Name : Kreena Shah
Sapid : 60004210243
Batch : Comps C’32
Subject : Advance Database Management System (ADBMS)

# Experiment No 2 - Query Optimization

Aim : Simulate Query optimization by applying an SQL Query on any database

Write Ups :

Kreena Shah
60004210243
C'32

19/10/2023 ADBMS Exp 2 9 12/15 Page No. Date Submission: 26/10/2023

Aim : Simulate Query Optimization By applying an SQL query on any database

Theory :

| Sr No | Query | Normal | Optimized | Indexed |
|---|---|---|---|---|
| | SELECT COUNT(*) FROM user_details ; | 0.000166 | 0.0019 | 0.000157 |
| | SELECT AVG(user_id) FROM user_details ; | 0.1495 | 0.0174 | 0.0167 |
| | SELECT * FROM user_details ORDER BY last_name DESC | 0.5025 | 0.50426 | 0.1218 |
| | SELECT * FROM user_details WHERE password IN ( SELECT password FROM user_details WHERE password LIKE '2%'); | 0.9533 | 0.9581 | 0.1625 |
| | SELECT u1.username AS user1_username, u2.username AS user2_username, FROM User_details u1 JOIN user_details u2 ON u1.first_name = u2.first_name AND u1.user_id = u2.user_id | 1.0647 | 1.065 | 1.060 |

Page No.
Date

## SQL Query Optimization

SQL query optimization is defined as the iterative process of enhancing the performance of a query in terms of execution time, number of disk accesses & other cost measuring criteria

## Optimization of Queries

Optimization in the context of database & query performance involves refining & finetuning various aspects of database operations to make them more efficient. This process aims to reduce query execution time, enhances system responsiveness & optimizes resource utilization.

Optimization can include rewriting queries to improve their logic, creating or adjusting indexes on database tables, optimizing data storage & finetuning database configuration. Optimization results improves system performance, reduced response time, better scalability & ensures that the database can handle heavy workloads efficiently

## Indexing in Queries

Indexing is a fundamental database optimization technique that enhances query performance by efficiently organizing & accessing data. Instead of scanning the entire dataset, indexes provide a better & optimized way to accem the desired data records

There are various types of indexes

(1) Primary Indexing

(2) Clustered Indexing
(3) Secondary / Non-clustered Indexing
(4) Multilevel Indexing
Properly implemented indexing can significantly enhance database performance.

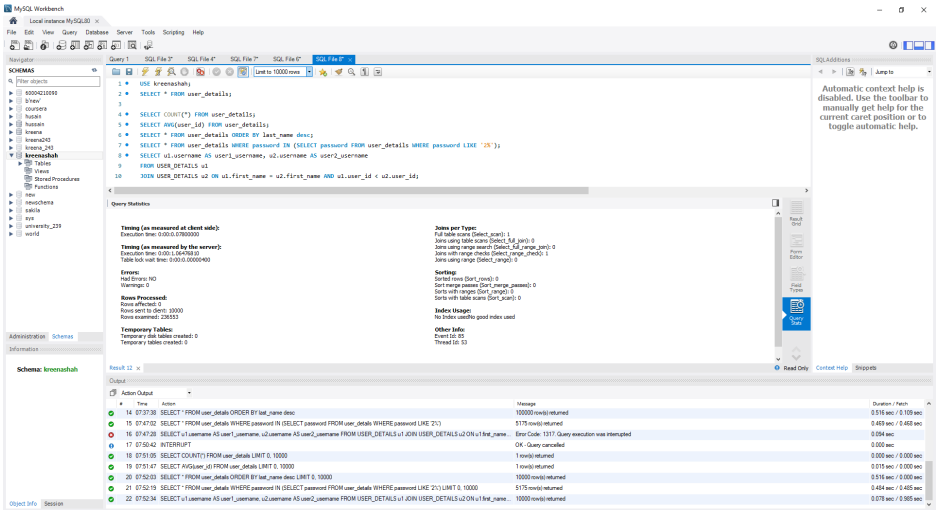
Observations :
The mentioned query table provides performance for a set of SQL queries of a normal select query, using aggregate functions, using ORDER BY for ordering on specific attribute, nested queries & using JOINS.
We implemented same set of queries by running queries, using optimization & then by indexing. After optimization of table, not much effect was observed. But after indexing, significant decrease in response time was observed

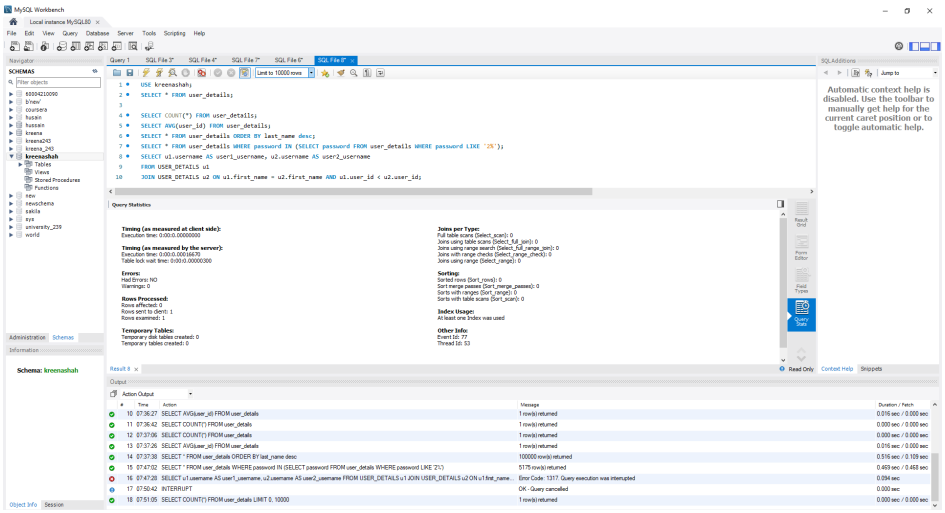
Conclusion :
The experiment concluded that optimization efforts improved query performance to varying degrees, with some queries showing significant gains while other has modest improvements.
Continuous monitoring & trade off considerations are essential for achieving desired performance

