Uni Freiburg, Web Science Group
Prof. Peter Fischer
Systems Infrastructure for Data Science - Winter 2013/14

Exercise Sheet #6: Query Optimization

Dec 4, 2013

Exercise 6.1 : From Queries to Execution Plans

Consider the following partial schema of a database which keeps track of students and the courses they take:

$Student(\underline{matrikel\_no}, first\_name, last\_name)$
$Class(\underline{class\_id}, class\_name)$
$Takes(\underline{class\_id}, \underline{matrikel\_no})$

Primary keys are underlined. You may assume the obvious foreign-key relations and indexes on primary and foreign key columns. The table cardinalities are as follows:

$|Student| = 1,000$
$|Class| = 100$
$|Takes| = 5,000$

A. Express the following natural-language query in SQL: Give me the last names of all students who take the course called "Information Systems".

B. Convert the SQL statement to a straight-forward query-plan using only cartesian products, not joins. Estimate the cardinality of each intermediate result and the final result.

C. Push-down the selection operators below or into the cartesian products (turning them into joins). Estimate the cardinalities of each intermediate result using your knowledge of table cardinalities and foreign-key relations.

D. Choose access paths and physical implementations of the various operators (particularly joins), taking advantage of indexes or sort order. Try to find a near-optimal plan. Revise your join order if necessary.

Exercise 6.2 : Query Planning

Consider the database containing the following tables:

$Courses(\underline{ID}, Name, Room, Time)$
$Exercises(\underline{ID}, C\_ID, A\_ID, Room, Time)$
$Assistants(\underline{ID}, Firstname, Lastname)$

and the query:

```
SELECT C.Name, A.Firstname, A.Lastname, E.Room, E.Time
FROM Courses C, Assistants A, Exercises E
WHERE C.ID = E.C_ID AND A.ID = E.A_ID AND C.Room like '10%' AND E.Room not like 'CAB%';
```

A. Show the logical canonical form for this query expressed in relational algebra. The logical canonical form is the straight-forward translation of the SQL query into a relational algebra query plan containing only selection, projection and cross product operators.

B. Perform Query Rewrite on part [A]. Which rules would you apply to get to this intermediate representation? Show the logical query plan of the resulting query.

C. Assume that there is a non-clustered index on $C.Room$ and a clustered, direct index on $E.A\_ID$. Based on part [B], show possible physical query execution plans (i.e., access paths, join implementations) that take advantage of the indexes. Discuss the efficiency of your plans based on the following cost metrics: disk I/O, intermediate result size.