Why is data independence (still) so important?

Julian Hyde @julianhyde

http://github.com/julianhyde/optiq
http://github.com/julianhyde/optiq-splunk

Apache Drill Meeting
2012/9/13

Data independence

This is my opinion about data management systems in general. I don't claim that it is the right answer for Apache Drill.

I claim that a logical/physical separation can make a data management system more widely applicable, therefore more widely adopted, therefore better.

What “data independence” means in today's “big data” world.

About me

Julian Hyde

Database hacker (Oracle, Broadbase, SQLstream, LucidDB)

Open source hacker (Mondrian, olap4j, LucidDB, Optiq)

@julianhyde

http://github.com/julianhyde

http://www.flickr.com/photos/torkildr/3462606643

http://www.flickr.com/photos/sylvar/31436961/

“Big Data”

Right data, right time

Diverse data sources / Performance / Suitable format

Volume / Velocity / Variety

Volume – solved :)

Velocity – not one of Drill's goals (?)

Variety – ?

Variety

Variety of source formats (csv, avro, json, weblogs)

Variety of storage structures (indexes, projections, sort order, materialized views) now or in future

Variety of query languages (DrQL, SQL)

Combine with other data (join, union)

Embed within other systems, e.g. Hive

Source for other systems, e.g. Drill | Cascading > Teradata

Tools generate SQL

Use case: Optiq* at Splunk

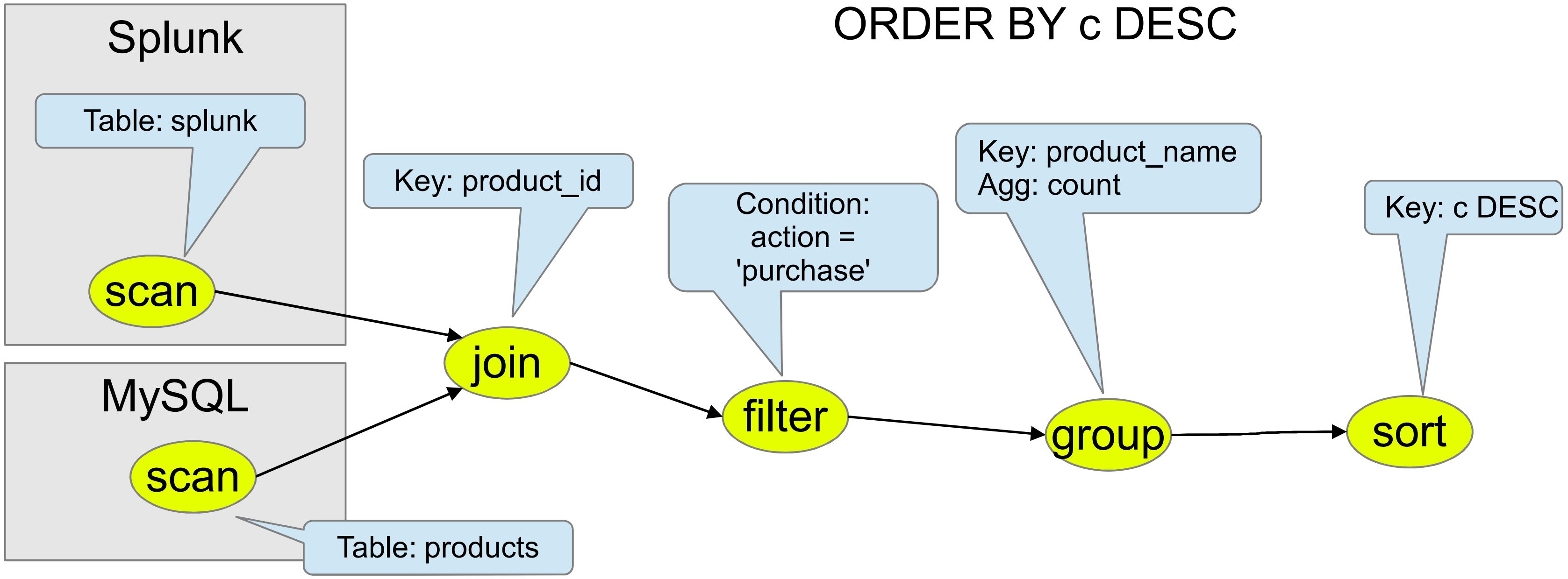
SQL interface on NoSQL system

“Smart” JDBC driver – pushes processing down to Splunk

** Truth in advertising: I am the author of Optiq.*

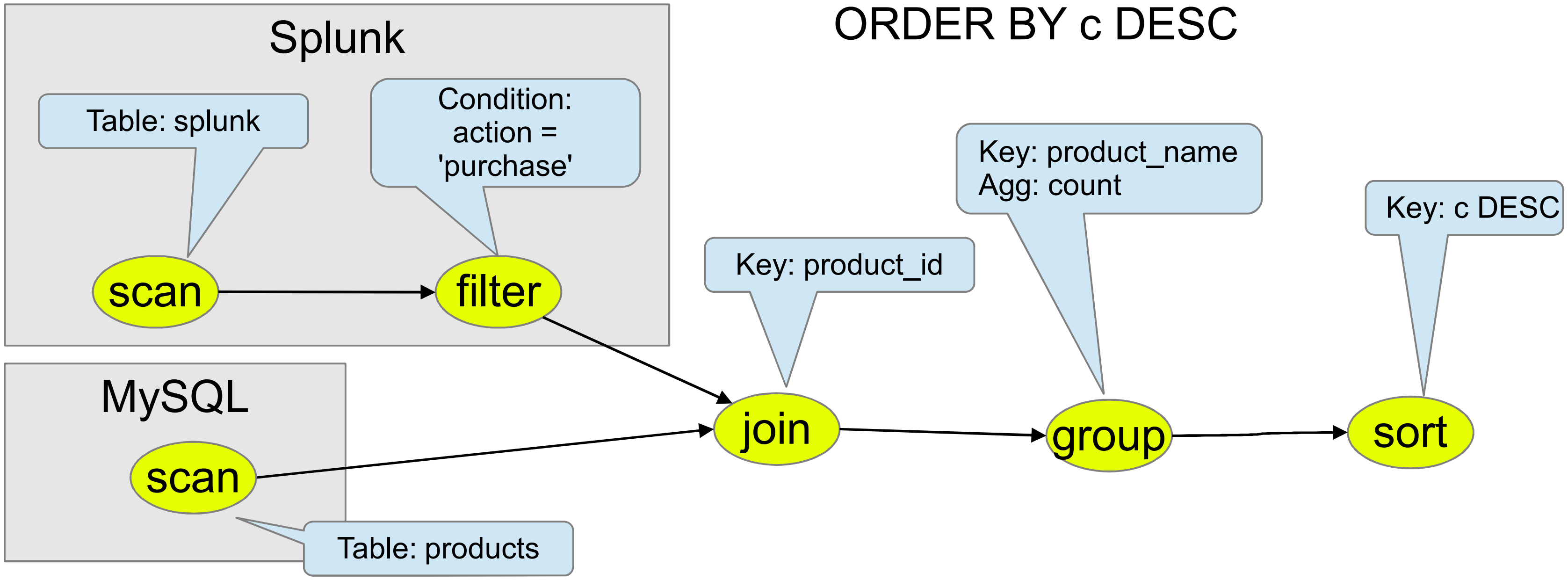
Expression tree

```
SELECT p.“product_name”, COUNT(*) AS c
FROM “splunk”.”splunk” AS s
    JOIN “mysql”.”products” AS p
    ON s.”product_id” = p.”product_id”
WHERE s.“action” = 'purchase'
GROUP BY p.”product_name”
ORDER BY c DESC
```

