Estudos de Banco de Banco de Dados I - Lista de exercícios sobre Funções

Professora: Renata Arantes

Nome: João Gabriel da Silva Leite Ra: 24787392

Observação: Printar a sintaxe de criação e invocação da função, do select e resultados

1. Crie uma função para conversão em dólar (US$) do preço do produto da tabela produto. O atributo preço do produto deve estar inserido em alguma tabela, então se necessário acrescente ou crie uma tabela.

```oracle
-- Criação da tabela e inserção de dados
CREATE TABLE PRODUTO (
    COD_PROD NUMBER NOT NULL PRIMARY KEY,
    DESC_PROD VARCHAR2(250),
    VALOR_PROD NUMBER NOT NULL
);

INSERT INTO PRODUTO (COD_PROD, DESC_PROD, VALOR_PROD)
VALUES (1, 'Prodcut One', 2000);

INSERT INTO PRODUTO (COD_PROD, DESC_PROD, VALOR_PROD)
VALUES (2, 'Product Two', 1000);
```

```oracle
-- Sintaxe de Criação da Função
CREATE OR REPLACE FUNCTION convertRealtoDolar (valorProduto NUMBER)
RETURN NUMBER IS
    newPrice NUMBER;
BEGIN
    newPrice := valorProduto * 5.25;
    RETURN newPrice;
END;
```

```sql
-- Invocação da Função
UPDATE PRODUTO
SET VALOR_PROD = convertRealtoDolar(VALOR_PROD);
```

```sql
-- Comando SELECT para Verificar o Resultado
SELECT * FROM PRODUTO;
```

Query executada:

```
SQL>  CREATE TABLE PRODUTO (
  2    COD_PROD NUMBER NOT NULL PRIMARY KEY,
  3    DESC_PROD VARCHAR2(250),
  4    VALOR_PROD NUMBER NOT NULL
  5 );

TABLE created.

SQL>  INSERT INTO PRODUTO (COD_PROD, DESC_PROD, VALOR_PROD)
  2  VALUES (1, 'Prodcut One', 2000);

1 row created.

Commit complete.

SQL>  INSERT INTO PRODUTO (COD_PROD, DESC_PROD, VALOR_PROD)
  2  VALUES (2, 'Product Two', 1000);

1 row created.

Commit complete.

SQL>  CREATE OR REPLACE FUNCTION convertRealtoDolar (valorProduto NUMBER)
  2  RETURN NUMBER IS
  3    newPrice NUMBER;
  4  BEGIN
  5    newPrice := valorProduto * 5.25;
  6    RETURN newPrice;
  7  END;

FUNCTION created.

Commit complete.

SQL>  SELECT * FROM PRODUTO;
```

Max Rows: 500 Columns autosize: Cell contents Save as: JSON

	COD_PROD	DESC_PROD	VALOR_PROD
1	1	Prodcut One	2000
2	2	Product Two	1000

```
SQL>  UPDATE PRODUTO
  2  SET VALOR_PROD = convertRealtoDolar(VALOR_PROD);

2 rows updated.

Commit complete.

SQL>  SELECT * FROM PRODUTO;
```

