

**Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology**

**(Deemed to be University Estd. u/s 3 of UGC Act, 1956)**

**WEEKLY LESSON PLAN**

**Department:** Computer Science and Engineering

**Year / Semester:** 2025-26 / Winter

**Course Code / Course Name:** 10213CS224 / Full Stack Application Development

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unit I: Database Management and Web Development basics**  **Course Outcome: • Understand and use database queries and web technologies to build interactive applications.** | | | | |
| **Week** | **Session** | **Topics to be Covered** | **Problems to Practice (Hands-On)** | **Resources** |
| Week 1 | Session 1 | Git | 1. Practice basic Git Commands 2. Upload single document to github using bash. 3. Upload and update a folder in github. 4. Create branch and push/commit one folder in Eclipse. 5. Create branch and push/commit one folder in VS Code. | <https://youtu.be/vA5TTz6BXhY> |
| Session 2 | SQL | Install SQL tool, MySQL and VS Code.  1. Create a database.  2. Create at least two tables:  \* Student (VTU Number, Name, Email, Phone, Department)  \* Course (Course Code, Course Name, Faculty Id, Student ID, Faculty Email)  3. Insert minimum 5 records into each table.  4. Select records using different CLAUSE. | <https://www.programiz.com/sql/online-compiler>  <https://www.w3schools.com> |
| Session 3 | SQL | 1. Write SELECT queries to display all records.  2. Write queries using aggregate functions.  3. Sort data and display based in ascending/descending order of the VTU number.  4. Display student records belonging to a particular department.  5. Map VTU no. with course name and faculty using joins. | <https://www.w3schools.com/> |
| Session 4 | SQL | 1. Select set of students based on course and count the number of students. 2. Select set of courses a faculty is handling and insert into the new table. 3. Select top 5 students from same department and update their phone no. with country code. | <https://youtu.be/EQbhKjBmW88> |
| Session 5 | SQL | 1. Initiate and commit a transaction. 2. Set a savepoint, perform transactions and rollback to the save point. 3. Create a user-defined function to perform simple mathematical calculations. 4. Create a user-defined function to execute a select query to return a value. | <https://www.w3schools.com/mysql/default.asp> |

Session 1:

**Prerequisites**

1. Install Git for Windows: <https://git-scm.com/download/win>
2. Install VS Code: <https://code.visualstudio.com/>
3. Restart VS Code after installing Git

**Part 1: First Time Setup**

**Open VS Code and Configure Git**

1. **Open VS Code Terminal**
   * Press Ctrl+` (backtick key, usually above Tab)
   * Or: View → Terminal
2. **Configure Git** (one-time setup)

bash

git config --global user.name "VTU#####"

git config --global user.email "your.email@example.com"

*# Verify*

git config --list

**Part 2: Create Your First Git Project**

**Method 1: Using VS Code UI (Beginner Friendly)**

**Step 1: Create Project Folder**

1. File → Open Folder
2. Navigate to: C:\Users\YourName\Documents
3. Click "New Folder" button
4. Name it: my-first-git-project
5. Click "Select Folder"

**Step 2: Initialize Git**

1. Click **Source Control** icon (left sidebar) - looks like branches
   * Or press Ctrl+Shift+G
2. Click **"Initialize Repository"** button
3. Done! You now have a Git repository

**Step 3: Create a File**

1. Click **Explorer** icon (top of left sidebar)
2. Click "New File" icon
3. Name it: README.md
4. Add content:

markdown

# My First Git Project

This is my first project using Git and VS Code!

1. Save: Ctrl+S

**Step 4: Commit Your First File**

1. Go to **Source Control** (Ctrl+Shift+G)
2. You'll see README.md under "Changes"
3. Hover over the file and click **"+"** (Stage Changes)
4. File moves to "Staged Changes"
5. In the message box at top, type: Initial commit
6. Click **✓** checkmark or press Ctrl+Enter
7. First commit done!

**Step 5: Create a Branch**

1. Look at **bottom-left corner** of VS Code
2. You'll see a branch icon with "main" (or "master")
3. Click on it
4. Select: **"Create new branch..."**
5. Type: feature/my-new-feature
6. Press Enter
7. You're now on the new branch!

**Step 6: Add More Files to Your Branch**

1. Create a new folder: Right-click in Explorer → New Folder → src
2. Inside src, create: app.js
3. Add code:

javascript

console.log("Hello from my branch!");

1. Save the file

**Step 7: Commit Changes on Branch**

1. Go to Source Control (Ctrl+Shift+G)
2. Stage the new files (click "+" on each, or "+" on "Changes" to stage all)
3. Commit message: Add src folder with app.js
4. Click ✓ checkmark

**Part 3: Connect to GitHub and Push**

**Create GitHub Repository**

1. Go to: <https://github.com>
2. Click **"+"** (top-right) → New repository
3. Name: my-first-git-project
4. **Do NOT** check "Initialize with README"
5. Click "Create repository"

**Connect VS Code to GitHub**

**Using VS Code UI:**

1. In Source Control panel, click **"..."** (three dots menu)
2. Select: **"Remote" → "Add Remote"**
3. Paste your GitHub URL:

https://github.com/yourusername/my-first-git-project.git

1. Name it: origin
2. Press Enter

**Or using Terminal:**

bash

git remote add origin https://github.com/yourusername/my-first-git-project.git

**Push to GitHub**

**Using VS Code UI:**

1. Click **"..."** menu in Source Control
2. Select: **"Push"**
3. If asked to publish branch, click **"OK"**
4. VS Code may ask for GitHub authentication - follow the prompts

**Or using Terminal:**

bash

*# Push main branch*

git push -u origin main

*# Push feature branch*

git checkout feature/my-new-feature

git push -u origin feature/my-new-feature

**Part 4: Daily Git Workflow in VS Code**

**Working on Your Project**

1. Open VS Code

2. Open your project folder

3. Check current branch (bottom-left corner)

4. Make changes to files

5. Go to Source Control (Ctrl+Shift+G)

6. Stage changes (click +)

7. Write commit message

8. Commit (click ✓)

9. Push (click ... → Push)

**Switch Between Branches**

1. Click branch name (bottom-left)
2. Select branch from list
3. Or create new branch

**Useful Keyboard Shortcuts**

| **Action** | **Shortcut** |
| --- | --- |
| Open Terminal | Ctrl+` |
| Source Control | Ctrl+Shift+G |
| Save File | Ctrl+S |
| Save All | Ctrl+K S |
| Command Palette | Ctrl+Shift+P |
| Find | Ctrl+F |
| New File | Ctrl+N |

**Part 5: Common Tasks in VS Code**

**See What Changed**

**Visual Diff:**

1. In Source Control, click on any changed file
2. VS Code shows side-by-side comparison
3. Red = removed, Green = added

**Undo Changes**

**Before Committing:**

1. Right-click file in Source Control
2. Select "Discard Changes"

**After Committing:**

* Use terminal: git reset --soft HEAD~1

**View Commit History**

**Install GitLens Extension (Recommended):**

1. Click Extensions icon (left sidebar)
2. Search: "GitLens"
3. Click "Install"
4. Now you can see commit history, blame, and more!

