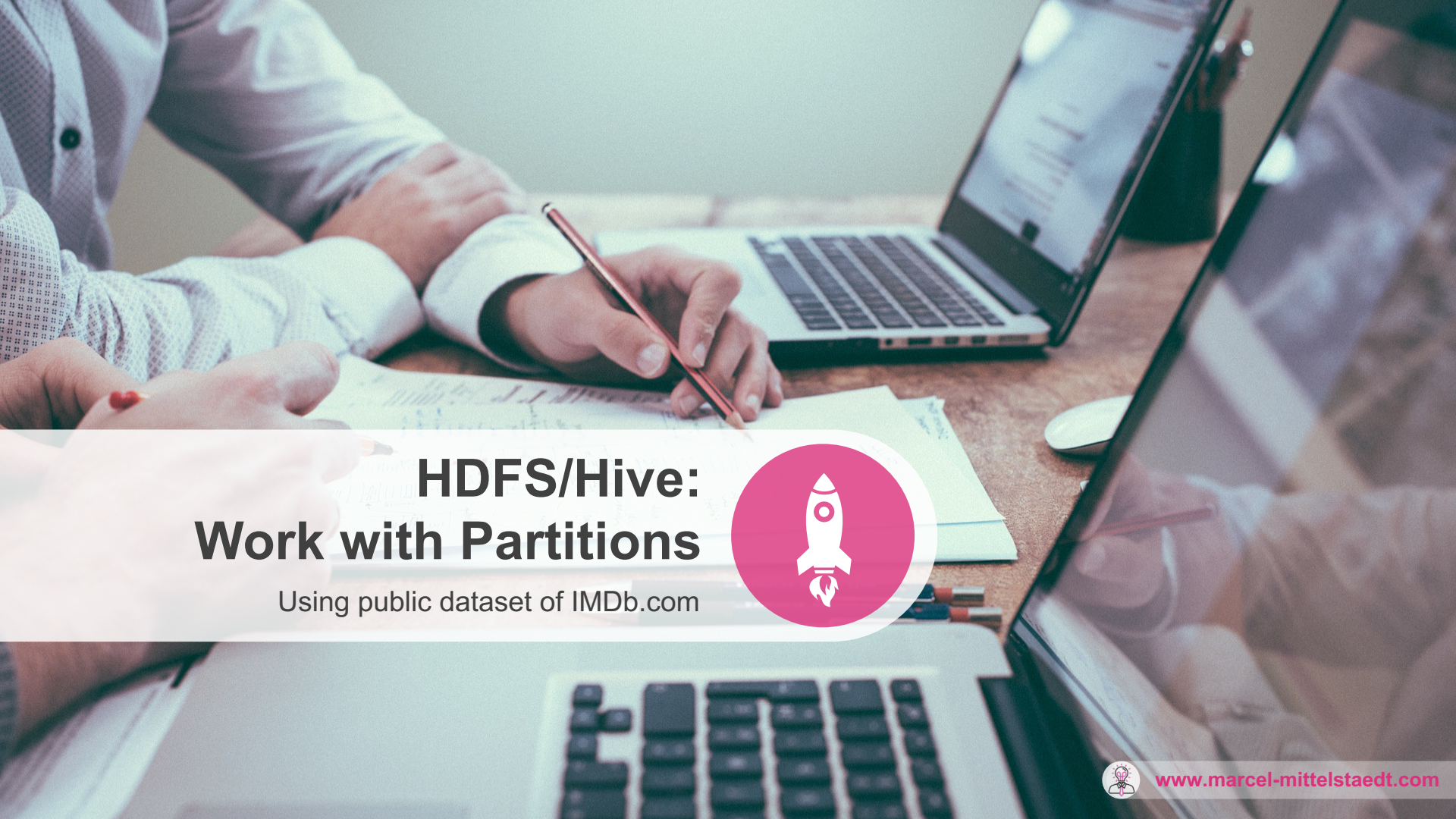
HandsOn – HDFS/Hive Partitioning and HiveServer2