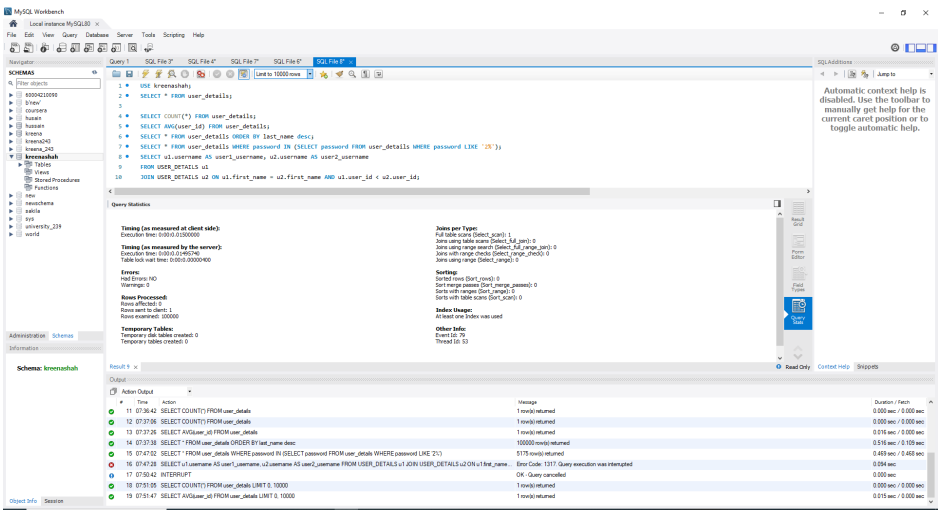


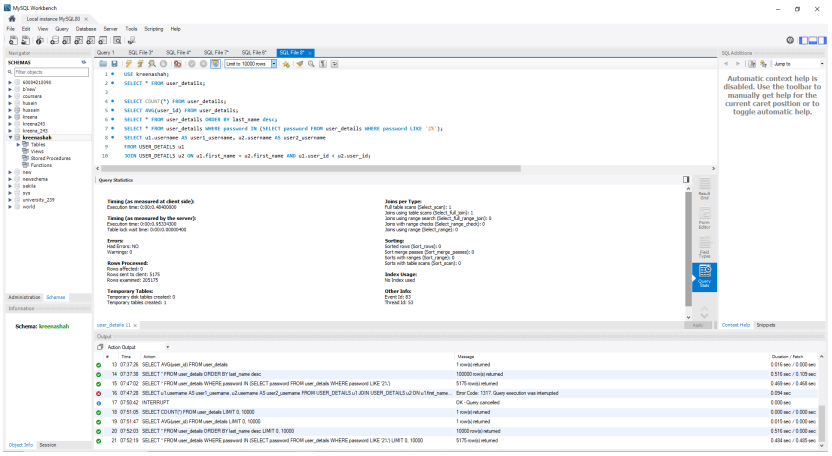
Screenshots :

