Cost-based query optimization in Apache Hive

2014 HADOOP SUMMIT

Julian Hyde

June 4th, 2014

About me

Julian Hyde

Architect at Hortonworks

Open source:

- Founder & lead, Apache Optiq (query optimization framework)
- Founder & lead, Pentaho Mondrian (analysis engine)
- Committer, Apache Drill
- Contributor, Apache Hive
- Contributor, Cascading Lingual (SQL interface to Cascading)

Past:

- SQLstream (streaming SQL)
- Broadbase (data warehouse)
- Oracle (SQL kernel development)

(Thanks to
John Pullokkaran,
Harish Butani
for presentation content
and actually doing the work.)

Apache Hive

The original "SQL on Hadoop"

Undergoing extensive renovation

- Tez execution engine
- YARN execution environment
- Vectorized data representation
- Column-oriented data storage (ORC)
- ACID transactions
- SQL standards compliance
- SQL authorization model
- **Cost-based query optimization (CBO)**

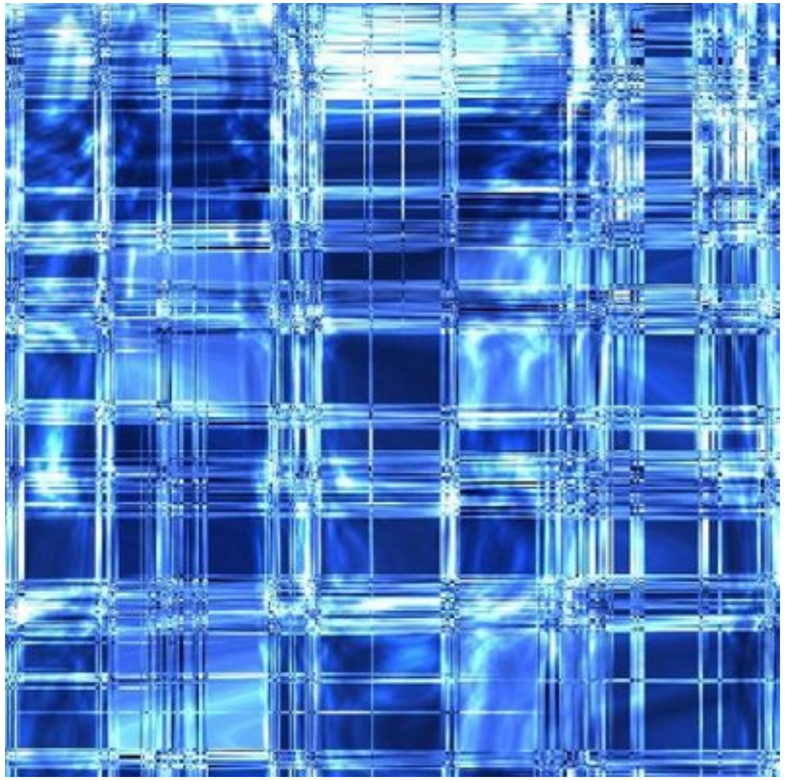
"Stinger Initiative"

What? Why? How? When?

Incremental cutover to cost-based optimization

Release	Date	Remarks
Apache Hive 0.12	October 2013	• Rule-based Optimizations • No join reordering • Main optimizations: predicate push-down & partition pruning • Semantic info and operator tree tightly coupled
Apache Hive 0.13	April 2014	“Old-style” JOIN & push-down conditions: *… FROM t1, t2 WHERE …* CBO just missed the deadline ☹
HDP 2.1	April 2014	Cost-based ordering of joins • HIVE-6439 “Introduce CBO step in Semantic Analyzer” • HIVE-5775 “Introduce Cost Based Optimizer in Hive”
Apache Hive 0.14	?	CBO patches More rework of internals More cost-based features…

Apache Optiq (incubating)



