МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА - Российский технологический университет»
РТУ МИРЭА

Практическое занятие № 5

Тема занятия: 1) Администрирование в СУБД PostgreSQL. Табличные пространства
2) Администрирование PostgreSQL - управление ролями и привилегиями
3) Наследование прав и предопределенные роли

Выполнил студент 3 курса группы
ЭФБО-10-23 Ефремов А.И.
Проверил доцент Бочаров М.И.

Москва
2025 г.

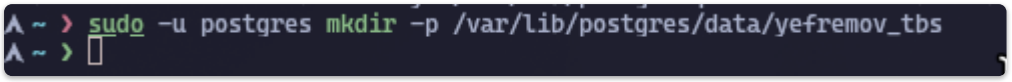
Ефремов ЭФБО-10-23 Практическая 5

Практика 5.1

Часть 1. Содержимое табличного пространства

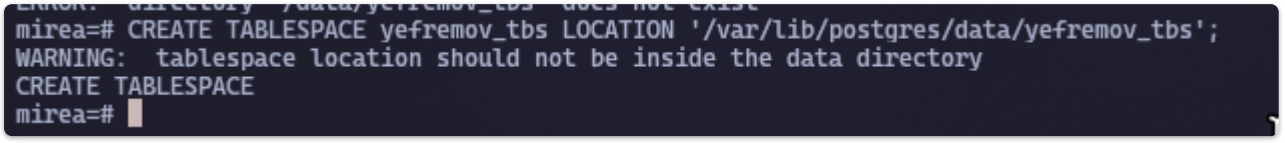
1. Создайте директорию test

```
sudo -u postgres mkdir -p /var/lib/postgres/data/yefremov_tbs
```



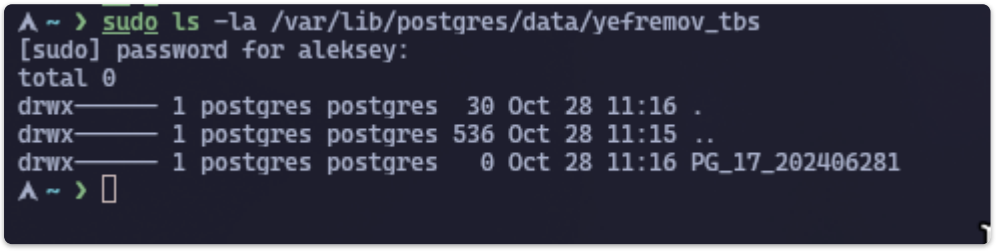
2. Создайте табличное пространство: CREATE TABLESPACE u01tbs LOCATION 'd:\test';

```
CREATE TABLESPACE yefremov_tbs LOCATION
'/var/lib/postgres/data/yefremov_tbs';
```



3. Посмотрите содержимое директории табличного пространства

```
ls -la /var/lib/postgres/data/yefremov_tbs
```



> *Была создана поддиректория с названием PG_17_202406281. В имени поддиректории присутствует номер основной версии СУБД. Такие директории создаются и удаляются автоматически, чтобы упростить обновление программного обеспечения на новую основную версию.*

4. Создайте в табличном пространстве таблицу:

```
CREATE TABLE tb_yefremov (id bigserial, t text) TABLESPACE
yefremov_tbs;
```

```
mirea=# CREATE TABLE tb_yefremov (id bigserial, t text) TABLESPACE yefremov_tbs;
CREATE TABLE
mirea=#
```

5. Заполните таблицу данными:

```
INSERT INTO tb_yefremov(t) SELECT
encode((floor(random()*1000)::numeric ^ 100::numeric)::text::bytea,
'base64') from generate_series(1,5000000);
```

```
mirea=# INSERT INTO tb_yefremov(t) SELECT encode((floor(random()*1000)::numeric ^ 100::numeric)::text::bytea, 'base64') from
generate_series(1,5000000);
INSERT 0 5000000
```

6. Посмотрим, какие файлы появились

```
sudo ls -la
/var/lib/postgres/data/yefremov_tbs/PG_17_202406281/16384
```

```
~ ❯ sudo ls -la /var/lib/postgres/data/yefremov_tbs/PG_17_202406281/16384
total 1947624
drwx------ 1 postgres postgres         78 Oct 28 11:26 .
drwx------ 1 postgres postgres         10 Oct 28 11:23 ..
-rw------- 1 postgres postgres 1073741824 Oct 28 11:26 16618
-rw------- 1 postgres postgres  920043520 Oct 28 11:27 16618.1
-rw------- 1 postgres postgres     507904 Oct 28 11:25 16618_fsm
-rw------- 1 postgres postgres      65536 Oct 28 11:27 16618_vm
-rw------- 1 postgres postgres          0 Oct 28 11:23 16622
-rw------- 1 postgres postgres       8192 Oct 28 11:23 16623
~ ❯
```

> *Файл с суффиксом ".1". Это второй файл основного слоя (main fork).*

7. Вставьте ещё миллион строк

```
INSERT INTO tb_yefremov(t) SELECT
encode((floor(random()*1000)::numeric ^ 100::numeric)::text::bytea,
'base64') from generate_series(1,1000000);
```

```
mirea=# INSERT INTO tb_yefremov(t) SELECT encode((floor(random()*1000)::numeric ^ 100::numeric)::text::bytea, 'base64') from
generate_series(1,1000000);
INSERT 0 1000000
```

```
sudo ls -la
/var/lib/postgres/data/yefremov_tbs/PG_17_202406281/16384
```

```
~ > sudo ls -la /var/lib/postgres/data/yefremov_tbs/PG_17_202406281/16384
total 2337104
drwx------ 1 postgres postgres         92 Oct 28 11:29 .
drwx------ 1 postgres postgres         10 Oct 28 11:23 ..
-rw------- 1 postgres postgres 1073741824 Oct 28 11:30 16618
-rw------- 1 postgres postgres 1073741824 Oct 28 11:30 16618.1
-rw------- 1 postgres postgres  245030912 Oct 28 11:30 16618.2
-rw------- 1 postgres postgres     606208 Oct 28 11:29 16618_fsm
-rw------- 1 postgres postgres      65536 Oct 28 11:29 16618_vm
-rw------- 1 postgres postgres          0 Oct 28 11:23 16622
-rw------- 1 postgres postgres       8192 Oct 28 11:23 16623
~ >
```

8. Выполните удаление табличного пространства: drop tablespace u01tbs; Сделайте выводы.

```
DROP TABLESPACE yefremov_tbs;
```

```
mirea=# DROP TABLESPACE yefremov_tbs;
ERROR:  tablespace "yefremov_tbs" is not empty
mirea=#
```

Вывод: табличное пространство нельзя удалить, если в нем есть данные.

9. Посмотрите утилитой oid2name информацию о таблице: oid2name -t tb

```
oid2name -t tb_yefremov
```

```
~ > sudo -u postgres oid2name -t tb_yefremov
[sudo] password for aleksey:
From database "postgres":
  Filenode  Table Name
----------------------
~ >
```

10. Посмотрите, что выдаёт утилита про оставшиеся файлы: oid2name -f 83705

```
oid2name -f 16622
```

```
~ > sudo -u postgres oid2name -f 16622
From database "postgres":
  Filenode  Table Name
----------------------
~ >
```

Часть 2. Файл объекта «последовательность»

1. Посмотрите определение таблицы: \d tb

```
\d tb_yefremov
```

```
mirea=# \d tb_yefremov
                        Table "public.tb_yefremov"
 Column |  Type  | Collation | Nullable |                 Default
--------+--------+-----------+----------+-----------------------------------------
 id     | bigint |           | not null | nextval('tb_yefremov_id_seq'::regclass)
 t      | text   |           |          |
Tablespace: "yefremov_tbs"
```

2. Посмотрите определение последовательности: \ds+

```
\ds+ tb_yefremov_id_seq
```

```
mirea=# \ds+ tb_yefremov_id_seq
                                    List of relations
 Schema |        Name        |   Type   |  Owner   | Persistence |    Size    | Description
--------+--------------------+----------+----------+-------------+------------+-------------
 public | tb_yefremov_id_seq | sequence | postgres | permanent   | 8192 bytes |
(1 row)

mirea=#
```

3. Посмотрите характеристики последовательности как «объекта»: select * from pg_class where relname='tb_id_seq' \gx

```
SELECT oid, relname, relfilenode, reltablespace, relpages,
reltuples
FROM pg_class
WHERE relname = 'tb_yefremov_id_seq' \gx
```

```
mirea=# SELECT oid, relname, relfilenode, reltablespace, relpages, reltup
FROM pg_class
WHERE relname = 'tb_yefremov_id_seq' \gx
-[ RECORD 1 ]-+-------------------
oid           | 16617
relname       | tb_yefremov_id_seq
relfilenode   | 16617
reltablespace | 0
relpages      | 1
reltuples     | 1
```

4. Посмотрите путь к файлу последовательности: SELECT pg_relation_filepath(relfilenode);

```
SELECT pg_relation_filepath('tb_yefremov_id_seq');
```

```
mirea=# SELECT pg_relation_filepath('tb_yefremov_id_seq');
 pg_relation_filepath
----------------------
 base/16384/16617
(1 row)
```

Файл последовательности был создан и располагается в табличном пространстве pg_default, которое является табличным пространством по умолчанию для базы данных postgres:

```
SELECT dattablespace, datname FROM pg_database WHERE datname =
current_database();
SELECT oid, spcname FROM pg_tablespace WHERE oid = 1663;
```

```
mirea=# SELECT dattablespace, datname FROM pg_database WHERE datname = current_database();
SELECT oid, spcname FROM pg_tablespace WHERE oid = 1663;
 dattablespace | datname
---------------+---------
          1663 | mirea
(1 row)

 oid  |  spcname
------+------------
 1663 | pg_default
(1 row)
```

Часть 3. Перемещение таблицы в другое табличное пространство

1. Переместим таблицу t в табличное пространство pg_default.

```
ALTER TABLE tb_yefremov SET TABLESPACE pg_default;
```

```
mirea=# ALTER TABLE tb_yefremov SET TABLESPACE pg_default;
ALTER TABLE
mirea=#
```