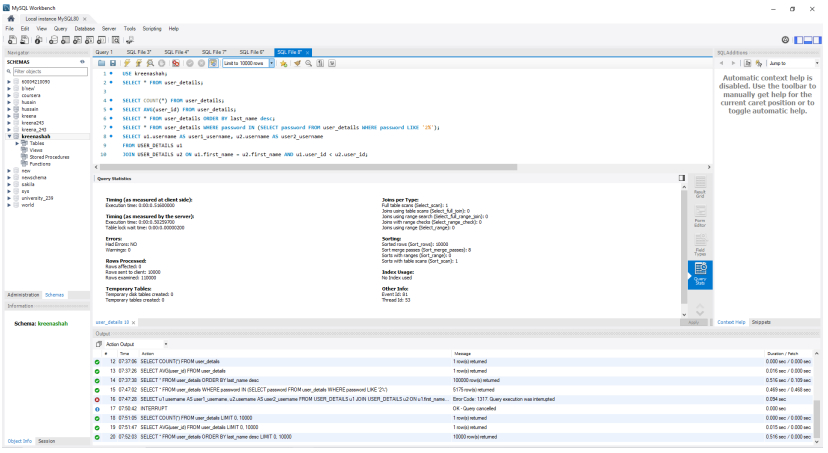
Simple Query


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
kreenashah
Tables
Views
Stored Procedures
Functions
Administration Schemas
Information
Schema: kreenashah
Query 1
SQL File 3*
SQL File 4*
SQL File 7*
SQL File 6*
SQL File 8*
Limit to 10000 rows
1 USE kreenashah;
2 SELECT * FROM user_details;
3
4 SELECT COUNT(*) FROM user_details;
5 SELECT AVG(user_id) FROM user_details;
6 SELECT * FROM user_details ORDER BY last_name desc;
7 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%');
8 SELECT u1.username AS user1_username, u2.username AS user2_username
9 FROM USER_DETAILS u1
10 JOIN USER_DETAILS u2 ON u1.first_name = u2.first_name AND u1.user_id < u2.user_id;
SQL Additions
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Query Statistics
Timing (as measured at client side):
Execution time: 0.000.0.07800000
Timing (as measured by the server):
Execution time: 0.000.1.064768.0
Table lock wait time: 0.000.0.00000400
Errors:
Had Errors: NO
Warnings: 0
Rows Processed:
Rows affected: 0
Rows sent to client: 10000
Rows examined: 236553
Temporary Tables:
Temporary disk tables created: 0
Temporary tables created: 0
Joins per Type:
Full table scans (Select_scan): 1
Joins using table scans (Select_full_join): 0
Joins using range search (Select_full_range_join): 0
Joins with range checks (Select_range_check): 0
Joins using range (Select_range): 0
Sorting:
Sorted rows (Sort_rows): 0
Sort merge passes (Sort_merge_passes): 0
Sorts with ranges (Sort_range): 0
Sorts with table scans (Sort_scan): 0
Index Usage:
No Index usedNo good index used
Other Info:
Event Id: 85
Thread Id: 53
Result 12
Read Only
Context Help
Snippets
Output
Action Output
# Time Action Message Duration / Fetch
14 07:37:38 SELECT * FROM user_details ORDER BY last_name desc 100000 row(s) returned 0.516 sec / 0.109 sec
15 07:47:02 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%') 5175 row(s) returned 0.469 sec / 0.468 sec
16 07:47:28 SELECT u1.username AS user1_username, u2.username AS user2_username FROM USER_DETAILS u1 JOIN USER_DETAILS u2 ON u1.first_name... Error Code: 1317. Query execution was interrupted 0.094 sec
17 07:50:42 INTERRUPT OK. Query cancelled 0.000 sec
18 07:51:05 SELECT COUNT(*) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.000 sec / 0.000 sec
19 07:51:47 SELECT AVG(user_id) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.015 sec / 0.000 sec
20 07:52:03 SELECT * FROM user_details ORDER BY last_name desc LIMIT 0, 10000 10000 row(s) returned 0.516 sec / 0.000 sec
21 07:52:19 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%') LIMIT 0, 10000 5175 row(s) returned 0.484 sec / 0.485 sec
22 07:52:34 SELECT u1.username AS user1_username, u2.username AS user2_username FROM USER_DETAILS u1 JOIN USER_DETAILS u2 ON u1.first_name... 10000 row(s) returned 0.078 sec / 0.985 sec
Object Info Session

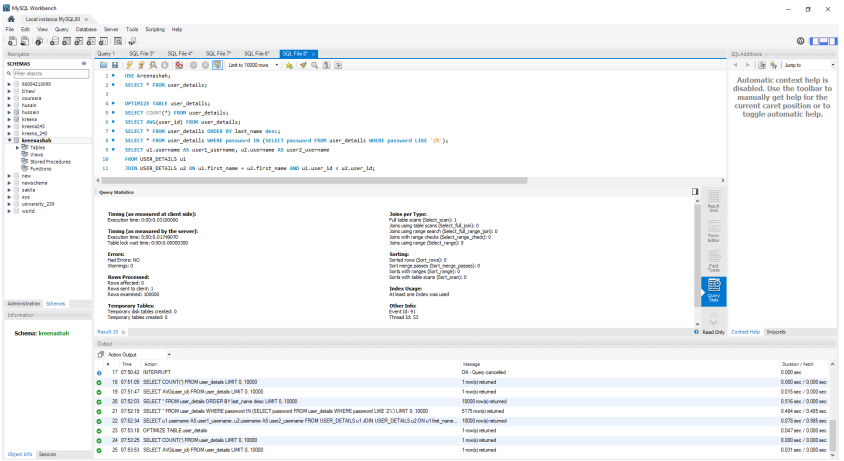

MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
kreenashah
Tables
Views
Stored Procedures
Functions
Administration Schemas
Information
Schema: kreenashah
Query 1
SQL File 3*
SQL File 4*
SQL File 7*
SQL File 6*
SQL File 8*
Limit to 10000 rows
1 USE kreenashah;
2 SELECT * FROM user_details;
3
4 SELECT COUNT(*) FROM user_details;
5 SELECT AVG(user_id) FROM user_details;
6 SELECT * FROM user_details ORDER BY last_name desc;
7 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%');
8 SELECT u1.username AS user1_username, u2.username AS user2_username
9 FROM USER_DETAILS u1
10 JOIN USER_DETAILS u2 ON u1.first_name = u2.first_name AND u1.user_id < u2.user_id;
SQL Additions
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Query Statistics
Timing (as measured at client side):
Execution time: 0.000.0.00000000
Timing (as measured by the server):
Execution time: 0.000.0.000166700
Table lock wait time: 0.000.0.00000300
Errors:
Had Errors: NO
Warnings: 0
Rows Processed:
Rows affected: 0
Rows sent to client: 1
Rows examined: 1
Temporary Tables:
Temporary disk tables created: 0
Temporary tables created: 0
Joins per Type:
Full table scans (Select_scan): 0
Joins using table scans (Select_full_join): 0
Joins using range search (Select_full_range_join): 0
Joins with range checks (Select_range_check): 0
Joins using range (Select_range): 0
Sorting:
Sorted rows (Sort_rows): 0
Sort merge passes (Sort_merge_passes): 0
Sorts with ranges (Sort_range): 0
Sorts with table scans (Sort_scan): 0
Index Usage:
At least one Index was used
Other Info:
Event Id: 77
Thread Id: 53
Result 8
Read Only
Context Help
Snippets
Output
Action Output
# Time Action Message Duration / Fetch
10 07:36:27 SELECT AVG(user_id) FROM user_details 1 row(s) returned 0.016 sec / 0.000 sec
11 07:36:42 SELECT COUNT(*) FROM user_details 1 row(s) returned 0.000 sec / 0.000 sec
12 07:37:06 SELECT COUNT(*) FROM user_details 1 row(s) returned 0.000 sec / 0.000 sec
13 07:37:26 SELECT AVG(user_id) FROM user_details 1 row(s) returned 0.016 sec / 0.000 sec
14 07:37:38 SELECT * FROM user_details ORDER BY last_name desc 100000 row(s) returned 0.516 sec / 0.109 sec
15 07:47:02 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%') 5175 row(s) returned 0.469 sec / 0.468 sec
16 07:47:28 SELECT u1.username AS user1_username, u2.username AS user2_username FROM USER_DETAILS u1 JOIN USER_DETAILS u2 ON u1.first_name... Error Code: 1317. Query execution was interrupted 0.094 sec
17 07:50:42 INTERRUPT OK. Query cancelled 0.000 sec
18 07:51:05 SELECT COUNT(*) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.000 sec / 0.000 sec
Object Info Session

