

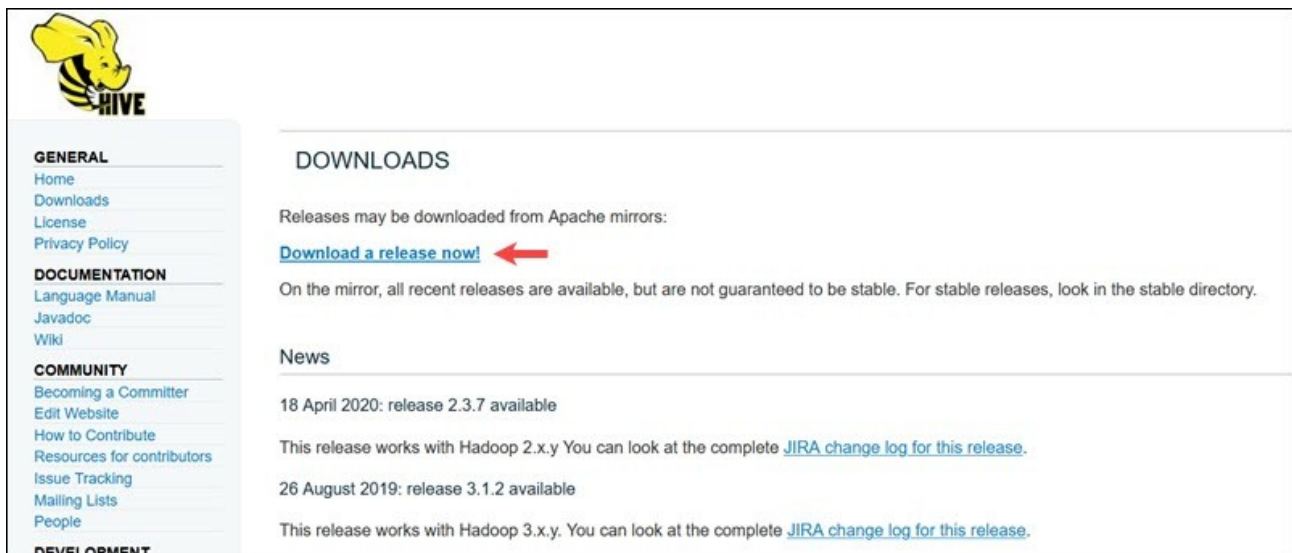
Install Apache Hive on Ubuntu

To configure Apache Hive, first you need to download and unzip Hive. Then you need to customize the following files and settings:

- Edit **.bashrc** file
- Edit **hive-config.sh** file
- Create **Hive directories** in HDFS
- Configure **hive-site.xml** file
- Initiate **Derby database**

Step 1: Download and Untar Hive

Visit the [Apache Hive official download page](#) and determine which Hive version is best suited for your Hadoop edition. Once you establish which version you need, select the **Download a Release Now!** option.



The screenshot shows the Apache Hive website's Downloads page. On the left is a navigation menu with sections: GENERAL (Home, Downloads, License, Privacy Policy), DOCUMENTATION (Language Manual, Javadoc, Wiki), COMMUNITY (Becoming a Committer, Edit Website, How to Contribute, Resources for contributors, Issue Tracking, Mailing Lists, People), and DEVELOPMENT. The main content area is titled 'DOWNLOADS' and contains the following text: 'Releases may be downloaded from Apache mirrors: [Download a release now!](#)' with a red arrow pointing to the link. Below this is a note: 'On the mirror, all recent releases are available, but are not guaranteed to be stable. For stable releases, look in the stable directory.' Under the 'News' section, there are two entries: '18 April 2020: release 2.3.7 available' with a link to the JIRA change log, and '26 August 2019: release 3.1.2 available' with a link to the JIRA change log.

The mirror link on the subsequent page leads to the directories containing available Hive tar packages. This page also provides useful instructions on how to validate the integrity of files retrieved from mirror sites.

News About Make a Donation The Apache Way Join Us Downloads

THE APACHE SOFTWARE FOUNDATION 20TH ANNIVERSARY

COMMUNITY-LED DEVELOPMENT "THE APACHE WAY"

Projects People Community License Sponsors

We suggest the following mirror site for your download:
<https://downloads.apache.org/hive/>

Other mirror sites are suggested below.

It is essential that you verify the integrity of the downloaded file using the PGP signature (.asc file) or a hash (.md5 or .sha* file).
 Please only use the backup mirrors to download KEYS, PGP signatures and hashes (SHA* etc) -- or if no other mirrors are working.

