Hortonworks Tutorial 5

Optimization on Hive

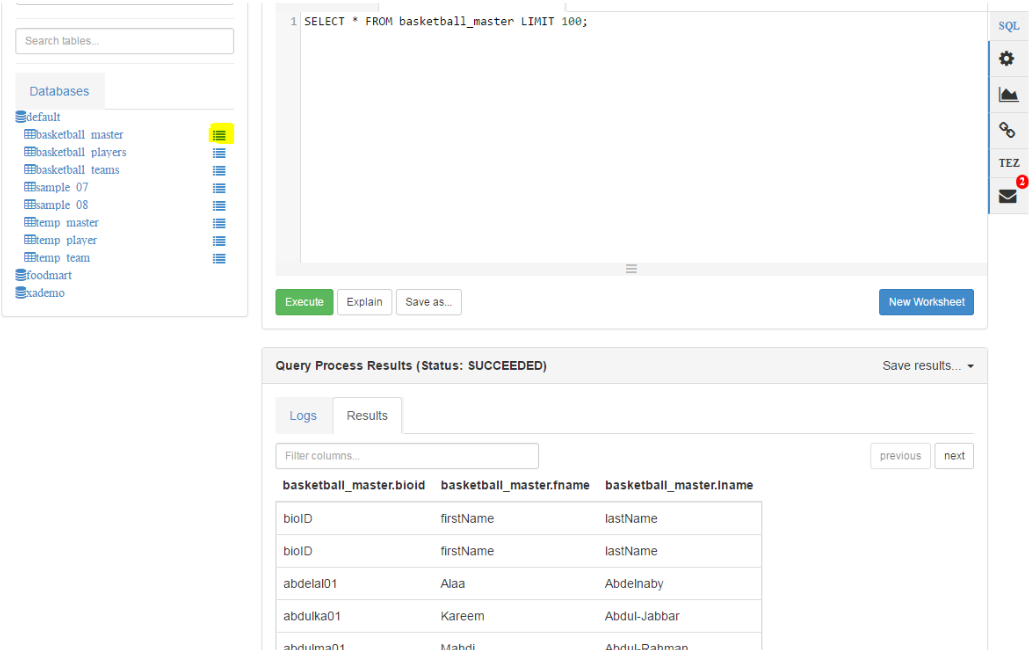
In this tutorial, we'll focus on taking advantage of performance improvements of Hive on **TEZ**. We are going to look at some of the new features that HDP 2.5 brings to Apache Hive.

Task 1. Using HDP to upload files

We will use the same dataset used in Tutorial 1. If you do not have the files, download the files from this link: `http://opensourcesports.com/files/basketball/BasketballDB-20130121.zip`. We will use the following 3 files: basketball_master.csv, basketball_players.csv and basketball_teams.csv. Make sure these files are uploaded to HDFS.

Select the "Hive View" menu. In Tutorial 1, you created three tables of basketball_teams, basketball_players, and basketball_master. You will use those tables in this tutorial. This means you should see these 3 tables in **Database Explorer**.

You can view the 100 sample records when clicking on the "Load sample data" (e.g. the one highlighted in yellow) icon right besides each table as seen in Fig. 1:



```
1 SELECT * FROM basketball_master LIMIT 100;
```

Query Process Results (Status: SUCCEEDED)

basketball_master.bioid	basketball_master.fname	basketball_master.lname
bioID	firstName	lastName
bioID	firstName	lastName
abdelal01	Alaa	Abdelnaby
abdulka01	Kareem	Abdul-Jabbar
abdulma01	Mahdi	Abdul-Rahman

Figure 1: Sample data in the table, basketball_master

Task 2. Speed Improvements

1. To take a look at the speed improvements of Hive on **TEZ**, we can run some sample queries. For this we will use the above three tables.
2. By default, the Hive view runs with **TEZ** as its execution engine. That is because **TEZ** has great speed improvements over the original MapReduce execution engine. But how much

are exactly these improvements? Well, lets find out! Click on the ”Settings”tab (see ”1” in Fig. 2). Then click on ”+Add” (see ”2” in Fig. 2).