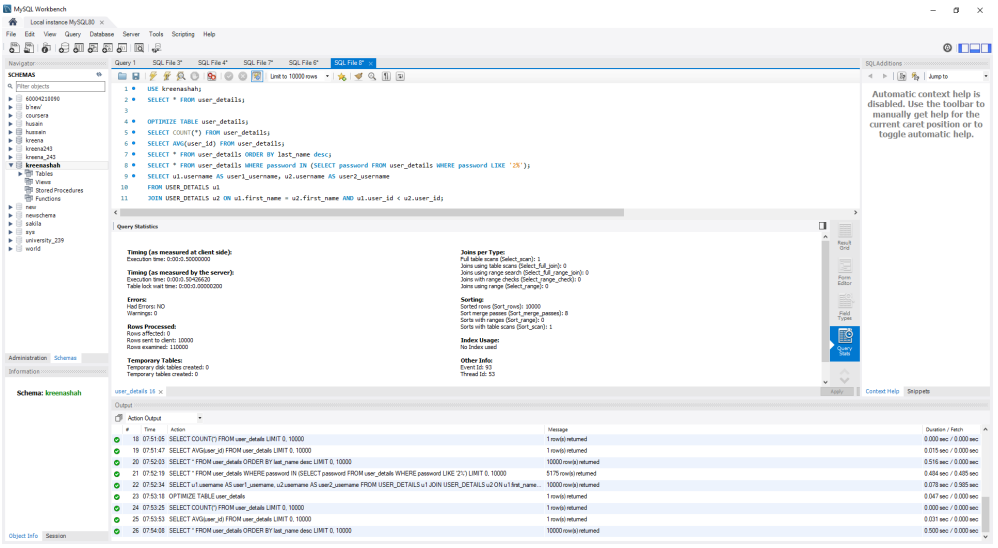

MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
kreenashah
Tables
Views
Stored Procedures
Functions
Administration Schemas
Information
Schema: kreenashah
Query 1
SQL File 3*
SQL File 4*
SQL File 7*
SQL File 6*
SQL File 8*
Limit to 10000 rows
1 USE kreenashah;
2 SELECT * FROM user_details;
3
4 SELECT COUNT(*) FROM user_details;
5 SELECT AVG(user_id) FROM user_details;
6 SELECT * FROM user_details ORDER BY last_name desc;
7 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%');
8 SELECT u1.username AS user1_username, u2.username AS user2_username
9 FROM USER_DETAILS u1
10 JOIN USER_DETAILS u2 ON u1.first_name = u2.first_name AND u1.user_id < u2.user_id;
SQL Additions
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Query Statistics
Timing (as measured at client side):
Execution time: 0.000.0.03500000
Timing (as measured by the server):
Execution time: 0.000.0.01495740
Table lock wait time: 0.000.0.00000400
Errors:
Had Errors: NO
Warnings: 0
Rows Processed:
Rows affected: 0
Rows sent to client: 1
Rows examined: 100000
Temporary Tables:
Temporary disk tables created: 0
Temporary tables created: 0
Joins per Type:
Full table scans (Select_scan): 1
Joins using table scans (Select_full_join): 0
Joins using range search (Select_full_range_join): 0
Joins with range checks (Select_range_check): 0
Joins using range (Select_range): 0
Sorting:
Sorted rows (Sort_rows): 0
Sort merge passes (Sort_merge_passes): 0
Sorts with ranges (Sort_range): 0
Sorts with table scans (Sort_scan): 0
Index Usage:
At least one Index was used
Other Info:
Event Id: 79
Thread Id: 53
Result 9
Read Only
Context Help
Snippets
Output
Action Output
# Time Action Message Duration / Fetch
11 07:36:42 SELECT COUNT(*) FROM user_details 1 row(s) returned 0.000 sec / 0.000 sec
12 07:37:06 SELECT COUNT(*) FROM user_details 1 row(s) returned 0.000 sec / 0.000 sec
13 07:37:26 SELECT AVG(user_id) FROM user_details 1 row(s) returned 0.016 sec / 0.000 sec
14 07:37:38 SELECT * FROM user_details ORDER BY last_name desc 100000 row(s) returned 0.516 sec / 0.109 sec
15 07:47:02 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%') 5175 row(s) returned 0.469 sec / 0.468 sec
16 07:47:28 SELECT u1.username AS user1_username, u2.username AS user2_username FROM USER_DETAILS u1 JOIN USER_DETAILS u2 ON u1.first_name... Error Code: 1317. Query execution was interrupted 0.094 sec
17 07:50:42 INTERRUPT OK. Query cancelled 0.000 sec
18 07:51:05 SELECT COUNT(*) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.000 sec / 0.000 sec
19 07:51:47 SELECT AVG(user_id) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.015 sec / 0.000 sec
Object Info Session