HTTP
<https://downloads.apache.org/hive/>

BACKUP SITES
 Please only use the backup mirrors to download KEYS, PGP signatures and hashes (SHA* etc) -- or if no other mirrors are working.
<https://downloads.apache.org/hive/>

The full listing of mirror sites is also available.

The Ubuntu system presented in this guide already has **Hadoop 3.2.1** installed. This Hadoop version is compatible with the **Hive 3.1.2** release.

Index of /hive

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 hive-1.2.2/	2018-05-04 20:51	-	
 hive-2.3.7/	2020-04-17 22:03	-	
 hive-3.1.2/ 	2019-08-26 20:21	-	
 hive-standalone-metastore-3.0.0/	2018-06-07 18:12	-	
 hive-storage-2.6.1/	2018-05-11 22:26	-	
 hive-storage-2.7.2/	2020-05-11 16:00	-	
 stable-2/	2020-04-17 22:03	-	
 KEYS	2020-01-17 22:47	82K	

Select the **apache-hive-3.1.2-bin.tar.gz** file to begin the download process.

Index of /hive/hive-3.1.2

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 apache-hive-3.1.2-bin.tar.gz	2019-08-26 20:20	266M	
 apache-hive-3.1.2-bin.tar.gz.asc	2019-08-26 20:20	833	
 apache-hive-3.1.2-bin.tar.gz.sha256	2019-08-26 20:20	95	
 apache-hive-3.1.2-src.tar.gz	2019-08-26 20:20	24M	
 apache-hive-3.1.2-src.tar.gz.asc	2019-08-26 20:20	833	
 apache-hive-3.1.2-src.tar.gz.sha256	2019-08-26 20:20	95	

Alternatively, access your Ubuntu command line and download the compressed Hive files using and the **wget** command followed by the download path:

```
wget https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
```

```
hadoop@phoenixnap:~$ wget https://downloads.apache.org/hive/hive-3.1.2/apache-hi
ve-3.1.2-bin.tar.gz
--2020-06-01 08:11:30-- https://downloads.apache.org/hive/hive-3.1.2/apache-hi
ve-3.1.2-bin.tar.gz
Resolving downloads.apache.org (downloads.apache.org)... 88.99.95.219, 2a01:4f8
:10a:201a::2
Connecting to downloads.apache.org (downloads.apache.org)|88.99.95.219|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 278813748 (266M) [application/x-gzip]
Saving to: 'apache-hive-3.1.2-bin.tar.gz'

apache-hive-3.1.2-b 100%[=====>] 265.90M  10.9MB/s   in 25s

2020-06-01 08:11:55 (10.7 MB/s) - 'apache-hive-3.1.2-bin.tar.gz' saved [2788137
48/278813748]
```

Once the download process is complete, untar the compressed Hive package:

```
tar xzf apache-hive-3.1.2-bin.tar.gz
```

The Hive binary files are now located in the *apache-hive-3.1.2-bin* directory.

```
hadoop@phoenixnap:~$ tar xzf apache-hive-3.1.2-bin.tar.gz
hadoop@phoenixnap:~$ ls
apache-hive-3.1.2-bin  dfsdata  hadoop-3.2.1.tar.gz  tmpdata
apache-hive-3.1.2-bin.tar.gz  hadoop-3.2.1  hadoop-3.2.1.tar.gz.1
```

Step 2: Configure Hive Environment Variables (bashrc)

