

ESTRUTURAS DE DADOS EM SERVIÇOS DE REDES SOCIAIS *ONLINE*

UMA ABORDAGEM METODOLÓGICA DE ANÁLISE

Fernando de Assis Rodrigues



**CULTURA
ACADÊMICA**
Editora

ESTRUTURAS DE DADOS EM SERVIÇOS DE REDES SOCIAIS *ONLINE*

UMA ABORDAGEM METODOLÓGICA DE ANÁLISE

ESTRUTURAS DE DADOS EM SERVIÇOS DE REDES SOCIAIS *ONLINE*

UMA ABORDAGEM METODOLÓGICA DE ANÁLISE

Fernando de Assis Rodrigues

Marília/Oficina Universitária
São Paulo/Cultura Acadêmica
2024



**CULTURA
ACADÊMICA**
Editora



UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de Marília

Diretora

Profa. Dra. Claudia Regina Mosca Giroto

Vice-Diretora

Profa. Dra. Ana Cláudia Vieira Cardoso

Conselho Editorial

Mariângela Spotti Lopes Fujita (Presidente)

Célia Maria Giacheti

Cláudia Regina Mosca Giroto

Edvaldo Soares

Franciele Marques Redigolo

Marcelo Fernandes de Oliveira

Marcos Antonio Alves

Neusa Maria Dal Ri

Renato Geraldi (Assessor Técnico)

Rosane Michelli de Castro

Parecerista:

Prof. Dr. Rogério Aparecido Sá Ramalho

Professor Associado do Departamento de Ciência da Informação (DCI) da Universidade Federal de São Carlos (UFSCar).

Ficha catalográfica

R696c Rodrigues, Fernando de Assis.
Estruturas de dados em serviços de redes sociais online : uma abordagem metodológica de análise / Fernando de Assis Rodrigues. – Marília : Oficina Universitária ; São Paulo : Cultura Acadêmica, 2024.
312 p.
Inclui bibliografia
ISBN 978-65-5954-468-4 (Impresso)
ISBN 978-65-5954-469-1 (Digital)
DOI: <https://doi.org/10.36311/2024.978-65-5954-469-1>

1. Redes sociais on-line. 2. Estruturas de dados (Computação). 3. Modelagem de dados. 4. Interface de programas aplicativos (Software). 5. Tecnologia da informação. 6. Ciência da Informação. I. Título.

CDD 005.73

Telma Jaqueline Dias Silveira –Bibliotecária – CRB 8/7867

Imagem capa: <https://stock.adobe.com/br> - Arquivo "AdobeStock_295088748". Acesso em 09/05/2024

Editora afiliada:



Associação Brasileira de
Editoras Universitárias

Cultura Acadêmica é selo editorial da Editora UNESP

Oficina Universitária é selo editorial da UNESP - campus de Marília



Este trabalho está licenciado sob uma licença Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

*Dedico este texto ao Alfredo de Assis e à
Amélia Constantino de Assis (in memoriam).*

AGRADECIMENTOS

Agradeço a todos familiares e amigos que contribuíram de forma direta ou indireta para a elaboração deste material, fruto de anos de pesquisa. Também agradeço o compartilhamento do conhecimento de todos os professores do Departamento de Ciência da Informação, aos quais em muitos momentos fui colega de profissão, e do Programa de Pós-Graduação em Ciência da Informação da Universidade Estadual Paulista, meus mentores. Um agradecimento especial ao Professor Ricardo César Gonçalves Sant'Ana, atualmente, um amigo querido. Recebam a minha fraterna gratidão. Paz e vida longa a todos vocês.

Verba volant scripta manent
– *Provérbio romano*

LISTA DE EXEMPLOS

Exemplo 1 – <i>Facebook Graph</i> API: Requisição para coleta de conjuntos de dados, via método GET	63
Exemplo 2 – <i>Facebook Graph</i> API: Resultado da requisição para coleta de conjuntos de dados, via método GET	64
Exemplo 3 – <i>Facebook Graph</i> API: Requisição para coleta de conjuntos de dados, com parâmetros adicionais, via método GET	65
Exemplo 4 – <i>Facebook Graph</i> API: Resultado da requisição para coleta de conjuntos de dados, com parâmetros adicionais, via método GET	66
Exemplo 5 – <i>Facebook Graph</i> API: Requisição para coleta de conjuntos de dados de visões relacionadas, via método GET	66
Exemplo 6 – <i>Facebook Graph</i> API: Resultado da requisição para coleta de conjuntos de dados de visões relacionadas, com parâmetros adicionais, via método GET..	67
Exemplo 7 – <i>Facebook Graph</i> API: Requisição para coleta de conjuntos de dados, por meio de descoberta de conteúdo, via método GET	70
Exemplo 8 – <i>Facebook Graph</i> API: Tratamento de erros de processamento ou de autorização	71
Exemplo 9 – <i>Facebook Graph</i> API: Requisição para coleta de conjuntos de dados, a partir de uma versão específica da API, via método GET	72
Exemplo 10 – <i>X REST</i> API: Requisição para coleta de conjuntos de dados, via método GET	100
Exemplo 11 – <i>X REST</i> API: Resultado da requisição para coleta de conjuntos de dados, via método GET	100

Exemplo 12 – X REST API: Requisição para coleta de conjuntos de dados de visões relacionadas, via método GET	102
Exemplo 13 – X REST API: Resultado da requisição para coleta de conjuntos de dados de visões relacionadas, com parâmetros adicionais, via método GET ..	103
Exemplo 14 – X REST API: Requisição para coleta de conjuntos de dados de tuítes, por meio de descoberta de conteúdo, via método GET	106
Exemplo 15 – X REST API: Tratamento de erros de processamento ou de autorização	106
Exemplo 16 – <i>LinkedIn</i> REST API: Requisição para coleta de conjuntos de dados, via método GET	133
Exemplo 17 – <i>LinkedIn</i> REST API: Resultado da requisição para coleta de conjuntos de dados em XML, via método GET	134
Exemplo 18 – <i>LinkedIn</i> REST API: Requisição para coleta de conjuntos de dados, em formato JSON, via método GET	134
Exemplo 19 – <i>LinkedIn</i> REST API: Resultado da requisição para coleta de conjuntos de dados em JSON, via método GET	135
Exemplo 20 – <i>LinkedIn</i> REST API: Requisição para coleta de conjuntos de dados, com parâmetros adicionais, via método GET	136
Exemplo 21 – <i>LinkedIn</i> REST API: Tratamento de erros de processamento ou de autorização	138
Exemplo 22 – Consulta o Total de Visões disponíveis em cada Versão de API, em sintaxe <i>Structured Query Language</i>	204
Exemplo 23 – Consulta o Total de Atributos disponíveis em cada Visão, para cada Versão de API, em sintaxe <i>Structured Query Language</i> , com recorte as Visões que apresentam 50 ou mais atributos	205
Exemplo 24 – Consulta a Quantidade de Visões disponíveis em cada conjunto de Autorização de Acesso e de Permissão, para cada Versão de API dos Serviços de Redes Sociais <i>Online Facebook</i> e <i>LinkedIn</i> , em sintaxe <i>Structured Query Language</i>	207

- Exemplo 25 – Modelagem de Segunda Ordem: Consulta o acesso as Visões disponíveis no Serviço de Rede Social *Online Facebook, Facebook Graph* API, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email, public_profile, user_about_me, user_actions.books* e *user_friends*, em sintaxe *Structured Query Language* 244
- Exemplo 26 – Modelagem de Segunda Ordem: Consulta o acesso as Visões disponíveis em diferentes graus no Serviço de Rede Social *Online Facebook, Facebook Graph* API, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email, public_profile, user_about_me, user_actions.books* e *user_friends*, em sintaxe *Structured Query Language* 253
- Exemplo 27 – Modelagem de Segunda Ordem: Consulta o acesso aos Atributos das Visões disponíveis no Serviço de Rede Social *Online Facebook, Facebook Graph* API, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email, public_profile, user_about_me, user_actions.books* e *user_friends*, em sintaxe *Structured Query Language* 270
- Exemplo 28 – Modelagem de Segunda Ordem: Consulta o acesso aos Atributos das Visões disponíveis em diferentes graus no Serviço de Rede Social *Online Facebook, Facebook Graph* API, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email, public_profile, user_about_me, user_actions.books* e *user_friends*, em sintaxe *Structured Query Language* 282

LISTA DE FIGURAS

Figura 1 – Representação visual do Ciclo de Vida dos Dados para o contexto dos Serviços de Redes Sociais <i>Online</i>	28
Figura 2 – Exemplo de coleta de dados de um Serviço de Rede Social <i>Online</i> , via API, por um aplicativo de um Agente Externo	54
Figura 3 – Exemplo de coleta de dados de um Serviço de Rede Social <i>Online</i> , via API, por um aplicativo de um Agente Externo, com o uso de visões	59
Figura 4 – <i>Meta Platforms, Inc.</i> : Página principal da área reservada aos desenvolvedores	60
Figura 5 – <i>Meta Platforms, Inc.</i> : Seções disponíveis dos documentos da <i>Facebook Graph API</i>	62
Figura 6 – <i>Facebook Graph API</i> : Atributos na primeira forma de apresentação, tabela HTML com três colunas	75
Figura 7 – <i>Facebook Graph API</i> : Atributos na segunda forma de apresentação, tabela HTML com duas colunas	76
Figura 8 – <i>Facebook Graph API</i> : Atributos na terceira forma de apresentação, com elemento textual e com tabela HTML com duas colunas	77
Figura 9 – <i>Facebook Graph API</i> : subseção contendo informações sobre as permissões da visão.....	93
Figura 10 – <i>Facebook Graph API</i> : Vínculos entre as entidades.....	95
Figura 11 – <i>X Corp.</i> : Página principal da área reservada aos desenvolvedores...	96
Figura 12 – <i>X Corp.</i> : Seções disponíveis dos documentos da <i>X REST API</i>	97

Figura 13 – X REST API: Atributos na primeira forma de apresentação, tabela HTML com três colunas	110
Figura 14 – X REST API: Atributos na segunda forma de apresentação, com o uso da marcação de bloco de citação do HTML	111
Figura 15 – X REST API: Subseção contendo informações sobre as permissões da visão	123
Figura 16 – X REST API: Vínculos entre as entidades	128
Figura 17 – <i>LinkedIn Corporation</i> : Página principal da área reservada aos desenvolvedores	129
Figura 18 – <i>LinkedIn Corporation</i> : Seções disponíveis dos documentos da <i>LinkedIn REST API</i>	131
Figura 19 – <i>LinkedIn REST API</i> : Atributos na primeira forma de apresentação, tabela HTML com duas colunas.....	141
Figura 20 – <i>LinkedIn REST API</i> : Atributos na segunda forma de apresentação, tabela HTML com duas colunas.....	142
Figura 21 – <i>LinkedIn REST API</i> : Atributos na segunda forma de apresentação, tabela HTML com três colunas	143
Figura 22 – <i>LinkedIn REST API</i> : Atributos na quarta forma de apresentação, com o uso da marcação de bloco de citação do HTML	144
Figura 23 – <i>LinkedIn REST API</i> : Parágrafo contendo informações sobre as permissões de acesso a visão	154
Figura 24 – <i>LinkedIn REST API</i> : Vínculos entre as entidades.....	159
Figura 25 – Sobreposição (<i>Overlay</i>) dos vínculos de entidades das APIs	164
Figura 26 – Fluxograma do procedimento padrão para coleta de dados de API de Serviços de Redes Sociais <i>Online</i>	167
Figura 27 – Exemplo de um Diagrama Entidade-Relacionamento no nível lógico	178
Figura 28 – Exemplo de um Diagrama Entidade-Relacionamento no nível físico.....	179
Figura 29 – Diagrama Entidade-Relacionamento da Modelagem Direta.....	182

Figura 30 – Diagrama Entidade-Relacionamento da Modelagem Direta com dados de auditoria	199
Figura 31 – Exemplo de Consulta a Modelagem Direta: Consulta as relações entre API, Versões de API, Autorizações de Acesso, Permissões, Visões (e suas relações) para o Serviço de Rede Social <i>Online LinkedIn</i> , em formato de grafo de vértices e arestas.....	212
Figura 32 – Exemplo de Consulta a Modelagem Direta: Consulta Visões, seus atributos e suas relações para todas as Versões de API dos Serviços de Rede Social <i>Online Facebook, LinkedIn e X</i> , em formato de grafo de vértices e arestas, com ampliação da região da Visão <i>Users</i> da <i>X REST API</i>	214
Figura 33 – Exemplo de visualização <i>crosswalk</i> de coleta de dados pela <i>Facebook Graph API</i> , com o uso da Autorização de Acesso <i>User Access Token</i>	222
Figura 34 – Exemplo de visualização <i>crosswalk</i> de coleta de dados pela <i>Facebook Graph API</i> , com o uso da Autorização de Acesso <i>User Access Token</i> , exibindo atributos, relacionamentos e cardinalidades	225
Figura 35 – Exemplo de um modelo dimensional em esquema estrela	229
Figura 36 – Exemplo simplificado das dimensões de um <i>Data Mart</i>	230
Figura 37 – Exemplo de um modelo dimensional em esquema floco de neve...231	
Figura 38 – Tabelas da Modelagem Direta selecionadas para a composição da Modelagem de Segunda Ordem	235
Figura 39 – Diagrama Entidade-Relacionamento para a Modelagem de Segunda Ordem – Fato Visões acessíveis em primeiro grau – Esquema Estrela.....	237
Figura 40 – Diagrama Entidade-Relacionamento para a Modelagem de Segunda Ordem – Fato Visões acessíveis em diferentes graus – Esquema Estrela.....	248
Figura 41 – Diagrama Entidade-Relacionamento para a Modelagem de Segunda Ordem – Fato Atributos acessíveis em primeiro grau – Esquema Floco de Neve	260
Figura 42 – Diagrama Entidade-Relacionamento para a Modelagem de Segunda Ordem – Fato Atributos acessíveis em diferentes graus – Esquema Floco de Neve	276

LISTA DE GRÁFICOS

Gráfico 1 – <i>Facebook Graph</i> API: Ocorrências de nomes de atributos iguais em diferentes visões	80
Gráfico 2 – <i>Facebook Graph</i> API: Ocorrências de tipos de dados em atributos...	82
Gráfico 3 – <i>X</i> REST API: Ocorrências de nomes de atributos iguais em diferentes visões.....	113
Gráfico 4 – <i>X</i> REST API: Ocorrências de tipos de dados em atributos.....	115
Gráfico 5 – <i>LinkedIn</i> REST API: Ocorrências de nomes de atributos iguais em diferentes visões.....	146
Gráfico 6 – <i>LinkedIn</i> REST API: Ocorrências de tipos de dados em atributos	148

LISTA DE QUADROS

Quadro 1 – <i>Facebook Graph</i> API: Referências de permissões disponíveis, versão 2.6	89
Quadro 2 –Referências de permissões da <i>X</i> REST API, versão 1.1	124
Quadro 3 – Referências de permissões da <i>LinkedIn</i> REST API, versão 1.0....	155
Quadro 4 – Exemplo de Dicionário de Dados para um Modelo Entidade-Relacionamento.....	180
Quadro 5 – Dicionário de Dados da Modelagem de Direta.....	187
Quadro 6 – Recorte do Dicionário de Dados da Modelagem de Direta com auditoria: tabelas <i>coletor</i> e <i>parametro</i>	200
Quadro 7 – Dicionário de Dados da Modelagem de Segunda Ordem – Fato Visões acessíveis em primeiro grau.....	239
Quadro 8 – Recorte do Dicionário de Dados da Modelagem de Segunda Ordem – Fato Visões acessíveis em diferentes graus: tabelas <i>acesso</i> e <i>relacao_grau</i>	251
Quadro 9 – Dicionário de Dados da Modelagem de Segunda Ordem – Fato Atributos acessíveis em primeiro grau	263
Quadro 10 – Recorte do Dicionário de Dados da Modelagem de Segunda Ordem – Fato Atributos acessíveis em diferentes graus: tabelas <i>acesso</i> e <i>relacao_grau</i> ..	279

LISTA DE TABELAS

Tabela 1 – Modelagem Direta: Total de Visões disponíveis em cada Versão de API.....	00
Tabela 2 – Modelagem Direta: Total de Atributos disponíveis em cada Visão, para cada Versão de API, com recorte as Visões que apresentam 50 ou mais atributos (n >= 50).....	206
Tabela 3 – Modelagem Direta: Quantidade de Visões disponíveis em cada conjunto de Autorização de Acesso e de Permissão, para cada Versão de API dos Serviços de Redes Sociais <i>Online Facebook</i> e <i>LinkedIn</i>	209
Tabela 4 – Modelagem de Segunda Ordem: Visões disponíveis no Serviço de Rede Social <i>Online Facebook</i> , <i>Facebook Graph</i> API, versão 2.6, por meio da Autorização de Acesso <i>User Access Token</i> e das permissões <i>email</i> , <i>public_profile</i> , <i>user_about_me</i> , <i>user_actions.books</i> e <i>user_friends</i>	246
Tabela 5 – Modelagem de Segunda Ordem: Visões disponíveis em diferentes graus no Serviço de Rede Social <i>Online Facebook</i> , <i>Facebook Graph</i> API, versão 2.6, por meio da Autorização de Acesso <i>User Access Token</i> e das permissões <i>email</i> , <i>public_profile</i> , <i>user_about_me</i> , <i>user_actions.books</i> e <i>user_friends</i>	256
Tabela 6 – Modelagem de Segunda Ordem: Atributos de Visões disponíveis no Serviço de Rede Social <i>Online Facebook</i> , <i>Facebook Graph</i> API, versão 2.6, por meio da Autorização de Acesso <i>User Access Token</i> e das permissões <i>email</i> , <i>public_profile</i> , <i>user_about_me</i> , <i>user_actions.books</i> e <i>user_friends</i>	273
Tabela 7 – Modelagem de Segunda Ordem: Atributos de Visões disponíveis em diferentes graus no Serviço de Rede Social <i>Online Facebook</i> , <i>Facebook Graph</i> API, versão 2.6, por meio da Autorização de Acesso <i>User Access Token</i> e das permissões <i>email</i> , <i>public_profile</i> , <i>user_about_me</i> , <i>user_actions.books</i> e <i>user_friends</i>	285

LISTA DE ABREVIATURAS E SIGLAS

API	<i>Application Programming Interface</i>
APIs	<i>Application Programming Interfaces</i>
ASP	<i>Application Service Provider</i>
CODASYL	<i>Committee on Data Systems Languages</i>
GPS	<i>Global Positioning System</i>
HTML	<i>Hyper Text Markup Language</i>
HTTP	<i>Hyper Text Transfer Protocol</i>
HTTPS	<i>Hyper Text Transfer Protocol Secure</i>
IBM	<i>International Business Machines</i>
IP	<i>Internet Protocol</i>
JSON	<i>JavaScript Object Notation</i>
SaaS	<i>Software as a Service</i>
SOC	<i>Service-Oriented Computing</i>
SQL	<i>Structured Query Language</i>
SQL Injection	<i>Structured Query Language Injection</i>
TCP/IP	<i>Transmission Control Protocol/Internet Protocol</i>
TIC	Tecnologias de Informação e Comunicação
UNESCO	<i>United Nations Educational, Scientific and Cultural Organization</i>
URL	<i>Uniform Resource Locator</i>
WDDX	<i>Web Distributed Data Exchange</i>
WIDL	<i>Web Interface Definition Language</i>
XML	<i>eXtensible Markup Language</i>
XML-RPC	<i>eXtensible Markup Language - Remote Procedure Calling</i>

SUMÁRIO

Introdução	21
1 Os serviços de redes sociais <i>online</i>	39
1.1 Modelo de negócios baseado na oferta de serviço	47
1.2 O uso de <i>Application Programming Interfaces</i>	51
1.3 A <i>Facebook Graph</i> API	60
1.4 A <i>X</i> REST API	96
1.5 A <i>LinkedIn</i> REST API	129
2 Modelagem Direta	161
2.1 Sistema de Gerenciamento de Banco de Dados e o Modelo Entidade-Relacionamento	169
2.2 Aplicação do Modelo Entidade-Relacionamento na Modelagem Direta	182
2.3 Exemplos de uso de dados da Modelagem Direta	202

3	Modelagem de segunda ordem	217
3.1	<i>Data Warehouse, Data Mart e Business Intelligence</i> na análise de conjuntos de dados no contexto dos Serviços de Redes Sociais <i>Online</i>	227
3.2	Exemplos de aplicação do Modelo Entidade-Relacionamento na Modelagem de Segunda Ordem	233
3.2.1	Modelagem de Segunda Ordem – <i>Data Mart</i> de Visões Diretamente Acessíveis	236
3.2.2	Modelagem de Segunda Ordem – <i>Data Mart</i> de Visões Diretamente ou Indiretamente Acessíveis	247
3.2.3	Modelagem de Segunda Ordem – <i>Data Mart</i> de Atributos de Visões Diretamente Acessíveis	258
3.2.4	Modelagem de Segunda Ordem – <i>Data Mart</i> de Atributos de Visões Diretamente ou Indiretamente Acessíveis	275
4	Conclusão	289
	Referências	297
	Resumo da biografia do autor	309

INTRODUÇÃO

O desenvolvimento de redes de inter-relacionamento entre indivíduos, seja formada por afinidades em comum ou pela participação em estruturas formais como em uma sociedade organizada, se mistura com a história do desenvolvimento do ser humano.

As sociedades gregas e romanas – influenciadoras dos fundamentos basilares das democracias ocidentais – já demonstravam interesse em desenvolver ambientes próprios à exposição de ideias e de discussões, com o uso de argumentos lógicos e do discurso racional (Habermas, 2014), ainda que permitidas apenas a uma parcela da população, com temas em áreas como a política e a filosofia, e até de interesses sociais ou culturais (Malkin; Constantakopoulou; Panagopoulou, 2011). Exemplos de localizações que formalizam este processo são a ágora grega e o fórum romano, que se constituem, dentre outros aspectos, como ambientes públicos de inter-relacionamento de indivíduos com interesses em comuns, e; o senado, como a instrumentalização de um local público para a inter-relação de indivíduos por interesses formais, no caso, interesses políticos.

Entretanto, as bases científicas que moldaram o que contemporaneamente se denomina como redes sociais só tomaram forma no final do sé-

culo XIX, especialmente na Sociologia, com pesquisas de Émile Durkheim sobre a integração social e de Ferdinand Tönnies sobre a teoria de comunidade (*Gemeinschaft*) e de sociedade (*Gesellschaft*). Ao final do século XX, adiciona-se ao tema das redes sociais a penetração do uso de Tecnologias de Informação e Comunicação (TIC) nas ações e atividades humanas – inseridos no contexto de uma sociedade denominada Sociedade em Rede, em que seu principal ativo é a informação (Castells, 2018, 2021) – o que possibilitou o estabelecimento de sistemas de informação *online*, ofertados na forma de serviços oferecidos por instituições do sistema capitalista, que suportasse a formação de redes sociais com bilhões de usuários conectados, com a finalidade de troca de informações sobre inúmeros temas, além de representar um novo local de organização social e cultural: as Redes Sociais *Online* (Fumero; Vacas; Roca, 2007; Lévy, 2001).

Impulsionados pela disponibilidade de conexão da internet, surgiram sistemas de informação desenvolvidos com o objetivo de fornecer suporte às redes de informações e de inter-relacionamento entre indivíduos, grupos de indivíduos e instituições, denominadas, *Sites* de Redes Sociais (em princípio) e, posteriormente de Redes Sociais *Online* e ou somente Redes Sociais¹ (Boyd; Ellison, 2007).

Estas Redes Sociais *Online* se tornaram elementos-chave na realização de atividades profissionais e de entretenimento. São formadas por diversos serviços *online*, complexos, muitas vezes interligados a outros serviços pela internet, desenvolvidos não só com o uso de recursos tecnológicos, mas também com recursos humanos e financeiros, que coletam, armazenam, processam e compartilham informações (Wellman; Haythornthwaite, 2002).

A criação de sistemas de informação voltados para a operacionalização de Redes Sociais *Online* estão presentes desde o início da internet (Adamic; Adar, 2003; Wellman; Haythornthwaite, 2002). Com o ama-

¹ Especialmente em fontes secundárias (e.g. *jornais*) há um uso deliberado do termo Redes Sociais para denominar Redes Sociais *Online* ou até mesmo um determinado serviço. Neste livro, adotaremos Redes Sociais *Online* para tratar de quaisquer instituições que ofereçam um conjunto de serviços com a finalidade de formar Redes Sociais por meio de TIC.

durecimento das TIC e o aumento da disponibilidade de conexão – especialmente a partir da década de 2000 – as instituições começaram a migrar o acesso a estes sistemas de informação para um modelo de negócio baseado na oferta de serviços específicos para o inter-relacionamento de seus usuários e para a troca de informações em formato multimídia, tais como imagens, vídeos, áudios, *hyperlinks* para conteúdos externos e textos (Mislove *et al.*, 2007).

Estes serviços e seus sistemas de informação utilizam recursos computacionais para automatizar determinados processos, como, por exemplo, as funções de desligar a tela de um dispositivo ao perceber pela câmera que o indivíduo não está mais utilizando-a, de sugerir um restaurante de acordo com os gostos em comuns com seus amigos ou pela geolocalização do dispositivo de acesso, e entre outros. Em uma parcela destes serviços, as funções realizam suas ações a partir da coleta de conjuntos dados de indivíduos, trafegando os dados aos seus servidores e bancos de dados, através da infraestrutura da Internet – no qual aqui propõe-se um nome para este tipo de serviço: Serviços de Redes Sociais *Online*.

Afinal, por que se aborda as Redes Sociais *Online* como um serviço e não uma ferramenta ou uma plataforma? A resposta é simples. O indivíduo é obrigado a assinar e concordar com um Termo de Serviço ao acessar pela primeira vez o sistema. Trata-se de um contrato social entre o indivíduo e a instituição detentora do serviço, tal qual é realizado com uma operadora de internet ou com a assinatura de um serviço de eletricidade. O que pode dar certa opacidade neste contrato é que os Serviços de Redes Sociais *Online* são gratuitos para utilização, na maioria dos casos.

A partir do ano de 2010, determinados Serviços de Redes Sociais *Online* apresentavam números expressivos de usuários ativos; com alta representatividade destes usuários em relação a totalidade da população mundial (Statista, 2023a) – denominadas por Donath (2007) como super-redes sociais, parte integrante do conjunto dos serviços mais acessados e utilizados através da infraestrutura da internet (ALEXA, 2021).

Entretanto, os Serviços de Redes Sociais *Online* são, prioritariamente, elaborados e mantidos por instituições privadas, no qual a troca de informações pessoais nestes serviços suscitam preocupações já existentes em outros contextos como: a exposição de conjuntos de dados sobre usuários para outras instituições, governos e, inclusive, para outros usuários; crimes sexuais e abusos contra a criança e a juventude; a perseguição online de pessoas; as ações e as atividades resultantes de intolerância; entre outras (Acquisti; Gross, 2006; Barnes, 2006; Boyd, 2008; Boyd; Ellison, 2007; Dinev; Hart, 2004; Fogel; Nehmad, 2009; Krasnova *et al.*, 2009; Rodrigues; Sant'ana, 2016; Tufekci, 2007; Viseu; Clement; Aspinall, 2004; Young; Quan-Haase, 2009). Em todos estes cenários, ocorrem exposições de dados pessoais que perpassam questões inerentes à privacidade, capacidade de resguardo de informações de indivíduos, grupos ou instituições perante outros entes, situações que se enquadram em aspectos sobre a privacidade de indivíduos, identificados na literatura do Direito, especialmente a ocidental (Solove, 2009; Westin; Solove, 2015).

Um dos pontos que torna o problema de privacidade grave é a dificuldade em controlar o que de fato existe de dados sobre um indivíduo nos Serviços de Redes Sociais *Online*. Existe uma percepção que o indivíduo está imune a ter seus dados circulando nos serviços ao não se tornar um usuário. Contudo, este indivíduo não-usuário de um Serviço de Rede Social *Online* pode ter seus dados em conteúdos de outros indivíduos, tais como em conteúdos audiovisuais divulgados, em citações e em outros conteúdos com origem em outros usuários. Um exemplo deste tipo de indivíduo 'não-usuário' é encontrar conteúdos *post mortem* da Clarice Lispector e de Machado de Assis (mesmo que muitas vezes falsamente associados a produção literária dos mesmos).

Para aumentar a complexidade do cenário, adiciona-se a capacidade de algoritmos em realizar reconhecimento facial e textual deste conteúdo, sejam desenvolvidos pela instituição detentora do serviço ou por Agentes Externos.

Portanto, o indivíduo possuir ou não uma relação direta com Serviços de Redes Sociais *Online* não o torna imune das preocupações existentes para este novo contexto social. Neste sentido, denominamos como Referenciado tanto os indivíduos não-usuários como os usuários, que são sujeitos passíveis de identificação pelos seus identificadores e seus potenciais identificadores em um determinado contexto, no qual este processo de identificação pode potencializar os problemas relacionados a sua privacidade.

Em pesquisas anteriores, Rodrigues e Sant’Ana (2016, p. 286, tradução nossa) destacam que:

[...] problemas relacionados a privacidade de usuários em redes sociais não são causados somente pelo uso de TIC (que pode fazer papel de agente catalisador na coleta automatizada de grandes quantidades de dados sobre usuários), mas também pela atividade de usuários, de Agentes Externos e de controladores que possuam conhecimento suficiente para coletar e processar estes dados com outras fontes, estabelecendo novos dados potencialmente prejudiciais à privacidade.

Soma-se a esta complexidade do cenário a figura do Agente Externo: são indivíduos ou instituições externas que possuem acesso parcial dos dados circulantes em um determinado serviço de internet. Diversos serviços disponíveis através da internet apresentam interfaces para o compartilhamento de seus dados com outros sistemas de informação, incluindo os Serviços de Redes Sociais *Online*. Estas interfaces são definidas para permitir regras de interoperabilidade de dados com Agentes Externos – fazendo com que as instituições detentoras dos serviços possam gerar receita por meio do acesso aos dados ou pela oferta de análises realizadas a partir destes dados, uma das formas corriqueiras das instituições em permitir que não existam taxas para a utilização (plena ou parcial) de seus serviços.

As *Application Programming Interfaces*² (APIs) são responsáveis pelo acesso para coleta de conjuntos de dados por aplicações elaboradas pelos Agentes Externos – potencializando as preocupações legais sobre o que é passível de realização com estes dados, e quais são os conjuntos de dados de Referenciados que estão disponíveis no momento da coleta.

Para Sheth (1999), a interoperabilidade de dados é importante, principalmente: a) pelo progresso que a internet proporcionou na distribuição de dados e na interconexão de sistemas de informação distribuídos – ou seja – sistemas de informação desenvolvidos e gerenciados por diferentes instituições; b) pela própria especialização destes sistemas de informação, que tem objetivos voltados para atividades específicas, como no caso dos Serviços de Redes Sociais *Online*, e; c) pelo reuso e análise de dados para a criação de novas informações e o posterior reuso e compartilhamento destas novas informações ou dos dados originais em outros sistemas de informação.

Entretanto, a interoperabilidade de conjuntos dados é complexa, já que o cenário de coleta de dados é heterogêneo, pois não há estruturas ou formas fixas ou tipos de conteúdo obrigatórios (Wiederhold, 1993). Sheth (1999) divide o desenvolvimento da interoperabilidade entre sistemas de informação em gerações, na qual as duas primeiras são voltadas ao delineamento do processo de coleta e armazenamento de dados com o uso das arquiteturas *mainframe* e cliente-servidor.

A terceira geração de interoperabilidade entre sistemas de informação, paradigma atual, possui como características principais (Sheth, 1999):

- a) preocupações com a formação de conhecimento e informações, a partir da manipulação de dados de diversos sistemas de informação, incluindo dados sobre indivíduos;
- b) possibilidade de interoperabilidade de conjuntos de dados, associados a metadados ou ontologias para agregar carga semântica;

² Termo no idioma inglês para Interface de Programação de Aplicação.

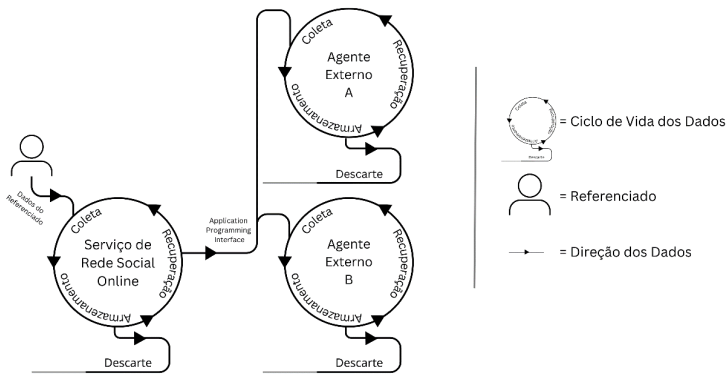
- c) o uso da internet como rede de distribuição de dados e o compartilhamento de dados em escala global;
- d) a disponibilidade de acesso aos dados armazenados em servidores, através do uso de softwares desenvolvidos para diversos dispositivos, tais como: microcomputadores, *laptops*, *tablets*, *smartphones* e *wearables*;
- e) a construção e a adoção de formatos, para a explicitação de conjuntos de dados, como linguagens de marcação;
- f) a independência tecnologia específica para a interpretação ou o uso dos conjuntos de dados;
- g) o surgimento de profissões voltadas à interpretação e à correlação de conjuntos de dados.

A interoperabilidade de dados depende diretamente de processos relacionados as fases de coletar, de armazenar e de recuperar conjuntos de dados de forma cíclica. Em certos casos, há a possibilidade de descarte dos dados, parte de um Ciclo de Vida dos Dados (Sant'ana, 2016). Seguindo a proposta de que um dado está relacionado a um ciclo de vida, pode-se extrapolar a proposta cíclica para os dados em Serviços de Redes Sociais *Online*.

Os dados dos Referenciados são coletados e armazenados pelas instituições detentoras. As instituições detentoras podem criar interfaces que permitam a sua exclusão (descarte) e acesso (recuperação) – seja pelos próprios usuários (pelos aplicativos) ou por Agentes Externos, via *Application Programming Interface* (API). Os Agentes Externos podem criar seus próprios ciclos, a partir da recuperação do ciclo anterior. Ou seja, pela perspectiva do Agente Externo, o seu ciclo de vida de dados inicia na coleta de um dado a partir recuperação de outro ciclo. A Figura 1 estabelece uma representação deste cenário, em uma hipotética coleta de dados de dois

Agentes Externos (A e B), a partir de um Serviço de Rede Social *Online*, via API.

Figura 1 – Representação visual do Ciclo de Vida dos Dados para o contexto dos Serviços de Redes Sociais *Online*



Fonte: Elaborado pelo Autor, adaptado de Sant’Ana (2016).

As fases de coleta, de armazenamento, de recuperação e de descarte são permeadas por objetivos – delimitações temáticas de análise, sendo: Preservação, Disseminação, Direitos Autorais, Qualidade, Integração e Privacidade (Sant’Ana, 2016). Ou seja, é possível analisar quaisquer fases por quaisquer objetivos. Neste livro, trataremos a coleta dos dados em Serviços de Redes Sociais *Online* pela temática da privacidade, já que ciclos de vida de dados interligados por APIs podem ser o elemento-chave em um processo que não se tenha controle com quem de fato tem acesso aos conjuntos de dados de Referenciados.

Por que a privacidade? Quando o Referenciado utiliza um serviço disponível na internet, ele deve estar de acordo com as regras estabelecidas em um conjunto de documentos que compõem os Termos de Serviço (Reidenberg, 1998).

Os Termos de Serviço estabelecem

As regras que uma pessoa ou uma organização devem observar em ordem para utilizar o serviço. Geralmente vinculado a legalidade jurídica a não ser que este viole leis federais ou locais, os Termos de Uso podem variar de tempo em tempo, e é responsabilidade do provedor do serviço avisar quando ocorrer quaisquer alterações. Um *website* que disponibiliza apenas informações ou vendas de produtos geralmente não possui um termo de uso. Entretanto, os provedores de Internet e todos os *websites* que armazenam dados pessoais para usuários possuem; em particular, *sites* de redes sociais, de leilões e de transações financeiras (PC MAGAZINE, 2023, tradução nossa).

A aceitação das condições é realizada no momento da assinatura do contrato de adesão ao serviço, e pode ser realizado de duas formas: *clickwrap*, quando há somente uma marcação para o aceite, mesmo sem a obrigatoriedade de leitura das regras (largamente utilizado pelos Serviços de Redes Sociais *Online*) e *browsewrap*, quando o acesso é permitido mesmo sem o Referenciado manifestar aceitação de forma ativa (Garcia, 2013; Harrison, 1998; Lemley, 2012).

São nos Termos de Serviço que estão estabelecidas as regras de coleta de dados por Agentes Externos e não há transparência ao Referenciado sobre quais serão os processamentos que serão realizados por estes Agentes Externos, e quais serão os novos conjuntos de dados gerados a partir destes processamentos, aumentando potencialmente a possibilidade de ações e de atividades prejudiciais à privacidade, a partir deste ambiente de coleta de dados. Para tornar o processo mais complexo, os Agentes Externos podem estabelecer seus próprios Termos de Serviço – com regras diferentes dos Serviços de Redes Sociais *Online*.

Os Termos de Serviço dos Agentes Externos podem descrever novos processos de coleta ou de compartilhamento de dados a seus parceiros e outros Agentes Externos – e estes outros Agentes Externos também podem ter seus dados acessíveis por outros entes – criando um efeito em cascata

de identificação do Referenciado que pode potencializar ações e atividades prejudiciais à privacidade, a partir da singular existência de um dado como o e-mail do Referenciado adquirido na coleta de dados inicial (no caso, dos Serviços de Redes Sociais *Online*).

Ou seja, os conjuntos de dados de Referenciados compartilhados aos Agentes Externos estarão vinculados a novos Termos de Serviço e a outras jurisdições, que podem estabelecer outros critérios de processamento e compartilhamento dos dados de Referenciados ou estarem submetidos a legislação de outros países: um ambiente de alta opacidade mesmo quando analisados por profissionais experientes.

Os Termos de Serviço têm duplo papel neste processo: pacificador, enquanto elemento que fortalece a percepção de segurança aos Referenciados, ao estabelecer os limites e as garantias legais sobre o que é realizado com conjuntos de dados, e; como elemento de opacidade entre Referenciados e as instituições sobre o *modus operandi* do uso e compartilhamento de dados, diluído em uma alta complexidade dos sistemas de informação, da linguagem jurídica dos Termos de Serviço e na variedade de ações e atividades passíveis de realização por meio das APIs (Rodrigues; Sant'ana, 2016).

Portanto, mesmo com o auxílio de profissionais com habilidades e conhecimentos prévios sobre a forma de coletar conjuntos de dados de Referenciados e sobre os aspectos legais de compartilhamento de dado, não há um modelo estruturado para análise sobre os dados destes serviços e uma sistematização apropriada para o acompanhamento dos aspectos relacionados à privacidade de Referenciados no processo de coleta de dados.

Uma outra pergunta que pode ser realizada nesta etapa é: trata-se de analisar dados de um determinado Referenciado ou de um grupo? A resposta é não. Quando se coloca em voga a ideia de estudar a coleta de dados é comum pensar que se trata de estudar os dados que circulam nos serviços, como, por exemplo, desde coletar dados de tendências de comportamento até métricas que permitam identificar agrupamentos (*clusters*). Estes

exemplos são uma pequena fração de inúmeras pesquisas sobre os diversos fenômenos socioculturais, políticos e econômicos que os Serviços de Redes Sociais *Online* são submetidos, realizados por pesquisadores consagrados tanto no Brasil como em outros países. Não são sobre estes tipos de análise que este livro trata. A perspectiva deste livro é outra.

Em discussões que tratam sobre o assunto de coleta de dados em Serviços de Redes Sociais *Online* é quase que uma unanimidade o discurso que as instituições detentoras destes serviços têm acesso quase ilimitado a dados de Referenciados. Esta afirmação é um mantra dito e repetido de tal forma que se tornou um senso comum afirmar que determinado serviço está ou pode estar violando a privacidade de Referenciados, especialmente aqueles que de fato possuem conta de usuário.

Todavia, seria interessante que existisse uma abordagem metodológica para estabelecer um modelo sistematizado de análise que permitisse i) determinar qual o dado é coletado, ii) sistematizar estes dados de uma forma que seja possível consultar os dados disponíveis em cada serviço, iii) elaborar um processo unificado que possa habilitar uma comparação entre os dados coletados e disponíveis em serviços distintos, além de poder acompanhar as mudanças na coleta de dados de um mesmo serviço ao longo do tempo, e iv) permitir a identificação de qual é o conjunto de dados que estará disponível para acesso aos Agentes Externos. É sobre este tipo de problema que este livro trata.

A ausência de uma abordagem metodológica com esta finalidade dificulta o processo de identificação de fragilidades sobre questões de privacidade de dados disponíveis nos Serviços de Redes Sociais *Online*, adquiridos no momento da coleta e disponíveis aos Agentes Externos por meio da interoperabilidade das APIs, com acesso, inclusive, a conjuntos de dados dos Referenciados; o que também ofusca análises nesta temática e não permite validar estas percepções de senso comum, ou seja, de verificar potenciais prejuízos à privacidade.

Mas, como ter acesso a informações oficiais das instituições sobre que tipo de conjunto de dados está disponível em cada serviço, evitando conjecturas? A resposta para esta pergunta é simples e, ao mesmo tempo, complexa: as instituições disponibilizam estas informações em uma série de documentos em seus respectivos *websites*. Estes documentos têm a finalidade de orientar os Agentes Externos, ou seja, são documentos que explicitam informações estruturalistas sobre quais dados foram coletados e armazenados pelos serviços, e estão disponíveis para acesso via API. Popularmente são denominados como documentação técnica ou documentos de referência, já que são encontrados geralmente em guias de referência para programadores de software externos ao serviço.

Os Termos de Serviço não apresentam os conjuntos de dados coletados, armazenados e compartilhados com Agentes Externos, pelo menos não em sua totalidade. Já a documentação técnica, que estabelece as definições e os processos para a troca de conjuntos de dados via APIs dos serviços disponíveis na Internet, tem como foco primário descrever as normas técnicas de funcionamento, em linguagem e com o uso de terminologias voltadas principalmente para desenvolvedores de software por Agentes Externos. Ou seja, ao promover o uso dos dados pelos Agentes Externos, a documentação técnica precisa pormenorizar todos os dados disponíveis, bem como as condições de acesso – por isso a importância de levar em consideração este tipo de documentação para evitar conjecturas sobre o que de fato é ou não é compartilhado.

Neste sentido, o objetivo é estabelecer uma abordagem metodológica que permita ao investigador análises sobre quais são as estruturas de dados de Serviços de Redes Sociais *Online* que estão disponíveis ao acesso, a partir da identificação das características da coleta de dados de Referenciados via APIs.

Para a construção desta abordagem, os objetivos específicos estabelecidos são:

- i) Identificar as características do processo de coleta de dados de Referenciados, a partir de informações contidas na documentação técnica voltada para interoperabilidade com Agentes Externos, disponíveis nos *websites* dos Serviços de Redes Sociais *Online*;
- ii) Sistematizar um procedimento padronizado de coleta de dados de informações sobre as APIs, a partir da identificação de similaridades entre as amostras coletadas;
- iii) Estruturar um modelo para armazenamento dos dados sobre o processo de coleta de dados de Referenciados – denominado como Modelagem Direta;
- iv) Estruturar um modelo para armazenamento dos dados da Modelagem Direta, em uma estrutura de armazenamento orientada à recuperação de informações específicas para análise – denominado como Modelagem de Segunda Ordem.

Para desenvolver estas metas, optou-se pelo estudo das APIs dos Serviços de Redes Sociais *Online Facebook*³, *X*⁴ e *LinkedIn*⁵, com dados coletados entre o primeiro semestre de 2016 e o primeiro semestre de 2023. Estes serviços foram escolhidos pelo acesso gratuito⁶ de seus serviços pela perspectiva de seus Referenciados, e seus *websites* estarem classificados entre os 20 serviços mais acessados da Internet, segundo a ALEXA (2021)⁷.

³ *Facebook* é marca registrada de *Meta Platforms, Inc.*

⁴ *X* é marca registrada de *X Corp.*. Até o ano de 2023, o *X* era denominado como *Twitter*, de propriedade da *Twitter, Inc.*

⁵ *LinkedIn* é marca registrada de *LinkedIn Corporation*.

⁶ Apesar de serem serviços de acesso gratuito, o *LinkedIn* possui acesso prêmio mediante o pagamento de uma taxa mensal, e *X* e *Facebook* possuem *in-app purchases* – conteúdos e serviços que podem ser adquiridos mediante o pagamento de uma taxa.

⁷ Apesar do Serviço de Rede Social *Online Baidu Tieba* estar classificado entre os serviços mais acessados da internet, a sua posição nesta classificação contabiliza o total de acessos a todos os produtos da Baidu Inc., diferente dos três serviços analisados, no qual foram contabilizados somente os acessos aos serviços com a finalidade de Rede Social *Online* – e, portanto, não sendo analisada por esta pesquisa. Entretanto, não há obstruções para a aplicação dos conceitos e do modelo apresentado neste estudo em qualquer rede social, de qualquer tamanho ou em qualquer idioma.

Como as funcionalidades disponíveis nas APIs estão em constante desenvolvimento – com a adição de funcionalidades de coleta a partir de novos recursos oferecidos e a remoção de funcionalidades – as instituições adotam a nomenclatura das versões de revisão das APIs, baseadas no sistema de controle de versões numéricas. As versões de APIs analisadas neste estudo foram: *Facebook Graph* API, nas versões 2.6; *X REST* API⁸, na versão 1.1 e *LinkedIn REST* API⁹, na versão 1.0.

Estas versões foram escolhidas por se tratarem de um recorte de um momento importante para o assunto aqui abordado. São as versões das APIs vigentes no momento em que se antecedeu o primeiro escândalo global de acesso aos dados de Referenciados para uso diferente da finalidade coletada: o escândalo da *Cambridge Analytica*¹⁰.

Com relação a documentação de referência, foram analisadas as APIs que possuem acesso à coleta de dados de Referenciados, descartando as APIs com acessos privilegiados para empresas parceiras, e que são, a princípio, concedidos pela origem¹¹. Além disso, as seções da documentação técnica que está relacionada as requisições de gerenciamento de conteúdo nos serviços por aplicativos externos – tais como os métodos de inserção, de alteração e de exclusão de dados – foram descartadas, pois não são elementos vinculados com a coleta de dados por Agentes Externos, e sim vinculados com a manipulação de conjuntos de dados armazenados nos bancos de dados destas redes.

Outro ponto importante de descarte é desconsiderar a subversão de regras destes serviços. Exemplos de subversão são o uso de técnicas de invasão ou o uso de acessos não autorizados às APIs existentes, tais como: o *brute force*, a exaustão de tentativa de entrada no sistema por tentativa –

⁸ REST é acrônimo do termo no idioma inglês *Representational State Transfer*. Como *LinkedIn* e o *Twitter* utilizam esta arquitetura, adotou o acrônimo como o nome de sua API.

⁹ Ver nota 8.

¹⁰ Ver Isaak e Hanna (2018).

¹¹ Por exemplo, o Facebook possui uma API denominada *Meta Marketing* API, que necessita que as empresas sejam autorizadas previamente e façam parte do programa *Facebook Marketing Partner* (Meta Platforms, INC., 2023a).

acerto/erro; os *exploits*, como a exploração de falhas em códigos-fonte, e; o *packet sniffing*, utilizado para a análise e a captura de pacotes de dados em uma rede à revelia do emissor e do receptor.

Com relação ao método empregado, trata-se de uma pesquisa de abordagem qualitativa, de natureza aplicada, de objetivo exploratório, com procedimento metodológico cunhado na pesquisa documental, com o uso de métodos combinados a partir da exploração das características das APIs, da leitura da documentação técnica e do desenvolvimento dos modelos propostos por meio do uso de abstrações de modelos de armazenamento de dados.

Propõe-se três etapas, sendo a primeira relacionada à explicitação das características do processo de coleta de dados apresentadas pela documentação técnica das APIs, identificando conjuntos de dados e funcionalidades existentes nas APIs, e a sistematização dos procedimentos realizados pelos coletores para acesso aos dados em cada API.

Na segunda etapa, propõe-se a Modelagem Direta para armazenamento dos dados sobre o processo de coleta de dados, as características das estruturas existentes e as funcionalidades apresentadas pelas APIs. A Modelagem Direta é concomitante aos conceitos de modelagem de dados para aplicação em Sistemas de Gerenciamento de Banco de Dados (Silberschatz; Korth; Sudarshan, 2020).

Todavia, explicitar o contexto apenas a partir da elaboração de uma Modelagem Direta torna complexo o processo de identificação dos aspectos relacionados à privacidade dos dados de Referenciados nas APIs, principalmente em função do volume de funcionalidades, de dados e demais características.

Neste sentido, propõe-se uma terceira etapa, com base na construção de um *Data Warehouse* alimentado pelas características dos dados da Modelagem Direta, porém orientado a análise de dados – gerando uma modelagem denominada como Modelagem de Segunda Ordem, com a

aplicação dos esquemas estrela e floco de neve (Inmon, 2005; Kimball; Ross, 2011).

Para Modelagens Direta e de Segunda Ordem foram desenvolvidos como instrumentos os Diagrama Entidade-Relacionamento e Dicionário de Dados, além um ensaio de avaliação do acesso aos dados dos Serviços de Redes Sociais *Online* por Agentes Externos, a partir a população dos dados amostrais nas modelagens propostas. O ensaio tem caráter exemplificativo, e tem como objetivo auxiliar na compreensão dos aspectos que podem prejudicar a privacidade de dados de Referenciados nestes serviços, no momento da coleta de dados.

Este livro está segmentado em quatro capítulos. O primeiro capítulo trata sobre os aspectos ligados aos Serviços de Redes Sociais *Online*, apresentando um breve contexto histórico do início de estudos científicos de identificação das características das redes sociais, de teorias e de processos de visualização de redes sociais; a diferença do uso do termo rede social após o surgimento de *websites*, as características gerais da coleta de dados destes serviços e o uso das APIs. Neste capítulo também são apresentadas as características da *Facebook Graph* API da *X REST* API e da *LinkedIn REST* API – de forma individualizada (atendendo a primeira etapa).

O segundo capítulo apresenta as similaridades no processo de coleta de dados entre as APIs analisadas, delimita suas características e demais elementos que compõem o processo de entrada e da coleta de dados. Apresenta conceitos elementares Sistema de Gerenciamento de Banco de Dados e o Modelo Entidade-Relacionamento, e a aplicação destes na Modelagem Direta. Detalha cinco exemplos de uso de dados da Modelagem Direta (atendendo a segunda etapa).

O terceiro capítulo é reservado para o desenvolvimento da Modelagem de Segunda Ordem, em atendimento a terceira etapa, explicando em seu *caput* a necessidade de seu desenvolvimento e, em sequência apresentando os elementos do *Data Warehouse* e do *Business Intelligence* que foram empregados na estruturação da modelagem. Ao final, apresen-

ta-se quatro exemplos de análises, sendo dois em esquema estrela e dois em esquema floco de neve.

No quarto capítulo são apresentadas as conclusões e reflexões sobre o percurso trilhado no desenvolvimento deste livro. Também são propostos possíveis caminhos para a aplicação da abordagem metodológica de análise.

**OS SERVIÇOS DE
REDES SOCIAIS *ONLINE***

OS SERVIÇOS DE REDES SOCIAIS *ONLINE*

Como mencionado na introdução deste livro, pesquisas relacionadas as redes sociais não é algo novo e muito menos surgiram a partir do desenvolvimento das tecnologias digitais. Não é intenção deste capítulo explorar um contexto historicista do desenvolvimento das pesquisas de forma exaustiva, mas sim pontuar elementos importantes para compreender os fenômenos contemporâneos.

A partir do século XX, diversas áreas do conhecimento influenciaram a forma que os Serviços de Redes Sociais *Online* foram concebidos. Um destaque especial é a influência de Moreno (1955) nas áreas de Psiquiatria e de Psicologia, por ser um dos precursores do processo de explicitação das relações de indivíduos por visualizações baseadas na Teoria dos Grafos (com origem na Matemática), denominado como sociograma.

O sociograma é uma forma visual para representar vínculos entre os indivíduos. Pode representar visualmente parte dos dados e das relações em uma rede social. Esta forma de representação influenciou diretamente à disposição dos conjuntos de dados em Serviços de Redes Sociais *Online*, especialmente aqueles que estão disponíveis para a coleta de dados por Agentes Externos.

Seguindo a lógica proposta pela Teoria dos Grafos (Biggs; Lloyd; Wilson, 1999), as formas geométricas circulares de um sociograma representam os indivíduos em uma rede. O tamanho ou a coloração do círculo podem ser alterados para destacar grupos ou outros tipos de categorização, e incluir o uso de rótulos para designar informações sobre cada entidade, como o nome ou outro atributo.

As ligações entre os círculos são denominadas arestas, explicitadas na forma de vetores retos ou angulados, com início e término em bordas de duas entidades diferentes, no qual cada ligação representa um vínculo entre dois indivíduos. Não há limites mínimos ou máximos estabelecidos para restringir a quantidade de vínculos de um indivíduo (Biggs; Lloyd; Wilson, 1999; Wilson, 2010).

Com o surgimento da infraestrutura de rede da internet – que permite o fluxo de inúmeros tipos de dados e documentos – e impulsionados pela criação em 1989 da linguagem de marcação *HyperText Markup Language* (HTML) e do protocolo *HyperText Transfer Protocol* (HTTP), surgiram sistemas de informação desenvolvidos com o objetivo de fornecer suporte às redes de informações e de inter-relacionamento entre indivíduos, grupos de indivíduos e instituições, denominados Redes Sociais *Online* (no idioma inglês, *Online Social Networks*), *Sites* de Redes Sociais (no idioma inglês, *Social Network Sites*) ou, de forma sintética, por Redes Sociais (Adamic; Adar, 2003; Boyd; Ellison, 2007; Conseil Européen Pour La Recherche Nucléaire (CERN), 2015) (Flake; Lawrence; Giles, 2000).

Apesar da existência das Redes Sociais *Online* desde o início da infraestrutura da internet, somente no início da década de 2000 as Redes Sociais *Online* começaram a ganhar destaque, principalmente influenciado pelo surgimento de uma série de aplicativos para dispositivos móveis e de *websites* oferecendo serviços com o propósito de proporcionar o fluxo de conteúdo digital entre os participantes da rede. As estruturas de dados nestes serviços são similares as dinâmicas do sociograma: os Referenciados possuem perfis de usuários (nós da rede) e os preenchem com dados pes-

soais (atributos do nó). Os perfis podem ser relacionados com outros usuários da rede e com conteúdo que trafegam no sistema (arestas) (Adamic; Adar, 2003).

Para Boyd e Ellison (2007, p. 210), as Redes Sociais *Online* são,

[...] serviços baseados na *web* que permitem indivíduos (1) construir um perfil público ou semipúblico com de um sistema limitado, (2) articular uma lista de outros usuários com quem compartilham uma conexão e (3) visualizar e percorrer sua lista de conexões e aquelas feitas por outros dentro do sistema. A natureza e a nomenclatura dessas conexões podem variar de site para site.

Além disso, Jorente, Santos e Vidotti (2009, p. 10) adicionam que as Redes Sociais *Online* também se tratam de ambientes de troca de informações, a partir dos

[...] aparatos tecnológicos informacionais da transferência (ambientes digitais, estruturas de produção, tratamento, armazenamento e reprodução de recursos ou mensagens, produção de novos sistemas e modelos de armazenagem e acesso à informação, entre outros).

Com o amadurecimento das TIC, diversas iniciativas acabaram por se tornarem um negócio, visando a lucratividade, sendo gerido por instituições privadas. Com isso, as Redes Sociais *Online* são serviços oferecidos de forma gratuita ou por meio de assinatura, aonde o Referenciado é obrigado a concordar com um Termo de Serviço ao acessar pela primeira vez o serviço.

Neste sentido, as Redes Sociais *Online* são entendidas como instituições que elaboram um ou mais serviços. Os serviços são disponibilizados por meio de ferramentas na forma de *websites* e de aplicativos para dispositivos móveis, eletrodomésticos e *wearables*, com o intuito de subsidiar um espaço que permita a comunicação e o inter-relacionamento de Referenciados, que podem ser pessoas, grupos ou instituições

(LinkedIn Corporation, 2023a; Meta Platforms, INC., 2022; Rodrigues; Sant'ana, 2018, 2023; X CORP., 2023a).

Nos Serviços de Redes Sociais *Online*, os Referenciados podem optar por estabelecer relações com outros Referenciados, aonde estas relações podem representar uma amizade, uma conexão profissional ou até mesmo um interesse em receber conteúdo de outros Referenciados.

As conexões ocorrem pelo relacionamento entre perfis de Referenciados, como por exemplo, uma área que descreve os atributos de um indivíduo ou de uma instituição – pois os Serviços de Redes Sociais *Online* permitem que instituições participem como usuários do serviço, e em alguns casos, com perfis de acesso, conjunto de dados e funcionalidades distintas das disponíveis para perfis para pessoas. Por exemplo, parte dos Referenciados também podem ser administradores de grupos e de páginas de conteúdo. Estes Referenciados são administradores, que enviam informações sobre grupos ou páginas de conteúdo à área própria para o armazenamento e o compartilhamento com os demais Referenciados. As páginas de conteúdo podem ser personalizadas para representar uma instituição, um produto, um serviço, entre outras modalidades.

Para cada Referenciado acessar o conteúdo – que pode ser formado pelos mais variados recursos multimídia – é necessário relacionar algum tipo de identificador único, como um documento de identidade, um número de telefone ou um endereço de e-mail – e ao identificar-se, a instituição proprietária do Serviço de Rede Social *Online* tem acesso a dados com potencial de identificação, como a data e hora de acesso; a localização no Sistema de Posicionamento Global (GPS, acrônimo do termo inglês *Global Positioning System*); o endereço *Internet Protocol* (IP); dados sobre a rede de acesso; entre outros.

Os Serviços de Redes Sociais *Online* não restringem os conteúdos veiculados apenas a informações ou aos dados dos Referenciados. Ou seja, os Referenciados – indivíduos não-usuários e os usuários dos Serviços de Redes Sociais *Online* – são sujeitos que são passíveis de identificação em

um determinado contexto pelos seus identificadores, seus potenciais identificadores e pelos recursos multimídia que circulam no serviço, no qual este processo de identificação pode potencializar os problemas relacionados a sua privacidade.

Por exemplo, os conjuntos de dados pessoais de um Referenciado são vinculados com identificadores únicos, sendo possível a identificação do relacionamento dos Referenciados com os recursos multimídias disponíveis no serviço, como páginas, grupos, curtidas, entre outros tipos de conteúdo, em diversos formatos. Em síntese, a instituição detentora do serviço consegue isolar e agrupar quaisquer recursos multimídia que circulem na rede, de qualquer Referenciado, independentemente de ter ou não conta no serviço.

A internet ampliou o público que gera e que têm acesso ao conteúdo dos Referenciados, e as instituições proprietárias destes serviços desenvolveram ferramentas para análises estatísticas e canais de acesso de dados dos Referenciados por Agentes Externos, além de oferecer plataformas com sistemas de entrada contendo permissões de acesso gratuito e prêmio¹ a estes serviços.

Nessas ferramentas e canais oferecidos nos Serviços de Redes Sociais *Online* é possível configurar uma série de opções sobre a forma de controle de acesso, e sobre os requisitos necessários para visualizar e para interagir com os conteúdos dos Referenciados. No entanto, a forma de operação e a troca de conteúdo com Agentes Externos – ou seja, quaisquer Referenciados que queiram acessar os conteúdos – ficam em segundo plano, em parte devido à complexidade da interface de comunicação entre instituição detentora do serviço e Agentes Externos. A quantidade disponível de recursos, de telas, de formulários e de documentos técnicos relacionados com a forma de compartilhamento dos conteúdos, diminui a percepção e a transparência dos caminhos que estes conteúdos trafegam e os participantes desta interação (Rodrigues; Sant’ana, 2016).

¹ Popularmente se utiliza o termo na língua inglesa, *premium*.

Este processo de troca de dados entre dois sistemas de informação é chamado de interoperabilidade, uma

[...] característica de um produto ou sistema, na qual as interfaces são completamente compreendidas, para funcionar com outros produtos ou sistemas, no presente ou futuro, em qualquer aplicação ou acesso, sem quaisquer restrições (Interoperability Working Group, 2016, tradução nossa).

A quantidade de Referenciados, o volume de conteúdos relacionados aos seus perfis, e a quantidade de Agentes Externos, com aplicativos desenvolvidos para intercambiar conjuntos de dados aumentaram de forma análoga a uma progressão geométrica. A partir do ano de 2010, os Serviços de Redes Sociais *Online* apresentam quantidades expressivas de Referenciados, como relatado em pesquisas de Adamic e Adar (2003), de Acquisti e Gross (2006), de Mislove *et al.* (2007), de Boyd e Ellison (2007), de Fogel e Nehmad (2009) e de Rodrigues e Sant'Ana (2016).

O total de Referenciados utilizando esses serviços tem alta representatividade em relação ao total da população mundial (Statista, 2023a, 2023b). Para Donath (2007), os Serviços de Redes Sociais *Online* que atingem bilhões e centenas de milhares de Referenciados passam a ser super-redes sociais, parte integrante do conjunto dos serviços mais acessados e utilizados através da infraestrutura da internet (ALEXA, 2021), oferecidos por instituições multinacionais, com escritórios regionais localizados em cidades de diversos países.

Como se trata de serviços ofertados por instituições privadas, os *websites* e os aplicativos disponibilizam uma série de documentos contendo os Termos de Serviço, e outras seções vinculadas, voltadas a oferecer informações sobre os dados pessoais, o processo de compartilhamento de dados a Agentes Externos, os guias de convivência, os processos de remoção de conteúdo e de exclusão de conta, as orientações sobre a veiculação de propagandas e material publicitário, entre outros.

1.1 MODELO DE NEGÓCIOS BASEADO NA OFERTA DE SERVIÇO

O modelo de negócio de venda de software passou por uma transformação, a partir da década de 2000, aonde a aquisição tradicional de software – também conhecida como venda de prateleira, no qual a aquisição é realizada sob a forma de compra de produtos em redes varejistas – começa a conviver com outras formas de aquisição e distribuição (Buxmann; Hess; Lehmann, 2008; Dubey; Wagle, 2007).

Nos primeiros anos da década de 2000, essa transformação na forma em que se vendia softwares sofreu forte influência pela disponibilidade de conexão de internet – globalizada e com maior capacidade de transmissão – pela quantidade e portabilidade de dispositivos – aumentando tanto o uso de aplicativos e a variedade de tipos desenvolvidos, incluindo preocupações estéticas como o uso de interfaces para navegadores e aplicativos voltados a telas de tamanho reduzido, tais como os dispositivos móveis.

Esta nova percepção sobre a forma de venda de software fez com que as instituições comesçassem a adotar uma preferência pela liberação do pagamento de taxas para o uso dos aplicativos e, ao mesmo tempo, iniciassem um processo de desenvolvimento de novos modelos para a rentabilidade dos negócios. As formas de aquisição de software foram beneficiadas pelo estabelecimento da nova infraestrutura propiciada pela internet, principalmente ao utilizar essa infraestrutura tanto como uma forma de distribuição de software *online* aos clientes, quanto para a elaboração de novos serviços.

Este modelo novo modelo beneficiou as instituições, ao continuar possuindo o controle dos bancos de dados e dos códigos-fonte de seus sistemas de informação – a partir do uso da arquitetura cliente/servidor – além de centralizar as questões de controle de acesso aos usuários, permitir a atualização e correção de erros à distância no código-fonte pela origem e combater à pirataria.

A forma de distribuição *online* de aplicativos é denominada como Distribuição de Software *Online*², porém é parte integrante de um modelo de negócio mais amplo, o Software como Serviço³ – o SaaS, parte integrante da Computação Orientada a Serviços⁴ (Dubey; Wagle, 2007; Papazoglou, 2003). O SaaS tem origem no conceito *Application Service Provider* (ASP), da década de 1990, diferenciando-se principalmente deste em sua abrangência em número de usuários.

