

## PROGRAMAÇÃO DE BANCO DE DADOS

### Criação de programas para banco de dados, procedimentos, funções, gatilhos

1.

Os comandos SQL categorizados como DCL são utilizados para controlar a segurança dos dados, ou seja, definem quem pode acessar cada operação dos objetos do banco de dados. Dentre os comandos abaixo, qual é um comando DCL?

A.

CREATE.

---

B.

ALTER TABLE.

---

C.

DROP TABLE.

---

D.

GRANT.

---

E.

RENAME TABLE.

2.

O comando SELECT do SQL permite a realização de consultas nas tabelas do banco de dados. Como utiliza a linguagem de programação SQL, esse comando tem sintaxe definida. Para realizar

uma consulta em uma tabela chamada *Imoveis*, procurando o imóvel cujo código é 1090, qual destas sintaxes do comando SELECT está correta?

**A.**

**SELECT Codigo, Endereco , Proprietario FROM Imoveis WHERE Codigo = 1090.**

---

**B.**

**CREATE Codigo, Endereco , Proprietario FROM Imoveis WHERE Codigo = 1090.**

---

**C.**

**SELECT Codigo FROM Imoveis WHERE Endereco= 1090.**

---

**D.**

**SELECT Codigo, Endereco , Proprietario FROM Codigos WHERE Codigo = 1090.**

---

**E.**

**SELECT Codigo, Endereco , Proprietario FROM Imoveis WHERE Proprietario= 1090.**

**3.**

Durante o desenvolvimento de aplicações complexas que acessam banco de dados, a execução de rotinas robustas para manipulação dos dados pode otimizar o desempenho do banco de dados. Para isso, é necessário executar várias instruções SQL de modo sequencial para solucionar os problemas oriundos da programação do banco de dados. Diante desse contexto, os procedimentos armazenados são um recurso importante na programação de banco

de dados.

Marque a opção que é uma RESTRIÇÃO para utilização de procedimentos armazenados em banco de dados:

A.

Falta de comandos SQL.

---

B.

Restrição de *hardware* na máquina onde vai rodar o banco de dados.

---

C.

Restrição da linguagem de programação utilizada na aplicação.

---

D.

Falta de padronização da sintaxe.

---

E.

Impossibilidade de manipulação de dados.

4.

Qual o resultado do comando SQL 'CREATE DATABASE Locadora\_NovoTempo' após a sua execução?

A.

Será criada uma tabela no banco de dados Locadora\_NovoTempo.

---

B.

Será criada uma nova coluna no banco de dados Locadora\_NovoTempo.

---

**C.**

Será criado um banco de dados com o nome Locadora\_NovoTempo.

---

**D.**

Será criado um banco de dados novo sem tabelas e sem nome.

---

**E.**

Será criado um novo procedimento armazenado no banco de dados Locadora\_NovoTempo.

**5.**

Qual o resultado do comando SQL 'DROP TABLE Carros' após a sua execução?

**A.**

Cria uma tabela no banco de dados chamada Carros.

---

**B.**

Cria um novo registro na tabela Carros.

---

**C.**

Exclui o banco de dados chamado Carros.

---

**D.**

Exclui a tabela Carros do banco de dados.

---

E.

Exclui todos os registros com a palavra Carros do banco de dados.

### Conhecer os desafios da Administração de Banco de Dados

1.

Sabendo que um banco de dados tem perda de *performance* nas VIEWS, qual técnica pode ser utilizada para obter melhora?

A.

Checar a tabela.

---

B.

Criar índices das views.

---

C.

Excluir dados não utilizados.

---

D.

Criar uma outra tabela para relacionar buscas.

---

E.

Criar triggers.

2.

É uma tecnologia atual que necessita de qualificação do administrador de banco de dados:

A.

Tuning de banco de dados.

B.

Dividir o banco de dados principal em pequenos bancos de dados.

C.

Utilizar BigData.

D.

Utilizar bancos NoSQL.

E.

Replicação de bancos de dados. 3.

Qual a tecnologia atual que está consumindo muito a pesquisa do banco de dados?

A.

IoT.

---

B.

Cloud.

---

C.

Backup.

---

D.

Restore.

---

E.

Paralelismo.

4.

Entre os bancos de dados existentes, qual é o que se destaca pelo maior uso e abrangência a novas tecnologias?

A.

Oracle.

---

B.

MySQL.

---

C.

MSSQL.

---

D.

MongoDB.

---

E.

Amazon Dynamo DB.

5.

Qual comando pode ser utilizado para monitorar o banco de dados MySQL?

