Хранимые процедуры

и функции (SQL)

Хранимые процедуры и функции

Объекты базы данных, представляющие собой набор SQL- инструкций, которые компилируются один раз и хранятся на сервере

Основное отличие: *функции могут использоваться,* как любое другое выражение в операторах **SQL** (*например **sum()***), а хранимые процедуры *должны вызываться* с помощью оператора **CALL**.

Процедуры могут и возвращать, и не возвращать значения, функции всегда возвращают значения, которые затем могут использоваться в других операторах **SQL**

Триггер представляет собой хранимую процедуру, которая активизируется при наступлении определенного события.

Создание процедуры CREATE:

```
CREATE PROCEDURE название (аргумент тип_аргумента)
BEGIN
Тело процедуры
END
```

Удаление процедуры DROP:

```
DROP PROCEDURE IF EXIST название_процедуры;
```

Создание функции CREATE:

```
CREATE FUNCTION название (аргумент тип_аргумента)
RETURNS возвращаемый_тип_данных
BEGIN
Тело процедуры
    RETURN значение
END
```

Удаление функции DROP:

```
DROP FUNCTION IF EXIST название_функции;
```

Ключевые слова SQL

Вызов процедуры - CALL

```
CALL название (аргумент тип_аргумента);
```

Объявление переменной - DECLARE

```
DECLARE var INT DEFAULT 0;
```

```
DECLARE var INT;
```

Присваивание значения переменной - SET

```
SET var = 100;
```

Объявление блока для выполнения - BEGIN .. END

```
BEGIN
Последовательность SQL инструкций для выполнения
END;
```

Пример создания функции и ее вызов

```
DELIMITER $$ /*изменяет разделитель оператора с ; на $$*/
CREATE FUNCTION multiplication_10 (starting_value INT) RETURNS int(11)
BEGIN
   DECLARE income INT;
   SET income = 10 * starting_value;
   RETURN income;
END $$
DELIMITER ;
```

Пример: имеется таблица **Customers** (покупатели) с полями **cnum** (уникальный номер заказчика), **cname** (имя), **city** (город), **rating** (рейтинг), **snum** (номер продавца)
Выведем все поля этой таблицы, с рейтингом покупателей умноженным на 10:

```
SELECT cnum, cname, city, multiplication_10(rating), snum FROM Customers
```

Пример создания процедуры и ее вызов

```
DELIMITER $$
CREATE PROCEDURE get_customers_by_city(city_arg VARCHAR(45))
BEGIN
    SELECT c.cname, c.city FROM Customers c WHERE c.city = city_arg;
END $$
DELIMITER ;
```

Пример: имеется таблица **Customers** (покупатели) с полями **cnum** (уникальный номер заказчика), **cname** (имя), **city** (город), **rating** (рейтинг), **snum** (номер продавца)
Получим всех покупателей по названию города:

```
CALL get_customers_by_city('Москва');
```

Пример создания процедуры и ее вызов

```
DROP PROCEDURE change name salespeople; /* На случай, если процедура существует, предварительно удалим ее */

DELIMITER $$
CREATE PROCEDURE change_name_salespeople()
BEGIN
  DECLARE max comm FLOAT;
  SELECT MAX(comm) INTO max comm FROM Salespeople;
  UPDATE Salespeople SET sname = CONCAT(sname, " BOSS") WHERE comm = max comm;
END $$
DELIMITER ;
```

Пример: имеется таблица **Salespeople** (продавцы) с полями **sname** (имя продавца), **comm** (комиссия)
Добавим к имени продавца, с наибольшей комиссией слово “BOSS”:

```
CALL change_name_salespeople;
```

Операторы хранимых процедур

В хранимых процедурах и функциях есть возможность использовать условные операторы и циклы, такие как **IF-ELSE, CASE, WHILE.**

Пример использования конструкции WHILE:

```
DELIMITER //
CREATE FUNCTION func_calc ( starting_value INT )
RETURNS INT

BEGIN
   DECLARE income INT;

   SET income = 0;

   label1: WHILE income <= 8000 DO
     SET income = income + starting_value;
   END WHILE label1;

   RETURN income;
END; //
DELIMITER ;
```

Пример использования конструкции WHILE:

Имеются таблицы **Salespeople** (продавцы) с полями **sname** (имя продавца), **snum** (номер продавца) **и Orders (продажи)** с полями **snum** (номер продавца), **amt** (сумма продажи). Вывести продавцов с общими продажами на большие/средние/маленькие суммы

```
DELIMITER $$
CREATE PROCEDURE get_salespeople (IN str VARCHAR(45))
BEGIN
    CASE str
    WHEN "Маленькие"
    THEN
        SELECT s.sname as "Имя продавца", SUM(amt) as "Суммарные продажи"
        FROM Salespeople s
        INNER JOIN Orders o on s.snum = o.snum
        group by s.sname
        HAVING SUM(o.amt) < 1000;
    WHEN "Средние"
    THEN
        SELECT s.sname as "Имя продавца", SUM(o.amt) as "Суммарные продажи"
        FROM Salespeople s
        INNER JOIN Orders o on s.snum = o.snum
        group by s.sname
        HAVING SUM(o.amt) >= 1000 and SUM(o.amt) < 1500;
    WHEN "Большие"
    THEN
        SELECT s.sname as "Имя продавца", SUM(o.amt) as "Суммарные продажи"
        FROM Salespeople s
        INNER JOIN Orders o on s.snum = o.snum
        group by s.sname
        HAVING SUM(o.amt) >= 1500;
    END CASE;
END$$
DELIMITER ;
```