**Without Extension:**

* Terminal: git log --oneline

**Part 6: Real Example - Complete Workflow**

**Scenario: Add a New Feature**

bash

*# In VS Code Terminal (Ctrl+`)*

*# 1. Make sure you're on main branch*

git checkout main

*# 2. Pull latest changes*

git pull origin main

*# 3. Create feature branch*

git checkout -b feature/add-login-page

*# Now in VS Code:*

*# 4. Create new files:*

*# - src/login.html*

*# - src/login.css*

*# - src/login.js*

*# 5. Go to Source Control (Ctrl+Shift+G)*

*# 6. Stage all files (click + on "Changes")*

*# 7. Commit message: "Add login page with styling and validation"*

*# 8. Click ✓ to commit*

*# 9. Push to GitHub*

git push -u origin feature/add-login-page

*# 10. Go to GitHub and create Pull Request*

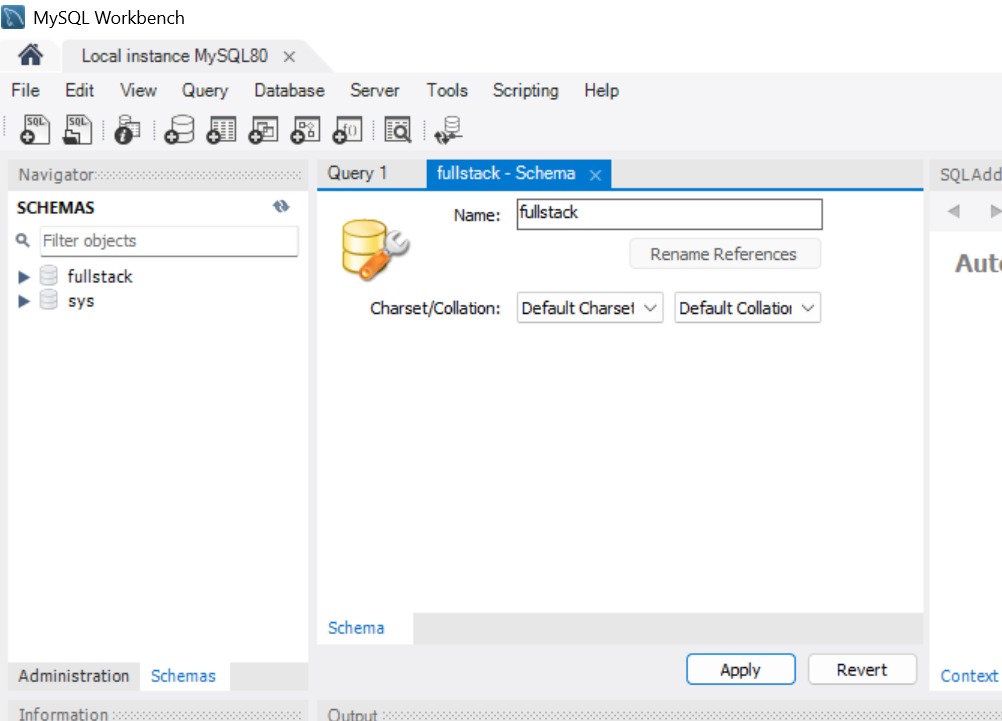
**Session 2:**

Install SQL tool, MySQL and VS Code.

1. Create a database.

**CREATE DATABASE fullstack;**

**USE fullstack;**



2. Create at least two tables:

Table 1: Student (VTU Number, Name, Email, Phone, Department)

**CREATE TABLE Student (**

**VTU\_Number VARCHAR(20) PRIMARY KEY,**

**Name VARCHAR(100),**

**Email VARCHAR(100),**

**Phone VARCHAR(15),**

**Department VARCHAR(50)**

**);**

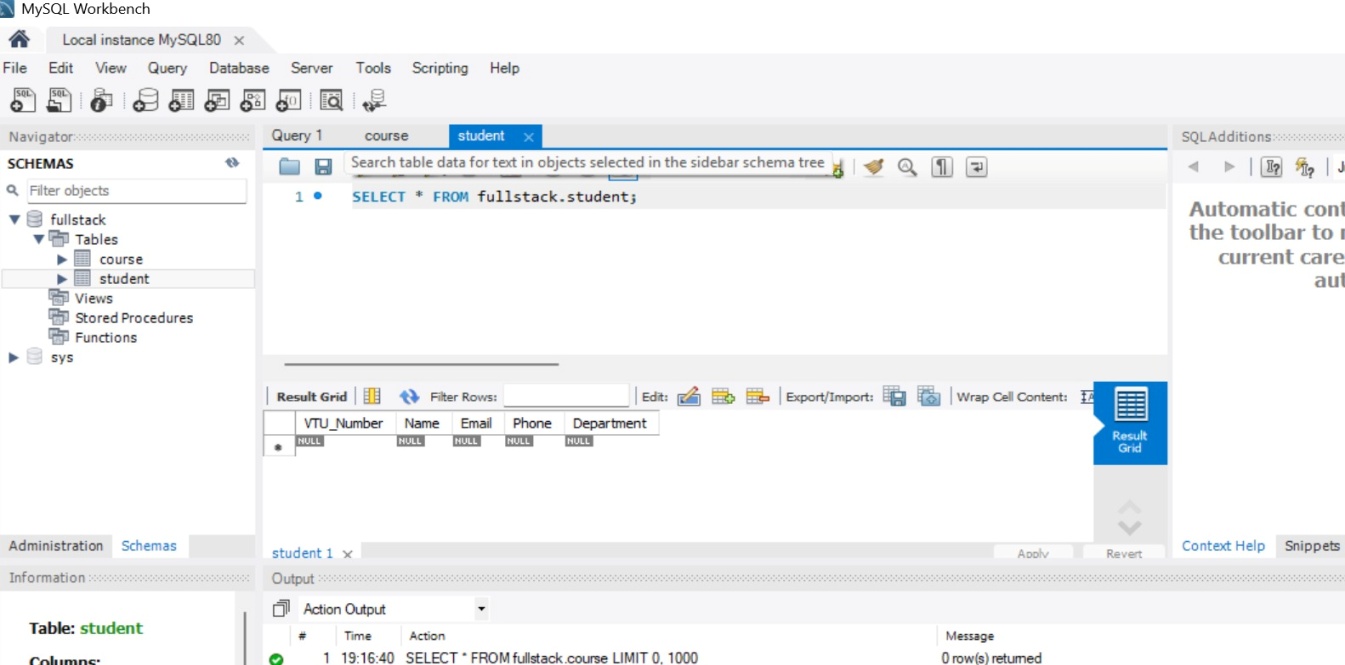


Table 2: Course (Course Code, Course Name, Faculty Id, Student ID, Faculty Email)

