StreamSQL on Spark: Manipulating Streams by "SQL" using Spark

Grace Huang, Jerry Shao
Intel/SSG/Big Data Technology
Contact us: jie.huang@intel.com

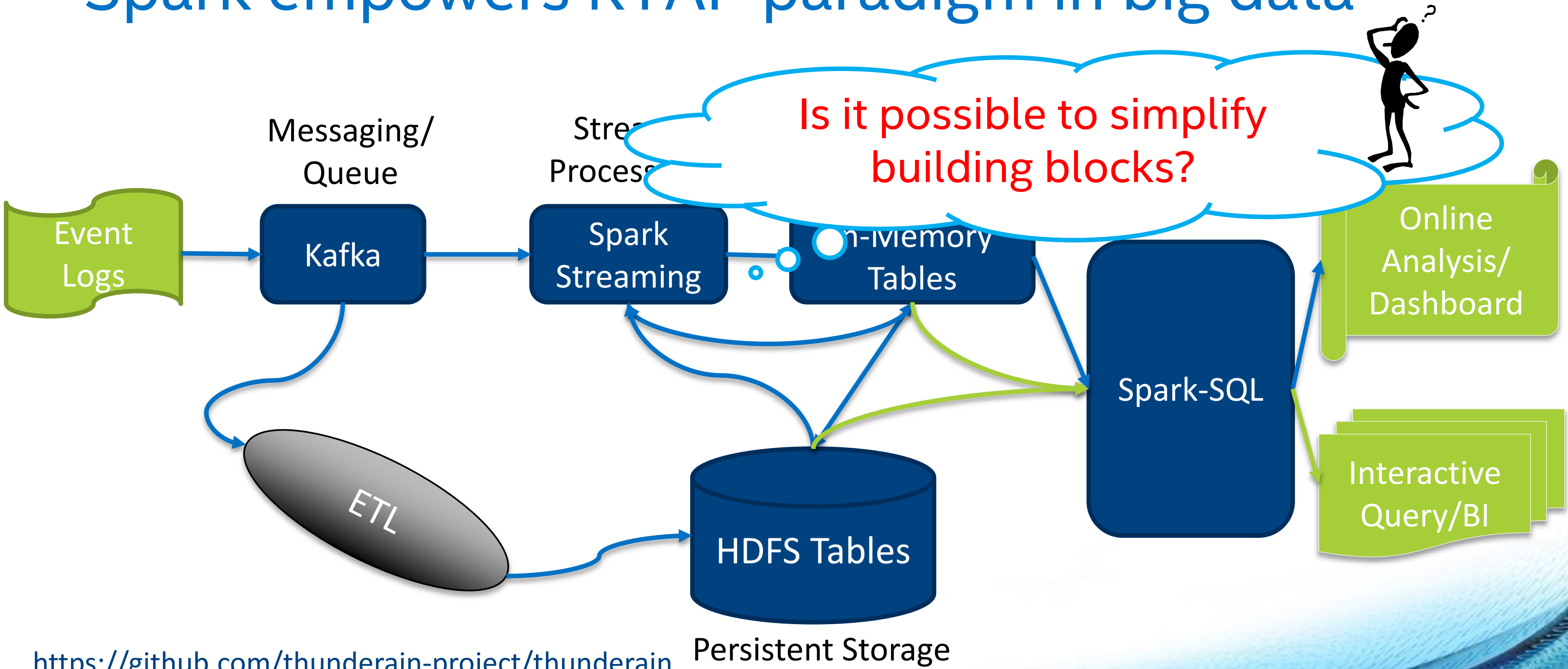
Agenda

- About us
- Problem statement
- What is StreamSQL & its benefit
- Inside POC & HelloWorld
- Next step

