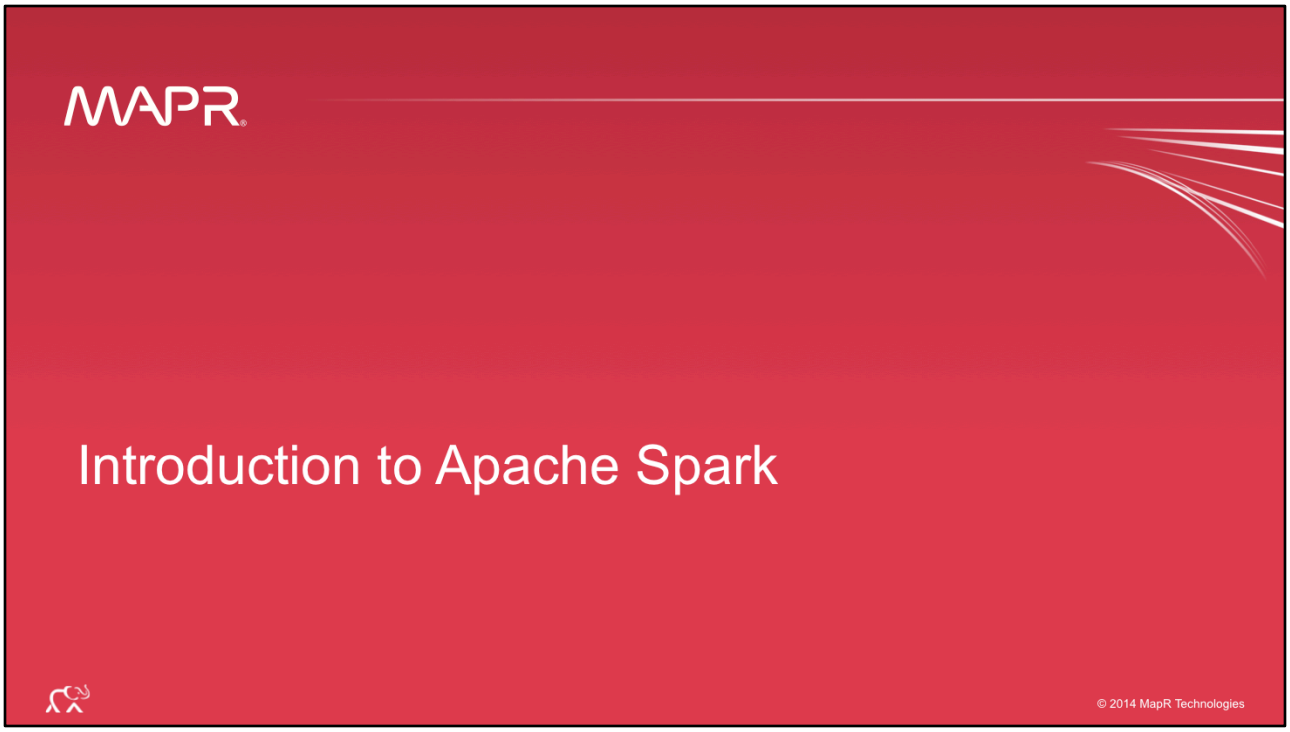
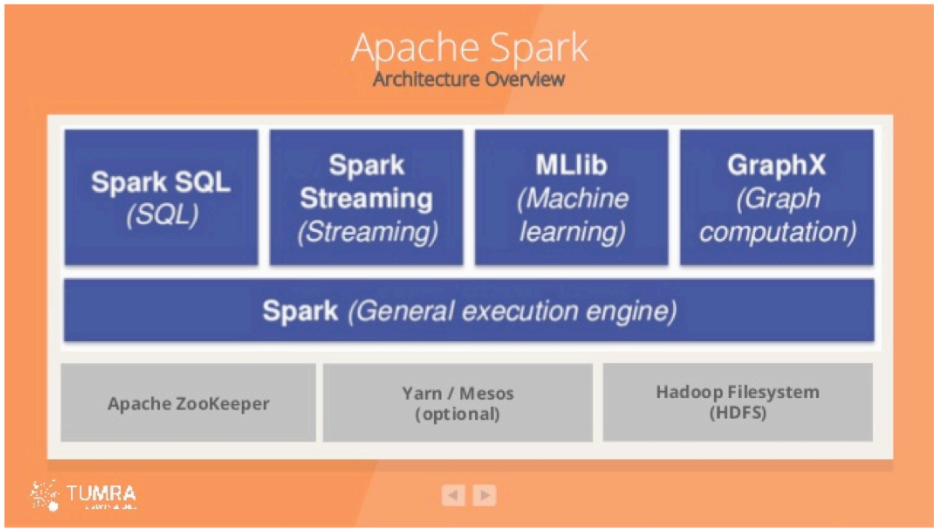


