CS 744: SCOPE

Shivaram Venkataraman

Fall 2022

ADMINISTRIVIA

- Assignment 1 grades out! → Roger
- Course Project Proposal: Due soon! → what resources do you need
  - Google Cloud Credits ($150) CloudLab

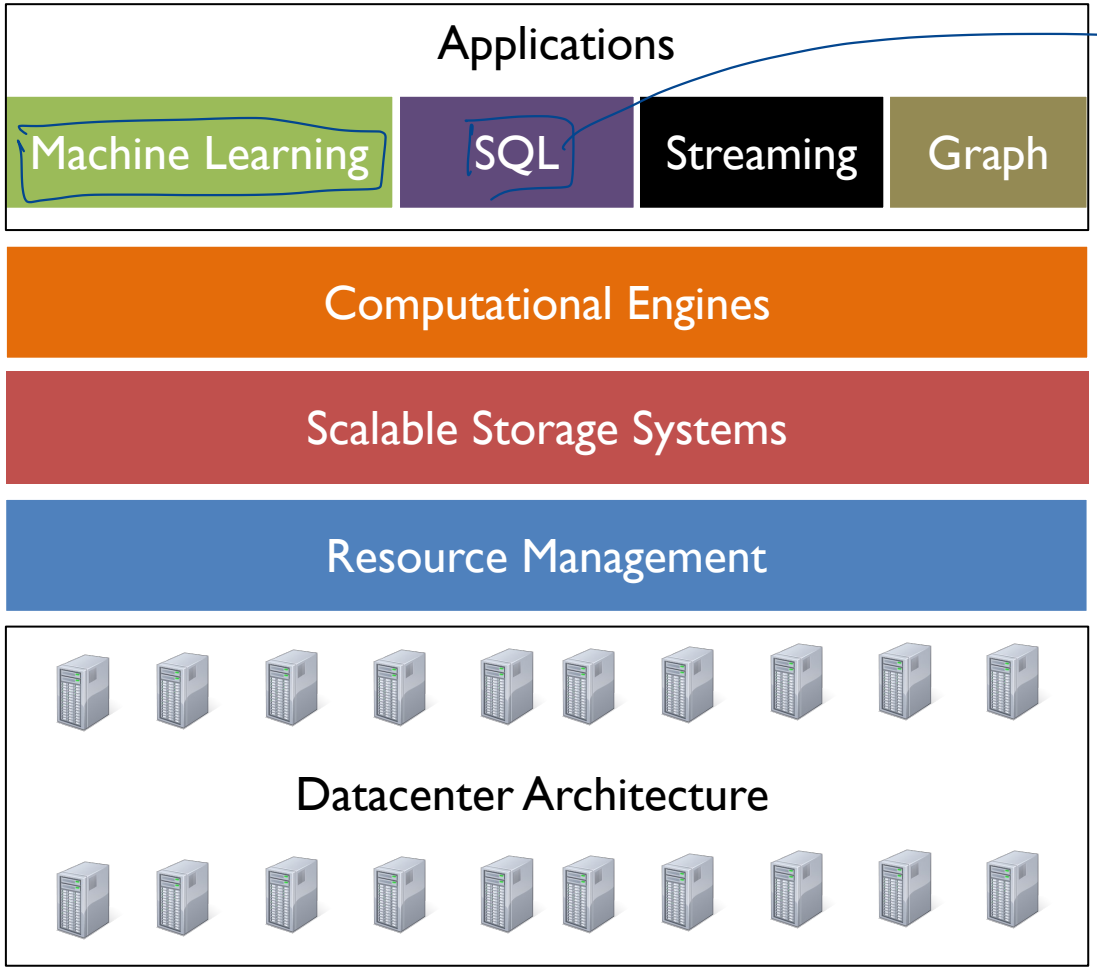
- Midterm details are on Piazza. 1pm to 2.15pm CS 1221. Oct 27th

- No reviews for Tuesday (Snowflake)!

