Relatório de Banco de Dados

Renato Lousan da Silva, David Pereira Bessa

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)

Manaus – Am – Brasil

1. Banco de Dados e Esquemas

Inicialmente, foi escolhido a Terceira Forma Normal (FN3) para a criação do esquema do banco de dados. Os domínios dos atributos de cada esquema foram escolhidos através de observação da base de dados para que atendesse da melhor forma os dados da Amazon product co-purchasing network metadata.

O dicionário de dados está representado na figura abaixo :

Tabela : Costumer			
Campo	Descrição	Tipo	NOT NULL
ID_Costumer (PK)	Identificador de cliente	TEXT	SIM

Tabela : GroupProducts			
Campo	Descrição	Tipo	NOT NULL
Title	Descrição do grupo	TEXT	NÃO
ID_Group (PK)	Identificador de grupo	INTEGER	SIM

Tabela : Category			
Campo	Descrição	Tipo	NOT NULL
Title	Descrição da categoria	TEXT	NÃO
ID_Category (PK)	Identificador de categoria	INTEGER	SIM

Tabela : CategoriesByProduct			
Campo	Descrição	Tipo	NOT NULL
ID_CategoriesByProduct (PK)	Identificador de produto por categoria	INTEGER	SIM
ID_Prod (FK)	Identificador do produto	INTEGER	SIM
ID_Cat(FK)	Identificador de categoria	INTEGER	SIM

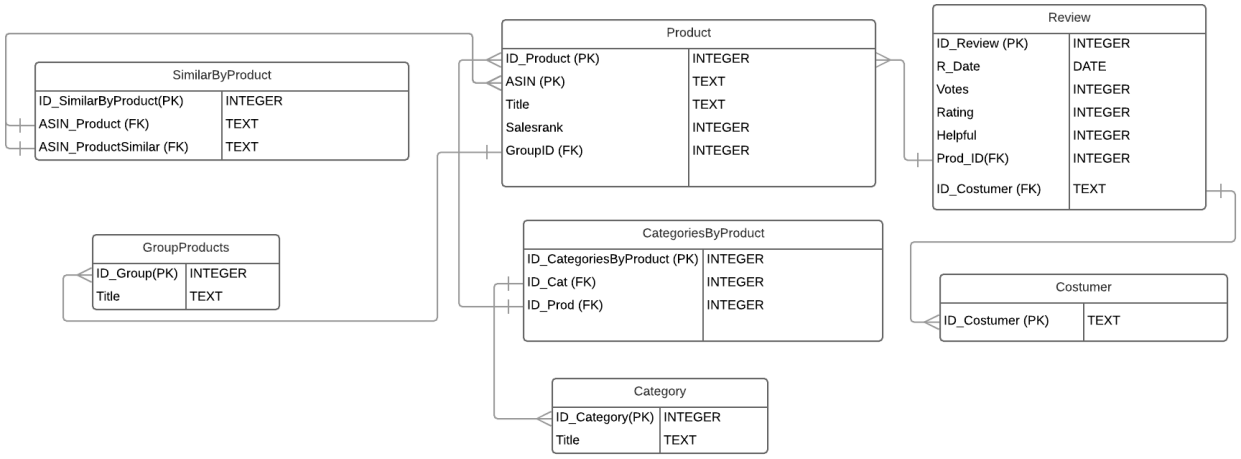
Tabela : SimilarByProduct			
Campo	Descrição	Tipo	NOT NULL
ID_SimilarByProduct (PK)	Identificador de tupla produto	INTEGER	SIM
ASIN_Product (FK)	ASIN do produto	TEXT	SIM
ASIN_ProductSimilar (FK)	ASIN do produto	TEXT	SIM

Tabela : Product			
Campo	Descrição	Tipo	NOT NULL
ID_Product (PK)	O identificador do produto	INTEGER	SIM
ASIN (UK)	ASIN do produto	TEXT	SIM
Title	Titulo do produto	TEXT	NÃO
Salerank	Rank de vendas do produto	INTEGER	NÃO
ID_Group (FK)	Identificador do grupo	INTEGER	SIM

Tabela : Review			
Campo	Descrição	Tipo	NOT NULL
ID_Review	Identificador da avaliação	INTEGER	SIM
R_Date	Data da avaliação	DATE	NÃO
Votes	Quantidade de avaliações	INTEGER	NÃO
Rating	Nota do produto	INTEGER	NÃO
Helpful	Nota de utilidade da avaliação	INTEGER	NÃO
Prod_ID (FK)	Identificador do produto	INTEGER	SIM
ID_Costumer (FK)	Identificador de quem avaliou	INTEGER	SIM

O esquema do banco extraído da massa de dados está representado na figura abaixo :

Figure 1. Figura 2



2. Restrições de Integridade

Para cada tabela foi especificado as seguintes restrições de integridade, as mesmas estão sendo mostradas na figura 2 .