**A.**

**mytop.**

---

**B.**

**mysqltop.**

---

**C.**

**mysqlcheck.**

---

**D.**

**mysqldump.**

---

**E.**

**mysqltree.**

### **Necessidades das políticas de manutenção de banco de dados**

**1.**

**São exemplos de rotinas de manutenção em banco de dados:**

**A.**

**Rotina de limpeza e rotina de rede.**

---

**B.**

**Rotina de reindexação e rotina de rede.**

---

**C.**

**Rotina de manutenção de arquivos e rotina de backup.**

---

**D.**

**Rotina de backup e rotina de criação de novos bancos de dados.**

---

**E.**

**Rotina de limpeza e rotina de ocupação de espaço em disco.**

**2.**

**Em um banco de dados sem manutenção corretiva e preventiva, o banco de dados da empresa passou a apenas aceitar ações de leitura dos dados. O que poderia ocasionar isso?**

**A.**

**Espaço do banco sem capacidade de expansão.**

**B.**

**Uma manutenção em alguma tabela.**

**C.**

**Uma atualização de sistema.**

**D.**

**Alguma alteração feita pelo usuário.**

E.

Uma manutenção feita pelo programador.

3.

Em um banco de dados MySQL, quais são os comandos que fazem manutenção em tabelas do banco de dados?

A.

Check table, repair table, analyze table.

---

B.

Check table, repair table, drop table e analyze table.

---

C.

Check table, repair table, optimize table e analyze table.

---

---

D.

Check table, repair table, create table.

---

E.

Repair table e analyze table.

4.

Uma forma de fazer manutenção no banco de dados, sem afetar os usuários, seria?

A.

Acessar o banco de dados e efetuar um check no banco.

---

**B.**

**Copiar o banco de dados em produção e efetuar sua manutenção.**

---

**C.**

**Identificar o erro e fazer a manutenção em uma base de testes, gerada por meio da restauração de backup.**

---

**D.**

**Identificar o erro e efetuar a manutenção no banco de dados em produção.**

---

**E.**

**Informar o programador para bloquear a aplicação.**

**5.**

**Atualmente, temos uma praga virtual chamada ransomware que efetua o sequestro dos dados. Seria uma forma plausível de fazer a manutenção preventiva do banco evitando a destruição causada por esta praga?**

**A.**

**Ter o backup do banco de dados em outra partição do disco.**

---

**B.**

**Efetuar o backup do banco de dados em um HD USB conectado ao host.**

---

**C.**

Efetuar o backup do banco de dados dentro do próprio disco.

---

D.

Efetuar o backup na nuvem com um aplicativo que fica conectado full time.

---

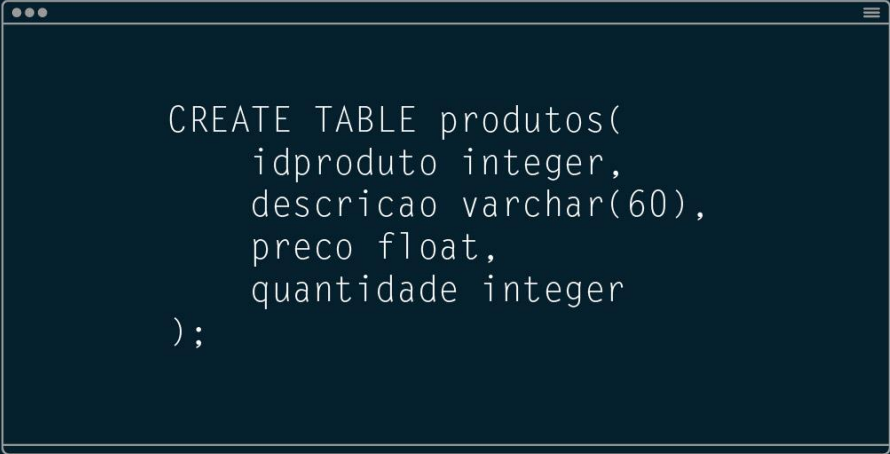
E.

Efetuar um backup na nuvem periodicamente, porém, não manter o acesso disponível.

### Stored procedures e functions

1.

Considere a seguinte tabela:



```
CREATE TABLE produtos(  
    idproduto integer,  
    descricao varchar(60),  
    preco float,  
    quantidade integer  
);
```

O administrador do banco de dados que trabalha em um supermercado criou um *stored procedure* para facilitar a inserção dos registros dos produtos no banco de dados.

