Apache Drill

Haridas N

Agenda

- Get your cluster ready using easy scripts
- Try out different use-cases
- Where Drill fits best
- Drill vs Spark : Try SQL on spark and Drill

Get your cluster ready for Session: Apache Drill

NOTE: For Mac or Windows environment, ensure docker got enough CPU and RAM, change it from docker settings. Minimum machine spec expected is 8GB RAM, 4 Core.

```
$ git clone https://github.com/haridas/hadoop-env

$ cd hadoop-env

$ bash ./bin/clean-all.sh

$ bash ./bin/start-hadoop.sh

$ bash ./bin/start-spark.sh

$ bash ./bin/start-drill.sh

$ docker ps
```

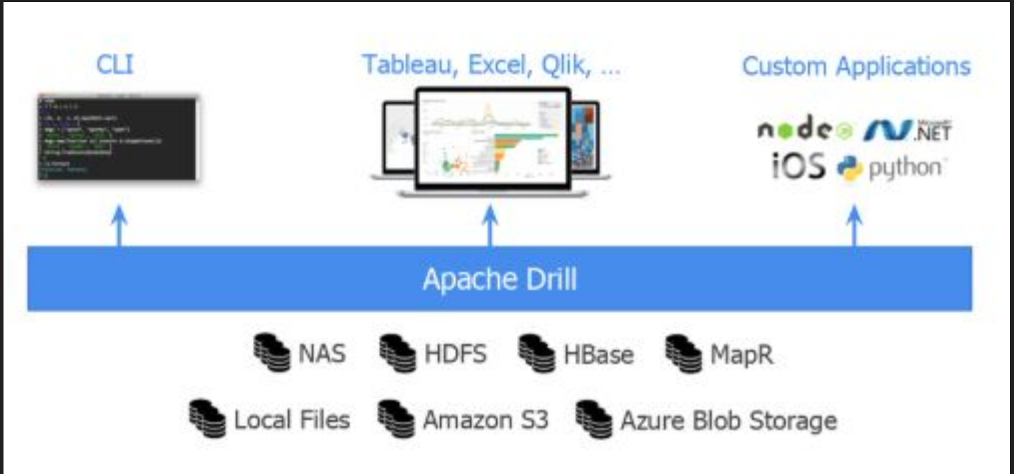
* Don’t be afraid to check inside above scripts. !

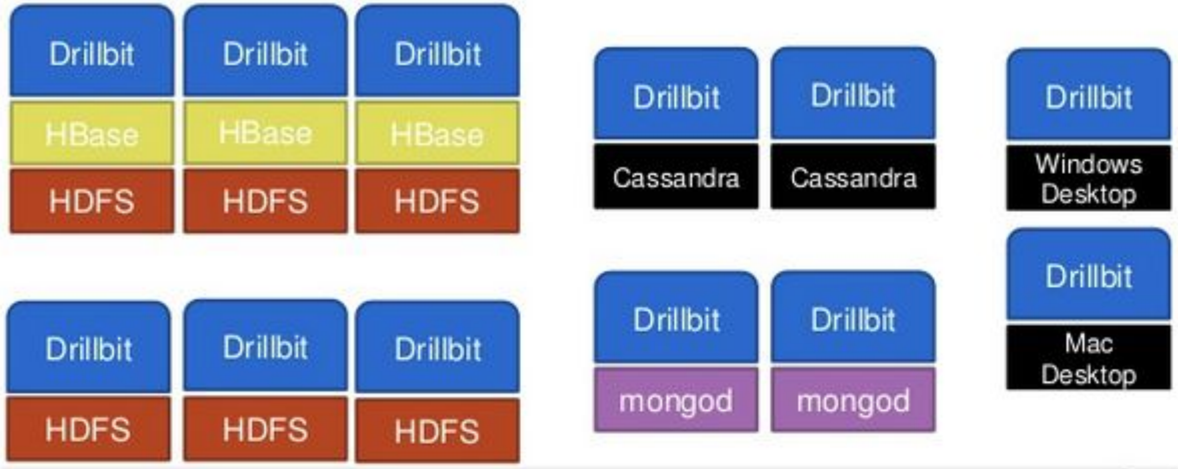
What’s Drill

- Apache Drill is inspired from the Dremel query engine
- Dremel is the engine for BigQuery
- It brings its on computation framework which is optimized for SQL query execution in the wild.
- Apache Impala another tool inspired from Dremel
- Map-reduce paper published on - 2004 (Hadoop on 2011)
- Dremel paper published on - 2010
- http://prestodb.github.io/docs/current/

Drill (cont.)