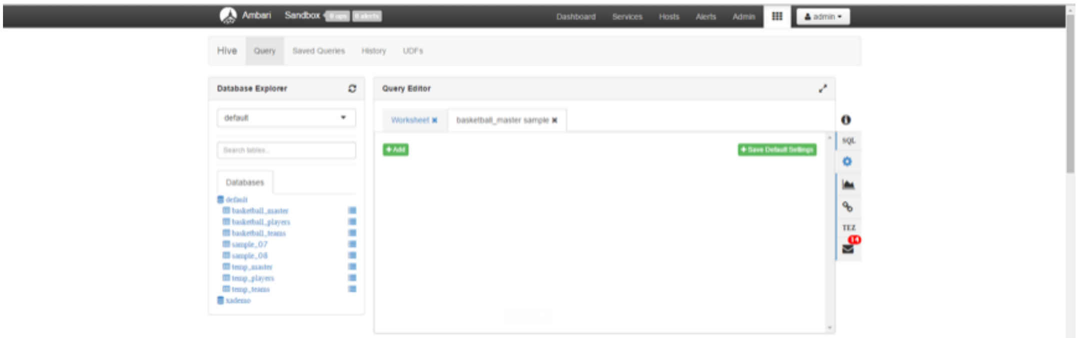
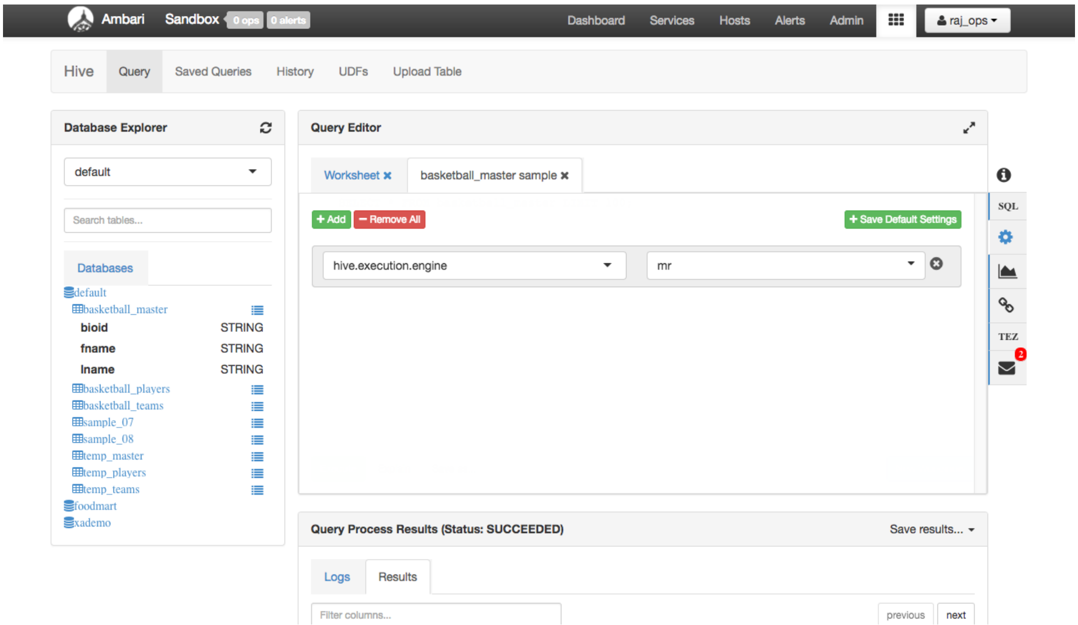


Figure 2: The ”Settings” menu in Hive View

3. Try to find the property which is”hive.execution.engine”. Select this property and then for its value select”mr”(short for MapReduce) (see Fig. 3).



bioid	STRING
fname	STRING
lname	STRING

hive.execution.engine mr

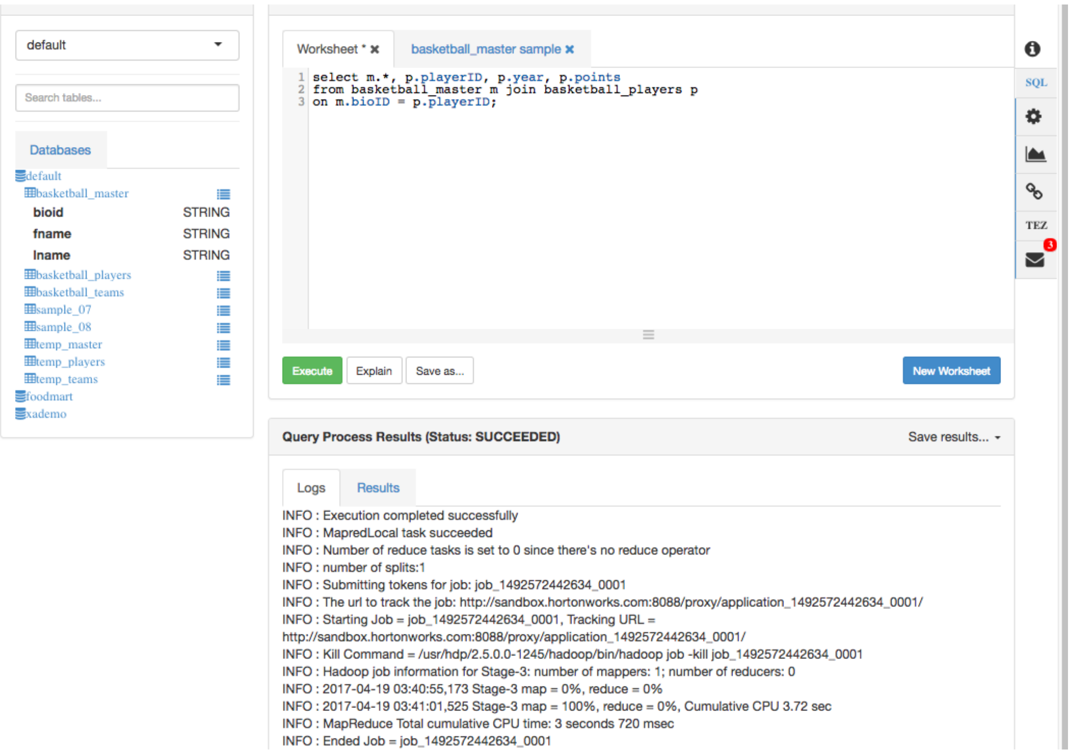
Query Process Results (Status: SUCCEEDED)

