

## Lesson 05 Demo 06

### Working with Subqueries and Keys

**Objective:** To demonstrate how to effectively utilize subqueries and work with different types of keys in MySQL for complex data operations

**Tools required:** MySQL

**Prerequisites:** None

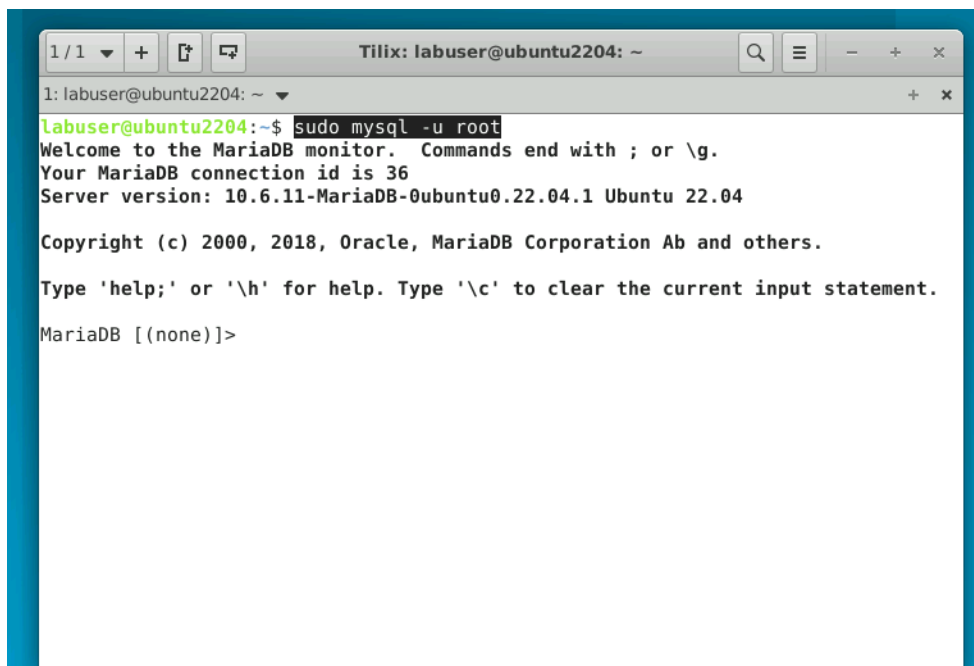
Steps to be followed:

1. Set up a database and table
2. Insert data with a subquery
3. Use subqueries in the SELECT statement
4. Work with keys

#### Step 1: Set up a database and table

1.1 Open a terminal window and access MySQL as root user:

**sudo mysql -u root**



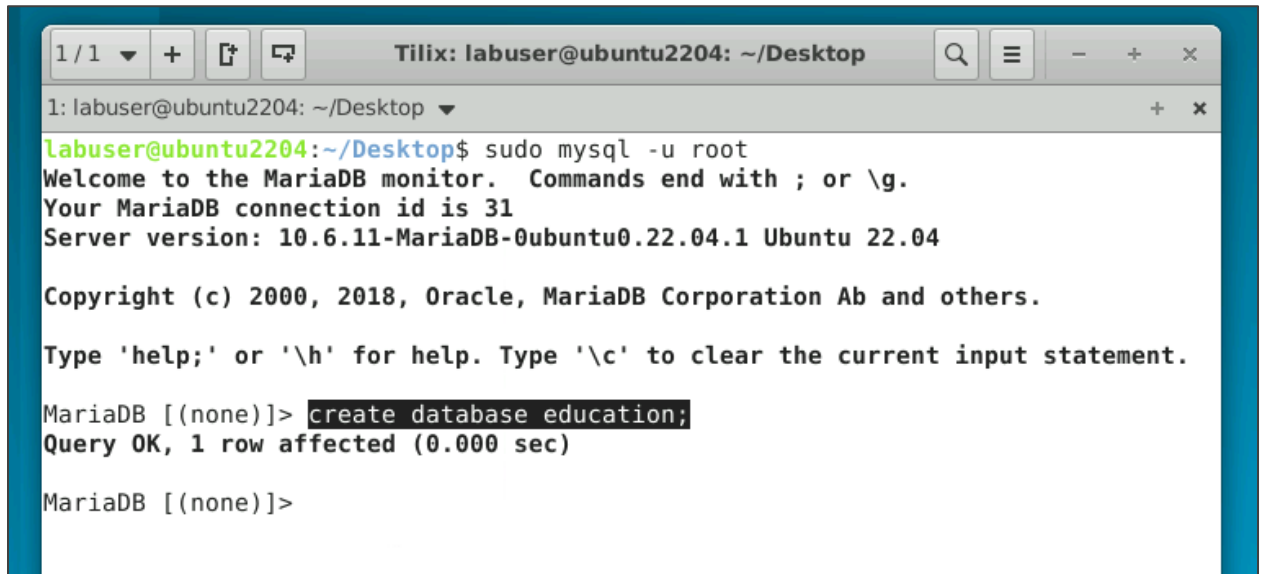
```
1 / 1 ▼ + [ ] [ ] Tilix: labuser@ubuntu2204: ~
1: labuser@ubuntu2204: ~ ▼
labuser@ubuntu2204:~$ sudo mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 36
Server version: 10.6.11-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

- 1.2 Create a new database named **education**:  
**create database education;**



A terminal window titled 'Tilix: labuser@ubuntu2204: ~/Desktop' showing a MySQL session. The user 'labuser@ubuntu2204' runs 'sudo mysql -u root'. The MySQL prompt shows 'MariaDB [(none)]>' where the command 'create database education;' is entered. The output is 'Query OK, 1 row affected (0.000 sec)'.

