Lecture #14: Query Planning & Optimization I

15-445/645 Database Systems (Fall 2019)
https://15445.courses.cs.cmu.edu/fall2019/
Carnegie Mellon University
Prof. Andy Pavlo

1 Overview

SQL is declarative. This means that the user tells the DBMS what answer they want, not how to get the answer. Thus, the DBMS needs to translate a SQL statement into an executable query plan. But there are different ways to execute a query (e.g., join algorithms) and there will be differences in performance for these plans. Thus, the DBMS needs a way to pick the "best" plan for a given query. This is the job of the DBMS's optimizer.

There are two types of optimization strategies:

- **Heuristics/Rules:** Rewrite the query to remove inefficiencies. Does not require a cost model.
- **Cost-based Search:** Use a cost model to evaluate multiple equivalent plans and pick the one with the smallest cost.

2 Rule-based Query Optimization

Two relational algebra expressions are equivalent if they generate the same set of tuples. Given this, the DBMS can identify better query plans without a cost model. This is technique often called **query rewriting**. Note that most DBMSs will rewrite the query plan and not the raw SQL string.

Examples of query rewriting:

- **Predicate Push-down:** Perform predicate filtering before join to reduce size of join).
- **Projections Push down:** Perform projections early to create smaller tuples and reduce intermediate results. You can project out all attributes except the ones requested or required (e.g. join attributes).
- **Expression Simplification:** Exploit the transitive properties of boolean logic to rewrite predicate expressions into a more simple form.