

Лабораторная работа №1

Теоретическая часть

База данных (БД) - это упорядоченный набор структурированных данных, хранящихся в электронном виде.

Системы управления базами данных (СУБД) - это совокупность программных средств, обеспечивающих доступ и управление базами данных.

Типы баз данных

- *реляционный базы данных;*
- *объектно-ориентированные базы данных;*
- *NoSQL базы данных;*
- *графовые базы данных.*

PostgreSQL

PostgreSQL (или `Postgres`) - это **объектно-реляционная** система управления базами данных с открытым исходным кодом, разрабатываемая сообществом и распространяемая бесплатно под свободной лицензией **BSD**.

Установка PostgreSQL

`PostgreSQL` входит в состав подавляющего числа дистрибутивов `Linux`.

В **Debian** и его производных (e.g. **Ubuntu**) можно воспользоваться менеджером пакетов `apt`.

Info

Для того чтобы `установить/удалить` программные пакеты необходимо перейти в режим суперпользователя - `root`.

Для этого используется команды `su` , значение пароля `1234` :

```
user@dbms:~$ su
Пароль: 
```



Для установки PostgreSQL выполните команду :

```
apt-get install postgresql
```

```
Для его удаления используйте «apt autoremove».
Предлагаемые пакеты:
 postgresql-doc
Следующие НОВЫЕ пакеты будут установлены:
 postgresql
Обновлено 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 3 пакета
в не обновлено.
Необходимо скачать 0 B/64,7 kB архивов.
После данной операции объем занятого дискового пространства возрастет на 69,6 kB.
Выбор ранее не выбранного пакета postgresql.
(Чтение базы данных ... на данный момент установлено 57867 файлов и каталогов.)
Подготовка к распаковке .../postgresql_13+225ubuntu1.deb ...
Распаковывается postgresql (13+225) ...
Настраивается пакет postgresql (13+225) ...
Сканирование процессов...
Сканирование образов linux...

Запущено ядро последней версии.

Службы не требуют перезапуска.

Контейнеры не требуют перезапуска.

В сеансах пользователей нет устаревших процессов.
root@dbms:/home/user# 
```



Использование клиента `psql`

`psql` - текстовый клиент, позволяющий взаимодействовать с PostgreSQL .

Вывести набор поддерживаемых аргументов можно с помощью

```
psql --help
```

`psql` использует *UNIX-socket* и *TCP-socket*.

Для подключения к локальной СУБД (используется *UNIX-socket*) необходимо выполнить следующие действия:

1. Необходимо открыть сессию `root` (пароль: **1234**):

```
su
```

2. Открыть сессию пользователя postgres:

```
su postgres
```

3. Запустить клиента:

```
psql
```

Для использования удалённого подключения - подключения с другого компьютера необходимо произвести настройку PostgreSQL .

Настройка PostgreSQL

По умолчанию возможность подключения к сервер PostgreSQL по протоколу TCP выключена.

Для активации возможности принятия соединения от внешних клиентов необходимо отредактировать файл:

```
/etc/postgresql/${VERSION}/main/postgresql.conf
```

Где `${VERSION}` - число, являющиеся версией PostgreSQL. На одном сервере может работать несколько версий СУБД одновременно.

В файле конфигурации `postgresql.conf` необходимо строку

```
#listen_addresses = 'localhost'
```

убрать символ `#`, (данный символ комментирует строку) и вписать вместо `localhost = 'localhost'`

```
listen_addresses = '*'
```

Также в этом файле можно настроить значение сетевого порта, по умолчанию это значение соответствует **5432**.

Перед внесением каких либо изменений необходимо сделать резервную копию файла настроек:

```
cp /etc/postgresql/${VERSION}/main/postgresql.conf  
/etc/postgresql/${VERSION}/main/postgresql.conf.bck
```

Для редактирования файла можно воспользоваться редактором `nano`:

```
nano /etc/postgresql/${VERSION}/main/postgresql.conf
```

Info

Основные команды текстового редактора `nano`:

- `Ctrl+W` - поиск текста;
- `Ctrl+\` - найти и заменить.
- `Ctrl+O` - запись файла;

- `Ctrl+X` - выход из редактора;

⚠ Warning

При изменении **любых конфигурационных файлов СУБД PostgreSQL** необходимо выполнить перезагрузку сервера:

```
pg_ctl reload
```

или

```
systemctl restart postgresql.service
```

```
user@dbms:~$ su
Пароль:
root@dbms:/home/user# ls /etc/postgresql/13/main/postgresql.conf
/etc/postgresql/13/main/postgresql.conf
root@dbms:/home/user# PG_CFG=/etc/postgresql/13/main/postgresql.conf
root@dbms:/home/user# cp $PG_CFG $PG_CFG.bck
root@dbms:/home/user# ls -l $PG_CFG*
/etc/postgresql/13/main/postgresql.conf
/etc/postgresql/13/main/postgresql.conf.bck
root@dbms:/home/user# nan$PG_CFG
```



Создание роли

Помимо настройки `listen_address` необходимо создать роль в СУБД. По умолчанию в `PostgreSQL` существует одна единственная роль - `postgres`.

Роль - это понятие, которое обобщает такие понятия как *пользователь* и *группа*, т.е. о ролях можно думать одновременно как о пользователях, так и группах.

Роли создаются со специализированными параметрами, которые накладывают какие-либо ограничения или дают дополнительные возможности по взаимодействию с `Postgres`.

Роль `postgres` имеет атрибут `NOLOGIN` - это означает, что для данной роли доступны подключения только по `UNIX-socket`. Осуществить подключение по `TCP` нельзя.

Новые роли создаются посредством выполнения `SQL` запроса в `psql`.

К примеру, данный `SQL` создаёт новую роль `new_user` с возможностью подключения к базам `PostgreSQL` по `TCP` и паролем `1234`:

```
CREATE ROLE new_user WITH LOGIN PASSWORD '1234';
```

Обратите внимание, что специализированные параметры роли записываются после ключевого слова `WITH`.

Можно добавить параметр `SUPERUSER`, который задает привилегии суперпользователя для создаваемой роли.

Полный список параметров представлен [в документации к PostgreSQL](#).

Важно заметить, что создавать новые роли или изменять старые могут только те роли, для которых установлен параметр `SUPERUSER`.

```
user@dbms:~$ su
Пароль:
root@dbms:/home/user# su postgres
postgres@dbms:/home/user$ nano /tmp/script.sql
postgres@dbms:/home/user$ cat /tmp/script.sql
```



Настройка `pg_hba.conf`

После создание роли с параметром `LOGIN` необходимо отредактировать файл:

```
/etc/postgresql/${VERSION}/main/pg_hba.conf
```

`pg_hba.conf` (host-based authentication) содержит правила аутентификации пользователей, подключаемых к определённой базе данных с определённых компьютеров (`host`).

Записи в этом файле задаются следующим образом

#	TYPE	DATABASE	USER	ADDRESS	METHOD
---	------	----------	------	---------	--------

1. Type (`host` | `local`) - тип подключения. Для `UNIX-socket` - `local`, а для `TCP` - `host`
2. DATABASE - база данных, к которой производится подключение. `all` соответствует всем базам (подключаться можно к любой).
3. USER - пользователь от которого происходит подключение. `all` соответствует всем пользователям (подключаться можно от

любого).

4. ADDRESS - ip адрес и маска (подробная информация изложена ниже).
5. METHOD - метод аутентификации, для TCP - md5 (для версий 13 и младше) и scram-sha-256 (для версии 14 и старше).

На параметре ADDRESS необходимо остановится подробнее.

Данный параметр может задавать как один компьютер, так и все компьютеры, входящие в локальную вычислительную сеть.

Например: имеется компьютер с которого необходимо осуществлять подключение к серверу баз данных и его ip - **192.168.0.2**, тогда значения параметра будет иметься следующий вид:

```
192.168.0.2/32
```

/32 - обозначает маску, для конкретного компьютера она соответствует 32 .

Если необходимо подключить всё компьютеры сети, запись будет иметь следующий вид:

```
192.168.0.0/24
```

192.168.0.0 - обозначает сеть (в конце 0), маска для сети будет соответствовать 24 .

Практическая часть

Установка программного обеспечения

Список необходимого программного обеспечения для ОС Windows :

1. VirtualBox
2. PuTTY
3. Git

4. WinSCP

5. DBeaver

В приложении 1 изложен порядок настройки рабочего окружения.

Задания

1. Выполнить установку программного обеспечения.
2. Настроить рабочее окружения.
3. Настройка доступа к Postgres для любого компьютера из вашей сети.
4. Создать роли admin с правами суперпользователя и с возможностью подключения по TCP .
5. Отредактировать соответствующие файлы конфигурации: postgresql.conf и pg_hba.conf .
6. Выполнить подключение к базе по TCP используя psql со следующими аргументами:

```
psql -U admin -h ${IP} -p ${PORT} -d postgres
```

- IP - необходимо запросить получить из операционной системы Debian Linux (см. Приложение 1)
- PORT - значение, значение которого берётся из postgresql.conf

Вопросы

1. Что такое PostgreSQL ?
2. Как получить PostgreSQL ? Сколько она стоит?
3. Что такое psql ?
4. Какие аргументы psql вы знаете и для чего используются?
5. Как посмотреть полный список аргументов psql ?
6. Что такое роли в PostgreSQL ?
7. Как создать роль?
8. Для чего нужны параметры SUPERUSER и LOGIN ?

9. Какие методы аутентификации пользователей PostgreSQL вам известны?
10. Предположим, что в PostgreSQL существует база test . Как открыть доступ для пользователя user к этой базе с компьютера, имеющего ip 192.168.11.5?
11. Как открыть доступ для всех пользователей из сети 192.168.2.0 к базе test ?
12. Как перезапустить PostgreSQL ?

ПРИЛОЖЕНИЕ 1. Настройка рабочего окружения

Монтирование образа виртуальной машины

1. Скачайте образ виртуальной машины и соберите его.

