PROVA 3 03/02/2022 - SQL

NOME: KLYSMAN REZENDE ALVES

MATRICULA: 2017108779

1. A - Considere o seguinte esquema relacional, referente a um banco de dados que suporta uma indústria de equipamentos eletrônicos produzidos sob encomenda, e produza comandos (SQL) para resolver o que se pede:

1.1.

```
▶ Run SQL
CREATE TABLE CLIENTE (
    CODCLI INTEGER NOT NULL PRIMARY KEY,
    NOME VARCHAR(50) NOT NULL,
    EMAIL VARCHAR(50) NOT NULL
);
```

1.2.

```
▶ Run SQL
CREATE TABLE PRODUTO (      You, 22 minutes a
    CP INTEGER NOT NULL PRIMARY KEY,
    NOMEPROD VARCHAR(50) NOT NULL,
    DESCRI_PROD VARCHAR(150) NOT NULL
);
```

1.3.

```
▶ Run SQL
CREATE TABLE VENDA (
    CODCLI INTEGER NOT NULL,
    CP INTEGER NOT NULL,
    DATA DATE NOT NULL,
    QUANT INTEGER NOT NULL,
    VALOR FLOAT NOT NULL CHECK (VALOR >= 100),
    FOREIGN KEY (CODCLI) REFERENCES CLIENTE(CODCLI) ON DELETE CASCADE,
    FOREIGN KEY (CP) REFERENCES PRODUTO(CP) ON DELETE CASCADE
);
```

1.4.

```
▶ Run SQL
CREATE TABLE CAT_COMP (
    CATEG VARCHAR(50) NOT NULL PRIMARY KEY,
    DESCR_CATEGORIA VARCHAR(50) NOT NULL
);
```

1.5.

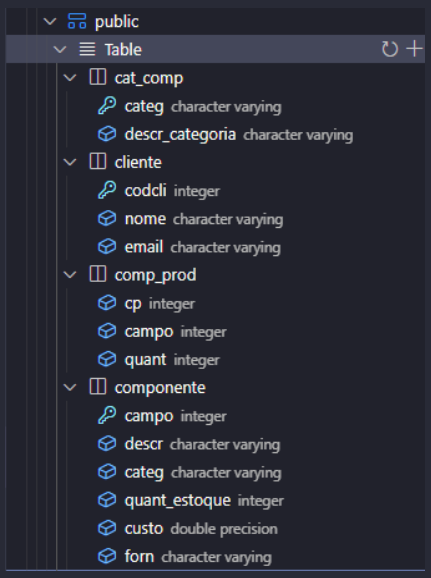
```
▶ Run SQL
CREATE TABLE COMPONENTE (
    CAMPO INTEGER NOT NULL PRIMARY KEY,
    DESCR VARCHAR(150) NOT NULL,
    CATEG VARCHAR(50) NOT NULL,
    QUANT_ESTOQUE INTEGER NOT NULL,
    CUSTO FLOAT NOT NULL,
    FORN VARCHAR(50) NOT NULL,
    FOREIGN KEY (CATEG) REFERENCES CAT_COMP(CATEG)
);
```

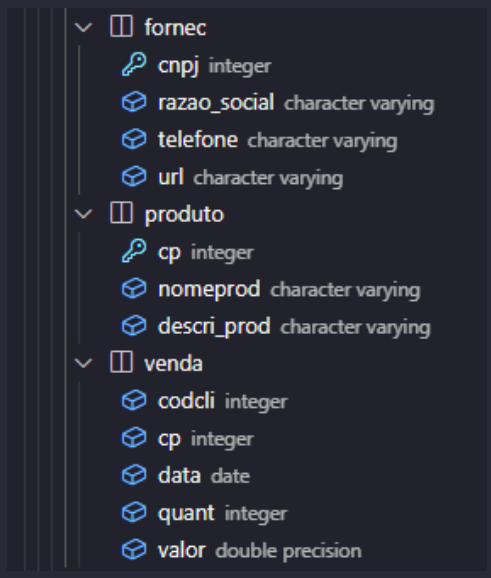
1.6.

```
▶ Run SQL
CREATE TABLE FORNEC (
    CNPJ INTEGER NOT NULL PRIMARY KEY,
    RAZAO_SOCIAL VARCHAR(50) NOT NULL,
    TELEFONE VARCHAR(50) NOT NULL,
    URL VARCHAR(50) NOT NULL
);
```

1.7.

```
▶ Run SQL
CREATE TABLE COMP_PROD (
    CP INTEGER NOT NULL,
    CAMPO INTEGER NOT NULL,
    QUANT INTEGER NOT NULL,
    FOREIGN KEY (CP) REFERENCES PRODUTO(CP) ON DELETE CASCADE,
    FOREIGN KEY (CAMPO) REFERENCES COMPONENTE(CAMPO) ON DELETE CASCADE
);
```





B. Listar o nome dos produtos que tiveram mais de 100 unidades vendidas para o mesmo cliente, na mesma data. Listar também o nome do cliente, a data da venda, e a quantidade vendida.

```
▶ Run SQL
SELECT nome, data, nomeprod, quant
FROM cliente, venda, produto
WHERE cliente.codcli = venda.codcli
AND produto.cp = venda.cp
AND quant > 100;
```

C. Listar os códigos e descrições dos componentes usados para fabricar o produto cujo nome é “SENSOR UV”, juntamente com as quantidades usadas no produto e a quantidade disponível em estoque de cada componente. Produzir a saída em ordem alfabética reversa (Z->A) de código do componente.

```
▶ Run SQL
SELECT nomeprod, comp_prod.cp, comp_prod.campo, comp_prod.quant, componente.descr, componente.quant_estoque
FROM produto, comp_prod, componente
WHERE produto.cp = comp_prod.cp
AND componente.campo = comp_prod.campo
AND nomeprod = "SENSOR_UV"
ORDER BY cp ASC;
```

D. Listar, sem repetições, o código e a descrição de todos os produtos que usam componentes da categoria “OPTOELETRÔNICOS”

```
▶ Run SQL
SELECT DISTINCT produto.cp, descri_prod
FROM produto, comp_prod, componente
WHERE produto.cp = comp_prod.cp
AND componente.campo = comp_prod.campo
AND componente.categ = "OPTOELETRÔNICOS"
GROUP BY cp, descri_prod;
```

E. Produzir um relatório indicando o nome de cada produto e o custo total de componentes utilizados por ele (quantidade multiplicada pelo custo do componente, acumulada para todo o produto), listando apenas produtos cujo custo total fique acima de R$100,00.

```
▶ Run SQL
SELECT nomeprod, SUM(custo * quant)
FROM produto, comp_prod, componente
WHERE produto.cp = comp_prod.cp
AND componente.campo = comp_prod.campo
GROUP BY nomeprod
HAVING SUM(custo * quant) > 100;
```

F. Calcular e apresentar o custo médio dos componentes de cada categoria, junto ao seu código e descrição.

```
▶ Run SQL
SELECT cat_comp.categ, descr_categoria, SUM(custo * quant) / SUM(quant) as Media
FROM cat_comp, componente, comp_prod
WHERE cat_comp.categ = componente.categ
AND componente.campo = comp_prod.campo
GROUP BY cat_comp.categ, descr_categoria;
```

G. Promover um reajuste dos valores (CUSTO) de todos os componentes, aumentando os valores dos semicondutores (CATEG = ‘SEMICOND”) em 10% e reduzir os custos dos demais componentes em 5%. (Dica: use dois comandos)

```
▶ Run SQL
UPDATE componente
SET custo = custo * 1.1
WHERE categ = "SEMICOND";

▶ Run SQL
UPDATE componente
SET custo = custo * 0.95
WHERE categ != "SEMICOND";
```

H. Remover o produto cujo código é 37711321. Explique as possíveis consequências para as tabelas relacionadas, considerando as opções da cláusula ON DELETE que você usou na criação das tabelas (letra a desta questão).

```
▶ Run SQL
111  DELETE FROM produto
112  WHERE cp = 37711321;
```

- Ao criar a tabala utilizando o recurso **ON DELETE CASACADE**, é garantido que todos os dados relacionados ao elemento sejam apagados nas demais tabelas que existe um relacionamento. Caso essa função não seja declarada, teremos um erro SQL state ao executar o comando DELETE.

I. Informar os códigos e as quantidades totais de cada produto vendido entre 01/01/2021 e 31/12/2021, inclusive produtos que estão registrados na tabela PRODUTO e não foram vendidos.

```
▶ Run SQL
115  SELECT cp, SUM(quant)
116  FROM venda
117  WHERE YEAR(data) = 2021
118  GROUP BY cp
119  ORDER BY SUM(quant) DESC;
120
```

QUESTÃO 2 (5 pontos) Imagine a implementação do banco de dados cujo esquema relacional está descrito na Questão 1 usando NoSQL.

A. Escolha um dos tipos de gerenciadores NoSQL descritos em aula (document store, key-value, wide-column store, graph database) e explique como seria possível adaptar o esquema para usá-lo.

DynamoDB - tipo Key-Value

Esse gerenciador é compatível com modelos de armazenamento de chave-valor. Para cada chave existe **n** valores e uma coleção de atributos associados a chave. Assim como nos bancos relacionais podemos criar uma tabela que possui itens (chaves) que possuem **n** atributos. Esses atributos, por exemplo, podem conter as colunas listadas na questão A. Como chave poderíamos atribuir os produtos ou categorias e por fim, os demais atributos e valores para cada produto do banco.

B. Descreva possíveis vantagens e desvantagens no uso da alternativa que você escolheu para implementar este banco de dados, considerando que se pretende usar o gerenciador para carregar um pequeno volume de dados, porém a quantidade de atualizações é mais significativa que a quantidade de consultas.

Nesse projeto, utilizando DynamoDB, teremos como vantagens consultas mais eficientes, desde que o banco seja projetado de maneira adequada. Além disso, ele é rápido e flexível. DynamoDB tem a capacidade de escalonamento automático, caso seja necessário. Este recurso ajuda a sistema a se ajustar conforme a quantidade de tráfego de dados e melhora o desempenho da aplicação, além de reduzir custos.

Como desvantagem, para consultas não previstas no esquema, o custo será alto e lento. E quando pensamos em atualizações, o DynamoDB possui estruturas de dados internas ajustadas aos requisitos levantados o que torna esse procedimento custoso.