```
1 / 1 ▼ + [?] [x] Tilix: labuser@ubuntu2204: ~/Desktop 🔍 ☰ - + ×
1: labuser@ubuntu2204: ~/Desktop ▼ + ×
labuser@ubuntu2204:~/Desktop$ sudo mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.6.11-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04

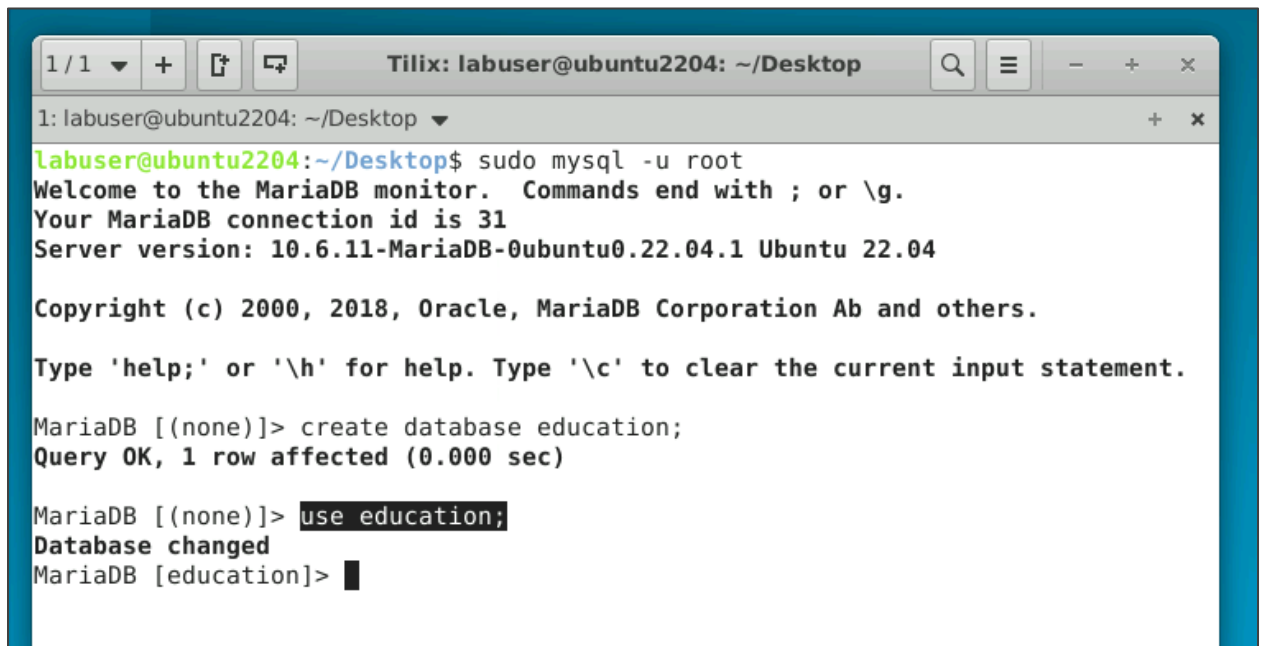
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database education;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]>
```

- 1.3 Select the **education** database:  
**use education;**



A terminal window titled 'Tilix: labuser@ubuntu2204: ~/Desktop' showing a MySQL session. The user 'labuser@ubuntu2204' runs 'sudo mysql -u root'. The MySQL prompt shows 'MariaDB [(none)]>' where the command 'create database education;' is entered, resulting in 'Query OK, 1 row affected (0.000 sec)'. Then, the command 'use education;' is entered, resulting in 'Database changed'.

```
1 / 1 ▼ + [?] [x] Tilix: labuser@ubuntu2204: ~/Desktop 🔍 ☰ - + ×
1: labuser@ubuntu2204: ~/Desktop ▼ + ×
labuser@ubuntu2204:~/Desktop$ sudo mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.6.11-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

