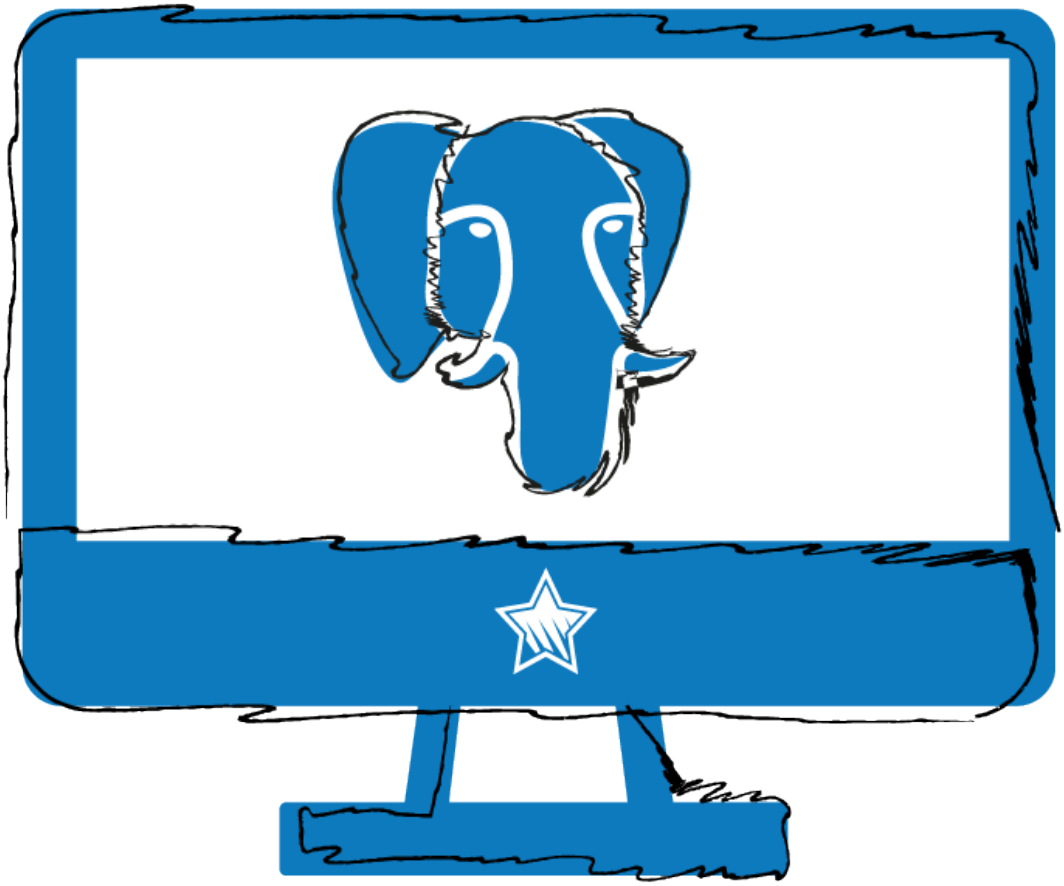
Курс AL-1707

Сборник практических заданий для курса



ASTRA LINUX®

Оглавление

Сборник практических заданий для курса «AL-1707. СУБД PostgreSQL в Astra Linux Special Edition 1.7: установка, администрирование и мониторинг»....3
Модуль 1. Установка и настройка СУБД PostgreSQL....5
Модуль 2. Создание баз данных....16
Модуль 3. Подключение и аутентификация....24
Модуль 4. Роли в СУБД PostgreSQL....30
Модуль 5. Дискреционное управление доступом в СУБД PostgreSQL в Astra Linux Special Edition....34
Модуль 6. Мандатное управление доступом в СУБД PostgreSQL в Astra Linux Special Edition....39
Часть 1. Мандатное управление доступом в ОС Astra Linux Special Edition....39
Часть 2. Мандатное управление доступом в СУБД PostgreSQL....42
Модуль 7. Журналирование и аудит доступа к данным....52
Модуль 8. Логическое резервное копирование и восстановление....61
Часть 1. Подготовка к резервному копированию, экспорт и импорт данных таблицы с помощью команды psql \copy....61
Часть 2. Создание резервных копий с помощью утилиты pg_dump в различных форматах. Восстановление данных....66
Модуль 9. Физическое резервное копирование и восстановление....74
Модуль 10. Регламентные работы....88
Модуль 11. Автоматизация обслуживания....95

Сборник практических заданий для курса «AL-1707. СУБД PostgreSQL в Astra Linux Special Edition 1.7: установка, администрирование и мониторинг»

Требования:

- Компьютер со следующими характеристиками:
  - процессор архитектуры AMD64 (x86_64);
  - RAM, не менее 4 ГБ;
  - диски, не менее 40 ГБ свободного места;
- ОС GNU/Linux или Windows 10 (или более новые версии);
- VirtualBox 6.1 или новее.

Материалы:

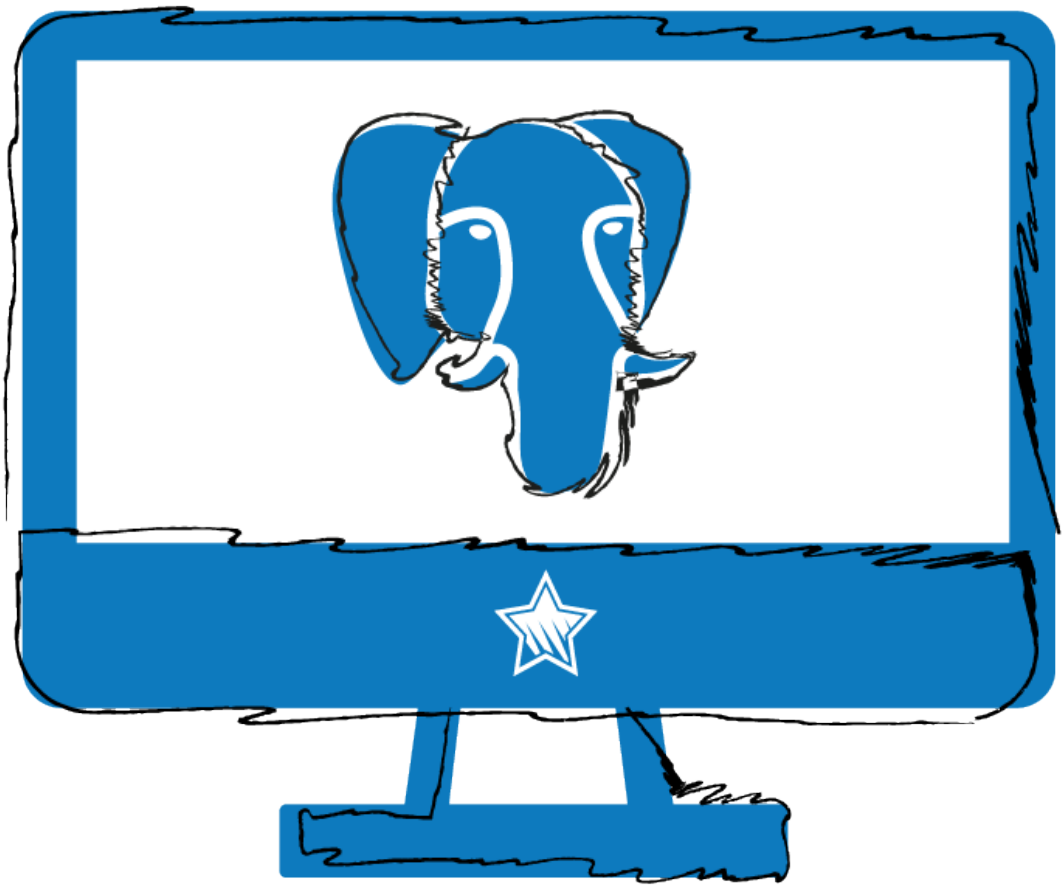
- ОС — релиз Astra Linux Special Edition 1.7.4.

Дополнительная информация:

- Могут быть использованы другие системы виртуализации для проведения учебных занятий.
- Виртуальная машина для проведения занятий подготавливается преподавателем и раздается слушателям, при необходимости, слушатель сам может подготовить виртуальную машину на основе практической работы по курсу AL-1702 (Модуль 2).
- Если используется система виртуализации VirtualBox, то рекомендуется после установки ОС Astra Linux установить дополнения гостевой ОС.
- Рекомендуется развернуть *основной (main)* и *базовый (base)* репозитории, а также кумулятивное оперативное обновление на сетевом ресурсе и обеспечить доступ по ftp-, http- или https-протоколам.

Модуль 1

Установка и настройка СУБД PostgreSQL



ASTRA LINUX®

Модуль 1. Установка и настройка СУБД PostgreSQL

Описание практического задания:

Оценка времени выполнения: 40 мин.

- Установка PostgreSQL.
- Создание кластера БД reserve и запуск обслуживающего его экземпляра.
- Подключение к кластеру БД main через psql.
- Создание супрепользователя PostgreSQL.
- Установка pgAdmin.
- Изменение параметров конфигурации под учетной записью администратора ОС, суперпользователя PostgreSQL.

Установка PostgreSQL

Задание: Создать ВМ 1707. Сделать снимок виртуальной машины «Начальное состояние». Установить PostgreSQL, проверить, работает ли сервер PostgreSQL после установки. Просмотреть список всех TCP-сокетов (IPv4 и IPv6), которые прослушиваются сервером PostgreSQL, с отображением имени сервиса и без.

1 Создать ВМ Postgres путем клонирования ВМ *template_1707_1_7_4.ova*. Сетевая карта – одна, тип подключения — NAT. Запустить ВМ.

2 Сделать снимок виртуальной машины. Назвать снимок «Начальное состояние».

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Установить PostgreSQL.

```
sudo apt install postgresql
```

3 Проверить, что PostgreSQL работает:

```
sudo pg_ctlcluster 11 main status
```

```
sa@astra:/$ sudo pg_ctlcluster 11 main status
pg_ctl: сервер работает (PID: 16288)
/usr/lib/postgresql/11/bin/postgres "-D" "/var/lib
/postgresql/11/main" "-c" "config_file=/etc/postgr
esql/11/main/postgresql.conf"
```

4 Просмотреть список всех TCP-сокетов (IPv4 и IPv6), которые прослушиваются сервером PostgreSQL (с отображением имени сервиса).

```
sudo ss -tlp | grep postgres
```

5 Просмотреть список всех TCP-сокетов (IPv4 и IPv6), которые прослушиваются сервером PostgreSQL (с без отображения имени сервиса).

```
sudo ss -tlpn | grep postgres
```

Создание дополнительного кластера БД

Задание: Создать кластер БД reserve. Запустить экземпляр PostgreSQL, обслуживающий кластер БД reserve. Отследить состояние всех экземпляров PostgreSQL в ОС. Просмотреть список всех TCP-сокетов (IPv4 и IPv6), которые прослушиваются экземплярами PostgreSQL.

1 Создать новый кластер БД версии 11 с именем *reserve:*

```
sudo pg_createcluster 11 reserve
```

2 Просмотреть состояние всех кластеров БД в ОС.

```
pg_lsclusters
```

3 Запустить экземпляр PostgreSQL, обслуживающий кластер БД reserve.

```
sudo pg_ctlcluster 11 reserve start
```

4 Повторно просмотреть состояние всех кластеров БД в ОС.

```
pg_lsclusters
```

5 Просмотреть список всех TCP-сокетов (IPv4 и IPv6), которые прослушиваются сервером PostgreSQL (с без отображения имени сервиса).

```
sudo ss -tlpn | grep postgres
```

Подключение к кластеру БД main через psql

Задание: Подключиться через утилиту psql к кластеру БД main под учетной записью postgres. Создать роль sa с правами суперпользователя и назначить ей пароль 'P@ssw0rd'. Выдать права на чтение информации о мандатных метках пользователей и их привилегиях в ОС пользователю postgres. Назначить мандатную метку пользователю sa с уровнем конфиденциальности 0. Подключиться к кластеру БД main, БД postgres, используя роль sa. Отобразить информацию о подключении. Подключиться к БД template1.

1 Подключиться к кластеру БД main под учетной записью postgres:

```
sudo -u postgres psql
```

2 Создать роль sa с правами суперпользователя и назначить ей пароль 'P@ssw0rd':

```
CREATE ROLE sa WITH LOGIN SUPERUSER PASSWORD 'P@ssw0rd';
```

3 Вывести список существующих в кластере main баз данных в консоль:

```
\l
```

4 Вывести список существующих в кластере main ролей в консоль.

```
\du
```

5 Выйти из psql и вывести список существующих в кластере main баз данных без интерактивного входа в консоль:

```
\q
sudo -u postgres psql -l
```

6 Попытаться подключиться к кластеру БД main, БД postgres, используя роль sa:

```
psql -h localhost -d postgres -U sa
```

7 Переключиться на root и выполнить следующие команды:

```
sudo -i
setfacl -d -m u:postgres:r /etc/parsec/macdb
setfacl -R -m u:postgres:r /etc/parsec/macdb
setfacl -m u:postgres:rx /etc/parsec/macdb
setfacl -d -m u:postgres:r /etc/parsec/capdb
setfacl -R -m u:postgres:r /etc/parsec/capdb
setfacl -m u:postgres:rx /etc/parsec/capdb
exit
```

Выполняя эти команды, вы выдали пользователю postgres права на чтение информации об уровнях доступа, категориях и привилегиях, присвоенных локальным пользователям в ОС. Подробнее подключение и авторизация будут рассмотрены в модуле 2, мандатное управление доступом – в модуле 6.

8 Назначить мандатную метку пользователю sa:

```
sudo pdpl-user -l 0:0 sa
```

Назначение мандатных меток будет подробно рассмотрено в модуле 6.

9 Попытаться подключиться к кластеру БД main, БД postgres, используя роль sa повторно:

```
psql -h localhost -d postgres -U sa
```

Подключение пройдет успешно.

10 Отобразить информацию о подключении.

```
\conninfo
```

11 Подключиться к БД template1.

```
\connect template1
```

Обратить внимание на то, как изменилось приглашение .

12 Выйти из psql.

```
\q
```

Создание суперпользователя PostgreSQL

Задание: Создать роль iivanov в кластере БД main, назначить ей атрибуты SUPERUSER, PASSWORD. В ОС создать пользователя *iivanov*. Задать ему пароль, уровень конфиденциальности: минимальный – **0**, максимальный – **3**. Указать, что пользователь имеет доступ к категориям информации: 0 (без категории), 1, 2. При создании указать, что должна быть создана домашняя директория. В качестве оболочки для пользователя указать /bin/bash. Предоставить доступ к категориям конфиденциальности 1, 2. Войти в ОС с учетной записью пользователя iivanov. При входе выбрать уровень конфиденциальности *3, 0:Категория 1, 1:Категория 2*. Выполнить подключение к БД postgres от имени роли iivanov.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться к БД postgres через утилиту psql от имени пользователя sa.

```
psql -h localhost -d postgres -U sa
```

3 Создать роль *iivanov*. В качестве атрибутов роли указать SUPERUSER, PASSWORD:

```
CREATE ROLE iivanov WITH LOGIN SUPERUSER PASSWORD
'P@ssw0rd';
```

4 Проверить, что роль создана, и выйти из psql.

```
\du
\q
```

5 Создать в ОС пользователя iivanov.

Задать ему пароль (например, P@ssw0rd) и уровень конфиденциальности.

Минимальный уровень конфиденциальности установить в значение 0, максимальный уровень конфиденциальности установить в значение 3.

Предоставить доступ к категориям конфиденциальности 1, 2:

```
sudo useradd -m -s /bin/bash iivanov
sudo passwd iivanov
```

```
sudo pdpl-user -l 0:3 -c 0:3 iivanov
```

6 Выйти из сеанса ОС и войти с учетной записью iivanov. При входе выбрать уровень конфиденциальности 3, 0:Категория 1, 1:Категория 2.

7 Подключиться через утилиту psql к базе postgres от имени пользователя `iivanov`:

```
psql -h localhost -d postgres -U iivanov
```

Подключение пройдет успешно.

Установка pgAdmin

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Установить pgadmin3, добавить новое подключение к кластеру БД main от имени роли sa, подключиться, ознакомиться с графическим интерфейсом утилиты pgadmin3.

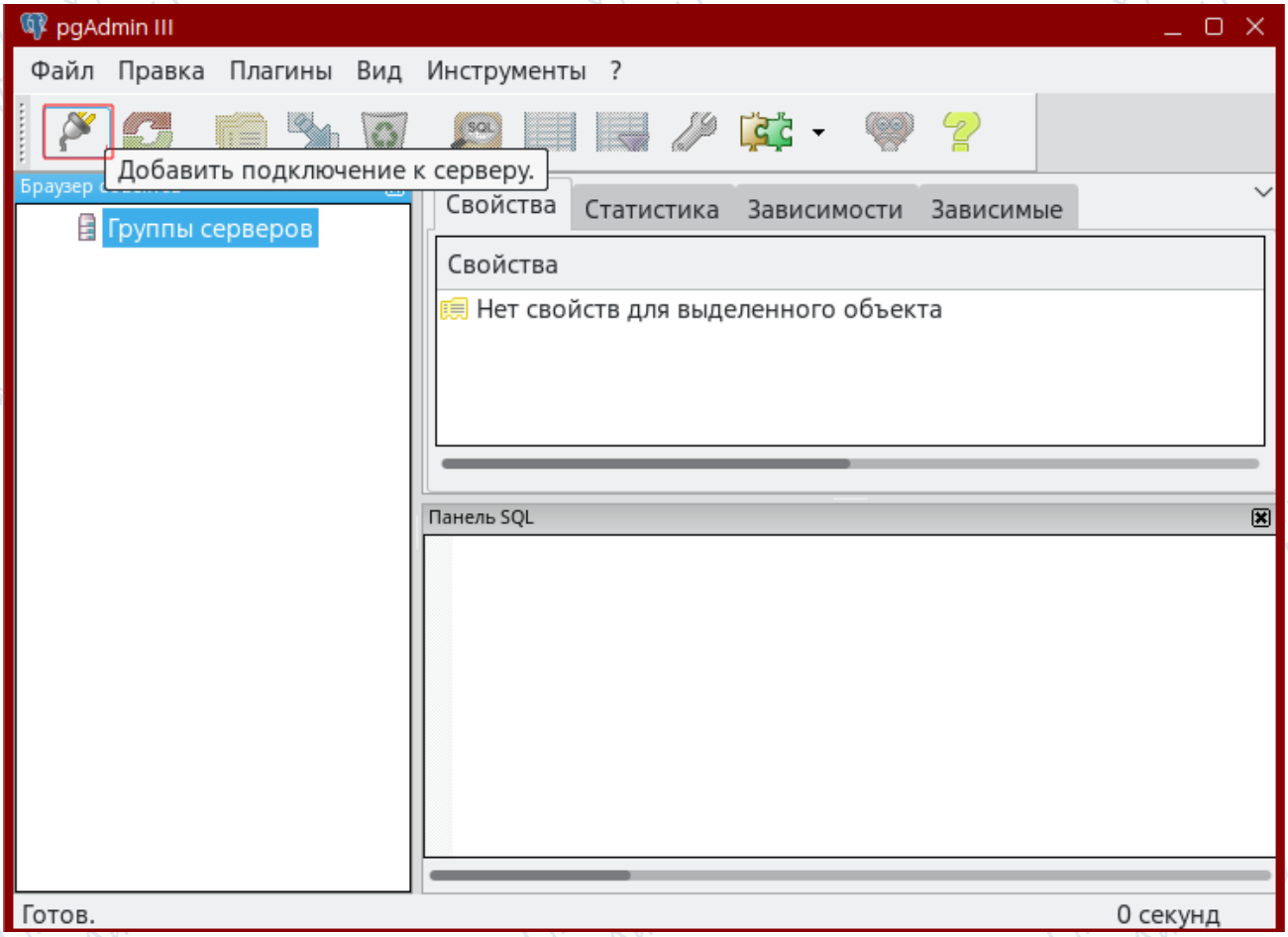
1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Установить pgadmin3.

```
sudo apt install pgadmin3
```

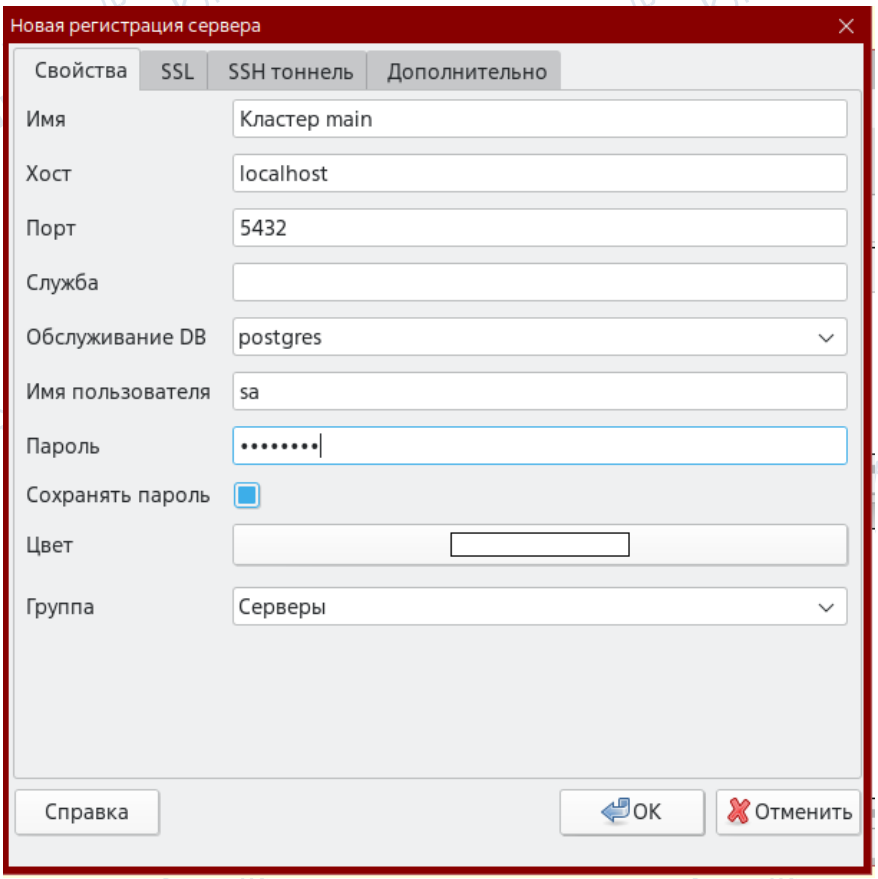
3 Через графическое меню **Пуск ➞ Разработка** запустить утилиту pgAdmin III.

4 Создать новое подключение, используя кнопку панели или пункт меню **Файл➞Добавить сервер**.



5 В окне **Новая регистрация сервера** укажите:

- в поле **Имя** – *«Кластер main»*,
- в поле **Хост** – *localhost*,
- в поле **Порт** – *5432*,
- в поле **Обслуживание DB** – *postgres,*
- в поле **Имя пользователя** – *sa*,
- в поле **Пароль** – *P@ssw0rd*.

