MySQL

HW5

#### **Домашнее задание Семинар 5**

(описание таблиц см. презентацию к семинару)

1. Создайте представление, в которое попадут автомобили стоимостью до 25 000 долларов CREATE VIEW CheapCars AS SELECT Name FROM Cars WHERE Cost<25000;

CREATE DATABASE hw5;

USE hw5;

CREATE TABLE cars(

id INT PRIMARY KEY AUTO\_INCREMENT NOT NULL,

name varchar(50),

cost int);

INSERT INTO cars (name, cost) VALUES(

"AUDI", 52642),

("MERCEDES", 57127),

("SKODA", 9000),

("VOLVO", 2900),

("BENTLEY", 350000),

("CITROEN", 21000),

("HAMMER", 41400),

("VOLKSWAGEN", 21600);

CREATE VIEW economy\_cars AS

(SELECT \* FROM cars WHERE cost <25000);

SELECT\* FROM economy\_cars;

2. Изменить в существующем представлении порог для стоимости: пусть цена будет до 30 000 долларов (используя оператор ALTER VIEW) ALTER VIEW CheapCars AS SELECT Name FROM CarsWHERE Cost<30000;

ALTER VIEW economy\_cars AS

(SELECT \* FROM cars WHERE cost <49000);

3. Создайте представление, в котором будут только автомобили марки “Шкода” и “Ауди” (аналогично)

ALTER VIEW economy\_cars AS

(SELECT \* FROM cars WHERE cost <49000 AND name = "SKODA" OR name = "AUDI");

4. Вывести название и цену для всех анализов, которые продавались 5 февраля 2020 и всю следующую неделю. Есть таблица анализов Analysis: an\_id — ID анализа; an\_name — название анализа; an\_cost — себестоимость анализа; an\_price — розничная цена анализа; an\_group — группа анализов. Есть таблица групп анализов Groups: gr\_id — ID группы; gr\_name — название группы; gr\_temp — температурный режим хранения. Есть таблица заказов Orders: ord\_id — ID заказа; ord\_datetime — дата и время заказа; ord\_an — ID анализа.

-- 4

CREATE TABLE a\_groups(

gr\_id INT PRIMARY KEY AUTO\_INCREMENT NOT NULL,

gr\_name VARCHAR (150),

gr\_temp DECIMAL (8, 4));

CREATE TABLE Analysis (

an\_id INT PRIMARY KEY AUTO\_INCREMENT NOT NULL,

an\_name VARCHAR (150),

an\_cost DECIMAL (16, 2),

an\_price DECIMAL (16,2),

an\_group INT,

FOREIGN KEY (an\_group) REFERENCES a\_groups (gr\_id));

CREATE TABLE Orders(

ord\_id INT PRIMARY KEY AUTO\_INCREMENT NOT NULL,

ord\_datetime DATETIME,

ord\_an INT,

FOREIGN KEY (ord\_an) REFERENCES Analysis(an\_id));

INSERT INTO Analysis (an\_name, an\_cost, an\_price, an\_group)

VALUES ("ОАК", 560, 650, 1),

("gemoglobin", 300, 450, 1),

("kalcitonin", 900, 1200, 1),

("TTG", 300, 700, 1),

("PH", 200, 400, 2),

("leikocyt", 300, 500, 2),

("eozonofil", 400, 500, 1);

SELECT \* FROM Analysis;

INSERT INTO a\_groups (gr\_id, gr\_name, gr\_temp) VALUES

(1, "BLOOD", 36.3), (2, "URINE", 35.4), (3, "salvia", 28.6);

INSERT INTO Orders (ord\_datetime, ord\_an) VALUES

('2017-02-01 18:00:00', 8),

('2017-02-01 19:15:00', 9),

('2018-01-12 17:00:00', 10),

('2018-01-12 14:00:00', 11),

('2018-01-12 12:00:00', 12),

('2019-02-01 11:15:00', 13),

('2019-02-01 12:15:00', 11),

('2019-02-01 10:15:00', 12),

('2020-01-10 17:00:00', 12),

('2020-02-01 21:00:12', 13),

('2020-01-12 13:25:00', 14),

('2021-05-02 14:45:00', 14),

('2022-06-10 18:21:00', 11),

('2022-02-01 14:45:00', 9),

('2022-01-12 12:14:00', 8),

('2022-05-02 15:41:00', 14);

SELECT an\_name, an\_price FROM Analysis

JOIN Orders

ON Orders.ord\_an = Analysis.an\_id

WHERE DATE(ord\_datetime) BETWEEN '2020-05-02' AND '2020-05-09';

5. Добавьте новый столбец под названием «время до следующей станции». Чтобы получить это значение, мы вычитаем время станций для пар смежных станций. Мы можем вычислить это значение без использования оконной функции SQL, но это может быть очень сложно. Проще это сделать с помощью оконной функции LEAD . Эта функция сравнивает значения из одной строки со следующей строкой, чтобы получить результат. В этом случае функция сравнивает значения в столбце «время» для станции со станцией сразу после нее.

CREATE TABLE trains (

tr\_id INT,

station VARCHAR(20),

station\_time TIME);

INSERT INTO trains(tr\_id, station, station\_time)

VALUES (110, "SAN FRANCISCO", '10:00:00'),

(110, 'REDWOOD CITY', '10:54:00'),

(110, 'PLATO ALTO', '11:02:00'),

(110, 'SAN JOSE', '12:35:00'),

(120, 'SAN FRANCISCO', '11:00:00'),

(120, 'PALO ALTO', '12:49:00'),

(120, 'SAN JOSE', '13:30:00');

SELECT \* FROM trains;

SELECT \*, TIME(

LEAD (station\_time) OVER (ORDER BY station\_time) - station\_time) AS time\_to\_the\_next\_station

FROM trains;