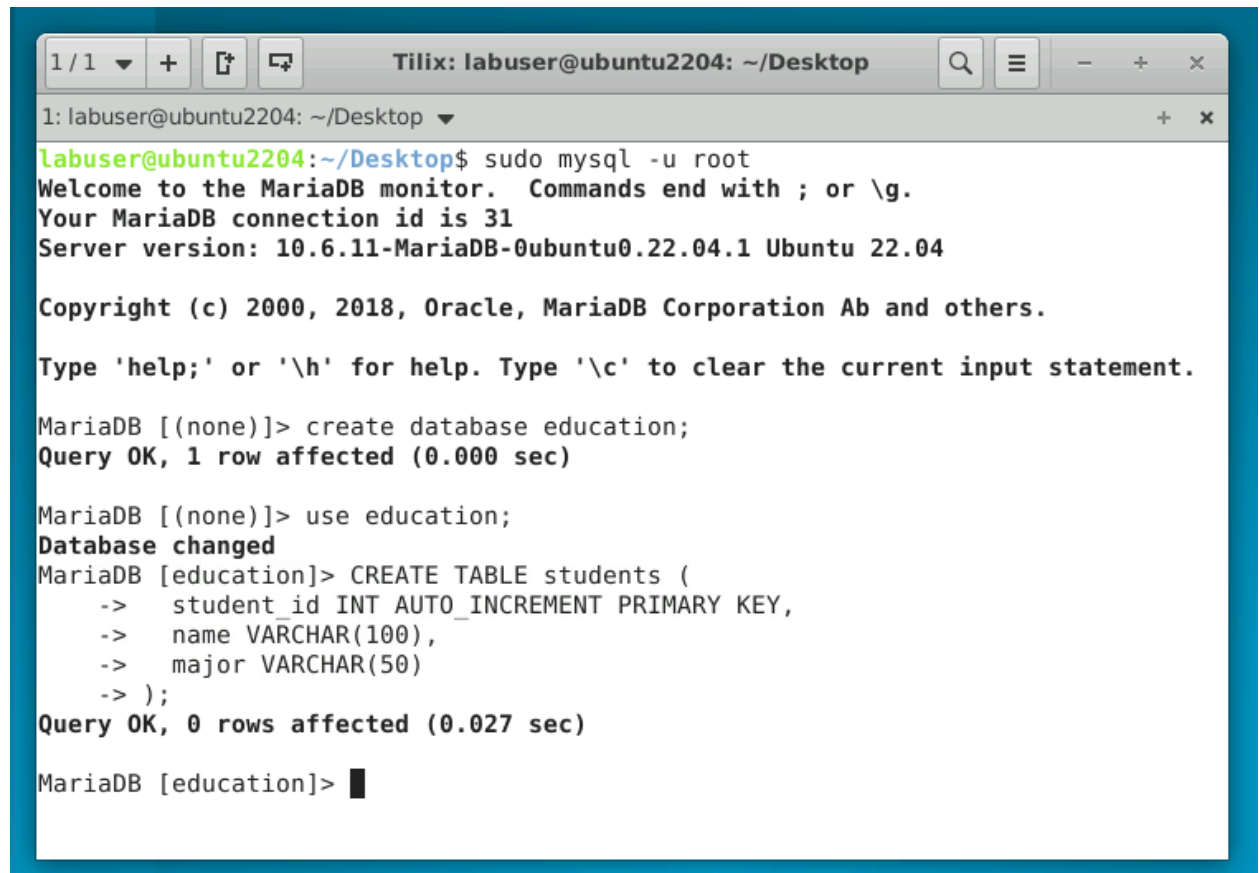
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database education;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> use education;
Database changed
MariaDB [education]> █
```

1.4 Create a **students** table with relevant fields:

```
CREATE TABLE students (  
  student_id INT AUTO_INCREMENT PRIMARY KEY,  
  name VARCHAR(100),  
  major VARCHAR(50)  
);
```

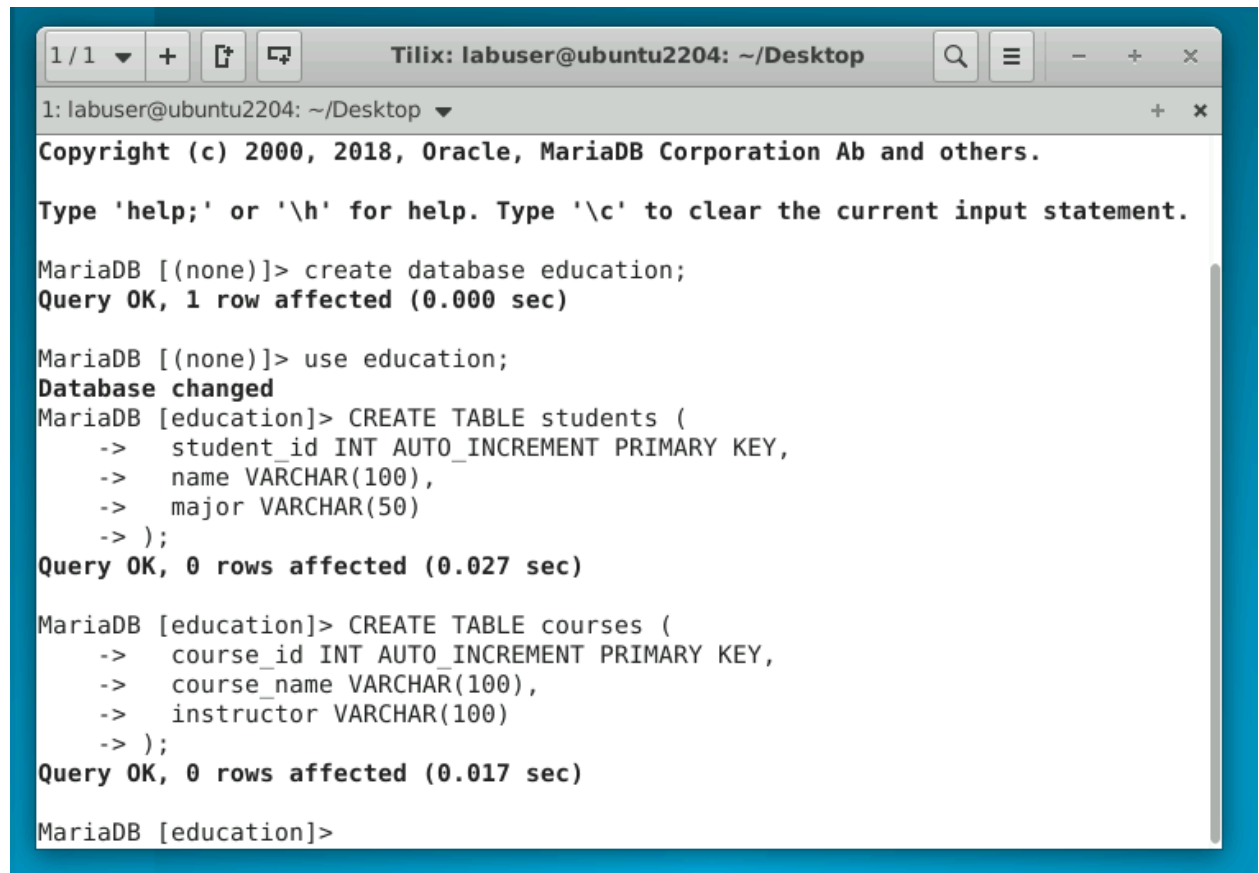


The screenshot shows a terminal window titled "Tilix: labuser@ubuntu2204: ~/Desktop". The user is logged in as labuser. The terminal shows the following commands and output:

```
labuser@ubuntu2204:~/Desktop$ sudo mysql -u root  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MariaDB connection id is 31  
Server version: 10.6.11-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MariaDB [(none)]> create database education;  
Query OK, 1 row affected (0.000 sec)  
  
MariaDB [(none)]> use education;  
Database changed  
MariaDB [education]> CREATE TABLE students (  
->   student_id INT AUTO_INCREMENT PRIMARY KEY,  
->   name VARCHAR(100),  
->   major VARCHAR(50)  
-> );  
Query OK, 0 rows affected (0.027 sec)  
  
MariaDB [education]> █
```

1.5 Create a **courses** table with relevant fields:

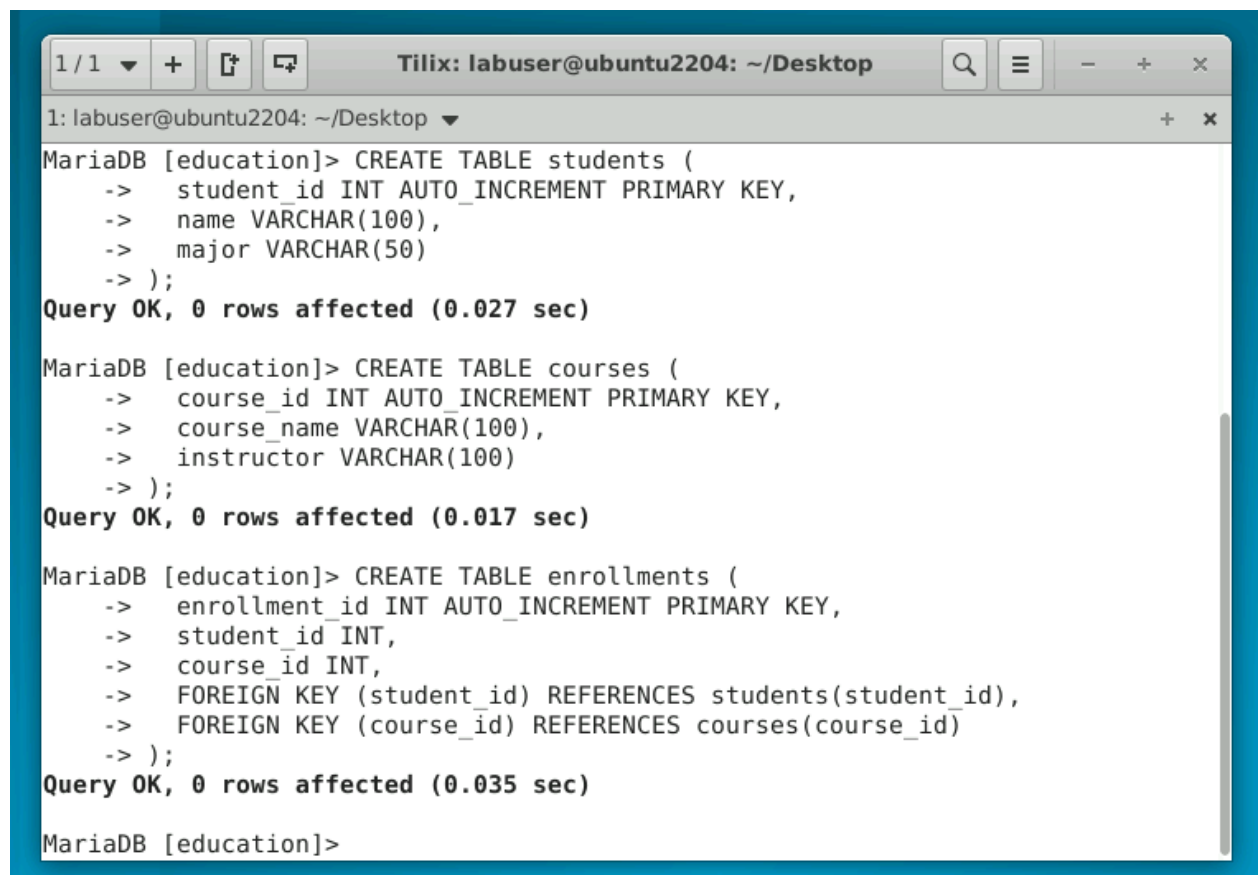
```
CREATE TABLE courses (  
  course_id INT AUTO_INCREMENT PRIMARY KEY,  
  course_name VARCHAR(100),  
  instructor VARCHAR(100)  
);
```

A screenshot of a terminal window titled 'Tilix: labuser@ubuntu2204: ~/Desktop'. The terminal shows a series of MariaDB commands and their outputs. The commands include creating a database named 'education', using that database, creating a 'students' table, and then creating a 'courses' table. The outputs show the success of each command, including the number of rows affected and the execution time.