2. В окне psql в целях оценки сколько журнальных данных сгенерируется посмотрим текущий LSN:

```
SELECT pg_current_wal_lsn() AS before_lsn;
```

```
mirea=# SELECT pg_current_wal_lsn() AS before_lsn;
 before_lsn
------------
 1/2968E130
(1 row)
```

3. В окне psql дайте команду перемещения. Воспользуйтесь, например, синтаксисом перемещения всех таблиц:

```
ALTER TABLE tb_yefremov SET TABLESPACE yefremov_tbs;
```

```
mirea=# ALTER TABLE tb_yefremov SET TABLESPACE yefremov_tbs;
ALTER TABLE
```

4. Посмотрите текущий LSN

```
SELECT pg_current_wal_lsn() AS after_lsn;
```

```
mirea=# SELECT pg_current_wal_lsn() AS after_lsn;
 after_lsn
------------
 1/B8B999C0
(1 row)
```

5. Посчитайте какой объем данных прошел через журналы:

```
SELECT pg_size_pretty('2/B8B999C0'::pg_lsn - '2/2968E130'::pg_lsn)
AS wal_volume;
```

```
mirea=# SELECT pg_size_pretty('2/B8B999C0'::pg_lsn - '2/2968E130'::pg_lsn) AS wal_volume;
 wal_volume
------------
 2293 MB
(1 row)
```

6. Посмотрите размер таблицы:

```
SELECT pg_size_pretty(pg_total_relation_size('tb_yefremov')) AS
table_size;
```

```
mirea=# SELECT pg_size_pretty(pg_total_relation_size('tb_yefremov')) AS table_size;
 table_size
------------
 2282 MB
(1 row)
```

Практика 5.2

Часть 1: Подготовка и создание ролей

Задание 1.1: Создание ролей

1. Подключитесь к вашей базе данных mirea под пользователем postgres (или другим суперпользователем).

```
psql -U postgres -d mirea
```

```
~ ❯ psql -U postgres -d mirea
psql (17.6)
Type "help" for help.

mirea=#
```

2. Создайте три новые роли, используя ваши индивидуальные данные:

```
CREATE ROLE yefremov WITH LOGIN PASSWORD 'secure_password_1';
CREATE ROLE aleksey WITH LOGIN PASSWORD 'secure_password_2';
CREATE ROLE igorevich WITH NOLOGIN;
```

```
mirea=# CREATE ROLE yefremov WITH LOGIN PASSWORD 'secure_password_1';
CREATE ROLE aleksey WITH LOGIN PASSWORD 'secure_password_2';
CREATE ROLE igorevich WITH NOLOGIN;
CREATE ROLE
CREATE ROLE
CREATE ROLE
mirea=#
```

3. Назначьте роли yefremov и aleksey членом групповой роли igorevich.

```
GRANT igorevich TO yefremov, aleksey;
```

```
mirea=# GRANT igorevich TO yefremov, aleksey;
GRANT ROLE
mirea=#
```

Часть 2: Базовые права доступа и управление привилегиями

Задание 2.1: Базовые права доступа

1. Предоставьте групповой роли igorevich базовые права на подключение к БД и использование схемы.

```
-- Разрешение на подключение к БД
GRANT CONNECT ON DATABASE mirea TO igorevich;

-- Разрешение на использование схемы 'public' (по умолчанию)
GRANT USAGE ON SCHEMA public TO igorevich;
```

```
mirea=# -- Разрешение на подключение к БД
GRANT CONNECT ON DATABASE mirea TO igorevich;

-- Разрешение на использование схемы 'public' (по умолчанию)
GRANT USAGE ON SCHEMA public TO igorevich;
GRANT
GRANT
mirea=#
```

Задание 2.2: Права на чтение данных

2. Назначьте роли yefremov права на чтение данных (SELECT) из ВСЕХ таблиц базы данных. Сделайте это, выдав право групповой роли igorevich.

```
-- Выдача права SELECT на все таблицы
GRANT SELECT ON ALL TABLES IN SCHEMA public TO igorevich;
```

```
mirea=# -- Выдача права SELECT на все таблицы
GRANT SELECT ON ALL TABLES IN SCHEMA public TO igorevich;
GRANT
mirea=#
```

Задание 2.3: Расширенные права для aleksey

3. Назначьте роли aleksey более широкие права. Она должна иметь право не только читать данные, но и добавлять новые записи (INSERT) и обновлять существующие (UPDATE) в таблицах property, evaluation и sale. Выдайте эти права напрямую роли aleksey.

```
-- Выдача прав на конкретные таблицы напрямую пользователю
GRANT INSERT, UPDATE ON property, evaluation, sale TO aleksey;
```

```
mirea=# -- Выдача прав на конкретные таблицы напрямую пользователю
GRANT INSERT, UPDATE ON property, evaluation, sale TO aleksey;
GRANT
mirea=#
```

Задание 2.4: Запрет на удаление данных

4. Убедитесь, что ни у кого из новых пользователей нет права удалять данные (DELETE) из таблиц. Отзовите это право, если оно есть по умолчанию.

```
-- Отзыв права DELETE у роли PUBLIC (всех пользователей) и у
групповой роли
REVOKE DELETE ON ALL TABLES IN SCHEMA public FROM PUBLIC;
REVOKE DELETE ON ALL TABLES IN SCHEMA public FROM igorevich;
```

```
mirea=# -- Отзыв права DELETE у роли PUBLIC (всех пользователей) и у групповой роли
REVOKE DELETE ON ALL TABLES IN SCHEMA public FROM PUBLIC;
REVOKE DELETE ON ALL TABLES IN SCHEMA public FROM igorevich;
REVOKE
REVOKE
mirea=#
```

Часть 3: Проверка базовых прав доступа

Задание 3.1: Тестирование прав пользователя yefremov

1. Подключитесь к БД mirea под пользователем yefremov (в отдельной вкладке psql или другом клиенте).

```
psql -U yefremov -d mirea
```

```
~ ❯ psql -U yefremov -d mirea
psql (17.6)
Type "help" for help.

mirea=>
```

```sql
-- Тест 1: Чтение данных (должно выполниться успешно)
SELECT * FROM property LIMIT 5;
```

```
mirea=> -- Тест 1: Чтение данных (должно выполниться успешно)
SELECT * FROM property LIMIT 5;
 id | district_id |      address      | floor | room_count | type_id | status |  price   |          description          | b
uilding_material_id | area | listing_date | test_column
----+-------------+-------------------+-------+------------+---------+--------+----------+-------------------------------+--
--------------------+------+--------------+-------------
  1 |           1 | ул. Ленина, 10    |     3 |          2 |       1 | t      |  5000000 | Квартира в центре города      |
                  2 |   45 | 2023-01-15   |
  2 |           1 | ул. Советская, 25 |     5 |          3 |       1 | t      |  7500000 | Просторная квартира с ремонтом |
                  2 |   65 | 2023-02-10   |
  3 |           2 | ул. Северная, 5   |     2 |          1 |       1 | f      |  3000000 | Квартира в спальном районе    |
                  1 |   35 | 2023-03-05   |
  4 |           2 | ул. Лесная, 15    |     1 |          2 |       3 | t      |  6000000 | Апартаменты с мебелью         |
                  2 |   50 | 2023-01-20   |
  5 |           3 | ул. Южная, 30     |     4 |          4 |       2 | t      | 12000000 | Частный дом с участком        |
                  2 |  120 | 2023-02-25   |
(5 rows)

mirea=> 2
```

```sql
-- Тест 2: Добавление новой записи (должна возникнуть ошибка)
INSERT INTO property (district_id, address, floor, room_count,
type_id, status, price, description, building_material_id, area,
listing_date)
VALUES (1, 'ул. Тестовая, 1', 1, 2, 1, true, 4000000, 'Тестовая
запись', 1, 40, '2025-11-06');
```

```
mirea=> -- Тест 2: Добавление новой записи (должна возникнуть ошибка)
INSERT INTO property (district_id, address, floor, room_count, type_id, status, price, description, building_material_id, are
a, listing_date)
VALUES (1, 'ул. Тестовая, 1', 1, 2, 1, true, 4000000, 'Тестовая запись', 1, 40, '2025-11-06');
ERROR:  permission denied for table property
mirea=>
```

```sql
-- Тест 3: Удаление записи (должна возникнуть ошибка)
DELETE FROM property WHERE id = 1;
```

```
mirea=> -- Тест 3: Удаление записи (должна возникнуть ошибка)
DELETE FROM property WHERE id = 1;
ERROR:  permission denied for table property
mirea=>
```

Задание 3.2: Тестирование прав пользователя aleksey

2. Подключитесь к БД mirea под пользователем aleksey.

```sql
psql -U aleksey -d mirea
```

```
~ ❯ psql -U aleksey -d mirea
psql (17.6)
Type "help" for help.

mirea=>
```

```sql
-- Тест 1: Чтение данных (должно выполниться успешно)
SELECT * FROM sale LIMIT 5;
```

```
mirea=> -- Тест 1: Чтение данных (должно выполниться успешно)
SELECT * FROM sale LIMIT 5;
 id | property_id | sale_date  | realtor_id |  price  | realtor_commission
----+-------------+------------+------------+---------+--------------------
 21 |          18 | 2023-05-20 |          1 | 4500000 |
 22 |          21 | 2023-05-20 |          2 | 5500000 |
 25 |          30 | 2020-01-01 |          1 | 5000000 |
 31 |          23 | 2025-10-01 |          1 | 1000000 |
 33 |          28 | 2025-10-01 |          1 | 1000000 |
(5 rows)

mirea=>
```

```sql
-- Тест 2: Обновление стоимости объекта (должно выполниться успешно)
UPDATE property SET price = 6000000 WHERE id = 3;
```

```
mirea=> -- Тест 2: Обновление стоимости объекта (должно выполниться успешно)
UPDATE property SET price = 6000000 WHERE id = 3;
UPDATE 1
mirea=>
```

```sql
-- Тест 3: Вставка новой оценки (должно выполниться успешно)
INSERT INTO evaluation (property_id, evaluation_date,
criterion_id, score)
VALUES (2, '2025-11-06', 1, 9.1);
```

```
mirea=> -- Тест 3: Вставка новой оценки (должно выполниться успешно)
INSERT INTO evaluation (property_id, evaluation_date, criterion_id, score)
VALUES (2, '2025-11-06', 1, 9.1);
INSERT 0 1
mirea=>
```