As características do SaaS são (Buxmann; Hess; Lehmann, 2008; Dubey; Wagle, 2007; Papazoglou, 2003; Turner; Budgen; Brereton, 2003):

- a) O uso da infraestrutura da internet para o acesso aos serviços oferecidos, a sua distribuição e a atualização, bem como a adição e a subtração de funcionalidades;
- b) A neutralidade tecnológica e a concomitância com o *cross-platform*, ou seja, uma aplicação deve requerer o mínimo possível de pré-requisitos locais de equipamentos para o seu funcionamento e a menor dependência possível de tecnologias instaladas. Exemplos são os aplicativos que funcionam na maioria dos sistemas operacionais (incluindo os sistemas operacionais de dispositivos móveis) ou que funcionam na maioria dos navegadores;
- c) O desenvolvimento de novos módulos deve ser de livre acoplamento nos existentes, evitando que a adição ou a remoção de novas funcionalidades exijam a alteração de todo o código-fonte, sendo comum o uso de técnicas como a programação orientada a objetos;
- d) A existência de uma camada de transporte dos conjuntos de dados armazenados nestes serviços para a aplicação (em que o usuário acessa), através do uso de linguagens de marcação (como o

² No idioma inglês, *On-line delivery of software*.

³ No idioma inglês, *Software as a Service* (SaaS).

⁴ No idioma inglês, *Service-Oriented Computing* (SOC).

JavaScript Object Notation – JSON e o *eXtensible Markup Language* – XML), de padrões fechados ou abertos⁵ e de protocolos;

- e) Em alguns casos, apresenta interfaces de interoperabilidade para a comunicação com outros aplicativos ou sistemas de informação, como as APIs.

Ou seja, o SaaS se beneficia da infraestrutura pós-2000 da internet – de abrangência global e presente em quase todos os países – para que um único aplicativo seja utilizado por inúmeras instituições, facilitando não só a atualização e a manutenção, mas também aumentando a lucratividade de seus produtos, como no caso dos Serviços de Redes Sociais *Online*, dos serviços de *streaming*, e de outras aplicações que atuam em diferentes países e em diferentes idiomas (Buxmann; Hess; Lehmann, 2008; Gold *et al.*, 2004).

O SaaS possui íntima relação com a Computação orientada a Serviços, definida como:

[...] o paradigma computacional que utiliza serviços como elementos fundamentais para o desenvolvimento de aplicações e soluções. Serviços são autodescritivos, independentes de plataforma computacional própria que suporta a composição rápida e de baixo custo de aplicações distribuídas (Papazoglou, 2003, p. 1, tradução nossa).

Gold *et al.* (2004) complementam que a Computação orientada a Serviços também inclui elementos como a terceirização – de funções especializadas de uma instituição para outros parceiros – e o fornecimento de aplicações prontas para o uso – por outras instituições, como o desenvolvimento e a manutenção de aplicativos, e o armazenamento dos bancos de dados. Por exemplo, uma instituição pode terceirizar a função de relacionamento com seus clientes a outra instituição – especializada neste tipo de

⁵ No idioma inglês, *Open Standards*.

atividade – e não adquirir um software para controlar estas atividades, e sim adquirir um serviço *online* para acompanhar e gerenciar esta atividade. É análogo ao serviço de distribuição de energia elétrica: no momento da assinatura do contrato, não se estabelece um contrato de aquisição de equipamentos para recepção da energia elétrica da rede, e sim a contratação do serviço de distribuição de energia elétrica.

No ambiente doméstico, um indivíduo pode adquirir um espaço mensal em servidores de empresas especializadas em gerenciamento de conteúdo para armazenar suas fotografias pessoais. Neste cenário, as fotografias do indivíduo não estarão mais armazenadas em um computador pessoal. O indivíduo passa a acessar um serviço *online* que gerencia, armazena, organiza e controla o acesso de suas fotografias.

Portanto, a Computação orientada a Serviços transforma o modo de uso de e comercialização de aplicativos em uma relação de prestação de serviço entre instituição e usuários – ao elaborar um mecanismo de instalação e de atualização de aplicativos através da infraestrutura da internet, as instituições que desenvolvem estes aplicativos passam a comercializar não mais o produto, e sim o serviço prestado pelo aplicativo, mesmo os casos que necessitam da instalação de um aplicativo (cliente) local como, por exemplo, ao adquirir um antivírus: o usuário instala um aplicativo local, mas está comprando o serviço de proteção e não mais o aplicativo de antivírus (Buxmann; Hess; Lehmann, 2008; Dubey; Wagle, 2007).

Um outro ponto fundamental no modelo de negócios é que o SaaS e a Computação orientada a Serviços permitiram que serviços começassem a ter parte dos seus dados interoperáveis, como, por exemplo, serviços postais *online*, que um aplicativo externo pode enviar uma requisição de coleta de dados para um determinado serviço postal, contendo como variável o valor de um código postal (*e.g.* um Código de Endereçamento Postal (CEP) de um endereço), e receber um conjunto de dados com a localização daquele código postal (*e.g.* o tipo de logradouro, o nome do logradouro, o

bairro, o município e o estado). A interoperabilidade é fruto da implementação das APIs neste contexto.

1.2 O USO DE *APPLICATION PROGRAMMING INTERFACES*

Como a maioria dos Serviços de Redes Sociais *Online* surgiram no contexto de um modelo de negócio orientado a serviço, as instituições detentoras elaboram ferramentas para habilitar o desenvolvimento por Agentes Externos de aplicações, permitindo interoperabilidade de conjuntos de dados de Referenciados, inclusive disponibilizando em seus *websites* documentação técnica dos processos de autorização e de permissão de acesso, e da forma de requisição para a coleta e a manipulação de dados, sendo que a principal forma de instrumentalizar este processo de interoperabilidade é por meio das APIs. As instituições detentoras dos serviços também atrelam documentos jurídicos para delimitar as condições legais de interoperabilidade, também inclusos nos Termos de Serviço, por meio de documentação adicional.

A origem do uso de API para interoperabilidade de dados entre sistemas de informação não é consensual na literatura (Lane, 2016). Entretanto, a forma de utilização de API em serviços disponíveis através da internet surge a partir de estudos de Fielding (2000), que propõe uma arquitetura para interoperabilidade de conjuntos de dados por aplicações baseadas no uso da internet como uma infraestrutura de comunicação entre sistemas. É baseada em diversas estruturas anteriores para a troca de dados, desenvolvidas até o final dos anos 1990, tais como o *Web Interface Definition Language* (WIDL), o *eXtensible Markup Language - Remote Procedure Calling* (XML-RPC), o *Web Distributed Data Exchange* (WDDX), entre outros (Box *et al.*, 2000; Cover, 1998; Fielding, 2000; Gourraud, 2002; Lane, 2016; Merrick; Allen, 1997; Wikipedia Contributors, 2023; Winer, 2003).

Considera-se que a API é uma estrutura formal de regras e protocolos para proporcionar a interoperabilidade de conjunto de dados, por dois

ou mais serviços, independentes de plataforma, de acesso público, privado ou misto, que utiliza padrões abertos ou fechados para o intercâmbio dos dados e contém documentação disponível na origem para o entendimento de todas as partes sobre o seu modo de operacionalização.

Além disso, uma API pode possuir várias versões e são designadas numerações para controlar a manutenção e a atualização das operações disponíveis. Por exemplo, em uma determinada versão, uma operação de coleta de conjuntos de dados pode ser inserida, atualizada removida.

As APIs de Serviços de Redes Sociais *Online* precisam receber um ponto inicial de consulta nos mecanismos para iniciar o processo de requisição e de interoperabilidade seus conjuntos de dados. Por exemplo, ao coletar dados de Referenciados via API, a ideia primária é iniciar a coleta de dados a partir do envio de um identificador único de um Referenciado para a solicitação de seus dados (Bolosky *et al.*, 2005; Fielding, 2000).

Esta característica de acesso e coleta de conjuntos de dados em diversos formatos propiciado pela API é denominado *Wire Protocol*, uma espécie de protocolo de ligações por arestas. A diferença deste protocolo em comparação aos outros protocolos de transmissão de dados é a necessidade do uso das arquiteturas disponíveis dos sistemas operacionais instalados na origem e no destino para realizar a transmissão (Bolosky *et al.*, 2005; Fielding, 2000; Gourraud, 2002; Linthicum; O'Brien, 2000).

Ou seja, as APIs se beneficiam de protocolos disponíveis nos dispositivos e nos sistemas operacionais para transmissão de dados (na origem e no destino) – para o envio e recebimento das consultas. Isso dá uma vantagem aos desenvolvedores de software, ao permitir a utilização de estruturas já existentes para a transmissão e a recepção de dados, permitindo que Agentes Externos se preocupem com camadas mais altas de abstração, ou seja, se preocupem somente em implementar a interoperabilidade via API, ao invés de ter que desenvolver protocolos de funcionamento da rede, como o Protocolo de Controle de Transmissão/Protocolo Internet (em inglês, *Transmission Control Protocol/Internet Protocol*) – o TCP/IP.

Na prática, o *Wire Protocol* proporciona que os conjuntos de dados dos serviços se comuniquem com as aplicações através de um sistema de comunicação ponto a ponto, aonde as operações disponíveis compõem os pontos de entrada disponíveis para a coleta de dados, com as etapas (Bolosky *et al.*, 2005; Fielding, 2000; Singla; Richardson, 2008):

- a) Requisição do conjunto de dados: a aplicação envia uma mensagem contendo parâmetros para designar quais são os conjuntos de dados a serem coletados, geralmente com o uso de um identificador único, e de credenciais de autorização e de permissões de acesso;
- b) Processamento: o serviço processa se a requisição é válida, a partir de regras preestabelecidas e disponíveis na documentação da API;
- c) Resposta e envio do conjunto de dados: o serviço envia para a aplicação a resposta da requisição, através da aplicação de estruturas de transferência de dados, como o *Representational State Transfer*⁶ (REST), o uso de linguagens de marcação e de esquemas para a validação do conteúdo transferido.

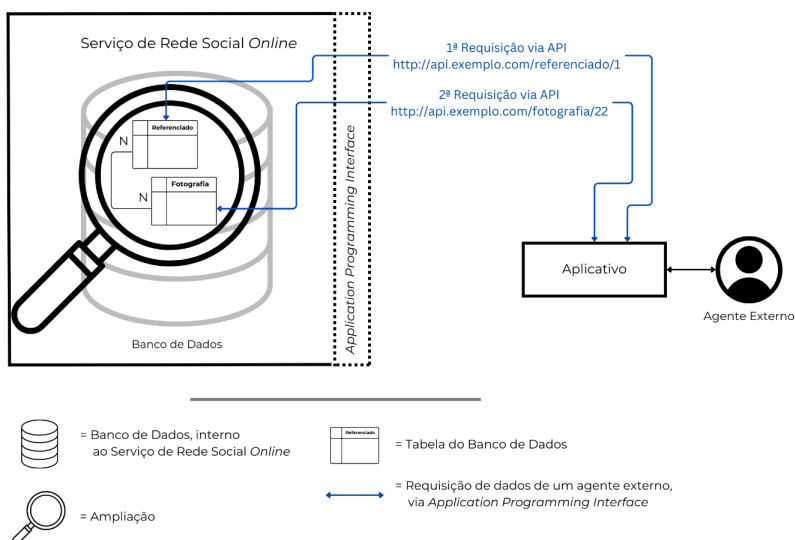
Em adição, as requisições são temáticas e utilizam o conceito de encapsulamento: construtores que facilitam a elaboração destas funções específicas de requisição/processamento/resposta. Para cada tipo de conteúdo a ser coletado em uma API é apresentada uma função específica, em que deve se utilizar parâmetros de entrada, que são variáveis pré-definidas contendo dados informativos sobre o que se deseja coletar. O acesso a estas funções de requisição pode ser controlado, com as seguintes condições: públicas (acessíveis), protegidas (controlado o acesso externo) e privadas (sem acesso externo), além de não ser obrigatório o retorno de todos os dados disponíveis (Connolly; Begg, 2015; Silberschatz; Korth; Sudarshan, 2020; Singla; Richardson, 2008).

⁶ No caso das API de Serviços de Redes Sociais *Online*, o REST é uma das formas mais utilizadas para o intercâmbio de dados.

Portanto, em uma API, os conjuntos de dados que estão contidos em um Banco de Dados de um Serviço de Rede Social *Online* podem ser recuperados por uma aresta, em uma requisição encapsulada, em que o acesso pode ser público ou controlado, no qual o Agente Externo deve enviar variáveis sobre quais conjuntos de dados deseja coletar, e a resposta da requisição encapsulada não necessariamente retornará todos os dados disponíveis.

A Figura 2 apresenta um exemplo da coleta de dados em uma API de um Serviço de Rede Social *Online* por um aplicativo de um Agente Externo, utilizando o protocolo de REST. Este aplicativo externo, hipotético, foi desenvolvido por uma instituição diferente da mantenedora do serviço, e possui, a princípio: a) autorização de acesso a API, b) códigos-fonte desenvolvidos a partir de conhecimentos adquiridos na documentação de referência técnica do funcionamento da coleta de dados pela API, e c) o identificador único de um determinado Referenciado (no exemplo, o número 1).

Figura 2 – Exemplo de coleta de dados de um Serviço de Rede Social *Online*, via API, por um aplicativo de um Agente Externo



Fonte: Elaborado pelo Autor.

Para coletar os dados de um Referenciado (1ª Requisição), o aplicativo do Agente Externo solicita uma resposta contendo um conjunto de dados de Referenciados, utilizando a requisição com estrutura pré-definida e encapsulada (protegida) <http://api.exemplo.com/referenciado/>, enviando como variável de consulta o identificador único de um Referenciado (identificado no banco de dados como Referenciado número 1), através da composição 'requisição + identificador único', com por meio do protocolo de comunicação HTTP.

No exemplo, cada Referenciado pode estar vinculado de zero a infinitas fotografias, e as fotografias podem estar vinculadas a um ou infinitos Referenciados (no exemplo, esta relação é explicitada pelas letras N). Caso a resposta da primeira requisição apresente uma lista contendo os identificadores únicos das fotografias vinculadas ao usuário número 1, a aplicação do Agente Externo pode solicitar uma segunda requisição para coletar os dados das fotografias.

Entretanto o endereço de requisição com estrutura pré-definida e encapsulada (protegida) difere-se da primeira (<http://api.exemplo.com/fotografia/>), pois serão coletados conjuntos de dados distintos (no caso, identificado no banco de dados como fotografia número 22).

No exemplo, os números 1 e 22 representam identificadores, ou seja, dados que representam dois registros, respectivamente: um Referenciado e uma fotografia. Isto só é possível pois certos dados podem subsidiar a identificação de cada registro, sendo classificados em identificadores, semi-identificadores e identificadores em potencial (Affonso; Sant'ana, 2015; Fung, 2011; Samarati; Sweeney, 1998).

Os identificadores são dados que possui a condição de identificar um conjunto de dados dentro e fora de domínio original, ou seja, além de identificar unicamente um conjunto de dados no banco de dados de origem, é possível relacioná-lo com conjunto de dados originários de outros bancos de dados. Um exemplo é a disponibilidade da existência de dados sensíveis na ocasião da coleta de dados em uma API (*e.g.* um código de

identificação único, um número documento de identidade e um endereço de e-mail) que permite, *a posteriori*, relacionar este conjunto de dados com outros bancos de dados.

Já os semi-identificadores são dados que aparentam não um determinado conjunto de dados, porém seu poder de identificação é potencializado ao combiná-lo com dados originários de outros bancos de dados. Por exemplo, no conjunto de dados de mensagens postadas em um Serviço de Rede Social *Online* podem existir informações como a data de postagem, dados de georreferenciamento, dados do Referenciado e o corpo de texto de mensagens que relatam a infecção de uma determinada doença (*e.g.* uma mensagem de um Referenciado dizendo que se contaminou com o vírus da COVID-19). Um Agente Externo pode combinar estes dados coletados com bancos de dados de recursos humanos, como dados de seleção e recrutamento, e determinar o percentual de probabilidade de um candidato a uma vaga de trabalho ter contraído este tipo de doença (por morar na mesma localidade) ou até mesmo de ser o mesmo indivíduo.

Os identificadores em potencial são dados que não possuem uma característica de identificação, porém ao combiná-los com outros dados do mesmo banco de dados – e com dados externos – podem potencializar o poder de identificação. Por exemplo, ao combinar dados de Referenciados de um Serviço de Rede Social *Online* para perfis profissionais com informações sobre *check-in* em restaurantes, associá-los ao nome do Referenciado e a outras informações derivadas de outros bancos de dados, é possível identificar o Referenciado em outros domínios, como o uso de dados de Referenciados em publicidades à comunidade, comportamento ligado a máxima: ‘se Referenciado já esteve aqui, é porque deve ser bom’. Separados, os conjuntos de dados não têm poder de identificação, mas ao agregá-los seu poder de identificação é ampliado, inclusive identificando o Referenciado no mundo real.

Cabe aqui uma reflexão. No caso dos Serviços de Rede Sociais *Online*, o fator complicador está no uso cada vez mais intenso de um ser-

viço de inter-relacionamento que, no fundo, é um software negociado nos moldes do século XX: acesso gratuito aos Referenciados, com a lucratividade voltada à venda de anúncios, de contas prêmio e de acesso pago a dados dos Referenciados. A cada dia se expõe mais dados pessoais em Serviços de Rede Sociais *Online* e outros serviços de forma deliberada, consciente ou inconsciente, e parte destes dados são originários de informações da esfera privada: opiniões sobre uma fotografia, locais frequentados, comidas favoritas, atividades que deram certo e que deram errado; as opiniões políticas, filosóficas, científicas e artísticas – e, *quod erat demonstrandum*, são serviços que possuem API com acesso legalizado a Agentes Externos, que utilizam estes dados para – cotidianamente – conhecer melhor a todos.

Winsborough, Lovric e Chamorro-Premuzic (2016, tradução nossa) expõem dois cenários para os próximos anos sobre aspectos relacionados à personalidade, à privacidade e ao ‘eu digital’.

No primeiro cenário, nossos eu digital estão fora do nosso controle. Nossos dados pessoais é propriedade do hardware e do software que utilizamos, e são vendidos pelo maior lance. A anonimidade e a privacidade estão em demanda, mas é muito caro – não há nenhuma cláusula para saída viável. Neste mundo, todo nosso engajamento com o mundo digital cria comida para o marketing e para a engenharia social em proporções Orwellianas. Aqui nós estamos falando sobre a perda fundamental da agência, um mundo no qual a nossa individualidade é usada como forragem de base para o consumo e manipulação. Quando Sócrates nos exortou a conhece-te a ti mesmo, é duvidoso que ele já entretido a possibilidade de que possamos viver em um mundo onde os outros nos conhece melhor do que nós podemos conhecer a nós mesmos, e usar esse conhecimento para assumir a nossa agência.

É importante ressaltar que as estruturas das APIs dos Serviços de Redes Sociais *Online* podem restringir o acesso à parte dos conjuntos de dados e também apresentar acesso a conjuntos de dados de outras temáticas, como os baseados de análises estatísticas e os de variáveis da configura-

ção de contas e perfis de usuários – de forma análoga ao uso de visões em um Sistemas de Gerenciamento de Bancos de Dados⁷.

Em outras palavras, as restrições de acesso a partir do encapsulamento das requisições não são suficientes para garantir aspectos relacionados a segurança dos dados. Principalmente no caso dos Serviços de Redes Sociais *Online*, as APIs são acessíveis a diversas instituições para um largo espectro de uso e de compartilhamento dos conjuntos de dados pessoas de Referenciados – ou até com parte dos dados sendo acessíveis de forma pública.

Além disso, não é recomendado que acessos externos sejam realizados diretamente nas tabelas dos Sistemas de Gerenciamento de Bancos de Dados desses serviços, podendo ser passíveis de ataques que podem comprometer a segurança dos conjuntos de dados armazenados, como o uso de técnicas similares: ao *Structured Query Language Injection* (SQL Injection), que permite a execução de uma consulta e alteração dos conjuntos de dados à revelia das regras de encapsulamento ou das regras de restrições de segurança adotadas e ao XML *rewriting attack* que permite a interceptação e alteração de conjuntos de dados transmitidos em formato XML, além de outras formas de exploração (Boyd, 2015; Rahaman; Schaad; Rits, 2006).

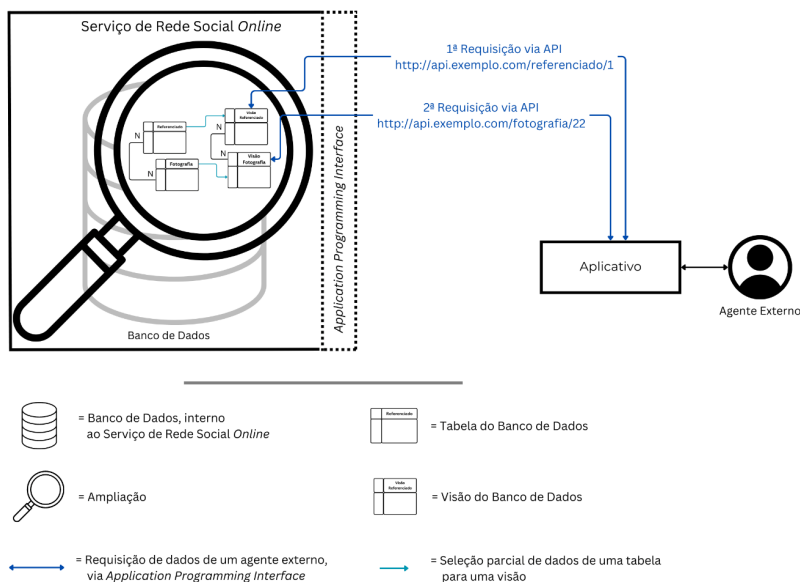
Os serviços devem oferecer acesso via API apenas de uma camada de abstração mais alta, em que são apresentadas apenas as visões (Figura 3). A instituição detentora do Serviço de Rede Social *Online* prepara uma camada de visão das tabelas do Sistema de Gerenciamento de Bancos de Dados (Figura 3, elementos com as nomenclaturas Visão Referenciado e Visão Fotografia), contendo uma seleção pré-definida dos conjuntos de dados das tabelas, por meio de uma supressão parcial de dados considerados sensíveis a segurança, como, por exemplo, a possibilidade de coleta da senha do Referenciado, via API, pelo Agente Externo.

⁷ No idioma inglês, *Data Base Management Systems*.

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Figura 3 – Exemplo de coleta de dados de um Serviço de Rede Social *Online*, via API, por um aplicativo de um Agente Externo, com o uso de visões



Fonte: Elaborado pelo Autor.

Portanto, neste contexto as visões são uma seleção de atributos e de relacionamentos personalizados, a partir de dados contidos em uma ou mais tabelas do banco de dados. As visões têm a capacidade de restringir o acesso a elementos do nível lógico e físico, e o acesso e a coleta dos dados contidos nas visões são idênticas ao acesso e a coleta de dados em tabelas, exceto que nas visões não é possível inserir, alterar ou excluir linhas (Silberschatz; Korth; Sudarshan, 2020).

No exemplo, a requisição encapsulada de coleta aos conjuntos de dados de Referenciados e suas fotografias, realizada por uma Agente Externo, não acessa diretamente as tabelas no Sistema de Gerenciamento de Bancos de Dados. Portanto, cada requisição é ponto de entrada pré-delimitado, com acesso a conjuntos de dados específicos, como, por exemplo, conjuntos de dados relacionados a eventos que o Referenciado participará, ou da-

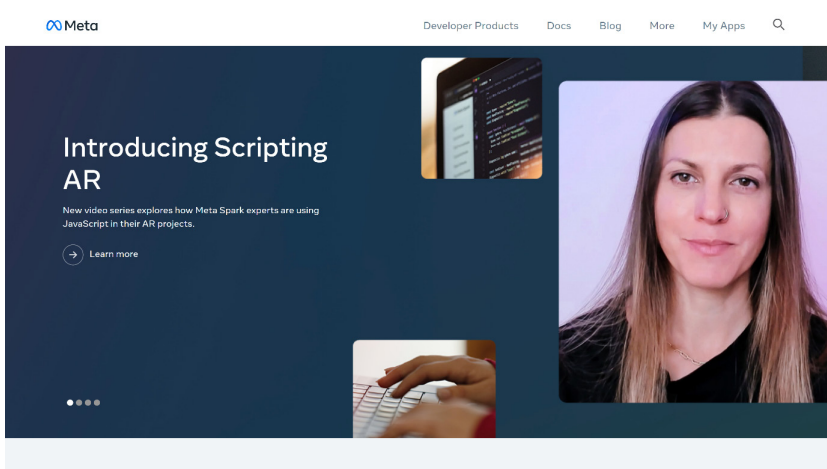
dos relacionados as fotografias do Referenciado; e estes conjuntos de dados podem ter identificadores que os vinculam a outros conteúdos, permitindo a coleta de dados destes vínculos – todavia com uma seleção pré-definida de conjuntos de dados que estarão disponíveis aos Agentes Externos.

Portanto, é uma falsa percepção determinar que todo e qualquer dado em um Serviço de Rede Social *Online* está disponível para coleta por API. Portanto, as próximas seções detalham as estruturas das APIs dos Serviços de Redes Sociais *Online* analisados: a *Facebook Graph* API, a *X REST* API e a *LinkedIn REST* API.

1.3 A FACEBOOK GRAPH API

Para acessar os documentos técnicos da *Facebook Graph* API, a *Meta Platforms, Inc.* possui uma área denominada *Meta for Developers* em seu *website* (Figura 4). Esta área é exclusiva para a concentração de documentos, de referências, de exemplos e de demais informações referentes ao processo de utilização de serviços oferecidos para Agentes Externos e parceiros.

Figura 4 – *Meta Platforms, Inc.*: Página principal da área reservada aos desenvolvedores



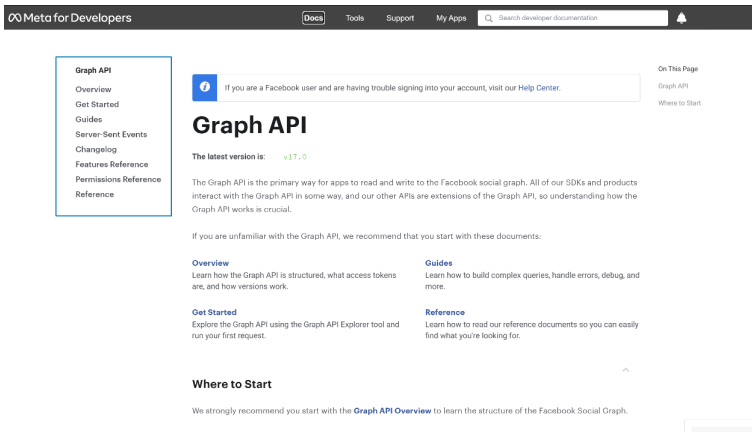
Fonte: Recorte de *Meta Platforms, Inc.* (2023b), elaborado pelo Autor.

A seção Documentação (*Docs*) contém as coleções de documentos técnicos referentes aos produtos oferecidos pela *Meta Platforms, Inc.* aos Agentes Externos, especificamente à comunidade de desenvolvedores de aplicações externas. É formado pela documentação de referência dos Kits de Desenvolvimento de Software (no idioma inglês, *Software Development Kit*), política de uso, informações sobre questões de privacidade e mecanismos automáticos de tradução, e documentação e guia de referência sobre as APIs disponíveis.

A *Facebook Graph* API é a principal API para o Serviço de Rede Social *Online Facebook*. Foi desenvolvida como o objetivo de realizar a interoperabilidade de conjuntos de dados originários do serviço com aplicações externas. As requisições e as respostas das requisições estão acessíveis através do protocolo HTTP, tanto para a coleta de dados quanto para realização de operações orientadas a inserção, a alteração ou a exclusão de dados (Meta Platforms, INC., 2023c). Para a versão 2.6 *Facebook* da *Graph* API, os documentos estão disponíveis somente em inglês.

O conjunto de documentos contendo as descrições de sua funcionalidade estão divididos em oito seções (Figura 5, destaque no retângulo azul): Visão Geral, Começar, Guias, Eventos enviados pelo Servidor, *Log* de Diário de Mudanças; Referências em Destaque; Referência de Permissões, e; Referência.

Figura 5 – *Meta Platforms, Inc.*: Seções disponíveis dos documentos da *Facebook Graph API*



Fonte: Recorte de *Meta Platforms, Inc.* (2023c), elaborado pelo Autor.

As informações relacionadas com os conjuntos de dados disponíveis, atributos, relacionamentos entre atributos e demais conjuntos de dados, bem como a descrição destes, encontram-se na seção Referências (*Meta Platforms, INC.*, 2023d) – sendo esta seção o escopo deste livro.

No momento da coleta, os conjuntos de dados da *Facebook Graph API* são apresentados na forma de um sociograma. Os conjuntos de dados disponíveis para coleta são pré-determinados e selecionados pela instituição, na forma de visões. Cada visão é tratada como um nó do sociograma.

Para coletar os conjuntos de dados de um Referenciado, o Agente Externo deve realizar uma requisição de acesso ao nó *User*. Esse nó possui características idênticas com o processo de requisição de uma visão (ver Figura 3, seção 1.1). Portanto, cada nó contém uma coleção de atributos e registros, unicamente identificáveis. Os atributos possuem um número como identificador de cada registro armazenado, independente da visão, denominado *id*. Não há atributos *id* com valores iguais, mesmo para atributos pertencentes a diferentes visões.

Os relacionamentos entre as visões são denominados arestas – que representam as conexões entre os nós. As visões estão relacionadas através de valores em um ou mais atributos na visão de origem e na visão de destino.

Os atributos disponíveis em uma visão também são denominados como campos na documentação. Os campos são idênticos aos atributos, pois contém um nome e um valor, que pode ser simples ou composto. Por exemplo, para um determinado Referenciado, o atributo nome possui o valor Albert Einstein, sendo do tipo simples. Já um atributo que permite a coleta dos amigos do Referenciado possui como valor uma lista de *ids*, sendo que cada *id* representa um amigo, sendo do tipo composto.

As requisições são realizadas por meio do uso do protocolo *Hyper Text Transfer Protocol Secure* (HTTPS), pela *Uniform Resource Locator* (URL) *https://graph.facebook.com*. Para coletar dados de uma visão é necessário realizar a composição dos seguintes elementos: a) URL da API, b) separador, representado pelo símbolo de barra, e c) valor do atributo *id*.

Por exemplo, para acessar os conjuntos de dados do Referenciado de nome Fernando de Assis Rodrigues (visão *User*), é necessário enviar pelo método GET do protocolo HTTPS a requisição do Exemplo 1.

Exemplo 1 – *Facebook Graph* API: Requisição para coleta de conjuntos de dados, via método GET

```
GET graph.facebook.com/100000309621709
```

Fonte: Elaborado pelo Autor.

O número 100000309621709 representa o valor do atributo *id*, sendo que este atributo é o identificador que representa unicamente este Referenciado na visão *User*. Entretanto, é importante observar que não é necessário enviar um parâmetro contendo o nome da visão, pois para a *Facebook Graph* API cada valor de *id* é único.

Os resultados das requisições da *Facebook Graph* API são apresentados no formato de notação da linguagem de marcação JSON. No exemplo da requisição solicitada de conjunto de dados do Referenciado com o *id* de número 100000309621709, a resposta da requisição é apresentada conforme o Exemplo 2.

Exemplo 2 – *Facebook Graph* API: Resultado da requisição para coleta de conjuntos de dados, via método GET

```
{
  "name": "Fernando de Assis Rodrigues",
  "id": "100000309621709"
}
```

Fonte: Elaborado pelo Autor.

Os caracteres chaves { e } indicam o início e o final do registro. Tanto os nomes dos atributos como os seus respectivos valores são apresentados dentro das duas chaves – uma no início e outra no final de cada resposta de requisição.

Cada atributo é representado da seguinte forma: o nome do atributo entre aspas duplas, seguido do símbolo de dois pontos (separador entre nome do atributo e valor) e do valor do atributo (que pode ter ou não aspas duplas no início e no final de seu valor, dependendo do tipo de dado). Os atributos de uma mesma visão são separados entre si pelo símbolo de vírgula.

No Exemplo 2, a solicitação retornou um registro da visão *User*, a partir do uso como parâmetro o valor do atributo *id*. O resultado apresentado para a coleta de dados contém dois atributos, de nomes *name* e *id*, com os valores (respectivamente): Fernando de Assis Rodrigues e 100000309621709.

Os atributos *name* e *id* são retornados automaticamente, pois são parte integrante do conjunto de atributos marcados como Padrão (no idioma inglês, *Default*) para esta visão. Caso a coleta de dados necessite de mais atributos da visão *User*, é necessário a utilização um novo parâmetro no momento da requisição, denominado *fields*, e o seu valor deve ser preenchido com os nomes dos atributos que se deseja coletar, separados pelo símbolo vírgula.

Por exemplo, para coletar os conjuntos de dados do Referenciado com o *id* de número 100000309621709 (visão *User*), contendo na requisição os valores dos atributos *birthday* (data de aniversário), *email* (e-mail do usuário) e *name* (nome do usuário), será necessário realizar uma nova composição, conforme o Exemplo 3.

Exemplo 3 – *Facebook Graph* API: Requisição para coleta de conjuntos de dados, com parâmetros adicionais, via método GET

```
GET graph.facebook.com/100000309621709?fields=birthday,email,name
```

Fonte: Elaborado pelo Autor.

O símbolo de interrogação separa o valor do atributo *id* e o primeiro parâmetro. No caso, o parâmetro adicional tem o nome de *fields*, um separador representado pelo símbolo de igualdade, e o valor sendo uma lista de três palavras, separadas pelo símbolo da vírgula. Caso seja necessário a adição de outro parâmetro, deve-se utilizar um separador entre os parâmetros, representado pelo símbolo *&*.

As respostas das requisições com a adição do parâmetro *fields* não sofrem alterações em sua estrutura, exceto a adição dos atributos na resposta, conforme os valores do parâmetro, ilustrado no Exemplo 4.

Exemplo 4 – *Facebook Graph* API: Resultado da requisição para coleta de conjuntos de dados, com parâmetros adicionais, via método GET

```
{  
  "birthday": "01/01/1900",  
  "email": "fernando.assis@unesp.br",  
  "name": "Fernando de Assis Rodrigues",  
  "id": "100000309621709"  
}
```

Fonte: Elaborado pelo Autor.

A *Facebook Graph* API permite a realização de requisições contendo dados de outras visões, a partir de seus relacionamentos entre visões. Para o acesso a estes dados, deve-se realizar a requisição destas visões (doravante denominadas como visões de destino), a partir do acesso a uma visão de origem, com o envio do parâmetro *id* contendo o valor deste atributo na visão de origem.

Para a requisição e a coleta de dados de visões de destino, é necessário realizar a composição dos seguintes elementos (Exemplo 5): a) URL da API, b) separador, representado símbolo de barra, c) valor do atributo *id* da visão de origem, d) separador, representado símbolo de barra e, e) nome do relacionamento com a visão de destino (informação disponível na documentação de cada visão).

Exemplo 5 – *Facebook Graph* API: Requisição para coleta de conjuntos de dados de visões relacionadas, via método GET

```
GET graph.facebook.com/100000309621709/friends
```

Fonte: Elaborado pelo Autor.

Por exemplo, para a requisição dos conjuntos de dados sobre relacionamentos do Referenciado de nome Fernando de Assis Rodrigues (visão de origem *User*) com outros Referenciados do *Facebook*, é necessário enviar tanto o *id* da visão *User* como o nome de relacionamento *friends*. O número 100000309621709 representa o valor do atributo *id* para o

Referenciado em questão, sendo este atributo o identificador da visão de origem *User*.

O atributo *friends* é responsável por relacionar duas visões: de origem e de destino. Delimita que a resposta da requisição deve retornar os conjuntos de dados da visão de destino, no caso, conjuntos de dados originários da visão de destino *User* que estão relacionados com o Referenciado de nome Fernando de Assis Rodrigues. Ou seja, a partir de um Referenciado, coletar conjuntos de dados sobre seus amigos, outros Referenciados.

A resposta deste tipo de requisição segue os mesmos princípios das requisições para coleta de atributos da visão de origem, ilustrado no Exemplo 6. As chaves { e } indicam o início e o final dos registros recuperadas. Neste caso, o Referenciado com o nome Fernando de Assis Rodrigues possui um relacionamento de amizade com outros Referenciados, no qual cada relação de amizade representa dados da visão do tipo *User*, porém agora representando cada um de seus amigos.

Exemplo 6 – *Facebook Graph* API: Resultado da requisição para coleta de conjuntos de dados de visões relacionadas, com parâmetros adicionais, via método GET

```
{
  "friends": {
    "data": [
      {
        "name": "John Connor",
        "id": "641418535"
      },
      {
        "name": "Bilbo Bolseiro",
        "id": "681150264"
      },
      {
        "name": "Morpheus",
        "id": "1344445431"
      }
    ]
  }
}
```

```
]
},
"paging": {
  "cursors": {
    "before": "QVFUkRIc3ZAfenlYUnZA6Rzly",
    "after": "QVFUUm9NQy1CdIZACNI9ldXc3U"
  }
},
"summary": {
  "total_count": 1467
}
}
```

Fonte: Elaborado pelo Autor.

A visão de origem contém três atributos:

- a) *friends*: contém um atributo composto como valor, denominada *data*. O atributo *data* apresenta nenhum, um ou mais amigos (visão de destino *User*), em que cada bloco entre chaves { e } representa o conjunto de dados de um amigo, contendo os atributos *name* e *id*. Os conjuntos de dados das visões de destino (destacado em azul) são apresentados dentro de duas chaves e a cada registro é separado pelo símbolo de vírgula, representando Referenciados, de forma individual.
- b) *paging*: contém informações para a iteração (paginação) do aplicativo que coletará estes conjuntos de dados, pois a *Facebook Graph API* não permite a coleta de todos os relacionamentos entre a visão de origem e as visões de destino em uma única requisição. Ou seja, para realização de coleta de dados sobre todos os amigos relacionados com o Referenciado de nome Fernando de Assis Rodrigues, terão de ser realizadas coletas em blocos, aonde cada requisição irá permitir a coleta de 3 amigos em cada requisição;

- c) *summary*: atributo composto, que contém como valor um atributo denominado *total_count*. O valor de *total_count* representa a quantidade de vezes que a visão *User* (origem) está relacionada com visões *User* (destino) – a quantidade total de amigos do Referenciado.

Para coletar as relações de amizade pela paginação, é necessário adicionar os parâmetros *after* para relações antecessoras a requisição atual e *before* para acessar as relações sucessoras a requisição atual (informações disponíveis como valor do atributo *paging*, disponível na primeira requisição).

Também é possível adicionar na requisição outros parâmetros e formas de paginação de conteúdo, variando a disponibilidade de acordo com cada visão. Em certos cenários de coleta, estão disponíveis os parâmetros:

- a) *order* para controlar a ordem de classificação dos resultados, como, os valores *chronological* e *reverse_chronological* para ordenar – respectivamente – em ordem cronológica (mais atual para mais antigo) ou inversamente cronológica (mais antigo para mais atual). Por exemplo, controlar na coleta a ordem de classificação dos comentários de fotografias de um Referenciado;
- b) *locale* para que, na coleta de dados, os valores dos atributos sejam traduzidos automaticamente para o idioma que foi determinado no valor deste parâmetro.

A *Facebook Graph* API também possui um sistema de descoberta de conteúdo, quando não se sabe, a princípio, o valor do atributo *id* de um registro. Para coletar os dados pela descoberta de conteúdo, é necessário realizar a composição dos seguintes elementos:

- a) URL da API;
- b) Separador, representado pelo símbolo de barra;
- c) A palavra reservada *search*;

- d) Separador, representado pelo símbolo de interrogação;
- e) O parâmetro *q*, seguido do símbolo de igualdade e o texto que se deseja pesquisar nos atributos *name* das visões;
- f) Símbolo & para inserir outros parâmetros na requisição, com a finalidade de separar dois ou mais parâmetros na mesma requisição;
- g) O parâmetro *type*, seguido do símbolo de igualdade e tipo de visão que se deseja ser descoberta. Por exemplo: a palavra *page* para pesquisar na visão *Page*, a palavra *group* para pesquisar na visão *Group* e a palavra *user* para pesquisar na visão *User*.

Por exemplo, para pesquisar e coletar dados de grupos públicos do *Facebook* que contenham no atributo *name* o valor Carro, deve-se enviar a requisição do Exemplo 7.

Exemplo 7 – *Facebook Graph* API: Requisição para coleta de conjuntos de dados, por meio de descoberta de conteúdo, via método GET

```
GET graph.facebook.com/search?q=Carro&type=group
```

Fonte: Elaborado pelo Autor.

As requisições da *Facebook Graph* API também possuem um sistema para o tratamento de erros de processamento ou de autorização. Por exemplo, no caso de uma requisição estar estruturada em desacordo com as normas estabelecidas em seus documentos técnicos ou no acesso a visões não disponíveis, o resultado da requisição (Exemplo 8) estará disponível em formato de notação da linguagem de marcação JSON, com atributos específicos para esta finalidade.

Exemplo 8 – *Facebook Graph* API: Tratamento de erros de processamento ou de autorização

```
{
  "error": {
    "message": "Message describing the error",
    "type": "OAuthException",
    "code": 190,
    "error_subcode": 460,
    "error_user_title": "A title",
    "error_user_msg": "A message",
    "fbtrace_id": "EJplcsCHuLu"
  }
}
```

Fonte: Elaborado pelo Autor.

Portanto, caso uma requisição apresente erro, a *Graph* API retorna o atributo *error*, composto, contendo os seguintes atributos:

- a) *message*: contendo a descrição do erro;
- b) *type*: contendo o tipo de erro ocorrido, com a lista de possibilidades descritas na seção Usando a *Facebook Graph* API;
- c) *code*: contendo o código do erro, com a lista de possibilidades descritas na seção Usando a *Facebook Graph* API;
- d) *error_subcode*: contendo o código do erro, contendo especificações do contexto da requisição, com a lista de possibilidades descritas na seção Usando a *Facebook Graph* API;
- e) *error_user_title*: contendo o título do erro a ser apresentado ao usuário, no caso de a coleta de dados estar vinculada a um aplicativo externo;
- f) *error_user_msg*: com a mensagem do erro a ser apresentada ao usuário, no caso de a coleta de dados estar vinculada a um aplicativo externo;

- g) *fbtrace_id*: caso o Agente Externo não compreenda a razão da ocorrência, o valor deste atributo auxiliará na identificação do fato ocorrido no momento do atendimento das equipes de desenvolvimento da *Meta Platforms, Inc.*

É possível acessar funcionalidades de versões específicas da *Facebook Graph API* como, por exemplo, o uso da composição do Exemplo 9, aonde antes do primeiro separador é necessário a inserção do código da versão, seguido de um segundo separador (símbolo de barra). No exemplo, o valor *v2.6* é referente ao uso da versão 2.6.

Exemplo 9 – *Facebook Graph API*: Requisição para coleta de conjuntos de dados, a partir de uma versão específica da API, via método GET

```
GET graph.facebook.com/v2.6/
```

Fonte: Elaborado pelo Autor.

Os códigos das versões disponíveis estão na seção *Log de Mudanças*, no conjunto de documentos com as descrições de funcionalidades. Após a explicitação da versão, a forma de uso dos parâmetros é igual aos apresentados nos exemplos dessa seção.

A seção Referências apresenta a descrição de cada visão disponível para coleta de dados. Cada visão possui um documento próprio, contendo as seguintes informações (Meta Platforms, INC., 2023d):

- a) URL do documento: cada um destes documentos possui um endereço eletrônico para acesso individualizado de suas informações, concomitante ao formato URL;
- b) Título: elemento textual contendo o título da visão. Este elemento não garante a unicidade dos documentos pois há casos em que o mesmo Título da Visão define duas visões (homônimos), como no

caso do termo *Live Video*, que é o título tanto da visão que relaciona usuários a vídeos ao vivo, quanto da visão que relaciona páginas a vídeos ao vivo;

- c) Descrição da Visão (elemento não obrigatório): elemento textual, na forma de um ou mais parágrafos, descrevendo a visão;
- d) Permissões para o acesso (elemento não obrigatório): explicita as permissões necessárias para o acesso à coleta de dados da visão, e para a criação, a alteração e a exclusão de conteúdo;
- e) Leitura (elemento não obrigatório): contém exemplos, parâmetros, atributos, tipos de dados e códigos de erro específicos da visão, e relacionamentos com outras visões disponíveis no momento da coleta de dados;
- f) Criação (elemento não obrigatório): contém exemplos, parâmetros, tipos de dados e códigos de erro específicos da visão disponíveis no momento da criação de novos conjuntos de dados;
- g) Atualização (elemento não obrigatório): contém exemplos, parâmetros, tipos de dados e códigos de erro específicos da visão disponíveis no momento da atualização de conjuntos de dados;
- h) Exclusão (elemento não obrigatório): contém exemplos, parâmetros, tipos de dados e códigos de erro específicos da visão disponíveis no momento da exclusão de conjuntos de dados.

Foram identificadas 291 visões, sendo 279 vinculadas com a *Facebook Graph* API. Dentre as características identificadas, destacam-se que:

- a) Um total de 168 visões não possuem descrição;
- b) Os documentos que descrevem as visões *Live Video*, *Page Live Videos* e *User Live Videos* contém títulos iguais (homônimos). Somente a visão *Live Video* possui elemento textual com a descrição de seu

significado e está fora do escopo da análise, pois se trata de uma visão da *Facebook Live API*;

- c) Os documentos que descrevem as visões *Page Settings* e *Page Setting* contém títulos iguais: *Page Settings*. Somente a visão *Page Settings* possui elemento textual com a descrição de seu significado;
- d) Os documentos que descrevem as visões *Image Source* e *Platform Image Source* contém títulos iguais (homônimos). Ambas não possuem elemento textuais com a descrição para compreensão de seu significado e possível distinção das visões;
- e) 12 visões possuem referências e documentação de referência na *Facebook Graph API*, porém são parte integrante de outras APIs. Os Títulos destas visões são: *Ad Account*, *Ad Account Group*, *Ad Campaign*, *Ad Contract*, *App Events Export API*, *Business Manager*, *Article*, *Live Video*, *Open Graph Action*, *Open Graph Object Type*, *Video Copyright Rule* e *Video Format*.

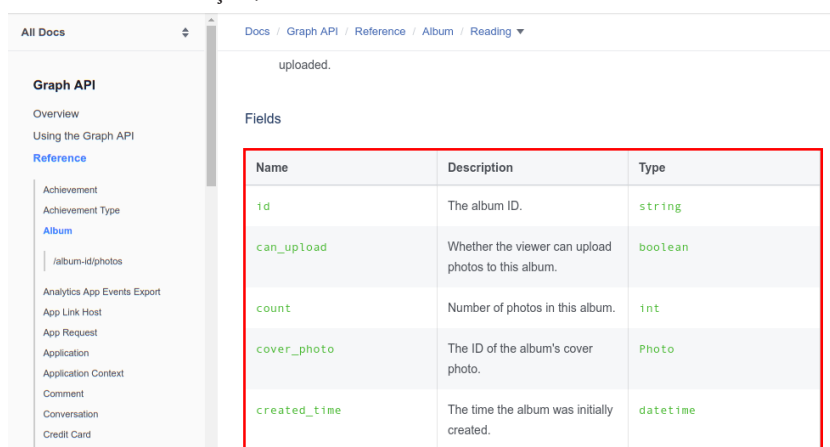
Não foram encontradas na documentação as referências para as visões de nome *Favorite Request*, *Games IAP Product*, *Instagram User*, *Lead Gen Data*, *Lead Gen Qualifiers*, *Live Video Error*, *Mobile Sdk Error Item*, *Open Graph Object*, *Page Label Users*, *Page Saved Filter*, *Rich Media Document*, *Saved Message Response Macro*, *Video Copyright Condition Group* e *Payment Address* – todas são citadas em visões da *Facebook Graph API*.

É importante ressaltar que do total de 111 visões que contém elementos textuais voltados a descrição. Algumas destas visões têm descrições que não possuem carga informacional suficiente para servir como elemento que sustente a plena compreensão do conteúdo a ser coletado. Por exemplo, as descrições das visões *App Link Hosts* e *App Event Types for an App* que contém como descrição uma cópia do título da própria visão.

Na seção Leitura, contida dentro da documentação técnica das visões, são encontradas as informações sobre os atributos e tipos de dados disponíveis no momento da coleta de dados.

Estas informações são disponibilizadas no formato HTML, no próprio corpo do documento, e possuem três formas de apresentação: duas em quadro informativo, no formato de tabela HTML; e uma como elemento textual acrescido de quadro informativo, no formato de tabela HTML.

Figura 6 – *Facebook Graph API*: Atributos na primeira forma de apresentação, tabela HTML com três colunas



Name	Description	Type
id	The album ID.	string
can_upload	Whether the viewer can upload photos to this album.	boolean
count	Number of photos in this album.	int
cover_photo	The ID of the album's cover photo.	Photo
created_time	The time the album was initially created.	datetime

Fonte: Recorte de *Meta Platforms, Inc.* (2023d), elaborado pelo Autor.

Na primeira forma de apresentação (Figura 6), o quadro informativo contendo a descrição dos atributos disponíveis na coleta de dados possui três colunas (ênfatisadas por um retângulo com a borda na cor vermelha): a) *name*, contendo o nome do atributo; b) *description*, com a descrição das informações contidas no atributo, e; c) *type*, contendo o tipo de dado aceito para o atributo.

Figura 7 – Facebook Graph API: Atributos na segunda forma de apresentação, tabela HTML com duas colunas

Field	Description
<code>backdated_time</code> datetime	The time when the video post was created.
<code>backdated_time_granularity</code> enum	Accuracy of the backdated time.
<code>created_time</code> datetime	The time the video was initially published.
<code>description</code> string	The description of the video. <small>Default</small>
<code>embed_html</code> string	The HTML element that may be embedded in a Web page to play the video.

Fonte: Recorte de Meta Platforms, Inc. (2023d), elaborado pelo Autor.

A segunda forma de apresentação, em quadro informativo na forma de tabela HTML com duas colunas (Figura 7), exibe as mesmas informações da primeira forma, porém com o uso de duas colunas (ênfatisadas por um retângulo com a borda na cor vermelha).

A primeira coluna, denominada *field*, tem seu conteúdo dividido em duas linhas: a linha superior contém o nome do atributo, e a linha inferior o tipo de dado aceito para ser valor do atributo. A segunda coluna contém a descrição das informações contidas nos valores.

Figura 8 – *Facebook Graph API*: Atributos na terceira forma de apresentação, com elemento textual e com tabela HTML com duas colunas

Docs / Graph API / Reference / User / Family / Reading ▾

Reading from this edge will return a JSON formatted result:

```
{  
  "data": [],  
  "paging": {}  
}
```

data

A list of User nodes.
The following fields will be added to each node that is returned:

Field	Description
relationship	The relationship between user and family member
string	string <small>(Default)</small>

paging

For more details about pagination, see the [Graph API guide](#).

Fonte: Recorte de *Meta Platforms, Inc.* (2023d), elaborado pelo Autor.

A terceira forma de apresentação, com a associação de elemento textual e com quadro informativo em formato de tabela HTML com duas colunas (Figura 8), aparece somente para as visões que possuem como objetivo relacionar duas visões, ou seja, visões elaboradas para relacionar um ou mais registros da visão de origem com um ou mais registros da visão de destino. Por exemplo, a visão *User Family* tem o objetivo de relacionar registros da visão *User* com um ou mais registros da visão *User* para composição dos membros familiares.

Os atributos disponíveis para a coleta são apresentados em formato de notação da linguagem de marcação JSON (ênfatisados com um retângulo com a borda na cor amarela), contendo os nomes dos atributos disponíveis. O tipo de dado esperado para os atributos é descrito nos elementos textuais da seção (seta na cor roxa).

Caso o relacionamento contenha outros atributos específicos do contexto de relacionamento entre as visões de origem e destino, esses são apresentados em um quadro informativo, na forma de uma tabela em HTML

(ênfâtizados por um retângulo com a borda na cor vermelha), contendo duas colunas: *field*, com seu conteúdo dividido em duas linhas, sendo a superior o nome do atributo e a inferior o tipo de dado aceito para o atributo, e; *description* com a descrição das informações contidos no atributo.

A referência dos elementos descritivos para os atributos também pode apresentar alguns qualificadores – marcações especiais que agregam características específicas aos atributos.

Os qualificadores identificados podem ser do tipo:

- a) *Core*: o atributo marcado com este qualificador não sofrerá nenhuma alteração e não terá seu acesso removido no período de 24 meses, contados a partir do lançamento da versão da API;
- b) *Default*: o atributo marcado com este qualificador ficará disponível automaticamente na resposta da requisição. Ou seja, não haverá necessidade de explicitar o nome do atributo no valor do parâmetro *fields* para que esteja disponível no momento da requisição de coleta de dados;
- c) *Deprecated*: o atributo marcado com este qualificador foi considerado obsoleto pela equipe de desenvolvimento da API e, caso disponível na coleta de dados, o seu valor será nulo na resposta da requisição.

Como já mencionado nesta seção, os valores dos atributos podem ser simples – quando o valor é um número, signos, data e outros formatos – ou composto. No caso os atributos compostos, podem variar entre:

- a) Vetores: quando o valor do atributo é um conjunto de valores, representado com o uso de vetores computacionais (*arrays*). Os atributos com este tipo de valor apresentam no seu tipo de dado o sufixo [];

- b) Listas: quando o valor do atributo é um conjunto de valores com tipo de dados pré-fixados, com o uso de listas computacionais (*list*). Os atributos com este tipo de valor apresentam no seu tipo de dado o prefixo *list*;
- c) Visões: quando o valor do atributo é um registro de outra visão. Os atributos com este tipo de valor apresentam como tipo de dado o nome de uma Visão.

Foram identificados 1231 atributos, sendo 792 atributos com valor do tipo simples, 436 atributos do tipo composto e 3 atributos que não foi possível realizar esse tipo de identificação.

Com relação aos qualificadores, 417 atributos são retornados automaticamente pelas requisições no momento da coleta de dados, 430 atributos precisam ser explicitados na requisição para a coleta de dados (através do uso do parâmetro *fields*) e 384 atributos não foi possível realizar esse tipo de identificação.

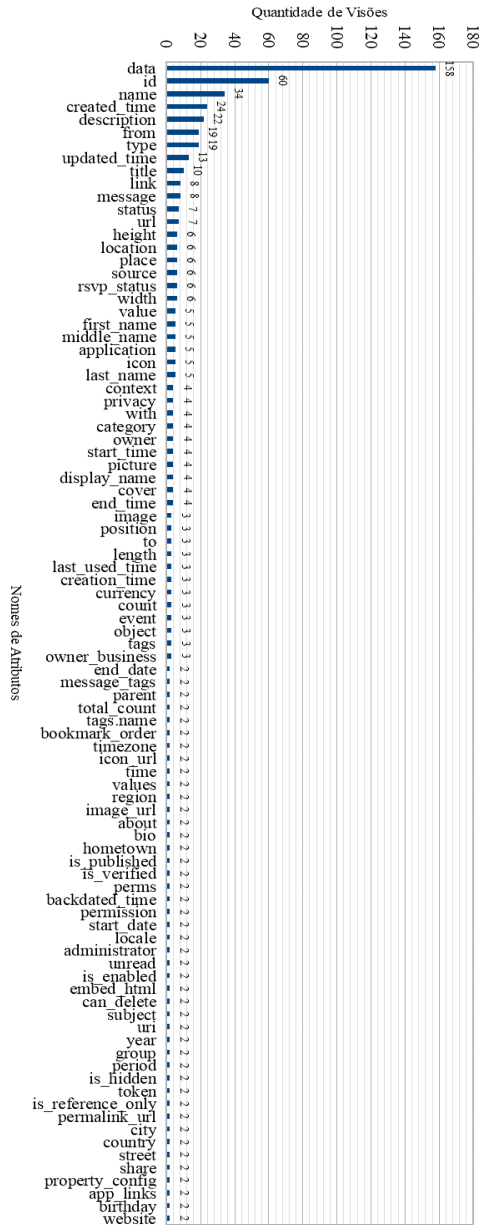
Foram identificados sete atributos obsoletos e 840 atributos disponíveis para a coleta de dados. Entretanto, 384 atributos não foi possível realizar esse tipo de identificação.

Um total de 31 atributos são parte integrante do *Core* e em 1200 atributos não foi possível realizar a identificação desse tipo de informação.

Dos atributos que não foram possíveis identificar os qualificadores, 156 possuem como nome o texto *data* e são parte de visões elaboradas para vincular o relacionamento entre visões.

Foram encontrados 167 nomes distintos para atributos da *Facebook Graph* API, porém um mesmo nome de atributo pode aparecer em visões distintas.

Gráfico 1 – Facebook Graph API: Ocorrências de nomes de atributos iguais em diferentes visões



Fonte: Elaborado pelo Autor.

O Gráfico 1 exibe as ocorrências de nomes de atributos que foram utilizados em duas ou mais visões. Destacam-se que nove nomes de atributos são utilizados 10 em ou mais visões (*data*, *id*, *name*, *created_time*, *description*, *from*, *type*, *updated_time* e *title*) e 16 nomes são utilizados entre cinco e nove visões (*link*, *message*, *status*, *url*, *height*, *location*, *place*, *source*, *rspv_status*, *width*, *value*, *first_name*, *middle_name*, *application*, *icon* e *last_name*). Um total de 66 nomes de atributos são utilizados entre duas e quatro visões e 612 nomes de atributos são utilizados em apenas uma visão da API.

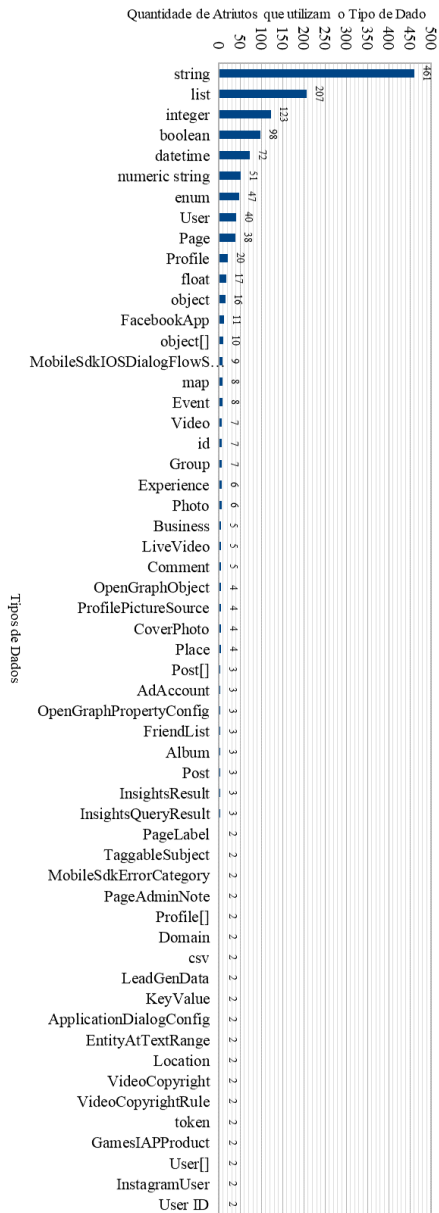
A *Facebook Graph* API contém 159 tipos de dados, que podem ser classificados segundo suas características, em três categorias:

- a) Tipos de dados de Sistema de Gerenciamento de Bancos de Dados: são encontrados na maioria dos Sistemas de Gerenciamento de Bancos de Dados, como números inteiros, *string* para caracteres, datas, textos longos, booleanos, entre outros;
- b) Tipos de dados de algoritmos de Linguagens de Programação: são tipos de dados encontrados em linguagens de programação (como o Java e o C) e também próprios do contexto da API;
- c) Tipos de dados para relacionar conteúdos de duas visões: são tipos de dados para atributos que apresentam como valor um subconjunto de atributos originários de uma outra visão no momento da coleta.

Um mesmo tipo de dado pode ter até duas características. Por exemplo, o tipo de dado *list<Page>* é uma lista (tipo de dado específico) que apresenta subconjuntos da visão *Page* (tipo de dado que apresenta conteúdo de uma visão) – mesclando duas categorias.

Os tipos de dados precisam ser concomitantes com a estrutura identificadas nos atributos e, portanto, seus valores podem ser simples – quando o valor é um número, símbolos, textos ou datas – ou compostos – quando o valor é composto por um subconjunto de atributos.

Gráfico 2 – Facebook Graph API: Ocorrências de tipos de dados em atributos



Fonte: Elaborado pelo Autor.

O Gráfico 2 exibe as ocorrências de tipos de dados que foram utilizados em dois ou mais atributos. Com relação a ocorrências destes tipos de dados nos atributos, foram identificados:

- a) 14 tipos de dados utilizados em 10 ou mais atributos, sendo sete simples e sete compostos: *string* (simples), *list* (composto), *integer* (simples), *boolean* (simples), *datetime* (simples), *numeric string* (simples), *enum* (simples), *User* (composto), *Page* (composto), *Profile* (composto), *float* (simples), *object* (composto), *FacebookApp* (composto) e *object[]* (composto);
- b) 11 de tipos de dados utilizados entre cinco a nove atributos, sendo dois simples e nove compostos: *MobileSdkIOSDialogFlowSettings* (composto), *map* (simples), *Event* (composto), *Video* (composto), *id* (simples), *Group* (composto), *Experience* (composto), *Photo* (composto), *Business* (composto), *LiveVideo* (composto) e *Comment* (composto).

Apesar da existência de um tipo de dado *id*, não há uma relação direta entre atributos de mesmo nome. Ou seja, nem todo atributo *id* possui o tipo de dado *id*.

Também foram identificados 31 tipos de dados utilizados de dois a quatro atributos, além de 103 tipos de dados utilizados apenas uma vez. Não foi possível identificar os tipos de dados dos atributos *from* (da visão *Live Video*), *open_graph_story* e *reviewer* (da visão *Open Graph Rating*).

Os relacionamentos entre visões seguem princípios similares ao de cardinalidade entre tabelas nos Sistemas de Gerenciamento de Bancos de Dados (Silberschatz; Korth; Sudarshan, 2020). Foram identificados o uso das cardinalidades: 1-para-1, quando um registro da visão de origem relaciona-se com um registro da visão de destino; 1-para-N, quando um registro da visão de origem relaciona-se com uma ou mais registros da visão

de destino, e; N-para-N, quando um ou mais registros da visão de origem relacionam-se com um ou mais registros da visão de destino.

Os relacionamentos podem ser explicitados na documentação técnica das visões de duas formas diferentes:

- a) A partir dos atributos: quando o tipo de dado de um atributo é uma visão, considera-se que o valor do atributo representa um relacionamento com outra visão, proporcionando que, ao coletar conjuntos de dados da visão de origem, os dados da visão de destino também podem ser coletados através deste atributo;
- b) A partir das arestas: são relacionamentos a partir de arestas disponíveis na visão de origem que interligam os seus registros com registros de outras visões, porém com a obrigatoriedade da execução de uma nova requisição na coleta de dados. Ou seja, ao coletar os conjuntos de dados de uma determinada visão, é possível realizar outras requisições para coletar conjuntos de dados das visões relacionadas por arestas.

Foram identificados 484 relacionamentos, sendo: 300 com o uso de atributos e 184 com o uso de arestas. Nos relacionamentos com o uso de atributos, 262 possuem a cardinalidade 1-para-N e 38 a cardinalidade N-para-N. Já nos relacionamentos por aresta, 183 possuem a cardinalidade 1-para-N e um possui a cardinalidade 1-para-1. Os relacionamentos por cardinalidade 1-para-1 só foram identificados com o uso de arestas e os relacionamentos por cardinalidade N-para-N somente com o uso de atributos.

As requisições disponíveis para o acesso às visões funcionam a partir do uso do protocolo HTTPS, método GET, tanto para a transmissão das informações das requisições, como para o recebimento dos conjuntos de dados coletados. Como as requisições estão diretamente vinculadas com as visões, a descrição das funcionalidades e a lista com parâmetros de consulta estão disponíveis na documentação técnica, nas seções:

- a) **Leitura:** na subseção Parâmetros, contendo informações sobre o método de consulta (denominado GET, homônimo ao método) aos conjuntos de dados disponíveis na visão, parte integrante do escopo analisado;
- b) **Criação:** contendo informações sobre o método de inserção de novos conjuntos de dados (denominado INSERT), fora do escopo de análise;
- c) **Atualização:** contendo informações sobre o método de atualização de conjuntos de dado (denominado UPDATE), fora do escopo de análise;
- d) **Exclusão:** contendo informações sobre o método de exclusão de conjuntos de dados (denominado DELETE), fora do escopo de análise.

Das 279 visões disponíveis para a coleta por API, um total de 19 visões não possuem requisições para coleta dos conjuntos de dados: *FavoriteRequest*, *GamesIAPProduct*, *InstagramUser*, *LeadGenData*, *LeadGenQualifiers*, *LiveVideoError*, *MobileSdkErrorItem*, *OpenGraphObject*, *PageCategory*, *PageSavedFilter*, *PaymentAddress*, *Profile*, *RichMediaDocument* e *SavedMessageResponseMacro*.