Training

Scheduling

Inference



SCOPE

SparkSQL

Cloud-based

Snowflake

SQL: STRUCTURED QUERY LANGUAGE

DATABASE SYSTEMS

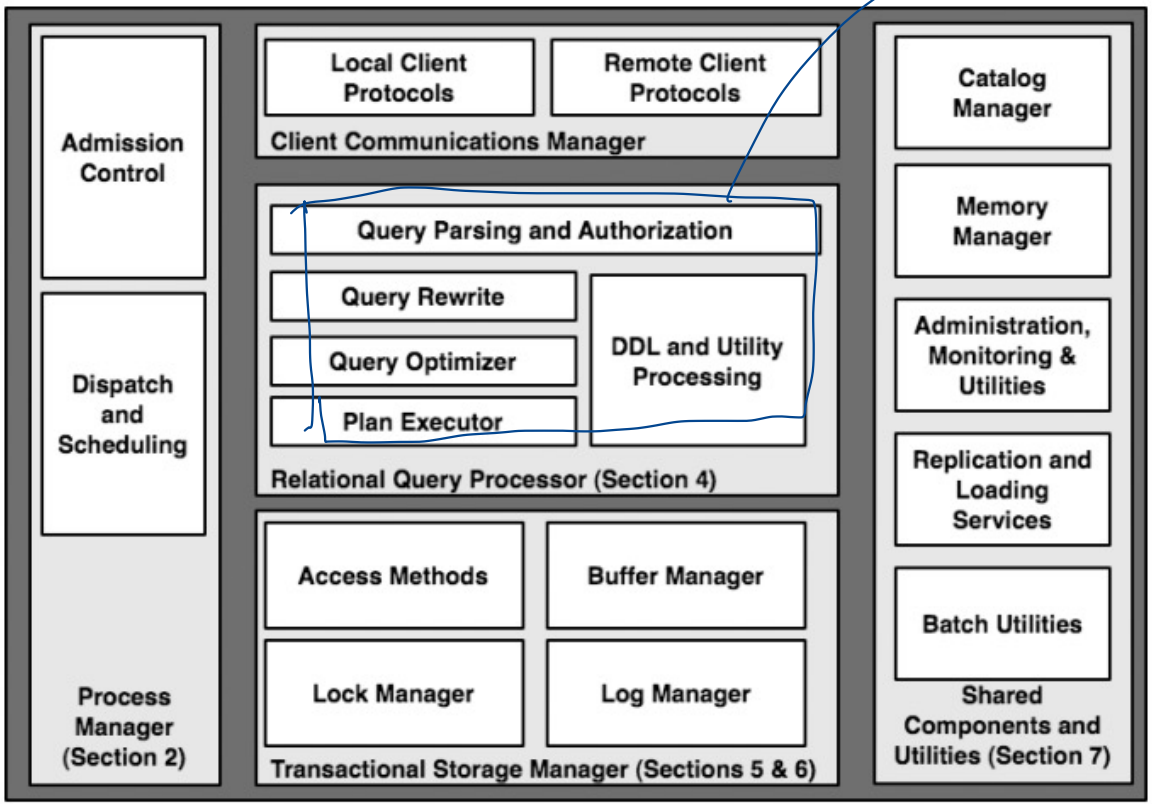
Relational Model

Declarative query

submit

Query Processor

Tuples or rows



PROCEDURAL VS. RELATIONAL

- Programing style
- Flexibility → Re-use of data
  → Custom Functions

```
lines = sc.textFile(“users")
csv = lines.map(x =>
    x.split(‘,’))
young = csv.filter(x =>
    x(1) < 21)
println(young.count())
```

Declarative → "What you want"

System figures out "How"

```
SELECT COUNT(*)
FROM “users”
WHERE age < 21
```

you want to do this

↳ lowers barrier to entry

SCOPE

Transaction (OLTP)

