Assignment 5

Name: Yinsheng Dong

Student Number: 11148648

NSID: yid164

Lecture: CMPT 355

Part 1

a. Fill out tables

Index 1

Execution Time	Without index	With index
Query 1	0.623 ms	0.688 ms
Query 2	1.236 ms	1.233 ms

Index 2

Execution Time	Without index	With index
Query 3	2.069 ms	0.447 ms

Index 3

Execution Time	Without index	With index
Query 4	27.407 ms	19.434 ms

b. Did adding the index change the explain plans? What changed?

Answer: Yes, adding the index changed the explain plans in query 3 and query 4, but no effective in query 1 and query 2.

1. The query 1 in index and without index, the explains are same.
2. The query 2 in index and without index, the explains are same.
3. In the query 3 without the index, the query executed by a nested loop, join filter by employee_status_type_id in employees table equal to primary key of employee_status_type table, then sequential scanning in employees table and materialize. But in the query 3 with the index, the explain plan is different from before. It used the hash join the two tables, and use the bitmap heap for scanning, then hash them to get the result.
4. In the query 4 without the index, the query executed by merge join, merge join condition by id from employees table equals to employee_jobs's employee_id, and use the nest loop and sort them, and it finally aggregate the data. In the query 4 with the index executed by nested loop. It firstly uses hash join by the conditions those are employee_id in employee_jobs table equal to id in the employees table and effective_date equals to sub-plan, the sub-plan is the aggregate the index_2 on employee_jobs ej2, and use hash to sequence scan on the employee e, in the second part it use the index scan to get employment_status_types_pkey on employment_status_types es.

c. Was this what you expected happen for the timing and the execution plans? What is a possible reason for this change (or lack of change)?

Answer: Yes, there as something expected happen for timing and the execution plans.

1. There were no execution plans change between query 1 with index and without index, they both use sequential scan in to search table, and the execution time got pretty similar, and I think the query with index even takes longer than the query without index, because there was no expected execution plan changed.
2. There were also no execution plans change between query 2 with index and without index, similar as query 1, they both use sequential scan in to search table, and the execution time got very similar, and I think the query with index even takes longer than the query without index, because there was no expected execution plan changed.
3. In the query 3 with index, it uses the hash join for the execution plan, it uses the bitmap index scan on the employees table. It is efficient way than the query without index. And the actually execution time is faster than the query without index. The possible reason is this is the query is using for larger table, but the nested loop is not efficient for this query, that is why the index changed the way for the query.
4. In the query 4 with index, it uses the nested loop in the execution plan instead the merge-join in the without index query. According to the result (the execution time), the nested loop is faster than the merge-join in this condition. I think the possible reason for this change is because of the merge join is using for the larger table, and it before the join, both tables are sorted by the join attribute, but this way will waste

time, so the with index way transfer it to default as nested loop, in the smaller tables, it use the inner sequential scan to help the efficient.

Part 2

1st normal:

Team Member Id	Team Member First Name	Team Member Last Name	Project Code	Project Name	Project Status	Project Manager	Task Number	Task Status
1	John	Smith	DDL	Darren & Darren Ltd	Active	Garth Butler	10	Resolved
1	John	Smith	DDL	Darren & Darren Ltd	Active	Garth Butler	132	In Progress
1	John	Smith	DDL	Darren & Darren Ltd	Active	Garth Butler	133	Not Started
1	John	Smith	DDL	Darren & Darren Ltd	Active	Garth Butler	134	In Progress
2	Dave	Richter	DDL	Darren & Darren Ltd	Active	Garth Butler	100	In Progress
2	Dave	Richter	DDL	Darren & Darren Ltd	Active	Garth Butler	110	Not Started
2	Dave	Richter	KMI	Kristen Motors Inc.	Active	Jim David	10	Not Started
2	Dave	Richter	KMI	Kristen Motors Inc.	Active	Jim David	13	Resolved
3	Janie	Klotter	KMI	Kristen Motors Inc.	Active	Jim David	1	In Progress
3	Janie	Klotter	KMI	Kristen Motors Inc.	Active	Jim David	2	Resolved
3	Janie	Klotter	KMI	Kristen Motors Inc.	Active	Jim David	15	Resolved

2nd Normal

Project Relation

Project Code	Project Name	Project Status	Project Manager
DDL	Darren & Darren Ltd	Active	Garth Butler
KMI	Kristen Motors Inc.	Active	Jim David

Task Relation

Project Code	Task Number	Task Status	Team Member Id
DDL	10	Resolved	1
DDL	132	In Progress	1
DDL	133	Not Started	1
DDL	134	In Progress	1
DDL	100	In Progress	2
DDL	110	Not Started	2
KMI	10	Not Started	2
KMI	13	Resolved	2
KMI	1	In Progress	3

KMI	2	Resolved	3
KMI	15	Resolved	3

Team Member Relation

Project Code	Task Number	Team Member Id	Team Member First Name	Team Member Last Name
DDL	10	1	John	Smith
DDL	132	1	John	Smith
DDL	133	1	John	Smith
DDL	134	1	John	Smith
DDL	100	2	Dave	Richter
DDL	110	2	Dave	Richter
KMI	10	2	Dave	Richter
KMI	13	2	Dave	Richter
KMI	1	3	Janie	Klotter
KMI	2	3	Janie	Klotter
KMI	15	3	Janie	Klotter

3rd Normal

Project Relation

Project Code	Project Name	Project Status	Project Manager
DDL	Darren & Darren Ltd	Active	Garth Butler
KMI	Kristen Motors Inc.	Active	Jim David

Task Relation

Project Code	Task Number	Task Status	Team Member Id
DDL	10	Resolved	1
DDL	132	In Progress	1
DDL	133	Not Started	1
DDL	134	In Progress	1
DDL	100	In Progress	2
DDL	110	Not Started	2
KMI	10	Not Started	2
KMI	13	Resolved	2
KMI	1	In Progress	3
KMI	2	Resolved	3

KMI	15	Resolved	3

Team Member Relation

Team Member Id	Team Member First Name	Team Member Last Name
1	John	Smith
2	Dave	Richter
3	Janie	Klotter

Project-Task-Team Member Relation

Project Code	Task Number	Team Member Id
DDL	10	1
DDL	132	1
DDL	133	1
DDL	134	1
DDL	100	2
DDL	110	2
KMI	10	2
KMI	13	2
KMI	1	3

KMI	2	3
KMI	15	3

Part 3: Concurrency

1. Scenario – Transaction A and B are being run concurrently in separate sessions
    a. What would the Accounts table look like after these transactions are finished?

Account Number	Account Nickname	Account Balance
1	Chequing	350
2	Chequing	100

    b. What type of data inconsistency is caused in this case?
    Answer: This is the dirty read inconsistency and lost update issue are caused in this case, because while the transaction B is using select to read wrong data while transaction A is update, it is caused the dirty read, and after B transaction updated and B and A commit in same time, A and B will lost update.
2. Transaction C and D
    a. Answer: This one should be non-repeatable read and phantom read, because transaction C re-read data that has been previously read, but transaction D changed the data in the meant time, also transaction D insert a newly row into the table that was seen in the transaction C first time.
3. Transaction E and F
    a. Answer: This one should be dirty read and lost update, at the transaction F first query, transaction F first query read the uncommitted data from transaction E, in this part, the dirty read happened. After the transaction E ROLLBACK, the data went back to original, but after this, transaction F insert data to the table, it will lose data from transaction E, so in this part, the lost update happened.