# Primeiro Trabalho Prático - Banco de Dados I

**Ayumi Aoki Santana**[1], **Erllison Reis**[1], **Maria Luiza Saldanha**[1]

[1]Universidade Federal do Amazonas (UFAM)
Av. Gen. Rodrigo Octávio, 6200 Setor Norte do Campus Universitário -
Coroado, Manaus - AM, 69080-900

## 1. Introdução

Neste trabalho apresentaremos a implementação de um banco de dados voltado para consultas de informações relevantes sobre diferentes produtos, suas categorias e avaliações. O objetivo é desenvolver o esquema do Banco de Dados Relacional que armazene também as informações que foram consideradas úteis para os clientes e as relações entre produtos similares. Além disso será apresentado um Dashboard interativo para a visualização das consultas propostas pela especificação. Os dados para o banco de dados serão extraídos do *"Amazon product co-purchasing network metadata"* que contêm mais de 548.000 produtos e 7.000.000 reviews do site *Amazon*. Para a implementação, foi utilizada a linguagem Python para os scripts de criação das tabelas e consultas, bem como o Sistema de Gerenciamento de Banco de Dados PostgreSQL.

## 2. Esquema do Banco de Dados

O esquema escolhido para esse banco de dados foi construído tendo como base primeiramente a primeira, segunda e terceira formas normais e posteriormente a Forma Normal de Boyce-Codd. O esquema foi estruturado da seguinte forma:

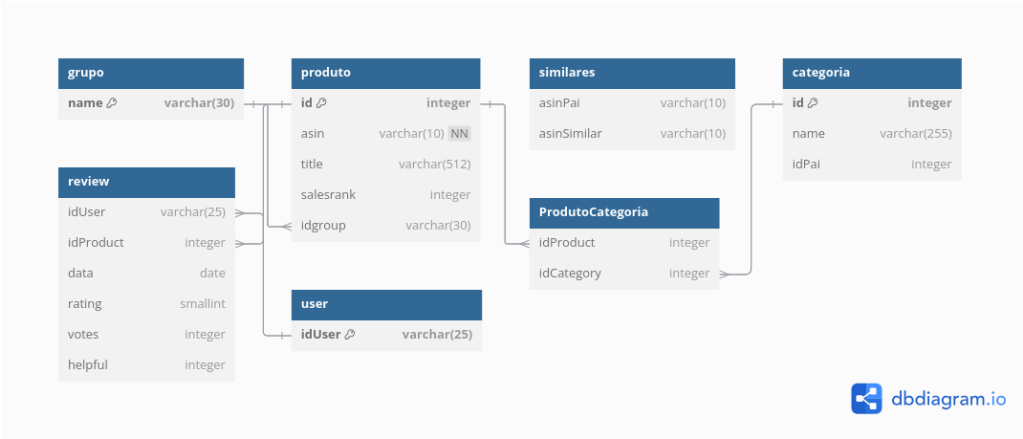


**Figure 1. Esquema do Banco de Dados - Produtos Amazon**

Para a eliminação das redundâncias do banco de dados foi escolhida a Forma Normal de Boyce-Codd, que assegura que o banco estará também na terceira forma. Especificamente, para o BCNF, qualquer depedência funcional deve identificar as tuplas de uma relação. Ou seja, uma tabela em BCNF não permite que existam dependência funcionais no qual o determinante não seja uma superchave.

## 2.1. Tabelas

### 2.1.1. Tabela Grupo

Armazena grupos de produtos.

- **Colunas**:
  - **name** (VARCHAR(30), PK): Nome do grupo, que é a chave primária.

### 2.1.2. Tabela Produto

Contém informações sobre os produtos em si.

- **Colunas**:
  - **id** (INT, PK): Identificador do produto.
  - **asin** (VARCHAR(10), UNIQUE, NOT NULL): Identificador único do produto para a Amazon.
  - **title** (VARCHAR(512)): Título do produto.
  - **salesrank** (INT): Classificação de vendas.
  - **idgroup** (VARCHAR(30), FK): Nome do grupo ao qual o produto pertence (referência à tabela grupo).

A tabela produto possui uma restrição de chave estrangeira que relaciona idgroup com grupo.name. Isso garante que o grupo referenciado exista na tabela grupo e que não seja possível inserir um produto com um id que não esteja na tabela grupo.

A coluna produto.asin é única e não pode ter valores duplicados, marcado com UNIQUE e NOT NULL.

### 2.1.3. Tabela Similares

Armazena relacionamento entre produtos similares.

- **Colunas**:
  - **asinPai** (VARCHAR(10), PK): ASIN do produto pai.
  - **asinSimilar** (VARCHAR(10), PK): ASIN do produto similar.

### 2.1.4. Tabela Categoria

Armazena categorias de produtos.

- **Colunas**:
  - **id** (INT, PK): Identificador único da categoria.
  - **name** (VARCHAR(255)): Nome da categoria.
  - **idPai** (INT): Identificador da categoria pai.

#### 2.1.5. Tabela Review

Contém avaliações de produtos feitas por clientes.

