14 – Introdução

- Uma **stored function** é um tipo de programa armazenado que sempre retorna um valor. Diferentemente das stored procedures (retorna valores apenas com parâmetros OUT e INOUT), uma função retorna dados por meio de um único valor **RETURN** presente em seu corpo.

- Uma stored function pode ser usada na estrutura de operações básicas de SQL (SELECT, UPDATE, etc.) como se fosse uma função nativa do banco de dados.

- Desta forma, a stored function permite que cálculos complexos sejam encapsulados dentro de uma função, facilitando futuras consultas. Caberá ao usuário apenas inserir a função na estrutura de seu comando SELECT.

- A estrutura de uma stored function é muito parecida com a de uma stored procedure, endo a diferença mais aparente a obrigatoriedade do comando RETURN.

14.1 – Sintaxe de uma Stored Function

- O quadro abaixo mostra a estrutura de uma stored function:

```
DELIMITER $
DROP FUNCTION IF EXISTS <nome da function>$
CREATE FUNCTION <nome da function> RETURNS <tipo>
BEGIN
  <corpo da function>
  RETURN <valor>
END$
DELIMITER ;
```

14.1 – Utilizando uma Stored Function

- O quadro abaixo mostra uma stored function que retorna a área de um quadrado, cujo os lados foi passado como parâmetro:

```
DELIMITER $
DROP FUNCTION IF EXISTS area_retangulo$
CREATE FUNCTION area_retangulo(a INT, b INT) RETURNS INT
BEGIN
  DECLARE c INT;
  SET c=a*b;
  RETURN c;
END$
DELIMITER ;
```

```
SELECT area_retangulo(2,3);
```

area_retangulo(2,3)
6

14.1 – Utilizando uma Stored Function

- O quadro abaixo mostra uma stored function que efetua um reajuste no salario de determinado funcionário, retornando o novo salário para o programa principal:

```
DELIMITER $
CREATE FUNCTION reajuste(id INT, taxa INT) RETURNS DOUBLE
BEGIN
  DECLARE novo_salario FLOAT;
  DECLARE salario_atual FLOAT;
  SELECT salario_fixo INTO salario_atual FROM funcionario WHERE id_funcionario=id;
  SET novo_salario=salario_atual+(salario_atual*taxa/100);
  UPDATE funcionario SET salario_fixo=novo_salario WHERE id_funcionario=id;
  RETURN novo_salario;
END$
DELIMITER ;
```

```
SELECT reajuste(2,50); //aumenta em 50% o salario do funcionário 2
```

14.1 – Utilizando uma Stored Function

- A function pode ser usada para implementar funções que não existem por padrão no banco de dados. O exemplo abaixo mostra uma função que retorna a data em português:

```
DELIMITER $
CREATE FUNCTION `data_portugues`(data DATE) RETURNS varchar(25)
BEGIN
  DECLARE dia, mes1 CHAR(2);
  DECLARE ano CHAR(4);
  DECLARE mes2 VARCHAR(8);
  DECLARE data_final VARCHAR(25);
  SET dia=day(data);
  SET mes1=month(data);
  SET ano=year(data);
```

14.1 – Utilizando uma Stored Function

```
CASE
 WHEN mes1='1' THEN SET mes2='Janeiro';
 WHEN mes1='2' THEN SET mes2='Fevereiro';
 WHEN mes1='3' THEN SET mes2='Março';
 WHEN mes1='4' THEN SET mes2='Abril';
 WHEN mes1='5' THEN SET mes2='Maio';
 WHEN mes1='6' THEN SET mes2='Junho';
 WHEN mes1='7' THEN SET mes2='Julho';
 WHEN mes1='8' THEN SET mes2='Agosto';
 WHEN mes1='9' THEN SET mes2='Setembro';
 WHEN mes1='10' THEN SET mes2='Outubro';
 WHEN mes1='11' THEN SET mes2='Novembro';
 WHEN mes1='12' THEN SET mes2='Dezembro';
END CASE;
```

14.1 – Utilizando uma Stored Function

```
  SET data_final=CONCAT(dia,' de ',mes2,' de ',ano);
  RETURN data_final;
END$
DELIMITER ;
```

```
SELECT nome, data_portugues(data_contratacao) AS 'Contratação',
FORMAT(salario_fixo,2) FROM funcionario;
```

#	nome	Contratação	FORMAT(salario_fixo,2)
1	Paulo	14 de Janeiro de 2016	14,026.32
2	Zé	3 de Fevereiro de 2016	1,089.00

14.1 – Utilizando uma Stored Function

- g

```
  SET data_final=CONCAT(dia,' de ',mes2,' de ',ano);
  RETURN data_final;
END$
DELIMITER ;
```

```
SELECT reajuste(2,50); //aumenta em 50% o salario do funcionário 2
```