1. Задание (сделано)

В базе данных shop и sample присутствуют одни и те же таблицы, учебной базы данных. Переместите запись id = 1 из таблицы shop.users в таблицу sample.users. Используйте транзакции.

SELECT \* FROM shop.users;

SELECT \* FROM sample.users;

START TRANSACTION;

SELECT \* FROM shop.users WHERE id = 1;

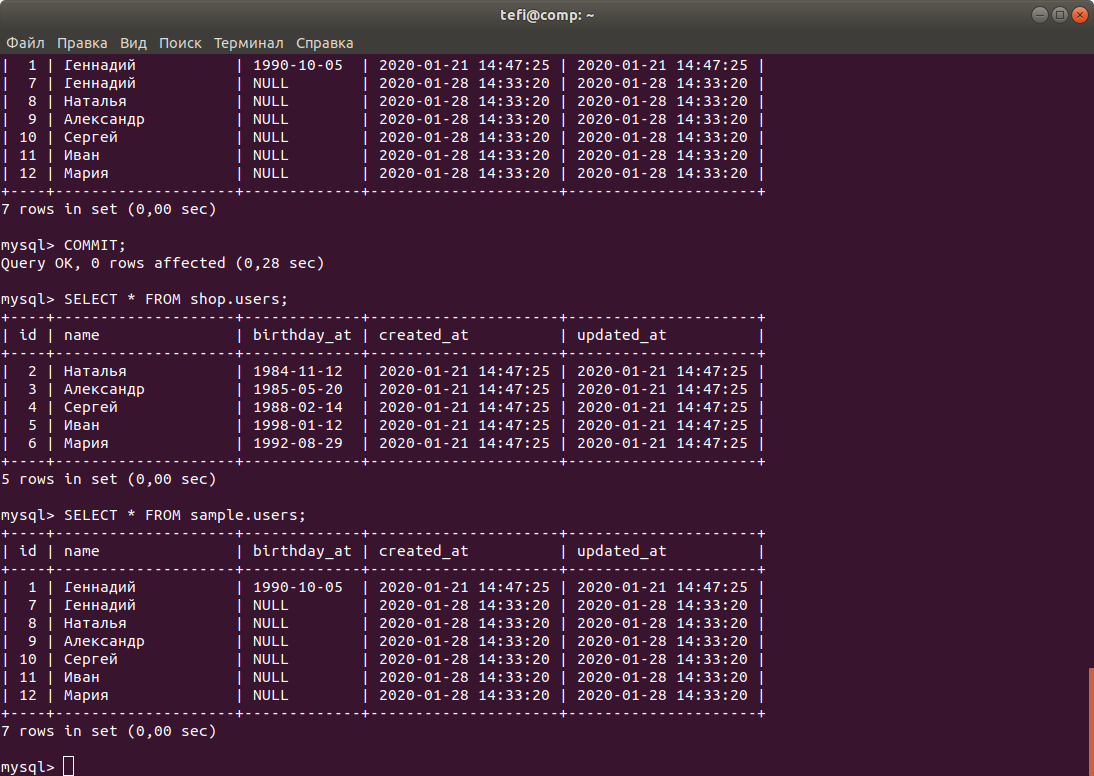
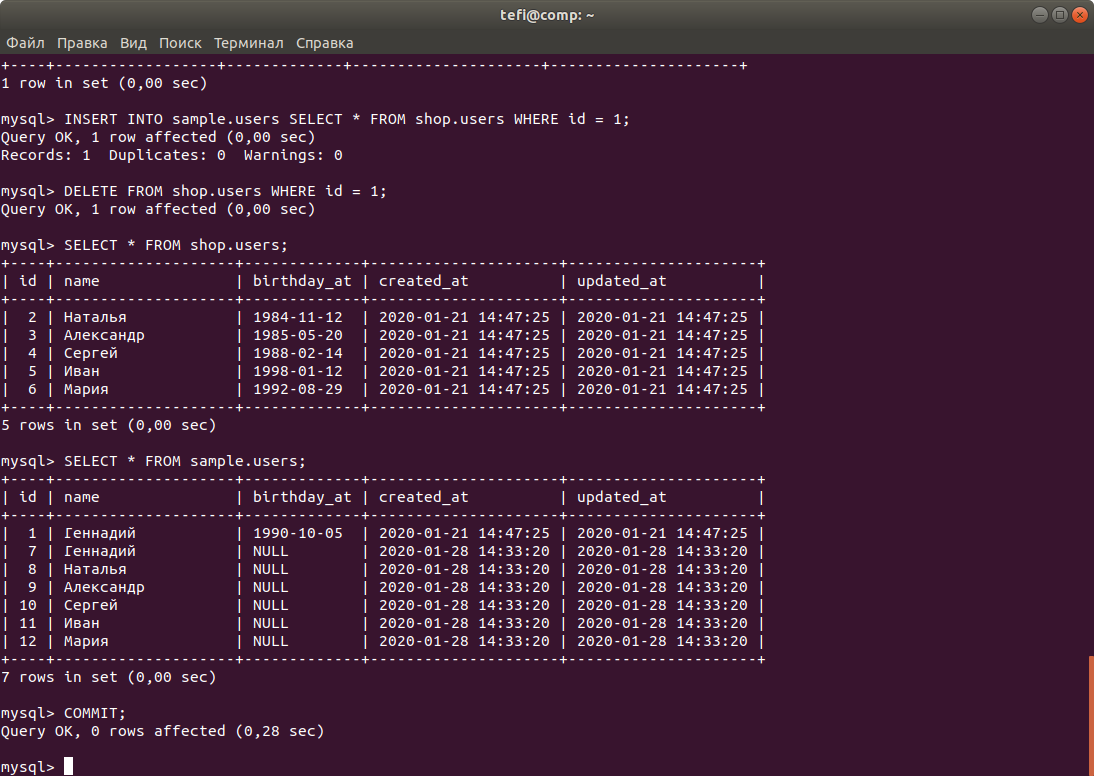
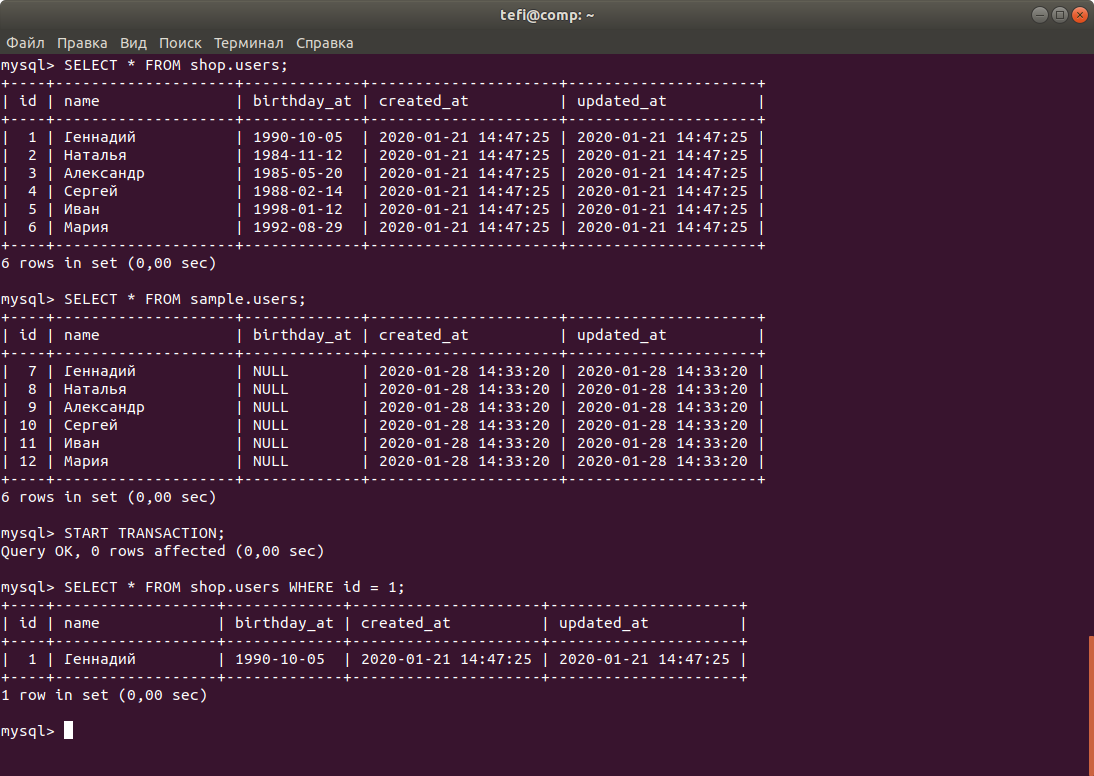
INSERT INTO sample.users SELECT \* FROM shop.users WHERE id = 1;