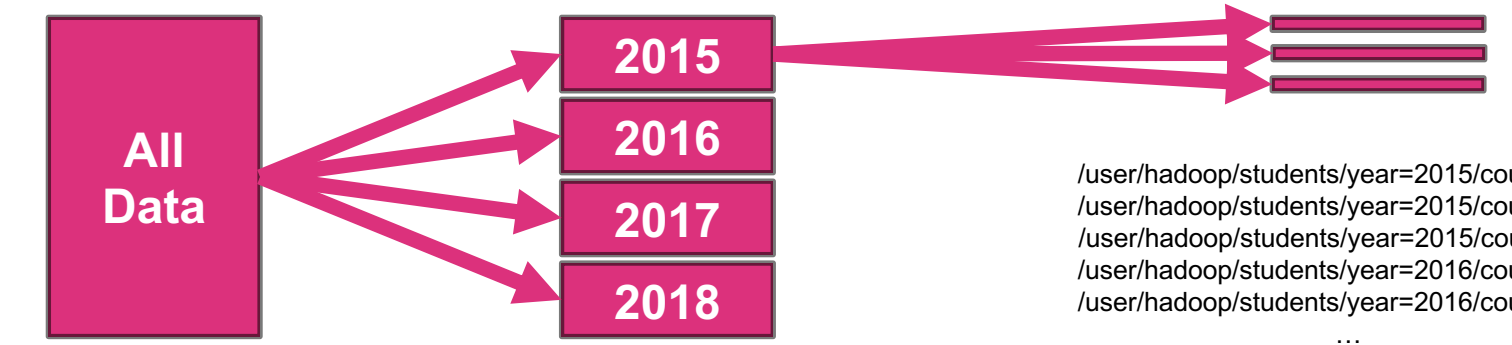


HDFS/Hive: Work with Partitions

Using public dataset of IMDb.com

HDFS/Hive - **Partitioning**

- Partitioning of data distributes load and speeds up data processing
- A table can have one or more partition columns, defined by the time of creating a table (`CREATE TABLE student(id Int, name STRING) PARTITIONED BY (year STRING)… STORED AS TEXTFILE LOCATION '/user/hadoop/students';`)
- partitioning can be done either **static** or **dynamic**
- each distinct value of a partition column is represented by a **HDFS directory**



Static Partitioning – **Create Partitioned Table**

1. Create partitioned version of table `imdb_ratings`: **`imdb_ratings_partitioned:`**

```
hive >    CREATE TABLE IF NOT EXISTS imdb_ratings_partitioned(
                tconst STRING,
                average_rating DECIMAL(2,1),
                num_votes BIGINT
          ) PARTITIONED BY (partition_quality STRING)
          STORED AS ORCFILE LOCATION '/user/hadoop/imdb/ratings_partitioned';
```