```
1 / 1  +  [?] [x]  Tilix: labuser@ubuntu2204: ~/Desktop  🔍  ≡  -  +  x  
1: labuser@ubuntu2204: ~/Desktop  +  x  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MariaDB [(none)]> create database education;  
Query OK, 1 row affected (0.000 sec)  
  
MariaDB [(none)]> use education;  
Database changed  
MariaDB [education]> CREATE TABLE students (  
->   student_id INT AUTO_INCREMENT PRIMARY KEY,  
->   name VARCHAR(100),  
->   major VARCHAR(50)  
-> );  
Query OK, 0 rows affected (0.027 sec)  
  
MariaDB [education]> CREATE TABLE courses (  
->   course_id INT AUTO_INCREMENT PRIMARY KEY,  
->   course_name VARCHAR(100),  
->   instructor VARCHAR(100)  
-> );  
Query OK, 0 rows affected (0.017 sec)  
  
MariaDB [education]>
```

1.6 Create an **enrollments** table to link **students** with **courses**:

```
CREATE TABLE enrollments (  
  enrollment_id INT AUTO_INCREMENT PRIMARY KEY,  
  student_id INT,  
  course_id INT,  
  FOREIGN KEY (student_id) REFERENCES students(student_id),  
  FOREIGN KEY (course_id) REFERENCES courses(course_id)  
);
```



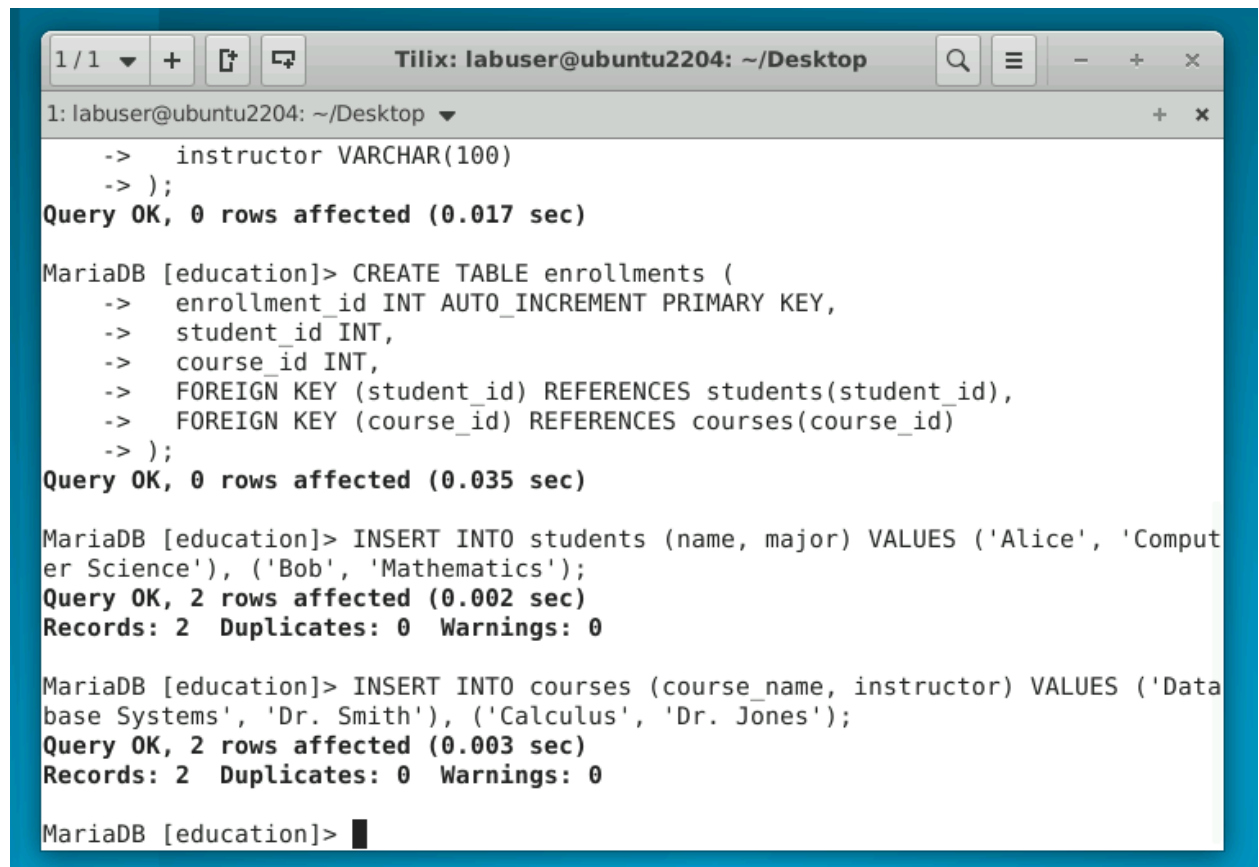
The screenshot shows a terminal window titled "Tilix: labuser@ubuntu2204: ~/Desktop". The terminal displays the following SQL commands and their outputs:

```
1: labuser@ubuntu2204: ~/Desktop  
MariaDB [education]> CREATE TABLE students (  
->   student_id INT AUTO_INCREMENT PRIMARY KEY,  
->   name VARCHAR(100),  
->   major VARCHAR(50)  
-> );  
Query OK, 0 rows affected (0.027 sec)  
  
MariaDB [education]> CREATE TABLE courses (  
->   course_id INT AUTO_INCREMENT PRIMARY KEY,  
->   course_name VARCHAR(100),  
->   instructor VARCHAR(100)  
-> );  
Query OK, 0 rows affected (0.017 sec)  
  
MariaDB [education]> CREATE TABLE enrollments (  
->   enrollment_id INT AUTO_INCREMENT PRIMARY KEY,  
->   student_id INT,  
->   course_id INT,  
->   FOREIGN KEY (student_id) REFERENCES students(student_id),  
->   FOREIGN KEY (course_id) REFERENCES courses(course_id)  
-> );  
Query OK, 0 rows affected (0.035 sec)  
  