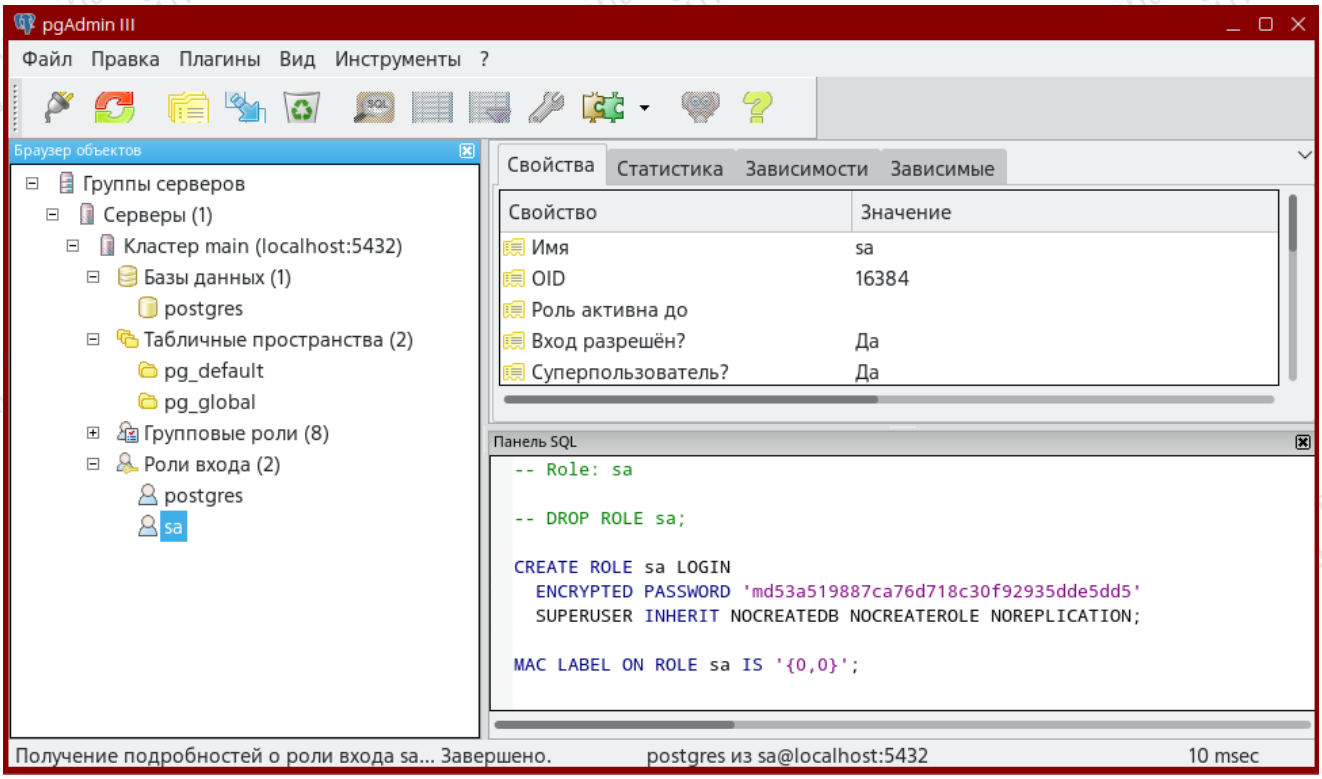


6 Нажмите кнопку **OK**.

7 В окне **Подсказка гуру – Сохранение паролей** нажмите **OK**.

8 После подключения в окне **Браузер объектов** разверните «Группы серверов», «Серверы (1)», «Кластер main (localhost:5432)».

9 Исследуйте объекты кластера БД. Обратите внимание на то, какую информацию можно получить через интерфейс.



Изменение параметров конфигурации под учетной записью — администратором ОС, суперпользователем PostgreSQL

Задание: Подключиться к кластеру БД main, БД postgres, используя роль sa. Отобразить текущее значение параметра config_file для кластера БД main. Ознакомиться со списком параметров и их значениями, сделав выборку из представления pg_file_settings. Изменить значение параметра max_connections на 1000 в файле /etc/postgresql/11/main/postgresql.conf. Перезапустить экземпляр сервера PostgreSQL, обслуживающий кластер БД main. Убедиться, что значение параметра изменилось.

1 Подключиться к кластеру БД main, БД postgres, используя роль sa.

```
psql -h localhost -d postgres -U sa
```

2 Отобразить текущее значение параметра config_file для кластера БД main.

```
SHOW config_file;
```

3 Сделать выборку из представления pg_file_settings.

```
SELECT * FROM pg_file_settings;
```

Ознакомьтесь со списком параметров и их значениями.

4 Выйти из psql.

```
\q
```

5 Изменить значение параметра max_connections (задает максимальное количество одновременных соединений, которое экземпляр PostgreSQL может обслуживать) на 1000 в файле /etc/postgresql/11/main/postgresql.conf.

```
sudo mcedit /etc/postgresql/11/main/postgresql.conf
Изменить значение параметра max_connections на 1000.
Нажать F10 для выхода. При выходе сохранить изменения.
```

6 Повторно считать конфигурацию кластера БД main.

```
sudo pg_ctlcluster 11 main reload
```

7 Подключиться к кластеру БД main, БД postgres, используя роль sa.

```
psql -h localhost -d postgres -U sa
```

8 Вывести текущее значение параметра max_connections.

```
SHOW max_connections;
```

Оно останется неизменным. Для изменения этого параметра требуется перезапуск сервера.

9 Выйти из psql.

```
\q
```

10 Перезапустить экземпляр сервера PostgreSQL, обслуживающий кластер БД main.

```
sudo pg_ctlcluster 11 main restart
```

11 Подключиться к кластеру БД main, БД postgres, используя роль sa.

```
psql -h localhost -d postgres -U sa
```

12 Вывести текущее значение параметра max_connections.

```
SHOW max_connections;
```

Значение параметра изменится.

Изменение параметров конфигурации под учетной записью — суперпользователем PostgreSQL

Задание: Войти в ОС с учетной записью iivanov. При входе выбрать уровень конфиденциальности 3, 0:Категория 1, 1:Категория 2. Подключиться к кластеру БД main, БД postgres, используя роль iivanov. Изменить значение параметра maintenance_work_mem на 1MB, используя команду ALTER SYSTEM. Повторно считать конфигурацию кластера БД main через функцию pg_reload_conf(). Убедиться, что значение параметра изменилось. Вернуть значение параметра к значению, определенному по умолчанию. Убедиться, что значение параметра изменилось.

1 Выйти из сеанса ОС и войти с учетной записью iivanov. При входе выбрать уровень конфиденциальности 3, 0:Категория 1, 1:Категория 2.

2 Подключиться через утилиту psql к базе postgres от имени пользователя iivanov.

```
psql -h localhost -d postgres -U iivanov
```

Подключение пройдет успешно.

3 Изменить значение параметра maintenance_work_mem на `'1MB'`, используя команду ALTER SYSTEM:

```
ALTER SYSTEM SET maintenance_work_mem = '1MB';
```

Примечание: параметр maintenance_work_mem в PostgreSQL определяет **максимальный объем** памяти, разрешенный для использования операциями обслуживания, такими как VACUUM, CREATE INDEX и другими. Изменение этого параметра влияет на производительность этих операций.

4 Повторно считать конфигурацию кластера БД main через функцию pg_reload_conf().

```
SELECT pg_reload_conf();
```

5 Вывести текущее значение параметра maintenance_work_mem.

```
SHOW maintenance_work_mem;
```

Убедиться, что значение было изменено.

6 Вернуть значение параметра к значению, определенному по умолчанию.

```
ALTER SYSTEM RESET maintenance_work_mem;
```

7 Повторно считать конфигурацию кластера БД main через функцию pg_reload_conf().

```
SELECT pg_reload_conf();
```

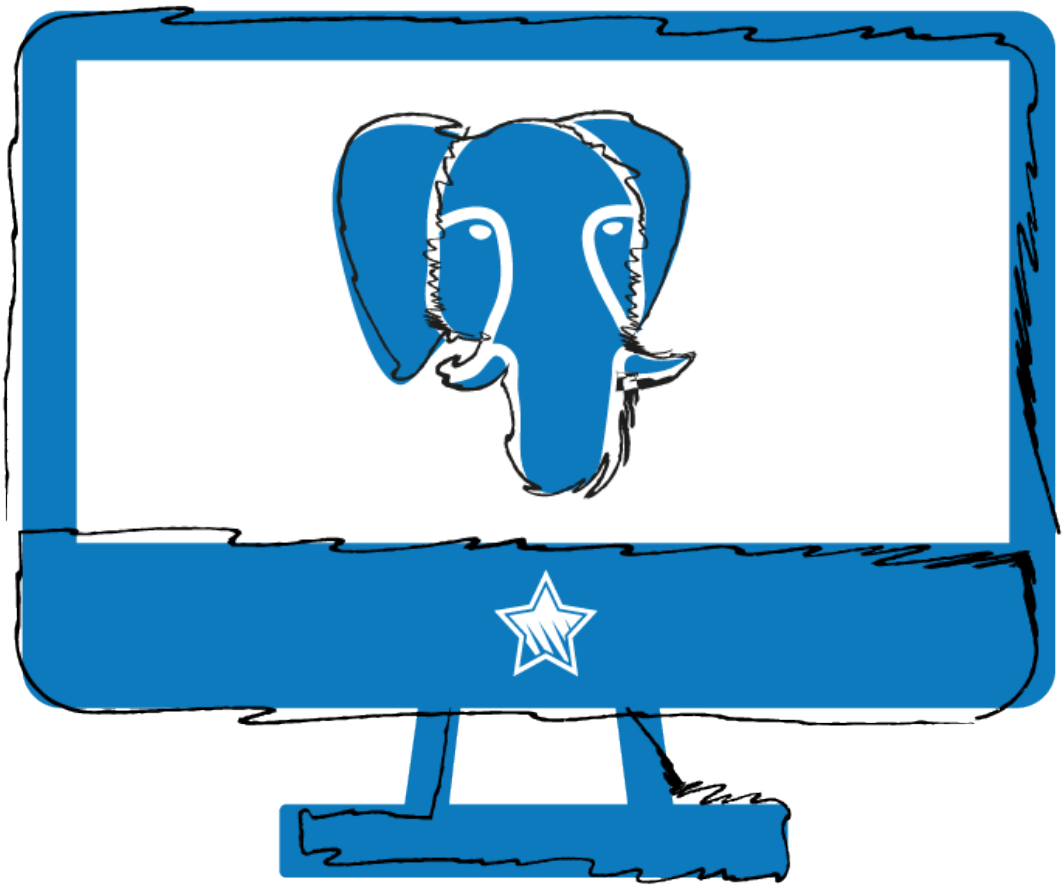
8 Вывести текущее значение параметра max_connections.

```
SHOW maintenance_work_mem;
```

Значение параметра вновь станет 64MB.

Модуль 2

Создание баз данных



ASTRA LINUX®

Модуль 2. Создание баз данных

Описание практического задания:

Оценка времени выполнения: 40 мин.

- Расширение шаблона template1 и создание БД на основе измененного шаблона.
- Переименование БД, изменение параметров БД, определение размера БД.
- Работа со схемами.
- Работа с табличными пространствами.
- Перенос данных и удаление ТП.

Расширение шаблона template1 и создание БД на основе измененного шаблона

Задание: Войти в ОС с учетной записью iivanov. При входе выбрать уровень конфиденциальности 3, 0:Категория 1, 1:Категория 2. Установить расширение pgcrypto в шаблон БД template1. На основе шаблона template1 создать БД new_db_from_template1. Проверить наличие функций из расширения pgcrypto в БД new_db_from_template1.

1 Войти в ОС с учетной записью iivanov. При входе выбрать уровень конфиденциальности 3, 0:Категория 1, 1:Категория 2.

2 Подключиться через утилиту psql к базе template1 от имени пользователя iivanov.

```
psql -h localhost -d template1 -U iivanov
```

Подключение пройдет успешно.

3 Проверить наличие функций генерации соли и хэширования пароля:

```
SELECT crypt('new password', gen_salt('md5'));
```

```
template1=# SELECT crypt('new password', gen_salt('md5'));
ОШИБКА:  функция gen_salt(unknown) не существует
СТРОКА 1: SELECT crypt('new password', gen_salt('md5'));
                                       ^
ПОДСКАЗКА:  Функция с данными именем и типами аргументов не найдена. Возможно,
 вам следует добавить явные приведения типов.
```

4 Посмотреть список расширений, доступных для установки.

```
SELECT * FROM pg_available_extensions;
\q
```

5 Установить расширение pgcrypto.

```
CREATE EXTENSION pgcrypto;
```

6 Проверить наличие функций генерации соли и хэширования пароля:

```
SELECT crypt('new password', gen_salt('md5'));
```

```
template1=# SELECT crypt('new password', gen_salt('md5'));
                crypt
-----------------------------------
 $1$YhCRGjJN$pBd8MGM2O.RYfR/DTCuw..
(1 row)
```

Соль – это случайные данные, которые используются в качестве дополнительного входа для хеш-функции, которая шифрует пароль.

Цель использования соли — предотвратить атаки, которые основываются на предварительном вычислении хешей для известных входных данных и последующем сравнении их с хешами, полученными из зашифрованных данных.

При использовании соли, даже если два пароля идентичны, их хеши будут отличаться, если соль разная.

7 Отключиться от шаблона template1, чтобы была возможность использовать его при создании БД — подключиться к БД postgres.

```
\c postgres
```

8 Создать БД new_db_from_template1.

```
CREATE DATABASE new_db_from_template1;
```

9 Просмотреть список всех БД кластера main.

```
\l
```

10 Подключиться к БД new_db_from_template1. Проверить наличие функций генерации соли и хэширования пароля.

```
\c new_db_from_template1
SELECT crypt('new password', gen_salt('md5'));
```

Функции будут присутствовать.

Переименование БД new_db_from_template1, изменение параметров БД, определение размера БД

Задание: Переименовать БД new_db_from_template1 в inventory. Установить максимальное количество возможных одновременных подключений к БД inventory в значение 100. Определить размер БД inventory. Вывести ответ в удобном человеку формате.

1 Подключиться к БД postgres, поскольку нельзя изменять имя текущей БД.

```
\c postgres
```

2 Изменить имя БД new_db_from_template1 на *inventory*.

```
ALTER DATABASE new_db_from_template1 RENAME TO inventory;
```

3 Просмотреть список всех БД кластера main. Убедиться, что БД new_db_from_template1 переименована в inventory.

```
\l
```

4 Отобразить максимальное количество возможных одновременных подключений к БД inventory:

```
SELECT datname AS db_name, datconnlimit AS connection_limit
FROM pg_database
WHERE datname = 'inventory';
```

Количество возможных одновременных подключений к БД inventory будет иметь значение -1. Т.е. количество не ограничено.

5 Изменить количество возможных одновременных подключений к БД inventory на 100:

```
ALTER DATABASE inventory CONNECTION LIMIT 100;
```

6 Повторно отобразить максимальное количество возможных одновременных подключений к БД inventory. Убедиться, что значение было изменено:

```
SELECT datname AS db_name, datconnlimit AS connection_limit
FROM pg_database
WHERE datname = 'inventory';
```

7 Определить размер БД inventory. Вывести ответ в удобном человеку формате.

```
SELECT pg_size_pretty(pg_database_size('inventory'));
```

8 Выйти из psql.

```
\q
```

Работа со схемами

Задание: Создать схему iivanov. Создать таблицу equipment, не указывая явным образом, к какой схеме она привязана.

Таблица должна включать следующие столбцы:

- `id int PRIMARY KEY,`
- `name VARCHAR (150),`
- `description VARCHAR (250).`

Создать схему supp_dept. Привязать таблицу equipment к схеме supp_dept.

1 Подключиться через утилиту psql к БД inventory кластера БД main от имени пользователя iivanov:

```
psql -h localhost -d inventory -U iivanov
```

2 Вывести текущее значение search_path.

```
SHOW search_path;
```

3 Вывести имена схем в пути поиска, включая схемы, добавляемые в него неявно можно с помощью функции current_schemas(boolean):

```
SELECT current_schemas(true);
```

4 Создать схему `iivanov`.

```
CREATE SCHEMA iivanov;
```

5 Создать таблицу *equipmen*t. Не указывать явным образом, к какой схеме она привязана:

```
CREATE TABLE equipment (
    id int PRIMARY KEY,
    name VARCHAR (150),
    description VARCHAR (250)
);
```

6 Определить в какой схеме была создана таблица equipment. Выполнить запрос:

```
SELECT
    table_schema as schema_name,
    table_name
FROM
    information_schema.tables
WHERE table_name = 'equipment';
```

7 Создать схему *supp_dept*. Привязать таблицу equipment к схеме supp_dept.

```
CREATE SCHEMA supp_dept;
ALTER TABLE iivanov.equipment SET SCHEMA supp_dept;
```

8 Определить к какой схеме привязана таблица supp_dept теперь.

```
SELECT
    table_schema as schema_name,
    table_name
FROM
    information_schema.tables
WHERE table_name = 'equipment';
```

Работа с табличными пространствами

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Выяснить, где размещается каталог данных для кластера main. Создать каталог /var/lib/postgresql/11/fastspace. Установить в качестве владельца каталога fastspace пользователя postgres. Установить на каталог /fastspace разрешения drwx---------. Создать ТП fastspace в каталоге fastspace. Перенести таблицу equipment в ТП fastspace.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться к БД postgres через утилиту psql от имени пользователя sa.

```
psql -h localhost -d postgres -U sa
```

3 Выяснить размещение каталога данных (значение PGDATA) для кластера БД main.

```
SHOW data_directory;
```

4 Вывести информацию о табличных пространствах, которые есть в кластере БД main на данный момент.

```
SELECT * FROM pg_tablespace;
```

5 Создать каталог `/var/lib/postgresql/11/fastspace`.

```
\q
sudo mkdir /var/lib/postgresql/11/fastspace
```

6 Установить в качестве владельца каталога /var/lib/postgresql/11/fastspace пользователя postgres:

```
sudo chown -R postgres:postgres
/var/lib/postgresql/11/fastspace
```

7 Установить на каталог /var/lib/postgresql/11/fastspace разрешения drwx---------:

```
sudo chmod 700 /var/lib/postgresql/11/fastspace
```

8 Просмотреть какие дискреционные права доступа выставлены на каталог /var/lib/postgresql/11/fastspace в результате конфигурирования. Обратить внимание, что присутствует информация и о мандатной метке каталога:

```
sudo pdp-ls -M  /var/lib/postgresql/11/
```

9 Подключиться через утилиту psql к БД inventory кластера БД main от имени пользователя sa.

```
psql -h localhost -d inventory -U sa
```

10 Создать ТП fastspace в каталоге /var/lib/postgresql/11/fastspace.

```
CREATE TABLESPACE fastspace LOCATION
'/var/lib/postgresql/11/fastspace';
```

11 Вывести информацию о табличных пространствах, проверить, что ТП fastspace было создано.

```
SELECT * FROM pg_tablespace;
```

12 Перенести таблицу equipment в ТП fastspace.

```
ALTER TABLE supp_dept.equipment
SET TABLESPACE fastspace;
```

13 Проверить, что таблица equipment теперь размещена в табличном пространстве fastspace:

```
SELECT tablespace
FROM pg_tables
WHERE tablename = 'equipment';
```

14 Просмотреть размер табличного пространства fastspace через функцию pg_tablespace_size. Вывести результат в удобном человеку формате.

```
SELECT pg_size_pretty(pg_tablespace_size('fastspace'));
```

15 Вывести информацию о табличных пространствах через команду psql \db+.

```
\db+
```

16 Вывести информацию о размере подкаталогов в каталоге /var/lib/postgresql/11/fastspace:

```
\q
sudo du -h /var/lib/postgresql/11/fastspace/
```

Перенос данных и удаление ТП

Задание: Перенести таблицу equipment в ТП pg_default. Удалить ТП fastspace.

1 Подключиться к БД inventory через утилиту psql от имени пользователя sa.

```
psql -h localhost -d inventory -U sa
```

2 Перенести таблицу equipment в ТП pg_default:

```
ALTER TABLE supp_dept.equipment
SET TABLESPACE pg_default;
```

3 Удалить ТП fastspace:

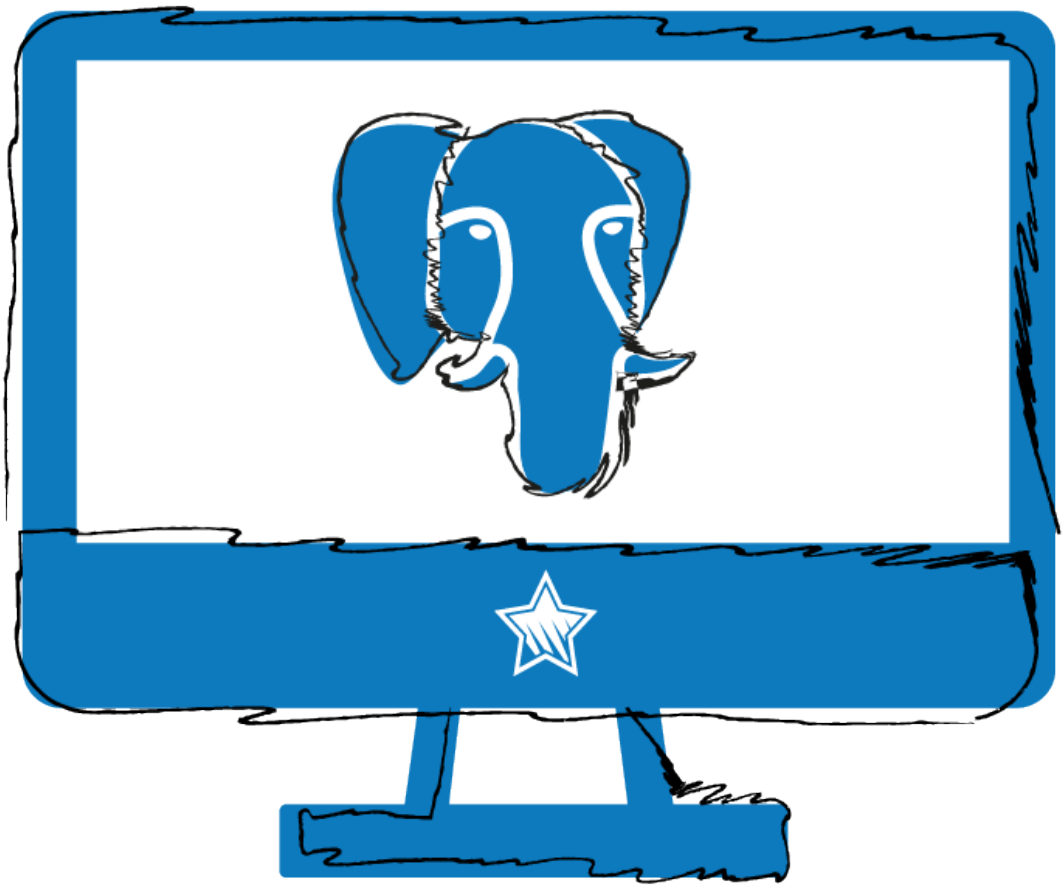
```
DROP TABLESPACE fastspace;
```

4 Выйти из psql.

```
\q
```

Модуль 3

Подключение и аутентификация



ASTRA LINUX®

Модуль 3. Подключение и аутентификация

Описание практического задания:

Оценка времени выполнения: 50 мин.

- Хранение паролей в СУБД PostgreSQL.
- Влияние конфигурации в файле /etc/parsec/mswitch.conf на возможность подключения к PostgreSQL.
- Настройка авторизации в файле pg_hba.conf.
- Создание файла с паролями ~/.pgpass на ОС пользователя.

Хранение паролей в СУБД PostgreSQL

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Создать роли bborisov, ggerasimov в кластере БД main. В качестве атрибутов и привилегий роли указать LOGIN, CREATEDB, REPLICATION, PASSWORD. Написать запрос на выборку к таблице pg_authid системного каталога, вывести значения в стобцах: rolname, rolsuper, rolcanlogin, rolpassword. Вывести из таблицы данные данные о ролях bborisov, ggerasimov.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться к БД postgres через утилиту psql от имени пользователя sa.

```
psql -h localhost -d postgres -U sa
```

3 Создать роль *bborisov*. В качестве атрибутов и привилегий роли указать LOGIN, CREATEDB, REPLICATION, PASSWORD:

```
CREATE ROLE bborisov WITH LOGIN CREATEDB REPLICATION
PASSWORD 'P@ssw0rd';
```

4 Создать роль *ggerasimov*. В качестве атрибутов и привилегий роли указать LOGIN, CREATEDB, REPLICATION, PASSWORD:

```
CREATE ROLE ggerasimov WITH LOGIN CREATEDB REPLICATION
PASSWORD 'P@ssw0rd';
```

5 Проверить, что роли созданы.

```
\du
```

6 Написать запрос на выборку к таблице pg_authid системного каталога, вывести значения в столбцах: rolname, rolsuper, rolcanlogin, rolpassword. Вывести данные о ролях bborisov, ggerasimov:

```
SELECT rolname, rolsuper, rolcanlogin, rolpassword FROM
pg_authid WHERE rolname='ggerasimov' OR rolname='bborisov';
```

```
postgres=# SELECT rolname, rolsuper, rolcanlogin, rolpassword FROM pg_authid
postgres-# WHERE rolname='ggerasimov' OR rolname='bborisov';
  rolname   | rolsuper | rolcanlogin |             rolpassword
------------+----------+-------------+-------------------------------------
 bborisov   | f        | t           | md5ddfa08c070b25f897829820df3189298
 ggerasimov | f        | t           | md53276d4ab8495ce91a698e8443c652dce
(2 строки)
```

Влияние конфигурации в файле /etc/parsec/mswitch.conf на возможность подключения к PostgreSQL

Задание: Попытаться подключиться к базе postgres от имени роли bborisov. В ОС создать пользователя bborisov. Задать ему пароль, уровень конфиденциальности: минимальный – 0, максимальный – 0. Указать, что пользователь имеет доступ к информации с категориями конфиденциальности: 0 (без категории).
При создании указать, что должна быть создана домашняя директория.
В качестве оболочки для пользователя указать /bin/bash. Выполнить подключение к БД postgres.