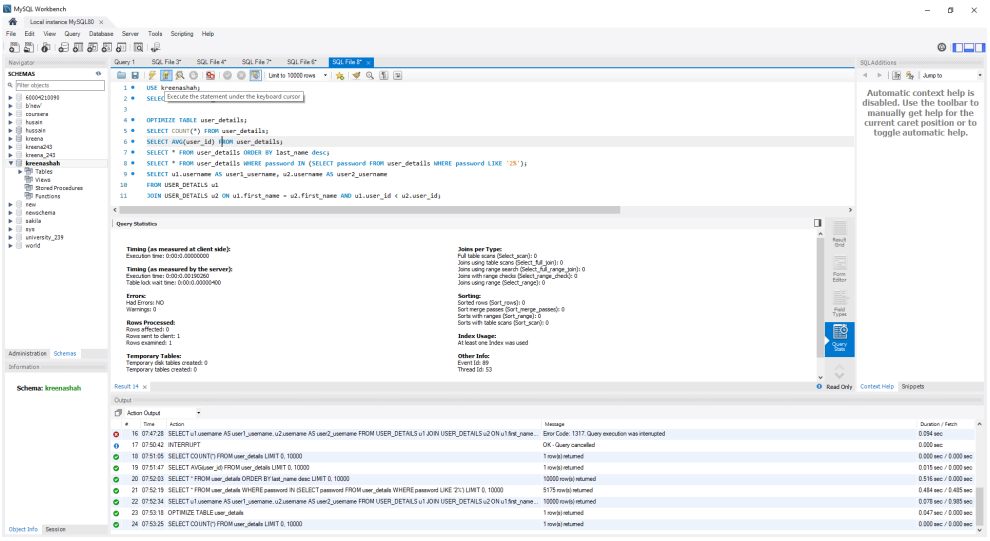


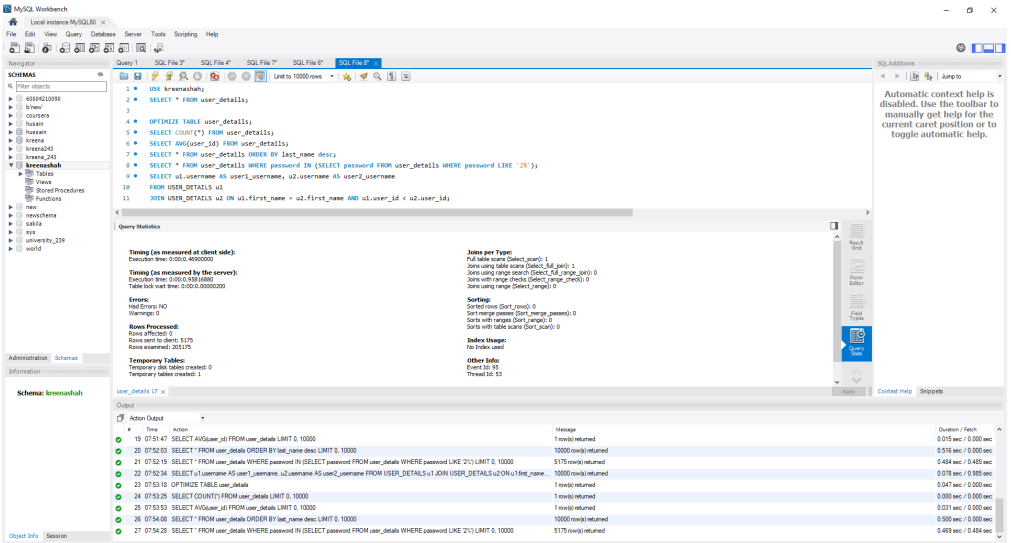


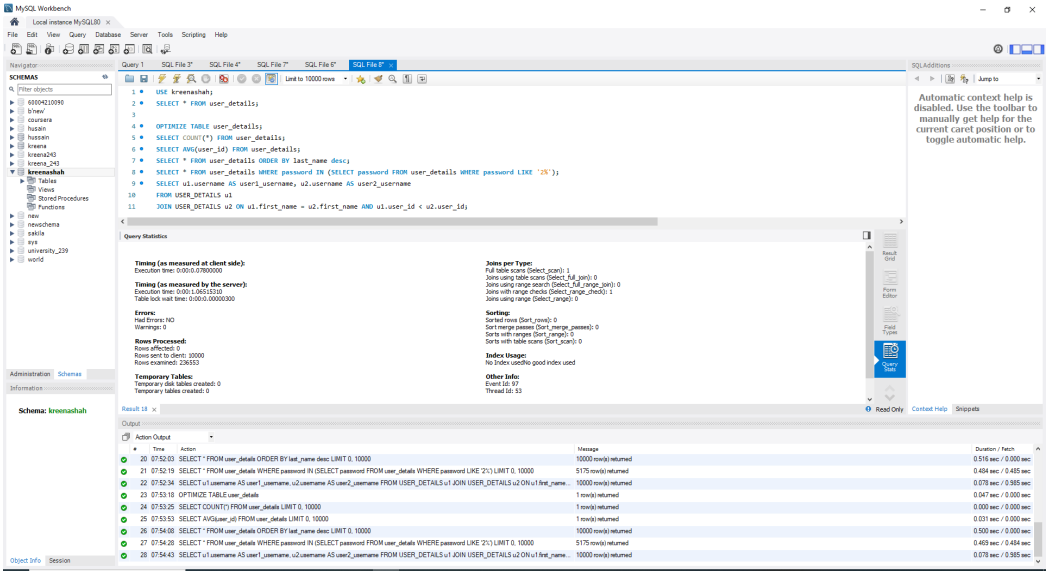
# Optimized Query



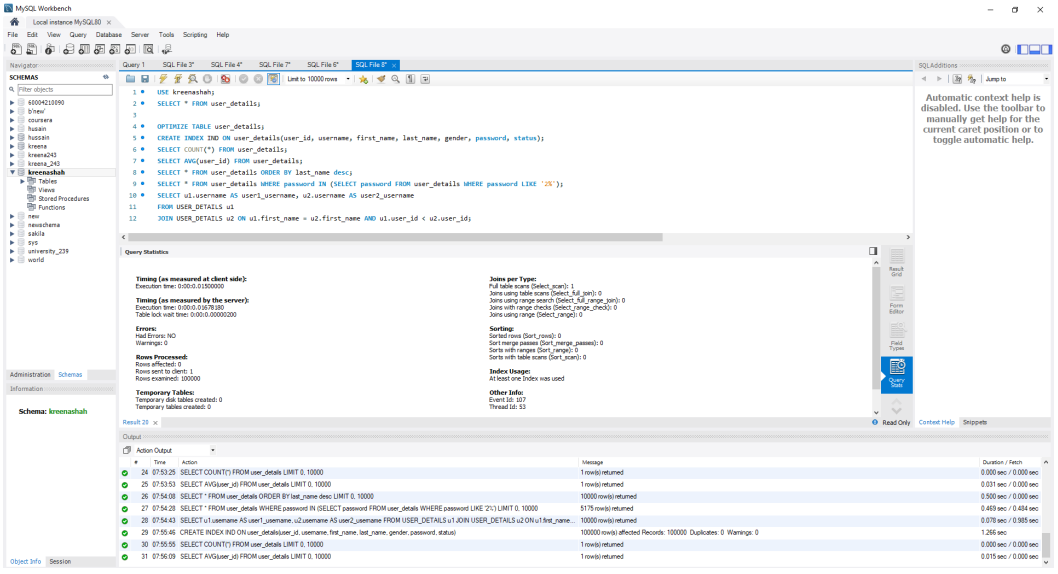

MySQL Workbench
USE kreenashah;
SELECT * FROM user_details;
OPTIMIZE TABLE user_details;
SELECT COUNT(*) FROM user_details;
SELECT AVG(user_id) FROM user_details;
SELECT * FROM user_details ORDER BY last_name desc;
SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%');
SELECT u1.username AS user1_username, u2.username AS user2_username
FROM USER_DETAILS u1
