Documentação do Banco de Dados de Metadados de Produtos da Amazon

---

Sumário

1. Introdução
2. Diagrama de Esquema do Banco de Dados
3. Dicionário de Dados
    - 3.1. Tabela: Grupo
    - 3.2. Tabela: Produto
    - 3.3. Tabela: Similare
    - 3.4. Tabela: Categoria
    - 3.5. Tabela: Produto_categoria
    - 3.6. Tabela: Cliente
    - 3.7. Tabela: Review
4. Explicação do Código
    - 4.1. Extração de Dados e População do Banco de Dados
        - 4.1.1. `CriaTabelas.py`
        - 4.1.2. `populate_tables.py`
    - 4.2. Implementação do Dashboard
        - 4.2.1. `dashboard.py`
5. Consultas Realizadas no Dashboard
6. Conclusão
7. Referências

---

1. Introdução

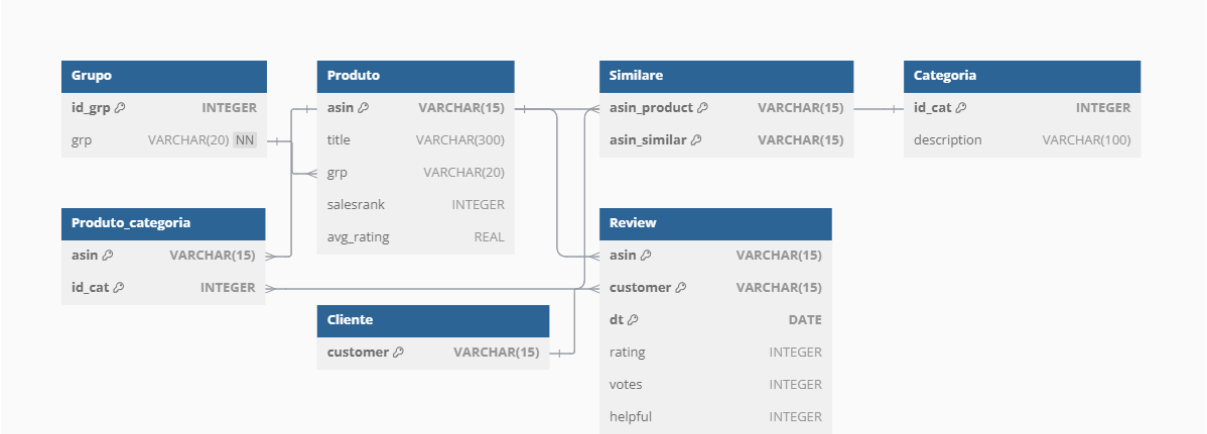
O objetivo deste projeto é projetar e implementar um banco de dados relacional para armazenar e analisar metadados de produtos da Amazon, incluindo avaliações e classificações de usuários. O banco de dados é populado com dados do "Amazon product co-purchasing network metadata", fornecido pelo Stanford Network Analysis Project (SNAP). O projeto inclui a criação do esquema do banco de dados, a inserção dos dados fornecidos e o desenvolvimento de um dashboard para realizar consultas específicas e gerar relatórios.

O ambiente de desenvolvimento consiste em Python para scripting e PostgreSQL como o sistema de gerenciamento de banco de dados relacional (RDBMS). Os scripts Python interagem diretamente com o banco de dados PostgreSQL utilizando comandos SQL, sem camadas intermediárias de software.

---

2. Diagrama de Esquema do Banco de Dados

Abaixo está o diagrama Entidade-Relacionamento (ER) representando o esquema do banco de dados:



Importante notar que o esquema do banco de dados acima segue o modelo relacional e foi construído visando respeitar as regras de projeto capazes de atingir pelo menos a **Terceira Forma Normal**, forma de alto nível no tocante ao processo de normalização de relações em bancos de dados.

Nota-se que o esquema como um todo respeita desde a **Primeira Forma Normal** na medida em que, em todas as relações previstas, seus domínio somente admitem valores atômicos (indivisíveis).

A **Segunda Forma Normal** também se faz observada, já que a relação **Review**, como a única relação com chave primária composta por mais de um atributo e que possui atributos adicionais não-chaves, para a qual faz sentido **conferir existência de dependência funcional total**, caso, por exemplo, seja removido unicamente qualquer atributo não-chave da relação, haverá perda total da dependência dos atributos restantes em relação ao conjunto dos atributos que representam a chave primária.

E, por fim, fica evidente que as relações acima também alcançam a **Terceira Forma Normal**, já que estando na Segunda Forma Normal não possuem também **nenhuma relação com dependência transitiva** (não exclusivamente direta) entre atributos não-principais e o(s) que compõe(m) cada chave primária.