Max Rows: 500 Columns autosize: Cell contents Save as: JSON

	COD_PROD	DESC_PROD	VALOR_PROD
1	1	Prodcut One	10500
2	2	Product Two	5250

2. Crie uma função para calcular o IMC (Índice de massa corpórea). Para calcular utilize a seguinte fórmula **IMC = peso[kg] / altura²(m)**. O campo peso e altura devem estar inseridos em alguma tabela, então se necessário acrescente ou crie uma tabela.

```oracle_sql
-- Criação da Tabela PESSOA e Inserção de Dados
CREATE TABLE PESSOA (
    ID NUMBER NOT NULL PRIMARY KEY,
    NOME VARCHAR2(100),
    PESO_KG NUMBER NOT NULL,
    ALTURA_M NUMBER NOT NULL
);

INSERT INTO PESSOA (ID, NOME, PESO_KG, ALTURA_M)
VALUES (1, 'João Gabriel', 65, 1.90);

INSERT INTO PESSOA (ID, NOME, PESO_KG, ALTURA_M)
VALUES (2, 'Renata Arantes', 55, 1.55);

-- Sintaxe de Criação da Função IMC
CREATE OR REPLACE FUNCTION calcularIMC (peso NUMBER, altura NUMBER)
RETURN NUMBER IS
    imc NUMBER;
BEGIN
    imc := peso / (altura * altura);
    RETURN imc;
END;

-- Invocação da Função com SELECT para Verificar o Resultado
SELECT NOME, PESO_KG, ALTURA_M, round(calcularIMC(PESO_KG, ALTURA_M), 2) AS IMC FROM PESSOA;
```

```
SQL>   -- Criação da Tabela PESSOA e Inserção de Dados
SQL>   CREATE TABLE PESSOA (
  2     ID NUMBER NOT NULL PRIMARY KEY,
  3     NOME VARCHAR2(100),
  4     PESO_KG NUMBER NOT NULL,
  5     ALTURA_M NUMBER NOT NULL
  6 );

TABLE created.

SQL>   INSERT INTO PESSOA (ID, NOME, PESO_KG, ALTURA_M)
  2  VALUES (1, 'João Gabriel', 65, 1.90);

1 row created.

Commit complete.

SQL>   INSERT INTO PESSOA (ID, NOME, PESO_KG, ALTURA_M)
  2  VALUES (2, 'Renata Arantes', 55, 1.55);

1 row created.

Commit complete.

SQL>   -- Sintaxe de Criação da Função IMC
SQL>   CREATE OR REPLACE FUNCTION calcularIMC (peso NUMBER, altura NUMBER)
  2  RETURN NUMBER IS
  3      imc NUMBER;
  4  BEGIN
  5      imc := peso / (altura * altura);
  6      RETURN imc;
  7  END;

FUNCTION created.

Commit complete.

SQL>   -- Invocação da Função com SELECT para Verificar o Resultado
SQL>   SELECT NOME, PESO_KG, ALTURA_M, round(calcularIMC(PESO_KG, ALTURA_M), 2) AS IMC FROM PESSOA;
```

Max Rows: 500 Columns autosize: Cell contents Save as: JSON

	NOME	PESO_KG	ALTURA_M	IMC
1	João Gabriel	65	1.9	18.01
2	Renata Arantes	55	1.55	22.89

3. Crie uma função que calcule a velocidade de movimento uniformemente variável. Os atributos velocidade inicial (v0), a aceleração (a) e o tempo decorrido (t), devem estar em tabelas. Para efetuar o cálculo utilize a fórmula abaixo:

Velocidade = V0+a*t

```sql
-- Criação da Tabela MOVIMENTO e Inserção de Dados
CREATE TABLE MOVIMENTO (
    ID_MOV NUMBER NOT NULL PRIMARY KEY,
    V0 NUMBER NOT NULL,  -- Velocidade inicial
    A NUMBER NOT NULL,   -- Aceleração
    T NUMBER NOT NULL   -- Tempo
);

INSERT INTO MOVIMENTO (ID_MOV, V0, A, T)
VALUES (1, 10, 2, 5);

INSERT INTO MOVIMENTO (ID_MOV, V0, A, T)
VALUES (2, 15, 3, 4);

-- Sintaxe de Criação da Função para Calcular Velocidade
CREATE OR REPLACE FUNCTION calcularVelocidade (velocidade_inicial NUMBER, aceleracao NUMBER, tempo NUMBER)
RETURN NUMBER IS
    velocidade NUMBER;
BEGIN
    velocidade := velocidade_inicial + aceleracao * tempo;
    RETURN velocidade;
END;

-- Invocação da Função com SELECT para Verificar o Resultado
SELECT V0 AS VelocidadeInicial, A AS Aceleracao, T AS Tempo, calcularVelocidade(V0, A, T) AS VelocidadeFinal
FROM MOVIMENTO;

```

```
SQL>  -- Criação da Tabela MOVIMENTO e Inserção de Dados
SQL>  CREATE TABLE MOVIMENTO (
  2     ID_MOV NUMBER NOT NULL PRIMARY KEY,
  3     V0 NUMBER NOT NULL,  -- Velocidade inicial
  4     A NUMBER NOT NULL,   -- Aceleração
  5     T NUMBER NOT NULL   -- Tempo
  6 );

TABLE created.

SQL>  INSERT INTO MOVIMENTO (ID_MOV, V0, A, T)
  2  VALUES (1, 10, 2, 5);

1 row created.

Commit complete.

SQL>  INSERT INTO MOVIMENTO (ID_MOV, V0, A, T)
  2  VALUES (2, 15, 3, 4);

1 row created.

Commit complete.

SQL>  -- Sintaxe de Criação da Função para Calcular Velocidade
SQL>  CREATE OR REPLACE FUNCTION calcularVelocidade (velocidade_inicial NUMBER, aceleracao NUMBER, tempo NUMBER)
  2  RETURN NUMBER IS
  3     velocidade NUMBER;
  4  BEGIN
  5     velocidade := velocidade_inicial + aceleracao * tempo;
  6     RETURN velocidade;
  7  END;

FUNCTION created.

Commit complete.

SQL>  -- Invocação da Função com SELECT para Verificar o Resultado
SQL>  SELECT V0 AS VelocidadeInicial, A AS Aceleracao, T AS Tempo, calcularVelocidade(V0, A, T) AS VelocidadeFinal
  2  FROM MOVIMENTO;
```