About US – Big Data Technology in Intel

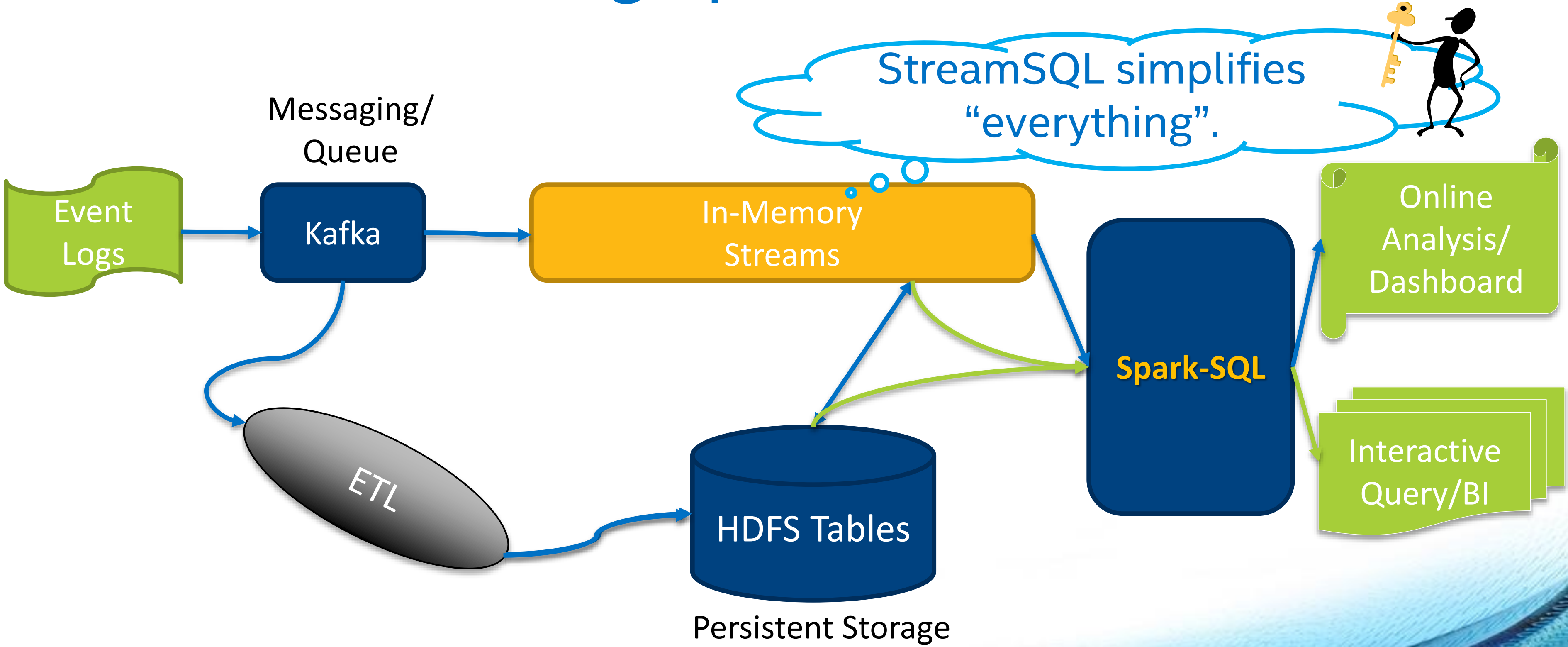
- Long history with Spark & plays key contributions to grow itself and its ecosystem
- Usability & Operability
  - Metrics & monitors in Spark-streaming
  - CLI supports in Spark-sql
  - More advanced functions in Spark-sql
  - Better Tachyon integration
- Performance
  - New pluggable shuffle (with Cloudera) & Pluggable storage
  - Improve scheduler for better locality
  - Join &Aggregation optimization in Spark-sql

Spark empowers RTAP paradigm in big data



https://github.com/thunderain-project/thunderain

Build DSMS using Spark



What is StreamSQL?

Streams ≈ Tables means to …

- To define structures in the stream-in records

```
CREATE TABLE words ( word STRING) STORED AS LOCATION ‘hdfs://…‘;
```

- To manipulate stream data like a table

```
SELECT word, COUNT(*) FROM words;
```

- To drop stream similarly

```
DROP TABLE IF EXISTS words;
```

Streams ≈ Tables means to …

- To define structures in the stream-in records

```
CREATE STREAM words ( word STRING) STORED AS LOCATION ‘kafka://…'
```

- To manipulate stream data like a table

```
SELECT word, COUNT(*) FROM words;
```

- To drop stream similarly

```
DROP STREAM IF EXISTS words;
```

What's StreamSQL?

