Engenharia de Dados

Exercícios:

1) Cenário
A empresa quer disponibilizar um banco de dados PostgreSQL para uso em produção. Temos um time pequeno de dados e esse banco não pode ter downtime. Ele será utilizado por colaboradores em São Paulo, Nova York e Londres com alta frequência de leitura e gravação.

Tarefa
Quais aspectos e configurações devem ser levadas em consideração na disponibilização desse banco de dados? Explique-os.

Alta disponibilidade e latência global - usar alguma ferramenta como o Amazon RDS e AWS Route53 para habilitar a criação de réplicas do banco para caso de falha e de réplicas em assíncrona em regiões geográficas chave, que permitam o roteamento das requisições de leitura para a réplica mais próxima, diminuindo a latência.

Ajuste fino de parâmetros do banco de dados - ajustar configurações de cache, máximo de conexões e autovaccum para evitar overload do banco já que existe alta frequência de leitura e gravação.

Monitoramento - usar alguma ferramenta de monitoramento como o AWS Cloudwatch, para gerar alertas para consumo de disco, falhas de replicação, etc.

2) Cenário
Uma asset manager está expandindo sua operação de gestão de investimentos e precisa de um sistema robusto para gerenciar seus ativos, clientes, portfólios e transações. O sistema deve ser capaz de armazenar informações detalhadas sobre os ativos sob gestão, incluindo diferentes tipos de investimentos como ações, títulos de dívida (bonds) e fundos imobiliários. Além disso, deve gerenciar as contas dos clientes, os portfólios atribuídos a cada cliente, e todas as transações realizadas.

Requisitos Específicos
• **Clientes:** O sistema deve armazenar informações básicas de cadastro dos clientes.
• **Ativos:** Os ativos devem incluir as informações do ativo como nome, preço atual e moeda.
• **Portfólios:** Cada cliente pode ter um ou mais portfólios. Um portfólio pode incluir diversos ativos. É necessário registrar o peso ou a porcentagem de cada ativo no portfólio.
• **Transações:** Todas as compras e vendas de ativos devem ser registradas. •

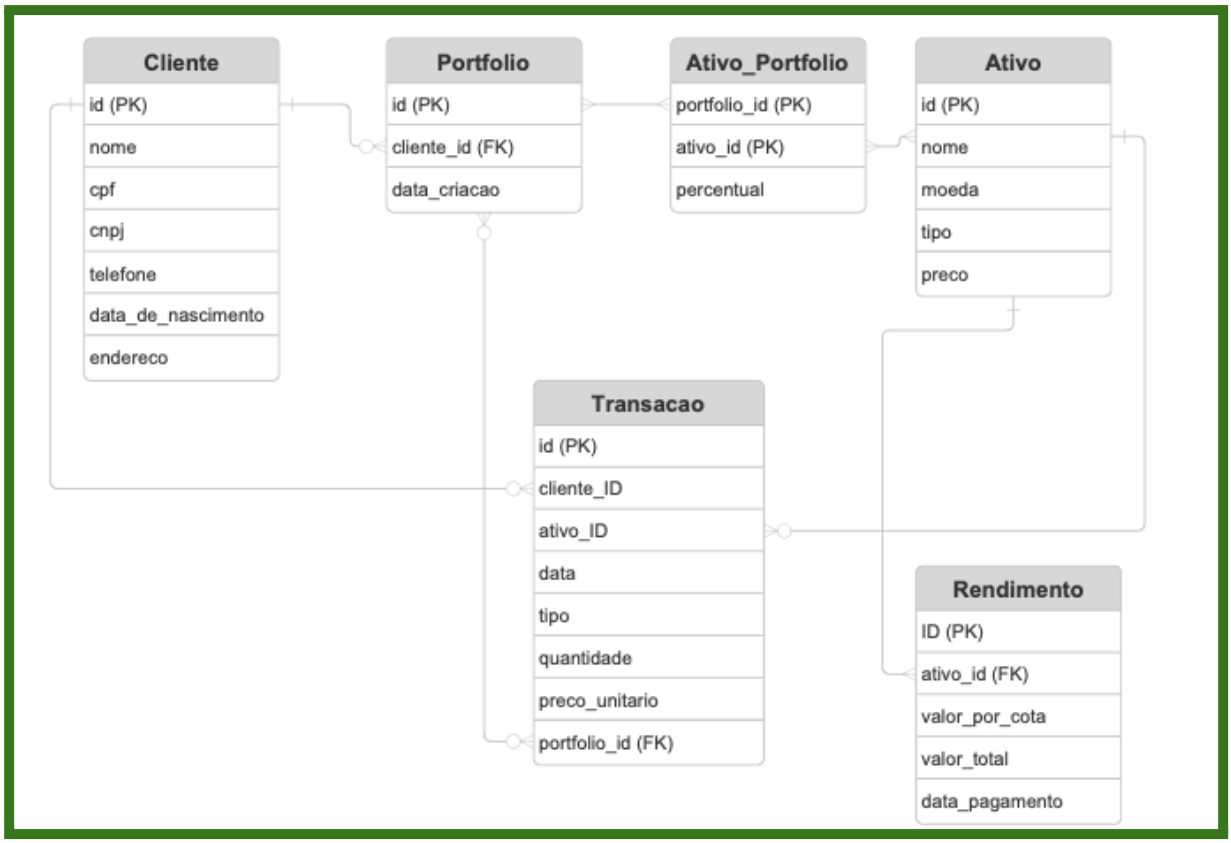
Rendimentos: O sistema deve ser capaz de registrar os rendimentos dos ativos, como dividendos para ações e fundos imobiliários, ou pagamento de cupons para bonds.