Apache Optiq

Apache incubator project since May, 2014

Query planning framework

- Extensible
- Usable standalone (JDBC) or embedded

Adoption

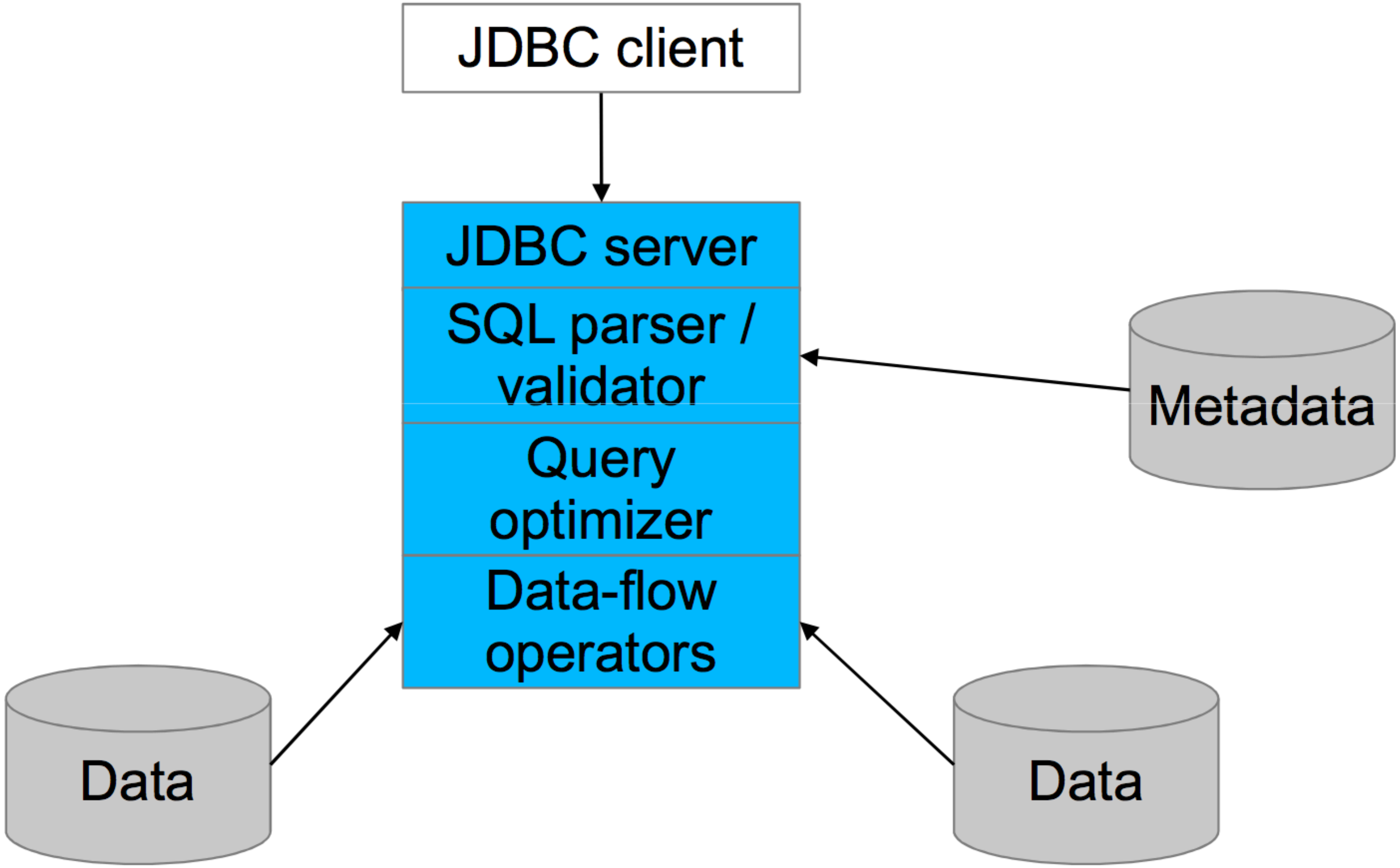
Lingual – SQL interface to Cascading

Apache Drill

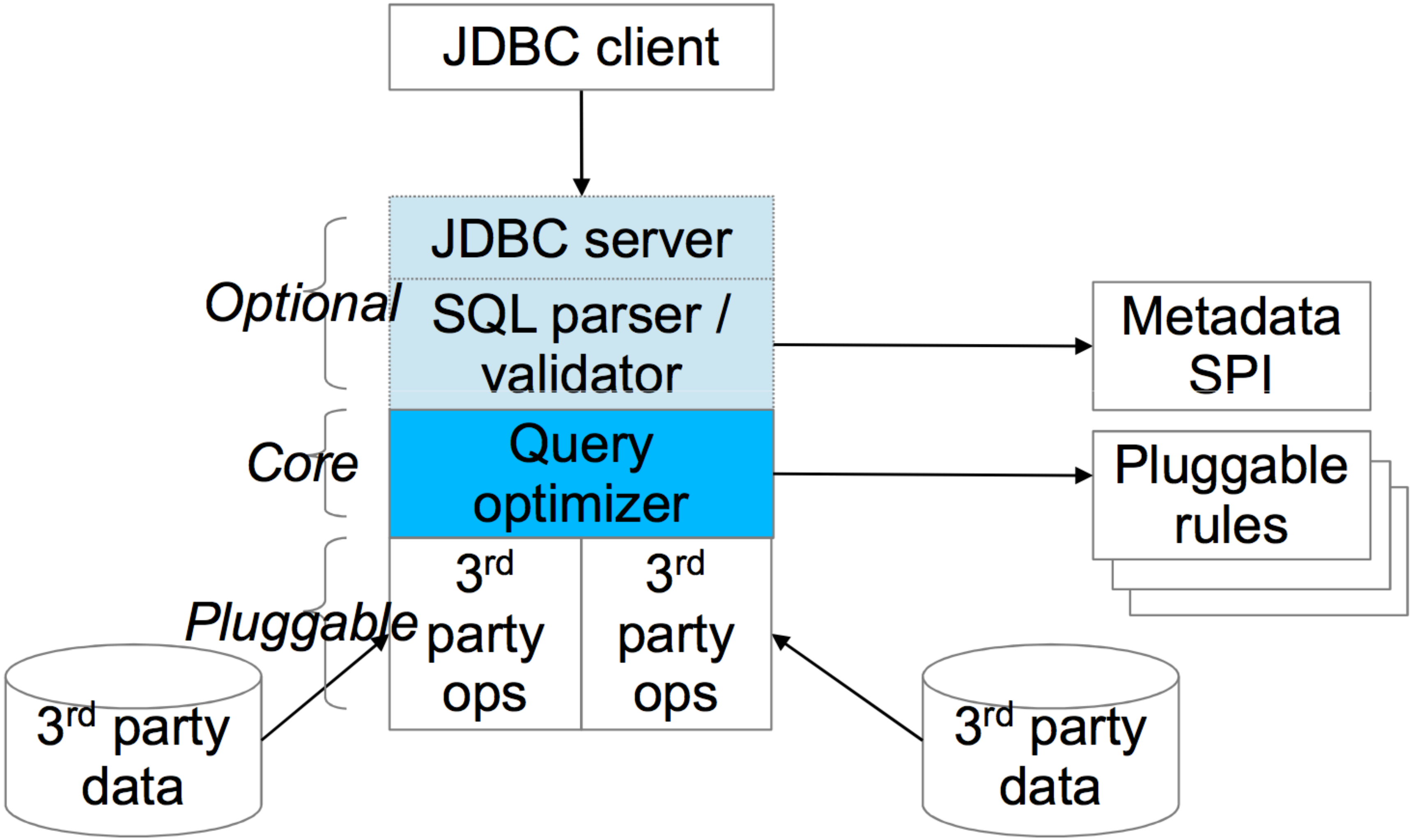
Apache Hive

Adapters: Splunk, Spark, MongoDB, JDBC, CSV, Web tables, In-memory data

Conventional DB architecture



Optiq architecture



Optiq – APIs and SPIs

Relational algebra

RelNode (operator)

- TableScan
- Filter
- Project
- Union
- Aggregate
- …

RelDataType (type)

RexNode (expression)

RelTrait (physical property)

- RelConvention (calling-convention)
- RelCollation (sortedness)
- TBD (bucketedness/distribution)

Transformation rules

RelOptRule

- MergeFilterRule
- PushAggregateThroughUnionRule
- RemoveCorrelationForScalarProjectRule
- 100+ more

Unification (materialized view)

Column trimming

Cost, statistics

RelOptCost

RelOptCostFactory

RelMetadataProvider

- RelMdColumnUniquensss
- RelMdDistinctRowCount
- RelMdSelectivity

SQL parser

SqlNode

SqlParser

SqlValidator

Metadata

Schema

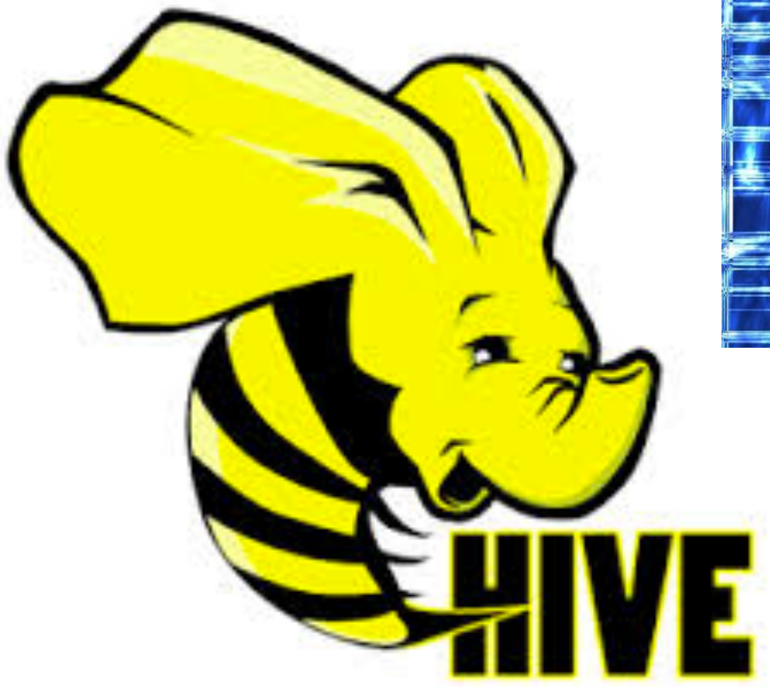
Table

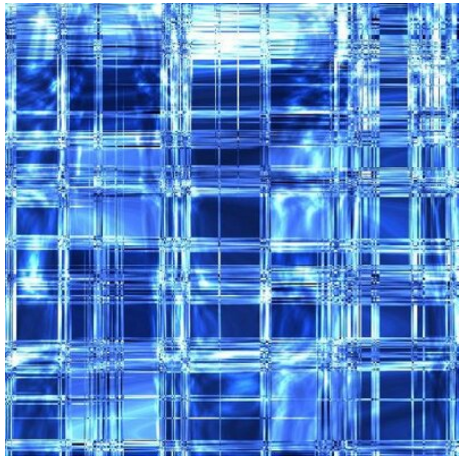
Function

- TableFunction
- TableMacro

JDBC driver

Now… back to Hive





CBO in Hive

Why cost-based optimization?