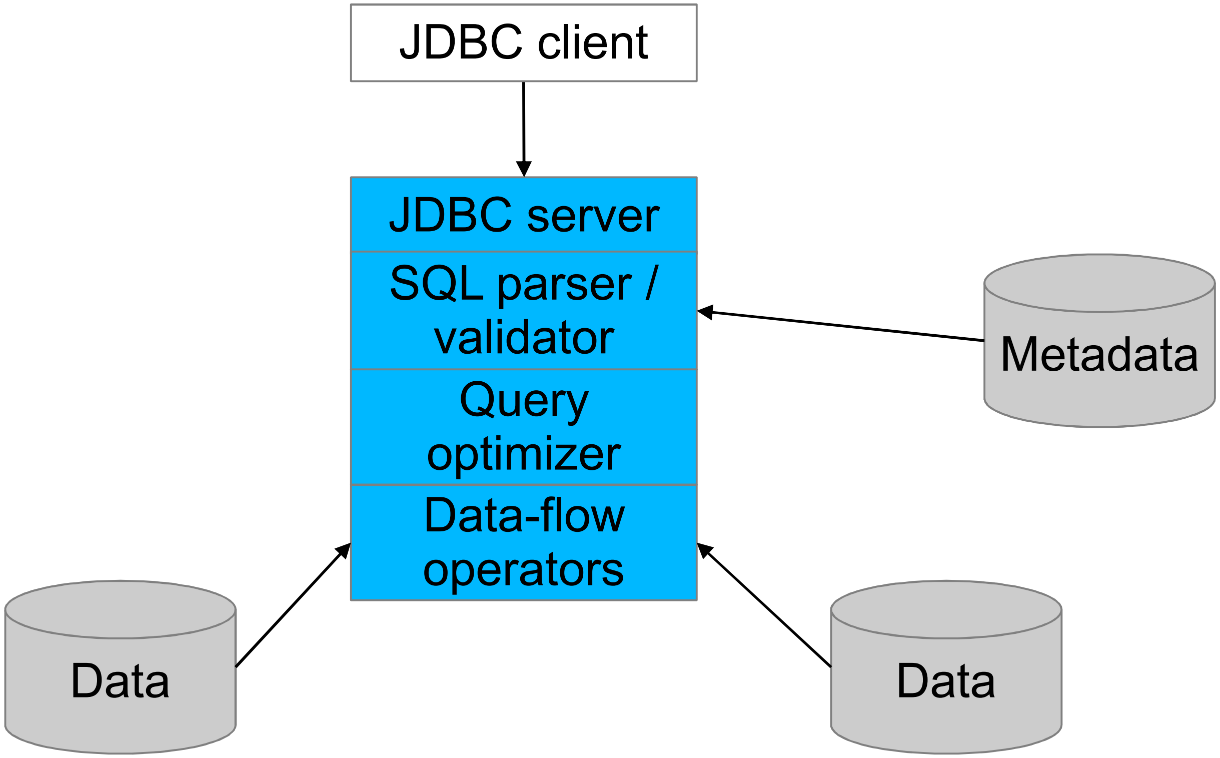


Expression tree (optimized)

