12

OVERVIEW OF QUERY EVALUATION

Exercise 12.1 Briefly answer the following questions:

1. Describe three techniques commonly used when developing algorithms for relational operators. Explain how these techniques can be used to design algorithms for the selection, projection, and join operators.
2. What is an access path? When does an index *match* an access path? What is a *primary conjunct*, and why is it important?
3. What information is stored in the system catalogs?
4. What are the benefits of storing the system catalogs as relations?
5. What is the goal of query optimization? Why is optimization important?
6. Describe *pipelining* and its advantages.
7. Give an example query and plan in which pipelining *cannot* be used.
8. Describe the *iterator* interface and explain its advantages.
9. What role do statistics gathered from the database play in query optimization?
10. What were the important design decisions made in the System R optimizer?
11. Why do query optimizers consider only left-deep join trees? Give an example of a query and a plan that would not be considered because of this restriction.

Answer 12.1 1. Answer not available yet.

2. Anwswer not available yet.
3. Information about relations, indexes, and views is stored in the system catalogs. This includes file names, file sizes, and file structure, the attribute names and data types, lists of keys, and constraints.

   Some commonly stored statistical information includes:

(a) Cardinality - the number of tuples for each relation

(b) Size - the number of pages in each relation

(c) Index Cardinality - the number of distinct key values for each index

(d) Index Size - the number of pages for each index (or number of leaf pages)

(e) Index Height - the number of nonleaf levels for each tree index

(f) Index Range - the minimum present key value and the maximum present key value for each index.

4. There are several advantages to storing the system catatlogs as relations. Relational system catalogs take advantage of all of the implementation and management benefits of relational tables: effective information storage and rich querying capabilities. The choice of what system catalogs to maintain is left to the DBMS implementor.

5. The goal of query optimization is to avoid the worst plans and find a good plan. The goal is usually not to find the optimal plan. The difference in cost between a good plan and a bad plan can be several orders of magnitude: a good query plan can evaluate the query in seconds, whereas a bad query plan might take days!

6. Pipelining allows us to avoid creating and reading temporary relations; the I/O savings can be substantial.

7. Bushy query plans often cannot take advantage of pipelining because of limited buffer or CPU resources. Consider a bushy plan in which we are doing a selection on two relations, followed by a join. We cannot always use pipelining in this strategy becuase the result of the selection on the first selection may not fit in memory, and we must wait for the second relation's selection to complete before we can begin the join.

8. The iterator interface for an operator includes the functions *open*, *get_next*, and *close*; it hides the details of how the operator is implemented, and allows us to view all operator nodes in a query plan uniformly.

9. The query optimizer uses statistics to improve the chances of selecting an optimum query plan. The statistics are used to calculate reduction factors which determine the results the optimizer may expect given different indexes and inputs.

10. Some important design decisions in the System R optimizer are:

    (a) Using statistics about a database instance to estimate the cost of a query evaluation plan.

    (b) A decision to consider only plans with binary joins in which the inner plan is a base relation. This heuristic reduces the often significant number of alternative plans that must be considered.

(c) A decision to focus optimization on the class of SQL queries without nesting and to treat nested queries in a relatively ad hoc way.

(d) A decision not to perform duplicate elimination for projections (except as a final step in the query evaluation when required by a `DISTINCT` clause).

(e) A model of cost that accounted for CPU costs as wellas I/O costs.

Exercise 12.2 Consider a relation R(*a,b,c,d,e*) containing 5,000,000 records, where each data page of the relation holds 10 records. R is organized as a sorted file with secondary indexes. Assume that *R.a* is a candidate key for R, with values lying in the range 0 to 4,999,999, and that R is stored in *R.a* order. For each of the following relational algebra queries, state which of the following three approaches is most likely to be the cheapest:

- Access the sorted file for R directly.
- Use a (clustered) B+ tree index on attribute *R.a*.
- Use a linear hashed index on attribute *R.a*.

1. $\sigma_{a<50,000}(R)$
2. $\sigma_{a=50,000}(R)$
3. $\sigma_{a>50,000 \wedge a<50,010}(R)$
4. $\sigma_{a\neq 50,000}(R)$

Answer 12.2 Answer omitted.