- DSMS ≈ DBMS
- To manipulate stream-in data like static structured data in database
- To conduct query continuously

To grasp streaming processing like SQL

- Spark-streaming

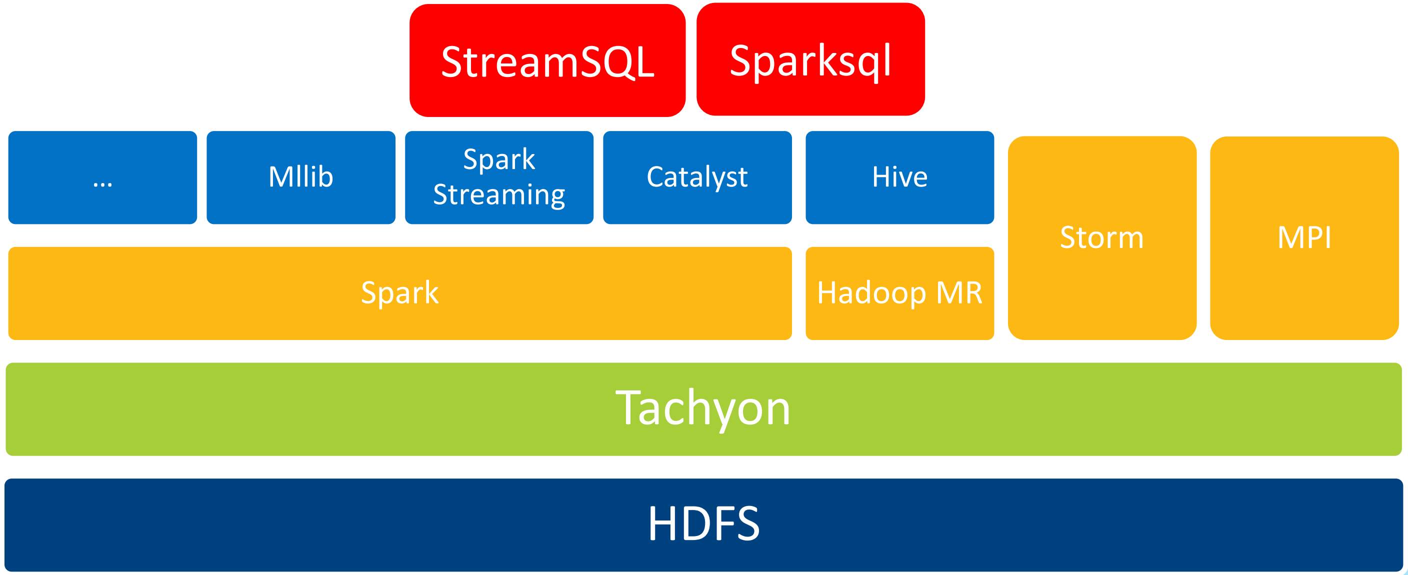
```
val weblog= KafkaUtils.createStream(…)
 .map(_._2)
 .flapMap(_.split(“|”)).map(_._2, _)

val category=
  sc.textFile(…).flatMap(_.split(“|”))

val result
  = weblog.transform(r => r.join(category))
   .map(_._2._2, 1L))
   .reduceByKey(_ + _)

result.print()
```