This lesson provides a brief introduction to Apache Spark.

Using Spark

Review High-Level Spark Architecture



 3

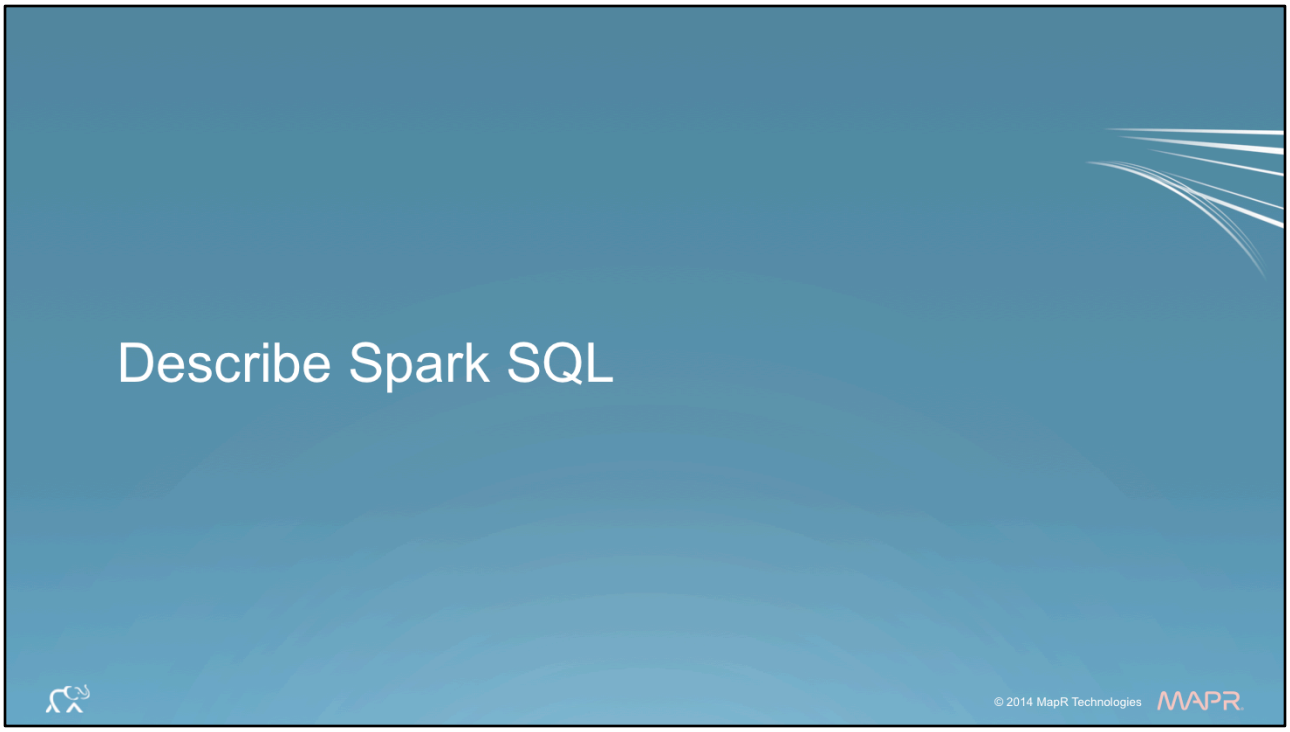
Spark example: word count

```
// Create a Java Spark Context.
    JavaSparkContext sc = new JavaSparkContext(masterURL, "wordcount");
    JavaRDD<String> input = sc.textFile(inputFile);
    // Split up into words.
    JavaRDD<String> words = input.flatMap(
      new FlatMapFunction<String, String>() {
        public Iterable<String> call(String x) {
          return Arrays.asList(x.split(" "));
        }});
    // Transform into word and count.
    JavaPairRDD<String, Integer> counts = words.mapToPair(
      new PairFunction<String, String, Integer>(){
        public Tuple2<String, Integer> call(String x){
          return new Tuple2(x, 1);
        }}).reduceByKey(new Function2<Integer, Integer, Integer>(){
            public Integer call(Integer x, Integer y){ return x + y;}});
    // Save the word count back out to a text file, causing evaluation.
    counts.saveAsTextFile(outputFile);
```