Max Rows: 500 Columns autosize: Column headers Save as: CSV

	VELOCIDADEINICIAL	ACELERACAO	TEMPO	VELOCIDADEFINAL
1	10	2	5	20
2	15	3	4	27

4. Crie uma função para calcular o comprimento, Para efetuar o cálculo utilize a fórmula abaixo. O atributo raio da roda deve estar inserido em alguma tabela, então se necessário acrescente ou crie uma tabela.

comprimento = 2*pi*raio_roda

```oracle
-- Criação da Tabela RODA e Inserção de Dados
CREATE TABLE RODA (
    ID_RODA NUMBER NOT NULL PRIMARY KEY,
    RAIO NUMBER NOT NULL
);

INSERT INTO RODA (ID_RODA, RAIO)
VALUES (1, 2);

INSERT INTO RODA (ID_RODA, RAIO)
VALUES (2, 3);

-- Sintaxe de Criação da Função para Calcular Comprimento
CREATE OR REPLACE FUNCTION calcularComprimento (raio_roda NUMBER)
RETURN NUMBER IS
    comprimento NUMBER;
BEGIN
    comprimento := 2 * 3.14 * raio_roda;
    RETURN comprimento;
END;

-- Invocação da Função com SELECT para Verificar o Resultado
SELECT RAIO AS Raio_Roda, calcularComprimento(RAIO) AS Comprimento
FROM RODA;
```

```
SQL>   -- Criação da Tabela RODA e Inserção de Dados
SQL>   CREATE TABLE RODA (
  2     ID_RODA NUMBER NOT NULL PRIMARY KEY,
  3     RAIO NUMBER NOT NULL
  4 );

TABLE created.

SQL>   INSERT INTO RODA (ID_RODA, RAIO)
  2  VALUES (1, 2);

1 row created.

Commit complete.

SQL>   INSERT INTO RODA (ID_RODA, RAIO)
  2  VALUES (2, 3);

1 row created.

Commit complete.

SQL>   -- Sintaxe de Criação da Função para Calcular Comprimento
SQL>   CREATE OR REPLACE FUNCTION calcularComprimento (raio_roda NUMBER)
  2  RETURN NUMBER IS
  3     comprimento NUMBER;
  4  BEGIN
  5     comprimento := 2 * 3.14 * raio_roda;
  6     RETURN comprimento;
  7  END;

FUNCTION created.

Commit complete.

SQL>   -- Invocação da Função com SELECT para Verificar o Resultado
SQL>   SELECT RAIO AS Raio_Roda, calcularComprimento(RAIO) AS Comprimento
  2  FROM RODA;

Max Rows: 500   Columns autosize: Cell contents   Save as: JSON
```

	RAIO_RODA	COMPRIMENTO
1	2	12.56
2	3	18.84

5. Crie uma função que receba duas notas de avaliação, o atributo nota1 e nota 2 deve estar inserido em alguma tabela, então se necessário acrescente ou crie uma tabela. Calcular:

média (nota1 + nota2)/2.

```sql
-- Criação da Tabela AVALIACAO e Inserção de Dados
CREATE TABLE AVALIACAO (
    ID_AVALIACAO NUMBER NOT NULL PRIMARY KEY,
    NOTA1 NUMBER NOT NULL,
    NOTA2 NUMBER NOT NULL
);

INSERT INTO AVALIACAO (ID_AVALIACAO, NOTA1, NOTA2)
VALUES (1, 8, 10);

INSERT INTO AVALIACAO (ID_AVALIACAO, NOTA1, NOTA2)
VALUES (2, 10, 9.5);

-- Sintaxe de Criação da Função para Calcular a Média
CREATE OR REPLACE FUNCTION calcularMedia (nota1 NUMBER, nota2 NUMBER)
RETURN NUMBER IS
    media NUMBER;
BEGIN
    media := (nota1 + nota2) / 2;
    RETURN media;
END;

-- Invocação da Função com SELECT para Verificar o Resultado
SELECT NOTA1, NOTA2, calcularMedia(NOTA1, NOTA2) AS Media
FROM AVALIACAO;
```

```
SQL>   -- Criação da Tabela AVALIACAO e Inserção de Dados
SQL>   CREATE TABLE AVALIACAO (
  2     ID_AVALIACAO NUMBER NOT NULL PRIMARY KEY,
  3     NOTA1 NUMBER NOT NULL,
  4     NOTA2 NUMBER NOT NULL
  5 );

TABLE created.

SQL>   INSERT INTO AVALIACAO (ID_AVALIACAO, NOTA1, NOTA2)
  2  VALUES (1, 8, 10);

1 row created.

Commit complete.

SQL>   INSERT INTO AVALIACAO (ID_AVALIACAO, NOTA1, NOTA2)
  2  VALUES (2, 10, 9.5);

1 row created.

Commit complete.

SQL>   -- Sintaxe de Criação da Função para Calcular a Média
SQL>   CREATE OR REPLACE FUNCTION calcularMedia (nota1 NUMBER, nota2 NUMBER)
  2  RETURN NUMBER IS
  3     media NUMBER;
  4  BEGIN
  5     media := (nota1 + nota2) / 2;
  6     RETURN media;
  7  END;

FUNCTION created.

Commit complete.

SQL>   -- Invocação da Função com SELECT para Verificar o Resultado
SQL>   SELECT NOTA1, NOTA2, calcularMedia(NOTA1, NOTA2) AS Media
  2  FROM AVALIACAO;
```

