**ТЕМА 7. ПРЕДСТАВЛЕНИЯ В SQL**

1. **Введение в представления SQL**

Представление SQL (SQL view) - это виртуальная таблица, создаваемая на основе одной или нескольких таблиц, которая позволяет пользователям выполнять запросы к данным, как если бы они работали с обычной таблицей. Представления упрощают доступ к данным и обеспечивают абстракцию от фактической структуры базы данных.

1. **Создание представления SQL**

Для создания представления SQL используется оператор CREATE VIEW. Вот пример создания представления, которое показывает список сотрудников:

CREATE VIEW products\_suppliers\_categories AS

SELECT product\_name, quantity\_per\_unit, unit\_price, units\_in\_stock, company\_name, contact\_name, phone,

category\_name, description

FROM products

JOIN suppliers USING (supplier\_id)

JOIN categories USING (category\_id);

SELECT \*

FROM products\_suppliers\_categories;

SELECT \*

FROM products\_suppliers\_categories

WHERE unit\_price > 20;

DROP VIEW products\_suppliers\_categories;

DROP VIEW IF EXISTS products\_suppliers\_categories;

1. **Обновляемые представления SQL**

Обычно представления SQL предназначены только для чтения, и изменения данных в них не разрешены. Однако в некоторых случаях можно создать обновляемые представления с помощью правил или триггеров. Пример:

SELECT \* FROM orders;

CREATE OR REPLACE VIEW heavy\_orders AS

SELECT \*

FROM orders

WHERE freight > 50;

SELECT \*

FROM heavy\_orders

ORDER BY freight;

CREATE OR REPLACE VIEW heavy\_orders AS

SELECT \*

FROM orders

WHERE freight > 100;

CREATE OR REPLACE VIEW products\_suppliers\_categories AS

SELECT product\_name, quantity\_per\_unit, unit\_price, units\_in\_stock, discontinued, company\_name, contact\_name, phone, country,

category\_name, description

FROM products

JOIN suppliers USING (supplier\_id)

JOIN categories USING (category\_id);

-----

SELECT MAX(order\_id)

FROM orders;

INSERT INTO heavy\_orders

VALUES (11078, 'VINET', 5, '2019-12-10', '2019-12-15', '2019-12-14', 1, 120,

'Hanari Carnes', 'Rua do Paco', 'Bern', NULL, 3012, 'Switzerland');

SELECT \*  
FROM heavy\_orders

ORDER BY order\_id DESC;

--показать, что мы не можем удалить то, чего нет в самой вьюхе

SELECT MIN(freight)

FROM orders;

DELETE FROM heavy\_orders WHERE freight < 0.05;

--а теперь удаляем то, что есть во вьюхе

SELECT MIN(freight)

FROM heavy\_orders;

DELETE FROM heavy\_orders WHERE freight < 100.25;

DELETE FROM order\_details

WHERE order\_id = 10854;

В этом примере создаётся правило update\_employee\_view, которое перенаправляет операции обновления из представления employee\_view к соответствующей таблице employees.

1. **Опция CHECK в представлениях**

Опция CHECK в представлениях SQL позволяет применять ограничения к данным, возвращаемым представлением. Например, можно использовать CHECK для фильтрации строк по определенному условию. Пример:

CREATE OR REPLACE VIEW heavy\_orders AS

SELECT \*

FROM orders

WHERE freight > 100;

SELECT \*

FROM heavy\_orders

ORDER BY freight;

INSERT INTO heavy\_orders

VALUES(11900, 'FOLIG', 1, '2000-01-01', '2000-01-05', '2000-01-04', 1, 80, 'Folies gourmandes', '184, chaussee de Tournai', 'Lille', NULL, 59000, 'FRANCE');

SELECT \*

FROM heavy\_orders

WHERE order\_id = 11900;

CREATE OR REPLACE VIEW heavy\_orders AS

SELECT \*

FROM orders

WHERE freight > 100

WITH LOCAL CHECK OPTION;

CREATE OR REPLACE VIEW heavy\_orders AS

SELECT \*

FROM orders

WHERE freight > 100

WITH CASCADE CHECK OPTION;