DATABRICKS

Spark SQL 1.2 Improvements and New Features

Cheng Lian — Spark Meetup Beijing, Dec 2014

Agenda

- External data source API
- Enhanced in-memory columnar storage
- Enhanced Parquet support
- Enhanced Hive support
- Misc.
- Next steps

Agenda

- > External data source API
- > Enhanced in-memory columnar storage
- > Enhanced Parquet support
- > Enhanced Hive support
- > Misc.
- > Next steps

DATABRICKS

External data source API

- New @DeveloperApi introduced in 1.2
- Define new input sources for Spark SQL
  - JSON, Avro, CSV, ...
  - Parquet, ORC, ...
  - JDBC, C*, HBase, …

External data source API

- Mixing multiple types of data sources easily:

  -
    ```
    CREATE TEMPORARY TABLE jtable
    USING org.apache.spark.sql.json
    (path = "...");
    ```

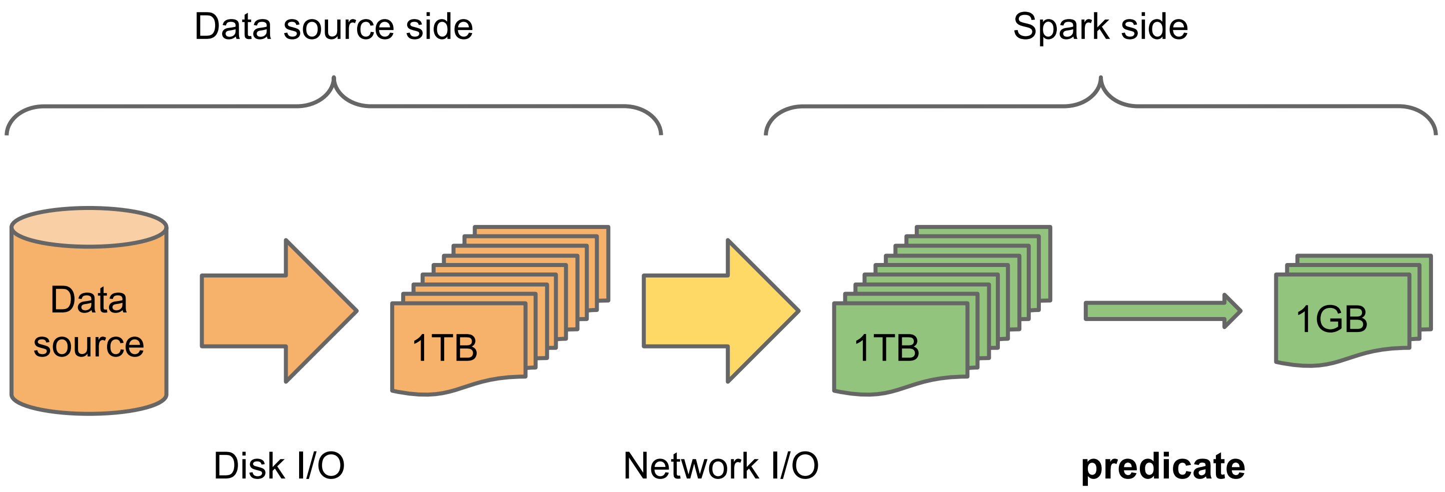
  -
    ```
    CREATE TEMPORARY TABLE ptable
    USING org.apache.spark.sql.parquet
    (path = "...");
    ```

  -
    ```
    SELECT jtable.key, ptable.value
    FROM jtable JOIN ptable
    ON jtable.key = ptable.key;
    ```

External data source API

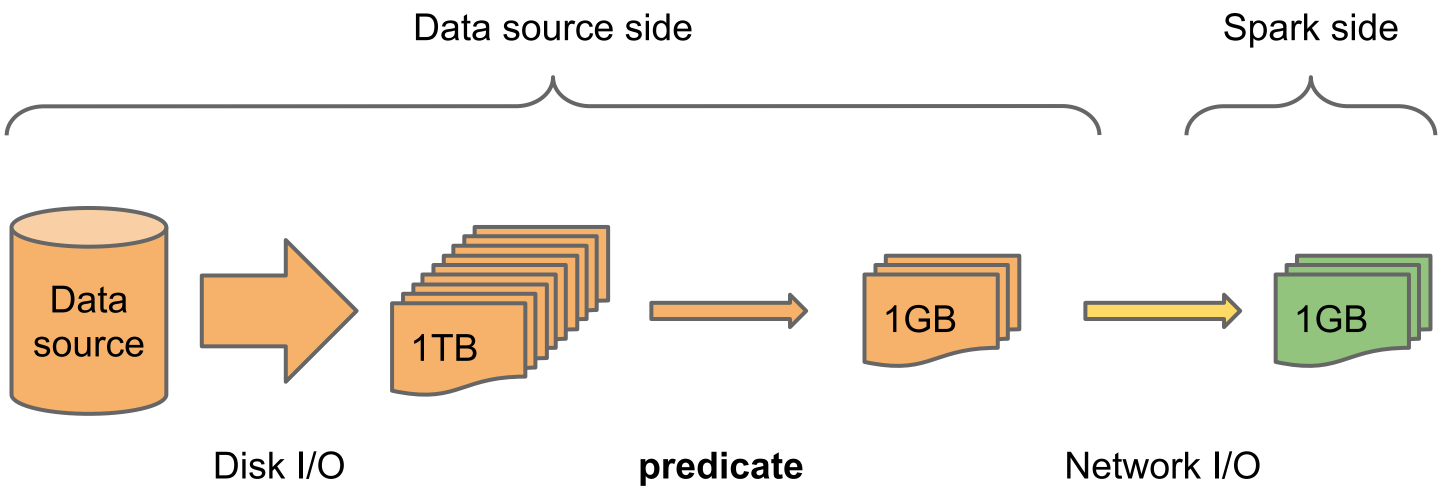
- Supports data source specific optimizations
  - Column pruning
  - Pushing predicates to datasources (filter pushdown)
  - Partition pruning (coming soon)

External data source API



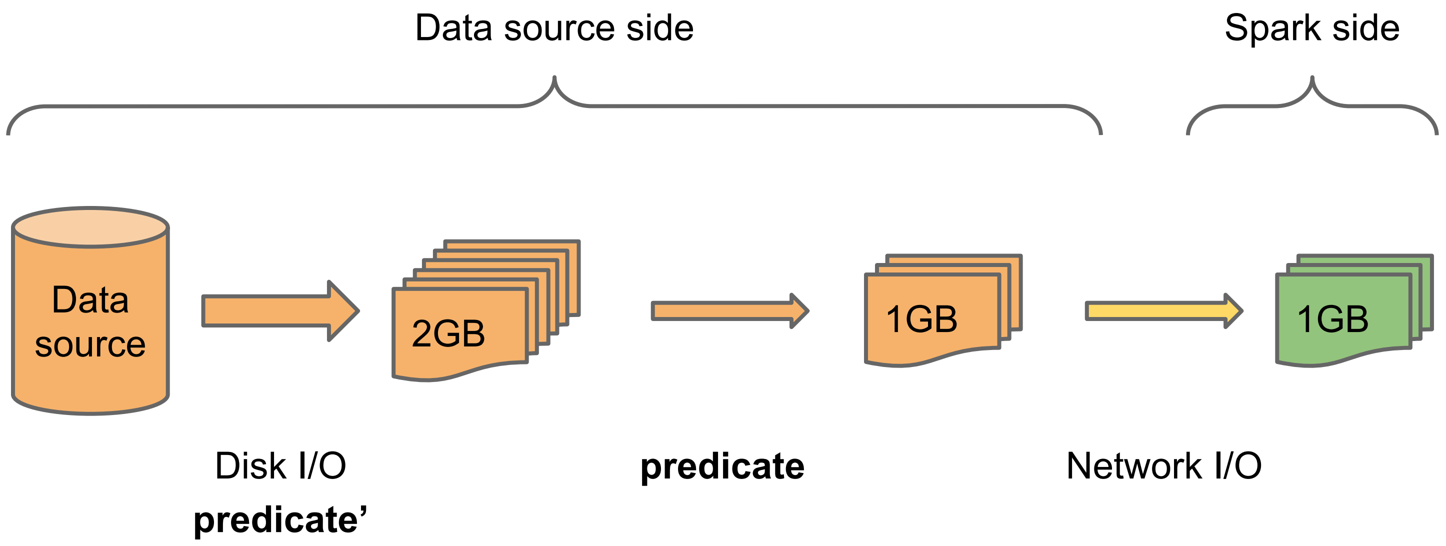
Before...

External data source API



After...

External data source API



**With smart data formats like Parquet and ORC,
wen can even achieve this...**

External data source API