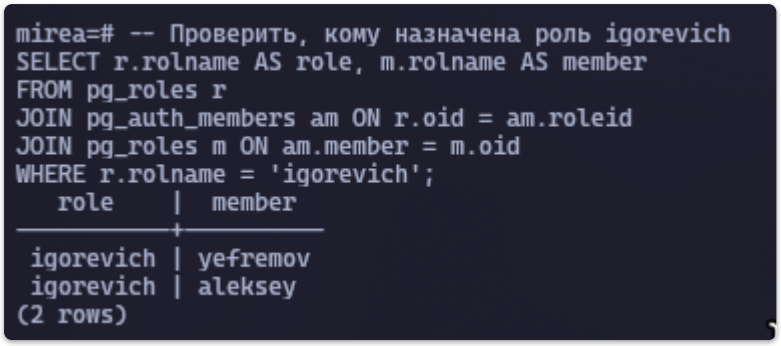
```sql
-- Тест 4: Удаление риэлтора (должна возникнуть ошибка)
DELETE FROM realtor WHERE id = 5;
```

```
mirea=> -- Тест 4: Удаление риэлтора (должна возникнуть ошибка)
DELETE FROM realtor WHERE id = 5;
ERROR:  permission denied for table realtor
mirea=>
```

Задание 3.3: Проверка членства в ролях

3. Вернитесь в сеанс под пользователем postgres и проверьте членство в ролях:

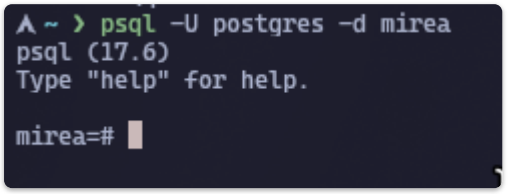
```sql
-- Проверить, кому назначена роль igorevich
SELECT r.rolname AS role, m.rolname AS member
FROM pg_roles r
JOIN pg_auth_members am ON r.oid = am.roleid
JOIN pg_roles m ON am.member = m.oid
WHERE r.rolname = 'igorevich';
```



Часть 4: Ограничение доступа к столбцам (Column-Level Security)

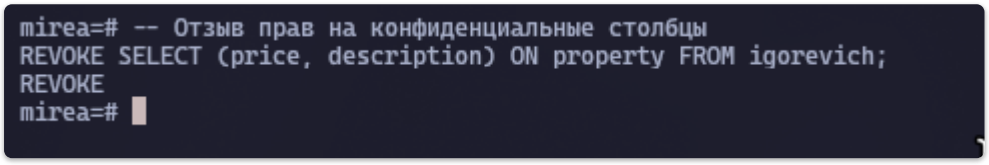
Задание 4.1: Защита конфиденциальных данных

1. Создайте политику конфиденциальности: столбцы `price` (стоимость) и `description` (описание) в таблице `property` являются конфиденциальными.



2. Отзовите право на просмотр этих столбцов у групповой роли `igorevich`:

```sql
-- Отзыв прав на конфиденциальные столбцы
REVOKE SELECT (price, description) ON property FROM igorevich;
```



3. Предоставьте право на просмотр этих столбцов только роли `aleksey`:

```sql
-- Выдача прав на конфиденциальные столбцы конкретному
пользователю
GRANT SELECT (price, description) ON property TO aleksey;
```

```
mirea=# -- Выдача прав на конфиденциальные столбцы конкретному пользователю
GRANT SELECT (price, description) ON property TO aleksey;
GRANT
mirea=#
```

Задание 4.2: Проверка ограничений на уровне столбцов

4. Проверка работы ограничений:

Под пользователем `yefremov`

```
psql -U yefremov -d mirea
```

```
~ ❯ psql -U yefremov -d mirea
psql (17.6)
Type "help" for help.

mirea=>
```

```
SELECT id, address, price FROM property LIMIT 3;
```

```
mirea=> SELECT id, address, price FROM property LIMIT 3;
ERROR:  permission denied for table property
mirea=>
```

Под пользователем `aleksey`

```
psql -U aleksey -d mirea
```

```
~ ❯ psql -U aleksey -d mirea
psql (17.6)
Type "help" for help.

mirea=>
```

```
SELECT id, address, price FROM property LIMIT 3;
```

```
mirea=> SELECT id, address, price FROM property LIMIT 3;
 id |      address       |  price
----+--------------------+---------
  1 | ул. Ленина, 10     | 5000000
  2 | ул. Советская, 25  | 7500000
  4 | ул. Лесная, 15     | 6000000
(3 rows)
```

Часть 5: Ограничение доступа к строкам (Row-Level Security)

Задание 5.1: Включение RLS

1. Включите RLS для таблицы `property`:

```sql
-- От суперпользователя (postgres)
ALTER TABLE property ENABLE ROW LEVEL SECURITY;
```

```
mirea=# -- От суперпользователя (postgres)
ALTER TABLE property ENABLE ROW LEVEL SECURITY;
ALTER TABLE
mirea=#
```

Задание 5.2: Создание политик доступа

2. Создайте политику для роли `yefremov`, которая позволяет ему видеть только объекты недвижимости, расположенные в "Центральном" районе.

```sql
-- Только Центральный район (district_id = 1)
CREATE POLICY yefremov_policy ON property
    FOR ALL
    TO yefremov
    USING (district_id = 1);
```

```
mirea=# -- Только Центральный район (district_id = 1)
CREATE POLICY yefremov_policy ON property
    FOR ALL
    TO yefremov
    USING (district_id = 1);
CREATE POLICY
mirea=#
```

3. Создайте политику для роли `aleksey`, которая позволяет ему видеть только активные объявления.

```sql
-- Только активные объявления
CREATE POLICY aleksey_policy ON property
    FOR ALL
    TO aleksey
    USING (status = true);
```

```
mirea=# -- Только активные объявления
CREATE POLICY aleksey_policy ON property
    FOR ALL
    TO aleksey
    USING (status = true);
CREATE POLICY
mirea=#
```

4. Создайте политику по умолчанию для администраторов:

```sql
-- Администратор видит всё
CREATE POLICY admin_policy ON property
```

```
    FOR ALL
    TO postgres
    USING (true);
```

```
mirea=# -- Администратор видит всё
CREATE POLICY admin_policy ON property
    FOR ALL
    TO postgres
    USING (true);
CREATE POLICY
mirea=#
```

5. Создайте самостоятельно какую-либо политику для группы `igorevich`.

```
-- Группа igorevich видит только объекты дешевле 10 млн
CREATE POLICY igorevich_price_policy ON property
    FOR ALL
    TO igorevich
    USING (price < 10000000);
```

```
mirea=# -- Группа igorevich видит только объекты дешевле 10 млн
CREATE POLICY igorevich_price_policy ON property
    FOR ALL
    TO igorevich
    USING (price < 10000000);
CREATE POLICY
mirea=#
```

Задание 5.3: Проверка RLS

Под пользователем `yefremov`:

```
psql -U yefremov -d mirea
```

```
~ ❯ psql -U yefremov -d mirea
psql (17.6)
Type "help" for help.

mirea=>
```

```
SELECT id, address, district_id, price FROM property;
```

```
mirea=> SELECT id, address, district_id, price FROM property;
 id |       address        | district_id |  price
----+----------------------+-------------+---------
  1 | ул. Ленина, 10       |           1 | 5000000
  2 | ул. Советская, 25    |           1 | 7500000
  9 | пр. Мира, 20         |           1 | 8000000
 13 | ул. Центральная, 11  |           1 | 5200000
 17 | пр. Победы, 17       |           1 | 9000000
 21 | ул. Новая, 21        |           1 | 7200000
 25 | ул. Торговая, 26     |           1 | 7800000
 29 | ул. Береговая, 28    |           1 | 8500000
(8 rows)

mirea=>
```

Под пользователем `aleksey`:

```
psql -U aleksey -d mirea
```

```
mirea=> \q
➜ ~ ❯ psql -U aleksey -d mirea
psql (17.6)
Type "help" for help.

mirea=>
```

```
SELECT id, address, status, price FROM property;
```

```
mirea=> SELECT id, address, status, price FROM property;
 id |       address        | status |  price
----+----------------------+--------+----------
  1 | ул. Ленина, 10       | t      |  5000000
  2 | ул. Советская, 25    | t      |  7500000
  4 | ул. Лесная, 15       | t      |  6000000
  5 | ул. Южная, 30        | t      | 12000000
  6 | ул. Садовая, 8       | t      |  5500000
  8 | ул. Речная, 7        | t      |  4000000
  9 | пр. Мира, 20         | t      |  8000000
 10 | ул. Сосновая, 3      | t      |  4500000
 12 | ул. Горная, 22       | t      |  6500000
 13 | ул. Центральная, 11  | t      |  5200000
 15 | ул. Дачная, 14       | t      |  8500000
 16 | ул. Водная, 6        | t      |  5000000
 17 | пр. Победы, 17       | t      |  9000000
 19 | ул. Солнечная, 19    | t      | 11000000
 20 | ул. Луговая, 13      | t      |  5800000
 21 | ул. Новая, 21        | t      |  7200000
 23 | ул. Зелёная, 16      | t      | 10500000
 24 | ул. Каменная, 23     | t      |  5100000
 25 | ул. Торговая, 26     | t      |  7800000
 27 | ул. Парковая, 24     | t      |  9800000
 28 | ул. Моховая, 27      | t      |  5400000
 29 | ул. Береговая, 28    | t      |  8500000
(22 rows)

mirea=>
```