↳ Analytics (OLAP

Microsoft

~ 2008 - 2010

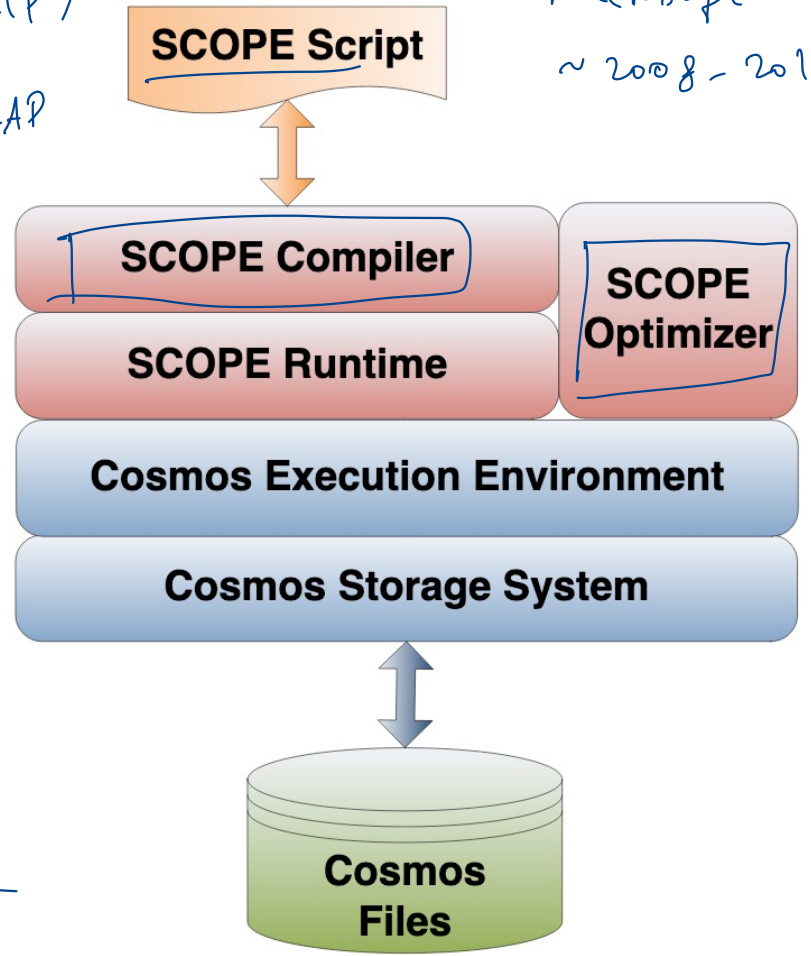
Example

```
SELECT query, COUNT(*)
AS count
FROM "search.log"

USING LogExtractor
GROUP BY query
HAVING count > 1000
ORDER BY count DESC;
```

Similar to MapReduce

Similar to GFS



SCOPE OPERATORS

Input reading: What is different?

- Inputs from many sources
  - Providers / Connectors
  - SQL ≡ Schema on the data you are processing

```
EXTRACT column[:<type>][, ...]
FROM <input_stream(s) >
USING <Extractor> [(args)]
[HAVING <predicate>]
```

Specify schema for the data we will process

Custom class that specifies logic for getting one row

SQL OPERATORS

Select – read rows that satisfy some predicate

Join – Support for Inner and Outer join

GroupBy – Group by some column

OrderBy – Sorting the output

Aggregations – COUNT, SUM, MAX etc.

Subset of the SQL standard

Familiar for users

- easy to use

LANGUAGE INTEGRATION → LINQ

↳ Language integrated Queries

numbers String function

```
R1 = SELECT A+C AS ac, B.Trim() AS B1
FROM R
WHERE StringOccurs(C,"xyz") > 2
```

→ Query planning / execution deals with this.

```
#CS
public static int StringOccurs(string str, string ptrn){
 int cnt=0; int pos=-1;
 while (pos+1 < str.Length) {
    pos = str.IndexOf(ptrn, pos+1);
    if (pos < 0) break;
    cnt++;
  }
  return cnt;
}
#ENDCS
```

→ libraries

→ Compile this & ship it along with query

MAPREDUCE-LIKE? → Mix and match

Process (Map)

Input rows given to a user defined class → One or more rows output (with schema)

