```sql
DROP TABLE IF EXISTS catalogs;
CREATE TABLE catalogs (
  id SERIAL PRIMARY KEY,
  name VARCHAR(255) COMMENT 'Название раздела',
  UNIQUE unique_name(name(10))
) COMMENT = 'Разделы интернет-магазина';

INSERT INTO catalogs VALUES
  (NULL, 'Процессоры'),
  (NULL, 'Материнские платы'),
  (NULL, 'Видеокарты'),
  (NULL, 'Жесткие диски'),
  (NULL, 'Оперативная память');

DROP TABLE IF EXISTS users;
CREATE TABLE users (
  id SERIAL PRIMARY KEY,
  name VARCHAR(255) COMMENT 'Имя покупателя',
  birthday_at DATE COMMENT 'Дата рождения',
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
) COMMENT = 'Покупатели';

INSERT INTO users (name, birthday_at) VALUES
  ('Геннадий', '1990-10-05'),
  ('Наталья', '1984-11-12'),
  ('Александр', '1985-05-20'),
  ('Сергей', '1988-02-14'),
  ('Иван', '1998-01-12'),
  ('Мария', '1992-08-29');

DROP TABLE IF EXISTS products;
CREATE TABLE products (
  id SERIAL PRIMARY KEY,
  name VARCHAR(255) COMMENT 'Название',
  desription TEXT COMMENT 'Описание',
  price DECIMAL (11,2) COMMENT 'Цена',
  catalog_id INT UNSIGNED,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
  KEY index_of_catalog_id (catalog_id)
) COMMENT = 'Товарные позиции';

INSERT INTO products
  (name, desription, price, catalog_id)
VALUES
  ('Intel Core i3-8100', 'Процессор для настольных персональных
компьютеров, основанных на платформе Intel.', 7890.00, 1),
  ('Intel Core i5-7400', 'Процессор для настольных персональных
компьютеров, основанных на платформе Intel.', 12700.00, 1),
```

```
  ('AMD FX-8320E', 'Процессор для настольных персональных
компьютеров, основанных на платформе AMD.', 4780.00, 1),
  ('AMD FX-8320', 'Процессор для настольных персональных
компьютеров, основанных на платформе AMD.', 7120.00, 1),
  ('ASUS ROG MAXIMUS X HERO', 'Материнская плата ASUS ROG MAXIMUS
X HERO, Z370, Socket 1151-V2, DDR4, ATX', 19310.00, 2),
  ('Gigabyte H310M S2H', 'Материнская плата Gigabyte H310M S2H,
H310, Socket 1151-V2, DDR4, mATX', 4790.00, 2),
  ('MSI B250M GAMING PRO', 'Материнская плата MSI B250M GAMING
PRO, B250, Socket 1151, DDR4, mATX', 5060.00, 2);

DROP TABLE IF EXISTS orders;
CREATE TABLE orders (
  id SERIAL PRIMARY KEY,
  user_id INT UNSIGNED,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
  KEY index_of_user_id(user_id)
) COMMENT = 'Заказы';

DROP TABLE IF EXISTS orders_products;
CREATE TABLE orders_products (
  id SERIAL PRIMARY KEY,
  order_id INT UNSIGNED,
  product_id INT UNSIGNED,
  total INT UNSIGNED DEFAULT 1 COMMENT 'Количество заказанных
товарных позиций',
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
) COMMENT = 'Состав заказа';

DROP TABLE IF EXISTS discounts;
CREATE TABLE discounts (
  id SERIAL PRIMARY KEY,
  user_id INT UNSIGNED,
  product_id INT UNSIGNED,
  discount FLOAT UNSIGNED COMMENT 'Величина скидки от 0.0 до 1.0',
  started_at DATETIME,
  finished_at DATETIME,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
  KEY index_of_user_id(user_id),
  KEY index_of_product_id(product_id)
) COMMENT = 'Скидки';

DROP TABLE IF EXISTS storehouses;
CREATE TABLE storehouses (
  id SERIAL PRIMARY KEY,
  name VARCHAR(255) COMMENT 'Название',
```

```sql