Под пользователем `postgres`:

```
SELECT id, address, district_id, status, price FROM property;
```

```
mirea=# SELECT id, address, district_id, status, price FROM property;
 id |       address        | district_id | status |  price
----+----------------------+-------------+--------+----------
  1 | ул. Ленина, 10       |           1 | t      |  5000000
  2 | ул. Советская, 25    |           1 | t      |  7500000
  4 | ул. Лесная, 15       |           2 | t      |  6000000
  5 | ул. Южная, 30        |           3 | t      | 12000000
  6 | ул. Садовая, 8       |           3 | t      |  5500000
  7 | ул. Западная, 12     |           4 | f      |  4800000
  8 | ул. Речная, 7        |           4 | t      |  4000000
  9 | пр. Мира, 20         |           1 | t      |  8000000
 10 | ул. Сосновая, 3      |           2 | t      |  4500000
 11 | ул. Полевая, 18      |           3 | f      |  9500000
 12 | ул. Горная, 22       |           4 | t      |  6500000
 13 | ул. Центральная, 11  |           1 | t      |  5200000
 14 | ул. Берёзовая, 9     |           2 | f      |  3200000
 15 | ул. Дачная, 14       |           3 | t      |  8500000
 16 | ул. Водная, 6        |           4 | t      |  5000000
 17 | пр. Победы, 17       |           1 | t      |  9000000
 18 | ул. Снежная, 4       |           2 | f      |  4700000
 19 | ул. Солнечная, 19    |           3 | t      | 11000000
 20 | ул. Луговая, 13      |           4 | t      |  5800000
 21 | ул. Новая, 21        |           1 | t      |  7200000
 22 | ул. Старая, 2        |           2 | f      |  2900000
 23 | ул. Зелёная, 16      |           3 | t      | 10500000
 24 | ул. Каменная, 23     |           4 | t      |  5100000
 25 | ул. Торговая, 26     |           1 | t      |  7800000
 26 | ул. Школьная, 1      |           2 | f      |  4600000
 27 | ул. Парковая, 24     |           3 | t      |  9800000
 28 | ул. Моховая, 27      |           4 | t      |  5400000
 29 | ул. Береговая, 28    |           1 | t      |  8500000
 30 | ул. Хвойная, 29      |           2 | f      |  3100000
  3 | ул. Северная, 5      |           2 | f      |  6000000
(30 rows)

mirea=#
```

Практика 5.3

Часть 6: Наследование прав и каскадное управление привилегиями

Задание 6.1: Создание иерархии ролей

```
CREATE ROLE managers WITH NOLOGIN;
GRANT igorevich TO managers;
GRANT managers TO aleksey;
```

```
mirea=# CREATE ROLE managers WITH NOLOGIN;
GRANT igorevich TO managers;
GRANT managers TO aleksey;
CREATE ROLE
GRANT ROLE
GRANT ROLE
mirea=#
```

Задание 6.2: Наследование привилегий через иерархию

```
GRANT INSERT, UPDATE, DELETE ON realtor, sale TO managers;
```

```
mirea=# GRANT INSERT, UPDATE, DELETE ON realtor, sale TO managers;
GRANT
```

```sql
-- Проверка цепочки наследования
SELECT r.rolname AS user_role, m.rolname AS member_of, rm.rolname
AS inherited_from
FROM pg_roles r
JOIN pg_auth_members am ON r.oid = am.member
JOIN pg_roles m ON am.roleid = m.oid
LEFT JOIN pg_auth_members am2 ON m.oid = am2.member
LEFT JOIN pg_roles rm ON am2.roleid = rm.oid
WHERE r.rolname = 'aleksey';
```

```
mirea=# -- Проверка цепочки наследования
SELECT r.rolname AS user_role, m.rolname AS member_of, rm.rolname AS inherited_from
FROM pg_roles r
JOIN pg_auth_members am ON r.oid = am.member
JOIN pg_roles m ON am.roleid = m.oid
LEFT JOIN pg_auth_members am2 ON m.oid = am2.member
LEFT JOIN pg_roles rm ON am2.roleid = rm.oid
WHERE r.rolname = 'aleksey';
 user_role | member_of | inherited_from
-----------+-----------+----------------
 aleksey   | managers  | igorevich
(1 row)

mirea=#
```

Задание 6.3: Тестирование каскадного наследования

```sql
psql -U aleksey -d mirea
```

```
~ ❯ psql -U aleksey -d mirea
psql (17.6)
Type "help" for help.

mirea=>
```

```sql
SELECT * FROM district LIMIT 3;
```

```
mirea=> SELECT * FROM district LIMIT 3;
 id |    name
----+------------
  1 | Центральный
  2 | Северный
  3 | Южный
(3 rows)
```

```sql
INSERT INTO realtor (last_name, first_name, middle_name,
phone_number)
VALUES ('Тестов', 'Тест', 'Тестович', '+79160000000');
```

```
mirea=> INSERT INTO realtor (last_name, first_name, middle_name, phone_number)
VALUES ('Тестов', 'Тест', 'Тестович', '+79160000000');
INSERT 0 1
mirea=>
```

```
DELETE FROM realtor WHERE last_name = 'Тестов';
```

```
mirea=⇒ DELETE FROM realtor WHERE last_name = 'Тестов';
DELETE 3
mirea=⇒
```

Задание 6.4: Создание роли с ограниченным наследованием

```
CREATE ROLE viewer WITH NOINHERIT LOGIN PASSWORD 'viewer_pass';
GRANT SELECT ON property, district, type TO viewer;
GRANT igorevich TO viewer;
```

```
mirea=# CREATE ROLE viewer WITH NOINHERIT LOGIN PASSWORD 'viewer_pass';
GRANT SELECT ON property, district, type TO viewer;
GRANT igorevich TO viewer;
CREATE ROLE
GRANT
GRANT ROLE
mirea=#
```

```
psql -U viewer -d mirea
```

```
➜ ~ ❯ psql -U viewer -d mirea
psql (17.6)
Type "help" for help.

mirea=⇒
```

```
SELECT * FROM building_material;           -- ОШИБКА
SET ROLE igorevich;
SELECT * FROM building_material LIMIT 3;   -- УСПЕХ
RESET ROLE;
```

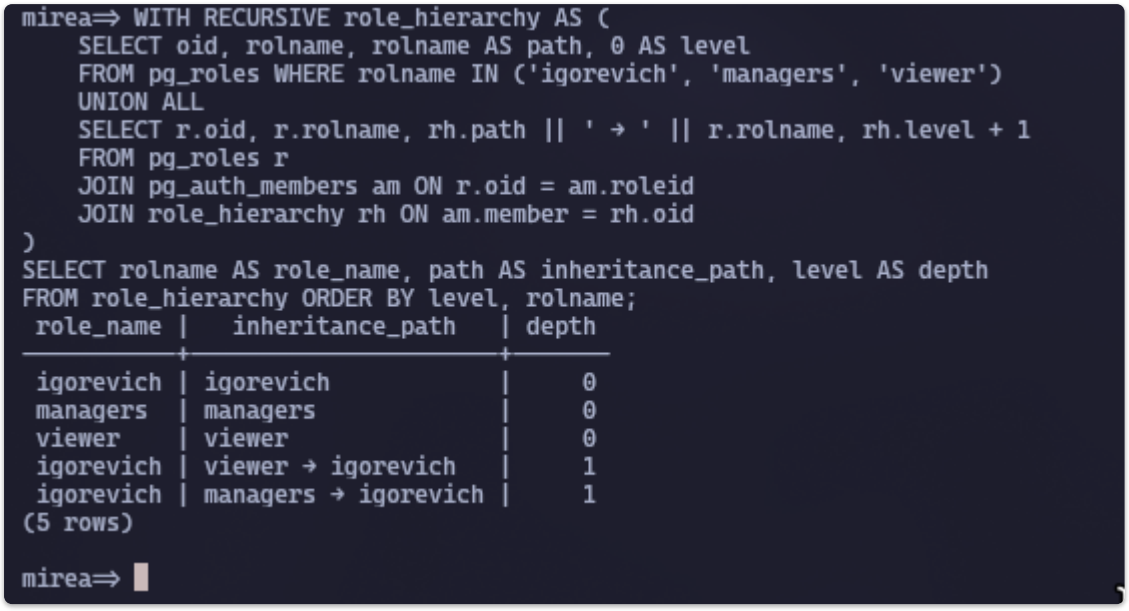
```
mirea=⇒ SELECT * FROM building_material;           -- ОШИБКА
SET ROLE igorevich;
SELECT * FROM building_material LIMIT 3;   -- УСПЕХ
RESET ROLE;
ERROR:  permission denied for table building_material
SET
 id |  name
----+--------
  1 | Панель
  2 | Кирпич
(2 rows)

RESET
mirea=⇒
```

Задание 6.5: Анализ эффективности иерархии

```
WITH RECURSIVE role_hierarchy AS (
    SELECT oid, rolname, rolname AS path, 0 AS level
    FROM pg_roles WHERE rolname IN ('igorevich', 'managers',
```

```
'viewer')
    UNION ALL
    SELECT r.oid, r.rolname, rh.path || ' → ' || r.rolname,
rh.level + 1
    FROM pg_roles r
    JOIN pg_auth_members am ON r.oid = am.roleid
    JOIN role_hierarchy rh ON am.member = rh.oid
)
SELECT rolname AS role_name, path AS inheritance_path, level AS
depth
FROM role_hierarchy ORDER BY level, rolname;
```



Задание 6.6: Практическое применение наследования

```
CREATE ROLE new_manager WITH LOGIN PASSWORD 'manager_pass123';
GRANT managers TO new_manager;

-- Проверка под new_manager
SELECT table_name, privilege_type
FROM information_schema.table_privileges
WHERE grantee = 'managers';
```

```
mirea=# CREATE ROLE new_manager WITH LOGIN PASSWORD 'manager_pass123';
GRANT managers TO new_manager;

-- Проверка под new_manager
SELECT table_name, privilege_type
FROM information_schema.table_privileges
WHERE grantee = 'managers';
CREATE ROLE
GRANT ROLE
 table_name | privilege_type
------------+----------------
 sale       | INSERT
 sale       | UPDATE
 sale       | DELETE
 realtor    | INSERT
 realtor    | UPDATE
 realtor    | DELETE
(6 rows)

mirea=#
```