**CREATE TABLE Course (**

**Course\_Code VARCHAR(10) PRIMARY KEY,**

**Course\_Name VARCHAR(100),**

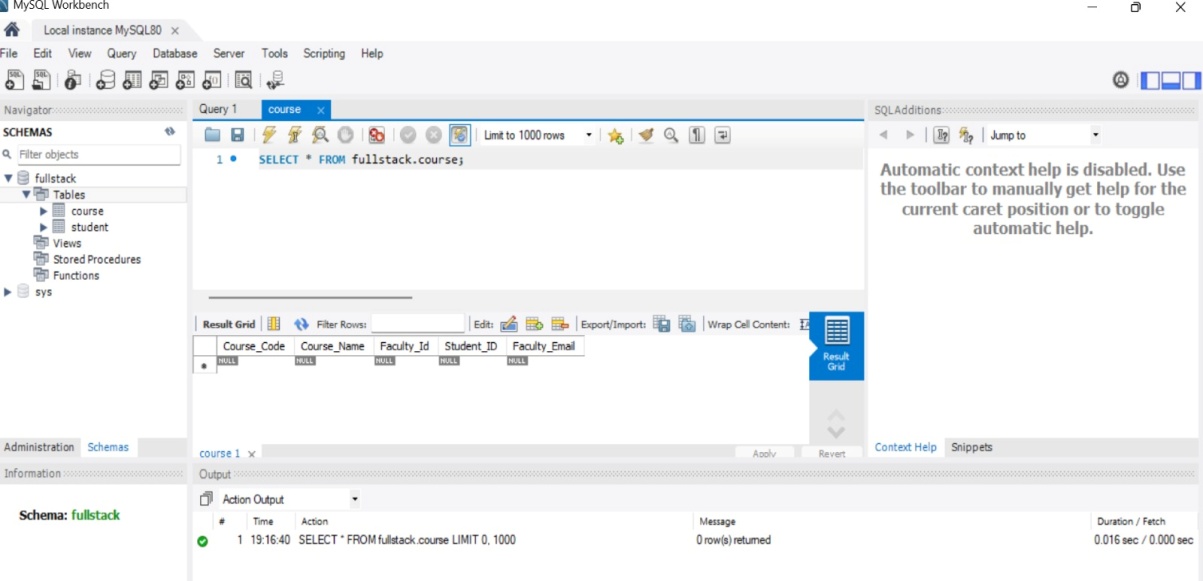
**Faculty\_Id INT,**

**Student\_ID VARCHAR(20),**

**Faculty\_Email VARCHAR(100),**

**FOREIGN KEY (Student\_ID) REFERENCES Student(VTU\_Number)**

**);**



1. Insert minimum 5 records into each table.

**-- Inserting into Student Table**

**INSERT INTO Student VALUES**

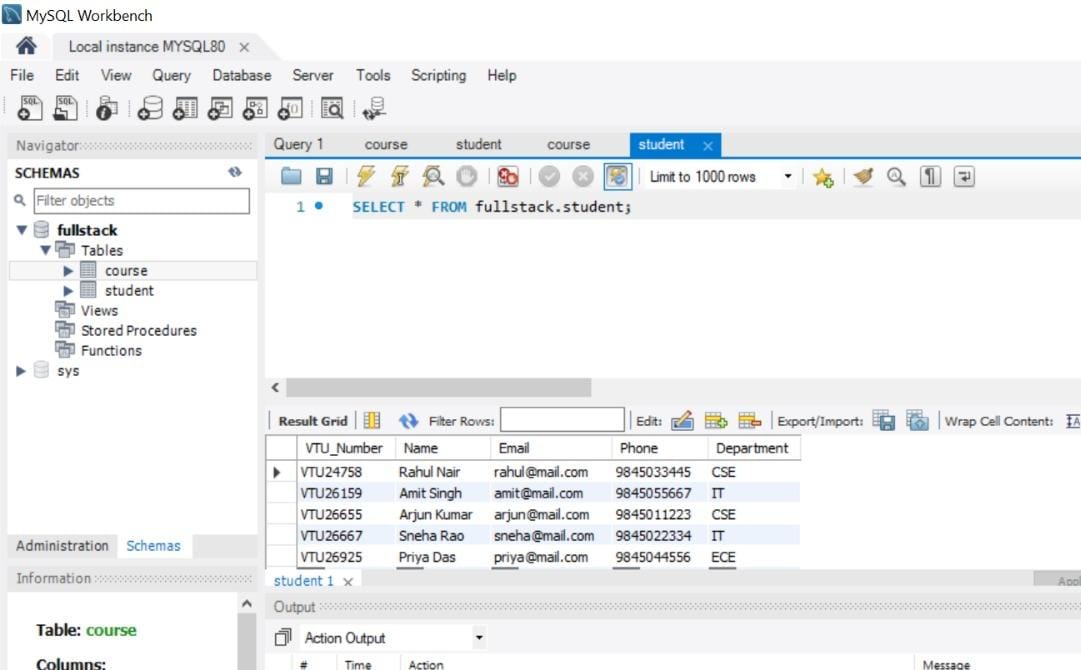
**('VTU26655', 'Arjun Kumar', 'arjun@mail.com', '9845011223', 'CSE'),**

**('VTU26667', 'Sneha Rao', 'sneha@mail.com', '9845022334', 'IT'),**

**('VTU24758', 'Rahul Nair', 'rahul@mail.com', '9845033445', 'CSE'),**

**('VTU26925', 'Priya Das', 'priya@mail.com', '9845044556', 'ECE'),**

**('VTU26159', 'Amit Singh', 'amit@mail.com', '9845055667', 'IT');**



**-- Inserting into Course Table**

**INSERT INTO Course VALUES**

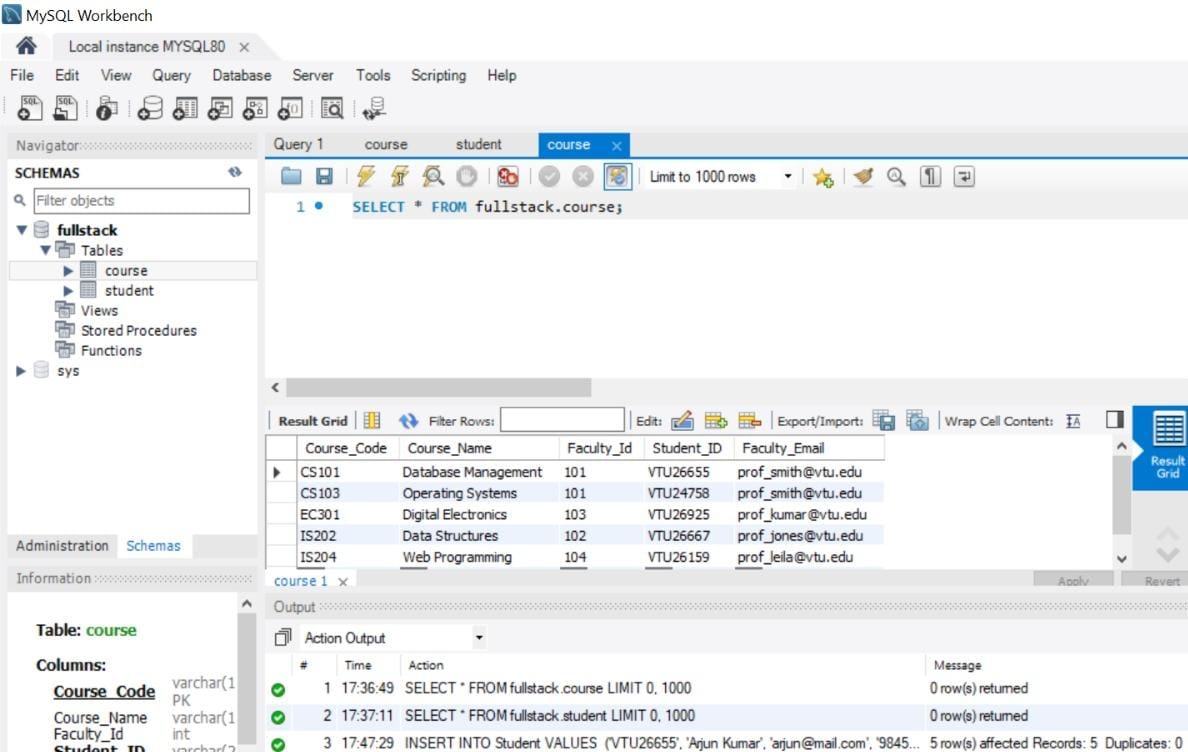
**('CS101', 'Database Management', 101, 'VTU26655', 'prof\_smith@vtu.edu'),**