Попытаться подключиться к базе postgres от имени роли ggerasimov. Изменить значение параметра *zero_if_notfound на yes* в файле /etc/parsec/mswitch.conf. Подключиться к базе postgres от имени роли ggerasimov. Сделать выводы.

1 Попытаться подключиться к базе postgres от имени роли bborisov.

```
\q
psql -h localhost -d postgres -U bborisov
```

При текущих настройках возникнет ошибка, поскольку одноименного пользователя в операционной системе нет, и для него не определены мандатные атрибуты.

```
sa@astra:~$ psql -h localhost -d postgres -U bborisov
Пароль пользователя bborisov:
psql: ВАЖНО:  error obtaining MAC configuration for user "bborisov"
```

2 Создать в ОС пользователя bborisov.

Задать ему пароль (например, P@ssw0rd) и уровень конфиденциальности.

Минимальный и максимальный уровень конфиденциальности установить в значение 0:

```
sudo useradd -m -s /bin/bash bborisov
sudo passwd bborisov
sudo pdpl-user -l 0:0 bborisov
```

3 Подключиться через утилиту psql к БД postgres от имени пользователя bborisov:

```
psql -h localhost -d postgres -U bborisov
```

Подключение пройдет успешно.

4 Ввести команду `\conninfo` для получения информации о подключении.

```
\conninfo
```

```
postgres=> \conninfo
Вы подключены к базе данных "postgres" как пользователь "bborisov" (сервер "localhost", порт "5432").
SSL-соединение (протокол: TLSv1.3, шифр: TLS_AES_256_GCM_SHA384, бит: 256, сжатие: выкл.)
```

5 Попытаться подключиться к базе postgres от имени роли ggerasimov.

```
\q
psql -h localhost -d postgres -U ggerasimov
```

При текущих настройках возникнет ошибка, поскольку одноименного пользователя в операционной системе нет, и для него не определены мандатные атрибуты.

6 Изменить значение параметра zero_if_notfound на yes в файле /etc/parsec/mswitch.conf.

```
sudo mcedit /etc/parsec/mswitch.conf
Изменить значение параметра zero_if_notfound на yes.
Нажать F10 для выхода. При выходе сохранить изменения.
```

7 Попытаться подключиться к базе postgres от имени роли ggerasimov.

```
psql -h localhost -d postgres -U ggerasimov
```

При текущих настройках ошибка не возникнет, несмотря на то, что одноименного пользователя в операционной системе нет, и для него не определены мандатные атрибуты. Но, таким образом, есть возможность подключиться к PostgreSQL только с уровнем конфиденциальности 0 и без доступа к информации с установленной категорией конфиденциальности.

Настройка авторизации в файле pg_hba.conf

Задание: Запретить пользователю bborisov подключение по TCP, IPv4 с адреса 127.0.0.1 к кластеру БД main.

1 Внести в файл pg_hba.conf запись, запрещающую пользователю bborisov подключение по TCP, IPv4 с адреса 127.0.0.1.

```
\q
```

Внести в раздел #IPv4 local connections файла /etc/postgresql/11/main/pg_hba.conf строку:
host all bborisov 127.0.0.1/32 reject

```
cd /etc/postgresql/11/main/
sudo mcedit pg_hba.conf
Внести строку.
Закрыть файл, сохранив изменения.
```

```
# IPv4 local connections:
#host    all             all             127.0.0.1/32            md5
host    all             bborisov        127.0.0.1/32            reject
host    all             all             0.0.0.0/0               md5
```

2 Повторно считать конфигурация кластера БД main.

```
sudo pg_ctlcluster 11 main reload
```

1 Попытаться подключиться к базе postgres от имени роли bborisov.

```
psql -h localhost -d postgres -U bborisov
```

Подключение будет отвергнуто.

```
sa@astra:/etc/postgresql/11/main$ psql -h localhost -d postgres -U bborisov
psql: ВАЖНО:  pg_hba.conf отвергает подключение: компьютер "127.0.0.1", пользователь "bbori
sov", база данных "postgres", SSL вкл.
ВАЖНО:  pg_hba.conf отвергает подключение: компьютер "127.0.0.1", пользователь "bborisov",
база данных "postgres", SSL выкл.
```

2 Попытаться подключиться к базе postgres от имени роли ggerasimov.

```
psql -h localhost -d postgres -U ggerasimov
```

Подключение пройдет успешно.

Создание файла с паролями ~/.pgpass в ОС пользователя

Задание: В ОС создать пользователя ggerasimov. Задать ему пароль. Подключиться к базе postgres от имени роли ggerasimov, введя пароль. Войти в ОС от имени пользователя ggerasimov. В домашнем каталоге создать файл *.pgpass*. Внести в файл .pgpass запись:

```
localhost:5432:*:ggerasimov:P@ssw0rd
```

Установить на файл .pgpass доступ на чтение и запись только для пользователя ggerasimov. Подключиться к базе postgres от имени роли ggerasimov, не вводя пароль.

1 Создать в ОС пользователя ggerasimov.

Задать ему пароль (например, P@ssw0rd):

```
\q
```

```
sudo useradd -m -s /bin/bash ggerasimov
sudo passwd ggerasimov
```

2 Выйти из ОС. Войти в ОС от имени пользователя ggerasimov.

3 Подключиться к базе postgres от имени роли ggerasimov.

```
psql -h localhost -d postgres -U ggerasimov
```

Обратите внимание, что был запрошен пароль.

4 В домашнем каталоге пользователя ggerasimov создать файл .pgpass.

```
\q
touch .pgpass
```

5 Внести в файл .pgpass запись:

```
localhost:5432:*:ggerasimov:P@ssw0rd
```

```
mcedit .pgpass
Внести запись.
Закрыть файл, сохранив изменения.
```

6 Установить на файл .pgpass доступ только для пользователя ggerasimov.

Выполните команду:

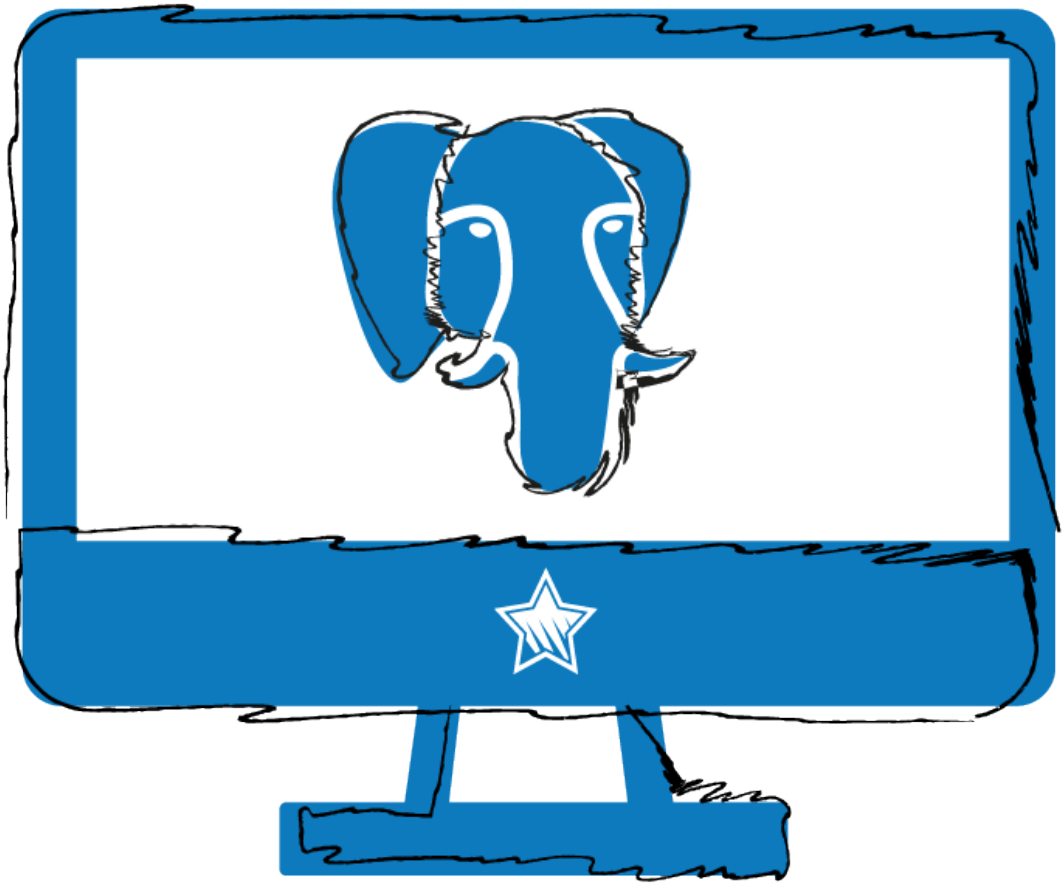
```
chmod 600 .pgpass
```

7 Подключиться к базе postgres от имени роли ggerasimov.

```
psql -h localhost -d postgres -U ggerasimov
```

Подключение будет выполнено без запроса пароля.

Модуль 4
Роли в СУБД PostgreSQL



ASTRA LINUX®

Модуль 4. Роли в СУБД PostgreSQL

Описание практического задания:

Оценка времени выполнения: 15 мин.

- Создание суперпользователя.
- Создание групповой роли.
- Создание ролей для пользователей и включение их в групповую роль.
- Включение ролей для пользователей в групповую роль.

Создание суперпользователя

Задание: Создать роль суперпользователя zzakharov в кластере БД main. В ОС создать пользователя zzakharov. Задать ему пароль, уровень конфиденциальности: минимальный – 0, максимальный – 0. При создании указать, что должна быть создана домашняя директория. В качестве оболочки для пользователя указать /bin/bash. Выполнить подключение к БД postgres от имени пользователя zzakharov. Создать базу данных zzakharov_data. Определить, какая роль является владельцем БД zzakharov_data.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться через утилиту psql к БД postgres от имени пользователя sa.

```
psql -h localhost -d postgres -U sa
```

3 Создать роль *zzakharov*. В качестве атрибутов и привилегий роли указать LOGIN, SUPERUSER, PASSWORD.

```
CREATE ROLE zzakharov WITH LOGIN SUPERUSER PASSWORD
'P@ssw0rd';
```

4 Проверить, что роль была создана и обладает привилегиями суперпользователя.

```
\du
```

5 Создать в ОС пользователя zzakharov.

Задать ему пароль (например, P@ssw0rd) и уровень конфиденциальности.

Минимальный и максимальный уровень конфиденциальности установить в значение 0:

```
\q
sudo useradd -m -s /bin/bash zzakharov
```

```
sudo passwd zzakharov
sudo pdpl-user -l 0:0 zzakharov
```

6 Подключиться к базе postgres от имени роли zzakharov.

```
psql -h localhost -d postgres -U zzakharov
```

7 Создать базу данных zzakharov_data.

```
CREATE DATABASE zzakharov_data;
```

8 Вывести информацию о владельцах БД в кластере main.

Определить, какая роль является владельцем БД zzakharov_data:

```
SELECT d.datname as "Database Name",
       pg_catalog.pg_get_userbyid(d.datdba) as "Database
Owner"
FROM pg_catalog.pg_database d
WHERE  d.datname='zzakharov_data';
```

Создание групповой роли

Задание: Создать групповую роль tech_support в кластере БД main.

1 Создать роль *tech_support*.

```
CREATE ROLE tech_support;
```

2 Проверить, что роль была создана и не обладает привилегией LOGIN (Вход запрещён).

```
\du
```

Создание ролей для пользователей

Задание: Создать роли oolegov, aantonov в кластере БД main. В качестве атрибутов и привилегий роли oolegov указать LOGIN, PASSWORD. В качестве атрибутов и привилегий роли aantonov указать LOGIN, NOINHERIT, PASSWORD.

1 Создать роль *oolegov*. В качестве атрибутов и привилегий роли указать LOGIN, PASSWORD.

```
CREATE ROLE oolegov WITH LOGIN PASSWORD 'P@ssw0rd';
```

2 Создать роль *aantonov*. В качестве атрибутов и привилегий роли указать LOGIN, NOINHERIT, PASSWORD.

```
CREATE ROLE aantonov WITH LOGIN NOINHERIT PASSWORD
'P@ssw0rd';
```

3 Проверить, что роли были созданы с нужными атрибутами.

```
\du
```

Включение ролей для пользователей в групповую роль

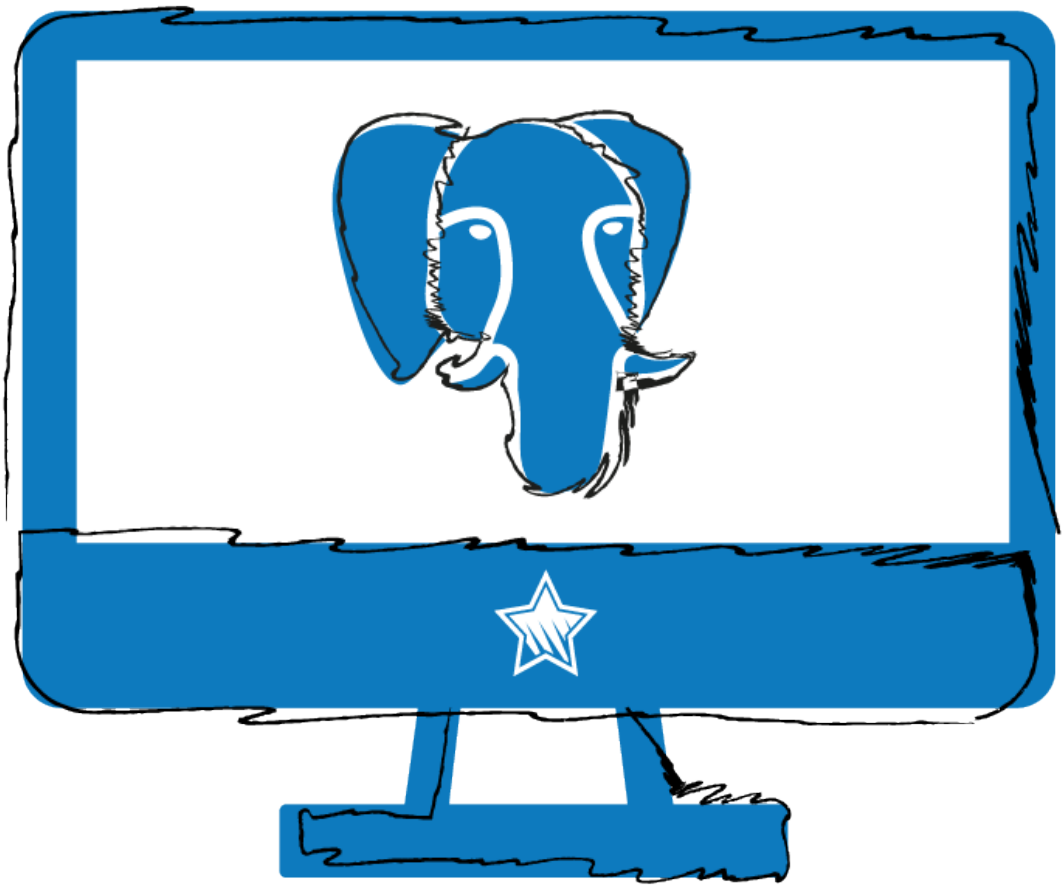
Задание: Включить роли oolegov, aantonov в групповую роль tech_support в кластере БД main.

1 Включить роли oolegov, aantonov в групповую роль tech_support.

```
GRANT tech_support TO oolegov, aantonov;
\q
```

Модуль 5

Дискреционное управление доступом в СУБД PostgreSQL в Astra Linux Special Edition



ASTRA LINUX®

Модуль 5. Дискреционное управление доступом в СУБД PostgreSQL в Astra Linux Special Edition

Описание практического задания:

Оценка времени выполнения: 50 мин.

- Автоматизированное тестирование функциональных возможностей по разграничению доступа в PostgreSQL с помощью пакета postgresql-se-test-11.
- Предоставление привилегий групповой роли tech_support на схему supp_dept и на таблицу equipment.
- Проверка привилегий на таблицу equipment для роли oolegov.
- Проверка привилегий на таблицу equipment для роли aantonov.

Автоматизированное тестирование функциональных возможностей по разграничению доступа в PostgreSQL с помощью пакета postgresql-se-test-11

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Установить пакет postgresql-se-test-11. Запустить тесты. Ознакомиться с результатами выполнения тестов. Открыть один из файлов результатов.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Установить пакет *postgresql-se-test-11*.

```
sudo apt -y install postgresql-se-test-11
```

3 Перейти в каталог с тестами и запустить их.

```
cd /usr/share/postgresql/11/test/pgacext/
sudo ./runtests -all
```

4 Результаты выполнения тестов будут сохранены в каталоге /home/sa/pg-11-se-test.

Для просмотра результатов тестирования нужно дождаться его завершения. Пока выполняется тестирование можно продолжить выполнение практического задания, используя другую вкладку терминала.

5 Для ознакомления с результатами тестирования перейти в каталог */home/sa/pg-11-se-test/results* и просмотреть содержимое каталога.

```
cd /home/sa/pg-11-se-test/results
```

```
ls
```

6 Просмотреть один из файлов результатов, например, *mac-createtableas.out*. Ознакомиться с теми операциями, которые выполнялись в ходе теста:

```
cat mac-createtableas.out
```

Предоставление привилегий групповой роли tech_support на схему supp_dept и на таблицу equipment

Задание: Подключиться к БД inventory от имени роли zzakharov. Просмотреть список управления доступом для таблицы supp_dept.equipment. Предоставить групповой роли tech_support право на запись, чтение и изменение данных из таблицы supp_dept.equipment. Повторно просмотреть список управления доступом для таблицы supp_dept.equipment.

1 Подключиться к БД *inventory* от имени роли zzakharov.

```
psql -h localhost -d inventory -U zzakharov
```

2 Предоставить групповой роли *tech_support* право на использование схемы tech_support.

```
GRANT USAGE ON SCHEMA supp_dept TO tech_support;
```

3 Вывести список управления доступом для таблицы *supp_dept.equipment*.

```
\dp supp_dept.equipment
Список окажется пустым.
```

4 Предоставить групповой роли tech_support право на запись, чтение и изменение данных из таблицы supp_dept.equipment.

```
GRANT SELECT,INSERT,UPDATE ON TABLE supp_dept.equipment TO
tech_support;
```

5 Повторно вывести список управления доступом для таблицы supp_dept.equipment. Ознакомиться с результатами.

```
\dp supp_dept.equipment
```

Проверка привилегий на таблицу equipment для роли oolegov

Задание: Создать в ОС пользователя oolegov. Задать ему пароль (например, P@ssw0rd) и уровни доступа: минимальный уровень доступа установить в значение 0, максимальный уровень установить в значение 3. Предоставить доступ к категориям конфиденциальности 1, 2. Подключиться к БД inventory от имени роли oolegov. Выполнить вставку строки данных в таблицу supp_dept.equipment. В столбцы id, name, description. и secret_level_comment вставить значения: 1,'Intel Core i7-9700K', 'Восьмиядерный процессор'. Выбрать все записи из таблицы supp_dept.equipment.

1 Подключиться к БД *postgres* от имени роли *oolegov*.

```
\q
psql -h localhost -d postgres -U oolegov
```

Подключение пройдет успешно.

2 Попытаться подключиться к БД inventory.

```
\c inventory
```

```
postgres=> \c inventory
ВАЖНО:  база данных "inventory" не существует
Сохранено предыдущее подключение
postgres=>
```

3 Создать в ОС пользователя oolegov.

Задать ему пароль (например, P@ssw0rd) и уровень конфиденциальности.

Минимальный уровень конфиденциальности установить в значение 0, максимальный уровень конфиденциальности установить в значение 3.

Предоставить доступ к категориям конфиденциальности 1, 2:

```
\q
sudo useradd -m -s /bin/bash oolegov
sudo passwd oolegov
sudo pdpl-user -l 0:3 -c 0:3 oolegov
```

4 Попытаться подключиться к БД inventory повторно.

```
psql -h localhost -d inventory -U oolegov
```

Подключение пройдет успешно.

5 Выполнить вставку строки в таблицу supp_dept.equipment строку.

```
INSERT INTO supp_dept.equipment (id, name, description)
VALUES (1,'Intel Core i7-9700K','Восьмиядерный процессор');
```

6 Выбрать все записи из таблицы supp_dept.equipment.

```
SELECT * FROM supp_dept.equipment;
```

Проверка привилегий на таблицу equipment для роли aantonov

Задание: Создать в ОС пользователя aantonov. Задать ему пароль (например, P@ssw0rd) и уровни доступа: минимальный уровень доступа установить в значение 0, максимальный уровень установить в значение 3. Предоставить доступ к категориям конфиденциальности 1, 2.

Подключиться к БД inventory от имени роли aantonov. Попытаться выполнить вставку строки данных в таблицу supp_dept.equipment.

В столбцы id, name, description. и secret_level_comment вставить значения: 2,'Asus ROG STRIX Z390-E','Материнская плата'.

Изменить роль на групповую tech_support. Выполнить вставку строки данных в таблицу supp_dept.equipment (с теми же данными, что и в прошлой попытке). Выбрать все записи из таблицы supp_dept.equipment.

1 Создать в ОС пользователя aantonov.

Задать ему пароль (например, P@ssw0rd) и уровень конфиденциальности.

Минимальный уровень конфиденциальности установить в значение 0, максимальный уровень конфиденциальности установить в значение 3:

Предоставить доступ к категориям конфиденциальности 1, 2.

```
sudo useradd -m -s /bin/bash aantonov
sudo passwd aantonov
sudo pdpl-user -l 0:3 -c 0:3 aantonov
```

2 Подключиться к БД inventory от имени роли aantonov.

```
psql -h localhost -d inventory -U aantonov
```

3 Попытаться вставить строку в таблицу supp_dept.equipment.

```
INSERT INTO supp_dept.equipment (id, name, description)
VALUES (2,'Asus ROG STRIX Z390-E','Материнская плата');
```

4 Измените роль на групповую tech_support.

```
SET ROLE tech_support;
```

5 Выполнить вставку в таблицу supp_dept.equipment строку.