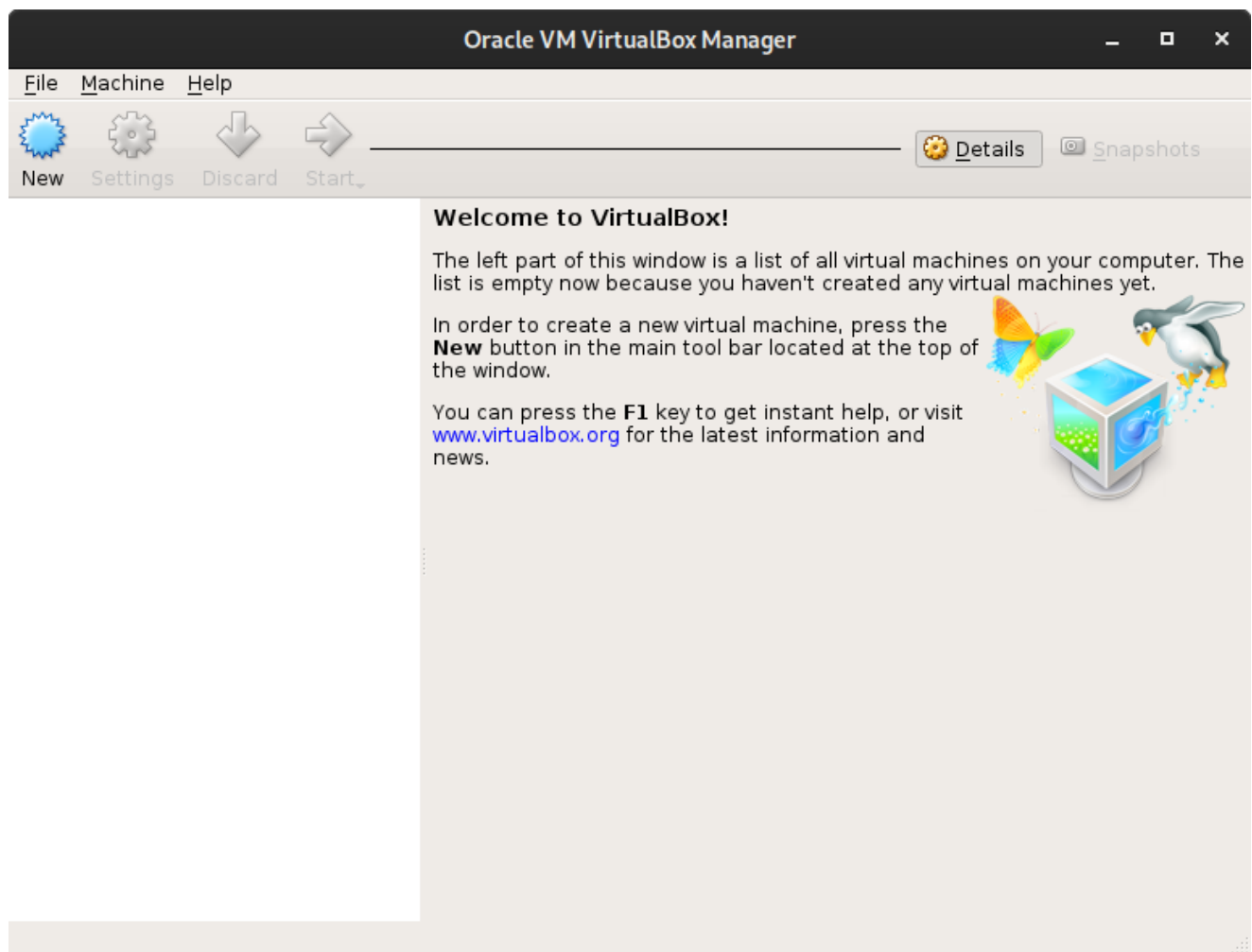
Info

По указанной ссылке доступны части vdi файла, который является виртуальным жестким диском для VirtualBox.

Сборка файла vdi

Для Windows необходимо запустить файл join_file.bat .
Для Linux и MacOS - join_file.sh .

3. Запустите VirtualBox



3. Создайте новую виртуальную машину

Create Virtual Machine



Name and operating system

Please choose a descriptive name for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name:


Type:

Microsoft Windows

Version:

Windows 7 (64-bit)

64




Expert Mode

< Back

Next >

Cancel

Create Virtual Machine



Name and operating system

Please choose a descriptive name for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name:

dbm


Type:

Linux

Version:

Debian (64-bit)

64



Expert Mode

< Back

Next >

Cancel

Create Virtual Machine



Memory size

Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.

The recommended memory size is **1024 MB**.

 MB

4 MB 8192 MB

< Back

Next >

Cancel

Create Virtual Machine



Hard disk

If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select one from the list or from another location using the folder icon.

If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

The recommended size of the hard disk is **8.00 GB**.

- ☐ Do not add a virtual hard disk
- ☐ Create a virtual hard disk now
- ☒ Use an existing virtual hard disk file

