**ТЕМА 9: ФУНКЦИИ SQL**

1. **Введение в SQL функции**

SQL функции представляют собой набор инструкций SQL, объединенных в одну единицу, которую можно вызывать и использовать повторно. Они могут принимать аргументы, выполнять операции и возвращать результат. В этом разделе мы рассмотрим различные типы SQL функций и приведем примеры их использования.

**Пример 1: Отображение пользовательских таблиц**

-- Этот запрос отображает все пользовательские таблицы в базе данных

SELECT \* FROM Sys.objects WHERE Type='u'

Этот запрос отображает все пользовательские таблицы в базе данных, где Sys.objects - системная таблица, содержащая информацию о объектах в базе данных.

**Пример 2: Создание и вызов функции**

-- Создание функции, которая обновляет регион во временной таблице create or replace function fix\_customer\_region() returns void AS $$

update tmp\_customers

set region = 'unknown'

where region is null

$$ language sql;

-- Вызов функции

select fix\_customer\_region()

В этом примере создана функция fix\_customer\_region, которая обновляет регион во временной таблице tmp\_customers. Затем вызывается эту функцию с помощью select fix\_customer\_region().

1. **Скалярные функции**

Скалярные функции - это функции, которые возвращают одно значение. Они могут принимать аргументы и выполнять операции над ними.

**Пример 1: Получение цены продукта по его имени**

-- Создание функции, которая возвращает цену продукта по его имени

create or replace function get\_product\_price\_by\_name(prod\_name varchar) returns real as $$

select unit\_price

from products

where product\_name = prod\_name

$$ language sql;

-- Вызов функции

select get\_product\_price\_by\_name('Chocolade')

В этом примере создана функция get\_product\_price\_by\_name, которая принимает имя продукта в качестве аргумента и возвращает его цену из таблицы products.

**Пример 2: Получение максимального количества заказа по стране** **доставки**

-- Создание функции, которая возвращает максимальное количество заказов по стране доставки create or replace function get\_order\_quantity(ship\_country varchar) returns smallint as $$

select MAX(quantity)

from orders

join order\_details using(order\_id)

where ship\_country = get\_order\_quantity.ship\_country

$$ language sql;

-- Вызов функции

select get\_order\_quantity('France');

В этом примере создана функция get\_order\_quantity, которая принимает страну доставки в качестве аргумента и возвращает максимальное количество заказов для этой страны.

1. **Аргументы функций SQL: IN, OUT, DEFAULT**

SQL функции могут иметь аргументы входного и выходного типов, а также значения по умолчанию.

**Пример 1: Функция с выходными параметрами**

-- Создание функции, которая возвращает максимальную и минимальную цену продукта

create or replace function get\_price\_boundaries(out max\_price real, out min\_price real) AS $$

SELECT MAX(unit\_price), MIN(unit\_price)

FROM products

$$ language sql;

-- Вызов функции

select get\_price\_boundaries()

В этом примере создана функция get\_price\_boundaries, которая возвращает максимальную и минимальную цену продукта в выходных параметрах.

**Пример 2: Функция с аргументом по умолчанию**

-- Создание функции с аргументом по умолчанию

create or replace function get\_price\_boundaries\_by\_discontinuity

(in is\_discontinued int DEFAULT 1, out max\_price real, out min\_price real) AS $$

SELECT MAX(unit\_price), MIN(unit\_price)

FROM products

where discontinued = is\_discontinued

$$ language sql;

-- Вызов функции с указанием аргумента по умолчанию

select get\_price\_boundaries\_by\_discontinuity(1);

-- Вызов функции без указания аргумента (используется значение по умолчанию) select get\_price\_boundaries\_by\_discontinuity();

В этом примере создана функция get\_price\_boundaries\_by\_discontinuity, которая принимает аргумент is\_discontinued со значением по умолчанию равным 1. Вызывая функцию без указания аргумента, используется значение по умолчанию.

1. **Возврат наборов данных**

SQL функции также могут возвращать наборы данных. Ниже приведен пример функции, возвращающей случайное число.

**Пример: Функция, возвращающая случайное число**

-- Создание функции, возвращающей случайное число в заданном диапазоне

CREATE OR REPLACE FUNCTION random\_between(low INT ,high INT)

RETURNS INT AS

$$

BEGIN

RETURN floor(random()\* (high-low + 1) + low);

END;

$$ language 'plpgsql' STRICT;

В этом примере создается функция random\_between, которая возвращает случайное число в заданном диапазоне от low до high. Функция использует PL/pgSQL (язык для PostgreSQL) для определения логики функции.