Qual é o código correto para a criação do *stored procedure* nesse caso? Assinale a alternativa correta:

**A.**

```
CREATE FUNCTION insere_produtos(idprodutos integer, descricao
varchar(60), preco float, quantidade integer)
AS $$
INSERT INTO produtos VALUES(idprodutos, descricao, preco,
quantidade);
$$ LANGUAGE SQL;
```

---

**B.**

```
CREATE PROCEDURE insere_produtos(idprodutos integer, descricao
varchar(60), preco float, quantidade integer)
AS $$
INSERT INTO produtos VALUES(idprodutos, descricao, preco,
quantidade);
RETURN produtos
$$ LANGUAGE SQL;
```

---

**C.**

```
CREATE PROCEDURE insere_produtos(idprodutos integer, descricao
varchar(60), preco float, quantidade integer)
AS $$
SET INTO produtos VALUES(idprodutos, descricao, preco,
quantidade);
$$ LANGUAGE SQL;
```

**D.**

```
CREATE PROCEDURE insere_produtos(idprodutos integer, descricao
varchar(60), preco float, quantidade integer)
AS $$
INSERT INTO produtos VALUES(idprodutos, descricao, preco,
quantidade);
$$ LANGUAGE SQL;
```

E.

```
CREATE PROCEDURE insere_produtos(produtos)
AS $$
INSERT INTO produtos VALUES(idprodutos, descricao, preco,
quantidade);
$$ LANGUAGE SQL;
```

2.

**Os *stored procedures* são blocos de código armazenados no servidor de banco de dados e podem ser chamados a qualquer momento, com o objetivo de realizar algum processamento em informações. Eles oferecem alguns benefícios ao banco de dados e às aplicações que o utilizam. Quais são esses benefícios? Assinale a alternativa correta:**

A.

**Descentralização, velocidade, performance, segurança e suporte a transferências.**

---

B.

**Descentralização, performance, segurança e suporte a transferências.**

---

**C.**

**Centralização, velocidade, performance, segurança e suporte a transações.**

---

**D.**

**Velocidade, performance, segurança e suporte a transferências.**

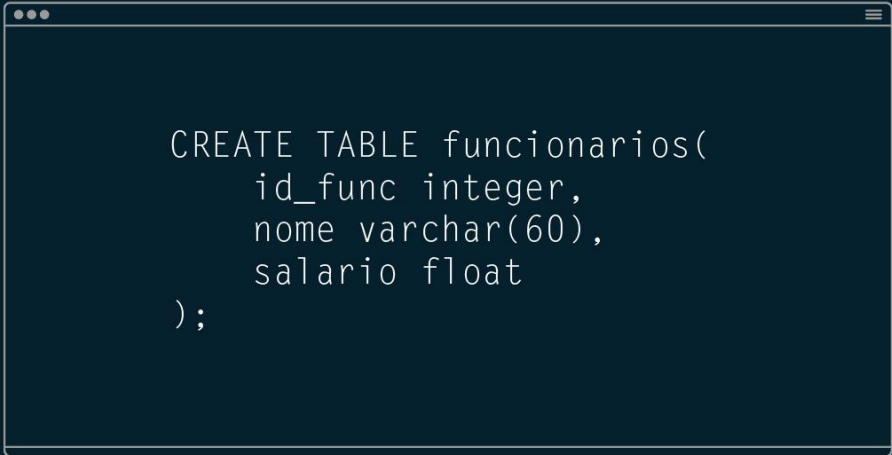
---

**E.**

**Centralização, velocidade, performance e suporte a transferências.**

**3.**

**Considere a seguinte tabela:**



```
CREATE TABLE funcionarios(  
    id_func integer,  
    nome varchar(60),  
    salario float  
);
```

**Suponha que você seja o administrador do banco de dados de uma empresa, e o gerente pediu que você entregue um relatório com o número total de funcionários da empresa.**

**Qual dos trechos de código a seguir apresenta a função correta?**

**A.**

```
CREATE FUNCTION totalFuncionarios () RETURNS integer AS $$  
DECLARE  
    total integer;  
BEGIN  
    SELECT count(*) into total FROM funcionarios;  
    RETURN total;  
END;  
$$ LANGUAGE plpgsql;
```

---

**B.**

```
CREATE FUNCTION totalFuncionarios () RETURNS void AS $$  
DECLARE  
    total integer;  
BEGIN  
    SELECT count(*) into total FROM funcionarios;  
    RETURN total;  
END;  
$$ LANGUAGE plpgsql;
```

---

C.

```
CREATE FUNCTION totalFuncionarios () RETURNS integer AS $$  
DECLARE  
    total integer;  
BEGIN  
    SELECT count(*) into total FROM funcionarios;  
END;  
    RETURN total;  
$$ LANGUAGE plpgsql;
```

---

D.

```
CREATE FUNCTION totalFuncionarios () RETURNS integer AS $$  
DECLARE  
BEGIN  
    total integer;  
    SELECT count(*) into total FROM funcionarios;  
    RETURN total;  
END;  
$$ LANGUAGE plpgsql;
```

---

E.

```
CREATE FUNCTION totalFuncionarios () RETURNS integer AS $$  
DECLARE  
    total integer;  
BEGIN  
    SELECT count(*) into total FROM funcionarios;  
END;  
$$ LANGUAGE plpgsql;
```

4.