Ease of Use – Join Reordering

View Chaining

Ad hoc queries involving multiple views

Enables BI Tools as front ends to Hive

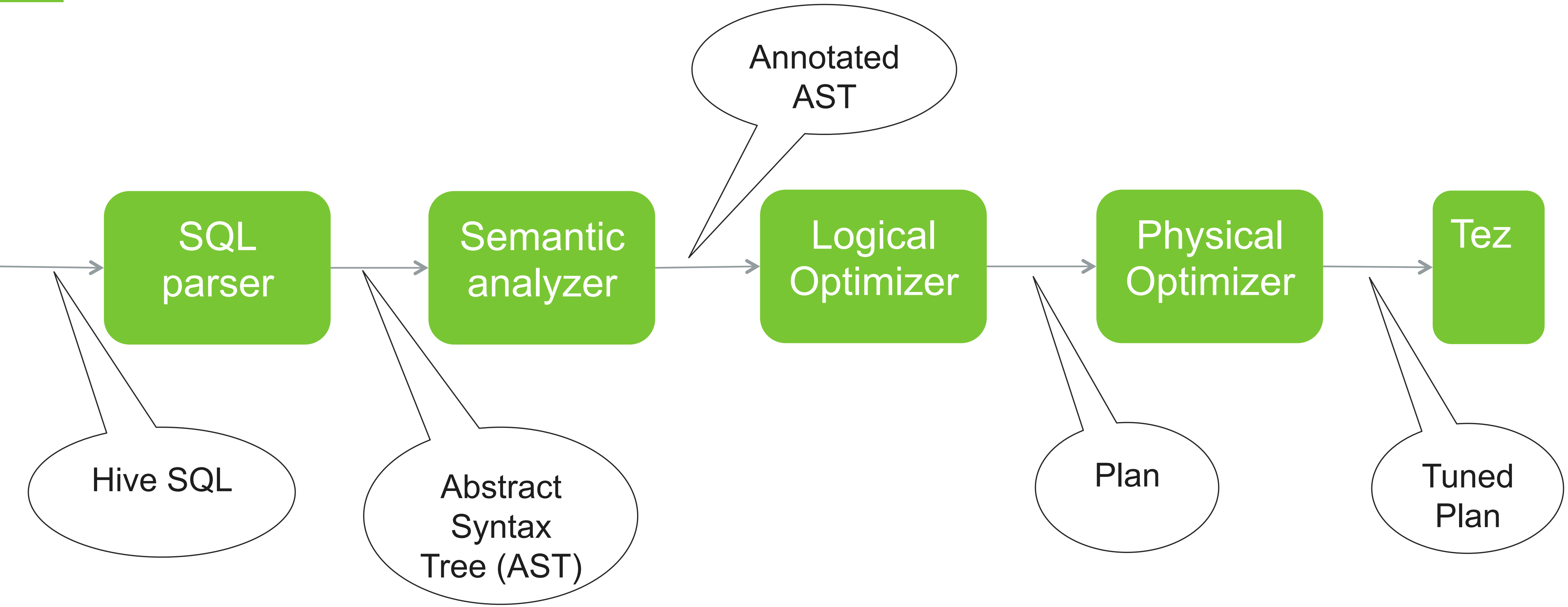
First version

Modest goal

Concrete results

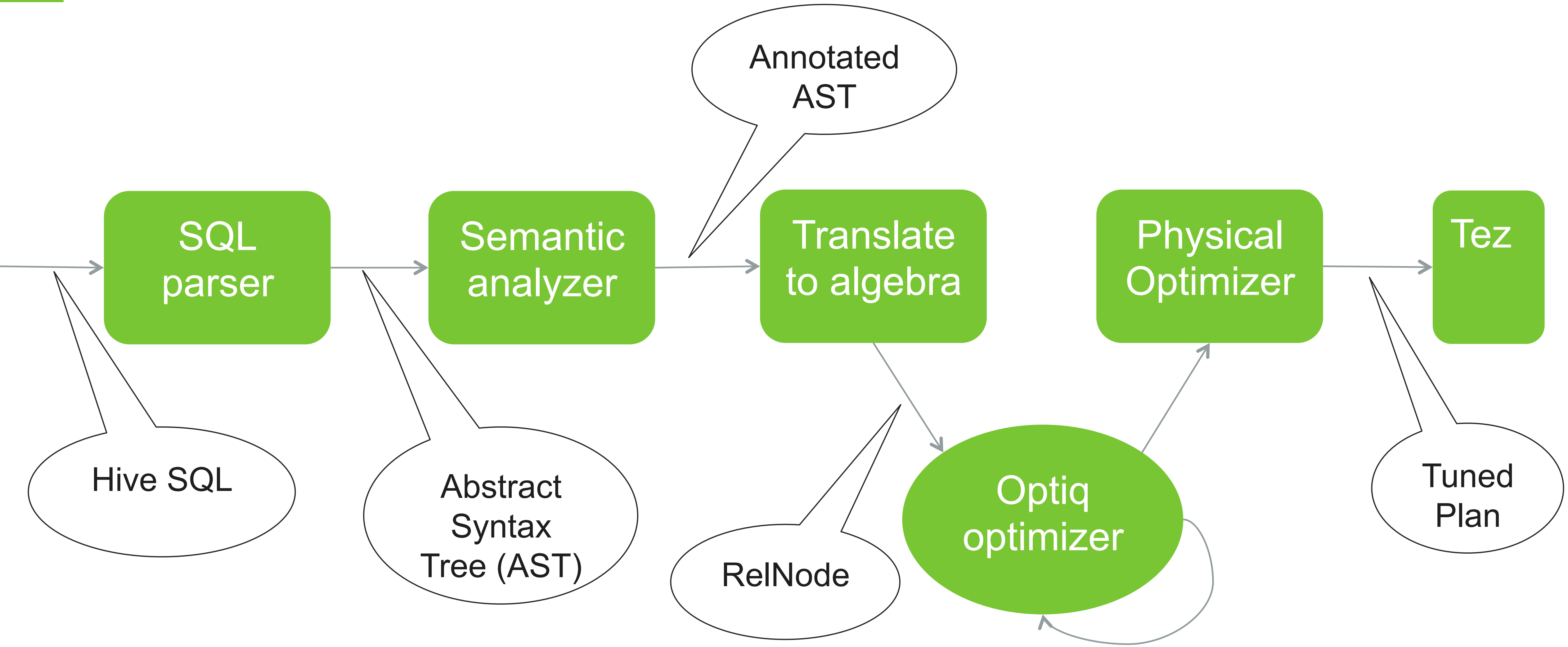
Join re-ordering

Query preparation – Hive 0.13



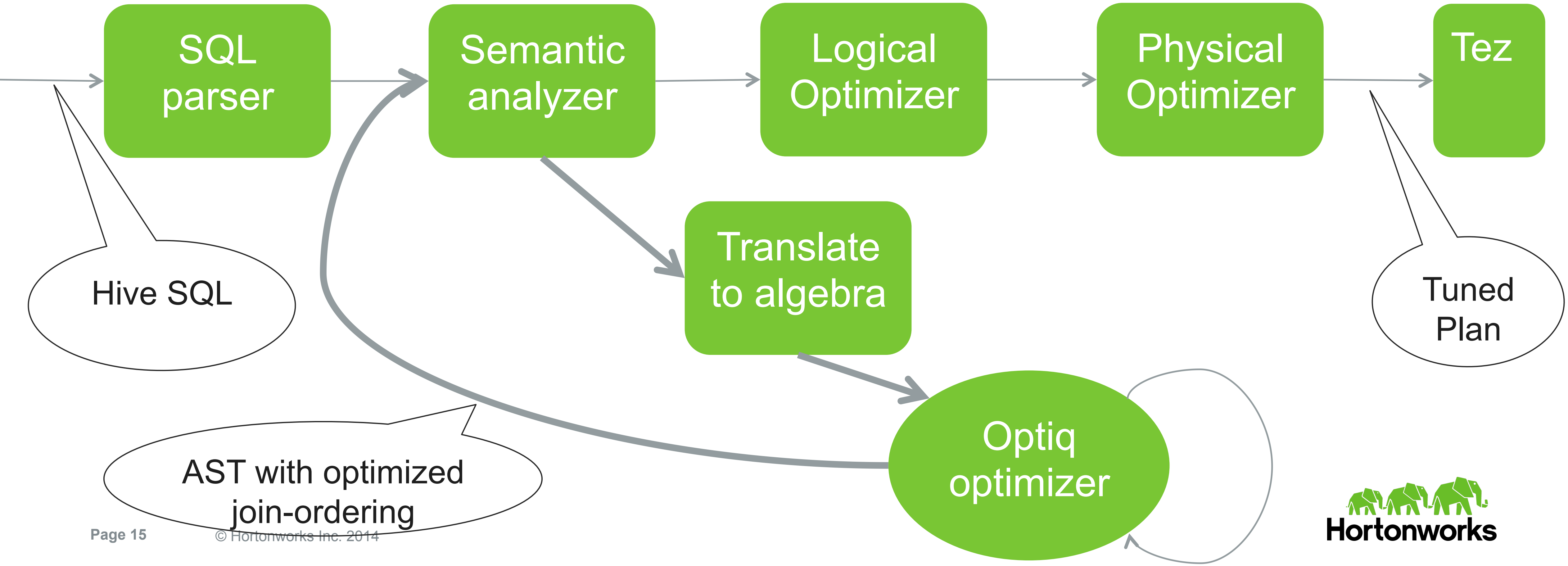
Hortonworks

Query preparation – full CBO



