GRP Members: - Bizima Peace 27778
-TWakira Agape Gift 27320
-Uwineza Delphine 27897

Class Quiz Report PL SQL

This report presents the solution to the class quiz group task. We created database tables with constraints, performed joins, created an index, built a view, and explained the results step by step. Spaces are reserved for screenshots taken from phpMyAdmin to demonstrate outputs.

1. Creating Tables with Constraints

We created three tables: students, courses, and enrollments. Each table has primary keys and relevant constraints like NOT NULL and UNIQUE. The enrollments table also has foreign keys linking students and courses.

SQL Used:

```
CREATE DATABASE class_quiz;
USE class_quiz;

CREATE TABLE students (
  student_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  email VARCHAR(100) UNIQUE NOT NULL
);

CREATE TABLE courses (
  course_id INT AUTO_INCREMENT PRIMARY KEY,
  course_name VARCHAR(100) NOT NULL
);

CREATE TABLE enrollments (
  enroll_id INT AUTO_INCREMENT PRIMARY KEY,
  student_id INT,
  course_id INT,
  grade VARCHAR(5),
  FOREIGN KEY (student_id) REFERENCES students(student_id),
  FOREIGN KEY (course_id) REFERENCES courses(course_id)
);
```

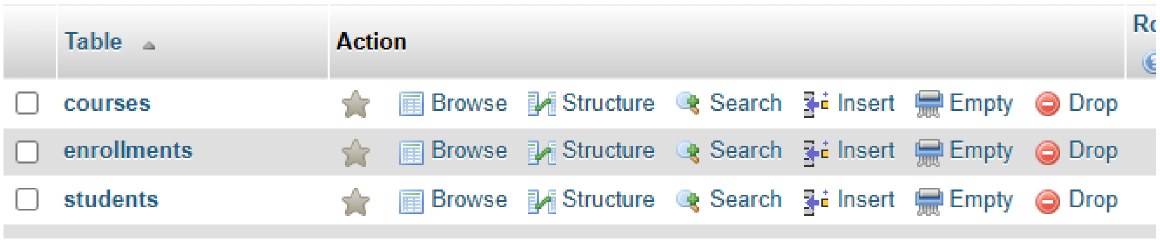


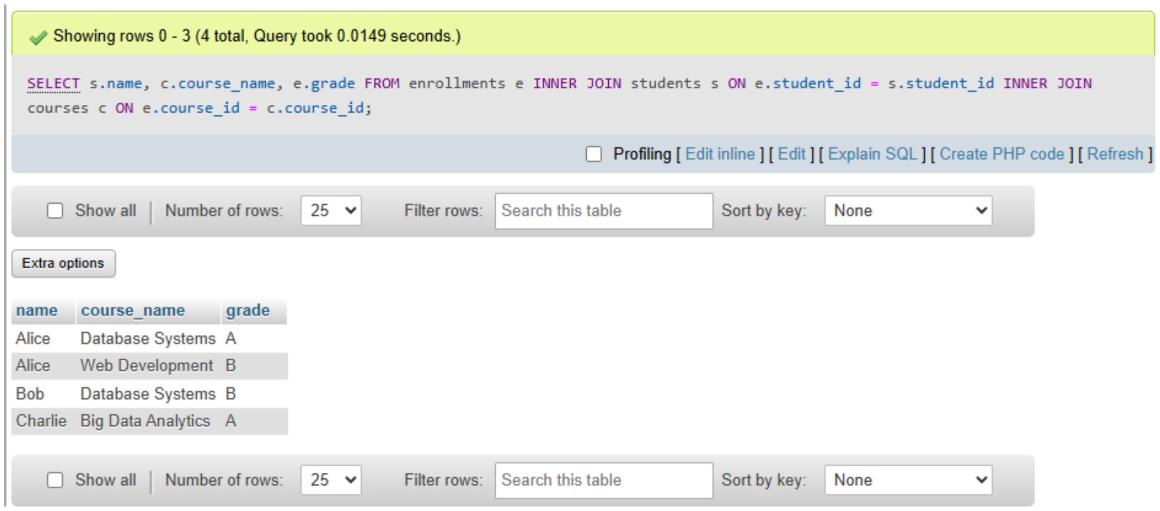
	Table	Action						Ro
☐	courses	☆	Browse	Structure	Search	Insert	Empty	Drop
☐	enrollments	☆	Browse	Structure	Search	Insert	Empty	Drop
☐	students	☆	Browse	Structure	Search	Insert	Empty	Drop

2. Performing Joins

We inserted sample data into the tables and demonstrated different joins: INNER, LEFT, RIGHT, and FULL (using UNION).

