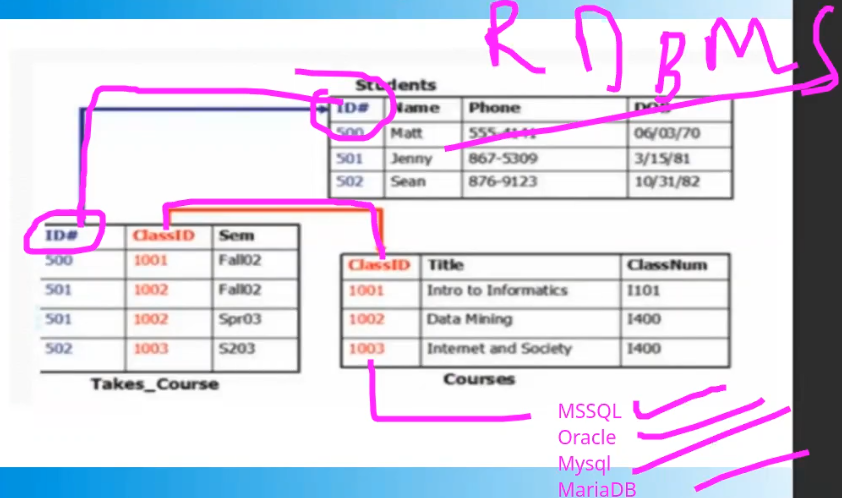
**Lecture 24**

MySQL - RDBMS Part2 (Live Session 25 September 2022)



* Relational database, interconnection between tables. Or a relation between tables.
  + MSSQL
  + Oracle
  + MySQL
  + MariaDB
* SQL (structured Querry Language) is used.
* Try to connect the sql server remotely.
* 
* It means SQL Service is not working on server.
* To check if service is on, on centos 6 or centos 7
* 
* .

If root password is not working or forgotten.

* It is admins job to recover or reset the root password.
* **This is not hacking.**
* An Admins role is to ***recover*** the password.
* 
* Steps are,

1. **Stop the MySQL service:**

sudo systemctl stop mysqld

1. **Start MySQL in safe mode:**

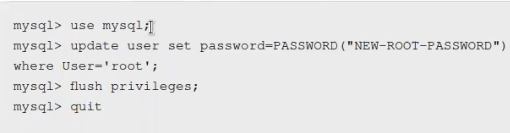
sudo mysqld\_safe --skip-grant-tables --skip-networking **&**

|  |
| --- |
| **mysqld\_safe** is a command used to start the MySQL server in a safe mode. The **--skip-grant-tables** option will start the MySQL server with all privileges granted to any user, bypassing the normal password authentication system. This is usually done when the MySQL root password has been lost or forgotten, and you need to reset it.  The **--skip-networking** option will disable networking, which means that the MySQL server will only be accessible from the local machine. This is useful in situations where you want to prevent remote connections to the MySQL server.  The **&** at the end of the command is a shell symbol that tells the operating system to run the command in the background, which means that you can continue to use the terminal window or shell prompt while the MySQL server is running.  In summary, the command **mysqld\_safe --skip-grant-tables --skip-networking &** starts the MySQL server in safe mode with all privileges granted to any user and networking disabled, and runs the command in the background. |

* + **Note:-** this step will take some time and return to the Linux prompt. It means SQL service is started in safe mode.

1. **Connect to MySQL:**

mysql -u root 🡪 it will take to the mySQL prompt but few tables which are critical would be skipped.



1. **Update the root password: 🡪** *running these commands separately is a good idea. (I pasted the copied commands)*
   1. use mysql;
   2. update user set authentication\_string=PASSWORD("NEW\_PASSWORD") where User='root'; 🡪 *user can be any user, since we are resetting “root” password so here root is mentioned.*
   3. flush privileges; 🡪 *to make the changes persistent 🡪 means flush from memory (RAM) and save to the storage.*
   4. exit;

this step is little bit different in CentOS 6.

Text

Description automatically generated

Replace "NEW\_PASSWORD" with your desired new password.

As I pasted this command, it changed the password and exited 🡪 in this case the password is **NEW\_PASSWORD**

Graphical user interface, text, application

Description automatically generated

1. **Stop the MySQL service:**

sudo systemctl stop mysqld

Text

Description automatically generated

1. **Start the MySQL service:**

sudo systemctl start mysqld



1. **Test the new root password:**

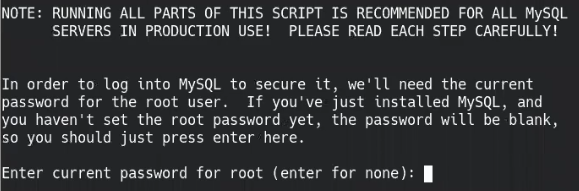
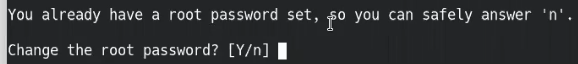
mysql -u root -p

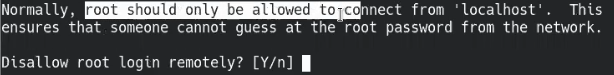
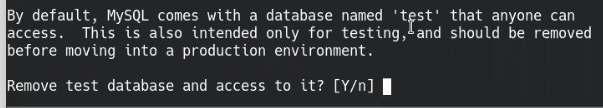
* + - I have checked the changed password, it is working.