Часть 7: Работа с предопределёнными ролями

Задание 7.1: Изучение встроенных ролей

```
SELECT rolname, rolsuper, rolcreaterole, rolcreatedb, rolcanlogin
FROM pg_roles WHERE rolname LIKE 'pg_%' AND rolname !=
'pg_signal_backend';
```

```
mirea=> SELECT rolname, rolsuper, rolcreaterole, rolcreatedb, rolcanlogin
FROM pg_roles WHERE rolname LIKE 'pg_%' AND rolname <> 'pg_signal_backend';
           rolname            | rolsuper | rolcreaterole | rolcreatedb | rolcanlogin
------------------------------+----------+---------------+-------------+-------------
 pg_database_owner            | f        | f             | f           | f
 pg_read_all_data             | f        | f             | f           | f
 pg_write_all_data            | f        | f             | f           | f
 pg_monitor                   | f        | f             | f           | f
 pg_read_all_settings         | f        | f             | f           | f
 pg_read_all_stats            | f        | f             | f           | f
 pg_stat_scan_tables          | f        | f             | f           | f
 pg_read_server_files         | f        | f             | f           | f
 pg_write_server_files        | f        | f             | f           | f
 pg_execute_server_program    | f        | f             | f           | f
 pg_checkpoint                | f        | f             | f           | f
 pg_maintain                  | f        | f             | f           | f
 pg_use_reserved_connections  | f        | f             | f           | f
 pg_create_subscription       | f        | f             | f           | f
(14 rows)

mirea=>
```

Задание 7.2: Назначение роли pg_read_all_data

```
GRANT pg_read_all_data TO yefremov;
```

```
mirea=# GRANT pg_read_all_data TO yefremov;
GRANT ROLE
```

```
SELECT grantee, privilege_type, table_schema, table_name
FROM information_schema.role_table_grants
WHERE grantee = 'yefremov'
```

```
ORDER BY table_name, privilege_type
LIMIT 10;
```

```
mirea=# SELECT grantee, privilege_type, table_schema, table_name
FROM information_schema.role_table_grants
WHERE grantee = 'yefremov'
ORDER BY table_name, privilege_type
LIMIT 10;
 grantee | privilege_type | table_schema | table_name
---------+----------------+--------------+------------
(0 rows)

mirea=#
```

Это правильная проверка:

```
SELECT
    has_table_privilege('yefremov', 'property', 'SELECT') AS
property_select,
    has_table_privilege('yefremov', 'sale', 'SELECT') AS
sale_select,
    has_table_privilege('yefremov', 'realtor', 'SELECT') AS
realtor_select,
    has_table_privilege('yefremov', 'evaluation', 'SELECT') AS
evaluation_select;
```

```
mirea=# SELECT
    has_table_privilege('yefremov', 'property', 'SELECT') AS property_select,
    has_table_privilege('yefremov', 'sale', 'SELECT') AS sale_select,
    has_table_privilege('yefremov', 'realtor', 'SELECT') AS realtor_select,
    has_table_privilege('yefremov', 'evaluation', 'SELECT') AS evaluation_select;
 property_select | sale_select | realtor_select | evaluation_select
-----------------+-------------+----------------+-------------------
 t               | t           | t              | t
(1 row)

mirea=#
```

Задание 7.3: Назначение роли pg_write_all_data

```
CREATE ROLE data_entry WITH LOGIN PASSWORD 'entry_pass123';
GRANT pg_write_all_data TO data_entry;
```

```
mirea=# CREATE ROLE data_entry WITH LOGIN PASSWORD 'entry_pass123';
GRANT pg_write_all_data TO data_entry;
CREATE ROLE
GRANT ROLE
mirea=#
```

```
INSERT INTO district (name) VALUES ('Новый район'); -- Должно
выполниться
UPDATE property SET price = price + 100000 WHERE id = 1; -- Должно
выполниться
```

```
mirea=> INSERT INTO district (name) VALUES ('Новый район'); -- Должно выполниться
UPDATE property SET price = price + 100000 WHERE id = 1; -- Должно выполниться
INSERT 0 1
UPDATE 0
mirea=>
```

Задание 7.4: Роль pg_read_all_settings

```
CREATE ROLE monitor WITH LOGIN PASSWORD 'monitor_pass123';
GRANT pg_read_all_settings TO monitor;
```

```
mirea=# CREATE ROLE monitor WITH LOGIN PASSWORD 'monitor_pass123';
GRANT pg_read_all_settings TO monitor;
CREATE ROLE
GRANT ROLE
mirea=#
```

```
SELECT name, setting FROM pg_settings WHERE name LIKE '%mem%';
```

```
mirea=> SELECT name, setting FROM pg_settings WHERE name LIKE '%mem%';
              name                 | setting
-----------------------------------+---------
 autovacuum_work_mem               | -1
 dynamic_shared_memory_type        | posix
 enable_memoize                    | on
 hash_mem_multiplier               | 2
 logical_decoding_work_mem         | 65536
 maintenance_work_mem              | 65536
 min_dynamic_shared_memory         | 0
 multixact_member_buffers          | 32
 shared_memory_size                | 143
 shared_memory_size_in_huge_pages  | 72
 shared_memory_type                | mmap
 work_mem                          | 4096
(12 rows)

mirea=>
```

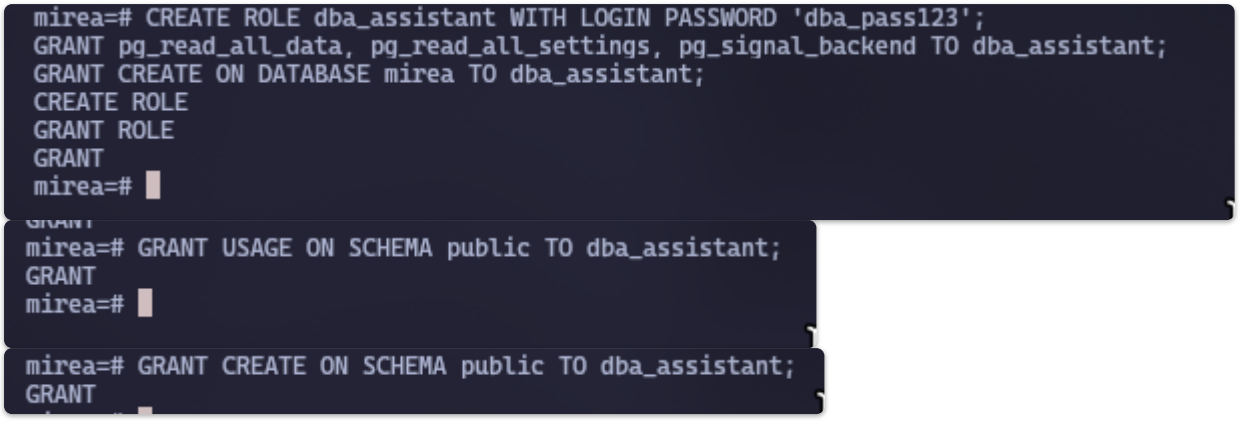
Задание 7.5: Роль pg_read_server_files

```
CREATE ROLE file_operator WITH LOGIN PASSWORD 'file_pass123';
GRANT pg_read_server_files TO file_operator;
GRANT pg_write_server_files TO file_operator;
GRANT SELECT ON ALL TABLES IN SCHEMA public TO file_operator;
```

```
mirea=# CREATE ROLE file_operator WITH LOGIN PASSWORD 'file_pass123';
GRANT pg_read_server_files TO file_operator;
GRANT pg_write_server_files TO file_operator;
GRANT SELECT ON ALL TABLES IN SCHEMA public TO file_operator;
CREATE ROLE
GRANT ROLE
GRANT ROLE
GRANT
mirea=#
```

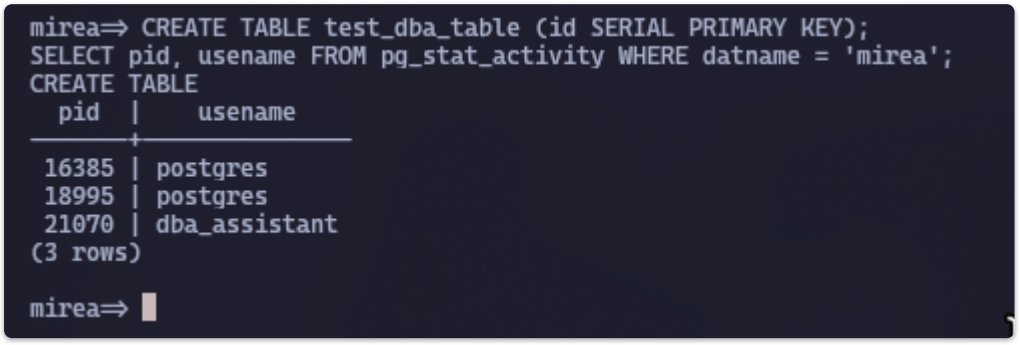
Задание 7.6: Комбинирование встроенных и пользовательских ролей

```
CREATE ROLE dba_assistant WITH LOGIN PASSWORD 'dba_pass123';
GRANT pg_read_all_data, pg_read_all_settings, pg_signal_backend TO
dba_assistant;
GRANT CREATE ON DATABASE mirea TO dba_assistant;
GRANT USAGE ON SCHEMA public TO dba_assistant;
GRANT CREATE ON SCHEMA public TO dba_assistant;
```



Задание 7.7: Проверка комплексных прав

```
CREATE TABLE test_dba_table (id SERIAL PRIMARY KEY);
SELECT pid, usename FROM pg_stat_activity WHERE datname = 'mirea';
```



Задание 7.8: Аудит использования предопределённых ролей

```
SELECT r.rolname AS user_role, m.rolname AS member_of,
       CASE WHEN m.rolname LIKE 'pg_%' THEN 'Встроенная роль' END
AS role_description
FROM pg_roles r
JOIN pg_auth_members am ON r.oid = am.member
JOIN pg_roles m ON am.roleid = m.oid
WHERE m.rolname LIKE 'pg_%';
```

```
mirea=# SELECT r.rolname AS user_role, m.rolname AS member_of,
       CASE WHEN m.rolname LIKE 'pg_%' THEN 'Встроенная роль' END AS role_description
FROM pg_roles r
JOIN pg_auth_members am ON r.oid = am.member
JOIN pg_roles m ON am.roleid = m.oid
WHERE m.rolname LIKE 'pg_%';
   user_role   |       member_of       | role_description
---------------+-----------------------+------------------
 dba_assistant | pg_read_all_data      | Встроенная роль
 data_entry    | pg_read_all_data      | Встроенная роль
 yefremov      | pg_read_all_data      | Встроенная роль
 data_entry    | pg_write_all_data     | Встроенная роль
 dba_assistant | pg_read_all_settings  | Встроенная роль
 monitor       | pg_read_all_settings  | Встроенная роль
 pg_monitor    | pg_read_all_settings  | Встроенная роль
 pg_monitor    | pg_read_all_stats     | Встроенная роль
 pg_monitor    | pg_stat_scan_tables   | Встроенная роль
 file_operator | pg_read_server_files  | Встроенная роль
 file_operator | pg_write_server_files | Встроенная роль
 dba_assistant | pg_signal_backend     | Встроенная роль
(12 rows)

mirea=#
```