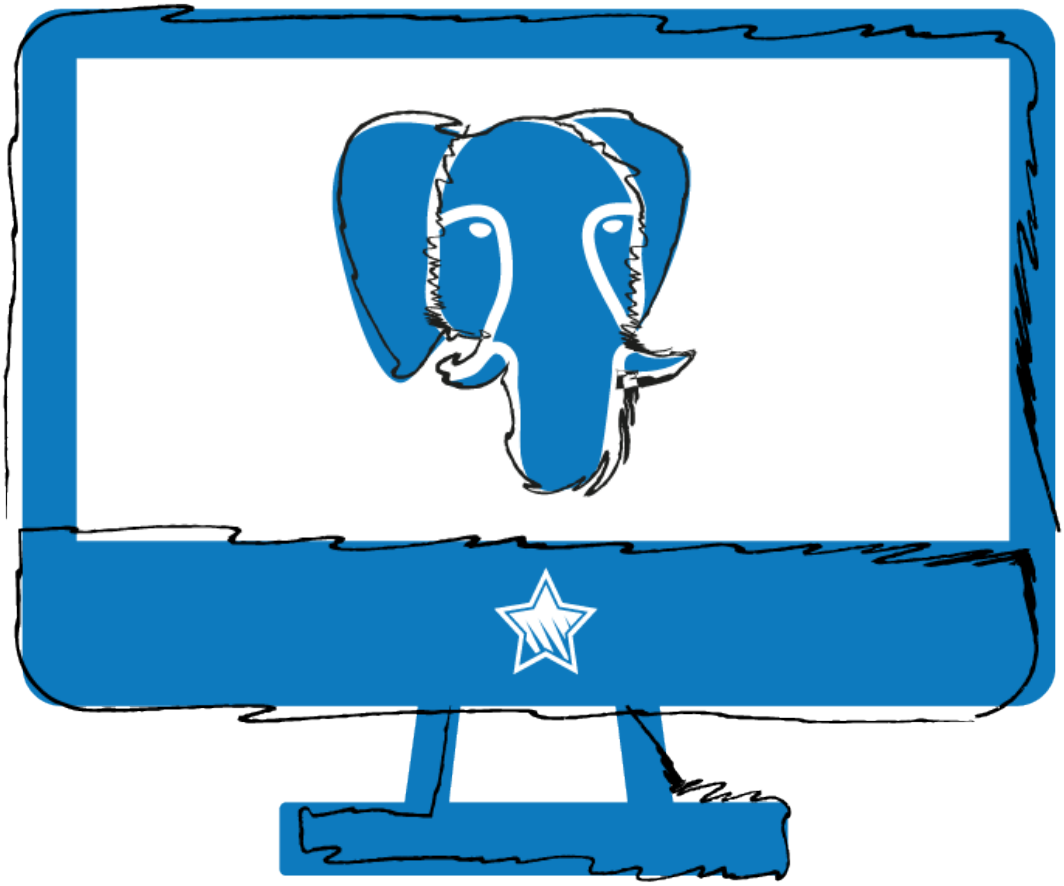
```
INSERT INTO supp_dept.equipment (id, name, description)
VALUES (2,'Asus ROG STRIX Z390-E','Материнская плата');
```

6 Выбрать все записи из таблицы supp_dept.equipment .

```
SELECT * FROM supp_dept.equipment;
```

Модуль 6

Мандатное управление доступом в СУБД PostgreSQL в Astra Linux Special Edition



ASTRA LINUX®

Модуль 6. Мандатное управление доступом в СУБД PostgreSQL в Astra Linux Special Edition

Описание практического задания:

Оценка времени выполнения: 120 мин.

- Просмотр и изменение уровней доступа пользователей ОС. Работа с файлами ОС.
- Создание пользователей в операционной системе и назначение им классификационных меток. Просмотр классификационных меток при подключении к PostgreSQL.
- Настройка кластера, базы и объектов БД для применения мандатного управления доступом в нужной конфигурации.
- Изменение таблицы supp_dept.equipment для поддержки защиты классификационными метками на уровне строк. Добавление столбцов для фиксации дополнительной информации.
- Вставка данных в таблицу supp_dept.equipment от имени пользователя zzakharov.
- Тестирование работы мандатного управления доступом от имени пользователя aantonov при значении конфигурационного параметра ac_ignore_socket_maclabel равном true.
- Изменение значения параметра ac_ignore_socket_maclabel на false.
- Тестирование работы мандатного управления доступом от имени пользователя aantonov при значении конфигурационного параметра ac_ignore_socket_maclabel равном false.
- Тестирование работы мандатного управления доступом от имени пользователя oolegov.

Часть 1. Мандатное управление доступом в ОС Astra Linux Special Edition

Просмотр и изменение уровней доступа пользователей ОС. Работа с файлами ОС

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Определить уровни доступа, заданные в ОС. Определить неиерархические категории, заданные в ОС. В терминале Fly проверить, в каком формате определены мандатные атрибуты пользователей. В графической утилите «Политика безопасности» для учетной записи пользователя zzakharov выбрать максимальный уровень доступа — «3:Уровень_3».

Войти в ОС с учётной записью пользователя zzakharov (уровень доступа – «Уровень 3», уровень целостности «Низкий»). В терминале Fly с помощью утилиты pdp-id отобразить мандатные атрибуты сеанса пользователя.
В домашнем каталоге создать файл *Уровень_3.txt*. С помощью утилиты pdp-ls вывести мандатный контекст безопасности файла Уровень_3.txt.

Выйти из сеанса ОС и войти в ОС с учётной записью пользователя zzakharov (уровень доступа – «Уровень 2», уровень целостности «Низкий»).

В терминале Fly с помощью утилиты pdp-id отобразить мандатные атрибуты сеанса пользователя. Отобразить файлы и каталоги в домашнем каталоге пользователя. Отобразить файлы и каталоги в домашнем каталоге пользователя.
В домашнем каталоге создать файл *Уровень_2.txt*. Внести в файл Уровень_2.txt запись «*Уровень_2*» и сохранить файл.

Выйти из сеанса ОС и войти в ОС с учётной записью пользователя zzakharov (уровень доступа – «Уровень 3», уровень целостности «Низкий»).

Вывести информацию о содержимом каталога /home/.pdp/zzakharov/l2i0c0x0t0x0. Вывести содержимое файла *Уровень_2.txt*. Попытаться внести в файл изменения и сохранить его.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Определить уровни доступа, заданные в ОС, для этого вывести в терминал Fly содержимое файла */etc/parsec/mac_levels*.

```
cat /etc/parsec/mac_levels
```

3 Определить неиерархические категории, заданные в ОС, для этого вывести в терминал Fly содержимое файла */etc/parsec/mac_categories*.

```
cat /etc/parsec/mac_categories
```

4 В терминале Fly проверить, в каком формате определены мандатные атрибуты пользователей. Например, просмотреть файл для пользователя sa. 1000 — это ID пользователя sa.

```
sudo cat /etc/parsec/macdb/1000
```

5 Запустить графическую утилиту «Политика безопасности».

6 Для учетной записи пользователя zzakharov выбрать максимальный уровень доступа — «3:Уровень_3».

7 Войти в ОС с учётной записью пользователя zzakharov (уровень доступа – «Уровень 3», уровень целостности «Низкий»).

8 В терминале Fly с помощью утилиты pdp-id отобразите мандатные атрибуты сеанса пользователя.

```
pdp-id
```

9 В домашнем каталоге создать файл *Уровень_3.txt*

```
touch Уровень_3.txt
```

10 С помощью утилиты pdp-ls вывести мандатный контекст безопасности файла Уровень_3.txt.

```
pdp-ls -M Уровень_3.txt
```

11 Выйти из сеанса ОС и войти в ОС с учётной записью пользователя zzakharov (уровень доступа – «Уровень 2», уровень целостности «Низкий»).

12 В терминале Fly с помощью утилиты pdp-id отобразить мандатные атрибуты сеанса пользователя.

```
pdp-id
```

13 Отобразить файлы и каталоги в домашнем каталоге пользователя.

```
ls
```

Есть ли среди них файл Уровень_3.txt?

```
Нет.
```

14 В домашнем каталоге создать файл *Уровень_2.txt*

```
touch Уровень_2.txt
```

15 С помощью утилиты pdp-ls выведите мандатный контекст безопасности файла Уровень_2.txt.

```
pdp-ls -M Уровень_2.txt
```

16 Внести в файл Уровень_2.txt запись «*Уровень_2*» и сохранить файл.

```
mcedit Уровень_2.txt
Внести текст, например, Уровень_2
F10, сохранить изменения.
```

17 Выйти из сеанса ОС и войти в ОС с учётной записью пользователя zzakharov (уровень доступа – «Уровень 3», уровень целостности «Низкий»).

18 Вывести информацию о содержимом каталога /home/.pdp/zzakharov/l2i0c0x0t0x0.

```
ls /home/.pdp/zzakharov/l2i0c0x0t0x0
```

Присутствует ли файл Уровень_2.txt в этом каталоге?

19 Вывести содержимое файла Уровень_2.txt.

```
cat /home/.pdp/zzakharov/l2i0c0x0t0x0/Уровень_2.txt
Файл доступен для чтения.
```

20 Попытаться внести в файл изменения и сохранить его.

```
mcedit /home/.pdp/zzakharov/l2i0c0x0t0x0/Уровень_2.txt
```

Удалось ли сохранить изменения?

```
Файл недоступен для редактирования. Потому
что редактирование файлов возможно, только если метка
конфиденциальности совпадает с меткой текущего мандатного
уровня пользователя. Но пользователь может скопировать
такой файл на свой уровень и после копирования
отредактировать. При копировании мандатная метка
автоматически меняется.
```

Часть 2. Мандатное управление доступом в СУБД PostgreSQL

Изменение классификационных меток пользователей. Просмотр классификационных меток при подключении к PostgreSQL

Задание: Изменить мандатные атрибуты пользователя oolegov.

Установить для него минимальный и максимальный уровень конфиденциальности в значение 0. Отключить доступ к данным с установленными категориями конфиденциальности. Подключиться через утилиту psql к БД postgres от имени пользователя oolegov и вывести значение классификационной метки. Выйти из утилиты psql.

Подключиться через утилиту psql к БД inventory от имени пользователя aantonov и вывести значение классификационной метки. Выйти из утилиты psql.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Установить для пользователя oolegov минимальный и максимальный уровень конфиденциальности в значение 0. Отключить доступ к данным с установленными категориями конфиденциальности:

```
sudo pdpl-user -l 0:0 -c 0:0 oolegov
```

3 Подключиться через утилиту psql к базе postgres от имени пользователя oolegov:

```
psql -h localhost -d postgres -U oolegov
```

Подключение пройдет успешно.

4 Просмотреть значение текущей классификационной метки сеанса пользователя:

```
SHOW ac_session_maclabel;
```

Значение уровня конфиденциальности будет равно нулю.

5 Выйти из утилиты psql.

```
\q
```

6 Подключиться к базе inventory через утилиту psql от имени пользователя aantonov:

```
psql -h localhost -d inventory -U aantonov
```

Подключение пройдет успешно.

7 Просмотреть значение текущей классификационной метки сеанса пользователя:

```
SHOW ac_session_maclabel;
```

Значение уровня конфиденциальности будет равно трём.

8 Выйти из утилиты psql.

```
\q
```

Настройка кластера, базы и объектов БД для применения мандатного управления доступом в нужной конфигурации

Задание: Подключиться к базе данных inventory через утилиту psql от имени пользователя zzakharov. Установить классификационные метки в значение {3,3} и мандатный признак CCR в значение OFF для кластера, табличного пространства pg_default, БД inventory, схемы supp_dept, таблицы equipment.

1 Подключиться к базе данных inventory через утилиту psql от имени пользователя zzakharov:

```
psql -h localhost -d inventory -U zzakharov
```

2 Изменить классификационную метку кластера:

```
MAC LABEL ON CLUSTER IS '{3,3}';
```

В результате выполнения команды изменится классификационная метка табличного пространства *pg_global*.

3 Установить мандатный признак CCR для кластера в значение **OFF**:

```
MAC CCR ON CLUSTER IS OFF;
```

4 Установить мандатный признак CCR для табличного пространства *pg_global в* значение **OFF**:

```
MAC CCR ON TABLESPACE pg_global IS OFF;
```

5 Изменить классификационную метку для табличного пространства *pg_default:*

```
MAC LABEL ON TABLESPACE pg_default IS '{3,3}';
```

6 Установить мандатный признак CCR для табличного пространства *pg_default* в значение **OFF**:

```
MAC CCR ON TABLESPACE pg_default IS OFF;
```

7 Изменить классификационную метку для БД inventory на {3,3}:

```
MAC LABEL ON DATABASE inventory IS '{3,3}';
```

8 Установить мандатный признак CCR для БД inventory в значение **OFF**:

```
MAC CCR ON DATABASE inventory IS OFF;
```

9 Изменить классификационную метку схемы *supp_dept*:

```
MAC LABEL ON SCHEMA supp_dept IS '{3,3}';
```

10 Установить мандатный признак CCR для схемы supp_dept в значение **OFF:**

```
MAC CCR ON SCHEMA supp_dept IS OFF;
```

11 Изменить классификационную метку таблицы *equipment*:

```
MAC LABEL ON TABLE supp_dept.equipment IS '{3,3}';
```

12 Установить мандатный признак CCR для таблицы *mactable* в значение **OFF**:

```
MAC CCR ON TABLE supp_dept.equipment IS OFF;
```

Изменение таблицы supp_dept.equipment для поддержки защиты классификационными метками на уровне строк. Добавление столбцов для фиксации дополнительной информации

Задание: Подключиться к кластеру через утилиту psql от имени пользователя zzakharov. Включить на таблице supp_dept.equipment защиту записей классификационными метками.

Изменить структуру таблицы *supp_dept.equipment* – добавить столбцы:

- secret_level_comment varchar (250),
- sess_user varchar (100);
- curr_user varchar (100).

Для столбца *sess_user* задайте значение по умолчанию – SESSION_USER,

для столбца *curr_user* – CURRENT_USER.

1 Подключиться к базе inventory через утилиту psql от имени пользователя zzakharov.

```
psql -h localhost -d inventory -U zzakharov
```

2 Включить на таблице supp_dept.equipment защиту записей классификационными метками.

```
ALTER TABLE supp_dept.equipment SET WITH MACS;
```

3 Добавить столбец secret_level_comment varchar (250) в таблицу supp_dept.equipment:

```
ALTER TABLE supp_dept.equipment ADD COLUMN
secret_level_comment varchar (150);
```

4 Добавить столбец sess_user varchar (100) в таблицу supp_dept.equipment:

```
ALTER TABLE supp_dept.equipment ADD COLUMN sess_user
varchar (100);
```

5 Добавить столбец curr_user varchar (100) в таблицу supp_dept.equipment:

```
ALTER TABLE supp_dept.equipment ADD COLUMN curr_user
varchar (100);
```

6 Для столбца таблицы supp_dept.equipment добавить значение по умолчанию SESSION_USER:

ALTER TABLE supp_dept.equipment ALTER COLUMN sess_user SET DEFAULT SESSION_USER;

7 Для столбца таблицы supp_dept.equipment добавить значение по умолчанию CURRENT_USER:

ALTER TABLE supp_dept.equipment ALTER COLUMN curr_user SET DEFAULT CURRENT_USER;

Вставка данных в таблицу supp_dept.equipment от имени пользователя zzakharov

Задание: Подключиться через утилиту psql к кластеру main, БД inventory под учетной записью zzakharov. Выполнить вставку строки данных в таблицу supp_dept.equipment БД inventory. В столбцы id, name, description, secret_level_comment вставить значения 3, 'Nvidia GeForce RTX 2080 Ti ', 'видеокарта', 'Уровень конфиденциальности 3'. Выбрать из таблицы все записи, а также значения классификационных меток этих записей.

Просмотреть значение атрибута (привилегии) ac_capable_chmac пользователя zzakharov.

Выполнить вставку строки данных в таблицу supp_dept.equipment. В столбцы id, name, description и secret_level_comment вставить значения: 4, 'NVIDIA GeForce RTX 3090', 'видеокарта', 'Уровень конфиденциальности 2'. Указать значение maclabel при вставке, равное '{2,0}'.

Выполнить вставку строки данных в таблицу supp_dept.equipment. В столбцы id, name, description и secret_level_comment вставить значения: 5, 'AMD Radeon RX 6800 XT', 'видеокарта', 'Уровень конфиденциальности 0'. Указать значение maclabel при вставке, равное '{0,0}'.

1 Подключиться к базе inventory через утилиту psql от имени пользователя zzakharov. Просмотреть значение текущей классификационной метки сеанса пользователя zzakharov.

```
psql -h localhost -d inventory -U zzakharov
SHOW ac_session_maclabel;
```

У пользователя zzakharov классификационная метка `{3,0}`.

2 Выполнить вставку строки данных в таблицу supp_dept.equipment:

```
INSERT INTO supp_dept.equipment (id, name, description,
secret_level_comment) VALUES (3, 'Nvidia GeForce RTX 2080
Ti', 'видеокарта', 'Уровень конфиденциальности 3');
```

3 Выбрать из таблицы все записи, а также значения классификационных меток этих записей:

```
SELECT maclabel, * FROM supp_dept.equipment;
```

4 Просмотреть значение атрибута (привилегии) ac_capable_chmac пользователя zzakharov:

```
SHOW ac_capable_chmac;
```

5 Выполнить вставку строк в таблицу supp_dept.equipment с явным указанием значения классификационной метки:

```
INSERT INTO supp_dept.equipment (maclabel, id, name,
description, secret_level_comment) VALUES ('{2,0}', 4,
'NVIDIA GeForce RTX 3090', 'видеокарта', 'Уровень
конфиденциальности 2');

INSERT INTO supp_dept.equipment (maclabel, id, name,
description, secret_level_comment) VALUES ('{0,0}', 5, 'AMD
Radeon RX 6800 XT', 'видеокарта', 'Уровень
конфиденциальности 0');
```

6 Выбрать из таблицы все записи, а также значения классификационных меток этих записей.

```
SELECT maclabel, * FROM supp_dept.equipment;
```

Тестирование работы мандатного управления доступом от имени пользователя aantonov при значении конфигурационного параметра ac_ignore_socket_maclabel равном true

Задание: Войти в операционную систему под учетной записью пользователя aantonov, выбрав уровень конфиденциальности 2.

Подключиться через утилиту psql к БД inventory под учетной записью aantonov.

Выбрать из таблицы все записи, а также значения классификационных меток этих записей.

1 Выйти из сеанса операционной системы и войти под учетной записью пользователя user_secret. При входе выбрать уровень конфиденциальности **2**.

2 Подключиться к БД inventory от имени пользователя aantonov:

```
psql -h localhost -d inventory -U aantonov
```

3 Просмотреть значение текущей классификационной метки сеанса пользователя aantonov:

```
SHOW ac_session_maclabel;
```

При подключении пользователя к БД уровень конфиденциальности был установлен в значении **3**. Есть доступ к категориям **1** и **2.**

4 Измените роль на групповую tech_support.

```
SET ROLE tech_support;
```

5 Выбрать из таблицы все записи, а также значения классификационных меток этих записей:

```
SELECT maclabel, * FROM supp_dept.equipment;
```

Обратите внимание на то, что, войдя в операционную систему с уровнем конфиденциальности 2, вы можете считать из таблицы БД записи с уровнем конфиденциальности 3.

6 Выйдите из сеанса операционной системы.

Изменение значения параметра ac_ignore_socket_maclabel на false

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Изменить значение параметра ac_ignore_socket_maclabel на false в файле postgresql.conf. Запустить команду для повторного считывания значений параметров, указанных в файле postgresql.conf.

7 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

8 Открыть файл postgresql.conf:

```
cd /etc/postgresql/11/main
sudo mcedit postgresql.conf
```

9 Изменить значение параметра ac_ignore_socket_maclabel на **false и** сохранить изменения при выходе.

10 Повторно считать параметры из файла конфигурации:

```
sudo pg_ctlcluster 11 main reload
```

Важно: при значении параметра ac_ignore_socket_maclabel, равном **true**, (устанавливается по умолчанию) при подключении к PostgreSQL не учитывается значение уровня конфиденциальности, с которым он вошел в операционную систему, а устанавливается максимальный уровень конфиденциальности, который может использовать пользователь, указанный в каталоге macdb в ОС.

Тестирование работы мандатного управления доступом от имени пользователя aantonov при значении конфигурационного параметра ac_ignore_socket_maclabel равном false

Задание: Войти в операционную систему под учетной записью пользователя aantonov, выбрав уровень конфиденциальности 2.

Подключиться через утилиту psql к БД inventory под учетной записью aantonov. Изменить роль на групповую tech_support.

Выполнить вставку строки данных в таблицу supp_dept.equipment БД inventory. В столбцы id, name, description и secret_level_comment вставить значения: 4, 'NVIDIA GeForce RTX 3060 Ti', 'Поддержка технологии RTX и DLSS', 'Уровень конфиденциальности 2'. Выбрать из таблицы все записи, а также значения классификационных меток этих записей.

1 Выйти из сеанса операционной системы и войти под учетной записью пользователя aantonov. При входе выбрать уровень конфиденциальности **2**.

2 Подключиться к БД inventory от имени пользователя aantonov:

```
psql -h localhost -d inventory -U aantonov
```

3 Просмотреть значение текущей классификационной метки сеанса пользователя user_secret:

```
SHOW ac_session_maclabel;
```

4 Изменить роль на групповую tech_support.

```
SET ROLE tech_support;
```

5 Выполнить вставку строки данных в таблицу `supp_dept.equipment`:

```
INSERT INTO supp_dept.equipment (id, name, description,
secret_level_comment) VALUES (6, 'NVIDIA GeForce RTX 3060
Ti', 'видеокарта', 'Уровень конфиденциальности 2');
```

6 Выбрать из таблицы все записи, а также значения классификационных меток этих записей:

```
SELECT maclabel, * FROM supp_dept.equipment;
```

Обратите внимание, что запрос возвратил записи с уровнем конфиденциальности **не выше** 2, для последней записи отличаются значения в столбцах sess_user и curr_user.

Тестирование работы мандатного управления доступом от имени пользователя oolegov

Задание: Войти в операционную систему под учетной записью пользователя oolegov. Подключиться через утилиту psql к БД inventory под учетной записью oolegov.

Выполнить вставку строки данных в таблицу supp_dept.equipment БД inventory.

В столбцы id, name, description и secret_level_comment вставить значения: 7, 'AMD Radeon RX 5700 XT', 'Видеокарта', 'Уровень конфиденциальности 0'. Выбрать из таблицы все записи, а также значения классификационных меток этих записей.

1 Выйти из сеанса операционной системы и войти под учетной записью пользователя oolegov.

2 Подключиться к базе от имени пользователя oolegov:

```
psql -h localhost -d inventory -U oolegov
```

3 Просмотреть значение текущей классификационной метки сеанса пользователя oolegov:

```
SHOW ac_session_maclabel;
```

4 Выполнить вставку строки данных в таблицу mactable:

```
INSERT INTO supp_dept.equipment (id, name, description,
secret_level_comment) VALUES (7, 'AMD Radeon RX 5700 XT',
'Видеокарта', 'Уровень конфиденциальности 0');
```

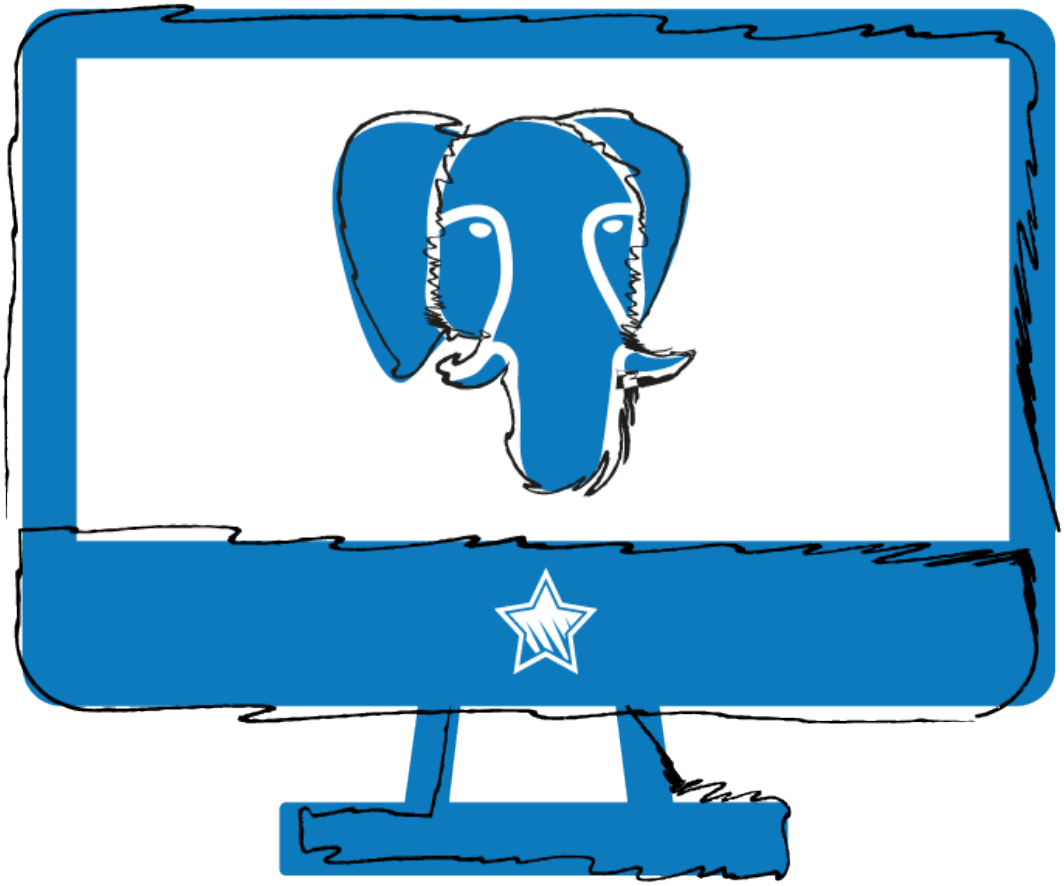
5 Выбрать из таблицы все записи, а также значения классификационных меток этих записей.

```
SELECT maclabel, * FROM supp_dept.equipment;
```

Обратите внимание, что записи с уровнем конфиденциальности выше 0 **не отобразились** в выборке.