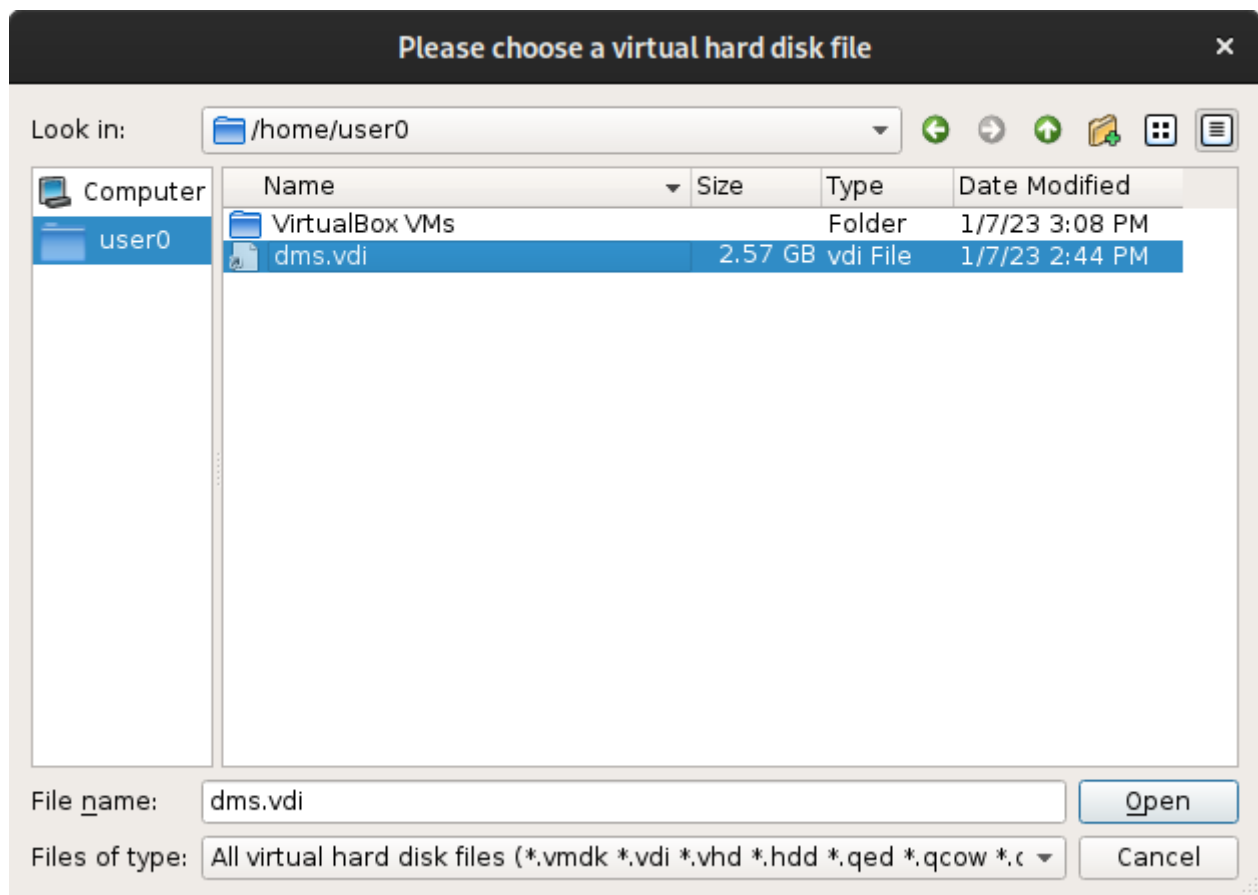
Empty

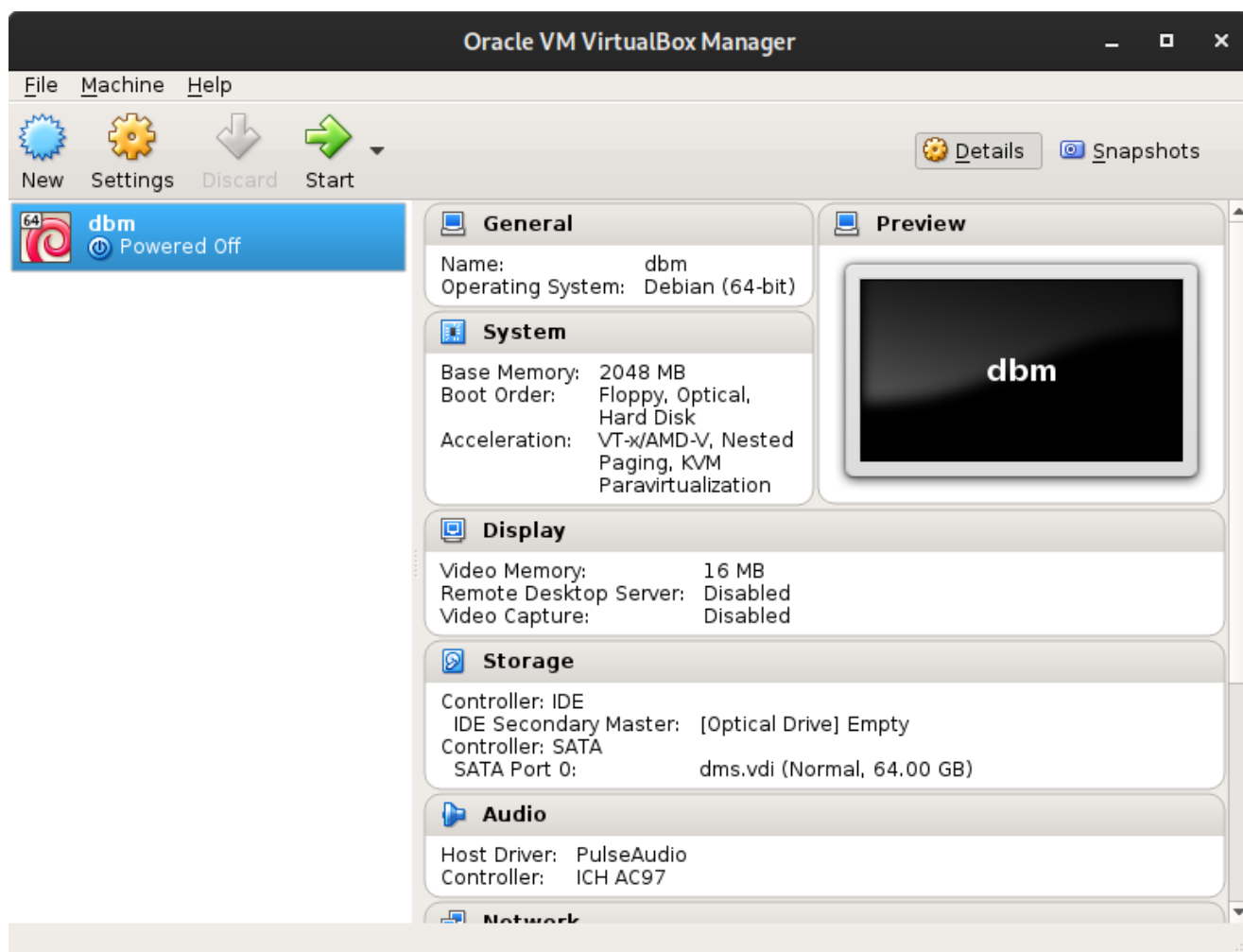


< Back

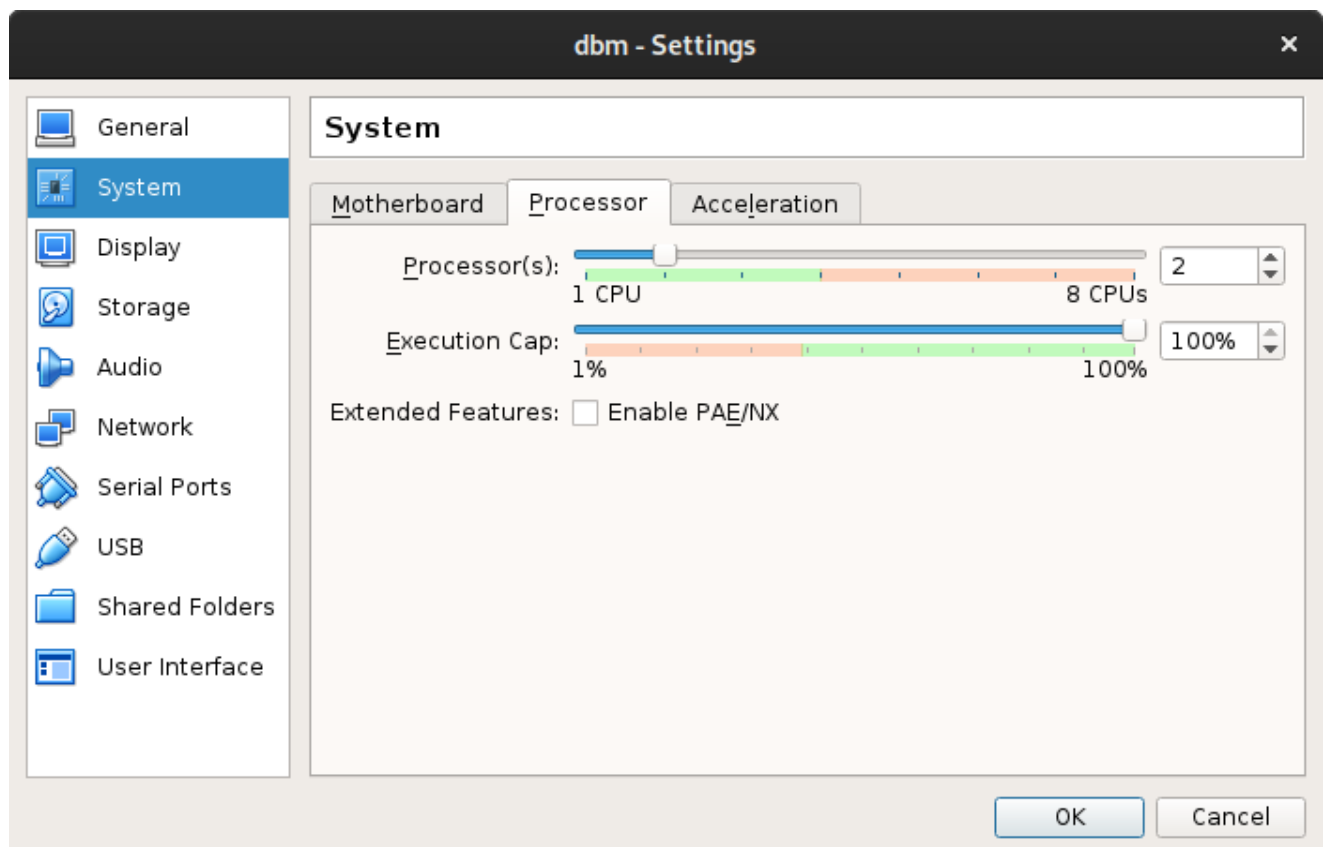
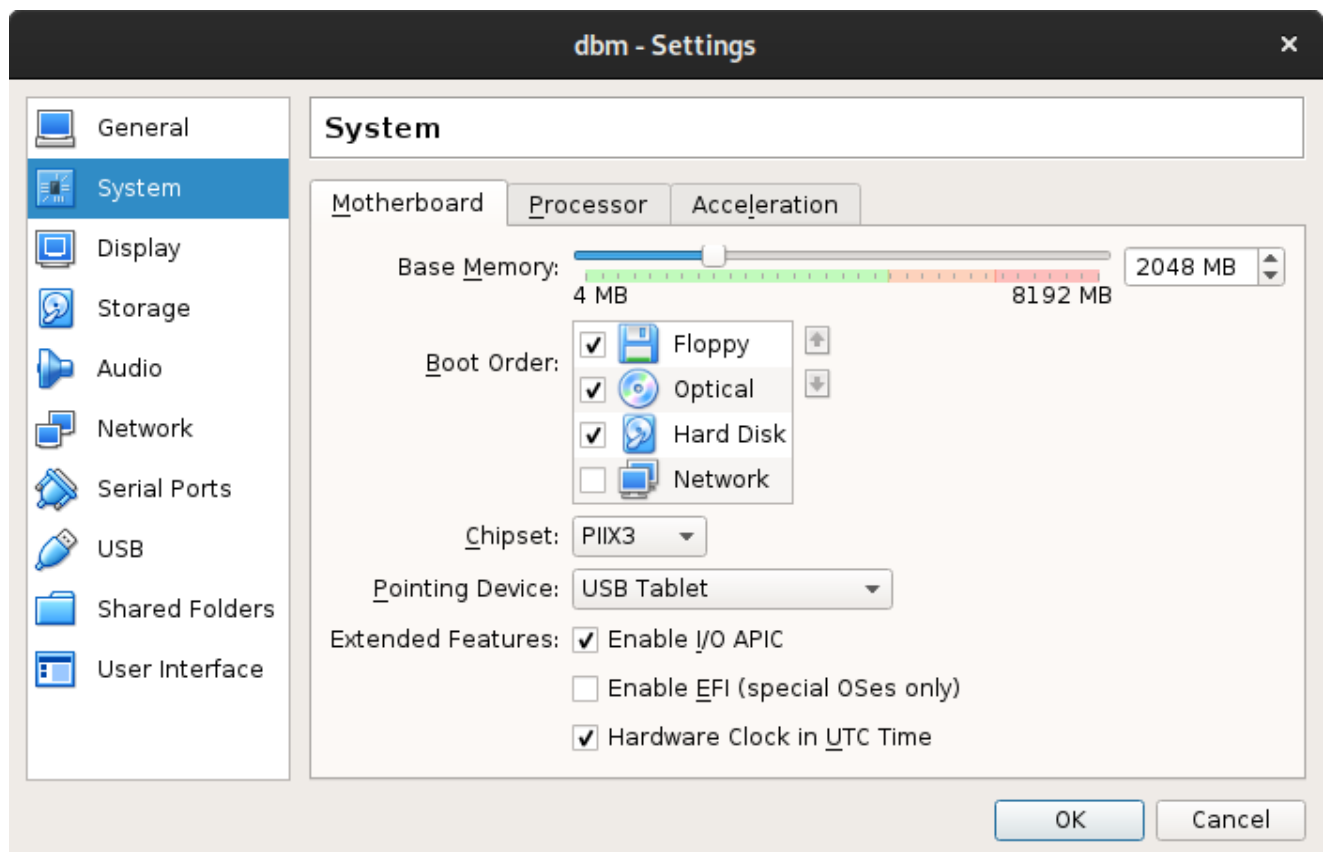
Create