Figure 3: Using MapReduce framework

4. We now test a query using the MapReduce framework. Execute the following query and wait for the results:

```
select m.*, p.playerID, p.year, p.points from basketball_master m
join basketball_players p on m.bioID = p.playerID;
```

This query is run using the MapReduce framework as seen in ”Logs” in Fig. 4 (see the last 5 lines):



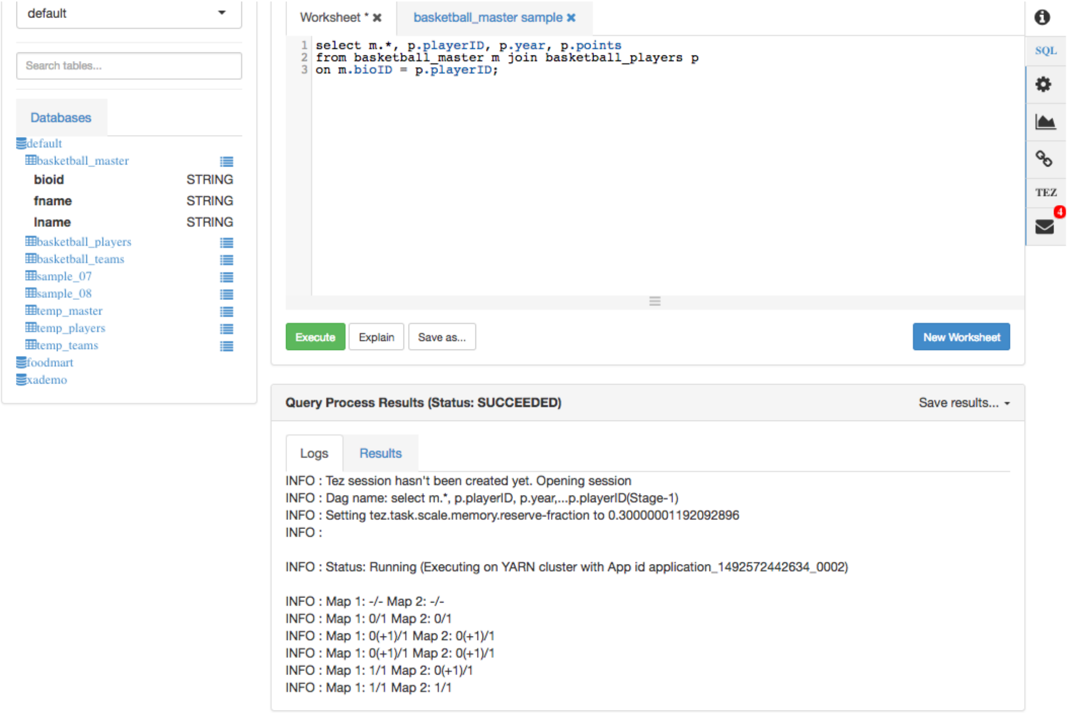
```
select m.*, p.playerID, p.year, p.points
from basketball_master m join basketball_players p
on m.bioID = p.playerID;
```

Query Process Results (Status: SUCCEEDED)

```
INFO : Execution completed successfully
INFO : MapredLocal task succeeded
INFO : Number of reduce tasks is set to 0 since there's no reduce operator
INFO : number of splits:1
INFO : Submitting tokens for job: job_1492572442634_0001
INFO : The url to track the job: http://sandbox.hortonworks.com:8088/proxy/application_1492572442634_0001/
INFO : Starting Job = job_1492572442634_0001, Tracking URL =
http://sandbox.hortonworks.com:8088/proxy/application_1492572442634_0001/
INFO : Kill Command = /usr/hdp/2.5.0.0-1245/hadoop/bin/hadoop job -kill job_1492572442634_0001
INFO : Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 0
INFO : 2017-04-19 03:40:55,173 Stage-3 map = 0%, reduce = 0%
INFO : 2017-04-19 03:41:01,525 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 3.72 sec
INFO : MapReduce Total cumulative CPU time: 3 seconds 720 msec
INFO : Ended Job = job_1492572442634_0001
```

Figure 4: Logs using the MapReduce framework

5. Now instead of using the MapReduce framework we enable TEZ. Click on the Settings in the Hive view, and then select ”hive.execution.engine” and ”tez”. Execute the same query as we had run earlier to see the speed improvements with TEZ. The printed log should be different to the log created when using MapReduce, and the time taken should be reduced. See Fig. 5.



```
select m.*, p.playerID, p.year, p.points
from basketball_master m join basketball_players p
on m.bioID = p.playerID;
```

```
INFO : Tez session hasn't been created yet. Opening session
INFO : Dag name: select m.*, p.playerID, p.year,...p.playerID(Stage-1)
INFO : Setting tez.task.scale.memory.reserve-fraction to 0.30000001192092896
INFO :

INFO : Status: Running (Executing on YARN cluster with App id application_1492572442634_0002)

INFO : Map 1: -/- Map 2: -/-
INFO : Map 1: 0/1 Map 2: 0/1
INFO : Map 1: 0(+1)/1 Map 2: 0(+1)/1
INFO : Map 1: 0(+1)/1 Map 2: 0(+1)/1
INFO : Map 1: 1/1 Map 2: 0(+1)/1
INFO : Map 1: 1/1 Map 2: 1/1
```

