CSE 135 Project Indices Report

Please note: The queries utilized by the Sales Analytics Page can be referred to on *SalesDAO.java* and *ProductDAO.java*. A simplified compilation was also created in the sqlScripts folder titled *sql\_sales\_analysis\_queries.sql*. Each query is labelled by a number according to that .sql file and will be referred to as such in the following.

**Listing of Possibly Beneficial Indices**:

For the following queries, it might be beneficial to index on…

*Query 1*:

* state table’s state\_name
* person table’s state\_id
* products\_in\_cart table’s product\_id
  + cart\_id
* shopping\_cart table’s person\_id
* product table’s category\_id

*Query 2*:

* products\_in\_cart table’s product\_id

*Query 3*: This is basically Query 2 but taking all products even if they have null prices thus the same index applies. It also applies a filter thus it might be simplified with an index on:

* product table’s category\_id

*Query 4*:

* person table’s state\_id
* products\_in\_cart table’s product\_id
  + cart\_id
* shopping\_cart table’s person\_id

*Query 5*:

* shopping\_cart table’s person\_id
* products\_in\_cart table’s product\_id
  + cart\_id

*Query 6*: The above query along with:

* product table’s category\_id

*Query 7*:

* shopping\_cart table’s person\_id
* products\_in\_cart table’s product\_id
  + cart\_id

**Index Testing:** Notes: Referring to indices listed in *indices.sql*. Testing was done via pgadmin and the data generator was used for each case. For Case 1 (small and hot), I generated 20 customers, 15 categories, 15 products, and 20 sales. For Case 2 (large and cold), I generated 5000 customers, 2700 categories, 4950 products, and 7280 sales. I averaged times over 5 runs.

*Index 1:*

Case 1:

a) running time

b) individual query times:

i)

ii)

iii)

iv)

v)

vi)

vii)

Case 2:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

*Index 2:*

Case 1:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

Case 2:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

*Index 3:*

Case 1:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

Case 2:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

*Index 4:*

Case 1:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

Case 2:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

*Index 5:*

Case 1:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

Case 2:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

*Index 6:*

Case 1:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

Case 2:

a) running time

b) individual query

i)

ii)

iii)

iv)

v)

vi)

vii)

o For each of the two cases report (a) running time of the overall jsp, (b) running time of the individual queries and (c) your index choices for these two cases.

▪ In the two extreme cases, adjust the Postgres sequential-Vs-random access ratio  
accordingly.  
  
o For each index Y that you are sure that it is beneficial, you need not contact any  
experiment to verify its usefulness. Just mark that you are sure and provide your  
reasoning. Nevertheless, debug: Use EXPLAIN to check whether some query (or queries)  
of your program indeed used the index Y

o Next, for each candidate index in the indices’ report where you are not sure whether it  
actually benefits performance, do an experiment. The simple way to experiment if an  
index *X* is beneficial is to first run the “Sales Analytics” page without having created the  
index *X* and then create the index *X* and run the page again. If you see no performance  
difference, the index was not worthy. Be careful of caching effects when you execute  
such experiments. It is very possible that an index may useful in the “small and hot” but  
not in the “large and cold” and vice versa.  
  
  
  
  
• You will need to provide the following and argue for the appropriateness of your solution:  
o Code  
o Best index choice.  
o For each query of your program, provide its running time for the small and the large  
database. To avoid experimental flukes, average over at least 3 runs.  
o The indices’ report: candidate indices, which one of them are the indices you are sure  
that are beneficial (and your reasoning), which ones you experimented with in order to  
decide whether they are (or they are not) useful, the experimentation results.