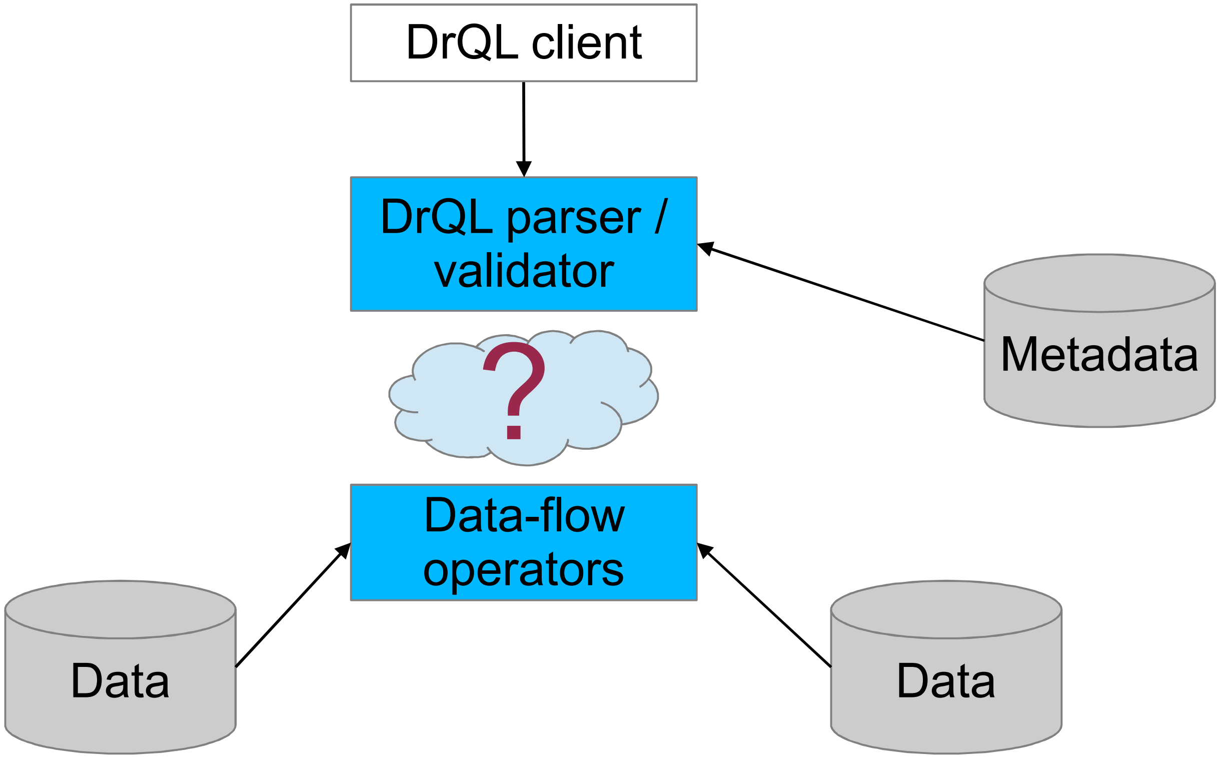
```
SELECT p.“product_name”, COUNT(*) AS c
FROM “splunk”.”splunk” AS s
   JOIN “mysql”.”products” AS p
   ON s.”product_id” = p.”product_id”
WHERE s.“action” = 'purchase'
GROUP BY p.”product_name”
ORDER BY c DESC
```