**('IS202', 'Data Structures', 102, 'VTU26667', 'prof\_jones@vtu.edu'),**

**('CS103', 'Operating Systems', 101, 'VTU24758', 'prof\_smith@vtu.edu'),**

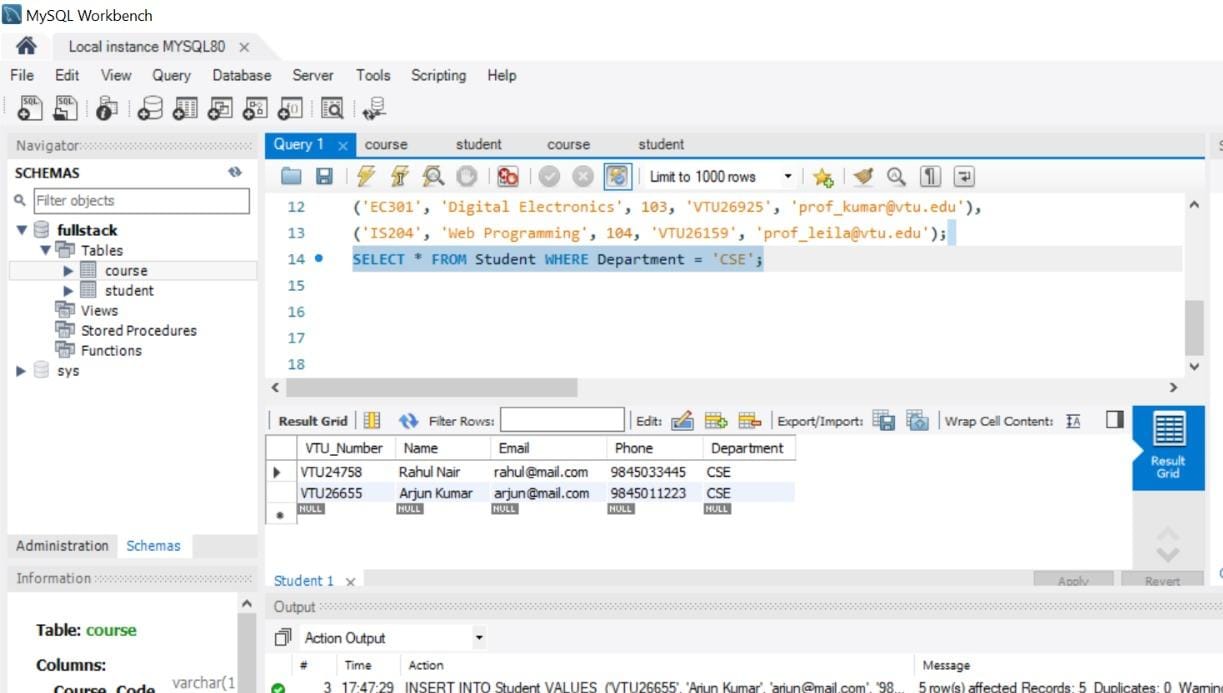
**('EC301', 'Digital Electronics', 103, 'VTU26925', 'prof\_kumar@vtu.edu'),**

**('IS204', 'Web Programming', 104, 'VTU26159', 'prof\_leila@vtu.edu');**



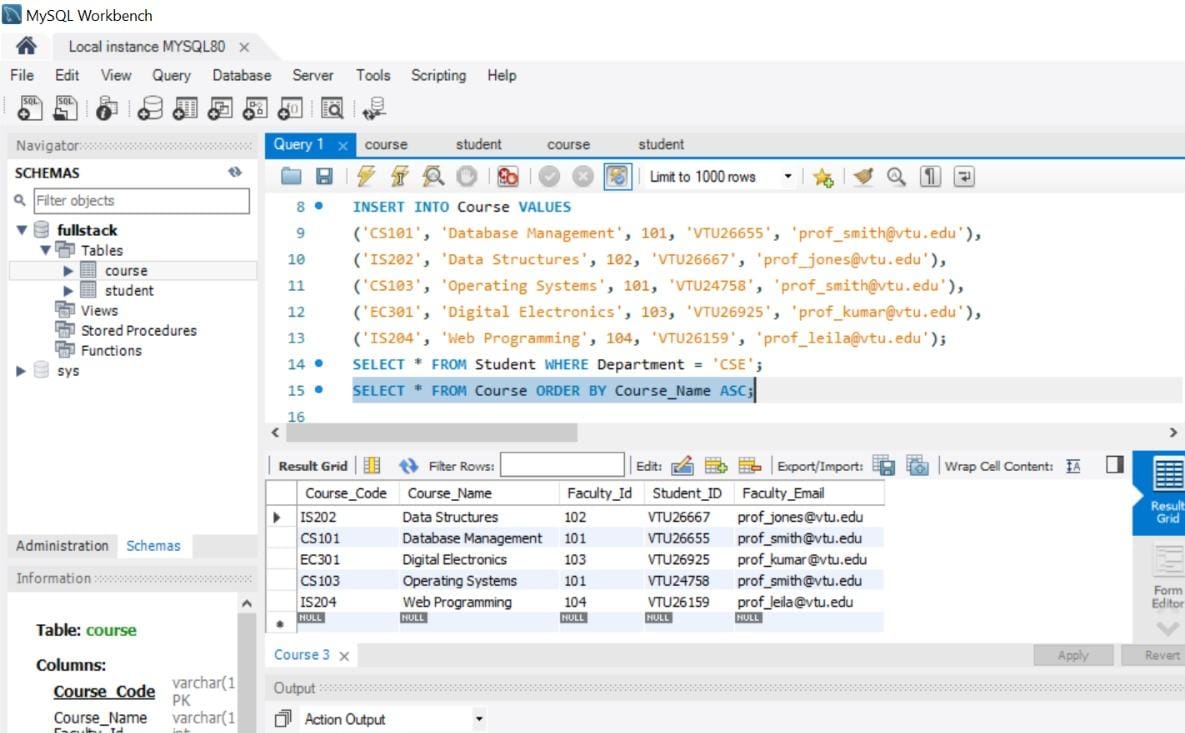
1. Select records using different CLAUSE.
2. **WHERE clause:**

SELECT \* FROM Student WHERE Department = 'CSE';



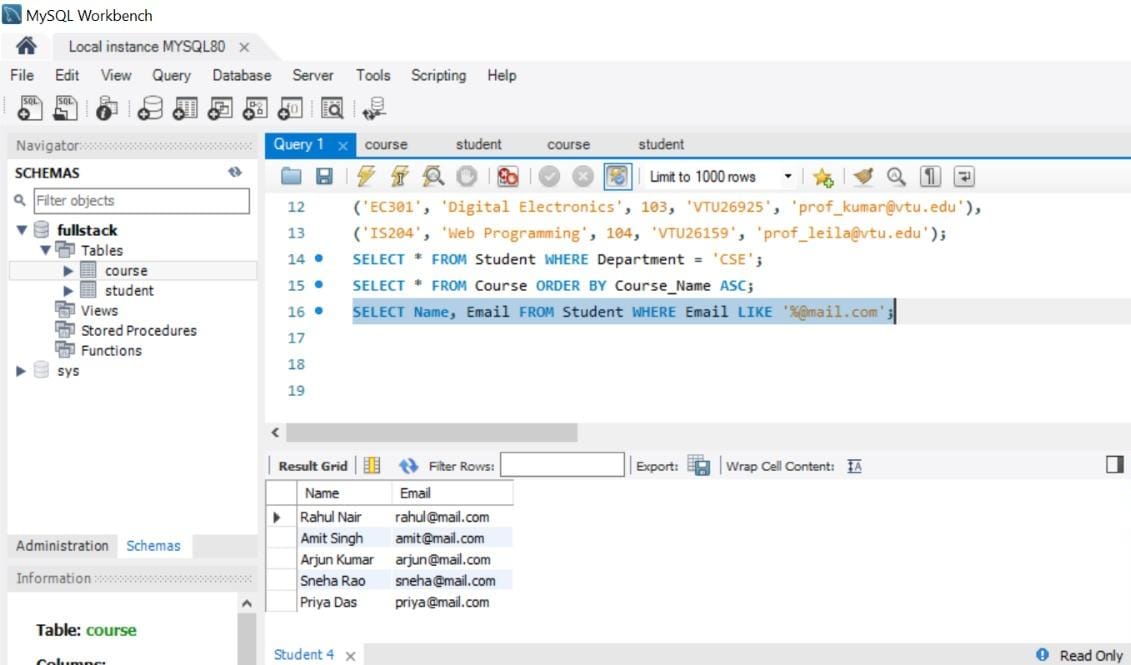
1. **ORDER BY Clause (Sorting):**

SELECT \* FROM Course ORDER BY Course\_Name ASC;