Nas 260 requisições disponíveis, foram identificados um total de 398 parâmetros para a coleta, com as seguintes características:

- a) 65 parâmetros não possuem descrição;
- b) 64 parâmetros não apresentam o tipo de dado esperado na entrada de dados para a coleta. São parte integrante dos parâmetros que não possuem descrição.

Foram encontrados 164 nomes de parâmetros distintos, sendo que um total de 122 são utilizados somente em um método. No caso de nomes de parâmetros utilizados em mais de um método:

- a) Seis nomes são utilizados em 10 ou mais métodos: *user-id*, *application-id*, *event-id*, *page-id*, *type* e *video-id*;
- b) Nove nomes são utilizados de cinco a nove métodos: *life-event-id*, *life-event-id*, *open-graph-context-id*, *period*, *photo-id*, *since*, *target_id*, *until* e *user*;
- c) Um total de 27 nomes são utilizados de dois a quatro métodos: *aggregateBy*, *album-id*, *breakdown*, *broadcast_status*, *business_id*, *conversation-id*, *ecosystem*, *event_name*, *fieldname_of_type_PlatformImageSource*, *filter*, *friend-list-id*, *group-id*, *height*, *is_published*, *live-video-id*, *metric*, *object-id*, *order*, *page-label-id*, *permission*, *post-id*, *redirect*, *saved-message-response-id*, *status*, *uid*, *video-list-id* e *width*.

Os parâmetros possuem tipos de dados, assim como os atributos. Apresentam as seguintes características:

- a) Sete tipos de dados são utilizados em 10 ou mais parâmetros: *numeric string*, *string*, *enum*, *integer*, *boolean*, *datetime* e *list*;
- b) Um de tipo de dado é utilizado de cinco a nove parâmetros: *Set of insights API metrics* – sendo uma lista parâmetros com métricas para a API;
- c) Dois tipos de dados são utilizados de dois a nove parâmetros: *id* (texto contendo o valor de *id* de qualquer visão) e *Open Graph Object type*;
- d) Quatro tipos de dados são utilizados apenas em um parâmetro: *Base64 UTF-8 encoded string*, *Permission*, *invite token* e URL – todos relacionados ao *Open Graph Object*.

Para acessar as requisições é necessário o uso de um conjunto de permissões. No caso da *Facebook Graph* API, as permissões para a coleta de dados estão vinculadas ao uso do conjunto de duas estruturas de permissão de acesso: os *tokens* de autorização de acesso e as permissões.

Os *tokens* de autorização de acesso são um conjunto finito de símbolos, gerados pelo serviço, com a finalidade de providenciar uma contrassenha para verificação de credenciais. Estes *tokens* de autorização de acesso são um dos itens que compõem o processo de verificação inicial de credenciais – ou seja, o processo do sistema de entrada que permite que o acesso seja concedido e que habilite aplicações para a coleta de conjuntos de dados. São obtidos pelo uso do protocolo HTTPS, sendo possível sua implementação em diversas linguagens de programação. Após a sua criação, os Agentes Externos têm acesso a informações complementares, como o tempo de validade do *token* e o aplicativo que solicitou a sua geração (Meta Platforms, INC., 2023e).

Já as permissões são autorizações de acesso a um aplicativo externo para coleta de conjuntos de dados de Referenciados, de páginas ou de grupos – e as visões relacionadas a todos estes entes. As permissões são concedidas no momento que um Referenciado conecta um aplicativo externo ou um código-fonte de *website* ao seu perfil, ou quando um Referenciado conecta um aplicativo externo ou um código-fonte de *website* as páginas do *Facebook* que administra (Meta Platforms, INC., 2023f).

As informações sobre quais são os *tokens* de autorização de acesso existentes e quais são as permissões necessárias para acessar conjuntos de dados são encontradas em de três documentos: *Tokens* de Acesso (no idioma inglês, *Access Tokens*), o conjunto de documentos Permissões com o *Login* do *Facebook* (no idioma inglês, *Permissions with Facebook Login*) e na documentação técnica das visões.

As informações sobre os *tokens* de autorização de acesso existentes encontram-se em um documento denominado *Tokens* de Acesso, que integra o conjunto de documentos que descrevem como as aplicações e os

códigos-fonte de *websites* podem interoperar dados com os conteúdos disponíveis no serviço, denominado *Facebook Login*. Existem quatro *tokens* de autorização de acesso (Meta Platforms, INC., 2023e):

- a) *User Access Token*: é o *token* de autorização de acesso mais utilizado e que permite acesso pela *Facebook Graph* API aos conjuntos de dados de Referenciados. Autoriza a coleta, a criação, a alteração e a exclusão de conteúdo. Em algumas operações, os Referenciados precisam autorizar o Agente Externo para realizar as operações de criação, alteração e exclusão de dados;
- b) *App Access Token*: utilizado para que Agentes Externos possam realizar requisições específicas ao funcionamento dos aplicativos, como a coleta de dados estatísticos do *Facebook* sobre a aplicação; a realização de testes de interação entre o serviço e o aplicativo, e; a modificação de configurações do aplicativo;
- c) *Page Access Token*: tem funcionamento similar ao *User Access Token*, porém diferencia-se por ser voltado à autorização do acesso aos conteúdos de páginas do *Facebook* que o Referenciado administra. Por exemplo, se um Referenciado é administrador de uma página do *Facebook* sobre culinária – e autorizar o Agente Externo por este *token* de autorização de acesso – o Agente Externo estará apto a realizar a leitura, e a criação, a alteração e a exclusão de conteúdo desta página;
- d) *Client Token*: são *tokens* de autorização de acesso inseridos nos códigos-fonte de aplicativos para dispositivos móveis e aplicativos para sistemas operacionais de computadores, voltados a conexão com o serviço. Não são utilizados para a coleta de dados de Referenciados, pois esta autorização concede o acesso a um conjunto de dados limitado⁸ pelo *Facebook*.

⁸ Não há informações nos documentos de referência do *Facebook* sobre quais são os conjuntos de dados disponíveis por este *token* de autorização de acesso (Meta Platforms, INC., 2023e).

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Entretanto, o Referenciado pode controlar a permissão de acesso a dados de algumas visões de forma individual, no momento da conexão para o início do processo de coleta de dados pelos Agentes Externos. Portanto, as permissões são os elementos preestabelecidos pelo *Facebook* que delimitam, nestas visões, quais serão os conjuntos de dados de Referenciados, de páginas e de grupos que estarão acessíveis no momento da coleta por Agentes Externos.

O documento de Referências de Permissões – *Graph* API (Meta Platforms, INC., 2023g) contém as descrições de funcionamento das 40 permissões existentes (Quadro 1).

Quadro 1 – *Facebook Graph* API: Referências de permissões disponíveis, versão 2.6

Nome da Permissão	Descrição da Permissão	Visões disponíveis	Atributos disponíveis
<i>email</i>	Acesso ao e-mail do Referenciado.	<i>User</i>	<i>e-mail</i>
<i>manage_pages</i>	Habilita o acesso aos dados de páginas e aplicativos que o Referenciado administra.	<i>Facebook App</i>	Todos da visão
		<i>Page</i>	Todos da visão
<i>pages_manage_cta</i>	Habilita a realização de <i>call-to-actions</i> nas páginas que o Referenciado administra.	<i>Page</i>	Todos da visão
<i>pages_manage_instant_articles</i>	Habilita o gerenciamento a artigos nas páginas que o Referenciado administra.	<i>Page</i>	Todos da visão
		<i>Page Instant Article</i>	Todos da visão
<i>pages_messaging</i>	Habilita ao aplicativo ler e enviar mensagens privadas da página que o Referenciado administra.	<i>Page</i>	Todos da visão
		<i>/{page-id}/conversations</i>	Todos da visão
<i>pages_messaging_phone_number</i>	Habilita ao aplicativo ler e enviar mensagens privadas (via <i>Short Message Service</i>) da página que o Referenciado administra.	<i>Page</i>	Todos da visão
		<i>/{page-id}/conversations</i>	Todos da visão
<i>pages_show_list</i>	Acesso a lista de páginas que um Referenciado administra.	<i>User</i>	Todos da visão
		<i>Page</i>	Todos da visão
<i>public_profile</i>	Acesso ao conjunto de dados pessoais públicos.	<i>User</i>	<i>age_range, first_name, gender, id, last_name, link, locale, name e timezone</i>

Nome da Permissão	Descrição da Permissão	Visões disponíveis	Atributos disponíveis
<i>publish_actions</i>	Habilita a publicação de postagens, ações <i>Open Graph</i> , conquistas e resultados por um aplicativo em nome de um Referenciado.	<i>Achievement</i>	Todos da visão
		<i>Open Graph Action</i>	Todos da visão
		<i>Post</i>	Todos da visão
		<i>User Feed</i>	Todos da visão
		<i>User</i>	Todos da visão
<i>publish_pages</i>	Habilita a publicação de postagens, comentários e curtidas em páginas e aplicativos que o Referenciado administra.	<i>Facebook App</i>	Todos da visão
		<i>Page</i>	Todos da visão
<i>read_audience_network_insights</i>	Acesso aos dados sobre a audiência de um aplicativo para a personalização de público-alvo.	<i>Facebook App</i>	Todos da visão
<i>read_custom_friendlists</i>	Acesso aos nomes das listas de amigos que o Referenciado elaborou para organizar seus contatos.	<i>User</i>	Todos da visão
		<i>User Friendlists</i>	Todos da visão
<i>read_insights</i>	Acesso aos aplicativos, páginas e websites que o Referenciado administra.	<i>Facebook App</i>	Todos da visão
		<i>Page</i>	Todos da visão
		<i>User</i>	Todos da visão
<i>read_page_mailboxes</i>	Acesso a caixa de entrada de conversas de uma página que um Referenciado administra.	<i>Page</i>	Todos da visão
		<i>Page Conversations</i>	Todos da visão
<i>rspv_event</i>	Habilita uma aplicação para modificar o atendimento a um evento em nome de um Referenciado.	<i>Event</i>	Todos da visão
		<i>User</i>	Todos da visão
		<i>User Events</i>	Todos da visão
<i>user_about_me</i>	Acesso a descrição sobre do Referenciado.	<i>User</i>	<i>bio</i>
<i>user_actions:{app_namespace}</i>	Acesso ao conjunto de ações do Referenciado para um aplicativo.	<i>Facebook App</i>	Todos da visão
		<i>Open Graph Action</i>	Todos da visão
		<i>User</i>	Todos da visão
<i>user_actions.books</i>	Acesso ao conjunto de obras literárias que um Referenciado citou, quer ler, classificou ou leu.	<i>User</i>	Todos da visão
		<i>User Books</i>	Todos da visão
<i>user_actions.fitness</i>	Acesso a ações e postagens de caminhadas, corridas e passeios de bicicleta do Referenciado.	<i>Open Graph Action</i>	Todos da visão
		<i>User</i>	Todos da visão
<i>user_actions.music</i>	Acesso a ações e postagens de músicas e listas de músicas do Referenciado.	<i>Open Graph Action</i>	Todos da visão
		<i>User Music</i>	Todos da visão
		<i>User</i>	Todos da visão

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

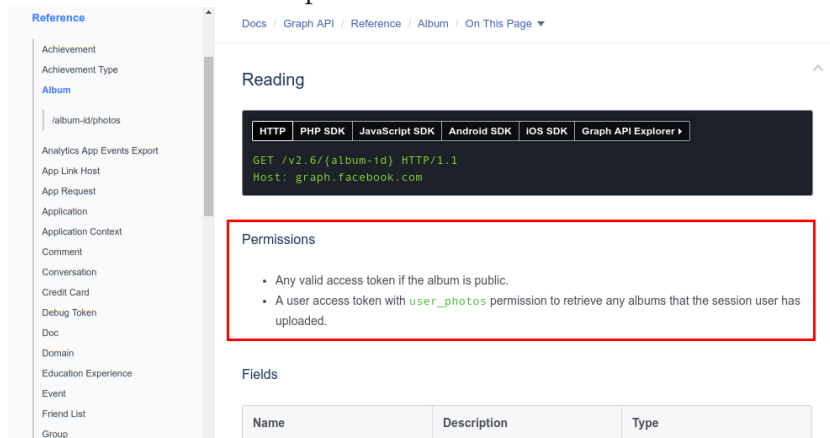
Nome da Permissão	Descrição da Permissão	Visões disponíveis	Atributos disponíveis
<i>user_actions.news</i>	Acesso ao conjunto de notícias que um Referenciado leu ou publicou.	<i>Open Graph Action</i>	Todos da visão
		<i>User</i>	Todos da visão
<i>user_actions.video</i>	Acesso ao conjunto de vídeos que um Referenciado assistiu, classificou, publicou ou quer assistir.	<i>Open Graph Action</i>	Todos da visão
		<i>User Videos</i>	Todos da visão
		<i>User</i>	Todos da visão
<i>user_birthday</i>	Acesso a data de nascimento do Referenciado.	<i>User</i>	<i>birthday</i>
<i>user_education_history</i>	Acesso as informações de escolaridade do Referenciado.	<i>EducationExperience</i>	Todos da visão
		<i>User</i>	Todos da visão
<i>user_events</i>	Acesso aos conjuntos de eventos que a Referenciado criou ou que participa.	<i>Event</i>	Todos da visão
		<i>User</i>	Todos da visão
		<i>User Events</i>	Todos da visão
<i>user_friends</i>	Acesso aos amigos relacionados a determinado Referenciado.	<i>User</i>	Todos da visão
		<i>User Friends</i>	Todos da visão
<i>user_games_activity</i>	Acesso a postagens e conquistas do Referenciado em jogos.	<i>Achievement</i>	Todos da visão
		<i>User</i>	Todos da visão
		<i>User Achievements</i>	Todos da visão
<i>user_hometown</i>	Acesso a cidade natal do Referenciado.	<i>User</i>	<i>hometown</i>
<i>user_likes</i>	Acesso aos objetos que o Referenciado curtiu.	<i>Page</i>	Todos da visão
		<i>User</i>	Todos da visão
		<i>User Likes</i>	Todos da visão
<i>user_location</i>	Acesso a localização do Referenciado.	<i>User</i>	<i>location</i>
<i>user_managed_groups</i>	Acesso aos grupos que o Referenciado administra.	<i>Group</i>	Todos da visão
		<i>User</i>	Todos da visão
		<i>User Groups</i>	Todos da visão
<i>user_photos</i>	Acesso as fotografias que o Referenciado enviou ao serviço ou que foi marcado.	<i>Photo</i>	Todos da visão
		<i>User</i>	Todos da visão
		<i>User Photos</i>	Todos da visão
<i>user_posts</i>	Acesso as postagens do Referenciado, as postagens que foram enviadas em sua linha do tempo ou em que foi marcado.	<i>Post</i>	Todos da visão
		<i>User</i>	Todos da visão
		<i>User Feed</i>	Todos da visão
<i>user_relationship_details</i>	Acesso ao interesse de relacionamento do Referenciado.	<i>User</i>	<i>interested_in</i>

Nome da Permissão	Descrição da Permissão	Visões disponíveis	Atributos disponíveis
<i>user_relationships</i>	Acesso ao estado de relacionamento e aos familiares do Referenciado.	<i>User</i>	<i>relationship_status</i>
	Acesso ao estado de relacionamento e aos familiares do Referenciado.	<i>User Family</i>	Todos da visão
<i>user_religion_politics</i>	Acesso a orientação religiosa e política do Referenciado.	<i>User</i>	<i>religion e political</i>
<i>user_tagged_places</i>	Acesso as localidades de fotografias, vídeos, postagens de estado e <i>hyperlinks</i> que o Referenciado foi marcado.	<i>Location</i>	Todos da visão
		<i>User</i>	Todos da visão
		<i>User Photos</i>	Todos da visão
		<i>User Likes</i>	Todos da visão
		<i>User Feed</i>	Todos da visão
		<i>User Videos</i>	Todos da visão
<i>user_videos</i>	Acesso aos vídeos que o Referenciado enviou ao serviço ou que foi marcado.	<i>Video</i>	Todos da visão
		<i>User</i>	Todos da visão
		<i>User Videos</i>	Todos da visão
<i>user_website</i>	Acesso ao endereço do <i>website</i> do Referenciado.	<i>User</i>	<i>website</i>
<i>user_work_history</i>	Acesso ao histórico de empregos do Referenciado e a lista de empregados destes empregos.	<i>User</i>	<i>work</i>
		<i>Work Experience</i>	Todos da visão

Fonte: Elaborado pelo Autor.

O conjunto de documentos com as descrições de funcionalidades das visões *Facebook Graph API* podem apresentar uma subseção denominada Permissões (Figura 9, destaque com o retângulo de bordas vermelhas), não obrigatória, parte integrante da seção Leitura, onde são explicitados os *tokens* de autorização de acesso e as permissões para permitir a coleta de dados por Agentes Externos. Estas informações são complementares às descrições de cada permissão, encontradas no documento Referências de Permissões – *Login do Facebook*.

Figura 9 – *Facebook Graph API*: subseção contendo informações sobre as permissões da visão



Fonte: Recorte de *Meta Platforms, Inc.* (2023d), elaborado pelo Autor.

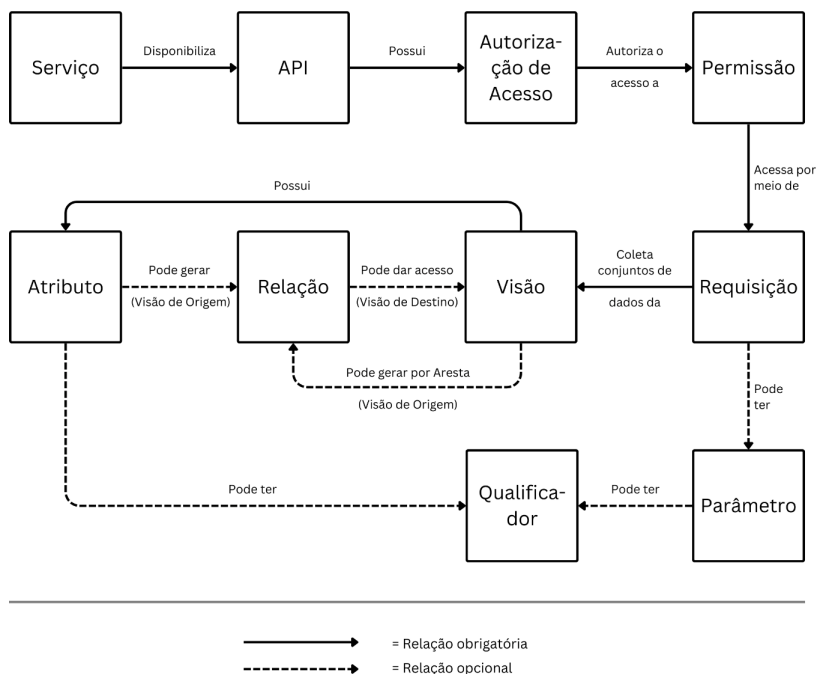
No exemplo da Figura 9, a subseção Permissões da visão *Album* apresenta informações sobre o acesso às operações de coleta de dados (seção Leitura) permitidas, com o uso de: a) qualquer *token* de autorização de acesso válido, no caso de o álbum ser público, ou; b) o uso de um *token* de autorização de acesso do tipo *User Access Token* para álbuns particulares, com prévia autorização do usuário pela habilitação da permissão *user_photos*.

A partir das características da *Facebook Graph API* apresentadas nesta seção, foram identificadas as seguintes entidades para a sistematização da coleta de dados:

- Serviço, com informações referentes ao Serviço de Rede Social *Online*: nome do serviço e a URL de acesso;
- API, com informações referentes a API: nome da API, versão, descrição e URL para acesso as requisições;
- Visão, contendo informações sobre as visões disponíveis: nome da visão, descrição e URL para acesso aos documentos;

- d) Atributos de cada visão, com informações dos atributos disponíveis nos resultados das requisições: nome do atributo, descrição, tipo de dado e qualificadores;
- e) Qualificadores, com informações sobre as características das marcações que diferenciam parte dos atributos das visões, e parte dos parâmetros da requisição: nome do qualificador e descrição;
- f) Requisições existentes em cada visão, contendo informações sobre quais operações de coleta de dados estão disponíveis para a visão: nome da requisição, tipo e nome do método e seus parâmetros de entrada e saída;
- g) Parâmetros existentes nas requisições, em um relacionamento que uma requisição pode ter nenhum, um ou mais parâmetros: nome do parâmetro, descrição e qualificadores;
- h) Relações, com informações sobre os relacionamentos entre as visões: nome da visão de origem, nome da visão de destino, tipo de relacionamento, nome do relacionamento, cardinalidade na origem e cardinalidade no destino;
- i) Autorizações de acesso, com informações sobre os tipos de acesso: nome, descrição e URL para acesso aos documentos;
- j) Permissões, definindo as informações sobre os tipos de permissões existentes e suas relações com cada visão: nome, descrição, visões e atributos relacionados, e URL para acesso aos documentos.

Figura 10 – Facebook Graph API: Vínculos entre as entidades



Fonte: Elaborado pelo Autor.

A Figura 10 apresenta uma representação gráfica das relações entre as entidades propostas. O serviço apresenta formas de acesso aos seus conjuntos de dados por meio da API, que possui um sistema de autorização de acesso baseado em *tokens* (com a função de autorizar o acesso) e permissões (que estabelecem quais conjuntos de dados podem ser coletados).

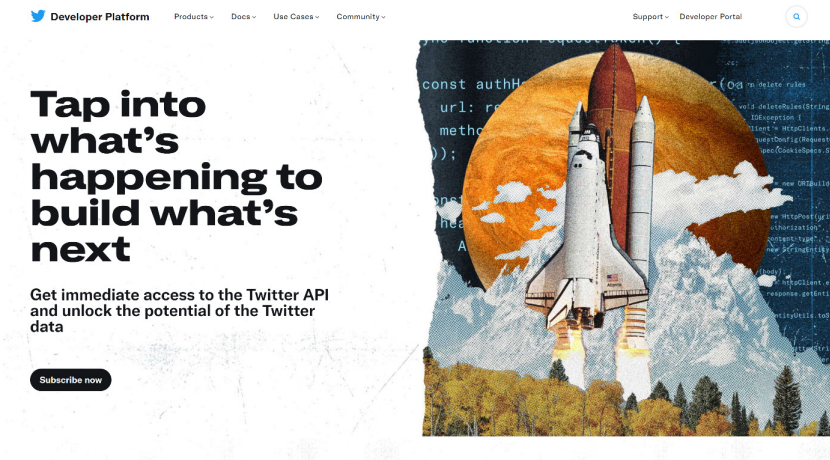
Por meio de pontos de entrada disponíveis, o sistema de autorização permite que seja realizada a execução de requisições para coletar dados, permitindo o acesso aos dados contidos nas visões. No momento do disparo da execução das requisições, podem existir parâmetros de entrada de dados, e estes parâmetros podem ter qualificadores – marcações que estabelecem características especiais a certos parâmetros – como o estado de obrigatoriedade de uso do parâmetro na requisição.

As visões possuem atributos, com tipologia de dados própria, e que também podem apresentar qualificadores (marcações que estabelecem características especiais a certos parâmetros); e relações com outras visões da API – que são estabelecidas a partir de arestas (com o uso de identificadores da visão de origem e da visão de destino) ou pelos valores compostos de atributos, que apresentam como tipo de dado a visão de destino.

1.4 A X REST API

Para acessar os documentos técnicos da X REST API, a X Corp. possui uma área denominada X Developer Platform em seu website (Figura 11). Esta área é exclusiva para a concentração de documentos, de referências, de exemplos e de demais informações referentes ao processo de utilização de serviços oferecidos para Agentes Externos e parceiros.

Figura 11 – X Corp.: Página principal da área reservada aos desenvolvedores



Fonte: Recorte de X Corp. (2023b), elaborado pelo Autor.

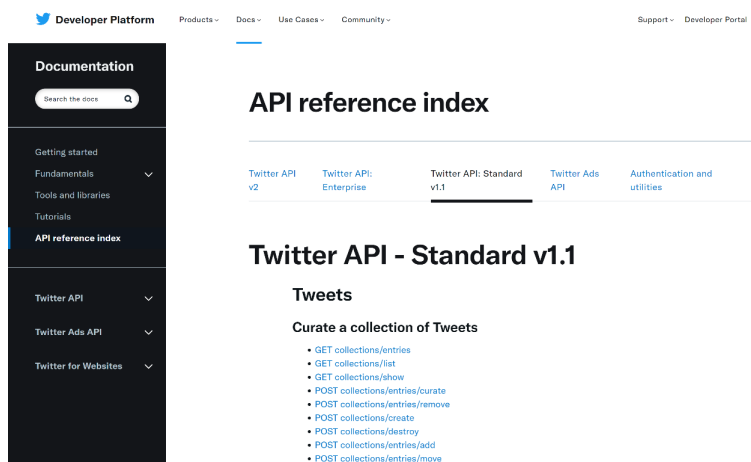
A seção Documentação (*Docs*) contém as coleções de documentos técnicos referentes aos produtos e serviços oferecidos pela X Corp. aos Agentes

Externos, especificamente à comunidade de desenvolvedores de aplicações externas. É formado pela documentação de recursos disponíveis, pelas APIs disponíveis e por referências sobre o uso de dados do *X* em *websites*.

A *X REST API* é a principal API do *X*, sendo desenvolvida com o objetivo de realizar a interoperabilidade de conjuntos de dados de Referenciados com Agentes Externos. As requisições e as respostas das requisições estão acessíveis através do protocolo HTTPS, tanto para a coleta de dados quanto para realização de operações orientadas a inserção, a alteração ou a exclusão de dados, em associação ao sistema de entrada *OAuth* (baseado no padrão de código aberto de mesmo nome) (X CORP., 2023c, 2023d). Os documentos estão disponíveis nos idiomas inglês, português e japonês. O idioma inglês é considerado o principal e, portanto, recebe atualizações antes dos demais (X CORP., 2023d).

O conjunto de documentos contendo as descrições de sua funcionalidade estão divididos em oito seções (Figura 12, coluna da esquerda com fundo preto): Começando, Fundamentos, Ferramentas e bibliotecas, Tutoriais, Índice de Referência da API, *X API*, *X Ads API* e *X para Websites*.

Figura 12 – *X Corp.*: Seções disponíveis dos documentos da *X REST API*



Fonte: Recorte de *X Corp.* (2023e), elaborado pelo Autor.

As informações relacionadas com os conjuntos de dados disponíveis, atributos, relacionamentos entre atributos e demais conjuntos de dados, bem como a descrição destes, encontram-se na seção Índice de Referência da API (X CORP., 2023e) – sendo esta seção o escopo deste livro.

No momento da coleta, os conjuntos de dados da X REST API são apresentados na forma de um sociograma. Os conjuntos de dados disponíveis para coleta são pré-determinados e selecionados pela instituição, na forma de visões. Cada visão é tratada como um nó do sociograma.

Para coletar os conjuntos de dados de um Referenciado, o Agente Externo deve realizar uma requisição de acesso ao nó *Users*. Esse nó possui características idênticas com o processo de requisição de uma visão (ver Figura 3, seção 1.1). Portanto, cada nó contém uma coleção de atributos e registros, unicamente identificáveis. Os atributos possuem um número como identificador de cada registro armazenado, independente da visão, denominado de duas formas distintas: i) pela junção do nome da visão com o sufixo *_id* ou ii) pelo termo *id*. Em cada visão, o *id* está associado a um identificador único somente para aquela visão e, portanto, podem existir valores iguais para o atributo *id* de diferentes visões.

Os relacionamentos entre as visões são denominados arestas – que representam as conexões entre os nós. As visões estão relacionadas através de valores em um ou mais atributos na visão de origem e na visão de destino.

Os atributos disponíveis em uma visão também são denominados como campos na documentação. Os campos são idênticos aos atributos, pois contém um nome e um valor, que pode ser simples ou composto. Por exemplo, para um determinado Referenciado, o atributo nome possui o valor Albert Einstein, sendo do tipo simples. Já um atributo que permite a coleta dos seguidores do Referenciado possui como valor uma lista de *ids*, sendo que cada *id* representa um seguidor, sendo do tipo composto.

As requisições são realizadas através do uso do protocolo HTTPS, pela URL <https://api.twitter.com>⁹. Para coletar dados de uma visão é neces-

⁹ Apesar da mudança de nome de *Twitter* para *X*, ainda é utilizado o domínio *twitter.com* para diversos serviços.

sário o uso de processo de transformação dos nomes de visões como parte dos parâmetros da requisição de coleta de dados, similares as diretrizes propostas no módulo *mod_rewrite* do servidor de páginas Apache, em sua segunda versão ou superior (The Apache Software Foundation, 2017).

A requisição é composta pelos seguintes elementos:

- a) URL da API;
- b) Separador, representado pelo símbolo de barra;
- c) Versão da API, em formato numérico representado pela versão de revisão maior e menor, separadas pelo símbolo de ponto;
- d) Separador, representado pelo símbolo de barra;
- e) Nome da visão, similar ao formato proposto no módulo de redirecionamento *mod_rewrite*;
- f) Representação da extensão do formato de arquivo JSON, através do uso dos símbolos *.json*;
- g) Separador, representado pelo símbolo de interrogação;
- h) Lista de parâmetros.

É importante observar que algumas requisições não possuem parâmetros obrigatórios, devido ao contexto da própria requisição ou por ser de uso exclusivo em conjuntos de dados do próprio coletor.

Por exemplo, para acessar os conjuntos de dados do Referenciado de nome Fernando de Assis Rodrigues (visão *Users*), é necessário enviar pelo método GET do protocolo HTTPS: i) a versão da API, ii) o nome da visão de exibição de informações de Referenciados de forma individual *Users Show* (no caso, *users*), no formato do módulo *mod_rewrite* (no caso, *show*), e; a chave de identificação do Referenciado na visão *Users* pelo parâmetro *user_id* (Exemplo 10).

Exemplo 10 – X REST API: Requisição para coleta de conjuntos de dados, via método GET

```
GET api.twitter.com/1.1/users/show.json?user_id=44249787
```

Fonte: Elaborado pelo Autor.

O número 44249787 representa o valor do atributo *id*, enviado na requisição pelo parâmetro de entrada *user_id*, sendo que este atributo é o identificador que representa unicamente este Referenciado na visão *Users*.

Os resultados das requisições da *X Graph* API são apresentados somente no formato de notação da linguagem de marcação JSON. No exemplo da requisição solicitada de conjunto de dados do Referenciado com o *id* de número 44249787, a resposta da requisição é apresentada conforme o Exemplo 11.

Exemplo 11 – X REST API: Resultado da requisição para coleta de conjuntos de dados, via método GET

```
{
  "id": 44249787,
  "id_str": "44249787",
  "name": "Fernando de Assis Rodrigues",
  "screen_name": "rodriguesprobr",
  "location": "Marília, SP",
  "profile_location": null,
  "description": "Bio and more: https://amazondatatech.com.br",
  "url": "https://t.co/ci3iDm2F5b"
  ...
}
```

Fonte: Elaborado pelo Autor.

Os caracteres chaves { e } indicam o início e o final do registro. Tanto os nomes dos atributos como os seus respectivos valores são apresentados dentro de duas chaves – uma no início e outra no final de cada resposta de requisição.

Cada atributo é representado da seguinte forma: o nome do atributo entre aspas duplas, seguido do símbolo de dois pontos (separador entre nome do atributo e valor) e do valor do atributo (que pode ter ou não aspas duplas no início e no final de seu valor, dependendo do tipo de dado). Os atributos de uma mesma visão são separados entre si pelo símbolo de vírgula.

No Exemplo 11, a solicitação retornou um registro da visão *Users*, a partir do uso como parâmetro *user_id* para a atributo *id* com o valor 44249787. O resultado apresentado para a coleta de dados contém oito atributos¹⁰, de nomes *id*, *id_str*, *name*, *screen_name*, *location*, *profile_location*, *description* e *url*, com os valores (respectivamente): 44249787, 44249787, Fernando de Assis Rodrigues, rodriguesprobr, Marília SP, Ø (valor nulo), *Bio and more: https://amazondatatech.com.br e https://t.co/ci3iDm2F5b*.

Os atributos supramencionados são retornados automaticamente, pois são parte integrante do conjunto de atributos marcados como Padrão (no idioma inglês, *Default*) para esta visão. Caso a coleta de dados necessite de mais atributos da visão *Users*, é necessário se consultar as permissões de acesso concedidas garantem a coleta destes dados. Caso seja necessária a adição de mais de um parâmetro na requisição, deve-se utilizar um separador entre os parâmetros, representado pelo símbolo &.

A X REST API permite a realização de requisições contendo dados de outras visões, a partir de seus relacionamentos entre visões. Para o acesso a estes dados, deve-se realizar a requisição destas visões (visões de destino), a partir do acesso a uma visão de origem, com o envio do parâmetro *id* contendo o valor deste atributo na visão de origem.

Para a requisição e a coleta de dados de visões de destino, a composição dos elementos de consulta é igual à composição das requisições para visões de origem. Entretanto, o Agente Externo deve se preocupar em compreender quais serão os pontos de entrada necessários para coletar os conjuntos de dados das visões de destino.

¹⁰ O exemplo de resposta da requisição foi encurtado por motivos estéticos ao texto. São apresentados oito atributos de um total de 46 atributos disponíveis para coleta de dados.

Por exemplo, para coletar conjuntos de dados de seguidores do Referenciado, é necessário utilizar o valor do atributo *id* da visão do *Users* (Origem) como parâmetro *user_id* da requisição para a visão *Followers*, conforme o Exemplo 12.

Exemplo 12 – X REST API: Requisição para coleta de conjuntos de dados de visões relacionadas, via método GET

```
GET api.twitter.com/1.1/followers/list.json?user_id=44249787
```

Fonte: Elaborado pelo Autor.

Por exemplo, para a requisição dos conjuntos de dados sobre relacionamentos do Referenciado de nome Fernando de Assis Rodrigues (visão de origem *Users*) com outros Referenciados do *X*, é necessário enviar o *user_id* da visão *Users* como parâmetro de entrada na requisição para visão *Followers*. O número 44249787 representa o valor do atributo *user_id* para o Referenciado em questão, sendo este atributo o identificador da visão de origem *Users*.

A resposta deste tipo de requisição segue os mesmos princípios das requisições para coleta de atributos da visão de origem, ilustrado no Exemplo 13. As chaves { e } indicam o início e o final dos registros recuperados. Neste caso, o Referenciado com o nome Fernando de Assis Rodrigues possui um relacionamento com outros Referenciados, no qual cada relação representa dados da visão do tipo *Users*, porém agora representando cada um de seus seguidores¹¹.

¹¹ Diferentemente do Serviço de Rede Social *Online Facebook*, o *X* utiliza relacionamentos unidirecionais. Um referenciado pode se relacionar com os demais de duas formas: seguindo outros indivíduos (*following*) ou sendo seguido pelos outros indivíduos (*followers*).

Exemplo 13 – X REST API: Resultado da requisição para coleta de conjuntos de dados de visões relacionadas, com parâmetros adicionais, via método GET

```
{
  "users": [
    {
      "id": 186568600,
      "id_str": "186568600",
      "name": "John Connor ",
      "screen_name": "john_connor_hates_skynet",
      "location": "Los Angeles",
      "description": "It's doomsday.",
      "url": "https://en.wikipedia.org/wiki/Skynet_(Terminator)",
      ...
    },
    {
      "id": 266966992,
      "id_str": "266966992",
      "name": "Bilbo Bolseiro",
      "screen_name": "bilbo_lotr",
      "location": "Condado",
      "description": "It's my birthday",
      "url": " https://lotr.fandom.com/wiki/Main_Page",
      ...
    },
    {
      "id": 67391507,
      "id_str": "67391507",
      "name": "Morpheus",
      "screen_name": "morpheus_matrix",
      "location": "Zion",
      "description": "Blue pill or Red pill?",
      "url": null,
      ...
    }
  ],
  "next_cursor": 1516655103312451000,
  "next_cursor_str": "1516655103312451188",
  "previous_cursor": 0,
  "previous_cursor_str": "0"
}
```

Fonte: Elaborado pelo Autor.

Neste caso, a visão de origem contém cinco atributos:

- a) *users*: atributo composto, que apresenta entre os símbolos [e] um ou mais seguidores (visão de destino *Users*), em que cada bloco entre chaves { e } representa o conjunto de dados de cada seguidor. Os conjuntos de dados das visões de destino (destacados na cor azul) são apresentados dentro de duas chaves e a cada linha é separada pelo símbolo de vírgula;
- b) *next_cursor*, *next_cursor_str*, *prev_cursor* e *prev_cursor_str*: contém informações para a iteração (paginação) do aplicativo que coletará estes conjuntos de dados, pois a X REST API não permite a coleta de todos os relacionamentos entre a visão de origem e as visões de destino em uma única requisição. Ou seja, para realização de coleta de dados sobre todos os seguidores relacionados com o Referenciado de nome Fernando de Assis Rodrigues, terão de ser realizadas coletas em blocos, onde cada requisição permitirá a coleta de 20 seguidores cada vez. Os atributos *next_cursor* e *next_cursor_str* apresentam os valores para acessar a próxima lista contendo os próximos 20 seguidores (em formato numérico e textual, respectivamente), e os atributos *prev_cursor* e *prev_cursor_str* apresentam os valores para acessar a lista anterior contendo 20 seguidores, em formato numérico e textual, respectivamente.

Para coletar pela paginação os registros da relação com seguidores é necessário adicionar o parâmetro *cursor* com o valor retornado nos atributos *prev_cursor* ou de *prev_cursor_str* para acessar a lista antecessora a requisição atual, e com o valor de *next_cursor* ou de *next_cursor_str* para acessar a lista sucessora a requisição atual.

Também é possível adicionar na requisição outros parâmetros e formas de paginação de conteúdo, variando a disponibilidade de acordo com cada visão. Em certos cenários de coleta, estão disponíveis os parâmetros:

- a) *include_entities*, para controlar a coleta de conjuntos de dados de visões de destino vinculadas automaticamente com a visão de origem, como, por exemplo, a inclusão de atributos do último tuíte que um Referenciado publicou, ao acessar conjuntos de dados da visão *Users*;
- b) *screen_name*, para a recuperação de conjuntos de dados sobre Referenciados pelo uso do apelido de sua conta no *X*;
- c) *lang*, para que na coleta de dados os valores dos atributos sejam traduzidos automaticamente para o idioma que foi determinado no valor deste parâmetro.

A *X* REST API também possui um sistema de descoberta de conteúdo, denominado *Standard Search* API, no qual é utilizada quando não se sabe, a princípio, o valor do atributo *id* de um registro. Todavia, só é possível realizar consultas aos tuítes¹² (visão *Tweets*).

Para coletar os dados pela descoberta de conteúdo, é necessário realizar a composição dos seguintes elementos (X CORP., 2023f):

- a) URL da API;
- b) Separador, representado pelo símbolo de barra;
- c) Versão da API, em formato numérico representado pela versão de revisão maior e menor, separadas pelo símbolo de ponto;
- d) Separador, representado pelo símbolo de barra;
- e) A palavra reservada *search*;
- f) Separador, representado pelo símbolo de barra;
- g) A palavra reservada *tweets*;
- h) Representação da extensão do formato de arquivo JSON, através do uso dos símbolos *.json*;

¹² Tradicionalmente um tuíte é o nome dado a uma publicação de uma postagem no *X*.

- i) Separador, representado pelo símbolo de interrogação;
- j) O parâmetro q , seguido do símbolo de igualdade e o texto que se deseja pesquisar nos tuítes.

Por exemplo, para pesquisar e coletar dados de tuítes de quaisquer Referenciados, marcados como públicos, que mencionaram o perfil no X da Organização das Nações Unidas para a Educação, a Ciência e a Cultura (UNESCO¹³), deve-se enviar a requisição ilustrada no Exemplo 14.

Exemplo 14 – X REST API: Requisição para coleta de conjuntos de dados de tuítes, por meio de descoberta de conteúdo, via método GET

```
GET api.twitter.com/search/tweets.json?p=@UNESCO
```

Fonte: Elaborado pelo Autor.

As requisições da X REST API também possuem um sistema para o tratamento de erros de processamento ou de autorização. Por exemplo, no caso de uma requisição ser enviada com um *token* de autorização de acesso inválido, o resultado da requisição (Exemplo 15) estará disponível em formato de notação da linguagem de marcação JSON, com atributos específicos para esta finalidade.

Exemplo 15 – X REST API: Tratamento de erros de processamento ou de autorização

```
{
  "errors" : [
    {
      "message" : "The access token used in the request is incorrect or has expired",
      "code" : 89
    }
  ]
}
```

Fonte: Elaborado pelo Autor.

¹³ Acrônimo com origem na língua inglesa da *United Nations Educational, Scientific and Cultural Organization*.

Portanto, caso uma requisição apresente erro, a REST API retorna um atributo composto com o nome *errors*, contendo os seguintes atributos:

- a) *message*: contendo a descrição do erro;
- b) *code*: com o código do erro. A lista de possibilidades de valores está descrita na seção Visão Geral da API (X CORP., 2023g).

A seção Índice de Referência da API apresenta a descrição de cada visão disponível para coleta de dados (X CORP., 2023e). Foram identificadas 64 visões disponíveis para coleta de dados, sendo complementada por cinco documentos oriundos da seção Visão Geral da API, contendo 10 visões extras: *Entities*, *Entities in Objects (Extended Entities)*, *Entities in Objects (Hashtags)*, *Entities in Objects (Media)*, *Entities in Objects (Symbols)*, *Entities in Objects (URLs)*, *Entities in Objects (User Mentions)*, *Places*, *Tweets* e *Users*.

Cada visão possui documento próprio, com as informações:

- a) URL do documento: cada um destes documentos possui um endereço eletrônico para acesso individualizado de suas informações, concomitante ao formato URL;
- b) Título: elemento textual contendo o título da visão. Não há ocorrências da X REST API de títulos iguais para visões distintas (homônimos);
- c) Descrição da Visão (elemento não obrigatório): elemento textual, na forma de um ou mais parágrafos, descrevendo a visão;
- d) Recurso URL (elemento não obrigatório): elemento, em formato de URL, contendo o endereço disponível, com o protocolo HTTPS, para a realização de requisições de conjuntos de dados da visão, em formato JSON;
- e) Informação do Recurso (elemento não obrigatório): apresenta um quadro informativo com cinco linhas, informando: i) formato da

- reposta da requisição; ii) se o acesso aos conjuntos de dados da visão requer autenticação/autorização prévia; iii) se existe limitação de quantidade de linhas em cada requisição (ou seja, se necessita de elementos de paginação); iv) limites de requisição para acessar dados de cada Referenciado, e; v) limites de requisição para cada aplicação;
- f) Parâmetros (elemento não obrigatório): área contendo um quadro informativo de parâmetros, informando o nome, a obrigatoriedade de uso, o valor padrão e exemplos de quais são os parâmetros disponíveis na visão;
 - g) Exemplo de Requisição (elemento não obrigatório): elemento, em formato de URL, contendo um exemplo do uso, com o protocolo HTTPS, e de parâmetros para a realização de requisições de conjuntos de dados da visão, em formato JSON;
 - h) Exemplo de Resposta de Requisição (elemento não obrigatório): um exemplo da resposta da requisição de exemplo (item g), contendo os atributos da visão expressos em uma resposta no formato JSON;
 - i) Guia de Campos (elemento não obrigatório): área contendo um quadro informativo, de três colunas (nome, tipo de dado, descrição), com informações sobre os atributos disponíveis na visão: o nome, o tipo de dado esperado e uma descrição para os atributos disponíveis na visão no momento da coleta de dados.

As informações sobre ações de criação, atualização e exclusão de dados não aparecem nos documentos, pois são tratadas de maneira separada das ações de coleta de dados.

Foram identificadas 74 visões vinculadas com a X REST API, sendo 10 visões focadas em classes de conteúdo consideradas principais para o X e 64 visões voltadas a auxiliar a consulta aos demais tipos de conteúdo. Dentre as características identificadas, destacam-se que:

- a) A visão *Coordinates* não possui descrição;
- b) As visões *Entities in Objects (Extended Entities)*, *Entities in Objects (Hashtags)*, *Entities in Objects (Media)*, *Entities in Objects (Symbols)*, *Entities in Objects (URLs)* e *Entities in Objects (User Mentions)* estão descritas em um único documento;
- c) As visões *Coordinates* e *Users* estão descritas juntas, em um único documento.

Na documentação técnica, foram encontradas as referências dos tipos de dados para as visões *Entities*, *Entities in Objects (Extended Entities)*, *Entities in Objects (Hashtags)*, *Entities in Objects (Media)*, *Entities in Objects (Symbols)*, *Entities in Objects (URLs)*, *Entities in Objects (User Mentions)*, *Places*, *Tweets* e *Users*. As demais visões apresentam os nomes de atributos e tipos de valores apenas no formato de exemplo em JSON, na área Exemplo de Resposta de Requisição.

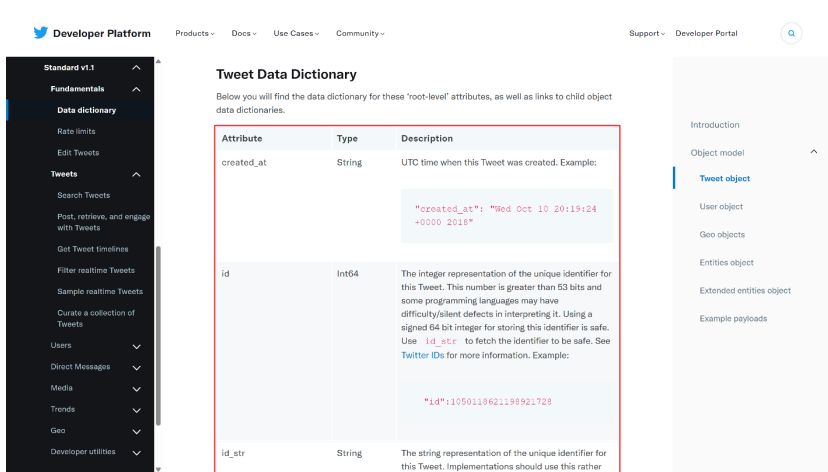
É importante ressaltar que do total de 73 visões que contém elementos textuais voltados a sua descrição, algumas descrições não dão subsídio suficiente para servir como elemento de sustentação a compreensão do conteúdo a ser coletado, como, por exemplo, a descrição da visão *Geo ID Place ID*.

Nas subseções Dicionário de Dados e Exemplo de Resposta são encontradas as informações sobre os atributos e tipos de dados disponíveis no momento da coleta de dados.

Estas informações são disponibilizadas no formato HTML, no próprio corpo documental, e possuem duas formas de apresentação: um quadro informativo no formato de tabela HTML e uma com o uso da marcação de bloco de citação do HTML.

Na primeira forma de apresentação (Figura 13), o quadro informativo contendo a descrição dos atributos disponíveis na coleta de dados possui três colunas (ênfatisadas por um retângulo com a borda na cor vermelha): a) *Field*, contendo o nome do atributo; b) *Type*, contendo o tipo de dado aceito para atributo, e; c) *Description*, com a descrição do atributo.

Figura 13 – XREST API: Atributos na primeira forma de apresentação, tabela HTML com três colunas



Fonte: Recorte de X Corp. (2023h), elaborado pelo Autor.

A segunda forma de apresentação, com o uso da marcação de bloco de citação do HTML (Figura 14), exhibe as informações sobre os atributos somente na forma de um exemplo de resposta de requisição a consultas de conjuntos de dados da visão (via método GET). Os exemplos estão inseridos dentro da marcação *pre*¹⁴ do HTML (ênfatisadas por um retângulo com a borda na cor vermelha) e seu texto está estruturado na forma de conjunto de dados no formato de notação da linguagem de marcação JSON.

¹⁴ Ver World Wide Web Consortium (2023).

Figura 14 – X REST API: Atributos na segunda forma de apresentação, com o uso da marcação de bloco de citação do HTML



Fonte: Recorte de X Corp. (2023i), elaborado pelo Autor.

Portanto, neste tipo de apresentação não há informações explícitas sobre o tipo de dado aceito em cada atributo. Para a identificação do tipo de dado de cada atributo é necessário a interpretação do valor dos exemplos.

A referência dos elementos descritivos para os atributos também pode apresentar alguns qualificadores – marcações especiais que agregam características específicas aos atributos.

Os qualificadores identificados podem ser do tipo:

- Default*: o atributo marcado com este qualificador ficará disponível automaticamente na resposta da requisição, ou seja, na requisição de coleta de dados de uma visão não haverá necessidade de explicitar a API retorno do valor do atributo;
- Deprecated*: o atributo marcado com este qualificador foi considerado obsoleto pela equipe de desenvolvimento da API e, caso disponível na coleta de dados, o seu valor será nulo na resposta da requisição.

Os qualificadores não possuem uma área própria para sua descrição. Em todos os casos os qualificadores foram identificados nos elementos textuais de descrição dos atributos.

Os valores dos atributos podem ser simples – quando o valor é um número, signos, data e outros formatos – ou composto. No caso os atributos compostos, podem variar entre:

- a) Vetores: quando o valor do atributo é um conjunto de valores, representado com o uso de vetores computacionais (*arrays*). Os atributos com este tipo de valor apresentam no seu tipo de dado o sufixo [];
- b) Listas: quando o valor do atributo é um conjunto de valores com tipo de dados pré-fixados, com o uso de listas computacionais (*list*). Os atributos com este tipo de valor apresentam no seu tipo de dado o prefixo *list*;
- c) Visões: quando o valor do atributo é um registro de outra visão. Os atributos com este tipo de valor apresentam como tipo de dado o nome de uma Visão.

Foram identificados 803 atributos, sendo que um total de 616 atributos possuem valor do tipo simples, 185 atributos possuem valor do tipo composto e dois atributos que não foi possível realizar esta identificação. Um total de 178 atributos são utilizados para relacionar diferentes visões.

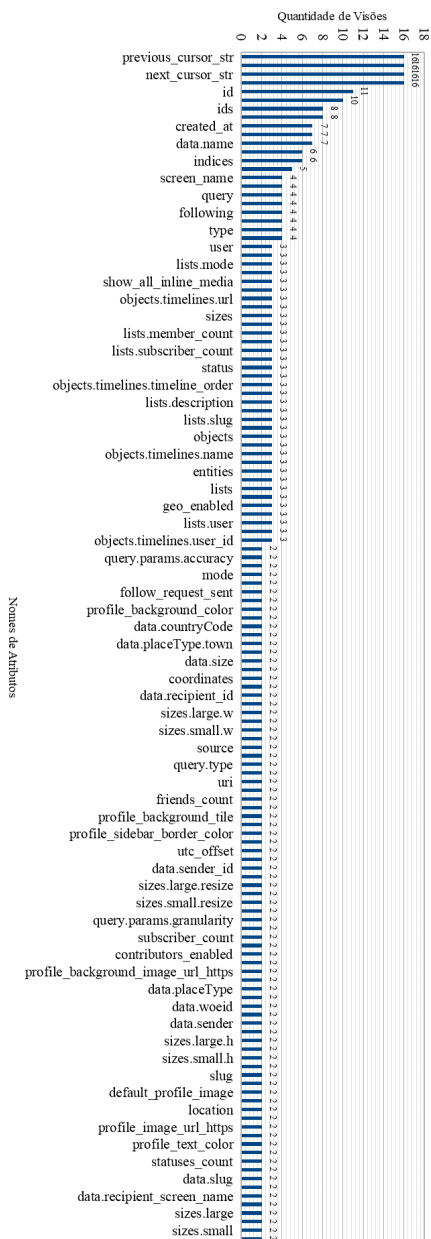
Com relação aos qualificadores, 11 atributos são retornados automaticamente pelas requisições no momento da coleta de dados, 49 atributos precisam ser explicitados na requisição para a coleta de dados e 743 atributos em que não foi possível realizar esse tipo de identificação. Foram encontrados quatro atributos obsoletos e 799 atributos que não foi possível realizar esta identificação.

Foram encontrados 477 nomes distintos para atributos da X REST API, porém um mesmo nome de atributo pode aparecer em visões distintas.

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Gráfico 3 – XREST API: Ocorrências de nomes de atributos iguais em diferentes visões



Fonte: Elaborado pelo Autor.

O Gráfico 3 exibe as ocorrências de nomes de atributos que foram utilizados em duas ou mais visões. Destacam-se que seis nomes são utilizados em 10 ou mais visões (*id*, *id_str*, *next_cursor*, *next_cursor_str*, *previous_cursor* e *previous_cursor_str*) e oito nomes são utilizados entre cinco e nove visões (*created_at*, *data.id*, *data.name*, *ids*, *indices*, *name*, *url* e *users*). Um total de 124 nomes de atributos são utilizados entre 2 e 4 visões e 338 nomes de atributos são utilizados em apenas uma visão da API.

A XREST API contém 25 tipos de dados, que podem ser classificados segundo suas características, em três categorias:

- a) Tipos de dados de Sistema de Gerenciamento de Bancos de Dados: são encontrados na maioria dos Sistemas de Gerenciamento de Bancos de Dados, como números inteiros, *string* para caracteres, datas, textos longos, booleanos, entre outros;
- b) Tipos de dados de algoritmos de Linguagens de Programação: são tipos de dados encontrados em linguagens de programação e também próprios do contexto da API;
- c) Tipos de dados para relacionar conteúdos de duas visões: são tipos de dados para atributos que apresentam como valor um subconjunto de atributos originários de uma outra visão no momento da coleta.

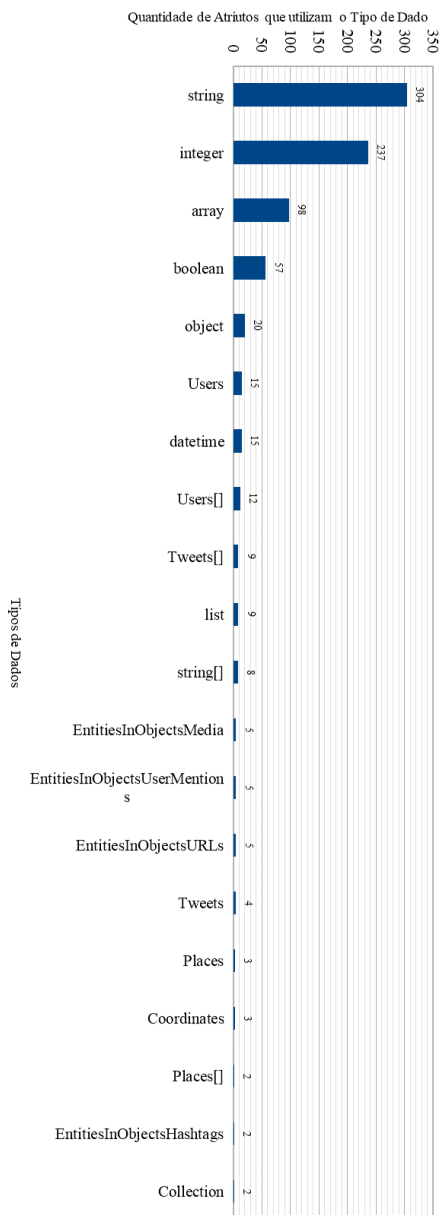
Um mesmo tipo de dado pode ter até duas características. Por exemplo, o tipo de dado *list<EntitiesInObjectsURLs>* é uma lista (tipo de dado específico) que apresenta subconjuntos da visão *Entities in Objects (URLs)* (tipo de dado que apresenta conteúdo de uma visão) – mesclando duas categorias.

Os tipos de dados precisam ser concomitantes com a estrutura identificadas nos atributos e, portanto, seus valores podem ser simples – quando o valor é um número, símbolos, textos ou datas – ou compostos – quando o valor é composto por um subconjunto de atributos.

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Gráfico 4 – X REST API: Ocorrências de tipos de dados em atributos



Fonte: Elaborado pelo Autor.

O Gráfico 4 exibe as ocorrências de tipos de dados que foram utilizados em dois ou mais atributos. Com relação a ocorrências destes tipos de dados nos atributos, foram identificados:

- a) Oito tipos de dados utilizados em mais de 10 atributos, sendo quatro simples e quatro compostos: *string* (simples), *integer* (simples), *array* (composto), *boolean* (simples), *object* (composto), *Users* (composto), *datetime* (simples) e *Users[]* (composto);
- b) Seis tipos de dados utilizados entre cinco a nove atributos, sendo todos compostos: *EntitiesInObjectsMedia*, *EntitiesInObjectsURLs*, *EntitiesInObjectsUserMentions*, *list*, *string[]* e *Tweets[]*;
- c) Seis tipos de dados utilizados entre dois e quatro atributos, sendo todos compostos: *Collection*, *Coordinates*, *EntitiesInObjectsHashtags*, *Places*, *Places[]* e *Tweets*.

Um total de cinco tipos de dados são utilizados em apenas um atributo, sendo um do simples e quatro compostos: *float* (simples), *Entities[]* (composto), *Contributors* (composto), *Coordinates[]* (composto) e *integer[]* (composto). Não foi possível identificar os tipos de dados dos atributos *data.trends.events* e *data.trends.promoted_content* da visão *Trends Place (trends/place)*.

Os relacionamentos entre visões seguem princípios similares ao de cardinalidade entre tabelas nos Sistemas de Gerenciamento de Bancos de Dados (Silberschatz; Korth; Sudarshan, 2020). Foram identificados o uso das cardinalidades: 1-para-1, quando um registro da visão de origem relaciona-se com um registro da visão de destino; 1-para-N, quando um registro da visão de origem relaciona-se com uma ou mais registros da visão de destino, e; N-para-N, quando um ou mais registros da visão de origem relacionam-se com um ou mais registros da visão de destino.

Os relacionamentos podem ser explicitados na documentação técnica das visões de duas formas diferentes:

- a) A partir dos atributos: quando o tipo de dado de um atributo é uma visão, considera-se que o valor do atributo representa um relacionamento com outra visão, proporcionando que, ao coletar conjuntos de dados da visão de origem, os dados da visão de destino também podem ser coletados através deste atributo;
- b) A partir das arestas: são relacionamentos a partir de arestas disponíveis na visão de origem que interligam os seus registros com registros de outras visões, porém com a obrigatoriedade da execução de uma nova requisição na coleta de dados. Ou seja, ao coletar os conjuntos de dados de uma determinada visão, é possível realizar outras requisições para coletar conjuntos de dados das visões relacionadas por arestas.

Foram identificados 111 relacionamentos, sendo: 105 com o uso de atributos e seis com o uso de arestas. Nos relacionamentos com o uso de atributos, 58 possuem a cardinalidade N-para-N, 45 a cardinalidade 1-para-N e dois possuem a cardinalidade 1-para-1. Já nos relacionamentos por aresta, todos possuem a cardinalidade 1-para-1.

As requisições disponíveis para o acesso às visões funcionam a partir do uso do protocolo HTTPS, método GET, tanto para a transmissão das informações das requisições, como para o recebimento dos conjuntos de dados coletados. Como as requisições estão diretamente vinculadas com as visões, a descrição das funcionalidades e a lista com parâmetros de consulta estão disponíveis na documentação técnica, nas seções:

- a) Recurso URL: elemento, em formato de URL, contendo o endereço disponível, com o protocolo HTTPS, para a realização de requisições (denominado GET) de conjuntos de dados da visão, em formato de notação da linguagem de marcação JSON;
- b) Parâmetros (elemento não obrigatório): área contendo um quadro informativo de parâmetros, com cinco colunas, contendo o nome,

a obrigatoriedade de uso, a descrição, o valor padrão e exemplos de quais são os parâmetros disponíveis na visão;

- c) Exemplo de Requisição (elemento não obrigatório): elemento, em formato de URL, contendo um exemplo do uso do endereço disponível, com o protocolo HTTPS, e de parâmetros para a realização de requisições de conjuntos de dados da visão, em formato de notação da linguagem de marcação JSON;
- d) Exemplo de Resposta de Requisição (elemento não obrigatório): um exemplo da resposta da requisição de exemplo (item c), contendo os atributos da visão expressos em uma resposta no formato de notação da linguagem de marcação JSON.

Os documentos do tipo POST, que contém informações sobre os métodos de inserção, de atualização e de exclusão de conjuntos de dados, estão separados dos documentos de consulta aos dados (X CORP, 2023d), fora do escopo de análise.

Das 74 visões disponíveis para a coleta por API, um total de 10 visões não possuem requisições para coleta dos conjuntos de dados: *Entities*, *Entities in Objects (Extended Entities)*, *Entities in Objects (Hashtags)*, *Entities in Objects (Media)*, *Entities in Objects (Symbols)*, *Entities in Objects (URLs)*, *Entities in Objects (User Mentions)*, *Places*, *Tweets e Users*. O acesso ao conjunto de dados destas visões deve ser realizado a partir de outras visões, pelo processo de relacionamento por atributos ou por arestas.

Nas 64 requisições disponíveis, foram identificados um total de 234 parâmetros para a coleta, com as seguintes características:

- a) Todos os parâmetros possuem descrição e tipo de dado esperado;
- b) Não possuem parâmetros de entrada: *Account Settings*, *Help Configuration*, *Help Languages*, *Help Privacy*, *Help TOS*, *Saved Searches List* e *Trends Available*.

Foram encontrados 63 nomes de parâmetros distintos, sendo que um total de 37 são utilizados somente em um método. No caso de nomes de parâmetros utilizados em mais de um método:

- a) Cinco nomes são utilizados em 10 ou mais métodos: *count*, *include_entities*, *user_id*, *cursor* e *screen_name*;
- b) Nove nomes são utilizados entre cinco a nove métodos: *id*, *skip_status*, *since_id*, *slug*, *max_id*, *stringify_ids*, *trim_user*, *owner_screen_name*, *list_id* e *owner_id*;
- c) Um total de 12 nomes são utilizados entre dois a quatro métodos: *lang*, *long*, *lat*, *callback*, *include_rts*, *page*, *granularity*, *max_results*, *q*, *include_user_entities*, *exclude_replies* e *accuracy*.

Os parâmetros possuem tipos de dados, assim como os atributos. Apresentam as seguintes características:

- a) Três tipos de dados são utilizados em 10 ou mais parâmetros: *integer*, *string* e *boolean*;
- b) Dois de tipos de dados são utilizados entre cinco a nove parâmetros: *list* e *long*;
- c) Um tipo de dado é utilizado entre dois e quatro parâmetros: *json*;
- d) Um tipo de dado é utilizado apenas em um parâmetro: *datetime*.

Para acessar as requisições é necessário o uso de um conjunto de permissões. No caso da X REST API, as permissões para a coleta de dados estão vinculadas ao uso dos *tokens* de autorização de acesso.

Os *tokens* de autorização de acesso são um conjunto finito de símbolos, gerados pelo serviço, com a finalidade de providenciar uma contrassenha para verificação de credenciais. Estes *tokens* de autorização de acesso são um dos itens que compõem o processo de verificação inicial de

credenciais – ou seja, o processo do sistema de entrada que permite que o acesso seja concedido e que habilite aplicações para a coleta de conjuntos de dados. São obtidos pelo uso do protocolo HTTPS, sendo possível sua implementação em diversas linguagens de programação. Após a sua criação, os Agentes Externos têm acesso a informações complementares, como o tempo de validade do *token* e o aplicativo que solicitou a sua geração (X CORP., 2023j). No *X*, os *tokens* de autorização são parte de um sistema de entrada denominado *OAuth*¹⁵, desenvolvido com o propósito de auxiliar no processo de identificação digital (entrada) entre duas ou mais partes, dentro de aplicativos e *websites* (X CORP., 2023c).

No caso do *X*, as permissões de acesso são pré-definidas em cada autorização de acesso. Ou seja, as permissões de acesso a um aplicativo externo para coleta de conjuntos de dados de Referenciados, de tuítes ou de recursos audiovisuais – e as visões relacionadas a todos estes entes – são concedidas no momento que um Referenciado utiliza um *tokens* de autorização para conectar um aplicativo externo ou um código-fonte de *website* ao seu perfil, ou quando um Referenciado utiliza um *tokens* de autorização para conectar um aplicativo externo ou um código-fonte de *website* com uma conta vinculada a um dos produtos e serviços disponíveis na plataforma do *X* (X CORP., 2023c).

As informações sobre quais são os *tokens* de autorização de acesso existentes para acessar conjuntos de dados são encontradas em de dois documentos: nos documentos contendo informações sobre o sistema de entrada *OAuth* e na documentação técnica das visões.

No caso dos *tokens* de autorização de acesso existentes, as informações encontram-se em um documento denominado *OAuth*, que integra o conjunto de documentos que descrevem como aplicações e códigos-fonte de *websites* podem conectarem por meio do sistema de entrada *OAuth* para autorização da interoperabilidade com os conteúdos disponíveis no serviço (X CORP., 2023j).