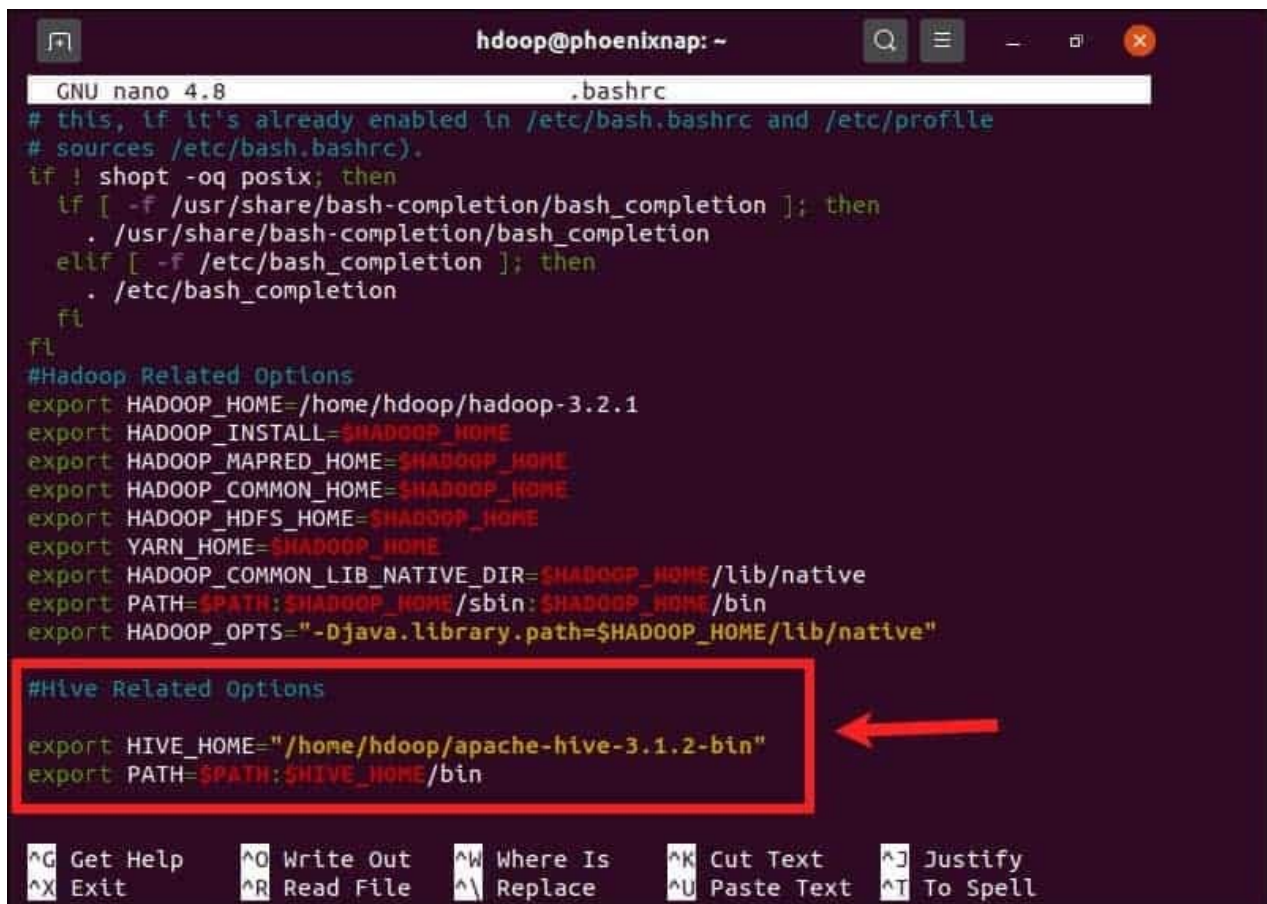
The `$HIVE_HOME` environment variable needs to direct the client shell to the `apache-hive-3.1.2-bin` directory. Edit the `.bashrc` shell configuration file using a text editor of your choice (we will be using nano):

```
sudo nano .bashrc
```

Append the following Hive environment variables to the `.bashrc` file:

```
export HIVE_HOME= "home/hadoop/apache-hive-3.1.2-bin"  
export PATH=$PATH:$HIVE_HOME/bin
```

The Hadoop environment variables are located within the same file.



```
hadoop@phoenixnap: ~  
GNU nano 4.8 .bashrc  
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile  
# sources /etc/bash.bashrc).  
if ! shopt -oq posix; then  
  if [ -f /usr/share/bash-completion/bash_completion ]; then  
    . /usr/share/bash-completion/bash_completion  
  elif [ -f /etc/bash_completion ]; then  
    . /etc/bash_completion  
  fi  
fi  
#Hadoop Related Options  
export HADOOP_HOME=/home/hadoop/hadoop-3.2.1  
export HADOOP_INSTALL=$HADOOP_HOME  
export HADOOP_MAPRED_HOME=$HADOOP_HOME  
export HADOOP_COMMON_HOME=$HADOOP_HOME  
export HADOOP_HDFS_HOME=$HADOOP_HOME  
export YARN_HOME=$HADOOP_HOME  
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native  
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin  
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"  
  
#Hive Related Options  
export HIVE_HOME="/home/hadoop/apache-hive-3.1.2-bin"  
export PATH=$PATH:$HIVE_HOME/bin
```

Save and exit the `.bashrc` file once you add the Hive variables. Apply the changes to the current environment with the following command:

```
source ~/.bashrc
```

Step 3: Edit hive-config.sh file

Apache Hive needs to be able to interact with the Hadoop Distributed File System. Access the *hive-config.sh* file using the previously created **\$HIVE_HOME** variable:

```
sudo nano $HIVE_HOME/bin/hive-config.sh
```

Note: The *hive-config.sh* file is in the *bin* directory within your Hive installation directory.