Модуль 7

Журналирование и аудит доступа к данным



ASTRA LINUX®

Модуль 7. Журналирование и аудит доступа к данным

Описание практического задания:

Оценка времени выполнения: 45 мин.

- Просмотр текущего состояния настроек, журнала сообщений.
- Подключение к кластеру БД и просмотр записи о подключении в журнале сообщений.
- Изменение настроек аудита и фиксация запросов в журнале сообщений.
- Генерация событий средствами PostgreSQL.

Просмотр текущего состояния настроек, журнала сообщений

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Просмотреть текущие настройки в файле конфигурации postgresql.conf кластера БД main для параметров log_directory, log_destination, logging_collector, log_connections, log_line_prefix, log_rotation_age.

Просмотреть текущий журнал сообщений (за сегодня) для кластера БД main.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Просмотреть текущие настройки в файле конфигурации postgresql.conf кластера БД для параметров:

- log_directory
- log_destination
- logging_collector
- log_connections
- log_line_prefix
- log_rotation_age

```
cd /etc/postgresql/11/main
grep log_directory postgresql.conf
grep log_destination postgresql.conf
grep logging_collector postgresql.conf
grep log_connections postgresql.conf
grep log_line_prefix postgresql.conf
grep log_rotation_age postgresql.conf
```

3 Просмотреть текущий журнал (за сегодня) для кластера БД main.

```
cd /var/lib/postgresql/11
```

```
sudo ls -l main/pg_log/
sudo cat main/pg_log/postgresql-<день_недели>.log
```

Подключение к кластеру БД и просмотр записи о подключении в журнале сообщений

Задание: Подключиться к БД postgres кластера БД main от имени роли zzakharov через psql, затем отключиться. Найти в журнале сообщений записи, о подключении и отключении от кластера БД main пользователя zzakharov.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться к БД postgres кластера main от имени роли zzakharov через psql:

```
psql -h localhost -d postgres -U zzakharov
```

3 Выйти из psql.

```
/q
```

4 Найти в журнале сообщений записи о подключении и отключении от кластера БД main пользователя zzakharov.

```
sudo grep zzakharov main/pg_log/postgresql-
<день_недели>.log
```

В журнале будут присутствовать записи, содержащие информацию о:

- штампе времени,
- идентификаторе процесса,
- успешно или нет прошло действие,
- какое действие производилось,
- к какой БД производилось подключение, от имени какой роли, от имени какого пользователя, и какая классификационная метка использовалась при подключении.

```
sa@astra:/var/lib/postgresql/11$ sudo grep zzakharov main/pg_log/postgresql-Sat.log
2023-11-25 10:03:29 [696] AUDIT: SUCCESS, CONNECT, 127.0.0.1, "postgres", SU = "zzakharov" (16443), CU =
"zzakharov" (16443): maclabel {0,0} ignore_socket_maclabel
2023-11-25 10:04:03 [696] AUDIT: SUCCESS, DISCONNECT, 127.0.0.1, "postgres", SU = "zzakharov" (16443), CU
 = "zzakharov" (16443)
```

Изменение настроек аудита и фиксация запросов в журнале сообщений

Задание: Включить вывод всех команд и времени их выполнения, изменив настройку log_min_duration_statement=0. Выполнить выборку всех данных из таблицы supp_dept.equipment. Найти в журнале сообщений запись о выполнении запроса на выборку данных из таблицы supp_dept.equipment пользователем zzakharov.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться к БД postgres кластера main от имени роли iivanov через psql:

```
psql -h localhost -d postgres -U iivanov
```

3 Включить вывод всех команд и времени их выполнения, изменив настройку *log_min_duration_statement=0*. Для включения используйте команду ALTER SYSTEM.

```
ALTER SYSTEM SET log_min_duration_statement=0;
```

4 Повторно считайте конфигурацию через функцию *pg_reload_conf()*.

```
SELECT pg_reload_conf();
```

5 Подключиться к БД inventory кластера main от имени роли zzakharov через psql.

```
\q
psql -h localhost -d inventory -U zzakharov
```

6 Выполнить выборку всех данных из таблицы supp_dept.equipment.

```
SELECT * FROM supp_dept.equipment;
```

7 Найти в журнале сообщений запись о выполнении запроса на выборку данных из таблицы supp_dept.equipment пользователем zzakharov.

```
\q
cd /var/lib/postgresql/11
sudo grep zzakharov main/pg_log/postgresql-
<день_недели>.log
```

В журнале будет присутствовать запись о выполнении пользователем запроса на выборку.

```
2023-11-25 10:16:52.364 MSK [17854] zzakharov@inventory СООБЩЕНИЕ:  продолжительность: 12.039 мс, операто
p: SELECT * FROM supp_dept.equipment;
```

Использование расширенного аудита СУБД PostgreSQL в дистрибутиве Astra Linux Special Edition

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Просмотреть текущие настройки в файле конфигурации postgresql.conf кластера БД main для параметров ac_audit_mode. Подключиться к БД inventory кластера main от имени роли iivanov через psql. Просмотреть значение атрибута сеанса ac_session_audit. Выполните выборку данных из таблицы pg_db_role_settings. Назначить маску регистрации событий для пользователя iivanov {SsRVrawdDxtXUCTEMce:SsRVrawdDxtXUCTEMce}. Просмотреть значение атрибута сеанса ac_session_audit. Подключиться к БД inventory кластера main от имени роли iivanov через psql.

Выполнить следующую вставку данных в таблицу supp_dept.equipment:

```
INSERT INTO supp_dept.equipment (id, name, description) VALUES
(7,'Intel Core i7-9700K','Восьмиядерный процессор');
```

Просмотреть журнала аудита через графическую утилит «Системный журнал». Найти сообщение о вставке данных в таблицу пользователем iivanov в журнале аудита. Просмотреть журнал аудита через терминал. Найти сообщение о вставке данных в таблицу пользователем iivanov.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Просмотреть текущие настройки в файле конфигурации postgresql.conf кластера БД main для параметра ac_audit_mode.

```
cd /etc/postgresql/11/main
grep ac_audit_mode postgresql.conf
```

1 Подключиться к БД inventory кластера main от имени роли iivanov через psql.

```
psql -h localhost -d inventory -U iivanov
```

2 Просмотреть значение атрибута сеанса ac_session_audit.

```
SHOW ac_session_audit;
```

3 Выполните выборку данных из таблицы pg_db_role_settings.

```
SELECT * from pg_db_role_setting;
```

4 Назначить маску регистрации событий для пользователя iivanov {SsRVrawdDxtXUCTEMce:SsRVrawdDxtXUCTEMce}.

```
ALTER ROLE iivanov
SET ac_session_audit TO
'{SsRVrawdDxtXUCTEMce:SsRVrawdDxtXUCTEMce}';
```

5 Просмотреть значение атрибута сеанса ac_session_audit.

```
SHOW ac_session_audit;
```

6 Выполните выборку данных из таблицы pg_db_role_settings. Обратить внимание на изменения.

```
SELECT * from pg_db_role_setting;
```

7 Просмотреть настройки в файле /var/lib/postgresql/11/main/pg_audit.conf

```
\q
sudo cat /var/lib/postgresql/11/main/pg_audit.conf
```

В файле окажется не закомментирована одна запись:

success event mask = F0707 failure events mask = FFFFF

8 Подключиться к БД inventory кластера main от имени роли iivanov через psql.

```
psql -h localhost -d inventory -U iivanov
```

9 Выполнить вставку данных в таблицу supp_dept.equipment;

```
INSERT INTO supp_dept.equipment (id, name, description)
VALUES (8,'Intel Core i7-9700K','Восьмиядерный процессор');
\q
```

10 Для просмотра журнала аудита через графическую утилиту в терминале выполнить команду:

```
sudo ksystemlog
```

Будет запущена графическая утилита «Системный журнал». Утилиту также можно вызвать через меню ОС Системные/Просмотр системных журналов KsystemLog.

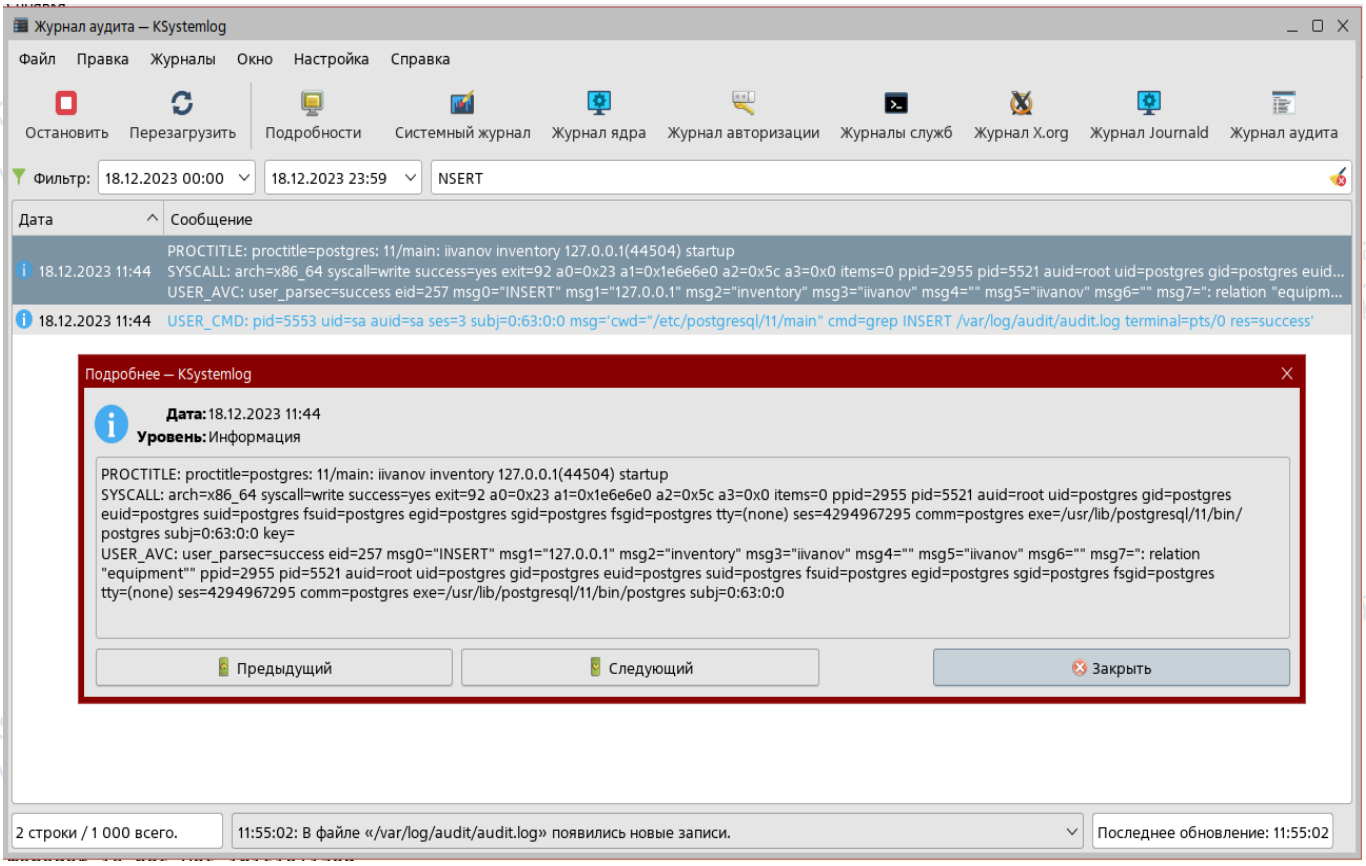
11 Найти сообщение о вставке данных в таблицу пользователем iivanov в журнале аудита.

```
Щелкнуть на кнопке «Журнал аудита».
В строке поиска можно указать, например,INSERT.
```

Среди последних сообщений в журнале аудита будет сообщение вида:

PROCTITLE: proctitle=postgres: 11/main: iivanov inventory 127.0.0.1(44504) startup SYSCALL: arch=x86_64 syscall=write success=yes exit=92 a0=0x23 a1=0x1e6e6e0 a2=0x5c a3=0x0 items=0 ppid=2955 pid=5521 auid=root uid=postgres gid=postgres euid=postgres suid=postgres fsuid=postgres egid=postgres sgid=postgres fsgid=postgres tty=(none) ses=4294967295 comm=postgres exe=/usr/lib/postgresql/11/bin/postgres subj=0:63:0:0 key=

USER_AVC: user_parsec=success eid=257 msg0="INSERT" msg1="127.0.0.1" msg2="inventory" msg3="iivanov" msg4="" msg5="iivanov" msg6="" msg7=": relation "equipment"" ppid=2955 pid=5521 auid=root uid=postgres gid=postgres euid=postgres suid=postgres fsuid=postgres egid=postgres sgid=postgres fsgid=postgres tty=(none) ses=4294967295 comm=postgres exe=/usr/lib/postgresql/11/bin/postgres subj=0:63:0:0



12 Просмотреть журнал аудита через терминал. Найти сообщение о вставке данных в таблицу пользователем iivanov:

```
sudo grep INSERT /var/log/audit/audit.log
```

```
sa@astra:~$ sudo grep INSERT /var/log/audit/audit.log
type=USER_AVC msg=audit(1702889055.006:825): user_parsec=success eid=257 msg0="INSERT" msg1="127.0.0.1
" msg2="inventory" msg3="iivanov" msg4="" msg5="iivanov" msg6="" msg7=": relation "equipment"" ppid=29
55 pid=5521 auid=4294967295 uid=115 gid=127 euid=115 suid=115 fsuid=115 egid=127 sgid=127 fsgid=127 tt
y=(none) ses=4294967295 comm="postgres" exe="/usr/lib/postgresql/11/bin/postgres" subj=0:63:0:0
```

13 Найти то же сообщение с помощью утилиты `ausearch`:

```
sudo ausearch -x postgres -i | grep INSERT
```

```
sa@astra:~$ sudo ausearch -x postgres -i | grep INSERT
type=USER_AVC msg=audit(18.12.2023 11:44:15.006:825) : user_parsec=success eid=257 msg0="INSERT" msg1=
"127.0.0.1" msg2="inventory" msg3="iivanov" msg4="" msg5="iivanov" msg6="" msg7=": relation " quipment
"" ppid=2955 pid=5521 auid=unset uid=postgres gid=postgres euid=postgres suid=postgres fsuid=postgres
egid=postgres sgid=postgres fsgid=postgres tty=(none) ses=unset comm=postgres exe=/usr/lib/postgresql/
11/bin/postgres subj=0:63:0:0
```

Генерация событий средствами PostgreSQL

Задание: Подключиться через утилиту psql к кластеру PostgreSQL под учетной записью iivanov. В БД sales создать таблицу *price* с включенной поддержкой мандатного управления доступом, включающую следующие столбцы:

- id int PRIMARY KEY,
- lot VARCHAR (50),
- price floatd,
- secret_level_comment varchar (150) NOT NULL,
- sess_user varchar (100) DEFAULT SESSION_USER,
- curr_user varchar (100) DEFAULT CURRENT_USER.

1 Подключиться к БД postgres кластера main от имени роли iivanov через psql.

```
psql -h localhost -d postgres -U iivanov
```

2 Создать БД sales:

```
CREATE DATABASE sales;
```

3 Подключиться к БД sales.

```
\c sales
```

4 Создать таблицу *price*:

```
CREATE TABLE price (
    id int PRIMARY KEY,
    lot VARCHAR (250),
    price float,
    secret_level_comment varchar (150) NOT NULL,
    sess_user varchar (100) DEFAULT SESSION_USER,
    curr_user varchar (100) DEFAULT CURRENT_USER
) WITH (MACS=TRUE);
```

5 Создать функцию, которую будет вызывать триггер:

```
CREATE FUNCTION notify_price_change()
  RETURNS trigger AS $$
  BEGIN
    NOTIFY "price_change";
    RETURN NEW;
  END;
$$ LANGUAGE plpgsql;
```

6 Создать триггер, который будет срабатывать при изменении таблицы:

```
CREATE TRIGGER price_change
AFTER UPDATE OR INSERT OR DELETE ON price
FOR EACH ROW
EXECUTE PROCEDURE notify_price_change();
```

7 Начать прослушивать канала *"price_change"*.

```
LISTEN "price_change";
```

8 Выполнить вставку данных в таблицу price:

```
INSERT INTO price (id, lot, price, secret_level_comment)
VALUES (1,'Вооружение 1', 257.5,'уровень конфиденциальности
3');
```

После выполнения запроса вы получите сообщение вида: *«Получено асинхронное уведомление "price_change" от серверного процесса с указанием его PID»*.

Создавая триггерную функцию, можно использовать функции:

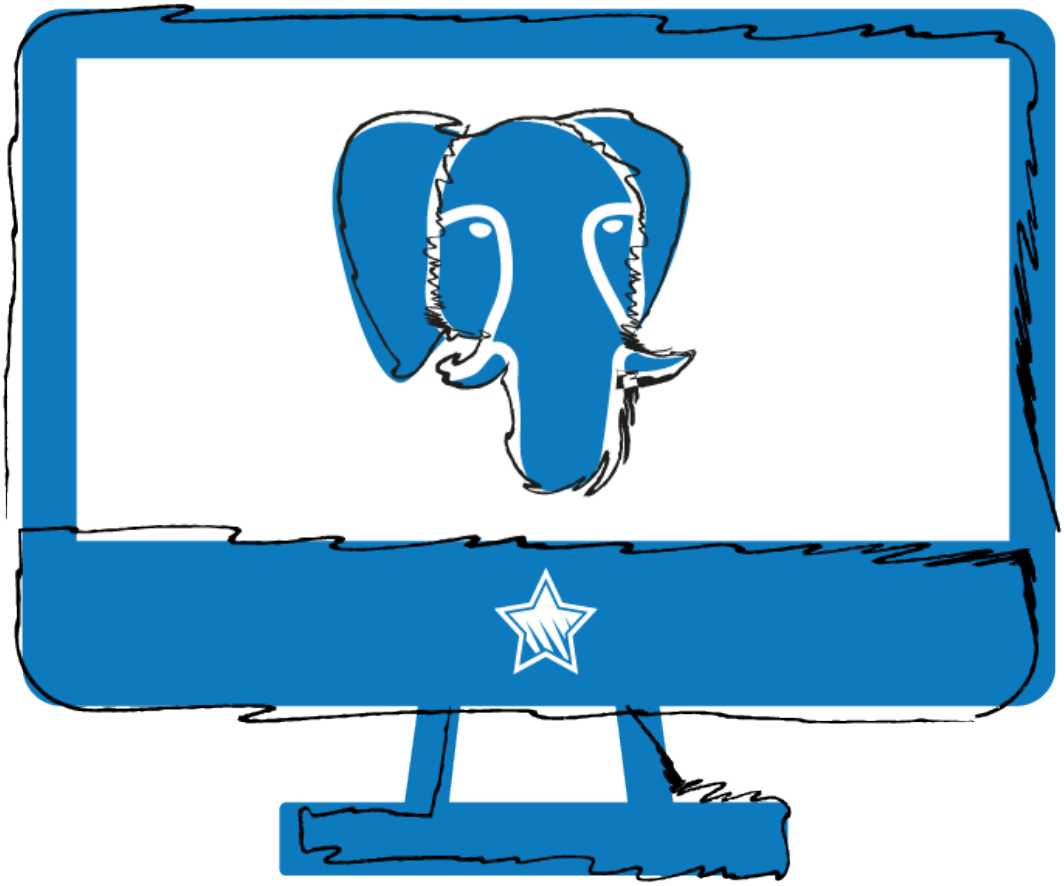
- TG_OP – строка, содержащая значение типа операции (INSERT, UPDATE, DELETE или TRUNCATE),
- OLD – старые данные изменяемой записи,
- NEW – новые данные изменяемой записи,

и добавлять в сообщение данные о старом и новом значении изменяемой записи.

Предполагается, что прослушивание канала будет осуществляться из **другого** сеанса, например, приложением.

Модуль 8

Логическое резервное копирование и восстановление



ASTRA LINUX®

Модуль 8. Логическое резервное копирование и восстановление

Описание практического задания:

Оценка времени выполнения: 80 мин.

- Создание пользователя для выполнения резервного копирования.
- Создание базы данных backup_db и таблицы test_copy.
- Выгрузка информации через команду /copy.
- Загрузка информации через команду /copy.
- Создание резервной копии таблицы из БД с помощью команды pg_dump в формате plain (SQL файл)
- Создание резервной копии БД с помощью команды pg_dump в формате plain (SQL файл).
- Создание резервной копии БД с помощью команды pg_dump в формате custom.
- Создание резервной копии БД с помощью команды pg_dump в формате tar.
- Создание резервной копии БД с помощью команды pg_dump в формате directory.
- Восстановление данных таблицы в БД backup_db_from_plain из резервной копии в формате plain (SQL файл).

Часть 1. Подготовка к резервному копированию, экспорт и импорт данных таблицы с помощью команды psql \copy

Создание пользователя для выполнения резервного копирования

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Создать роль ssidorov в кластере БД main, назначить ей атрибуты LOGIN CREATEDB REPLICATION PASSWORD. В ОС создать пользователя ssidorov. Задать ему пароль, уровень конфиденциальности: минимальный – 0, максимальный – 3. Указать, что пользователь имеет доступ к категориям информации: 0 (без категории), 1, 2. При создании указать, что должна быть создана домашняя директория. В качестве оболочки для пользователя указать /bin/bash. Выполнить подключение к БД postgres от имени пользоватля ssidorov.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться к БД postgres через утилиту psql от имени пользователя sa.

```
psql -h localhost -d postgres -U sa
```

3 Создать роль ssidorov, в контексте безопасности которой будет выполняться резервное копирование. В качестве атрибутов роли указать LOGIN, CREATEDB, REPLICATION, PASSWORD.

```
CREATE ROLE ssidorov WITH LOGIN CREATEDB REPLICATION
PASSWORD 'P@ssw0rd';
```

4 Проверить, что роль создана и выйти из psql.

```
\du
\q
```

5 Создать в ОС пользователя ssidorov.

Задать ему пароль (например, P@ssw0rd) и уровень конфиденциальности.

Минимальный уровень конфиденциальности установить в значение **0**, максимальный уровень конфиденциальности установить в значение **3**:

Указать, что пользователь имеет доступ к категориям **Категория_1, Категория_2.**

```
sudo useradd -m -s /bin/bash ssidorov
sudo passwd ssidorov
sudo pdpl-user -l 0:3 -c 0:3 ssidorov
```

6 Выйти из сеанса ОС и войти с учетной записью ssidorov. При входе выбрать уровень конфиденциальности 3, 0:Категория_1, 1:Категория_2.