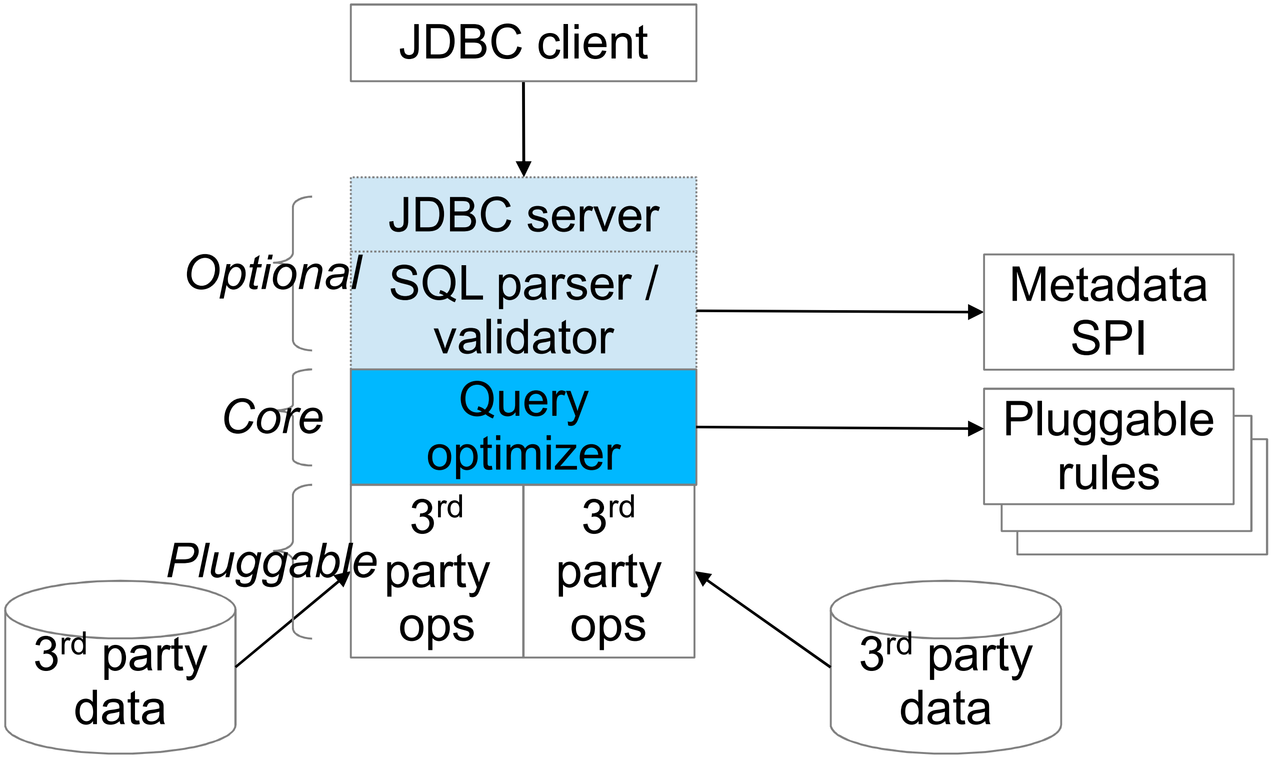
Conventional DBMS architecture



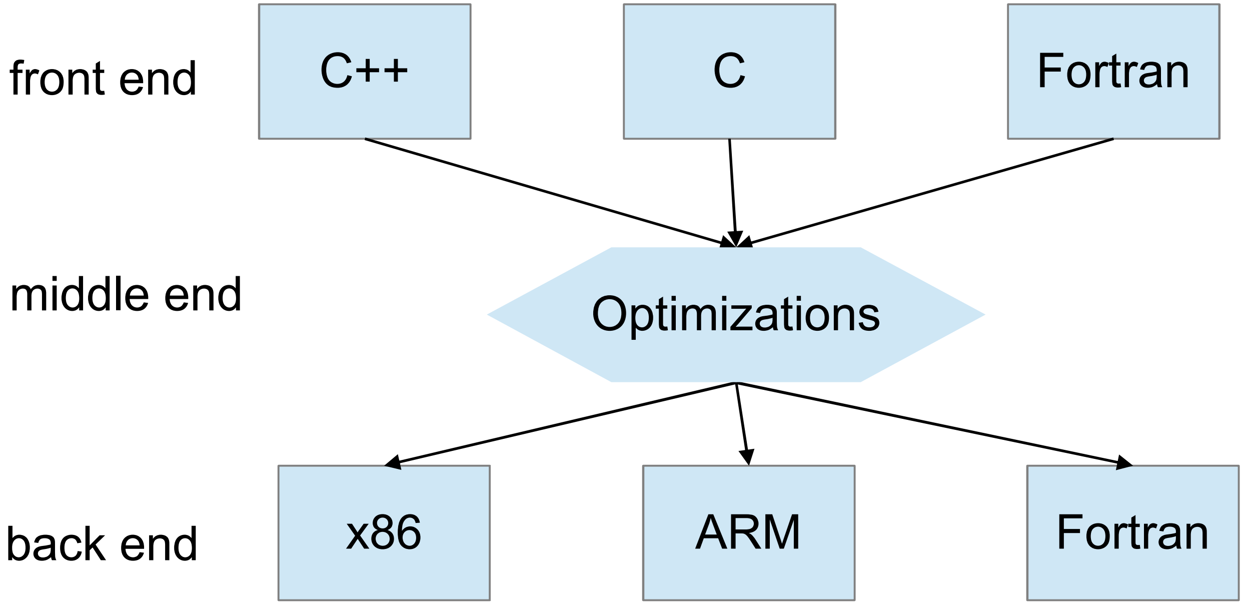
Drill architecture



Optiq architecture



Analogy: Compiler architecture



Conclusions

Clear logical / physical separation allows a data management system to handle a wider variety of data, query languages, and packaging.

Also provides a clear interface between the sub-teams working on query language and operators.

A query optimizer allows new operators, and alternative algorithms and data structures, to be easily added to the system.

Extra material follows...

Writing an adapter

Driver – if you want a vanity URL like “jdbc:drill:”

Schema – describes what tables exist

Table – what are the columns, and how to get the data.

Operators (optional) – non-relational operators, if any

Rules (optional, but recommended) – improve efficiency by changing the question

Parser (optional) – additional source languages