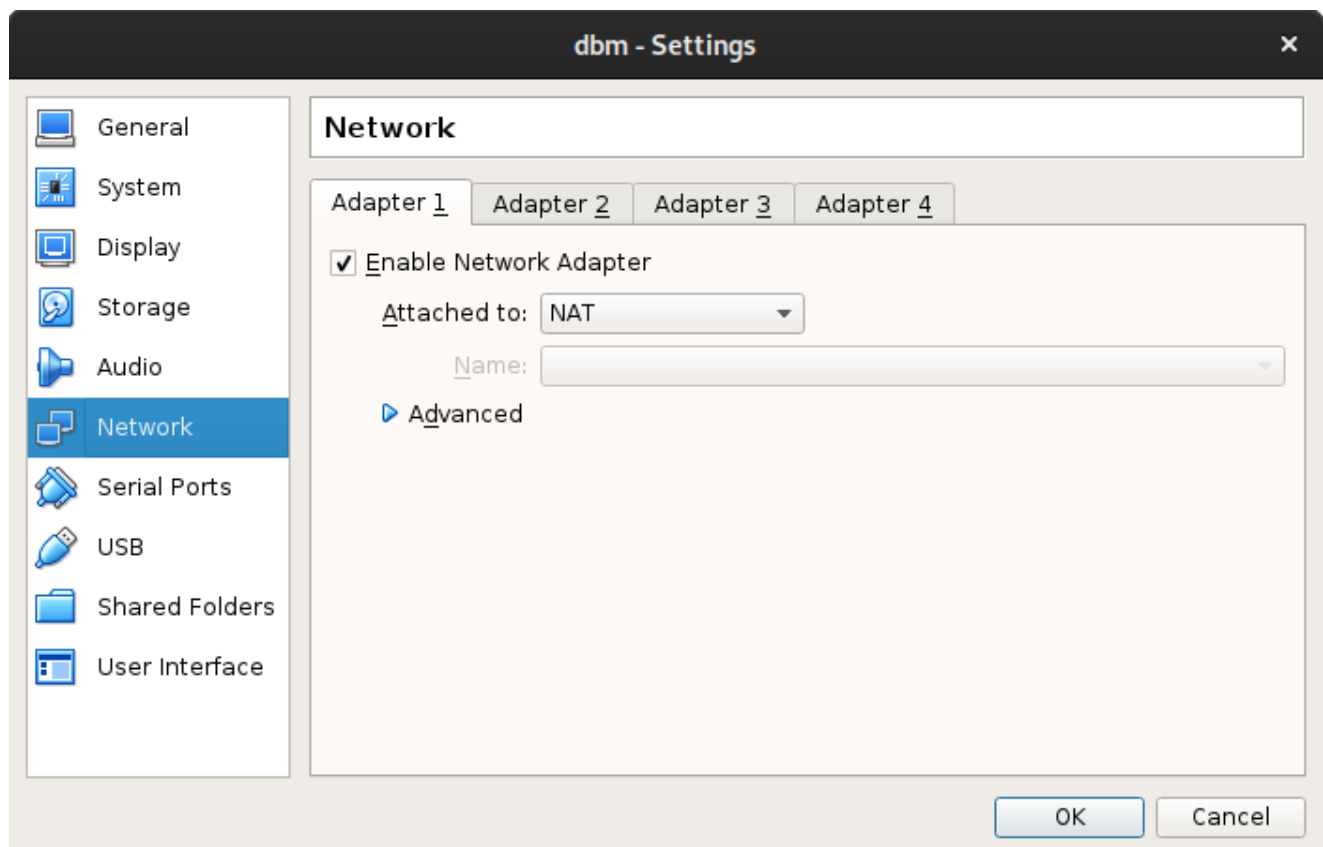
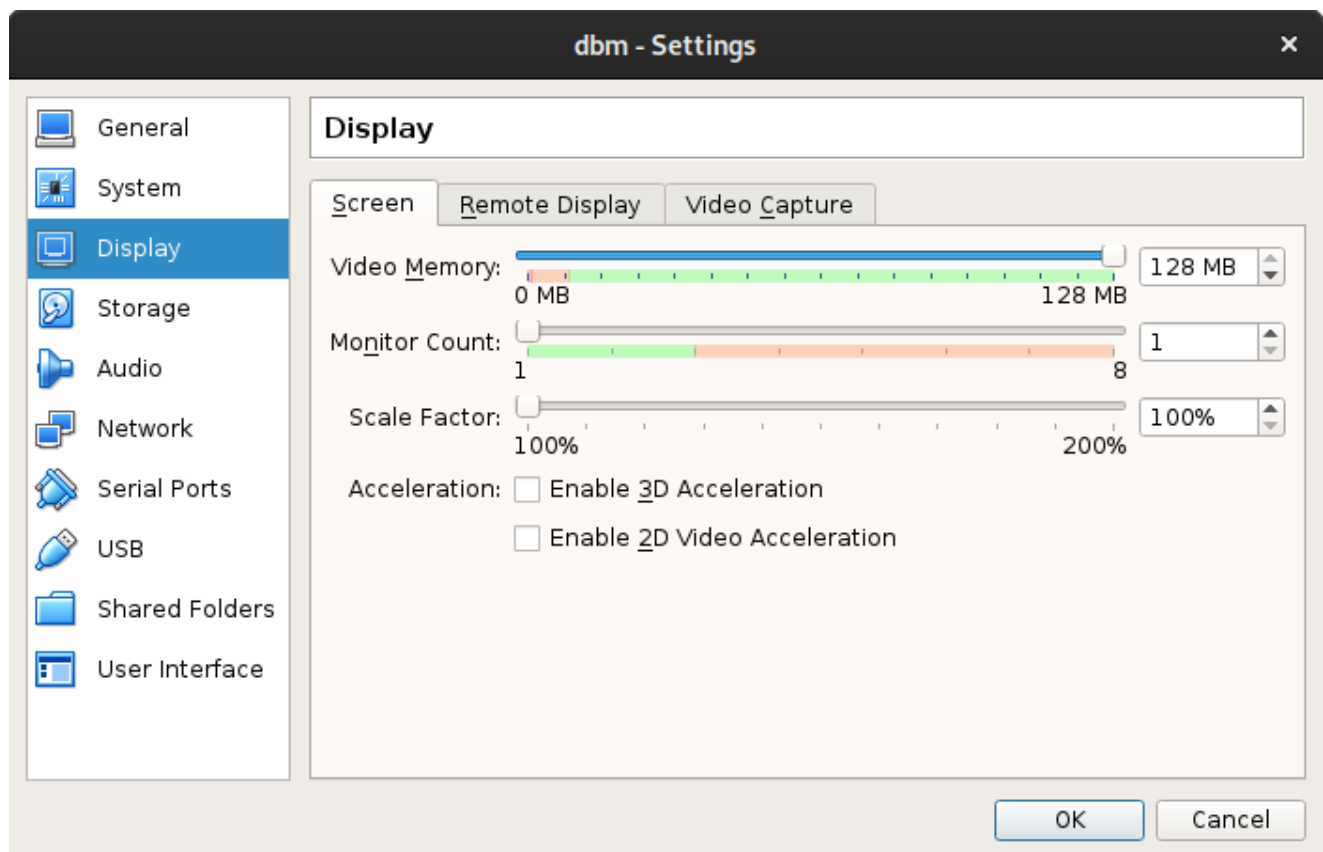
Cancel

