SQL assignment report

Group Members:

1. AKARIZA GASANA Leslie 27413
2. TETA Kevine 27973
3. UMUTONIWASE Aliane

INTRODUCTION

This report shows the use of SQL to create and manipulate a relational database. The tasks include:

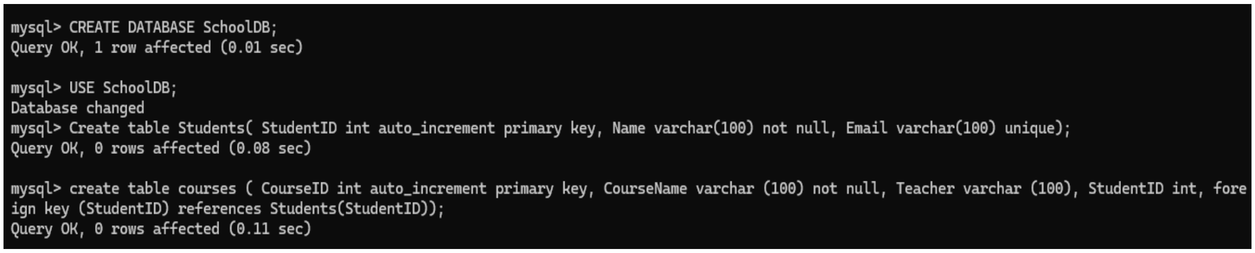
- Creating tables with constraints.
- Performing different types of joins.
- Creating and index to improve performance.
- Creating a view to simplify queries

Database design and creation of tables

We created a new database named SchoolDB and two tables, Students and Courses. They were defined by these constraints;

- Primary key: both tables have primary keys.
- Not null: which ensures important fields like name and course name are not empty.
- Unique: the email field in Students table must be unique.
- Foreign key: StudentID in courses references StudentID in Students.

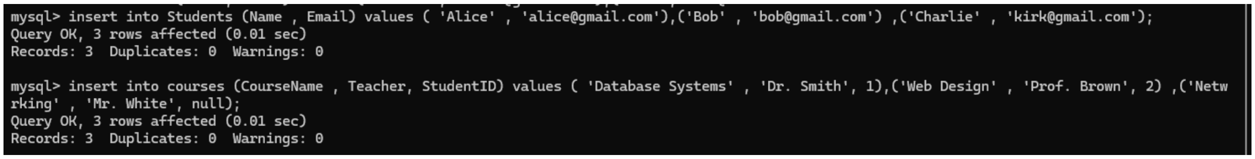
SQL code:



```
mysql> CREATE DATABASE SchoolDB;
Query OK, 1 row affected (0.01 sec)

mysql> USE SchoolDB;
Database changed
mysql> Create table Students( StudentID int auto_increment primary key, Name varchar(100) not null, Email varchar(100) unique);
Query OK, 0 rows affected (0.08 sec)

mysql> create table courses ( CourseID int auto_increment primary key, CourseName varchar (100) not null, Teacher varchar (100), StudentID int, fore
ign key (StudentID) references Students(StudentID));
Query OK, 0 rows affected (0.11 sec)
```

Data insertion

Sample data were inserted:



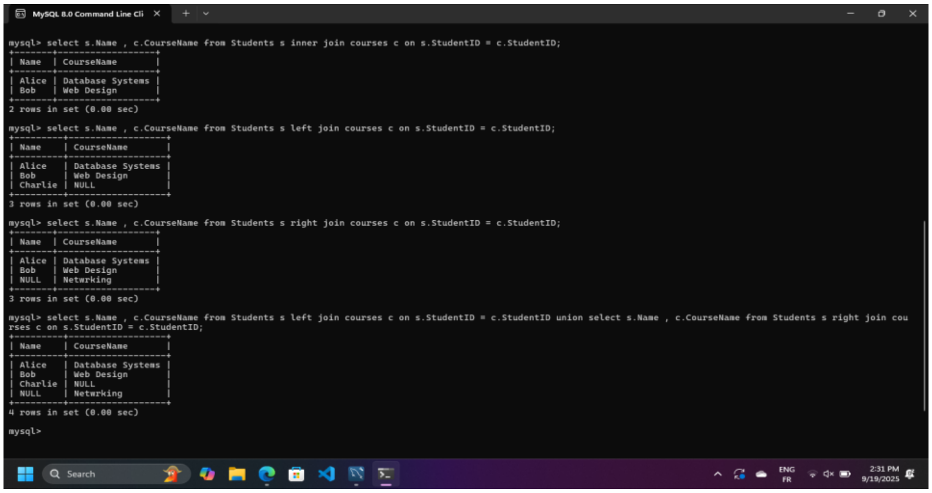
```
mysql> insert into Students (Name , Email) values ( 'Alice' , 'alice@gmail.com'),('Bob' , 'bob@gmail.com') ,('Charlie' , 'kirk@gmail.com');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> insert into courses (CourseName , Teacher, StudentID) values ( 'Database Systems' , 'Dr. Smith', 1),('Web Design' , 'Prof. Brown', 2) ,('Netw
rking' , 'Mr. White', null);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

Join operations

Different joins were tested to demonstrate relationships:

- Inner join: returns only students enrolled in a course.
- Left join: shows all students, even those who aren’t enrolled.
- Right join: shows all courses, even if no students are enrolled.
- Full outer join (simulated with union): combines both sides (left and right join).

Example and results;

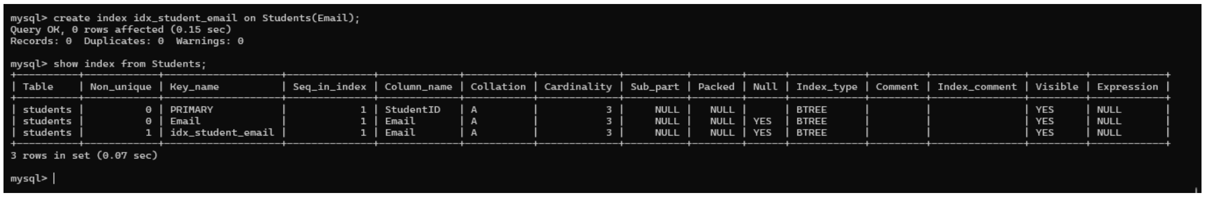


