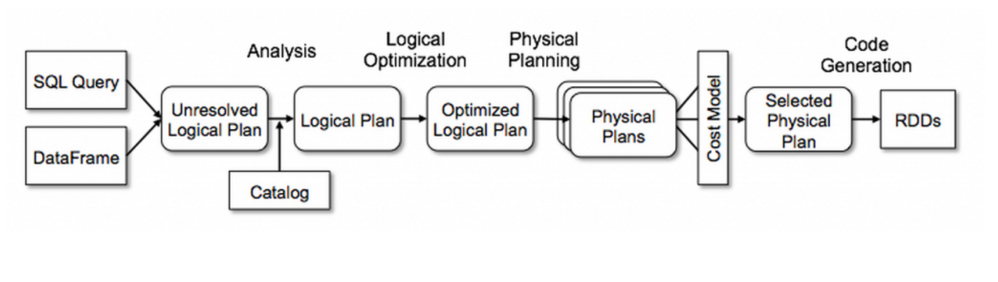
***Catalyst optimizer – Role of Logical and Physical Planning in Dataframes/Spark SQL***



***What happens behind the scenes***

Moment a query is written , internally catalyst optimizer takes the query and runs through series of steps

* Analysis – Analyses the plain query itself , looks to see if the columns are valid etc. ( Based on a rule book )
* Logical planning and optimization – Logically evaluates and modifies . Ex: casting the expressions, simplying boolean expressions etc . Also optimizes the expressions ex: if there are two filter conditions, it simplifies using AND condition etc.
* Physical planning - Generates multiple Physical plans as output

Based on the cost metrics, best physical plan is chosen

***Sample Execution Plan***

val input = sc.textFile("marketing\_data.csv");

val dataDF = input.map(x=>x.replace("\"","").split(';')).map(y=>(y(0),y(1),y(2),y(3),y(4),y(5),y(6),y(7),y(8),y(9),y(10),y(11),y(12),y(13),y(14),y(15),y(16))).toDF("age", "job","marital","education","default","balance","housing","loan","contact","month","day\_of\_week","duration","campaign","pdays","previous","poutcome","subscribed");

scala> val newDataDF = dataDF.where(col("age") !== "age").explain(true);

**== Parsed Logical Plan ==**

'Filter NOT ('age = age)

+- Project [\_1#0 AS age#17,\_2#1 AS job#18,\_3#2 AS marital#19,\_4#3 AS education#20,\_5#4 AS default#21,\_6#5 AS balance#22,\_7#6 AS housing#23,\_8#7 AS loan#24,\_9#8 AS contact#25,\_10#9 AS month#26,\_11#10 AS day\_of\_week#27,\_12#11 AS duration#28,\_13#12 AS campaign#29,\_14#13 AS pdays#30,\_15#14 AS previous#31,\_16#15 AS poutcome#32,\_17#16 AS subscribed#33]

+- LogicalRDD [\_1#0,\_2#1,\_3#2,\_4#3,\_5#4,\_6#5,\_7#6,\_8#7,\_9#8,\_10#9,\_11#10,\_12#11,\_13#12,\_14#13,\_15#14,\_16#15,\_17#16], MapPartitionsRDD[4] at rddToDataFrameHolder at <console>:29

**== Analyzed Logical Plan ==**

age: string, job: string, marital: string, education: string, default: string, balance: string, housing: string, loan: string, contact: string, month: string, day\_of\_week: string, duration: string, campaign: string, pdays: string, previous: string, poutcome: string, subscribed: string

Filter NOT (age#17 = age)

+- Project [\_1#0 AS age#17,\_2#1 AS job#18,\_3#2 AS marital#19,\_4#3 AS education#20,\_5#4 AS default#21,\_6#5 AS balance#22,\_7#6 AS housing#23,\_8#7 AS loan#24,\_9#8 AS contact#25,\_10#9 AS month#26,\_11#10 AS day\_of\_week#27,\_12#11 AS duration#28,\_13#12 AS campaign#29,\_14#13 AS pdays#30,\_15#14 AS previous#31,\_16#15 AS poutcome#32,\_17#16 AS subscribed#33]

+- LogicalRDD [\_1#0,\_2#1,\_3#2,\_4#3,\_5#4,\_6#5,\_7#6,\_8#7,\_9#8,\_10#9,\_11#10,\_12#11,\_13#12,\_14#13,\_15#14,\_16#15,\_17#16], MapPartitionsRDD[4] at rddToDataFrameHolder at <console>:29

**== Optimized Logical Plan ==**

Project [\_1#0 AS age#17,\_2#1 AS job#18,\_3#2 AS marital#19,\_4#3 AS education#20,\_5#4 AS default#21,\_6#5 AS balance#22,\_7#6 AS housing#23,\_8#7 AS loan#24,\_9#8 AS contact#25,\_10#9 AS month#26,\_11#10 AS day\_of\_week#27,\_12#11 AS duration#28,\_13#12 AS campaign#29,\_14#13 AS pdays#30,\_15#14 AS previous#31,\_16#15 AS poutcome#32,\_17#16 AS subscribed#33]

+- Filter NOT (\_1#0 = age)

+- LogicalRDD [\_1#0,\_2#1,\_3#2,\_4#3,\_5#4,\_6#5,\_7#6,\_8#7,\_9#8,\_10#9,\_11#10,\_12#11,\_13#12,\_14#13,\_15#14,\_16#15,\_17#16], MapPartitionsRDD[4] at rddToDataFrameHolder at <console>:29

**== Physical Plan ==**

Project [\_1#0 AS age#17,\_2#1 AS job#18,\_3#2 AS marital#19,\_4#3 AS education#20,\_5#4 AS default#21,\_6#5 AS balance#22,\_7#6 AS housing#23,\_8#7 AS loan#24,\_9#8 AS contact#25,\_10#9 AS month#26,\_11#10 AS day\_of\_week#27,\_12#11 AS duration#28,\_13#12 AS campaign#29,\_14#13 AS pdays#30,\_15#14 AS previous#31,\_16#15 AS poutcome#32,\_17#16 AS subscribed#33]

+- Filter NOT (\_1#0 = age)

+- Scan ExistingRDD[\_1#0,\_2#1,\_3#2,\_4#3,\_5#4,\_6#5,\_7#6,\_8#7,\_9#8,\_10#9,\_11#10,\_12#11,\_13#12,\_14#13,\_15#14,\_16#15,\_17#16]

newDataDF: Unit = ()