MariaDB [education]>
```

1.7 Insert data into the **students** and **courses** tables:

```
INSERT INTO students (name, major) VALUES ('Alice', 'Computer Science'), ('Bob', 'Mathematics');
```

```
INSERT INTO courses (course_name, instructor) VALUES ('Database Systems', 'Dr. Smith'), ('Calculus', 'Dr. Jones');
```



```
1 / 1 + [ ] [ ] Tilix: labuser@ubuntu2204: ~/Desktop
1: labuser@ubuntu2204: ~/Desktop
-> instructor VARCHAR(100)
-> );
Query OK, 0 rows affected (0.017 sec)

MariaDB [education]> CREATE TABLE enrollments (
-> enrollment_id INT AUTO_INCREMENT PRIMARY KEY,
-> student_id INT,
-> course_id INT,
-> FOREIGN KEY (student_id) REFERENCES students(student_id),
-> FOREIGN KEY (course_id) REFERENCES courses(course_id)
-> );
Query OK, 0 rows affected (0.035 sec)

MariaDB [education]> INSERT INTO students (name, major) VALUES ('Alice', 'Computer Science'), ('Bob', 'Mathematics');
Query OK, 2 rows affected (0.002 sec)
Records: 2 Duplicates: 0 Warnings: 0

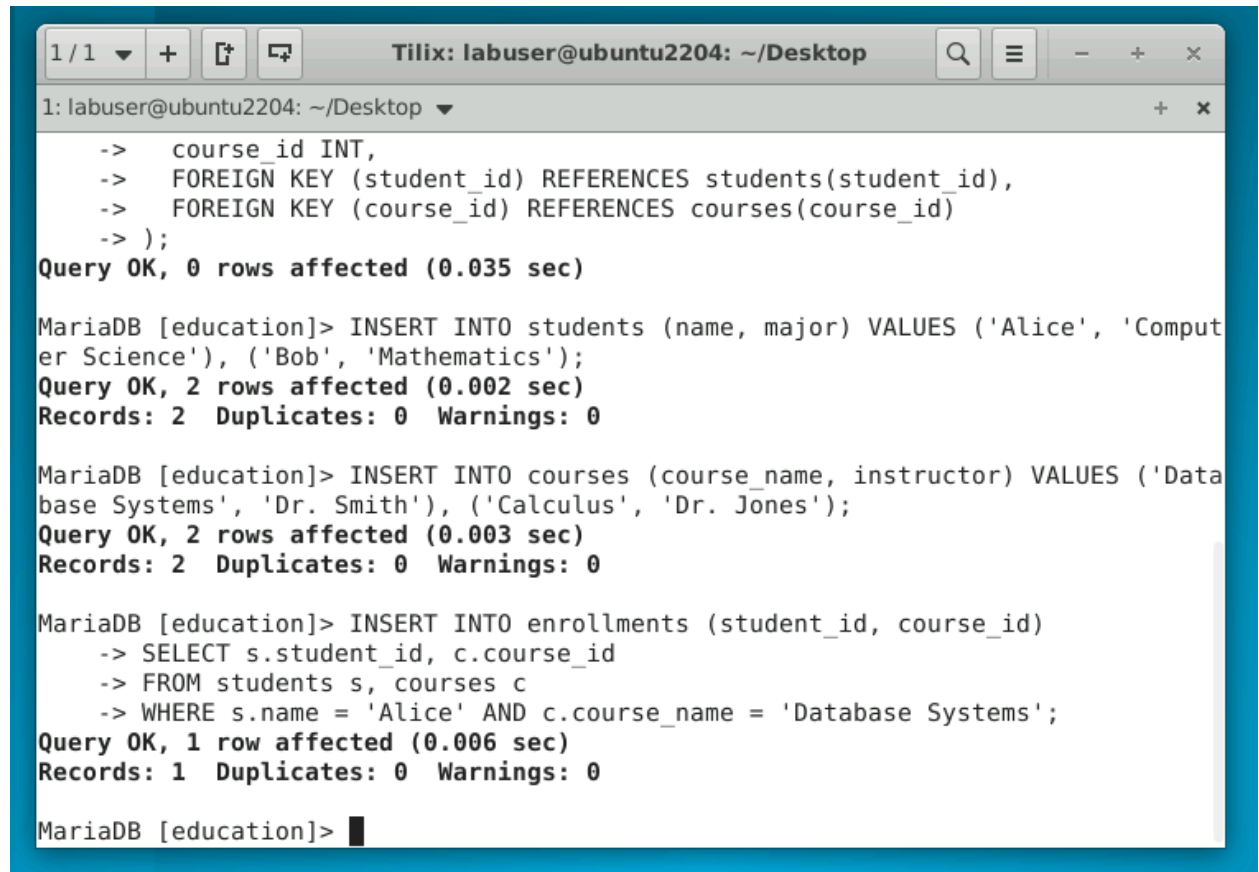
MariaDB [education]> INSERT INTO courses (course_name, instructor) VALUES ('Database Systems', 'Dr. Smith'), ('Calculus', 'Dr. Jones');
Query OK, 2 rows affected (0.003 sec)
Records: 2 Duplicates: 0 Warnings: 0

MariaDB [education]> 
```

## Step 2: Insert data with a subquery

2.1 Insert data into the **enrollments** table using a subquery:

```
INSERT INTO enrollments (student_id, course_id)
SELECT s.student_id, c.course_id
FROM students s, courses c
WHERE s.name = 'Alice' AND c.course_name = 'Database Systems';
```