Hortonworks

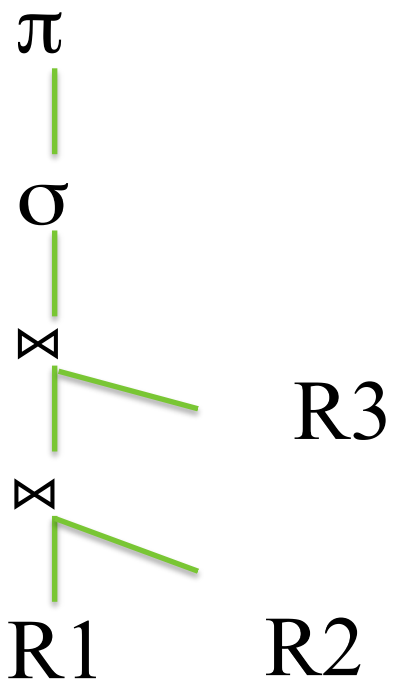
Query preparation – initial CBO



Hortonworks

Query Execution – The basics

```
SELECT R1.x
FROM R1
  JOIN  R2 ON R1.x = R2.x
JOIN R3 on R1.x = R3.x AND R2.x = R3.x
WHERE R1.z > 10;
```



π

σ

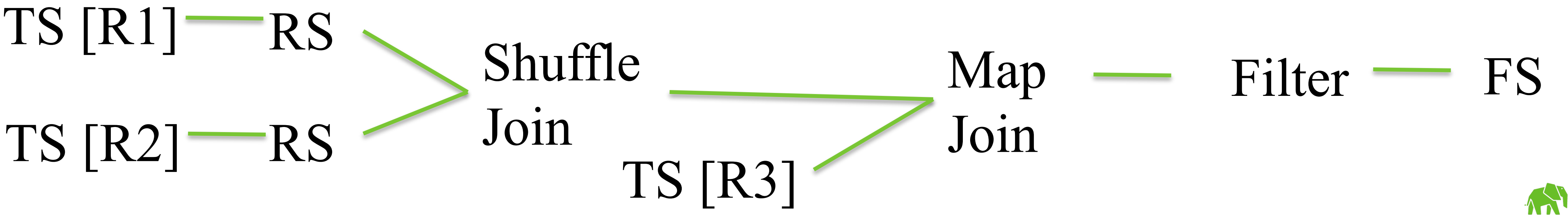
⋈

R3

⋈

R1

R2



TS [R1]

RS

TS [R2]

RS

Shuffle Join

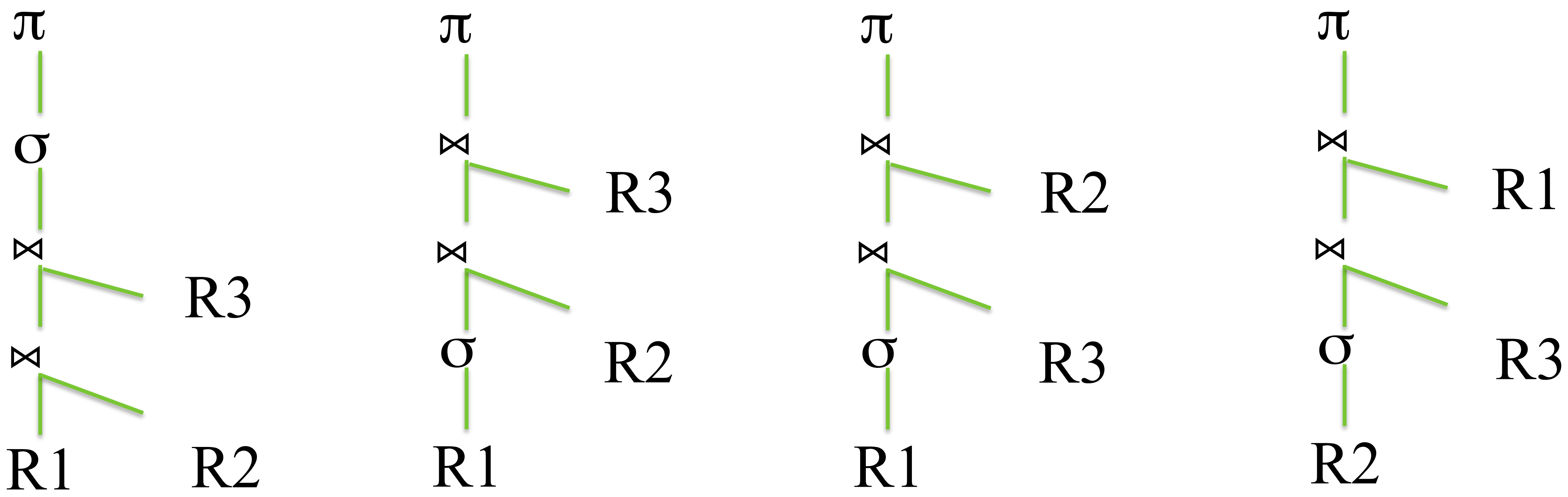
TS [R3]