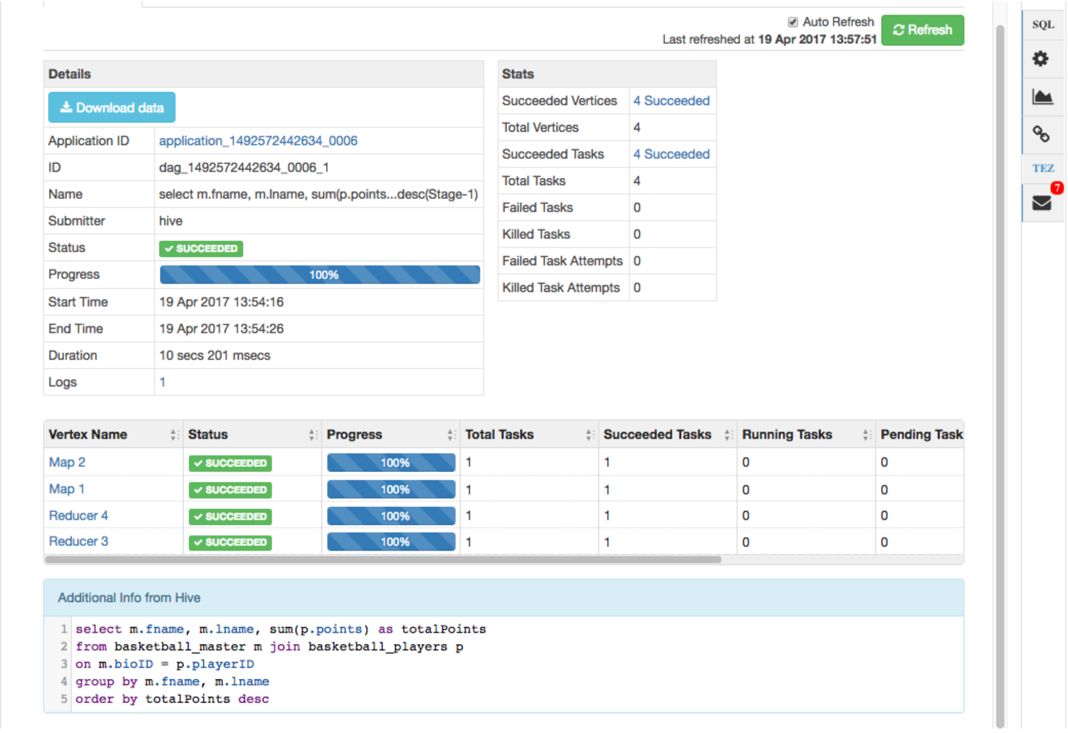
Figure 5: Logs using TEZ

6. Now let's try the following new query to work with. First execute the query first using "mr", then using "tez". The completion time should be significantly reduced when "tez" is used. Of course, the difference might be differently measured depending on the performance of the machine you use.

   ```
   select m.fname, m.lname, sum(p.points) as totalPoints
   from basketball_master m
   join basketball_players p on m.bioID = p.playerID
   group by m.fname, m.lname order by totalPoints desc;
   ```

7. When TEZ is enabled, you can also click "TEZ" tab on the right to see the DAG detail including the time taken, which is not available if executing Hive on "mr". See Fig. 6.

8. You can even obtain the time taken for each Map job by clicking on the "All Verticies" or "All Tasks" tab (e.g. Fig. 7 shows what happens when clicking on "All Vertices").



Last refreshed at 19 Apr 2017 13:57:51

Details	
Download data	
Application ID	application_1492572442634_0006
ID	dag_1492572442634_0006_1
Name	select m.fname, m.lname, sum(p.points...desc(Stage-1)
Submitter	hive
Status	✓ SUCCEEDED
Progress	100%
Start Time	19 Apr 2017 13:54:16
End Time	19 Apr 2017 13:54:26
Duration	10 secs 201 msecs
Logs	1

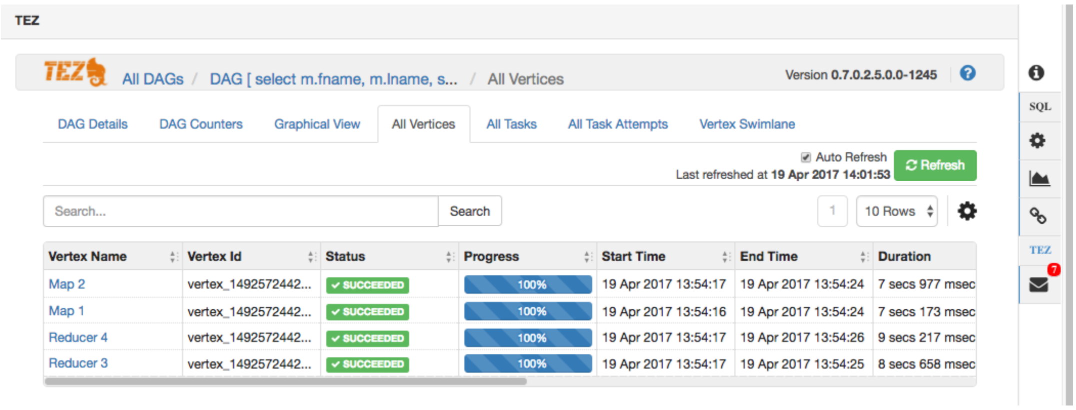
Stats	
Succeeded Vertices	4 Succeeded
Total Vertices	4
Succeeded Tasks	4 Succeeded
Total Tasks	4
Failed Tasks	0
Killed Tasks	0
Failed Task Attempts	0
Killed Task Attempts	0