3. Dicionário de Dados

O dicionário de dados fornece uma descrição detalhada de cada tabela, incluindo atributos, tipos de dados, chaves primárias, chaves estrangeiras e quaisquer restrições.

3.1. Tabela: `Grupo`

- **Descrição**: Armazena os diferentes grupos de produtos (ex.: Livro, DVD, Música).
- **Atributos**:

- **`id_grp`** (INTEGER, Chave Primária, Auto-incremento): Identificador único para cada grupo.
- **`grp`** (VARCHAR(20), Único, Não Nulo): Nome do grupo de produtos.

3.2. Tabela: `Produto`

- **Descrição**: Contém informações sobre cada produto.
- **Atributos**:
  - **`asin`** (VARCHAR(15), Chave Primária): Número Padrão de Identificação da Amazon.
  - **`title`** (VARCHAR(300)): Título do produto.
  - **`grp`** (VARCHAR(20), Chave Estrangeira): Grupo ao qual o produto pertence.
  - **`salesrank`** (INTEGER): Posição no ranking de vendas do produto.
  - **`avg_rating`** (REAL): Classificação média do produto.
- **Restrições**:
  - Chave Estrangeira **(`grp`)** referencia **`Grupo(grp)`** com **`ON UPDATE CASCADE`**

3.3. Tabela: `Similare`

- **Descrição**: Representa a relação entre produtos e seus produtos similares.
- **Atributos**:
  - **`asin_product`** (VARCHAR(15), Chave Primária, Chave Estrangeira): ASIN do produto principal.
  - **`asin_similar`** (VARCHAR(15), Chave Primária, Chave Estrangeira): ASIN do produto similar.
- **Restrições**:
  - Chave Primária composta por (**`asin_product`, `asin_similar`**).
  - Chave Estrangeira (**`asin_product`**) referencia **`Produto(asin)`** com **`ON UPDATE CASCADE`**.
  - Chave Estrangeira (**`asin_similar`**) referencia **`Produto(asin)`** com **`ON UPDATE CASCADE.`**

3.4. Tabela: `Categoria`

- **Descrição**: Armazena categorias e subcategorias de produtos.
- **Atributos**:
  - **`id_cat`** (INTEGER, Chave Primária): Identificador único para cada categoria.
  - **`description`** (VARCHAR(100)): Descrição da categoria.
  - **`cat_sup`** (INTEGER, Chave Estrangeira): Identificador da categoria superior.
- **Restrições**:
  - Chave Estrangeira **(`cat_sup`**) referencia **`Categoria(id_cat)`** com **`ON UPDATE CASCADE`**.

3.5. Tabela: `Produto_categoria`

- **Descrição**: Relação muitos-para-muitos entre produtos e categorias.
- **Atributos**:
  - **`asin`** (VARCHAR(15), Chave Primária, Chave Estrangeira): ASIN do produto.

    - **`id_cat`** (INTEGER, Chave Primária, Chave Estrangeira): Identificador da categoria.
- **Restrições**:
    - Chave Estrangeira (**`asin`**) referencia **`Produto(asin)`** com **`ON UPDATE CASCADE`**.
    - Chave Estrangeira (**`id_cat`**) referencia **`Categoria(id_cat)`** com **`ON UPDATE CASCADE`**.

3.6. Tabela: `Cliente`

- **Descrição**: Contém informações sobre clientes.
- **Atributos**:
    - **`customer`** (VARCHAR(15), Chave Primária): Identificador único para cada cliente.

3.7. Tabela: `Review`

- **Descrição**: Armazena avaliações feitas por clientes sobre produtos.
- **Atributos**:
    - **`asin`** (VARCHAR(15), Chave Primária, Chave Estrangeira): ASIN do produto avaliado.
    - **`customer`** (VARCHAR(15), Chave Primária, Chave Estrangeira): Identificador do cliente que fez a avaliação.
    - **`dt`** (DATE, Chave Primária): Data da avaliação.
    - **`rating`** (INTEGER): Classificação dada pelo cliente.
    - **`votes`** (INTEGER): Número de votos que a avaliação recebeu.
    - **`helpful`** (INTEGER): Número de votos úteis.
- **Restrições**:
    - Chave Primária composta por **(`asin`, `customer`, `dt`)**.
    - Chave Estrangeira (**`asin`**) referencia **`Produto(asin)`** com **`ON UPDATE CASCADE`**.
    - Chave Estrangeira (**`customer`**) referencia **`Cliente(customer)`** com **`ON UPDATE CASCADE`**.