7 Подключиться через утилиту psql к базе postgres от имени пользователя ssidorov:

```
psql -h localhost -d postgres -U ssidorov
```

Подключение пройдет успешно.

8 Просмотреть значение текущей классификационной метки сеанса пользователя:

SHOW ac_session_maclabel;

Создание базы данных backup_db и таблицы test_copy

Задание: Создать БД backup_db, схему ssidorov, проверить их классификационные метки.

В базе схеме ssidorov создать таблицу test_copy с включенной поддержкой мандатного управления доступом, включающую следующие столбцы:

- id int PRIMARY KEY,
- secret_level_comment varchar (150) NOT NULL,
- sess_user varchar (100) DEFAULT SESSION_USER,
- curr_user varchar (100) DEFAULT CURRENT_USER.

Проверить ее классификационную метку.

Выполнить вставку строку данных в таблицу test_copy. В столбцы id и secret_level_comment вставить значения: 1,'Уровень конфиденциальности 3, категория 1,2'. Выполнить вставку в таблицу test_copy строк данных, где в столбце secret_level_comment будет пустая строка – '' и нулевой указатель – **NULL.** Выполнить выборку из таблицы sidorov.test_copy. Вывести информацию о классификационных метках.

1 Создать БД backup_db.

```
CREATE DATABASE backup_db;
```

2 Написать запрос, выводящий информацию о классификационной метке и признаке CCR для БД backup_db:

```
SELECT datname,maclabel,datmacccr FROM pg_database
WHERE datname='backup_db';
```

3 Подключиться к БД backup_db и создать схему *sidorov*.

```
\connect backup_db
CREATE SCHEMA ssidorov;
```

4 Написать запрос, выводящий информацию о классификационной метке и признаке CCR для схемы ssidorov:

```
SELECT nspname, nspmacccr, maclabel FROM pg_namespace
WHERE nspname='ssidorov';
```

5 В схеме ssidorov создать таблицу *test_copy с* включенной поддержкой мандатного управления доступом, включающую следующие столбцы:

- id int PRIMARY KEY,
- secret_level_comment varchar (150),
- sess_user varchar (100) DEFAULT SESSION_USER,
- curr_user varchar (100) DEFAULT CURRENT_USER:

```
CREATE TABLE ssidorov.test_copy (
id int PRIMARY KEY,
secret_level_comment varchar (150),
```

```
sess_user varchar (100) DEFAULT SESSION_USER,
curr_user varchar (100) DEFAULT CURRENT_USER
) WITH (MACS=TRUE);
```

6 Написать запрос, выводящий информацию о классификационной метке и признаке CCR таблицы test_copy:

```
SELECT relname, relmacccr, maclabel FROM pg_class
WHERE relname='test_copy';
```

7 Выполнить вставку строки данных в таблицу test_copy:

```
INSERT INTO ssidorov.test_copy (id, secret_level_comment)
VALUES (1,'Уровень конфиденциальности 3, категория 1,2');
```

8 Выполнить вставку в таблицу test_copy строк данных, где в столбце secret_level_comment будет пустая строка – **''** и нулевой указатель – **NULL**:

```
INSERT INTO ssidorov.test_copy (id, secret_level_comment)
VALUES (2,'');
INSERT INTO ssidorov.test_copy (id, secret_level_comment)
VALUES (3,NULL);
```

9 Выполнить выборку из таблицы sidorov.test_copy. Вывести информацию о классификационных метках.

```
SELECT maclabel, * FROM ssidorov.test_copy;
```

Выгрузка информации через команду /copy

Задание: Вывести данные таблицы ssidorov.test_copy в стандартный поток вывода. Выгрузить данные таблицы ssidorov.test_copy в файл /home/ssidorov/tables_backup/test_copy.csv с помощью клиентской команды /copy, с разделителем ',' вывести строку заголовка с именами каждого столбца в файле. Выйти из psql. Просмотреть содержимое файла /home/ssidorov/tables_backup/test_copy.csv. Просмотреть, с каким уровнем конфиденциальности и категорией был создан файл /home/ssidorov/tables_backup/test_copy.csv.

1 Вывести данные таблицы ssidorov.test_copy в **стандартный** поток вывода.

```
COPY ssidorov.test_copy TO STDOUT;
```

Обратите внимание, что пустая строка и NULL отобразились **по-разному.**

2 Создать каталог /home/ssidorov/tables_backup.

```
\q
mkdir /home/ssidorov/tables_backup
```

3 Просмотреть, с каким уровнем конфиденциальности и категорией был создан каталог tables_backup.

```
pdp-ls -M /home/ssidorov/
```

При верном выполнении задания уровень конфиденциальности должен быть 3, категория 1, категория 2.

4 Подключиться через утилиту psql к базе backup_db от имени пользователя `ssidorov`.

```
psql -h localhost -d backup_db -U ssidorov
```

5 Выгрузить данные таблицы ssidorov.test_copy в файл test_copy.csv с помощью клиентской команды /copy, с разделителем ',' вывести строку заголовка с именами каждого столбца в файле:

```
\copy ssidorov.test_copy TO
/home/ssidorov/tables_backup/test_copy.csv DELIMITER ','
CSV HEADER;
```

6 Выйти из psql.

```
\q
```

7 Просмотреть содержимое файла /home/ssidorov/tables_backup/test_copy.csv.

```
cat /home/ssidorov/tables_backup/test_copy.csv
```

8 Просмотреть, с каким уровнем конфиденциальности и категорией был создан файл /home/ssidorov/tables_backup/test_copy.csv.

```
pdp-ls -M /home/ssidorov/tables_backup/test_copy.csv
```

При верном выполнении задания уровень конфиденциальности должен быть **3**, категория **1,** категория **2.**

Загрузка информации через команду /copy

Задание: Подключиться через утилиту psql к базе backup_db от имени пользователя ssidorov. В схеме ssidorov создать таблицу input_test с включенной поддержкой мандатного управления доступом, включающую следующие столбцы:

id int PRIMARY KEY,

secret_level_comment varchar (150),

sess_user varchar (100) DEFAULT SESSION_USER,

curr_user varchar (100) DEFAULT CURRENT_USER.

Загрузить данные из файла /home/ssidorov/tables_backup/test_copy.csv с помощью клиентской команды /copy в таблицу input_test. Выполнить выборку из таблицы ssidorov.input_test. Вывести информацию о классификационных метках.

1 Подключиться через утилиту psql к базе backup_db от имени пользователя ssidorov.

```
psql -h localhost -d backup_db -U ssidorov
```

2 В схеме ssidorov создать таблицу *input_test c* включенной поддержкой мандатного управления доступом, включающую следующие столбцы:

- id int PRIMARY KEY,
- secret_level_comment varchar (150),
- sess_user varchar (100) DEFAULT SESSION_USER,
- curr_user varchar (100) DEFAULT CURRENT_USER:

```
CREATE TABLE ssidorov.input_test (
id int PRIMARY KEY,
secret_level_comment varchar (150),
sess_user varchar (100) DEFAULT SESSION_USER,
curr_user varchar (100) DEFAULT CURRENT_USER
) WITH (MACS=TRUE);
```

3 Загрузить данные из файла /home/ssidorov/tables_backup/test_copy.csv с помощью клиентской команды /copy в таблицу input_test:

```
\copy ssidorov.input_test FROM
/home/ssidorov/tables_backup/test_copy.csv DELIMITER ','
CSV HEADER;
```

4 Выполнить выборку из таблицы ssidorov.input_test. Вывести информацию о классификационных метках.

```
SELECT maclabel, * FROM ssidorov.input_test;
```

Данные о классификационных метках были сформированы в момент вставки записей в таблицу ssidorov.input_test.

Часть 2. Создание резервных копий с помощью утилиты pg_dump в различных форматах. Восстановление данных.

Создание резервной копии таблицы из БД с помощью команды pg_dump в формате plain (SQL файл)

Задание: Войти в ОС под учетной записью sa, с нулевым уровнем конфиденциальности, с высоким уровнем целостности. Создать каталог /home/logical_backups. Установить на для каталога logical_backups мандатную метку с максимальным уровнем конфиденциальности 3, доступом к категориям: 0 (без категории), 1, атрибут CCNR.

Создать логическую резервную копию таблицы ssidorov.test_copy из БД backup_db в формате plain, сохранить в файле /home/logical_backups/backup_db_test_copy.sql, использовать ключ --create, к кластеру main обращаться от имени роли sa. Просмотреть, с каким уровнем и категорией конфиденциальности был создан файл backup_db_test_copy.sql.

1 Войти в ОС под учетной записью sa, с нулевым уровнем конфиденциальности, с высоким уровнем целостности.

2 Переключиться на **root**.

```
sudo -i
```

3 Создать каталог */home/logical_backups*.

```
mkdir /home/logical_backups
```

4 Установить на для каталога logical_backups мандатную метку с максимальным уровнем конфиденциальности **3**, доступом к категориям: **0** (без категории), 1, 2, атрибут **CCNR**:

```
cd /home
pdpl-file 3:0:3:CCNR logical_backups
```

5 Проверить установленную метку безопасности для каталога logical_backups:

```
pdp-ls -M
```

6 Создать логическую резервную копию таблицы ssidorov.test_copy из БД backup_db в plain формате, сохранить в файле /home/logical_backups/backup_db_test_copy.sql:

```
pg_dump -U sa -h localhost -d backup_db -t
ssidorov.test_copy -f
/home/logical_backups/backup_db_test_copy.sql
```

7 Просмотреть, с каким уровнем и категорией конфиденциальности был создан файл backup_db_test_copy.sql.

```
pdp-ls -M logical_backups
```

При верном выполнении задания уровень конфиденциальности должен быть 3, категории 1,2.

8 Просмотреть сценарий в файле backup_db_test_copy.sql.

```
cat /home/logical_backups/backup_db_test_copy.sql
```

Обратите внимание, что в сценарии присутствуют команды выставления классификационных меток и мандатного признака CCR.

Создание резервной копии БД с помощью команды pg_dump в формате plain (SQL файл)

Задание: Создать логическую резервную копию БД backup_db в формате plain, сохранить в файле /home/logical_backups/backup_db.sql, к кластеру main обращаться от имени роли sa. Просмотреть, с каким уровнем и категориями конфиденциальности был создан файл backup_db.sql. Просмотреть сценарий в файле backup_db.sql.

1 Создать логическую резервную копию БД backup_db в *plain* формате, сохранить в файле /home/logical_backups/backup_db.sql:

```
pg_dump -U sa -h localhost -d backup_db -f
/home/logical_backups/backup_db.sql
```

2 Просмотреть, с каким уровнем и категориями конфиденциальности был создан файл `backup_db.sql`.

```
pdp-ls -M /home/logical_backups
```

При верном выполнении задания уровень конфиденциальности должен быть **3**, категории **1,2**.

3 Просмотреть сценарий в файле backup_db.sql.

```
cat /home/logical_backups/backup_db.sql
```

Обратите внимание, что в сценарии присутствуют команды выставления классификационных меток и мандатного признака CCR.

Создание резервной копии БД с помощью команды pg_dump в формате custom

Задание: Создать логическую резервную копию БД backup_db в формате custom, сохранить в файле /home/logical_backups/custom, к кластеру main обращаться от имени роли sa. Просмотреть, с каким уровнем конфиденциальности и категорией был создан файл custom. Просмотреть оглавление файла резервной копии.

1 Создать логическую резервную копию БД backup_db в формате *custom,* сохранить в файле /home/logical_backups/custom:

```
pg_dump -U sa -h localhost -d backup_db -F c -f
/home/logical_backups/custom
```

2 Просмотреть, с каким уровнем конфиденциальности и категорией был создан файл custom.

```
pdp-ls -M logical_backups
```

При верном выполнении задания уровень конфиденциальности должен быть **3**, категория **1, 2**.

3 Просмотреть оглавление файла резервной копии:

```
pg_restore --list /home/logical_backups/custom
```

Создание резервной копии БД с помощью команды pg_dump в формате tar

Задание: Создать логическую резервную копию БД backup_db в формате tar, сохранить в файле /home/logical_backups/backup_db.tar, к кластеру main обращаться от имени роли sa. Просмотреть, с каким уровнем конфиденциальности и категорией был создан файл backup_db.tar.

1 Создать логическую резервную копию БД backup_db в формате *tar,* сохранить в файле /home/logical_backups/backup_db.tar:

```
pg_dump -U sa -h localhost -d backup_db -F t -f
/home/logical_backups/backup_db.tar
```

2 Просмотреть, с каким уровнем конфиденциальности и категорией был создан файл `backup_db.tar`

```
pdp-ls -M logical_backups
```

При верном выполнении задания уровень конфиденциальности должен быть **3**, категории **1,2.**

Создание резервной копии БД с помощью команды pg_dump в формате directory

Задание: Создать каталог /home/logical_backups/backup_db_dir. Установить для каталога backup_db_dir мандатную метку с максимальным уровнем конфиденциальности 3, доступом к категориям: 0 (без категории), 1, 2, атрибут CCNR. Создать логическую резервную копию БД backup_db в формате directory, сохранить в каталоге /home/logical_backups/backup_db_dir, к кластеру main обращаться от имени роли sa.

1 Создать каталог /home/logical_backups/backup_db_dir.

```
mkdir /home/logical_backups/backup_db_dir
```

2 Установить для каталога backup_db_dir мандатную метку с максимальным уровнем конфиденциальности **3**, доступом к категориям: 0 (без категории), 1, 2, атрибут CCNR:

```
pdpl-file 3:0:3:CCNR /home/logical_backups/backup_db_dir
```

3 Проверить установленную метку безопасности:

```
pdp-ls -M /home/logical_backups/
```

4 Создать логическую резервную копию БД backup_db в формате *directory*, сохранить в каталоге /home/logical_backups/backup_db_dir:

```
pg_dump -U sa -h localhost -d backup_db -F d -f
/home/logical_backups/backup_db_dir
```

5 Просмотреть, с каким уровнем и категорией конфиденциальности были созданы файлы внутри каталога /home/logical_backups/backup_db_dir.

```
cd /home/logical_backups/backup_db_dir
pdp-ls -M
```

При верном выполнении задания уровень конфиденциальности должен быть **3**, категории **1,2.**

Восстановление данных таблицы в БД backup_db_from_plain из резервной копии в формате plain (SQL файл)

Задание: Определить, какой порт использует кластер reserve. Подключиться к кластеру БД reserve под учетной записью postgres. Создать роль sa с правами суперпользователя и назначить ей пароль 'P@ssw0rd'. Подключиться через утилиту psql к базе postgres от имени роли sa. Создать БД backup_db_from_plain. Установить классификационную метку для БД backup_db_from_plain в значение {3,3}. Создать схему ssidorov. Установить классификационную метку на схему ssidorov '{3,3}'. Восстановить данные таблицы ssidorov.test_copy из резервной копии backup_db_test_copy.sql в БД backup_db_from_plain кластера reserve.

1 Определить, какой порт использует кластер reserve.

```
pg_lsclusters
```

1 Подключиться к кластеру БД reserve под учетной записью postgres:

```
sudo -u postgres psql -p <порт кластера БД reserve>
```

2 Создать роль sa с правами суперпользователя и назначить ей пароль 'P@ssw0rd':

```
CREATE ROLE sa WITH LOGIN SUPERUSER PASSWORD 'P@ssw0rd';
```

3 Создать БД backup_db_from_plain.

```
CREATE DATABASE backup_db_from_plain;
```

4 Написать запрос, выводящий информацию о классификационной метке и признаке CCR для БД backup_db:

```
SELECT datname,maclabel,datmacccr FROM pg_database
WHERE datname='backup_db_from_plain';
```

5 Изменить классификационную метку кластера reserve на {3,3}:

```
MAC LABEL ON CLUSTER IS '{3,3}';
```

6 Установить классификационную метку для БД backup_db_from_plain в значение {3,3}.

```
\connect backup_db_from_plain
MAC LABEL ON DATABASE backup_db_from_plain IS '{3,3}';
```

7 Создать схему ssidorov.

```
CREATE SCHEMA ssidorov;
```

8 Установить классификационную метку на схему ssidorov '{3,3}'.

```
MAC LABEL ON SCHEMA ssidorov IS '{3,3}';
```

9 Восстановить данные таблицы ssidorov.test_copy из резервной копии backup_db_sidorov.test_copy.sql в БД backup_db_from_plain кластера reserve.

```
\q
cd /home/logical_backups
psql -U sa -h localhost -p <порт кластера БД reserve> -d
backup_db_from_plain -f backup_db_test_copy.sql
```

10 Подключиться через утилиту psql к БД backup_db_from_plains кластера reserve от имени роли sa.

```
psql -h localhost -d backup_db_from_plain -U sa -p <порт
кластера БД reserve>
```

11 Выполнить выборку из таблицы *ssidorov.test_copy*. Вывести информацию о классификационных метках.

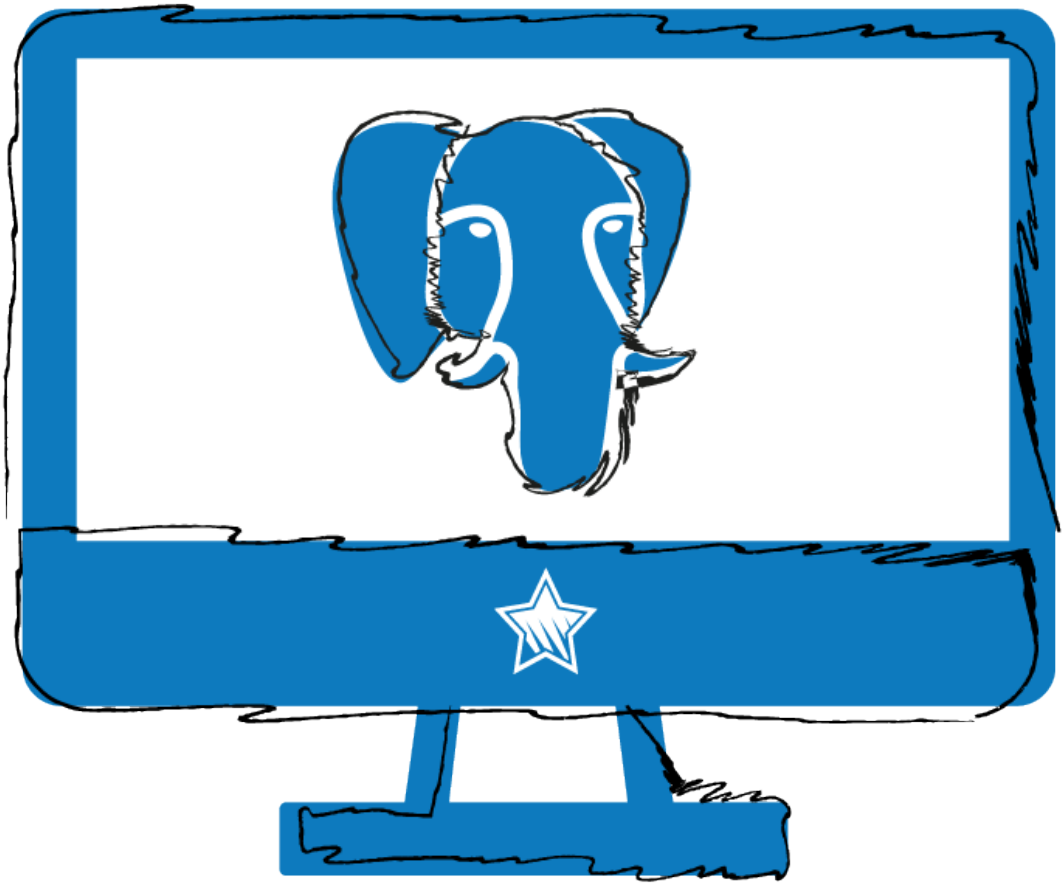
```
SELECT maclabel, * FROM ssidorov.test_copy;
```

Убедиться, что записи, которые присутствовали в таблице ssidorov.test_copy БД backup_db, присутствуют в БД backup_db_from_plain, в таблице ssidorov.test_copy.

Astra Linux для АНО УЦ ДПО « Айти Клауд»

Модуль 9

Физическое резервное копирование и восстановление



ASTRA LINUX®

Модуль 9. Физическое резервное копирование и восстановление

Описание практического задания:

Оценка времени выполнения: 120 мин.

- Создание холодной резервной копии.
- Создание автономной (базовой) резервной копии.
- Восстановление из автономной (базовой) резервной копии.
- Настройка файлового архивирования журналов предзаписи.
- Создание базовой резервной копии при настроенном потоковом архивировании журналов предзаписи.
- Восстановление из базовой резервной копии при настроенном потоковом архивировании журналов предзаписи.

Создание холодной резервной копии

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Остановить процесс postmaster.pid кластера main. Остановить кластер БД reserve. Удалить все файлы данных кластера reserve. Скопировать все файлы данных кластера main в каталог данных кластера reserve. Запустить кластер БД reserve. Подключиться к БД backupdb кластера reserve от имени пользователя sa и сделать выборку из таблицы sidorov.test_copy. Убедиться на примере таблицы ssidorov.test_copy, что данные из кластера БД main восстановлены в кластере БД reserve.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Узнать, какой PID у процесса postmaster кластера БД main:

```
sudo head -n 1 /var/lib/postgresql/11/main/postmaster.pid
```

3 Остановить процесс postmaster.pid кластера main, используя идентификатор, полученный на предыдущем шаге (имитация аварийного останова кластера):

```
sudo kill -9 PID_процесса_postmaster
```

4 Убедиться, что кластер main остановлен.

```
sudo pg_ctlcluster 11 main status
```

5 Остановить кластер БД reserve.

```
sudo pg_ctlcluster 11 reserve stop
```

6 Убедиться, что кластер reserve остановлен.

```
sudo pg_ctlcluster 11 reserve status
```

7 Удалить все файлы данных кластера reserve:

```
sudo rm -rf /var/lib/postgresql/11/reserve
```

8 Скопировать все файлы данных кластера main в каталог данных кластера reserve:

```
sudo cp -rp /var/lib/postgresql/11/main
/var/lib/postgresql/11/reserve
```

9 Запустить кластер БД reserve.

```
sudo pg_ctlcluster 11 reserve start
```

Согласованность данных на кластере reserve будет восстановлена, кластер запустится.

10 Определить, какой порт использует кластер reserve.

```
pg_lsclusters
```

11 Подключиться к БД backupdb кластера reserve от имени пользователя sa и сделать выборку из таблицы sidorov.test_copy:

```
psql -h localhost -p <порт_кластера_reserve> -U sa -d
backup_db
SELECT * FROM ssidorov.test_copy;
```

Создание автономной (базовой) резервной копии

Задание: Запустить кластер БД main. Просмотреть параметры протокола репликации кластера main, подключившись от имени роли sa. Просмотреть разрешения на локальное подключение по протоколу репликации в pg_hba.conf. В каталоге /home создать подкаталог basebackup. Создать базовую резервную копию кластера БД main, подключившись к кластеру от имени роли ssidorov. Резервную копию разместите в каталоге /home/backups/basebackup.

1 Запустить кластер БД *main*.

```
\q
sudo pg_ctlcluster 11 main start
```

2 Просмотреть параметры протокола репликации кластера main, подключившись от имени роли sa:

```
psql -h localhost -U sa -d postgres
SELECT name, setting
```

```
FROM pg_settings
WHERE name IN
('wal_level','max_wal_senders','max_replication_slots');
```

3 Просмотреть разрешения на локальное подключение по протоколу репликации в pg_hba.conf:

```
SELECT type, database, user_name, address, auth_method
FROM pg_hba_file_rules()
WHERE 'replication' = ANY(database);
```

```
postgres=#       SELECT type, database, user_name, address, auth_method
postgres-#       FROM pg_hba_file_rules()
postgres-#       WHERE 'replication' = ANY(database);
 type  |   database    | user_name |  address  | auth_method
-------+---------------+-----------+-----------+-------------
 local | {replication} | {all}     |           | trust
 host  | {replication} | {all}     | 127.0.0.1 | md5
 host  | {replication} | {all}     | ::1       | md5
(3 строки)
```

Разрешения на локальное подключение по протоколу репликации в pg_hba.conf по умолчанию предоставлены.

