**Qatar University**

**College of Engineering**

**Department of Computer Science and Engineering**

**CMPS 451 – Fall 2023**

**Assignment 3**

**SID**: 202005204 **STUDENT NAME:** Salma Eletreby **Email**: Se2005204@qu.edu.qa

**SID**: 202003532 **STUDENT NAME:** Aya Muhanad Hassan **Email**: ah2003532@qu.edu.qa

**SID**: 202001942 **STUDENT NAME:** Hissa Al Muhannadi **Email**: ha2001942@qu.edu.qa

**Submission date**: 6th December 2023

**Class section:** L51

Q2:

1. Transaction Logging (redo log in MySQL):

* **Similarities:**
* A record of every transaction is saved. If there is a system failure, the transactions from the log will be replayed.
* Modifications that did not finish updating data files before an unexpected shutdown are replayed automatically.[[1]](#footnote-1)
* **Differences:**
* If all changes are flushed from the buffer to the tablespaces at the time of the crash, redo log application is skipped. Also, InnoDB skips redo log application if redo log files are missing at startup.
* During recovery, InnoDB scans the redo log to collect counter value changes and applies the changes to the in-memory table object.
* When encountering index tree corruption, InnoDB writes a corruption flag to the redo log, which makes the corruption flag crash safe.[[2]](#footnote-2)

1. Checkpointing:

* **Similarities:**
* During crash recovery, InnoDB looks for a checkpoint label written on the log files. Then InnoDB scans the log files forward from the checkpoint, applying the logged modifications to the database.
* **Differences:**
* InnoDB writes checkpoint information to the first log file at each checkpoint.
* InnoDB creates checkpoints and this often involves flushing of modified database pages to disk.[[3]](#footnote-3)

1. undo log:

* **Similarities:**
* undo log is a collection of undo log records associated with a single read-write transaction.
* **Differences:**
* undo log contains information about how to undo the latest change by a transaction to a clustered index record.[[4]](#footnote-4)

MySQL has three primary ways for applying security mechanisms: role-based access, authentication, and encryption. As discussed during the lecture, role-based access enforces policies and privileges based on organization roles. As discussed in the lecture, privileges are permissions to be granted to a user to allow them to perform certain functionalities. In MySQL, roles are created by grouping different privileges together [[5]](#footnote-5) and when a user is created by database administrator (DBA) they are assigned a role which immediately grants them all the privileges that this role has without having to grant each privilege individually [[6]](#footnote-6).

MySQL ‘s connection allows for it to be encrypted using Transport Layer Security (TLS) protocol. This allows MySQL to ensure that data received over a network is secure and trusted. The algorithms used by TLS allow MySQL to detect changes or losses in data or replay a transaction.[[7]](#footnote-7)

There are many ways to identify and authenticate a user. First is using electronic certificates granted by Certificate Authority (CA). These certificates use a public key and a secret key so when a user presents their certificate as a proof of their identity, if they are authentic, then using the secret corresponding key it can be decrypted [[8]](#footnote-8). Second is using the combination of access control and password to authenticate the user and ensure which transactions they can perform which is called the privilege system. When a program attempts to connect to MySQL server, MySQL will accept or reject based on verification of identity using the combination of correct username and password. If everything is correct, the connection will be accepted and any statement that is executed is double checked with privileges that the client has as discussed in the lecture[[9]](#footnote-9). The security of database can be enhanced further by using the validate\_password component that allows a DBA to specify the password’s length and other strength requirements to prevent attackers from guessing a password[[10]](#footnote-10). Third is using authentication plugins which allows for the creation of proxy users. Upon connection the plugin will request that the connecting user is treated as a proxy user for checking privileges [[11]](#footnote-11). This proxy user allows for better security by hiding the database and easier auditing as it centralizes the connection.

In MySQL, DBAs have the authority to lock and unlock an account [[12]](#footnote-12). As discussed during the lecture, there might be login sessions or audits that record transactions, so if in any of these logs suspicious activity was found, a DBA has the authority to lock the account in order to investigate further and prevent the suspected user from preforming more transactions. Once the suspicion is cleared, the account can be unlocked. This helps in securing the database by preventing a suspected attacker from doing more harm.

Lastly, this option is available to MySQL enterprise which is data masking and de-identification. As discussed in the lecture, some sensitive data may exist that should be hidden in most cases, using those features we can hide the data yet still be able to use it in our transactions. For example, using the data masking feature we can transform the digits of a credit card number into an X or using random data for payment card numbers using the de-identification[[13]](#footnote-13). This would allow all non-authorized users to view the data, generate statistics or report without compromising the individual’s privacy.

1. <https://dev.mysql.com/doc/refman/8.0/en/innodb-redo-log.html#:~:text=The%20redo%20log%20is%20a,or%20low%2Dlevel%20API%20calls> [↑](#footnote-ref-1)
2. <https://dev.mysql.com/doc/refman/8.0/en/innodb-recovery.html> [↑](#footnote-ref-2)
3. <http://download.nust.na/pub6/mysql/doc/mysql-backup-excerpt/5.1/en/innodb-checkpoints.html> [↑](#footnote-ref-3)
4. <https://dev.mysql.com/doc/refman/8.0/en/innodb-undo-logs.html> [↑](#footnote-ref-4)
5. <https://dev.mysql.com/doc/refman/8.0/en/roles.html> [↑](#footnote-ref-5)
6. <https://dev.mysql.com/doc/refman/8.0/en/roles.html> [↑](#footnote-ref-6)
7. <https://dev.mysql.com/doc/mysql-security-excerpt/8.0/en/encrypted-connections.html> [↑](#footnote-ref-7)
8. <https://dev.mysql.com/doc/mysql-security-excerpt/8.0/en/encrypted-connections.html> [↑](#footnote-ref-8)
9. <https://dev.mysql.com/doc/refman/8.0/en/access-control.html> [↑](#footnote-ref-9)
10. <https://dev.mysql.com/doc/refman/8.0/en/validate-password.html> [↑](#footnote-ref-10)
11. <https://dev.mysql.com/doc/refman/8.0/en/proxy-users.html> [↑](#footnote-ref-11)
12. <https://dev.mysql.com/doc/refman/8.0/en/account-locking.html> [↑](#footnote-ref-12)
13. <https://dev.mysql.com/doc/refman/8.0/en/data-masking.html> [↑](#footnote-ref-13)