Big Data Technologies

Agenda

- Hive
- Spark

Hive

Word Count

```
hadoop fs -mkdir -p /user/nilesh/wordcount/input

hadoop fs -put $HADOOP_HOME/LICENSE.txt /user/nilesh/wordcount/input
```

```
CREATE EXTERNAL TABLE wordcount(lines STRING)
STORED AS TEXTFILE
LOCATION '/user/nilesh/wordcount/input';

SELECT * FROM wordcount
LIMIT 20;

SELECT word, COUNT(word) cnt FROM wordcount
LATERAL VIEW EXPLODE(SPLIT(LOWER(lines), '[^a-z]')) v_words AS word
WHERE LENGTH(word) > 1 AND word NOT IN
('is','are','you','we','over','on','under','and','to','by','from','of','an','the','as','your','above','or','or','a
ny','for','in','this','that','such','shall')
GROUP BY word
ORDER BY cnt DESC
LIMIT 20;
```

Partitioning

```
SET hive.exec.dynamic.partition.mode=strict;

TRUNCATE TABLE emp_part_dept_job;

INSERT INTO emp_part_dept_job PARTITION(deptno,job)
SELECT empno,ename,mgr,hire,sal,comm,deptno,job FROM emp_staging;
-- Dynamic partition strict mode requires at least one static partition column.

INSERT INTO emp_part_dept_job PARTITION(deptno=10,job)
SELECT empno,ename,mgr,hire,sal,comm,job FROM emp_staging WHERE deptno=10;

INSERT INTO emp_part_dept_job PARTITION(deptno=20,job)
SELECT empno,ename,mgr,hire,sal,comm,job FROM emp_staging WHERE deptno=20;

INSERT INTO emp_part_dept_job PARTITION(deptno=30,job)
SELECT empno,ename,mgr,hire,sal,comm,job FROM emp_staging WHERE deptno=30;

SELECT * FROM emp_part_dept_job;
-- bad practice -- always specify partition

SELECT * FROM emp_part_dept_job
WHERE deptno=20;
```

Bucketing

```
CREATE TABLE emp_bucketed(
empno INT,
ename STRING,
job STRING,
mgr INT,
hire STRING,
sal DOUBLE,
comm DOUBLE,
deptno INT
)
CLUSTERED BY (empno) INTO 2 BUCKETS
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

INSERT INTO emp_bucketed
SELECT * FROM emp_staging;

EXPLAIN
SELECT * FROM emp_bucketed
WHERE empno=7499;

SELECT deptno, SUM(sal) FROM emp_bucketed
GROUP BY deptno;
```

```
CREATE TABLE emp_part_bucketed(
empno INT,
ename STRING,
mgr INT,
hire STRING,
sal DOUBLE,
comm DOUBLE,
deptno INT
)
PARTITIONED BY (job STRING)
CLUSTERED BY (empno) INTO 2 BUCKETS
ROW FORMAT DELIMITED
```

```
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

DESCRIBE emp_part_bucketed;

SHOW CREATE TABLE emp_part_bucketed;

INSERT INTO emp_part_bucketed PARTITION(job)
SELECT empno,ename,mgr,hire,sal,comm,deptno,job FROM emp_staging;

SELECT * FROM emp_part_bucketed;

SELECT * FROM emp_part_bucketed WHERE job='CLERK';
```

Indexing

```
CREATE INDEX emp_job_index ON TABLE emp_staging (job) AS 'COMPACT' WITH DEFERRED REBUILD;

ALTER INDEX emp_job_index ON emp_staging REBUILD;
-- will execute MR job for building index

SHOW FORMATTED INDEX ON emp_staging;

INSERT INTO emp_staging VALUES(...);
-- DML operations don't modify indexes in Hive

ALTER INDEX emp_job_index ON emp_staging REBUILD;

SELECT * FROM emp_staging
WHERE job='ANALYST';
-- faster query execution (use EXPLAIN)

DROP INDEX emp_job_index ON emp_staging;
```

Spark

Spark Installation

- Download "spark-3.3.1-bin-hadoop3.tgz" and extract it.
- In ~/.bashrc
  - export SPARK_HOME=/path/of/spark-3.3.1-bin-hadoop3
  - export PATH=$SPARK_HOME/bin:$SPARK_HOME/sbin:$PATH
  - Close the terminal and open a new one.
  - which spark-shell
- In $SPARK_HOME/conf/spark-defaults.conf
  - spark.master spark://localhost:7077
- In $SPARK_HOME/conf/spark-env.sh
  - export SPARK_MASTER_HOST=localhost
  - export SPARK_LOCAL_IP=localhost
- In $SPARK_HOME/conf/workers
  - localhost

Executing Spark Scala Shell in Local mode

- terminal> spark-shell --master local

```
val file=sc.textFile("/home/nilesh/setup/bigdata/spark-3.3.1-bin-hadoop3/LICENSE")

val lines=file.map(line=>line.toLowerCase())

val words=lines.flatMap(line=>line.split("[^a-z]"))

val word1s=words.map(word=>(word,1))

val wordcounts=word1s.reduceByKey((a,x)=>a+x)
val capwordcounts=wordcounts.map(wc=>(wc._1.toUpperCase(), wc._2))

val result=capwordcounts.collect()

capwordcounts.toDebugString

:quit
```

`

PySpark Installation

- terminal> pip3 install pyspark
- In ~/.profile file
  - export PYSPARK_PYTHON=python3
  - export PYSPARK_DRIVER_PYTHON=python3
  - export SPARK_HOME=$HOME/.local/lib/python3.8/site-packages/pyspark
  - export PATH=$SPARK_HOME/bin:$SPARK_HOME/sbin:$PATH
- terminal> pyspark --master local

```
file=sc.textFile("/home/nilesh/setup/bigdata/spark-3.3.1-bin-hadoop3/LICENSE")

lines=file.map(lambda line:line.lower())

words=lines.flatMap(lambda line:line.split())

word1s=words.map(lambda word:(word,1))

wordcounts=word1s.reduceByKey(lambda a,x:a+x)

capwordcounts=wordcounts.map(lambda wc:(wc[0].upper(), wc[1]))

result=capwordcounts.collect()
```

```
print(result)

capwordcounts.toDebugString
```

Assignment

- Create emp_bucketed table in which emps are stored in two buckets by empno in ORC format. Also make table bucketed. Try INSERT, UPDATE and DELETE operations on the table. Observe number of reducers.
- Write a spark application to find top 10 movie ids (with max number of ratings).
- Write a spark application to find max salary per department.
- Write a spark application to find min salary per job. Take input from HDFS.
- Write a spark application to find top 20 words from a license file (other than stop words). Take input from HDFS.