```
1 / 1  +  [?] [x]  Tilix: labuser@ubuntu2204: ~/Desktop  🔍  ≡  -  +  x
1: labuser@ubuntu2204: ~/Desktop  +  x
->  course_id INT,
->  FOREIGN KEY (student_id) REFERENCES students(student_id),
->  FOREIGN KEY (course_id) REFERENCES courses(course_id)
-> );
Query OK, 0 rows affected (0.035 sec)

MariaDB [education]> INSERT INTO students (name, major) VALUES ('Alice', 'Computer Science'), ('Bob', 'Mathematics');
Query OK, 2 rows affected (0.002 sec)
Records: 2  Duplicates: 0  Warnings: 0

MariaDB [education]> INSERT INTO courses (course_name, instructor) VALUES ('Database Systems', 'Dr. Smith'), ('Calculus', 'Dr. Jones');
Query OK, 2 rows affected (0.003 sec)
Records: 2  Duplicates: 0  Warnings: 0

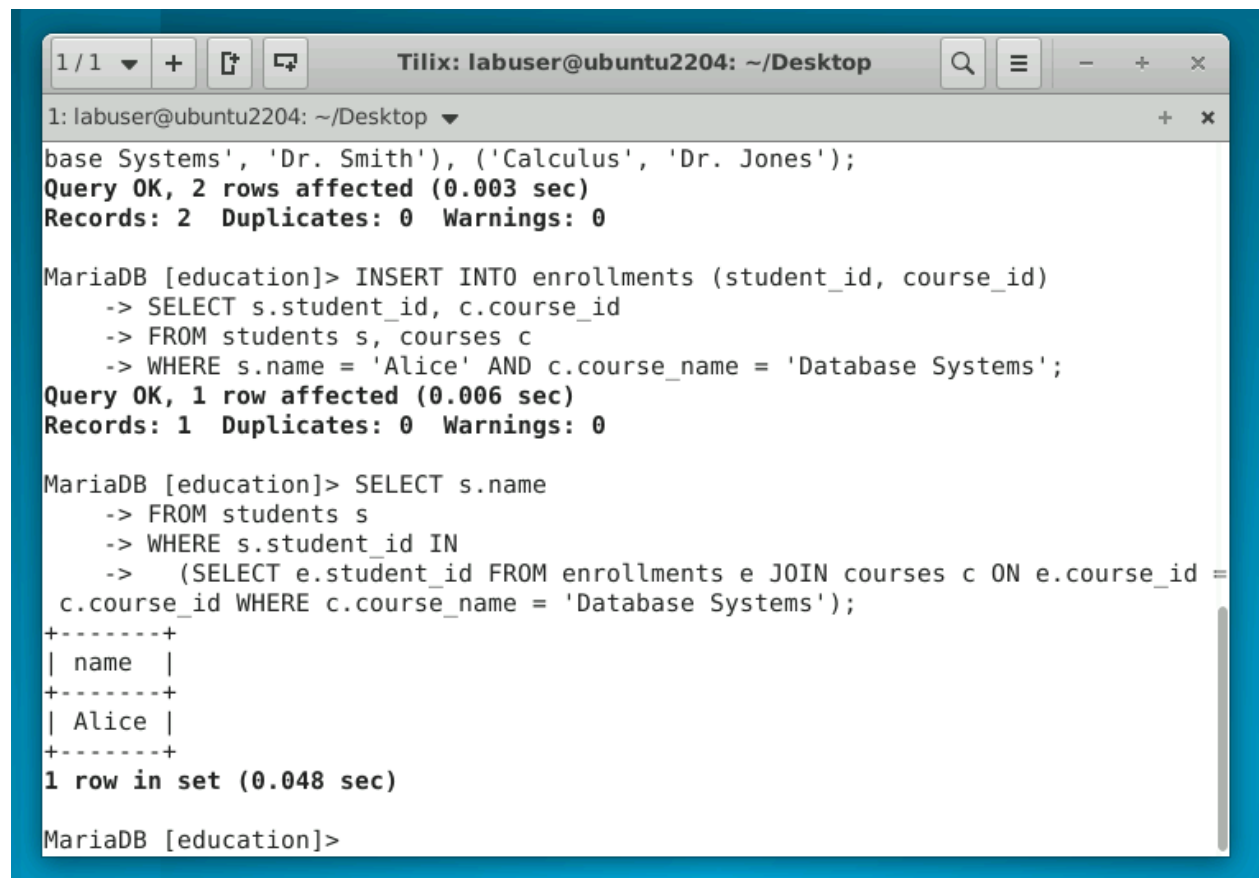
MariaDB [education]> INSERT INTO enrollments (student_id, course_id)
-> SELECT s.student_id, c.course_id
-> FROM students s, courses c
-> WHERE s.name = 'Alice' AND c.course_name = 'Database Systems';
Query OK, 1 row affected (0.006 sec)
Records: 1  Duplicates: 0  Warnings: 0

MariaDB [education]> █
```

### Step 3: Use subqueries in the SELECT statement

3.1 Retrieve students enrolled for the **Database Systems** course:

```
SELECT s.name
FROM students s
WHERE s.student_id IN
  (SELECT e.student_id FROM enrollments e JOIN courses c ON e.course_id =
c.course_id WHERE c.course_name = 'Database Systems');
```



The screenshot shows a terminal window titled 'Tilix: labuser@ubuntu2204: ~/Desktop'. The terminal displays the following SQL commands and their outputs:

```
1: labuser@ubuntu2204: ~/Desktop
base Systems', 'Dr. Smith'), ('Calculus', 'Dr. Jones');
Query OK, 2 rows affected (0.003 sec)
Records: 2  Duplicates: 0  Warnings: 0