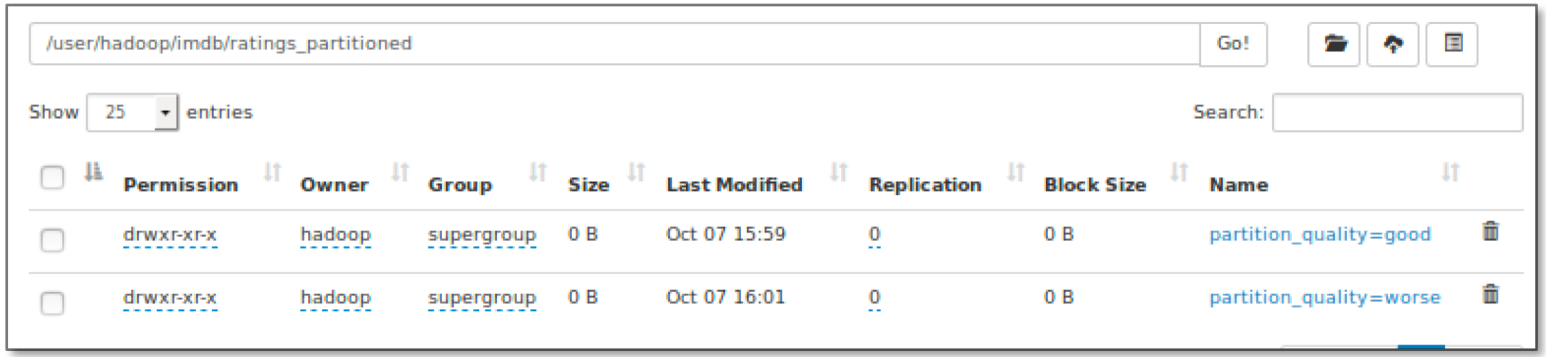

Static Partitioning – INSERT Into Table via Hive

1. Migrate and partition data of table `imdb_ratings` to table **`imdb_ratings_partitioned:`**

```
INSERT OVERWRITE TABLE imdb_ratings_partitioned partition(partition_quality='good')
SELECT r.tconst, r.average_rating, r.num_votes FROM imdb_ratings r WHERE r.average_rating >= 7;

INSERT OVERWRITE TABLE imdb_ratings_partitioned partition(partition_quality='worse')
SELECT r.tconst, r.average_rating, r.num_votes FROM imdb_ratings r WHERE r.average_rating < 7;
```

2. Check Success on HDFS:



Static Partitioning – **INSERT Into Table via Hive**

3. Check Success via Hive:

```
select distinct average_rating from imdb_ratings_partitioned where partition_quality = 'good';

7.0
7.1
7.2
7.3
7.4
7.5
7.6
7.7
[...]
9.0
9.1
9.2
9.3
9.4
9.5
9.6
9.7
9.8
9.9
```


Dynamic Partitioning – **Create Partitioned Table**

1. Create partitioned version of table `imdb_movies`: **`imdb_movies_partitioned:`**

```
hive >  CREATE TABLE IF NOT EXISTS imdb_movies_partitioned(
          tconst STRING,
          title_type STRING,
          primary_title STRING,
          original_title STRING,
          is_adult DECIMAL(1,0),
          start_year DECIMAL(4,0),
          end_year STRING,
          runtime_minutes INT,
          genres STRING
) PARTITIONED BY (partition_year int)
STORED AS ORCFILE
LOCATION '/user/hadoop/imdb/name_partitioned';
```

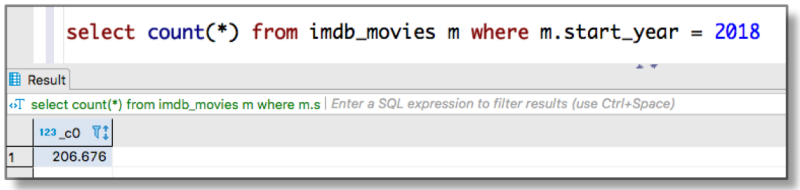

Dynamic Partitioning – INSERT Into Table via Hive

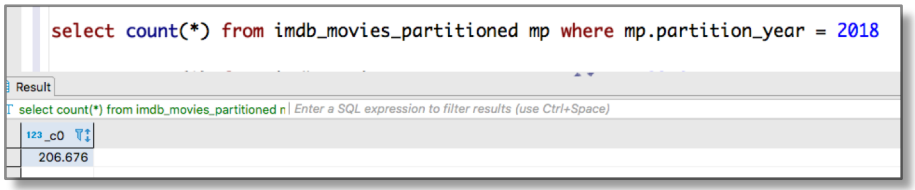
1. Migrate and partition data of table `imdb_movies` to table **`imdb_movies_partitioned:`**

```
set hive.exec.dynamic.partition.mode=nonstrict; -- enable dynamic partitioning

INSERT OVERWRITE TABLE imdb_movies_partitioned partition(partition_year)
SELECT m.tconst, m.title_type, m.primary_title, m.original_title, m.is_adult,
m.start_year, m.end_year, m.runtime_minutes, m.genres,
m.start_year -- last column = partition column
FROM imdb_movies m
```