JOIN USER_DETAILS u2 ON u1.first_name = u2.first_name AND u1.user_id < u2.user_id;
Schema: kreenashah

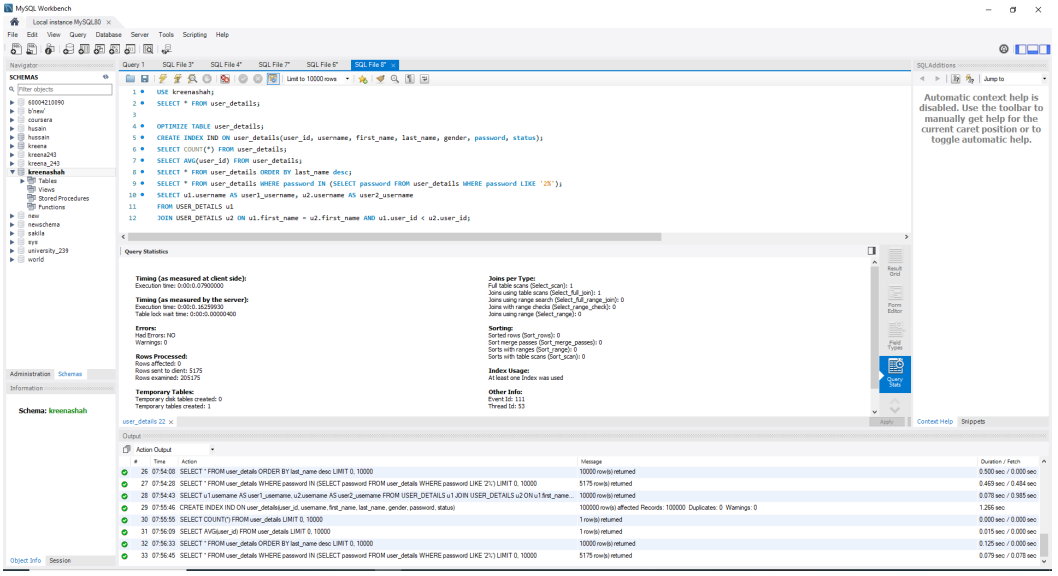

MySQL Workbench
USE kreenashah;
SELECT * FROM user_details;
OPTIMIZE TABLE user_details;
SELECT COUNT(*) FROM user_details;
SELECT AVG(user_id) FROM user_details;
SELECT * FROM user_details ORDER BY last_name desc;
SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%');
SELECT u1.username AS user1_username, u2.username AS user2_username
FROM USER_DETAILS u1
JOIN USER_DETAILS u2 ON u1.first_name = u2.first_name AND u1.user_id < u2.user_id;
Schema: kreenashah

