Chapter-2 (DDL-DML in Command Line)

DDL

1. **Check MySQL Version:**
- mysql -u root -p > enter > enter
- MariaDB [(none)]> select version();

2. **Create Database:**
- cd C:\ > cd Xampp > cd mysql > cd bin
- C:\xampp\mysql\bin
- Type CMD on the above location
- mysql -u root -p > enter > enter
- create database pos;

3. **Show Databases:**
- show databases;

4. **Enter into Database:**
- show databases;
- use database pos;

5. **Create Table:**

```
create table customer (
   id int not null  auto_increment primary key,
   name varchar (45) NOT NULL,
   dob date,
   address varchar (100),
   cell int (13),
   email varchar (40)
);
```

6. **Show Table's Column:**
- desc student;

7. **Show Tables:**
- show tables;

8. **Create user & Privileges:**
- create user 'faruq'@localhost identified by 'Abc@12345';
- grant all privileges on pos.* to 'faruq'@'localhost';
- show grants for 'faruq'@'localhost';
- drop user 'faruq'@'localhost';
- flush privileges;

9. **Show Users:**
- select user from mysql.user;

10. **Insert Data:**

insert into customer (name, dob, address, cell, email) values ('faruq', 11-10-2021, 'Sylhet', 0124585214, 'faruq@gmail.com');

11. Add column:

- alter table customers add email varchar (255);

12. Drop column:

- alter table customers drop column email;

13. Modify column:

- alter table customer modify column email varchar(40);

14. Change column name:

- alter table student change column name student_name varchar(40);

15. Set Value:

- update student set dept=10 where id =1000;

16. Make unique attribute to make sure data redundancy:

- alter table student add unique(email);

17. Define primary key:

- alter table dept add primary key (id);

18. Define foreign key:

- alter table student add foreign key(dept) references dept(dept);

19. Check:

- alter table student add check (age>=18);

20. Set Default:

- alter table student alter address set default "Sylhet";

21. Index:

- create index idx_student_id on student (id);

22. Auto Increment:

- create index idx_student_id on student (id);

23. View:

A view is an empty database object. Its information is based on the foundation table. It has rows and columns that are comparable to those seen in a real table. In MySQL, a View is a virtual table produced by connecting one or more tables in a query. It works in the same way as the basic table but doesn't have any data of its own. The major distinction between a view and a table is that views are definitions constructed on top of other tables (or views). Any modifications made to the underlying table are reflected in the View as well.

- create view lu_cse as select name, dept from student;
- show tables; (lu_cse view has been created)
- select * from lu_cse

24. Lock Table:

- lock table student write;

25. Unlock Table:

- unlock tables;

DML

1. **Select Data:**
   - select * from customer;

2. **Where Clause:**
   - select * from customer where id = 1;
   - select name, cell from customer where id =1;

3. **Order By:**
   - select * from student order by student_name;
   - select * from student order by student_name desc;

4. **AND-NOT-OR:**
   - select * from student where student_name ="Rony" and id = 4;
   - select * from student where not address = "Dhaka";
   - select * from student where name ="Faruq" or age = 20;

5. **Null-Not Null:**
   - select * from student where address is null;
   - select * from student where address is not null;

6. **Limit:**
   - select * from student limit 2;

7. **Min-Max-Sum-Count:**
   - select min(age) as youngest from student;
   - select max(age) as young from student;
   - select count(id) from student;
   - select count(id) from student group by age;
   - select sum(age) from student;
   - select sum(age) from student group by age;

8. **Like:**
   - select * from student where name like "F%";
   - select * from student where age in (55,20);

9. **Between:**
   - select * from student where dob between "2021-01-01" and "2021-12-30";
   - select * from student where dob not between "2021-01-01" and "2021-12-30";

10. Update:

- update student set dept = 05 where id =7;

11. Join:

- select s.name, s.email, d.name from student s, dept d where s.dept=d.dept;

12. Inner Join:

- select student.name, dept.name from student inner join dept on student.dept=dept.dept;

13. Left Join:

- select student.name, dept.dept from student left join dept on student.dept=dept.dept;

14. Right Join:

- select student.name, dept.dept from student right join dept on student.dept=dept.dept;

15. Union:

- select name from student union select dept from dept;

16. Group By:

- select count(id), age from student group by age;
- select name, count(*) from student group by age;

17. Having:

- select name, count(*) from student group by age having count(age)>1;

18. Distinct:

- select distinct age from student;

19. Concat:

- select concat (name, "-", id) as student from student;

20. Subquery:

- select name, dept from student where dept in (select dept from dept);
- Example of a big subquery:

```
SELECT
    product_name,
    list_price
FROM
    production.products
WHERE
    list_price > (
        SELECT
            AVG (list_price)
        FROM
            production.products
        WHERE
            brand_id IN (
                SELECT
                    brand_id
                FROM
                    production.brands
                WHERE
                    brand_name = 'Strider'
                OR brand_name = 'Trek'
            )
    )
ORDER BY
    list_price;
```

Chapter-3 (MongoDB)

Database, Collection & Document

1. **Database:** Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.
2. **Collection:** Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.
3. **Document:** A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

The following table shows the relationship of RDBMS terminology with MongoDB.

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by MongoDB itself)

1. MongoDB Version:

- C:\Program Files\MongoDB\Server\5.0\bin>mongod --version

2. Start MongoDB:

- C:\Program Files\MongoDB\Server\5.0\bin>mongo

3. Clear Screen:

- > cls

4. View Databases:

- > show dbs

5. Create Database:

- > user pos

6. Drop Database:

- db.dropDatabase()

7. Create Collection:

- db.createCollection("student")

8. Show Collections:

- show collections

9. View Data:

- db.student.find()

10. Drop Collection:
- db.student.drop()

11. Delete Data:
- db.student.remove({})

12. Insert Data:
- db.student.insert({"name": "Faruq", "cell": "0124521254", "email": "faruq@gmail.com", "courses":["CSE-1111","CSE-2319","CSE-4219"] })

13. Find Data Conditionally:
- db.student.findOne({"name":"Faruq"})