DELETE FROM shop.users WHERE id = 1;

SELECT \* FROM shop.users;

SELECT \* FROM sample.users;

COMMIT;



1. Задание (сделано)

Создайте представление, которое выводит название name товарной позиции из таблицы products и соответствующее название каталога name из таблицы catalogs.

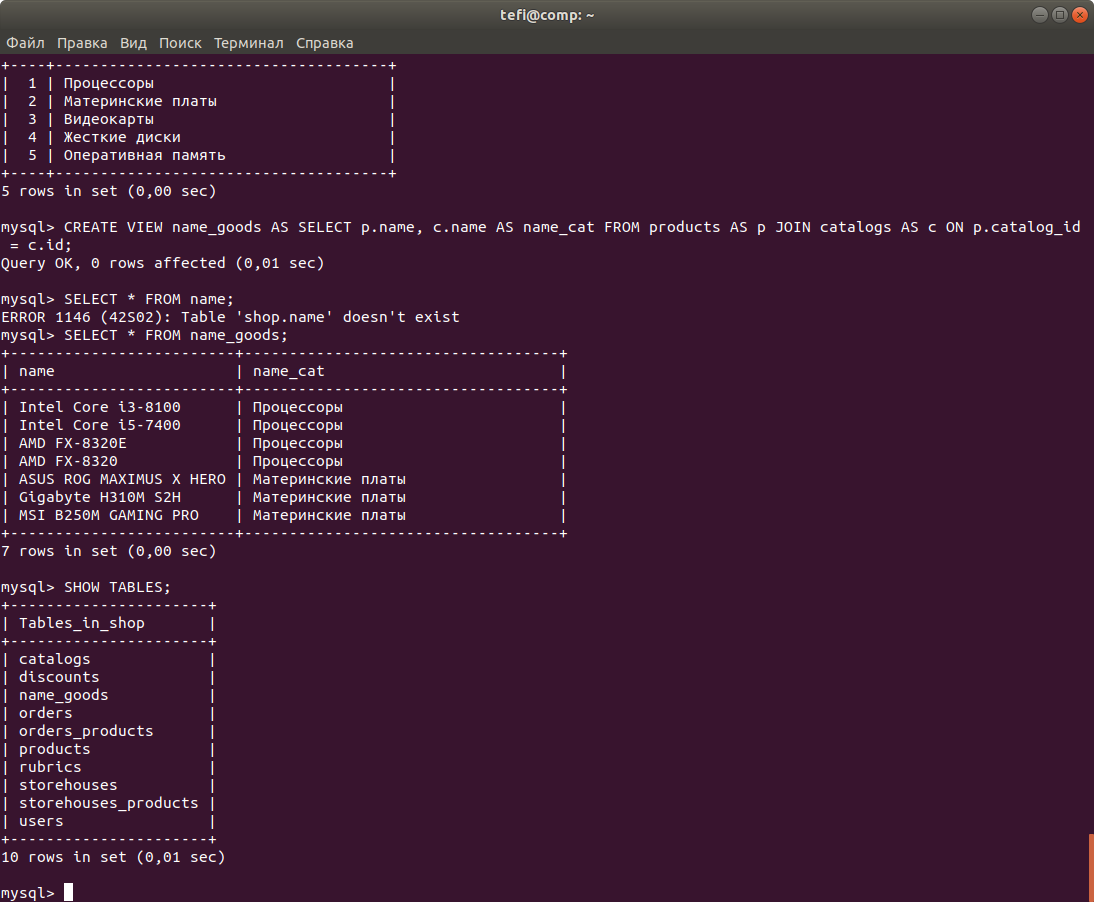
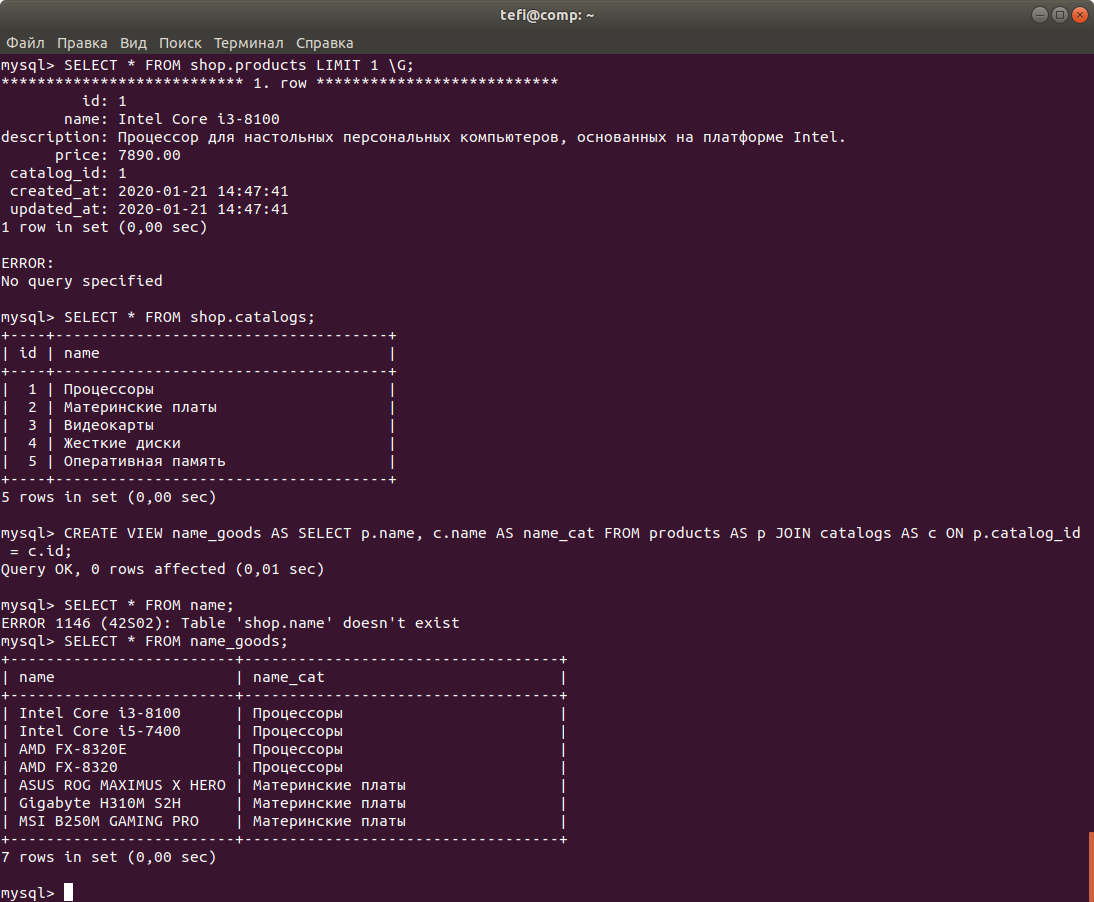
SELECT \* FROM shop.products LIMIT 1 \G;