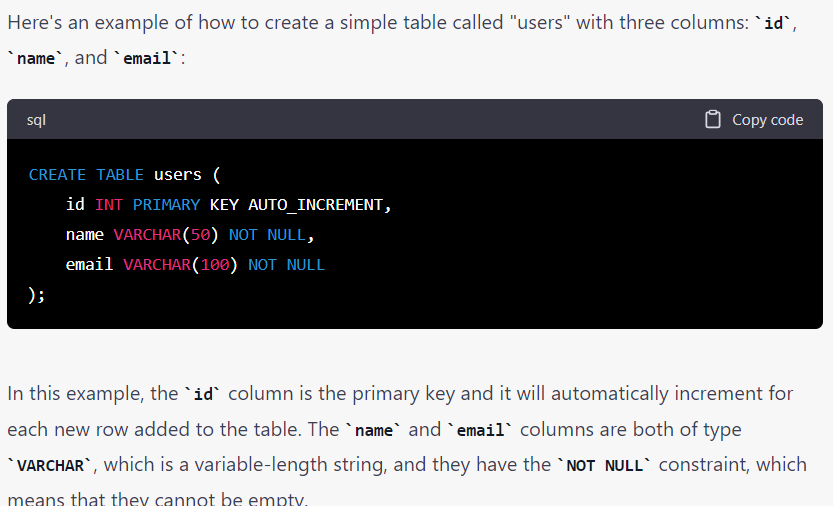
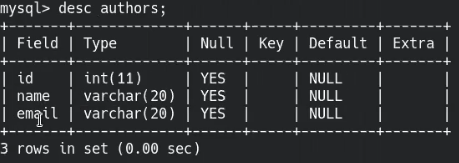
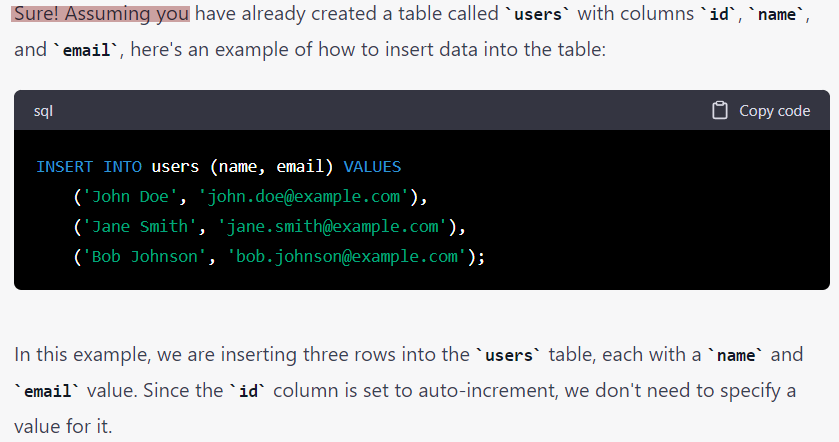
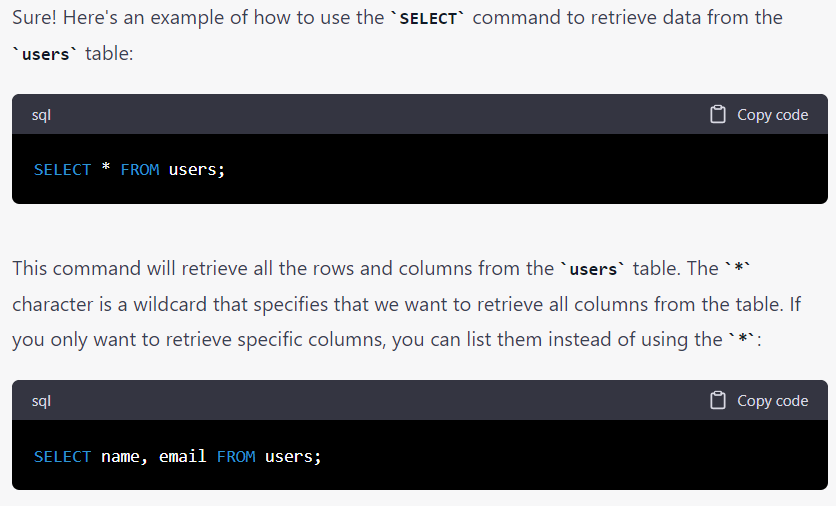
You should be prompted to enter your new password. If you can successfully log in, your new root password has been set.