Map Join

Filter

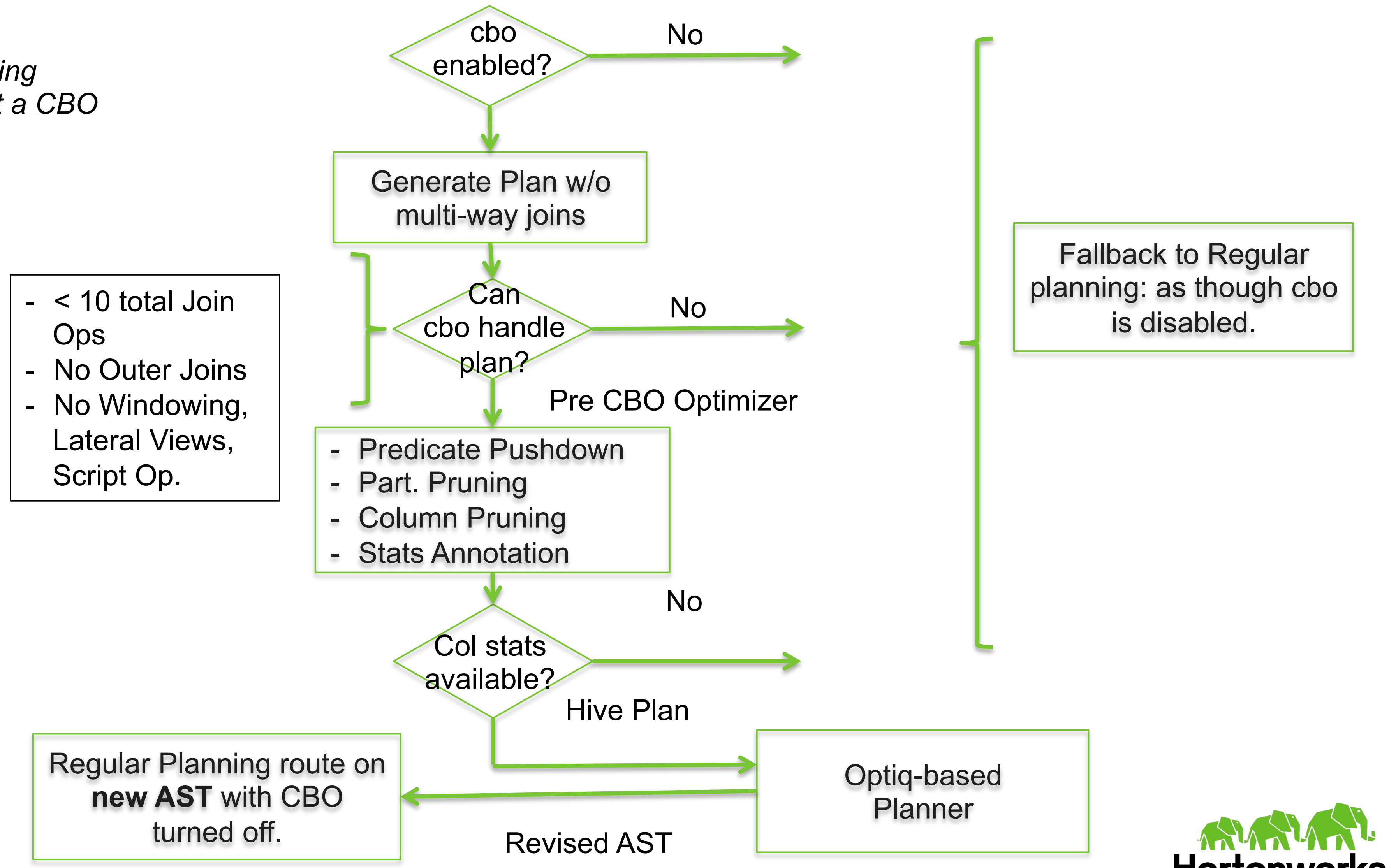
FS

Query Optimization – Rule Based vs. Cost Based



Introduction of CBO into Hive Planning

Series of gating factors to get a CBO Plan.



cbo enabled?

No

Generate Plan w/o multi-way joins

- < 10 total Join Ops
- No Outer Joins
- No Windowing, Lateral Views, Script Op.

Can cbo handle plan?

No

Pre CBO Optimizer

- Predicate Pushdown
- Part. Pruning
- Column Pruning
- Stats Annotation

No

Col stats available?

Hive Plan

Fallback to Regular planning: as though cbo is disabled.

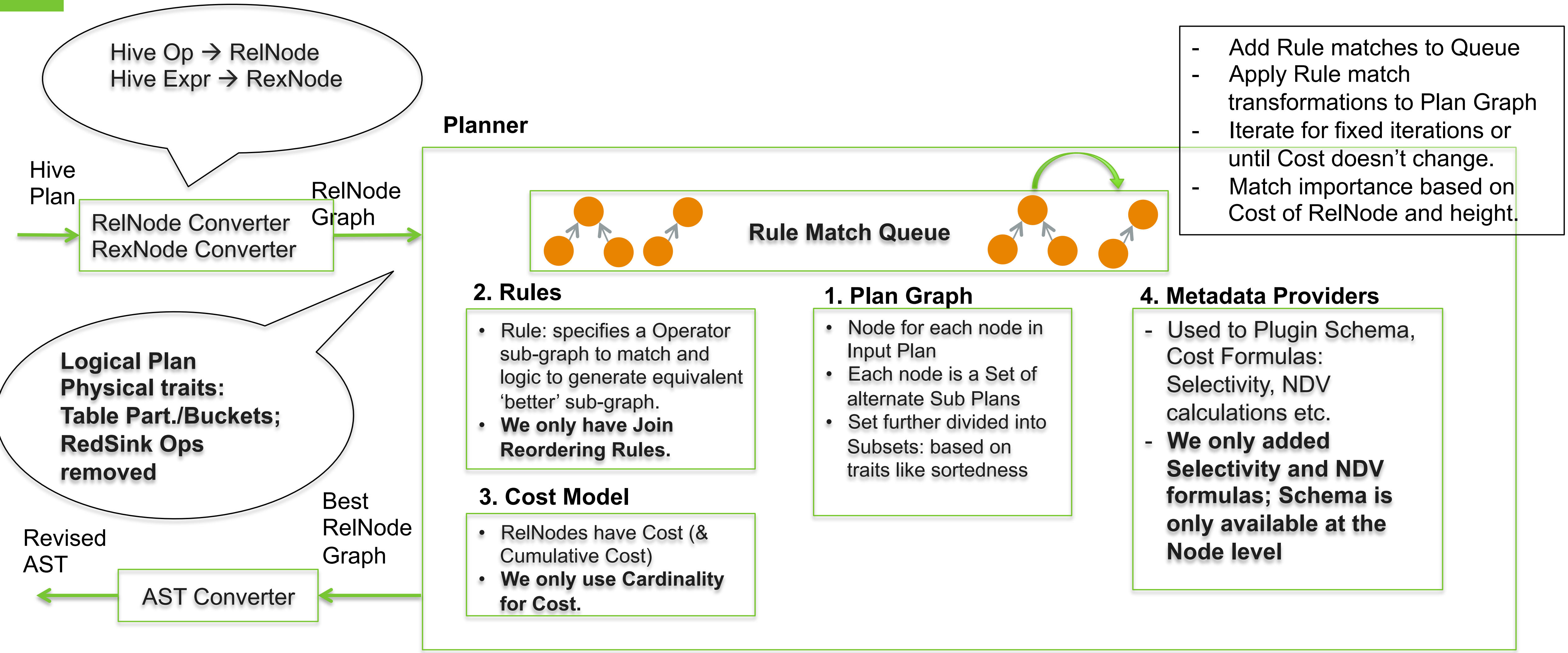
Optiq-based Planner

Revised AST

Regular Planning route on **new AST** with CBO turned off.

Hortonworks

Optiq Planner Process



Hive Op → RelNode
Hive Expr → RexNode

Hive Plan

RelNode Converter
RexNode Converter

RelNode Graph

Logical Plan
Physical traits:
Table Part./Buckets;
RedSink Ops removed

Revised AST

AST Converter

Best RelNode Graph

Planner

Rule Match Queue

- Add Rule matches to Queue
- Apply Rule match transformations to Plan Graph
- Iterate for fixed iterations or until Cost doesn't change.
- Match importance based on Cost of RelNode and height.