Run Spark example

```
$ /opt/mapr/spark/spark-1.2.1/bin/spark-submit \
--class "CS286.WordCount" \
spark-1.0-jar-with-dependencies.jar \
/user/user01/hosts \
/user/user01/wc.out \
yarn-client

$ cat /user/user01/wc.out/part-00000
(127.0.0.1,1)
(cs286,1)
(,2)
(10.0.2.15,1)
(localhost,1
```

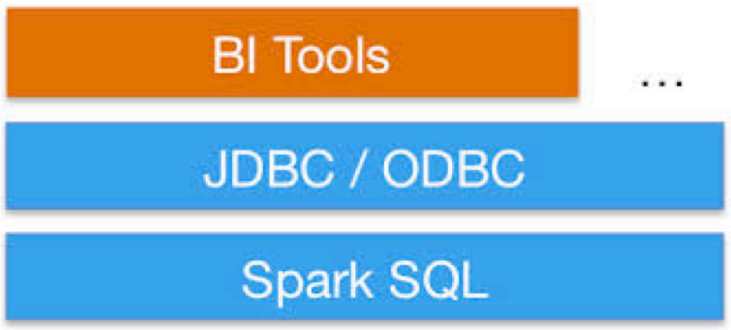
Describe Spark SQL



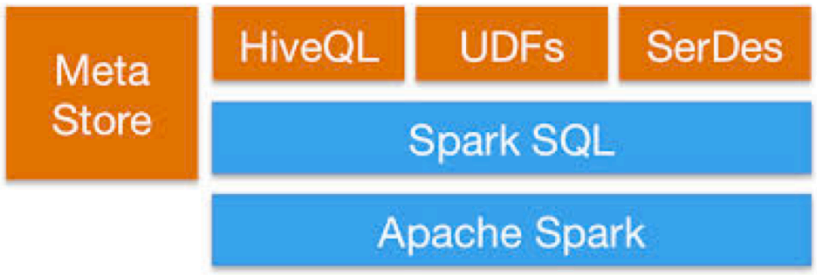
What is SQL anyway?

- SQL = structured query language
- Semi-strict specification managed by ANSI
- Used to interact with RDBMS (relational database management systems)
- Query = read-only operation (select)
- DDL = data definition language (create, alter, rename, drop)
- DCL = data control language (grant, revoke)
- DML = data manipulation language (insert, update, delete)

Spark SQL Use Case



Spark SQL Connectivity



Spark SchemaRDD

- SchemaRDD = RDD for SQL operations
- Composed of Row objects + schema that defines data types of each column in row
- Similar to a table in traditional database
- Can be created from:
  - Existing RDD
  - Parquet file
  - JSON dataset
  - Running HiveQL statement against Hive