Reduce

↳ grouped data

All rows that belong to a group → One or more rows output

Combine → NOT SIMILAR TO MR COMBINE

```
COMBINE S1 WITH S2
ON S1.A==S2.A AND S1.B==S2.B AND S1.C==S2.C
USING MultiSetDifference
PRODUCE A, B, C
```

→ Take two tables which are co-partitioned → One table output

EXECUTION: COMPILER

```
SELECT query, COUNT() AS count
FROM "search.log"
USING LogExtractor
GROUP BY query
HAVING count > 1000
ORDER BY count DESC;
```

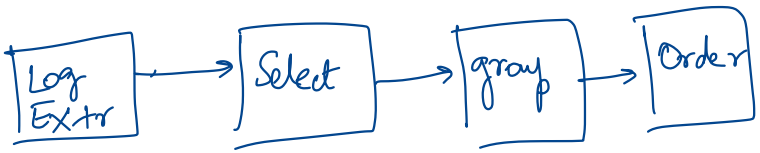
Check syntax, resolve names

Checks if columns have been defined

Result: Internal parse tree

check if column is a number if + on it

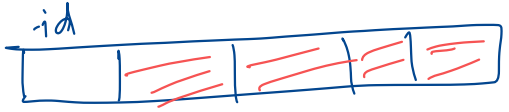
logical query plan



Logical Plan → Physical Plan

OPTIMIZER

row



Cost estimator

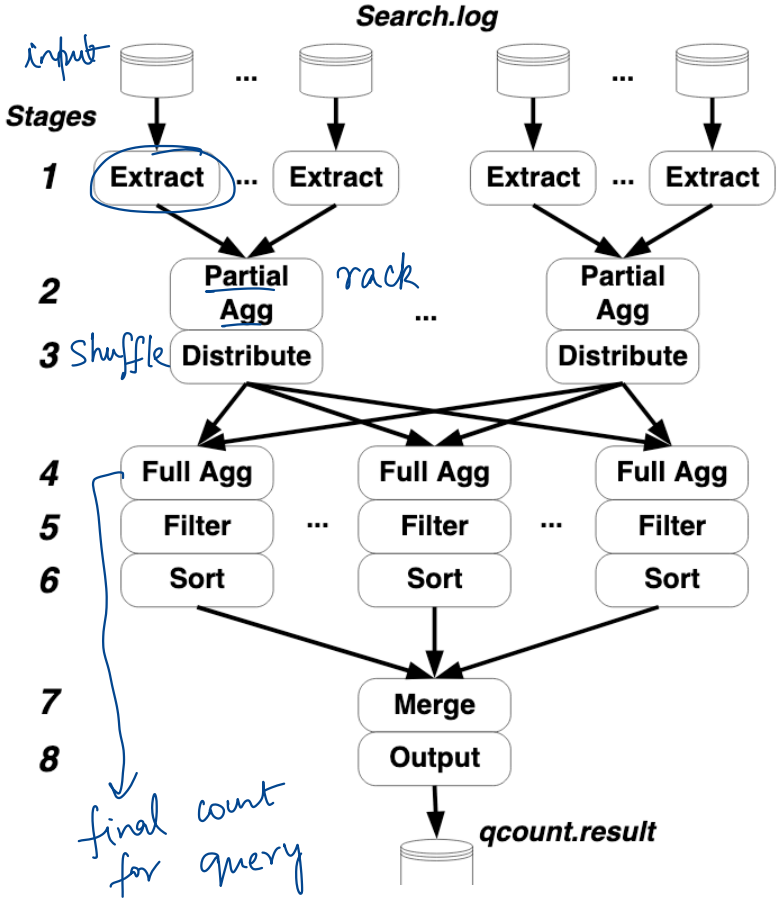
Rewrite the query expression → lowest cost

Examples:

- Removing unnecessary columns
- Pushing down selection predicates
- Pre-aggregating

"Distributed"

```
SELECT query, COUNT() AS count
FROM "search.log"
USING LogExtractor
GROUP BY query
HAVING count > 1000
ORDER BY count DESC;
```



