2.3.2

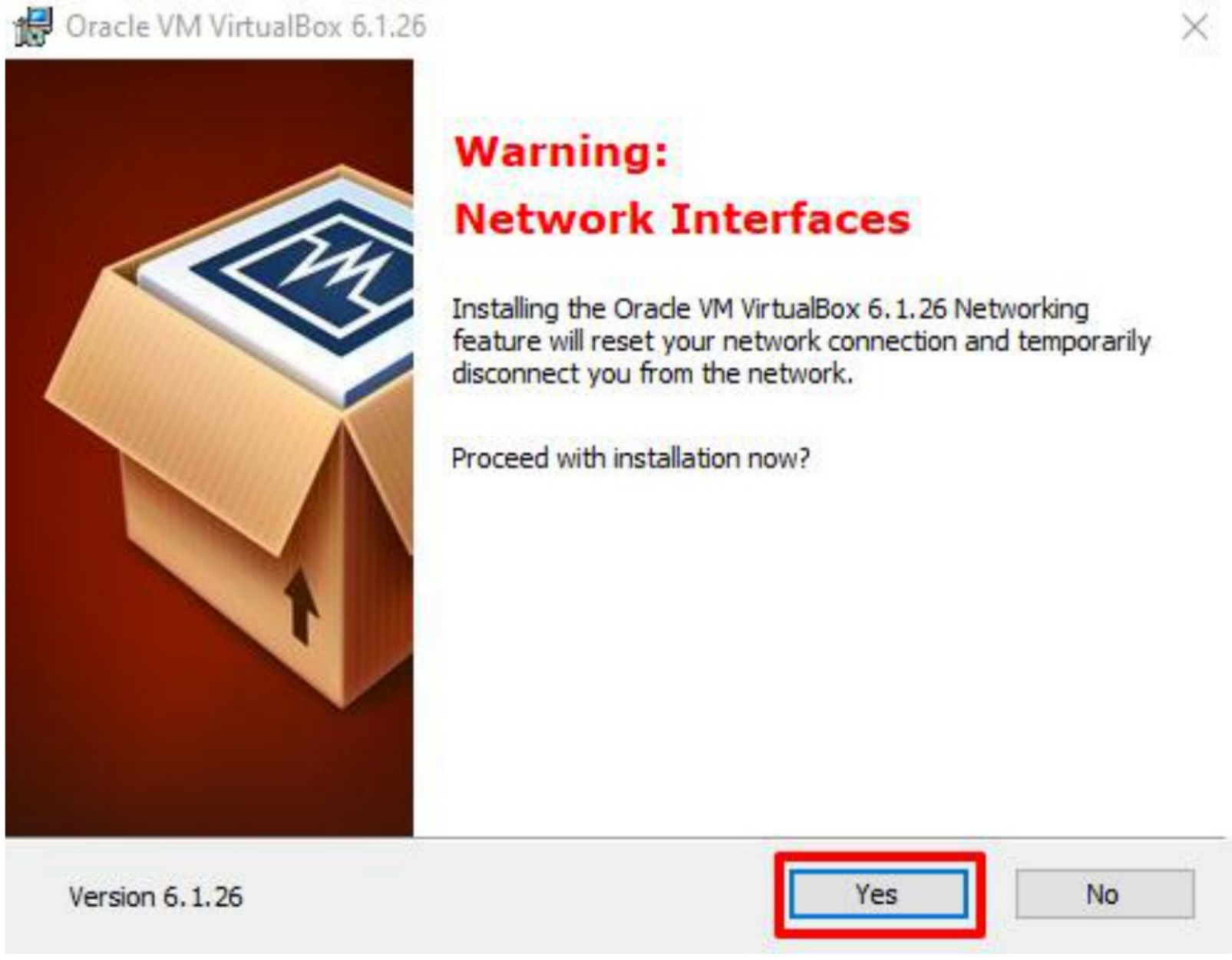
Установим VirtualBox

Заходим на сайт https://www.virtualbox.org/wiki/Downloads

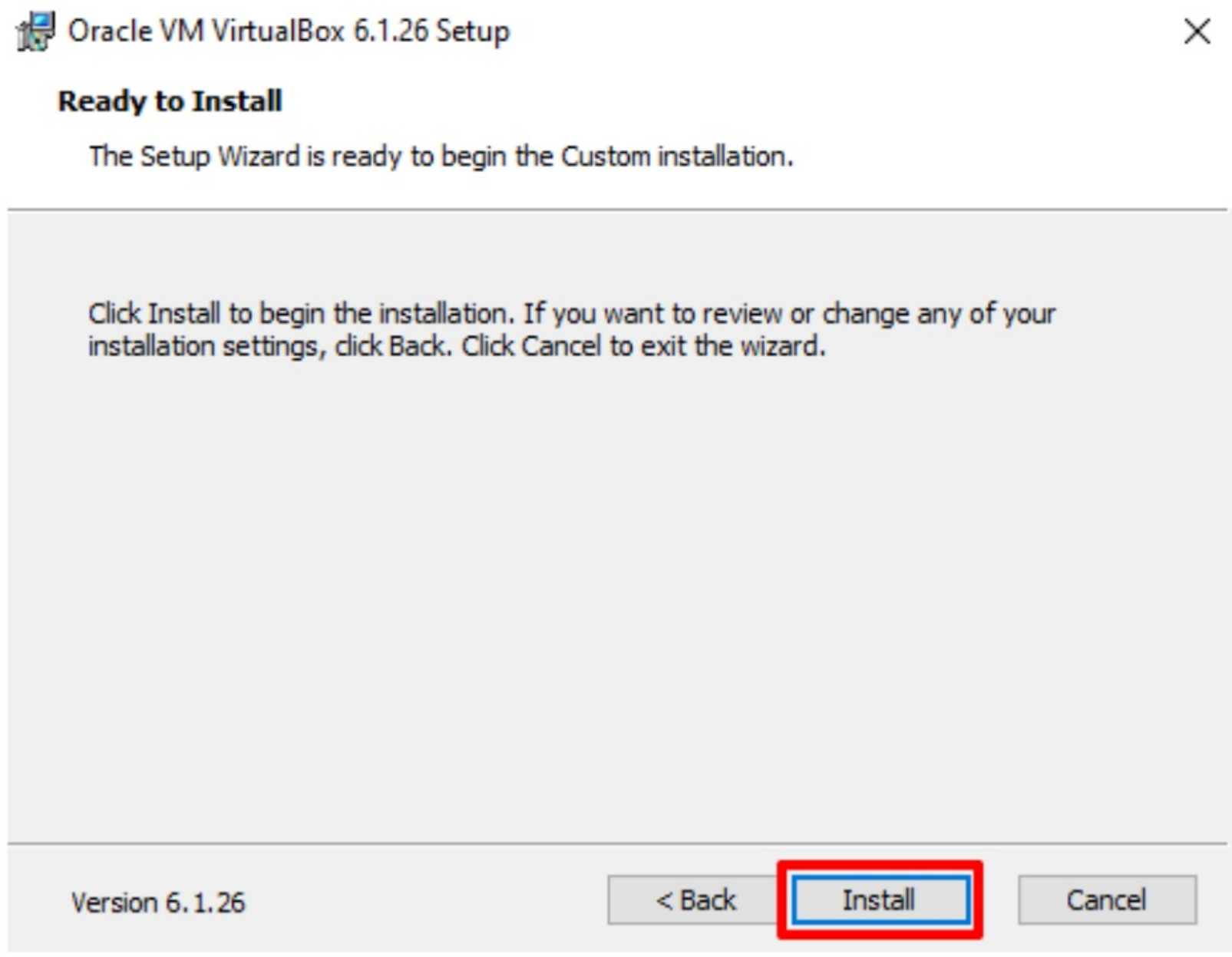
Скачиваем нужный дистрибутив для вашей ОС.