```
SELECT Name WHERE ID < 3
```

ID	Name	Age
1	Alice	21
2	Bob	30
3	Cart	28

Column pruning

ID	Name	Age
1	Alice	21
2	Bob	30
3	Cart	28

Predicate pushdown

DATABRICKS

External data source API

- Existing data sources
  - Simple formats
    - JSON, Avro, CSV
  - Smart formats with column pruning and filter pushdown
    - Parquet
    - ORC (PR #2576 by @scwf)

External data source API

- Roadmap
  - First class partitioning support with partition pruning
  - Data sink (insertion) API
  - Making Hive as an external data source

Agenda

> External data source API
> **Enhanced in-memory columnar storage**
> Enhanced Parquet support
> Enhanced Hive support
> Misc.
> Next steps

In-memory columnar storage

- Unified caching semantics
  - `SchemaRDD.cache()` now uses in-memory columnar storage (finally!!!)
  - `SQLContext.cacheTable("tbl")` is now eager by default
  - `CACHE [LAZY] TABLE tbl [AS SELECT …]`
- Don't need to trigger cache materialization manually anymore

  -
    ```
    CACHE TABLE src;
    ```
    ~~`SELECT COUNT(*) FROM src;`~~

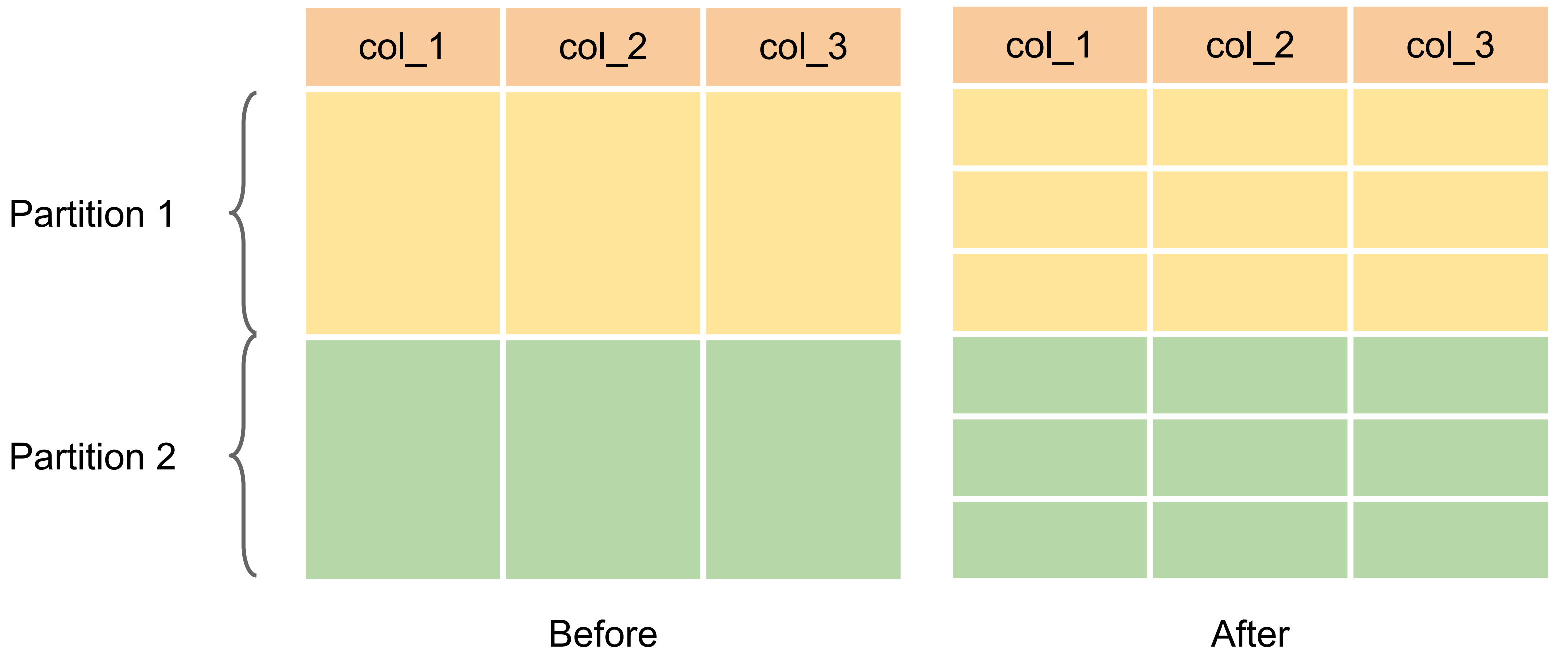
In-memory columnar storage

- Query plan based cache sharing
  - Cached queries with exactly the same result share the same underlying cache
    - `CACHE TABLE t1 AS SELECT * FROM src;`
    - `CACHE TABLE src;`

In-memory columnar storage

- Cleaner memory footprint
  - Eliminated most boxing costs when building and accessing column buffers
  - Introduced batched column buffer builder to avoid OOM when caching large tables

In-memory columnar storage



DATABRICKS

In-memory columnar storage

- Table statistics
  - Predicate pushdown
    - Faster table scan
  - Auto broadcast join
    - Faster table join

Agenda

> External data source API