Add the **HADOOP_HOME** variable and the full path to your Hadoop directory:

```
export HADOOP_HOME=/home/hadoop/hadoop-3.2.1
```

```
# Allow alternate conf dir location.
HIVE_CONF_DIR="${HIVE_CONF_DIR:-$HIVE_HOME/conf}"

export HIVE_CONF_DIR="/home/hadoop/apache-hive-3.1.2-bin/conf"
export HADOOP_HOME=/home/hadoop/hadoop-3.2.1
```

Save the edits and exit the *hive-config.sh* file.

Step 4: Create Hive Directories in HDFS

Create two separate directories to store data in the HDFS layer:

- The temporary, *tmp* directory is going to store the intermediate results of Hive processes.
- The *warehouse* directory is going to store the [Hive related tables](#).

Create tmp Directory

Create a *tmp* directory within the HDFS storage layer. This directory is going to store the intermediary data Hive sends to the HDFS:

```
hdfs dfs -mkdir /tmp
```

Add write and execute permissions to tmp group members:

```
hdfs dfs -chmod g+w /tmp
```

Check if the permissions were added correctly:

```
hdfs dfs -ls /
```

The output confirms that users now have write and execute permissions.

```
hadoop@phoenixnap:~$ hdfs dfs -ls /
Found 4 items
drwxr-xr-x  - hadoop supergroup    0 2020-06-02 02:37 /ExampleDir
drwxrwxr-x  - hadoop supergroup    0 2020-06-02 07:26 /tmp
```

Create warehouse Directory

Create the *warehouse* directory within the */user/hive/* parent directory:

```
hdfs dfs -mkdir -p /user/hive/warehouse
```


Add **write** and **execute** permissions to *warehouse* group members:

```
hdfs dfs -chmod g+w /user/hive/warehouse
```

Check if the permissions were added correctly:

```
hdfs dfs -ls /user/hive
```

The output confirms that users now have write and execute permissions.

```
hdoop@phoenixnap:~$ hdfs dfs -ls /user/hive
Found 1 items
drwxrwxr-x - hdoop supergroup          0 2020-06-02 09:06 /user/hive/warehouse
```

Step 5: Configure *hive-site.xml* File (Optional)

Apache Hive distributions contain template configuration files by default. The template files are located within the Hive *conf* directory and outline default Hive settings.

Use the following command to locate the correct file:

```
cd $HIVE_HOME/conf
```

List the files contained in the folder using the **ls** command.

```
hdoop@phoenixnap:~$ cd $HIVE_HOME/conf
hdoop@phoenixnap:~/apache-hive-3.1.2-bin/conf$ ls
beeline-log4j2.properties.template      ivysettings.xml
hive-default.xml.template                llap-cli-log4j2.properties.template
hive-env.sh.template                    llap-daemon-log4j2.properties.template
hive-exec-log4j2.properties.template    parquet-logging.properties
hive-log4j2.properties.template
```

Use the *hive-default.xml.template* to create the *hive-site.xml* file:

```
cp hive-default.xml.template hive-site.xml
```

Access the *hive-site.xml* file using the nano text editor:

```
sudo nano hive-site.xml
```

Note: The *hive-site.xml* file controls every aspect of Hive operations. The number of available advanced settings can be overwhelming and highly specific. Consult the [official Hive Configuration Documentation](#) regularly when customizing Hive and Hive Metastore settings.

Using Hive in a stand-alone mode rather than in a real-life Apache Hadoop cluster is a safe option for newcomers. You can configure the system to use your local storage rather than the HDFS layer by setting the *hive.metastore.warehouse.dir* parameter value to the location of your Hive *warehouse* directory.

```
GNU nano 4.8          hive-site.xml          Modified
<property>
  <name>hive.metastore.db.type</name>
  <value>DERBY</value>
  <description>
    Expects one of [derby, oracle, mysql, mssql, postgres].
    Type of database used by the metastore. Information schema & JDBCSto
  </description>
</property>
<property>
  <name>hive.metastore.warehouse.dir</name>
  <value>/user/hive/warehouse</value>
  <description>location of default database for the warehouse</description>
</property>
<property>
  <name>hive.metastore.warehouse.external.dir</name>
  <value/>
  <description>Default location for external tables created in the warehouse
</property>
<property>
  <name>hive.metastore.uris</name>
  <value/>
  <description>Thrift URI for the remote metastore. Used by metastore client
</property>
<property>
  <name>hive.metastore.uri.selection</name>
```

Step 6: Initiate Derby Database

Apache Hive uses the Derby database to store metadata. Initiate the Derby database, from the Hive *bin* directory using the **schematool** command:

```
$HIVE_HOME/bin/schematool -dbType derby -initSchema
```

The process can take a few moments to complete.

```
Initialization script completed
schemaTool completed
hdoop@phoenixnap:~/apache-hive-3.1.2-bin/bin$
```



Derby is the default metadata store for Hive. If you plan to use a different database solution, such as [MySQL](#) or [PostgreSQL](#), you can specify a database type in the *hive-site.xml* file.