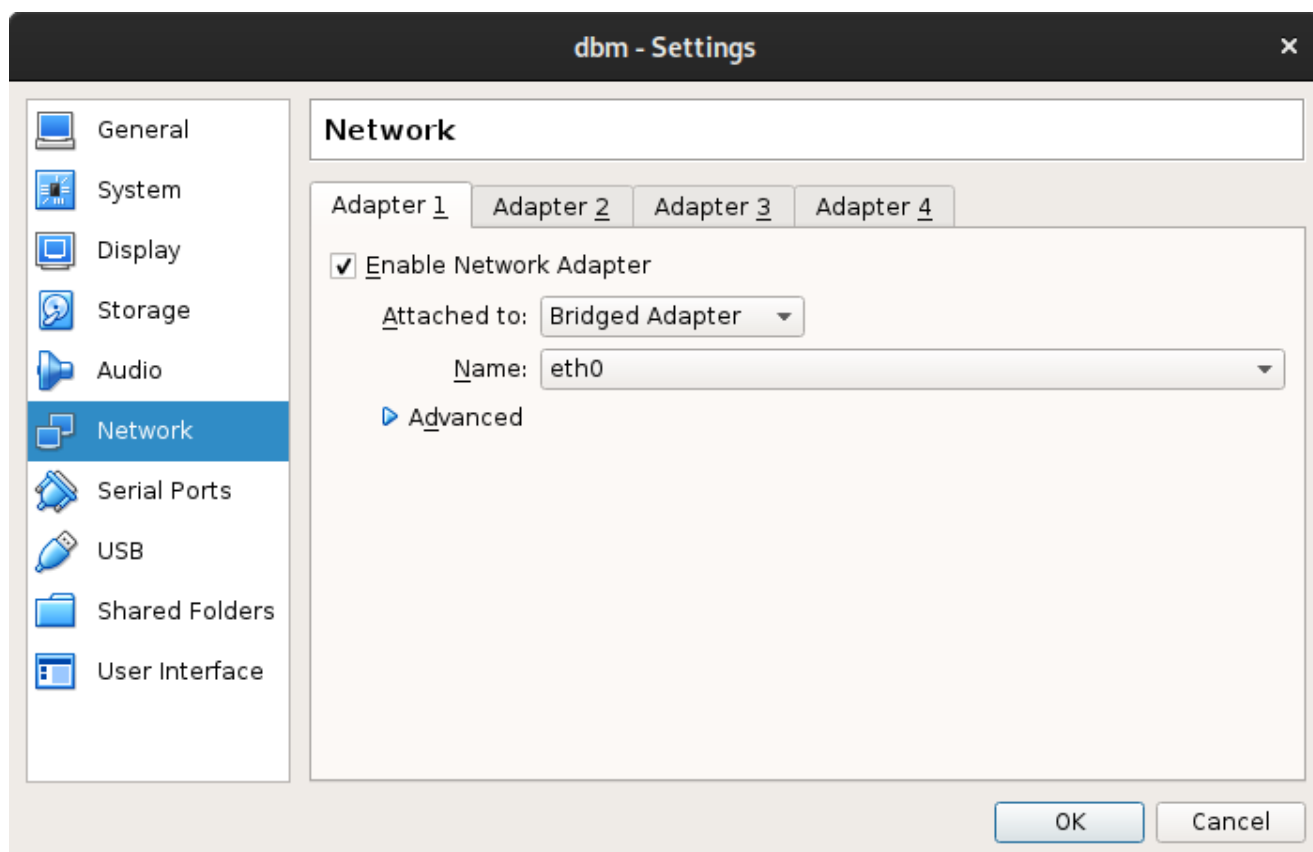




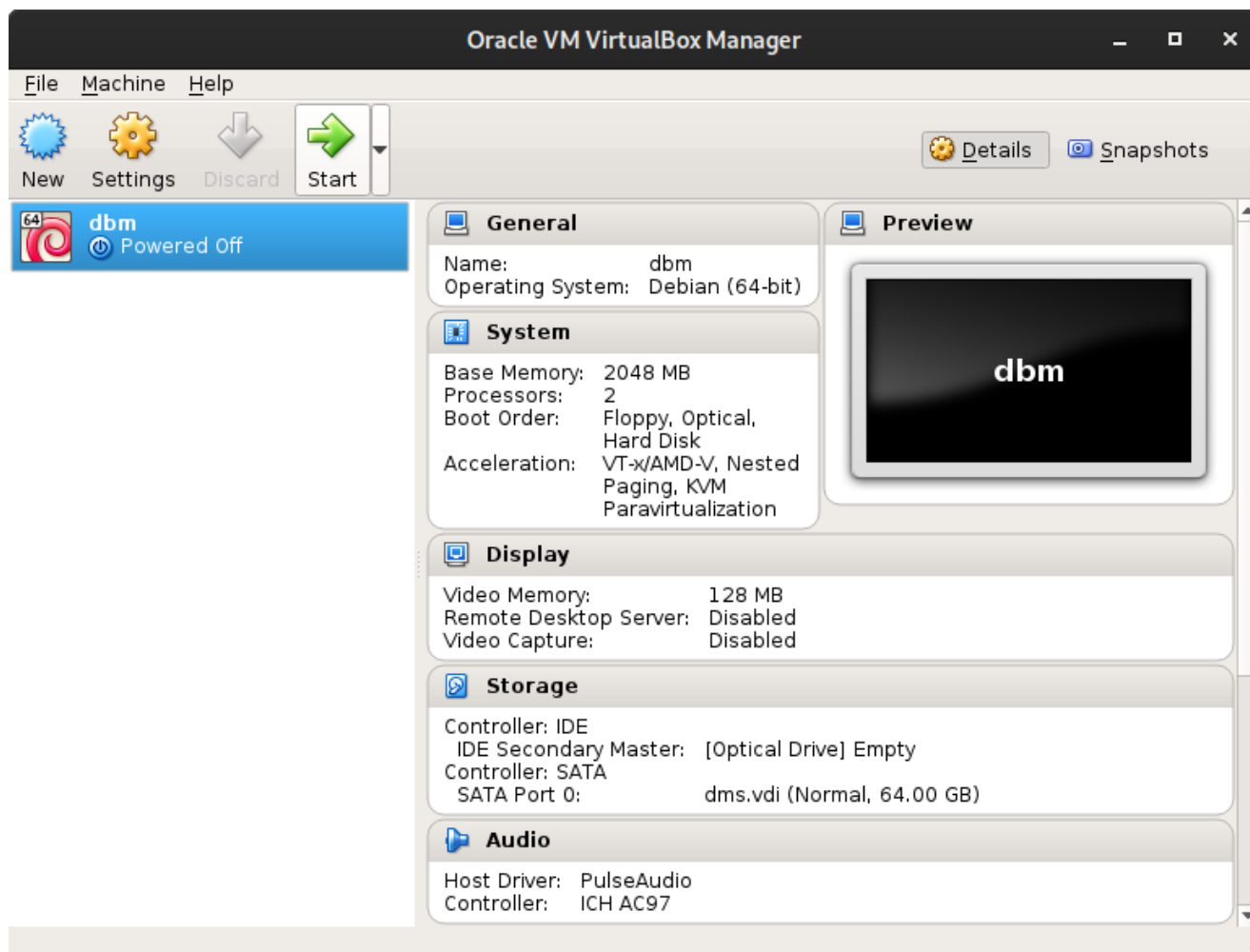
4. Выполните настройку виртуальной машины





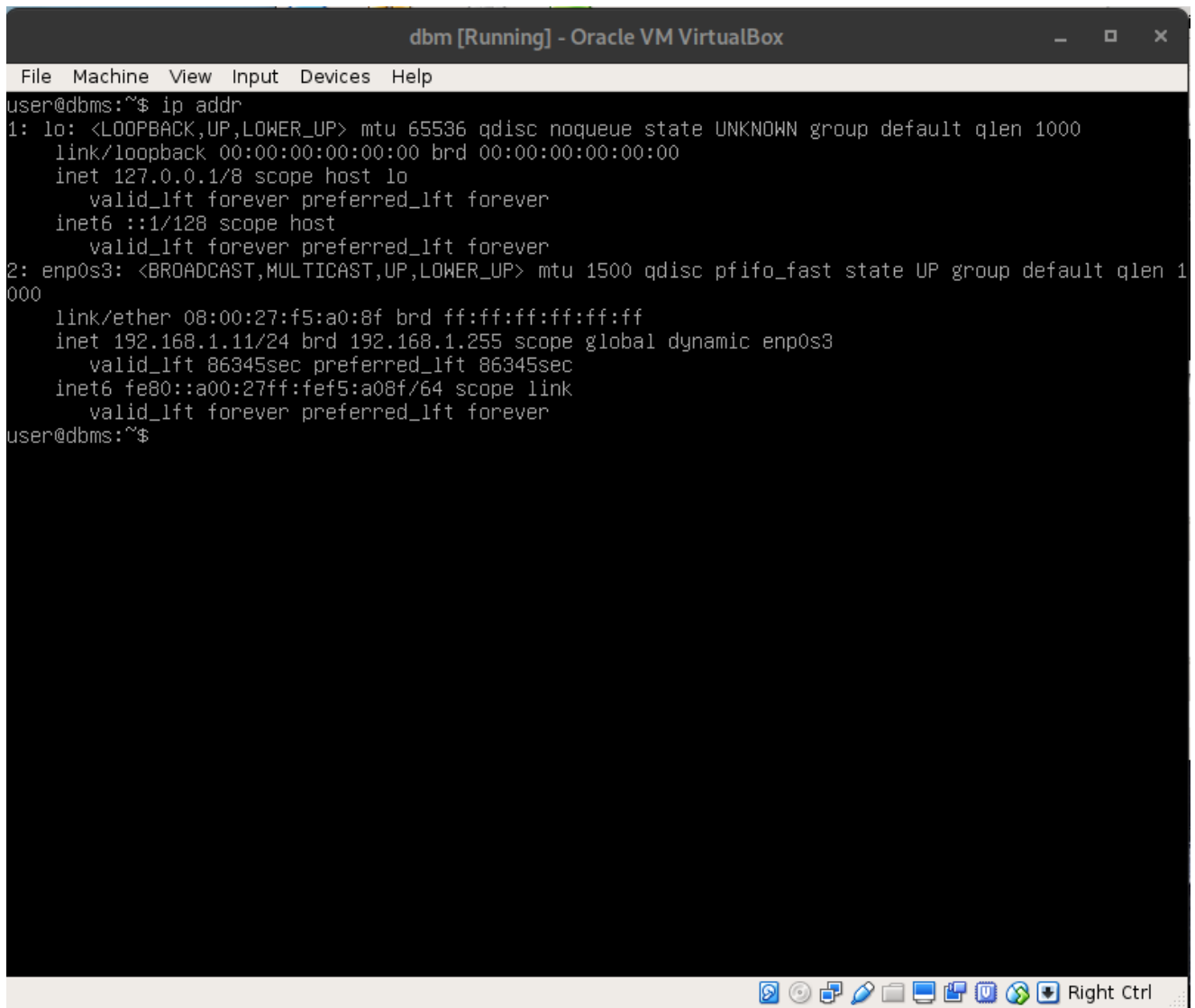


4. Запустите виртуальную машину



5. Выполните вход в систему и вывести ip адрес

```
login: user  
password: 1234
```



The screenshot shows a terminal window titled "dbm [Running] - Oracle VM VirtualBox". The terminal output is as follows:

```
user@dbms:~$ ip addr  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host  
        valid_lft forever preferred_lft forever  
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000  
    link/ether 08:00:27:f5:a0:8f brd ff:ff:ff:ff:ff:ff  
    inet 192.168.1.11/24 brd 192.168.1.255 scope global dynamic enp0s3  
        valid_lft 86345sec preferred_lft 86345sec  
    inet6 fe80::a00:27ff:fef5:a08f/64 scope link  
        valid_lft forever preferred_lft forever  
user@dbms:~$
```

📌 Получение IP адреса

В командной оболочке сервера (виртуальной машины) выполните следующую команду:

```
ip addr show enp0s3
```