- **Product**: possui restrição de integridade referencial com GroupProducts e de Category
- **Category** : não possui restrição de integridade referencial direcionada para outra tabela .
- **Review** : possui restrição de integridade referencial com Costumer e Product.
- **Costumer**: não possui restrição de integridade referencial direcionada para outra tabela .
- **GroupProducts** : não possui restrição de integridade referencial direcionada para outra tabela .
- **SimilarByProduct** : possui restrição de integridade com a tabela Product.
- **CategoriesByProduct**: possui restrição de integridade referencial com Category e Product.

3. Consultas do Dashboard

1. **Dado produto, listar os 5 comentários mais úteis e com maior avaliações e os 5 comentários mais úteis e com menor avaliação**

```
( select * from review natural join product where asin
= '%s' order by rating desc, helpful desc limit 5)
union all
(select * from review natural join product where asin
= '%s' order by rating desc, helpful desc limit 5);
```

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

```
SELECT p2.id_product, p2.asin, p2.title, p2.salesrank
FROM product p, similarbyproduct, product p2     where
p.asin = asin_product and p2.asin = asin_productsimilar
and p.salesrank <= p2.salesrank and p2.salesrank > 0
and p.asin='%s';
```

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

```
SELECT r_date, avg(rating) from product join review
r on prod_id = id_product where asin ='%s ' group by
r_date order by r_date;
```

4. **Listar os 10 produtos lideres de venda em cada grupo de produtos**

```
 SELECT * FROM (
 SELECT *,rank() OVER ( PARTITION BY id_group ORDER BY
salesrank ASC) FROM
(SELECT * from groupproducts join product p on
p.groupid = groupproducts.id_group where p.salesrank
> 0) as subaux
 ) rank_filter where rank <= 10;
```

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

```
SELECT title, avg(rating) as rating_avg, avg(helpful)
as helpful_avg
FROM review join product p on prod_id = p.id_product
group by p.id_product, p.title order by rating_avg
desc, helpful_avg desc LIMIT 10
```

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

```
 SELECT title, AVG(helpful) as AVG_helpful
 FROM (category JOIN ( categoriesbyproduct JOIN
(SELECT prod_id,helpful from review) AS
produto_avaliacoes on
id_prod = prod_id ) as seila on id_category = id_cat )
AS avaliacoes_categoria
GROUP BY title ORDER BY AVG_helpful DESC LIMIT 5;
```

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

```
SELECT * FROM ( SELECT *,
rank() OVER ( PARTITION BY id_group ORDER BY c DESC)
FROM ( select groupproducts.title,
id_costumer,id_group, count(*) as c from product
join review on prod_id = id_product join groupproducts
on groupid = id_group
natural join costumer group by id_costumer, id_group
order by c desc) as did ) rank_filter
natural join costumerwhere rank <= 10 order by
id_group, rank;
```

4. Arquivos

Para este trabalho, 9 arquivos *.py* foram gerados. 4 deles são apenas classes para a abstração de objetos do escopo deste trabalho (Produto, Review, etc) e estão presentes na pasta **models**. Os outros 5 arquivos são estes:

- **Builder.py** - arquivo principal da construção do banco, responsável por chamar todos os métodos necessários para esta função.

- **Parser.py** - usado para realizar o *parsing* dos dados.
- **ManagerDB.py** - classe que agrupa os métodos de acesso a base de dados.
- **Query.py** - Armazenas os diversos comandos em SQL usados neste trabalho.
- **DashboardServer.py** - implementação do dashboard para consulta dos dados.

5. Como Executar

O projeto foi desenvolvido em Python 3.6. A biblioteca usada para conexão com o SGBD foi a *psycopg2*.

5.1. Construtor do banco

Para executar construtor do banco de dados é bastar executar o arquivo **Builder.py** com os seguintes parâmetros:

- argv[1] - Host
- argv[2] - Usuário
- argv[3] - Nome do banco
- argv[4] - Senha do usuário
- argv[5] - Localização do arquivo que contém os dados.

Exemplo: python3 Builder.py localhost user dashboard senha123 /data/amazon-meta.txt

5.2. Dashboard

O dashboard é executado por meio do arquivo **DashboardServer.py**, os seguintes parâmetros são necessários:

- argv[1] - Host
- argv[2] - Usuário
- argv[3] - Nome do banco
- argv[4] - Senha do usuário

Exemplo: python3 DashboardServer.py localhost user dashboard senha123