ADALTAS

Data Analysis with APACHE Spark™

2. SparkSQL & DataFrames

RDDs: Pros and Cons

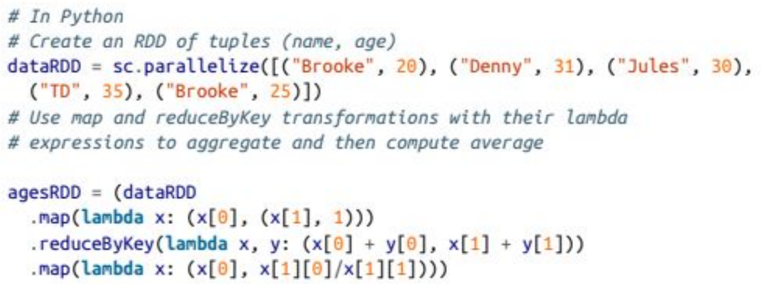
- For user:
  - complicated to express complex ideas
  - difficult to understand the code
- For Spark: lambda functions are opaque (no optimization)

+ Developers: low level control of execution

DataFrames

- Structured dataset:
  - In-memory, distributed tables
  - Named and typed columns: schema
  - Collection of `Row`s
- Sources available: structured files, Hive tables, RDBMS (MySQL, PostgreSQL, ...), RDDs
- High-level APIs

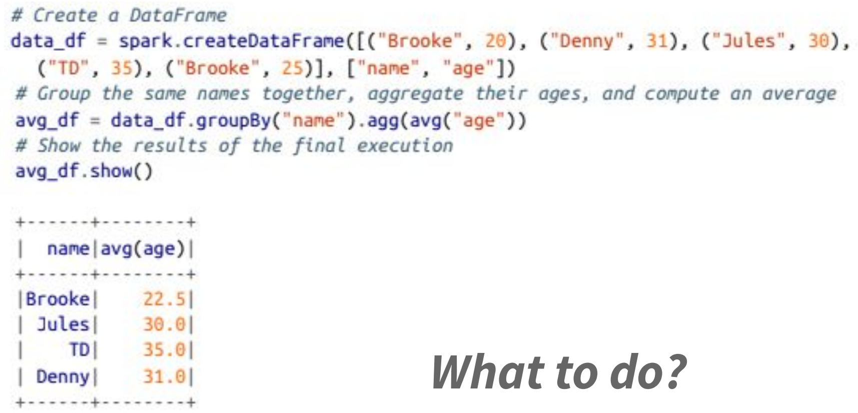
RDDs vs DataFrames: code



```python
# In Python
# Create an RDD of tuples (name, age)
dataRDD = sc.parallelize([("Brooke", 20), ("Denny", 31), ("Jules", 30),
  ("TD", 35), ("Brooke", 25)])
# Use map and reduceByKey transformations with their lambda
# expressions to aggregate and then compute average

agesRDD = (dataRDD
  .map(lambda x: (x[0], (x[1], 1)))
  .reduceByKey(lambda x, y: (x[0] + y[0], x[1] + y[1]))
  .map(lambda x: (x[0], x[1][0]/x[1][1])))
```

How to do it?



```python
# Create a DataFrame
data_df = spark.createDataFrame([("Brooke", 20), ("Denny", 31), ("Jules", 30),
  ("TD", 35), ("Brooke", 25)], ["name", "age"])
# Group the same names together, aggregate their ages, and compute an average
avg_df = data_df.groupBy("name").agg(avg("age"))
# Show the results of the final execution
avg_df.show()
```