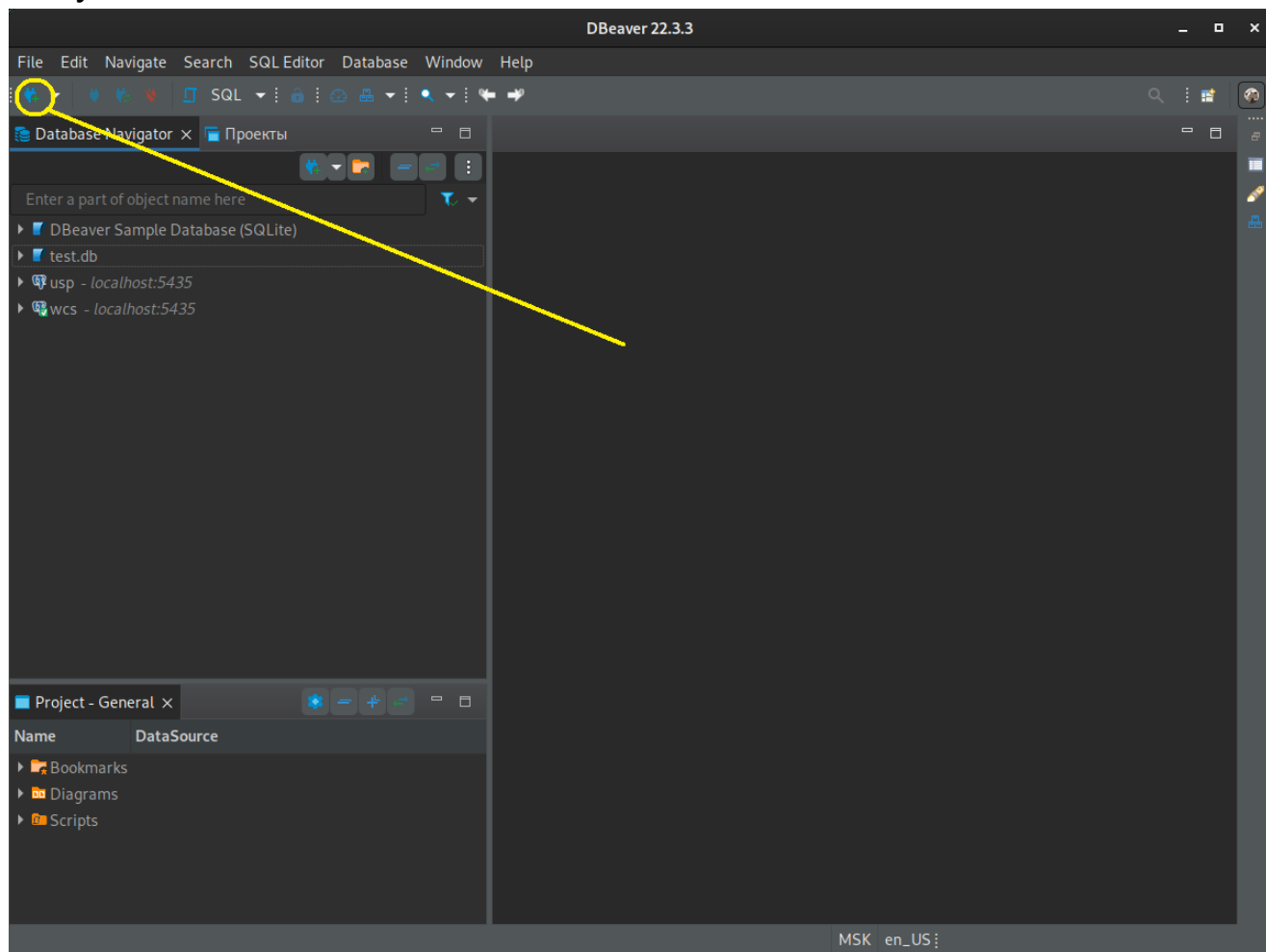
и найдите строчку начинающуюся с `inet`

```
user@dbms:~$ ip addr
address    addrlabel
user@dbms:~$ ip addr
address    addrlabel
user@dbms:~$ ip addr
add        change  del      flush    help     replace  show
user@dbms:~$ ip addr
```

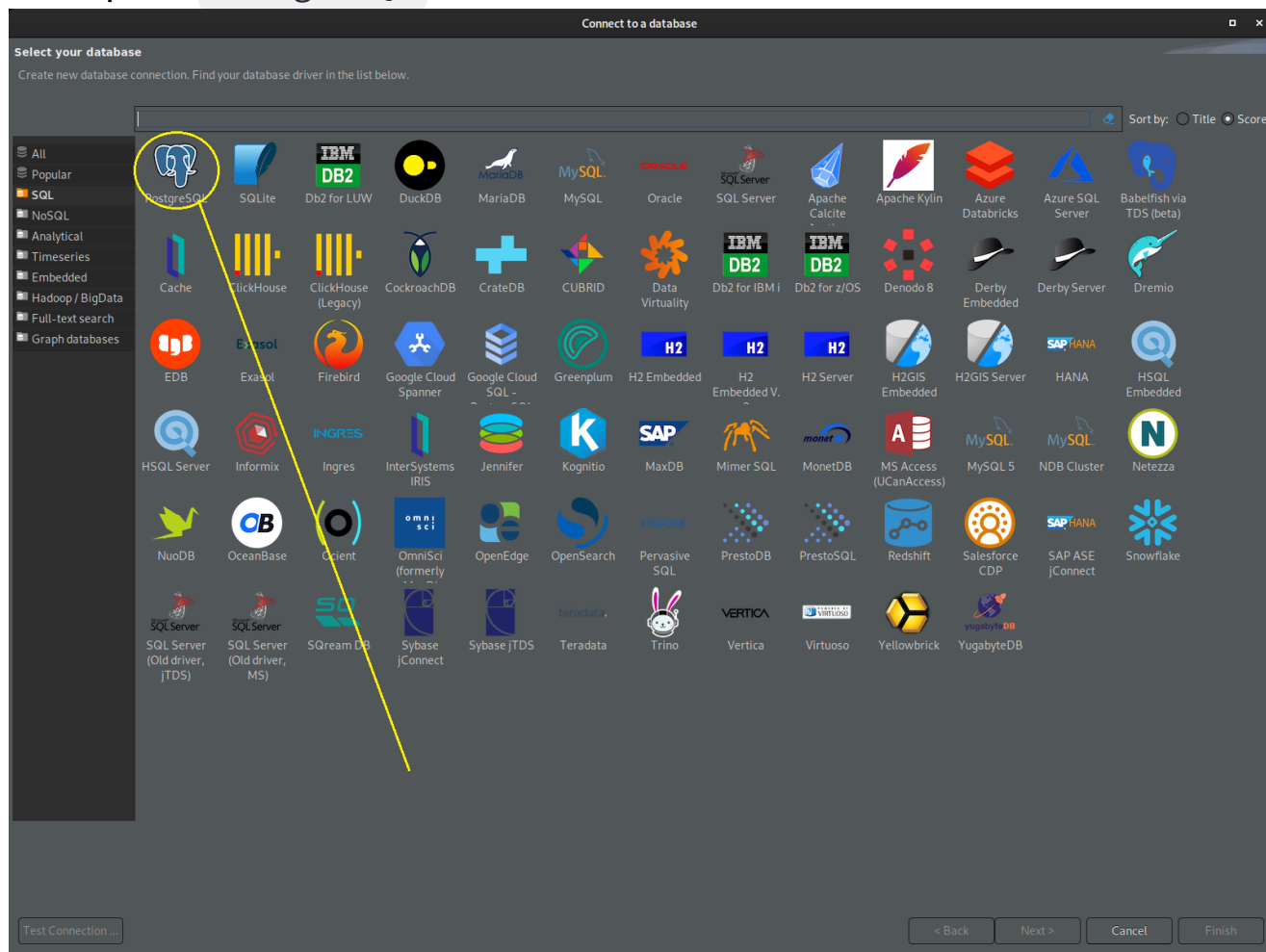


Настройка DBeaver

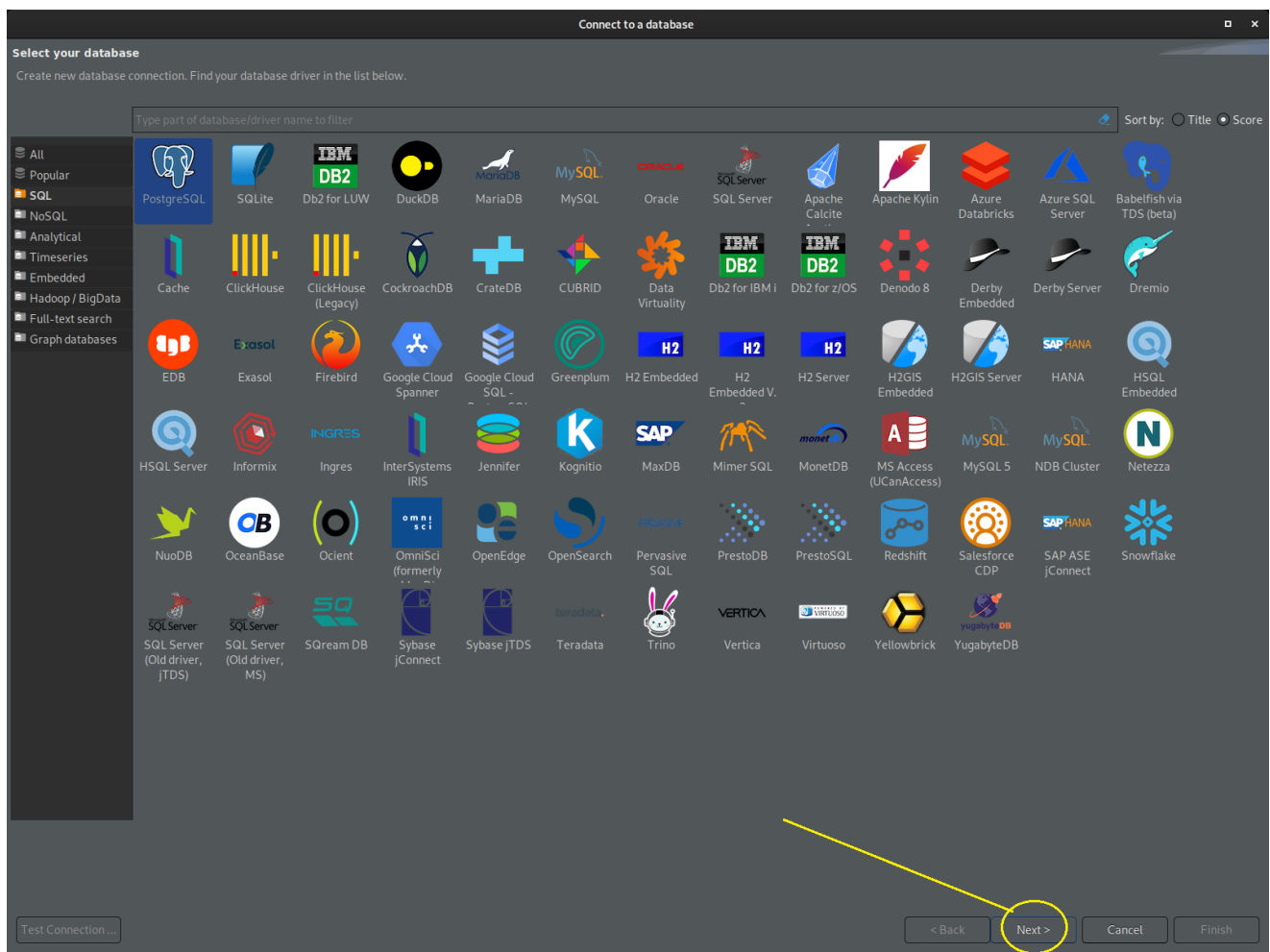
Запустите **DBeaver** и создайте новое соединение