Max Rows: 500 Columns autosize: Cell contents Save as: JSON

	NOTA1	NOTA2	MEDIA
1	8	10	9
2	10	9.5	9.75

6. Crie uma função que receba o valor da temperatura em Celsius e converta seu valor em Fahrenheit. O atributo temperatura em Celsius deve estar inserido em alguma tabela, então se necessário acrescente ou crie uma tabela.

$$°F = °C * 1{,}8 + 32.$$

```sql
-- Criação da Tabela TEMPERATURA e Inserção de Dados
CREATE TABLE TEMPERATURA (
    ID_TEMP NUMBER NOT NULL PRIMARY KEY,
    TEMP_CELSIUS NUMBER NOT NULL
);

INSERT INTO TEMPERATURA (ID_TEMP, TEMP_CELSIUS)
VALUES (1, 25);

INSERT INTO TEMPERATURA (ID_TEMP, TEMP_CELSIUS)
VALUES (2, 30);

INSERT INTO TEMPERATURA (ID_TEMP, TEMP_CELSIUS)
VALUES (3, 20);

-- Sintaxe de Criação da Função para Converter Celsius para Fahrenheit
CREATE OR REPLACE FUNCTION converterCelsiusParaFahrenheit (temperaturaCelsius NUMBER)
RETURN NUMBER IS
    temperaturaFahrenheit NUMBER;
BEGIN
    temperaturaFahrenheit := temperaturaCelsius * 1.8 + 32;
    RETURN temperaturaFahrenheit;
END;

-- Invocação da Função com SELECT para Verificar o Resultado
SELECT TEMP_CELSIUS AS Temperatura_Celsius, converterCelsiusParaFahrenheit(TEMP_CELSIUS) AS Temperatura_Fahrenheit
FROM TEMPERATURA;
```

```
SQL>   -- Criação da Tabela TEMPERATURA e Inserção de Dados
SQL>   CREATE TABLE TEMPERATURA (
  2     ID_TEMP NUMBER NOT NULL PRIMARY KEY,
  3     TEMP_CELSIUS NUMBER NOT NULL
  4 );

TABLE created.

SQL>   INSERT INTO TEMPERATURA (ID_TEMP, TEMP_CELSIUS)
  2  VALUES (1, 25);

1 row created.

Commit complete.

SQL>   INSERT INTO TEMPERATURA (ID_TEMP, TEMP_CELSIUS)
  2  VALUES (2, 30);

1 row created.

Commit complete.

SQL>   INSERT INTO TEMPERATURA (ID_TEMP, TEMP_CELSIUS)
  2  VALUES (3, 20);

1 row created.

Commit complete.

SQL>   CREATE OR REPLACE FUNCTION converterCelsiusParaFahrenheit (temperaturaCelsius NUMBER)
  2  RETURN NUMBER IS
  3     temperaturaFahrenheit NUMBER;
  4  BEGIN
  5     temperaturaFahrenheit := temperaturaCelsius * 1.8 + 32;
  6     RETURN temperaturaFahrenheit;
  7  END;

FUNCTION created.

Commit complete.

SQL>   -- Invocação da Função com SELECT para Verificar o Resultado
SQL>   SELECT TEMP_CELSIUS AS Temperatura_Celsius, converterCelsiusParaFahrenheit(TEMP_CELSIUS) AS Temperatura_Fahrenheit
  2  FROM TEMPERATURA;
```

Max Rows: 500 Columns autosize: Cell contents Save as: JSON

	TEMPERATURA_CELSIUS	TEMPERATURA_FAHRENHEIT
1	25	77
2	30	86
3	20	68