CS 744: SCOPE

Shivaram Venkataraman
Fall 2022
- Assignment 1 grades out!
- Course Project Proposal: Due soon!
  - Google Cloud Credits

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

- No reviews for Tuesday (Snowflake)!
SQL: STRUCTURED QUERY LANGUAGE
DATABASE SYSTEMS
PROCEDURAL VS. RELATIONAL

lines = sc.textFile("users")
csv = lines.map(x =>
    x.split(',
)
young = csv.filter(x =>
    x(1) < 21
println(young.count())
SELECT query, COUNT(*) AS count
FROM "search.log"
USING LogExtractor
GROUP BY query
HAVING count > 1000
ORDER BY count DESC;
SCOPE OPERATORS

Input reading: What is different?

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

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

#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
MAPREDUCE-LIKE?

Process

Reduce

Combine

COMBINE S1 WITH S2
USING MultiSetDifference
PRODUCE A, B, C
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
Rewrite the query expression $\rightarrow$ lowest cost

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

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

Hierarchical aggregation

Locality-sensitive task placement
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
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?

Row Storage

<table>
<thead>
<tr>
<th>Last Name</th>
<th>First Name</th>
<th>E-mail</th>
<th>Phone #</th>
<th>Street Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Columnar Storage

<table>
<thead>
<tr>
<th>Last Name</th>
<th>First Name</th>
<th>E-mail</th>
<th>Phone #</th>
<th>Street Address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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?
Next class: Elastic Data Warehousing with SnowFlake
Project proposals due soon! See Piazza!
Midterm: next week