Потом запускаем скачанный файл с правами администратора на установку и жмем **Далее\Next**.

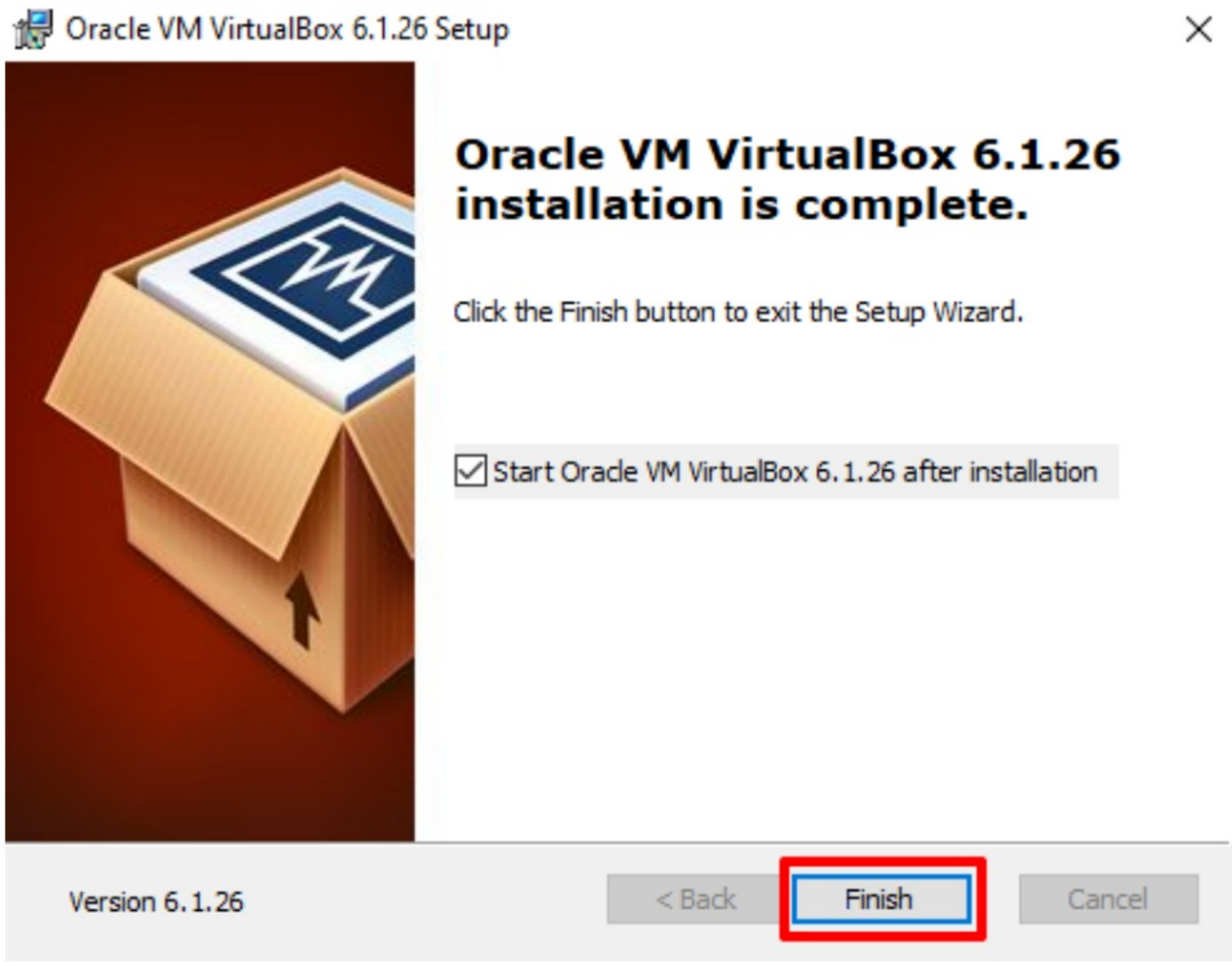
Соглашаемся с предупреждением **Yes**.