- Queries are pushed down to the nodes where the data belongs.
- There is no map-reduce conversion.
- Drill cluster can run standalone,





picture source : Drilling into Data with Apache Drill

WHY Drill

- SQL 2003 syntax fully supported, with custom UDFs on Java.
- Schema free like Elasticsearch / Mongodb, Read from any file.
- Query Semi-structured or/and nested data (Json, csv, txt etc)
- Provides standard APIs, jdbc, http.
-

https://www.dremio.com/apache-drill-explained/

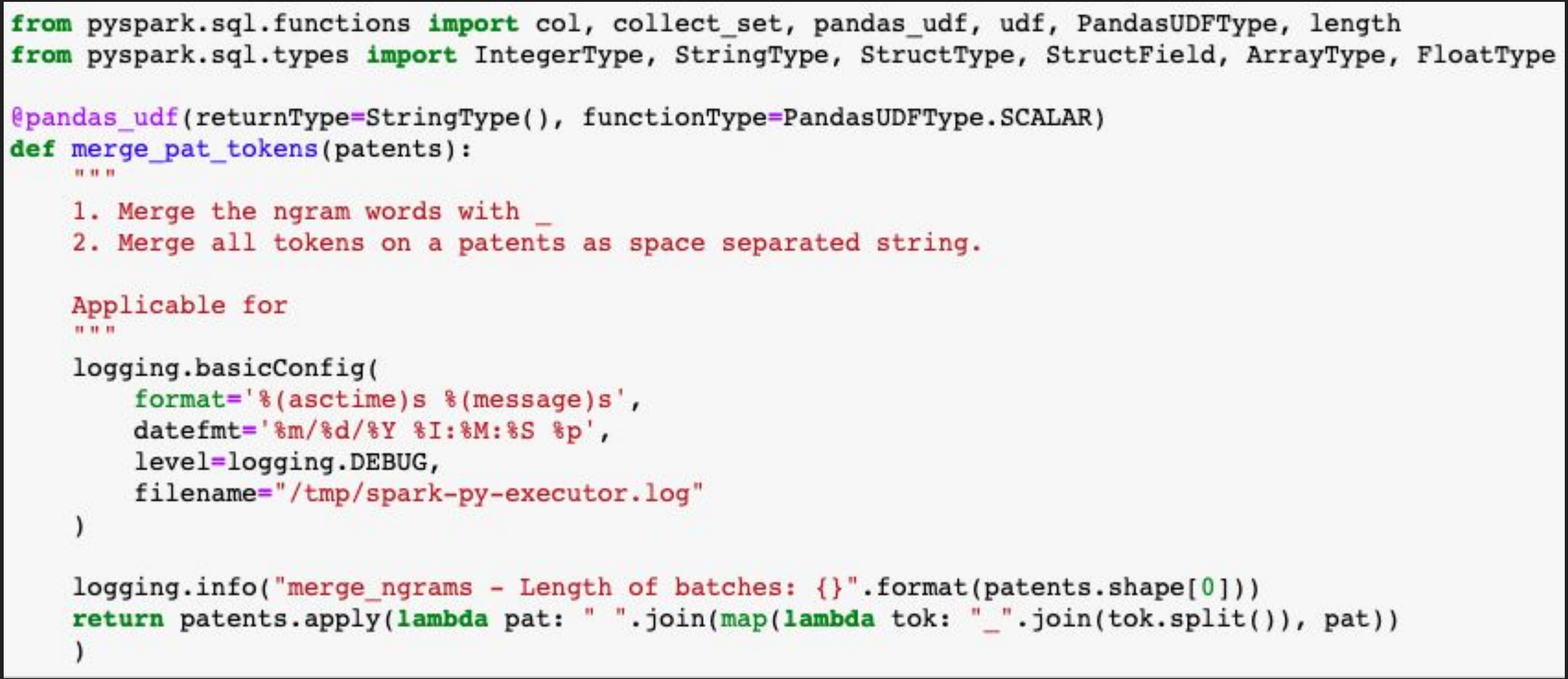
Drill vs Spark

- Spark is general purpose analytic engine, which make uses optimised version MR.
- Spark supports SQL, via its APIs but not fully standard complaint.
- Drill has optimised engine specifically for the query push-down and optimisation.
- Drill specially for SQL related use-cases, Spark has SQL interface but more into general graph computation.
- We can imagine Drill as a BigQuery Database, Spark is not for these kind of use cases.

Workshop

Thank you

Pyspark example



```python
from pyspark.sql.functions import col, collect_set, pandas_udf, udf, PandasUDFType, length
from pyspark.sql.types import IntegerType, StringType, StructType, StructField, ArrayType, FloatType

@pandas_udf(returnType=StringType(), functionType=PandasUDFType.SCALAR)
def merge_pat_tokens(patents):
    """
    1. Merge the ngram words with _
    2. Merge all tokens on a patents as space separated string.

    Applicable for
    """
    logging.basicConfig(
        format='%(asctime)s %(message)s',
        datefmt='%m/%d/%Y %I:%M:%S %p',
        level=logging.DEBUG,
        filename="/tmp/spark-py-executor.log"
    )

    logging.info("merge_ngrams - Length of batches: {}".format(patents.shape[0]))
    return patents.apply(lambda pat: " ".join(map(lambda tok: "_".join(tok.split()), pat))
    )
```