Lesson 3

Data Analytics using Apache® Spark™ Components Spark SQL for Querying the data objects

Spark Supported File Formats

- Text file, Sequence File, CSV (Comma Sepereted Values) File, JSON file, Object file (for structured data, serializable and deserializable), TSV (Tab Sepereted Values) File

Spark SQL

- A component of Spark Big Data Stack for the structured data, for example, columnar, tabular
- Spark SQL components─ DataFrames (SchemaRDDs), SQLContext and JDBC server

SparkSQL

- The SQL runs the queries on Spark data in the traditional business analytics and visualization applications

SparkSQL

- Enables Spark datasets to use JDBC or ODBC API
- HQL queries also run in Spark SQL
- Runs UDFs for inline SQL, distributed DataFrames, Parquet, Hive and Cassandra Data Stores

Spark SQL

- Runs SQL like scripts for query processing, using *catalyst optimizer and tungsten execution engine*
- Processes structured data using flexible Spark SQL APIs for support for many types of data sources

Spark SQL

- Runs ETL operations by creating ETL pipeline on the data from different file-formats, such as JSON, Parquet, Hive, Cassandra and then run ad-hoc querying

Use of Aggregation functions

- Hive consists of count (*), count (expr); sum (col), sum (DISTINCT col), avg (col), avg (DISTINCT col), min (col) and DOUBLE max(col)
- Stdev( ), Samplestdev( ), variance, samplevariance( )

Use of Statistical Functions

- The statistical functions stdev(), sampleStdev(), variance, sampleVariance() for analysis with DataFrames in input

Spark SQL Features

- SparkR, PySpark, Python, Java and other language support for coding for data analysis
- Spark SQL enables users to extract their data from different formats, such as Hive, JSON and Parquet, and then transform that into required formats for ad hoc querying.

Ad hoc query

- A query ‘just for this purpose’ or query ‘on the fly.’
- For example, var newToyQuery = “SELECT * FROM table WHERE id = “ + toy_puzzleId.
- newToyQuery result will be different each time this code executes, depending on the toy_puzzleId

Spark SQL processing support

- Inclusion of Hive, HiveQL and Cassandra CQL
- Enables the use of Hive tables, database and data warehouse, UDFs and SerDe (serialization and deserialization)