- StreamSQL

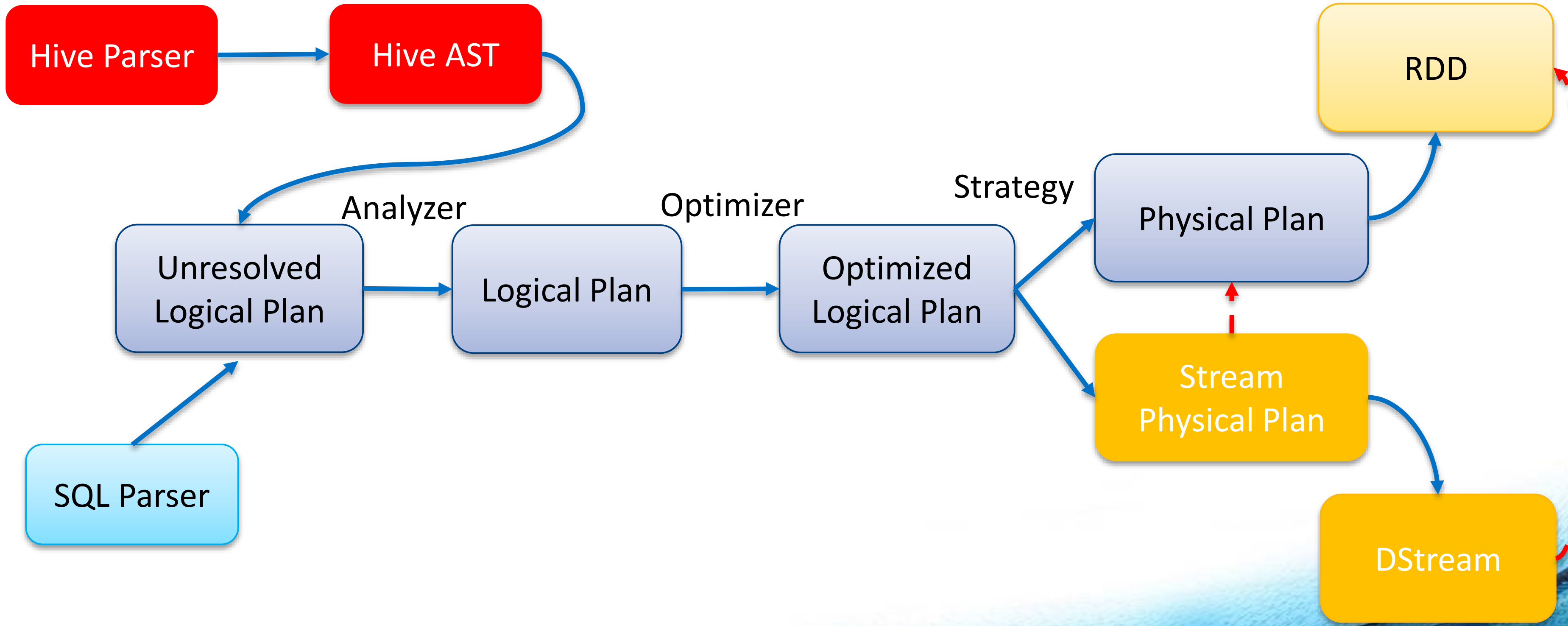
```
CREATE STREAM weblog ( sourcep_ip STRING,
cookie_id INT, item_id INT …) STORED AS LOCATION
'kafka://…'

CREATE TABLE category (item_id INT, category
STRING);

SELECT category, COUNT(*) FROM weblog JOIN
category ON category.item_id = weblog.item_id
GROUP BY category.category;
```

To build StreamSQL using Spark

- The building block on top of Spark Streaming and Catalyst



StreamSQL (POC) Architecture



DDL design (POC)

- Modified Hive parser to support newly introduced Stream syntax
- Shared Hive meta-store for easy integration

- Highly customized Stream Input/Output Format
  - To extract records from streams (equivalent to MR InputFormat)
  - To stream-out with specified write type