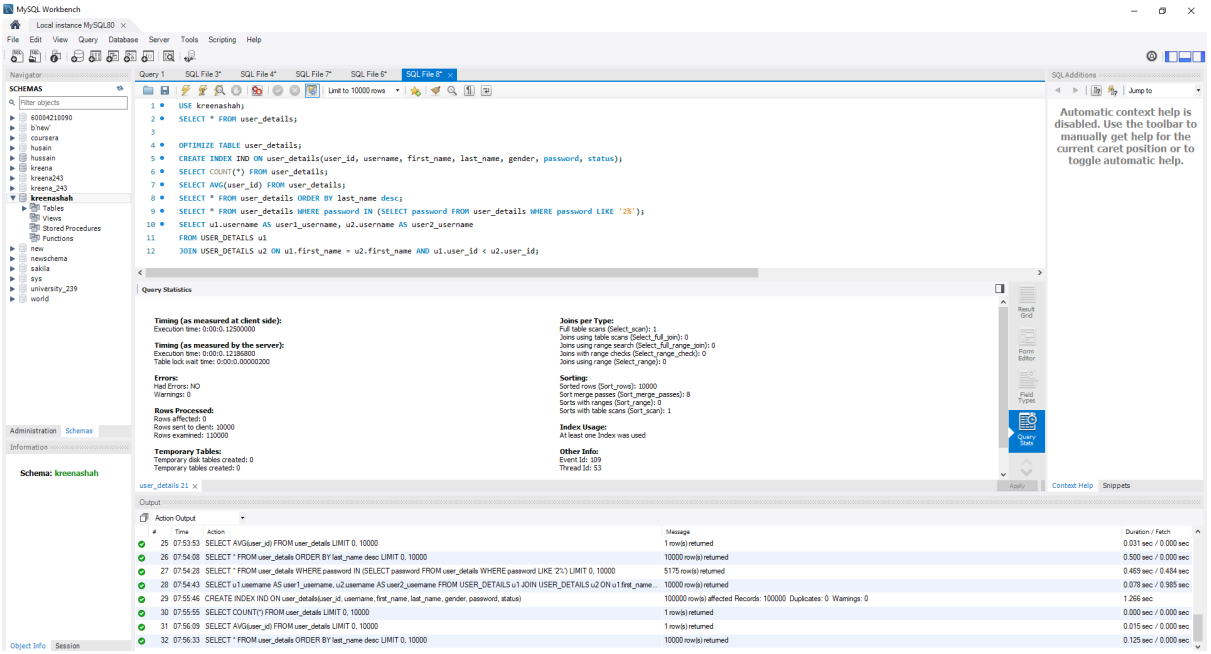

MySQL Workbench
USE kreenashah;
SELECT * FROM user_details;
OPTIMIZE TABLE user_details;
SELECT COUNT(*) FROM user_details;
SELECT AVG(user_id) FROM user_details;
SELECT * FROM user_details ORDER BY last_name desc;
SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%');
SELECT u1.username AS user1_username, u2.username AS user2_username
FROM USER_DETAILS u1
JOIN USER_DETAILS u2 ON u1.first_name = u2.first_name AND u1.user_id < u2.user_id;
Schema: kreenashah