Задание 7.9: Отзыв встроенных ролей

```
REVOKE pg_read_all_data, pg_read_all_settings FROM yefremov;
REVOKE pg_write_all_data FROM data_entry;
```

```
mirea=# REVOKE pg_read_all_data FROM yefremov;
REVOKE pg_write_all_data FROM data_entry;
REVOKE ROLE
REVOKE ROLE
mirea=#
```

от `yefremov`:

```
SELECT * FROM pg_settings LIMIT 5;
```

```
mirea=> SELECT * FROM pg_settings LIMIT 5;
           name            | setting | unit |                           category                           |                          short_desc                          |                                   extra_desc
                           |  context  | vartype | source  | min_val | max_val | enumvals | boot_val | reset_val | sourcefile | sourceline | pending_restart
---------------------------+---------+------+---------------------------------------------------------+---------------------------------------------------------------+------------------------------------------------
 allow_alter_system         | on      |      | Version and Platform Compatibility / Other Platforms and Clients | Allows running the ALTER SYSTEM command.                      | Can be set to off for environments where global configuration changes should
be made using a different method. | sighup    | bool    | default |         |         |          | on       | on        |            |            | f
 allow_in_place_tablespaces | off     |      | Developer Options                                        | Allows tablespaces directly inside pg_tblspc, for testing.    |
                           | superuser | bool    | default |         |         |          | off      | off       |            |            | f
 allow_system_table_mods    | off     |      | Developer Options                                        | Allows modifications of the structure of system tables.       |
                           | superuser | bool    | default |         |         |          | off      | off       |            |            | f
 application_name           | psql    |      | Reporting and Logging / What to Log                      | Sets the application name to be reported in statistics and logs. |
                           | user      | string  | client  |         |         |          |          | psql      |            |            | f
 archive_cleanup_command    |         |      | Write-Ahead Log / Archive Recovery                       | Sets the shell command that will be executed at every restart point. |
                           | sighup    | string  | default |         |         |          |          |           |            |            | f
(5 rows)

mirea=>
```

Пояснение: pg_settings доступно и без привелегий pg_read_all_data и pg_read_all_settings, это базовые настройки postgres, по умолчанию видны всем пользователям

Задание 7.10: Создание роли для резервного копирования

```
CREATE ROLE backup_operator WITH LOGIN PASSWORD 'backup_pass123';
GRANT pg_read_all_data, pg_read_server_files, pg_write_server_files
TO backup_operator;
```

```
mirea=# CREATE ROLE backup_operator WITH LOGIN PASSWORD 'backup_pass123';
GRANT pg_read_all_data, pg_read_server_files, pg_write_server_files TO backup_operator;
CREATE ROLE
GRANT ROLE
mirea=#
```

от backup_operator:

```
COPY property TO '/tmp/property_backup.csv' WITH CSV HEADER;
```

```
~ ❯ psql -U backup_operator -d mirea
psql (17.6)
Type "help" for help.

mirea=> COPY property TO '/tmp/property_backup.csv' WITH CSV HEADER;
COPY 0
mirea=>
```

Часть 8: Ограничение подключений и безопасность

Задание 8.1: Ограничение количества подключений

```
ALTER ROLE yefremov CONNECTION LIMIT 3;
ALTER ROLE aleksey CONNECTION LIMIT 5;
ALTER ROLE data_entry CONNECTION LIMIT 2;
```

```
mirea=# ALTER ROLE yefremov CONNECTION LIMIT 3;
ALTER ROLE aleksey CONNECTION LIMIT 5;
ALTER ROLE data_entry CONNECTION LIMIT 2;
ALTER ROLE
ALTER ROLE
ALTER ROLE
mirea=#
```

```
SELECT rolname, rolconnlimit, rolconfig
FROM pg_roles
WHERE rolname IN ('yefremov', 'aleksey', 'data_entry');
```

```
mirea=# SELECT rolname, rolconnlimit, rolconfig
FROM pg_roles
WHERE rolname IN ('yefremov', 'aleksey', 'data_entry')
mirea-# ;
  rolname   | rolconnlimit | rolconfig
------------+--------------+-----------
 yefremov   |            3 |
 aleksey    |            5 |
 data_entry |            2 |
(3 rows)

mirea=#
```

Задание 8.2: Временные ограничения сессий

```
ALTER ROLE yefremov VALID UNTIL '2025-12-31';
ALTER ROLE yefremov SET idle_in_transaction_session_timeout =
```

```
'10min';
```

```
mirea=# ALTER ROLE yefremov VALID UNTIL '2025-12-31';
ALTER ROLE yefremov SET idle_in_transaction_session_timeout = '10min';
ALTER ROLE
ALTER ROLE
mirea=#
```

```
SELECT rolname, rolvaliduntil, rolconfig
FROM pg_roles
WHERE rolname IN ('yefremov');
```

```
mirea=# SELECT rolname, rolvaliduntil, rolconfig
FROM pg_roles
WHERE rolname IN ('yefremov');
 rolname  |     rolvaliduntil      |                  rolconfig
----------+------------------------+---------------------------------------------
 yefremov | 2025-12-31 00:00:00+03 | {idle_in_transaction_session_timeout=10min}
(1 row)

mirea=#
```

Задание 8.3: Ограничения по IP-адресам (симуляция)

```
COMMENT ON ROLE yefremov IS 'Allowed: 192.168.1.0/24';
CREATE OR REPLACE VIEW role_network_restrictions AS
SELECT rolname, description::text AS network_restrictions
FROM pg_roles LEFT JOIN pg_shdescription ON pg_roles.oid = objoid
WHERE rolname IN ('yefremov', 'aleksey');
```

```
mirea=# COMMENT ON ROLE yefremov IS 'Allowed: 192.168.1.0/24';
CREATE OR REPLACE VIEW role_network_restrictions AS
SELECT rolname, description::text AS network_restrictions
FROM pg_roles LEFT JOIN pg_shdescription ON pg_roles.oid = objoid
WHERE rolname IN ('yefremov', 'aleksey');
COMMENT
CREATE VIEW
mirea=#
```

```
CREATE OR REPLACE VIEW role_network_restrictions AS
SELECT
rolname,
description::text as network_restrictions
FROM pg_roles
LEFT JOIN pg_shdescription ON pg_roles.oid = objoid
WHERE rolname IN ('yefremov', 'aleksey')
AND classoid = (SELECT oid FROM pg_class WHERE relname =
'pg_authid');
SELECT * FROM role_network_restrictions;
```

```
mirea=# CREATE OR REPLACE VIEW role_network_restrictions AS
SELECT
rolname,
description::text as network_restrictions
FROM pg_roles
LEFT JOIN pg_shdescription ON pg_roles.oid = objoid
WHERE rolname IN ('yefremov', 'aleksey')
AND classoid = (SELECT oid FROM pg_class WHERE relname = 'pg_authid');
SELECT * FROM role_network_restrictions;
CREATE VIEW
 rolname  |  network_restrictions
----------+-------------------------
 yefremov | Allowed: 192.168.1.0/24
(1 row)

mirea=#
```

Часть 9: Управление сессиями пользователей

Задание 9.1: Мониторинг активных сессий

```sql
CREATE OR REPLACE VIEW active_sessions_monitor AS
SELECT
    pid,
    usename,
    application_name,
    client_addr,
    client_hostname,
    backend_start,
    state,
    query_start,
    state_change,
    CASE
        WHEN state = 'active' THEN EXTRACT(EPOCH FROM (now() -
query_start))
        ELSE EXTRACT(EPOCH FROM (now() - state_change))
    END AS state_duration_seconds,
    query,
    wait_event_type,
    wait_event
FROM pg_stat_activity
WHERE datname = 'mirea'
  AND usename != 'postgres'
ORDER BY state_duration_seconds DESC;
```

```
mirea=# CREATE OR REPLACE VIEW active_sessions_monitor AS
SELECT
    pid,
    usename,
    application_name,
    client_addr,
    client_hostname,
    backend_start,
    state,
    query_start,
    state_change,
    CASE
        WHEN state = 'active' THEN EXTRACT(EPOCH FROM (now() - query_start))
        ELSE EXTRACT(EPOCH FROM (now() - state_change))
    END AS state_duration_seconds,
    query,
    wait_event_type,
    wait_event
FROM pg_stat_activity
WHERE datname = 'mirea'
  AND usename ≠ 'postgres'
ORDER BY state_duration_seconds DESC;
CREATE VIEW
mirea=#
```

```sql
-- Проверка
SELECT pid, usename, state, state_duration_seconds, left(query, 50) AS query_preview
FROM active_sessions_monitor LIMIT 10;
```

```
mirea=# -- Проверка
SELECT pid, usename, state, state_duration_seconds, left(query, 50) AS query_preview
FROM active_sessions_monitor LIMIT 10;
  pid  |     usename     | state | state_duration_seconds |                    query_preview
-------+-----------------+-------+------------------------+-----------------------------------------------------
 23683 | backup_operator | idle  |           7614.279740 | COPY property TO '/tmp/property_backup.csv' WITH C
(1 row)

mirea=#
```