¹⁵ Nome dado em função de ser uma implementação de sistema baseada no protocolo de padrão aberto *OAuth*.

Segundo o documento, existem dois tipos de *tokens* de autorização de acesso:

- a) *User Access Token*: é o *token* de autorização de acesso mais utilizado, que permite acesso pela *X* REST API aos conjuntos de dados de Referenciados. Autoriza a leitura, e a criação, a alteração e exclusão de conteúdo. Em algumas operações, os Referenciados precisam autorizar a aplicação externa para realizar as operações de manipulação de dados: criação, alteração e exclusão;
- b) *App Access Token*: utilizado para que as aplicações externas possam realizar requisições aos conjuntos de dados dos Referenciados, porém sem um alvo definido, como, por exemplo, um Referenciado específico. Entretanto, possui taxas de requisição maiores (o que permite maior volume de dados na coleta) e é indicada para ações tais como: a leitura de tuítes na linha do tempo; acessar a lista de seguidores de qualquer Referenciado; o acesso a recursos de listas, e; pesquisas de tuítes públicos. Não autoriza as operações de manipulação de conjuntos de dados (X CORP., 2023k).

Em ambos os casos, para a obtenção do conjunto de dados necessários para autorização via sistema de entrada *OAuth*, é necessário um cadastro prévio da aplicação ou do código-fonte na plataforma Gerenciamento de Aplicações do *X* (no idioma inglês, *X Application Management*), pela URL <https://apps.twitter.com>. Ao completar o cadastro, o *X* autorizará a conexão a seus produtos e serviços (incluindo as APIs disponíveis), e o Agente Externo receberá os seguintes dados:

- a) *oauth_consumer_key*: contém como valor a chave para a identificação de qual aplicação ou do *website* realizou a requisição;
- b) *oauth_signature*: contém como valor a chave pública para a verificação de assinatura;

- c) *oauth_signature_method*: contém como valor o tipo de método de criptografia utilizado para a geração da chave pública.

Além disso, o Agente Externo deve informar em cada requisição os parâmetros:

- a) *oauth_version*: contém como valor a versão do protocolo do sistema de entrada *OAuth* que será utilizado;
- b) *oauth_nonce*: contendo uma chave para cada requisição solicitada;
- c) *oauth_timestamp*: seu valor representa a data e o horário que a requisição foi criada, no formato *UNIX epoch*;
- d) *oauth_token*: contém a chave para acesso aos dados de um Referenciado, sendo que seu preenchimento é opcional em certos usos (como no caso do *App Access Token*).

Não há um documento único que estabeleça quais são as permissões de acesso de cada *token* de autorização de acesso. Entretanto na documentação técnica das visões estão preestabelecidas quais requisições serão aceitas para o acesso a dados das visões de forma individual, no momento da conexão para o início do processo de coleta de dados pelos aplicativos externos e, portanto, estas permissões são os elementos já estabelecidos pelo *X* que delimitam nas visões os conjuntos de dados de Referenciados, de tuítes e de entidades multimídia (e as visões relacionadas a estes entes) que estarão acessíveis para a coleta por aplicações externas.

Figura 15 – X REST API: Subseção contendo informações sobre as permissões da visão

The screenshot shows the Twitter Developer Platform documentation for the 'Retweets of Me' resource. The page is titled 'Resource URL' and 'Resource Information'. The 'Requires authentication?' field is highlighted with a red box, indicating that this resource requires authentication. The 'Parameters' table lists the following parameters:

Name	Required	Description	Default Value	Example
count	optional	Specifies the number of records to retrieve. Must be less than or equal to 100. If omitted, 20 will be assumed.	5	
since_id	optional	Returns results with an ID greater than (that is, more recent than) the specified ID. There are	12345	

Fonte: Recorte de X Corp. (2023i), elaborado pelo Autor.

Na subseção Informações sobre o Recurso (Figura 15), contidas nos documentos das visões, é possível identificar quais são os pontos de entrada que funcionam somente com o *token* de autorização de acesso *User Access Token*. Os demais pontos de acesso disponíveis são acessíveis por qualquer *token* de autorização de acesso (X CORP., 2023j).

O Quadro 2 apresenta a síntese com informações identificadas sobre as permissões para cada *token* de autorização de acesso, contendo: a) o nome da visão; b) critério condicional, indicando se a visão está disponível para o acesso via *User Access Token*; c) critério condicional, indicando se a visão está disponível para o acesso via *App Access Token*, e; d) critério condicional, indicando se a visão está disponível para o acesso público, sem a necessidade de identificação no X.

Quadro 2 –Referências de permissões da X REST API, versão 1.1

Nome da Visão	Acessível pelo Token de Autorização de Acesso		Acesso Público?
	User Access Token	App Access Token	
Account Settings	Sim	Não	Não
Account Verify Credentials	Sim	Não	Não
Application Rate Limit Status	Sim	Sim	Não
Blocks IDs	Sim	Não	Não
Blocks List	Sim	Não	Não
Collections Entries	Sim	Não	Não
Collections List	Sim	Não	Não
Collections Show	Sim	Não	Não
Coordinates	Sim*	Sim*	Não
Direct Messages	Sim	Não	Não
Direct Messages Sent	Sim	Não	Não
Direct Messages Show	Sim	Não	Não
Entities	Sim*	Sim*	Não
Entities in Objects (Extended Entities)	Sim*	Sim*	Não
Entities in Objects (Hashtags)	Sim*	Sim*	Não
Entities in Objects (Media)	Sim*	Sim*	Não
Entities in Objects (Symbols)	Sim*	Sim*	Não
Entities in Objects (URLs)	Sim*	Sim*	Não
Entities in Objects (User Mentions)	Sim*	Sim*	Não
Favorites List	Sim	Sim	Não
Followers IDs	Sim	Sim	Não
Followers List	Sim	Sim	Não
Friends IDs	Sim	Sim	Não
Friends List	Sim	Sim	Não
Friendships Incoming	Sim	Não	Não
Friendships Lookup	Sim	Não	Não
Friendships No Retweets IDs	Sim	Não	Não
Friendships Outgoing	Sim	Não	Não
Friendships Show	Sim	Sim	Não
Geo ID Place ID	Sim	Não	Não
Geo Reverse Geocode	Sim	Não	Não

Estruturas de dados em serviços de redes sociais online
 Uma abordagem metodológica de análise

Nome da Visão	Acessível pelo Token de Autorização de Acesso		Acesso Público?
	User Access Token	App Access Token	
Geo Search	Sim	Não	Não
Help Configuration	Sim	Sim	Não
Help Languages	Sim	Sim	Não
Help Privacy	Sim	Sim	Não
Help TOS	Sim	Sim	Não
Lists List	Sim	Sim	Não
Lists Members	Sim	Sim	Não
Lists Members Show	Sim	Sim	Não
Lists Memberships	Sim	Sim	Não
Lists Ownerships	Sim	Sim	Não
Lists Show	Sim	Sim	Não
Lists Statuses	Sim	Sim	Não
Lists Subscribers	Sim	Sim	Não
Lists Subscribers Show	Sim	Sim	Não
Lists Subscriptions	Sim	Sim	Não
Mutes Users IDs	Sim	Não	Não
Mutes Users List	Sim	Não	Não
Places	Sim*	Sim*	Não
Projects	Sim	Sim	Não
Saved Searches List	Sim	Não	Não
Saved Searches Show ID	Sim	Não	Não
Search Tweets	Sim	Sim	Não
Statuses Home Timeline	Sim	Não	Não
Statuses Lookup	Sim	Sim	Não
Statuses Mentions Timeline	Sim	Não	Não
Statuses Oembed	Sim	Sim	Sim
Statuses Retweeters IDs	Sim	Sim	Não
Statuses Retweets of Me	Sim	Não	Não
Statuses Retweeters ID	Sim	Sim	Não
Statuses Show ID	Sim	Sim	Não
Statuses User Timeline	Sim	Sim	Não
Trends Available	Sim	Sim	Não
Trends Closest	Sim	Sim	Não

Nome da Visão	Acessível pelo Token de Autorização de Acesso		Acesso Público?
	User Access Token	App Access Token	
Trends Place	Sim	Sim	Não
Tweets	Sim*	Sim*	Não
Users	Sim*	Sim*	Não
Users Lookup	Sim	Sim	Não
Users Profile Banner	Sim	Não	Não
Users Search	Sim	Não	Não
Users Show	Sim	Sim	Não
Users Suggestions	Sim	Sim	Não
Users Suggestions Slug	Sim	Sim	Não
Users Suggestions Slug Members	Sim	Sim	Não

* Somente por acesso indireto (via relacionamento).

Fonte: Elaborado pelo Autor, a partir de *X Corp.* (2023d).

Com o uso do *User Access Token*, todas as visões estão disponíveis para o acesso aos conjuntos de dados. O *User Access Token* possui restrições para 26 pontos de entrada: *Account Settings*, *Account Verify Credentials*, *Blocks IDs*, *Blocks List*, *Collections Entries*, *Collections List*, *Collections Show*, *Direct Messages*, *Direct Messages Sent*, *Direct Messages Show*, *Friendships Incoming*, *Friendships Lookup*, *Friendships No Retweets IDs*, *Friendships Outgoing*, *Geo ID Place ID*, *Geo Reverse Geocode*, *Geo Search*, *Mutes Users IDs*, *Mutes Users List*, *Saved Searches List*, *Saved Searches Show ID*, *Statuses Home Timeline*, *Statuses Mentions Timeline*, *Statuses Retweets of Me*, *Users Profile Banner* e *Users Search*.

Somente a requisição da visão *Statuses Oembed* é acessível sem o uso do sistema de entrada, pois trata-se de uma requisição com o propósito de retornar um tuíte público, estruturado em JSON, a partir de uma URL ou *id* de um tuíte.

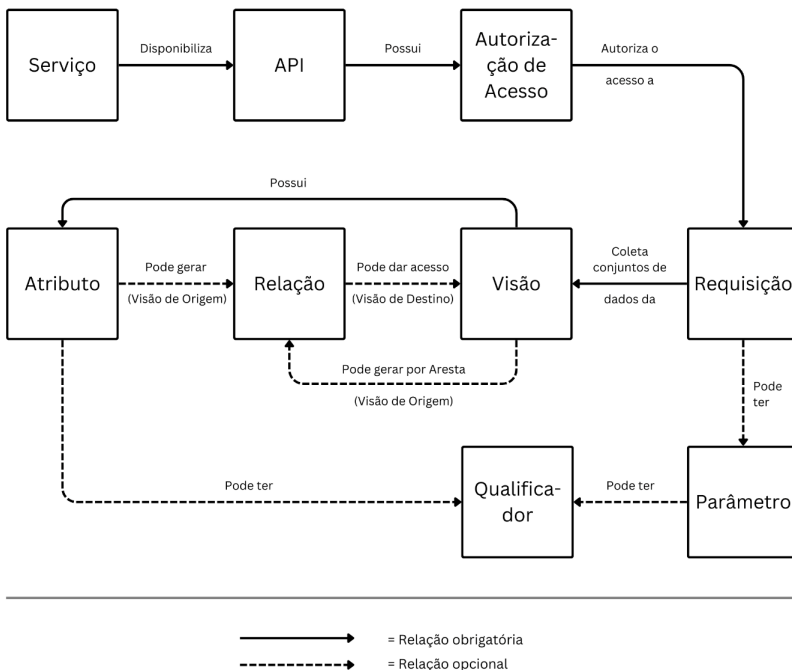
A partir das características da X REST API apresentadas nesta seção, foram identificadas as seguintes entidades para a sistematização da coleta de dados:

- a) Serviço, com informações referentes ao Serviço de Rede Social *Online*: nome do serviço e a URL de acesso;
- b) API, com informações referentes a API: nome da API, versão, descrição e URL para acesso as requisições;
- c) Visão, contendo informações sobre as visões disponíveis: nome da visão, descrição e URL para acesso aos documentos;
- d) Atributos de cada visão, com informações dos atributos disponíveis nos resultados das requisições: nome do atributo, descrição, tipo de dado e qualificadores;
- e) Qualificadores, com informações sobre as características das marcações que diferenciam parte dos atributos das visões, e parte dos parâmetros da requisição: nome do qualificador e descrição;
- f) Requisições existentes em cada visão, contendo informações sobre quais operações de coleta de dados estão disponíveis para a visão: nome da requisição, tipo e nome do método e seus parâmetros de entrada e saída;
- g) Parâmetros existentes nas requisições, em um relacionamento que uma requisição pode ter nenhum, um ou mais parâmetros: nome do parâmetro, descrição e qualificadores;
- h) Relações, com informações sobre os relacionamentos entre as visões: nome da visão de origem, nome da visão de destino, tipo de relacionamento, nome do relacionamento, cardinalidade na origem e cardinalidade no destino;
- i) Autorizações de acesso, com informações sobre os tipos de acesso: nome, descrição e URL para acesso aos documentos.

A Figura 16 apresenta uma representação gráfica das relações entre as entidades propostas. O serviço apresenta formas de acesso aos seus conjuntos de dados por meio da API, que possui um sistema de entrada *OAuth* baseado em *tokens* (com a função de autorizar o acesso e estabelecer quais conjuntos de dados podem ser coletados).

Por meio de pontos de entrada disponíveis, o sistema de autorização permite que seja realizada a execução de requisições para coletar dados, permitindo o acesso aos dados contidos nas visões. No caso desta API, o sistema de autorização não faz uso de Permissões. No momento do disparo da execução das requisições, podem existir parâmetros de entrada de dados, e estes parâmetros podem ter qualificadores – marcações que estabelecem características especiais a certos parâmetros – como o estado de obrigatoriedade de uso do parâmetro na requisição.

Figura 16 – X REST API: Vínculos entre as entidades



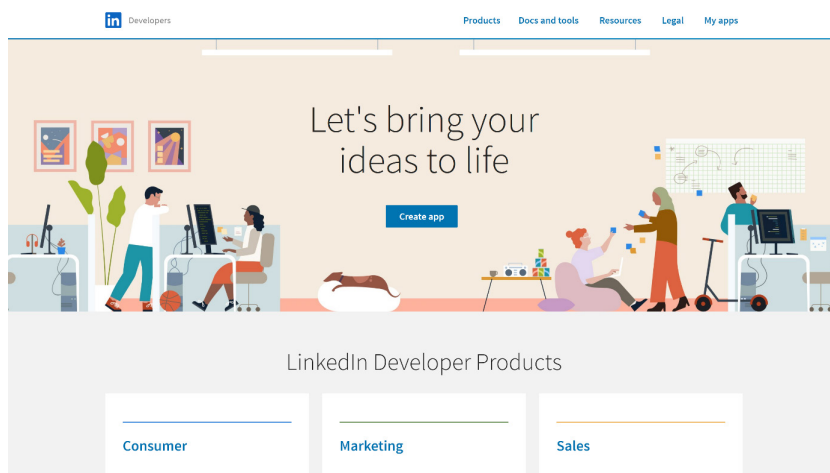
Fonte: Elaborado pelo Autor.

As visões possuem: atributos, com tipologia de dados própria, e que também podem apresentar qualificadores (marcações que estabelecem características especiais a certos parâmetros); e relações com outras visões da API – que são estabelecidas a partir de arestas (com o uso de identificadores da visão de origem e da visão de destino) ou pelos valores compostos de atributos, que apresentam como tipo de dado a visão de destino.

1.5 A *LINKEDIN* REST API

Para acessar os documentos técnicos da *LinkedIn* REST API, o *LinkedIn Corporation* possui uma área denominada *LinkedIn Developer Solutions* em seu *website* (Figura 17). Esta área é exclusiva para a concentração de documentos, de referências, de exemplos e de demais informações referentes ao processo de utilização de serviços oferecidos para Agentes Externos e parceiros.

Figura 17 – *LinkedIn Corporation*: Página principal da área reservada aos desenvolvedores



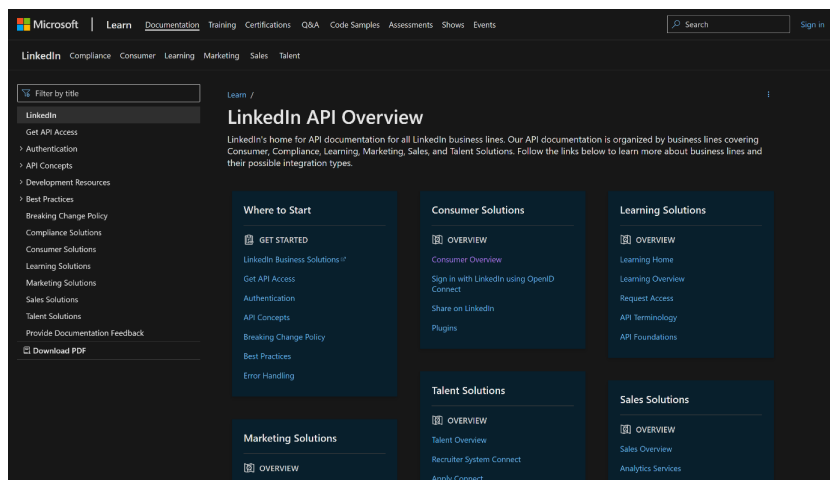
Fonte: Recorte de *LinkedIn Corporation* (2023b), elaborado pelo Autor.

Conforme as informações encontradas nos rótulos do menu de navegação superior, esta área está dividida em cinco seções: Produtos, Documentos e Ferramentas, Recursos, Legal e Meus Aplicativos (LinkedIn Corporation, 2023b). O botão *View Consumer Docs* da seção Produtos dá acesso as coleções de documentos técnicos referentes aos produtos e serviços oferecidos pelo *LinkedIn*, Inc. aos Agentes Externos, especificamente à comunidade de desenvolvedores de aplicações externas – denominada *Microsoft Learn - LinkedIn - LinkedIn API Documentation* (LinkedIn Corporation, 2023c).

A *LinkedIn* REST API é a principal API do *LinkedIn*, desenvolvida como o objetivo de realizar a interoperabilidade de conjuntos de dados de Referenciados e de páginas de companhias, com Agentes Externos. As requisições e as respostas das requisições estão acessíveis através do protocolo HTTPS, tanto para a coleta de dados quanto para realização de operações orientadas a inserção, a alteração ou a exclusão de dados, em associação ao uso do sistema de entrada *OAuth* (baseado no padrão de código aberto de mesmo nome) (LinkedIn Corporation, 2023d, 2023e). Os documentos estão disponíveis nos idiomas inglês e mandarim.

O conjunto de documentos contendo as descrições de sua funcionalidade estão divididos em 13 seções (Figura 18, coluna da esquerda com fundo preto): Obter Acesso a API, Autenticação, Conceitos da API, Recursos para Desenvolvimento, Boas Práticas, Política de Quebra de Mudança, Soluções de *Compliance*, Soluções de Consumo, Soluções de Aprendizagem, Soluções de *Marketing*, Soluções de Venda, Soluções de Talento e Fornecer *Feedback* da Documentação.

Figura 18 – *LinkedIn Corporation*: Seções disponíveis dos documentos da *LinkedIn REST API*



Fonte: Recorte de *LinkedIn Corporation* (2023c), elaborado pelo Autor.

As informações relacionadas com os conjuntos de dados disponíveis, atributos, relacionamentos entre atributos e demais conjuntos de dados, bem como a descrição destes, encontram-se na seção Conceitos da API (LinkedIn Corporation, 2023f) – sendo esta seção o escopo deste livro.

No momento da coleta, os conjuntos de dados da *LinkedIn REST API* são apresentados na forma de um sociograma. Os conjuntos de dados disponíveis para coleta são pré-determinados e selecionados pela instituição, na forma de visões. Cada visão é tratada como um nó do sociograma.

Para coletar os conjuntos de dados de um Referenciado, o Agente Externo deve realizar uma requisição de acesso ao nó *Member*. Esse nó possui características idênticas com o processo de requisição de uma visão (ver Figura 3, seção 1.1). Portanto, cada nó contém uma coleção de atributos e registros, unicamente identificáveis. Os atributos possuem um número como identificador de cada registro armazenado, independente da visão, denominado *id*. Em cada visão, o *id* está associado a um identificador único somente para aquela visão e, portanto, podem existir valores iguais para o atributo *id* de diferentes visões.

Os relacionamentos entre as visões são denominados arestas – que representam as conexões entre os nós. As visões estão relacionadas somente por meio do uso de valores em um ou mais atributos da visão de origem.

Os atributos disponíveis em uma visão também são denominados como campos na documentação. Os campos são idênticos aos atributos, pois contém um nome e um valor, que pode ser simples ou composto. Por exemplo, para um determinado Referenciado, o atributo nome possui o valor Albert Einstein, sendo do tipo simples. Já um atributo que permite a coleta dos amigos do Referenciado possui como valor uma lista de *ids*, sendo que cada *id* representa um amigo, sendo do tipo composto.

As requisições são realizadas através do uso do protocolo HTTPS, pela URL *https://api.linkedin.com*. Para coletar dados de uma visão é necessário o uso de processo de transformação dos nomes de visões como parte dos parâmetros da requisição de coleta de dados, na forma de parâmetros do HTTPS (método GET) (LinkedIn Corporation, 2023f).

A requisição é composta pelos seguintes elementos:

- a) URL da API;
- b) Separador, representado pelo símbolo de barra;
- c) Versão da API, em formato texto, representado pelo símbolo *v* e o número da versão de revisão maior, sem símbolo para separação;
- d) Separador, representado pelo símbolo de barra;
- e) Nome do método disponível para a visão;
- f) Separador, representado pelo símbolo de barra;
- g) Parâmetro obrigatório, com o valor contendo a chave de identificação do atributo *id* para a visão;
- h) Separador, representado pelo símbolo de interrogação;
- i) Lista de parâmetros.

É importante observar que algumas requisições não possuem parâmetros obrigatórios, devido ao contexto da própria requisição ou por ser de uso exclusivo em conjuntos de dados do próprio coletor.

Por exemplo, para acessar os conjuntos de dados do Referenciado de nome Fernando de Assis Rodrigues (visão *Member*), é necessário enviar pelo método GET do protocolo HTTPS: i) a versão da API (v1), ii) *people*, sendo o apelido do método para acesso à visão de membros, utilizada para a exibição de informações de Referenciados de forma individual, e; a chave de identificação do Referenciado na visão *Member* pelo parâmetro obrigatório (Exemplo 10).

Exemplo 16 – *LinkedIn* REST API: Requisição para coleta de conjuntos de dados, via método GET.

```
GET api.linkedin.com/v1/people/nmTVOeVYDE
```

Fonte: Elaborado pelo Autor.

Os caracteres nmTVOeVYDE representa o valor do atributo *id*, enviado na requisição pelo parâmetro de entrada *id*, sendo que este atributo é o identificador que representa unicamente este Referenciado na visão *Members*.

Os resultados das requisições da *LinkedIn Graph* API são apresentados no formato de notação da linguagem de marcação XML. No exemplo da requisição solicitada de conjunto de dados do Referenciado com o *id* de número nmTVOeVYDE, a resposta da requisição é apresentada conforme o Exemplo 17.

Exemplo 17 – *LinkedIn* REST API: Resultado da requisição para coleta de conjuntos de dados em XML, via método GET

```
<?xml version="1.0" encoding="UTF-8"?>
<person>
  <id>nmTVOeVYDE</id>
  <first-name>Fernando</first-name>
  <last-name>de Assis Rodrigues</last-name>
  <headline>Bio and more: https://amazondatatech.com.br</headline>
  <site-standard-profile-request>
    <url>https://www.linkedin.com/profile/view?id=AAoAAAcdz_gBB4zxRNjXQNmjfu1loC2-
dKLVfoI&authType=name&authToken=24SV&trk=api*a3227641*s3301901*</url>
  </site-standard-profile-request>
</person>
```

Fonte: Elaborado pelo Autor.

A primeira linha representa o cabeçalho obrigatório para o formato de notação da linguagem de marcação XML. Cada marcação – formada por palavras entre os símbolos de < (menor que) e > (maior que) – apresenta um atributo da visão, e os valores são apresentados pelo conteúdo disponível entre marcações de palavras iguais. Por exemplo, a marcação *first-name* (quarta linha da resposta da requisição) representa o atributo de nome *first-name* e tem como valor a palavra Fernando.

A *LinkedIn* API também disponibiliza os conjuntos de dados em formato de notação da linguagem de marcação JSON. É necessário acrescentar o parâmetro *format* na requisição para coletar os conjuntos de dados neste formato de notação, preenchendo o parâmetro com o valor textual *json* (Exemplo 18).

Exemplo 18 – *LinkedIn* REST API: Requisição para coleta de conjuntos de dados, em formato JSON, via método GET

```
GET api.linkedin.com/v1/people/nmTVOeVYDE?format=json
```

Fonte: Elaborado pelo Autor.

No exemplo da requisição de coleta dos conjuntos de dados do membro com o *id* contendo o valor nmTVOeVYDE, a resposta da requisição é apresentada no formato de notação da linguagem de marcação JSON, conforme o Exemplo 19.

Exemplo 19 – *LinkedIn* REST API: Resultado da requisição para coleta de conjuntos de dados em JSON, via método GET

```
{
  "firstName": "Fernando",
  "headline": "Bio and more: https://amazondatatech.com.br",
  "id": "nmTVOeVYDE",
  "lastName": "de Assis Rodrigues",
  "siteStandardProfileRequest": {
    "url": "https://www.linkedin.com/profile/view?id=AAoAAAcdz_gBB4zxRNjXQNmjfu1loC2-dKLVfoI&authType=name&authToken=24SV&trk=api*a3227641*s3301901*"
  }
}
```

Fonte: Elaborado pelo Autor.

Os caracteres chaves { (da primeira linha) e } (da última linha) indicam, respectivamente, o início e o final do registro. Tanto os nomes dos atributos como os seus respectivos valores são apresentados dentro destas duas chaves.

Cada atributo é representado da seguinte forma: o nome do atributo entre aspas duplas, seguido do símbolo de dois pontos (separador entre nome do atributo e valor) e do valor do atributo (que pode ter ou não aspas duplas no início e no final de seu valor, dependendo do tipo de dado). Os atributos de uma mesma visão são separados entre si pelo símbolo de vírgula.

No Exemplo 19, a solicitação retornou um registro da visão *Members*, a partir do uso como parâmetro *user_id* para o atributo *id* com o valor nmTVOeVYDE. O resultado apresentado para a coleta de da-

dos contém cinco atributos (*firstName*, *headline*, *id*, *lastName* e *siteStandardProfileRequest*), com os respectivos valores: Fernando; *Bio and more*: <https://amazondatatatech.com.br>; nmTVOeVYDE, de Assis Rodrigues, e um valor composto para o atributo *siteStandardProfileRequest*. Este último atributo possui como valor composto por um conjunto de atributos (apresentados entre duas chaves). No exemplo, o valor composto possui apenas um atributo, de título *url*, com o valor formado pela URL https://www.linkedin.com/profile/view?id=AAoAAAcdz_gBB-4zxRNjXQNmjfu1loC2-dKLVfoI&authType=name&authToken=24S-V&trk=api*a3227641*s3301901*.

Os atributos supramencionados são retornados automaticamente, pois são parte integrante do conjunto de atributos marcados como Padrão (no idioma inglês, *Default*) para esta visão. Caso a coleta de dados necessite de mais atributos da visão *Members*, é preciso utilizar um parâmetro especial, contendo como valor os nomes dos atributos desejados na coleta de dados entre símbolos de parênteses. Na requisição, este bloco de atributos deve ser inserido após o valor do parâmetro *id* para o método, separado pelo símbolo de dois pontos, conforme ilustra o Exemplo 20.

Exemplo 20 – *LinkedIn* REST API: Requisição para coleta de conjuntos de dados, com parâmetros adicionais, via método GET

```
GET api.linkedin.com/v1/people/nmTVOeVYDE:(id,num-connections,picture-url)?format=json
```

Fonte: Elaborado pelo Autor.

No Exemplo 20, a resposta da requisição de dados de um Referenciado diferencia-se das demais por causa dos atributos que estarão disponíveis no momento da coleta: *id*, *num-connections* e *picture-url*.

Caso seja necessário a adição de mais de um parâmetro na requisição, deve-se utilizar um separador entre os parâmetros no final da requisição, representado pelo símbolo &.

A *LinkedIn* REST API não permite a realização de requisições contendo conjuntos de dados de outras visões, a partir dos relacionamentos entre visões disponíveis. Para o acesso a estes conjuntos de dados, deve-se verificar a documentação de referência das visões de destino. Ou seja, ao receber um atributo composto, pode ser necessário a execução de um processo manual de consulta as visões relacionadas. Apesar de não ser um processo automático, a *LinkedIn* REST API relaciona valores de atributos compostos das visões de origem com estas visões. A única restrição técnica é que o Agente Externo deve pré-carregar em seu aplicativo os valores dos atributos das visões relacionadas, com as suas devidas cardinalidades explicitadas.

As respostas das requisições também podem ser segmentadas por um sistema de paginação, que utiliza os parâmetros *start* e *count* para realizar a iteração, pois a *LinkedIn* REST API não permite a coleta de todas os registros de uma visão em uma única requisição.

Para realização de coleta de dados sobre todas as postagens relacionadas de uma companhia, terão de ser realizadas coletas em blocos, aonde cada requisição permitirá a coleta de conjuntos de dados sobre 10 postagens. A quantidade de registros existentes em um bloco deve ser igual ao valor informado no parâmetro *count*.

O parâmetro *start* será preenchido com o valor da primeira linha do bloco. Ou seja, na primeira requisição o valor do parâmetro será 1, no segundo 11 (pois já estariam coletadas as dez primeiras postagens), no terceiro 21 (pois já estariam coletadas as vinte primeiras postagens) e assim por diante.

O sistema de descoberta de conteúdo só está disponível para acesso pelo uso de contas prêmio. A documentação técnica do sistema de descoberta não está acessível as demais contas e é parte de outra API do serviço.

As requisições da *LinkedIn* REST API também possuem um sistema para o tratamento de erros de processamento ou de autorização. Por exemplo, no caso de uma requisição não ser permitida ao *token* de autorização de acesso, o resultado da requisição (Exemplo 21) será uma estrutura em

formato de notação das linguagens de marcação JSON ou XML com atributos específicos para esta atividade.

Exemplo 21 – *LinkedIn* REST API: Tratamento de erros de processamento ou de autorização

```
{
  "errorCode": 0,
  "message": "Member does not have permission to get company.",
  "requestId": "KTU26IQZXT",
  "status": 403,
  "timestamp": 1484327071912
}
```

Fonte: Elaborado pelo Autor.

Portanto, caso uma requisição apresente algum erro, a *LinkedIn* REST API retorna um conjunto de atributos:

- a) *errorCode*: com o código do erro. Não há lista contendo as possibilidades de erro nos documentos da API;
- b) *message*: contendo a descrição do erro;
- c) *requestId*: com identificador único para representar o coletor;
- d) *status*: com o código HTTP, quando relacionado aos erros de processamento do servidor;
- e) *timestamp*: data e horário do recebimento da requisição, no formato *UNIX epoch*.

Na área Conceitos da API (LinkedIn Corporation, 2023f), encontra-se o conjunto de documentos contendo as descrições de funcionalidades da *LinkedIn* REST API. A seção apresenta a descrição de 21 visões disponíveis para a coleta de dados, com um total de 20 visões possuindo documento próprio, contendo as seguintes características:

- a) URL do documento: cada um dos documentos possui um endereço eletrônico para acesso individualizado de suas informações, concomitante ao formato URL;
- b) Título: elemento textual contendo o título da visão. Não há ocorrências da *LinkedIn* REST API de títulos iguais para visões (homônimos);
- c) Descrição da Visão (elemento não obrigatório): elemento textual, na forma de um ou mais parágrafos, descrevendo a visão;
- d) Descrição da Permissão de Acesso (elemento não obrigatório): elemento textual, na forma de um ou mais parágrafos, descrevendo as condições de acesso para coleta de dados da visão;
- e) Guia de Campos (elemento não obrigatório): área contendo um quadro informativo, com duas colunas (campo, descrição), com informações sobre os atributos disponíveis na visão: o nome e uma descrição para os atributos disponíveis na visão no momento da coleta de dados;
- f) Guia de Valores (elemento não obrigatório): área contendo um quadro informativo, com três colunas com títulos variáveis, contendo informações sobre os atributos disponíveis na visão: o nome dos atributos e a lista de valores válidos para o uso dos atributos tanto disponíveis na visão como também no momento da coleta de dados.

A visão *Share* não possui um documento de referência, porém os campos disponíveis aparecem, na forma de exemplos, em dois locais: a) no tutorial de compartilhamento de conteúdo no serviço, e; b) no tutorial de gerenciamento de páginas de companhias (LinkedIn Corporation, 2023g, 2023h).

As informações sobre ações de criação, atualização e exclusão de dados não aparecem nos documentos, pois são tratadas de maneira separada das ações de coleta de dados.

Foram identificadas 21 visões vinculadas com a *LinkedIn* REST API, sendo duas visões focadas em classes de conteúdo consideradas principais para o *LinkedIn* – as visões *Member* e *Company* – e 19 visões voltadas a auxiliar a consulta aos demais tipos de conteúdo. Dentre as características identificadas, destacam-se que:

- a) Somente as visões *Geography Codes* e *Industry Codes* possuem descrição;
- b) As referências sobre a visão *Member* estão divididas em três documentos: um principal, contendo informações básicas sobre a visão, e dois documentos auxiliares com acesso a informações sobre os atributos disponíveis e sobre informações de contato;
- c) Os documentos das visões *Company Size Codes*, *Currency Codes*, *Geography Codes*, *Industry Codes*, *Job Function Codes*, *Language Codes* e *Seniority Codes* são parte de um agrupamento de documentos tratados como referência pelo *LinkedIn*. Os documentos deste agrupamento têm como foco tornar disponível informações sobre os valores permitidos para os atributos destas visões – e estas farão papel de visões auxiliares, sem ponto de entrada automatizado, relacionadas com as visões *Member* e *Company*.

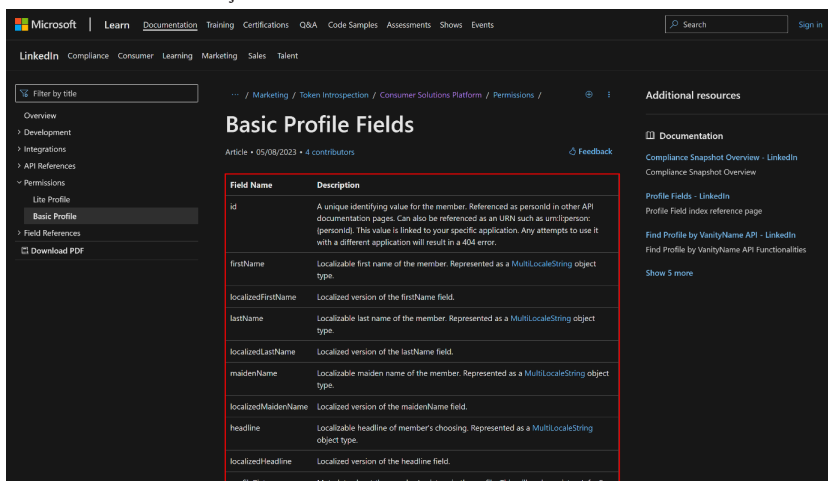
Foram encontradas, na documentação, as referências dos tipos de dados de duas formas: a) através de informações contidas na subseção Guia de Campos, na coluna descrição do quadro informativo, e; b) pela interpretação da lista de valores válidos, em informações contidas na subseção Guia de Valores.

Também na Guia de Campos e na Guia de Valores são encontradas as informações sobre os atributos e tipos de dados disponíveis no momento da coleta de dados.

Estas informações são disponibilizadas no formato HTML, no próprio corpo documental, e possuem quatro formas de apresentação: três quadros informativos no formato de tabela HTML e uma com o uso da marcação de bloco de citação do HTML.

Na primeira forma de apresentação (Figura 19), o quadro informativo contendo a descrição dos atributos disponíveis na coleta de dados possui duas colunas (ênfatisadas por um retângulo com a borda na cor vermelha): a) *Field Name*, contendo o nome do atributo, e; b) *Description*, com a descrição do atributo.

Figura 19 – *LinkedIn* REST API: Atributos na primeira forma de apresentação, tabela HTML com duas colunas



Field Name	Description
id	A unique identifying value for the member. Referenced as personId in other API documentation pages. Can also be referenced as an URN such as urn:li:person:(personId). This value is linked to your specific application. Any attempts to use it with a different application will result in a 404 error.
firstName	Localizable first name of the member. Represented as a MultiLocaleString object type.
localizedFirstName	Localized version of the firstName field.
lastName	Localizable last name of the member. Represented as a MultiLocaleString object type.
localizedLastName	Localized version of the lastName field.
maidenName	Localizable maiden name of the member. Represented as a MultiLocaleString object type.
localizedMaidenName	Localized version of the maidenName field.
headline	Localizable headline of member's choosing. Represented as a MultiLocaleString object type.
localizedHeadline	Localized version of the headline field.
profilePicture	Metadata about the member's picture in the profile. This will resolve pictureInfo. See

Fonte: Recorte de *LinkedIn Corporation* (2023d), elaborado pelo Autor.

Um total de 12 visões utilizam esta forma de apresentação para exibir as informações de seus atributos: *Certification*, *Company*, *Course*, *Education*, *Language*, *Location*, *Member*, *Patent*, *Position*, *Publication*, *Recommendation* e *Skill*.

A segunda forma de apresentação (exemplificada na Figura 20) também utiliza um quadro informativo para a descrição das características seus atributos (ênfatisados por um retângulo com a borda na cor vermelha). Todavia, os títulos das colunas do quadro representam os nomes dos atributos disponíveis nestas visões sem explicitar suas características e as linhas do quadro representam os valores permitidos para cada um dos atributos.

Figura 20 – *LinkedIn* REST API: Atributos na segunda forma de apresentação, tabela HTML com duas colunas

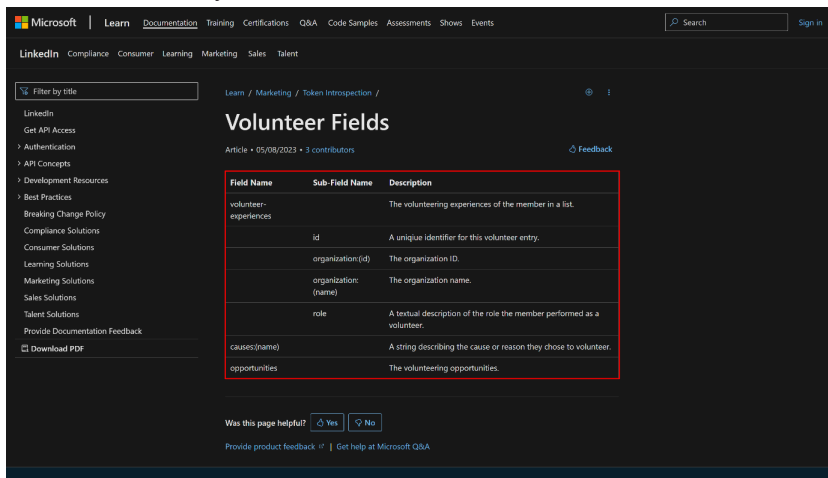
Code / ID	Description
acct	Accounting / Auditing
adm	Administrative
advr	Advertising
anls	Analyst
art	Art / Creative
bd	Business Development
cnd	Consulting
cust	Customer Service
dist	Distribution
dsgn	Design
edu	Education
eng	Engineering
fin	Finance
hrsh	Human Resources

Fonte: Recorte de *LinkedIn Corporation* (2023i), elaborado pelo Autor.

Esta variante de apresentação está vinculada com sete visões – *CompanySizeCodes*, *CurrencyCodes*, *GeographyCodes*, *IndustryCodes*, *JobFunctionCodes*, *LanguageCodes* e *SeniorityCodes*. São visões que apoiam outros conteúdos do serviço, como os conteúdos das visões relacionadas com os membros e as companhias.

A visão *Volunteer* é a única visão relacionada com a terceira apresentação (Figura 21), que também utiliza um quadro informativo para a descrição de seus atributos (ênfatisados por um retângulo com a borda na cor vermelha).

Figura 21 – *LinkedIn* REST API: Atributos na segunda forma de apresentação, tabela HTML com três colunas



Fonte: Recorte de *LinkedIn Corporation* (2023j), elaborado pelo Autor.

Nesta forma de apresentação, o quadro informativo está dividido em três colunas, apresentando: a) *Field Name*, contendo o nome do atributo; b) *Sub-Field Name*, os nomes dos atributos, no caso de serem parte integrante de um valor composto, e; c) *Description* com a descrição das informações contidas no atributo.

A quarta forma de apresentação, com o uso da marcação de bloco de citação do HTML (Figura 22), exhibe as informações sobre os atributos a partir de: a) um quadro informativo para a descrição de seus atributos, e b) na forma de um exemplo de resposta de requisição a consultas de conjuntos de dados da visão (via método GET). Os exemplos estão inseridos dentro da marcação *pre*¹⁶ do HTML e seu texto está estruturado na forma de conjunto de dados no formato de notação da linguagem de marcação JSON. O quadro está dividido em três colunas: a) *Field Name*, contendo o nome do atributo; b) *Type*, contendo o tipo de dado esperado para o atributo, e; c) *Description*, com a descrição das informações contidas no atributo.

¹⁶ Ver World Wide Web Consortium (2023).

Figura 22 – *LinkedIn* REST API: Atributos na quarta forma de apresentação, com o uso da marcação de bloco de citação do HTML

Field Name	Type	Description
activity	Activity URN	URN of the activity associated with this share. Activities act as a wrapper around shares and articles to represent content in the LinkedIn feed. Read only.
agent	URN	Agent is the Sponsored Ad Account that created the Direct Sponsored Content (DSC) Share on behalf of an organization.
content	Share Content	Referenced content such as articles and images.
created	Audit Stamp	Time of creation. Read only.
distribution, linkedInDistributionTarget	Share Distribution Target	Distribution target for the share.
edited	boolean	A flag that indicates if this share was edited by a member. Read only.
id	string	Unique ID for the share. Read only.
lastModified	Audit Stamp	Time of last modification. Read only.
originalShare	Share URN	If this share is a reshare, then this is the URN of the original/root share that was reshared.
owner	Person or Organization	Owner of the share. Required on create.

Sample Response
<pre> JSON { "activity": "urn:li:activity:12345657", "content": { "contentEntities": [{ "entity": "urn:li:article:", "entityLocation": "https://www.example.com/content.html", "thumbnails": [{ "imageSpecificContent": {}, "resolvedUrl": "https://www.example.com/image.jpg" }] }] }, "description": "content description", "title": "Test Share with Content" }, "created": { "actor": "urn:li:person:Abx03Qc10", "time": 1471907230000 }, "distribution": { "linkedInDistributionTarget": {} }, }, {id": "61738780592864350", "lastModified": { "actor": "urn:li:person:Abx03Qc10", "time": 1471907237000 }, "owner": "urn:li:organization:123456789", "text": { "text": "Test Share!" } } </pre>

Fonte: Colagem de recortes a partir de *LinkedIn Corporation* (2023k), elaborado pelo Autor.

A referência dos elementos descritivos para os atributos também pode apresentar alguns qualificadores – marcações especiais que agregam características específicas aos atributos.

Os qualificadores identificados podem ser do tipo:

- a) *Default*: o atributo marcado com este qualificador ficará disponível automaticamente na resposta da requisição, ou seja, na requisição de coleta de dados de uma visão não haverá necessidade de explicitar a API retorno do valor do atributo;
- b) *Deprecated*: o atributo marcado com este qualificador foi considerado obsoleto pela equipe de desenvolvimento da API e, caso disponível na coleta de dados, o seu valor será nulo na resposta da requisição.

Os qualificadores não possuem uma área própria para sua descrição. Em todos os casos os qualificadores foram identificados nos elementos textuais de descrição dos atributos.

Os valores dos atributos podem ser simples – quando o valor é um número, signos, data e outros formatos – ou composto. No caso os atributos compostos, podem variar entre:

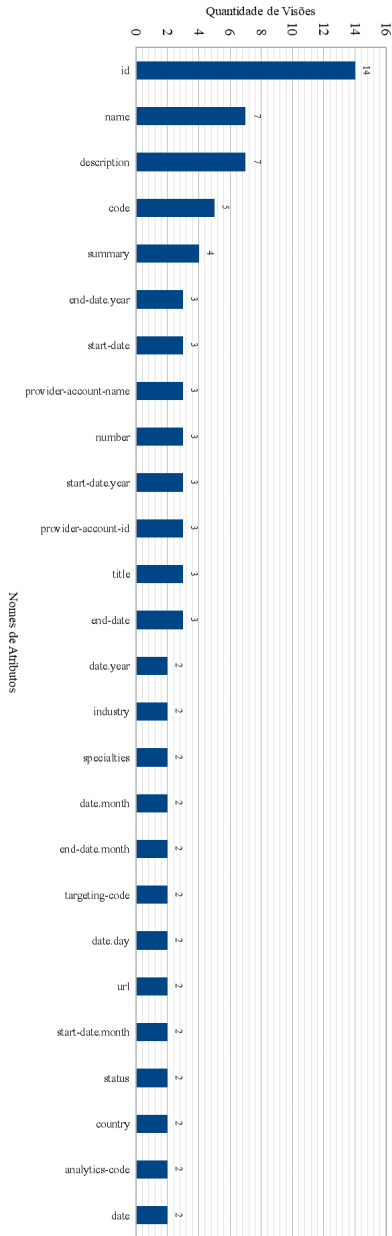
- a) Vetores: quando o valor do atributo é um conjunto de valores, representado com o uso de vetores computacionais (*arrays*). Os atributos com este tipo de valor apresentam no seu tipo de dado o sufixo [];
- b) Listas: quando o valor do atributo é um conjunto de valores com tipo de dados pré-fixados, com o uso de listas computacionais (*list*). Os atributos com este tipo de valor apresentam no seu tipo de dado o prefixo *list*;
- c) Visões: quando o valor do atributo é um registro de outra visão. Os atributos com este tipo de valor apresentam como tipo de dado o nome de uma Visão.

Foram identificados 231 atributos, sendo que um total de 176 atributos possuem valor do tipo simples, 54 atributos possuem valor do tipo composto e um atributo que não foi possível realizar esta identificação.

Com relação aos qualificadores, dois atributos são retornados automaticamente pelas requisições no momento da coleta de dados, dois atributos precisam ser explicitados na requisição para a coleta de dados e 227 atributos que não foi possível realizar esta identificação. Foi encontrado um atributo marcado como obsoleto e em 230 atributos não foi possível realizar a identificação.

Foram encontrados 170 nomes distintos para atributos da *LinkedIn* REST API, porém um mesmo nome de atributo pode aparecer em visões distintas.

Gráfico 5 – *LinkedIn* REST API: Ocorrências de nomes de atributos iguais em diferentes visões



Fonte: Elaborado pelo Autor.

O Gráfico 5 exibe as ocorrências de nomes de atributos que foram utilizados em duas ou mais visões. Destacam-se que um nome é utilizado em 10 ou mais visões (*id*), três nomes são utilizados entre cinco e nove visões (*code*, *description* e *name*), 22 nomes são utilizados entre duas e quatro visões (*analytics-code*, *country*, *date*, *date.day*, *date.month*, *date.year*, *end-date*, *end-date.month*, *end-date.year*, *industry*, *number*, *provider-account-id*, *provider-account-name*, *specialties*, *start-date*, *start-date.month*, *start-date.year*, *status*, *summary*, *targeting-code*, *title* e *url*). Um total de 144 nomes de atributos são utilizados em apenas uma visão da API.

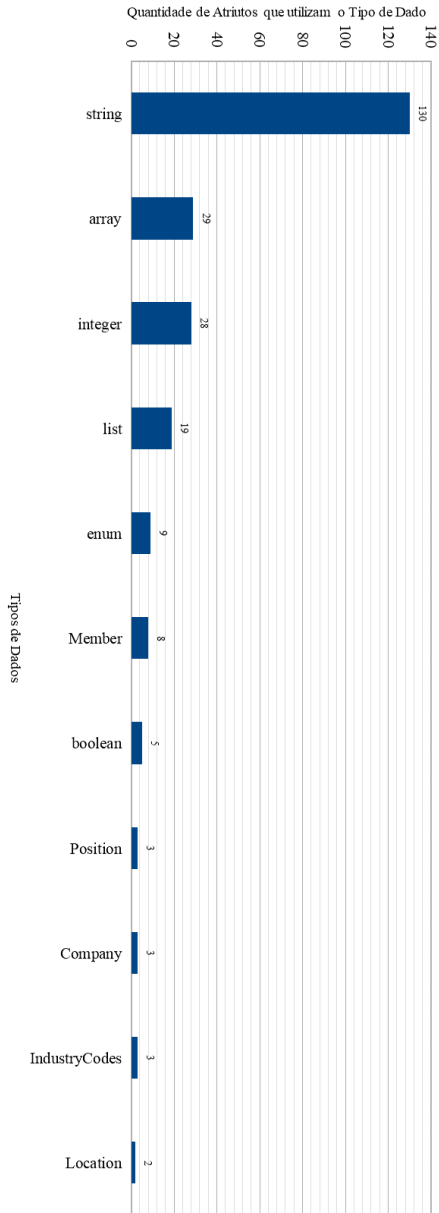
A *LinkedIn* REST API contém 22 tipos de dados, que podem ser classificados segundo suas características, em três categorias:

- a) Tipos de dados de Sistema de Gerenciamento de Bancos de Dados: são encontrados na maioria dos Sistemas de Gerenciamento de Bancos de Dados, como números inteiros, *string* para caracteres, datas, textos longos, booleanos, entre outros;
- b) Tipos de dados de algoritmos de Linguagens de Programação: são tipos de dados encontrados em linguagens de programação e também próprios do contexto da API;
- c) Tipos de dados para relacionar conteúdos de duas visões: são tipos de dados para atributos que apresentam como valor um subconjunto de atributos originários de uma outra visão no momento da coleta.

Um mesmo tipo de dado pode ter até duas características. Por exemplo, o tipo de dado *list<Education>* é uma lista (tipo de dado específico) que apresenta subconjuntos da visão *Education (URLs)* (tipo de dado que apresenta conteúdo de uma visão) – mesclando duas categorias.

Os tipos de dados precisam ser concomitantes com a estrutura identificadas nos atributos e, portanto, seus valores podem ser simples – quando o valor é um número, símbolos, textos ou datas – ou compostos – quando o valor é composto por um subconjunto de atributos.

Gráfico 6 – *LinkedIn* REST API: Ocorrências de tipos de dados em atributos



Fonte: Elaborado pelo Autor.

O Gráfico 6 exibe as ocorrências de tipos de dados que foram utilizados em dois ou mais atributos. Com relação a ocorrências destes tipos de dados nos atributos, foram identificados:

- a) Quatro tipos de dados utilizados em mais de 10 atributos, sendo dois simples e dois compostos: *string* (simples), *array* (composto), *integer* (simples) e *list* (composto);
- b) Três de tipos de dados utilizados entre cinco a nove atributos, sendo dois simples e um composto: *enum* (simples), *Member* (composto) e *boolean* (simples);
- c) Quatro tipos de dados utilizados entre dois e quatro atributos, sendo todos compostos: *Position*, *Company*, *Industry Codes* e *Location*.

Um total de doze tipos de dados são utilizados em apenas um atributo, sendo um do simples e onze compostos: *Certification* (composto), *Course* (composto), *date* (simples), *Education* (composto), *Language* (composto), *Patent* (composto), *Publication* (composto), *Recommendation* (composto), *Share* (composto), *Skill* (composto), *string[]* (composto e único caso de vetor) e *Volunteer* (composto).

Os relacionamentos entre visões seguem princípios similares ao de cardinalidade entre tabelas nos Sistemas de Gerenciamento de Bancos de Dados (Silberschatz; Korth; Sudarshan, 2020). Foram identificados o uso das cardinalidades: 1-para-N, quando um registro da visão de origem relaciona-se com uma ou mais registros da visão de destino, e; N-para-N, quando um ou mais registros da visão de origem relacionam-se com um ou mais registros da visão de destino. Totalizam-se 29 relacionamentos, sendo que 24 possuem a cardinalidade N-para-N e cinco a cardinalidade 1-para-N. Não foram identificadas relações utilizando a cardinalidade 1-para-1.

Os relacionamentos podem ser explicitados na documentação técnica das visões apenas por atributos: quando o tipo de dado de um atributo é uma visão, considera-se que o valor do atributo representa um relacio-

namento com outra visão, proporcionando que, ao coletar conjuntos de dados da visão de origem, os dados da visão de destino também podem ser coletados através deste atributo. Esta informação pode ser identificada de duas formas: i) no corpo do texto do documento de referência da visão, e ii) em informações do atributo *description*, localizada nos quadros informativos contendo as descrições de cada atributo (quando disponível).

As requisições disponíveis para o acesso às visões funcionam a partir do uso do protocolo HTTPS, método GET, tanto para a transmissão das informações das requisições, como para o recebimento dos conjuntos de dados coletados. Como as requisições estão diretamente vinculadas com as visões, a descrição das funcionalidades e a lista com parâmetros de consulta estão disponíveis no documento separado, intitulado Iniciando: REST API, acessível pela página inicial da área de documentação técnica, com *hyperlink* na página inicial, dentro da seção que contém informações sobre como utilizar Kits de Desenvolvimento de Aplicações e API.

Os métodos de requisição das visões podem ter nomenclaturas diferentes do nome da visão. Por exemplo, a requisição de conjuntos de dados de Referenciados, oriundos da visão *Members*, é acessível pela execução do método intitulado *people*.

Os documentos do tipo POST, que contém informações sobre os métodos de inserção, de atualização e de exclusão de conjuntos de dados, estão separados dos documentos de consulta aos dados (LinkedIn Corporation, 2023b, 2023c, 2023d, 2023e), fora do escopo de análise.

Das visões da *LinkedIn* REST API disponíveis para a coleta, um total de 19 visões não possuem pontos de entrada para consulta aos conjuntos de dados: *Certification, Company, Company Size Codes, Course, Currency Codes, Education, Geography Codes, Industry Codes, Job Function Codes, Language, Language Codes, Location, Member, Patent, Position, Publication, Recommendation, Seniority Codes, Share, Skill* e *Volunteer*. O acesso ao conjunto de dados destas visões deve ser realizado pelo processo de relacionamento por atributos de outras visões.

Nas duas requisições, foram identificados um total de seis parâmetros para a coleta. Todas as requisições disponíveis possuem parâmetros, e todos os parâmetros possuem descrição e tipo de dado esperado.

Foram encontrados quatro nomes distintos para os parâmetros disponíveis na *LinkedIn* REST API: *count*, *format*, *id* e *start*. É importante ressaltar que um nome de parâmetro pode ser repetido em métodos distintos. No caso, os parâmetros *format* e *id* aparecem nos dois métodos das requisições disponíveis.

Com relação as ocorrências do uso dos tipos de dado nos parâmetros, quatro parâmetros possuem o tipo de dado *string* e dois parâmetros possuem o tipo de dado *integer*.

Para acessar as requisições é necessário o uso de um conjunto de permissões. No caso da *LinkedIn* REST API, as permissões para a coleta de dados estão vinculadas ao uso do conjunto de duas estruturas de permissão de acesso: os *tokens* de autorização de acesso e as permissões.

Os *tokens* de autorização de acesso são um conjunto finito de símbolos, gerados pelo serviço, com a finalidade de providenciar uma contrassenha para verificação de credenciais. Estes *tokens* de autorização de acesso são um dos itens que compõem o processo de verificação inicial de credenciais – ou seja, o processo do sistema de entrada que permite que o acesso seja concedido e que habilite aplicações para a coleta de conjuntos de dados. São obtidos pelo uso do protocolo HTTPS, sendo possível sua implementação em diversas linguagens de programação. Após a sua criação, os Agentes Externos têm acesso a informações complementares, como o tempo de validade do *token* e o aplicativo que solicitou a sua geração (Linkedin Corporation, 2023¹). No *LinkedIn*, os *tokens* de autorização são parte de um sistema de entrada denominado *OAuth*¹⁷, desenvolvido com o propósito de auxiliar no processo de identificação digital (entra-

¹⁷ Nome dado em função de ser uma implementação de sistema baseada no protocolo de padrão aberto *OAuth*.

da) entre duas ou mais partes, dentro de aplicativos e *websites* (LinkedIn Corporation, 2023e, 2023l).

Já as permissões são autorizações de acesso a um aplicativo externo para coleta de conjuntos de dados de Referenciados, de companhias ou de visões relacionadas a todos estes entes. As permissões são concedidas no momento que um Referenciado conecta um aplicativo externo ou um código-fonte de *website* ao seu perfil, ou quando um Referenciado conecta um aplicativo externo ou um código-fonte de *website* com uma conta vinculada ao *LinkedIn* (LinkedIn Corporation, 2023l).

As informações sobre quais são os *tokens* de autorização de acesso existentes e quais são as permissões necessárias para acessar conjuntos de dados são encontradas em dois documentos: nos documentos contendo informações sobre o sistema de entrada *OAuth* e na documentação técnica das visões.

No caso dos *tokens* de autorização de acesso existentes, as informações encontram-se em um documento denominado *OAuth*, que integra o conjunto de documentos que descrevem como aplicações e códigos-fonte de *websites* podem conectarem por meio do sistema de entrada *OAuth* para autorização da interoperabilidade com os conteúdos disponíveis no serviço (LinkedIn Corporation, 2023d, 2023l).

Segundo o documento, existe somente um tipo de *token* de autorização de acesso: o *Access Token*. Este é o *token* de autorização de acesso que permite acesso pela *LinkedIn* REST API aos conjuntos de dados de Referenciados e companhias, e também autoriza a leitura, e a criação, a alteração e exclusão de conteúdo. Para a obtenção da autorização pelo sistema de entrada *OAuth*, é necessário realizar um cadastro prévio da aplicação ou do código-fonte na plataforma Minhas Aplicações, pela URL <https://www.linkedin.com/developers/apps>. Ao completar o cadastro, o *LinkedIn Corporation* autorizará a conexão a seus produtos e serviços (incluindo com as APIs disponíveis), e o Agente Externo receberá os seguintes dados (LinkedIn Corporation, 2023l):

- a) *oauth_consumer_key*: contém como valor a chave para a identificação de qual aplicação ou *website* realizou a requisição;
- b) *oauth_signature*: contém como valor a assinatura pública (chave pública) para que o *LinkedIn Corporation* reconheça o aplicativo, utilizando um algoritmo de verificação de assinaturas;
- c) *oauth_signature_method*: contém como valor o tipo de método de criptografia utilizado para a geração da chave pública.

Além disso, o Agente Externo deve informar em cada requisição os parâmetros:

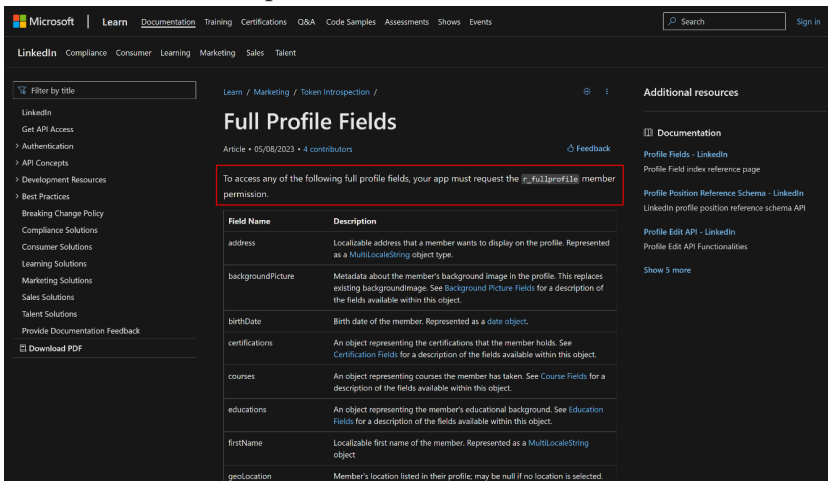
- a) *oauth_version*: contém como valor a versão do protocolo do sistema de entrada *OAuth* que será utilizado na requisição;
- b) *oauth_nonce*: contendo uma chave única de identificação para cada requisição solicitada;
- c) *oauth_timestamp*: com um valor que representa a data e o horário que a requisição foi iniciada, no formato *UNIX epoch*;
- d) *oauth_token*: com a assinatura pública do aplicativo (chave pública) para acesso aos conjuntos de dados de um Referenciado.

Na documentação técnica das visões está preestabelecido quais requisições serão aceitas para o acesso a dados de forma individual, para cada visão, no momento da conexão para iniciar o processo de coleta de dados pelos aplicativos externos. Portanto, as permissões de acesso aos conjuntos de dados de cada visão (e seus atributos) são elementos já estabelecidos pelo *LinkedIn Corporation* que delimitam nas visões os conjuntos de dados de Referenciados, de companhias e de visões relacionadas a estes entes, que estarão acessíveis para a coleta por aplicações externas.

O conjunto de documentos contendo as descrições de funcionalidades das visões do *LinkedIn* REST API podem apresentar um elemento

textual, na forma de um ou mais parágrafos, descrevendo as condições de acesso para coleta de dados da visão (Figura 23, enfatizado por um retângulo de bordas vermelhas), não obrigatório, onde são explicitadas as permissões que precisam ser previamente autorizadas pelos usuários para que a coleta de dados possa ser realizada por aplicativos externos.

Figura 23 – *LinkedIn* REST API: Parágrafo contendo informações sobre as permissões de acesso a visão



Fonte: Recorte de *LinkedIn Corporation* (2023m), elaborado pelo Autor.

Por exemplo, a Figura 23 exibe a descrição das permissões para o acesso à visão *Members*, para a coleta dos conjuntos de dados dos atributos considerados básicos, aonde o acesso às operações de coleta de dados é autorizado com o uso de um *token* de autorização de acesso válido, com prévia autorização do usuário da permissão *r_fullprofile*.

O Quadro 3 apresenta a síntese das informações identificadas sobre as permissões, contendo: a) o nome da permissão; b) a descrição, contendo o objetivo da permissão, quando concedida; c) as visões que ficarão disponíveis ao aplicativo externo, e; d) os atributos das visões que ficarão disponíveis ao aplicativo externo.

Estruturas de dados em serviços de redes sociais online
 Uma abordagem metodológica de análise

Quadro 3 – Referências de permissões da *LinkedIn* REST API, versão 1.0

Nome da Permissão	Descrição da Permissão	Visões relacionadas	Atributos Relacionados
<i>r_basicprofile</i>	Acesso ao perfil básico do membro.	<i>Member</i>	<i>api-standard-profile-request</i>
			<i>current-share</i>
			<i>first-name</i>
			<i>formatted-name</i>
			<i>formatted-phonetic-name</i>
			<i>headline</i>
			<i>id</i>
			<i>industry</i>
			<i>last-name</i>
			<i>location</i>
			<i>maiden-name</i>
			<i>num-connections</i>
			<i>num-connections-capped</i>
			<i>phonetic-first-name</i>
			<i>phonetic-last-name</i>
			<i>picture-url</i>
			<i>picture-urls:::(original)</i>
			<i>positions</i>
<i>public-profile-url</i>			
<i>site-standard-profile-request</i>			
<i>specialties</i>			
<i>summary</i>			

Nome da Permissão	Descrição da Permissão	Visões relacionadas	Atributos Relacionados
<i>r_fullprofile</i>	Acesso ao perfil completo do membro.	<i>Member</i>	<i>api-standard-profile-request</i> <i>associations</i> <i>certifications</i> <i>courses</i> <i>current-share</i> <i>date-of-birth</i> <i>educations</i> <i>first-name</i> <i>following</i> <i>formatted-name</i> <i>formatted-phonetic-name</i> <i>headline</i> <i>honors-awards</i> <i>id</i> <i>industry</i> <i>interests</i> <i>job-bookmarks</i> <i>languages</i> <i>last-modified-timestamp</i> <i>last-name</i> <i>location</i> <i>maiden-name</i> <i>member-url-resources</i> <i>num-connections</i> <i>num-connections-capped</i> <i>num-recommenders</i> <i>patents</i> <i>phonetic-first-name</i> <i>phonetic-last-name</i> <i>picture-url</i> <i>picture-urls::(original)</i> <i>positions</i> <i>proposal-comments</i> <i>public-profile-url</i> <i>publications</i> <i>recommendations-received</i> <i>related-profile-views</i> <i>site-standard-profile-request</i> <i>skills</i> <i>specialties</i> <i>suggestions</i> <i>summary</i> <i>three-current-positions</i> <i>three-past-positions</i> <i>volunteer</i>
<i>r_emailaddress</i>	Acesso ao e-mail do membro.	<i>Member</i>	<i>email-address</i>

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

Nome da Permissão	Descrição da Permissão	Visões relacionadas	Atributos Relacionados
<i>r_contactinfo</i>	Acesso aos contatos do membro.	<i>Member</i>	<i>bound-account-types</i> <i>bound-account-types.account-type</i> <i>bound-account-types.binding-status</i> <i>bound-account-types.id</i> <i>bound-account-types.is-primary</i> <i>bound-account-types.provider-account-id</i> <i>bound-account-types.provider-account-name</i> <i>im-accounts</i> <i>im-accounts.im-account-name</i> <i>im-accounts.im-account-type</i> <i>main-address</i> <i>phone-numbers</i> <i>phone-numbers.phone-number</i> <i>phone-numbers.phone-type</i> <i>primary-twitter-account</i> <i>primary-twitter-account.provider-account-id</i> <i>primary-twitter-account.provider-account-name</i> <i>twitter-accounts</i> <i>twitter-accounts.provider-account-id</i> <i>twitter-accounts.provider-account-name</i>
<i>rw_company_admin</i>	Acesso ao perfil da companhia.	<i>Company</i>	Todos os atributos da visão.