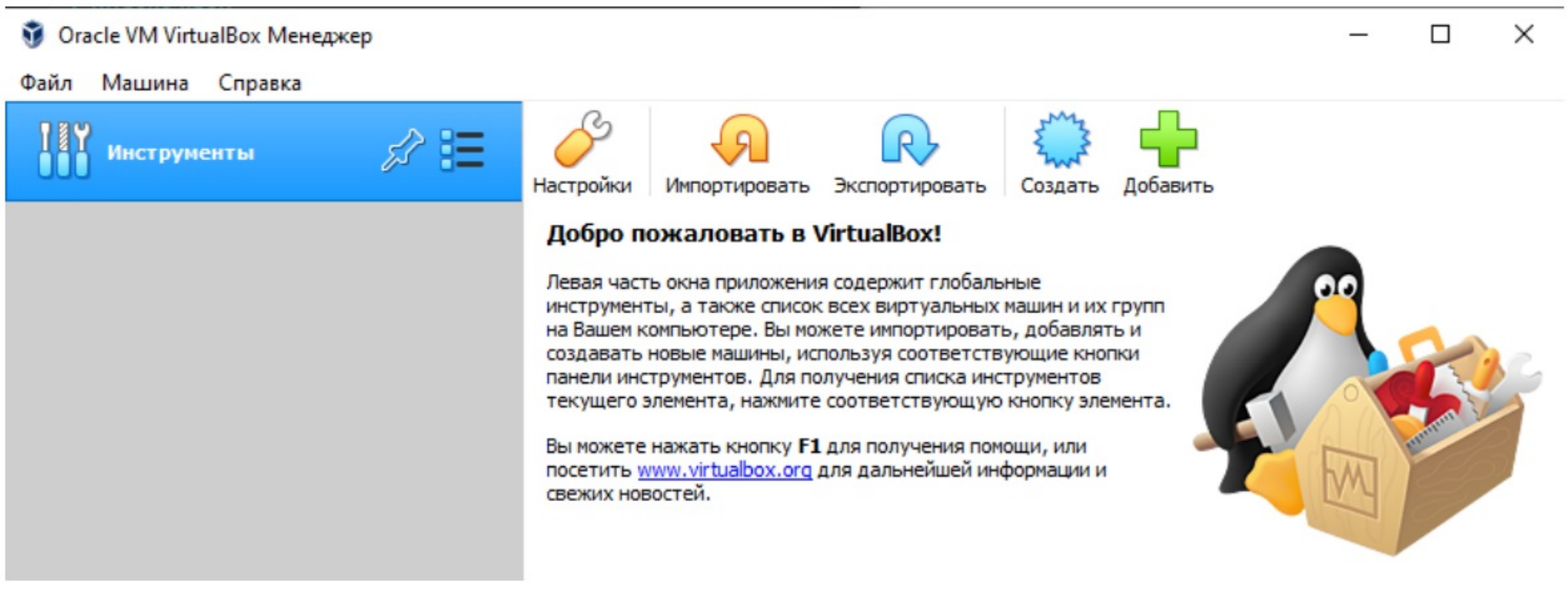
Запускаем установку **Install**



После успешной установки будет окно, где жмём **Finish**.



Если не убрали галочку «старт программы после установки», она запустится сразу.



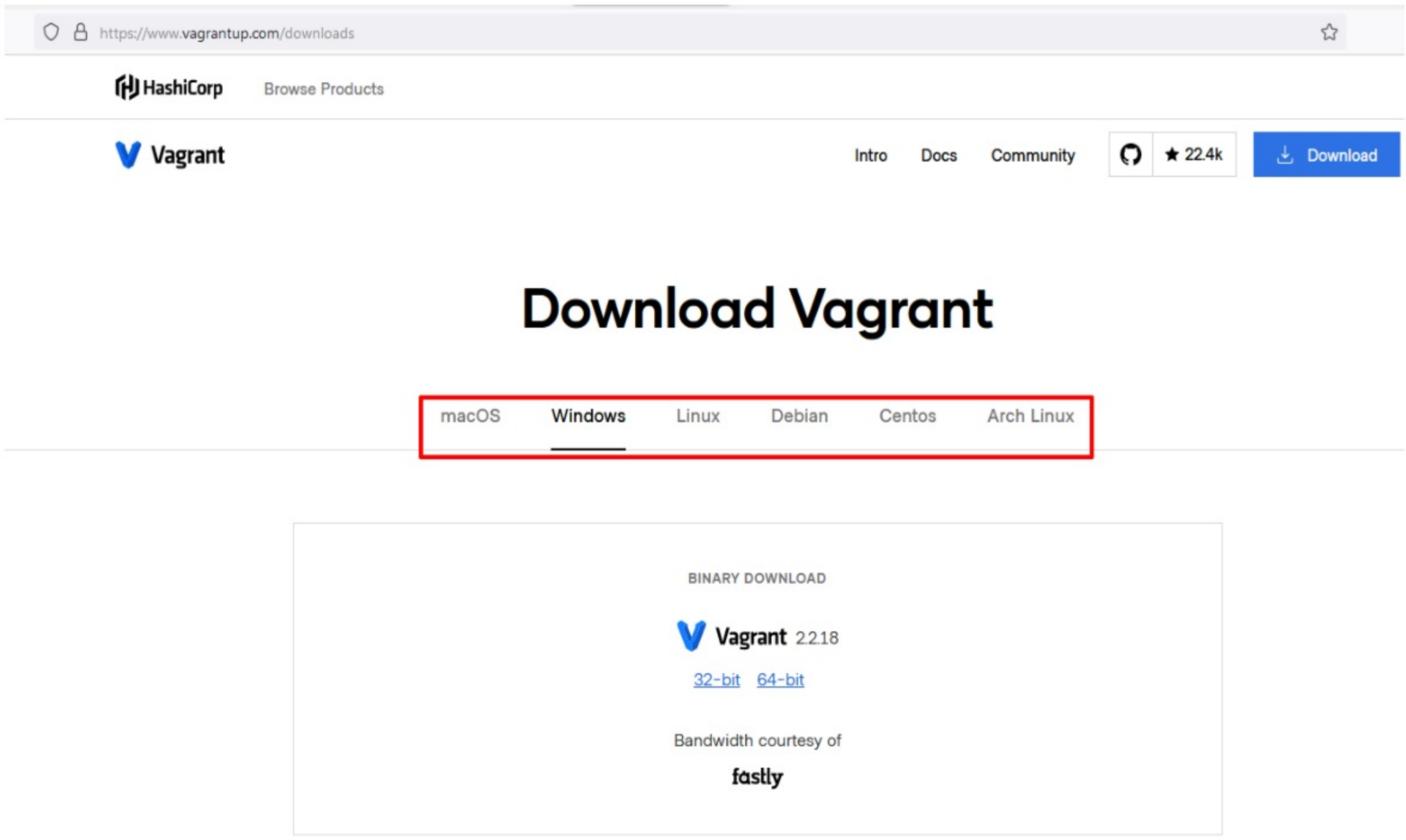
И мы видим заветное окно интерфейса VirtualBox. Погнали в следующий шаг!

