

Query Evaluation & Optimization

Nov 9, 2011

Chapter 12

Catalog

- Meta data about the tables
 - names
 - column name, domain
 - indexes
 - size
- Cardinality or NTuples(R)
- Size or NPages(R)
- Index Cardinality or NKeys(I)
- Index Size or INPages(I)
- Index Height or IHeight(I)
- Index Range or ILow(I)/IHigh(I)

Setup

- Sailors(sid, sname, rating, age)
- Reserves(sid, bid, day, rname)

- Reserves: 40 bytes per tuple
 - NPages(Reserves) = 1000
 - NTuples(Reserves) = 100000
 - NKeys(<rname, bid, sid>) = 100

- Sailors: 50 bytes per tuple
 - NPages(Sailors) = 500
 - NTuples(Sailors) = 40000

Simple Heuristics

- Indexing
- Iteration
- Partitioning

Goal:
Low **cost**

Access Path

- How to retrieve a tuple from a table

- File Scan

OR

- index plus matching selection condition

Matching

- Conjunctive Normal Form
 - may only match subset
 - primary conjuncts

- Hash index

- Tree index

So you would guess InnoDB uses what types of indexes?

- Why?

Cost

- Selectivity of access path
 - most selective
 - reduction factor
- Index File
- Data File

Operations

- Selection
- Projection
 - remove duplicates
 - `SELECT DISTINCT(FName) FROM Students`
 - `SELECT COUNT(DISTINCT(FName)) FROM Students`
 - partitioning: scan then sort
 - with index
 - with clustered index

Operations, cont p 403

- Join
 - index nested loops join
 - Reserves.sid=Sailors.sid
 - how many I/O operations are needed?
 - What do we know about Reserves, Sailors, sid?

 - Reserves.rname = Sailors.sname
 - how many I/O operations are needed?
 - sort-merge join

```
SELECT S.sname  
FROM Reserves as R, Sailors as S  
WHERE R.sid=S.sid AND R.bid=100 AND S.rating > 5
```

- with hash indexes on bid another hash index on Sailors.sid
- what if we had a tree index on rating?