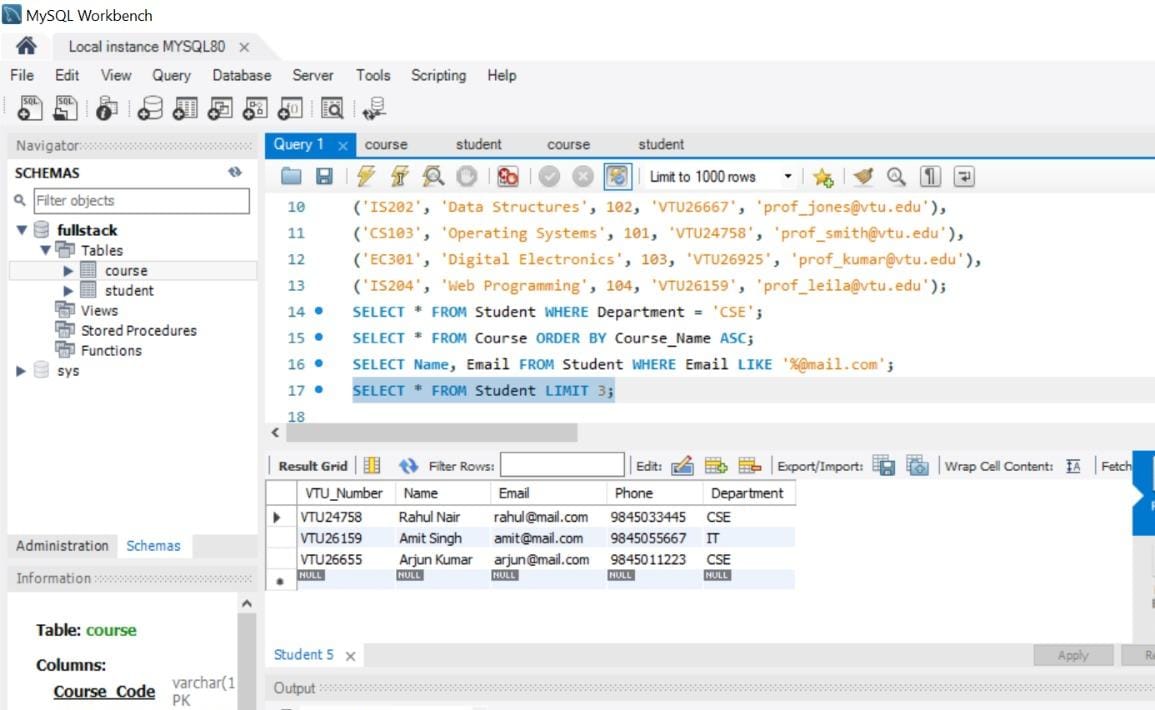
1. **LIKE Clause (Pattern Matching):**

SELECT Name, Email FROM Student WHERE Email LIKE '%@mail.com';



1. **LIMIT Clause (Restricting Results):**

SELECT \* FROM Student LIMIT 3;



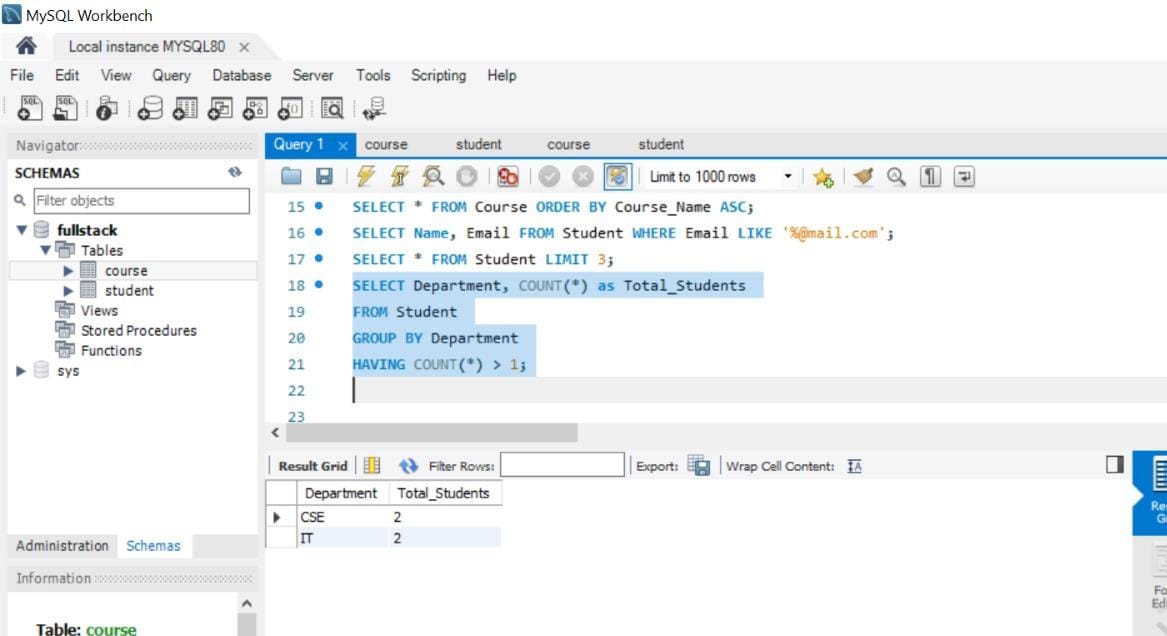
1. **GROUP BY & HAVING Clause (Aggregating):**

SELECT Department, COUNT(\*) as Total\_Students

FROM Student

GROUP BY Department

HAVING COUNT(\*) > 1;