Fonte: Elaborado pelo Autor.

A permissão *w_share* está vinculada com a manipulação de conjuntos de dados visões da *LinkedIn* REST API, e foi considerada fora de escopo.

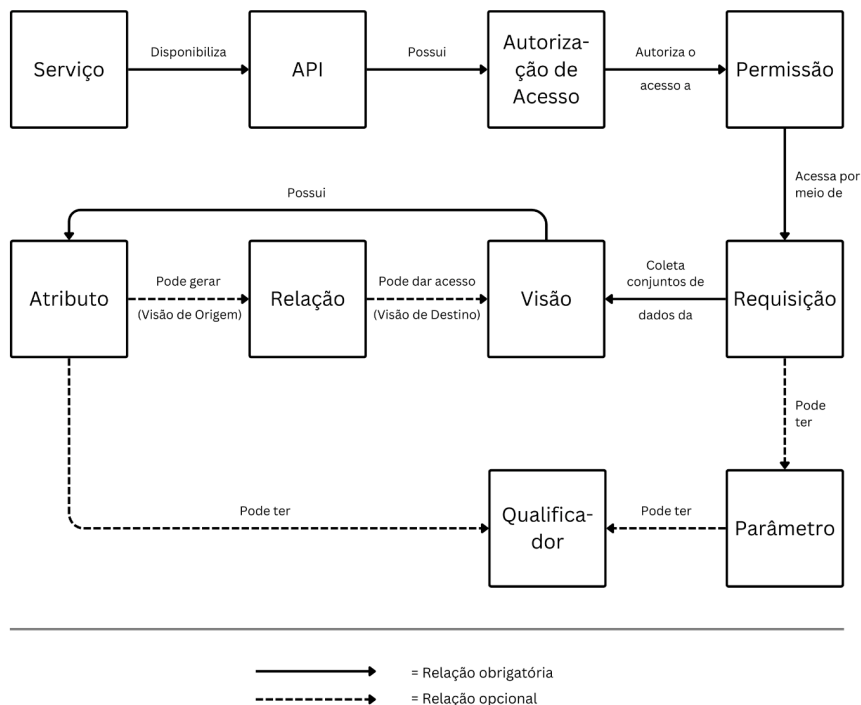
A partir das características da *X* REST API, foram identificadas as seguintes entidades para a sistematização da coleta de dados:

- a) Serviço, com informações referentes ao Serviço de Rede Social *Online*: nome do serviço e a URL de acesso;
- b) API, com informações referentes a API: nome da API, versão, descrição e URL para acesso as requisições;

- c) Visão, contendo informações sobre as visões disponíveis: nome da visão, descrição e URL para acesso aos documentos;
- d) Atributos de cada visão, com informações dos atributos disponíveis nos resultados das requisições: nome do atributo, descrição, tipo de dado e qualificadores;
- e) Qualificadores, com informações sobre as características das marcações que diferenciam parte dos atributos das visões, e parte dos parâmetros da requisição: nome do qualificador e descrição;
- f) Requisições existentes em cada visão, contendo informações sobre quais operações de coleta de dados estão disponíveis para a visão: nome da requisição, tipo e nome do método e seus parâmetros de entrada e saída;
- g) Parâmetros existentes nas requisições, em um relacionamento que uma requisição pode ter nenhum, um ou mais parâmetros: nome do parâmetro, descrição e qualificadores;
- h) Relações, com informações sobre os relacionamentos entre as visões: nome da visão de origem, nome da visão de destino, tipo de relacionamento, nome do relacionamento, cardinalidade na origem e cardinalidade no destino;
- i) Autorizações de acesso, com informações sobre os tipos de acesso: nome, descrição e URL para acesso aos documentos;
- j) Permissões, definindo as informações sobre os tipos de permissões existentes e suas relações com cada visão: nome, descrição, visões e atributos relacionados, e URL para acesso aos documentos.

A Figura 24 apresenta uma representação gráfica das relações entre as entidades propostas. O serviço apresenta formas de acesso aos seus conjuntos de dados por meio da API, que possui um sistema de entrada *OAuth* baseado em *tokens* (com a função de autorizar o acesso) e permissões (que estabelecem quais conjuntos de dados podem ser coletados).

Figura 24 – *LinkedIn* REST API: Vínculos entre as entidades



Fonte: Elaborado pelo Autor.

Por meio de pontos de entrada disponíveis, o sistema de autorização permite que seja realizada a execução de requisições para coletar dados, permitindo o acesso aos dados contidos nas visões. No momento do disparo da execução das requisições, podem existir parâmetros de entrada de dados, e estes parâmetros podem ter qualificadores – marcações que estabelecem características especiais a certos parâmetros – como o estado de obrigatoriedade de uso do parâmetro na requisição.

As visões possuem: atributos, com tipologia de dados própria, e que também podem apresentar qualificadores (marcações que estabelecem características especiais a certos parâmetros); e relações com outras visões

da API – que são estabelecidas somente a partir de valores compostos de atributos, que apresentam como tipo de dado a visão de destino.

2

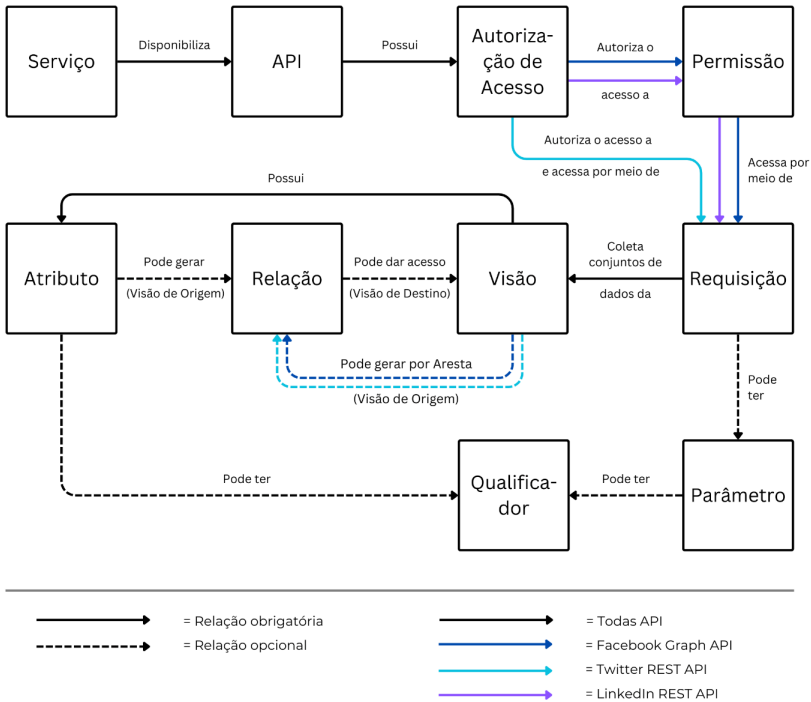
MODELAGEM
DIRETA

MODELAGEM DIRETA

A partir das características identificadas nas APIs dos Serviços de Redes Sociais *Online Facebook*, *X* e *LinkedIn*, foi possível perceber similaridades no processo de coleta de dados entre as APIs, bem como delimitar características inerentes as APIs, as visões, os atributos, as requisições, os parâmetros, demais elementos que compõem o processo de entrada e da coleta de dados.

No processo de descrição da coleta de dados das APIs, a identificação de características inerentes a cada API (ilustradas na Figura 10 – seção 1.3, Figura 16 – seção 1.4, e Figura 24 – seção 1.5) permitiu estabelecer uma síntese das relações entre as suas entidades. Desenvolvida a partir destas análises individualizadas das estruturas das APIs, a Figura 25 ilustra o resultado do processo de sobreposição das entidades e das relações identificadas em cada API, considerando, em princípio, que entidades com o mesmo nome são entidades semelhantes.

Figura 25 – Sobreposição (*Overlay*) dos vínculos de entidades das APIs



Fonte: Elaborado pelo Autor.

No caso de relações entre entidades iguais para todas as APIs, utilizou-se linhas únicas para explicitar estas relações, coloridas na cor preta. As linhas que representam as relações específicas de cada API foram coloridas conforme a legenda incluída na própria ilustração. As linhas contínuas representam relações obrigatórias, e as linhas pontilhadas representam as relações opcionais.

Com esta sobreposição foi possível revelar graficamente as diferenças na forma de coleta de dados entre as APIs analisadas. A primeira diferença é que a *Facebook Graph API* e *LinkedIn REST API* diferenciam-se da *X REST API* no processo de entrada e autorização a coleta de dados.

A X REST API não utiliza uma lista de permissões de acesso para as visões: o *token* de autorização de acesso já contém as permissões de coleta de dados das visões e dos atributos, intrínsecas ao seu próprio funcionamento, ou seja, a concessão do *token* de autorização de acesso já contém uma lista de visões e atributos acessíveis, independente de outros elementos.

A segunda diferença é que a *Facebook Graph* API e a X REST API se diferenciam da *LinkedIn* REST API no processo de relacionamento entre visões. A *LinkedIn* REST API só possui dois pontos de entrada, vinculados às visões de membros e companhias, e não possui arestas para acesso a outras visões, como um tipo de relacionamento.

Na *LinkedIn* REST API, caso o Agente Externo necessite conjuntos de dados de visões de destino, existem duas opções: i) a coleta de dados da visão de destino na forma de valores em um atributo composto, disponível na visão de origem, ii) a coleta de dados da visão de destino de forma manual, acessando a lista de todos os valores possíveis para seus atributos, disponível na documentação de referência.

A partir da análise das três APIs que dão acesso à coleta de dados originários dos principais Serviços de Redes Sociais *Online*, propõe-se um procedimento padronizado para coletar dados sobre as estruturas das APIs. É o primeiro passo para compreensão de como estes dados estão disponíveis e quais podem ser coletados.

Aqui é necessário realizar uma reflexão. Por que é necessário estabelecer um procedimento padronizado para coletar dados das APIs? São quatro argumentos principais. Em primeiro lugar, uma análise que se propõe a possuir um caráter científico deve permitir a replicabilidade das pesquisas. Sendo assim, a realização de distintos procedimentos de coleta de dados para cada API pode apresentar diferentes resultados ou estabelecer um viés, mesmo que não seja de forma deliberada.

Em segundo lugar, se não há um procedimento pré-definido de como serão realizadas as coletas de dados das estruturas das APIs, então diferentes equipes que analisam estas estruturas ficariam mais dependentes em conhecer formas de trabalho das demais equipes para executar a mesma coleta.

Ou seja, explicitar um procedimento padronizado de como coletar os dados permite, por exemplo, que equipes que não se conheçam consigam replicar análises de uma mesma API ou até estabelecer análises complementares.

Além disso, outro argumento é que se não há padronização no processo de coleta de dados das estruturas, então se diminui a capacidade de comparar características entre APIs de diferentes Serviços de Redes Sociais *Online*. Por exemplo, uma coleta com o objetivo de analisar as permissões de acesso a dados no Serviço de Rede Social *Online A* disponíveis para Agentes Externos pode desconsiderar a necessidade de coletar dados sobre as estruturas de relacionamentos. Uma segunda coleta, com o objetivo de analisar o acesso a visões por meio dos relacionamentos no Serviço de Rede Social *Online B* pode desconsiderar, de forma deliberada, a necessidade de coletar dados sobre as estruturas de permissão de acesso. Um terceiro observador terá dificuldades de relacionar as duas pesquisas, pois haverá lacunas nos dados coletados, como, por exemplo, no intuito de realizar uma meta-análise comparativa para verificar o grau de privacidade de acesso a dados por Agentes Externos nestes dois serviços. A pesquisa sobre o Serviço de Rede Social *Online A* desconsiderou a importância dos relacionamentos (acessar uma visão de destino por meio do acesso a outra visão – de origem) e a pesquisa sobre o Serviço de Rede Social *Online B* desconsiderou que as permissões são importantes para restringir o acesso a determinadas visões ou atributos.

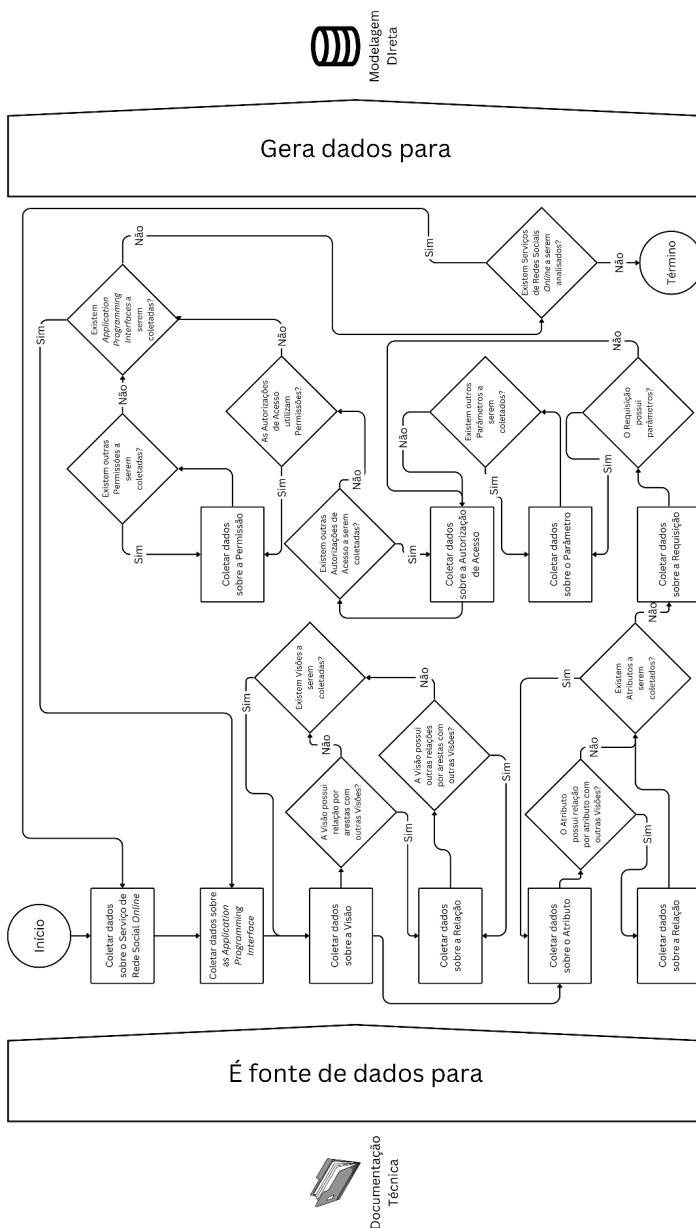
Por último, um procedimento padronizado permite que seja possível analisar o ciclo de vida da API ao longo do tempo, como, por exemplo, permitir a comparação de quais visões e atributos estão disponíveis em cada versão de API – um acompanhamento ao longo do curso da existência da API.

Propõe-se uma coleta sistematizada em etapas, com processos individualizados para a aquisição de dados dos Serviços de Redes Sociais *Online*, das APIs dos Serviços, das Visões disponíveis nas APIs, dos Atributos de cada Visão, das Relações entre as Visões, das Requisições passíveis de serem realizadas na API, das Autorizações de Acesso as Requisições e das Permissões necessárias para a execução da coleta de dados por Agentes Externos.

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Figura 26 – Fluxograma do procedimento padrão para coleta de dados de API de Serviços de Redes Sociais *Online*



Fonte: Elaborado pelo Autor.

A Figura 26 exibe o fluxograma do procedimento padronizado para coleta de dados de API de Serviços de Redes Sociais *Online*. As formas geométricas representam, respectivamente:

- a) Círculos: o início e o término do ciclo de coleta completo, ou seja, são os delimitadores temporais que circundam em totalidade à coleta de dados dos Serviços de Redes Sociais *Online*, das APIs e de suas características;
- b) Retângulos: representam individualmente os 10 processos necessários para a coleta de dados. Cada processo é concomitante as necessidades de coleta identificadas na sobreposição em camadas dos vínculos de entidades das APIs. São processos interligados – com início a partir do término do processo anterior, sendo formado pelo conjunto de processos de coleta de dados sobre Serviços de Redes Sociais *Online*, API, Visões, Atributos, Relações (por arestas e por atributos), das Requisições, dos Parâmetros, das Autorizações de Acesso e das Permissões;
- c) Losangos: representam as questões entre os processos, que definirão o caminho a ser seguido no procedimento padrão. Por exemplo, ao término do processo de coleta de dados de uma determinada visão, o losango subsequente (com o título *A Visão possui relação por arestas com outras Visões?*) questiona se aquela visão apresenta algum tipo de relação por aresta com outras visões da API. Caso a resposta para a questão for positiva, será necessário coletar os dados de cada relação por aresta disponível para a visão. Ao contrário, o coletor será direcionado a outra questão (com o título *Existem Visões a serem coletadas?*).

Além destas formas geométricas, o fluxograma apresenta o material analisado ao lado esquerdo (Documentação Técnica dos Serviços de Redes Sociais *Online*) – a entrada de dados – e os dados que serão gerados para posterior análise ao lado direito – a saída de dados, que será sistematizada e

armazenada em Sistema de Gerenciamento de Banco de Dados – no caso, estruturada na forma de uma Modelagem Direta por meio da aplicação do Modelo Entidade-Relacionamento em um Sistema de Gerenciamento de Banco de Dados. O nome Modelagem Direta foi escolhido por se tratar de uma modelagem que espelha diretamente as características do que está disponível nas APIs.

2.1 SISTEMA DE GERENCIAMENTO DE BANCO DE DADOS E O MODELO ENTIDADE-RELACIONAMENTO

Os computadores têm a capacidade de armazenar e organizar dados de diversas formas. As formas disponíveis coexistem desde a invenção do primeiro computador, porém na década de 1960 inicia-se um processo de consolidação do uso de modelos baseados no uso de um sistema que gerenciasse exclusivamente bancos de dados – e, portanto, contendo funções específicas para esta finalidade.

Nesta década, instituições e comitês, tais como a *International Business Machines* (IBM) e o *Committee on Data Systems Languages* (CODASYL), possuíam equipes de pesquisa e desenvolvimento de novos produtos e serviços voltados a melhoria no processo de gerenciamento de bancos de dados em computadores de grande escala. Começaram a surgir os primeiros bancos de dados utilizando arquivos armazenados nos discos rígidos dos computadores, na forma de arquivos armazenados, contendo conjuntos de dados estruturados (Grad; Bergin, 2010).

Contudo, foram nas décadas de 1970 e 1980 que os Sistemas de Gerenciamento de Bancos de Dados ganharam destaque na elaboração e no armazenamento de conjuntos de dados para uso em sistemas de informação.

É importante destacar que o termo banco de dados significa uma coleção de peças de dados que são organizadas e utilizadas em um computador (Merriam-Webster, 2023) – e que não deve ser confundido com os Sistemas de Gerenciamento de Bancos de Dados.

Segundo Silberschatz, Korth e Sudarshian (2020, p. 1, tradução nossa), os Sistemas de Gerenciamento de Bancos de Dados são:

[...] coleções de dados inter-relacionados e um conjunto de programas para acessar estes dados [...] são projetados para gerenciar grandes corpos de informação. O gerenciamento de dados envolve tanto a definição de estruturas para armazenamento da informação quanto mecanismos para a manipulação da informação.

Ainda nas décadas de 1970 e 1980, já existiam formas diferentes para modelar a forma que os conjuntos de dados destes Sistemas de Gerenciamento de Bancos de Dados estariam dispostos, tais como os modelos: o *flat file* (do inglês, arquivo plano), o hierárquico, o dimensional e o relacional (e suas variantes), principalmente influenciados pelos trabalhos desenvolvidos por Codd (1990), quando ainda era pesquisador da IBM, e, posteriormente, por Kimball e Ross (2011).

Os Sistemas de Gerenciamento de Bancos de Dados são desenvolvidos para o uso genérico, não sendo obrigatório um cenário, um único tipo de estudo de caso ou um determinado sistema de informação específico. São aplicações que gerenciam quaisquer conjuntos de dados, estruturados por instruções previamente estabelecidas pelos desenvolvedores¹ como, por exemplo, uma empresa que deseja elaborar um sistema de gerenciamento de pessoas de uma rede social e, posteriormente, está apto a receber instruções para a inserção, a alteração, a exclusão e a consulta a estes conjuntos (Silberschatz; Korth; Sudarshan, 2020).

Nas décadas seguintes, os Sistemas de Gerenciamento de Banco de Dados continuaram a ser aperfeiçoados e, atualmente, utilizam tanto estruturas baseadas em armazenamento em arquivos, quanto baseadas em sistemas de armazenamento fechados (com o uso de um protocolo exclusivo de acesso e manutenção) – este último, se solidificando como a principal forma de armazenamento e popularizando a adoção do modelo relacional.

¹ O termo desenvolvedores se refere aos desenvolvedores de aplicações, que podem ser de formação específica ao gerenciamento de bancos de dados (como um *Data Base Administrator*) ou pelo próprio desenvolvedor de algoritmos (como um programador), dependendo da abordagem adotada pela instituição.

O modelo hierárquico influenciou a forma de armazenamento de conjunto de dados de outros métodos de armazenamento disponíveis, tais como a estrutura hierárquica dos arquivos em formato compatível com as linguagens de marcação JSON e XML.

As vantagens do uso de Sistemas de Gerenciamento de Bancos de Dados para o desenvolvimento de bancos de dados são (Silberschatz; Korth; Sudarshan, 2020):

- a) Evitar a inconsistência de dados, pois possuem mecanismos que permitem ao desenvolvedor elaborar estruturas e estabelecer alertas para impedir a inserção de cópias de um mesmo registro – ou seja, de forma redundante – evitando a inconsistência de dados;
- b) Auxiliar na consulta unificada aos conjuntos de dados, pois os sistemas que utilizam um Sistema de Gerenciamento de Bancos de Dados para armazenar seus dados podem realizar consultas cruzando diferentes conjuntos de dados, por exemplo, ao realizar consultas vinculando dados de pessoas com dados de suas postagens em um serviço;
- c) Evitar problemas de integridade, pois os bancos de dados estruturados em um Sistema de Gerenciamento de Bancos de Dados podem utilizar constantes para controlar a consistência dos dados, como, por exemplo, restrições de inserção de valores alfanuméricos para dados com tipo data e hora;
- d) Evitar problemas relacionados a atomicidade, pois permitem a realização de operações em formato de transação, ou seja, caso ocorra falhas no decorrer das operações executadas, os conjuntos de dados podem voltar ao seu estado original (técnica denominada *rollback*²), também evitando o surgimento de inconsistências;

² Função que integra parte dos Sistemas Gerenciadores de Banco de Dados e que permite o cancelamento de todas as operações de inserção, de atualização ou de exclusão contidas em uma transação, realizadas por um aplicativo.

- e) Gerenciar acessos concorrentes aos dados, pois possuem funcionalidades para gerenciar o acesso um ou mais usuários ao mesmo tempo (concorrentes) aos conjuntos de dados armazenados. São componentes essenciais para a construção de serviços e aplicativos voltados ao funcionamento via internet, onde é cenário comum aos aplicativos e aos serviços a utilização por mais de um usuário, ao mesmo tempo;
- f) Evitar problemas de segurança, pois o acesso direto aos conjuntos de dados armazenados em Sistemas de Gerenciamento de Bancos de Dados depende da criação e uso de contas de usuários para acesso aos dados. Os aplicativos devem utilizar estas contas de usuário – que são internas e invisíveis aos usuários do sistema – para acessar os conjuntos de dados; e estas contas podem ter níveis de acesso diferentes. Por exemplo, uma aplicação pode ter acesso restrito a leitura dos conjuntos de dados, com o uso de contas que só permitem consultas aos dados armazenados no Sistema de Gerenciamento de Bancos de Dados, sem as permissões de inserção, alteração ou exclusão;
- g) Aumentar a eficiência de resposta no acesso aos dados, pois diferente de propostas anteriores de armazenamento de conjuntos de dados, um Sistema de Gerenciamento de Bancos de Dados pode processar grandes quantidades de registros em um espaço de tempo considerado bom ou aceitável, para a maioria dos cenários.

Os bancos de dados em um Sistema de Gerenciamento de Bancos de Dados podem ser modelados seguindo princípios de abstração – formas que permitem visualizar as estruturas, os detalhes e a forma da disposição dos conjuntos de dados a serem armazenados. Estão divididos em três níveis (Codd, 1990; Silberschatz; Korth; Sudarshan, 2020):

- a) Nível Físico: é o nível de abstração vinculado diretamente com a forma que será armazenado os conjuntos de dados no Sistema de

Gerenciamento de Bancos de Dados. Um exemplo de explicitação de bancos de dados em nível físico são os esquemas, diagramas e instruções elaborados para o funcionamento em um determinado Sistema de Gerenciamento de Bancos de Dados;

- b) **Nível Lógico:** é o nível de abstração que explicita o que será armazenado no nível físico. Por exemplo, são os modelos conceituais para explicitar entidades, atributos, tipos de dados e relações, tais como o Modelo Entidade-Relacionamento (Kingberg; Mccubbin; Martin, 1998);
- c) **Nível Visão:** é o nível de abstração, opcional, geralmente representando parte do modelo conceitual de um banco de dados. Geralmente é utilizado para descrever grandes bancos de dados, de alta complexidade. Por exemplo, o modelo conceitual de um banco de dados de um Serviço de Rede Social *Online* pode conter um número expressivo de entidades e relações, e, não necessariamente, há interesse em revelar a instituições parceiras a totalidade destas informações. Portanto, uma API pode explicitar um recorte do modelo conceitual do banco de dados, contendo apenas uma visão parcial das entidades e das relações existentes e, restringindo a visão (consequentemente, o acesso) de outras entidades e relações.

Existem diferentes abordagens para a elaboração de modelos com técnicas e conceitos próprios para a construção de abstrações dos dados. A partir de reflexões de Silberschatz, Korth e Sudarshan (2020) e de Codd (1990), as abordagens podem ser divididas em quatro categorias:

- a) **Relacionais:** modelo baseado no conceito matemático de relação – por Codd (1990) – que utiliza uma coleção de tabelas para a representação dos conjuntos de dados e suas relações;

- b) Orientados a Objetos: modelo baseado no aporte de conceito originário de Orientação a Objetos – utilizado na construção de aplicativos por linguagens de programação (Atkinson *et al.*, 1990);
- c) Semiestruturado: modelo com características de aceitação de flexibilidade das informações contidas em conjuntos de dados de mesma tipologia, como as representações de conjuntos de dados com o uso de linguagens de marcação XML ou JSON (Abello; Pardalos; Resende, 2002);
- d) Entidade-Relacionamento: derivado do modelo relacional, o Modelo Entidade-Relacionamento utiliza objetos para explicitar, respectivamente: as entidades (objetos do mundo real) e seus relacionamentos (com outras entidades) através do uso de chaves.

Apesar da existência de Sistemas de Gerenciamento de Bancos de Dados com outras abordagens para a elaboração de modelos, tais como os modelos para armazenamentos de tabelas não-normalizadas, estes modelos ainda seguem os princípios baseados nas categorias apresentadas, como o uso de tabelas para a explicitação de entidades ou o uso de restrições por chave.

O Modelo Entidade-Relacionamento é uma forma de modelagem lógica de banco de dados que utiliza coleções de objetos – denominados entidades – para explicitar as estruturas de armazenamento de conjuntos de dados e possui elementos que permitem relacioná-los entre si – denominados relações (Silberschatz; Korth; Sudarshan, 2020). Este modelo foi uma melhoria baseada nos conceitos do modelo relacional, proposto por Codd (1990), e é uma das formas mais populares de explicitação de bancos de dados.

As entidades são compostas por conjuntos de atributos de um objeto pré-definido pelo desenvolvedor do banco de dados (Date, 2016). Por exemplo: *url*, *nome* e *descricao* podem ser atributos da entidade *visao* (Visão), e *codigo*, *nome* e *descricao* podem ser atributos da entidade *atributo*.

Na camada física de um Sistema de Gerenciamento de Bancos de Dados, entidades do Modelo Entidade-Relacionamento se tornam tabelas: são representações para cada entidade, que possuem a estrutura similar a apresentadas nas estruturas de aplicativos para planilhas eletrônicas. A tabela representa a entidade, e cada coluna da tabela representa um atributo da entidade. As linhas representarão os conjuntos de dados que serão armazenados (denominados registros) a serem armazenados em cada tabela (Codd, 1990; Date, 2016; Silberschatz; Korth; Sudarshan, 2020). Os elementos da composição das entidades <Tabela, Coluna e Linha> também podem ser denominadas como <Relação, Atributo e Tupla> ou <Arquivo, Campo e Registro>.

Cada coluna representa uma característica para a tabela e não podem possuir nomes iguais em uma mesma tabela. Além disso, a consulta aos dados da tabela pode declarar quais colunas deseja recuperar e o Sistema de Gerenciamento de Bancos de Dados, por outro lado, pode restringir ou não o acesso a cada coluna (Silberschatz; Korth; Sudarshan, 2020).

As colunas também possuem restrições individuais dos tipos de valores que serão armazenados. Podem ser simples – com seu valor sendo formado por um número, um texto, uma data, entre outros tipos – ou compostas – com seu valor formado por um conjunto de valores, que podem dar acesso a subcolunas (Silberschatz; Korth; Sudarshan, 2020). As restrições das colunas são denominadas tipos de dado e, segundo a escola de aprimoramento de desenvolvedores de aplicativos W3SCHOOLS (2023), os tipos de dados mais populares são:

- a) *boolean*, utilizado no armazenamento de números binários para compor a lógica booleana (como as condições sim ou não, verdadeiro ou falso, ligado ou desligado, entre outras);
- b) *integer*, para armazenamento de números inteiros;
- c) *float*, para armazenamento de números racionais de escrita finita;
- d) *currency*, para armazenamento de valores em formato de moeda;

- e) *string*, para armazenamento de conjuntos de símbolos, com tamanho reservado de forma fixa ou variável;
- f) *binary object*, para armazenamento de conjuntos de símbolos que possuem tamanhos maiores que o limite máximo para o tipo de dado *string*.

Todavia, os Sistemas de Gerenciamento de Bancos de Dados possuem outros tipos de dados que podem variar de acordo com as especificações de cada fabricante destas tecnologias (Date, 2016; Silberschatz; Korth; Sudarshan, 2020).

Para garantir a unicidade e a identificação de cada linha em uma tabela, uma coluna ou um conjunto de colunas devem ser estabelecidos como chaves, também conhecidas como identificadores, que são colunas com restrições especiais (Date, 2016; Silberschatz; Korth; Sudarshan, 2020).

As chaves possuem as seguintes características:

- a) Superchaves: são colunas ou conjuntos de colunas que identificam unicamente uma linha. Por exemplo, a coluna *url* é a superchave da tabela *visao*, pois cada visão possui um *Uniform Resource Locator* único para acesso;
- b) Chaves candidatas: quando somente uma coluna possui a capacidade de identificar unicamente uma linha, esta coluna se torna uma chave em potencial, denominada chave candidata. Esta chave candidata pode ser escolhida (ou não) como chave primária pelo indivíduo desenvolvedor do banco de dados. Por exemplo, a coluna *url* é potencialmente uma chave candidata para a tabela *visao*, pois não existe a possibilidade de uma visão não possuir um *Uniform Resource Locator* único para acesso;
- c) Chaves primárias: são as colunas escolhidas para a identificação única de cada linha. Podem ser formadas tanto por colunas das chaves candidatas, quanto por um identificador único artificial,

como, por exemplo, um código sequencial único para cada linha. Independente do critério adotado, o valor contido nestas colunas deve ser único e não podem ser nulos (valor vazio). Por exemplo, a tabela *atributo* precisa de um código artificial como superchave, um código gerado pelo próprio Sistemas de Gerenciamento de Bancos de Dados, já que pode existir atributos homônimos, mas distintos;

- d) Chaves estrangeiras: são as chaves primárias que participam como colunas de outras tabelas para formalizar o relacionamento entre as tabelas. Por exemplo, em uma *visao* (tabela) é necessário listar seus *atributos* (tabela *atributo*). Portanto, a coluna *url* estará tanto na tabela de origem *visao* (como chave primária) como na tabela de destino *atributo* (como chave estrangeira). Assim, ao recuperar linhas da tabela *atributo*, a coluna *url* será o elemento de ligação entre as linhas da tabela *atributo* e as linhas da tabela *visao*.

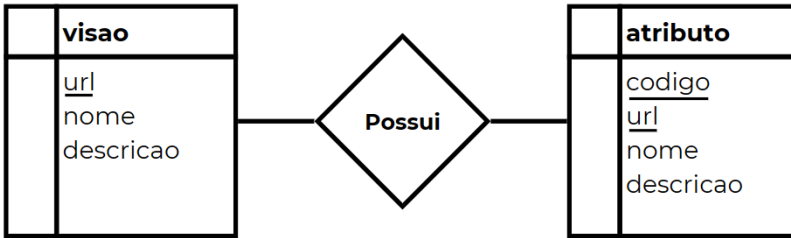
Portanto, as tabelas podem relacionar suas linhas com outras tabelas, com o uso de chaves primárias e estrangeiras (Codd, 1990; Date, 2016; Silberschatz; Korth; Sudarshan, 2020), e no exemplo, o Modelo Entidade-Relacionamento pode relacionar a tabela *atributo* com a tabela *visao*, pois cada linha inserida na tabela *atributo* deverá relacionar-se com uma linha da tabela *visao*, pois sempre um *atributo* deve participar de *visao*. As colunas *url* e *codigo* são colunas identificadoras (chaves primárias) destas tabelas e a tabela *atributo* deverá conter uma coluna identificadora extra para receber a chave primária da tabela *Visão*: *url*.

O valor coluna *url*, representando o *Uniform Resource Locator* único para acesso a *visao*, será a chave estrangeira da tabela *atributo*, funcionando como elo entre uma *visao* e um *atributo*, para cada linha.

A Figura 27 exibe este exemplo Modelo de Entidade-Relacionamento em nível lógico, na forma de diagrama – denominado Diagrama Entidade-Relacionamento, um dos elementos visuais existentes para a expressão lógica das tabelas, das colunas e de suas relações. No exemplo, está delimita-

do que a tabela *visao* possui *atributo*, sendo que as chaves são as colunas *url* (para a tabela *visao*), *codigo e url* (para a tabela *atributo*).

Figura 27 – Exemplo de um Diagrama Entidade-Relacionamento no nível lógico



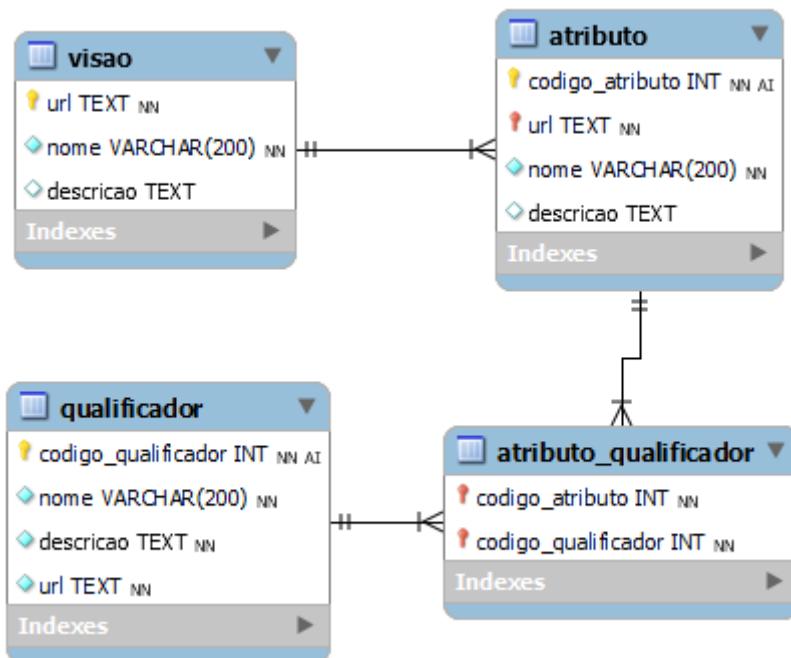
Fonte: Elaborado pelo Autor.

A forma do relacionamento entre as tabelas pode variar de acordo com a cardinalidade – a explicitação da quantidade de linhas pode ser relacionada em uma determinada relação entre tabelas (Date, 2016; Silberschatz; Korth; Sudarshan, 2020).

O mapeamento das restrições para a cardinalidade dos relacionamentos auxilia o processo lógico de relacionamento entre as tabelas e é representado por um conjunto fixo de símbolos, sendo (Codd, 1990; Date, 2016; Silberschatz; Korth; Sudarshan, 2020):

- a) 1-para-1: quando o relacionamento entre as tabelas é permitido, mas restrito, no máximo, entre duas linhas, uma em cada tabela;
- b) 1-para-N: quando a linha da tabela de origem pode relacionar-se com uma ou mais linhas da tabela de destino;
- c) N-para-N: quando as linhas de duas tabelas podem se relacionar diversas vezes entre si.

Figura 28 – Exemplo de um Diagrama Entidade-Relacionamento no nível físico



Fonte: Elaborado pelo Autor.

No nível físico, o Diagrama Entidade-Relacionamento (Figura 28) passa a expressar elementos tais como: os tipos de dado (a direita dos nomes de colunas); as restrições de conteúdo (a direita dos tipos de dado, com signos *NN* para as colunas com preenchimento obrigatório e com os signos *AI* para colunas no qual o conteúdo será controlado automaticamente pelo sistema, no caso, um número autoincrementado); as chaves primárias (chaves douradas à esquerda das colunas); as chaves estrangeiras (losangos ou chaves vermelhas à esquerda das colunas); as colunas de preenchimento obrigatório, que não podem ser nulas (losangos azuis à esquerda das colunas), e; as relações.

Para as relações, o exemplo exibe um relacionamento 1-para-N entre as tabelas *visao* e *atributo*, e; N-para-N entre as tabelas *atributo* e *qualificador*. Para relações N-para-N, surge uma tabela de ligação, denominada *atributo_qualificador*, que recebe as chaves primárias das duas tabelas, já que um *atributo* pode possuir de zero a infinitos *qualificadores*, e um *qualificador* pode estar associado de zero a infinitos *atributos*.

Outra representação para auxiliar a compreensão da estrutura de um banco de dados e a compreensão do Diagrama Entidade-Relacionamento é o Dicionário de Dados, que são instrumentos elaborados para auxiliar a compreensão das colunas e dos seus: tipos de dado, tamanho máximo de caracteres, a obrigatoriedade de inserção de valores na criação de novas linhas, restrições de chave primária e a descrição sobre o significado da coluna e/ou a expectativa dos valores a serem inseridos.

Portanto, o Dicionário de Dados é um dos elementos importantes na manutenção ou alteração das características das tabelas e das colunas em um Sistema de Gerenciamento de Bancos de Dados (Codd, 1990; Date, 2016; Silberschatz; Korth; Sudarshan, 2020). O Quadro 4 apresenta um exemplo de Dicionário de Dados para a entidade *visao*.

Quadro 4 – Exemplo de Dicionário de Dados para um Modelo Entidade-Relacionamento

Tabela: <i>visao</i>						
Descrição: Entidade que armazena dados sobre cada Visão.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>url</i>	TEXT	Não	Sim	Não	-	O <i>Uniform Resource Locator</i> único para acesso à visão.
<i>nome</i>	VARCHAR(200)	Não	Não	Não	-	Nome da visão, de acordo com a documentação técnica da <i>Application Programming Interface</i> .

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

<i>descricao</i>	TEXT	Sim	Não	Não	NULL	Descrição da visão, de acordo com a documentação técnica da <i>Application Programming Interface</i> . Em certos casos, a documentação técnica poderá não apresentar uma descrição para a visão.
------------------	------	-----	-----	-----	------	--

Fonte: Elaborado pelo Autor.

Portanto, o armazenamento de conjuntos de dados em ambientes digitais está sedimentado em uma tríade de relação Entidade/Atributo/Valor <E, A, V> - sinónímia a relação Tabela/Coluna/Valor³ <T, C, V> nos Sistemas de Gerenciamento de Banco de Dados. Os elementos desta tríade são considerados como: tabela, um objeto do mundo real, com características que o distingue de outros objetos; colunas, características intrínsecas do objeto, e: valor, que representa um valor de uma coluna de uma tabela específica, com determinada aceitação a um tipo de dado (Santos; Sant’Ana, 2015; Silberschatz; Korth; Sudarshan, 2020).

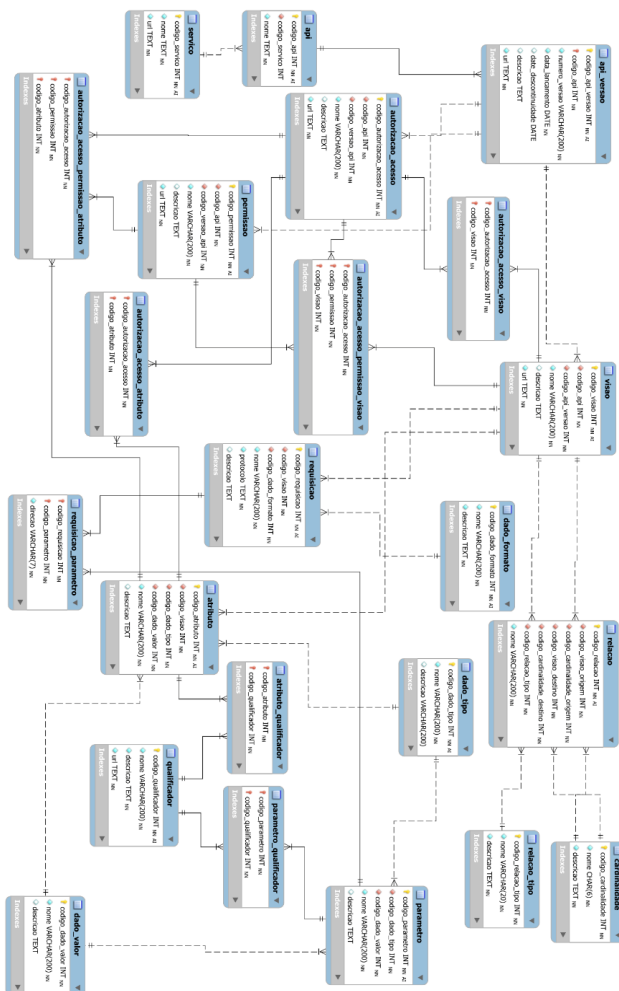
Esta tríade é a base conceitual tanto para modelagens de banco de dados e como para a interoperabilidade de conjuntos de dados entre sistemas de informação para a internet (Berners-Lee; Hendler; Lassila, 2001; Codd, 1990; Santos; Sant’Ana, 2015; Silberschatz; Korth; Sudarshan, 2020) – além de sua aplicação ser compatível para explicitar estruturas dos Serviços de Redes Sociais *Online*.

³ Valor representando o valor contido em uma linha para uma determinada coluna.

2.2 APLICAÇÃO DO MODELO ENTIDADE-RELAÇONAMENTO NA MODELAGEM DIRETA

A Figura 29 apresenta o Diagrama Entidade-Relacionamento da Modelagem Direta, contendo as tabelas, as colunas, os tipos de dados, e os relacionamentos para instrumentalizar procedimento padronizado para coletar dados sobre as estruturas das APIs.

Figura 29 – Diagrama Entidade-Relacionamento da Modelagem Direta



Fonte: Elaborado pelo Autor.

O Diagrama Entidade-Relacionamento é composto de 23 tabelas, e em todas optou-se por utilizar chaves primárias artificiais, geradas automaticamente pelo Sistema de Gerenciamento de Banco de Dados. Este processo foi definido para facilitar ao coletor – pessoa ou grupo no qual será responsável por aplicar o procedimento padrão – ao deixar a cargo Sistema de Gerenciamento de Banco de Dados a definição de valores das chaves primárias para cada linha cadastrada em cada tabela.

Por padrão, o Sistema de Gerenciamento de Banco de Dados se utiliza de uma sequência numérica, formada por um conjunto infinito de números inteiros positivos, iniciando pelo número um – e sendo somado mais um a cada nova linha adicionada em cada tabela.

As tabelas foram agrupadas em três tipos: 11 tabelas relacionadas ao procedimento padrão, sete tabelas de entidades associativas e cinco tabelas auxiliares.

As tabelas relacionadas ao procedimento padrão foram elaboradas para atender o armazenamento de dados do fluxo da coleta:

1. *servico*: tabela com dados sobre o Serviço de Rede Social *Online*;
2. *api*: tabela com dados sobre cada API;
3. *api_versao*: tabela com dados sobre cada versão das APIs;
4. *autorizacao_acesso*: tabela com dados sobre autorizações de acesso existentes em uma versão de API;
5. *permissao*: tabela com dados sobre as permissões existentes em uma versão de API;
6. *visao*: tabela contendo dados sobre visões existentes em uma versão de API;
7. *requisicao*: tabela com dados sobre as requisições disponíveis em uma visão;
8. *parametro*: tabela contendo dados referentes aos parâmetros disponíveis nas requisições de uma visão;

9. *atributo*: tabela com dados sobre os atributos de uma visão;
10. *qualificador*: tabela contendo dados sobre cada qualificador disponível para associação com atributos ou parâmetros;
11. *relacao*: tabela contendo dados sobre as relações entre visões. Uma linha da tabela *relacao* relaciona-se duas vezes com a tabela *visao* (um relacionamento com uma linha da tabela *visao* determinando a visão de origem e um relacionamento com uma linha da tabela *visao* determinando a visão de destino).

Já as tabelas de entidades associativas foram elaboradas para permitir o relacionamento N-para-N entre tabelas do procedimento padrão:

1. *atributo_qualificador*: tabela para o relacionamento de cardinalidade N-para-N entre linhas da tabela *atributo* e *qualificador*. As linhas da tabela *atributo* podem relacionar-se com a cardinalidade N-para-N com as linhas da tabela *qualificador* (já que um atributo pode conter um ou mais qualificadores e os qualificadores podem participar de uma ou mais atributos);
2. *autorizacao_acesso_atributo*: tabela para o relacionamento de cardinalidade N-para-N entre linhas da tabela *autorizacao_acesso* e *atributo*. As linhas da tabela *autorizacao_acesso* podem relacionar-se com a cardinalidade N-para-N com as linhas da tabela *atributo*, já que uma autorização de acesso pode dar acesso a um ou mais atributos e os atributos podem ser acessíveis por uma ou mais autorizações de acesso;
3. *autorizacao_acesso_visao*: tabela para o relacionamento de cardinalidade N-para-N entre linhas da tabela *autorizacao_acesso* e *visao*. As linhas da tabela *autorizacao_acesso* podem relacionar-se com a cardinalidade N-para-N com as linhas da tabela *visao*, já que uma autorização de acesso pode dar acesso a uma ou mais

visões e as visões podem ser acessíveis por uma ou mais autorizações de acesso;

4. *autorizacao_acesso_permissao_atributo*: tabela para o relacionamento de cardinalidade N-para-N entre linhas da tabela *autorizacao_acesso*, *permissao* e *atributo*. As linhas da tabela *autorizacao_acesso*, *permissao* e *atributo* podem relacionar-se com a cardinalidade N-para-N, já que a combinação entre uma autorização de acesso e uma permissão pode dar acesso a um ou mais atributos e os atributos podem ser acessíveis por uma ou mais combinações entre autorizações de acesso e permissões;
5. *autorizacao_acesso_permissao_visao*: tabela para o relacionamento de cardinalidade N-para-N entre linhas da tabela *autorizacao_acesso*, *permissao* e *visao*. As linhas da tabela *autorizacao_acesso*, *permissao* e *visao* podem relacionar-se com a cardinalidade N-para-N, já que a combinação entre uma autorização de acesso e uma permissão pode dar acesso a uma ou mais visões e as visões podem ser acessíveis por uma ou mais combinações entre autorizações de acesso e permissões;
6. *parametro_qualificador*: tabela para o relacionamento de cardinalidade N-para-N entre linhas da tabela *parametro* e *qualificador*. As linhas da tabela *parametro* podem relacionar-se com a cardinalidade N-para-N com as linhas da tabela *qualificador*, já que um parâmetro pode conter um ou mais qualificadores e os qualificadores podem participar de uma ou mais parâmetros;
7. *requisicao_parametro*: tabela para o relacionamento de cardinalidade N-para-N entre linhas da tabela *requisicao* e *parametro*. As linhas da tabela *requisicao* podem relacionar-se com a cardinalidade N-para-N com as linhas da tabela *parâmetro*, já que uma requisição pode conter um ou mais parâmetros e os parâmetros podem participar de uma ou mais requisições.

No caso das tabelas auxiliares, elas foram desenvolvidas para auxiliar o processo de funcionamento da Modelagem Direta, em sinergia com os fluxos propostos no procedimento padronizado. O conteúdo destas tabelas deve ser preenchido *a priori* do início do procedimento padrão para coletar dados sobre as estruturas das APIs:

1. *cardinalidade*: tabela contendo dados sobre os tipos de cardinalidade existentes para explicitar os relacionamentos de vinculação de conteúdos entre duas visões. As linhas desta tabela se relacionam com a tabela *relacao*;
2. *dado_formato*: tabela com dados sobre os formatos de dados disponíveis para a coleta de dados de uma requisição, tais como o uso de formatos de notação das linguagens de marcação JSON e XML;
3. *dado_tipo*: tabela com dados sobre os tipos de dados existentes para delimitar as características esperadas para parâmetros e atributos. No caso das linhas desta tabela, poderão ser inclusos tipos de dados específicos de cada API, como os tipos de dados que apresentam conteúdo de uma visão;
4. *dado_valor*: tabela com dados sobre os valores de dados existentes para delimitar as características esperadas para valores de parâmetros e atributos – se os valores serão simples, compostos ou se não é possível uma definição;
5. *relacao_tipo*: tabela com dados sobre os tipos de relação das visões, como o uso de arestas ou de atributos para a vinculação de conteúdos entre duas visões.

O Quadro 5 aprofunda o detalhamento técnico destas tabelas, apresentando o Dicionário de Dados contendo informações sobre tabelas, colunas e relacionamentos do Diagrama Entidade-Relacionamento. Para cada entidade proposta, apresenta-se um cabeçalho, contendo o nome da

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

tabela, a descrição de seu conteúdo, seguido de uma descrição das colunas. Para cada coluna é apresentado seu nome, o tipo de dado esperado, o tamanho máximo do valor, se a coluna aceita valores nulos, se a coluna é chave primária ou estrangeira, suas configurações padrão e a descrição do tipo de dado que é esperado para armazenar.

Quadro 5 – Dicionário de Dados da Modelagem de Direta

Tabela: <i>api</i>						
Descrição: Entidade que contém as informações sobre cada <i>Application Programming Interface</i> .						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_api</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_servico</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre Serviço de Rede Social Online e <i>Application Programming Interface</i> , com origem na tabela <i>servico</i> , coluna <i>codigo_servico</i> .
<i>Nome</i>	TEXT	Não	Não	Não		O nome da <i>Application Programming Interface</i> .

Tabela: <i>api_versao</i>						
Descrição: Entidade que armazena dados sobre cada Versão de <i>Application Programming Interface</i> .						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_api_versao</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_api</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a <i>Application Programming Interface</i> e a Versão da <i>Application Programming Interface</i> , com origem na tabela <i>api</i> , coluna <i>codigo_api</i> .
<i>numero_versao</i>	VARCHAR (200)	Não	Não	Não		Número da Versão da <i>Application Programming Interface</i> .
<i>data_lancamento</i>	DATE	Não	Não	Não		Data de início do funcionamento da Versão da <i>Application Programming Interface</i> .

<i>data_descontinuidade</i>	DATE	Sim	Não	Não	NULL	Data do encerramento da disponibilidade da Versão da <i>Application Programming Interface</i> .
<i>descricao</i>	TEXT	Sim	Não	Não		A descrição original da Versão da <i>Application Programming Interface</i> (quando disponível), conforme documentação de referência.
<i>url</i>	TEXT	Não	Não	Não		O principal endereço de acesso ao documento da Versão da <i>Application Programming Interface</i> , no formato <i>Uniform Resource Locator</i> .

Tabela: <i>autorizacao_acesso</i>						
Descrição: Entidade que armazena dados sobre cada Autorização de Acesso.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_autorizacao_acesso</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_api</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a <i>Application Programming Interface</i> e a Autorização de Acesso, com origem na tabela <i>api-versao</i> , coluna <i>codigo_api</i> .
<i>codigo-versao_api</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a Versão da <i>Application Programming Interface</i> e a Autorização de Acesso, com origem na tabela <i>api-versao</i> , coluna <i>codigo-versao_api</i> .
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome da Autorização de Acesso.
<i>descricao</i>	TEXT	Sim	Não	Não		A descrição original da Autorização de Acesso (quando disponível), conforme documentação de referência.
<i>url</i>	TEXT	Não	Não	Não		O endereço principal de acesso ao documento da Autorização de Acesso, no formato <i>Uniform Resource Locator</i> .

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Tabela: <i>autorizacao_acesso_atributo</i>						
Descrição: Entidade associativa de união entre Autorizações de Acesso e Atributos. Utilizada para relacionar quais Atributos estão disponíveis em cada Autorização de Acesso.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_ autorizacao_ acesso</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso e o Atributo, com origem na tabela <i>autorizacao_acesso</i> , coluna <i>codigo_ autorizacao_ acesso</i> .
<i>codigo_ atributo</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso e o Atributo, com origem na tabela <i>atributo</i> , coluna <i>codigo_ atributo</i> .

Tabela: <i>autorizacao_acesso_permissao_atributo</i>						
Descrição: Entidade associativa de união entre Autorizações de Acesso, Permissões e Atributos. Utilizada para relacionar quais Atributos estão disponíveis com o uso de Autorização de Acesso e Permissão.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_ autorizacao_ acesso</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso, a Permissão e o Atributo, com origem na tabela <i>autorizacao_acesso</i> , coluna <i>codigo_ autorizacao_ acesso</i> .
<i>codigo_ permissao</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso, a Permissão e o Atributo, com origem na tabela <i>permissao</i> , coluna <i>codigo_ permissao</i> .
<i>codigo_ atributo</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso, a Permissão e o Atributo, com origem na tabela <i>atributo</i> , coluna <i>codigo_ atributo</i> .

Tabela: <i>autorizacao_acesso_permissao_visao</i>						
Descrição: Entidade associativa de união entre Autorizações de Acesso, Permissões e Visões. Utilizada para relacionar quais Visões estão disponíveis com o uso de Autorização de Acesso e Permissão.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_ autorizacao_ acesso</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso, a Permissão e a Visão, com origem na tabela <i>autorizacao_acesso</i> , coluna <i>codigo_ autorizacao_ acesso</i> .
<i>codigo_permissao</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso, a Permissão e a Visão, com origem na tabela <i>permissao</i> , coluna <i>codigo_permissao</i> .
<i>codigo_visao</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso, a Permissão e a Visão, com origem na tabela <i>visao</i> , coluna <i>codigo_visao</i> .

Tabela: <i>autorizacao_acesso_visao</i>						
Descrição: Entidade associativa de união entre Autorizações de Acesso e Visões. Utilizada para relacionar quais Visões estão disponíveis em cada Autorização de Acesso.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_ autorizacao_ acesso</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso e a Visão, com origem na tabela <i>autorizacao_acesso</i> , coluna <i>codigo_ autorizacao_ acesso</i> .
<i>codigo_visao</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Autorização de Acesso e a Visão, com origem na tabela <i>visao</i> , coluna <i>codigo_ autorizacao_ acesso</i> .

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

Tabela: cardinalidade						
Descrição: Tabela Auxiliar utilizada para armazenar dados que descrevem as Cardinalidades das Relações entre as Visões.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_cardinalidade</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	CHAR (6)	Não	Não	Não		O nome da Cardinalidade, podendo ser 1-para-1, 1-para-N ou N-para-N.
<i>descricao</i>	TEXT	Não	Não	Não		A descrição das Cardinalidades: 1-para-1, 1-para-N ou N-para-N.

Tabela: atributo						
Descrição: Entidade que armazena dados sobre cada Atributo disponível nas Visões.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_atributo</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_visao</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a Visão e o Atributo, com origem na tabela <i>visao</i> , coluna <i>codigo_visao</i> .
<i>codigo_dado_tipo</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre o Atributo e o Tipo de Dado para os seus valores, com origem na tabela <i>dado_tipo</i> , coluna <i>codigo_dado_tipo</i> .
<i>codigo_dado_valor</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre o Atributo e o Valor de Dado para os seus valores, com origem na tabela <i>dado_valor</i> , coluna <i>codigo_dado_valor</i> .
<i>nome</i>	VARCHAR (200)	Não	Não	Não		Nome original do Atributo, conforme descrito na documentação de referência.

<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original do Atributo (quando disponível), conforme documentação de referência.
------------------	------	-----	-----	-----	------	--

Tabela: *atributo_qualificador*

Descrição: Entidade associativa de união entre Atributos e Qualificadores. Utilizada para relacionar quais são os Qualificadores de cada Atributo.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_atributo</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre o Atributo e o Qualificador, com origem na tabela <i>atributo</i> , coluna <i>codigo_atributo</i> .
<i>codigo_qualificador</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre o Atributo e o Qualificador, com origem na tabela <i>qualificador</i> , coluna <i>codigo_qualificador</i> .

Tabela: *dado_formato*

Descrição: Tabela Auxiliar utilizada para armazenar dados que descrevem os Formatos de Dados disponíveis no momento da coleta de dados por meio das Requisições.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_dado_formato</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome do formato que os dados podem ser coletados.
<i>descricao</i>	TEXT	Não	Não	Não		A descrição do formato que os dados podem ser coletados.

Tabela: *servico*

Descrição: Entidade que armazena dados sobre os Serviços de Redes Sociais *Online* aos quais as *Application Programming Interfaces* estão vinculadas.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_servico</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	TEXT	Não	Não	Não		O nome do Serviço de Rede Social <i>Online</i> .

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

<i>url</i>	TEXT	Não	Não	Não		O principal endereço de acesso ao Serviço de Rede Social <i>Online</i> , no formato <i>Uniform Resource Locator</i> .
------------	------	-----	-----	-----	--	---

Tabela: *parametro*

Descrição: Entidade que armazena dados sobre cada Parâmetro disponível nas Requisições.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_parametro</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_dado_tipo</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre o Parâmetro e o Tipo de Dado para os seus valores, com origem na tabela <i>dado_tipo</i> , coluna <i>codigo_parametro</i> .
<i>codigo_dado_valor</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre o Parâmetro e o Valor de Dado para os seus valores, com origem na tabela <i>dado_valor</i> , coluna <i>codigo_dado_valor</i> .
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome do Parâmetro.
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original do Parâmetro (quando disponível), conforme documentação de referência.

Tabela: *parametro_qualificador*

Descrição: Entidade associativa de união entre Parâmetros e Qualificadores. Utilizada para relacionar quais são os Qualificadores de cada Parâmetro.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_parametro</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre o Parâmetro e o Qualificador, com origem na tabela <i>parametro</i> , coluna <i>codigo_parametro</i> .
<i>codigo_qualificador</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre o Parâmetro e o Qualificador, com origem na tabela <i>qualificador</i> , coluna <i>codigo_qualificador</i> .

Tabela: <i>permissao</i>						
Descrição: Entidade que armazena dados sobre cada Permissão.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_permissao</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_api</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a <i>Application Programming Interface</i> e a Permissão, com origem na tabela <i>api-versao</i> , coluna <i>codigo_api</i> .
<i>codigo-versao_api</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a Versão da <i>Application Programming Interface</i> e a Permissão, com origem na tabela <i>api-versao</i> , coluna <i>codigo-versao_api</i> .
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome da Permissão.
<i>descricao</i>	TEXT	Sim	Não	Não		A descrição original da Permissão (quando disponível), conforme documentação de referência.
<i>url</i>	TEXT	Não	Não	Não		O endereço de acesso da documentação contendo a referência da Permissão, no formato <i>Uniform Resource Locator</i> .

Tabela: <i>qualificador</i>						
Descrição: Entidade que armazena dados sobre cada Qualificador.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_qualificador</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR (200)	Não	Não	Não		Nome original do Qualificador, conforme descrito na documentação de referência.
<i>descricao</i>	TEXT	Não	Não	Não		A descrição original do Qualificador (quando disponível), conforme documentação de referência.

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

<i>url</i>	TEXT	Não	Não	Não		O endereço principal de acesso ao documento do qualificador, no formato <i>Uniform Resource Locator</i> .
------------	------	-----	-----	-----	--	---

Tabela: *relacao*

Descrição: Entidade que armazena dados sobre cada Relação entre as Visões.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_relacao</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_visao_origem</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre o Relacionamento e a Visão de origem, com origem na tabela <i>visao</i> , coluna <i>codigo_visao</i> .
<i>codigo_cardinalidade_origem</i>	INT	Não	Não	Sim		A Cardinalidade do relacionamento com a Visão de origem.
<i>codigo_visao_destino</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre o Relacionamento e a Visão de destino, com origem na tabela <i>visao</i> , coluna <i>codigo_visao</i> .
<i>codigo_cardinalidade_destino</i>	INT	Não	Não	Sim		A Cardinalidade do relacionamento com a Visão de destino.
<i>codigo_relacao_tipo</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a Relação e o Tipo de Relação entre as Visões, com origem na tabela <i>tipo_relacao</i> , coluna <i>codigo_relacao</i> .
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome do relacionamento.

Tabela: *requisicao*

Descrição: Entidade que armazena dados sobre cada Requisição.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_requisicao</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_visao</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a Requisição e a Visão, com origem na tabela <i>visao</i> , coluna <i>codigo_visao</i> .

<i>codigo_dado_formato</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a Requisição e o Formato do Dado, com origem na tabela <i>dado_formato</i> , coluna <i>codigo_dado_formato</i> .
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome da Requisição.
<i>protocolo</i>	TEXT	Não	Não	Não		Nome do protocolo utilizado para realizar a requisição e coletar os dados.
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original da Requisição (quando disponível), conforme documentação de referência.

Tabela: *requisicao_parametro*

Descrição: Entidade associativa de união entre Requisições e Parâmetros. Utilizada para relacionar quais são os Parâmetros de cada Requisição.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_requisicao</i>	INT	Não	Sim	Sim	0	Chave estrangeira do relacionamento entre a Requisição e o Parâmetro, com origem na <i>tabela requisicao</i> , coluna <i>codigo_requisicao</i> .
<i>codigo_parametro</i>	INT	Não	Sim	Sim		Chave estrangeira do relacionamento entre a Requisição e o Parâmetro, com origem na <i>tabela parametro</i> , coluna <i>codigo_parametro</i> .
<i>direcao</i>	VARCHAR (7)	Não	Não	Não		Sentido do parâmetro no momento da requisição. Ele pode ser de entrada, de saída, de entrada e saída.

Tabela: *dado_tipo*

Descrição: Tabela Auxiliar utilizada para armazenar dados que descrevem os Tipos de Dados utilizados por Atributos das Visões ou por Parâmetros das Requisições.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_dado_tipo</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome do Tipo de Dado.

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

<i>descricao</i>	VARCHAR (200)	Sim	Não	Não		A descrição das características do Tipo de Dado.
------------------	------------------	-----	-----	-----	--	--

Tabela: relacao_tipo

Descrição: Tabela Auxiliar utilizada para armazenar dados que descrevem os Tipos de Relação entre as Visões.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_relacao_tipo</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR (20)	Não	Não	Não		O nome do Tipo de Relação entre as Visões.
<i>descricao</i>	TEXT	Não	Não	Não		Descrição sobre as características do Tipo de Relação entre as Visões.

Tabela: visao

Descrição: Entidade que armazena dados sobre cada Visão.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_visao</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_api</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a <i>Application Programming Interface</i> e a Visão, com origem na tabela <i>api-versao</i> , coluna <i>codigo_api</i> .
<i>codigo_api-versao</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre a Versão da <i>Application Programming Interface</i> e a Visão, com origem na tabela <i>api-versao</i> , coluna <i>codigo_api-versao</i> .
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome original da Visão, conforme descrito na documentação de referência.
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original da Visão (quando disponível), conforme documentação de referência.
<i>url</i>	TEXT	Não	Não	Não		O principal endereço de acesso ao documento da Visão, no formato <i>Uniform Resource Locator</i> .