Vertex Name	Status	Progress	Total Tasks	Succeeded Tasks	Running Tasks	Pending Task
Map 2	✓ SUCCEEDED	100%	1	1	0	0
Map 1	✓ SUCCEEDED	100%	1	1	0	0
Reducer 4	✓ SUCCEEDED	100%	1	1	0	0
Reducer 3	✓ SUCCEEDED	100%	1	1	0	0

Additional Info from Hive

```
select m.fname, m.lname, sum(p.points) as totalPoints
from basketball_master m join basketball_players p
on m.bioID = p.playerID
group by m.fname, m.lname
order by totalPoints desc
```

Figure 6: DAG Details in TEZ

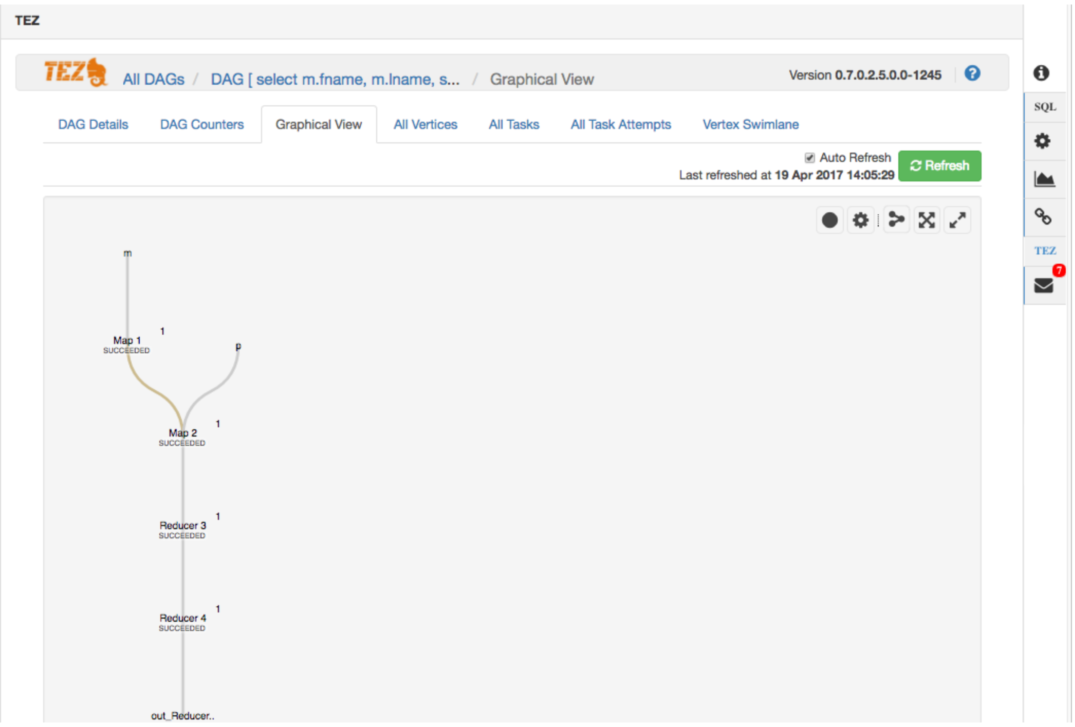


Last refreshed at 19 Apr 2017 14:01:53

Vertex Name	Vertex Id	Status	Progress	Start Time	End Time	Duration
Map 2	vertex_1492572442...	✓ SUCCEEDED	100%	19 Apr 2017 13:54:17	19 Apr 2017 13:54:24	7 secs 977 msec
Map 1	vertex_1492572442...	✓ SUCCEEDED	100%	19 Apr 2017 13:54:16	19 Apr 2017 13:54:24	7 secs 173 msec
Reducer 4	vertex_1492572442...	✓ SUCCEEDED	100%	19 Apr 2017 13:54:17	19 Apr 2017 13:54:26	9 secs 217 msec
Reducer 3	vertex_1492572442...	✓ SUCCEEDED	100%	19 Apr 2017 13:54:17	19 Apr 2017 13:54:25	8 secs 658 msec

Figure 7: The ”All Vertices” tab

9. Try clicking on the different parts above, such as”Graphical View”and explore some of the other execution information from Tez (see Fig. 8).