**Considere a seguinte função escrita em SQL. Esta função recebe como parâmetro dois valores inteiros e retorna a subtração deles. Note que, em SQL, a palavra reservada SELECT vai funcionar como o RETURN na linguagem PL/pgSQL.**

```
CREATE FUNCTION subtrai(a integer, b integer)  
RETURNS integer AS $$  
    SELECT a - b;  
$$ LANGUAGE SQL;
```

**Quais das funções a seguir executam a mesma operação dessa função, mas escrita na linguagem PL/pgSQL?**

**A.**

```
CREATE FUNCTION subtrai() RETURNS integer AS $$  
DECLARE  
    subtracao integer;  
BEGIN  
    subtracao := a - b;  
    RETURN subtracao;  
END;  
$$ LANGUAGE plpgsql;
```

---

**B.**

```
CREATE FUNCTION subtrai(a integer, b integer) AS $$  
DECLARE  
    subtracao integer;  
BEGIN  
    subtracao := a - b;  
    RETURN subtracao;  
$$ LANGUAGE plpgsql;
```

---

**C.**

```
CREATE FUNCTION subtrai(a integer, b integer)
RETURNS integer AS $$
BEGIN
    subtracao := a - b;
    RETURN subtracao;
END;
$$ LANGUAGE plpgsql;
```

---

**D.**

```
CREATE FUNCTION subtrai(a integer, b integer )
RETURNS integer AS $$
DECLARE
    subtracao integer;
BEGIN
    subtracao := a - b;
    RETURN subtracao;
$$ LANGUAGE plpgsql;
```

---

**E.**

```
CREATE FUNCTION subtrai(a integer, b integer)
RETURNS integer AS $$
DECLARE
    subtracao integer;
BEGIN
    subtracao := a - b;
    RETURN subtracao;
END;
$$ LANGUAGE plpgsql;
```

5.

Considere a seguinte tabela:

```
CREATE TABLE alunos(
    matricula integer,
    nome varchar(60),
    mediaFinal float
);
```

Foi criada uma *function* para retornar a maior média obtida entre os alunos, conforme o seguinte trecho de código:

```
CREATE FUNCTION maiorMedia() RETURNS float
AS $$
DECLARE
    media alunos.mediaFinal%type;
BEGIN
    SELECT MAX(mediaFinal) into media from
public.alunos;
    RETURN media;
END;
$$ LANGUAGE plpgsql ;
```

Qual comando a seguir apresenta a forma correta de invocar essa *function*?

A.

CALL maiorMedia();

---

B.

SELECT INTO maiorMedia();

---

C.

SELECT maiorMedia(20200420, 'Marcela da Silva', 8.7);

---

D.

SELECT maiorMedia();

---

---

E. SELECT maiorMedia(alunos);

## **Linguagem SQL (noções básicas)**

**1.**

Sabemos que a SQL (Linguagem de Consulta Estruturada) está dividida em categorias, sendo que uma dessas categorias é a DCL (Data Control Language ou Linguagem de Controle de Dados). A DCL é uma subcategoria de instruções utilizada para o controle dos privilégios de usuários, de forma, que o administrador do banco de dados possa determinar o nível de acesso de um usuário ou objeto do banco de dados. Aponte qual das opções a seguir contém somente instruções (comandos) de DCL.

**A.**

- Create
  - Grant
- 

**B.**

- Revoke
  - Commit
- 

**C.**

- Drop
  - Revok
- 

**D.**

- Delete
  - Grant
-

**E.**

**- Grant**

**- Revoke**

**2.**

A instrução Select faz parte da categoria DML (Linguagem de Manipulação de Dados), sendo uma instrução básica para recuperar informações de uma ou mais tabelas de um banco de dados.

Existem muitas opções e tipos de instruções Select em SQL, de modo que podemos utilizar estas opções de forma gradual e conforme o cenário e a necessidade. Aponte qual das opções a seguir contém somente palavras-chave válidas para utilizar junto à instrução Select.

**A.**

**- FROM**

**- SHOW**

**- ORDER BY**

---

**B.**

**- CREATE**

**- WHERE**

**- ORDER BY**

---

**C.**

**-FROM**

**-WHERE**

**-DESCRIBE**

---

**D.**

**-FROM**

**-WHERE**

**-ORDER BY**

---

**E.**

**-FROM**

**-MODIFY**

**-ORDER BY**

**3.**

A TCL (Transact Control Language ou Linguagem de Controle de Transações) é uma categoria de instruções utilizada para o controle de transações considerando um conjunto de uma ou mais operações de manipulação de dados (DML) realizadas em um banco de dados. Aponte qual das opções a seguir contém somente instruções (comandos) de TCL.