2.3.3

Установим Vargant

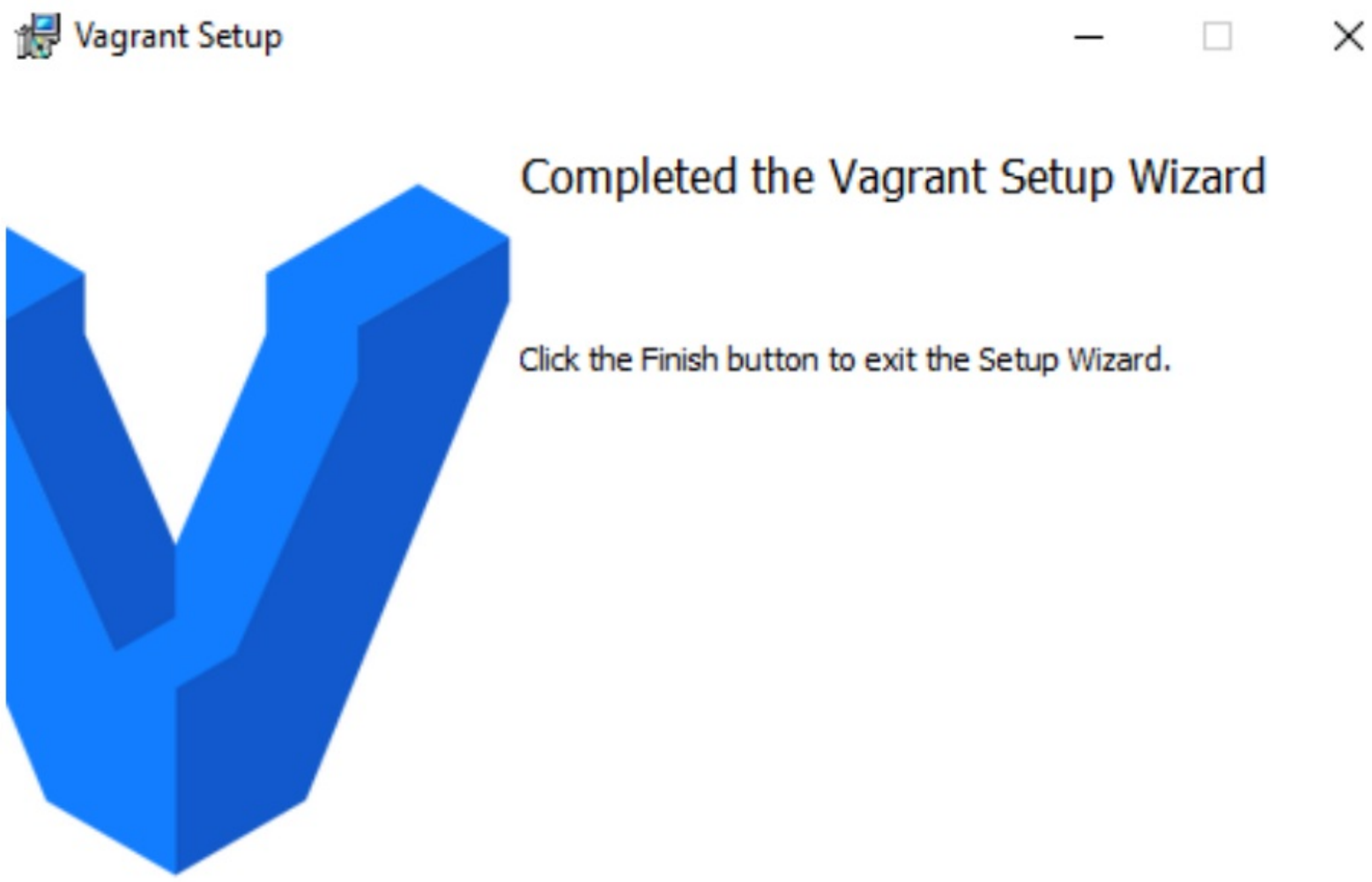
Переходим на сайт - https://www.vagrantup.com/downloads

Скачиваем дистрибутив, подходящий под вашу ОС.



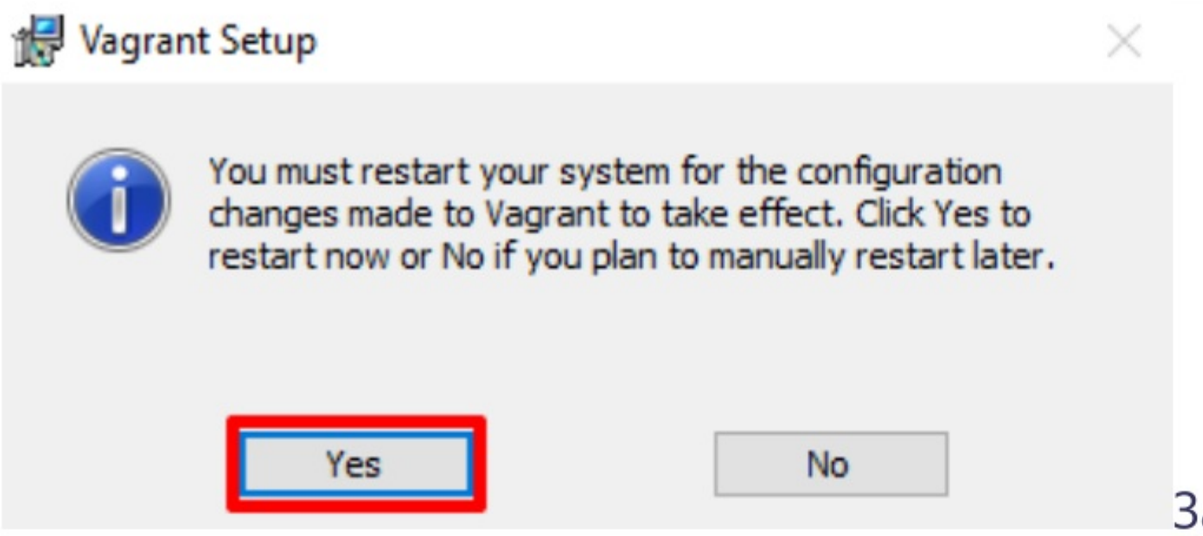
Запускаем файл на установку с правами администратора и жмём **Далее/Next**, попутно принимаем соглашение.

И жмём в конце заветную **Install**. На этот раз установка будет не такая быстрая, как VirtualBox. Чай, кофе, потанцуем?



Установка завершена, жмём **Finish**.

Тут ещё не всё, выйдет окно с запросом на перезагрузку ПК. Выбираем **Yes**.



Загружаемся и подрубаемся снова.