TEZ

All DAGs / DAG [select m.fname, m.lname, s... / Graphical View

Version 0.7.0.2.5.0.0-1245

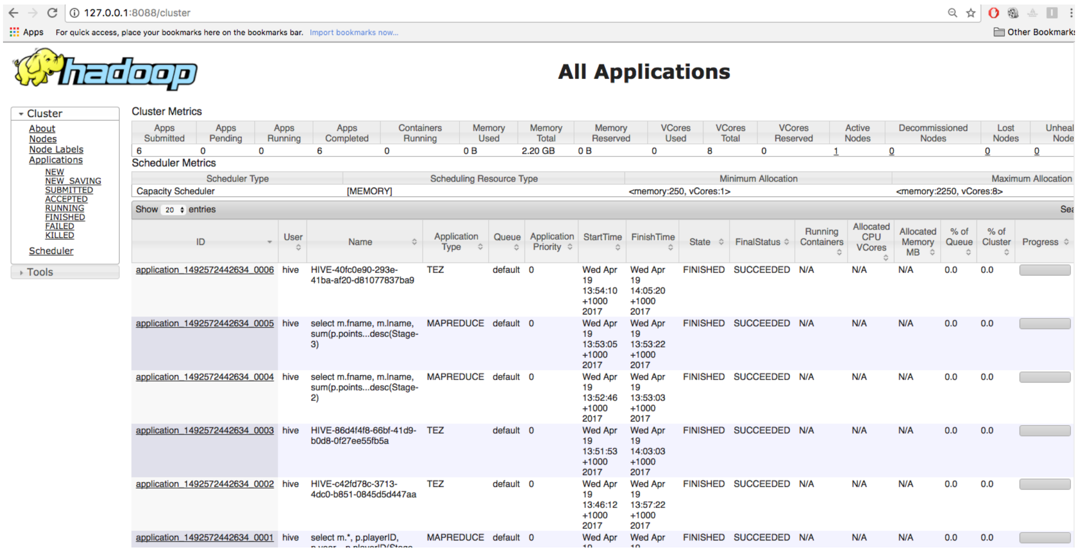
DAG Details | DAG Counters | Graphical View | All Vertices | All Tasks | All Task Attempts | Vertex Swimlane

☑ Auto Refresh

Last refreshed at 19 Apr 2017 14:05:29

Figure 8: The "Graphical View" tab

10. You can obtain further details about Tez and MapReduce jobs in the HDP Sandbox Web UI. If you go to this page `http://127.0.0.1:8088/cluster`,you can track your jobs. You can click on your job and see further details.



127.0.0.1:8088/cluster

Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

Other Bookmarks

All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unheal Node
6	0	0	6	0	0 B	2.20 GB	0 B	0	8	0	1	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:250, vCores:1>	<memory:2250, vCores:8>

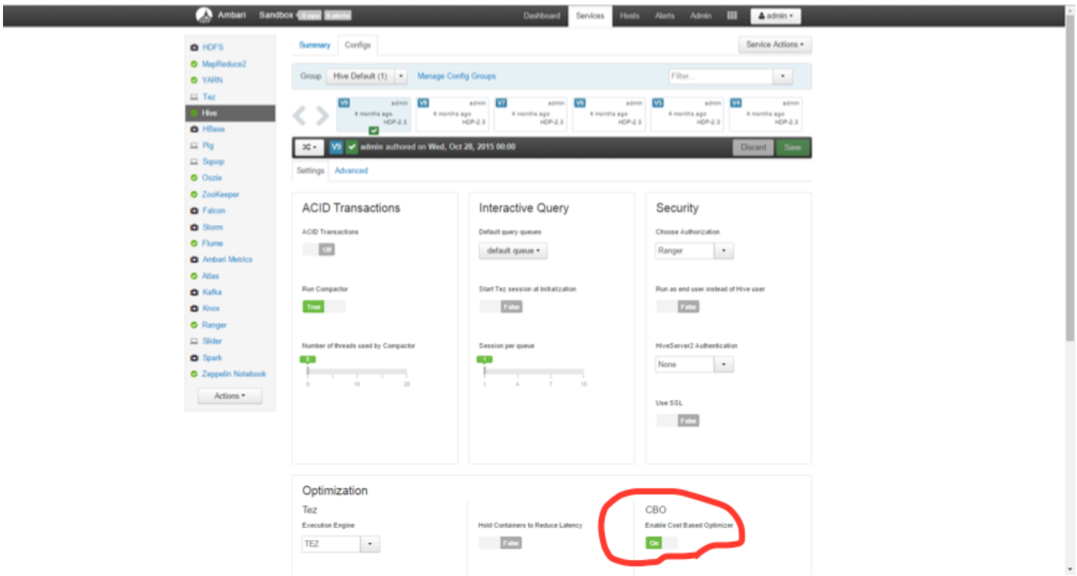
Show 20 entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	% of Queue	% of Cluster	Progress
application_1492572442634_0006	hive	HIVE-40fc0e90-293e-41ba-af20-d81077837ba9	TEZ	default	0	Wed Apr 19 13:54:10 +1000 2017	Wed Apr 19 14:05:20 +1000 2017	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0	
application_1492572442634_0005	hive	select m.fname, m.lname, sum(p.points...desc(Stage-3)	MAPREDUCE	default	0	Wed Apr 19 13:53:05 +1000 2017	Wed Apr 19 13:53:22 +1000 2017	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0	
application_1492572442634_0004	hive	select m.fname, m.lname, sum(p.points...desc(Stage-2)	MAPREDUCE	default	0	Wed Apr 19 13:52:46 +1000 2017	Wed Apr 19 13:53:03 +1000 2017	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0	
application_1492572442634_0003	hive	HIVE-86d4f4f8-66bf-41d9-b0d8-0f27ee55fb5a	TEZ	default	0	Wed Apr 19 13:51:53 +1000 2017	Wed Apr 19 14:03:03 +1000 2017	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0	
application_1492572442634_0002	hive	HIVE-c42fd78c-3713-4dc0-b851-0845d5d447aa	TEZ	default	0	Wed Apr 19 13:46:12 +1000 2017	Wed Apr 19 13:57:22 +1000 2017	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0	
application_1492572442634_0001	hive	select m.*, p.playerID,	MAPREDUCE	default	0	Wed Apr	Wed Apr	FINISHED	SUCCEEDED	N/A	N/A	N/A	0.0	0.0	