**A.**

**- Update**

**- Commit**

---

**B.**

**- Drop**

**- Rollback**

---

**C.**

**- Commit**

**- Rollback**

---

D.

- Alter
  - Commit
- 

E.

- Rollback
- Delete

4.

A partir da figura a seguir, que traz a estrutura da Tabela Pessoa\_Física (simplificada), onde estão indicadas as colunas, tipo de dados e tamanho do campo, podemos imaginar qual a sintaxe de código SQL equivalente e que dará forma a tal Tabela. Aponte qual das opções a seguir contém o código/sintaxe correto para tal situação.

Coluna	Tipo de dado	Tamanho
num_pessoa_pf	numérico inteiro	5
nom_pessoa	alfanumérico	30
num_CPF	numérico inteiro	11
dat_nascimento	data	padrão
idt_sexo	caractere	1

A.

(num\_pessoa\_pf NUMBER (5),  
nom\_pessoa VARCHAR (30),  
num\_CPF NUMBER (11),  
dat\_nascimento DATE,  
idt\_sexo CHAR (1));

---

B.

```
CREATE TABLE Pessoa_Fisica  
(num_pessoa_pf NUMBER (5),  
nom_pessoa VARCHAR (30),  
num_CPF NUMBER (11),  
dat_nascimento DATE,  
idt_sexo CHAR (1));
```

---

C.

```
CREATE TABLE Pessoa_Fisica  
(nom_pessoa VARCHAR (30),  
num_CPF NUMBER (11),  
dat_nascimento DATE,  
idt_sexo CHAR (1));
```

---

D.

```
CREATE TABLE Pessoa_Fisica  
(num_pessoa_pf NUMBER (5),  
nom_pessoa VARCHAR (30),  
dat_nascimento DATE,  
idt_sexo CHAR (1));
```

---

E.

```
CREATE TABLE Pessoa_Fisica  
(num_pessoa_pf NUMBER (5),
```

**nom\_pessoa VARCHAR (30),**

**num\_CPF NUMBER (11),**

**idt\_sexo CHAR (1));**

**5.**

Em determinadas situações, há necessidade de adicionarmos um novo campo a uma Tabela já existente. Na figura abaixo (Tabela Pessoa\_Fisica), precisamos de mais um campo, onde será armazenada a UF (Unidade da Federação, ou Estado) onde a pessoa nasceu. Aponte qual das opções a seguir contém o código/sintaxe correto para tal situação.

Coluna	Tipo de dado	Tamanho
num_pessoa_pf	numérico inteiro	5
nom_pessoa	alfanumérico	30
num_CPF	numérico inteiro	11
dat_nascimento	data	padrão
idt_sexo	caractere	1

**A.**

**ALTER TABLE Pessoa\_Fisica**

**ADD (UF\_pessoa CHAR (2));**

**B.**

**ALTER TABLE Pessoa\_Fisica**

**MODIFY (UF\_pessoa CHAR (2));**

**C.**

**ALTER TABLE Pessoa\_Fisica**

**DROP (UF\_pessoa CHAR (2));**

D.

```
ALTER TABLE Pessoa_Fisica  
DELET (UF_pessoa CHAR (2));
```

---

E.

```
ALTER TABLE Pessoa_Fisica  
CREATE (UF_pessoa CHAR (2));
```

### Linguagem SQL (Consultas)

1.

Quando da utilização da instrução SELECT para consultas em banco de dados, podemos utilizar vários argumentos de sintaxe para compor e estruturar a consulta das informações as quais são necessárias. Um destes argumentos é o BETWEEN-AND, um operador de atalho para testar uma coluna de data ou numérica em relação a uma gama de valores. Aponte qual das opções a seguir contém a sintaxe SQL correta para a "saída" conforme a figura abaixo, onde o objetivo é listar o Nome e a Data de contratação dos professores, contratados entre 1999 e 2000.

NomeProf	SobrenomeProf	DataAdmProf
João	Silva	01-mar-1999
Maria	Santos	15-out-1999
Julia	Bastos	10-set-2000

A.

```
SELECT NomeProf, SobrenomeProf, DataAdmProf  
FROM Professor  
WHERE DataAdmProf BETWEEN #1/1/1999# AND #31/12/2000#
```

---

**B.**

```
SELECT NomeProf, SobrenomeProf, DataAdmProf
FROM Professor
WHERE DataAdmProf BETWEEN #1/1/1999# AND #31/12/2005#
```

---

**C.**

```
SELECT NomeProf, SobrenomeProf, DataAdmProf
FROM Professor
WHERE DataAdmProf BETWEEN #1/1/1997# AND #31/12/2000#
```

---

**D.**

```
SELECT NomeProf, SobrenomeProf, DataAdmProf
WHERE DataAdmProf BETWEEN #1/1/1999# AND #31/12/2000#
```

---

**E.**

```
SELECT NomeProf, SobrenomeProf, DataAdmProf
FROM Professor
WHERE DataAdmProf #1/1/1999# AND #31/12/2000#
```

**2.**

O argumento GROUP BY pode ser utilizado junto à instrução SELECT para consultas em banco de dados quando se deseja agrupar informações a partir de determinadas colunas. Aponte qual das opções a seguir contém a sintaxe SQL correta para a "saída" conforme a figura abaixo, onde o objetivo é consultar a média geral de notas dos estudantes, agrupados por especialização.