How to Fix guava Incompatibility Error in Hive

If the Derby database does not successfully initiate, you might receive an error with the following content:

```
“Exception in thread “main” java.lang.NoSuchMethodError:  
com.google.common.base.Preconditions.checkArgument(ZLjava/lang/String;Ljava/lang/Object;)V”
```

This error indicates that there is most likely an incompatibility issue between Hadoop and Hive *guava* versions.

Locate the **guava jar** file in the Hive *lib* directory:

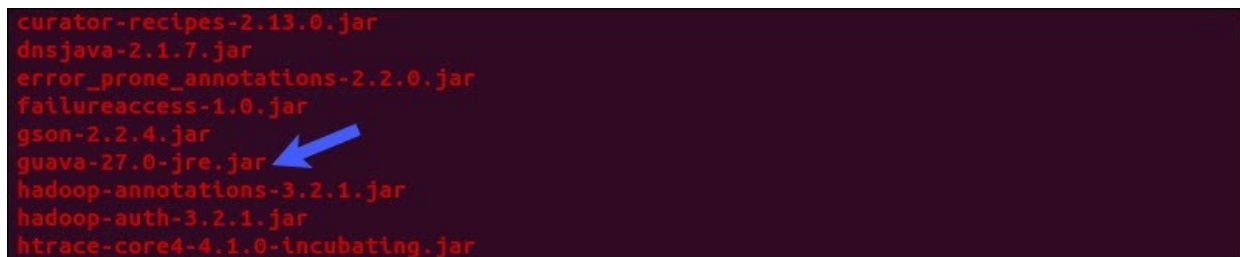
```
ls $HIVE_HOME/lib
```



```
esri-geometry-api-2.0.0.jar  
findbugs-annotations-1.3.9-1.jar  
flatbuffers-1.2.0-3f79e055.jar  
groovy-all-2.4.11.jar  
gson-2.2.4.jar  
guava-19.0.jar  
hbase-client-2.0.0-alpha4.jar  
hbase-common-2.0.0-alpha4.jar  
hbase-common-2.0.0-alpha4-tests.jar  
hbase-hadoop2-compat-2.0.0-alpha4.jar
```

Locate the **guava jar** file in the Hadoop *lib* directory as well:

```
ls $HADOOP_HOME/share/hadoop/hdfs/lib
```



```
curator-recipes-2.13.0.jar  
dnsjava-2.1.7.jar  
error_prone_annotations-2.2.0.jar  
failureaccess-1.0.jar  
gson-2.2.4.jar  
guava-27.0-jre.jar  
hadoop-annotations-3.2.1.jar  
hadoop-auth-3.2.1.jar  
htrace-core4-4.1.0-incubating.jar
```

The two listed versions are not compatible and are causing the error. Remove the existing **guava** file from the Hive *lib* directory:

```
rm $HIVE_HOME/lib/guava-19.0.jar
```

Copy the **guava** file from the Hadoop *lib* directory to the Hive *lib* directory:

```
cp $HADOOP_HOME/share/hadoop/hdfs/lib/guava-27.0-jre.jar $HIVE_HOME/lib/
```

Use the **schematool** command once again to initiate the Derby database:

```
$HIVE_HOME/bin/schematool -dbType derby -initSchema
```

Launch Hive Client Shell on Ubuntu

Start the Hive command-line interface using the following commands:


```
cd $HIVE_HOME/bin
```

```
hive
```

You are now able to issue SQL-like commands and directly interact with HDFS.

```
hdoop@phoenixnap:~$ cd $HIVE_HOME/bin
hdoop@phoenixnap:~/apache-hive-3.1.2-bin/bin$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hdoop/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/hdoop/hadoop-3.2.1/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation
.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 43f09b9d-bc36-4e29-a1d6-045a92e66b98

Logging initialized using configuration in jar:file:/home/hdoop/apache-hive-3.1.2-bin/lib/hive-common-3.1.2.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Hive Session ID = 39a404e5-f53f-47c4-9ff7-7581a413edd6
hive> 
```

Conclusion

You have successfully installed and configured Hive on your Ubuntu system. Use HiveQL to query and manage your Hadoop distributed storage and perform SQL-like tasks. Your Hadoop cluster now has an easy-to-use gateway to previously inaccessible RDBMS.