Tabela: <i>dado_valor</i>						
Descrição: Tabela Auxiliar utilizada para armazenar dados que descrevem os Valores de Dados utilizados por Atributos das Visões ou por Parâmetros das Requisições.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_dado_valor</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome do Valor de Dado esperado para cada dado.
<i>descricao</i>	TEXT	Sim	Não	Não		A descrição das características do Valor de Dado esperado para cada dado.

Fonte: Elaborado pelo Autor, a partir de Rodrigues (2017) e Rodrigues e Sant’Ana (2023).

Todavia, foram consideradas necessidades adicionais para a aplicação do Modelo Entidade-Relacionamento na Modelagem Direta, que permita relacionar os coletores em uma eventual auditoria dos dados, ou seja, possibilite determinar quem são os membros que participaram da coleta dos dados inseridos na Modelagem Direta. Além disso, controlar os coletores é necessário, pois a coleta de dados trata-se de uma atividade passível de colaboração (*e.g.* diferentes membros de uma equipe podem trabalhar ao mesmo tempo para agilizar o procedimento padrão de coleta de dados).

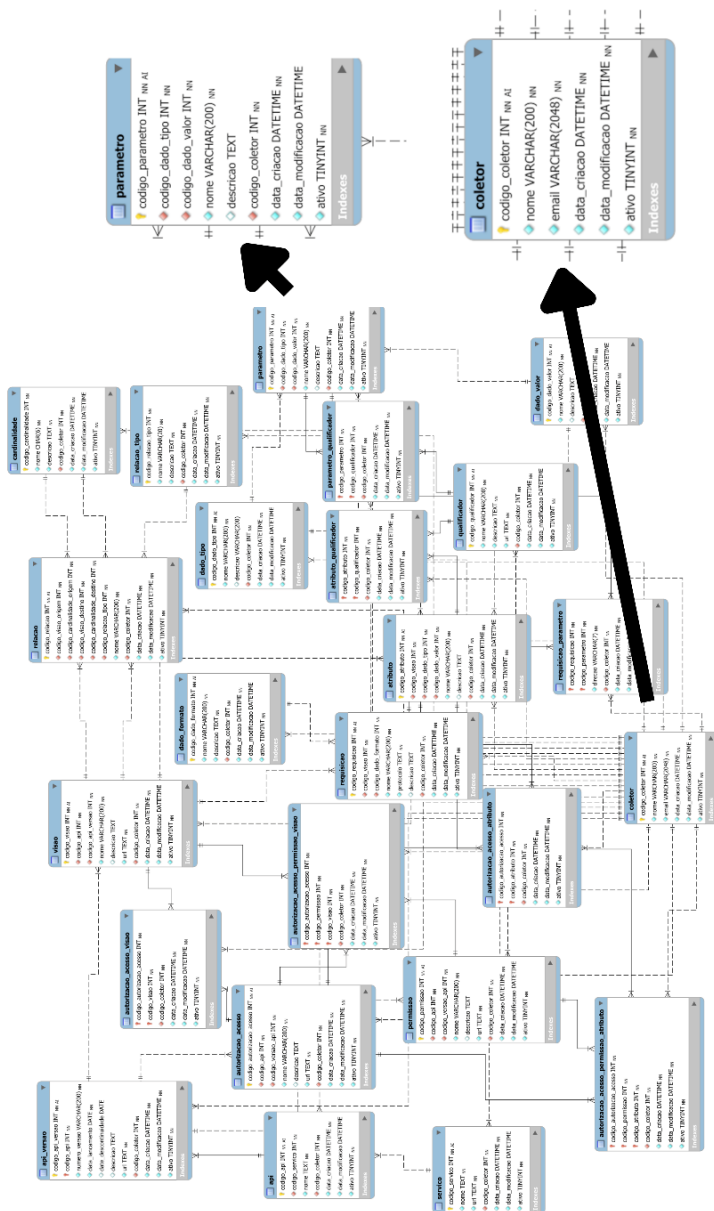
Neste sentido, elaborou-se uma tabela denominada *coletor* e relacionou-a com todas as tabelas da Modelagem Direta, utilizando sua chave primária como chave estrangeira. Também foi necessário inserir colunas extras em todas as tabelas para o controle de datas de criação e de alteração dos registros pelos coletores, bem como a ativação ou inativação dos dados registrados, adotando o princípio de exclusão lógica – no qual nenhuma linha da tabela é excluída, e sim inativada.

A Figura 30 ilustra o Diagrama Entidade-Relacionamento da Modelagem Direta, contendo as modificações para autoria dos dados coletados, além de ampliar a tabela *coletor* e a tabela *parametro*, esta última servindo como exemplo das modificações realizadas em todas as tabelas da Modelagem Direta. A tabela *coletor* poderá receber novas colunas, dependendo das características de cada equipe.

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Figura 30 – Diagrama Entidade-Relacionamento da Modelagem Direta com dados de auditoria



Fonte: Elaborado pelo Autor.

O Quadro 6 exibe informações um recorte do Dicionário de Dados, contendo informações sobre a tabela *coletor* e a tabela *parametro*. Apresenta-se um cabeçalho, contendo o nome da tabela, a descrição de seu conteúdo, seguido de uma descrição das colunas. Para cada coluna é apresentado seu nome, o tipo de dado esperado, o tamanho máximo do valor, se a coluna aceita valores nulos, se a coluna é chave primária ou estrangeira, suas configurações padrão e a descrição do tipo de dado que é esperado para armazenar.

Quadro 6 – Recorte do Dicionário de Dados da Modelagem de Direta com auditoria: tabelas *coletor* e *parametro*

Tabela: <i>parametro</i>						
Descrição: Entidade que armazena dados sobre cada Parâmetro disponível nas Requisições.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_parametro</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_dado_tipo</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre o Parâmetro e o Tipo de Dado para os seus valores, com origem na tabela <i>dado_tipo</i> , coluna <i>codigo_parametro</i> .
<i>codigo_dado_valor</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre o Parâmetro e o Valor de Dado para os seus valores, com origem na tabela <i>dado_valor</i> , coluna <i>codigo_dado_valor</i> .
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome do Parâmetro.
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original do Parâmetro (quando disponível), conforme documentação de referência.
<i>codigo_coletor</i>	INT	Não	Não	Sim		Chave estrangeira do relacionamento entre o Parâmetro e o Coletor para os seus valores, com origem na tabela <i>coletor</i> , coluna <i>codigo_coletor</i> .

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

<i>data_criacao</i>	DATETIME	Não	Não	Não	CURRENT_TIMESTAMP	A data e horário de criação de cada linha da tabela, gerado automaticamente.
<i>data_modificacao</i>	DATETIME	Não	Não	Não	CURRENT_TIMESTAMP	A data e horário de modificação de cada linha da tabela, gerado automaticamente na inserção. O seu valor deve ser alterado para o valor padrão <i>CURRENT_TIMESTAMP</i> no caso de algum dado da linha ser alterado.
ativo	TINYINT	Não	Não	Não	1	Dado representando se os dados da linha da tabela estão ativos ou inativos, gerado automaticamente na inserção (1 - Ativa). Para inativar o coletor, seu valor deve ser alterado para o número zero (0 - Inativa).

Tabela: coletor

Descrição: Entidade que armazena dados sobre os membros da equipe que coletarão os dados sobre os Serviços de Redes Sociais *Online*.

Colunas

Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_coletor</i>	INT	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome completo do coletor.
<i>email</i>	VARCHAR (2048)	Não	Não	Não		O endereço de e-mail do coletor.
<i>data_criacao</i>	DATETIME	Não	Não	Não	CURRENT_TIMESTAMP	A data e horário de criação de cada linha da tabela, gerado automaticamente.
<i>data_modificacao</i>	DATETIME	Não	Não	Não	CURRENT_TIMESTAMP	A data e horário de modificação de cada linha da tabela, gerado automaticamente na inserção. O seu valor deve ser alterado para o valor padrão <i>CURRENT_TIMESTAMP</i> o caso de algum dado da linha ser alterado.

<i>ativo</i>	TINYINT	Não	Não	Não	1	Dado representando se os dados da linha da tabela estão ativos ou inativos, gerado automaticamente na inserção (1 - Ativa). Para inativar o coletor, seu valor deve ser alterado para o número zero (0 - Inativa).
--------------	---------	-----	-----	-----	---	--

Fonte: Elaborado pelo Autor, a partir de Rodrigues (2017) e Rodrigues e Sant’Ana (2023).

A tabela *parametro* exemplifica as colunas adicionadas em todas as tabelas da Modelagem Direta, sendo: *codigo_coletor*, *data_criacao*, *data_modificacao* e *ativo*, respectivamente, armazenando o código de identificação de cada coletor, a data de criação da linha, a data de modificação da linha e se a linha está ativa ou inativa.

2.3 EXEMPLOS DE USO DE DADOS DA MODELAGEM DIRETA

A estrutura proposta para a Modelagem Direta foi instanciada, ou seja, sua estrutura foi aplicada ao Sistema de Gerenciamento de Banco de Dados *Oracle MySQL*. Isso é possível por meio da transformação para das informações do Diagrama Entidade-Relacionamento e do Dicionário de Dados em uma sintaxe válida da Linguagem de Controle de Dados (em inglês, *Data Control Language* – DCL) que é parte integrante da Linguagem Estruturada para Consultas, popularmente conhecida pelo seu nome original, em inglês, *Structured Query Language* (SQL). A Linguagem de Controle de Dados transforma entidades e relações propostas no Diagrama Entidade-Relacionamento e o detalhamento das tabelas e colunas a partir das informações do Dicionário de Dados em códigos compreensíveis pelo Sistema de Gerenciamento de Banco de Dados *Oracle MySQL*.

É possível automatizar esta transformação por meio de aplicativos específicos, tais como o *MySQL Workbench*, desenvolvido para auxiliar na construção de Diagramas Entidade-Relacionamento e de Dicionário de

Dados, e que inclui um módulo específico para este tipo de transformação – via Engenharia de Avanço (*Forward Engineering*). Em geral, os Sistemas de Gerenciamento de Banco de Dados possuem um aplicativo para esta finalidade, seja desenvolvido pelos próprios detentores ou por terceiros.

Para a construção de exemplos da Modelagem Direta proposta neste livro, utilizou-se dados coletados dos Serviços de Redes Sociais *Online Facebook, LinkedIn e X*, especificamente das Versões das APIs analisadas (ver seções 1.3 a 1.5). Os dados foram coletados a partir da aplicação do procedimento padronizado de coleta (ver Figura 26, localizada no início deste capítulo). Em suma, os exemplos aqui descritos são fruto de análise dos dados das próprias APIs analisadas, populados na Modelagem Direta, e que servirão como uma forma de validação da proposta estabelecida.

Os exemplos foram divididos em duas partes. Na primeira parte se utilizou somente instruções de consulta na Modelagem Direta em sintaxe SQL. Estes exemplos têm o intuito de exibir dados quantitativos sobre as APIs. Já a segunda parte está centrada em desenvolver exemplos de visualizações de dados, elaboradas a partir de consulta na Modelagem Direta em sintaxe SQL, com o auxílio de linguagens de programação e bibliotecas de geração de visualização de dados estatísticos⁴.

Não se propõe aqui um esforço em esgotar as possibilidades de consulta a Modelagem Direta, mas sim indicar a potencialidade de consultas ao se estruturar tanto os procedimentos de coleta como os dados coletados.

⁴ Os algoritmos desenvolvidos para os exemplos da Modelagem Direta propostos neste livro podem ser visualizados, utilizados, adaptados e distribuídos pela Licença *Creative Commons 4.0 BY-SA-ND*, disponível no endereço eletrônico <https://github.com/rodriguesprobr/livro-srso>.

Exemplo 22 – Consulta o Total de Visões disponíveis em cada Versão de API, em sintaxe *Structured Query Language*

```

SELECT
  api.nome AS 'Nome da API',
  api_versao.numero_versao AS 'Número da Versão',
  (
    SELECT
      COUNT(*)
    FROM visao
    WHERE visao.codigo_api = api.codigo_api
      AND visao.codigo_api_versao = api_versao.codigo_api_versao
  ) AS `Total de Visões`
FROM api_versao
INNER JOIN api ON api.codigo_api = api_versao.codigo_api
ORDER BY api.nome ASC, api_versao.numero_versao ASC;

```

Fonte: Elaborado pelo Autor.

No primeiro exemplo, estabelece-se uma consulta a Modelagem Direta para verificar o total de visões disponíveis em cada versão de API. O Exemplo 22 exhibe as codificações em sintaxe SQL para esta consulta, com as seguintes características: as colunas *api.nome* e *api_versao.numero_versao* foram alteradas para que os resultados apresentem nomes mais humanizados, respectivamente Nome da API e Número da Versão. Foi necessário realizar uma operação de junção em álgebra relacional do tipo junção interna (*INNER JOIN*) entre as tabelas *api_versao* e *api* (Silberschatz; Korth; Sudarshan, 2020). Os resultados da consulta foram classificados pelo Nome da API e pelo Número da Versão, em ordem crescente (no caso, em ordem alfabética).

Tabela 1 – Modelagem Direta: Total de Visões disponíveis em cada Versão de API

Nome da API	Número da Versão	Total de Visões
<i>Facebook Graph</i> API	2.6	291
<i>LinkedIn</i> REST API	1.0	21
<i>X</i> REST API	1.1	74

Fonte: Elaborado pelo Autor.

A Tabela 1 exibe os resultados desta consulta a Modelagem Direta, a partir dos dados populados. Como é possível observar, a Tabela 1 apresenta o total de Visões disponíveis em cada Versão de API. Os dados recuperados na Modelagem Direta foram iguais aos apresentados no primeiro capítulo. Isso é importante para verificar que o procedimento padronizado de coleta – e o uso da Modelagem Direta – não só pode facilitar a coleta dos dados das APIs, mas também permite uma comparabilidade direta entre APIs de diferentes Serviços de Redes Sociais *Online*, corroborando com o objetivo proposto neste livro, de permitir este tipo de análise.

Exemplo 23 – Consulta o Total de Atributos disponíveis em cada Visão, para cada Versão de API, em sintaxe *Structured Query Language*, com recorte as Visões que apresentam 50 ou mais atributos

```
SELECT
  api.nome AS 'Nome da API',
  api_versao.numero_versao AS 'Número da Versão',
  visao.nome AS 'Nome da Visão',
  (
    SELECT
      COUNT(*)
    FROM atributo
    WHERE atributo.codigo_visao = visao.codigo_visao
  ) AS 'Total de Atributos'
FROM api_versao
INNER JOIN api ON api.codigo_api = api_versao.codigo_api
INNER JOIN visao ON visao.codigo_api = api.codigo_api AND visao.codigo_api_versao =
api_versao.codigo_api_versao
WHERE
  (
    SELECT
      COUNT(*)
    FROM atributo
    WHERE atributo.codigo_visao = visao.codigo_visao
  ) >= 50
ORDER BY 'Total de Atributos' DESC, api.nome ASC, api_versao.numero_versao ASC, visao.
nome ASC;
```

Fonte: Elaborado pelo Autor.

Em sequência, propõe-se uma nova consulta na Modelagem Direta, para verificar o total de Atributos disponíveis em cada Visão, segmentados em cada Versão de API. O Exemplo 23 exhibe as codificações em sintaxe SQL para realizar esta consulta, com as seguintes características: as colunas *api.nome*, *api_versao.numero_versao* e *visao.nome* foram alteradas para que os resultados apresentem nomes mais humanizados, respectivamente Nome da API, Número da Versão e Nome da Visão; uma quarta coluna foi desenvolvida, com o nome Total de Atributos, sendo que os valores apresentados nesta coluna são fruto de uma subconsulta (*nested query*), com a finalidade de contabilizar o total de Atributos para cada Visão (Silberschatz; Korth; Sudarshan, 2020).

Foi necessário realizar duas operações de junção em álgebra relacional do tipo junção interna (*INNER JOIN*) entre as tabelas das dimensões *api_versao* e *api*, e *api_versao* e *visao*. Os resultados da consulta foram classificados pelo Total de Atributos, em ordem decrescente (ordenados do maior valor ao menor valor), pelo Nome da API, Número da Versão e Nome da Visão, todos em ordem crescente e em ordem alfabética.

Também foi utilizado um valor mínimo como recorte amostral: a consulta somente retornará Visões que possuam um total de 50 ou mais Atributos.

Tabela 2 – Modelagem Direta: Total de Atributos disponíveis em cada Visão, para cada Versão de API, com recorte as Visões que apresentam 50 ou mais atributos ($n \geq 50$)

Nome da API	Número da Versão	Nome da Visão	Total de Atributos
<i>Facebook Graph</i> API	2.6	<i>Page</i>	112
<i>XREST</i> API	1.1	<i>application/rate_limit_status</i>	99
<i>Facebook Graph</i> API	2.6	<i>FacebookApp</i>	81
<i>LinkedIn REST</i> API	1.0	<i>Member</i>	68
<i>XREST</i> API	1.1	<i>account/verify_credentials</i>	57
<i>Facebook Graph</i> API	2.6	<i>User</i>	53

Fonte: Elaborado pelo Autor.

A Tabela 2 exhibe os resultados da consulta a Modelagem Direta, a partir dos dados populados. Neste caso, é observável que as APIs analisadas possuem certas Visões centrais, com um número considerável de atributos – três relacionadas ao Serviço de Rede Social *Online Facebook* (*Page*, *FacebookApp* e *User*), duas relacionadas ao Serviço de Rede Social *Online X* (*application/rate_limit_status* e *account/verify_credentials*) e uma relacionada ao Serviço de Rede Social *Online LinkedIn* (*Member*) – corroborando com o senso comum que dados de perfis de Referenciados e de páginas possuem um alto grau de detalhamento (por meio de uma quantidade significativa de Atributos) e são elementos centrais destes serviços.

Exemplo 24 – Consulta a Quantidade de Visões disponíveis em cada conjunto de Autorização de Acesso e de Permissão, para cada Versão de API dos Serviços de Redes Sociais *Online Facebook* e *LinkedIn*, em sintaxe *Structured Query Language*

```
SELECT
    api.nome AS 'Nome da API',
    api-versao.numero-versao AS 'Número da Versão',
    autorizacao-acesso.nome AS 'Autorização de Acesso',
    permissao.nome AS 'Permissão',
    count(codigo-visao) AS 'Quantidade de Visões Acessíveis'
FROM autorizacao-acesso-permissao-visao
INNER JOIN autorizacao-acesso ON autorizacao-acesso.codigo-autorizacao-acesso =
    autorizacao-acesso-permissao-visao.codigo-autorizacao-acesso
INNER JOIN permissao ON permissao.codigo-permissao = autorizacao-acesso-permissao-
    visao.codigo-permissao
INNER JOIN api-versao ON api-versao.codigo-api-versao = autorizacao-acesso.codigo-api-
    versao
INNER JOIN api ON api.codigo-api = api-versao.codigo-api
GROUP BY 'Nome da API', 'Número da Versão', 'Autorização de Acesso', 'Permissão'
ORDER BY 'Quantidade de Visões Acessíveis' DESC;
```

Fonte: Elaborado pelo Autor.

Já o Exemplo 24 aborda uma outra perspectiva (codificada em sintaxe SQL), a de estabelecer uma consulta para verificar a quantidade Visões

que estão disponíveis com o conjunto formado entre as combinações de Autorizações de Acesso e de Permissões nas Versões de API dos Serviços de Redes Sociais *Online Facebook* e *LinkedIn* – lembrando que o Serviço de Rede Social *Online X* não utiliza este tipo de sistema de entrada (somente Autorizações de Acesso).

As colunas *api.nome* (com origem na API), *api_versao.numero_versao* (com origem na Versão da API), *autorizacao_acesso.nome* (com origem na Autorização de Acesso) e *permissao.nome* (com origem na Permissão) foram alteradas para que os resultados apresentem nomes mais humanizados, respectivamente Nome da API, Número da Versão, Autorização de Acesso e Permissão; uma quinta coluna foi desenvolvida, com o nome Quantidade de Visões Acessíveis, sendo que os valores apresentados nesta coluna são fruto do uso de uma instrução de contabilização (*count*), com a finalidade de totalizar as Visões acessíveis, a partir das ocorrências de valores distintos da coluna *codigo_visao* (Silberschatz; Korth; Sudarshan, 2020).

Foi necessário realizar quatro operações de junção em álgebra relacional do tipo junção interna (*INNER JOIN*) entre as tabelas *autorizacao_acesso_permissao_visao* e *autorizacao_acesso*; *autorizacao_acesso_permissao_visao* e *permissao*; *autorizacao_acesso* e *api_versao*, e; *api_versao* e *api* (Silberschatz; Korth; Sudarshan, 2020).

Para que fosse possível a utilização da instrução de contabilização do total de Visões acessíveis, pelo conjunto formado entre as combinações de Autorizações de Acesso e de Permissões, foi estabelecida uma função de agrupamento (*GROUP BY*) das colunas Nome da API, Número da Versão, Autorização de Acesso e Permissão (Silberschatz; Korth; Sudarshan, 2020).

Os resultados da consulta foram classificados pela Quantidade de Visões Acessíveis, em ordem decrescente (ordenados do maior valor ao menor valor).

Estruturas de dados em serviços de redes sociais online
 Uma abordagem metodológica de análise

Tabela 3 – Modelagem Direta: Quantidade de Visões disponíveis em cada conjunto de Autorização de Acesso e de Permissão, para cada Versão de API dos Serviços de Redes Sociais *Online Facebook e LinkedIn*

Nome da API	Número da Versão	Autorização de Acesso	Permissão	Quantidade de Visões Acessíveis
Facebook Graph API	2.6	User Access Token	user_tagged_places	6
Facebook Graph API	2.6	User Access Token	publish_actions	5
Facebook Graph API	2.6	User Access Token	user_actions:{app_namespace}	3
Facebook Graph API	2.6	User Access Token	user_posts	3
Facebook Graph API	2.6	User Access Token	user_photos	3
Facebook Graph API	2.6	User Access Token	user_managed_groups	3
Facebook Graph API	2.6	User Access Token	user_games_activity	3
Facebook Graph API	2.6	User Access Token	user_events	3
Facebook Graph API	2.6	User Access Token	user_videos	3
Facebook Graph API	2.6	User Access Token	user_actions.video	3
Facebook Graph API	2.6	User Access Token	user_actions.music	3
Facebook Graph API	2.6	User Access Token	rsvp_event	3
Facebook Graph API	2.6	User Access Token	read_insights	2
Facebook Graph API	2.6	Page Access Token	manage_pages	2
Facebook Graph API	2.6	Page Access Token	pages_manage_instant_articles	2
Facebook Graph API	2.6	User Access Token	user_relationships	2
Facebook Graph API	2.6	User Access Token	user_likes	2
Facebook Graph API	2.6	Page Access Token	pages_messaging	2
Facebook Graph API	2.6	Page Access Token	pages_messaging_phone_number	2
Facebook Graph API	2.6	User Access Token	user_friends	2
Facebook Graph API	2.6	Page Access Token	pages_show_list	2
Facebook Graph API	2.6	User Access Token	user_education_history	2
Facebook Graph API	2.6	User Access Token	user_actions.news	2
Facebook Graph API	2.6	User Access Token	user_actions.fitness	2
Facebook Graph API	2.6	User Access Token	user_actions.books	2
Facebook Graph API	2.6	Page Access Token	read_page_mailboxes	2
Facebook Graph API	2.6	User Access Token	user_work_history	2
Facebook Graph API	2.6	User Access Token	read_custom_friendlists	2
LinkedIn REST API	1.0	Access Token	r_emailaddress	1
LinkedIn REST API	1.0	Access Token	r_contactinfo	1
LinkedIn REST API	1.0	Access Token	r_fullprofile	1
LinkedIn REST API	1.0	Access Token	rw_company_admin	1
Facebook Graph API	2.6	User Access Token	user_website	1
LinkedIn REST API	1.0	Access Token	r_basicprofile	1
Facebook Graph API	2.6	User Access Token	user_religion_politics	1
Facebook Graph API	2.6	User Access Token	email	1
Facebook Graph API	2.6	User Access Token	user_relationship_details	1
Facebook Graph API	2.6	User Access Token	user_location	1
Facebook Graph API	2.6	Page Access Token	user_likes	1

Nome da API	Número da Versão	Autorização de Acesso	Permissão	Quantidade de Visões Acessíveis
<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>user_hometown</i>	1
<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>user_birthday</i>	1
<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>user_about_me</i>	1
<i>Facebook Graph API</i>	2.6	<i>Page Access Token</i>	<i>read_insights</i>	1
<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>read_audience_network_insights</i>	1
<i>Facebook Graph API</i>	2.6	<i>Page Access Token</i>	<i>publish_pages</i>	1
<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>publish_pages</i>	1
<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>public_profile</i>	1
<i>Facebook Graph API</i>	2.6	<i>Page Access Token</i>	<i>pages_manage_cta</i>	1

Fonte: Elaborado pelo Autor.

A Tabela 3 exhibe os resultados da consulta a Modelagem Direta, a partir dos dados populados. Neste caso, é possível perceber que as duas APIs analisadas possuem certos conjuntos de Visões com a finalidade de serem os pontos de acesso iniciais aos dados disponíveis nos Serviços de Redes Sociais *Online*. Por exemplo, no Serviço de Rede Social *Online Facebook*, seis Visões são acessíveis quando se utiliza a Autorização de Acesso *User Access Token* e a Permissão *user_tagged_places*. Já no Serviço de Rede Social *Online LinkedIn*, uma Visão fica acessível quando se utiliza a Autorização de Acesso *Access Token* e a Permissão *r_emailaddress*.

A partir dos dados das Tabelas 1 a 3, é possível ter uma percepção sobre as possibilidades quantitativas que a estruturação dos dados oferece ao coletor. A partir dos dados populados, em termos quantitativos, os dados do Serviço de Rede Social *Online Facebook* são mais expressivos que os demais, o que corrobora o senso comum que de fato é esse serviço não só um maior número de usuários ativos – conforme visto na introdução – mas também um serviço com maior número de funcionalidades disponíveis e que, portanto, exige mais entidades e mais relacionamentos nas estruturas de dados, aumentando as possibilidades de pontos de acesso a partir das Autorizações de Acesso e das Permissões.

Em um segundo momento, exemplifica-se possibilidades de analisar os dados da Modelagem Direta por meio de visualizações de dados. Para

isto, utilizou-se a linguagem de programação *Python* associado ao aplicativo de construção de grafos *GraphViz*, além de consultas codificadas em sintaxe SQL para acessar os dados da Modelagem Direta. É importante ressaltar que a linguagem de programação *Python* possui bibliotecas que integram seus algoritmos ao Sistema de Gerenciamento de Banco de Dados *Oracle MySQL* e ao *GraphViz*.

A Figura 31 apresenta uma possibilidade de visualização dos dados da Modelagem Direta, ao ilustrar graficamente as possíveis relações internas da *LinkedIn* REST API do Serviço de Rede Social *Online LinkedIn*, suas Versões de API, Autorizações de Acesso e Permissões disponíveis, Visões acessíveis pelo conjunto formado entre as combinações de Autorizações de Acesso e de Permissões, bem como outras Visões que possam ser acessadas por meio de Relações entre Atributos das Visões⁵.

⁵ O algoritmo utilizado para desenvolver o grafo pode ser visualizado no endereço eletrônico <https://github.com/rodriguesprobr/livro-srso>. Está sob Licença *Creative Commons 4.0 BY-SA-ND*.

A partir da visualização, é possível estabelecer relações entre informações do grafo gerado com os dados da Tabela 2 e 3. No grafo, a Visão *Member* é o principal ponto de acesso, além de ser a Visão que mais possui relações com as demais visões. De acordo com os dados da Tabela 2, a visão *Member* também é a que possui maior número de atributos – um total de 68 – além de ser a principal forma de acesso das permissões existentes.

Ao cruzar informações do grafo com dados da Tabela 3, verifica-se que das cinco permissões disponíveis pela Autorização de Acesso *Access Token*, quatro dão acesso aos dados da Visão *Member*: *r_emailaddress*, *r_contactinfo*, *r_fullprofile*, *user_website* e *r_basicprofile* – também reforçando a importância da Visão *Member*, que congrega dados dos Referenciados, como principal ponto de acesso a dados na API.

Este exemplo ilustra a importância da complementariedade da visualização de dados aos dados qualitativos analíticos no contexto de análise das APIs, por meio da Modelagem Direta, no qual existe uma indissociabilidade entre dados quantitativos e qualitativos. Todavia, aumentar o escopo de análise pode causar uma dificuldade de visualização por parte do coletor, mesmo com as novas possibilidades propiciadas ao estruturar a coleta de dados em uma Modelagem Direta.

A Figura 32 é um exemplo deste tipo de dificuldade a ser enfrentada, no qual foram adicionados dados das três Versões de API em um mesmo grafo, exibindo Visões disponíveis, seus Atributos e Relações.

Começa a ficar humanamente impraticável a geração de certos tipos de visualização devido a escala. A Figura 32 possui um tamanho de 6,2 metros de largura por 5,2 metros de altura (17.829 *pixels* de largura por 14.861 *pixels* de altura), quando utilizado fontes em tamanho 10 para nomes das Visões e tamanho oito para nomes de Atributos, além de bordas com um *pixel* de espessura. Para ilustrar a grandeza dada, a Figura 32 apresenta uma ampliação de parte de seu conteúdo, especificamente a região no qual se concentra a Visão *Users* da X REST API, com dados parciais de seus Atributos e de suas Relações.

Escalas de grandeza deste porte também trazem outras dificuldades. Por exemplo, se adicionar outras informações à visualização, tais como a relação entre Autorizações de Acesso, Permissões e as Visões disponíveis, poderão causar um excesso informacional, e deixar opaco ao coletor informações importantes, tais como um determinado acesso a dados ou mesmo caminhos de acesso possíveis por meio de relações entre Visões de destino que ficam acessíveis por meio de Visões de origem.

Portanto, deixar disponível ao coletor possibilidades de consulta a partir de dados da Modelagem Direta abre novas possibilidades de análises sobre as estruturas de dados das APIs, porém acarreta desafios complexos no processo de identificação de aspectos relacionados aos dados que circulam nestes serviços, principalmente em função do número de funcionalidades, de Visões, de Atributos e demais características do contexto.

Propõe-se uma segunda etapa de análise, denominada Modelagem de Segunda Ordem, que é alimentada pelas características identificadas na Modelagem Direta das APIs analisadas, propondo um novo ciclo de vida de dados (com fases de coleta, armazenamento e recuperação), iniciando-se a partir dos dados da Modelagem Direta, para permitir ao coletor o desenvolvimento de outras formas de análise, por meio da aplicação de técnicas de análise de dados voltadas ao *Business Intelligence*.

3

**MODELAGEM DE
SEGUNDA ORDEM**

MODELAGEM DE SEGUNDA ORDEM

O problema mais complexo da realização de análises das estruturas de dados das APIs dos Serviços de Redes Sociais *Online*, por meio de consulta aos dados coletados e armazenados na Modelagem Direta, seja por consultas com o uso exclusivo de SQL ou com o auxílio de linguagens de programação e de bibliotecas para visualização de dados, está na fase de recuperação.

Por mais que a Modelagem Direta e o procedimento padronizado de coleta de dados sejam os elementos centrais para a aplicabilidade desta proposta – ao possibilitar uma análise com rigor científico, permitindo a elaboração de consultas de comparabilidade entre APIs e de comparabilidade de Versões de uma mesma API ou da construção de visualizações para, por exemplo, verificar quais são as Visões e Atributos que ficam disponíveis em cada conjunto de Autorização de Acesso e Permissão – o sucesso de sua aplicação resulta em problemas de desenho de modelos de dados complexos. São quatro pontos importantes a serem apresentados.

Somente com a inserção de dados das estruturas das três APIs da amostra, a Modelagem Direta já possui uma quantidade significativa de dados passíveis de serem analisados. Por exemplo, a Figura 32, que exempli-

fica consulta contendo as relações entre API, Versões de API, Autorizações de Acesso, Permissões, Visões (e suas relações) para o Serviço de Rede Social *Online LinkedIn*, em formato de grafo de vértices e arestas, é a menor da amostra. Se a mesma consulta for executada para a API do Serviço de Rede Social *Online Facebook*, a visualização imagética não seria possível para ser inserida neste livro, devido ao tamanho da imagem necessário para atender a complexidade dada pela quantidade de dados das entidades selecionadas.

O segundo ponto é a quantidade de entidades diferentes na Modelagem Direta. São 24 entidades (lembrando que o Sistema de Gerenciamento de Bando de Dados as intitula como tabelas) na Modelagem Direta, em que muitas consultas terão o critério de cruzar dados de duas ou mais entidades.

Por exemplo, a partir dos dados das estruturas das três APIs da amostra, existem sete Autorizações de Acesso, 386 Visões e 2.269 Atributos. Um coletor pode decidir verificar quais Atributos (a) são ou não são acessíveis pelas Visões (v), utilizando cada uma das Autorizações de Acesso (aa), estabelecendo um critério de independência da relação entre Atributos e Visões, a partir da premissa que Atributos de uma Visão de destino pode ser acessado por Relação com uma Visão de origem.

Essa verificação pode ser representada pela seguinte expressão

$$T_p = aa_x \times v_y \times a_z$$

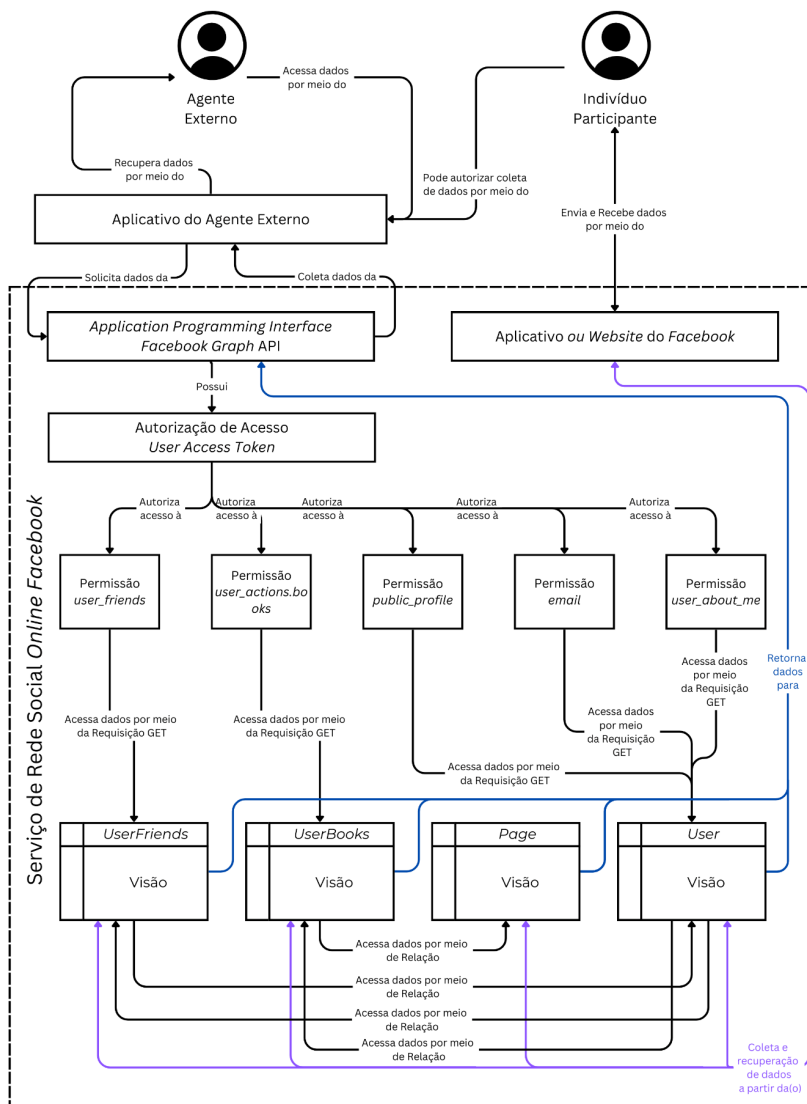
onde T significa o total de possibilidades p de acesso aos Atributos x , por meio das Visões y e pelas Autorizações de Acesso z . Na prática, para as três APIs da amostra, T teria um valor de 6.130.838 possibilidades, sendo que em cada possibilidade geraria uma linha com quatro colunas (Autorização de Acesso, Visão, Atributo e Acesso), no qual a coluna Acesso poderia ter como resultado um valor verdadeiro (Atributo acessível) ou um valor falso (Atributo inacessível), a partir da lógica booleana.

O terceiro ponto, relacionado ao exemplo acima, é a escalabilidade da Modelagem Direta. A escalabilidade está relacionada a capacidade da Modelagem Direta em crescer para atender as demandas de análise. Se o exemplo do segundo ponto apresentou mais de seis milhões de resultados para uma fórmula relativamente simples, ao adicionar dados de estruturas outras APIs – ou de outras Versões de uma API já coletada – ocasionará em maior dificuldade em visualizar dados, ao longo do tempo.

Para finalizar, o quarto ponto é que a própria capacidade humana em visualizar informações é limitada. Para exemplificar essa dificuldade em elaborar visualizações do processo voltadas para a análise por seres humanos, utiliza-se, como exemplo, a construção de elementos para a visualização de quais Atributos e Visões estão acessíveis por uma única Autorização de Acesso, por meio do uso de uma visualização do tipo *crosswalk* – em que o desenvolvimento da visualização parte de um objeto ou de uma situação como ponto de partida e, posteriormente, se vão interligando pontos que poderão ser transitados a partir do ponto inicial – e que, no contexto deste livro, pode se tornar complexa, mesmo se for aplicado um recorte dos conjuntos de dados que estão disponíveis na Modelagem Direta.

A Figura 33 ilustra um recorte temático da *Facebook Graph* API, apresentando o caminho do acesso de um Agente Externo as Visões *User*, *UserBooks*, *UserFriends* e *Page* por meio da Autorização de Acesso *User Access Token* e as permissões *user_friends*, *user_actions.books*, *public_profile*, *email* e *user_about_me*. Os dados utilizados para a elaboração da ilustração foram coletados nas entidades disponíveis na Modelagem Direta.

Figura 33 – Exemplo de visualização *crosswalk* de coleta de dados pela *Facebook Graph API*, com o uso da Autorização de Acesso *User Access Token*



Fonte: Elaborado pelo Autor.

O ponto de entrada determinado foi um Agente Externo, externo ao Serviço de Rede Social *Online*, e as relações entre a Autorização de Acesso, as Permissões, e entre as Permissões e as Visões e seus relacionamentos estão explicitadas com vetores de linhas pretas. O retorno dos dados para a API – e conseqüentemente para o Agente Externo – estão explicitados por vetores na cor azul. Já o caminho dos dados coletados e recuperados pelos Referenciados está explicitado por vetores na cor lilás.

No exemplo, o Agente Externo elabora um aplicativo externo ao Serviço de Rede Social *Online Facebook*, e os Referenciados podem conceder o acesso aos seus conjuntos de dados. Quando o aplicativo acessa a *Facebook Graph* API com o uso *token* de Autorização de Acesso *User Access Token*, as permissões escolhidas no recorte permitem executar requisições de coleta de dados das seguintes visões:

- a) *User*: com o uso das permissões *public_profile*, *email* e *user_about_me*;
- b) *UserFriends*: com o uso da permissão *user_friends*;
- c) *User*, a partir do relacionamento de *UserFriends*: indiretamente acessível pela permissão *user_friends*, pois o atributo *data* da visão *UserFriends* retorna como valor uma lista com linhas da visão *User*, para relacionar os amigos de um Referenciado;
- d) *UserFriends*, a partir do relacionamento de *User*: indiretamente acessível pela permissão *public_profile*, pois a aresta *friends* da visão *User* retorna como valor uma lista com linhas da visão *UserFriends*, para relacionar os amigos de um Referenciado;
- e) *UserBooks*: com o uso da permissão *user_actions.books*;
- f) *Page*: indiretamente acessível pela permissão *user_actions.books*, pois o atributo *data* da visão *UserBooks* retorna como valor uma lista com linhas da visão *Page*, para relacionar as páginas no *Facebook* que representam os livros que um Referenciado curtiu.

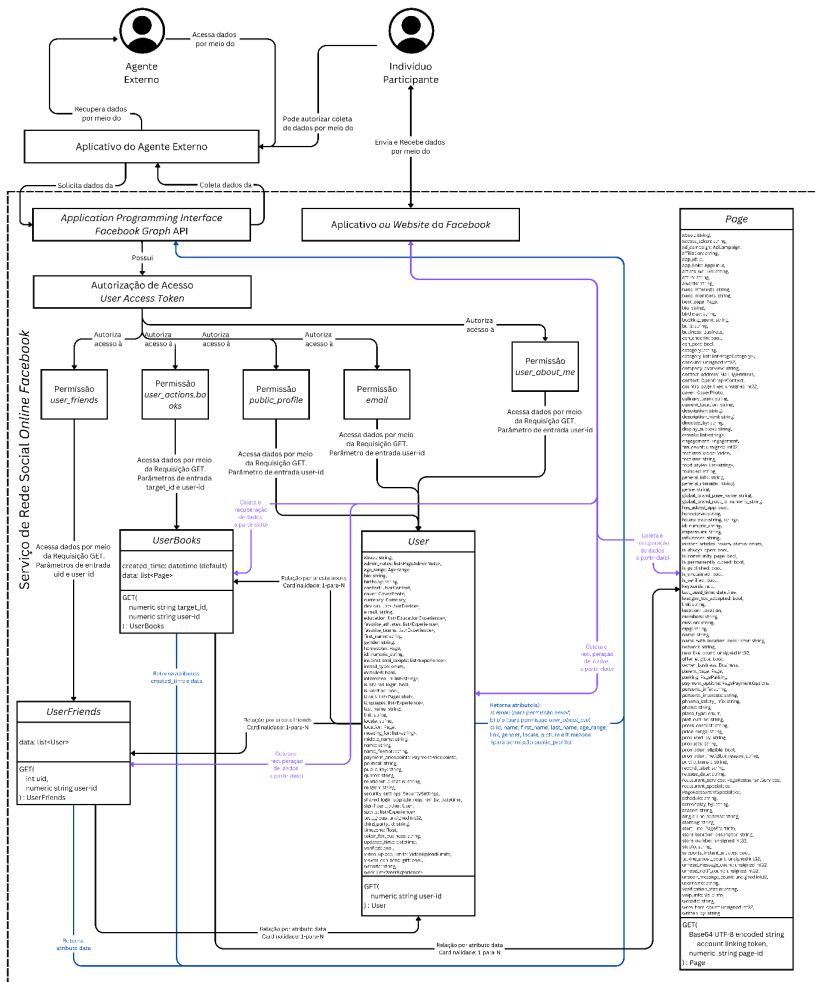
Para um maior detalhamento das características deste tipo de coleta por Agente Externo, é necessário explicitar mais elementos para este recorte, como quais são os Atributos que cada Visão possui e quais são os Atributos retornados em cada Requisição de coleta, bem como os Parâmetros de entrada para as Requisições.

Portanto, a Figura 34 – que é complementar a Figura 33 – exhibe informações sobre as visões *User*, *UserBooks*, *UserFriends* e *Page* e como elas se relacionam com o *token* de Autorização de Acesso *User Access Token* e com as permissões *user_friends*, *user_actions.books*, *public_profile*, *email* e *user_about_me*. Também são explicitadas a cardinalidade dos relacionamentos entre as Visões, o tipo de Relação, o nome dos Atributos ou da Aresta, as Requisições disponíveis, incluindo os Parâmetros de entrada e os Atributos recuperados em cada Requisição.

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Figura 34 – Exemplo de visualização *crosswalk* de coleta de dados pela Facebook Graph API, com o uso da Autorização de Acesso *User Access Token*, exibindo atributos, relacionamentos e cardinalidades



Fonte: Elaborado pelo Autor.

As relações entre a Autorização de Acesso, as Permissões, e entre as Permissões e as Visões e seus relacionamentos estão explicitadas com vetores de linhas pretas. Neste exemplo, os vetores passaram a incluir os

Parâmetros de entrada necessários para a execução da coleta de dados. No caso das Relações entre Visões, foram acrescentados aos vetores o tipo de Relação, bem como sua cardinalidade. O retorno dos dados para a API – e consequentemente para o Agente Externo – estão explicitados por vetores na cor azul – incluindo os Atributos que serão coletados pelo Agente Externo por meio da Permissão e da Autorização de Acesso vinculadas.

Já o caminho dos dados coletados e recuperados pelos Referenciados está explicitado por vetores na cor lilás. O caminho de acesso do Agente Externo e as Relações entre as Visões são os mesmos. Todavia, foi acrescentado todos os Atributos de cada Visão, bem como os Tipos de Dado de cada Atributo, bem como a Requisição e os Parâmetros de entrada necessários para acesso aos dados de cada Visão, identificados no rodapé das Visões.

É importante ressaltar que a Figura 34 apresenta o caminho de acesso de uma única Autorização de Acesso e de uma única Versão de API. Ao ampliar os limites destes modelos de visualização – processando um maior número de Autorizações de Acesso e de Permissões disponíveis com a totalidade das Visões e dos Atributos disponíveis – o volume e a variedade de informações a serem exibidas seria suficiente para ofuscar ao coletor detalhes de sua estrutura, dificultando ou mesmo impossibilitando a realização de uma análise a partir deste cenário.

Aqui se estabelece um critério desejável à Modelagem Direta: de permitir a construção de novos bancos de dados a partir dos dados da Modelagem Direta, ou seja, novas camadas de interpretação. As camadas de interpretação podem utilizar dados originários da Modelagem Direta para a construção de novos bancos de dados para a utilização de algoritmos que auxiliem o coletor em processar análises de forma semiautomática ou automática.

Para resolver o problema da dificuldade de construção de um processo de visualização a partir dos dados coletados na Modelagem Direta, propõe-se uma nova camada de interpretação, denominada Modelagem de Segunda Ordem. A Modelagem de Segunda Ordem se apropria de uma

forma de análise de dados consagrada tanto na área de negócios como na ciência: o *Data Warehouse* e o *Data Mart*, que são derivados de técnicas relacionadas ao tema de *Business Intelligence* – comumente utilizadas para a construção de análises de dados voltadas para tomada de decisão.

3.1 DATA WAREHOUSE, DATA MART E BUSINESS INTELLIGENCE NA ANÁLISE DE CONJUNTOS DE DADOS NO CONTEXTO DOS SERVIÇOS DE REDES SOCIAIS ONLINE

O *Business Intelligence* está relacionado com a necessidade de gestores tomarem decisões a partir de dados, associados à sua capacidade crítica de tomada de decisão; ao uso de Sistema de Gerenciamento de Bancos de Dados e com a construção de banco de dados compatíveis aos conceitos de *Data Warehouse*, para geração de relatórios e análises; e para processos de descobrimento de padrões nos conjuntos de dados armazenados (Barbieri, 2011; Liautaud, 2000; Maribel; Ramos, 2009; Watson; Wixom, 2007).

O *Data Warehouse* é um banco de dados integrado, orientado a assuntos, variável no tempo e não-volátil para auxiliar ao processo de tomada de decisão. A estrutura de sua modelagem não é voltada ao uso e apoio as atividades diárias de uma instituição, comumente utilizadas para o armazenamento de dados destas atividades na forma de dados de transação (denominados como dados transacionais). Exemplos de dados transacionais vão desde dados de uma venda em um estabelecimento comercial até dados da postagem de um Referenciado em um Serviço de Rede Social *Online*.

No caso do *Data Warehouse*, o foco é diferente. A estrutura de sua modelagem em nível lógico tem foco específico para a realização de consultas, e análises orientadas a uma demanda de análise específica, organizada por dimensões (Inmon, 2005; Kimball; Ross, 2011).

Para a elaboração de uma modelagem organizada de forma dimensional, é importante a identificação de dois elementos de base: o fato e a dimensão. O *Data Warehouse* é formado por diversas tabelas (entidades)

em um Sistema de Gerenciamento de Bancos de Dados, aonde as tabelas representam fatos ou dimensões – e cada conjunto interligado de fatos e dimensões é denominado *Data Mart* (Barbieri, 2011; Inmon, 2005; Kimball; Ross, 2011; Maribel; Ramos, 2009).

O *Data Mart* é um conjunto de dados flexível (em seu estado mais granular¹), extraídos de um banco de dados transacional e apresentados em um modelo dimensional mais adaptável a consultas do que em modelagens de banco de dados voltadas a operacionalização de atividades transacionais. Cada *Data Mart* pode representar dados de apenas um único processo de negócio de uma instituição, por exemplo, uma venda ou uma postagem (Kimball; Ross, 2011).

A primeira etapa para a construção de um *Data Mart*, é a identificação de atributos das tabelas que sejam fatos. A tabela de fatos (tabela fato ou fato) é a tabela primária, central, e é ponto de partida na construção de um modelo dimensional, onde devem ser armazenados apenas valores quantificáveis, por exemplo, os totais de cada venda de um estabelecimento comercial ou a total de postagens, e as chaves estrangeiras das tabelas de dimensões.

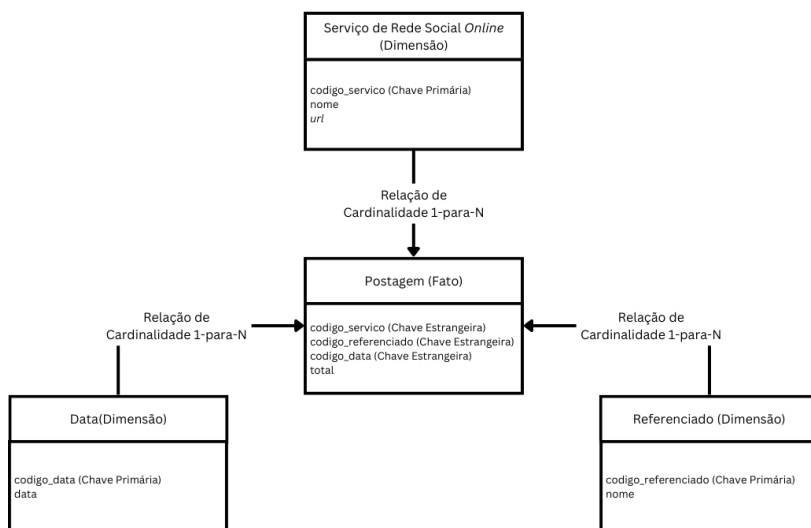
As tabelas de dimensões também são componentes essenciais para o funcionamento de um *Data Mart*, contendo as descrições textuais para cada uma das chaves estrangeiras identificadas na tabela fato. Portanto, as chaves estrangeiras da tabela fato devem se relacionar com uma coleção de tabelas de dimensão. As dimensões são as interfaces para a realização de consultas aos conjuntos de dados registrados em um *Data Mart* e as colunas dessas tabelas servem como recurso primário de pesquisa e refinamento para consultas a serem realizadas, são os pontos de acesso as consultas (Kimball; Ross, 2011).

O relacionamento das tabelas que representam as dimensões com a tabela de fato gera um modelo dimensional, que pode ser desenvolvido

¹ A granularidade mais fina – em seu estado mais granular possível – refere-se à incapacidade de dividir os valores contidos nas colunas de um conjunto de dados em outros conjuntos, por se tratarem de dados primários, em seu estado mais bruto (Santos; Sant’ana, 2015).

com diversas técnicas, que variam de acordo com o tipo de necessidade e o contexto de análise. O modelo de representação que contém a tabela de fato ao centro e as tabelas de dimensões dispostas no seu entorno é conhecido como esquema estrela (no idioma inglês, *star scheme*).

Figura 35 – Exemplo de um modelo dimensional em esquema estrela

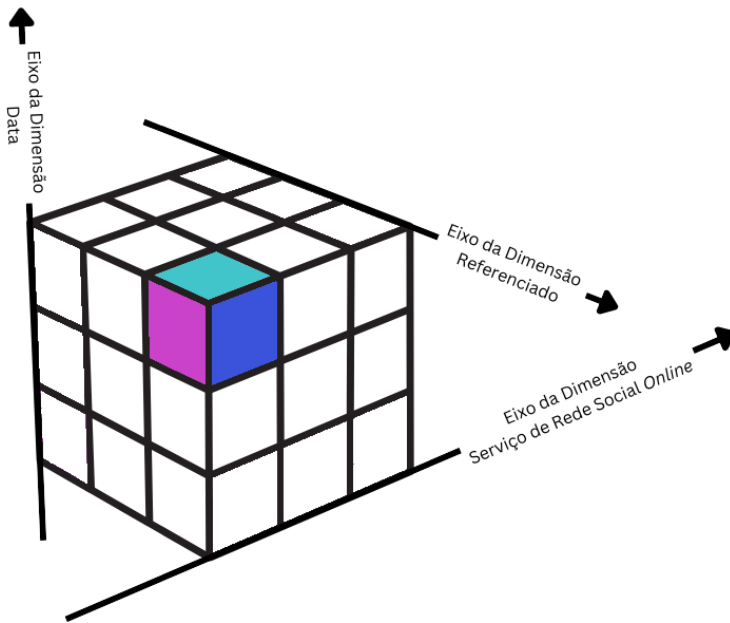


Fonte: Elaborado pelo Autor.

No exemplo ilustrado na Figura 35, o esquema estrela apresenta um *Data Mart* para o fato *Postagem*, em que são possíveis a realização de análises dos dados a partir das dimensões *Serviço de Rede Social Online*, *Data* (da *Postagem*) e *Referenciado*. Com esta disposição de dimensões, um indivíduo com acesso ao *Data Mart* – e com os conhecimentos prévios necessários para a operacionalização – pode, *a priori*, popular o *Data Mart* a partir de dados transacionais de postagens sobre um determinado assunto (*e.g.* uma campanha política, uma campanha de divulgação de um produto e um novo serviço). Com isso, seria possível realizar consultas sobre quais foram os Referenciados que mais relevantes em um determinado mês, quais são os Serviços de Redes Sociais *Online* que foram locais

preferidos pelos Referenciados para produzir conteúdo sobre o assunto, quais datas foram mais importantes, entre outras informações que o coletor achar necessário.

Figura 36 – Exemplo simplificado das dimensões de um *Data Mart*



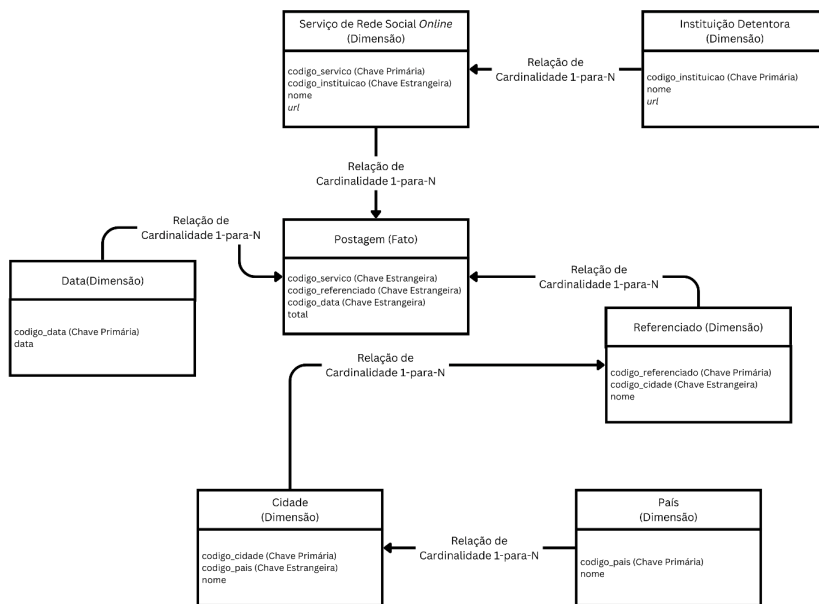
Fonte: Elaborado pelo Autor.

Em analogia, o exemplo descrito geraria um *Data Mart* na forma de cubo tridimensional, conforme a Figura 36. Cada face representa as dimensões propostas. Por exemplo, o cubo colorido representa o total de postagens para a Data x (cor lilás), do Referenciado y (cor ciano) no Serviço de Rede Social *Online* z (cor azul). Cada um dos demais cubos representará a quantidade de postagens para diferentes Datas, Referenciados ou Serviços de Redes Sociais *Online*. No caso de *Data Marts* com mais de

três dimensões, em esquema estrela, não é possível a utilização do cubo como representação gráfica.

Também é comum na construção de *Data Marts* o processo de normalização das dimensões disponíveis no esquema estrela, denominado como *snowflaking*, termo derivado da palavra inglesa para floco de neve (*snowflake*) – formando, desta forma, o esquema floco de neve (no idioma inglês, *snowflake schema*). O processo de normalização das dimensões é resultado de um desconforto ao coletor que poderá ocorrer no uso do esquema estrela, pois a relação direta de todas as dimensões a uma tabela fato poderá produzir o efeito de criar repetições de valores das dimensões desnecessárias para as análises (Kimball; Ross, 2011).

Figura 37 – Exemplo de um modelo dimensional em esquema floco de neve



Fonte: Elaborado pelo Autor.

O exemplo da Figura 35 poderia apresentar este tipo de repetição ao adicionar as novas dimensões Instituição Detentora dos Serviços de Redes Sociais *Online* disponíveis, País e Cidade que o Referenciado reside. Caso

estas dimensões fossem relacionadas diretamente na tabela fato, poderiam produzir valores desnecessários, como, por exemplo, dados na tabela fato com combinações de Referenciados que não residem em certas Cidades. Desta forma, ao adicionar estas dimensões no exemplo da Figura 35, o *Data Mart* para o fato Postagem pode ser normalizado em um esquema floco de neve, conforme a Figura 37.

É importante salientar que a aplicação e o uso do esquema estrela não é desestimulado aos coletores. O que deve ser levado em conta são os objetivos para a construção da Modelagem de Segunda Ordem, ou seja, o que se pretende investigar.

Outra pergunta recorrente é: Se estes dados já não estão disponíveis em um banco de dados transacional, não seria possível executar as mesmas consultas (ou similares) em um banco de dados transacional? A resposta é sim. Todavia, o *Data Mart* é mais eficiente na recuperação dos dados por parte do coletor, já que a quantidade de registros armazenados na estrutura de bancos de dados orientados à operacionalização de coleta de postagens, e a própria disposição da modelagem para estas operações, podem dificultar o acesso das respostas a estas perguntas (Inmon, 2005; Kimball; Ross, 2011; Maribel; Ramos, 2009).

O mesmo caso ocorre na Modelagem Direta. Quão maior for o número de APIs e Versões de APIs coletados, maior será a dificuldade em responder com eficiência os questionamentos de uma pesquisa do coletor. Portanto, propõe-se uma apropriação deste tipo de modelagem, em concomitância ao Modelo Entidade-Relacionamento, com o uso dos artefatos Diagrama Entidade-Relacionamento e Dicionário de Dados, associados aos conceitos de *Business Intelligence*, *Data Warehouse* e *Data Mart* para a estruturação da Modelagem de Segunda Ordem. A escolha do nome (Modelagem Segunda Ordem) foi devido a esta nova organização estruturante depender da primeira organização de dados sobre as características do que está disponível nas APIs – a Modelagem Direta.

3.2 EXEMPLOS DE APLICAÇÃO DO MODELO ENTIDADE-

RELACIONAMENTO NA MODELAGEM DE SEGUNDA ORDEM

Voltando as implicações e as proposições descritas no início deste capítulo, propõe-se nesta seção detalhar quatro exemplos de construção de *Data Marts* para compor a Modelagem de Segunda Ordem. Estes exemplos têm o objetivo principal de auxiliar a compreensão de aspectos ligados a recuperação de dados de Referenciados por Agentes Externos aos Serviços de Redes Sociais *Online*, por meio das APIs disponíveis.

Os exemplos aqui descritos são possibilidades de análises para definição de quais são as Visões e os Atributos que estão acessíveis para Agentes Externos nas APIs destes serviços, suas características e meios de acesso, que permitirão estabelecer, a critério do coletor, elementos que podem ou não ser prejudiciais à privacidade dos Referenciados. Não cabe aqui uma análise sobre se há prejuízos ou não a privacidade dos Referenciados, mas sim mostrar meios que possam estabelecer um processo comparativo sobre o que está ou não está disponível no momento da coleta de dados externa aos Serviços de Redes Sociais *Online*, sem recorrer a métodos escusos, tais como a invasão de sistemas de informação ou a exploração de brechas de código-fonte².

Define-se, portanto, a primeira atividade a qualquer análise em uma Modelagem de Segunda Ordem: saber o que se pretende analisar, a partir dos dados da Modelagem Direta. Quando se possui uma quantidade significativa de dados já estruturados sobre algum serviço na *web*, é importante saber o que se pretende investigar sobre. Neste caso propomos quatro exemplos para a resolução de quatro perguntas:

- a) Quais são as Visões acessíveis, de forma direta, do Serviço de Rede Social *Online Facebook* por meio do uso de um conjunto formado pela Autorização de Acesso *User Access Token* e pelas Permissões *email, public_profile, user_about_me, user_actions.books*, coletados pela *Facebook Graph API* na versão 2.6?

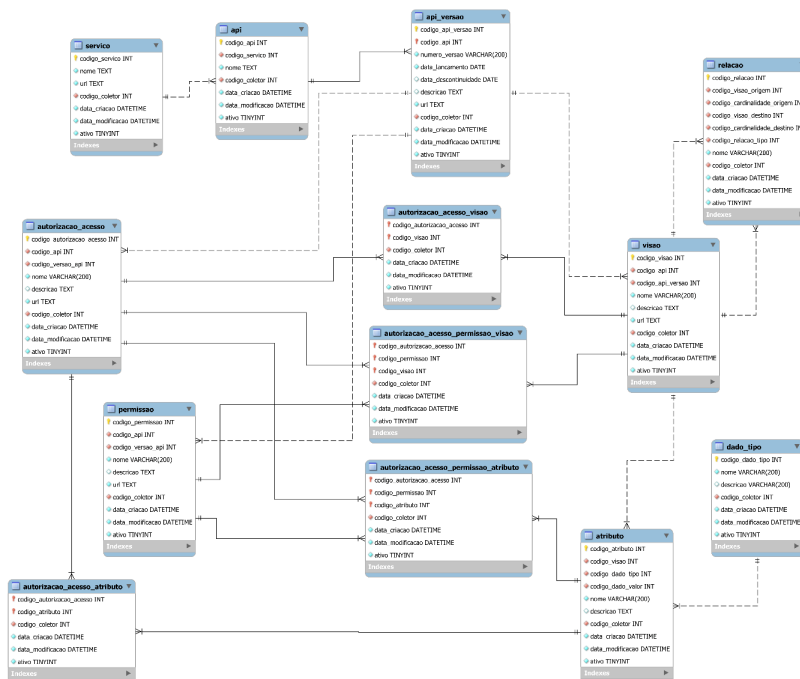
² Ver Rodrigues e Sant'Ana (2016, 2023).

- b) Quais são as Visões acessíveis, de forma direta e indireta (por meio de Relação entre Visão de Origem e Visão de Destino), do Serviço de Rede Social *Online Facebook* por meio do uso de um conjunto formado pela Autorização de Acesso *User Access Token* e pelas Permissões *email, public_profile, user_about_me, user_actions.books*, coletados pela *Facebook Graph* API na versão 2.6?
- c) Quais são os Atributos das Visões acessíveis, de forma direta, do Serviço de Rede Social *Online Facebook* por meio do uso de um conjunto formado pela Autorização de Acesso *User Access Token* e pelas Permissões *email, public_profile, user_about_me, user_actions.books*, coletados pela *Facebook Graph* API na versão 2.6?
- d) Quais são os Atributos das Visões acessíveis, de forma direta e indireta (por meio de Relação entre Visão de Origem e Visão de Destino), do Serviço de Rede Social *Online Facebook* por meio do uso de um conjunto formado pela Autorização de Acesso *User Access Token* e pelas Permissões *email, public_profile, user_about_me, user_actions.books*, coletados pela *Facebook Graph* API na versão 2.6?

Para responder estas quatro perguntas, a Modelagem de Segunda Ordem é formada por um *Data Warehouse* contendo quatro *Data Marts* – sendo que cada um possui dados selecionados especificamente para resolver cada pergunta.

O primeiro passo para a construção do *Data Warehouse* foi selecionar na Modelagem Direta as tabelas que possuem os conjuntos de dados necessários para a construção dos *Data Marts*. Não se trata de um passo obrigatório, mas esta ação auxilia ao coletor a remover de seu foco as tabelas da Modelagem Direta com dados que não terão utilidade nos exemplos propostos para a Modelagem de Segunda Ordem.