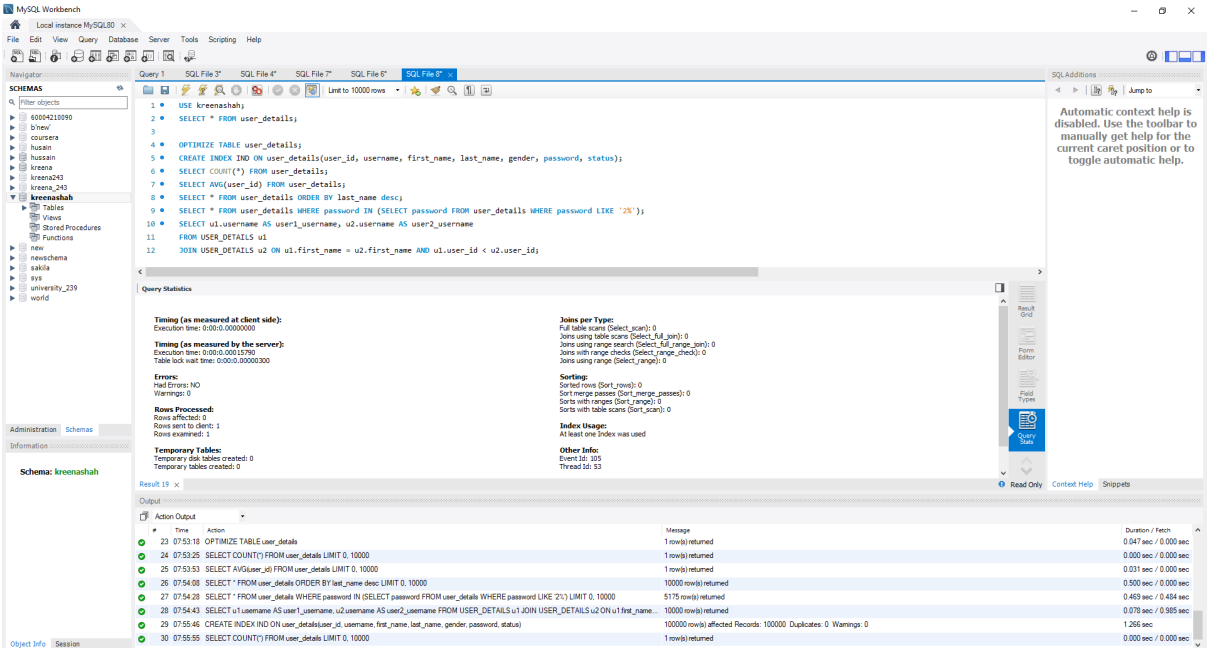



# Indexed Query




MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
60004210090
b'new'
coursera
husain
hussain
kreena
kreena243
kreena_243
kreenashah
Tables
Views
Stored Procedures
Functions
new
newschema
sakila
sys
university_239
world
Administration Schemas
Information
Schema: kreenashah
Object Info Session
Query 1 SQL File 3* SQL File 4* SQL File 7* SQL File 6* SQL File 8*
Limit to 10000 rows
1 USE kreenashah;
2 SELECT * FROM user_details;
3
4 OPTIMIZE TABLE user_details;
5 CREATE INDEX IND ON user_details(user_id, username, first_name, last_name, gender, password, status);
6 SELECT COUNT(*) FROM user_details;
7 SELECT AVG(user_id) FROM user_details;
8 SELECT * FROM user_details ORDER BY last_name desc;
9 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%');
10 SELECT u1.username AS user1_username, u2.username AS user2_username
11 FROM USER_DETAILS u1
12 JOIN USER_DETAILS u2 ON u1.first_name = u2.first_name AND u1.user_id < u2.user_id;
Query Statistics
Timing (as measured at client side):
Execution time: 0:00:0.12500000
Timing (as measured by the server):
Execution time: 0:00:0.12186800
Table lock wait time: 0:00:0.00000200
Errors:
Had Errors: NO
Warnings: 0
Rows Processed:
Rows affected: 0
Rows sent to client: 10000
Rows examined: 110000
Temporary Tables:
Temporary disk tables created: 0
Temporary tables created: 0
Joins per Type:
Full table scans (Select_scan): 1
Joins using table scans (Select_full_join): 0
Joins using range search (Select_full_range_join): 0
Joins with range checks (Select_range_check): 0
Joins using range (Select_range): 0
Sorting:
Sorted rows (Sort_rows): 10000
Sort merge passes (Sort_merge_passes): 8
Sorts with ranges (Sort_range): 0
Sorts with table scans (Sort_scan): 1
Index Usage:
At least one Index was used
Other Info:
Event Id: 109
Thread Id: 53
Result Grid
Form Editor
Field Types
Query Stats
Apply
user_details 21
SQL Additions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
# Time Action Message Duration / Fetch
25 07:53:53 SELECT AVG(user_id) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.031 sec / 0.000 sec
26 07:54:08 SELECT * FROM user_details ORDER BY last_name desc LIMIT 0, 10000 10000 row(s) returned 0.500 sec / 0.000 sec
27 07:54:28 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%') LIMIT 0, 10000 5175 row(s) returned 0.469 sec / 0.484 sec
28 07:54:43 SELECT u1.username AS user1_username, u2.username AS user2_username FROM USER_DETAILS u1 JOIN USER_DETAILS u2 ON u1.first_name... 10000 row(s) returned 0.078 sec / 0.985 sec
29 07:55:46 CREATE INDEX IND ON user_details(user_id, username, first_name, last_name, gender, password, status) 100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0 1.266 sec
30 07:55:55 SELECT COUNT(*) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.000 sec / 0.000 sec
31 07:56:09 SELECT AVG(user_id) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.015 sec / 0.000 sec
32 07:56:33 SELECT * FROM user_details ORDER BY last_name desc LIMIT 0, 10000 10000 row(s) returned 0.125 sec / 0.000 sec


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
60004210090
b'new'
coursera
husain
hussain
kreena
kreena243
kreena_243
kreenashah
Tables
Views
Stored Procedures
Functions
new
newschema
sakila
sys
university_239
world
Administration Schemas
Information
Schema: kreenashah
Object Info Session
Query 1 SQL File 3* SQL File 4* SQL File 7* SQL File 6* SQL File 8*
Limit to 10000 rows
1 USE kreenashah;
2 SELECT * FROM user_details;
3
4 OPTIMIZE TABLE user_details;
5 CREATE INDEX IND ON user_details(user_id, username, first_name, last_name, gender, password, status);
6 SELECT COUNT(*) FROM user_details;
7 SELECT AVG(user_id) FROM user_details;
8 SELECT * FROM user_details ORDER BY last_name desc;
9 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%');
10 SELECT u1.username AS user1_username, u2.username AS user2_username
11 FROM USER_DETAILS u1
12 JOIN USER_DETAILS u2 ON u1.first_name = u2.first_name AND u1.user_id < u2.user_id;
Query Statistics
Timing (as measured at client side):
Execution time: 0:00:0.00000000
Timing (as measured by the server):
Execution time: 0:00:0.00015790
Table lock wait time: 0:00:0.00000300
Errors:
Had Errors: NO
Warnings: 0
Rows Processed:
Rows affected: 0
Rows sent to client: 1
Rows examined: 1
Temporary Tables:
Temporary disk tables created: 0
Temporary tables created: 0
Joins per Type:
Full table scans (Select_scan): 0
Joins using table scans (Select_full_join): 0
Joins using range search (Select_full_range_join): 0
Joins with range checks (Select_range_check): 0
Joins using range (Select_range): 0
Sorting:
Sorted rows (Sort_rows): 0
Sort merge passes (Sort_merge_passes): 0
Sorts with ranges (Sort_range): 0
Sorts with table scans (Sort_scan): 0
Index Usage:
At least one Index was used
Other Info:
Event Id: 105
Thread Id: 53
Result Grid
Form Editor
Field Types
Query Stats
Result 19
Read Only
SQL Additions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
# Time Action Message Duration / Fetch
23 07:53:18 OPTIMIZE TABLE user_details 1 row(s) returned 0.047 sec / 0.000 sec
24 07:53:25 SELECT COUNT(*) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.000 sec / 0.000 sec
25 07:53:53 SELECT AVG(user_id) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.031 sec / 0.000 sec
26 07:54:08 SELECT * FROM user_details ORDER BY last_name desc LIMIT 0, 10000 10000 row(s) returned 0.500 sec / 0.000 sec
27 07:54:28 SELECT * FROM user_details WHERE password IN (SELECT password FROM user_details WHERE password LIKE '2%') LIMIT 0, 10000 5175 row(s) returned 0.469 sec / 0.484 sec
28 07:54:43 SELECT u1.username AS user1_username, u2.username AS user2_username FROM USER_DETAILS u1 JOIN USER_DETAILS u2 ON u1.first_name... 10000 row(s) returned 0.078 sec / 0.985 sec
29 07:55:46 CREATE INDEX IND ON user_details(user_id, username, first_name, last_name, gender, password, status) 100000 row(s) affected Records: 100000 Duplicates: 0 Warnings: 0 1.266 sec
30 07:55:55 SELECT COUNT(*) FROM user_details LIMIT 0, 10000 1 row(s) returned 0.000 sec / 0.000 sec