2.3.4

Первая буква: L

Запускаем PowerShell ISE\Visual Studio Code, если у вас Windows, если нет, то другой терминал.

Создаём конфигурационный файл для Vagrant. И сразу задаём ему какой нужен нам образ.

```
vagrant init ubuntu/focal64

A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.
```

Проверяем статус виртуальной машины. Текущий запрос ищет конфигурационный файл Vagrant и смотрит состояние машины, исходя от этого файла.

```
vagrant status

Current machine states:

default                   not created (virtualbox)

The environment has not yet been created. Run `vagrant up` to
create the environment. If a machine is not created, only the
default provider will be shown. So if a provider is not listed,
then the machine is not created for that environment.
```

Смотрим конфигурацию файла.

```
cat .\Vagrantfile

# -*- mode: ruby -*-
# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure("2") do |config|
```

```
  # The most common configuration options are documented and commented below.
  # For a complete reference, please see the online documentation at
  # https://docs.vagrantup.com.

  # Every Vagrant development environment requires a box. You can search for
  # boxes at https://vagrantcloud.com/search.
  config.vm.box = "ubuntu/focal64"

...

  config.vm.network "forwarded_port", guest: 80, host: 8080

...
```

Тут всё верно должно быть, т.к. ранее запускали команду с наименованием образа - **config.vm.box**

config.vm.network "forwarded_port", guest: 80, host: 8080 - это нужно раскомментировать

Открываем файл в текстовом редакторе и раскомментируем.

```
notepad.exe .\Vagrantfile
```

При запуске команды `vagrant up` может выйти подобная ошибка. Vagrant не может получить доступ к общим папкам VirtualBox.

```
vagrant : Vagrant was unable to mount VirtualBox shared folders. This is usually
строка:1 знак:1
+ vagrant up
+ ~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (Vagrant was una...This is
usually:String) [], RemoteException
    + FullyQualifiedErrorId : NativeCommandError

because the filesystem "vboxsf" is not available. This filesystem is
made available via the VirtualBox Guest Additions and kernel module.
Please verify that these guest additions are properly installed in the
guest. This is not a bug in Vagrant and is usually caused by a faulty
Vagrant box. For context, the command attempted was:
mount -t vboxsf -o uid=1000,gid=1000,_netdev vagrant /vagrant
The error output from the command was:
/sbin/mount.vboxsf: mounting failed with the error: Invalid argument
```

Решается установкой плагина.

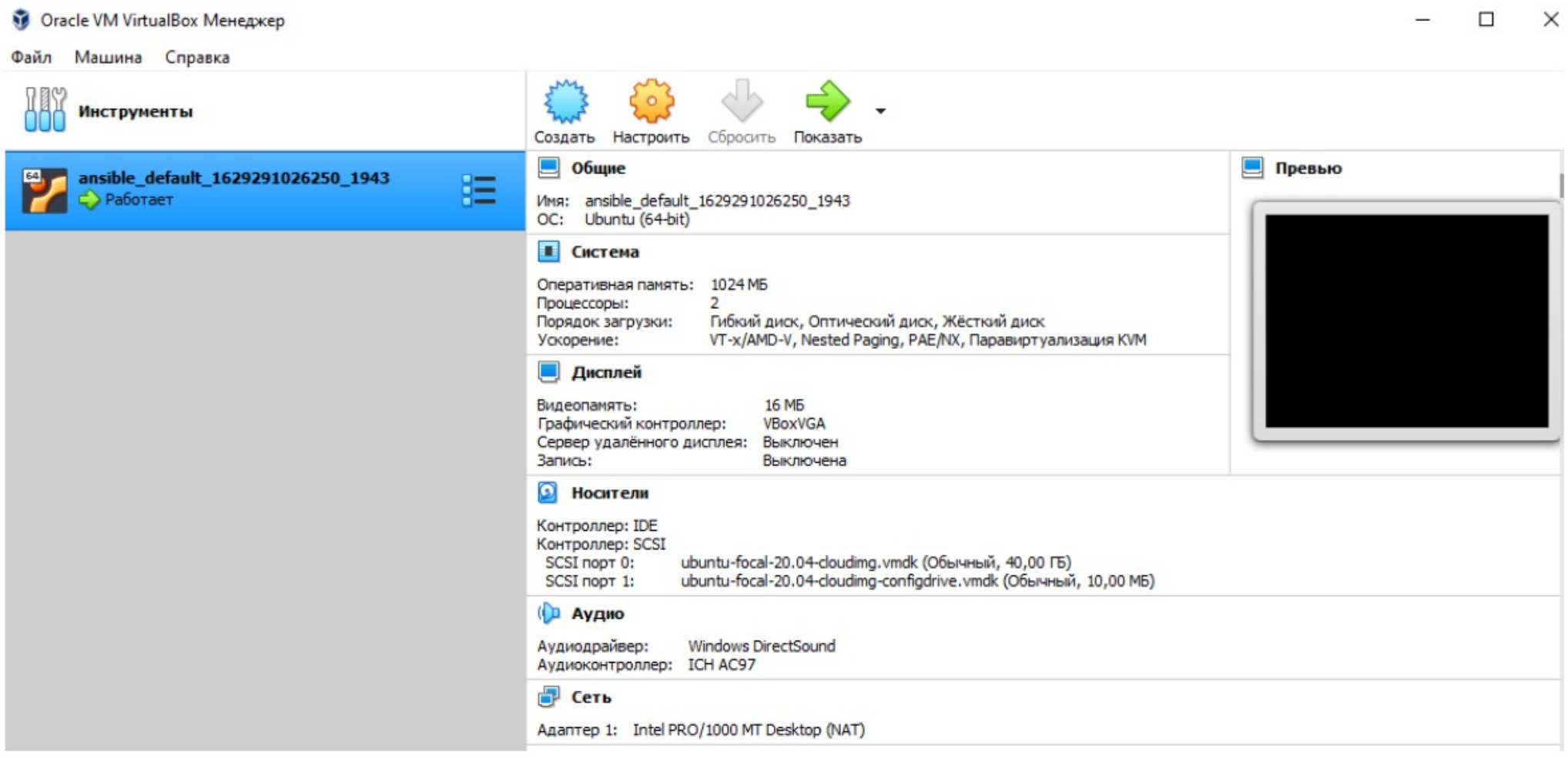
```
vagrant plugin install vagrant-vbguest

Installing the 'vagrant-vbguest' plugin. This can take a few minutes...
Installed the plugin 'vagrant-vbguest (0.30.0)'!
```

Запускаем установку машины.

```
vagrant up

Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'ubuntu/focal64'...

[K==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'ubuntu/focal64' version '20210816.0.0' is up to date...
```

```
==> default: Setting the name of the VM: ansible_default_1629288803611_15214
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
==> default: Forwarding ports...

....
```

После того как перестанут бежать строчки можно запустить VirtualBox и посмотреть на созданную машину.