Figura 38 – Tabelas da Modelagem Direta selecionadas para a composição da Modelagem de Segunda Ordem



Fonte: Elaborado pelo Autor.

A Figura 38 apresenta um recorte do Diagrama Entidade-Relacionamento da Modelagem Direta, contendo apenas a seleção de tabelas necessárias tanto para a composição das dimensões quanto para as tabelas fato. Para a composição das dimensões, foram selecionadas nove tabelas da Modelagem Direta: *api*, *api-versao*, *atributo*, *autorizacao-acao*, *dado-tipo*, *permissao*, *relacao*, *servico* e *visao*. Para composição das tabelas fato, foram selecionadas quatro tabelas: *autorizacao-acao-atributo*, *autorizacao-acao-permissao-atributo*, *autorizacao-acao-permissao-visao* e *autorizacao-acao-visao*.

Complementarmente, é importante a construção de um roteiro (do inglês, *script*) para a conversão dos conjuntos de dados armazenados nas estruturas da Modelagem Direta, e o envio destes conjuntos de dados ao *Data Warehouse* da Modelagem de Segunda Ordem e, posteriormente, o armazenamento dos *Data Marts* no Sistema de Gerenciamento de Bancos de Dados.

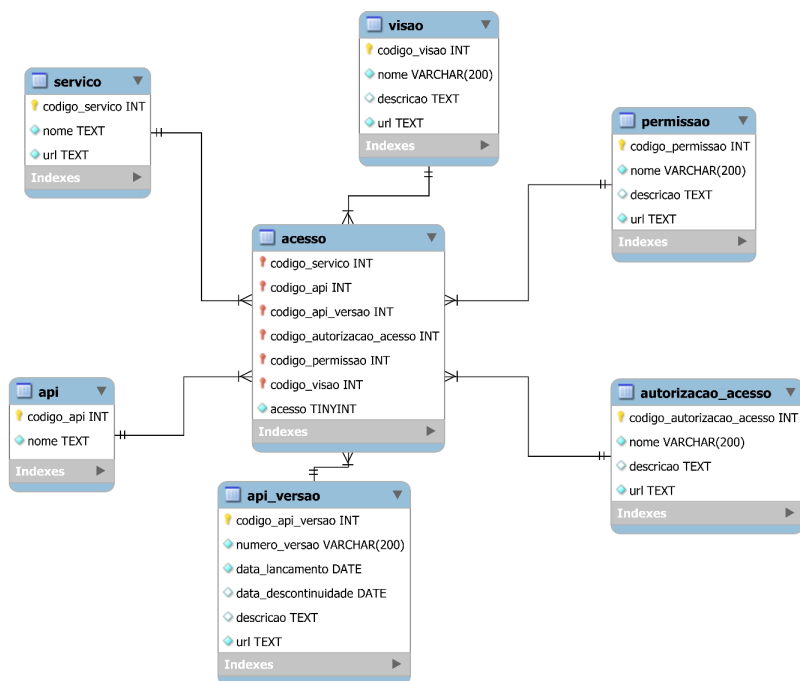
Existem diversas formas de realizar a conversão dos conjuntos de dados entre a Modelagem Direta e de Segunda Ordem, sendo os principais: i) o uso de aplicativos desenvolvidos especificamente para a finalidade de auxiliar a construção deste tipo de análise para *Business Intelligence*, b) a construção de códigos em sintaxe SQL e c) a construção de algoritmos em linguagem de programação e de códigos em sintaxe SQL que convertem os dados. Neste livro, optou-se pelos algoritmos em linguagem de programação e códigos em sintaxe SQL, já que é possível o uso de linguagens de programação de código-aberto³, sem a necessidade de aquisição ou de uso de uma plataforma ou sistema operacional específico.

3.2.1 MODELAGEM DE SEGUNDA ORDEM – *DATA MART* DE VISÕES DIRETAMENTE ACESSÍVEIS

O primeiro *Data Mart* da Modelagem de Segunda Ordem está relacionado a responder à pergunta “Quais são as Visões acessíveis, de forma direta, do Serviço de Rede Social *Online Facebook* por meio do uso de um conjunto formado pela Autorização de Acesso *User Access Token* e pelas Permissões *email*, *public_profile*, *user_about_me*, *user_actions.books*, coletados pela *Facebook Graph API* na versão 2.6?”. Utiliza o esquema estrela e possui seis dimensões e uma tabela fato.

³ Os algoritmos desenvolvidos para os exemplos da Modelagem de Segunda Ordem propostos neste livro podem ser visualizados, utilizados, adaptados e distribuídos pela Licença *Creative Commons 4.0 BY-SA-ND*, disponível no endereço eletrônico <https://github.com/rodriguesprobr/livro-srso>.

Figura 39 – Diagrama Entidade-Relacionamento para a Modelagem de Segunda Ordem – Fato Visões acessíveis em primeiro grau – Esquema Estrela



Fonte: Elaborado pelo Autor.

Esse *Data Mart* é apresentado na forma de Diagrama Entidade-Relacionamento na Figura 39. A tabela fato é denominada *acesso* e possui relação com as seguintes dimensões:

1. Serviço (tabela *servico*), relaciona o fato (*acesso* as Visões) com os Serviços de Redes Sociais *Online*. Por meio desta dimensão é possível filtrar Visões que estão relacionadas a um ou mais Serviços de Redes Sociais *Online*;
2. API (tabela *api*), relaciona o fato (*acesso* as Visões) com as APIs. Por meio desta dimensão é possível filtrar Visões que estão acessíveis por meio de uma ou mais APIs;

3. Versão da API (tabela *api_versao*), relaciona o fato (acesso as Visões) com as Versões das APIs. Por meio desta dimensão é possível filtrar Visões que estão acessíveis por meio de uma ou mais Versões das APIs;
4. Autorização de Acesso (tabela *autorizacao_acesso*), relaciona o fato (acesso as Visões) com as Autorizações de Acesso. Por meio desta dimensão é possível filtrar Visões que estão acessíveis por meio de uma ou mais Autorizações de Acesso;
5. Permissão (tabela *permissao*), relaciona o fato (acesso as Visões) com as Permissões. Por meio desta dimensão é possível filtrar Visões que estão acessíveis por meio de uma ou mais Permissões. No caso dos dados das APIs do *X*, que não utilizam Permissão, foi inserido um registro na dimensão de *codigo_permissao* número 0, com *nome* “Sem o uso de Permissão”;
6. Visão (tabela *visao*): relaciona o fato (acesso as Visões) com as Visões. Por meio desta dimensão é possível filtrar uma ou mais Visões das demais disponíveis.

A tabela fato *acesso* possui uma coluna denominada *acesso*, no qual seu valor só poderá ser 0 (Falso) ou 1 (Verdadeiro), representada pela expressão

$$Acesso_f = Servico_a, API_b, Versao_c, Autorizacao_d, Permissao_e, Visao_f$$

onde valor de *Acesso* da Visão *f* poderá ser Verdadeiro ou Falso dependendo da combinação do Serviço de Rede Social *Online* (*Servico_a*), da API (*A_b*), da Versão de API (*Versao_c*), da Autorização de Acesso (*Autorizacao_d*), da Permissão (*Permissao_e*) e da Visão (*Visao_f*). Os valores disponíveis para *a* a *f* são compostos por todas as possibilidades disponíveis na Modelagem Direta, como, por exemplo, *Servico_a* é forma-

do pelo conjunto de dados disponíveis sobre Serviços de Redes Sociais *Online* da Modelagem Direta (tabela *servico*). Por esta expressão, se o valor do acesso da Visão f é 1 significa que esta combinação dá acesso à Visão. Se o valor é 0, o acesso é negado.

Já as dimensões de análise funcionam como pontos de entrada para realização de consultas a tabela fato *acesso*. Foram escolhidas para permitir a filtragem dos conjuntos de dados. Por exemplo, se for definido um valor para S_a em uma consulta, o *Data Mart* somente recuperará combinações para o Serviço de Rede Social *Online* determinado.

O Quadro 7 apresenta o Dicionário de Dados do *Data Mart*, contendo o detalhamento técnico sobre tabelas, colunas e relacionamentos do Diagrama Entidade-Relacionamento. Apresenta-se um cabeçalho, contendo o nome da tabela, a descrição de seu conteúdo, seguido de uma descrição das colunas. Para cada coluna é apresentado seu nome, o tipo de dado esperado, o tamanho máximo do valor, se a coluna aceita valores nulos, se a coluna é chave primária ou estrangeira, suas configurações padrão e a descrição do tipo de dado que é esperado armazenar.

Quadro 7 – Dicionário de Dados da Modelagem de Segunda Ordem – Fato Visões acessíveis em primeiro grau

Tabela: <i>acesso</i>						
Descrição: Tabela Fato <i>acesso</i> que representará as diversas combinações possíveis de acesso às Visões, combinando dados dos Serviços de Redes Sociais <i>Online</i> , <i>Application Programming Interfaces</i> , Versões das <i>Application Programming Interfaces</i> , Autorizações de Acesso, Permissões e Visões.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
codigo_servico	INT (11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Serviço de Rede Social <i>Online</i> e a tabela fato de acesso as Visões, com origem na tabela <i>servico</i> , coluna <i>codigo_servico</i> .

<i>codigo_api</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão <i>Application Programming Interface</i> e a tabela fato de acesso as Visões, com origem na tabela <i>api</i> , coluna <i>codigo_api</i> .
<i>codigo_api_versao</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Versão da <i>Application Programming Interface</i> e a tabela fato de acesso as Visões, com origem na tabela <i>api_versao</i> , coluna <i>codigo_api_versao</i> .
<i>codigo_autorizacao_acesso</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Autorização de Acesso e a tabela fato de acesso as Visões, com origem na tabela <i>autorizacao_acesso</i> , coluna <i>codigo_autorizacao_acesso</i> .
<i>codigo_permissao</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Permissão e a tabela fato de acesso as Visões, com origem na tabela <i>permissao</i> , coluna <i>codigo_permissao</i> .
<i>codigo_visao</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Visão e a tabela fato de acesso as Visões, com origem na tabela <i>visao</i> , coluna <i>codigo_visao</i> .
<i>acesso</i>	TINYINT (4)	Não	Não	Não		O valor do atributo acesso representará se quais são as Visões estão disponíveis para acesso pela combinação entre o conjunto Serviço de Rede Social Online, <i>Application Programming Interface</i> , Versão da <i>Application Programming Interface</i> , Autorização de Acesso e Permissão. Opcionalmente, o coletor poderá adicionar a consulta dados uma determinada Visão, por meio da dimensão Visão.

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

Tabela: <i>api</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as <i>Application Programming Interfaces</i> que estão vinculados às Visões.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
codigo_api	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
nome	TEXT	Não	Não	Não		O nome da <i>Application Programming Interface</i> .

Tabela: <i>api_versao</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as Versões das <i>Application Programming Interfaces</i> que estão vinculados às Visões.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
codigo_api_versao	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
numero_versao	VARCHAR(200)	Não	Não	Não		Número da Versão da <i>Application Programming Interface</i> .
data_lancamento	DATE	Não	Não	Não		Data de início do funcionamento da Versão da <i>Application Programming Interface</i> .
data_descontinuidade	DATE	Sim	Não	Não	NULL	Data do encerramento da disponibilidade da Versão da <i>Application Programming Interface</i> .
descricao	TEXT	Sim	Não	Não	NULL	A descrição original da Versão da <i>Application Programming Interface</i> (quando disponível), conforme documentação de referência.
url	TEXT	Não	Não	Não		O principal endereço de acesso ao documento da Versão da <i>Application Programming Interface</i> , no formato <i>Uniform Resource Locator</i> .

Tabela: <i>autorizacao_acesso</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as Autorizações de Acesso que dão acesso às Visões.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
codigo_ autorizacao_ acesso	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
nome	VARCHAR (200)	Não	Não	Não		O nome da Autorização de Acesso.
descricao	TEXT	Sim	Não	Não	NULL	A descrição original da Autorização de Acesso (quando disponível), conforme documentação de referência.
url	TEXT	Não	Não	Não		O endereço principal de acesso ao documento da Autorização de Acesso, no formato <i>Uniform Resource Locator</i> .

Tabela: <i>permissao</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as Permissões que dão acesso às Visões.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
codigo_ permissao	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
nome	VARCHAR (200)	Não	Não	Não		O nome da Permissão.
descricao	TEXT	Sim	Não	Não	NULL	A descrição original da Permissão (quando disponível), conforme documentação de referência.
url	TEXT	Não	Não	Não		O endereço de acesso da documentação contendo a referência da Permissão, no formato <i>Uniform Resource Locator</i> .

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Tabela: <i>servico</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre os Serviços de Redes Sociais <i>Online</i> que estão vinculados às <i>Visões</i> .						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_servico</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	TEXT	Não	Não	Não		O nome do Serviço de Rede Social <i>Online</i> .
<i>url</i>	TEXT	Não	Não	Não		O principal endereço de acesso ao Serviço de Rede Social <i>Online</i> , no formato <i>Uniform Resource Locator</i> .

Tabela: <i>visao</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as <i>Visões</i> acessíveis pela Modelagem Direta.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_visao</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR(200)	Não	Não	Não		O nome original da <i>Visão</i> , conforme descrito na documentação de referência.
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original da <i>Visão</i> (quando disponível), conforme documentação de referência.
<i>url</i>	TEXT	Não	Não	Não		O principal endereço de acesso ao documento da <i>Visão</i> , no formato <i>Uniform Resource Locator</i> .

Fonte: Elaborado pelo Autor.

A partir desta estrutura e do carregamento dos dados da amostra dos Serviços de Redes Sociais *Online*, disponíveis na Modelagem Direta, é possível realizar consultas para verificação do acesso as *Visões*, semelhante ao exemplo apresentado na Figura 33 (localizada no início do terceiro capítulo), formando uma consulta para representar um recorte temá-

tico da *Facebook Graph* API, apresentando o caminho do acesso direto de um Agente Externo as Visões *User*, *UserBooks* e *UserFriends*, por meio da Autorização de Acesso *User Access Token* e as permissões *user_friends*, *user_actions.books*, *public_profile*, *email* e *user_about_me*.

Exemplo 25 – Modelagem de Segunda Ordem: Consulta o acesso as Visões disponíveis no Serviço de Rede Social *Online Facebook*, *Facebook Graph* API, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends*, em sintaxe *Structured Query Language*

```

SELECT
    servico.nome AS 'Serviço de Rede Social Online',
    api.nome AS 'API',
    api_versao.numero_versao AS 'Número da Versão',
    autorizacao_acesso.nome AS 'Autorização de Acesso',
    permissao.nome AS 'Permissão',
    visao.nome AS 'Visão'
FROM acesso
INNER JOIN servico ON servico.codigo_servico = acesso.codigo_servico
INNER JOIN api ON api.codigo_api = acesso.codigo_api
INNER JOIN api_versao ON api_versao.codigo_api_versao = acesso.codigo_api_versao
INNER JOIN autorizacao_acesso ON autorizacao_acesso.codigo_autorizacao_acesso = acesso.codigo_autorizacao_acesso
INNER JOIN permissao ON permissao.codigo_permissao = acesso.codigo_permissao
INNER JOIN visao ON visao.codigo_visao = acesso.codigo_visao
WHERE acesso.codigo_servico = 1 /* Identificador do Serviço de Rede Social Online Facebook
*/
AND acesso.codigo_api = 1 /* Identificador da API Facebook Graph API */
AND acesso.codigo_api_versao = 1 /* Identificador da Versão da API 2.6 */
AND acesso.codigo_autorizacao_acesso = 1 /* Identificador da Autorização de Acesso User
Access Token */
AND acesso.codigo_permissao IN (
    1, /* Identificador da Permissão email */
    8, /* Identificador da Permissão public_profile */
    16, /* Identificador da Permissão user_about_me */
    18, /* Identificador da Permissão user_actions.books */
    26 /* Identificador da Permissão user_friends */
)

```

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

```
AND acesso.acesso = 1 /* Filtro para retornar apenas as Visões que estão acessíveis pelos critérios estabelecidos */
ORDER BY
    servico.nome ASC,
    api.nome ASC,
    api_versao.numero_versao ASC,
    autorizacao_acesso.nome ASC,
    permissao.nome ASC,
    visao.nome ASC;
```

Fonte: Elaborado pelo Autor.

O Exemplo 25 codifica em sintaxe SQL uma consulta ao *Data Mart* da Modelagem de Segunda Ordem para o recorte proposto. As colunas *servico.nome* (com origem na dimensão Serviço), *api.nome* (com origem na dimensão API), *api_versao.numero_versao* (com origem na dimensão Versão da API), *autorizacao_acesso.nome* (com origem na dimensão Autorização de Acesso), *permissao.nome* (com origem na dimensão Permissão) e *visao.nome* (com origem na dimensão Visão) foram alteradas para que os resultados apresentem nomes mais humanizados, respectivamente Serviço de Rede Social *Online*, API, Número da Versão, Autorização de Acesso, Permissão e Visão (Silberschatz; Korth; Sudarshan, 2020).

Foi necessário realizar seis operações de junção em álgebra relacional do tipo junção interna (*INNER JOIN*) entre a tabela fato *acesso* e as tabelas das dimensões *servico*, *api*, *api_nome*, *autorizacao_acesso*, *permissao* e *visao*, por meio das chaves estrangeiras da tabela fato *acesso* e das chaves primárias das dimensões. Como a tabela fato *acesso* só armazena as chaves estrangeiras destas tabelas, este tipo de relacionamento é necessário para exibir informações descritivas de cada dimensão (e.g. coluna *nome* da Visão ao invés do valor numérico da coluna *codigo_visao*) (Silberschatz; Korth; Sudarshan, 2020).

Como filtros para exemplificar o uso das dimensões, foram recuperados somente dados sobre Serviço de Rede Social *Online Facebook* (por meio do identificador 1 para a coluna *codigo_servico*), da *Facebook Graph*

API (por meio do identificador 1 para a coluna *codigo_api*), da Versão da API 2.6 (por meio do identificador 1 para a coluna *codigo_api-versao*), da Autorização de Acesso *User Access Token* (por meio do identificador 1 para a coluna *codigo_autorizacao_acesso*), com o uso das Permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends* (respectivamente, por meio dos identificadores 1, 8, 16, 18 e 26 para a coluna *codigo_permissao*).

Além disso, foi inserido um filtro adicional para que o *Data Mart* retorne apenas as Visões acessíveis para os critérios estabelecidos, por meio do valor 1 para a coluna *acesso*.

Os resultados da consulta foram classificados pelo Nome do Serviço de Rede Social *Online*, API, Número de Versão da API, Nome da Autorização de Acesso, Permissão e Visão, em ordem crescente (ordenados alfabeticamente), concatenados.

Tabela 4 – Modelagem de Segunda Ordem: Visões disponíveis no Serviço de Rede Social *Online Facebook, Facebook Graph API*, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email, public_profile, user_about_me, user_actions.books* e *user_friends*

Serviço de Rede Social <i>Online</i>	API	Número da Versão	Autorização de Acesso	Permissão	Visão
<i>Facebook</i>	<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>email</i>	<i>User</i>
<i>Facebook</i>	<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>
<i>Facebook</i>	<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>user_about_me</i>	<i>User</i>
<i>Facebook</i>	<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>user_actions.books</i>	<i>User</i>
<i>Facebook</i>	<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>user_actions.books</i>	<i>UserBooks</i>
<i>Facebook</i>	<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>user_friends</i>	<i>User</i>
<i>Facebook</i>	<i>Facebook Graph API</i>	2.6	<i>User Access Token</i>	<i>user_friends</i>	<i>UserFriends</i>

Fonte: Elaborado pelo Autor.

A Tabela 4 exibe os resultados da consulta, a partir dos dados populados no *Data Mart*. Neste caso, é possível verificar informações da Figura 33 (localizada no início do terceiro capítulo), porém no formato de tabela contendo dados analíticos que representam condições semelhantes da figura, ou seja, que as Visões *User*, *UserBooks* e *UserFriends* estão disponíveis para acesso por meio da Autorização de Acesso *User Access Token* – no qual as permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends* autorizam o acesso à Visão *User*, a permissão *user_actions.books* autoriza o acesso à Visão *UserBooks* e a permissão *user_friends* autoriza o acesso à Visão *UserFriends*.

Todavia, o coletor poderá personalizar esta consulta para recuperar dados de outros Serviços de Redes Sociais *Online*, APIs, Versões de APIs, Autorizações de Acesso, Permissões ou Visões. Para realizar esta personalização, basta o *Data Mart* possuir dados sobre estas entidades armazenadas e, ao coletor, personalizar as consultas em sintaxe SQL com os filtros desejados.

Se compararmos os dados da Tabela 4 com as informações visuais apresentadas pela Figura 33 (localizada no início do terceiro capítulo), pode-se perceber que há uma discrepância com relação às Visões. Nessa figura são apresentadas não só as Visões acessadas diretamente com o uso do conjunto Autorização de Acesso e Permissões. Também apresenta Visões disponíveis indiretamente (Visões de destino), por meio de acesso por arestas ou Atributos da Visão de origem. Um exemplo é a Visão *Page*, acessível por meio de uma Relação com a Visão *UserBooks*.

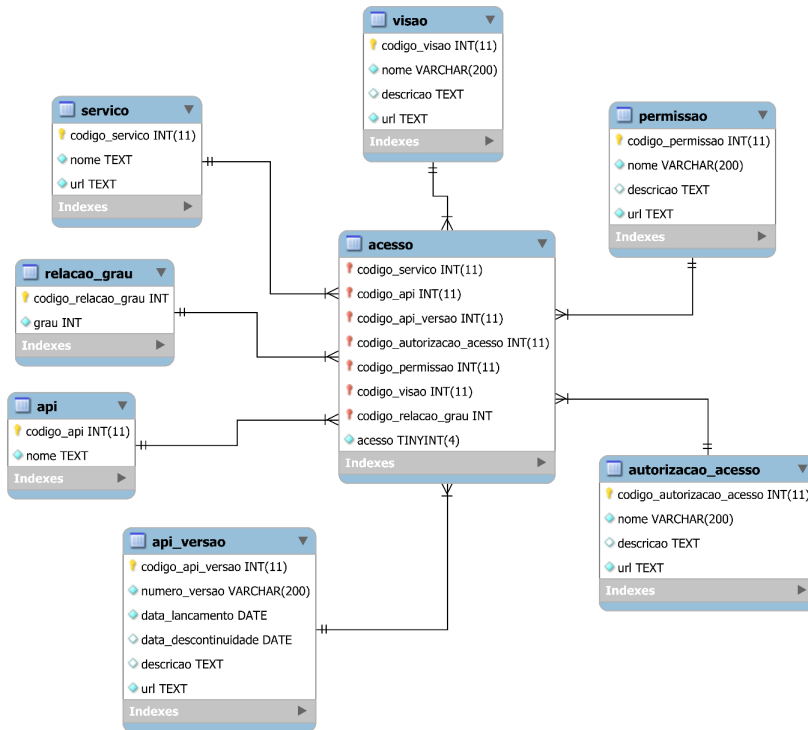
3.2.2 MODELAGEM DE SEGUNDA ORDEM – *DATA MART* DE VISÕES DIRETAMENTE OU INDIRETAMENTE ACESSÍVEIS

Para resolver a análise de Visões diretamente ou indiretamente acessíveis por um conjunto de Autorização de Acesso e Permissão é necessário modificar a estrutura proposta no primeiro exemplo. Portanto, o segundo

Data Mart da Modelagem de Segunda Ordem está relacionado a responder à pergunta “Quais são as Visões acessíveis, de forma direta e indireta (por meio de Relação entre Visão de Origem e Visão de Destino), do Serviço de Rede Social *Online Facebook* por meio do uso de um conjunto formado pela Autorização de Acesso *User Access Token* e pelas Permissões *email*, *public_profile*, *user_about_me*, *user_actions.books*, coletados pela *Facebook Graph API* na versão 2.6?”. Utiliza o esquema estrela e possui sete dimensões, uma dimensão adicional ao primeiro exemplo, e uma tabela fato.

Figura 40 – Diagrama Entidade-Relacionamento para a Modelagem de Segunda Ordem – Fato Visões acessíveis em diferentes graus – Esquema

Estrela



Fonte: Elaborado pelo Autor.

O *Data Mart* para a segunda pergunta é apresentado na Figura 40 na forma de Diagrama Entidade-Relacionamento. A tabela fato é denominada *acesso* e possui relação com as seis dimensões já propostas no primeiro exemplo (Serviço – tabela *servico*, API – tabela *api*, Versão da API – tabela *api_versao*, Autorização de Acesso – tabela *autorizacao_acesso*, Permissão – tabela *permissao* e Visão – tabela *visao*).

Uma sétima dimensão foi elaborada, denominada Grau da Relação (tabela *relacao_grau*), no qual seu propósito é identificar em cada registro da tabela *acesso* se determinada Visão é acessível diretamente ou por meio de Relacionamento com outras Visões. No momento da conversão dos conjuntos de dados entre a Modelagem Direta e de Segunda Ordem é necessário estabelecer um critério para identificar cada registro do *Data Mart*, da seguinte forma:

1. Se a Visão é acessível diretamente pelo uso de uma Autorização de Acesso e por uma Permissão ou somente por uma Autorização de Acesso, seu grau é definido como 1;
2. Se a Visão é acessível por uma Relação com uma Visão de Origem que é acessível diretamente pelo uso de uma Autorização de Acesso e por uma Permissão ou somente por uma Autorização de Acesso, seu grau é definido como 2;
3. Se a Visão é acessível por uma Relação com uma Visão de Origem que é acessível indiretamente, seu grau é definido como a soma do grau da Visão de Origem acrescido de 1 (Grau da Visão de Origem + 1).

A tabela fato *acesso* possui uma coluna denominada *acesso*, no qual seu valor só poderá ser 0 (Falso) ou 1 (Verdadeiro), representada pela expressão

$$Acesso_g = Serviço_a, API_b, Versão_c, Autorização_d, Permissão_e, Grau_f, Visão_g$$

onde valor de *Acesso* da Visão *g* poderá ser Verdadeiro ou Falso dependendo da combinação do Serviço de Rede Social *Online* (*Serviço_a*), da API (*A_b*), da Versão de API (*Versão*), da Autorização de Acesso (*Autorização_d*), da Permissão (*Permissão*), do Grau da Relação (*Grau_p*) e da Visão (*Visão_g*). Os valores disponíveis para *a* e *g* são compostos por todas as possibilidades disponíveis na Modelagem Direta.

Por exemplo, *Serviço_a* é formado pelo conjunto de dados disponíveis sobre Serviços de Redes Sociais *Online* da Modelagem Direta (tabela *servico*). Já outro exemplo é o *Grau_p*, que pode ser filtrado pelo coletor, para recuperar somente Visões diretamente acessíveis (grau 1) ou indiretamente (grau acima de 1).

Por esta expressão se o valor de acesso da Visão *g* é 1 significa que esta combinação dá acesso à Visão. Se o valor é 0, o acesso é negado.

Novamente as dimensões de análise funcionam como pontos de entrada para realização de consultas a tabela fato *acesso*. Foram escolhidas para permitir a filtragem dos conjuntos de dados. Por exemplo, se for definido um valor para *Serviço_a* em uma consulta, o *Data Mart* somente recuperará combinações para o Serviço de Rede Social *Online* determinado.

O Quadro 8 apresenta um recorte do Dicionário de Dados do *Data Mart*, contendo o detalhamento técnico da dimensão Grau da Relação – tabela *relacao_grau* e da tabela fato *acesso*. As outras dimensões são idênticas ao primeiro exemplo proposto (ver Quadro 7, na seção anterior). Apresenta-se um cabeçalho, contendo o nome da tabela, a descrição de seu conteúdo, seguido de uma descrição das colunas. Para cada coluna é apresentado seu nome, o tipo de dado esperado, o tamanho máximo do valor, se a coluna aceita valores nulos, se a coluna é chave primária ou estrangeira, suas configurações padrão e a descrição do tipo de dado que é esperado armazenar.

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

Quadro 8 – Recorte do Dicionário de Dados da Modelagem de Segunda Ordem – Fato Visões acessíveis em diferentes graus: tabelas *acesso* e *relacao_grau*

Tabela: <i>acesso</i>						
Descrição: Tabela Fato intitulada <i>acesso</i> que representará as diversas combinações possíveis de acesso às Visões, combinando dados dos Serviços de Redes Sociais <i>Online</i> , <i>Application Programming Interfaces</i> , Versões das <i>Application Programming Interfaces</i> , Autorizações de Acesso, Permissões, Visões e Grau da Relação.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_servico</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Serviço de Rede Social <i>Online</i> e a tabela fato de acesso as Visões, com origem na tabela <i>servico</i> , coluna <i>codigo_servico</i> .
<i>codigo_api</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão <i>Application Programming Interface</i> e a tabela fato de acesso as Visões, com origem na tabela <i>api</i> , coluna <i>codigo_api</i> .
<i>codigo_api_versao</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Versão da <i>Application Programming Interface</i> e a tabela fato de acesso as Visões, com origem na tabela <i>api_versao</i> , coluna <i>codigo_api_versao</i> .
<i>codigo_autorizacao_acesso</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Autorização de Acesso e a tabela fato de acesso as Visões, com origem na tabela <i>autorizacao_acesso</i> , coluna <i>codigo_autorizacao_acesso</i> .
<i>codigo_permissao</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Permissão e a tabela fato de acesso as Visões, com origem na tabela <i>permissao</i> , coluna <i>codigo_permissao</i> .
<i>codigo_visao</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Visão e a tabela fato de acesso as Visões, com origem na tabela <i>visao</i> , coluna <i>codigo_visao</i> .

<i>codigo_relacao_grau</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Grau da Relação e a tabela fato de <i>acesso</i> as Visões, com origem na tabela <i>relacao_grau</i> , coluna <i>codigo_relacao_grau</i> .
<i>acesso</i>	TINYINT(4)	Não	Não	Não		O valor do atributo <i>acesso</i> representará se quais são as Visões estão disponíveis para acesso pela combinação entre o conjunto Serviço de Rede Social <i>Online, Application Programming Interface</i> , Versão da <i>Application Programming Interface</i> , Autorização de Acesso, Permissão e Grau da Relação. Opcionalmente, o coletor poderá adicionar a consulta dados uma determinada Visão, por meio da dimensão Visão.

Tabela: *relacao_grau*

Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre os possíveis Graus de Relação entre Visões. O Valor 1 representa que a Visão é acessível diretamente por uma Autorização de Acesso ou um conjunto entre Autorização de Acesso e Permissão. Valores acima de 1 representam que a Visão é acessível por meio de relação com outra Visão, na modalidade Visão de Origem - Visão de Destino. Quanto maior o número, mais Relações são necessárias para acessar a Visão.

Colunas

Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_relacao_grau</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>grau</i>	INT(11)	Não	Não	Não		Grau da relação. O Valor 1 representa que a Visão é acessível diretamente por uma Autorização de Acesso ou um conjunto entre Autorização de Acesso e Permissão. Valores acima de 1 representam que a Visão é acessível por meio de relação com outra Visão, na modalidade Visão de Origem - Visão de Destino. Quanto maior o número, mais Relações são necessárias para acessar a Visão.

Fonte: Elaborado pelo Autor.

A partir da adaptação da estrutura do primeiro *Data Mart*, do estabelecimento das regras para os valores da dimensão Grau da Relação e do carregamento dos dados da amostra dos Serviços de Redes Sociais *Online*, disponíveis na Modelagem Direta, é possível realizar consultas para verificação do acesso as Visões.

Diferente do primeiro exemplo, as consultas nesse *Data Mart* recuperam os mesmos dados do exemplo apresentado na Figura 33 (localizada no início do terceiro capítulo): uma consulta para representar um recorte temático da *Facebook Graph* API, apresentando o caminho do acesso de um Agente Externo as Visões *User*, *UserBooks*, *UserFriends* e *Page* por meio da Autorização de Acesso *User Access Token* e as permissões *user_friends*, *user_actions.books*, *public_profile*, *email* e *user_about_me*.

Exemplo 26 – Modelagem de Segunda Ordem: Consulta o acesso as Visões disponíveis em diferentes graus no Serviço de Rede Social *Online Facebook*, *Facebook Graph* API, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends*, em sintaxe *Structured Query Language*

```
SELECT
servico.nome AS 'Serviço de Rede Social Online',
api.nome AS 'API',
api_versao.numero_versao AS 'Número da Versão',
autorizacao_acesso.nome AS 'Autorização de Acesso',
permissao.nome AS 'Permissão',
relacao_grau.grau AS 'Grau da Relação',
visao.nome AS 'Visão'
FROM acesso
INNER JOIN servico ON servico.codigo_servico = acesso.codigo_servico
INNER JOIN api ON api.codigo_api = acesso.codigo_api
INNER JOIN api_versao ON api_versao.codigo_api_versao = acesso.codigo_api_versao
INNER JOIN autorizacao_acesso ON autorizacao_acesso.codigo_autorizacao_acesso = acesso.codigo_autorizacao_acesso
INNER JOIN permissao ON permissao.codigo_permissao = acesso.codigo_permissao
INNER JOIN visao ON visao.codigo_visao = acesso.codigo_visao
```

```

INNER JOIN relacao_grau ON relacao_grau.codigo_relacao_grau = acesso.codigo_relacao_
grau
WHERE acesso.codigo_servico = 1 /* Identificador do Serviço de Rede Social Online Facebook
*/
AND acesso.codigo_api = 1 /* Identificador da API Facebook Graph API */
AND acesso.codigo_api-versao = 1 /* Identificador da Versão da API 2.6 */
AND acesso.codigo_autorizacao_acesso = 1 /* Identificador da Autorização de Acesso User
Access Token */
AND acesso.codigo_permissao IN (
    1, /* Identificador da Permissão email */
    8, /* Identificador da Permissão public_profile */
    16, /* Identificador da Permissão user_about_me */
    18, /* Identificador da Permissão user_actions.books */
    26 /* Identificador da Permissão user_friends */
)
AND acesso.codigo_relacao_grau IN (
    1, /* Identificador do Grau da Relação 1 - Acesso a Visão de forma Direta */
    2 /* Identificador do Grau da Relação 2 - Acesso a Visão de forma Indireta, via Visão
de Origem de Grau 1 */
)
AND acesso.acesso = 1 /* Filtro para retornar apenas as Visões que estão acessíveis pelos critérios
estabelecidos */
ORDER BY
    servico.nome ASC,
    api.nome ASC,
    api-versao.numero-versao ASC,
    autorizacao_acesso.nome ASC,
    permissao.nome ASC,
    relacao_grau.grau ASC,
    visao.nome ASC;

```

Fonte: Elaborado pelo Autor.

O Exemplo 26 codifica em sintaxe SQL uma consulta ao *Data Mart* da Modelagem de Segunda Ordem para o recorte proposto. As colunas *servico.nome* (com origem na dimensão Serviço), *api.nome* (com origem na dimensão API), *api-versao.numero-versao* (com origem na dimensão Versão da API), *autorizacao_acesso.nome* (com origem na dimensão Autorização de Acesso), *permissao.nome* (com origem na dimensão Permissão), *relacao_grau.grau* (com origem na dimensão Grau da Relação) e *visao.nome*

(com origem na dimensão Visão) foram alteradas para que os resultados apresentem nomes mais humanizados, respectivamente Serviço de Rede Social *Online*, API, Número da Versão, Autorização de Acesso, Permissão, Grau da Relação e Visão (Silberschatz; Korth; Sudarshan, 2020).

Foi necessário realizar sete operações de junção em álgebra relacional do tipo junção interna (*INNER JOIN*) entre a tabela fato *acesso* e as tabelas das dimensões *servico*, *api*, *api_nome*, *autorizacao_acesso*, *permissao*, *relacao_grau* e *visao*, por meio das chaves estrangeiras da tabela fato *acesso* e das chaves primárias das dimensões. Como a tabela fato *acesso* só armazena as chaves estrangeiras destas tabelas, este tipo de relacionamento é necessário para exibir informações descritivas de cada dimensão (e.g. coluna *nome* da Visão ao invés do valor numérico da coluna *codigo_visao*) (Silberschatz; Korth; Sudarshan, 2020).

Como filtros para exemplificar o uso das dimensões, foram recuperados somente dados sobre Serviço de Rede Social *Online Facebook* (por meio do identificador 1 para a coluna *codigo_servico*), da *Facebook Graph API* (por meio do identificador 1 para a coluna *codigo_api*), da Versão da API 2.6 (por meio do identificador 1 para a coluna *codigo_api_versao*), da Autorização de Acesso *User Access Token* (por meio do identificador 1 para a coluna *codigo_autorizacao_acesso*), com o uso das Permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends* (respectivamente, por meio dos identificadores 1, 8, 16, 18 e 26 para a coluna *codigo_permissao*).

Também foi filtrado a recuperação de dados sobre Visões acessíveis apenas em primeiro e segundo grau, respectivamente, Visões acessíveis diretamente pelo conjunto de Autorização de Acesso e Permissões selecionadas (grau de número 1), e Visões acessíveis indiretamente pelas Visões de acesso direto (grau de número 2).

Além disso, foi inserido um filtro adicional para que o *Data Mart* retorne apenas as Visões acessíveis para os critérios estabelecidos, por meio do valor 1 para a coluna *acesso*.

Os resultados da consulta foram classificados pelo Nome do Serviço de Rede Social *Online*, API, Número de Versão, Nome da Autorização de Acesso, Permissão, Grau da Relação e Visão, em ordem crescente (ordenados alfabeticamente ou, no caso do Grau da Relação, de forma numérica crescente), concatenados.

Tabela 5 – Modelagem de Segunda Ordem: Visões disponíveis em diferentes graus no Serviço de Rede Social *Online Facebook, Facebook Graph API*, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email, public_profile, user_about_me, user_actions.books* e *user_friends*

Serviço de Rede Social Online	API	Número da Versão	Autorização de Acesso	Permissão	Grau da Relação	Visão
Facebook	Facebook Graph API	2.6	User Access Token	email	1	User
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	1	User
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	2	UserFriends
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	2	UserBooks
Facebook	Facebook Graph API	2.6	User Access Token	user_about_me	1	User
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	1	User
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	1	UserBooks
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	2	Page
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	1	User
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	1	UserFriends
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	2	User

Fonte: Elaborado pelo Autor.

A Tabela 5 exhibe os resultados da consulta, a partir dos dados populados no *Data Mart*. Neste caso, é possível verificar as informações da Figura 33 (localizada no início do terceiro capítulo), no formato de tabela com dados analíticos que representam as mesmas condições exibidas na figura, ou seja, que as Visões *User*, *UserBooks* e *UserFriends* estão disponíveis para acesso por meio da Autorização de Acesso *User Access Token* – no qual as permissões *email, public_profile, user_about_me, user_actions.books* e *user_friends* autorizam o acesso a Visão *User*, a permissão *user_actions.books* autoriza o acesso a Visão *UserBooks* e a permissão *user_friends* autoriza

o acesso a Visão *UserFriends* – todas acessíveis em primeiro grau (coluna Grau da Relação, valor 1).

Diferentemente do primeiro exemplo de *Data Mart*, a Tabela 5 também exibe as Visões acessíveis em segundo grau, de forma indireta (coluna Grau da Relação, valor 2): Visões *UserFriends* e *UserBooks* acessíveis pela Relação com a Visão *User* para a permissão *public_profile*, Visão *Page* acessível pela Relação com a Visão *UserBooks* para a permissão *user_actions.books* e Visão *User* acessível pela Relação com a Visão *UserFriends* para a permissão *user_friends* – todas estas utilizando a Autorização de Acesso *User Access Token*.

Todavia, o coletor poderá personalizar esta consulta para recuperar dados de outros Serviços de Redes Sociais *Online*, APIs, Versões de APIs, Autorizações de Acesso, Permissões, Graus de Relação ou Visões. Para realizar esta personalização, basta o *Data Mart* possuir dados sobre estas entidades armazenadas e, ao coletor, personalizar as consultas em sintaxe SQL com os filtros desejados.

Ao compararmos os dados da Tabela 5 com as informações visuais apresentadas pela Figura 33 (localizada no início do terceiro capítulo), pode-se perceber que uma consulta sistematizada ao *Data Mart* da Modelagem de Segunda Ordem pode tanto ser subsídio para a elaboração da própria figura, como traz a possibilidade de recuperar dados analíticos, como os apresentados na tabela – deixando a critério do coletor escolher a melhor forma de analisar e interpretar os dados originários da Modelagem Direta, ou seja, mais plasticidade para atender suas demandas informacionais.

Ainda no caso da Figura 33, os vetores de retorno de dados para cada acesso do Agente Externo foram mesclados para facilitar a visualização. Todavia, em uma análise mais aprofundada sobre os dados que estarão disponíveis a estes entes, cada retorno de dados deveria, em tese, ser individualizado, pois permissões distintas apresentam acessos a diferentes conjuntos de Atributos de uma Visão. Por exemplo, a permissão *email* e

a permissão *public_profile* dão acesso direto a Visão *User*, mas retornam atributos diferentes, como será visto nos próximos exemplos.

Não obstante, o coletor poderá gerar uma imagem que possua vetores individuais para representar os conjuntos de atributos recuperados de cada Visão. Todavia, se a análise do coletor for derivada de uma quantidade significativa de Visões, acessíveis em diferentes graus, poderá ser dificultoso a visualização por meio de imagens, assim, sendo importante a possibilidade de acesso aos dados sistematizados, como os apresentados na Tabela 5. O importante aqui é destacar que os *Data Marts* propostos abrem possibilidades ao coletor de estruturar a recuperação dos dados a sua conveniência.

3.2.3 MODELAGEM DE SEGUNDA ORDEM – *DATA MART* DE ATRIBUTOS DE VISÕES DIRETAMENTE ACESSÍVEIS

Nos dois primeiros exemplos para a Modelagem de Segunda Ordem foram desenvolvidos *Data Marts* desenhados a partir do esquema estrela. O esquema estrela permite ao coletor uma maior velocidade na finalização da construção do *Data Mart*. Contudo, esquemas mais complexos podem não serem eficientes com o uso do esquema estrela, especialmente quando a tabela fato possuir relação com diversas dimensões para análise.

Ao adotar o esquema estrela, outro fator a ser levado em consideração é que quão maior o número de dimensões, maior será a quantidade de linhas na tabela fato – no qual não necessariamente todas as suas linhas serão úteis nas análises.

Um exemplo desse fator são os dois primeiros exemplos. De acordo com a proposta dos esquemas estrelas desenvolvidos, um número considerável de linhas poderia ser evitado, como, por exemplo, registros da tabela fato *acesso* que constatam que não há acesso em Visões de uma API do *Facebook* com o uso de Autorização de Acesso de uma API do *X*. Esta

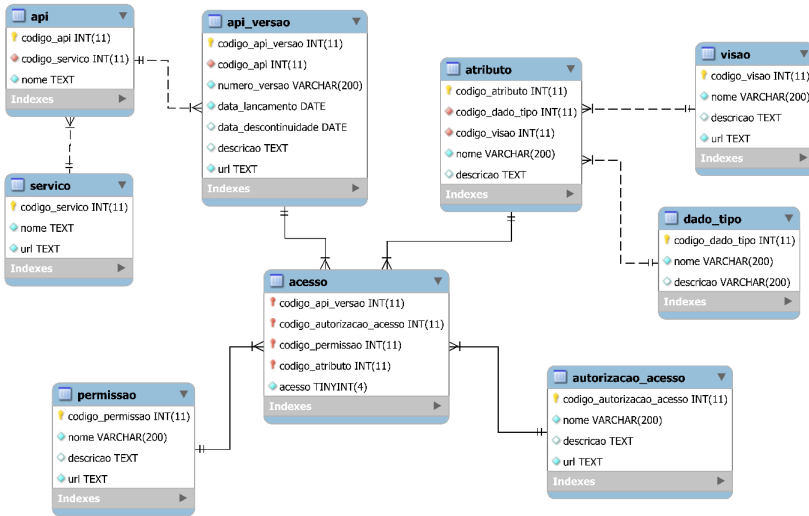
combinação provavelmente não existirá, pois são API relacionadas a instituições detentoras distintas.

Neste sentido, o terceiro e o quarto exemplo utilizam o esquema floco de neve, especialmente por serem mais complexos que os dois primeiros – ao adicionar um nível mais detalhado de análise, colocando em perspectiva central o acesso aos Atributos, de forma individualizada. Como uma Visão possui um ou mais Atributos, caso fosse utilizado o esquema estrela, a quantidade de dados sem utilidade seria mais significativa.

Portanto, o terceiro *Data Mart* da Modelagem de Segunda Ordem está relacionado a responder à pergunta “Quais são os Atributos das Visões acessíveis, de forma direta, do Serviço de Rede Social *Online Facebook* por meio do uso de um conjunto formado pela Autorização de Acesso *User Access Token* e pelas Permissões *email, public_profile, user_about_me, user_actions.books*, coletados pela *Facebook Graph* API na versão 2.6?”.

O *Data Mart* possui quatro dimensões principais, sendo que duas dimensões possuem relacionamentos com outras quatro dimensões, totalizando oito dimensões, e uma tabela fato. Este processo de dimensões dependentes de outras dimensões é parte integrante da normalização das dimensões, proposto por Silberschatz, Korth e Sudarshan (2020, p. 55–58), no qual os autores atentam a desencorajar o uso de muitas dimensões dependentes umas das outras, correndo o risco de tornar-se um processo que tem o objetivo de facilitar as análises em um processo complexo e moroso.

Figura 41 – Diagrama Entidade-Relacionamento para a Modelagem de Segunda Ordem – Fato Atributos acessíveis em primeiro grau – Esquema Floco de Neve



Fonte: Elaborado pelo Autor.

O *Data Mart* do terceiro exemplo é apresentado na forma de Diagrama Entidade-Relacionamento na Figura 41. A tabela fato também é denominada *acesso* e possui relação direta com as seguintes dimensões:

1. Versão da API (tabela *api-versao*), relaciona o fato (*acesso* aos Atributos) com as Versões das APIs. Por meio desta dimensão é possível filtrar Atributos que estão acessíveis por meio de uma ou mais Versões das APIs;
2. Autorização de Acesso (tabela *autorizacao_acesso*), relaciona o fato (*acesso* aos Atributos) com as Autorizações de Acesso. Por meio desta dimensão é possível filtrar Atributos que estão acessíveis por meio de uma ou mais Autorizações de Acesso;
3. Permissão (tabela *permissao*), relaciona o fato (*acesso* aos Atributos) com as Permissões. Por meio desta dimensão é pos-

sível filtrar Atributos que estão acessíveis por meio de uma ou mais Permissões. No caso dos dados das APIs do X , que não utilizam Permissão, foi inserido um registro na dimensão de *codigo_permissao* número 0, com *nome* “Sem o uso de Permissão”;

4. Atributos (tabela *atributo*): relaciona o fato (acesso aos Atributos) com os Atributos. Por meio desta dimensão é possível filtrar um ou mais Atributos dos demais disponíveis.

Além disso, as dimensões Versão da API e Atributo possuem relacionamentos com outras quatro dimensões, com as seguintes características:

1. API (tabela *api*), relaciona a dimensão Versão da API (tabela *api_versao*) com as APIs nas quais estão relacionadas. Por meio desta dimensão é possível filtrar Atributos que estão acessíveis por meio de uma ou mais APIs, independente de qual sejam as suas versões;
2. Serviço (tabela *servico*), relaciona a dimensão API (tabela *api*) com os Serviços de Redes Sociais *Online* nas quais as APIs são pertencentes. Por meio desta dimensão é possível filtrar Atributos que estão relacionados a um ou mais Serviços de Redes Sociais *Online*, independente em qual API ou Versão de API que o Atributo está vinculado;
3. Visão (tabela *visao*), relaciona a dimensão Atributo (tabela *atributo*) com as Visões nas quais são pertencentes. Por meio desta dimensão é possível filtrar Atributos que estão acessíveis por meio de uma ou mais Visões;
4. Tipo de Dado (tabela *dado_tipo*), relaciona a dimensão Atributo (tabela *atributo*) com os Tipos de Dados dos Atributos. Por meio desta dimensão é possível filtrar os Tipos de Dados dos Atributos.

A tabela fato *acesso* possui uma coluna denominada *acesso*, no qual seu valor só poderá ser 0 (Falso) ou 1 (Verdadeiro), representada pela expressão

$$Acesso_a = API_a, Autorização_b, Permissão_c, Atributo_d$$

onde valor de *Acesso* do Atributo *d* poderá ser Verdadeiro ou Falso dependendo da combinação da Versão de API (*Versão_a*), da Autorização de Acesso (*Autorização_b*), da Permissão (*Permissão_c*) e do Atributo (*Atributo_d*). Os valores disponíveis para *a* e *d* são compostos por todas as possibilidades disponíveis na Modelagem Direta, como, por exemplo, *Versão_a* é formado pelo conjunto de dados disponíveis sobre Versões de API da Modelagem Direta (tabela *api_versao*). Por esta expressão, se o valor do acesso do Atributo *d* é 1 significa que esta combinação dá acesso ao Atributo. Se o valor é 0, o acesso é negado.

As demais dimensões – Serviço de Rede Social *Online*, API, Visão e Tipo de Dado – servem como subsídio para filtragem de outras dimensões. Por exemplo, o coletor poderá utilizar a dimensão Visão para filtrar somente os Atributos de uma determinada Visão, o que afetará os resultados a serem exibidos da tabela fato *acesso*. Neste caso, a consulta ao *Data Mart* exibirá somente dados referentes as condições de acessos dos Atributos da Visão escolhida.

Portanto, todas as dimensões de análise funcionam como pontos de entrada para realização de consultas, independente se estão diretamente ou indiretamente ligadas a tabela fato *acesso*.

O Quadro 9 apresenta o Dicionário de Dados do *Data Mart*, contendo o detalhamento técnico destas tabelas, contendo informações sobre tabelas, colunas e relacionamentos do Diagrama Entidade-Relacionamento. Apresenta-se um cabeçalho, contendo o nome da tabela, a descrição de seu conteúdo, seguido de uma descrição das colunas. Para cada coluna

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

é apresentado seu nome, o tipo de dado esperado, o tamanho máximo do valor, se a coluna aceita valores nulos, se a coluna é chave primária ou estrangeira, suas configurações padrão e a descrição do tipo de dado que é esperado armazenar.

Quadro 9 – Dicionário de Dados da Modelagem de Segunda Ordem –
Fato Atributos acessíveis em primeiro grau

Tabela: <i>acesso</i>						
Descrição: Tabela Fato intitulada <i>acesso</i> que representará as diversas combinações possíveis de acesso aos Atributos, combinando dados dos Serviços de Redes Sociais <i>Online</i> , <i>Application Programming Interfaces</i> , Versões das <i>Application Programming Interfaces</i> , Autorizações de Acesso, Permissões e Visões.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
codigo_api_versao	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Versão da <i>Application Programming Interface</i> e a tabela fato de acesso aos Atributos, com origem na tabela <i>api_versao</i> , coluna <i>codigo_api_versao</i> .
codigo_autorizacao_acesso	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Autorização de Acesso e a tabela fato de acesso aos Atributos, com origem na tabela <i>autorizacao_acesso</i> , coluna <i>codigo_autorizacao_acesso</i> .
codigo_permissao	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Permissão e a tabela fato de acesso aos Atributos, com origem na tabela <i>permissao</i> , coluna <i>codigo_permissao</i> .

<i>codigo_atributo</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Atributo e a tabela fato de acesso aos Atributos, com origem na tabela <i>atributo</i> , coluna <i>codigo_atributo</i> .
<i>acesso</i>	TINYINT (4)	Não	Não	Não		O valor do atributo acesso representará se quais são os Atributos estão disponíveis para acesso pela combinação entre o conjunto Versão da <i>Application Programming Interface</i> , Autorização de Acesso e Permissão. Opcionalmente, o coletor poderá adicionar a consulta dados de a) um determinado Atributo, por meio da dimensão Atributo, b) um determinado Tipo de Dado para o Atributo, por meio da dimensão Tipo de Dado, c) uma determinada Visão para o Atributo, por meio da dimensão Visão, d) uma determinada <i>Application Programming Interface</i> , por meio da dimensão <i>Application Programming Interface</i> , e e) um determinado Serviço de Rede Social <i>Online</i> , por meio da dimensão Serviços de Redes Sociais <i>Online</i> .

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Tabela: <i>api</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as <i>Application Programming Interfaces</i> que estão vinculados às Visões.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_api</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_servico</i>	INT(11)	Não	Não	Sim		Chave estrangeira do relacionamento entre a dimensão <i>Application Programming Interface</i> e a dimensão Serviço de Rede Social <i>Online</i> , com origem na tabela <i>servico</i> , coluna <i>codigo_servico</i> .
<i>nome</i>	TEXT	Não	Não	Não		O nome da <i>Application Programming Interface</i> .

Tabela: <i>api_versao</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as Versões das <i>Application Programming Interfaces</i> que estão vinculados aos Atributos.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_api_versao</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_api</i>	INT(11)	Não	Não	Sim		
<i>numero_versao</i>	VARCHAR(200)	Não	Não	Não		Número da Versão da <i>Application Programming Interface</i> .
<i>data_lancamento</i>	DATE	Não	Não	Não		Data de início do funcionamento da Versão da <i>Application Programming Interface</i> .

<i>data_descontinuidade</i>	DATE	Sim	Não	Não	NULL	Data do encerramento da disponibilidade da Versão da <i>Application Programming Interface</i> .
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original da Versão da <i>Application Programming Interface</i> (quando disponível), conforme documentação de referência.
<i>url</i>	TEXT	Não	Não	Não		O principal endereço de acesso ao documento da Versão da <i>Application Programming Interface</i> , no formato <i>Uniform Resource Locator</i> .

Tabela: atributo

Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre os Atributos acessíveis pela Modelagem Direta.

Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_atributo</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>codigo_dado_tipo</i>	INT(11)	Não	Não	Sim		Chave estrangeira do relacionamento entre a dimensão Tipo de Dado e a dimensão Atributo, com origem na tabela <i>dado_tipo</i> , coluna <i>codigo_dado_tipo</i> .
<i>codigo_visao</i>	INT(11)	Não	Não	Sim		Chave estrangeira do relacionamento entre a dimensão Visão e a dimensão Atributo, com origem na tabela <i>visao</i> , coluna <i>codigo_visao</i> .

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

<i>nome</i>	VARCHAR (200)	Não	Não	Não		Nome original do Atributo, conforme descrito na documentação de referência.
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original do Atributo (quando disponível), conforme documentação de referência.

Tabela: *autorizacao_acesso*

Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as Autorizações de Acesso que dão acesso aos Atributos.

Colunas

Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_ autorizacao_ acesso</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR (200)	Não	Não	Não		O nome da Autorização de Acesso.
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original da Autorização de Acesso (quando disponível), conforme documentação de referência.
<i>url</i>	TEXT	Não	Não	Não		O endereço principal de acesso ao documento da Autorização de Acesso, no formato <i>Uniform Resource Locator</i> .

Tabela: <i>dado_tipo</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre os Tipos de Dados dos Atributos acessíveis pela Modelagem Direta.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_dado_tipo</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR(200)	Não	Não	Não		O nome do Tipo de Dado.
<i>descricao</i>	VARCHAR(200)	Sim	Não	Não	NULL	A descrição das características do Tipo de Dado.

Tabela: <i>permissao</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as Permissões que dão acesso aos Atributos.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_permissao</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR(200)	Não	Não	Não		O nome da Permissão.
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original da Permissão (quando disponível), conforme documentação de referência.
<i>url</i>	TEXT	Não	Não	Não		O endereço de acesso da documentação contendo a referência da Permissão, no formato <i>Uniform Resource Locator</i> .

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

Tabela: <i>servico</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre os Serviços de Redes Sociais <i>Online</i> que estão vinculados aos Atributos.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_servico</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	TEXT	Não	Não	Não		O nome do Serviço de Rede Social <i>Online</i> .
<i>url</i>	TEXT	Não	Não	Não		O principal endereço de acesso ao Serviço de Rede Social <i>Online</i> , no formato <i>Uniform Resource Locator</i> .

Tabela: <i>visao</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre as Visões acessíveis pela Modelagem Direta.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_visao</i>	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>nome</i>	VARCHAR(200)	Não	Não	Não		O nome original da Visão, conforme descrito na documentação de referência.
<i>descricao</i>	TEXT	Sim	Não	Não	NULL	A descrição original da Visão (quando disponível), conforme documentação de referência.
<i>url</i>	TEXT	Não	Não	Não		O principal endereço de acesso ao documento da Visão, no formato <i>Uniform Resource Locator</i> .

Fonte: Elaborado pelo Autor.

A partir desta estrutura e do carregamento dos dados da amostra dos Serviços de Redes Sociais *Online*, disponíveis na Modelagem Direta, é possível realizar consultas para verificação do acesso aos Atributos, semelhante ao exemplo apresentado na Figura 34 (localizada no início do terceiro capítulo), formando uma consulta para representar um recorte temático da *Facebook Graph API*, apresentando o caminho do acesso direto de um Agente Externo aos Atributos das Visões *User*, *UserBooks* e *UserFriends*, por meio da Autorização de Acesso *User Access Token* e as permissões *user_friends*, *user_actions.books*, *public_profile*, *email* e *user_about_me*.

Exemplo 27 – Modelagem de Segunda Ordem: Consulta o acesso aos Atributos das Visões disponíveis no Serviço de Rede Social *Online Facebook*, *Facebook Graph API*, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends*, em sintaxe *Structured Query Language*

```
SELECT
    servico.nome AS 'Serviço de Rede Social Online',
    api.nome AS 'API',
    api_versao.numero_versao AS 'Número da Versão',
    autorizacao_acesso.nome AS 'Autorização de Acesso',
    permissao.nome AS 'Permissão',
    visao.nome AS 'Visão',
    atributo.nome AS 'Atributo',
    dado_tipo.nome AS 'Tipo de Dado'
FROM acesso
INNER JOIN api_versao ON api_versao.codigo_api_versao = acesso.codigo_api_versao
INNER JOIN api ON api.codigo_api = api_versao.codigo_api
INNER JOIN servico ON servico.codigo_servico = api.codigo_servico
INNER JOIN autorizacao_acesso ON autorizacao_acesso.codigo_autorizacao_acesso = acesso.codigo_autorizacao_acesso
INNER JOIN permissao ON permissao.codigo_permissao = acesso.codigo_permissao
INNER JOIN atributo ON atributo.codigo_atributo = acesso.codigo_atributo
INNER JOIN visao ON visao.codigo_visao = atributo.codigo_visao
INNER JOIN dado_tipo ON dado_tipo.codigo_dado_tipo = atributo.codigo_dado_tipo
WHERE servico.codigo_servico = 1 /* Identificador do Serviço de Rede Social Online Facebook */
```

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

```
AND api.codigo_api = 1 /* Identificador da API Facebook Graph API */
AND acesso.codigo_api-versao = 1 /* Identificador da Versão da API 2.6 */
AND acesso.codigo_autorizacao_acesso = 1 /* Identificador da Autorização de Acesso User Access
Token */
AND acesso.codigo_permissao IN (
    1, /* Identificador da Permissão email */
    8, /* Identificador da Permissão public_profile */
    16, /* Identificador da Permissão user_about_me */
    18, /* Identificador da Permissão user_actions.books */
    26 /* Identificador da Permissão user_friends */
)
AND acesso.acesso = 1 /* Filtro para retornar apenas as Visões que estão acessíveis pelos critérios
estabelecidos */
ORDER BY
    servico.nome ASC,
    api.nome ASC,
    api-versao.numero-versao ASC,
    autorizacao-acesso.nome ASC,
    permissao.nome ASC,
    visao.nome ASC,
    atributo.nome ASC;
```

Fonte: Elaborado pelo Autor.

O Exemplo 27 codifica em sintaxe SQL uma consulta ao *Data Mart* da Modelagem de Segunda Ordem para o recorte proposto. As colunas *servico.nome* (com origem na dimensão Serviço), *api.nome* (com origem na dimensão API), *api-versao.numero-versao* (com origem na dimensão Versão da API), *autorizacao-acesso.nome* (com origem na dimensão Autorização de Acesso), *permissao.nome* (com origem na dimensão Permissão), *visao.nome* (com origem na dimensão Visão), *atributo.nome* (com origem na dimensão Atributo) e *dado-tipo.nome* (com origem na dimensão Tipo de Dado) foram alteradas para que os resultados apresentem nomes mais humanizados, respectivamente Serviço de Rede Social *Online*, API, Número da Versão, Autorização de Acesso, Permissão, Visão, Atributo e Tipo de Dado (Silberschatz; Korth; Sudarshan, 2020).

Foi necessário realizar quatro operações de junção em álgebra relacional do tipo junção interna (*INNER JOIN*) entre a tabela fato *acesso* e as tabelas das dimensões *api_versao*, *atributo*, *autorizacao_acesso*, *permissao* e *visao*, por meio das chaves estrangeiras da tabela fato *acesso* e das chaves primárias das dimensões. Como a tabela fato *acesso* só armazena as chaves estrangeiras destas tabelas, este tipo de relacionamento é necessário para exibir informações descritivas de cada dimensão (e.g. coluna *nome* do Atributo ao invés do valor numérico da coluna *codigo_atributo*) (Silberschatz; Korth; Sudarshan, 2020).

Adicionalmente foram realizadas quatro operações adicionais de junção em álgebra relacional do tipo junção (*INNER JOIN*) entre as tabelas das dimensões: i) *api_versao* e *api*; ii) *api* e *servico*; iii) *atributo* e *visao*, e; iv) *atributo* e *dado_tipo*. Todos estes vínculos são necessários para que o coletor possa utilizá-las como pontos de acesso para filtragem de outras dimensões, além de permitir a exibição dos rótulos de colunas *servico.nome* (com origem na dimensão Serviço), *api.nome* (com origem na dimensão API), *visao.nome* (com origem na dimensão Visão) e *dado_tipo.nome* (com origem da dimensão Tipo de Dado).

Como filtros para exemplificar o uso das dimensões, foram recuperados somente dados sobre Serviço de Rede Social *Online Facebook* (por meio do identificador 1 para a coluna *codigo_servico*), da *Facebook Graph API* (por meio do identificador 1 para a coluna *codigo_api*), da Versão da API 2.6 (por meio do identificador 1 para a coluna *codigo_api_versao*), da Autorização de Acesso *User Access Token* (por meio do identificador 1 para a coluna *codigo_autorizacao_acesso*), com o uso das Permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends* (respectivamente, por meio dos identificadores 1, 8, 16, 18 e 26 para a coluna *codigo_permissao*).

Além disso, foi inserido um filtro adicional para que o *Data Mart* retorne apenas os Atributos acessíveis para os critérios estabelecidos, por meio do valor 1 para a coluna *acesso*.

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Os resultados da consulta foram classificados pelo Nome do Serviço de Rede Social *Online*, API, Número de Versão da API, Nome da Autorização de Acesso, Permissão, Visão e Atributo, em ordem crescente (ordenados alfabeticamente), concatenados.

Tabela 6 – Modelagem de Segunda Ordem: Atributos de Visões disponíveis no Serviço de Rede Social *Online Facebook, Facebook Graph* API, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email, public_profile, user_about_me, user_actions.books* e *user_friends*

Serviço de Rede Social <i>Online</i>	API	Número da Versão	Autorização de Acesso	Permissão	Visão	Atributo	Tipo de Dado
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>email</i>	<i>User</i>	<i>e-mail</i>	<i>string</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>	<i>age_range</i>	<i>AgeRange</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>	<i>first_name</i>	<i>string</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>	<i>gender</i>	<i>string</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>	<i>id</i>	<i>numeric_string</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>	<i>last_name</i>	<i>string</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>	<i>link</i>	<i>string</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>	<i>locale</i>	<i>string</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>	<i>name</i>	<i>string</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>public_profile</i>	<i>User</i>	<i>timezone</i>	<i>float</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>user_about_me</i>	<i>User</i>	<i>bio</i>	<i>string</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>user_actions.books</i>	<i>UserBooks</i>	<i>created_time</i>	<i>datetime</i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>user_actions.books</i>	<i>UserBooks</i>	<i>data</i>	<i>list<Page></i>
<i>Facebook</i>	<i>Facebook Graph</i> API	2.6	<i>User Access Token</i>	<i>user_friends</i>	<i>UserFriends</i>	<i>data</i>	<i>list<User></i>

Fonte: Elaborado pelo Autor.