Spark SQL Streaming processing support

- OLTP and structured streaming

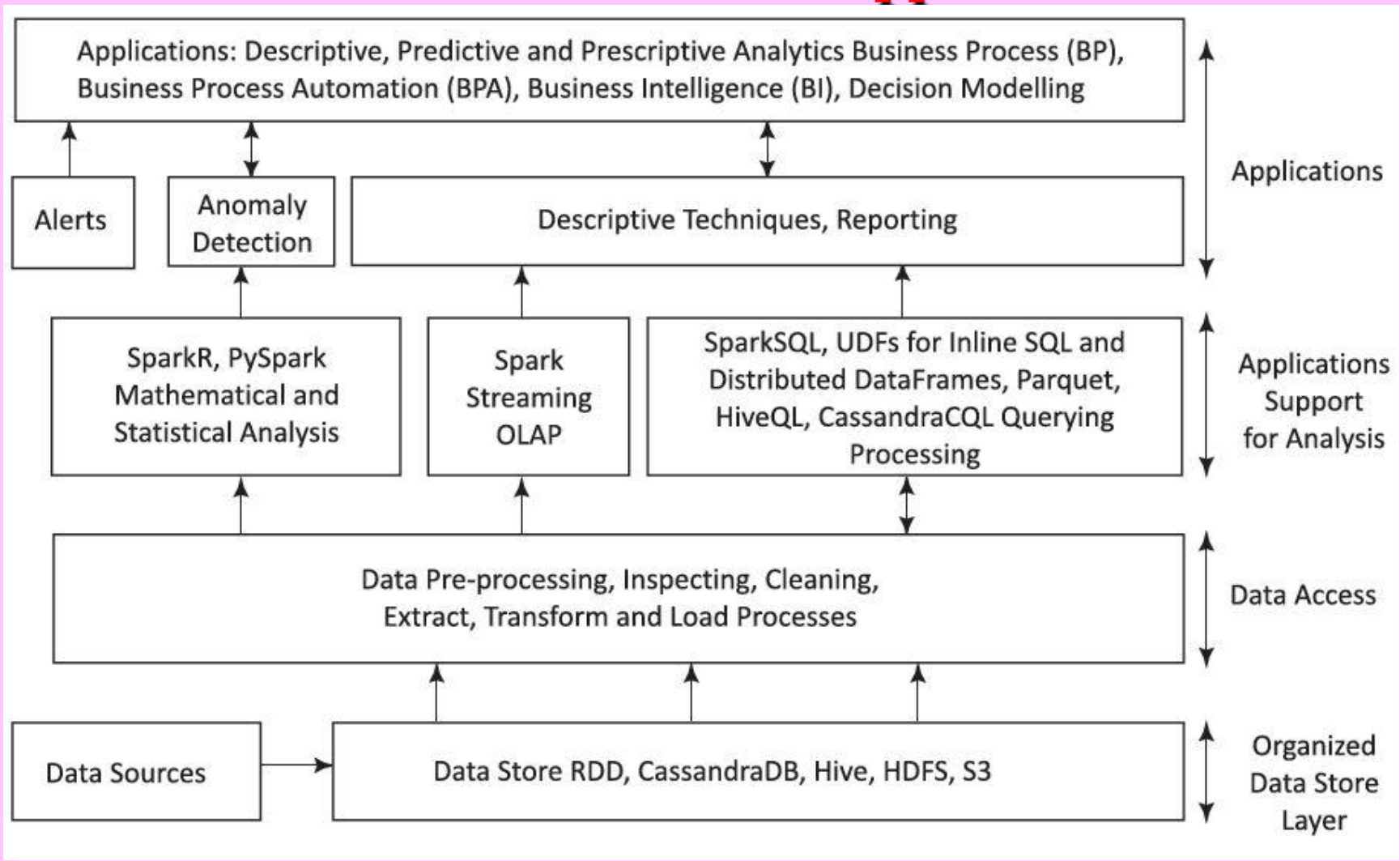
Steps For Data Analysis with Spark SQL

- Number of steps between acquisition of data from different sources and its applications
- Refer Figure 5.4 for the connectivity of applications with Spark SQL which connects to Object Stores in different formats.

Steps For Data Analysis

Refer Figure 5.4: Layer 1 Data Storage: Store of data from the multiple sources after acquisition. The Big Data storage may be in HDFS compatible files, Cassandra, Hive, HDFS or S3.

Figure 5.4 Steps between acquisition of data from different sources and its applications



Steps For Data Analysis

Refer Figure 5.4: Layer 1 Data Storage: Store of data from the multiple sources after acquisition. The Big Data storage may be in HDFS compatible files, Cassandra, Hive, HDFS or S3.

Steps For Data Analysis

Refer Figure 5.4: Layer 2a Preprocessing:
(a) dropping out of range, inconsistent and outlier values,
(b) filtering unreliable, irrelevant and redundant information,
(c) data cleaning, editing, reduction and/or wrangling,
(d) data-validation, transformation or transcoding.

Steps For Data Analysis

Refer Figure 5.4: Layer 2b ETL

Layer 3: Mathematical and statistical analysis of the data obtained after querying relevant data needing the analysis, Spark Streaming, OLAP, Spark SQL, UDFs for inline SQL, Distributed DataFrames, HiveQL, Parquet, Cassandra QL query processing

Steps For Data Analysis

Refer Figure 5.4: Layer 4 Alerts to Applications, Anomaly detection, Descriptive and Reporting

Steps For Data Analysis

Refer Figure 5.4: Layer 5 Applications for analyzing data, for example, descriptive, predictive and prescriptive analytics, business processes (BPs), business process automation (BPA), business intelligence (BI), decision modelling and knowledge discovery..