Задание 9.2: Управление проблемными сессиями

```sql
-- Функция поиска заблокированных сессий
CREATE OR REPLACE FUNCTION find_blocking_sessions()
RETURNS TABLE(
    blocked_pid integer,
    blocked_user text,
    blocking_pid integer,
    blocking_user text,
    blocked_query text,
    blocking_query text,
    duration interval
) LANGUAGE plpgsql AS $$
BEGIN
    RETURN QUERY
    SELECT
```

```
        blocked_locks.pid AS blocked_pid,
        blocked_activity.usename::text AS blocked_user,
        blocking_locks.pid AS blocking_pid,
        blocking_activity.usename::text AS blocking_user,
        blocked_activity.query AS blocked_query,
        blocking_activity.query AS blocking_query,
        now() - blocked_activity.query_start AS duration
    FROM pg_catalog.pg_locks blocked_locks
    JOIN pg_catalog.pg_stat_activity blocked_activity ON
blocked_activity.pid = blocked_locks.pid
    JOIN pg_catalog.pg_locks blocking_locks ON
blocking_locks.locktype = blocked_locks.locktype
        AND blocking_locks.DATABASE IS NOT DISTINCT FROM
blocked_locks.DATABASE
        AND blocking_locks.relation IS NOT DISTINCT FROM
blocked_locks.relation
        AND blocking_locks.page IS NOT DISTINCT FROM
blocked_locks.page
        AND blocking_locks.tuple IS NOT DISTINCT FROM
blocked_locks.tuple
        AND blocking_locks.virtualxid IS NOT DISTINCT FROM
blocked_locks.virtualxid
        AND blocking_locks.transactionid IS NOT DISTINCT FROM
blocked_locks.transactionid
        AND blocking_locks.classid IS NOT DISTINCT FROM
blocked_locks.classid
        AND blocking_locks.objid IS NOT DISTINCT FROM
blocked_locks.objid
        AND blocking_locks.objsubid IS NOT DISTINCT FROM
blocked_locks.objsubid
        AND blocking_locks.pid != blocked_locks.pid
    JOIN pg_catalog.pg_stat_activity blocking_activity ON
blocking_activity.pid = blocking_locks.pid
    WHERE NOT blocked_locks.granted;
END;
$$;

-- Функция завершения idle-сессий
CREATE OR REPLACE FUNCTION terminate_idle_sessions(
    max_idle_minutes integer DEFAULT 60,
    exclude_users text[] DEFAULT ARRAY['postgres']
) RETURNS TABLE(terminated_pid integer, username text, duration
```

```
interval)
LANGUAGE plpgsql AS $$
BEGIN
    RETURN QUERY
    WITH terminated AS (
        SELECT
            pid,
            usename,
            now() - state_change AS idle_duration,
            pg_terminate_backend(pid) AS terminated
        FROM pg_stat_activity
        WHERE state = 'idle'
          AND (now() - state_change) > (max_idle_minutes *
interval '1 minute')
          AND usename != ALL(exclude_users)
          AND datname = 'mirea'
    )
    SELECT pid, usename::text, idle_duration FROM terminated WHERE
terminated = true;
END;
$$;
```

```
mirea=# -- Функция поиска заблокированных сессий
CREATE OR REPLACE FUNCTION find_blocking_sessions()
RETURNS TABLE(
    blocked_pid integer,
    blocked_user text,
    blocking_pid integer,
    blocking_user text,
    blocked_query text,
    blocking_query text,
    duration interval
) LANGUAGE plpgsql AS $$
BEGIN
    RETURN QUERY
    SELECT
        blocked_locks.pid AS blocked_pid,
        blocked_activity.usename::text AS blocked_user,
        blocking_locks.pid AS blocking_pid,
        blocking_activity.usename::text AS blocking_user,
        blocked_activity.query AS blocked_query,
        blocking_activity.query AS blocking_query,
        now() - blocked_activity.query_start AS duration
    FROM pg_catalog.pg_locks blocked_locks
    JOIN pg_catalog.pg_stat_activity blocked_activity ON blocked_activity.pid = blocked_locks.pid
    JOIN pg_catalog.pg_locks blocking_locks ON blocking_locks.locktype = blocked_locks.locktype
        AND blocking_locks.DATABASE IS NOT DISTINCT FROM blocked_locks.DATABASE
        AND blocking_locks.relation IS NOT DISTINCT FROM blocked_locks.relation
        AND blocking_locks.page IS NOT DISTINCT FROM blocked_locks.page
        AND blocking_locks.tuple IS NOT DISTINCT FROM blocked_locks.tuple
        AND blocking_locks.virtualxid IS NOT DISTINCT FROM blocked_locks.virtualxid
        AND blocking_locks.transactionid IS NOT DISTINCT FROM blocked_locks.transactionid
        AND blocking_locks.classid IS NOT DISTINCT FROM blocked_locks.classid
        AND blocking_locks.objid IS NOT DISTINCT FROM blocked_locks.objid
        AND blocking_locks.objsubid IS NOT DISTINCT FROM blocked_locks.objsubid
        AND blocking_locks.pid ≠ blocked_locks.pid
    JOIN pg_catalog.pg_stat_activity blocking_activity ON blocking_activity.pid = blocking_locks.pid
    WHERE NOT blocked_locks.granted;
END;
$$;

-- Функция завершения idle-сессий
CREATE OR REPLACE FUNCTION terminate_idle_sessions(
    max_idle_minutes integer DEFAULT 60,
    exclude_users text[] DEFAULT ARRAY['postgres']
) RETURNS TABLE(terminated_pid integer, username text, duration interval)
LANGUAGE plpgsql AS $$
BEGIN
    RETURN QUERY
    WITH terminated AS (
        SELECT
            pid,
            usename,
            now() - state_change AS idle_duration,
            pg_terminate_backend(pid) AS terminated
        FROM pg_stat_activity
        WHERE state = 'idle'
          AND (now() - state_change) > (max_idle_minutes * interval '1 minute')
          AND usename ≠ ALL(exclude_users)
          AND datname = 'mirea'
    )
    SELECT pid, usename::text, idle_duration FROM terminated WHERE terminated = true;
END;
$$;
CREATE FUNCTION
CREATE FUNCTION
mirea=#
```

```sql
-- Тестирование
SELECT * FROM find_blocking_sessions();
SELECT * FROM terminate_idle_sessions(1, ARRAY['postgres',
'yefremov']);
```

```
mirea=# -- Тестирование
SELECT * FROM find_blocking_sessions();
SELECT * FROM terminate_idle_sessions(1, ARRAY['postgres', 'yefremov']);
 blocked_pid | blocked_user | blocking_pid | blocking_user | blocked_query | blocking_query | duration
-------------+--------------+--------------+---------------+---------------+----------------+----------
(0 rows)

 terminated_pid |    username     |    duration
----------------+-----------------+-----------------
          23683 | backup_operator | 02:08:59.888087
(1 row)

mirea=#
```

Задание 9.3: Статистика использования сессий

```sql
CREATE TABLE session_usage_stats (
    id SERIAL PRIMARY KEY,
    username TEXT NOT NULL,
    session_start TIMESTAMP DEFAULT now(),
    session_end TIMESTAMP,
    client_addr INET,
    application_name TEXT,
    query_count INTEGER DEFAULT 0
);

CREATE OR REPLACE FUNCTION log_session_usage()
RETURNS void LANGUAGE plpgsql AS $$
BEGIN
    INSERT INTO session_usage_stats (username, client_addr,
application_name, query_count)
    SELECT
        usename,
        client_addr,
        application_name,
        COUNT(*)::integer
    FROM pg_stat_activity
    WHERE datname = 'mirea'
      AND state = 'active'
      AND usename != 'postgres'
    GROUP BY usename, client_addr, application_name
    ON CONFLICT DO NOTHING;
END;
$$;
```

```
mirea=# CREATE TABLE session_usage_stats (
    id SERIAL PRIMARY KEY,
    username TEXT NOT NULL,
    session_start TIMESTAMP DEFAULT now(),
    session_end TIMESTAMP,
    client_addr INET,
    application_name TEXT,
    query_count INTEGER DEFAULT 0
);

CREATE OR REPLACE FUNCTION log_session_usage()
RETURNS void LANGUAGE plpgsql AS $$
BEGIN
    INSERT INTO session_usage_stats (username, client_addr, application_name, query_count)
    SELECT
        usename,
        client_addr,
        application_name,
        COUNT(*)::integer
    FROM pg_stat_activity
    WHERE datname = 'mirea'
      AND state = 'active'
      AND usename ≠ 'postgres'
    GROUP BY usename, client_addr, application_name
    ON CONFLICT (username, client_addr, application_name) DO UPDATE
    SET query_count = session_usage_stats.query_count + EXCLUDED.query_count,
        session_start = LEAST(session_usage_stats.session_start, now());
END;
$$;
CREATE TABLE
CREATE FUNCTION
mirea=#
```

```sql
SELECT log_session_usage();
SELECT * FROM session_usage_stats ORDER BY query_count DESC;
```

```
SELECT * FROM session_usage_stats ORDER BY query_count DESC;
 log_session_usage
-------------------

(1 row)

 id | username | session_start | session_end | client_addr | application_name | query_count
----+----------+---------------+-------------+-------------+------------------+-------------
(0 rows)

mirea=#
```

Часть 10: Аудит и логирование действий

Задание 10.1: Создание системы аудита

```sql
CREATE TABLE role_audit_log (
    id BIGSERIAL PRIMARY KEY,
    event_time TIMESTAMP DEFAULT now(),
    username TEXT NOT NULL,
    database_name TEXT NOT NULL,
    client_addr INET,
    client_port INTEGER,
    session_id TEXT,
    action_type TEXT NOT NULL,
    object_schema TEXT,
```

```
    object_name TEXT,
    command_tag TEXT,
    sql_text TEXT,
    parameters JSONB,
    application_name TEXT
);

-- DDL-аудит
CREATE OR REPLACE FUNCTION log_ddl_event()
RETURNS event_trigger LANGUAGE plpgsql AS $$
DECLARE
    r RECORD;
BEGIN
    FOR r IN SELECT * FROM pg_event_trigger_ddl_commands()
    LOOP
        INSERT INTO role_audit_log (
            username, database_name, action_type,
            object_schema, object_name, command_tag
        )
        VALUES (
            current_user,
            current_database(),
            'DDL',
            r.schema_name,
            r.object_identity,
            tg_tag
        );
    END LOOP;
END;
$$;

CREATE EVENT TRIGGER audit_ddl_events
    ON ddl_command_end
    EXECUTE FUNCTION log_ddl_event();
```