2. Rules

- Rule: specifies a Operator sub-graph to match and logic to generate equivalent 'better' sub-graph.
- **We only have Join Reordering Rules.**

1. Plan Graph

- Node for each node in Input Plan
- Each node is a Set of alternate Sub Plans
- Set further divided into Subsets: based on traits like sortedness

4. Metadata Providers

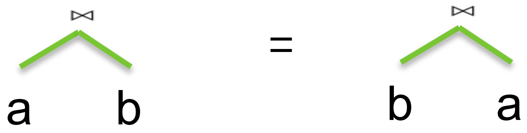
- Used to Plugin Schema, Cost Formulas: Selectivity, NDV calculations etc.
- **We only added Selectivity and NDV formulas; Schema is only available at the Node level**

3. Cost Model

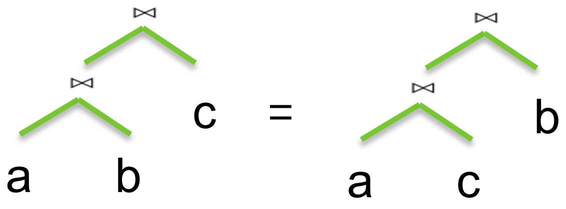
- RelNodes have Cost (& Cumulative Cost)
- **We only use Cardinality for Cost.**

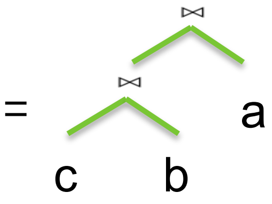
Join Reordering Rules

1. Swap Join Rule



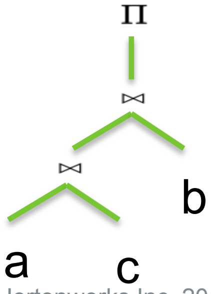
2. Push Join Through Join Rule



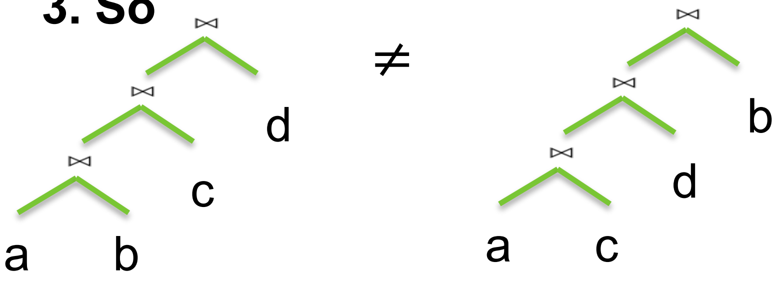


but is really:

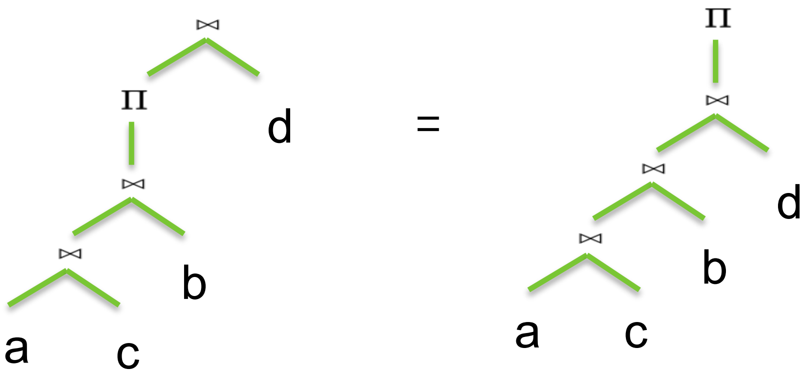
Optiq schema is position based



3. So



4. Pull Up Project above Join



Added bonus
Join permutations across sub-query blocks

5. Merge Projects

Summary

Join re-ordering

Join cardinality is used for cost

All other operators are assumed to have tiny cost

Cardinality of filter, join, group-by is based on selectivity

Selectivity is computed based on number-of-distinct-values (NDV)

Table Stats and Column stats are required

Current limitations

Only supports: filter, inner join, group-by, project, order-by, limit

Not all UDFs

Does not attempt all join permutations (e.g. bushy trees; 10-way joins or more)

May not work well for Bucket, SMB & Skew Joins

TPC-DS Query 50

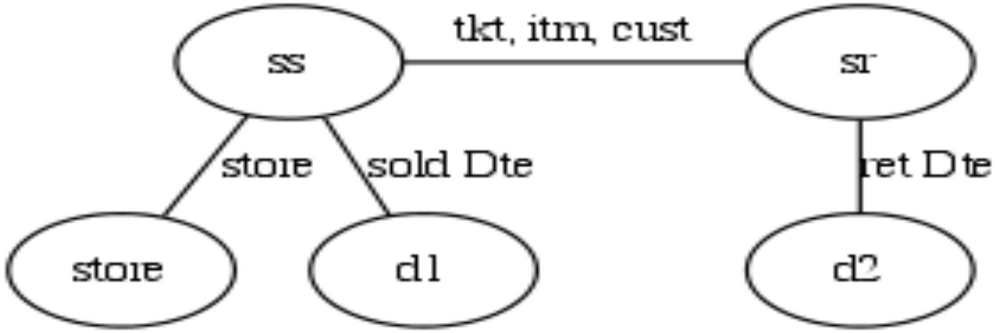
Joins Store Sales, and Store Returns fact tables.

Each of the fact tables are independently restricted by date.

Analysis at Store grain, so this dimension also joined in.

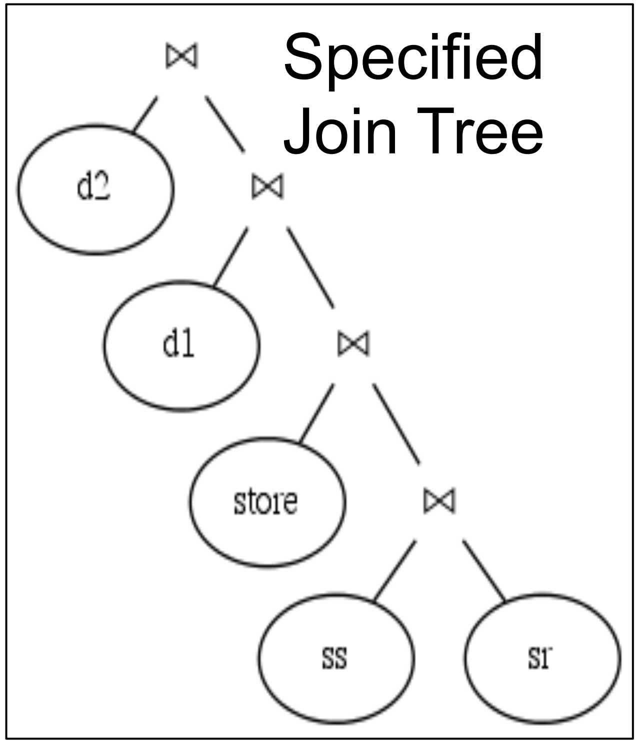
As specified Query starts by joining the 2 Fact tables.

Join Graph

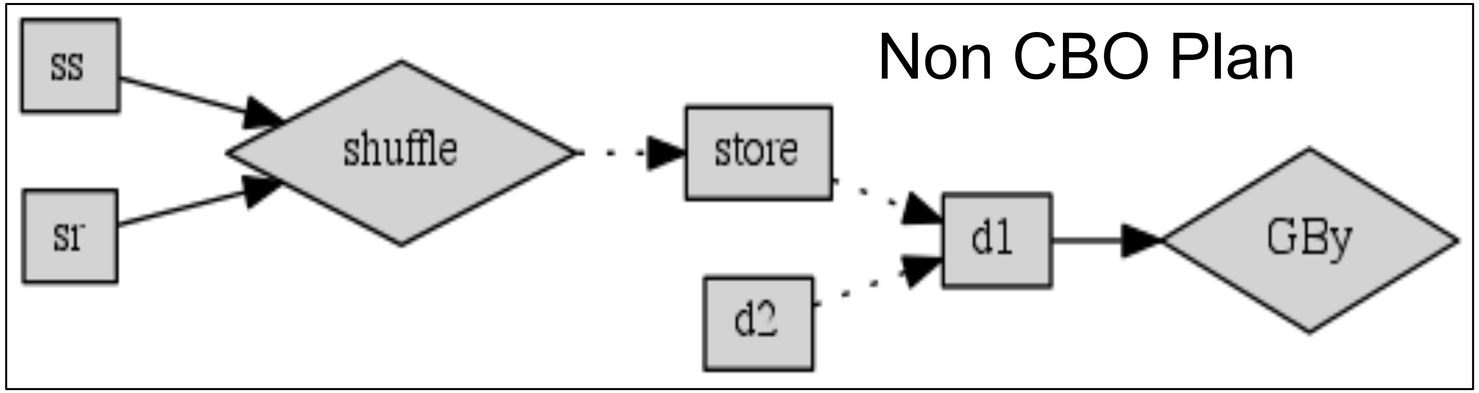


```
select
   s_store_name , ..  other store details
  ,sum(case when (sr_returned_date_sk - ss_sold_date_sk <= 30 ) then 1 else 0 end)  as `30 days`, …
from
   store_sales ss,store_returns sr,store s ,date_dim d1 ,date_dim d2
where
    d2.d_year = 2000 and d2.d_moy  = 9
and ss.ss_ticket_number = sr.sr_ticket_number and ss.ss_item_sk = sr.sr_item_sk
and ss.ss_sold_date_sk    = d1.d_date_sk and sr.sr_returned_date_sk    = d2.d_date_sk
and ss.ss_customer_sk = sr.sr_customer_sk and ss.ss_store_sk = s.s_store_sk
group by store details
order by store details limit 100;
```