- **Colunas**:
  - **idUser** (VARCHAR(25), PK): Identificador do usuário que fez a avaliação.
  - **idProduct** (INT, PK): Identificador do produto avaliado.
  - **data** (DATE): Data em que a avaliação foi feita.
  - **rating** (SMALLINT): Nota dada ao produto em estrelas.
  - **votes** (INT): Número de votos que a avaliação do cliente recebeu.
  - **helpful** (INT): Número de votos que consideraram a avaliação do cliente útil.

#### 2.1.6. Tabela User

Armazena informações sobre os usuários.

- **Colunas**:
  - **idUser** (VARCHAR(25), PK): Identificador do usuário.

#### 2.1.7. Tabela ProdutoCategoria

Conecta os produtos a suas respectivas categorias.

- **Colunas**:
  - **idProduct** (INT, PK): Identificador do produto.
  - **idCategory** (INT, PK): Identificador da categoria.

## 3. Consultas

Para o Dashboard foram propostas sete possíveis consultas a serem realizadas no banco de dados, sendo elas:

. Dado um produto, listar os 5 comentários mais úteis e com maior avaliação e os 5 comentários mais úteis e com menor avaliação.

```
'a': """
            (
                SELECT 'MAIOR' as tipo,
                * FROM review
                WHERE idproduct = %s
                ORDER BY helpful DESC, rating DESC
                LIMIT 5
            )
            UNION ALL
            (
                SELECT 'MENOR' as tipo,
                * FROM review
```

```
                WHERE idproduct = %s
                ORDER BY helpful DESC, rating ASC
                LIMIT 5
            )
            ORDER BY tipo, helpful DESC;
        """,
```

2. Dado um produto, listar os produtos similares com maiores vendas do que ele.

```
'b': """
                SELECT produtoSimilares .*
                FROM produto p
                JOIN similares ps ON ps.asinpai = p.asin
                JOIN produto produtoSimilares
                    ON produtoSimilares.asin = ps.asinsimilar
                WHERE produtoSimilares.salesrank < p.salesrank
                    AND p.id = %s;
             """,
```

3. Dado um produto, mostrar a evolução diária das médias de avaliação ao longo do intervalo de tempo coberto no arquivo de entrada.

```
    'c': """
                SELECT data, count(*) as qntdReview,
                       round(avg(rating), 4) as mediaAvaliacaoDia
                FROM review
                WHERE idproduct = %s
                GROUP BY data
                ORDER BY data;
             """,
```

4. Listar os 10 produtos líderes de venda em cada grupo de produtos.

```
    'd': """
            WITH RankedProducts AS (
                SELECT grupo.name AS nomeGrupo,
                    ROW_NUMBER() OVER (PARTITION BY produto.idgroup
                    ORDER BY produto.salesrank) AS ranking,
                    produto.*
                FROM produto
                INNER JOIN grupo ON produto.idgroup = grupo.name
                WHERE produto.salesrank > 0
            )
            SELECT *
            FROM RankedProducts
            WHERE ranking <= 10
            ORDER BY nomeGrupo, ranking;
        """,
```

5. Listar os 10 produtos com a maior média de avaliações úteis positivas por produto.

```
'e': """
        SELECT asin, title, idgroup, avgProductRating
        FROM
            (
                SELECT asin, title, idgroup, AVG(r.rating) AS avgPr
                FROM produto p
                INNER JOIN review r ON r.idproduct = p.id
                      AND r.rating > 4 AND r.helpful > 0
                GROUP BY asin, title, idgroup
            )
        AS aux_reviews
        ORDER BY avgProductRating DESC
        LIMIT 10;
    """,
```

6. Listar as 5 categorias de produto com a maior média de avaliações úteis positivas por produto.

```
'f': """
        WITH id_categories AS (
            SELECT round(AVG(review.helpful), 4) AS media_helpful,
            FROM review
            INNER JOIN produto
                   ON review.idproduct = produto.id
            INNER JOIN produtocategoria
                  ON produto.id = produtocategoria.idproduct
            WHERE review.rating >= 4
            GROUP BY produtocategoria.idcategory
            ORDER BY AVG(review.helpful) DESC
            LIMIT 5
        )
        SELECT categoria.name,
               categoria.id, id_categories.media_helpful
        FROM categoria
        JOIN id_categories
        ON categoria.id = id_categories.idcategory
        ORDER BY id_categories.media_helpful DESC;
    """,
```

7. Listar os 10 clientes que mais fizeram comentários por grupo de produto.

```
'g': """
    SELECT *
    FROM
    (
    SELECT *, ROW_NUMBER()
     OVER (PARTITION BY idgroup
```

```
      ORDER BY count_customer_reviews DESC) AS rows
       FROM
       (
     SELECT iduser, idgroup, COUNT(iduser) count_customer_reviews
     FROM produto p
     INNER JOIN review r ON r.idproduct = p.id
     GROUP BY iduser, idgroup
         ) AS aux_reviews
     )
AS aux WHERE rows <= 10;
     """
```