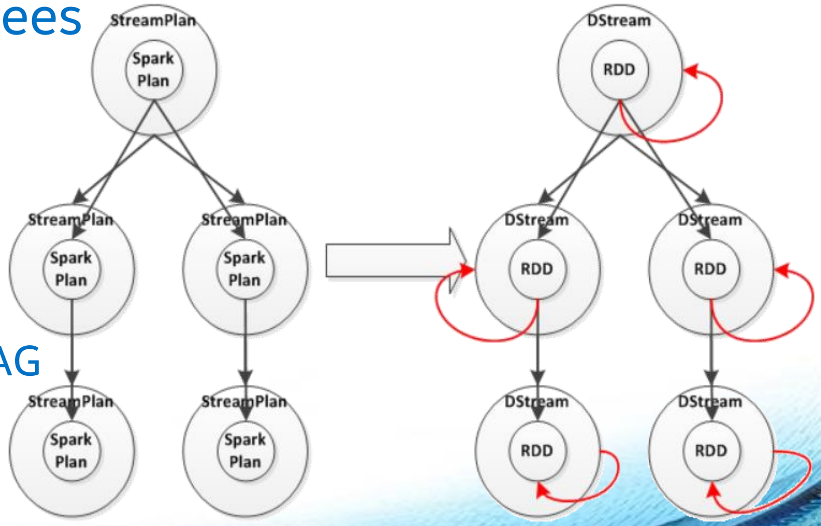
- Re-use Ser/Der to understand data structure

DML design (POC)

- Re-use original LogicalPlan and its analysis rule

- To build two coupled PhyscialPlan trees
  - Re-created StreamPlan
  - RDD-based core SparkPlan

- Upgraded execution workflow
  - Evolve the physical plan into DStream DAG
  - Turn out to be RDD DAG essentially



Software and Services

StreamSQL’s HelloWorld

Feature#1: HQL compatible

- Query snippet for Wordcount demo case

```
streamHiveContext.streamHql(
"""
    CREATE STREAM IF NOT EXISTS words( word STRING)
    ROW FORMAT DELIMITED  FIELDS TERMINATED BY ' '
    STORED AS
       INPUTFORMAT 'org.apache.hadoop.mapred.TextKafkaInputFormat'
       OUTPUTFORMAT 'org.apache.hadoop.mapred.DummyStreamOutputFormat'
       LOCATION 'kafka://localhost:2181/topic=test/group=demo'
  """)

streamHql("""SELECT word, COUNT(*) FROM words GROUPBY word""").print()
```

Feature#2: Mutual operations between streams & tables (1)

- Register Table

```
case class Category(item_id: Int,  category: String)

val category = sc.textFile(“category").map(_.split(“|"))
  .map(c => Category(c(0).toInt, c(1))
sqlContext.createSchemaRDD(category).registerAsTable("category")
```

- Register Stream

```
case class Weblog (source_ip: String,  item_id: Int, cookie: Int)

val weblog: DStream[Weblog] = ssc.socketTextStream(serverIP,serverPort)
   .map(_.split(“|"))
   .map(log => Weblog (log(0), log(1).toInt, log(2).toInt))
weblog.registerAsStream(“weblog")
```

Feature#2: Mutual operations between streams & tables (2)

- Query snippet for category visits

```
val query = sql("SELECT category, COUNT(*) FROM weblog JOIN
category ON category. item_id = weblog.item_id GROUP BY
category.category;")
```

Feature#3: LINQ-like Scala DSL

- Query snippet for visits in certain categories

```
val query = sql("SELECT category, COUNT(*) FROM weblog JOIN
category ON category. item_id = weblog.item_id GROUP BY
category.category;")

val visits = query.where(‘category === worldcup’
    || ’category === sports’
    || ‘category === family’).toDtream
```

Feature#4: Spark stack's integration

- Query snippet for filtering visits in certain categories

```
val query = sql("SELECT category, COUNT(*) FROM weblog JOIN
category ON category. item_id = weblog.item_id GROUP BY
category.category;")

val visits = query.where(‘category === worldcup’
    || ’category === sports’
    || ‘category === family’).toDtream

visits.filter(_._2 > 1000).map( c => “Category: ” + c._1 + “,
Visit Counts: ” + c._2).print()
```

Future

- Time-based window function
- More generic physical plan design (rule-based)
- Enrich DDL operations
    - E.g., CREATE STREAM AS TABLE or STREAM
- Support more streaming source (Flume)
- CLI for Catalyst and StreamSQL
- …