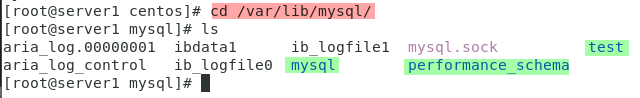
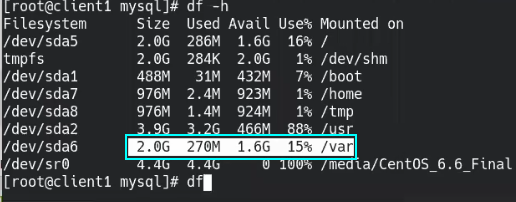
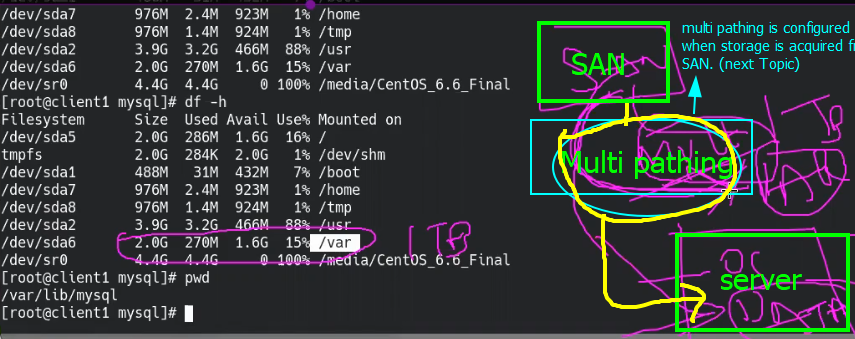
* .

Secure **configuration** of “mySQL”

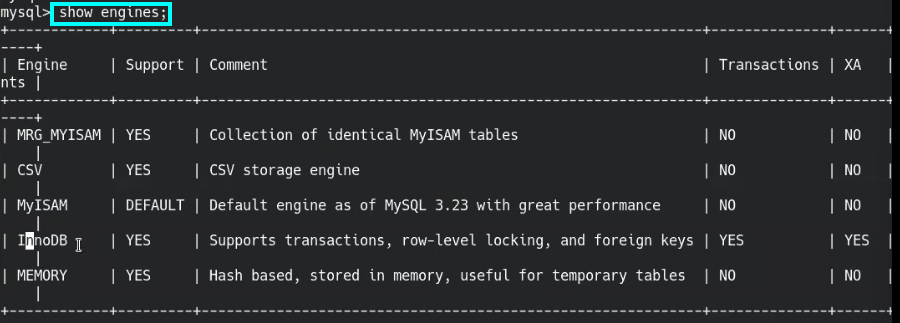
* How the work is done in practical environment.
* A meeting with the management & development team.
* For secure configuration the mySQL service must run successfully.
* $ mysql\_secure\_installation
* 
* .
* 
* .
* Graphical user interface, text, application, chat or text message

  Description automatically generated
* *We don’t need “anonymous” users in production environment.*
* 
* Root should not connect remotely.
* 
* Yes
* Text

  Description automatically generated with low confidence
* Yes
* Done!!!
* This is secure configuration of “mySQL” server.
* .
* Few queries for the sake of information.
* To create new db
  + $ create database alnafi;
* To remove a db
  + $ drop database test3;
* To connect to a db
  + $ use alnafi;
* To create a table.
  + $ create table employee;
  + >
* 
* To show description of a table
* mysql> desc <table\_name>; 🡪 structure of a table
* 
* Use INSERT command to enter data into the table,
* 
* Usage of SELCET command,🡪 to retrieve the data from table.
* 
* Graphical user interface, text, application

  Description automatically generated
* Interview Question
* Suppose INSERT query is high in a table (means that a large amount of data is being written.)
* In which RAID this database should be 🡪 to get best write speed and avoid slowness.
* Answer: RAID 1+0
* If read performance is required?
* Answer: RAID 5
* Where mySQL stores (save) its data? 🡪 it is important to know as an Admin.
* $ /var/lib/mysql/ 🡪 default location of mySQL.
* 
* Suppose the project manager during meeting askes to allow 1 Tb for data base size.
* $ df -h 🡪 tell about the situation of the storage
* Hence, default location of the mySQL data base is in /var 🡪 as and Admin the Admin will make sure that the /var should have this space (1Tb)
* 
* If enough storage space is not available 🡪 an Admin must migrate the data or attach a new storage.
* 
* .
* .