> Enhanced in-memory columnar storage
> **Enhanced Parquet support**
> Enhanced Hive support
> Misc.
> Next steps

Enhanced Parquet support

- Parquet filter2 API based filter pushdown
  - Simple predicates can be pushed down
    - Comparisons: attr cmp literal, literal cmp attr
    - Logical: AND, OR, NOT
  - Enables filtering entire row groups or pages of records without touching them
  - Unfortunately disabled by default because of PARQUET-136, expected to be re-enabled in 1.2.1

Enhanced Parquet support

- Ported to the data source API
  - With simple partitioning and partition skipping support
  - Aims to replace the old Parquet implementation

Agenda

> External data source API
> Enhanced in-memory columnar storage
> Enhanced Parquet support
> Enhanced Hive support
> Misc.
> Next steps

Enhanced Hive support

- Hive 0.13.1 support
- Dynamic partitioning support
  - Thanks to @baishuo from AsiaInfo!
- View support

Agenda

- External data source API
- Enhanced in-memory columnar storage
- Enhanced Parquet support
- Enhanced Hive support
- Misc.
- Next steps

DATABRICKS

Miscellaneous

- Fixed-precision decimal type support
- Date type support
- UDT support (mostly for Spark ML)
- UDF DSL

  -
    ```
    val triple = (n: Int) => n * 3
    table("src")
      .select(triple.call('key) as 'k, 'value)
      .where('key > 10)
    ```

Agenda

> External data source API
> Enhanced in-memory columnar storage
> Enhanced Parquet support
> Enhanced Hive support
> Misc.
> Next steps

DATABRICKS

Next steps

- External data source API
  - Data sink API
  - First class partitioning support
- Better way to support multiple Hive versions
- Window function and analytics features
- ...

DATABRICKS

THANK YOU!