2. Check Success via Hive:





Dynamic Partitioning – **INSERT Into Table via Hive**

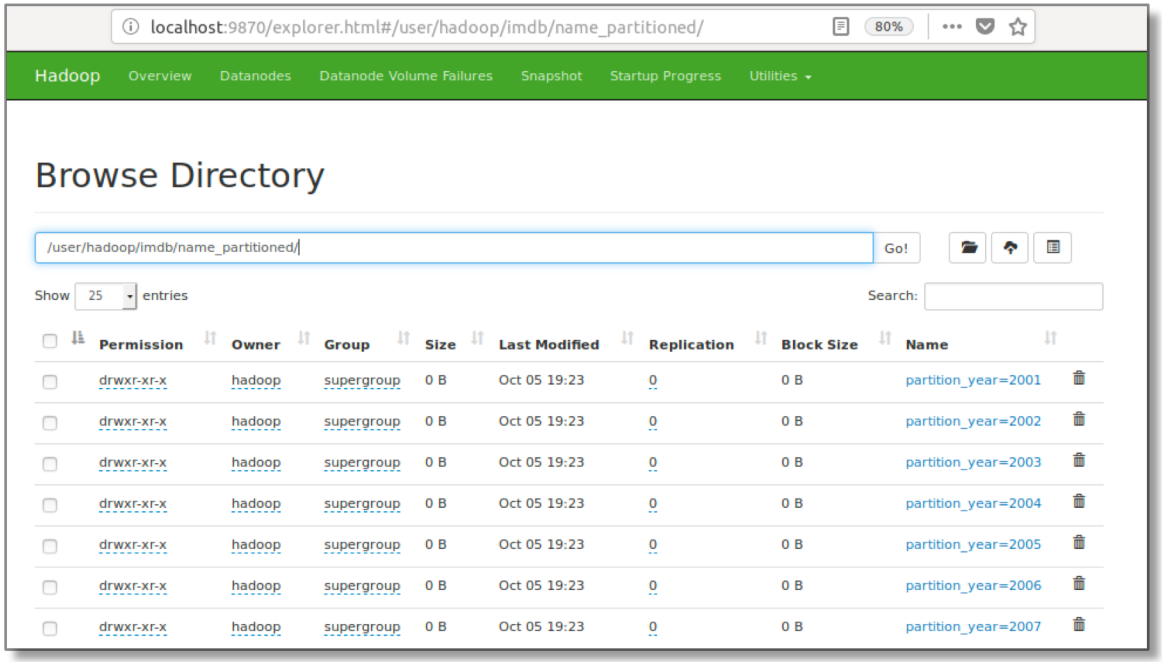
3. Check Success on HDFS:

```
hadoop fs -ls /user/hadoop/imdb/name_partitioned
[…]
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2011
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2012
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2013
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2014
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2015
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2016
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2017
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2018
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2019
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2020
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:24 /user/hadoop/imdb/name_partitioned/partition_year=2021
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:22 /user/hadoop/imdb/name_partitioned/partition_year=2022
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:22 /user/hadoop/imdb/name_partitioned/partition_year=2023
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:24 /user/hadoop/imdb/name_partitioned/partition_year=2024
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:22 /user/hadoop/imdb/name_partitioned/partition_year=2025
drwxr-xr-x   - hadoop supergroup          0 2018-10-05 19:22 /user/hadoop/imdb/name_partitioned/partition_year=2115

hadoop fs -ls /user/hadoop/imdb/name_partitioned/partition_year=2018Found 1 items
-rw-r--r--   1 hadoop supergroup    5371795 2018-10-05 19:23 /user/hadoop/imdb/name_partitioned/partition_year=2018/000
007_0
```

Dynamic Partitioning – INSERT Into Table via Hive

4. Check Success via HDFS Web Browser:



Exercises I

HDFS/Hive: Work with Partitions

HDFS/Hive Partitioning Exercises - **IMDB**

1. Execute Tasks of previous HandsOn Slides

2. Create a (*statically*) partitioned table **`imdb_actors_partitioned`**, which:
   - contains all columns of table imdb_actors
   - is statically partitioned by `partition_is_alive`, containing:
     - „*alive*“ in case actor is still alive
     - „*dead*“ in case actor is already dead

   Load all data from `imdb_actors` to table **`imdb_actors_partitioned`**

3. Create a (*dynamically*) partitioned table **`imdb_movies_and_ratings_partitioned`**, which:
   - contains all columns of the two tables `imdb_movies` and `imdb_ratings` and
   - is partitioned by start year of movie (create and add column `partition_year`).