SparkSQL example

```
JavaSparkContext sc = new JavaSparkContext(masterURL, "jsonsql");
JavaSQLContext sqlContext = new
org.apache.spark.sql.api.java.JavaSQLContext(sc);

// Create a JavaSchemaRDD from the file(s) pointed to by path
JavaSchemaRDD payload = sqlContext.jsonFile(inputFile);

// The inferred schema can be visualized using the printSchema() method.
payload.registerAsTable("mytable");

// SQL statements can be run by using the sql methods provided by
sqlContext.JavaSchemaRDD selectAll = sqlContext.sql("SELECT * FROM
mytable");
List<Row> allRows = selectAll.collect();
for(Row row: allRows) {
    System.out.println("row is " + row.toString());
}
```

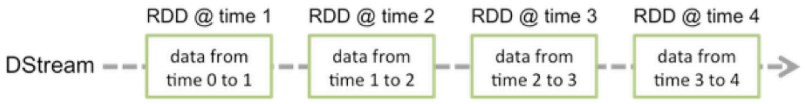
Run SparkSQL example

```
$ /opt/mapr/spark/spark-1.2.1/bin/spark-submit --class
"CS286.JsonSQL" spark-1.0-jar-with-dependencies.jar /user/
user01/payload.json  yarn-client
<output omitted>
row is
[[ArrayBuffer([[null,null,null,.done,null,com.mapr.solution.logp
arse.mrjob.RawLogToCsvParseJob,/user/mapr/fws/
data,null,RAW_TO_CSV,/user/mapr/fws/output_csv,/user/mapr/fws/
conf/patterns,SYSLOGBASE]], [[/user/mapr/fws/conf/
linuxsyslog.avsc,syslog/fws,localhost:
9201,null,RAW_TO_CSV,com.mapr.solution.logparse.mrjob.CsvToJsonE
lasticServerJob,/user/mapr/fws/
output_csv,com.mapr.solution.logparse.mrmapper.SyslogToEsMapper,
CSV_TO_ELASTICSEARCH,/user/mapr/fws/output_es,/user/mapr/fws/
conf/patterns,SYSLOGBASE]])]]
```

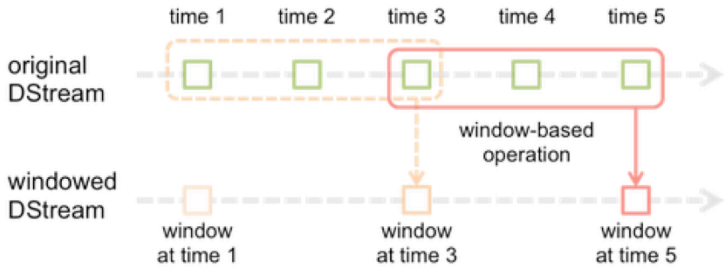
Describe Spark Streaming

Spark Streaming Data Flows

Dstreams (Discretized Streams)



Window Operations



Spark Streaming example

```
public class NetCat {
      private static final Pattern SPACE = Pattern.compile("[\\s+\\t]");

    public static void main(String[] args) {
          // check command-line args
           if(args.length != 2) {
                System.err.println("usage: NetCat <port-number> <master-url>");
                System.exit(1);
           }
        int port = Integer.parseInt(args[0]);
        String masterURL = args[1];
    // Create a StreamingContext with a local master
    JavaStreamingContext jssc = new JavaStreamingContext(masterURL, "NetCat", new
Duration(5000));
    // Create a DStream that will connect to serverIP:serverPort, like localhost:9999
    JavaReceiverInputDStream<String> lines = jssc.socketTextStream("localhost", port);
```

Spark streaming example (2)

```
JavaDStream<String> words = lines.flatMap(new FlatMapFunction<String, String>() {
       public Iterable<String> call(String x) {
             return Lists.newArrayList(SPACE.split(x));
           }
         });

         JavaPairDStream<String, Integer> wordCounts = words.mapToPair(
           new PairFunction<String, String, Integer>() {
             public Tuple2<String, Integer> call(String s) {
               return new Tuple2<String, Integer>(s, 1);
             }
           }).reduceByKey(new Function2<Integer, Integer, Integer>() {

             public Integer call(Integer i1, Integer i2) {
               return i1 + i2;
             }
           });

     wordCounts.print();
        jssc.start();
      jssc.awaitTermination();

     }
```