---

4. Explicação do Código

4.1. Extração de Dados e População do Banco de Dados

4.1.1. `CriaTabelas.py`

- **Objetivo**: Automatiza a criação do banco de dados e suas tabelas, e inicia o processo de inserção de dados.
- **Principais Funções**:
    1. **`connect_to_database()`**: Estabelece conexão com o banco de dados PostgreSQL.
    2. **`create_database()`**: Cria um novo banco de dados PostgreSQL.
    3. **`create_tables()`**: Executa comandos SQL para criar as tabelas.
    4. **povoar_tabelas()**:

- **Fluxo de Trabalho**:
    1. Conecta-se ao banco de dados padrão **`postgres`.**
    2. Cria um novo banco de dados para o projeto.
    3. Conecta-se ao novo banco de dados
    4. Cria tabelas executando comandos SQL chamados pela função **`create_tables`**.
    5. Chama a função **`povoar_tabelas`** para popular o banco de dados.

4.1.2. Função `povoar_tabelas`

- **Finalidade**: Analisa o arquivo de entrada **`amazon-meta.txt`** e insere dados nas tabelas do banco de dados de forma eficiente.
- **Melhorias Implementadas**:
    1. **Consistência nos Tipos de Dados**: Unificação do tipo de dado **`asin`** como **`VARCHAR(15)`** em todas as tabelas.
    2. **Chaves Estrangeiras Adicionadas**: Adicionada chave estrangeira em **`asin_similar`** na tabela **`Similare`** para garantir integridade referencial.
    3. **Chave Primária em `Review`**: Chave primária agora composta por **(`asin`, `customer`, `dt`)** para garantir unicidade das avaliações.
    4. **Chamamento de funções de inserção específicas para cada tipo de dado**.
    5. **Transações**: Utilização de transações para garantir a integridade dos dados.
    6. **Tratamento de Exceções**: Implementação de tratamento de exceções para capturar erros sem interromper todo o processo.
- **Fluxo de Trabalho**:
    1. Abre e lê o arquivo **`amazon-meta.txt`** linha por linha.
    2. Extrai os dados relevantes utilizando manipulação de strings e expressões regulares.
    3. Armazena os dados em listas para inserções em lote.
    4. Popula as tabelas do banco de dados executando inserções em lote.
    5. Comita as transações no banco de dados após cada lote.

4.2. Implementação do Dashboard

4.2.1. `dashboard.py`

- **Finalidade**: Fornece uma interface de linha de comando para os usuários realizarem consultas específicas no banco de dados e visualizarem os resultados.
- **Melhorias Implementadas**:
    1. **Correções nas Consultas SQL**: Ajustes nas consultas para garantir resultados corretos.
    2. **Visualizações Gráficas Aprimoradas**: Uso das bibliotecas **`matplotlib`** e **`seaborn`** para gráficos mais informativos.
    3. **Interatividade Melhorada**: Opções adicionais para o usuário, como exibir gráficos após consultas.
- **Principais Funções**:
    1. **Consultas**:
        - **`listar_comentarios_uteis()`**: Lista os 5 comentários mais úteis e positivos e negativos para um produto.
        - **`listar_produtos_similares()`**: Lista produtos similares com vendas superiores.

        - **`mostrar_evolucao_avaliacoes()`**: Mostra a evolução diária das classificações médias de um produto.
        - **`listar_produtos_lideres()`**: Lista os 10 produtos mais vendidos em cada grupo.
        - **`listar_produtos_melhores_avaliacoes()`**: Lista os 10 produtos com a maior média de avaliações úteis positivas.
        - **`listar_categorias_melhores_avaliacoes()`**: Lista as 5 categorias com a maior média de avaliações úteis positivas.
        - **`listar_clientes_mais_comentarios():`** Lista os 10 clientes que fizeram mais comentários por grupo de produto.
    2. **Visualização**:
        - Funções para exibir gráficos correspondentes às consultas, como **`exibir_grafico_evolucao_avaliacoes()`**.
    3. **Interface do Usuário**:
        - **`menu_interativo()`**: Fornece um menu para os usuários selecionarem consultas a serem executadas.
- **Fluxo de Trabalho**:
    1. Conecta-se ao banco de dados utilizando as configurações fornecidas.
    2. Exibe um menu de opções para o usuário.
    3. Executa a consulta selecionada e exibe os resultados.
    4. Oferece uma opção de visualização gráfica dos dados, quando aplicável.
    5. Continua a exibir opções ao usuário até que ele escolha sair.

---

5. Consultas Realizadas no Dashboard

O dashboard inclui as seguintes consultas, acessíveis a partir de uma interface interativa:

1. Listar os 5 Comentários Mais Úteis e Melhor Avaliados e os 5 Comentários Mais Úteis e Pior Avaliados

Descrição: Retorna os 5 comentários mais úteis com maior avaliação e os 5 mais úteis com menor avaliação para um produto específico.