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
) COMMENT = 'Склады';

DROP TABLE IF EXISTS storehouses_products;
CREATE TABLE storehouses_products (
  id SERIAL PRIMARY KEY,
  storehouse_id INT UNSIGNED,
  product_id INT UNSIGNED,
  value INT UNSIGNED COMMENT 'Запас товарной позиции на складе',
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
) COMMENT = 'Запасы на складе';

DROP TABLE IF EXISTS accounts;
CREATE TABLE accounts (
  id SERIAL PRIMARY KEY,
  user_id INT,
  total DECIMAL (11,2) COMMENT 'Счет',
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
) COMMENT = 'Счета пользователей и интернет магазина';

INSERT INTO accounts (user_id, total) VALUES
  (4, 5000.00),
  (3, 0.00),
  (2, 200.00),
  (NULL, 25000.00);


/*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.*/

USE sample;
USE lesson9;
SELECT * FROM users;


DROP TABLE IF EXISTS users;
CREATE TABLE users (
  id SERIAL PRIMARY KEY,
  name VARCHAR(255) COMMENT 'Имя покупателя',
  birthday_at DATE COMMENT 'Дата рождения',
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
) COMMENT = 'Покупатели';
```

```sql
START TRANSACTION;
INSERT INTO sample.users SELECT * FROM lesson9.users WHERE id = 1;
COMMIT;



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

SELECT * FROM products;
SELECT * FROM catalogs;
SELECT * FROM names;

DROP VIEW IF EXISTS names;
CREATE VIEW names (cat_name, prod_name) AS
 SELECT catalogs.name, products.name FROM products, catalogs WHERE
products.catalog_id = catalogs.id;

DROP VIEW IF EXISTS names;
CREATE VIEW names (cat_name, prod_name) AS
 SELECT catalogs.name, products.name FROM products
 LEFT JOIN catalogs ON products.catalog_id = catalogs.id;

/* (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.*/

DROP TABLE IF EXISTS august;
CREATE TABLE august (created_at DATE);

INSERT INTO august VALUES
('2018-08-01'), ('2016-08-04'),
('2018-08-16'), ('2018-08-17');

CREATE VIEW digit AS
     (
     SELECT 0 AS digit FROM dual UNION ALL
     SELECT 1 FROM DUAL UNION ALL
     SELECT 2 FROM DUAL UNION ALL
     SELECT 3 FROM DUAL UNION ALL
     SELECT 4 FROM DUAL UNION ALL
     SELECT 5 FROM DUAL UNION ALL
     SELECT 6 FROM DUAL UNION ALL
     SELECT 7 FROM DUAL UNION ALL
     SELECT 8 FROM DUAL UNION ALL
```

```
     SELECT 9 FROM DUAL
     );

SELECT
     ten_pos.digit * 10 + unit_pos.digit AS day_num
     ,IFNULL((
          SELECT 1
          FROM august a
          WHERE     a.created_at = DATE_ADD("2018-08-01", INTERVAL
(ten_pos.digit * 10 + unit_pos.digit - 1) DAY)
      ), 0) AS is_exist
FROM digit AS unit_pos
     JOIN digit AS ten_pos
WHERE ten_pos.digit * 10 + unit_pos.digit BETWEEN 1 AND 31;

/* (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.*/

DROP TABLE IF EXISTS del;
CREATE TABLE del SELECT * FROM vk.users;
SELECT * FROM del;
-- 1

CREATE OR REPLACE VIEW temp AS
SELECT del.*
FROM del
ORDER BY del.created_at DESC
LIMIT 18446744073709551615 offset 5;

DELETE
FROM del
WHERE EXISTS (SELECT 1 FROM temp WHERE temp.id = del.id);

-- 2
START TRANSACTION;

CREATE TEMPORARY TABLE temp (id BIGINT, created_at DATETIME);
TRUNCATE TABLE temp;

INSERT INTO temp
SELECT del.*
FROM del