Figure 9: Tracking jobs

Task 3. Cost Based Optimization (CBO)

Cost Based Optimisation (CBO) engine uses statistics to optimize query processing in Hive. In the new version of Hortonworks, CBO is enabled by default. You can check this by selecting Ambari → Hive → Configs (as shown in Fig. 10).



Group Hive Default (1)

V9 admin authored on Wed, Oct 28, 2015 00:00

ACID Transactions

ACID Transactions: Off

Run Compactor: True

Number of threads used by Compactor: 0

Interactive Query

Default query queues: default queue

Start Tez session at Initialization: False

Session per queue: 1

Security

Choose Authorization: Ranger

Run as end user instead of Hive user: False

HiveServer2 Authentication: None

Use SSL: False

Optimization

Tez

Execution Engine: TEZ

Hold Containers to Reduce Latency: False

CBO

Enable Cost Based Optimizer: On

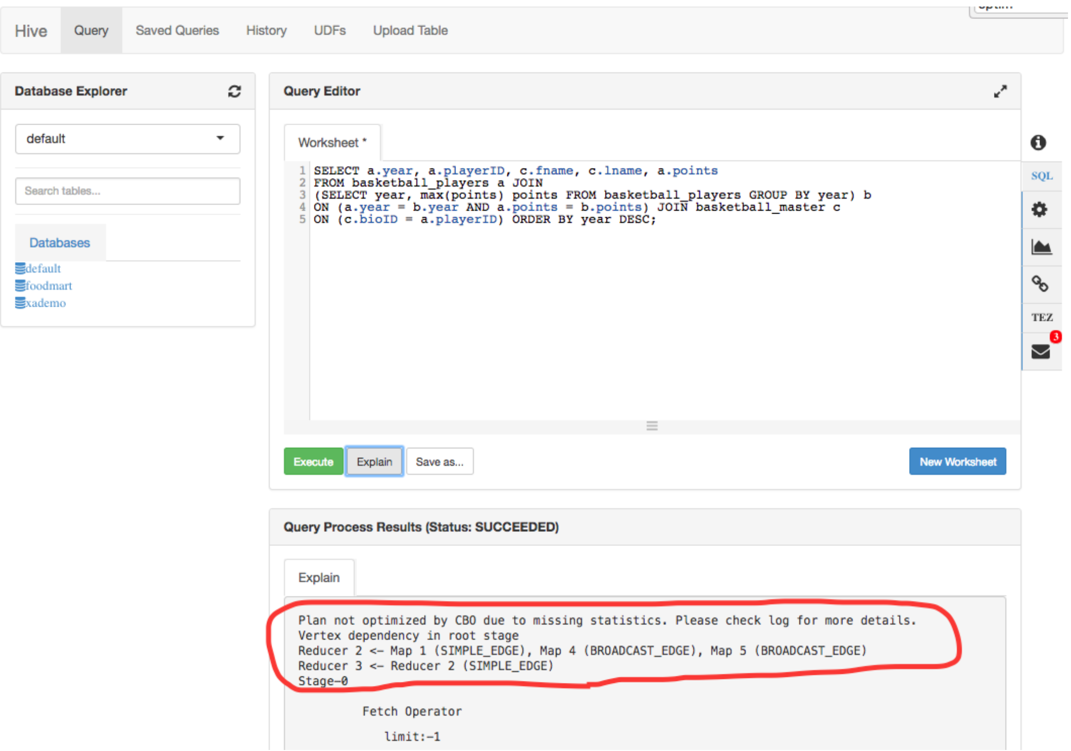
Figure 10: CBO Config

While CBO is enabled in Hive, it is not utilised for optimization due to missing statistics about tables and columns but by providing these statistics, it can be used, specially when there are joins between multiple tables.

1. **Test with queries**: Let's do a simple exercise. Lets run the following query in which on theSettingsin the Hive view, you need to choose "hive.execution.engine" and "tez".