Example SQL Queries:

INNER JOIN:
SELECT s.name, c.course_name, e.grade
FROM enrollments e
INNER JOIN students s ON e.student_id = s.student_id
INNER JOIN courses c ON e.course_id = c.course_id;



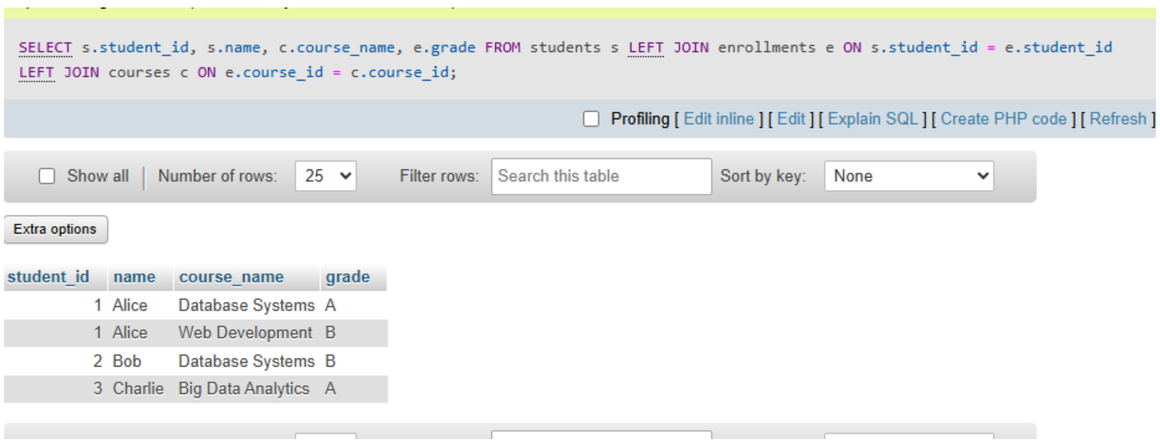
✓ Showing rows 0 - 3 (4 total, Query took 0.0149 seconds.)

```
SELECT s.name, c.course_name, e.grade FROM enrollments e INNER JOIN students s ON e.student_id = s.student_id INNER JOIN
courses c ON e.course_id = c.course_id;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Extra options

name	course_name	grade
Alice	Database Systems	A
Alice	Web Development	B
Bob	Database Systems	B
Charlie	Big Data Analytics	A

LEFT JOIN:
SELECT s.name, c.course_name, e.grade
FROM students s
LEFT JOIN enrollments e ON s.student_id = e.student_id
LEFT JOIN courses c ON e.course_id = c.course_id;

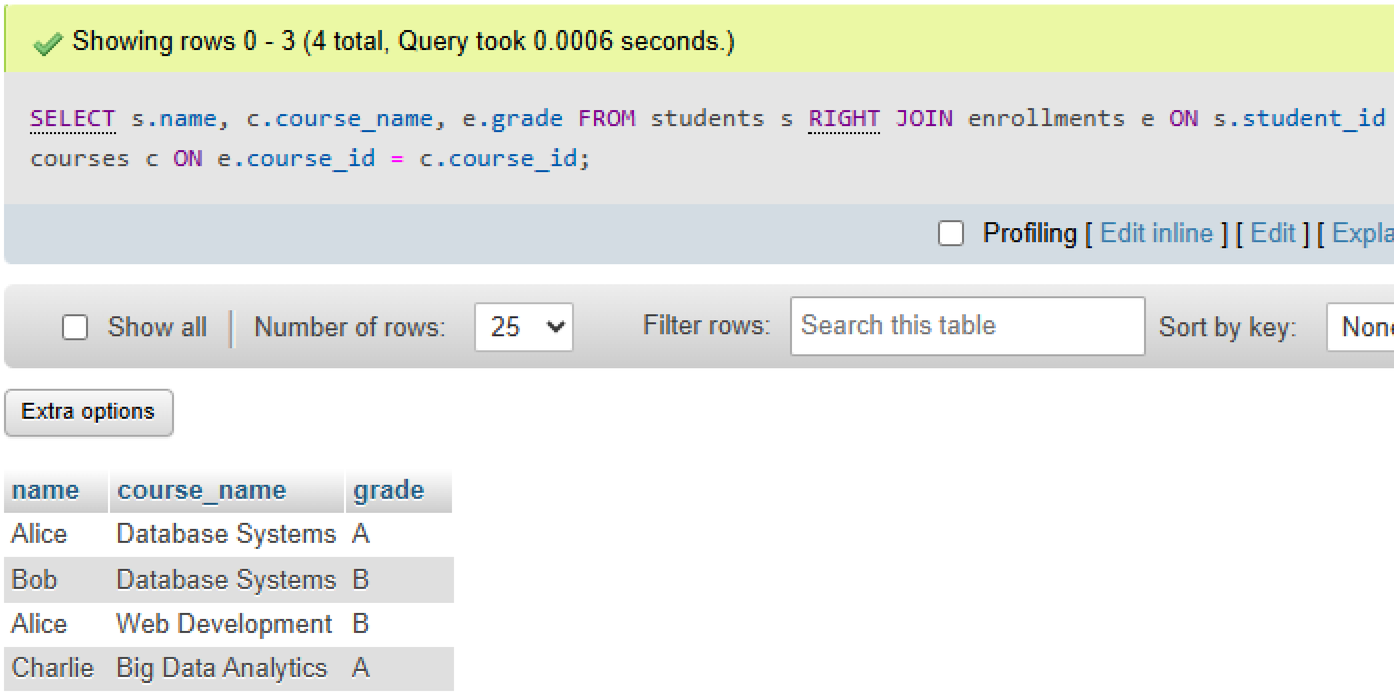