Especialização	MediaGeral
Contabilidade	3,339
Finanças	2,888
Marketing	3,658
Sistemas	3,223

A.

```
SELECT Especialização, AVG(MediaAluno) AS MediaGeral  
GROUP BY Especialização
```

---

B.

```
SELECT Especialização, AVG(MediaAluno) AS MediaGeral  
FROM Aluno  
GROUP BY Especialização
```

---

C.

```
SELECT Especialização, AVG(MediaAluno) AS MediaGeral  
FROM Aluno  
GROUP BY MediaGeral
```

---

D.

```
SELECT Especialização, AVG(MediaAluno) AS MediaGeral  
FROM Aluno  
GROUP BY Aluno
```

---

E.

```
SELECT Especialização, AS MediaGeral  
FROM Aluno  
GROUP BY Especialização
```

3.

O argumento **GROUP BY** pode ser utilizado junto a instrução **SELECT** para consultas em banco de dados; junto a este argumento, podemos utilizar também a instrução **HAVING**, que serve para aplicar uma determinada condição de seleção junto aos registros

agrupados. Aponte qual das opções a seguir contém a sintaxe SQL correta para a "saída" conforme a figura abaixo, onde o objetivo é consultar a quantidade e o valor das faturas de cada conveniada, porém, somente quando o valor da fatura for superior a 30.000.

COD CONVENIADA	COUNT (NUM FATURA)	SUM (VLR FATURA)
55122	4	81.344,12
55123	4	39.546,95
55124	6	65.489,78
55125	6	98.856,32

A.

```
SELECT cod_conveniada, count (num_fatura), sum(vlr_fatura)
FROM Fatura
GROUP BY cod_conveniada
HAVING sum (vlr_fatura) > 50000
```

---

B.

```
SELECT cod_conveniada, count (num_fatura), sum(vlr_fatura)
GROUP BY cod_conveniada
HAVING sum (vlr_fatura) > 30000
```

---

C.

```
SELECT cod_conveniada, count (num_fatura), sum(vlr_fatura)
FROM Fatura
GROUP BY cod_conveniada
HAVING sum (vlr_fatura) > 30000
```

---

D.

```
SELECT cod_conveniada, count (num_fatura), sum(vlr_fatura)
FROM Fatura
```

```
GROUP BY num_fatura
HAVING sum (vlr_fatura) > 30000
```

---

E.

```
SELECT cod_conveniada, count (num_fatura), sum(vlr_fatura)
FROM Fatura
GROUP BY cod_conveniada
```

4.

Quando da utilização da instrução SELECT para consultas em banco de dados, podemos utilizar vários argumentos de sintaxe para compor e estruturar a consulta das informações as quais são necessárias. Um destes argumentos é o NULL (IS NULL ou IS NOT NULL) que é utilizado para consultar valores nulos ou não nulos. Aponte qual das opções a seguir contém a sintaxe SQL correta para a "saída" conforme a figura abaixo, onde o objetivo é consultar o código (num\_pessoa\_pf) das pessoas cujo estado civil NÃO foi cadastrado (nulo).

NUM_PESSOA_P F
32451
23654
29784
12568

A.

```
SELECT num_pessoa_pf
FROM pessoa_juridica
WHERE cod_estado_civil IS NULL
```

---

B.

```
SELECT num_pessoa_pf  
FROM pessoa_fisica  
WHERE num_pessoa_pf IS NULL
```

---

C.

```
SELECT num_pessoa_pf  
FROM pessoa_fisica  
WHERE cod_estado_civil IS NOT NULL
```

---

D.

```
SELECT num_pessoa_pf  
FROM pessoa_fisica  
WHERE cod_estado_civil IS NULL
```

---

E.

```
SELECT num_pessoa_pf  
WHERE cod_estado_civil IS NULL
```

5.

Quando utilizamos a instrução SELECT para consultas em banco de dados, podemos utilizar também os argumentos de sintaxe MIN (mínimo), MAX (máximo) e AVG (média). Aponte qual das opções a seguir contém a sintaxe SQL correta para a "saída" conforme a figura abaixo, onde o objetivo é consultar a fatura com maior valor, a fatura com menor valor e o valor médio entre todas as faturas.