Tarefa

Baseando-se nos requisitos acima, crie um modelo Entidade-Relacionamento (ER) detalhado para o sistema de gestão de investimentos da asset manager. Seu modelo deve incluir todas as entidades mencionadas, seus atributos específicos e os relacionamentos entre elas. Considere as melhores práticas de normalização para evitar redundâncias e garantir a integridade dos dados.

Entregáveis

• Um diagrama do modelo ER, mostrando entidades, atributos e relacionamentos. • Uma breve descrição de cada entidade e relacionamento no seu modelo.



Entidades:

Cliente:

- Armazena dados cadastrais como nome, CPF/CNPJ, telefone, data de nascimento e endereço.

Ativo:

- Armazena o nome do ativo, a moeda em que ele é operado, seu tipo, que pode ser uma ação, título de dívida (bond) ou fundo imobiliário por exemplo, e o preço atual.

Portfolio:

- Armazena data de criação do portfólio e depende de um cliente como chave estrangeira.

Ativo_Portfolio:

- Armazena o peso percentual de cada ativo dentro do portfólio, dependendo de um portfólio e de um ativo como chave primária composta.

Transacao:

- Informa qual ativo foi negociado e por qual cliente (como duas chaves estrangeiras), a data, quantidade, preço unitário, tipo (compra/venda) e moeda.

- Pode ser vinculada a um portfólio específico como chave estrangeira opcional, se aplicável.

Rendimento:

- Inclui a data de pagamento, valor total e valor por cota/unidade do ativo de um certo ativo (chave estrangeira).

Relacionamentos:

Cliente 1:N Portfolio: Um cliente pode possuir vários portfólios, cada portfólio pertence a um único cliente.

Portfólio N:M Ativo (via Ativo_Portfolio): Um portfólio pode conter diversos ativos, um ativo pode estar presente em vários portfólios.

Cliente 1:N Transacao: Um cliente pode realizar múltiplas transações.

Transação N:1 Ativo: Cada transação se refere a um único ativo.

Transação N:1 Portfolio (opcional): A transação pode estar associada a um portfólio específico.

Ativo 1:1 Rendimento: Um ativo pode gerar múltiplos rendimentos ao longo do tempo.