Consulta SQL:

```
-- Comentários mais úteis e com maior avaliação
SELECT customer, rating, votes, helpful
FROM Review
WHERE asin = %s
ORDER BY helpful DESC, rating DESC
LIMIT 5;

-- Comentários mais úteis e com menor avaliação
SELECT customer, rating, votes, helpful
FROM Review
WHERE asin = %s
ORDER BY helpful DESC, rating ASC
LIMIT 5;
```

2. Listar Produtos Similares com Maiores Vendas

Descrição: Lista produtos similares a um determinado produto que possuem melhores vendas (menor **`salesrank`**).

Consulta SQL:

```
SELECT p2.asin, p2.title, p2.salesrank
FROM Produto p1
JOIN Similare s ON p1.asin = s.asin_product
JOIN Produto p2 ON s.asin_similar = p2.asin
WHERE p1.asin = %s AND p2.salesrank < p1.salesrank
ORDER BY p2.salesrank ASC;
```

3. Mostrar a Evolução Diária das Médias de Avaliação de um Produto

Descrição: Exibe a evolução das avaliações médias de um produto ao longo do tempo.

Consulta SQL:

```
SELECT dt, AVG(rating) AS avg_rating
FROM Review
WHERE asin = %s
GROUP BY dt
ORDER BY dt ASC;
```

4. Listar os 10 Produtos Líderes de Venda em Cada Grupo de Produto

Descrição: Lista os 10 produtos com melhores vendas em cada grupo de produtos.

Consulta SQL:

```
SELECT asin, title, salesrank, grp
FROM (
   SELECT p.asin, p.title, p.salesrank, p.grp,
        ROW_NUMBER() OVER (PARTITION BY p.grp ORDER BY p.salesrank ASC) AS rn
   FROM Produto p
) sub
WHERE rn <= 10
ORDER BY grp, salesrank;
```

5. Listar os 10 Produtos com a Maior Média de Avaliações Úteis Positivas

Descrição: Retorna os 10 produtos com a maior média de avaliações úteis positivas.

Consulta SQL:

```
SELECT r.asin, p.title, AVG(r.helpful) AS avg_helpful
FROM Review r
JOIN Produto p ON r.asin = p.asin
WHERE r.helpful > 0
GROUP BY r.asin, p.title
ORDER BY avg_helpful DESC
LIMIT 10;
```

6. Listar as 5 Categorias de Produto com a Maior Média de Avaliações Úteis Positivas

Descrição: Lista as 5 categorias com a maior média de avaliações úteis positivas por produto.

Consulta SQL:

```
SELECT c.description, AVG(r.helpful) AS avg_helpful
FROM Categoria c
JOIN Produto_categoria pc ON c.id_cat = pc.id_cat
JOIN Review r ON pc.asin = r.asin
WHERE r.helpful > 0
GROUP BY c.description
ORDER BY avg_helpful DESC
LIMIT 5;
```

7. Listar os 10 Clientes que Mais Fizeram Comentários por Grupo de Produto

Descrição: Lista os 10 clientes que mais fizeram comentários em cada grupo de produtos.

Consulta SQL:

```
SELECT grp, customer, num_comentarios
        FROM (
           SELECT p.grp, r.customer, COUNT(*) AS num_comentarios,
                ROW_NUMBER() OVER (PARTITION BY p.grp ORDER BY COUNT(*)
DESC) as rn
           FROM review r
           JOIN produto p ON r.asin = p.asin
           GROUP BY p.grp, r.customer
        ) sub
        WHERE rn <= 10
        ORDER BY grp, num_comentarios DESC;
```

---

6. Conclusão

Este projeto demonstra com sucesso o processo de projetar e implementar um banco de dados relacional utilizando PostgreSQL e Python para extração, transformação e carregamento de dados (ETL). O esquema do banco de dados está normalizado utilizando a terceira forma normal (3FN) e seguindo as melhores práticas, garantindo integridade e eficiência dos dados. O dashboard oferece uma interface amigável para consultas e análise dos dados, atendendo aos requisitos especificados.

---

7. Referências

1. Stanford Network Analysis Project (SNAP): Amazon product co-purchasing network metadata
2. Garcia-Molina, H., Ullman, J. D., & Widom, J. (2008). *Database Systems: The Complete Book* (2ª ed.). Prentice Hall.
3. Elmasri, R., & Navathe, S. B. (2010). *Fundamentals of Database Systems* (6ª ed.). Addison Wesley.