SELECT \* FROM shop.catalogs;

CREATE VIEW name\_goods AS SELECT p.name, c.name AS name\_cat FROM products AS p JOIN catalogs AS c ON p.catalog\_id = c.id;

SELECT \* FROM name\_goods;

SHOW TABLES;



1. Задание

(по желанию) Пусть имеется таблица с календарным полем created\_at. В ней размещены разряженые календарные записи за август 2018 года '2018-08-01', '2016-08-04', '2018-08-16' и 2018-08-17. Составьте запрос, который выводит полный список дат за август, выставляя в соседнем поле значение 1, если дата присутствует в исходном таблице и 0, если она отсутствует.

CREATE TABLE IF NOT EXISTS posts (

id SERIAL PRIMARY KEY,

name VARCHAR(255),

created\_at DATE NOT NULL

);

INSERT INTO posts VALUES

(NULL, 'первая запись', '2018-08-01'),

(NULL, 'вторая запись', '2018-08-04'),

(NULL, 'третья запись', '2018-08-16'),

(NULL, 'четвертая запись', '2018-08-17');

CREATE TEMPORARY TABLE last\_days (

day INT

);

INSERT INTO last\_days VALUES

(0), (1), (2), (3), (4), (5), (6), (7), (8), (9), (10),

(11), (12), (13), (14), (15), (16), (17), (18), (19), (20),

(21), (22), (23), (24), (25), (26), (27), (28), (29), (30);

SELECT

DATE(DATE('2018-08-31') - INTERVAL l.day DAY) AS day,

NOT ISNULL(p.name) AS order\_exist

FROM

last\_days AS l

LEFT JOIN

posts AS p

ON

DATE(DATE('2018-08-31') - INTERVAL l.day DAY) = p.created\_at

ORDER BY

day;

1. Задание (по желанию) Пусть имеется любая таблица с календарным полем created\_at. Создайте запрос, который удаляет устаревшие записи из таблицы, оставляя только 5 самых свежих записей.

INSERT INTO users\_1 (name, created\_at, updated\_at) VALUES ('Геннадий', '2017-10-12 21:47:25', '2018-10-12 17:47:25'), ('Наталья', '2019-10-12 14:47:25', '2020-10-12 01:47:25'), ('Александр', '2010-10-12 02:47:25', '2012-10-12 03:47:25'), ('Сергей', '2005-10-12 08:47:25', '2017-10-12 09:47:25'), ('Иван', '2017-10-12 12:47:25', '2020-10-12 10:47:25'), ('Мария', '2020-10-12 18:47:25', '2020-11-12 15:47:25');

SELECT \* FROM users\_1;

SELECT u.name, u.created\_at FROM users\_1 AS u ORDER BY created\_at DESC;

DELETE FROM users\_1 ORDER BY created\_at DESC LIMIT 5,1;

DROP TABLE IF EXISTS posts;

CREATE TABLE IF NOT EXISTS posts (

id SERIAL PRIMARY KEY,

name VARCHAR(255),

created\_at DATE NOT NULL

);

INSERT INTO posts VALUES

(NULL, 'первая запись', '2018-11-01'),

(NULL, 'вторая запись', '2018-11-02'),

(NULL, 'третья запись', '2018-11-03'),

(NULL, 'четвертая запись', '2018-11-04'),

(NULL, 'пятая запись', '2018-11-05'),

(NULL, 'шестая запись', '2018-11-06'),

(NULL, 'седьмая запись', '2018-11-07'),

(NULL, 'восьмая запись', '2018-11-08'),

(NULL, 'девятая запись', '2018-11-09'),

(NULL, 'десятая запись', '2018-11-10');

DELETE

posts

FROM

posts

JOIN

(SELECT

created\_at

FROM

posts

ORDER BY

created\_at DESC

LIMIT 5, 1) AS delpst

ON

posts.created\_at <= delpst.created\_at;

SELECT \* FROM posts;

1. Задание

Создайте двух пользователей которые имеют доступ к базе данных shop. Первому пользователю shop\_read должны быть доступны только запросы на чтение данных, второму пользователю shop — любые операции в пределах базы данных shop.

CREATE USER 'shop\_read'@'localhost';

GRANT SELECT, SHOW VIEW ON shop.\* TO 'shop\_read'@'localhost' IDENTIFIED BY '';

SHOW DATABASES;

USE shop;

SHOW TABLES;

SELECT \* FROM catalogs;

INSERT INTO catalogs (name) VALUES ('Оперативная память');

CREATE USER 'shop'@'localhost';

GRANT ALL ON shop.\* TO 'shop'@'localhost' IDENTIFIED BY '';

1. Задание

(по желанию) Пусть имеется таблица accounts содержащая три столбца id, name, password, содержащие первичный ключ, имя пользователя и его пароль. Создайте представление username таблицы accounts, предоставляющий доступ к столбца id и name. Создайте пользователя user\_read, который бы не имел доступа к таблице accounts, однако, мог бы извлекать записи из представления username.