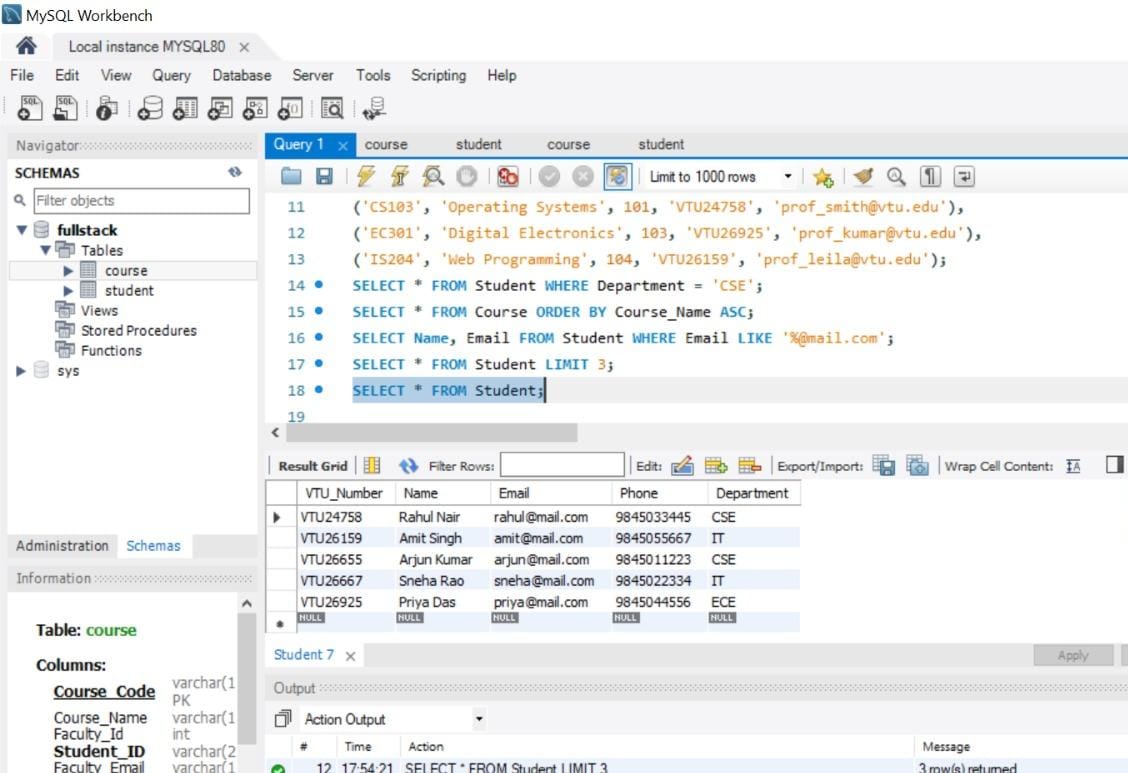


**Session 3:**

1. Write SELECT queries to display all records.

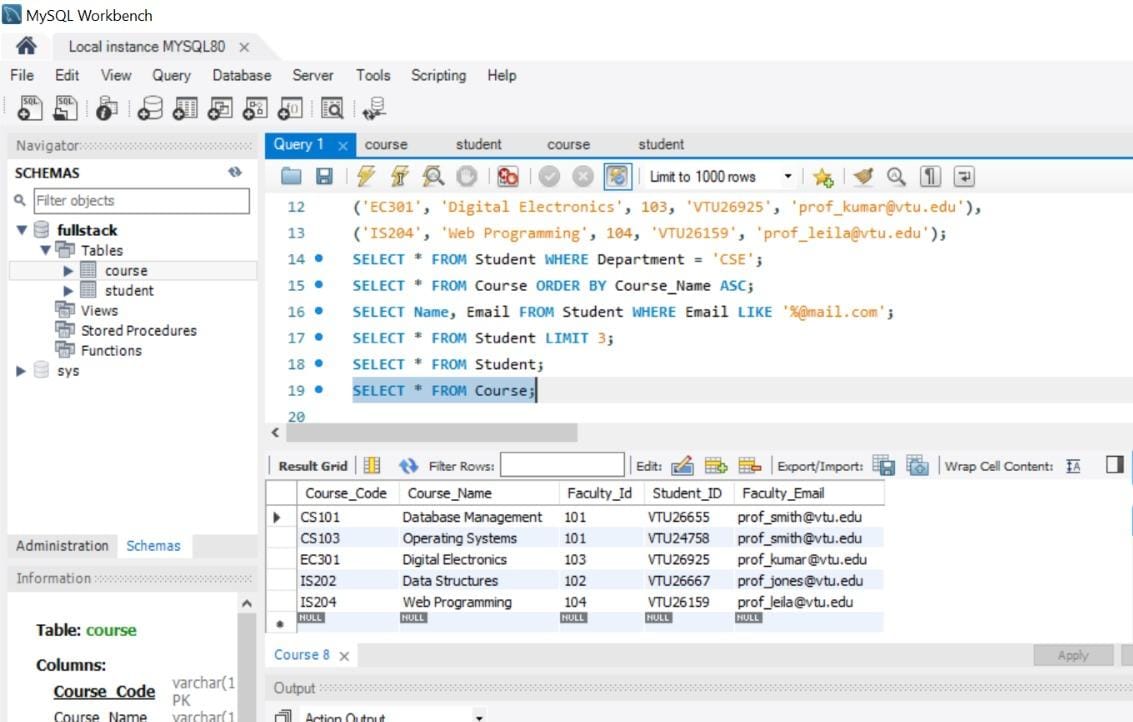
**-- Display all students**

**SELECT \* FROM Student;**



**-- Display all courses**

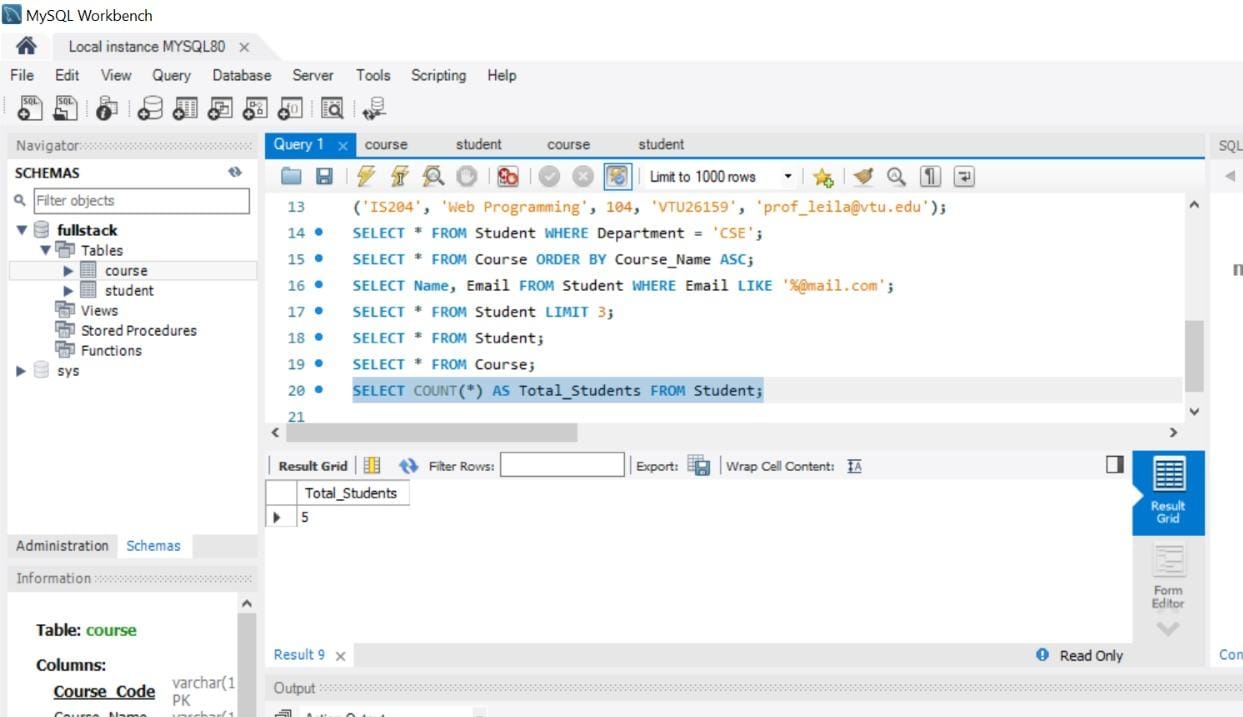
**SELECT \* FROM Course;**



1. Write queries using aggregate functions.

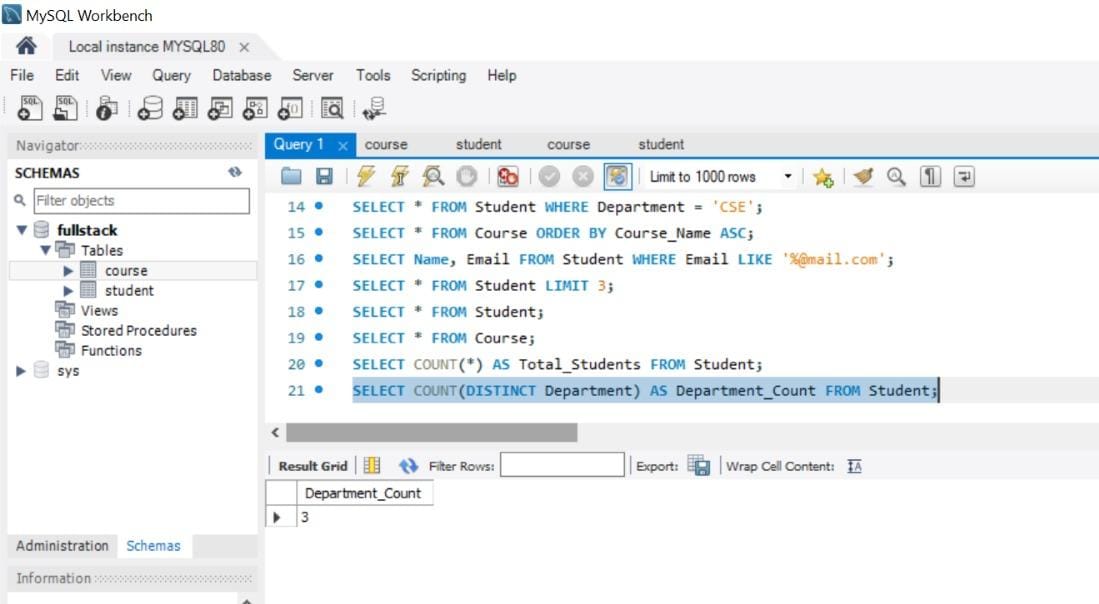
**-- Count total number of students**

**SELECT COUNT(\*) AS Total\_Students FROM Student;**



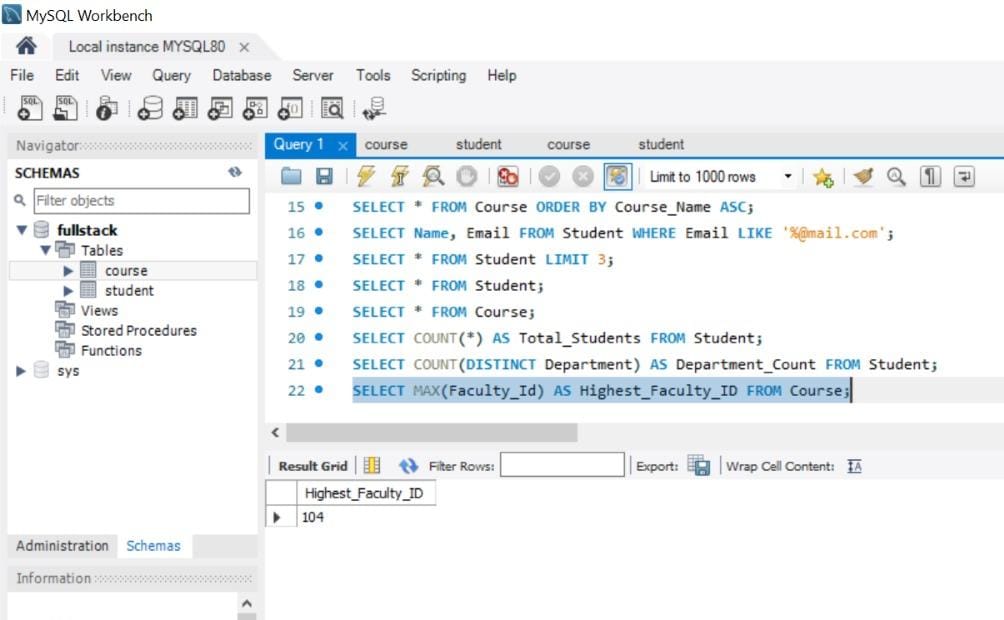
**-- Count unique departments**

**SELECT COUNT(DISTINCT Department) AS Department\_Count FROM Student;**



**-- Find the highest Faculty ID assigned**

**SELECT MAX(Faculty\_Id) AS Highest\_Faculty\_ID FROM Course;**



1. Sort data and display based in ascending/descending order of the VTU number.