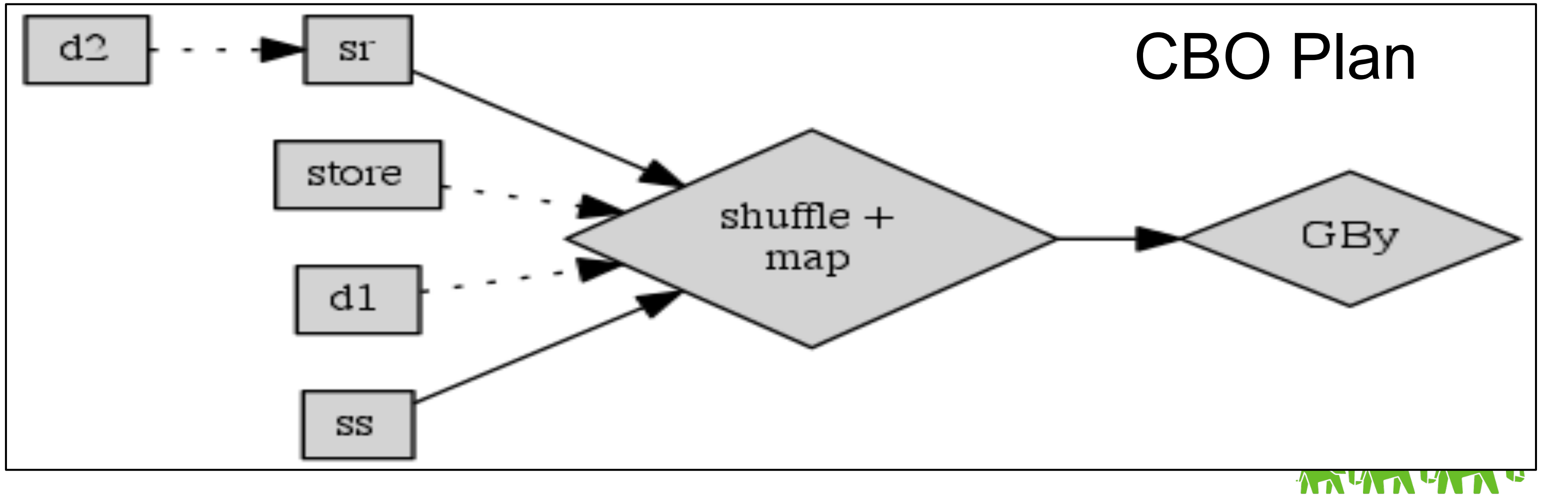
TPC-DS Query 50



Specified Join Tree



Non CBO Plan



CBO Plan

TPC-DS Query 50

Facts restricted to 3 months

	Run 1	Run 2
Non CBO	53.1	53.4
CBO	22.5	21.9

- 1 year test
  - > 10 mins for Non CBO
  - CBO time was about the same
- Fact tables
  - partitioned by Day,
  - bucketed by Item
- Bucketing off
  - Bucketing should help CBO plan.
  - SR table much smaller. Better chance of Bucket Join in place of Shuffle Join.

Orderings considered by Planner

Join Ordering	Cost Estimate
['d2', [[['store_sales', 'd1'], 'store_returns'], 'store']]	515074768.659
['d1', [[['store_sales', 'store'], 'store_returns'], 'd2']]	448155.355
…	
['store_returns', 'd2']	9938.93
['store_sales', 'store_returns']	156727295.634
['d1', 'store_sales']	123675664.449

Hortonworks

TPC-DS Query 17

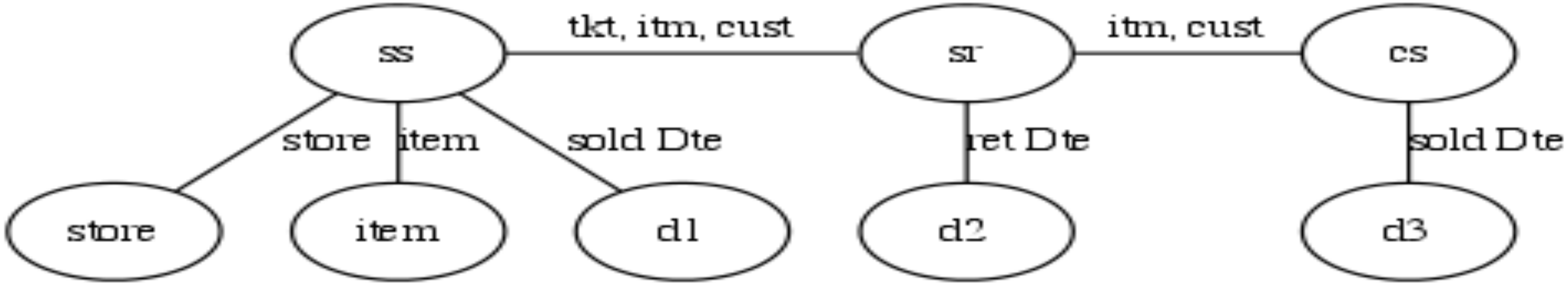
Joins Store Sales, Store Returns and Catalog Sales fact tables.

Each of the fact tables are independently restricted by time.

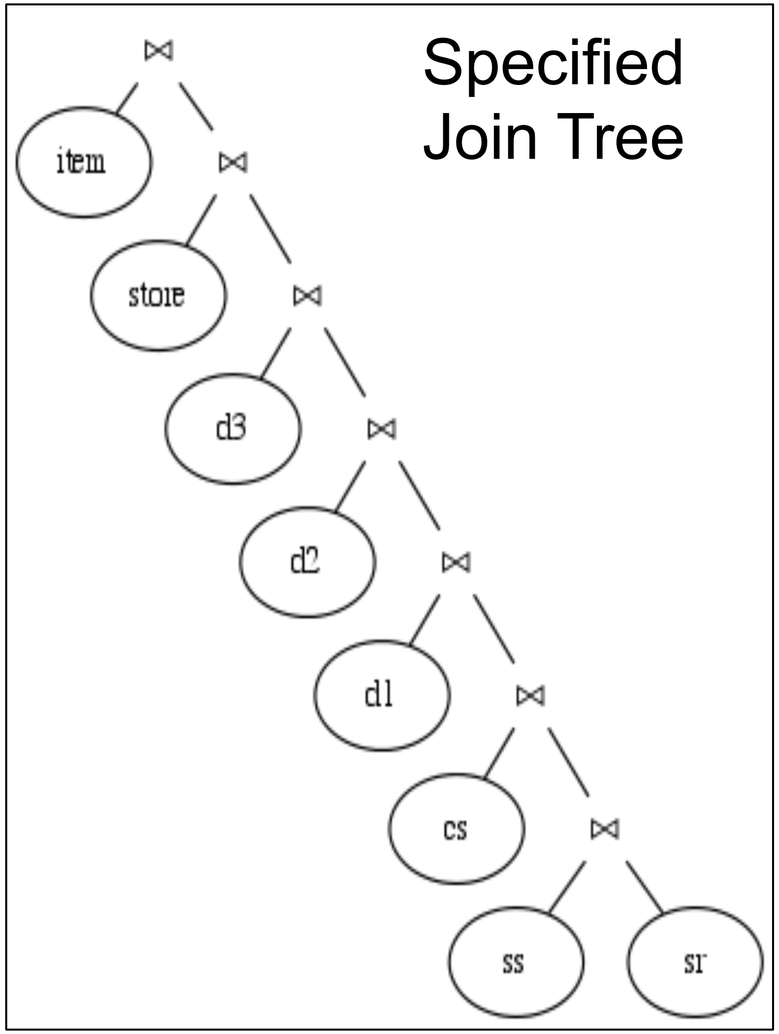
Analysis at Item and Store grain, so these dimensions are also joined in.