Можем проверить статус.

```
vagrant status

Current machine states:
default                   running (virtualbox)
```

Пробуем подключится к машине по ssh.

```
vagrant ssh default

Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-81-generic x86_64)
 ...
vagrant@ubuntu-focal:~$
```

После того как вы с ней поиграли, выходим `exit` и уничтожаем её `vagrant destroy`.

2.3.5

Основные команды Vagrant:

`vagrant init` — создать пустой Vagrantfile, который содержит комментарии и общую структуру виртуальной машины.

`vagrant up` — поднять виртуальные машины из Vagrantfile. Запускается в той же директории, где и физически лежит Vagrantfile.

`vagrant suspend` — приостанавливает машины. `vagrant up` запустит машину без ее пересоздания.

`vagrant reload` — останавливает и потом запускает её. Обычно требуется если были внесены правки в конфигурационный файл Vargant.

`vagrant destroy` — уничтожает машины и стирает их жесткие диски. `vagrant up` создаст виртуальные машины заново.

`vagrant ssh` — заходит на машину по ssh.

Адрес vagrant repo для просмотра виртуальных машин
- https://app.vagrantup.com/boxes/search

2.4.2

Вторая буква: E

Сейчас вы залогинены на свежеустановленной системе ubuntu.

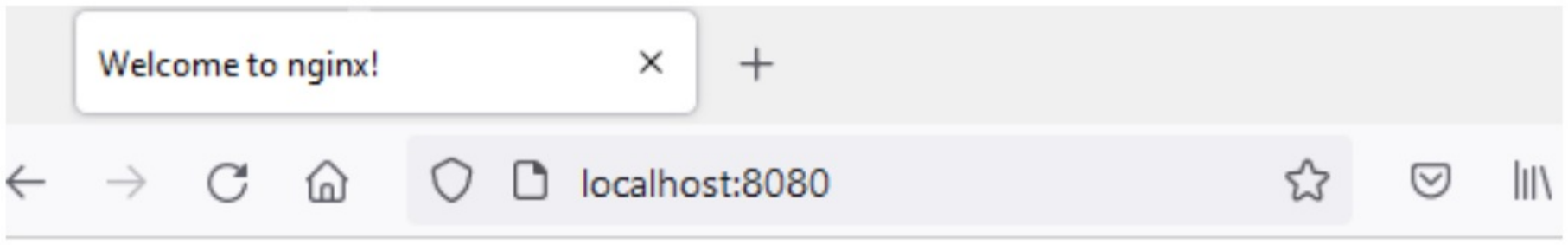
Обновляем архивы.

```
sudo apt -y update
```

Установим **nginx**. Также он автоматически настраивается и запускается.

```
sudo apt install -y nginx
```

Проверим его работу. Заходим на адрес localhost:8080



Меняем права папки **/var/www/html** на того пользователя с которого оперируем по ssh.

```
cd /var/www

sudo chown vagrant:vagrant html
```

проверяем

```
ls -l

drwxr-xr-x 2 vagrant vagrant 4096 Aug 18 13:32 html
```

Выходим из терминала виртуальной машины

```
exit
```

Скачиваем шаблон сайта какой-нибудь и бросаем файлы, к примеру, в папку **dist**, туда где лежит конфигурационный файл Vagrant. Или скачайте наш одностраничный сайт.

Скачиваем наш общедоступный репозиторий с материалами данного курса - https://gitlab.slurm.io/edu/ansible_course

```
git clone git@gitlab.slurm.io:edu/ansible_course.git
```

Копируем содержание данной папки - `ansible_base\config_files\1.lemp\one-page_site`. Это наш одностраничный сайт для тестирования.

Загружаем шаблон сайта

```
vagrant upload .\dist\ /var/www/html
Uploading .\dist\ to /var/www/html
Upload has completed successfully!

  Source: .\dist\
  Destination: /var/www/html
```

Заходим снова

```
vagrant ssh default

Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-81-generic x86_64)
```

Проверим, что всё корректно загрузилось.

```
cd /var/www/html/
ls -l

total 28
drwxrwxr-x 2 vagrant vagrant 4096 Aug 18 13:55 css
drwxrwxr-x 2 vagrant vagrant 4096 Aug 18 13:55 fonts
drwxrwxr-x 2 vagrant vagrant 4096 Aug 18 13:55 images
-rw-rw-r-- 1 vagrant vagrant 4955 Aug 18 13:55 index.html
-rw-r--r-- 1 root    root     612 Aug 18 13:32 index.nginx-debian.html
drwxrwxr-x 2 vagrant vagrant 4096 Aug 18 13:55 js
```

У вас скорее всего будет другой набор файлов.

Снова заходим сюда - localhost:8080 и любуемся своими трудами.

2.4.3

Кратко по пунктам из конфигурационного файла nginx.

```
server {
        listen       80; //порт который nginx будет слушать
           server_name  _; //имя сайта, на который nginx будет реагировать(например
www.mysite.ru). server_name _ - nginx будет использовать это для ответа на любые
запросы сайтов, даже несуществующие.
        root         /var/www/html; //папка в которой должны лежать файлы веб сайта,
которые nginx вернет
        location / { //показывает какие действия nginx должен совершать после
определенного адреса, данная команда будет применена к любым запросам после /, то
есть к любым запросам к сайту
        try_files $uri $uri/ =404; //просто возвращает документы по адресу урла,
например /test/test.html вернет содержимое test.html из папки test, если ничего нет,
то вернет 404
        }
}
```

Проверить запуск сервиса:

```
systemctl --type=service | grep nginx
```

Эта команда восстанавливает описание и состояние nginx сервиса.

