ICS 421 Spring 2010
Query Evaluation (i)

Asst. Prof. Lipyeow Lim
Information & Computer Science Department
University of Hawaii at Manoa
SELECT * FROM Reserves WHERE sid=101

Parse Query
Enumerate Plans
Estimate Cost
Choose Best Plan
Evaluate Query Plan

Result

Optimizer

Evaluate Plan A

Pick B

SCAN (sid=101)

IDXSCAN (sid=101)

Index(sid)

32.0

25.0
Parse Query

- **Input**: SQL
  - Eg. SELECT-FROM-WHERE, CREATE TABLE, DROP TABLE statements

- **Output**: Some data structure to represent the “query”
  - Relational algebra?

- **Also checks syntax, resolves aliases, binds names in SQL to objects in the catalog**

- **How?**
Enumerate Plans

• **Input**: a data structure representing the “query”

• **Output**: a collection of equivalent query evaluation plans

• **Query Execution Plan** (QEP): tree of database operators.
  – high-level: RA operators are used
  – low-level: RA operators with particular implementation algorithm.

• **Plan enumeration**: find **equivalent** plans
  – Different QEPs that return the same results
  – Query rewriting: transformation of one QEP to another equivalent QEP.
Estimate Cost

- **Input**: a collection of equivalent query evaluation plans
- **Output**: a cost estimate for each QEP in the collection
- **Cost estimation**: a mapping of a QEP to a cost
  - **Cost Model**: a model of what counts in the cost estimate. Eg. Disk accesses, CPU cost...
- Statistics about the data and the hardware are used.
Choose Best Plan

- **Input**: a collection of equivalent query evaluation plans and their cost estimate
- **Output**: best QEP in the collection
- The steps: enumerate plans, estimate cost, choose best plan collectively called the:

  - **Query Optimizer**:
    - Explores the space of equivalent plan for a query
    - Chooses the best plan according to a cost model

Diagram:

1. Parse Query
2. Enumerate Plans
3. Estimate Cost
4. Choose Best Plan
5. Evaluate Query Plan
6. Result
Evaluate Query Plan

- **Input**: a QEP (hopefully the best)
- **Output**: Query results
- Often includes a “code generation” step to generate a lower level QEP in executable “code”.
- **Query evaluation engine** is a “virtual machine” that executes some code representing low level QEP.
Query Execution Plans (QEPs)

• A tree of database operators: each operator is a RA operator with specific implementation

• Selection $\sigma$: Index Scan or Table Scan

• Projection $\pi$:
  – Without DISTINCT: Table Scan
  – With DISTINCT: requires sorting or index scan

• Join $\Join$:
  – Nested loop joins (naïve)
  – Index nested loop joins
  – Sort merge joins

• Sort:
  – In-memory sort
  – External sort
SELECT S.sname
FROM Reserves R, Sailors S
WHERE R.sid=S.sid AND R.bid=100 AND S.rating>5

On the fly
Nested Loop Join
Temp T1

(SCAN) Reserves

(SCAN) Sailors

(R.bid=100) Reserves

(Scan) Sailors

(R.sid=S.sid) Reserve

(R.sid=S.sid) Sailors

(S.rating>5) Reserve

(S.rating>5) Sailors

(R.bid=100) Reserve

(P.S.sname) Reserve

(P.S.sname) Sailors
Access Paths

- An **access path** is a method of retrieving tuples. Eg. Given a query with a selection condition:
  - File or table scan
  - Index scan

- **Index matching problem**: given a selection condition, which indexes can be used for the selection, i.e., matches the selection?
  - Selection condition normalized to conjunctive normal form (CNF), where each term is a *conjunct*
  - Eg. \((\text{day}<8/9/94 \ \text{AND} \ \text{rname}='Paul') \ \text{OR} \ \text{bid}=5 \ \text{OR} \ \text{sid}=3\)
  - CNF: \((\text{day}<8/9/94 \ \text{OR} \ \text{bid}=5 \ \text{OR} \ \text{sid}=3) \ \text{AND} \ (\text{rname}='Paul' \ \text{OR} \ \text{bid}=5 \ \text{OR} \ \text{sid}=3)\)
Index Matching

- A **tree index** matches a selection condition if the selection condition is a prefix of the index search key.
- A **hash index** matches a selection condition if the selection condition has a term `attribute=value` for every attribute in the index search key.

Q1: $\sigma_{a=5 \text{ AND } b=3}$
Q2: $\sigma_{a=5 \text{ AND } b>6}$
Q3: $\sigma_{b=3}$
Q4: $\sigma_{a=5 \text{ AND } b=3 \text{ AND } c=5}$
Q5: $\sigma_{a>5 \text{ AND } b=3 \text{ AND } c=5}$

I1: Tree Index (a,b,c)
I2: Tree Index (b,c,d)
I3: Hash Index (a,b,c)
One Approach to Selections

1. Find the *most selective access path*, retrieve tuples using it
2. Apply remaining terms in selection not matched by the chosen access path

- The **selectivity** of an access path is the size of the result set (in terms of tuples or pages).
  - Sometimes selectivity is also used to mean **reduction factor**: fraction of tuples in a table retrieved by the access path or selection condition.

- Eg. Consider the selection:
  - `day<8/9/94 AND bid=5 AND sid=3`
  - Tree Index(day)
  - Hash index (bid,sid)