Steps For Data Analysis

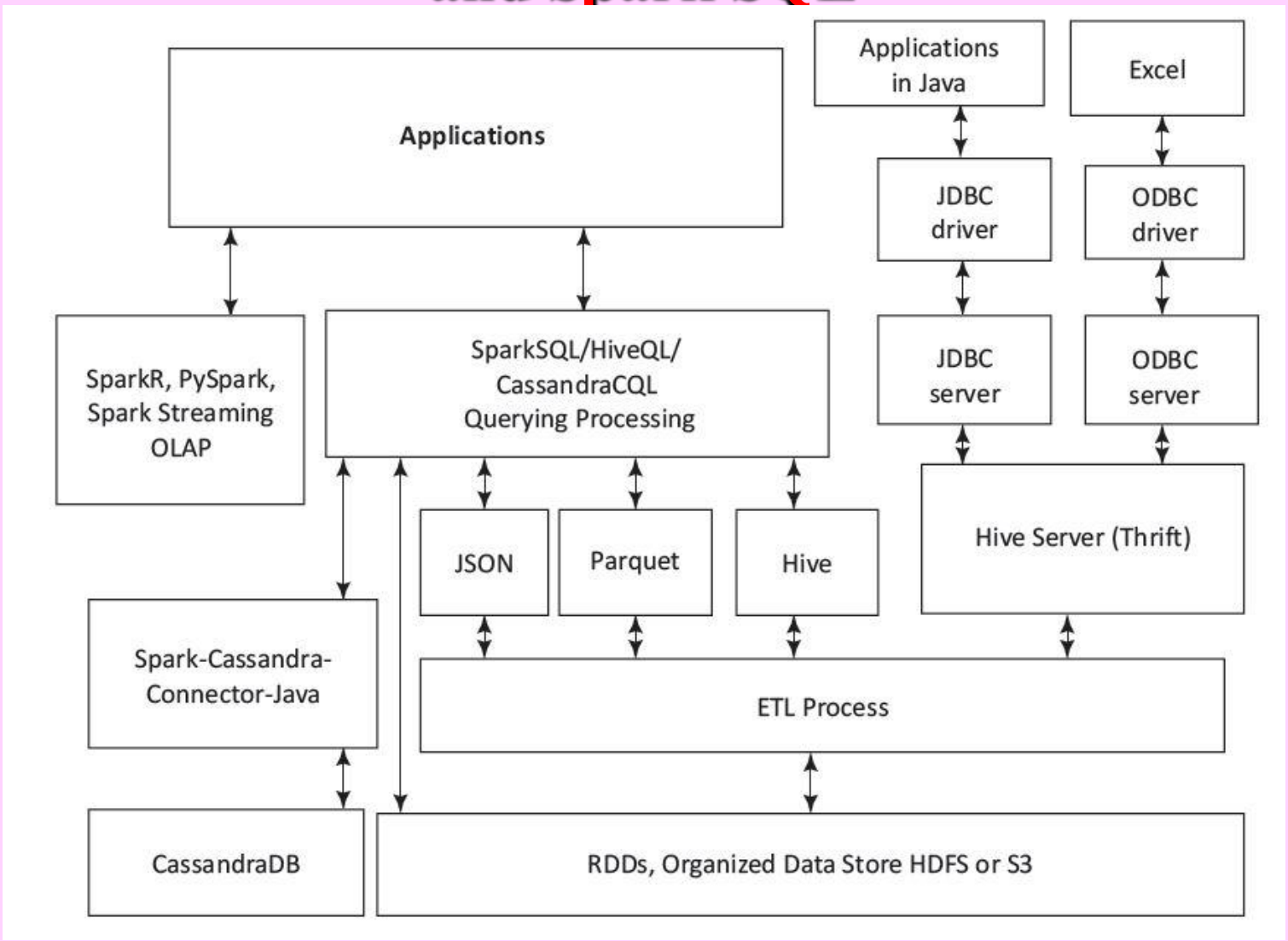
Refer Figure 5.5: Data Storage: Store of data from the multiple sources after acquisition. The Big Data storage may be in HDFS compatible files, Cassandra, Hive, HDFS or S3.

Spark SQL Connectivity to Inputs

Refer Figure 5.5 Data Flow

- Cassandra DB, DataFrames, RDDs
- Data into Spark SQL /HiveQL/ CassandraCQL for Querying Processing either through Cassandra-Spark Connector in Java or Data in Parquet, JSON or Hive tables after ETL pipeline

Figure 5.5 Connectivity between the applications and Spark SQL



Spark SQL/Hive Server (Thrift) Connectivity to outputs

- Spark SQL API JDBC connectivity using JDBC/ODBC drivers
- to the Applications

JDBC Server

- An application reads the data tables in RDBMS using a JDBC client (JDBC API at the application)
- Applications in Java connect to databases using JDBC driver and server

Hive Server (Thrift)

- Enables a remote Hive client or JDBC driver to send a request to Hive and the server sends response to that
- The client requests can be in Scala, Java, Python or R

JSON, Hive, Parquet Objects

- HDFS is highly reliable for very long running queries
- IO operations are slow
- Columnar storage used for faster IOs
- Columnar storage stores the data portion, presently required for the IOs.

JSON, Hive, Parquet Objects

- HDFS is highly reliable for very long running queries. However, IO operations are slow. Columnar storage is a solution for faster IOs. Columnar storage stores the data portion, presently required for the IOs. Load-only columns access during processing.

Columnar object Data Store

- Load-only columns access during processing
- Can be compressed or encoded according to the data type
- Also, executions of different columns or column partitions can be in parallel at the data nodes.

A nested hierarchical columnar storage concept

- Apache Parquet three projects specify the usages of files for query processing or applications
- The projects are (i) parquet-format and Thrift definitions of metadata, (ii) parquet-mr and (iii) parquet-compatibility for compatibly for read-write in multiple languages

Project parquet-mr

- Implements the sub-modules in the core components for reading and writing a nested column-oriented data stream

Use of UDFs

- Refer 10.4.2 for Analysis and Query-Processing Using UDFs in Hive and Pyspark

Summary

We learnt

- SQL Features and Support to Hive and Cassandra
- Steps between acquisition of data from different sources and its applications
- Connector in Java or Data in Parquet, JSON or Hive tables after processing in ETL pipeline

Summary

- Connectivity between the applications and Spark SQL
- JDBC/ODBC Driver
- Parquet, JSON and DataFrames as inputs to Spark SQL or Hive Server

End of Lesson 3 on

Data Analytics using Apache® Spark™ Components Spark SQL for Querying the data objects