MAIOR	MENOR	MEDIA
32.155,20	15.566,22	24.555,66

A.

```
SELECT MAX (vlr_fatura) AS MENOR,  
MIN (vlr_fatura) AS MAIOR,  
AVG (vlr_fatura) AS MEDIA  
FROM fatura;
```

---

B.

```
SELECT MAX (vlr_fatura) AS MAIOR,  
AVG (vlr_fatura) AS MEDIA  
FROM fatura;
```

---

C.

```
SELECT MAX (vlr_fatura) AS MAIOR,  
MIN (vlr_fatura) AS MENOR,  
FROM fatura;
```

---

D.

```
SELECT MAX (vlr_fatura) AS MAIOR,  
MIN (vlr_fatura) AS MENOR,  
AVG (vlr_fatura) AS MEDIA;
```

---

E.

```
SELECT MAX (vlr_fatura) AS MAIOR,  
MIN (vlr_fatura) AS MENOR,  
AVG (vlr_fatura) AS MEDIA  
FROM fatura;
```

## Funções de agregação e agrupamento de dados em SQL

1.

A linguagem SQL disponibiliza, dentre os seus recursos de consulta, várias instruções utilizadas para agregação. As funções de agregação têm como objetivo prever o resultado a partir da execução de uma operação.

Marque a alternativa que corresponde a uma função de agregação:

A.

ORDER BY.

---

B.

GROUP BY.

---

C.

COUNT.

---

---

D.

WHERE.

---

E.

WHEN.

2.

A Escola Técnica para menores implantou um banco de dados para armazenar os dados de seus alunos. Atualmente, as informações dos alunos são:

Tabela de alunos			
Matrícula	Nome	Idade	Cidade
10	José	25	Caxias
20	Maria	30	Niterói
30	João	19	Null
40	Joana	32	Rio de Janeiro

Marque a alternativa que apresenta a quantidade de alunos cadastrados na Escola Técnica:

A.

`select count(*) from Alunos;.`

---

B.

`select count(cidade) from Alunos;.`

---

C.

`select sum(*) from Alunos;.`

---

D.

`select sum(cidade) from Alunos;.`

---

E.

`select sum(idade) from Alunos;.`

3.

O sr. Manoel possui uma padaria, que já funciona há muitos anos na vila onde mora. Preocupado com os tempos difíceis que está vivendo, ele calcula diariamente o total de vendas realizadas por atendente. Com isso, pretende avaliar quantos atendentes são, de fato, necessários para o desempenho dos serviços da padaria.

Você, analista de sistema, sugere ao sr. Manoel a implantação de um sistema com uso de um banco de dados, para facilitar a sua rotina, argumentando que, ao final do dia, poderá executar uma consulta e obter as informações desejadas automaticamente. Após o levantamento dos requisitos, junto ao sr. Manoel, você propõe a seguinte estrutura para os dados:

Tabela: ATENDIMENTOS			
cpfAtendente	dataVenda	codProduto	valorVenda
11111111111	10/02/2020	10	15,00
22222222222	15/02/2020	20	25,00
33333333333	20/03/2020	10	30,00
11111111111	25/03/2020	30	55,00
22222222222	25/03/2020	10	45,00

Analise as sentenças a seguir e marque a alternativa com a instrução SQL que resultará na apresentação do valor total de vendas realizadas por atendente, como deseja o sr. Manoel:

A.

```
select cpfAtendente, count(valorVenda) from atendimentos group by cpfAtendente;
```

---

**B.**

```
select cpfAtendente, sum(valorVenda) from atendimentos group by  
cpfAtendente;.
```

---

**C.**

```
select cpfAtendente, sum(valorVenda) group by cpfAtendente from  
atendimentos;.
```

---

**D.**

```
select cpfAtendente,, count(valorVenda) group by cpfAtendente  
from atendimentos;.
```

---

**E.**

```
select cpfAtendente, sum(*) from atendimentos group  
by cpfAtendente;.
```

**4.**

**A linguagem SQL disponibiliza várias funções para a realização de consultas a bancos de dados.**

**Analise as alternativas apresentadas a seguir e marque a que indica uma função que, em um agrupamento, realiza a subseleção de linhas:**

**A.**

**MIN.**

---

**B.**

**GROUP BY.**

---

C.

AVG.

---

D.

MAX.

---

E.

HAVING.

5.

As funções agregadas na linguagem SQL têm como característica gerar um único resultado após o processamento de todos os dados armazenados no banco de dados. Suponha que você deseja calcular a média da idade de uma tabela de atletas da Federação Nacional de Atletismo.

Marque a alternativa que indica a função agregada apropriada para uso na consulta SQL:

A.