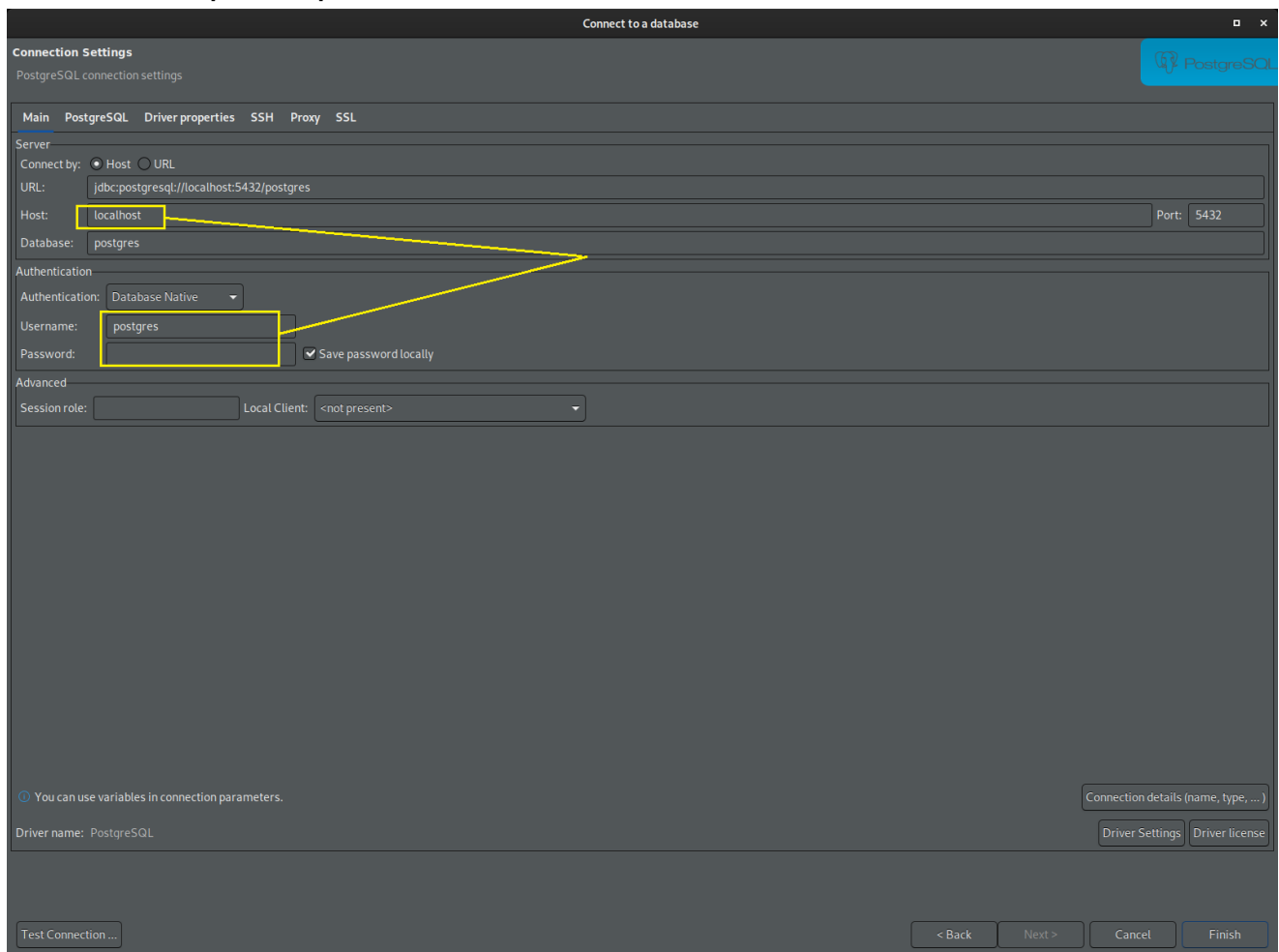
Выберите PostgreSQL из списка баз данных:



Нажмите next и перейдите к следующему экрану конфигурации соединения с PostgreSQL



Укажите параметры соединения:



Впишите IP адрес вашего сервера и нажмите **finish** :

Connect to a database

Connection Settings
PostgreSQL connection settings

Main PostgreSQL Driver properties SSH Proxy SSL

Server

Connect by: ☒ Host ☐ URL

URL: jdbc:postgresql://192.168.X.XX:5432/postgres

Host: 192.168.X.XX Port: 5432

Database: postgres

Authentication

Authentication: Database Native

Username: new_user

Password: ☒ Save password locally

Advanced

Session role: Local Client: <not present>

i You can use variables in connection parameters.

Driver name: PostgreSQL

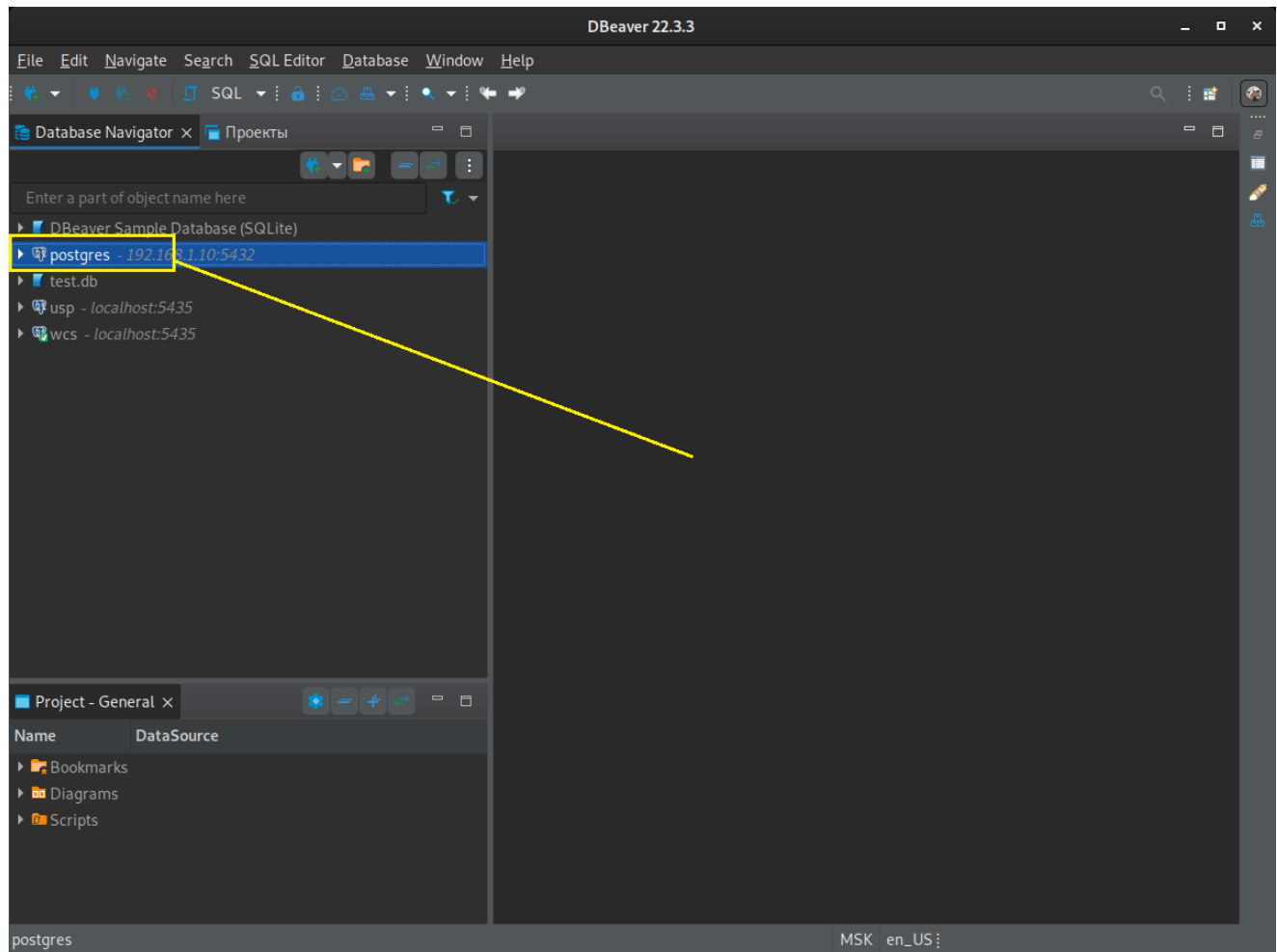
Connection details (name, type, ...)

Driver Settings Driver license

Test Connection ...

