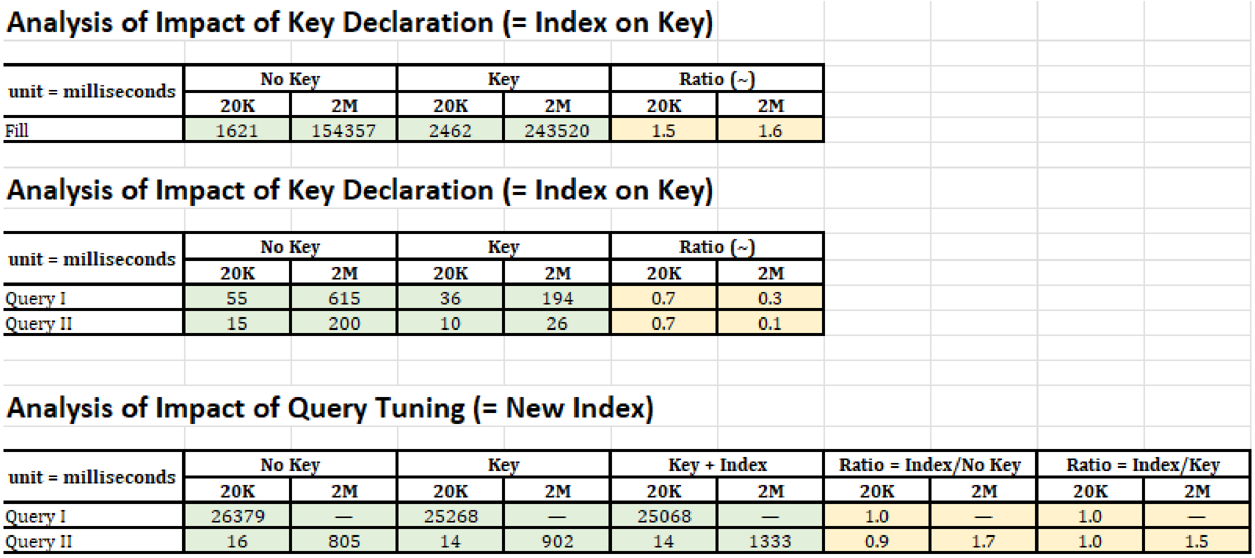
Exercise 10, indexes

Elísabet Jóhannesdóttir, Eva Natalía, Ómar Bessi



Analysis of Impact of Key Declaration (= Index on Key)

unit = milliseconds	No Key		Key		Ratio (~)	
	20K	2M	20K	2M	20K	2M
Fill	1621	154357	2462	243520	1.5	1.6

Analysis of Impact of Key Declaration (= Index on Key)

unit = milliseconds	No Key		Key		Ratio (~)	
	20K	2M	20K	2M	20K	2M
Query I	55	615	36	194	0.7	0.3
Query II	15	200	10	26	0.7	0.1

Analysis of Impact of Query Tuning (= New Index)

unit = milliseconds	No Key		Key		Key + Index		Ratio = Index/No Key		Ratio = Index/Key	
	20K	2M	20K	2M	20K	2M	20K	2M	20K	2M
Query I	26379	—	25268	—	25068	—	1.0	—	1.0	—
Query II	16	805	14	902	14	1333	0.9	1.7	1.0	1.5

Part II:

- Explanation for the difference in Query I and Query II:

It is because peopleID comes before sportID in the composite key (in the DDL) so searching is much faster since they first go through peopleID.

- Explanation for why the queries run faster in E10KS:

Because when there are no specified primary keys the DBMS will have to determine which is the primary key compared to specifying them in the DDL.

- The same without indexes?

With indexes:

Query I would be slower as that would be unclustered.

Query II would be the same as the search is clustered and an index would actually be covering.

Part III:

- Which query/database combo can complete within a reasonable time?

As seen in our analysis we can see that Query II ran at a much much faster time than Query I in all databases.

- Why is the execution faster in the E10KS database?

Because if there are no keys the DBMS make its own keys, taking a much longer time to process the tables vs where keys have already been determined.

- Why version II is much faster? Does it have anything to do with indexes?

Query I uses a subquery for each row while Query II as much better performance because of the group by. This groups the results by sportID first and then selecting the max only once.