ORIENTAÇÕES PARA ENTREGA DO RESULTADO DO TRABALHO

Commitar os scripts dentro de uma pasta chamada “BANCO”.

Parte 1 – SQL Conceitual

Questão 1.

Qual o código SQL que deve ser usado para exibir uma única combinação de DEPARTAMENTO e GRUPO da tabela USUARO?

1. SELECT DISTINCT departamento, grupo FROM usuario;
2. SELECT departamento, grupo DISTINCT FROM usuario;
3. SELECT departamento, grupo FROM usuario;
4. SELECT departamento, DISTINCT grupo FROM usuario;

Questão 2.

Marque quais afirmações estão corretas a respeito da clausula ORDER BY:

1. A ordenação é crescente por padrão;
2. A ordenação é decrescente por padrão;
3. A clausula ORDER BY deve ser escrita antes da clausula WHERE;
4. A clausula ORDER BY é executada no lado do cliente;
5. A clausula ORDER BY sempre deverá ser a última clausula de um SELECT;
6. A clausula ORDER BY é sempre a primeira a ser executada em um SELECT;

Questão 3.

Qual o código SQL que executa uma projeção, uma seleção e uma junção quando executado?

1. SELECT p.id\_number, f.fabricante\_id, f.city

FROM produto p, fabricante f

WHERE p.fabricante\_id = f.fabricante\_id

AND f.fabricante\_id = 15;

1. SELECT id\_number, fabricante\_id

FROM produto

ORDER BY fabricante\_id, id\_number;

1. SELECT id\_number, fabricante\_id

FROM produto

WHERE fabricante\_id = 15;

1. SELECT fabricante\_id, city

FROM fabricante

AND fabricante\_id = 15

ORDER BY city;

Questão 4.

Avalie o seguinte script SQL:

CREATE TABLE DEPARTAMENTO

(deptno NUMBER(2),

dnome VARCHAR2(14),

loc VARCHAR2(13));

ROLLBACK;

SELECT COUNT(\*) FROM DEPARTAMENTO;

É correto afirmar que:

1. O comando SELECT COUNT(\*) FROM DEPARTAMENTO exibe o valor 0 pois não existe registros.
2. O comando ROLLBACK libera todo espaço em disco utilizado pela tabela DEPARTAMENTO.
3. O comando SELECT COUNT(\*) FROM DEPARTAMENTO retornara um erro informando que a tabela departamento não existe.
4. O create table não vai funcionar, pois o nome “loc” da coluna é um comando interno de banco de dados.

Questão 5.

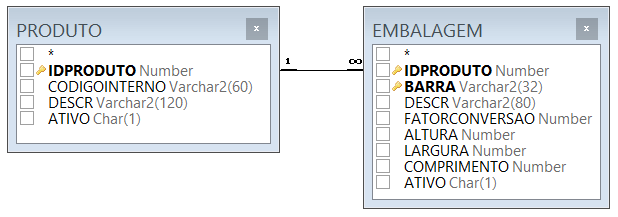
Caso necessite de usar uma função em uma consulta SELECT, qual a melhor função para representar uma lógica similar a IF-THEN-ELSE?

1. SQRT
2. DECODE
3. NEW\_TIME
4. ROWIDTOCHAR

Parte 2 – SQL Prático

Exercício 1.

Com a estrutura de dados abaixo e os exemplos de dados crie:



Dados da tabela produto:

|  |  |  |  |
| --- | --- | --- | --- |
| **IDPRODUTO** | **CODIGOINTERNO** | **DESCR** | **ATIVO** |
| 12 | 000001 | OLEO DE SOJA | S |
| 13 | 000002 | OLEO DE GIRASSOL | S |
| 57 | 000003 | TELEFONE SEM FIO | S |
| 382 | 000004 | MOUSE MICROSOFT | S |
| 435 | 000005 | JOHNNIE WALKER BLUE LABEL | S |