RUNTIME OPTIMIZATIONS

Hierarchical aggregation

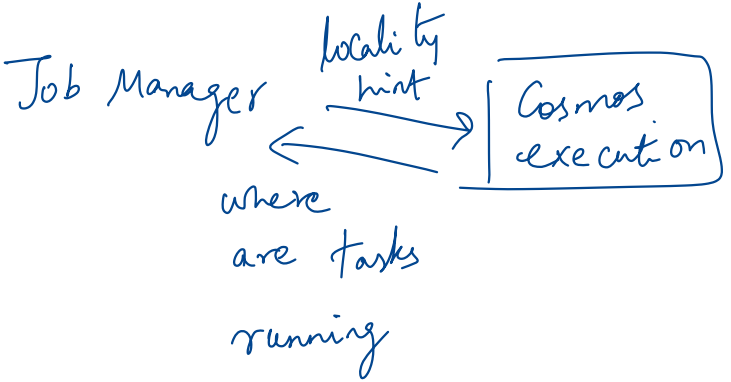
↳ Change the query plan at run time

↳ Partial agg. and full aggregation

→ Insert it based on need.

Locality-sensitive task placement

↳ Place task close to its inputs



SUMMARY, TAKEAWAYS

Relational API

- Enables rich space of optimizations
- Easy to use, integration with C#

Scope Execution

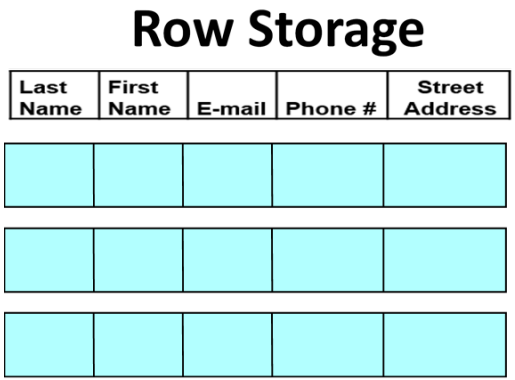
- Compiler to check for errors, generate DAG
- Optimizer to accelerate queries (static + dynamic)

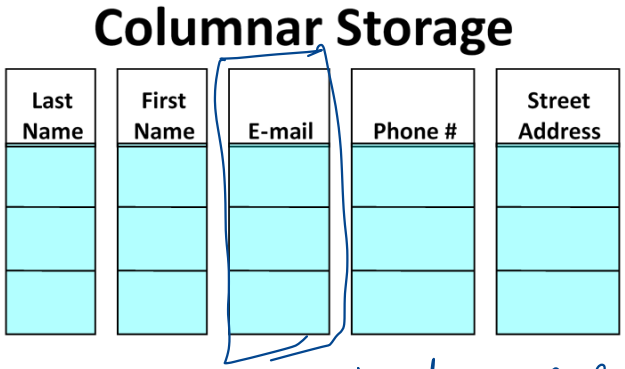
Precursor to systems like SparkSQL, Apache Hive

DISCUSSION

https://forms.gle/CCx6evn2LJAY8m9d9

Consider you have a column-oriented data layout on your storage system (Example below). What are some reasons that a SCOPE query might be faster than running equivalent MR program?





→ map stage extract one column, then aggregation

query only touches one column

↳ push down selection to only read 1 column from disk!

http://dbmsmusings.blogspot.com/2017/10/apache-arrow-vs-parquet-and-orc-do-we.html

Does SCOPE-like Optimizer help ML workloads? Consider the code in your Assignment2. What parts of your code would benefit and what parts would not?

Within 1 iter

- → Pick between All Reduce vs. Gather / Scatter
- → Insert rack or machine-level partial agg

→ Optimizer needs to be across-iterations

→ Pipeline I/O with compute | Improved input reading for ML workloads

→ Forward + Backward pass ?

→ I/O reduction

NEXT STEPS

Next class: Elastic Data Warehousing with SnowFlake

Project proposals due soon! See Piazza!

Midterm: next week