Summary

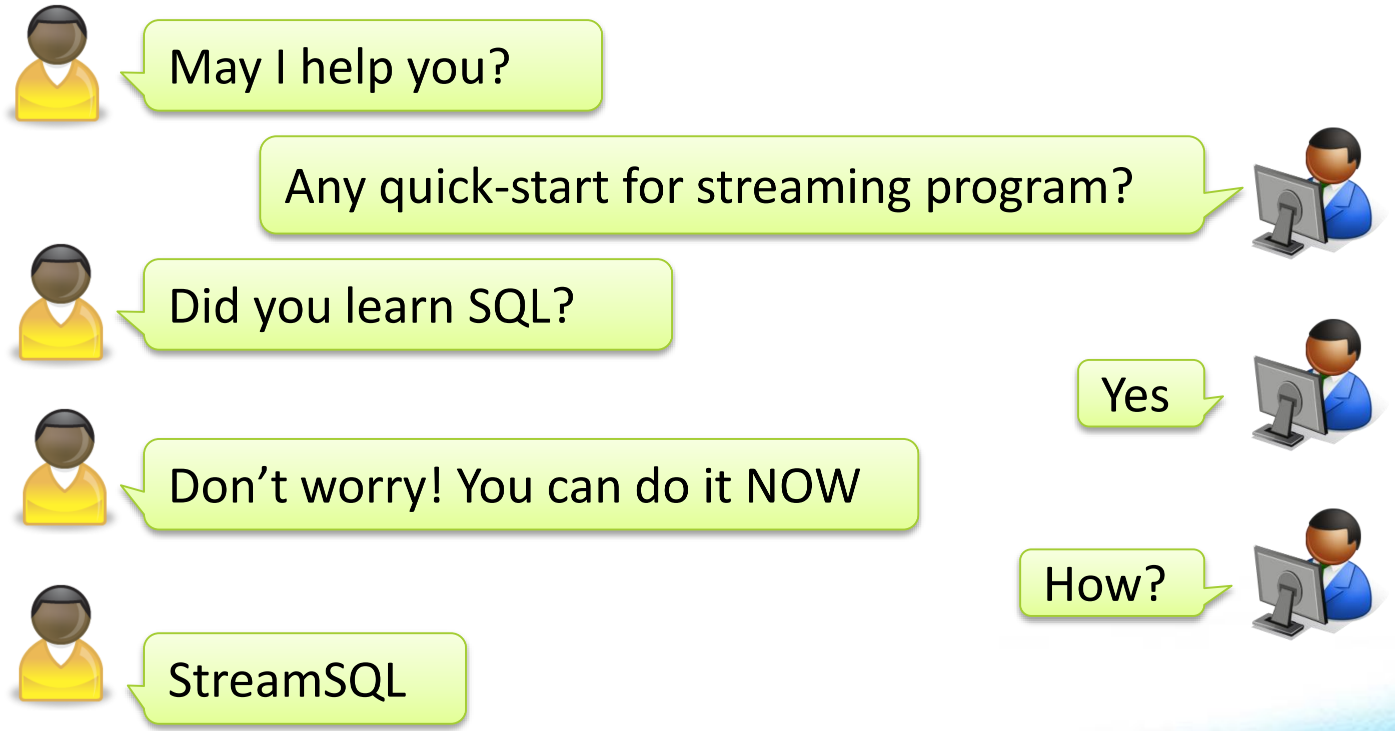
- To mine and explore data in a real-time, streaming manner
- StreamSQL to manipulate ***STREAM*** by SQL just like ***TABLE***
- Key supportive features in POC using Spark
  - HQL compatible & shared Hive metastore
  - Mutual operations between streams & tables
  - LINQ-like Scala DSL
  - Spark stack's integration
- StreamSQL stands on the shoulders of Spark and serves Spark with easy usage

Let's make both Better altogether!

Notices and Disclaimers

intel®

Why StreamSQL


May I help you?
Any quick-start for streaming program?
Did you learn SQL?
Yes
Don’t worry! You can do it NOW
How?
StreamSQL