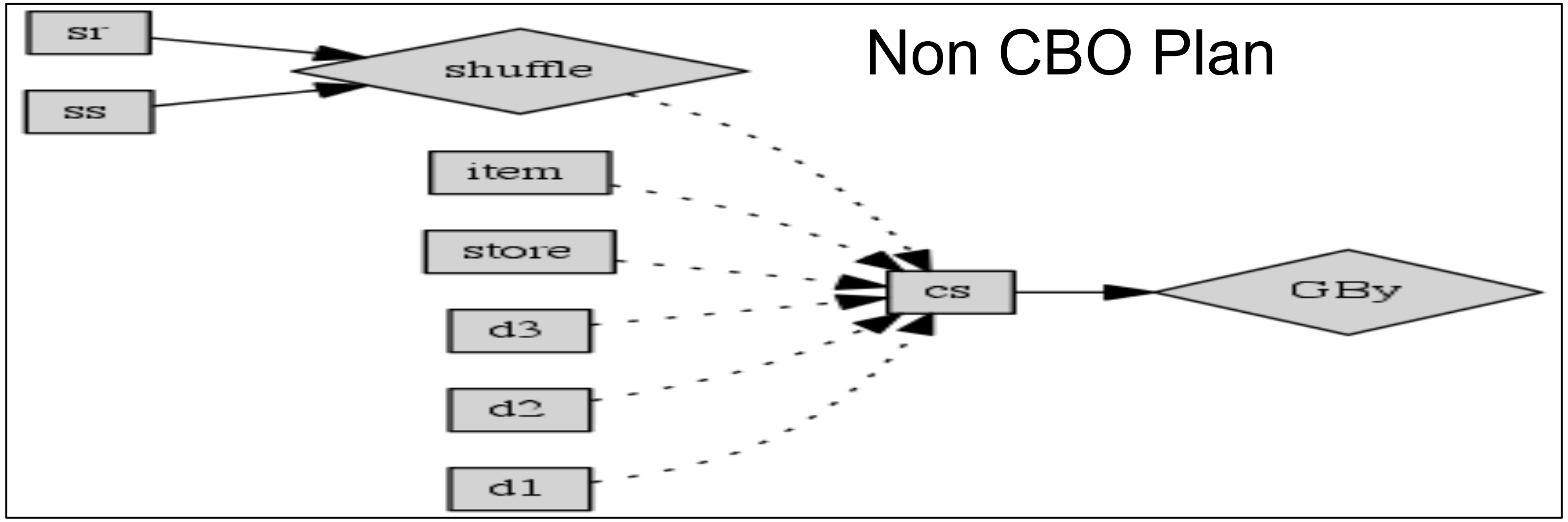
As specified Query starts by joining the 3 Fact tables.

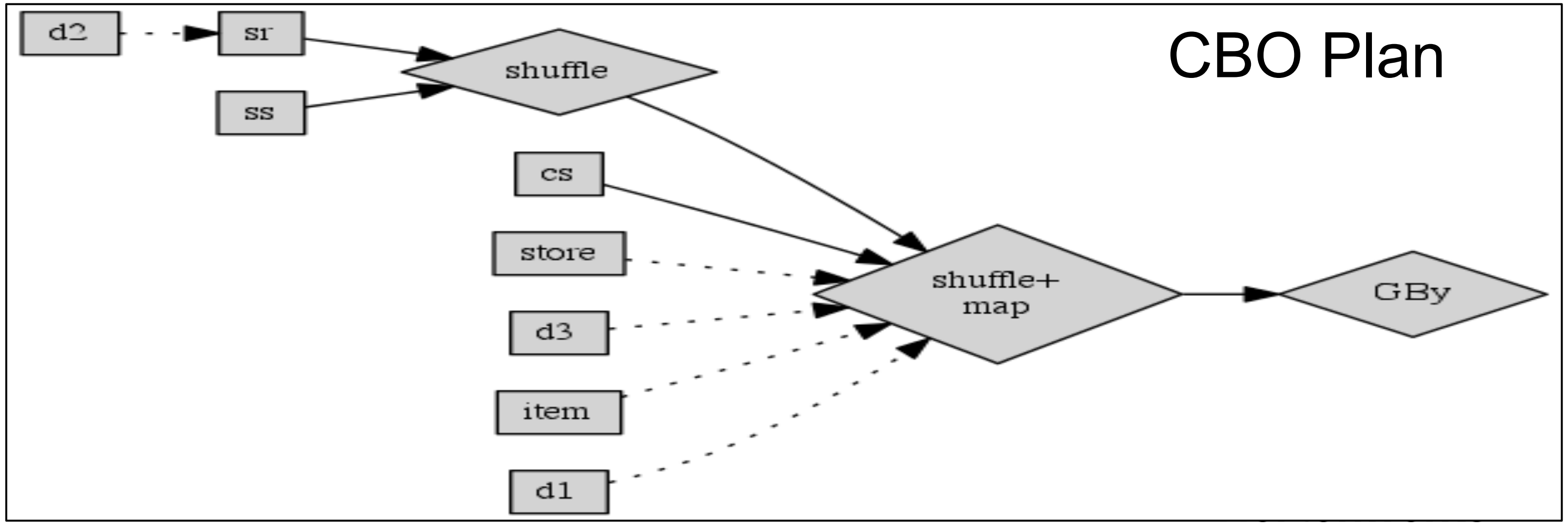
```
select  i_item_id
       ,i_item_desc
       ,s_state
       ,count(ss_quantity) as store_sales_quantitycount
       ,….
from store_sales ss ,store_returns sr, catalog_sales cs,
 date_dim d1, date_dim d2, date_dim d3, store s, item I
where d1.d_quarter_name = '2000Q1’
 and d1.d_date_sk = ss.ss_sold_date_sk
 and i.i_item_sk = ss.ss_item_sk and …
group by i_item_id ,i_item_desc, ,s_state
order by i_item_id ,i_item_desc, s_state
limit 100;
```



TPC-DS Query 17







TPC-DS Query 17

Facts restricted to 3 months

	Run 1	Run 2
Non CBO	100.71	127.53
CBO	50.9	44.52

- 1 year test
  - > 10 mins for Non CBO
  - CBO time was about the same
- Fact tables
  - partitioned by Day,
  - bucketed by Item
- Bucketing off
  - Bucketing should help CBO plan.
  - SR table much smaller. Better chance of Bucket Join in place of Shuffle Join.

Orderings considered by Planner

Join Ordering	Cost Estimate
['item', [[[[['d2', 'store_returns'], 'store_sales'], 'catalog_sales'], 'd1'], 'd3'], 'store']]	3547898.061
…	
['store_returns', 'd2’]	19224.71
['store_sales', 'store_returns’]	23057497.991
['d1', 'store_sales']	26142.943

Next?

Outer joins

Scale to larger numbers of joins

Support all expressions (UDFs)

Join algorithm selection

Sortedness & distribution as a trait

Trait propagation

Better cost model

More statistics

Move all pre-planning and logical planning to Optiq

Use Optiq costs/statistics to help physical planning

Constant reduction & tree pruning

Rewrite query to use materialized view

Thank you!

@julianhyde

http://hive.apache.org/

http://incubator.apache.org/projects/optiq.html