A Tabela 6 exibe os resultados da consulta, a partir dos dados populados no *Data Mart*. Neste caso, é possível verificar informações da Figura 34 (localizada no início do terceiro capítulo), porém no formato de tabela contendo dados analíticos que representam condições semelhantes da figura, ou seja, que os seguintes Atributos estão acessíveis:

- a) *e-mail*, da Visão *User*, por meio da Autorização de Acesso *User Access Token* e da permissão *email*;
- b) *age_range*, *first_name*, *gender*, *id*, *last_name*, *link*, *locale*, *name* e *timezone*, da Visão *User*, por meio da Autorização de Acesso *User Access Token* e da permissão *public_profile*;
- c) *bio*, da Visão *User*, por meio da Autorização de Acesso *User Access Token* e da permissão *user_about_me*;
- d) *created_time* e *data*, da Visão *UserBooks*, por meio da Autorização de Acesso *User Access Token* e da permissão *user_actions.books*;
- e) *data*, da Visão *UserFriends*, por meio da Autorização de Acesso *User Access Token* e da permissão *user_friends*.

Todavia, o coletor poderá personalizar esta consulta para recuperar dados de outros Serviços de Redes Sociais *Online*, APIs, Versões de APIs, Autorizações de Acesso, Permissões, Visões, Atributos e Tipos de Dados. Para realizar esta personalização, basta o *Data Mart* possuir dados sobre estas entidades armazenadas e, ao coletor, personalizar as consultas em sintaxe SQL com os filtros desejados.

Se compararmos os dados da Tabela 6 com os dois primeiros *Data Marts* elaborados para a Modelagem de Segunda Ordem, percebe-se que colocar o Atributo como elemento central da tabela fato *acesso* apresentou menor granularidade dos dados e, conseqüentemente, um enriquecimento na recuperação dos dados, especialmente detalhando explicitamente quais são os Atributos das Visões disponíveis em cada cenário de acesso.

Por exemplo, os resultados da Tabela 6 exibem que o acesso aos Atributos da Visão *User*, por meio da Autorização de Acesso *User Access Token* e da permissão *user_about_me*. Nos *Data Marts* anteriores, aonde a granularidade é maior, apresenta que este conjunto de Autorização de Acesso e Permissão dá acesso à Visão *User* – podendo dar uma falsa sensação ao coletor que todos os Atributos desta Visão estarão disponíveis para coleta de dados em uma requisição na API.

Em contrapartida ao enriquecimento, ao compararmos dados da Tabela 6 com as informações visuais apresentadas pela Figura 34 (localizada no início do terceiro capítulo), pode-se perceber que ainda há uma discrepância com relação aos Atributos das Visões. Nesta figura são apresentadas não só os Atributos das Visões acessadas diretamente com o uso do conjunto Autorização de Acesso e Permissões. Também apresenta Atributos de Visões disponíveis indiretamente (Visões de destino), por meio de acesso por arestas ou Atributos da Visão de origem. Um exemplo são os Atributos da Visão *Page*, acessíveis por meio de uma Relação com a Visão *UserBooks*.

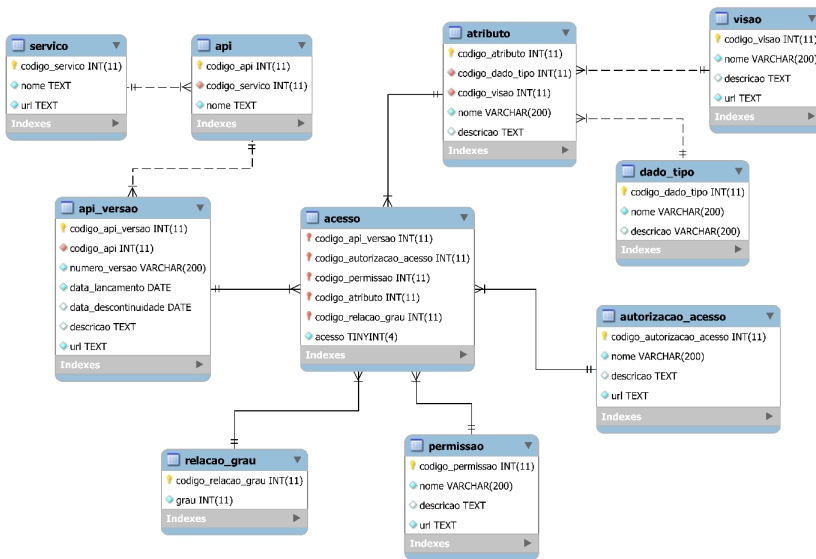
3.2.4 MODELAGEM DE SEGUNDA ORDEM – *DATA MART* DE ATRIBUTOS DE VISÕES DIRETAMENTE OU INDIRETAMENTE ACESSÍVEIS

Para resolver a análise de Atributos de Visões acessíveis diretamente ou indiretamente por um conjunto de Autorização de Acesso e Permissão é necessário modificar a estrutura proposta do terceiro exemplo, a mesma solução entre os dois primeiros exemplos de *Data Marts*. Neste sentido, o quarto *Data Mart* da Modelagem de Segunda Ordem está relacionado a responder à pergunta “Quais são os Atributos das Visões acessíveis, de forma direta e indireta (por meio de Relação entre Visão de Origem e Visão de Destino), do Serviço de Rede Social *Online Facebook* por meio do uso de um conjunto formado pela Autorização de Acesso *User Access Token* e pelas Permissões *email*, *public_profile*, *user_about_me*, *user_actions.books*, coletados pela *Facebook Graph API* na versão 2.6?”.

Utiliza o esquema floco de neve e possui cinco dimensões principais, uma a mais do que no terceiro exemplo, sendo que duas dimensões possuem relacionamentos com outras quatro dimensões, totalizando nove dimensões, e uma tabela fato.

O *Data Mart* elaborado para a quarta pergunta é apresentado na Figura 42 na forma de Diagrama Entidade-Relacionamento. A tabela fato é denominada *acesso* e possui relação com as quatro dimensões já propostas no primeiro exemplo (Versão da API – tabela *api-versao*, Atributo – tabela *atributo*, Autorização de Acesso – tabela *autorizacao-acesso* e Permissão – tabela *permissao*).

Figura 42 – Diagrama Entidade-Relacionamento para a Modelagem de Segunda Ordem – Fato Atributos acessíveis em diferentes graus – Esquema Floco de Neve



Fonte: Elaborado pelo Autor.

Uma quinta dimensão foi elaborada, denominada Grau da Relação (tabela *relacao-grau*), no qual seu propósito é identificar em cada registro

da tabela *acesso* se determinado Atributo é acessível diretamente ou por meio de Relacionamento com outras Visões. No momento da conversão dos conjuntos de dados entre a Modelagem Direta e de Segunda Ordem é necessário estabelecer um critério para identificar cada registro do *Data Mart*, da seguinte forma:

- a) Se o Atributo é acessível diretamente pelo uso de uma Autorização de Acesso e por uma Permissão ou somente por uma Autorização de Acesso, seu grau é definido como 1;
- b) Se o Atributo está relacionado a uma Visão que somente é acessível por uma Relação com uma Visão de Origem que é acessível diretamente pelo uso de uma Autorização de Acesso e por uma Permissão ou somente por uma Autorização de Acesso, seu grau é definido como 2;
- c) Se o Atributo é acessível por uma Relação com uma Visão de Origem que é acessível indiretamente, seu grau é definido como a soma do grau da Visão de Origem acrescido de 1 (Grau da Visão de Origem + 1).

Além disso, as dimensões Versão da API e Atributo possuem relacionamentos com outras quatro dimensões, igual ao terceiro exemplo: i) API (tabela *api*), relaciona a dimensão Versão da API (tabela *api_versao*); ii) Serviço (tabela *servico*), relaciona a dimensão API (tabela *api*); iii) Visão (tabela *visao*), relaciona a dimensão Atributo (tabela *atributo*), e; iv) Tipo de Dado (tabela *dado_tipo*), relaciona a dimensão Atributo (tabela *atributo*).

A tabela fato *acesso* possui uma coluna denominada *acesso*, no qual seu valor só poderá ser 0 (Falso) ou 1 (Verdadeiro), representada pela expressão

$$Acesso_e = API_a, Autorização_b, Permissão_c, Grau_d, Atributo_e$$

onde valor de *Acesso* do Atributo *e* poderá ser Verdadeiro ou Falso dependendo da combinação da Versão de API (*Versão_a*), da Autorização de Acesso (*Autorização_v*), da Permissão (*Permissão_z*), do Grau da Relação (*Grau_d*) e do Atributo (*Atributo_o*). Os valores disponíveis para *a* e *e* são compostos por todas as possibilidades disponíveis na Modelagem Direta.

Por exemplo, *Versão_a* é formado pelo conjunto de dados disponíveis sobre Versões de API da Modelagem Direta (tabela *api_versao*). Já outro exemplo é o *Grau_d*, que pode ser filtrado pelo coletor, para recuperar somente Atributos diretamente acessíveis (grau 1) ou indiretamente (grau acima de 1).

Por esta expressão se o valor de acesso do Atributo *e* é 1 significa que esta combinação dá acesso ao Atributo. Se o valor é 0, o acesso é negado.

De forma igualitária ao terceiro exemplo, as demais dimensões – Serviço de Rede Social *Online*, API, Visão e Tipo de Dado – servem como subsídio para filtragem de outras dimensões. Por exemplo, o coletor poderá utilizar a dimensão Visão para filtrar somente os Atributos de uma determinada Visão, o que afetará os resultados a serem exibidos da tabela fato *acesso*. Neste caso, a consulta ao *Data Mart* exibirá somente dados referentes os acessos dos Atributos da Visão escolhida.

Também para este exemplo, todas as dimensões de análise, independente se estão diretamente ou indiretamente ligadas a tabela fato *acesso*, funcionam como pontos de entrada para realização de consultas.

O Quadro 10 apresenta um recorte do Dicionário de Dados do *Data Mart*, contendo o detalhamento técnico da dimensão Grau da Relação – tabela *relacao_grau* e da tabela fato *acesso*. As outras dimensões são idênticas ao primeiro exemplo proposto (ver Quadro 9, na seção anterior). Apresenta-se um cabeçalho, contendo o nome da tabela, a descrição de seu conteúdo, seguido de uma descrição das colunas. Para cada coluna é apresentado seu nome, o tipo de dado esperado, o tamanho máximo do valor, se a coluna aceita valores nulos, se a coluna é chave primária ou estrangeira, suas configurações padrão e a descrição do tipo de dado que é esperado armazenar.

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Quadro 10 – Recorte do Dicionário de Dados da Modelagem de Segunda Ordem – Fato Atributos acessíveis em diferentes graus: tabelas *acesso* e *relacao_grau*

Tabela: <i>acesso</i>						
Descrição: Tabela Fato intitulada <i>acesso</i> que representará as diversas combinações possíveis de acesso aos Atributos, combinando dados dos Serviços de Redes Sociais <i>Online</i> , <i>Application Programming Interfaces</i> , Versões das <i>Application Programming Interfaces</i> , Autorizações de Acesso, Permissões, Visões e Grau da Relação.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
<i>codigo_api_versao</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Versão da <i>Application Programming Interface</i> e a tabela fato de acesso aos Atributos, com origem na tabela <i>api_versao</i> , coluna <i>codigo_api_versao</i> .
<i>codigo_autorizacao_acesso</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Autorização de Acesso e a tabela fato de acesso aos Atributos, com origem na tabela <i>autorizacao_acesso</i> , coluna <i>codigo_autorizacao_acesso</i> .
<i>codigo_permissao</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Permissão e a tabela fato de acesso aos Atributos, com origem na tabela <i>permissao</i> , coluna <i>codigo_permissao</i> .
<i>codigo_api_versao</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Versão da <i>Application Programming Interface</i> e a tabela fato de acesso aos Atributos, com origem na tabela <i>api_versao</i> , coluna <i>codigo_api_versao</i> .

<i>codigo_relacao_grau</i>	INT(11)	Não	Sim	Sim		Chave estrangeira do relacionamento entre a dimensão Grau da Relação e a tabela fato de <i>acesso</i> aos Atributos, com origem na tabela <i>relacao_grau</i> , coluna <i>codigo_relacao_grau</i> .
<i>acesso</i>	TINYINT (4)	Não	Não	Não		O valor do atributo <i>acesso</i> representará se quais são os Atributos estão disponíveis para acesso pela combinação entre o conjunto Versão da <i>Application Programming Interface</i> , Autorização de Acesso, Permissão e Grau da Relação. Opcionalmente, o coletor poderá adicionar a consulta dados de a) um determinado Atributo, por meio da dimensão Atributo, b) um determinado Tipo de Dado para o Atributo, por meio da dimensão Tipo de Dado, c) uma determinada Visão para o Atributo, por meio da dimensão Visão, d) uma determinada <i>Application Programming Interface</i> , por meio da dimensão <i>Application Programming Interface</i> , e) um determinado Serviço de Rede Social <i>Online</i> , por meio da dimensão Serviços de Redes Sociais <i>Online</i> , e f) um determinado Grau da Relação, por meio da dimensão Grau da Relação.

Estruturas de dados em serviços de redes sociais online
Uma abordagem metodológica de análise

Tabela: <i>relacao_grau</i>						
Descrição: Entidade que armazena dados sobre a dimensão que representará um ponto de acesso para consultas sobre os possíveis graus de Relação entre Atributos por meio das Visões. O Valor 1 representa que o Atributo é acessível por meio de uma Visão que está diretamente por uma Autorização de Acesso ou um conjunto entre Autorização de Acesso e Permissão. Valores acima de 1 representam que o atributo é acessível por meio da relação de sua Visão com outra Visão, na modalidade Visão de Origem - Visão de Destino. Quão maior o número, mais Relações são necessárias para acessar o Atributo.						
Colunas						
Nome da Coluna	Tipo de Dado e Tamanho Máximo	Aceita valores nulos?	Chave Primária?	Chave Estrangeira?	Configurações Padrão	Descrição
codigo_relacao_grau	INT(11)	Não	Sim	Não		Chave primária e artificial da tabela, gerada automaticamente.
<i>grau</i>	INT(11)	Não	Não	Não		Grau da relação. O Valor 1 representa que o Atributo é acessível por meio de uma Visão que está diretamente por uma Autorização de Acesso ou um conjunto entre Autorização de Acesso e Permissão. Valores acima de 1 representam que o atributo é acessível por meio da relação de sua Visão com outra Visão, na modalidade Visão de Origem - Visão de Destino. Quão maior o número, mais Relações são necessárias para acessar o Atributo.

Fonte: Elaborado pelo Autor.

A partir da adaptação da estrutura do terceiro *Data Mart*, do estabelecimento das regras para os valores da dimensão Grau da Relação e do carregamento dos dados da amostra dos Serviços de Redes Sociais *Online*, disponíveis na Modelagem Direta, é possível realizar consultas para verificação do acesso aos Atributos.

Diferente do terceiro exemplo, a consulta realizada nesse *Data Mart* recupera os mesmos dados do exemplo apresentado na Figura 34 (localizada no início do terceiro capítulo): representar um recorte temático da *Facebook Graph API*, apresentando o caminho do acesso de um Agente Externo aos Atributos das Visões *User*, *UserBooks*, *UserFriends* e *Page* por meio da Autorização de Acesso *User Access Token* e as permissões *user_friends*, *user_actions.books*, *public_profile*, *email* e *user_about_me*.

Exemplo 28 – Modelagem de Segunda Ordem: Consulta o acesso aos Atributos das Visões disponíveis em diferentes graus no Serviço de Rede Social *Online Facebook*, *Facebook Graph API*, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends*, em sintaxe *Structured Query Language*

```

SELECT
servico.nome AS 'Serviço de Rede Social Online',
api.nome AS 'API',
api_versao.numero_versao AS 'Número da Versão',
autorizacao_acesso.nome AS 'Autorização de Acesso',
permissao.nome AS 'Permissão',
visao.nome AS 'Visão',
relacao_grau.grau AS 'Grau da Relação',
atributo.nome AS 'Atributo',
dado_tipo.nome AS 'Tipo de Dado'
FROM acesso
INNER JOIN api_versao ON api_versao.codigo_api_versao = acesso.codigo_api_versao
INNER JOIN api ON api.codigo_api = api_versao.codigo_api
INNER JOIN servico ON servico.codigo_servico = api.codigo_servico
INNER JOIN autorizacao_acesso ON autorizacao_acesso.codigo_autorizacao_acesso = acesso.codigo_autorizacao_acesso
INNER JOIN permissao ON permissao.codigo_permissao = acesso.codigo_permissao
INNER JOIN atributo ON atributo.codigo_atributo = acesso.codigo_atributo
INNER JOIN visao ON visao.codigo_visao = atributo.codigo_visao
INNER JOIN dado_tipo ON dado_tipo.codigo_dado_tipo = atributo.codigo_dado_tipo
INNER JOIN relacao_grau ON relacao_grau.codigo_relacao_grau = acesso.codigo_relacao_grau

```

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

```
WHERE servico.codigo_servico = 1 /* Identificador do Serviço de Rede Social Online Facebook */
AND api.codigo_api = 1 /* Identificador da API Facebook Graph API */
AND acesso.codigo_api_versao = 1 /* Identificador da Versão da API 2.6 */
AND acesso.codigo_autorizacao_acesso = 1 /* Identificador da Autorização de Acesso User Access Token */
AND acesso.codigo_permissao IN (
    1, /* Identificador da Permissão email */
    8, /* Identificador da Permissão public_profile */
    16, /* Identificador da Permissão user_about_me */
    18, /* Identificador da Permissão user_actions.books */
    26 /* Identificador da Permissão user_friends */
)
AND acesso.codigo_relacao_grau IN (
    1, /* Identificador do Grau da Relação 1 - Acesso a Visão de forma Direta */
    2 /* Identificador do Grau da Relação 2 - Acesso a Visão de forma Indireta, via Visão de Origem de Grau 1 */
)

AND acesso.acesso = 1 /* Filtro para retornar apenas as Visões que estão acessíveis pelos critérios estabelecidos */

ORDER BY
    servico.nome ASC,
    api.nome ASC,
    api_versao.numero_versao ASC,
    autorizacao_acesso.nome ASC,
    permissao.nome ASC,
    relacao_grau.grau ASC,
    visao.nome ASC,
    atributo.nome ASC;
```

Fonte: Elaborado pelo Autor.

O Exemplo 28 codifica em sintaxe SQL uma consulta ao *Data Mart* da Modelagem de Segunda Ordem para o recorte proposto. As colunas *servico.nome* (com origem na dimensão Serviço), *api.nome* (com origem na dimensão API), *api_versao.numero_versao* (com origem na dimensão Versão da API), *autorizacao_acesso.nome* (com origem na dimensão Autorização de Acesso), *permissao.nome* (com origem na dimensão Permissão), *visao.nome* (com origem na dimensão Visão), *relacao_grau.grau* (com origem na dimensão Grau da Relação), *atributo.nome* (com origem na dimensão

Atributo) e *dado_tipo.nome* (com origem da dimensão Tipo de Dado) foram alteradas para que os resultados apresentem nomes mais humanizados, respectivamente Serviço de Rede Social *Online*, API, Número da Versão, Autorização de Acesso, Permissão, Visão, Grau da Relação, Atributo e Tipo de Dado (Silberschatz; Korth; Sudarshan, 2020).

Foi necessário realizar cinco operações de junção em álgebra relacional do tipo junção interna (*INNER JOIN*) entre a tabela fato *acesso* e as tabelas das dimensões *api_versao*, *atributo*, *autorizacao_acesso*, *permissao*, *relacao_grau* e *visao*, por meio das chaves estrangeiras da tabela fato *acesso* e das chaves primárias das dimensões. Como a tabela fato *acesso* só armazena as chaves estrangeiras destas tabelas, este tipo de relacionamento é necessário para exibir informações descritivas de cada dimensão (e.g. coluna *nome* do Atributo ao invés do valor numérico da coluna *codigo_atributo*) (Silberschatz; Korth; Sudarshan, 2020).

Adicionalmente foram realizadas quatro operações adicionais de junção em álgebra relacional do tipo junção (*INNER JOIN*) entre as tabelas das dimensões: i) *api_versao* e *api*; ii) *api* e *servico*; iii) *atributo* e *visao*, e; iv) *atributo* e *dado_tipo*. Todos estes vínculos são necessários para que o coletor possa utilizá-las como pontos de acesso para filtragem de outras dimensões, além de permitir a exibição dos rótulos de colunas *servico.nome* (com origem na dimensão Serviço), *api.nome* (com origem na dimensão API), *visao.nome* (com origem na dimensão Visão) e *dado_tipo.nome* (com origem da dimensão Tipo de Dado).

Como filtros para exemplificar o uso das dimensões, foram recuperados somente dados sobre Serviço de Rede Social *Online Facebook* (por meio do identificador 1 para a coluna *codigo_servico*), da *Facebook Graph API* (por meio do identificador 1 para a coluna *codigo_api*), da Versão da API 2.6 (por meio do identificador 1 para a coluna *codigo_api_versao*), da Autorização de Acesso *User Access Token* (por meio do identificador 1 para a coluna *codigo_autorizacao_acesso*), com o uso das Permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends* (respeti-

vamente, por meio dos identificadores 1, 8, 16, 18 e 26 para a coluna *codigo_permissao*).

Também foi filtrado a recuperação de dados sobre Atributos das Visões acessíveis apenas em primeiro e segundo grau, respectivamente, Atributos das Visões acessíveis diretamente pelo conjunto de Autorização de Acesso e Permissões selecionadas (grau de número 1), e Atributos das Visões acessíveis indiretamente pelas Visões de acesso direto (grau de número 2).

Além disso, foi inserido um filtro adicional para que o *Data Mart* retorne apenas os Atributos acessíveis para os critérios estabelecidos, por meio do valor 1 para a coluna *acesso*.

Os resultados da consulta foram classificados pelo Nome do Serviço de Rede Social *Online*, API, Número de Versão da API, Nome da Autorização de Acesso, Permissão, Grau da Relação, Visão e Atributo, em ordem crescente (ordenados alfabeticamente ou, no caso do Grau da Relação, de forma numérica crescente), concatenados.

Tabela 7 – Modelagem de Segunda Ordem: Atributos de Visões disponíveis em diferentes graus no Serviço de Rede Social *Online Facebook*, *Facebook Graph* API, versão 2.6, por meio da Autorização de Acesso *User Access Token* e das permissões *email*, *public_profile*, *user_about_me*, *user_actions.books* e *user_friends*

Serviço de Rede Social Online	API	Número da Versão	Autorização de Acesso	Permissão	Visão	Grau da Relação	Atributo	Tipo de Dado
Facebook	Facebook Graph API	2.6	User Access Token	email	User	1	e-mail	string
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	User	1	age_range	AgeRange
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	User	1	first_name	string
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	User	1	gender	string
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	User	1	id	numeric_string
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	User	1	last_name	string

Serviço de Rede Social Online	API	Número da Versão	Autorização de Acesso	Permissão	Visão	Grau da Relação	Atributo	Tipo de Dado
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	User	1	link	string
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	User	1	locale	string
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	User	1	name	string
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	User	1	timezone	float
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	UserBooks	2	created_time	datetime
Facebook	Facebook Graph API	2.6	User Access Token	public_profile	UserBooks	2	data	list<Page>
Facebook	Facebook Graph API	2.6	User Access Token	user_about_me	User	1	bio	string
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	UserBooks	1	created_time	datetime
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	UserBooks	1	data	list<Page>
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	Page	2	category	string
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	Page	2	checkins	unsigned int32
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	Page	2	description	string
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	Page	2	id	numeric_string
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	Page	2	link	string
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	Page	2	name	string
Facebook	Facebook Graph API	2.6	User Access Token	user_actions.books	Page	2	username	string
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	UserFriends	1	data	list<User>
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	User	2	age_range	AgeRange
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	User	2	first_name	string
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	User	2	gender	string
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	User	2	id	numeric_string
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	User	2	last_name	string
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	User	2	link	string
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	User	2	locale	string
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	User	2	name	string

Estruturas de dados em serviços de redes sociais online

Uma abordagem metodológica de análise

Serviço de Rede Social Online	API	Número da Versão	Autorização de Acesso	Permissão	Visão	Grau da Relação	Atributo	Tipo de Dado
Facebook	Facebook Graph API	2.6	User Access Token	user_friends	User	2	timezone	float

Fonte: Elaborado pelo Autor.

A Tabela 7 exhibe os resultados da consulta, a partir dos dados populados no *Data Mart*. Neste caso, é possível verificar as informações da Figura 34 (localizada no início do terceiro capítulo), no formato de tabela, com dados analíticos que representam as mesmas condições exibidas na figura, ou seja, que os Atributos Visões *User*, *UserBooks* e *UserFriends* estão disponíveis para acesso como demonstrado no terceiro exemplo, porém incrementando o acesso indireto aos Atributos das Visões em segundo grau, de forma indireta (coluna Grau da Relação, valor 2):

- created_time* e *data*, da Visão *UserBooks*, por meio da Autorização de Acesso *User Access Token* e da permissão *public_profile*;
- category*, *checkins*, *description*, *id*, *link*, *name* e *username* da Visão *Page*, por meio da Autorização de Acesso *User Access Token* e da permissão *user_actions.books*;
- age_range*, *first_name*, *gender*, *id*, *last_name*, *link*, *locale*, *name* e *timezone*, da Visão *User*, por meio da Autorização de Acesso *User Access Token* e da permissão *user_friends*.

Todavia, o coletor poderá personalizar esta consulta para recuperar dados de outros Serviços de Redes Sociais *Online*, APIs, Versões de APIs, Autorizações de Acesso, Permissões, Visões, Atributos e Tipos de Dados. Para realizar esta personalização, basta o *Data Mart* possuir dados sobre estas entidades armazenadas e, ao coletor, personalizar as consultas em sintaxe SQL com os filtros desejados.

Os dados exemplificados por meio deste *Data Mart* demonstram a complexidade do ambiente de análise sobre acessos aos Atributos das Visões dos Serviços de Redes Sociais *Online* por meio das APIs – fechando o ciclo proposto em contribuir com uma percepção mais detalhada sobre os possíveis caminhos de acesso com o uso de Autorizações de Acesso e Permissões.

Um exemplo importante a ser delineado sobre essa complexidade do o acesso pode ser verificado a partir dos Atributos disponíveis da Visão *User* por meio da Autorização de Acesso *User Access Token* e da Permissão *user_friends*.

Quando um Referenciado *a* concede acesso a um Agente Externo aos seus dados com estas permissões, um aplicativo externo ao Serviço de Rede Social *Online* pode coletar, via API, o Atributo *data* da Visão *UserFriends*, de forma direta.

O Atributo *data* é do Tipo de Dado *list<User>*, um tipo de dado composto que retorna ao Agente Externo uma lista contendo objetos do tipo Visão *User* que, em outras palavras, apresenta uma lista de Referenciados que possuem relação de amizade com o Referenciado *a*.

Em um segundo momento, o Agente Externo poderá realizar novas requisições pela API, solicitando, individualmente, dados públicos dos Referenciados que possuem relação de amizade com o Referenciado *a*. Se o Agente Externo utilizar a Autorização de Acesso *User Access Token* e da Permissão *public_profile* (uma das permissões mais flexíveis e genéricas a disposição), ele poderá coletar individualmente os Atributos *age_range* (faixa etária), *first_name* (primeiro nome), *gender* (gênero), *id* (código identificador do Referenciado no Serviço de Rede Social *Online*), *last_name* (sobrenome), *link* (URL do perfil do Referenciado no Serviço de Rede Social *Online*), *locale* (idioma escolhido pelo Referenciado para acesso aos serviços), *name* (nome completo) e *timezone* (fuso horário) dos Referenciados.

CONCLUSÃO

CONCLUSÃO

Nos Serviços de Redes Sociais *Online*, o acompanhamento sobre como os conjuntos de dados de Referenciados são disponibilizados pelas APIs para público externo (Agentes Externos) possui alta opacidade. Esta opacidade é reflexo da complexidade do contexto de análise dos serviços, aonde a complexidade é influenciada pelas diversas estruturas de dados necessárias para a própria operacionalização e funcionamento dos sistemas de informação dos Serviços de Redes Sociais *Online*, mas também por um processo de compartilhamento de parte dos dados destas estruturas para acesso externo aos serviços.

Aqui as APIs modulam a complexidade de forma dualista. Por um lado, contribui diretamente para ofuscar o que de fato é ou não é acessível por Agentes Externos – em documentos técnicos com linguagem específica a um público (desenvolvedores de software) – no qual identificar quem são estes agentes acaba por ser algo intrincado, já que dependerá de inúmeras condições de acesso, permissões, autorizações, acordos interinstitucionais, compartilhamento para empresas de publicidade, parceiros, além de transações de dados entre bancos de dados em diferentes países e, portanto, vinculado a diferentes legislações.

Por outro lado, a documentação técnica das APIs são documentos oficiais das próprias instituições, o que é um material bruto interessante para investigar o que é de fato armazenado sobre Referenciados pelos Serviços de Redes Sociais *Online*. Diferente do senso comum, estes documentos são detalhados, pormenorizados tanto em relação as condições de acesso quanto em relação a quais são e como estão estruturados os dados nos Sistemas de Gerenciamento de Banco de Dados das instituições detentoras dos Serviços de Redes Sociais *Online*.

De forma prática, pode-se observar a complexidade das estruturas de dados dos Serviços de Redes Sociais *Online* nas APIs pela sua a diversificação de tipos de conteúdo, de tipos de formatos, de formas de apresentação, de diferentes mídias, e da própria forma que estes conteúdos se relacionam com Referenciados, páginas, grupos e outras entidades – dados pessoais, postagens, curtidas, comentários, compartilhamentos, *hyperlinks* de conteúdos externos, fotografias, vídeos, vídeos ao vivo, entre outros. Esta diversificação também dificulta a compreensão de qual é a seleção de conjuntos de dados de Referenciados que estão disponíveis e circulantes tanto internamente as instituições detentoras como para Agentes Externos.

Soma-se a este processo as mais variadas habilidades necessárias ao profissional envolvido neste tipo de análise. Exige-se conhecimentos desenvolvidos em diversas Áreas do Conhecimento, com destaque a Ciência da Informação, a Ciência da Computação, o Direito, a Estatística, a Lógica e a Matemática, além habilidades ligadas a Áreas do Conhecimento contemporâneas e ainda em desenvolvimento, tais como a Ciência de Dados.

Em síntese, trata-se de uma análise dependente de conhecimentos multidisciplinares sobre protocolos de transferência e interoperabilidade de dados em sistemas de informação, linguagens de programação, construção e elaboração de consultas em banco de dados, desenvolvimento de algoritmos, aplicação de modelos de estatística e de lógica, sistemas de entradas para restrições de acesso, APIs voltadas ao protocolo HTTP e HTTPS, linguagens de marcação HTML, JSON e XML e *Business Intelligence*. Além

destes conhecimentos, adiciona-se a necessidade de conhecimentos sobre os serviços, suas características e – para além – uma seleção multidisciplinar de conhecimentos a depender do objetivo de cada análise.

Constatou-se que o desenvolvimento e a aplicação de abordagem metodológica e de um procedimento padronizado podem ser elementos-chave para sistematizar os dados coletados sobre estas estruturas e trazer, especialmente aquele que investiga – o coletor – um ambiente para uma análise mais criteriosa sobre o que acontece com os dados dentro e fora destes serviços, afetando diretamente aspectos inerentes a privacidade dos dados circulantes.

As Modelagens Direta e de Segunda Ordem tentam mitigar a críspação entre a complexidade do contexto e o potencial de demandas de análise a partir das estruturas de dados dos serviços, aonde fica evidente que esse processo fica encoberto por diversas camadas que ofuscam a compreensão do que de fato é ou não é acessível nestes serviços.

Mesmo observando dados da Modelagem Direta, há uma significativa opacidade dada em relação a complexidade de analisar diretamente dados das estruturas, como as que são referentes as Visões, os Atributos e as Relações dos Serviços de Redes Sociais *Online*, especialmente devido à quantidade de entidades disponíveis para acesso via API (ver Figura 32, seção 2.3) – por isso se entende como necessária um procedimento padronizado que permita a criação de novas camadas de interpretação, como a Modelagem de Segunda Ordem.

Os dados da Modelagem de Segunda Ordem (e demais dados que poderão ser elaborados a partir dos dados da Modelagem Direta) podem dar subsídios necessários para a análise sobre o que é ou não passível de coleta de dados nestes serviços e potenciais canais que (mesmo sem a intenção das instituições detentoras) são passíveis de diminuir a privacidade de dados dos Referenciados, independente se estes são pessoas físicas ou jurídicas.

Não cabe aqui um juízo de valor sobre aspectos inerentes a privacidade dos dados, menos ainda descreditar análises já realizadas sobre o assunto, mas auferir que uma abordagem metodológica sobre estas estruturas de dados podem apontar caminhos para análises mais padronizadas, especialmente abrindo espaço para uma possível comparabilidade analítica sobre os mais diversos serviços disponíveis.

A ideia central é contribuir para se sair de uma observação das estruturas dos serviços pelo senso comum. Se é difícil (ou mesmo impossível) a um investigador ter acesso aos bancos de dados das instituições detentoras dos Serviços de Redes Sociais *Online*, a saída é estruturar análises a partir de suas APIs (já que estas descrevem boa parte dos dados que estão armazenados) e um caminho possível é coletando dados de suas estruturas (Modelagem Direta) e transformando os dados coletados em uma Modelagem de Segunda Ordem e, assim, aumentando a capacidade de observação sobre como os Atributos das Visões dos Serviços de Redes Sociais *Online* estão acessíveis por meio das APIs.

Ou seja, conclui-se que, ao coletor, um dos contatos possíveis entre as estruturas de dados e as suas análises são as APIs: aonde requisições das APIs funcionam como ponto de entrada para a interoperabilidade de conjuntos de dados de Referenciados com estas instituições e, ao mesmo tempo, são quase invisíveis aos observadores e participantes dos serviços, especialmente em seu uso cotidiano.

Ao conectar um aplicativo externo para a elaboração de um *quiz*, ao realizar a entrada (*log in*) em um jogo com os dados do perfil, ao gerenciar uma página pessoal e em outras inúmeras atividades, são abertas conexões para coleta de dados por aplicações externas. Estas conexões podem servir como catalisador para a execução de potenciais ações e atividades prejudiciais à privacidade dos Referenciados? É no momento da coleta de dados que estes conjuntos podem ser copiados para outros serviços (e para outros bancos de dados), localizados em outras instituições, com sedes em outros

países. Cabe agora a cada coletor definir suas perguntas e analisá-las de acordo com seus objetivos.

Como reflexão para o futuro, espera-se que a abordagem proposta seja base para o início de estudos sobre aspectos relacionados ao compartilhamento de conjuntos de dados de Referenciados não só relacionados ao elo entre APIs e a privacidade de dados, mas também como método de investigação sobre direitos autorais e licenciamento, integração de dados e entre outros.

REFERÊNCIAS

REFERÊNCIAS

- ABELLO, J.; PARDALOS, P. M.; RESENDE, M. G. C. (ed.). **Handbook of massive data sets**. Norwell: Kluwer Academic Publishers, 2002.
- ACQUISTI, A.; GROSS, R. Imagined Communities: Awareness, Information Sharing, and Privacy on the Facebook. *In*: DANEZIS, G.; GOLLE, P. (ed.). **Privacy Enhancing Technologies**. Berlim: Springer Berlin Heidelberg, 2006. v. 4258, p. 36–58.
- ADAMIC, L. A.; ADAR, E. Friends and neighbors on the Web. **Social Networks**, Amsterdã, v. 25, n. 3, p. 211–230, Jul. 2003.
- AFFONSO, E. P.; SANT'ANA, R. C. G. **Anonimização de metadados de imagem digital por meio do modelo k-anonimato**. *In*: ENCONTRO NACIONAL DE PESQUISA EM CIÊNCIA DA INFORMAÇÃO, 16., 2015, João Pessoa. **Anais [...]**. João Pessoa: ANCIB, 2015, p. 1-8.
- ALEXA. **The top 500 sites on the web**. 2021. Disponível em: <https://www.alexa.com/topsites>. Acesso em: 10 mar. 2021.
- ATKINSON, M. *et al.* The Object-Oriented Database System Manifesto. *In*: KIM, W.; NICOLAS, J. M.; NISHIO, S. (ed.). **Deductive and Object-Oriented Databases**. Amsterdã: North-Holland, 1990. p. 223–240.
- BARBIERI, C. **BI2 - Business Intelligence: modelagem e qualidade**. Rio de Janeiro: Elsevier, 2011.
- BARNES, S. B. A privacy paradox: Social networking in the United States. **First Monday**, Chicago, v. 11, n. 9, Sept. 2006.
- BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The semantic web. **Scientific American**, Nova Iorque, v. 284, n. 5, p. 28–37, May 2001.

BIGGS, N. L.; LLOYD, E. K.; WILSON, R. J. **Graph Theory 1736-1936**. Oxford Oxfordshire; Nova Iorque: Clarendon Press, 1999.

BOLOSKY, W. J. *et al.* **Wire protocol for a media server system**. United States, mar. 2005. Disponível em: <https://patentimages.storage.googleapis.com/cf/5b/79/a437caa5a4ce93/US20020116447A1.pdf>. Acesso em: 15 jul. 2023.

BOX, D. *et al.* **Simple Object Access Protocol (SOAP) 1.1**. Cambridge: World Wide Web Consortium, 2000. Disponível em: <https://www.w3.org/TR/2000/NOTE-SOAP-20000508/>. Acesso em: 15 jul. 2023.

BOYD, D. Facebook's privacy trainwreck: exposure, invasion, and social convergence. **Convergence: The International Journal of Research into New Media Technologies**, Londres, v. 14, n. 1, p. 13–20, Feb. 2008.

BOYD, D.; ELLISON, N. Social network sites: definition, history, and scholarship. **Journal of Computer-Mediated Communication**, Oxford, v. 13, n. 1, p. 210–230, Oct. 2007.

BOYD, D. Making sense of teen life: strategies for capturing ethnographic data in a networked era. *In*: HARGITTAI, E.; SANDVIG, C. (ed.). **Digital research confidential**. Boston. The MIT Press, 2015. p. 79-102.

BUXMANN, P.; HESS, T.; LEHMANN, S. Software as a Service. **WIRTSCHAFTSINFORMATIK**, Amsterdã, v. 50, n. 6, p. 500–503, Dec. 2008.

CASTELLS, M. **O poder da identidade**. Rio de Janeiro: Paz e Terra, 2018. v. 2.

CASTELLS, M. **A sociedade em rede**. 23. ed. Rio de Janeiro: Paz e Terra, 2021. v. 1.

CODD, E. F. **The relational model for database management: version 2**. Reading, Mass: Addison-Wesley, 1990.

CONNOLLY, T. M.; BEGG, C. E. **Database systems: a practical approach to design, implementation, and management**. 6th. Boston: Pearson, 2015.

CONSEIL EUROPÉEN POUR LA RECHERCHE NUCLÉAIRE (CERN). **The birth of the web**. Portal. 2015. Disponível em: <http://home.cern/topics/birth-web>. Acesso em: 4 maio 2017.

COVER, R. **WDDX - Web Distributed Data Exchange**. [S. l.]: Cover Pages, 1998. Disponível em: <http://xml.coverpages.org/wddx.html>. Acesso em: 15 jul. 2023.

DATE, C. J. **The new relational database dictionary: a comprehensive glossary of concepts arising in connection with the relational model of data, with definitions and illustrative examples: [terms, concepts, and examples]**. Sebastopol: O'Reilly, 2016.

DINEV, T.; HART, P. Internet privacy concerns and their antecedents - measurement validity and a regression model. **Behaviour & Information Technology**, Londres, v. 23, n. 6, p. 413–422, Nov. 2004.

- DONATH, J. Signals in Social Supernets. **Journal of Computer-Mediated Communication**, Oxford, v. 13, n. 1, p. 231–251, Oct. 2007.
- DUBEY, A.; WAGLE, D. Delivering software as a service. **The McKinsey Quarterly**, Seattle, v. 6, n. 2007, May 2007.
- FIELDING, R. T. **Architectural Styles and the Design of Network-based Software Architectures**. Irvine: University of California, 2000.
- FLAKE, G. W.; LAWRENCE, S.; GILES, C. L. Efficient identification of Web communities. *In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING*, 6., 2000, Boston. **Proceedings** [...]. New York: ACM Press, 2000. Disponível em: <http://portal.acm.org/citation.cfm?doid=347090.347121>. Acesso em: 17 abr. 2017.
- FOGEL, J.; NEHMAD, E. Internet social network communities: Risk taking, trust, and privacy concerns. **Computers in Human Behavior**, Amsterdã, v. 25, n. 1, p. 153–160, Jan. 2009.
- FUMERO, A.; VACAS, F. S.; ROCA, G. **Web 2.0**. Madri: Fundación Orange, 2007.
- FUNG, B. C. M. (ed.). **Introduction to privacy-preserving data publishing: concepts and techniques**. Boca Ratón: Chapman & Hall/CRC, 2011.
- GARCIA, M. Browsewrap: a unique solution to the slippery slope of the clickwrap conundrum. **Campbell Law Review**, Raleigh, v. 36, n. 1, p. 31–74, Oct. 2013.
- GOLD, N. *et al.* Understanding service-oriented software. **IEEE Software**, Los Angeles, v. 21, n. 2, p. 71–77, Mar. 2004.
- GOURRAUD, C. **Application-programming-interface-based method and system including triggers**. United States, 2002. Disponível em: <https://www.google.com/patents/US20020026473>. Acesso em: 15 jul. 2023.
- GRAD, B.; BERGIN, T. J. History of Database Management Systems. **IEEE Annals of the History of Computing**, Baltimore, v. 31, n. 4, p. 3–5, Jan. 2010.
- HABERMAS, J. **Mudança estrutural da esfera pública investigações sobre uma categoria da sociedade burguesa**. São Paulo: Ed. da Unesp, 2014.
- HARRISON, Z. M. Just click here: Article 2B's failure to guarantee adequate manifestation of assent in click-wrap contracts. **Fordham Intellectual Property, Media and Entertainment Law Journal**, Nova Iorque, v. 8, n. 3, p. 907, Sept. 1998.
- INMON, W. H. **Building the data warehouse**. 4th. Indianápolis: Wiley, 2005.
- INTEROPERABILITY WORKING GROUP. **Definition of Interoperability**. [S. l.]: [S. n.], 2016.
- ISAAK, J.; HANNA, M. J. User data privacy: Facebook, Cambridge Analytica, and Privacy Protection. **Computer**, Washington, v. 51, n. 8, p. 56–59, Aug. 2018.

JORENTE, M. J. V.; SANTOS, P. L. V. A. C.; VIDOTTI, S. A. B. G. Quando as Webs se encontram: social e semântica – promessa de uma visão realizada? **Informação & Informação**, Londrina, v. 14, p. 1–24, 19 dez. 2009. Número suplementar.

KIMBALL, R.; ROSS, M. **The Data Warehouse Toolkit The Complete Guide to Dimensional Modeling**. Nova Iorque: John Wiley & Sons, 2011.

KINGBERG, D. G.; MCCUBBIN, E. M.; MARTIN, W. J. **Method and apparatus for logical data access to a physical relational database**. United Stats, 1998. Disponível em: <https://www.google.com/patents/US5734887>. Acesso em: 15 jul. 2023.

KRASNOVA, H. *et al.* Privacy concerns and identity in online social networks. **Identity in the Information Society**, Berlim, v. 2, n. 1, p. 39–63, Dec. 2009.

LANE, K. **History of APIs**. 2016. Disponível em: <https://history.apievangelist.com/>. Acesso em: 15 jul. 2023.

LEMLEY, M. A. Intellectual Property and Shrinkwrap Licenses. **SSRN Electronic Journal**, Stanford, v. 1995, n. 68, p. 1239–1294, Aug. 2012.

LÉVY, P. **Cyberculture**. Tradução: Robert Bononno. 1. ed. Minneapolis, Estados Unidos da América: University of Minnesota Press, 2001.

LIAUTAUD, B. **e-Business Intelligence: Turning Information into Knowledge into Profit**. Nova Iorque: McGraw-Hill, 2000.

LINKEDIN CORPORATION. **User Agreement**. Sunnyvale: LinkedIn Corporation, 1 fev. 2023a. Disponível em: <https://www.linkedin.com/legal/user-agreement>. Acesso em: 1 ago. 2023.

LINKEDIN CORPORATION. **LinkedIn Developer Solutions**. Sunnyvale: LinkedIn Corporation, 2023b. Disponível em: <https://developer.linkedin.com/>. Acesso em: 29 jul. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - LinkedIn API Documentation**. Sunnyvale: LinkedIn Corporation, 2023c. Disponível em: <https://learn.microsoft.com/en-us/linkedin/>. Acesso em: 29 jul. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - LinkedIn API Documentation - Basic Profile Fields**. Sunnyvale: LinkedIn Corporation, 2023d. Disponível em: <https://learn.microsoft.com/en-us/linkedin/shared/references/v2/profile/basic-profile?context=linkedin%2Fconsumer%2Fcontext>. Acesso em: 29 jul. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - LinkedIn API Documentation - Getting Access to LinkedIn APIs**. Sunnyvale: LinkedIn Corporation, 25 jul. 2023e. Disponível em: <https://learn.microsoft.com/en-us/linkedin/shared/authentication/getting-access?context=linkedin%2Fcontext>. Acesso em: 1 ago. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - LinkedIn API Documentation - LinkedIn API Concepts**. Sunnyvale: LinkedIn Corporation, 2023f. Disponível em: <https://learn.microsoft.com/en-us/linkedin/shared/api-guide/concepts?context=linkedin%2Fcontext>. Acesso em: 29 jul. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - LinkedIn Marketing API Program - Organizations and Brands Overview**. Sunnyvale: LinkedIn Corporation, 22 mar. 2023g. Disponível em: <https://learn.microsoft.com/en-us/linkedin/marketing/integrations/community-management/organizations?view=li-lms-2023-07>. Acesso em: 1 ago. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - Consumer Solutions Platform - Share on LinkedIn**. Sunnyvale: LinkedIn Corporation, 8 maio 2023h. Disponível em: <https://learn.microsoft.com/en-us/linkedin/consumer/integrations/self-serve/share-on-linkedin?context=linkedin%2Fconsumer%2Fcontext>. Acesso em: 1 ago. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - LinkedIn API Documentation - Job Function Codes**. Sunnyvale: LinkedIn Corporation, 2023i. Disponível em: <https://learn.microsoft.com/en-us/linkedin/shared/references/reference-tables/job-function-codes>. Acesso em: 29 jul. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - LinkedIn API Documentation - Volunteer Fields**. Sunnyvale: LinkedIn Corporation, 2023j. Disponível em: <https://learn.microsoft.com/en-us/linkedin/shared/references/fields/volunteer>. Acesso em: 29 jul. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - LinkedIn API Documentation - Share API**. Sunnyvale: LinkedIn Corporation, 2023k. Disponível em: <https://learn.microsoft.com/en-us/linkedin/marketing/integrations/community-management/shares/share-api?context=linkedin%2Fcompliance%2Fcontext&view=li-lms-unversioned&preserve-view=true&tabs=http#update-shares>. Acesso em: 29 jul. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - Authenticating with OAuth 2.0**. Sunnyvale: LinkedIn Corporation, 5 ago. 2023l. Disponível em: <https://learn.microsoft.com/en-us/linkedin/shared/authentication/authentication>. Acesso em: 1 ago. 2023.

LINKEDIN CORPORATION. **Microsoft Learn - LinkedIn - LinkedIn API Documentation - Full Profile Fields**. Sunnyvale: LinkedIn Corporation, 2023m. Disponível em: <https://learn.microsoft.com/en-us/linkedin/shared/references/v2/profile/full-profile?source=recommendations>. Acesso em: 31 jul. 2023.

LINTHICUM, D.; O-BRIEN, M. **B2B Application Integration: e-Business-Enable Your Enterprise**. Boston: Addison-Wesley Professional, 2000.

MALKIN, I.; CONSTANTAKOPOULOU, C.; PANAGOPOULOU, K. **Greek and Roman networks in the Mediterranean**. Londres: Routledge, 2011.

MARIBEL, Y. S.; RAMOS, I. **Business Intelligence: tecnologias da informação na gestão de conhecimento**. 2nd. Lisboa: FCA, 2009.

MERRIAM-WEBSTER. **Definition of DATABASE**. MERRIAM-WEBSTER, 2023. Disponível em: <https://www.merriam-webster.com/dictionary/database>. Acesso em: 15 jul. 2023.

MERRICK, P.; ALLEN, C. **Web interface definition language submission**. World Wide Web Consortium, 22 set. 1997. Disponível em: <https://www.w3.org/TR/NOTE-widl-970922>. Acesso em: 15 jul. 2023.

META PLATFORMS, INC. **Terms of Service**. Menlo Park: Meta Platforms, Inc., 26 jul. 2022. Disponível em: https://www.facebook.com/legal/terms/update?paipv=0&eav=AfZE3MbAZqfxdXI16AjKIVNt5Tc3tywORgYUqKQlgm9TdFK2naiKQav8MRra8Fs-Zk4&_rdr. Acesso em: 1 ago. 2023.

META PLATFORMS, INC. **Meta Marketing API**. Menlo Park: Meta Platforms, Inc., 2023a. Disponível em: <https://developers.facebook.com/docs/marketing-apis>. Acesso em: 1 ago. 2023.

META PLATFORMS, INC. **Meta for Developers**. Menlo Park: Meta Platforms, Inc., 2023b. Disponível em: <https://developers.facebook.com/>. Acesso em: 23 jul. 2023.

META PLATFORMS, INC. **Graph API Overview**. Menlo Park: Meta Platforms, Inc., 2023c. Disponível em: <https://developers.facebook.com/docs/graph-api/overview/>. Acesso em: 22 jul. 2023.

META PLATFORMS, INC. **Graph API Reference - v2.6**. Menlo Park: Meta Platforms, Inc., 2023d. Disponível em: <https://developers.facebook.com/docs/graph-api/reference>. Acesso em: 1 ago. 2023.

META PLATFORMS, INC. **Access Token Guide - Facebook Login**. Menlo Park: Meta Platforms, Inc., 2023e. Disponível em: <https://developers.facebook.com/docs/facebook-login/guides/access-tokens>. Acesso em: 1 ago. 2023.

META PLATFORMS, INC. **Permissions Guide - Facebook Login**. Menlo Park: Meta Platforms, Inc., 2023f. Disponível em: <https://developers.facebook.com/docs/facebook-login/guides/permissions>. Acesso em: 1 ago. 2023.

META PLATFORMS, INC. **Permissions Reference - Graph API**. Menlo Park: Meta Platforms, Inc., 2023g. Disponível em: <https://developers.facebook.com/docs/permissions/reference>. Acesso em: 1 ago. 2023.

MISLOVE, A. *et al.* Measurement and analysis of online social networks. *In: ACM SIGCOMM CONFERENCE ON INTERNET MEASUREMENT*, 7., 2007, São Diego. **Proceedings** [...]. São Diego: ACM Press, 2007. Disponível em: <http://portal.acm.org/citation.cfm?doid=1298306.1298311>. Acesso em: 14 ago. 2015.

- MORENO, J. L. **Who Shall Survive?:** Foundations of Sociometry, Group Psychotherapy, and Sociodrama. 3rd. Nova Iorque: Beacon House, 1955.
- PAPAZOGLU, M. P. **Service-oriented computing: Concepts, characteristics and directions.** *In:* INTERNATIONAL CONFERENCE ON WEB INFORMATION SYSTEMS ENGINEERING (WISE'03), 4., 2003, Roma. **Proceedings** [...]. Piscataway: IEEE, 2003.
- PC MAGAZINE. **Definition of terms of service.** Nova Iorque: ZIFF DAVIS, LLC., 2023.
- RAHAMAN, M. A.; SCHAAD, A.; RITS, M. Towards secure SOAP message exchange in a SOA. *In:* ACM WORKSHOP ON SECURE WEB SERVICES, 3., 2006, Alexandria. **Proceedings** [...]. Nova Iorque: Association for Computing Machinery, 2006. Disponível em: <https://doi.org/10.1145/1180367.1180382>. Acesso em: 15 jul. 2023.
- REIDENBERG, J. R. Lex informatica: The formulation of information policy rules through technology. **Texas Law Review**, Austin, v. 76, n. 3, p. 553, Feb. 1998.
- RODRIGUES, F. A. **Coleta de dados em redes sociais:** privacidade de dados pessoais no acesso via *Application Programming Interface*. Tese (Doutorado em Ciência da Informação) – Faculdade de Filosofia e Ciências, Universidade Estadual Paulista, Marília, 2017.
- RODRIGUES, F. A.; SANT'ANA, R. C. G. Use of Taxonomy of Privacy to Identify Activities Found in Social Network's Terms of Use. **Knowledge Organization**, Baden-Baden, v. 43, n. 4, p. 285–295, Apr. 2016.
- RODRIGUES, F. A.; SANT'ANA, R. C. G. Contextualização de conceitos teóricos no processo de coleta de dados de Redes Sociais Online. **Informação & Tecnologia**, João Pessoa, v. 5, n. 1, p. 18–36, fev. 2018.
- RODRIGUES, F. A.; SANT'ANA, R. C. G. Privacy and Online Social Network: A Model for Analysis of Collecting Personal Data. **Brazilian Journal of Information Science:** research trends, Marília, v. 17, p. 1–26, Jan. 2023.
- SAMARATI, P.; SWEENEY, L. **Protecting privacy when disclosing information:** k-anonymity and its enforcement through generalization and suppression. [S. l.]: Technical report, SRI International, 1998. Disponível em: https://epic.org/privacy/reidentification/Samarati_Sweeney_paper.pdf. Acesso em: 15 jul. 2023.
- SANT'ANA, R. C. G. Ciclo de vida dos dados: uma perspectiva a partir da ciência da informação. **Informação & Informação**, Londrina, v. 21, n. 2, p. 116-142, dez. 2016.
- SANTOS, P. L. V. A. DA C.; SANT'ANA, R. C. G. Dado e Granularidade na perspectiva da Informação e Tecnologia: uma interpretação pela Ciência da Informação. **Ciência da Informação**, Brasília, v. 42, n. 2, p. 11, maio 2015.

- SHETH, A. P. Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics. *In*: GOODCHILD, M. *et al.* (ed.). **Interoperating Geographic Information Systems**. The Springer International Series in Engineering and Computer Science. Boston: Springer US, 1999. p. 5–29.
- SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistema de Banco de Dados**. 7. ed. Barueri: GEN LTC, 2020.
- SINGLA, P.; RICHARDSON, M. Yes, There is a Correlation- From Social Networks to Personal Behavior on the Web. WWW 2008. *In*: WWW 2008, 17., 2008, Beijing. **Proceedings** [...]. Beijing, China: WWW 2008 Organizing Committee, 2008.
- SOLOVE, D. J. **Understanding privacy**. First Harvard University Press paperback edition ed. Cambridge: Harvard University Press, 2009.
- STATISTA. **Number of social media users worldwide from 2017 to 2027**. Hamburgo: Statista, 2023a. Disponível em: <https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/>. Acesso em: 13 jul. 2023.
- STATISTA. **Number of monthly active Twitter users worldwide from 1st quarter 2010 to 1st quarter 2019**. Hamburgo: Statista, 2023b. Disponível em: <https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/>. Acesso em: 16 jul. 2023.
- THE APACHE SOFTWARE FOUNDATION. **mod_rewrite - Apache HTTP Server Version 2.4**. 2017. Disponível em: https://httpd.apache.org/docs/2.4/mod/mod_rewrite.html#rewriterule. Acesso em: 17 jul. 2023.
- TUFEKCI, Z. Can You See Me Now? Audience and Disclosure Regulation in Online Social Network Sites. **Bulletin of Science, Technology & Society**, Jacksonville, v. 28, n. 1, p. 20–36, Dec. 2007.
- TURNER, M.; BUDGEN, D.; BRERETON, P. Turning software into a service. **Computer**, Washington, v. 36, n. 10, p. 38–44, Oct. 2003.
- WISEU, A.; CLEMENT, A.; ASPINALL, J. Situating privacy online: Complex perceptions and everyday practices. **Information, Communication & Society**, Londres, v. 7, n. 1, p. 92–114, 2004.
- W3SCHOOLS. **SQL Data Types for MySQL, SQL Server, and MS Access**. 2023. Disponível em: https://www.w3schools.com/sql/sql_datatypes.asp. Acesso em: 17 jul. 2023.
- WATSON, H. J.; WIXOM, B. H. The Current State of Business Intelligence. **Computer**, Washington, v. 40, n. 9, p. 96–99, Sept. 2007.
- WELLMAN, B.; HAYTHORNTHWAITE, C. A. (ed.). **The Internet in everyday life**. Malden: Blackwell Publishers Ltd., 2002.
- WESTIN, A. F.; SOLOVE, D. J. **Privacy and Freedom**. Nova Iorque: Ig Publishing, 2015.

WIEDERHOLD, G. **Intelligent integration of information**. *In*: THE 1993 ACM SIGMOD INTERNATIONAL CONFERENCE, 18., 1993, Washington. **Proceedings** [...]. Washington: ACM Press, 1993. Disponível em: <http://portal.acm.org/citation.cfm?doid=170035.170118>. Acesso em: 13 jul. 2023.

WIKIPEDIA CONTRIBUTORS. **API**. Wikimedia Foundation, 7 jul. 2023. (Nota técnica).

WILSON, R. J. **Introduction to Graph Theory**. 5th. Harlow: Pearson, 2010.

WINER, D. **XML-RPC Specification**. UserLand Software, 30 jun. 2003. Disponível em: <http://xmlrpc.com/spec.md>. Acesso em: 17 jul. 2023.

WINSBOROUGH, D.; LOVRIC, D.; CHAMORRO-PREMUZIC, T. Personality, privacy and our digital selves. **The Guardian**, Londres, 18 jul. 2016.

WORLD WIDE WEB CONSORTIUM. **4.5.3 The pre-element - HTML5**: Edition for Web Authors. Cambridge: World Wide Web Consortium, 2023. Disponível em: <https://www.w3.org/TR/2011/WD-html5-author-20110809/the-pre-element.html>. Acesso em: 28 jul. 2023.

X CORP. **Terms of Service**. São Francisco: X Corp., 18 maio 2023a. Disponível em: <https://twitter.com/en/tos> . Acesso em: 1 ago. 2023.

X CORP. **X Developer Platform**. São Francisco: X Corp., 2023b. Disponível em: <https://developer.twitter.com/en> . Acesso em: 28 jul. 2023.

X CORP. **X Developer Platform - Docs - Authorizing a request**. São Francisco: X Corp., 2023c. Disponível em: <https://developer.twitter.com/en/docs/authentication/oauth-1-0a/authorizing-a-request>. Acesso em: 1 ago. 2023.

X CORP. **X Developer Platform - Docs**. São Francisco: X Corp., 2023d. Disponível em: <https://developer.twitter.com/en/docs/api-reference-index#twitter-api-standard>. Acesso em: 29 jul. 2023.

X CORP. **API reference index**. São Francisco: X Corp., 2023e. Disponível em: <https://developer.twitter.com/en/docs/api-reference-index#twitter-api-standard>. Acesso em: 28 jul. 2023.

X CORP. **X Developer Platform - Docs - Standard search API**. São Francisco: X Corp., 2023f. Disponível em: <https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/api-reference/get-search-tweets>. Acesso em: 1 ago. 2023.

X CORP. **X Developer Platform - Docs - X API Documentation**. São Francisco: X Corp., 2023g. Disponível em: <https://developer.twitter.com/en/docs/twitter-api>. Acesso em: 1 ago. 2023.

X CORP. **X Developer Platform - Docs - Tweet object**. São Francisco: X Corp., 2023h. Disponível em: <https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/object-model/tweet>. Acesso em: 28 jul. 2023.

X CORP. **X Developer Platform - Docs - GET collections/list**. São Francisco: X Corp., 2023i. Disponível em: <https://developer.twitter.com/en/docs/twitter-api/v1/tweets/curate-a-collection/api-reference/get-collections-list>. Acesso em: 28 jul. 2023.

X CORP. **X Developer Platform - Docs - Authentication overview**. São Francisco: X Corp., 2023j. Disponível em: <https://developer.twitter.com/en/docs/authentication/overview>. Acesso em: 1 ago. 2023.

X CORP. **X Developer Platform - App only authentication and OAuth 2.0 Bearer Token**. São Francisco: X Corp., 2023k. Disponível em: <https://developer.twitter.com/en/docs/authentication/oauth-2-0/application-only>. Acesso em: 1 ago. 2023.

X CORP. **X Developer Platform - Docs - GET statuses/retweets_of_me**. São Francisco: X Corp., 2023l. Disponível em: https://developer.twitter.com/en/docs/twitter-api/v1/tweets/post-and-engage/api-reference/get-statuses-retweets_of_me. Acesso em: 29 jul. 2023.

YOUNG, A. L.; QUAN-HAASE, A. Information revelation and internet privacy concerns on social network sites: a case study of Facebook. *In*: INTERNATIONAL CONFERENCE ON COMMUNITIES AND TECHNOLOGIES. 4., 2009, Ogden. **Proceedings** [...]. University Park, United States: ACM Press, 2009. Disponível em: <http://portal.acm.org/citation.cfm?doid=1556460.1556499>. Acesso em: 17 abr. 2017.

SOBRE O AUTOR

SOBRE O AUTOR

Fernando de Assis Rodrigues é doutor e mestre em Ciência da Informação pela Universidade Estadual Paulista (UNESP). Possui especialização em Sistemas para Internet pelo Centro Universitário Eurípides de Marília (UNIVEM) e é bacharel em Sistemas de Informação pelo Centro Universitário Sagrado Coração (UNISAGRADO). Realizou estágio de pós-Doutoramento em Ciência da Informação na UNESP, desenvolvendo a pesquisa intitulada *Fundamentos Teóricos para Coleta de Dados de Redes Sociais Online*. É líder dos Grupos de Pesquisa do *AmazonDataTech* da Universidade Federal do Pará (UFPA) e do *Tecnologias de Acesso a Dados* (UNESP). Também é membro do Grupo de Pesquisa *Novas Tecnologias em Informação* (UNESP). É membro de diversos comitês da Universidade Federal do Pará, membro das Comissões Editoriais dos Periódicos *Revista Eletrônica Competências Digitais para a Agricultura Familiar*, *DATA-Read* e *Revista Conexões*, além de parecerista de periódicos científicos de diversas Áreas do Conhecimento. Atua nas áreas da Ciência da Informação e da Ciência da Computação, com ênfase em Coleta de Dados e Tecnologias de Informação e Comunicação, focado principalmente em Serviços de Redes Sociais *Online* e Privacidade. É Diretor da Faculdade de Arquivologia da Universidade Federal do Pará (mandatos 2020-2022 e 2023-2024).

Professor Adjunto no Instituto de Ciências Sociais Aplicadas, lotado na Faculdade de Arquivologia e no Programa de Pós-Graduação em Ciência da Informação da Universidade Federal do Pará. Também é estagiário de pós-doutoramento em Geofísica na Universidade Federal do Pará.

SOBRE O LIVRO

CATALOGAÇÃO NA PUBLICAÇÃO (CIP)

Telma Jaqueline Dias Silveira
CRB 8/7867

FORMATO

16 x 23cm

NORMALIZAÇÃO

Elizabete C. de Souza de Aguiar Monteiro
CRB - 8/7963
Camila Lopes

TIPOLOGIA

Adobe Garamond Pro

CAPA E DIAGRAMAÇÃO

Gláucio Rogério de Moraes

PRODUÇÃO GRÁFICA

Giancarlo Malheiro Silva
Gláucio Rogério de Moraes

ASSESSORIA TÉCNICA

Renato Geraldi

OFICINA UNIVERSITÁRIA

Laboratório Editorial
labeditorial.marilia@unesp.br

2024

O desenvolvimento de redes de inter-relacionamento entre indivíduos, seja formada por afinidades em comum ou pela participação em estruturas formais como em uma sociedade organizada, se mistura com a história do desenvolvimento do ser humano.

Impulsionados pela disponibilidade de conexão da internet, surgiram sistemas de informação desenvolvidos com o objetivo de fornecer suporte às redes de informações e de inter-relacionamento entre indivíduos, grupos de indivíduos e instituições, atualmente, na modalidade de serviços *online* - os Serviços de Redes Sociais *Online*.

Como em toda nova Tecnologia de Informação, o uso destes serviços traz diversas preocupações à sociedade, entre elas, questões relacionadas a privacidade dos dados pessoais – que, em muitas vezes, as afirmações sobre a privacidade dos dados nestes serviços são sustentadas apenas no senso comum.

Propõe-se uma abordagem metodológica que permita identificar as estruturas de dados disponíveis, padronizar a coleta de dados e estabelecer critérios lógicos para analisar os dados circulantes nos Serviços de Redes Sociais *Online*. Esta abordagem metodológica pode auxiliar no aprofundamento de pesquisas e de investigações realizadas por acadêmicos, por comunicólogos ou por profissionais de outras áreas do conhecimento que investigam o tema.

ISBN 978-65-5954-468-4



9 786559 544684