Dados da tabela embalagem:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **IDPRODUTO** | **BARRA** | **DESCR** | **FATORCONVERSAO** | **ALTURA** | **LARGURA** | **COMPRIMENTO** | **ATIVO** |
| 12 | 7891000315507 | OLEO DE SOJA LT | 1 | 200 | 100 | 100 | S |
| 12 | 7891000315508 | OLEO DE SOJA CX COM 10 | 10 | 250 | 400 | 400 | S |
| 13 | 7891000415510 | OLEO DE GIRASSOL LT | 1 | 200 | 100 | 100 | S |
| 13 | 7891000415511 | OLEO DE GIRASSOL CX COM 10 | 10 | 250 | 400 | 400 | S |
| 57 | 7891000415515 | TELEFONE SEM FIO | 1 | 170 | 180 | 190 | S |
| 382 | 7891000715501 | MOUSE MICROSOFT | 1 | 60 | 60 | 110 | S |
| 382 | 7891000715502 | MOUSE MICROSOFT CX COM 10 | 10 | 180 | 200 | 200 | S |
| 382 | 7891000715503 | MOUSE MICROSOFT CX COM 50 | 50 | 400 | 450 | 450 | N |
| 435 | 7891000745678 | JOHNNIE WALKER BLUE LABEL | 1 | 300 | 150 | 150 | S |
| 435 | 7891000745679 | JOHNNIE WALKER BLUE LABEL CX COM 12 | 12 | 350 | 450 | 450 | S |

1. Uma consulta onde mostre os 10 primeiros produtos.
2. Uma consulta onde mostre os produtos apenas com embalagens ativas.
3. Uma consulta que traga quantidade de embalagens de cada produto.
4. Insira um novo produto e uma nova embalagem para esse produto de acordo com a estrutura dados.
5. Altere a altura para 250, largura para 120 e comprimento para 150 das embalagens dos produtos cujo FATORCONVERSAO seja igual a 1.

Parte 3 – SQL Prático

Exercício 1.

Crie as seguintes tabelas, em um script SQL com o nome de SCRIPT1.SQL:

**EXAME\_NF**

|  |  |
| --- | --- |
| IDNF | NUMERO |
| NUMERO | NUMERO |
| DATACADASTRO | DATA |
| TOTALGERAL | NUMERO |

**EXAME\_ITEMNF**

|  |  |
| --- | --- |
| IDITEMNF | NUMERO |
| IDNF | NUMERO |
| IDPRODUTO | NUMERO |
| QTDE | NUMERO |
| VALOR | NUMERO |

Exercício 2.

Crie um segundo script (SCRIPT2.SQL) que possua um bloco de código para que gere registros fictícios para as tabelas criadas. Gere 1000 registros para a tabela EXAME\_NF. Para cada EXAME\_NF gere 3 registros. Faça com que a DATACADASTRO do EXAME\_NF comece em 10 dias atrás, fazendo com que a cada 100 registros a data seja aumentada em 1 dia, distribuindo assim os 1000 registros em 10 dias diferentes de cadastro.

Exercício 3.

Crie uma procedure chamada DEFINIR\_VALORES para que altere o valor dos itens da tabela EXAME\_ITEMNF, definindo números inteiros aleatórios entre 1 e 100. Faça com que a linha da tabela EXAME\_NF receba o valor da somatória de seus itens na coluna TOTALGERAL. Salve o fonte do script no arquivo SCRIPT3.SQL.

Exercício 4.

Crie uma consulta SELECT que exiba o total de linhas da tabela EXAME\_NF por dia. Exiba somente as linhas que possuam ao menos um item (EXAME\_ITEMNF) com valor inferior a 50. Salve a consulta no arquivo SCRIPT4.SQL.

Exercício 5.

Otimize a última consulta SELECT (Exercício 4), medindo o custo e melhore caso seja possível. Crie índices caso necessário. Salve todas alterações, caso existam, no arquivo SCRIPT5.SQL