**-- Sort - Ascending order (Default)**

**SELECT \* FROM Student**

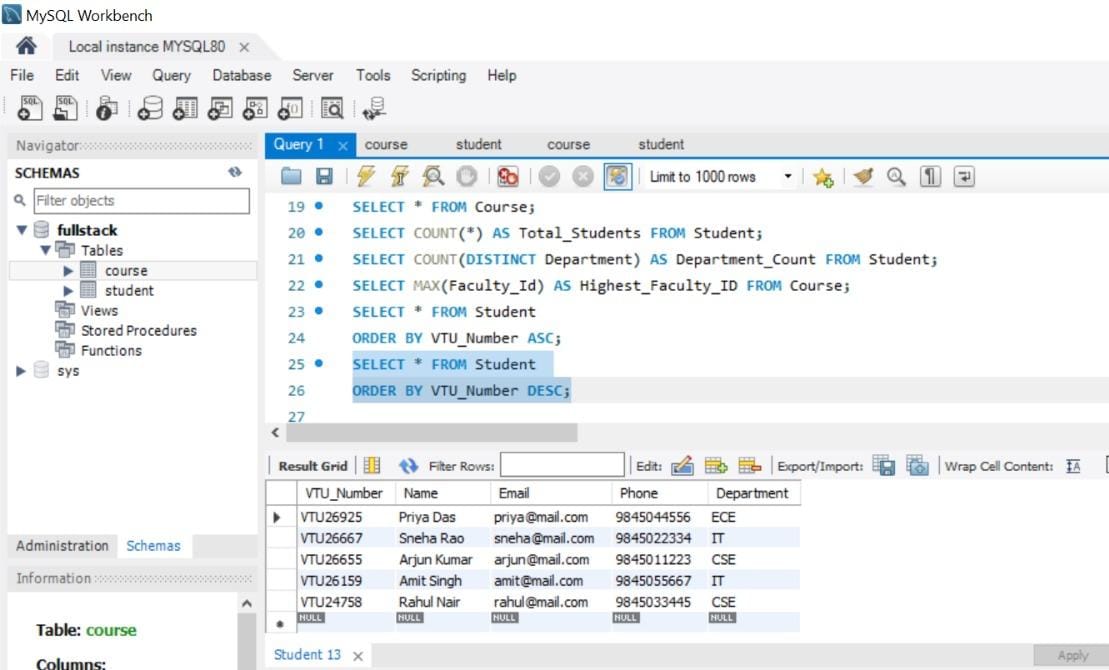
**ORDER BY VTU\_Number ASC;**



**-- Sort - in Descending order**

**SELECT \* FROM Student**

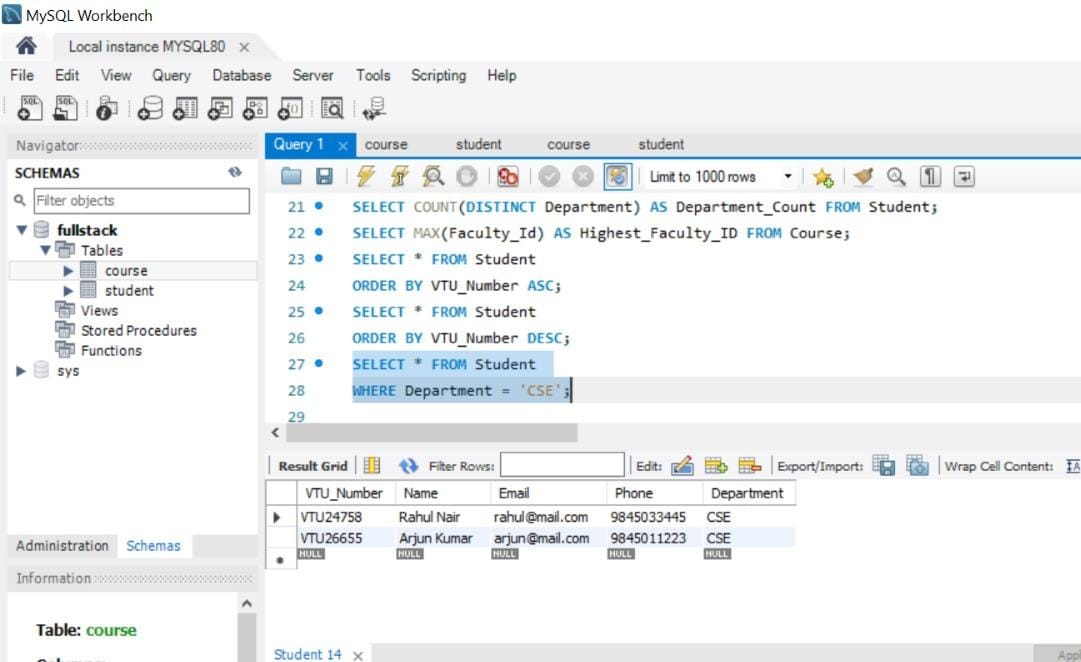
**ORDER BY VTU\_Number DESC;**



1. Display student records belonging to a particular department.

**SELECT \* FROM Student**

**WHERE Department = 'CSE';**



1. Map VTU no. with course name and faculty using joins.

**Joins Summary:**

|  |  |
| --- | --- |
| **Join Type** | **Records Included** |
| **Inner** | **Only matches found in both tables.** |
| **Left** | **Everything from Left + matches from Right.** |
| **Right** | **Everything from Right + matches from Left.** |
| **Full** | **Everything from both tables.** |
| **Cross** | **Every possible combination of both tables.** |