```
SELECT s.student_id, s.name, c.course_name, e.grade FROM students s LEFT JOIN enrollments e ON s.student_id = e.student_id
LEFT JOIN courses c ON e.course_id = c.course_id;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

student_id	name	course_name	grade
1	Alice	Database Systems	A
1	Alice	Web Development	B
2	Bob	Database Systems	B
3	Charlie	Big Data Analytics	A

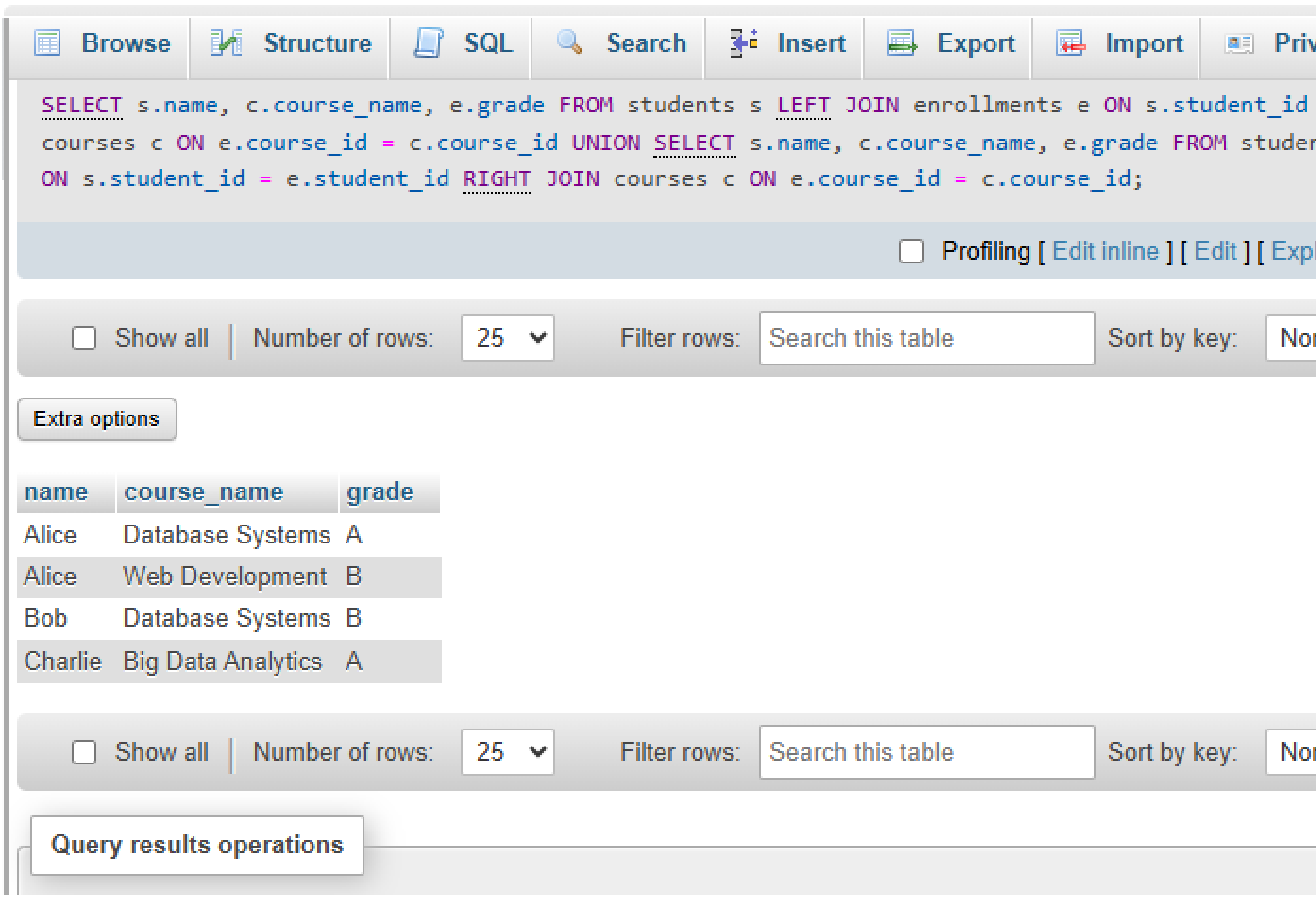


Showing rows 0 - 3 (4 total, Query took 0.0006 seconds.)

```
SELECT s.name, c.course_name, e.grade FROM students s RIGHT JOIN enrollments e ON s.student_id
courses c ON e.course_id = c.course_id;
```

Profiling [Edit inline] [Edit] [Expla
Show all | Number of rows: 25
Filter rows: Search this table
Sort by key: None
Extra options