MEDS.

---

B.

MAX.

---

C.

MED.

---

D.

AVG.

---

---

E.

SAVG.

### Consultas complexas em SQL

1.

As funções de agregação permitem contar ou realizar operações de soma entre os valores presentes em uma tabela, entre outras opções.

Entre as opções a seguir, a que utiliza corretamente a função utilizada para verificar o valor máximo de um produto em uma consulta SQL é:

A.

SELECT SUM(Valor) FROM PRODUTO;

---

B.

SELECT MIN(Salario) FROM FUNCIONARIO;

---

C.

SELECT MAX(Valor) FROM PRODUTO;

---

---

D.

SELECT AVG(Valor) FROM PRODUTO;

---

E.

SELECT COUNT(Valor) FROM PRODUTO;

2.

A cláusula GROUP BY é utilizada para formar grupos de acordo com atributos definidos na consulta, para que assim possa ser aplicada determinada função de agregação.

Entre as opções a seguir, assinale a que utiliza corretamente a cláusula GROUP BY e uma função de agregação para verificar a quantidade e os nomes dos produtos em estoque em uma consulta SQL:

A.

SELECT NProduto SUM(Valor) FROM PRODUTOS GROUP BY NProduto.

---

B.

SELECT NProduto, SUM(Quantidade) FROM PRODUTOS GROUP BY NProduto;

---

---

C.

SELECT SUM(Valor) FROM PRODUTOS GROUP BY NProduto;

---

D.

SELECT SUM(Valor) GROUP BY Nproduto FROM PRODUTOS;

---

E.

SELECT NProduto, SUM FROM PRODUTOS GROUP BY NProduto;

3.

Ao realizar consultas em um banco de dados em SQL, diferentes cláusulas podem ser utilizadas para encontrar o resultado esperado.

**Sobre as cláusulas em SQL em consultas, é correto afirmar que:**

**A.**

**Uma consulta pode ter até seis cláusulas, mas somente GROUP BY e ORDER BY são obrigatórias**

---

**B.**

**uma consulta pode ter até quatro cláusulas, mas somente SELECT e GROUP BY são obrigatórias.**

---

**C.**

**uma consulta pode ter até cinco cláusulas, mas somente SELECT e WHERE são obrigatórias.**

---

**D.**

**Uma consulta pode ter até seis cláusulas, mas somente SELECT e FROM são obrigatórias.**

---

**E.**

**uma consulta pode ter até sete cláusulas, mas somente FROM e HAVING são obrigatórias.**

**4.**

**Consultas em SQL podem ser escritas e executadas de diferentes formas. Por este motivo, existem vantagens e desvantagens dessa flexibilidade.**

**Sobre as vantagens e/ou desvantagens da flexibilidade de especificação de consultas em SQL, é correto afirmar que:**

**A.**

a disponibilidade de técnicas pode gerar desorganização do usuário, o que é uma vantagem.

---

B.

a possibilidade de a técnica escolhida pelo usuário ser menos eficiente é uma vantagem.

---

C.

o usuário poder escolher a técnica com a qual está acostumado é uma desvantagem.

---

D.

o usuário poder escolher a técnica com a qual está acostumado é uma vantagem.

---

E.

o programador do sistema pode receber a consulta organizada, o que é uma desvantagem.

5.

As consultas aninhadas têm subconsultas embutidas em seu texto.

Analise as seguintes consultas e assinale a opção na qual uma subconsulta na cláusula WHERE em SQL é apresentada na sintaxe correta.

A.

SELECT F.Nome

FROM ALUNOS AS A

WHERE R.Cpf IN

(SELECT F.Cpf  
FROM BY RESPONSVEL AS R  
WHERE R.Nome=F.Nome\_responsavel);

---

B.

SELECT SUM (Salario)  
FROM (FUNCIONARIO JOIN DEPARTAMENTO ON Dnr=Dnumero)  
WHERE D.Nome= 'Financeiro'  
(SELECT D.Nome  
FROM FUNCIONARIO AS A  
WHERE D.Nome=F.Nome\_dep;

---

C.

SELECT A.Nome  
FROM ALUNOS AS A  
WHERE R.Cpf IN RESPONSVEL AS R  
WHERE R.Nome=A.Nome\_responsavel;

---

D.

SELECT A.Nome  
FROM ALUNOS AS B  
WHERE A.Cpf IN  
(SELECT R.Cpf  
FROM RESPONSVEL AS A  
WHERE R.Nome=A.Nome\_responsavel);

---

E.

SELECT A.Nome

FROM ALUNOS AS A

WHERE R.Cpf IN

(SELECT R.Cpf

FROM RESPONSAVEL AS R

WHERE R.Nome=A.Nome\_responsavel);