   Load all data of `imdb_movies` and `imdb_ratings` to table:
   `imdb_movies_and_ratings_partitioned`

Exercises II Preparation

Setup HiveServer2 For Remote Connections

Setup HiveServer2

1. Edit Hive Config and add (*hive/conf/hive-site.xml*):

```
<property>
          <name>hive.server2.thrift.min.worker.threads</name>
          <value>3</value>
</property>
<property>
          <name>hive.server2.thrift.max.worker.threads</name>
          <value>5</value>
</property>
<property>
          <name>hive.server2.thrift.port</name>
          <value>10000</value>
</property>
<property>
          <name>hive.server2.thrift.bind.host</name>
          <value>localhost</value>
</property>
```


Setup HiveServer2

2. Update Hadoop Config and add (*hadoop/etc/hadoop/core-site.xml*):

```
<property>
          <name>hadoop.proxyuser.hadoop.hosts</name>
          <value>*</value>
 </property>
<property>
          <name>hadoop.proxyuser.hadoop.groups</name>
          <value>*</value>
</property>
```

3. Restart DFS and YARN:

```
stop-all.sh
start-dfs.sh
Start-yarn.sh
```


Setup HiveServer2

4. Start HiveServer2:

```
hive/bin/hiveserver2

2018-10-02 16:19:08: Starting HiveServer2
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hadoop/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4
j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/hadoop/hadoop/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 = b8d1efb3-fc8c-4ec8-bdf0-6a9a41e2ddaa
Hive Session ID = 32503981-a5fd-497e-b887-faf3ec1e686e
Hive Session ID = 00f7eab4-5a29-4ce4-ad97-e90904d9206f
Hive Session ID = 100e54c5-14c6-4acc-b398-040152b08ebf
[…]
```


Connect To HiveServer2 via JDBC

1. Download JDBC SQL Client, e.g. *DBeaver*:

Mac OSX:

```
wget https://dbeaver.io/files/dbeaver-ce-latest-installer.pkg
```

Linux (Debian):

```
wget https://dbeaver.io/files/dbeaver-ce_latest_amd64.deb
```

Linux (RPM):

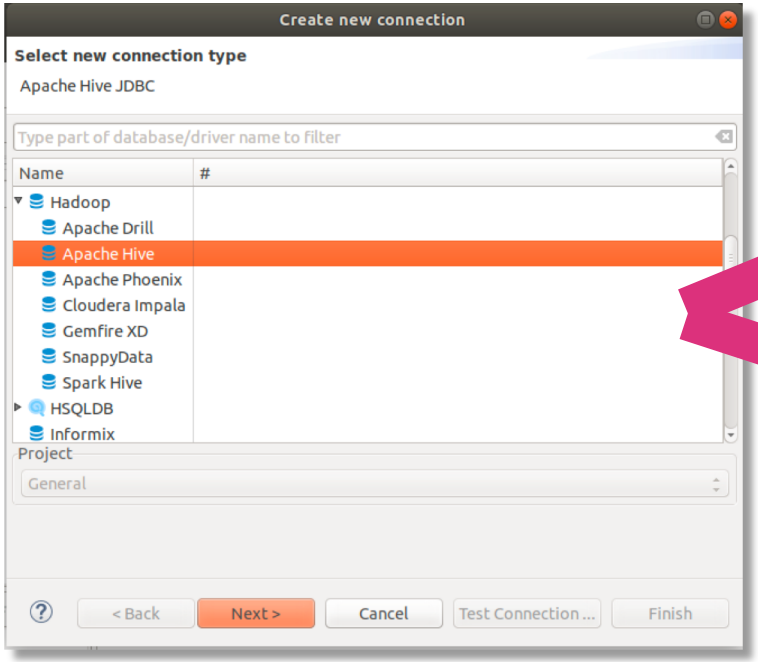
```
wget https://dbeaver.io/files/dbeaver-ce-latest-stable.x86_64.rpm
```

Windows:

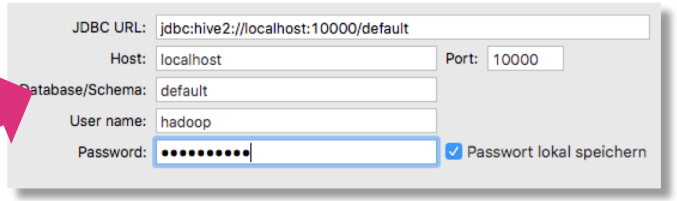
```
wget https://dbeaver.io/files/dbeaver-ce-latest-x86_64-setup.exe
```


Connect To HiveServer2 via JDBC

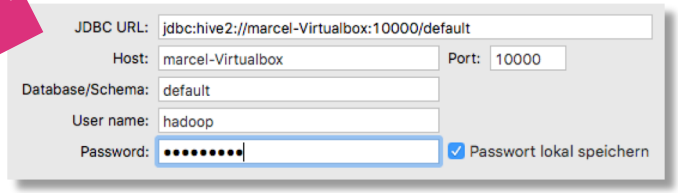
2. Configure Connection To Hive Server:



From **inside** VM:

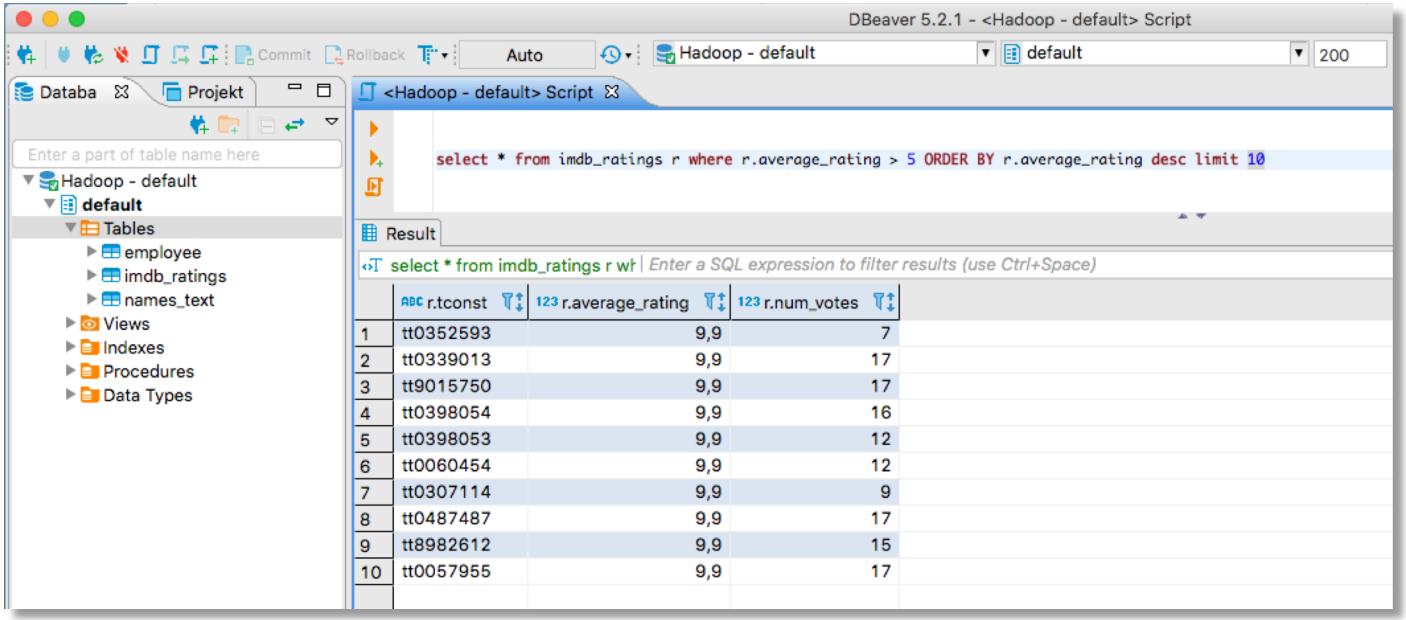


From **outside** VM:



Connect To HiveServer2 via JDBC

3. Query something, e.g.:



Exercises II

HiveServer2 For Remote Connections

HiveServer2 Exercises

1. Execute Tasks of previous HandsOn Slides.

2. Run any query.