Чтобы утилита pg_basebackup могла подключиться к серверу под ролью ssidorov, эта роль должна иметь атрибут REPLICATION.

4 Выйти из psql и переключиться на root:

```
\q
sudo -i
```

5 В каталоге /home создать подкаталог basebackup.

```
mkdir /home/basebackup
```

6 Создать базовую резервную копию кластера БД main, подключившись к кластеру от имени роли ssidorov. Резервную копию разместите в каталоге /home/basebackup:

```
pg_basebackup -U ssidorov -h localhost -D /home/basebackup
```

7 После завершения работы команды проверить содержимое каталога /home/basebackup, просмотреть мандатные метки файлов.

```
pdp-ls -M /home/basebackup
```

Восстановление из автономной (базовой) резервной копии

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Остановить кластер БД reserve. Удалить все файлы данных кластера reserve. Скопировать все файлы из базовой резервной копии в каталог данных кластера reserve. Проверить содержимое каталога /var/lib/postgresql/11/reserve, просмотреть мандатные метки файлов. Установить в качестве владельца каталога /var/lib/postgresql/11/reserve пользователя postgres. Установить на каталог reserve разрешения drwx---------. Запустить кластер БД reserve. Подключиться к кластеру reserve от имени пользователя zzakharov, БД backupdb и сделать выборку из таблицы sidorov.test_copy.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Переключиться на root:

```
sudo -i
```

3 Остановить кластер БД reserve.

```
pg_ctlcluster 11 reserve stop
```

4 Убедиться, что кластер reserve остановлен.

```
pg_ctlcluster 11 reserve status
```

5 Удалить все файлы данных кластера reserve:

```
rm -rf /var/lib/postgresql/11/reserve/*
```

6 Скопировать все файлы из базовой резервной копии в каталог данных кластера reserve:

```
cp -r /home/basebackup/* /var/lib/postgresql/11/reserve
```

7 Проверить содержимое каталога /var/lib/postgresql/11/reserve, просмотреть мандатные метки файлов :

```
pdp-ls -M /var/lib/postgresql/11/reserve
```

8 Установить в качестве владельца каталога /var/lib/postgresql/11/reserve и вложенных файлов пользователя postgres:

```
chown -R postgres:postgres /var/lib/postgresql/11/reserve
```

9 Запустить кластер БД reserve:

```
pg_ctlcluster 11 reserve start
```

Согласованность данных на кластере reserve будет восстановлена, кластер запустится.

10 Подключиться к кластеру reserve от имени пользователя sa – БД backupdb и сделать выборку из таблицы ssidorov.test_copy:

```
psql -h localhost -p <порт_кластера_reserve> -d backup_db -
U sa
```

```
SELECT * FROM ssidorov.test_copy;
```

Настройка файлового архивирования журналов предзаписи

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Создать каталог /home/WAL_file_arch для файлового архива журналов предзаписи. Изменить владельца каталога на postgres.

Установить значение параметров в файле postgresql.conf кластера БД main:

- archive_mode = on
- archive_command = 'cp %p /home/WAL_file_arch/%f'

Перезапустить кластер main БД.

Для кластера БД main определить, какой сегмент WAL используется в данный момент. Принудительно переключить сегмент WAL. Повторно определить, какой сегмент WAL используется. Просмотреть текущий статус архивации через представление pg_stat_archiver. Проверить, что предыдущий сегмент попал в архив и посмотреть его мандатную метку.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Переключиться на root:

```
sudo -i
```

3 Создать каталог для файлового архива журналов предзаписи — каталог /home/WAL_file_arch:

```
mkdir /home/WAL_file_arch
```

При проведении курса архив размещается в локальном каталоге.

Архив может размещаться на отдельном сервере или дисковой системе с доступом по сети.

4 Изменить владельца каталога /home/WAL_file_arch на postgres:

```
chown -R postgres:postgres /home/WAL_file_arch
```

5 Отредактировать файл postgresql.conf для кластера БД main. Установить значение параметров:

```
archive_mode = on
archive_command = 'cp %p /home/WAL_file_arch/%f'
```

Не забудьте раскомментировать соответствующие строки.

```
cd /etc/postgresql/11/main/
mcedit postgresql.conf
```

6 Перезапустить кластер main БД.

```
pg_ctlcluster 11 main restart
```

7 Подключиться к кластеру БД main от имени пользователя sa, БД postgres.

```
psql -h localhost -U sa -d postgres
```

8 Определить, какой сегмент WAL используется в данный момент:

```
SELECT pg_walfile_name(pg_current_wal_lsn());
```

```
postgres=# SELECT pg_walfile_name(pg_current_wal_lsn());
     pg_walfile_name
--------------------------
 000000010000000000000004
(1 строка)
```

9 В тестовых целях переключить сегмент WAL **принудительно**:

```
SELECT pg_switch_wal();
```

```
postgres=# SELECT pg_switch_wal();
 pg_switch_wal
---------------
 0/4002058
(1 строка)
```

10 Повторно определить, какой сегмент WAL используется в данный момент:

```
SELECT pg_walfile_name(pg_current_wal_lsn());
```

```
postgres=# SELECT pg_walfile_name(pg_current_wal_lsn());
     pg_walfile_name
--------------------------
 000000010000000000000005
(1 строка)
```

Сегмент сменился.

11 Просмотреть текущий статус архивации через представление pg_stat_archiver:

```
SELECT * FROM pg_stat_archiver \gx
```

```
postgres=# SELECT * FROM pg_stat_archiver \gx
-[ RECORD 1 ]------+------------------------------
archived_count     | 1
last_archived_wal  | 000000010000000000000004
last_archived_time | 2023-12-06 12:55:02.760076+03
failed_count       | 0
last_failed_wal    |
last_failed_time   |
stats_reset        | 2023-12-06 10:52:03.193868+03
```

12 Проверить, что предыдущий сегмент попал в архив в каталог /home/WAL_file_arch/ и посмотреть его мандатную метку:

```
pdp-ls -M /home/WAL_file_arch/
```

```
root@astra:~# pdp-ls -M /home/WAL_file_arch/
итого 16384
-rw---------  1 postgres postgres Уровень_0:Низкий:Нет:0x0 000000010000000000000004
```

Файл присутствует в нужном каталоге. Архивация файлов журнала предзаписи работает.

Настройка потокового архивирования журналов предзаписи

Задание: Остановить файловую репликацию.

Установить значение параметров в файле postgresql.conf кластера БД main:

- archive_mode = off
- archive_command = ''

Перезапустить кластер main БД.

Создать каталог для потокового архива журналов предзаписи — каталог /home/WAL_cont_arch. Изменить владельца каталога на postgres.

Создать слот репликации через утилиту pg_receivewal. Запустить утилиту pg_receivewal в фоном в режиме.

В другой вкладке консоли подключиться к БД postgres кластера БД main от имени пользователя sa, БД postgres.

Для кластера БД main определить, какой сегмент WAL используется в данный момент. Принудительно переключить сегмент WAL. Повторно определить, какой сегмент WAL используется. Проверить, что предыдущий сегмент попал в архив и посмотреть его мандатную метку. Проверить наличие файла с суффиксом .partial.

1 Остановить файловую репликацию: отредактировать файл postgresql.conf для кластера БД main. Установить значение параметров:

```
archive_mode = off
archive_command = ''
```

```
cd /etc/postgresql/11/main
mcedit postgresql.conf
```

2 Перезапустить кластер main БД.

```
pg_ctlcluster 11 main restart
```

3 Переключиться на root:

```
sudo -i
```

4 Создать каталог для потокового архива журналов предзаписи — каталог /home/WAL_cont_arch:

```
mkdir /home/WAL_cont_arch
```

5 Изменить владельца каталога на postgres:

```
chown -R postgres:postgres /home/WAL_cont_arch
exit
```

6 Создать слот репликации через утилиту pg_receivewal, чтобы гарантировать получение всех записей журнала:

```
sudo -u postgres /bin/pg_receivewal --create-slot --slot=archive
```

7 Запустить утилиту pg_receivewal в фоновом режиме:

```
sudo -u postgres /bin/pg_receivewal -D /home/WAL_cont_arch --slot=archive
```

Если в архиве уже будут файлы, утилита запросит у сервера **следующий** сегмент, чтобы в архиве не было пропусков.

8 В другой вкладке консоли подключиться к БД postgres кластера БД main от имени пользователя sa, БД postgres.

```
psql -h localhost -U sa -d postgres
```

9 Определить, какой сегмент WAL используется в данный момент:

```
SELECT pg_walfile_name(pg_current_wal_lsn());
```

```
postgres=# SELECT pg_walfile_name(pg_current_wal_lsn());
     pg_walfile_name
--------------------------
 000000010000000000000006
(1 строка)
```

10 Выполнить вставку строки данных в таблицу ssidorov.test_copy базы данных backup_db:

```
\c backup_db
```

```
INSERT INTO ssidorov.test_copy (id, secret_level_comment)
VALUES (4,'Уровень конфиденциальности 0, без категории.
Вставка после настройки потокового архивирования журналов
предзаписи');
```

11 В тестовых целях переключить сегмент WAL **принудительно:**

```
SELECT pg_switch_wal();
```

```
backup_db=# SELECT pg_switch_wal();
 pg_switch_wal
---------------
 0/6000578
(1 строка)
```

12 Повторно определить, какой сегмент WAL используется в данный момент:

```
SELECT pg_walfile_name(pg_current_wal_lsn());
```

```
backup_db=# SELECT pg_walfile_name(pg_current_wal_lsn());
     pg_walfile_name
--------------------------
 000000010000000000000007
(1 строка)
```

Сегмент сменился.

13 Просмотреть текущий статус архивации через представление pg_stat_archiver:

```
SELECT * FROM pg_stat_archiver \gx
```

```
backup_db=# SELECT * FROM pg_stat_archiver \gx
-[ RECORD 1 ]------+------------------------------
archived_count     | 2
last_archived_wal  | 000000010000000000000005
last_archived_time | 2023-12-06 13:04:36.54833+03
failed_count       | 0
last_failed_wal    |
last_failed_time   |
stats_reset        | 2023-12-06 10:52:03.193868+03
```

14 Проверить, что предыдущий сегмент попал в архив в каталог и посмотреть его мандатную метку:

```
\q
pdp-ls -M /home/WAL_cont_arch/
```

```
sa@astra:~$ pdp-ls -M /home/WAL_cont_arch/
итого 32768
-rw----------  1 postgres postgres Уровень_0:Низкий:Нет:0x0 000000010000000000000006
-rw----------  1 postgres postgres Уровень_0:Низкий:Нет:0x0 000000010000000000000007.partial
```

Файл присутствует в нужном каталоге. Потоковое архивирование журналов предзаписи работает.

В каталоге также есть файл с суффиксом *.partial*, куда ведется запись текущего журнала.

Создание базовой резервной копии при настроенном потоковом архивировании журналов предзаписи

Задание: В каталоге /home создать подкаталог basebackup_no_WAL. Создать базовую резервную копию кластера БД main, подключившись к кластеру от имени роли ssidorov. Дополнительно указать параметр --wal-method=none. Резервную копию разместите в каталоге /home/basebackup_no_WAL. Просмотреть содержимое каталога pg_wal резервной копии. Просмотреть содержимое каталога /home/WAL_cont_arch/ и мандатные метки файлов. Просмотреть содержимое файла метки резервной копии.

1 Переключиться на root:

```
sudo -i
```

2 В каталоге /home создать подкаталог basebackup_no_WAL.

```
mkdir /home/basebackup_no_WAL
```

3 Создать базовую резервную копию кластера БД main, подключившись к кластеру от имени роли ssidorov. Дополнительно указать параметр `--wal-method=none`. Резервную копию разместите в каталоге */home/basebackup_no_WAL*:

```
pg_basebackup -U ssidorov -h localhost -D
/home/basebackup_no_WAL --wal-method=none
```

4 Просмотреть содержимое каталога pg_wal резервной копии:

```
ls -l /home/basebackup_no_WAL/pg_wal
```

```
root@astra:~# ls -l /home/basebackup_no_WAL/pg_wal
итого 4
drwx------ 2 root root 4096 дек  6 13:35 archive_status
```

В нем не будет сегментов журнала.

5 Просмотреть содержимое каталога /home/WAL_cont_arch/ и мандатные метки файлов.

```
pdp-ls -M /home/WAL_cont_arch/
```

```
root@astra:~# pdp-ls -M /home/WAL_cont_arch/
итого 65536
-rw----------  1 postgres postgres Уровень_0:Низкий:Нет:0x0 000000010000000000000006
-rw----------  1 postgres postgres Уровень_0:Низкий:Нет:0x0 000000010000000000000007
-rw----------  1 postgres postgres Уровень_0:Низкий:Нет:0x0 000000010000000000000008
-rw----------  1 postgres postgres Уровень_0:Низкий:Нет:0x0 000000010000000000000009.partial
```

Обратите внимание, что количество сегментов в архиве **увеличилось**.

6 Просмотреть содержимое файла метки резервной копии:

```
cat /home/basebackup_no_WAL/backup_label
```

```
root@astra:~# cat /home/basebackup_no_WAL/backup_label
START WAL LOCATION: 0/8000028 (file 000000010000000000000008)
CHECKPOINT LOCATION: 0/8000060
BACKUP METHOD: streamed
BACKUP FROM: master
START TIME: 2023-12-06 13:35:36 MSK
LABEL: pg_basebackup base backup
START TIMELINE: 1
```

Строка START TIMELINE содержит номер линии времени.

Восстановление из базовой резервной копии при настроенном потоковом архивировании журналов предзаписи

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). В каталоге резервной копии basebackup_no_WAL создать файл recovery.conf. Внести в файл recovery.conf строку:

```
restore_command = 'cp /home/WAL_cont_arch/%f %p'
```

В каталоге резервной копии basebackup_no_WAL создать файл recovery.signal. Остановить кластер БД reserve. Удалить все файлы данных кластера БД reserve. Скопировать все файлы резервной копии в каталог данных кластера reserve. Установить в качестве владельца каталога /var/lib/postgresql/11/reserve и вложенных файлов пользователя postgres. Запустить кластер БД reserve. Подключиться к БД backup_db кластера reserve от имени пользователя sa. Сделать выборку из таблицы ssidorov.test_copy. Убедиться, что в таблице есть запись со значением id=4, внесенная после настройки потокового архивирования журналов предзаписи.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Переключиться на root:

```
sudo -i
```

3 В каталоге резервной копии basebackup_no_WAL создать файл recovery.conf:

```
cd /home/basebackup_no_WAL/
touch recovery.conf
```

4 Внести в файл recovery.conf строку:

```
restore_command = 'cp /home/WAL_cont_arch/%f %p'
```

```
mcedit recovery.conf
```

Не указывать целевую точку восстановления (один из параметров recovery_target*). В этом случае к базовой резервной копии будут применены записи WAL из **всех** файлов в архиве.

5 Остановить кластер БД reserve.

```
pg_ctlcluster 11 reserve stop
```

6 Удалить все файлы данных кластера БД reserve:

```
rm -rf /var/lib/postgresql/11/reserve/*
```

7 Скопировать все файлы резервной копии в каталог данных кластера reserve.

```
cp -rp /home/basebackup_no_WAL/*
/var/lib/postgresql/11/reserve
```

8 Просмотреть содержимое каталога /var/lib/postgresql/11/reserve.

Убедиться в наличии файла recovery.conf.

```
cd /var/lib/postgresql/11/reserve
ls
```

```
root@astra:/var/lib/postgresql/11/reserve# ls
backup_label      pg_commit_ts  pg_notify     pg_stat_tmp  pg_wal
base              pg_dynshmem   pg_replslot   pg_subtrans  pg_xact
current_logfiles  pg_log        pg_serial     pg_tblspc    postgresql.auto.conf
global            pg_logical    pg_snapshots  pg_twophase  recovery.conf
pg_audit.conf     pg_multixact  pg_stat       PG_VERSION
```

9 Установить в качестве владельца каталога /var/lib/postgresql/11/reserve и вложенных файлов пользователя postgres:

```
chown -R postgres:postgres /var/lib/postgresql/11/reserve
```

10 Изменить значение параметра *max_connections* (задает максимальное количество одновременных соединений, которое экземпляр PostgreSQL может обслуживать) на **1000** в файле /etc/postgresql/11/reserve/postgresql.conf.

```
sudo mcedit /etc/postgresql/11/reserve/postgresql.conf
Изменить значение параметра max_connections на 1000.
Нажать F10 для выхода. При выходе сохранить изменения.
```

11 Запустить кластер БД reserve.

```
pg_ctlcluster 11 reserve start
```

После успешного восстановления файл метки переименовывается в *backup_label.old*.

```
ls -l /var/lib/postgresql/11/reserve | egrep backup_label.*
```

12 Подключиться к БД backup_db кластера reserve от имени пользователя sa.

```
psql -h localhost -U sa -p <порт_кластера_reserve> -d
backup_db
```

13 Сделать выборку из таблицы ssidorov.test_copy.

```
SELECT * FROM ssidorov.test_copy;
```

Убедиться, что в таблице есть запись со значением **id=4**, внесенная после настройки потокового архивирования журналов предзаписи.

14 После успешного завершения восстановления остановить работу утилиты pg_receivewal.

15 Просмотреть список слотов репликации:

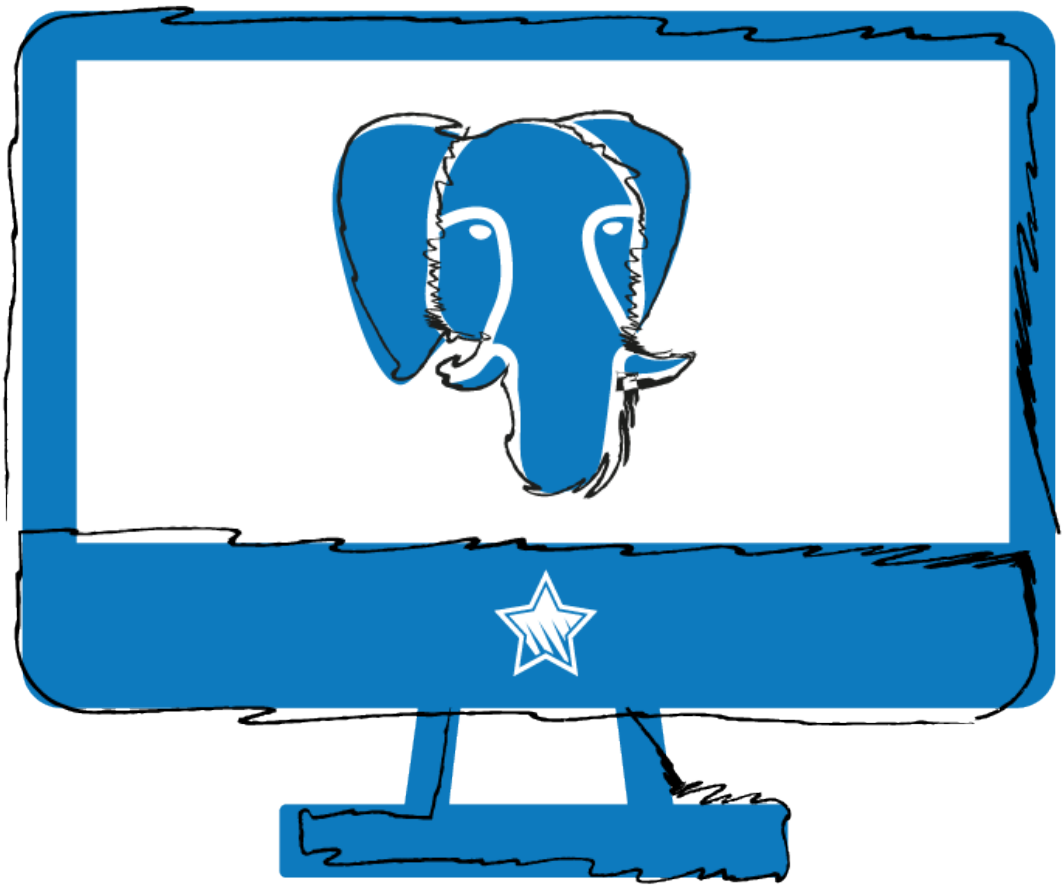
```
SELECT slot_name FROM pg_replication_slots;
```

16 Удалить имеющийся слот (слоты) репликации:

SELECT pg_drop_replication_slot('archive');

Модуль 10

Регламентные работы



ASTRA LINUX®

Модуль 10. Регламентные работы

Описание практического задания:

Оценка времени выполнения: 30 мин.

- Выполнение обычной очистки.
- Очистка большого количества строк.
- Полная очистка после удаления большого количества строк.

Выполнение обычной очистки

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Подключиться через утилиту psql к базе inventory кластера БД main от имени роли sa.

Создать таблицу supp_dept.for_vacuum с параметром autovacuum_enabled=off и одним столбцом id integer.

Вставить в таблицу supp_dept.for_vacuum, столбец id значения от 1 до 2000000.

Создать индекс на таблице supp_dept.for_vacuum по столбцу id.

Уменьшить размер памяти, выделяемой под массив идентификаторов до 1МБ (это замедлит процесс выполнения очистки).

Обновить все строки, увеличив значение id на 1.

В другом сеансе консоли подключиться через утилиту psql к БД inventory от имени роли sa.

Написать запрос на выборку из представления pg_stat_progress_vacuum, но не запускать его.

Запустить очистку командой VACUUM и как можно скорее переключить на сеанс с подготовленным запросом на выборку из представления pg_stat_progress_vacuum и запустить запрос.

Восстановить прежнее значение размера памяти, выделяемой под массив идентификаторов.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться через утилиту psql к базе inventory кластера main от имени роли sa.

```
psql -h localhost -d inventory -U sa
```

3 Создать таблицу *supp_dept.for_vacuum*. При создании указать параметр *autovacuum_enabled=off*:

```
CREATE TABLE supp_dept.for_vacuum(id integer) WITH
(autovacuum_enabled = off);
```

4 Вставить в таблицу значения. Для это сгенерировать ряд значений от 1 до 3000000 с шагом 1, используя функцию generate_series:

```
INSERT INTO supp_dept.for_vacuum SELECT gener.id FROM
generate_series(1,2000000) gener(id);
```

5 Создать индекс на таблице supp_dept.for_vacuum по столбцу **id**:

```
CREATE INDEX for_vacuum_id ON supp_dept.for_vacuum(id);
```

6 Уменьшить размер памяти, выделяемой под массив идентификаторов (это замедлит процесс выполнения очистки):

```
ALTER SYSTEM SET maintenance_work_mem = '1MB';
```

7 Перечитать конфигурацию:

```
SELECT pg_reload_conf();
```

8 Обновить все строки:

```
UPDATE supp_dept.for_vacuum SET id = id + 1;
```

9 В другом сеансе консоли подключиться через утилиту psql к БД inventory кластера main от имени роли sa.

```
psql -h localhost -d inventory -U sa
```

10 Написать запрос на выборку из представления pg_stat_progress_vacuum, но не запускать его:

```
SELECT * FROM pg_stat_progress_vacuum \gx
```

11 Запустить очистку командой VACUUM и как можно скорее переключить на сеанс с подготовленным запросом на выборку из представления pg_stat_progress_vacuum и запустить запрос:

```
VACUUM VERBOSE supp_dept.for_vacuum;
```

Команда VACUUM выдаст отчет следующего вида:

```
ИНФОРМАЦИЯ:  очистка "supp_dept.for_vacuum"
ИНФОРМАЦИЯ:  "for_vacuum": удалено версий строк: 2000000,
обработано страниц: 10811
ИНФОРМАЦИЯ:  "for_vacuum": найдено удаляемых версий строк:
2000000, неудаляемых - 2000000, обработано страниц: 21622,
всего страниц: 21622
ПОДРОБНОСТИ:  В данный момент нельзя удалить "мёртвых"
строк: 0, старейший xmin: 878
```

```
Найдено неиспользованных указателей: 0.
Пропущено страниц, закреплённых в буфере: 0,замороженных страниц: 0.
Полностью пустых страниц: 0.
CPU: пользов.: 0.12 с, система: 0.11 с, прошло: 0.62 с.
VACUUM
```

Запрос на выборку из представления pg_stat_progress_vacuum выдаст информацию вида:

```
inventory=# SELECT * FROM pg_stat_progress_vacuum \gx
-[ RECORD 1 ]------+-----------------
pid                | 30913
datid              | 98354
datname            | inventory
relid              | 114796
phase              | vacuuming indexes
heap_blks_total    | 10811
heap_blks_scanned  | 3776
heap_blks_vacuumed | 2831
index_vacuum_count | 3
max_dead_tuples    | 174762
num_dead_tuples    | 174640
```

12 Восстановить прежнее значение размера памяти, выделяемой под массив идентификаторов:

ALTER SYSTEM RESET maintenance_work_mem;

13 Перечитать конфигурацию:

SELECT pg_reload_conf();

Очистка большого количества строк

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Подключиться через утилиту psql к базе inventory кластера main от имени роли sa.