name	course_name	grade
Alice	Database Systems	A
Bob	Database Systems	B
Alice	Web Development	B
Charlie	Big Data Analytics	A



Browse
Structure
SQL
Search
Insert
Export
Import
Priv

```
SELECT s.name, c.course_name, e.grade FROM students s LEFT JOIN enrollments e ON s.student_id
courses c ON e.course_id = c.course_id UNION SELECT s.name, c.course_name, e.grade FROM studer
ON s.student_id = e.student_id RIGHT JOIN courses c ON e.course_id = c.course_id;
```

Profiling [Edit inline] [Edit] [Expl
Show all | Number of rows: 25
Filter rows: Search this table
Sort by key: Nor
Extra options

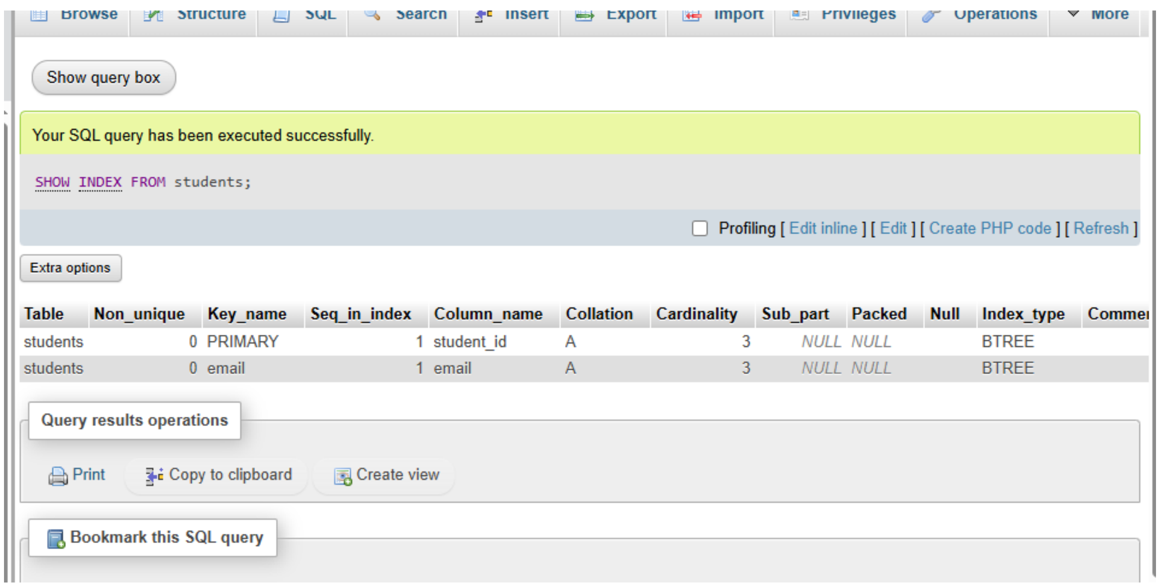
name	course_name	grade
Alice	Database Systems	A
Alice	Web Development	B
Bob	Database Systems	B
Charlie	Big Data Analytics	A

