Apache Hive (Part 4)

Part 4

Views and User Defined Functions (UDF) in Hive

Hive Views

Views are a logical construct (virtual table) that can be used to simplify queries by either abstracting away complexities such as joins or sub-queries or by pre-filtering data. Materialized views are not currently supported.

An example of subquery from Part 3:

```
SELECT ap.name, ap.Actually_Paid FROM
(
  SELECT name, salary, round(salary * (1 - deductions["Federal Taxes"]
  -deductions["State Taxes"]-deductions["Insurance"]),1) as Actually_Paid
  FROM employees23
) ap
WHERE ap.Actually_Paid > 80000;
```

Create a View and Do a Query

```
CREATE VIEW actual_pay AS
  SELECT name, salary, round(salary * (1 - deductions["Federal Taxes"]
  -deductions["State Taxes"]-deductions["Insurance"]),1) ap
  FROM employees23
```

```
SELECT name, ap from actual_pay
WHERE ap > 80000;
```

	name	ap
0	Jason Yang	97500.0
1	Ted Wang	123500.0
2	Tom McPheron	174300.0
3	Jason Jones	97500.0
4	Bob McPheron	124500.0
5	Jason Yang	97500.0
6	Tom McPheron	124500.0

User Defined Functions (UDF) in Hive

- User-Defined Functions (UDFs) are a powerful feature that allow users to **extend** HiveQL.
- A user can implement his/her own functions in Java, **Python** or other languages and add them to a session.

How to Use Python UDF

- A Python function can be used as a UDF in Hive through the HiveQL **TRANSFORM** statement.
- Hive UDFs written in Python must be accessed using the **Hadoop Streaming** concept, which allows any executable to be used as either the mapper or reducer in a MapReduce transformation. Any Python script can be adapted for Hive to make use of it, as long as the script is set up to read input and write output in a particular manner.
- The "sys" package must be imported.

Some Flight Delay Data (First 10 Rows of 88210 Rows)

.carrier_delay	weather_delay	L.nas_delay	security_delay	late_aircraft_delay
0	0	16	0	0
0	0	80	0	0
0	0	0	0	16
0	0	3	0	26
0	0	70	0	14
138	0	0	0	100
0	0	0	0	17
31	0	4	0	92
0	0	6	0	27
9	0	0	0	105

A Python UDF to Find the Max Flight Delay Type

FindMaxDelayType.py

```
#!/usr/bin/python

import sys

def findMostCommonDelayType(carrier, weather, nas, security, late_aircraft):
  delayTypes = {0 : 'carrier', 1 : 'weather', 2 : 'nas', 3 : 'security', 4 : 'late_aircraft'}
  delays = [carrier, weather, nas, security, late_aircraft]
  maxValue = max(delays)
  return delayTypes[delays.index(maxValue)]

for line in sys.stdin:
  line = line.strip()
  carrier, weather, nas, security, late_aircraft = line.split('\t')
  print findMostCommonDelayType(float(carrier), float(weather), float(nas), \
  float(security), float(late_aircraft))
```

How to Use Python UDF

The following HiveQL invokes the Python script stored in the FindMaxDelayType.py file.

```
ADD FILE /stsci5065/FindMaxDelayType.py;

SELECT TRANSFORM(carrier_delay, weather_delay, nas_delay,
                 security_delay, late_aircraft_delay)
  USING 'python FindMaxDelayType.py'
  AS maxDelayType
  FROM FLIGHT_DELAYS_2013_1_1;
```

The Result (Partial)

```
late_aircraft
carrier
nas
carrier
late_aircraft
late_aircraft
carrier
carrier
nas
late_aircraft
late_aircraft
security
late_aircraft
late_aircraft
nas
carrier
carrier
nas
nas
late_aircraft
carrier
weather
weather
Time taken: 17.786 seconds, Fetched: 88210 row(s)
```

UDF Improvement

- What are the counts of different types of delays?
- What is the most common delay type?

The Modified/Improved Python UDF

```
#!/usr/bin/python

import sys

def findMostCommonDelayType(carrier, weather, nas, security, late_aircraft):
   delayTypes = {0 : 'carrier', 1 : 'weather', 2 : 'nas', 3 : 'security', 4 : 'late_aircraft'}
   delays = [carrier, weather, nas, security, late_aircraft]
   maxValue = max(delays)
   return delayTypes[delays.index(maxValue)]

counter ={"carrier":0, "nas":0,"weather":0, "security":0, "late_aircraft":0}

for line in sys.stdin:
   line = line.strip()
   carrier, weather, nas, security, late_aircraft = line.split('\t')
   mfd = findMostCommonDelayType(carrier, weather, nas, security, late_aircraft)
   if mfd == "carrier":
      counter["carrier"] += 1
   elif mfd == "nas":
      counter["nas"] += 1
   elif mfd == "security":
      counter["security"] += 1
   elif mfd == "weather":
      counter["weather"] += 1
   elif mfd == "late_aircraft":
      counter["late_aircraft"] += 1

print counter
print "The most commom delay type is ", findMostCommonDelayType(counter["carrier"], \
counter["weather"], counter["nas"], counter["security"], counter["late_aircraft"])
```

The New Result (1)

```
hive> add file /stsci5065/FindMaxDelayType1.py;
Added resources: [/stsci5065/FindMaxDelayType1.py]
hive> select transform(carrier_delay,weather_delay,nas_delay,security_delay,late_aircraft_delay)
    > using 'python FindMaxDelayType1.py'
    > as maxDelayType
    > from flight_delays_2013_1_1;
Query ID = root_20160325152222_0076f0c8-822e-4099-87fa-a6af444ebd22
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1458917667801_0004, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/ap
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1458917667801_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2016-03-25 15:23:06,100 Stage-1 map = 0%,  reduce = 0%
2016-03-25 15:23:11,467 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.62 sec
MapReduce Total cumulative CPU time: 1 seconds 620 msec
Ended Job = job_1458917667801_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 1.62 sec   HDFS Read: 1881147 HDFS Write: 126 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 620 msec
OK
{'weather': 2772, 'nas': 32366, 'security': 188, 'carrier': 26258, 'late_aircraft': 26626}
The most commom delay type is  nas
Time taken: 13.204 seconds, Fetched: 2 row(s)
```

The New Result (2)

(omitting alias)

```
Added resources: [/stsci5065/FindMaxDelayType1.py]
hive> select transform(carrier_delay,weather_delay,nas_delay,security_delay,late_aircraft_delay)
    > using 'python FindMaxDelayType1.py'
    > from flight_delays_2013_1_1;
Query ID = root_20160325152525_9c14636a-dda3-4c5f-8406-038b208c7a41
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1458917667801_0005, Tracking URL = http://sandbox.hortonworks.com:8088/proxy/applic
Kill Command = /usr/hdp/2.2.0.0-2041/hadoop/bin/hadoop job  -kill job_1458917667801_0005
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2016-03-25 15:25:38,845 Stage-1 map = 0%,  reduce = 0%
2016-03-25 15:25:45,103 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.73 sec
MapReduce Total cumulative CPU time: 1 seconds 730 msec
Ended Job = job_1458917667801_0005
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 1.73 sec   HDFS Read: 1881147 HDFS Write: 132 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 730 msec
OK
{'weather': 2772, 'nas': 32366, 'security': 188, 'carrier': 26258, 'late_aircraft': 26626}      NULL
The most common delay type is  nas      NULL
Time taken: 12.613 seconds, Fetched: 2 row(s)
```