Database Tuning, ITU, Spring 2009

Rasmus Pagh

March 26, 2009

Project – deliverable 3

Deadline: April 21, 23.59 Danish time.

Time plan for deliverable 3

March 26. Description, question/answer session.

March 31, April 7, and April 21. Office hours for teaching assistant, 13-15 in room 2A18 (Mai Sun). NB! No office hours on April 14, due to easter break.

April 21. Hand-in by e-mail (pdf file) to `milan@itu.dk`

Purpose

The purpose of this deliverable is to gain experience in analyzing query plans, and applying methods for tuning queries.

Overview

In this deliverable you continue your work on the VXL database. Your first task is to look at and understand Oracle's query plan for each query. You should then consider the possibility of:

- Improving the available access methods (e.g., by making additional indexes, partitioning or denormalizing),
- rewriting queries,
- updating statistics,
- aggregate maintenance,
- ... and other ways of influencing the query plan.

For example, if a B-tree index is used only for an index nested loop join, consider replacing it with a hash index. If an equality condition consistently has low selectivity, consider the use of a bitmap index (not available in Oracle XE).

Testing framework

The only constants in QueryTestUnit that you will change are num_iterations and updatesPerIteration. Until you get everything to work correctly you may set them to the smallest possible values (1,1). When you perform *real* performance tests set num_iterations = 100 and updatesPerIteration = 2 or 15, depending on the workload scenario. The only other change in the source code that you may want to make is temporarily putting some method calls under comments. Yet, when doing "official" tests make sure that the entire set of testing methods is executed.

The times that you will report should be obtained as the averages of, say, 3 executions. If some test shows times that significantly deviate from the average, discard it and do another one. That will happen when some other task, either within Oracle or elsewhere in the system, consumes considerable resources. Still, on average the output times will be reasonable indicators of performance.

New queries

You will be working with the 10 parameterized queries from the second deliverable, plus some new queries found below.

11. Find the total number of hours worked per week by drivers in the region named `s`.
    public int queryHours(String region)

12. Find the number of customers that has not sent a parcel after date `d`.
    public int queryNotRecentCustomers(Date d)

13. Find all parcels sent from postal code `p1` to postal code `p2`
    public ResultSet queryParcelFromTo(int p1, int p2)

14. Find all vehicles that can carry more than 1000 kilograms, *and* have driven less than 100,000 kilometers.
    public ResultSet queryGoodVehicles()

15. Find the newest (most recently inserted) entry of the schedule of the bookable vehicle with id `vid`. This is the one with the highest value of schedule_id.
    public ResultSet queryLatestSchedule(int vid)

16. Find all storage locations at longitude between x and $x + 1$, and lattitude between y and $y + 1$, where x and y are integers. For simplicity, we assume that lattitude and longitude are computed from coordinates by dividing by 10,000 (the true conversion is far more difficult).
    public Resultset queryLongLat(int x, int y)

17. Find the name of every driver who picked up a parcel ordered by some some customer with ID `cid` on date `d`.
    public ResultSet queryCustomerDrivers(int cid, Date d)

18. Find, for every customer, the price and delivery address of the most expensive parcel sent. Report the result sorted by price in decreasing order.
    public ResultSet queryBiggestCustomers()

To be handed in

A pdf file with the following:

- Your data model (indicating any changes from the second deliverable).
- All 18 SQL queries (indicating any queries from the second deliverable that you revised). Please submit also the java files, as a separate attachment.

- A description of the indexes you created (attributes and type).
- For every query, a description of the query plan chosen by Oracle. You should not copy the plan from the browser window, but only state (in words) the most important choices made: What indexes are used, in what order, what join methods are used, etc.
- For every query, a brief discussion of the query plan: What are the competing alternatives (if any)? Have any of them been tried, and what were the results? Is there any change that could make this query faster, but was not chosen because of other constraints?