Тестирование

```
CREATE TABLE test_ddl_table (id serial);
ALTER TABLE test_ddl_table ADD COLUMN note TEXT;
DROP TABLE test_ddl_table;

SELECT event_time, action_type, object_name, command_tag
```

```
FROM role_audit_log
WHERE action_type = 'DDL'
ORDER BY id DESC LIMIT 5;
```

```
CREATE TABLE
ALTER TABLE
DROP TABLE
        event_time         | action_type |         object_name          | command_tag
---------------------------+-------------+------------------------------+-------------
 2025-11-06 20:42:13.372256 | DDL         | public.test_ddl_table        | ALTER TABLE
 2025-11-06 20:42:13.356913 | DDL         | public.test_ddl_table_id_seq | CREATE TABLE
 2025-11-06 20:42:13.356913 | DDL         | public.test_ddl_table        | CREATE TABLE
 2025-11-06 20:42:13.356913 | DDL         | public.test_ddl_table_id_seq | CREATE TABLE
(4 rows)

mirea=#
```

Задание 10.2: Аудит DML-операций

```
CREATE OR REPLACE FUNCTION audit_table_changes()
RETURNS TRIGGER LANGUAGE plpgsql AS $$
DECLARE
    old_data JSONB := NULL;
    new_data JSONB := NULL;
BEGIN
    IF TG_OP = 'INSERT' THEN new_data = to_jsonb(NEW);
    ELSIF TG_OP = 'UPDATE' THEN new_data = to_jsonb(NEW); old_data
= to_jsonb(OLD);
    ELSIF TG_OP = 'DELETE' THEN old_data = to_jsonb(OLD);
    END IF;

    INSERT INTO role_audit_log (
        username, database_name, client_addr, action_type,
        object_schema, object_name, command_tag, parameters,
application_name
    ) VALUES (
        current_user, current_database(), inet_client_addr(),
        TG_OP, TG_TABLE_SCHEMA, TG_TABLE_NAME, TG_OP,
        jsonb_build_object('old', old_data, 'new', new_data),
        current_setting('application_name', true)
    );
    RETURN COALESCE(NEW, OLD);
END;
$$;

-- Триггеры
CREATE TRIGGER audit_property_changes
    AFTER INSERT OR UPDATE OR DELETE ON property
```

```
    FOR EACH ROW EXECUTE FUNCTION audit_table_changes();

CREATE TRIGGER audit_sale_changes
    AFTER INSERT OR UPDATE OR DELETE ON sale
    FOR EACH ROW EXECUTE FUNCTION audit_table_changes();

CREATE TRIGGER audit_realtor_changes
    AFTER INSERT OR UPDATE OR DELETE ON realtor
    FOR EACH ROW EXECUTE FUNCTION audit_table_changes();
```

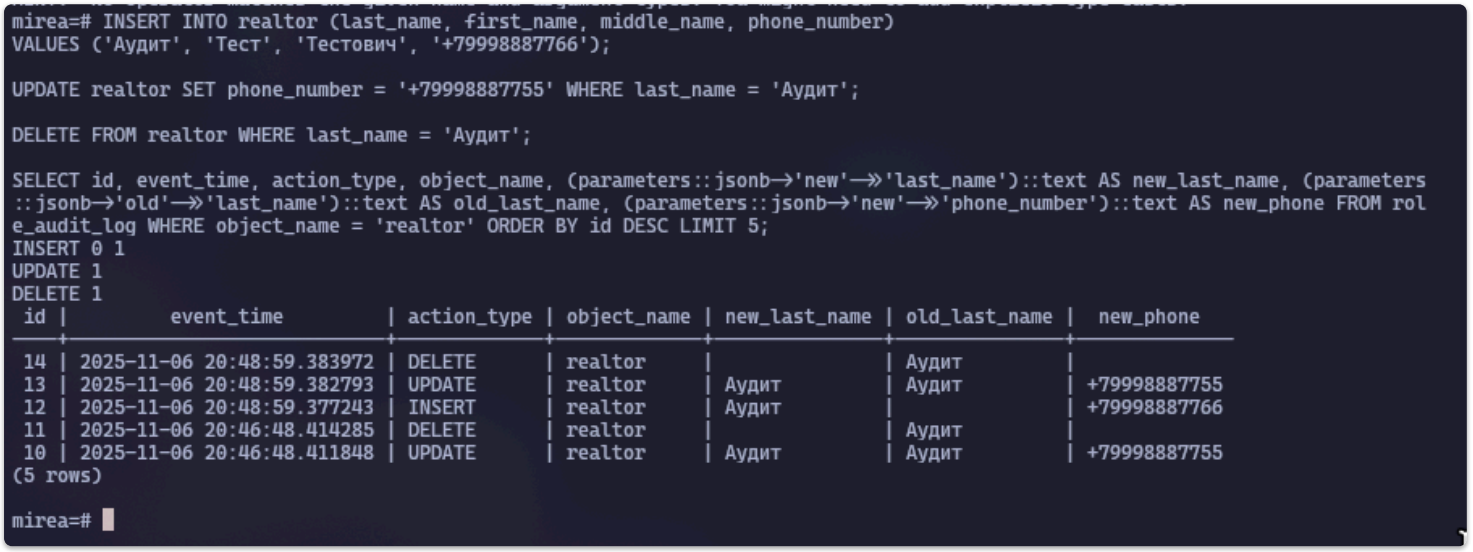
Тестирование

```
INSERT INTO realtor (last_name, first_name, middle_name,
phone_number)
VALUES ('Аудит', 'Тест', 'Тестович', '+79998887766');

UPDATE realtor SET phone_number = '+79998887755' WHERE last_name =
'Аудит';

DELETE FROM realtor WHERE last_name = 'Аудит';

SELECT id, event_time, action_type, object_name,
(parameters::jsonb->'new'->>'last_name')::text AS new_last_name,
(parameters::jsonb->'old'->>'last_name')::text AS old_last_name,
(parameters::jsonb->'new'->>'phone_number')::text AS new_phone
FROM role_audit_log WHERE object_name = 'realtor' ORDER BY id DESC
LIMIT 5;
```



Задание 10.3: Отчеты и анализ аудита

```
CREATE OR REPLACE VIEW user_activity_report AS
SELECT
    username,
    action_type,
    COUNT(*) AS action_count,
    MIN(event_time) AS first_action,
    MAX(event_time) AS last_action,
    COUNT(DISTINCT object_name) AS objects_affected
FROM role_audit_log
WHERE event_time >= now() - interval '7 days'
GROUP BY username, action_type
ORDER BY action_count DESC;

CREATE OR REPLACE VIEW suspicious_activity_report AS
SELECT
    username, client_addr, action_type, object_name, event_time,
sql_text
FROM role_audit_log
WHERE action_type IN ('DELETE', 'DROP', 'TRUNCATE')
   OR object_name IN ('realtor', 'sale', 'property')
   OR sql_text ILIKE '%password%' OR sql_text ILIKE '%secret%'
ORDER BY event_time DESC;

CREATE OR REPLACE FUNCTION cleanup_audit_log(retention_days
integer DEFAULT 90)
RETURNS bigint LANGUAGE plpgsql AS $$
DECLARE deleted_count bigint;
BEGIN
    DELETE FROM role_audit_log
    WHERE event_time < now() - (retention_days * interval '1
day');
    GET DIAGNOSTICS deleted_count = ROW_COUNT;
    RETURN deleted_count;
END;
$$;
```

Тестирование

```
SELECT * FROM user_activity_report LIMIT 5;
SELECT * FROM suspicious_activity_report LIMIT 10;
```

```
mirea=# SELECT * FROM user_activity_report LIMIT 5;
SELECT * FROM suspicious_activity_report LIMIT 10;
 username | action_type | action_count |        first_action        |        last_action         | objects_affected
----------+-------------+--------------+----------------------------+----------------------------+------------------
 postgres | DDL         |           11 | 2025-11-06 20:42:13.356913 | 2025-11-06 20:50:20.508022 |                9
 postgres | DELETE      |            2 | 2025-11-06 20:46:48.414285 | 2025-11-06 20:48:59.383972 |                1
 postgres | INSERT      |            2 | 2025-11-06 20:46:48.397523 | 2025-11-06 20:48:59.377243 |                1
 postgres | UPDATE      |            2 | 2025-11-06 20:46:48.411848 | 2025-11-06 20:48:59.382793 |                1
(4 rows)

 username | client_addr | action_type | object_name |         event_time         | sql_text
----------+-------------+-------------+-------------+----------------------------+----------
 postgres |             | DELETE      | realtor     | 2025-11-06 20:48:59.383972 |
 postgres |             | UPDATE      | realtor     | 2025-11-06 20:48:59.382793 |
 postgres |             | INSERT      | realtor     | 2025-11-06 20:48:59.377243 |
 postgres |             | DELETE      | realtor     | 2025-11-06 20:46:48.414285 |
 postgres |             | UPDATE      | realtor     | 2025-11-06 20:46:48.411848 |
 postgres |             | INSERT      | realtor     | 2025-11-06 20:46:48.397523 |
(6 rows)

mirea=#
```

Задание 10.4: Мониторинг попыток доступа

```sql
CREATE TABLE auth_attempts_log (
    id BIGSERIAL PRIMARY KEY,
    attempt_time TIMESTAMP DEFAULT now(),
    username TEXT,
    client_addr INET,
    success BOOLEAN,
    error_message TEXT
);

CREATE OR REPLACE VIEW failed_login_attempts AS
SELECT
    username,
    client_addr,
    COUNT(*) AS failed_attempts,
    MIN(attempt_time) AS first_attempt,
    MAX(attempt_time) AS last_attempt
FROM auth_attempts_log
WHERE success = false
  AND attempt_time >= now() - interval '1 hour'
GROUP BY username, client_addr
HAVING COUNT(*) > 5
ORDER BY failed_attempts DESC;
```

Тестирование

```sql
DO $$
BEGIN
    FOR i IN 1..7 LOOP
        INSERT INTO auth_attempts_log (username, client_addr,
```

```
success, error_message)
        VALUES ('aleksey', '192.168.1.100', false, 'invalid
password');
    END LOOP;
END $$;
SELECT * FROM failed_login_attempts;
```