```
mysql> select s.Name , c.CourseName from Students s inner join courses c on s.StudentID = c.StudentID;
+-------+------------------+
| Name  | CourseName       |
+-------+------------------+
| Alice | Database Systems |
| Bob   | Web Design       |
+-------+------------------+
2 rows in set (0.00 sec)

mysql> select s.Name , c.CourseName from Students s left join courses c on s.StudentID = c.StudentID;
+---------+------------------+
| Name    | CourseName       |
+---------+------------------+
| Alice   | Database Systems |
| Bob     | Web Design       |
| Charlie | NULL             |
+---------+------------------+
3 rows in set (0.00 sec)

mysql> select s.Name , c.CourseName from Students s right join courses c on s.StudentID = c.StudentID;
+-------+------------------+
| Name  | CourseName       |
+-------+------------------+
| Alice | Database Systems |
| Bob   | Web Design       |
| NULL  | Netwrking        |
+-------+------------------+
3 rows in set (0.00 sec)

mysql> select s.Name , c.CourseName from Students s left join courses c on s.StudentID = c.StudentID union select s.Name , c.CourseName from Students s right join cou
rses c on s.StudentID = c.StudentID;
+---------+------------------+
| Name    | CourseName       |
+---------+------------------+
| Alice   | Database Systems |
| Bob     | Web Design       |
| Charlie | NULL             |
| NULL    | Netwrking        |
+---------+------------------+
4 rows in set (0.00 sec)

mysql>
```

Index creation

An index was created on the Email field to optimize search performance. This improves query speed when searching by email.

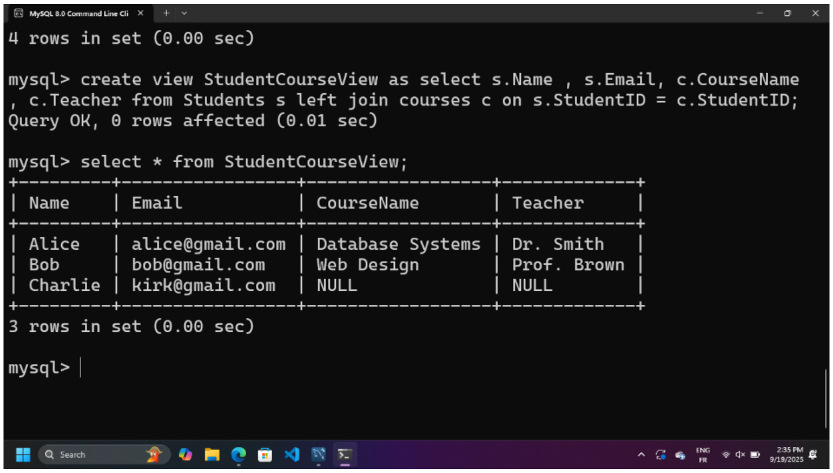
Demonstration:



```
mysql> create index idx_student_email on Students(Email);
Query OK, 0 rows affected (0.15 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> show index from Students;
+----------+------------+-------------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table    | Non_unique | Key_name          | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expression |
+----------+------------+-------------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| students |          0 | PRIMARY           |            1 | StudentID   | A         |           3 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| students |          0 | Email             |            1 | Email       | A         |           3 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
| students |          1 | idx_student_email |            1 | Email       | A         |           3 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
+----------+------------+-------------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
3 rows in set (0.07 sec)

mysql> |
```

View creation

A view was created to simplify queries and reporting. It allows quick access with:



```
4 rows in set (0.00 sec)

mysql> create view StudentCourseView as select s.Name , s.Email, c.CourseName
, c.Teacher from Students s left join courses c on s.StudentID = c.StudentID;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from StudentCourseView;
+---------+-----------------+------------------+-------------+
| Name    | Email           | CourseName       | Teacher     |
+---------+-----------------+------------------+-------------+
| Alice   | alice@gmail.com | Database Systems | Dr. Smith   |
| Bob     | bob@gmail.com   | Web Design       | Prof. Brown |
| Charlie | kirk@gmail.com  | NULL             | NULL        |
+---------+-----------------+------------------+-------------+
3 rows in set (0.00 sec)

mysql> 
```

Conclusion

Through this assignment, we demonstrated key SQL concepts:

- Database design with constraints ensures data accuracy and relationships.
- Join operations provide flexible ways to retrieve related data.
- Indexes improve performance for frequently queried fields.
- Views simplify reporting and abstract complex queries.

These tasks showed the practical importance of SQL in real-world database management and reporting.