DROP TABLE IF EXISTS accounts;

CREATE TABLE accounts (

id SERIAL PRIMARY KEY,

name VARCHAR(255),

password VARCHAR(255)

);

INSERT INTO accounts (name, password) VALUES

('Геннадий', 'Qt3X08VetW'),

('Наталья', 'hvg0b057Br'),

('Александр', 'a4YGUJjRLk'),

('Сергей', 'YYug1IeyWl'),

('Иван', 'oKoo7KXvTE'),

('Мария', 'w5r4yvfo9f');

CREATE VIEW username AS SELECT id, name FROM accounts;

SELECT \* FROM username;

CREATE USER 'user\_read'@'localhost';

GRANT SELECT (id, name) ON shop.username TO 'user\_read'@'localhost';

1. Задание Создайте хранимую функцию hello(), которая будет возвращать приветствие, в зависимости от текущего времени суток. С 6:00 до 12:00 функция должна возвращать фразу "Доброе утро", с 12:00 до 18:00 функция должна возвращать фразу "Добрый день", с 18:00 до 00:00 — "Добрый вечер", с 00:00 до 6:00 — "Доброй ночи".

DELIMITER //

DROP FUNCTION IF EXISTS hello//

CREATE FUNCTION hello()

RETURNS VARCHAR(255) DETERMINISTIC  
BEGIN

DECLARE time INT;

SET time = DATE\_FORMAT(NOW(), '%H');

CASE

WHEN time > -1 AND time < 6 THEN

RETURN "Доброй ночи";

WHEN time > 17 AND time < 24 THEN

RETURN "Добрый вечер";

WHEN time > 11 AND time < 18 THEN

RETURN "Добрый день";

WHEN time > 5 AND time < 12 THEN

RETURN "Доброе утро";

ELSE

RETURN "Доброго здравия!";

END CASE;

END//

SELECT hello()//

1. Задание(не заработало)

В таблице products есть два текстовых поля: name с названием товара и description с его описанием. Допустимо присутствие обоих полей или одно из них. Ситуация, когда оба поля принимают неопределенное значение NULL неприемлема. Используя триггеры, добейтесь того, чтобы одно из этих полей или оба поля были заполнены. При попытке присвоить полям NULL-значение необходимо отменить операцию.

DELIMITER //

CREATE TRIGGER validate\_name\_description\_insert BEFORE INSERT ON products

FOR EACH ROW BEGIN

IF NEW.name IS NULL AND NEW.description IS NULL THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Both name and description are NULL';

END IF;

END//

INSERT INTO products

(name, description, price, catalog\_id)

VALUES

(NULL, NULL, 9360.00, 2)//

INSERT INTO products

(name, description, price, catalog\_id)

VALUES

('ASUS PRIME Z370-P', 'HDMI, SATA3, PCI Express 3.0,, USB 3.1', 9360.00, 2)//

INSERT INTO products

(name, description, price, catalog\_id)

VALUES

(NULL, 'HDMI, SATA3, PCI Express 3.0,, USB 3.1', 9360.00, 2)//

CREATE TRIGGER validate\_name\_description\_update BEFORE UPDATE ON products

FOR EACH ROW BEGIN

IF NEW.name IS NULL AND NEW.description IS NULL THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Both name and description are NULL';

END IF;

END//

1. Задание

(по желанию) Напишите хранимую функцию для вычисления произвольного числа Фибоначчи. Числами Фибоначчи называется последовательность в которой число равно сумме двух предыдущих чисел. Вызов функции FIBONACCI(10) должен возвращать число 55.

Код на питоне (чтобы убедиться, что алгоритм работает):

num = int(input())

n\_1 = 0

n\_2 = 1

n = 0

coun = 1

while coun <= num:

n = n\_1 + n\_2

coun += 1

if coun == num:

print(n)

n\_1 = n\_2

n\_2 = n

DELIMITER //

DROP FUNCTION IF EXISTS fibonacci;

CREATE FUNCTION fibonacci(num INT)

RETURNS INT DETERMINISTIC

BEGIN

DECLARE n\_1, n\_2, n, coun INT;

SET n\_1 = 0;

SET n\_2 = 1;

SET n = 0;

SET coun = 1;

WHILE coun <= num DO

SET n = n\_1 + n\_2;

SET coun = coun + 1;

SET n\_1 = n\_2;

SET n\_2 = n;

IF coun = num THEN

RETURN n;

END IF;

END WHILE;

END//

SELECT fibonacci(10)//