Exercise 12.3 For each of the following SQL queries, for each relation involved, list the attributes that must be examined to compute the answer. All queries refer to the following relations:

Emp(*eid:* `integer`, *did:* `integer`, *sal:* `integer`, *hobby:* `char(20)`)
Dept(*did:* `integer`, *dname:* `char(20)`, *floor:* `integer`, *budget:* `real`)

1. `SELECT * FROM` Emp E
2. `SELECT * FROM` Emp E, Dept D
3. `SELECT * FROM` Emp E, Dept D `WHERE` E.did = D.did
4. `SELECT` E.eid, D.dname `FROM` Emp E, Dept D `WHERE` E.did = D.did

Answer 12.3 The answer to each question is given below.

1. E.eid, E.did, E.sal, E.hobby
2. E.eid, E.did, E.sal, E.hobby, D.did, D.dname, D.floor, D.budget
3. E.eid, E.did, E.sal, E.hobby, D.did, D.dname, D.floor, D.budget
4. E.eid, D.dname, E.did, D.did

Exercise 12.4 Consider the following schema with the Sailors relation:

Sailors(*sid:* `integer`, *sname:* `string`, *rating:* `integer`, *age:* `real`)

For each of the following indexes, list whether the index matches the given selection conditions. If there is a match, list the primary conjuncts.

1. A B+-tree index on the search key ⟨ Sailors.sid ⟩.
   (a) $\sigma_{Sailors.sid<50,000}(Sailors)$
   (b) $\sigma_{Sailors.sid=50,000}(Sailors)$
2. A hash index on the search key ⟨ Sailors.sid ⟩.
   (a) $\sigma_{Sailors.sid<50,000}(Sailors)$
   (b) $\sigma_{Sailors.sid=50,000}(Sailors)$
3. A B+-tree index on the search key ⟨ Sailors.sid, Sailors.age ⟩.
   (a) $\sigma_{Sailors.sid<50,000 \wedge Sailors.age=21}(Sailors)$
   (b) $\sigma_{Sailors.sid=50,000 \wedge Sailors.age>21}(Sailors)$
   (c) $\sigma_{Sailors.sid=50,000}(Sailors)$
   (d) $\sigma_{Sailors.age=21}(Sailors)$
4. A hash-tree index on the search key ⟨ Sailors.sid, Sailors.age ⟩.
   (a) $\sigma_{Sailors.sid=50,000 \wedge Sailors.age=21}(Sailors)$
   (b) $\sigma_{Sailors.sid=50,000 \wedge Sailors.age>21}(Sailors)$
   (c) $\sigma_{Sailors.sid=50,000}(Sailors)$
   (d) $\sigma_{Sailors.age=21}(Sailors)$

Answer 12.4 Answer omitted.

Exercise 12.5 Consider again the schema with the Sailors relation:

Sailors(*sid:* `integer`, *sname:* `string`, *rating:* `integer`, *age:* `real`)

Assume that each tuple of Sailors is 50 bytes long, that a page can hold 80 Sailors tuples, and that we have 500 pages of such tuples. For each of the following selection conditions, estimate the number of pages retrieved, given the catalog information in the question.

1. Assume that we have a B+-tree index T on the search key ⟨ Sailors.sid ⟩, and assume that $IHeight(T) = 4$, $INPages(T) = 50$, $Low(T) = 1$, and $High(T) = 100{,}000$.
   (a) $\sigma_{Sailors.sid<50{,}000}(Sailors)$
   (b) $\sigma_{Sailors.sid=50{,}000}(Sailors)$
2. Assume that we have a hash index T on the search key ⟨ Sailors.sid ⟩, and assume that $IHeight(T) = 2$, $INPages(T) = 50$, $Low(T) = 1$, and $High(T) = 100{,}000$.
   (a) $\sigma_{Sailors.sid<50{,}000}(Sailors)$
   (b) $\sigma_{Sailors.sid=50{,}000}(Sailors)$

Answer 12.5 Answer not available yet.

Exercise 12.6 Consider the two join methods described in Section **??**. Assume that we join two relations R and S, and that the systems catalog contains appropriate statistics about R and S. Write formulas for the cost estimates of the index nested loops join and sort-merge join using the appropriate variables from the systems catalog in Section **??**. For index nested loops join, consider both a B+ tree index and a hash index. (For the hash index, you can assume that you can retrieve the index page containing the rid of the matching tuple with 1.2 I/Os on average.)

Answer 12.6 Answer omitted.