Директиву *try_files* удобно использовать в случае, если необходимо проверить несколько папок перед отдачей файла - nginx.org

2.5.2

Третья буква: M

Устанавливаем **mysql-server**. Также автоматически сконфигурируется и запустится.


```
sudo apt install -y mysql-server
```

Получаем вывод от сервиса системного контроля с сервисами, которые сейчас работают и проверим наши уже запущенные сервисы **nginx** и **mysql-server**.

```
sudo systemctl --type=service
...
mysql.service                          loaded active running MySQL Community Server
nginx.service                          loaded active running A high performance web
server and a reverse proxy server
```

Повысим безопасность установки **mysql-server**

```
sudo mysql_secure_installation

ALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?
```

Первый вопрос. Хотим ли валидировать на сложность пароли? В продакшене надо, но в демонстрационных целях можно пропустить **No**.

Задаем пароль. Но это бессмысленно сейчас, root сейчас доступен изнутри этой машины.

```
New password:

Re-enter new password:
```

Есть пользователи с пустыми именами и это не безопасно. Просит подтвердить их удаление и мы соглашаемся **Yes.**

```
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.
```

Хотим ли мы из далеко подключатся по root доступу? Мы отказываемся **No**, приложение, которое будет подключаться к базе данных, находится локально.

```
Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.
```

Удаляем тестовые базы данных? Подтверждаем **Yes**.

```
By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.
```

Подтверждаем **Yes** перезагрузку таблиц, чтоб применились новые изменения.

```
Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.
```

Проверяем возможность подключится к БД.

```
mysql

ERROR 1045 (28000): Access denied for user 'vagrant'@'localhost' (using password: NO)

sudo mysql

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.26-0ubuntu0.20.04.2 (Ubuntu)

...

mysql>
```

Есть рутовая возможность подключения к базе. Сейчас мы это изменим. Смотрим, что у root подключение через сокет.

```
mysql> SELECT user, plugin FROM mysql.user;
+------------------+-----------------------+
| user             | plugin                |
+------------------+-----------------------+
| debian-sys-maint | caching_sha2_password |
| mysql.infoschema | caching_sha2_password |
| mysql.session    | caching_sha2_password |
| mysql.sys        | caching_sha2_password |
| root             | auth_socket           |
+------------------+-----------------------+
```

Сменим на обычную парольную авторизацию, т.к. сокет на стандартное подключение для php/python.

```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH 'mysql_native_password' BY
'password';
Query OK, 0 rows affected (0.02 sec)
```

Проверяем.

```
mysql> SELECT user, plugin FROM mysql.user;
...
root             | mysql_native_password
```

Выходим из подключения к mysql-server

```
mysql> exit
Bye
```

Проверяем снова авторизацию

```
mysql

ERROR 1045 (28000): Access denied for user 'vagrant'@'localhost' (using password: NO)

sudo mysql

ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)
```

И теперь нас не пускает через root, как ранее.

Теперь нужно ввести такую команду. **-u** - ключ с какого юзера авторизуемся по паролю и указываем его. **-p** - запросить у нас ввод пароля.

```
mysql -u root -p

Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
...
mysql>
```

У вас получилось и теперь осталась последняя буква. Давайте перейдём!

2.5.3

MySQL - система управления базами данных, необходимая разработчикам для хранения долгосрочной информации. Чтобы управлять базами данных система поддерживает язык SQL. Её основной модуль постоянно запущен на фоне и является сервисным процессом.

Полезные запросы MySQL:

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'password';
```

Изменит пароль пользователю root на password, позволит авторизоваться без помощи ключа.

```
sudo mysql_secure_installation
```

Защитит установку MySQL, путем удаления тестовых БД пустых пользователей.

2.6.2

Четвертая буква: P

Установим два модуля. **php-fpm** связывает язык php и nginx сервисом. **php-mysql** нужен для связи с базой данных.

```
sudo apt install -y php-fpm php-mysql
```

Проверим, что php установлен. Вводим команду.

```
php --version

PHP 7.4.3 (cli) (built: Jul  5 2021 15:13:35) ( NTS )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies
with Zend OPcache v7.4.3, Copyright (c), by Zend Technologies
```

Убедимся, что php-fpm создал файловый сокет - **php7.4-fpm.sock**

```
ls -la /var/run/php/

drwxr-xr-x  2 www-data www-data 100 Aug 18 15:06 .
drwxr-xr-x 30 root     root     960 Aug 18 15:06 ..
lrwxrwxrwx  1 root     root      30 Aug 18 15:06 php-fpm.sock ->
```

```
/etc/alternatives/php-fpm.sock
-rw-r--r--  1 root     root       5 Aug 18 15:06 php7.4-fpm.pid
srw-rw----  1 www-data www-data   0 Aug 18 15:06 php7.4-fpm.sock
```

Создадим отдельную конфигурацию для проверки php. В папке **sites-available**, т.к. в папке **sites-enabled** находятся линки в папку **sites-available**.

```
cd /etc/nginx/sites-available

sudo vim php_test.conf

server {
        listen 80;
        root /var/www/php_test;
        index index.php index.html index.htm index.nginx-debian.html; # определяем
индексный файл по умолчанию, в нашем случае мы хотим index.php
        server_name php-test.com;

        location / {
                try_files $uri $uri/ =404;
        }

        location ~ \.php$ {  # для любого окончания .php по запросу мы применяем
правила этого блока
                include snippets/fastcgi-php.conf; # включаем конфигурацию для php
                fastcgi_pass unix:/var/run/php/php7.4-fpm.sock; # обрабатываем с
помощью fpm, передавая данные в файловый сокет и получая их обратно
        }
}
```

Сохраняем файл и выходим. Делаем линку в папку **sites-enabled**.

```
sudo ln -s /etc/nginx/sites-available/php_test.conf /etc/nginx/sites-enabled/php_test
```

Проверяем линки.

```
ls -la /etc/nginx/sites-enabled/

lrwxrwxrwx 1 root root   34 Aug 18 13:32 default -> /etc/nginx/sites-
available/default
lrwxrwxrwx 1 root root   40 Aug 18 15:28 php_test -> /etc/nginx/sites-
available/php_test.conf
```

Проверяем через nginx конфигурацию.

```
sudo nginx -t

nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Перечитаем конфигурационные файлы для nginx.

```
sudo nginx -s reload
```

Следующим шагом создадим файлы php для тестового сайта.

```
cd /var/www

sudo mkdir php_test

sudo chown vagrant:vagrant php_test

cd php_test
```

```
vim index.php
```

Вставляем туда этот код. Код вернет скомпилированную информацию о среде PHP на вашем сервере.

```
<?php phpinfo(); ?>
```

Теперь нужно внести информацию в свой **hosts** файл на ПК. Иначе, если вы введете наименование сайта у себя в браузере php-test.com:8080, то он полезет наружу.