3) Cenário

Você é um analista de dados que foi encarregado de otimizar o armazenamento de dados para análises de vendas de uma grande rede de varejo. A empresa possui um sistema de banco de dados relacional tradicional, com um modelo Entidade Relacionamento (ER) que inclui as seguintes entidades principais: Produtos, Vendas, Clientes, Funcionários e Lojas.

Requisitos Específicos

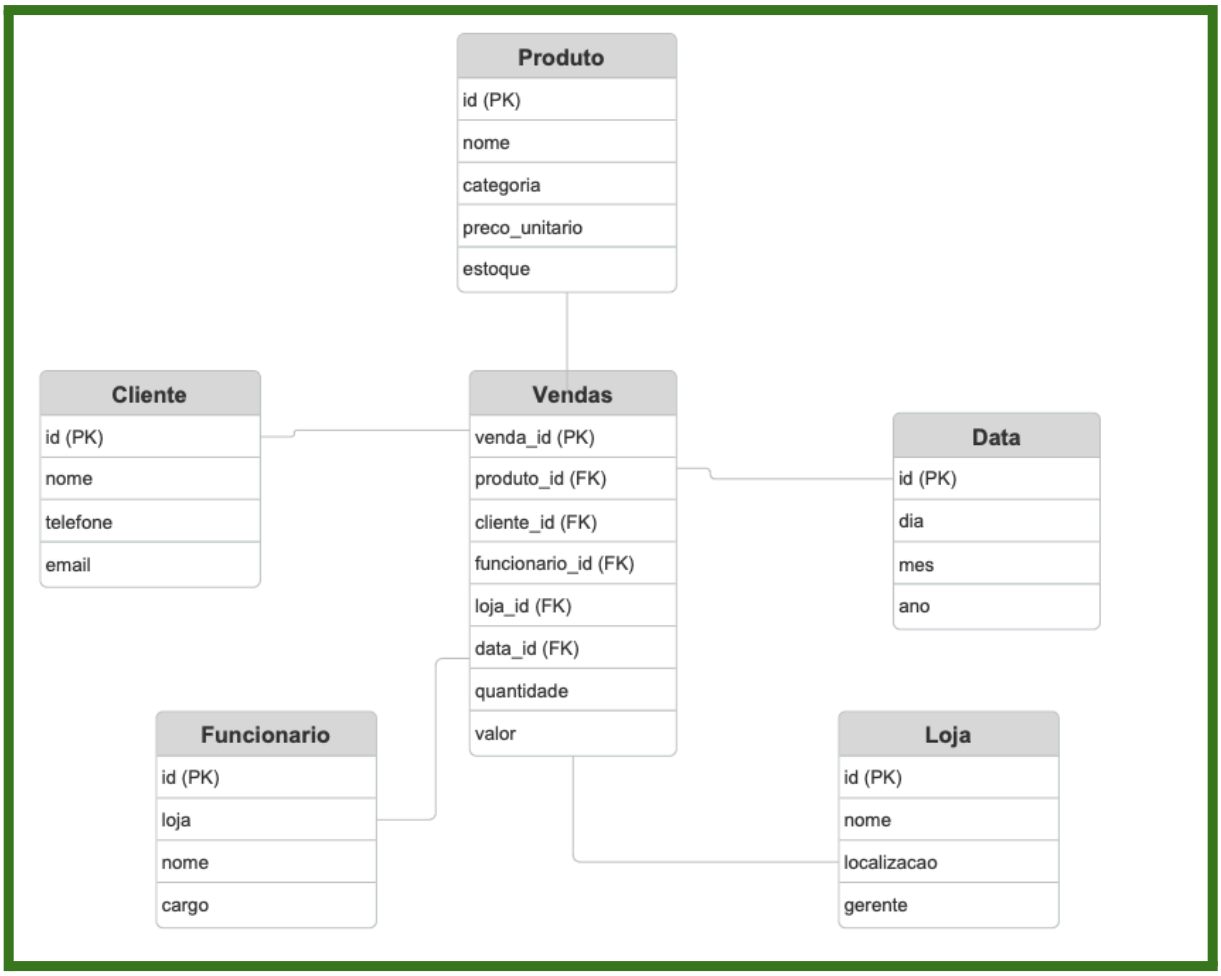
• **Produtos:** Armazena informações sobre os produtos vendidos, incluindo ID do Produto, Nome, Categoria, Preço Unitário e Estoque.
• **Vendas:** Registra cada venda, incluindo ID da Venda, ID do Produto, ID do Cliente, ID do Funcionário, ID da Loja, Data da Venda, Quantidade e Valor Total.
• **Clientes:** Contém dados dos clientes, como ID do Cliente, Nome, E-mail e Telefone.
• **Funcionários:** Mantém informações sobre os funcionários, incluindo ID do Funcionário, Nome, Cargo e ID da Loja em que trabalham.
• **Lojas:** Detalha as lojas da rede, com ID da Loja, Nome, Localização e Gerente.

Tarefa

A partir do modelo ER fornecido, desenvolva um Star Schema para otimizar as análises de vendas da empresa.

Entregáveis

• Um diagrama do Star Schema, identificando claramente a tabela de fatos e as dimensões.
• Uma breve descrição de cada tabelas
• Explique brevemente como você transformou o modelo ER em um Star Schema, destacando as decisões de design mais importantes.



A tabela Vendas é a tabela Fact, e contém apenas as chaves estrangeiras que se relacionam com as outras tabelas, e os dados quantitativos que provavelmente serão mais utilizados em análises, e Cliente, Funcionario, Data e Produto são as Dimensions, que contém as informações de cadastro.

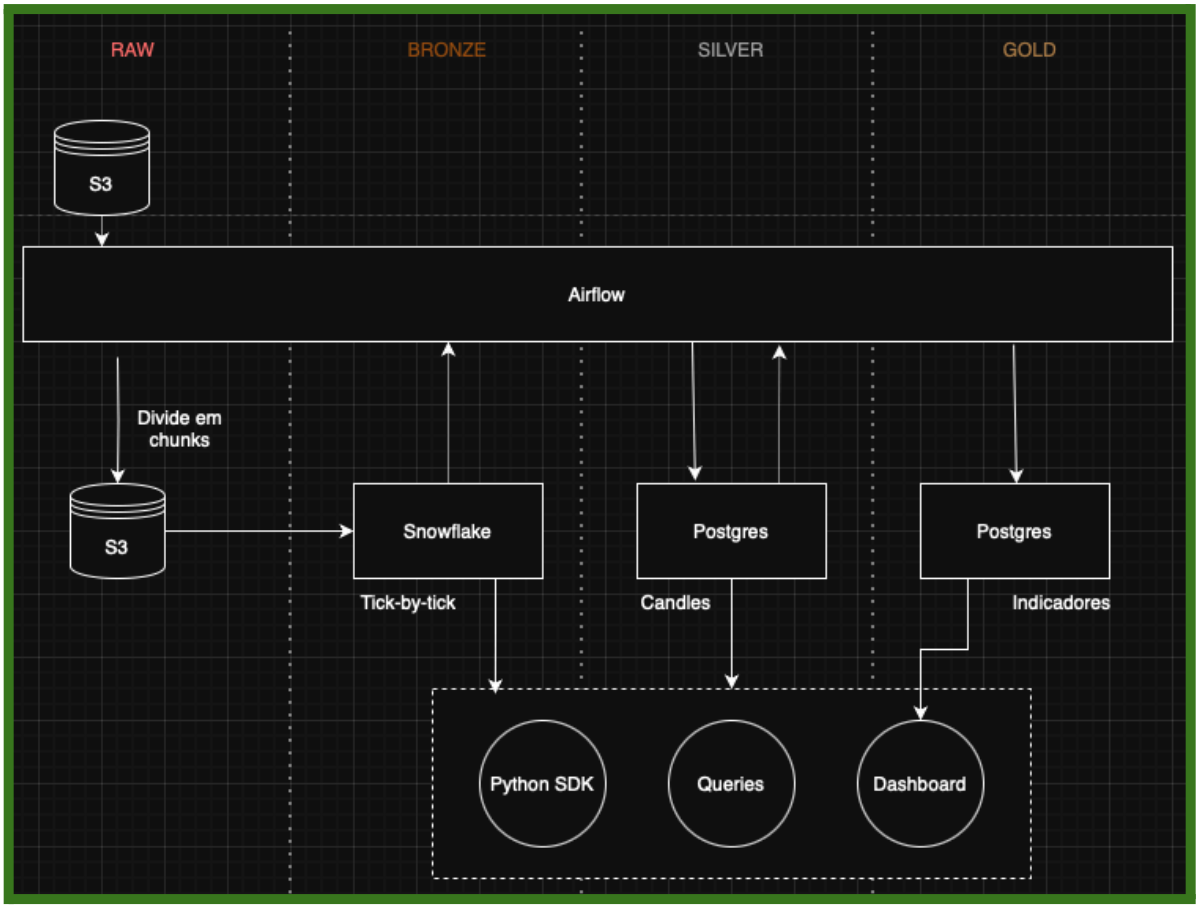
Ao transformar o ER em Star Schema algumas possíveis relações foram omitidas, como por exemplo a relação entre Funcionário e Loja: Todo funcionário é de uma loja, e todo Gerente de loja é um Funcionario, portanto essas tabelas poderiam conversar, mas isso foi omitido para simplificar o Star Schema. Decidi também criar a Dimension Data para guardar a data de cada venda, o que é uma boa prática em Star Schemas, pois facilita a busca em períodos de tempo especificados.

4) Cenário

Todo dia às 03h um arquivo .txt com 50gb é disponibilizado em um bucket S3 contendo informação de todas as operações que aconteceram no dia anterior. Esses dados precisam ser tratados e disponibilizados em um data warehouse para que os indicadores estejam disponíveis as 05h para os clientes consumirem.

Tarefa
Monte a arquitetura de um pipeline que faça a ingestão, tratamento e disponibilização desses dados de forma estruturada para consumo via queries e via dashboard.

Entregáveis
• Desenho da arquitetura contendo as ferramentas utilizadas
• Breve descrição do motivo da escolha das ferramentas



Apache Airflow - Orquestrador de tarefas open-source, intuitivo e possui integração com um grande leque de ferramentas, incluindo AWS S3, Snowflake e Postgres.

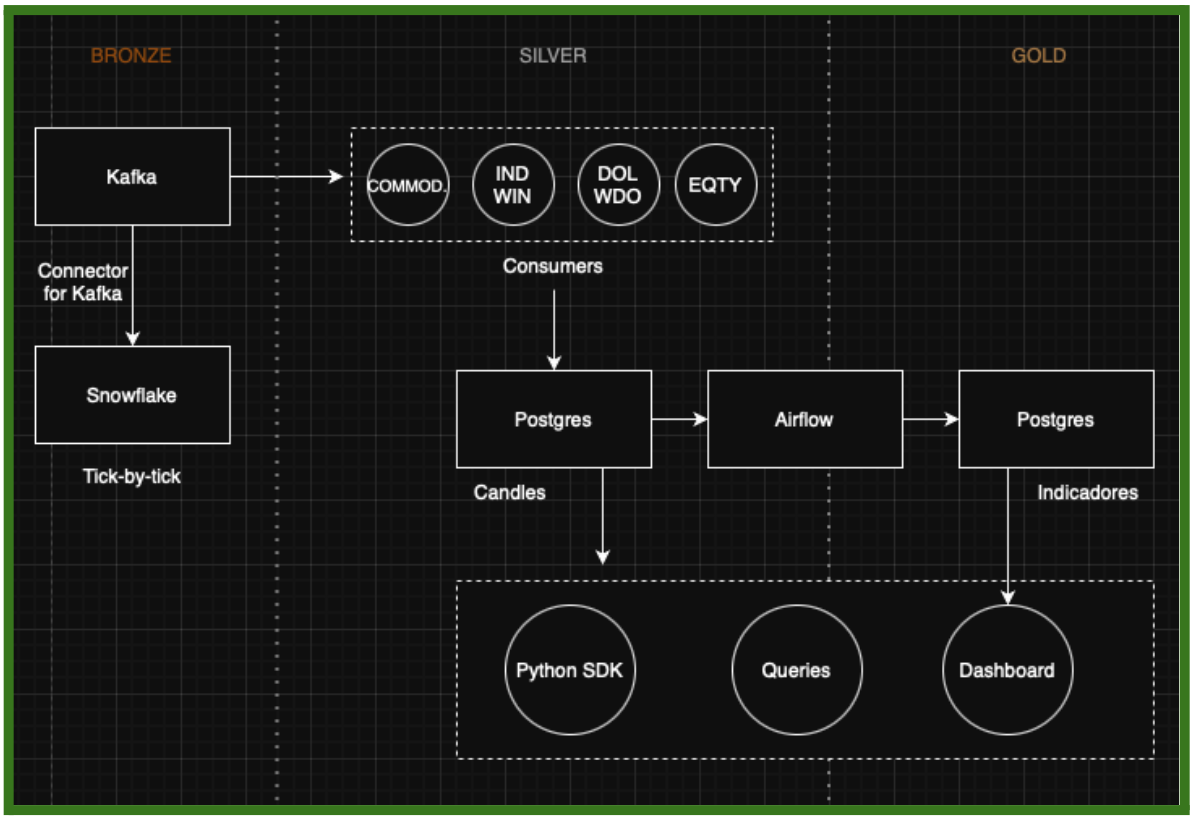
Snowflake - Interface intuitiva e integrável com diversas ferramentas e fácil de escalar.

Python SDK - Forma fácil de acesso aos dados por parte dos usuários, podendo escolher por exemplo a granularidade.

5) Cenário
Os valores de ativos listados na B3 são disponibilizados em tempo real em um tópico Kafka para consulta. Esses dados precisam estar disponíveis para a área de investimento conseguir tomar as decisões de compra ou venda de ativos.

Tarefa
Monte a arquitetura de um pipeline que faça a ingestão, tratamento e disponibilização desses dados de forma estruturada para consumo via queries e via dashboard.

Entregáveis
• Desenho da arquitetura contendo as ferramentas utilizadas
• Breve descrição do motivo da escolha das ferramentas



Python e consumers separados - Separar a escuta dos tópicos por conjuntos de ativo de grande liquidez vai diminuir o gargalo e paralelizar a leitura e inserção desses dados no banco. Em caso de falha de apenas um consumer, ele retoma a leitura de onde parou e não para o processo dos outros consumers.

Database Administration

Exercícios:

1) Explique, sucintamente, a diferença entre *BEGIN/END* dentro de um objeto de linguagem procedural PL/pgSQL e o *BEGIN* que se executa no cliente *psql* para execução de uma simples DML, como um update.

Em PL/pgSQL o BEGIN e END são usados para agrupamento dentro de um bloco que identifica uma função. A palavra reservada BEGIN em no psql é um comando que inicia uma transação, isto é, persistir em disco o que vier após o BEGIN até que outro comando como COMMIT ou ROLLBACK ordene a escrita ou cancelamento dos comandos.

2) Assinale a alternativa que corretamente concatena as palavras 'ABC' e 'def' para formar 'ABCdef'.

```
A. SELECT 'ABC' . 'def';
B. SELECT cat('ABC', 'def') FROM pg_operator;
C. SELECT 'ABC' + 'def' FROM dual;
D. SELECT 'ABC' + 'def' FROM dual;
E. SELECT 'ABC' || 'def';
```

3) O PostgresQL pode usar índices para acessar uma tabela, assinale duas alternativas erradas sobre índices:

A. Um índice é criado pelo 'CREATE INDEX' e eliminado pelo 'DROP INDEX'. B. Quando a query usa o índice, ela pode retornar as linhas de forma muito mais rápida.

C. Os tipos de índices são B-TREE, Hash, R-TREE e GiST.

D. Quando se cria um índice, a query que usa aquela coluna indexada fica sempre mais rápida.

E. Criar um índice que não esteja sendo utilizado por nenhuma query não altera de forma alguma o desempenho do banco de dados.

4) Assinale duas afirmações corretas sobre VIEWS no PostgreSQL:

A. Uma VIEW é criada pelo comando 'DECLARE VIEW' e eliminada pelo comando 'DROP VIEW'

B. Uma VIEW é uma tabela virtual que não existe no disco.

C. Uma VIEW ajuda a simplificar queries complicadas.

D. Uma VIEW pode ser criada com o mesmo nome de uma tabela no esquema em questão.

E. Uma VIEW só existe enquanto o processo postmaster está rodando, sendo eliminada quando o servidor para.

5) Baseado na tabela *EMPREGADOS* abaixo, escreva uma query (suando sub query) que retorna o *ID*, o *FIRST_NAME*, o *MANGER_ID* e o *SALARY* de todos os empregados que tem salário maior que o maior salário dos empregados com *MANAGER_ID* igual a 100. Ordene o resultado pelo salário.

```
+----+-------------+------------+------------+----------+----------------+------------+-------------
--+ | ID | FIRST_NAME | LAST_NAME | HIRE_DATE | SALARY | COMMISSION_PCT | MANAGER_ID | DEPARTMENT_ID
|
+----+-------------+------------+------------+----------+----------------+------------+-------------
--+ |100 | Marcelo | Goncalves | 1997-04-17 | 20000.00 | 0.00 | 101 | 80 | |101 | Carlos |
Macaranduba | 1997-05-18 | 17000.00 | 0.00 | 100 | 90 | |102 | Renato | Ucraniano | 1997-06-19 |
16000.00 | 0.00 | 100 | 90 |
.....................................................................................................
... |300 | Geraldo | Silva | 2001-10-01 | 8300.00 | 0.00 | 205 | 110 |
+----+-------------+------------+------------+----------+----------------+------------+-------------
                                                --+
```

```
SELECT ID, FIRST_NAME, MANAGER_ID, SALARY FROM EMPREGADOS
    WHERE SALARY >  (SELECT MAX(SALARY) FROM EMPREGADOS WHERE
    MANAGER_ID = 100) ORDER BY SALARY;
```

6) Baseado na pergunta anterior, responda V para verdadeiro e F para falso na afirmações abaixo:

A. (**V**) Um índice compostos nas colunas (manager_id, salary) é recomendado.

B. (**F**) A query requisitada será sempre lenta, não importando como está indexada a tabela.

C. (**F**) O PostgreSQL sempre fará a ordenação em disco, independente do índice usado.

D. (**V**) Um índice de função (índice com expressão) deixaria a query mais rápida.

E. (**V**) O PostgreSQL permite que se use tabelas temporárias para evitar o uso de subqueries.

7) Assinale a alternativa incorreta sobre PostgreSQL:

A. O PostgreSQL possui recurso para permitir execução de queries em paralelo.
B. O PostgreSQL possui recurso para permitir criação de índices de forma *ONLINE*, sem bloquear escrita.
C. Na criação de um banco de dados é obrigatório definir o proprietário com a palavra chave *OWNER* do banco.
D. Define-se o esquema que deseja trabalhar com *search_path* para evitar usar o nome do esquema nas tabelas o tempo todo.
E. Os valores do *search_path* podem conter esquemas separados por vírgula