**IPL Data Analysis | Apache Spark End-To-End Data Engineering Project**

We will proceed as per the below steps, one by one:

**1. Prepare your Data:**

* **Acquire the Data:** The video uses the IPL (Indian Premier League) cricket data set as an example [1]. You'll need to find or acquire your own data relevant to your project.
* **Upload the Data to Cloud Storage:** The video demonstrates uploading the data to Amazon S3, a cloud storage service [1]. There are other cloud storage options as well, depending on your preference.

**2. Set Up the Development Environment:**

* **Use Apache Spark:** The video utilizes Apache Spark for data processing [1]. You'll need to install and configure Apache Spark on your development machine or use a cloud-based Spark environment.

**3. Data Processing with Apache Spark:**

* **Write PySpark Code:** The video highlights using PySpark, a Python API for Apache Spark [1]. You'll write code to manipulate and transform your data according to your project's needs.

**4. Data Analysis:**

* **Use SQL for Analysis:** The video mentions using SQL queries to analyze the transformed data [1]. This indicates you can leverage your SQL skills to extract insights from the data.
* **Data Visualization:** The video suggests creating data visualizations to understand the data better [1]. Tools like Tableau or Power BI can be used to create charts and graphs.

**5. Find Insights and Make Decisions:**

* **Answer Questions:** Ultimately, the goal is to use the processed and analyzed data to answer specific questions and gain insights relevant to your project [1].
* **Make Decisions:** Based on the extracted knowledge from the data, you can make informed decisions related to your project goals.

**Additional Notes:**

* The video provides a high-level overview [18]. Each step might involve further research or breakdown into more detailed sub-steps depending on your specific project requirements and chosen tools.
* Consider referring to the resources mentioned in the video or exploring online tutorials for a deeper understanding of each step, especially regarding Apache Spark, PySpark coding, and data visualization tools.

**Architecture Diagram For the Project.**

A diagram of a company

Description automatically generated

A diagram of a computer program

Description automatically generated

Apache Spark Core : Heart of Apache. Responsible for executing every code submitted to Spark Application.

Spark SQL : Supports SQL queries inside Spark application.

Spark Streaming : to Handle and process the real time data. Like the ones we see on google maps or uber.

MLlib (Machine Learning) : supports running machine learning application on large data.

GraphX (Graph) : Supports processing the graphical data space inside Spark Application.

**Apache Spark** is a unified computing engine and a set of libraries for parallel data processing on computer clusters. It’s a framework that divides your data in multiple partitions and processes that data on to the large clusters of machines. It will combine the output at the end again.

**APACHE SPARK  
  
A diagram of a software development process

Description automatically generated with medium confidence**

**SPARK ARCHITECTURE.**

Diagram of a computer process

Description automatically generated

**Driver Process**

It’s the manager. The driver process runs your main() function, sits on a node in the cluster, and is responsible for three things:

* Maintaining information about the Spark Application
* Responding to a user's program or input
* Analyzing, distributing, and scheduling work across the executors

**Executor Processes**

The executors are responsible for carrying out the work that the driver assigns them.

* Executing code assigned to it by the driver.
* Reporting the state of the computation on that executor back to the driver node.

Spark, in addition to its cluster mode, also has a local mode. Which means that they can live on the same machine or different machines. In local mode, the driver and executors run (as threads) on your individual computer instead of a cluster.

**TheSparkSession:**  You control your Spark Application through a driver process called the SparkSession.

The SparkSession instance is the way Spark executes user-defined manipulations across the cluster. There is a one-to-one correspondence between a SparkSession and a Spark Application.

In Scala and Python, the variable is available as **spark** when you start the console.

**Spark DataFrame:**

Its similar to spreadsheets. A DataFrame is the most common Structured API and simply represents a table of data with rows and column.

In Spark all this data is stored distributed fashion. Fragments of data can be on different machines.

**Tranformations :** These are instructions used to specify the modifications to Spark about how we want to change the ‘DataFrame’.

In Spark, core data structures are immutable. They cannot be changed after they are created.

We will have to instruct.

* Spark keeps on stacking all the transformation logic(like filter, group.. etc) until there is an action (eg : show) demanded. Once we put an action to it, it executes all the transformations at once.

**DATABRICKS** is a software that support Apache Spark. It take cares of all the infrastructural component and just lets us work on the business side.

* + StructField("match\_id", IntegerType(), true) : **TRUE** here means that the filed

can have NULL values as well.

* + Once the schema has been provided, while reading the data frame, Spark will add the schema information. If there is a any inconsistencies in the data, Spark will automatically apply the schema on top of it.