Определить текущий размер файла данных таблицы supp_dept.for_vacuum.

Удалить из таблицы supp_dept.for_vacuum около 90% строк случайным образом.

Выполнить очистку таблицы supp_dept.for_vacuum.

Определить размер файла данных таблицы supp_dept.for_vacuum осле очистки.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться через утилиту psql к базе inventory кластера main от имени роли sa.

```
psql -h localhost -d inventory -U sa
```

3 Определить текущий размер файла данных таблицы supp_dept.for_vacuum:

```
SELECT pg_size_pretty(pg_table_size('supp_dept.for_vacuum'));
```

```
inventory=# SELECT pg_size_pretty(pg_table_size('supp_dept.for_vacuum'));
 pg_size_pretty
----------------
 169 MB
(1 строка)
```

4 Удалить из таблицы supp_dept.for_vacuum около 90% строк случайным образом:

```
DELETE FROM supp_dept.for_vacuum WHERE random() < 0.9;
```

```
inventory=# DELETE FROM supp_dept.for_vacuum WHERE random() < 0.9;
DELETE 1800429
```

5 При удалении строк случайным образом высока вероятность, что на каждой странице останутся какие-то неудаленные строки. Если с каких-то страниц будут удалены все строки, то размер файла будет **уменьшен** при очистке.

6 Выполнить очистку таблицы *supp_dept.for_vacuum*.

```
VACUUM supp_dept.for_vacuum;
```

7 Определить размер файла данных таблицы supp_dept.for_vacuum после очистки.

```
SELECT
pg_size_pretty(pg_table_size('supp_dept.for_vacuum'));
pg_size_pretty
----------------
169 MB
(1 строка)
Объем не изменился.
```

Полная очистка после удаления большого количества строк

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Подключиться через утилиту psql к базе inventory кластера main от имени роли sa.

Усечь таблицу supp_dept.for_vacuum.

Вставить в таблицу supp_dept.for_vacuum, столбец id значения от 1 до 2000000.

Определить размер файла данных таблицы supp_dept.for_vacuum. Удалить из таблицы supp_dept.for_vacuum около 90% строк случайным образом. Выполнить полную очистку таблицы supp_dept.for_vacuum. Определить текущий размер файла данных таблицы supp_dept.for_vacuum. Сделать выводы.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Подключиться через утилиту psql к базе *inventory* кластера main от имени роли sa.

```
psql -h localhost -d inventory -U sa
```

3 Усечь (опустошить) таблицу *supp_dept.for_vacuum*:

```
TRUNCATE supp_dept.for_vacuum;
```

4 Вставить в таблицу значения. Для это сгенерировать ряд значений от 1 до 2000000 с шагом 1, используя функцию `generate_series`:

```
INSERT INTO supp_dept.for_vacuum SELECT gener.id FROM
generate_series(1,2000000) gener(id);
```

5 Определить текущий размер файла данных таблицы supp_dept.for_vacuum:

```
SELECT pg_size_pretty(pg_table_size('supp_dept.for_vacuum'));
```

```
inventory=# SELECT pg_size_pretty(pg_table_size('supp_dept.for_vacuum'));
 pg_size_pretty
----------------
 85 MB
(1 строка)
```

Обратите внимание, что при последнем определении размера файла данных для таблицы supp_dept.for_vacuum, он был примерно в два раза больше. Эта разница обусловлена тем, что часть пространства была занята версиями строк, которые создала команда UPDATE .

6 Удалить из таблицы supp_dept.for_vacuum около 90% строк случайным образом:

```
DELETE FROM supp_dept.for_vacuum WHERE random() < 0.9;
```

7 Выполнить полную очистку таблицы supp_dept.for_vacuum:

```
VACUUM FULL supp_dept.for_vacuum;
```

8 Определить текущий размер файла данных таблицы supp_dept.for_vacuum:

```
SELECT
pg_size_pretty(pg_table_size('supp_dept.for_vacuum'));
```

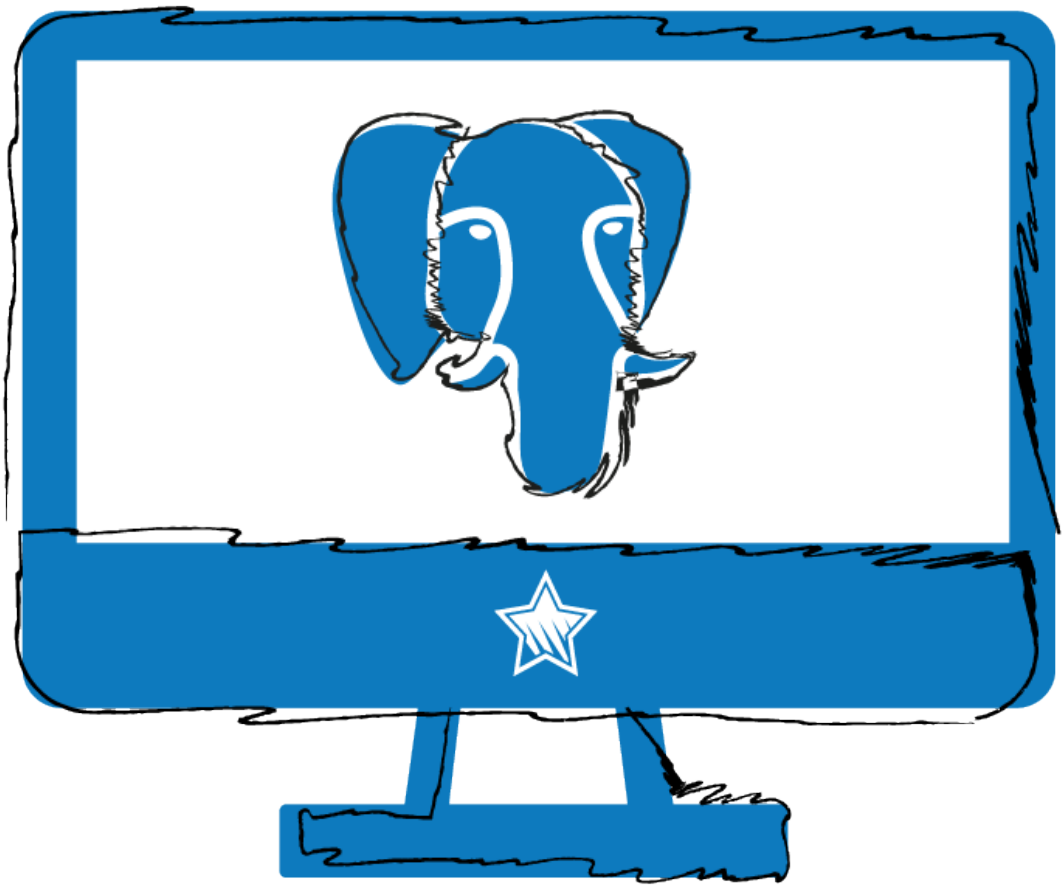
```
inventory=# SELECT pg_size_pretty(pg_table_size('supp_dept.for_vacuum'));
 pg_size_pretty
----------------
 8664 kB
(1 строка)
```

Размер уменьшился на 90%.

Astra Linux для АНО УЦ ДПО « Айти Клауд»

Модуль 11

Автоматизация обслуживания



ASTRA LINUX®

Модуль 11. Автоматизация обслуживания

Описание практического задания:

Оценка времени выполнения: 45 мин.

- Создание файла с паролями ~/.pgpass для учетной записи sa.
- Установка и запуск pgAgent.
- Подготовка каталога для выгрузки данных.
- Создание задания на экспорт таблицы в графическом интерфейсе pgAdmin.
- Изменение задания через запрос SQL: добавление шага.

Создание файла с паролями ~/.pgpass для учетной записи sa

Задание: Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»). Подключиться к базе postgres кластера main от имени роли sa, введя пароль. В домашнем каталоге создать файл .pgpass. Внести в файл .pgpass запись:

localhost:5432:*:sa:P@ssw0rd

Установить на файл .pgpass доступ на чтение и запись только для пользователя sa. Подключиться к базе postgres от имени роли sa не вводя пароль.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 В домашнем каталоге создать файл *.pgpass*.

```
touch .pgpass
```

3 Внести в файл .pgpass запись:

```
localhost:5432:*:sa:P@ssw0rd
```

```
mcedit .pgpass
Внести строку localhost:5432:*:sa:P@ssw0rd
Закрыть файл, сохранив изменения.
```

4 Установить на файл .pgpass доступ **только для** пользователя sa.

Выполните команду:

```
chmod 600 .pgpass
```

5 Подключиться к базе postgres кластера main от имени роли sa.

```
psql -h localhost -d postgres -U sa
```

Подключение будет выполнено без запроса пароля.

Установка расширения и запуск pgAgent

Задание: Подключиться через утилиту psql к базе данных postgres кластера main от имени роли sa. Установить расширение pgagent. Запустить pgAgent на сервере localhost, используя базу данных с именем postgres, подключиться от имени пользователя sa.

1 Подключиться через утилиту psql к базе postgres от имени роли sa.

```
psql -h localhost -d postgres -U sa
```

2 Установить расширение *pgagent*:

```
CREATE EXTENSION pgagent;
\q
```

3 Запустить pgAgent на сервере localhost, используя базу данных с именем postgres, подключиться от имени пользователя sa.

```
/usr/bin/pgagent hostaddr=localhost dbname=postgres user=sa
```

4 Чтобы убедиться, что процесс запустился, можно использовать команду:

```
ps -all | grep pgagent
```

Подготовка каталога для выгрузки данных

Задание: Создать каталог /home/scheduled_export. Установить для каталога scheduled_export мандатную метку с максимальным уровнем конфиденциальности 3, доступом к категориям: 0 (без категории), 1, 2, атрибут CCNR. Установить в качестве владельца каталога /home/scheduled_export пользователя postgres.

1 Войти в ОС с учётной записью пользователя sa (Уровень_0, «Высокий»).

2 Переключиться на root:

```
sudo -i
```

3 Создать каталог /home/scheduled_export.

```
mkdir /home/scheduled_export
```

4 Установить для каталога scheduled_export мандатную метку с максимальным уровнем конфиденциальности 3, доступом к категориям: 0 (без категории), 1, 2, атрибут CCNR:

```
pdpl-file 3:0:3:CCNR /home/scheduled_export
```

5 Установить в качестве владельца каталога /home/scheduled_export пользователя postgres:

```
chown -R postgres:postgres /home/scheduled_export
```

6 Проверить установленную метку безопасности и владельца для каталога scheduled_export:

```
pdp-ls -M /home/
```

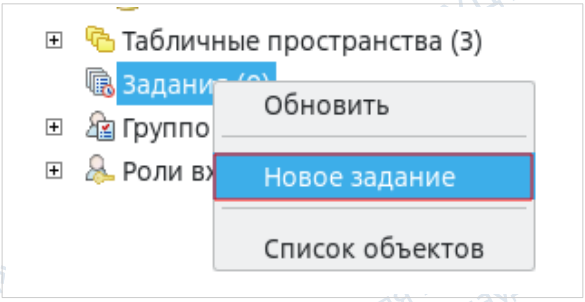
Создание задания на экспорт таблицы в графическом интерфейсе pgAdmin

Задание: Запустить утилиту pgAdmin. Подключиться к кластеру БД main под ролью sa. Создать задание с именем «Выгрузка таблицы», которое экспортирует таблицу supp_dept.equipment из БД inventory в рабочие дни в 18 часов в каталог /home/scheduled_export/ в файл с именем supp_det.equipment.copy. Запустить задание, проверить результат выполнения – его отображние в графическом интерфейсе и содержимое созданного файла supp_det.equipment.copy в каталоге /home/scheduled_export/.

1 Запустить утилиту pgAdmin. Подключиться к кластеру БД main под ролью sa.

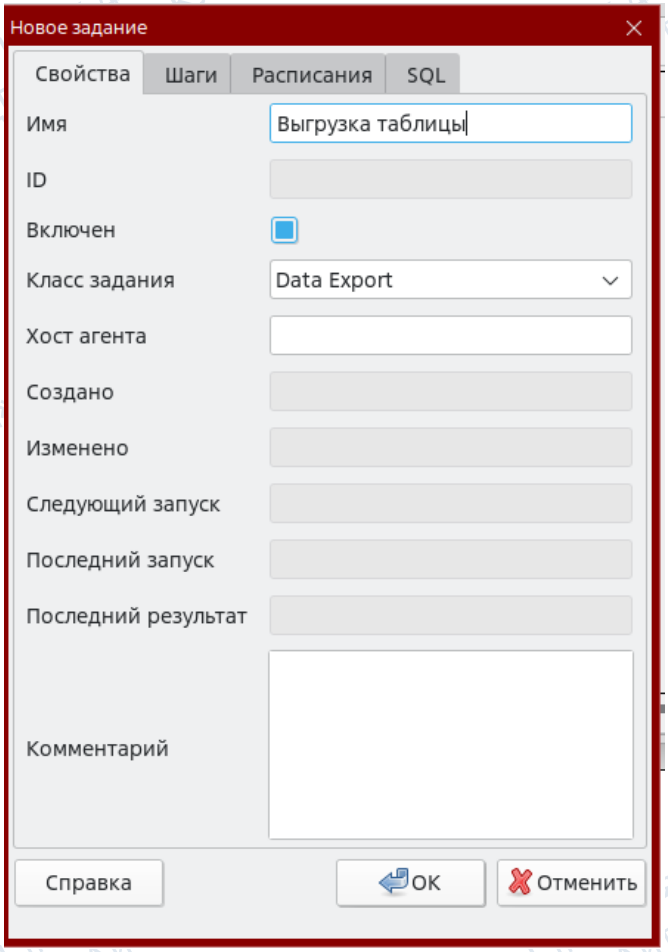
Пуск-Разработка-pgAdmin III
Настроить подключение от имени пользователя sa к кластеру БД main, если оно не было настроено.

2 Щелкнуть правой кнопкой мыши на узле «Задания». В контекстном меню выбрать «Новое задание».



3 В окне **Новое задание**, в поле **Имя** указать «Выгрузка таблицы».

4 В выпадающем списке **Класс задания** выбрать «Data Export».



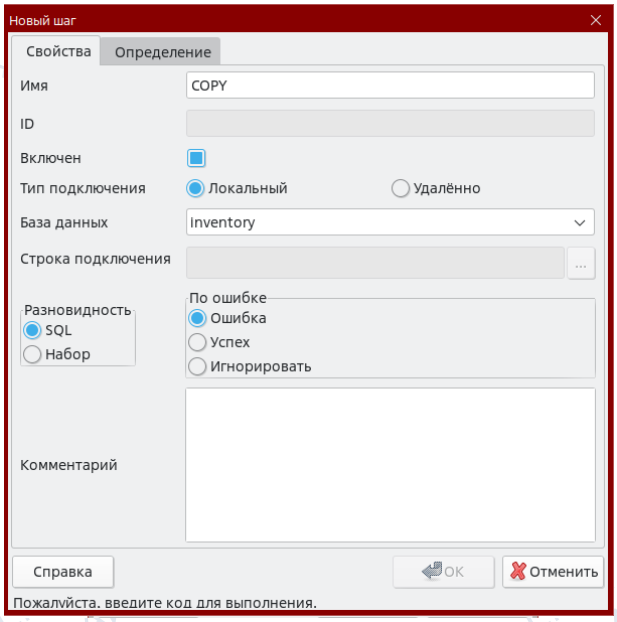
5 Во вкладке **Шаги** нажать кнопку «Добавить».

6 В окне **Новый шаг** укажите имя «COPY».

7 Тип подключения оставить без изменений (Локальный).

8 Разновидность оставить без изменений (SQL).

9 В выпадающем списке **База** указать «Inventory».



10 Во вкладке **Определение** вписать команду SQL:

```
COPY supp_dept.equipment TO
'/home/scheduled_export/supp_dept.equipment.copy';
```

11 Нажать **ОК**.

12 Во вкладке **Расписания** нажать кнопку **Добавить**.

13 В окне **Новое расписание**, на вкладке **Свойства**, в поле **Имя** ввести «В рабочие дни».

14 В поле **Конец** укажите дату – через месяц, от текущей даты.

15 На вкладке **Дни**, в блоке **Дни недели** отметить *Понедельник, Вторник, Среда, Четверг, Пятница*.

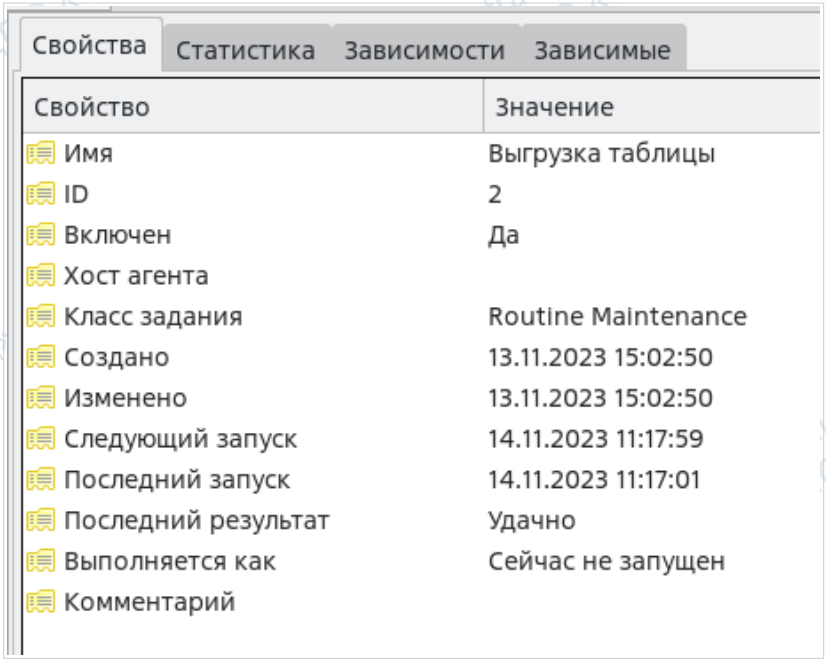
16 На вкладке **Время**, в блоке **Часов** отметить **18**, в блоке минут отметить **00**.

17 Нажать **ОК** на вкладке **Время**.

18 Нажать кнопку **ОК** для сохранения изменений в настройке задания.

19 Установить фокус на задание **Выгрузка таблицы** и через контекстное меню выбрать «Запустить сейчас».

20 В правой части окна обратить внимание на изменение данных в полях: *Последний запуск, Последний результат*. При необходимости обновите данные.



21 Проверить наличие файла с экспортированными данными и просмотреть мандатную метку файла supp_dept.equipment.copy:

```
pdp-ls -M /home/scheduled_export/
```

22 Просмотреть содержимое файла /home/scheduled_export/supp_dept.equipment.copy.

```
cat /home/scheduled_export/supp_dept.equipment.copy
```

Изменение задания через запрос SQL: добавление шага

Задание: Подключиться через утилиту psql к базе postgres кластера БД main от имени роли sa. Через запрос на SQL добавить к заданию «Выгрузка таблицы» еще один шаг, который изменит мандатную метку файла, созданного на предыдущем шаге и установит ее в значение: 3-й уровень конфиденциальности, Категория_1, Категория_2.

Запустить задание, проверить результат выполнения – его отображение в графическом интерфейсе и содержимое созданного файла supp_dept.equipment.copy в каталоге /home/scheduled_export/.

1 Подключиться через утилиту psql к базе postgres кластера БД main от имени роли sa.

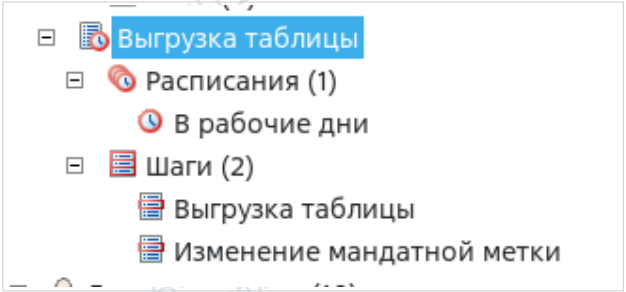
```
psql -h localhost -d postgres -U sa
```

2 Добавить к заданию «Выгрузка таблицы» еще один шаг, который изменит мандатную метку файла, созданного на предыдущем шаге и установит ее в значение: 3-й уровень конфиденциальности, Категория_1, Категория_2.

```
INSERT INTO pgagent.pga_jobstep (jstjobid, jstname,
jstdesc, jstenabled, jstkind, jstonerror, jstcode,
jstdbname, jstconnstr)
SELECT (SELECT jobid
        FROM pgagent.pga_job
        WHERE jobname = 'Выгрузка таблицы'), 'Изменение
мандатной метки', '', true, 'b', 'f', 'sudo pdpl-file 3:0:3
/home/scheduled_export/supp_dept.equipment.copy', '', '';
```

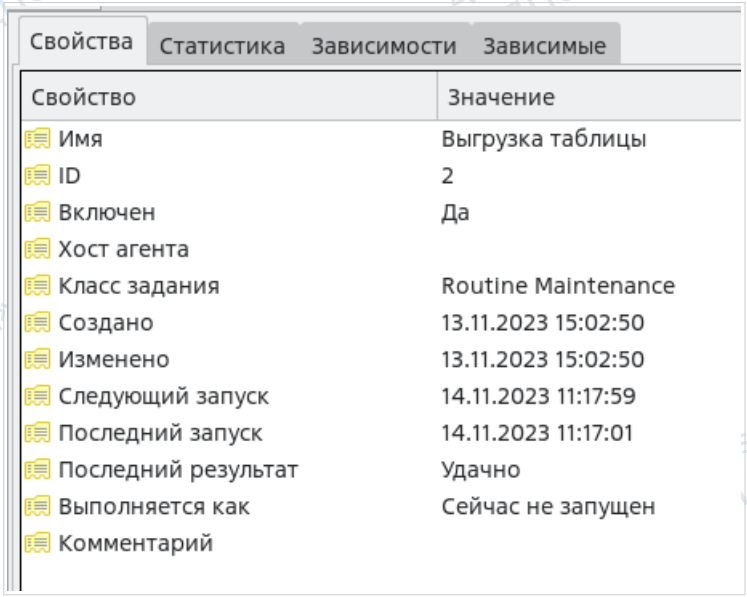
3 Переключиться в графический интерфейс утилиты pgAdmin.

4 Обновить данные. Убедиться, что в задании «Выгрузка таблицы» появился еще один шаг – «Изменение мандатной метки».



5 Установить фокус на задание **Выгрузка таблицы** и через контекстное меню выбрать «Запустить сейчас».

6 В правой части окна обратить внимание на изменение данных в полях: *Последний запуск, Последний результат*.



7 Проверить наличие файла с экспортированными данными и просмотреть мандатную метку файла supp_dept.equipment.copy:

```
sudo pdp-ls -M /home/scheduled_export/
```

8 Мандатная метка файла должны быть **изменена**. Данные должны быть следующего вида:

```
root@astra:/home/scheduled_export# pdp-ls -M /home/scheduled_export/
итого 4
-rw-r--r--m--  1 postgres postgres Уровень_3:Низкий:Категория_1,Категория_2:0x0 supp_dept.equipment.copy
```