|  |
| --- |
| **how to change default location of mysql?**  To change the default location of MySQL, you need to perform the following steps:   1. Stop the MySQL service: You should stop the MySQL service before changing the default location of the MySQL data directory. 2. Move the MySQL data directory: Move the MySQL data directory to the new location. The data directory is usually located in /var/lib/mysql/ in Linux systems.   For example, you can move the data directory from /var/lib/mysql/ to /home/mysql/ by running the following command as root:  **mv /var/lib/mysql/ /home/mysql/**   1. Edit the MySQL configuration file: Edit the MySQL configuration file (/etc/my.cnf or /etc/mysql/my.cnf) and modify the datadir parameter to point to the new location of the MySQL data directory.   For example, you can open the configuration file in a text editor and modify the datadir parameter as follows:  **[mysqld] datadir=/home/mysql/**  Save the changes and exit the editor.   1. Update file permissions: You may need to update file permissions to ensure that the new location of the MySQL data directory is accessible by the MySQL user. You can use the following command to set the appropriate permissions:   **chown -R mysql:mysql /home/mysql/**  This command changes the owner and group of the /home/mysql directory and its **contents to mysql:mysql.**   1. Restart the MySQL service: Finally, start the MySQL service again to apply the changes you made to the configuration file.   On a Linux system, you can restart the MySQL service by running the following command as root:  **systemctl restart mysql**  After completing these steps, the MySQL data directory will be located in the new location you specified in the configuration file. |

* .
* **How SQL works?**
* Engine based work.
* different engines 🡪 Every engine has different role.
* 
* .
* MyISAM is default engine.
* Hierarchy : Engine🡪 database 🡪 tables
* In production,
* **InnoDB** is used mostly. 🡪 its performance is great. 🡪 disadvantage 🡪 takes much memory (RAM) 🡪 ***database tuning*** is the ***solution***. 🡪 according to the application requirements.
* .
* Difference b/w MyISAM & InnoDB 🡪 Interview Question

|  |  |
| --- | --- |
| MyISAM | InnoDB |
| Table level locking  Table level locking is a type of locking mechanism used by some database management systems, including MySQL's MyISAM storage engine. In table level locking, when a user issues a query that involves a table, the entire table is locked for the duration of the query, preventing other users from modifying or accessing the same table. | Row level locking  Row level locking is a type of locking mechanism used by some database management systems, including MySQL's InnoDB storage engine. In row level locking, when a user issues a query that modifies a single row in a table, only that row is locked for the duration of the query, allowing other users to access or modify other rows in the same table. |
|  |  |

|  |
| --- |
| ***as an Admin (Linux Administrator : RHEL) what kind of queries (related to working of mySQL server only because the developer team will look after all the queries related to development) I may have in production environment?***  As a Linux administrator managing a MySQL server in a production environment, you may need to execute various queries related to the functioning of the MySQL server. Here are some examples of such queries:   1. Querying server status and configuration: You may need to retrieve information about the current status and configuration of the MySQL server, such as the version number, uptime, or configuration parameters.   sqlCopy code  SELECT VERSION(); -- retrieves the version number of the MySQL server SHOW STATUS; -- retrieves various status information about the server, such as uptime and query counts SHOW VARIABLES; -- retrieves configuration parameters for the server, such as buffer sizes and timeouts   1. Managing user accounts and privileges: You may need to create, modify, or delete user accounts, as well as grant or revoke privileges for specific databases or tables.   sqlCopy code  CREATE USER 'username'@'hostname' IDENTIFIED BY 'password'; -- creates a new user account with a specified username, hostname, and password GRANT privilege ON database\_name.table\_name TO 'username'@'hostname'; -- grants a specific privilege (such as SELECT, INSERT, or UPDATE) on a specific table to a user account REVOKE privilege ON database\_name.table\_name FROM 'username'@'hostname'; -- revokes a specific privilege on a specific table from a user account DROP USER 'username'@'hostname'; -- deletes a user account from the server   1. Monitoring server performance and troubleshooting issues: You may need to identify and resolve performance issues or errors on the MySQL server, such as slow queries, high CPU usage, or database corruption.   sqlCopy code  EXPLAIN SELECT \* FROM table\_name; -- analyzes the execution plan for a specific query and identifies any performance issues or bottlenecks SHOW PROCESSLIST; -- lists all active client connections to the server and the queries they are executing, which can help identify performance issues or long-running queries SHOW ENGINE INNODB STATUS; -- retrieves detailed information about the InnoDB storage engine, including any potential errors or warnings  These are just a few examples of the types of queries you may need to use as a Linux administrator managing a MySQL server in a production environment. The specific queries you use will depend on the tasks you need to perform and the issues you encounter. |