 MAPR 18

Run Spark Streaming example

```
$ nc -lk 9999
hello hadoop world
```

```
$ /opt/mapr/spark/spark-1.2.1/bin/spark-submit \
--class "CS286.NetCat"park-1.0-jar-with-dependencies.jar\
 9999 local
-------------------------------------------
Time: 1426178405000 ms
-------------------------------------------
(hello,1)
(world,1)
(hadoop,1)
```

Describe Spark MLlib

What is machine learning anyway?

- ML = algorithms that can learn from data
- Involves building a model to make predictions or decisions
- Discipline inclusive of
  - Statistics and math
  - Computer science
  - Some domain knowledge

How is machine learning used?

- Clustering (e.g. market segmentation)
- Collaborative filtering (e.g. product recommendation)
- Classification (e.g. SPAM email detection)
- Regression (e.g. sales forecast)

MLlib

Learning paradigm	Algorithms
Classification/ regression	Support vector machine, logistic regression, linear regression, Naïve Bayes, decision trees, ensembles of trees
Collaborative filtering	Alternating least squares
Clustering	K-means
Dimensionality reduction	Singular value decomposition, principal component analysis
Optimization	Gradient descent, limited memory BFGS

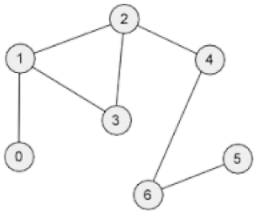
Spark MLlib example

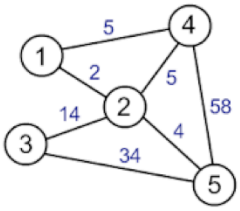
Run Spark MLlib example

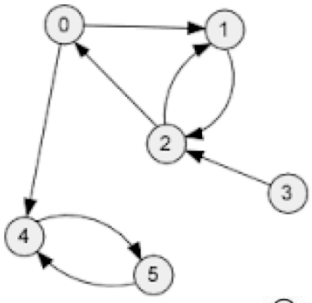
```
$ cat /user/user01/mypoints.txt
1.0 1.0
2.0 2.0
2.0 3.0
3.0 2.0
3.0 3.0
10.0 10.0

$ /opt/mapr/spark/spark-1.2.1/bin/spark-submit \
--class "CS286.JavaKMeans" \
spark-1.0-jar-with-dependencies.jar \
/user/user01/mypoints.txt \
2 \
200

Cluster centers:
 [2.2,2.2]
 [10.0,10.0]
Cost: 5.599999999999991
```

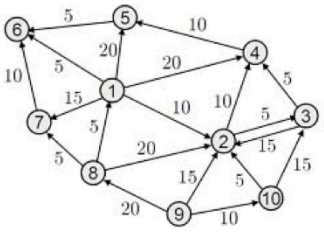
Describe Spark GraphX

What is a graph anyway?







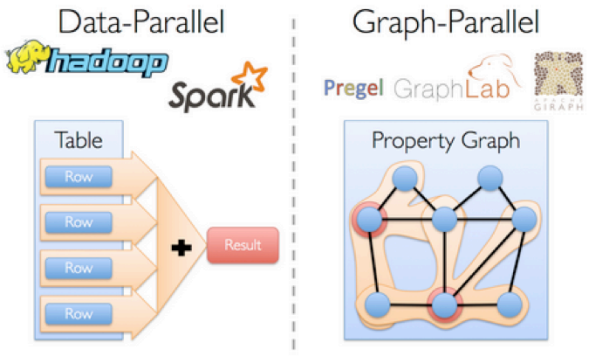


How are graphs used?

- Social network analysis
- Workflow expression
- Transportation optimization
- Networking optimization
- Targeted advertising

Spark GraphX

- https://amplab.cs.berkeley.edu/wp-content/uploads/2013/05/grades-graphx_with_fonts.pdf



Spark GraphX Use Case