   ```
   SELECT a.year, a.playerID, c.fname, c.lname, a.points
   FROM basketball_players a JOIN
   (SELECT year, max(points) points FROM basketball_players GROUP BY year) b
   ON (a.year = b.year AND a.points = b.points) JOIN basketball_master c
   ON (c.bioID = a.playerID) ORDER BY year DESC;
   ```

   Now, click on "Explain" for the above query. The result in the figure below shows the CBO was not used because of missing statistics (see Fig. 11).



```
SELECT a.year, a.playerID, c.fname, c.lname, a.points
FROM basketball_players a JOIN
(SELECT year, max(points) points FROM basketball_players GROUP BY year) b
ON (a.year = b.year AND a.points = b.points) JOIN basketball_master c
ON (c.bioID = a.playerID) ORDER BY year DESC;
```

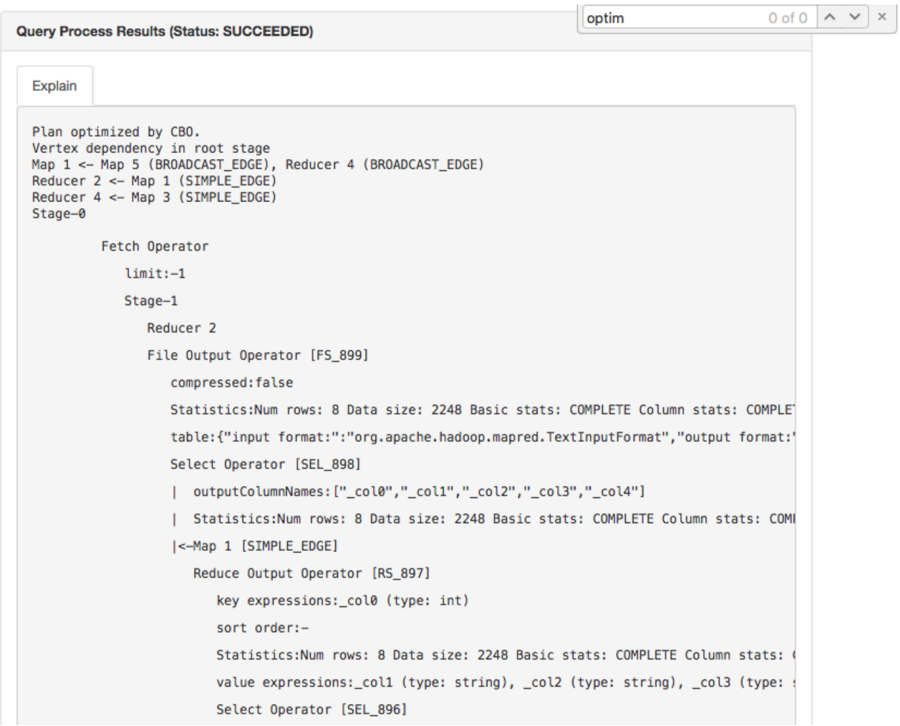
```
Plan not optimized by CBO due to missing statistics. Please check log for more details.
Vertex dependency in root stage
Reducer 2 <- Map 1 (SIMPLE_EDGE), Map 4 (BROADCAST_EDGE), Map 5 (BROADCAST_EDGE)
Reducer 3 <- Reducer 2 (SIMPLE_EDGE)
Stage-0

        Fetch Operator

           limit:-1
```

Figure 11: Not optimized result by CBO

To provide the missing statistics, write and execute the following commands ONE by ONE (it will give an error if you execute them together).

- *ANALYZE TABLE basketball_players COMPUTE STATISTICS;*
- *ANALYZE TABLE basketball_players COMPUTE STATISTICS FOR COLUMNS year, playerID, points;*
- *ANALYZE TABLE basketball_master COMPUTE STATISTICS;*
- *ANALYZE TABLE basketball_master COMPUTE STATISTICS FOR COLUMNS bioID, fname, lname;*

Now you can execute the query we tested earlier and after getting the results, click on Explain (see Fig. 12).



```
Plan optimized by CBO.
Vertex dependency in root stage
Map 1 <- Map 5 (BROADCAST_EDGE), Reducer 4 (BROADCAST_EDGE)
Reducer 2 <- Map 1 (SIMPLE_EDGE)
Reducer 4 <- Map 3 (SIMPLE_EDGE)
Stage-0

         Fetch Operator

            limit:-1

            Stage-1

               Reducer 2

               File Output Operator [FS_899]

                  compressed:false

                  Statistics:Num rows: 8 Data size: 2248 Basic stats: COMPLETE Column stats: COMPLET

                  table:{"input format:":"org.apache.hadoop.mapred.TextInputFormat","output format:'

                  Select Operator [SEL_898]

                  |  outputColumnNames:["_col0","_col1","_col2","_col3","_col4"]

                  |  Statistics:Num rows: 8 Data size: 2248 Basic stats: COMPLETE Column stats: COM

                  |<-Map 1 [SIMPLE_EDGE]

                     Reduce Output Operator [RS_897]

                        key expressions:_col0 (type: int)

                        sort order:-

                        Statistics:Num rows: 8 Data size: 2248 Basic stats: COMPLETE Column stats: (

                        value expressions:_col1 (type: string), _col2 (type: string), _col3 (type: s

                        Select Operator [SEL_896]
```

Figure 12: Optimized result by CBO

2. Visit and investigate following links to more fully understand TEZ and CBO:

- http://hortonworks.com/hadoop-tutorial/supercharging-interactive-queries-hive-tez/
- http://hortonworks.com/blog/hive-0-14-cost-based-optimizer-cbo-technical-overview/