ORDER BY del.created_at DESC
LIMIT 5;

TRUNCATE TABLE del;

INSERT INTO del
SELECT * FROM temp;
```

```sql
DROP TEMPORARY TABLE temp;

COMMIT;

-- Практическое задание по теме “Администрирование MySQL” (эта тема изучается по вашему желанию)

/* 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. */

DROP USER IF EXISTS shop_read;
CREATE USER shop_read;
GRANT SELECT ON lesson9.* TO shop_read;

DROP USER IF EXISTS shop;
CREATE USER shop;
GRANT ALL ON lesson9.* TO shop;

/* (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 USER IF EXISTS user_read;
DROP VIEW IF EXISTS username;
DROP TABLE IF EXISTS accounts;

CREATE TABLE accounts (
     id serial PRIMARY KEY,
    name VARCHAR(100) UNIQUE,
    password VARCHAR(100)
);

CREATE VIEW username AS SELECT id, name FROM accounts;

CREATE USER user_read;

GRANT SELECT ON lesson9.username to user_read;


-- Практическое задание по теме “Хранимые процедуры и функции, триггеры
```

```
/* 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". */

DROP FUNCTION IF EXISTS hello;

DELIMITER //

CREATE FUNCTION hello() RETURNS VARCHAR(255) DETERMINISTIC
BEGIN
 DECLARE times INT;
 SET times = CURTIME();
 CASE
     WHEN times BETWEEN 60001 AND 120000 THEN
          RETURN "Доброе утро";
     WHEN times BETWEEN 120001 AND 180000 THEN
          RETURN "Добрый день";
     WHEN times BETWEEN 180001 AND 000000 THEN
          RETURN "Добрый вечер";
     ELSE
          RETURN "Доброй ночи";
       END CASE;

END//

DELIMITER ;

SELECT hello();

/* 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.*/

SELECT * FROM products;

DROP TRIGGER IF EXISTS insert_null;

DELIMITER //

CREATE TRIGGER insert_null BEFORE INSERT ON products
    FOR EACH ROW
BEGIN
    IF NEW.name IS NULL AND NEW.description IS NULL THEN
```

```
        SIGNAL SQLSTATE '45000' SET ERROR = 'оба поля принимают
неопределенное значение NULL';
    END IF;
END//

DELIMITER ;

DROP TRIGGER IF EXISTS insert_update_null;

DELIMITER //

CREATE TRIGGER insert_update_null BEFORE UPDATE ON products
    FOR EACH ROW
BEGIN
    IF (NEW.name IS NULL AND NEW.description IS NULL)
        OR (OLD.name IS NULL AND NEW.description IS NULL)
        OR (OLD.description IS NULL AND NEW.name IS NULL)
    THEN
        SIGNAL SQLSTATE '45000' SET ERROR = 'оба поля принимают
неопределенное значение NULL';
    END IF;
END//

DELIMITER ;

/* (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.
0 1 2 3 4 5 6 7  8  9  10
0 1 1 2 3 5 8 13 21 34 55
*/

DROP FUNCTION IF EXISTS FIBONACCI;

DELIMITER //

CREATE FUNCTION FIBONACCI(num int)
RETURNS int DETERMINISTIC
BEGIN
     DECLARE result_2 int DEFAULT 0;
     DECLARE result_1 int DEFAULT 1;
     DECLARE result_S int DEFAULT 1;
     DECLARE i int DEFAULT 1;
     IF num < 2 THEN
          IF num = 0 THEN SET result_S = 0;
          END IF;
          IF num = 1 THEN SET result_S = 1;
          END IF;
     ELSE
          WHILE i < num DO
```

```
            SET result_S = result_2 + result_1;
            SET result_2 = result_1;
            SET result_1 = result_S;
            SET i = i + 1;
        END WHILE;
    END IF;
    RETURN result_S;
END //

DELIMITER ;

SELECT FIBONACCI(9);
```