3. Creating an Index

We created a composite index on (student_id, course_id) in the enrollments table to optimize join queries.

SQL Used:

CREATE INDEX idx_student_course ON enrollments(student_id, course_id);



Your SQL query has been executed successfully.

```
SHOW INDEX FROM students;
```

Profiling [Edit inline] [Edit] [Create PHP code] [Refresh]
Extra options

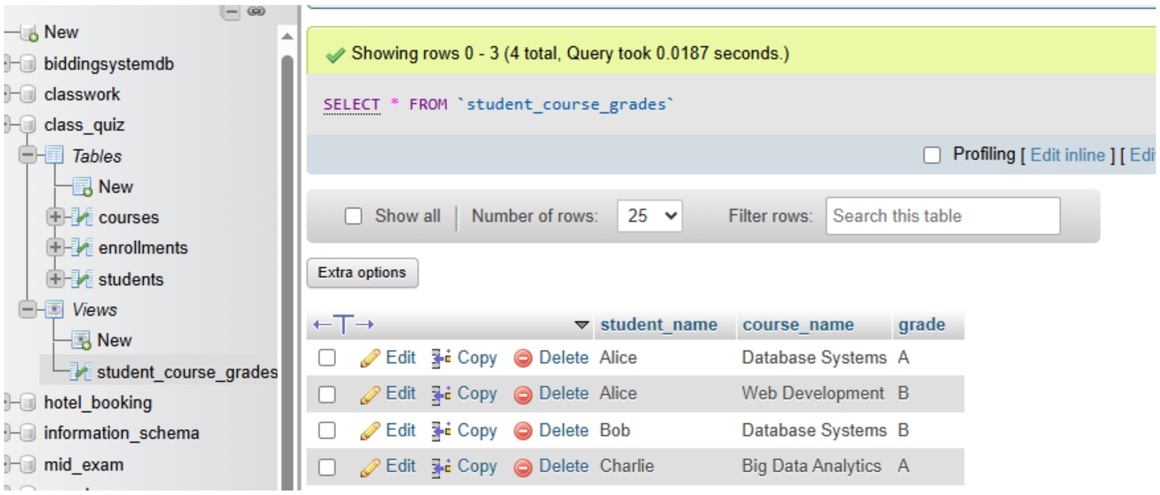
Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comme
students	0	PRIMARY	1	student_id	A	3	NULL	NULL		BTREE	
students	0	email	1	email	A	3	NULL	NULL		BTREE	

4. Creating a View

We created a view called student_course_grades to simplify data access. This view joins students, courses, and enrollments into a single query.

SQL Used:

CREATE VIEW student_course_grades AS
SELECT s.name AS student_name, c.course_name, e.grade
FROM enrollments e
INNER JOIN students s ON e.student_id = s.student_id
INNER JOIN courses c ON e.course_id = c.course_id;



Showing rows 0 - 3 (4 total, Query took 0.0187 seconds.)

```
SELECT * FROM `student_course_grades`
```

Profiling [Edit inline] [Edi
Show all
Number of rows: 25
Filter rows: Search this table
Extra options

	student_name	course_name	grade
Edit Copy Delete	Alice	Database Systems	A
Edit Copy Delete	Alice	Web Development	B
Edit Copy Delete	Bob	Database Systems	B
Edit Copy Delete	Charlie	Big Data Analytics	A

5. Report Conclusion

In this quiz, we successfully:
- Created tables with primary keys, foreign keys, unique, and not null constraints.
- Performed INNER, LEFT, RIGHT, and FULL joins and observed how they return results differently.
- Created an index to improve performance.
- Created a view to simplify future queries.

Conclusion: This task helped us understand the importance of constraints, joins, indexing, and views in databases. With screenshots included, the report clearly shows the SQL commands and their outputs.