MariaDB [education]> INSERT INTO enrollments (student_id, course_id)
-> SELECT s.student_id, c.course_id
-> FROM students s, courses c
-> WHERE s.name = 'Alice' AND c.course_name = 'Database Systems';
Query OK, 1 row affected (0.006 sec)
Records: 1  Duplicates: 0  Warnings: 0

MariaDB [education]> SELECT s.name
-> FROM students s
-> WHERE s.student_id IN
->   (SELECT e.student_id FROM enrollments e JOIN courses c ON e.course_id =
c.course_id WHERE c.course_name = 'Database Systems');
+-----+
| name |
+-----+
| Alice |
+-----+
1 row in set (0.048 sec)

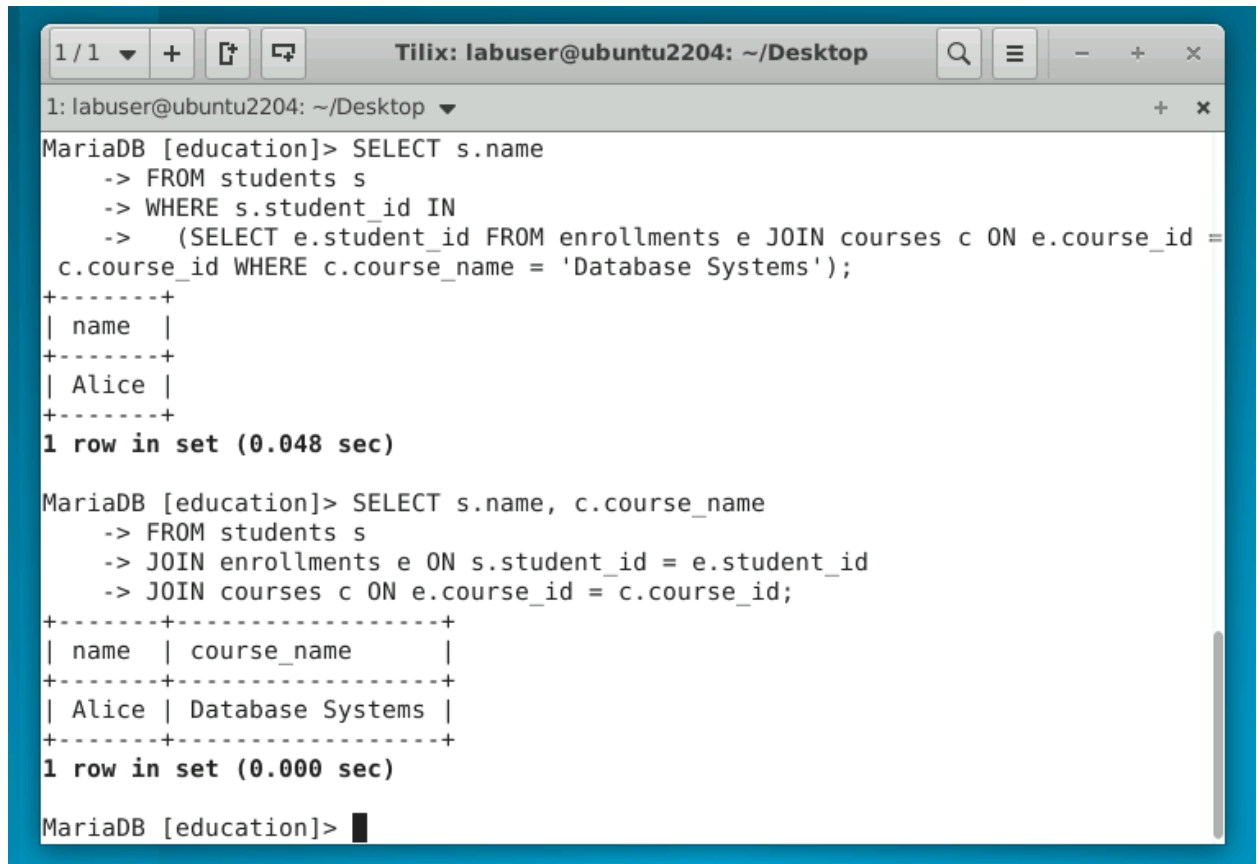
MariaDB [education]>
```



## Step 4: Work with keys

4.1 Demonstrate a JOIN operation using foreign keys:

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



The screenshot shows a terminal window titled 'Tilix: labuser@ubuntu2204: ~/Desktop'. The terminal displays two SQL queries executed in the MariaDB [education] database. The first query uses a subquery to find the student name for a specific course. The second query uses a JOIN operation to retrieve student names and course names for a specific course. Both queries return one row of data.

```
1: labuser@ubuntu2204: ~/Desktop
MariaDB [education]> SELECT s.name
-> FROM students s
-> WHERE s.student_id IN
-> (SELECT e.student_id FROM enrollments e JOIN courses c ON e.course_id =
c.course_id WHERE c.course_name = 'Database Systems');
+-----+
| name |
+-----+
| Alice |
+-----+
1 row in set (0.048 sec)

MariaDB [education]> SELECT s.name, c.course_name
-> FROM students s
-> JOIN enrollments e ON s.student_id = e.student_id
-> JOIN courses c ON e.course_id = c.course_id;
+-----+-----+
| name | course_name |
+-----+-----+
| Alice | Database Systems |
+-----+-----+
1 row in set (0.000 sec)

MariaDB [education]>
```

By following these steps, you have successfully learned how to utilize subqueries within SELECT statements and work effectively with different types of keys, such as primary and foreign keys.