**Inner Join:**

**SELECT**

**s.VTU\_Number,**

**s.Name AS Student\_Name,**

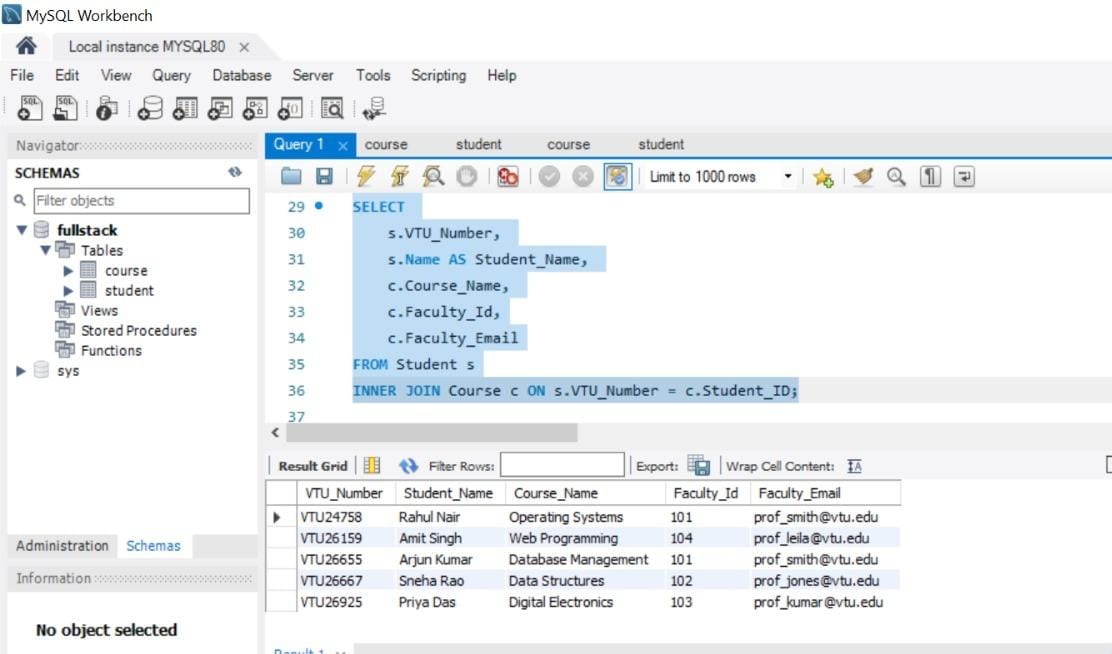
**c.Course\_Name,**

**c.Faculty\_Id,**

**c.Faculty\_Email**

**FROM Student s**

**INNER JOIN Course c ON s.VTU\_Number = c.Student\_ID;**



**Left Join:**

**SELECT**

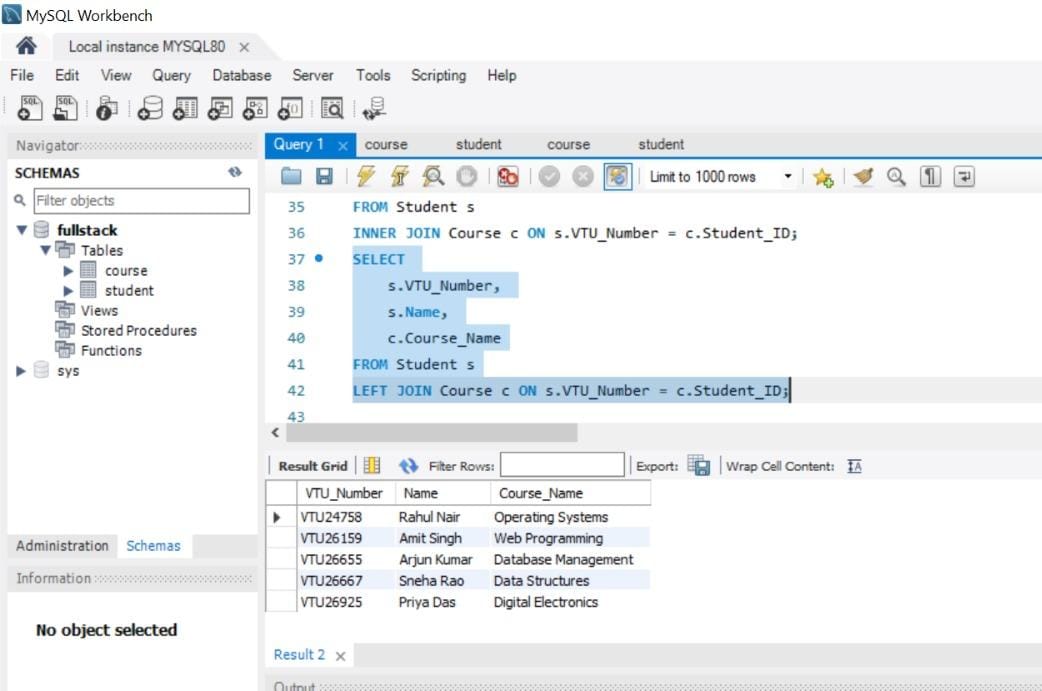
**s.VTU\_Number,**

**s.Name,**

**c.Course\_Name**

**FROM Student s**

**LEFT JOIN Course c ON s.VTU\_Number = c.Student\_ID;**



**Right Join:**

**SELECT**

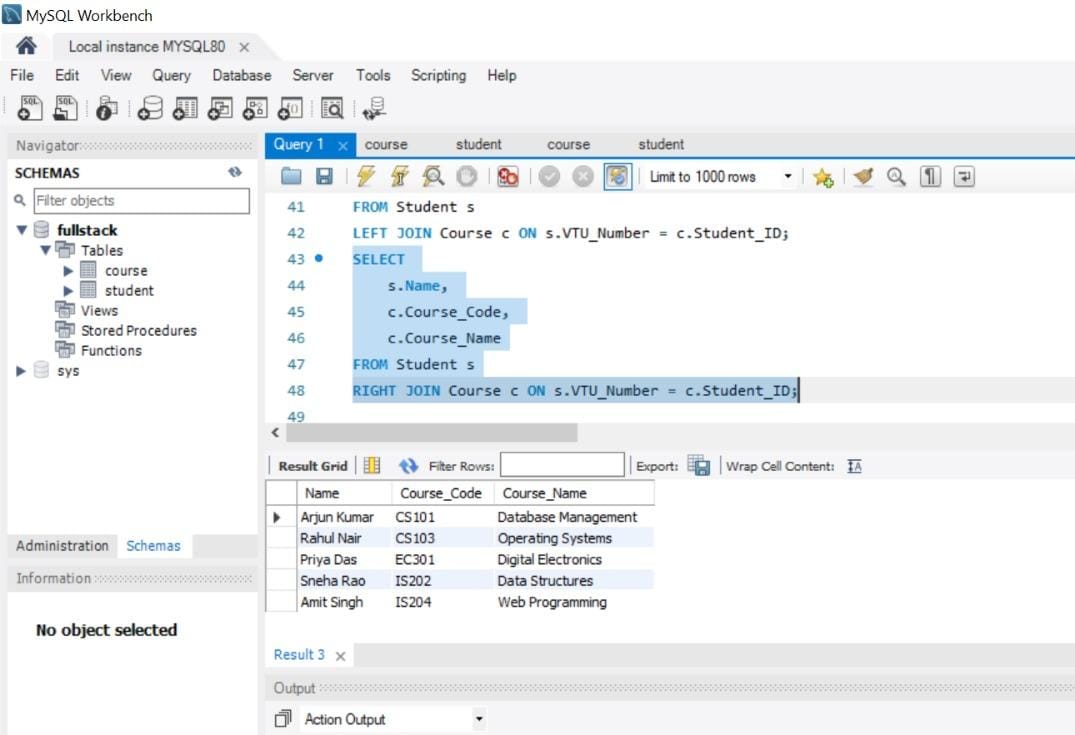
**s.Name,**

**c.Course\_Code,**

**c.Course\_Name**

**FROM Student s**

**RIGHT JOIN Course c ON s.VTU\_Number = c.Student\_ID;**



**FULL JOIN (Full Outer Join):**

**SELECT s.Name, c.Course\_Name FROM Student s**

**LEFT JOIN Course c ON s.VTU\_Number = c.Student\_ID**

**UNION**

**SELECT s.Name, c.Course\_Name FROM Student s**

**RIGHT JOIN Course c ON s.VTU\_Number = c.Student\_ID;**



**Cross Join(Cartesian Product):**

**SELECT s.Name, c.Course\_Name**

**FROM Student s**

**CROSS JOIN Course c;**