В Windows по пути **C:\Windows\System32\drivers\etc\hosts**

Вносим **127.0.0.1 php-test.com**

Сохраняем и смотрим php-test.com:8080 Ура!!!

Проверим подключение к БД. Копируем файл **test-connection.php** и кладем рядом с конфигурационным файлом Vagrant.

Он уже готов и находится в нашем скаченном репо, по пути `\ansible_course\1.lemp\test-connection.php`

Загружаем этот файл в виртуальную машину.

```
vagrant upload .\test-connection.php /var/www/php_test/

Uploading .\test-connection.php to /var/www/php_test/
Upload has completed successfully!

Source: .\test-connection.php
Destination: /var/www/php_test/
```

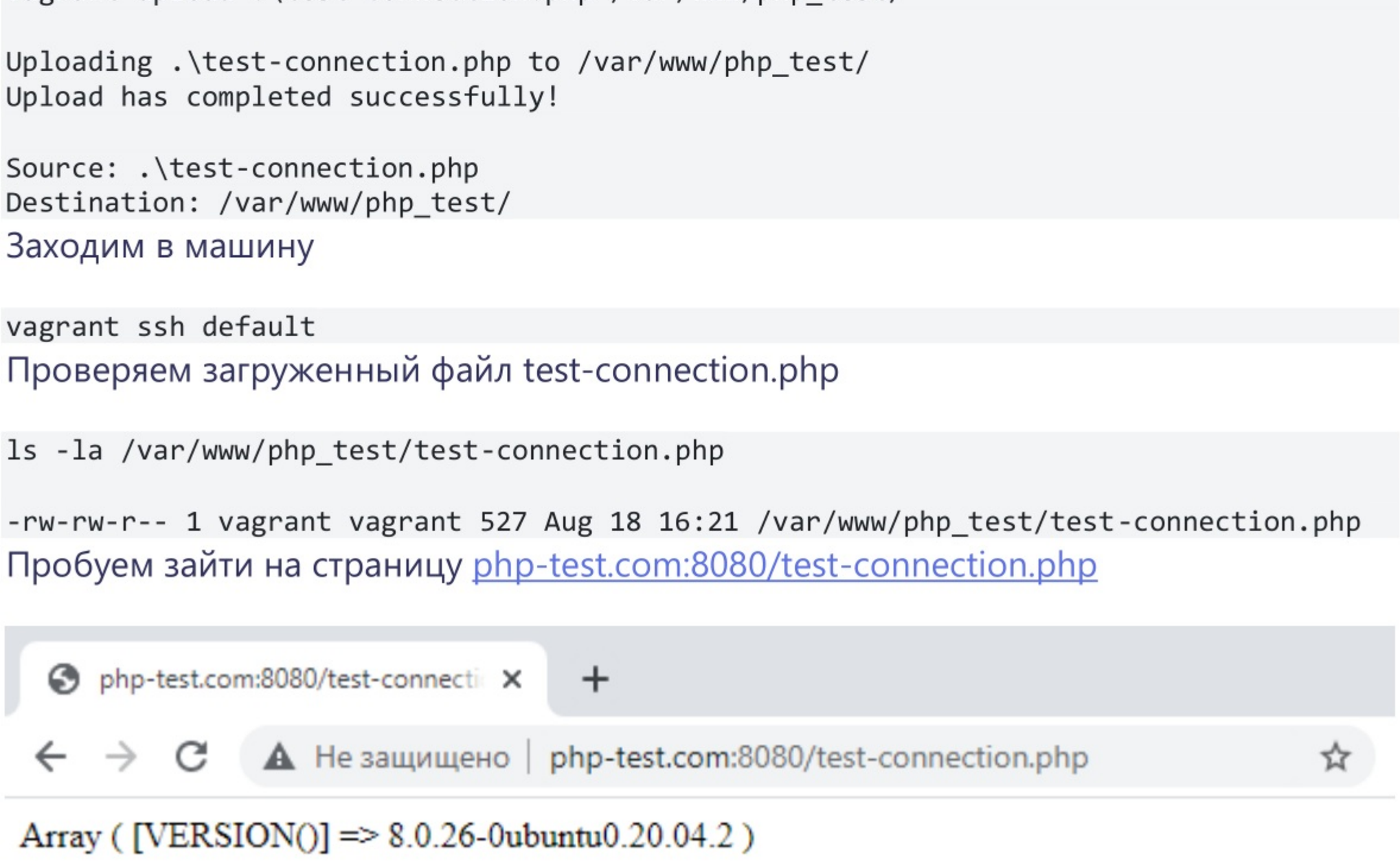
Заходим в машину

```
vagrant ssh default
```

Проверяем загруженный файл test-connection.php

```
ls -la /var/www/php_test/test-connection.php

-rw-rw-r-- 1 vagrant vagrant 527 Aug 18 16:21 /var/www/php_test/test-connection.php
```

Пробуем зайти на страницу php-test.com:8080/test-connection.php



Поздравляем вас!!!

Собрали ~~слово~~ аббревиатуру **LEMP**

2.9

Машина на CentOS 7

1. Создайте виртуальную машину в Vagrant с образом CentOS 7, используя образ https://app.vagrantup.com/bento/boxes/centos-7.5.

2. Внимание на вывод Vagrant. Какие отличия от Ubuntu вы видите?

3. После создания остановите и удалите виртуальную машину.

Редирект

Используя атрибут location сделайте редирект на **google.com** со страницы **/wantgoogle/**. localhost:8080**/wantgoogle/**
Что изменится в файле конфигурации относительно базового файла?

Подсказка: Для редиректа можете использовать `proxy_pass` который работает, в том числе и на веб сайты (P.S. не забудьте про `proxy_set_header`). В простом случае можно использовать `return 301`

Важно: Не забывайте про команды `nginx -s reload` и `nginx -t`. Первая перезагрузит конфигурации в nginx, вторая скажет вам верные ли они.

Вывод данных из БД

Задание повышенной сложности.

Используя команды CREATE DATABASE, CREATE TABLE создайте базу данных business_db и таблицу important_data в MySQL со следующей структурой.

```
CREATE TABLE important_data (
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) );
```

Используя PHP соединитесь с базой данных, с помощью запроса.

```
SELECT * FROM important_data;
```

и скрипта test-connection.php выведите данные. php-test.com:8080/test-connection.php

Как изменится скрипт **test-connection.php**?