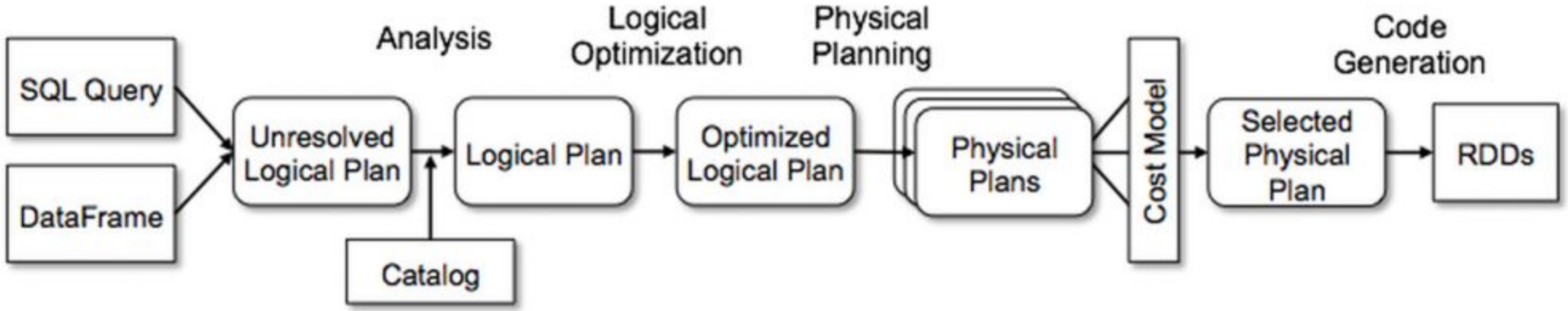
```
+------+--------+
|  name|avg(age)|
+------+--------+
|Brooke|    22.5|
| Jules|    30.0|
|    TD|    35.0|
| Denny|    31.0|
+------+--------+
```

What to do?

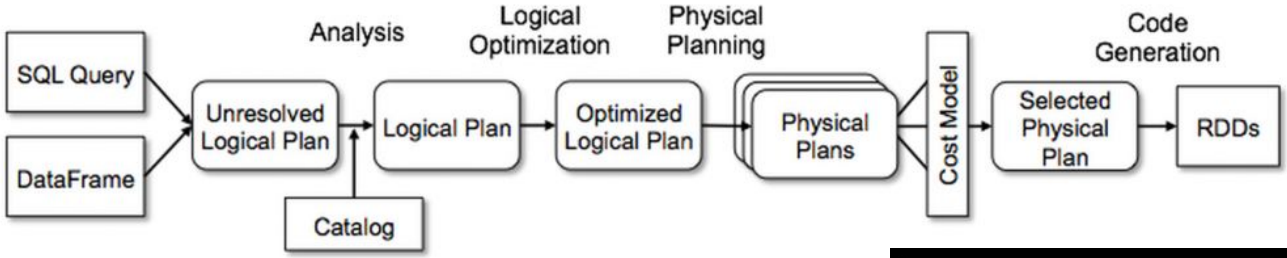
Catalyst Optimizer

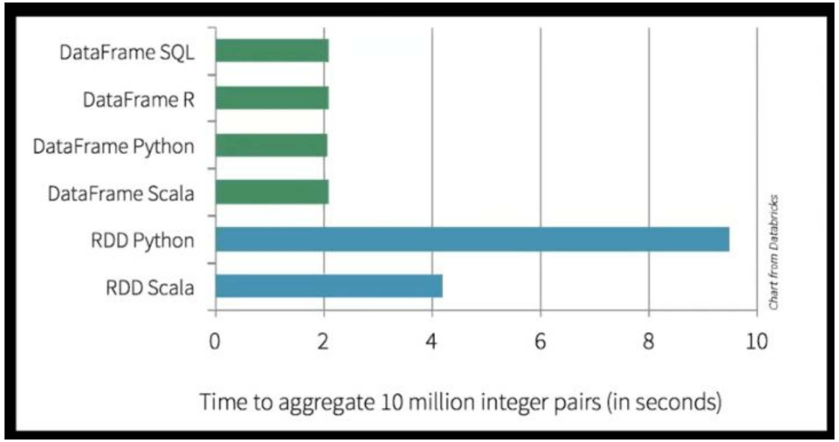
```
someRdd
    .reduceByKey(lambda x, y: ...)
    .filter(lambda x: ...)
```

```
someDF
    .groupBy("...")
    .filter(cond)
```



Catalyst Optimizer





Working with DataFrames

Querying DataFrames:

- By chaining functions
- By writing standard SQL strings

Why SQL?

- Around since the 70s
- Huge enterprise usage:
    - Lots of users
    - Lots of projects
- But: cannot be used for ML or graph analyses