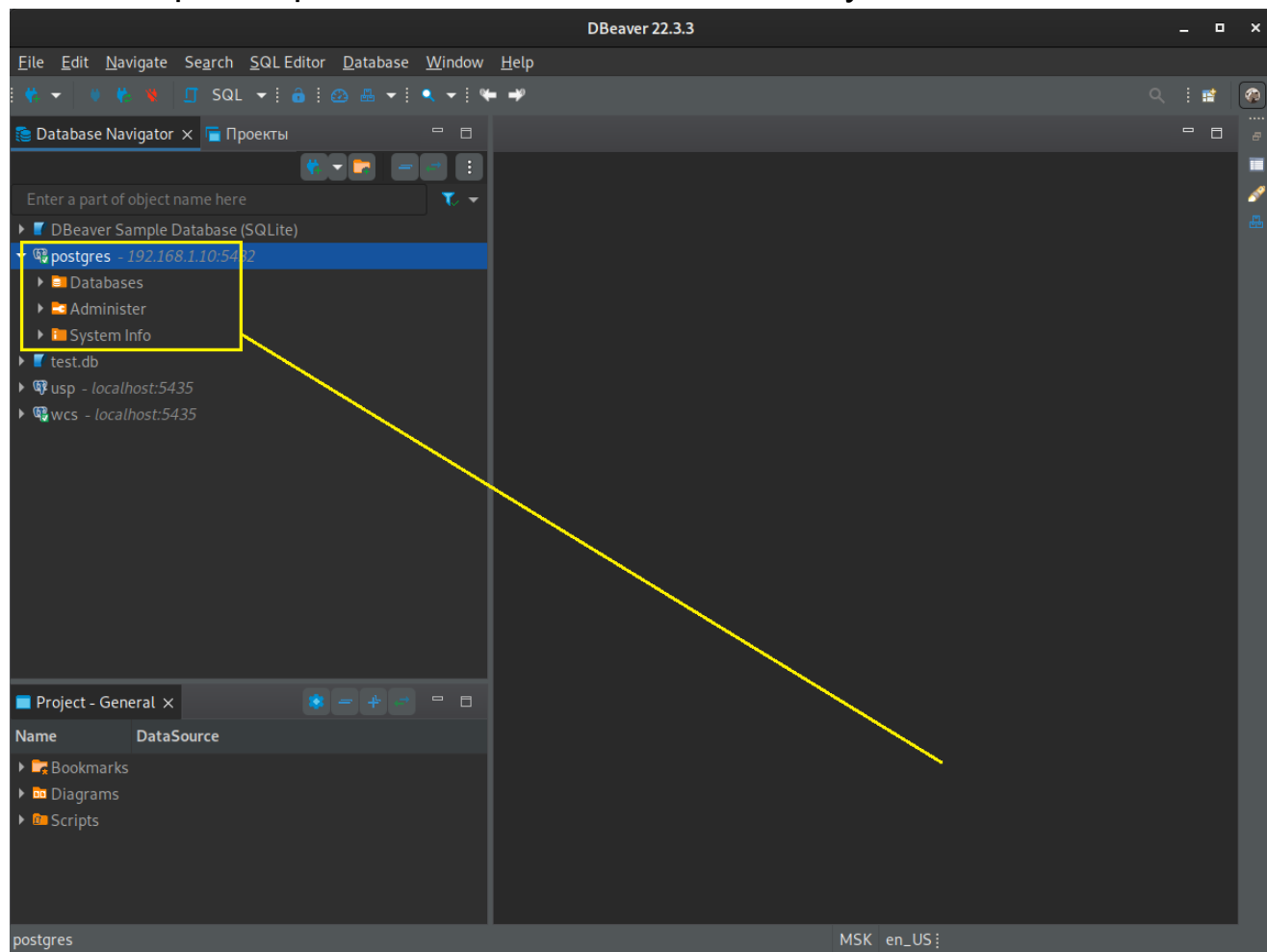
< Back Next > Cancel **Finish**

В боковой панели появится новое соединение, нажмите на него:

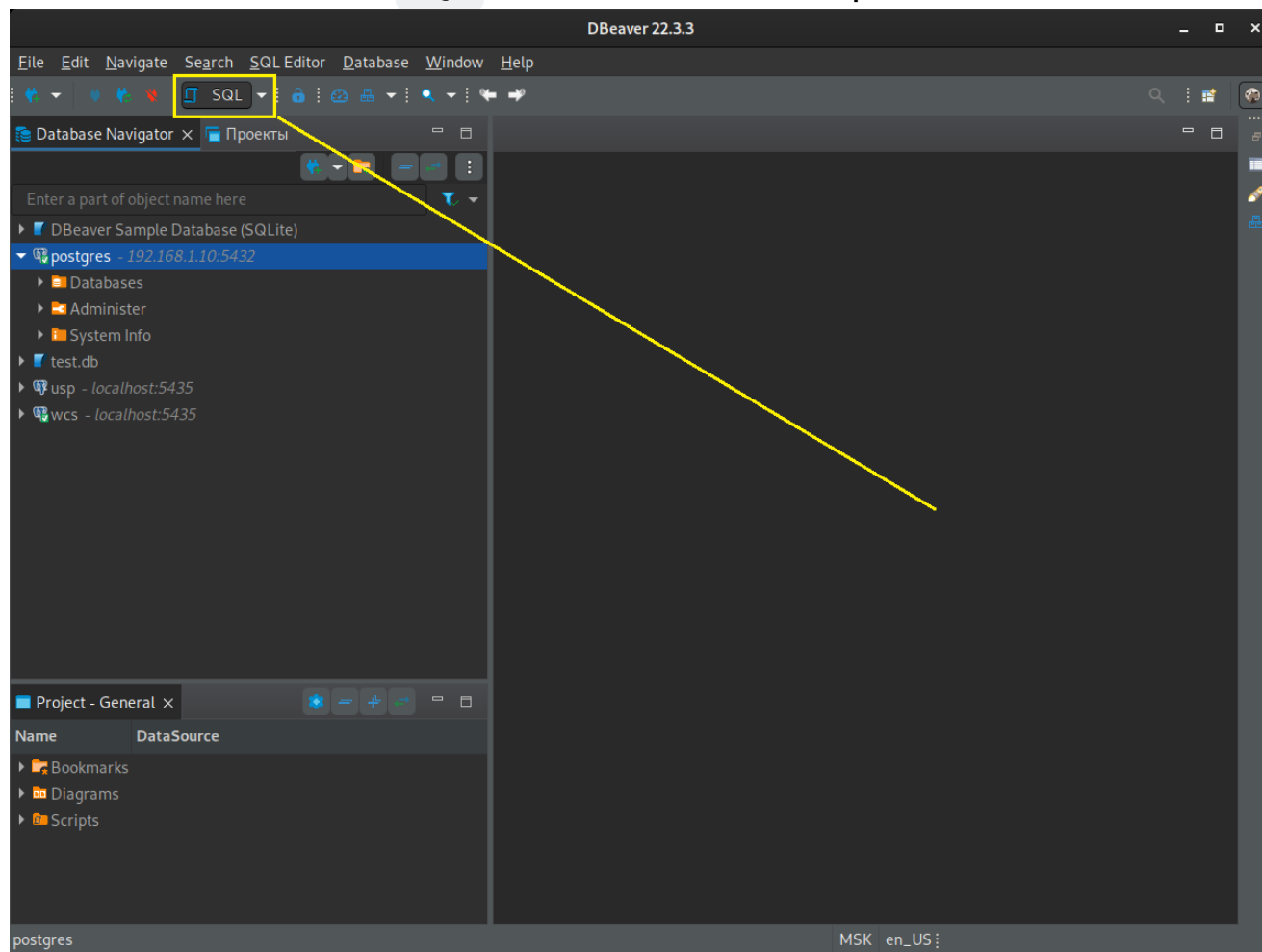


Если сервер настроен корректно и все конфигурационные параметры соединения указаны правильно, то можно будет увидеть зеленый

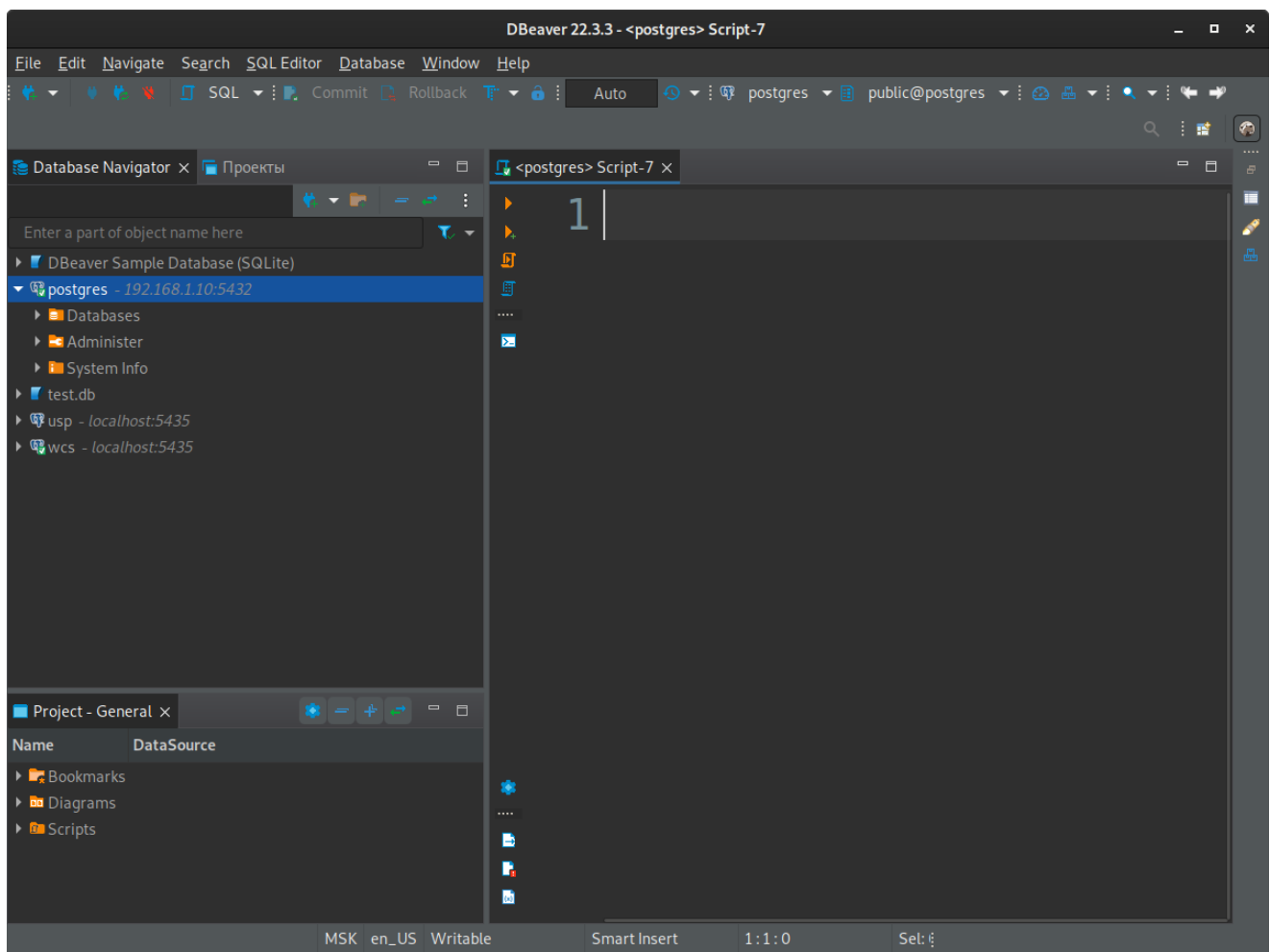
индикатор, который означает, что соединение установлено:



Нажмите на элемент **SQL** для создания сценария:



Воспользуйтесь появившейся вкладке редактора для написания **SQL** запросов:



К примеру, напишите запрос, выводящий текущую дату и время.
Для выполнения SQL нажмите **Ctrl + Enter**. Результат выполнения

ПОЯВИТЬСЯ В НИЖНЕЙ ПАНЕЛИ:

