Big Data Technologies

Agenda

- HBase
- Data warehousing

HBase

```
create 'emp', { NAME => 'ename' }, { NAME => 'sal', VERSIONS => 3 }

list

describe 'emp'

put 'emp', '001', 'ename', 'SMITH'
put 'emp', '001', 'sal', '800'
put 'emp', '002', 'ename', 'KING'
put 'emp', '002', 'sal', '5000'
put 'emp', '003', 'ename', 'FORD'
put 'emp', '003', 'sal', '3000'
put 'emp', '004', 'ename', 'MILLER'
put 'emp', '004', 'sal', '1250'
put 'emp', '005', 'ename', 'JAMES'
put 'emp', '005', 'sal', '900'

scan 'emp'

get 'emp', '001', 'ename', 'sal'
get 'emp', '001', { COLUMNS => ['ename', 'sal'] }

get 'emp', '001', 'sal'
put 'emp', '001', 'sal', '950'
```

```
get 'emp', '001', 'sal'
put 'emp', '001', 'sal', '1000'
get 'emp', '001', 'sal'

get 'emp', '001', { COLUMNS => ['sal'], VERSIONS=>3 }
put 'emp', '001', 'sal', '1100'
get 'emp', '001', { COLUMNS => ['sal'], VERSIONS=>3 }
put 'emp', '001', 'sal', '1200'
get 'emp', '001', { COLUMNS => ['sal'], VERSIONS=>3 }

get 'emp', '001', { COLUMNS => ['sal'], VERSIONS=>5 }
# display last 3 versions only (because at table creation we asked to store 3 versions)

disable 'emp'

enable 'emp'

put 'emp', '001', 'sal', '1250'
get 'emp', '001', { COLUMNS => ['sal'], VERSIONS=>3 }

put 'emp', '006', 'ename', 'TURNER'
put 'emp', '006', 'sal', '2200'
put 'emp', '007', 'ename', 'ADAM'
put 'emp', '007', 'sal', '2350'
put 'emp', '008', 'ename', 'JOHN'

deleteall 'emp', '005'

scan 'emp'
disable 'emp'

enable 'emp'

put 'emp', '009', 'ename', 'Emp1'
put 'emp', '009', 'sal', '1000'
put 'emp', '010', 'ename', 'Emp2'
```

```
put 'emp', '010', 'sal', '2000'
put 'emp', '011', 'ename', 'Emp3'
put 'emp', '011', 'sal', '3000'
```

Zookeeper

- terminal> hbase zkcli

```zookeeper` ls /

ls /hbase

ls /hbase/master

get /hbase/master

stat /hbase/master

ls /hbase/rs

ls /hbase/rs/nilesh-pc,16020,1675046473641

get /hbase/rs/nilesh-pc,16020,1675046473641

quit
```

```
### Importing CSV

```ruby
create 'books', 'name', 'author', 'subject', 'price'

list
```
```

```
terminal> hadoop fs -mkdir -p /user/nilesh/books

terminal> hadoop fs -put /path/to/books.csv /user/nilesh/books

terminal> start-yarn.sh

terminal> hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.separator=, -
Dimporttsv.columns="HBASE_ROW_KEY,name,author,subject,price" books /user/nilesh/books/
```

```
scan 'books'
```

User defined MR job

- We can implement MR job to process the data stored in HBase.
- For HBase data input & output, we should use TableInputFormat and TableOutputFormat (provided in HBase API).
- To create mapper (to read data from HBase table)
    - class MyHBaseTableMapper extends `TableMapper<KeyOut,ValueOut>`
        - Mapper input: Key=Row Key, Value=Column Values
        - All input data is available as byte-array.
- Implement driver class using TableMapReduceUtil
    - TableMapReduceUtil.initTableMapperJob("books", scan, MyHBaseTableMapper.class, Text.class, DoubleWritable.class, job);
    - It internally set up -- InputFormat as TableInputFormat.
- hadoop jar /path/to/job.jar pkg.HBaseBooksDriver /tmp/booksummary