

## SESSION 1 – GIT COMMANDS

Repository Name: University-Lab-Work

GitHub URL: <https://github.com/pprince/University-Lab-Work.git>

### 1. Practice basic Git commands

Initialize repository

```
git init
```

Check status

```
git status
```

Add file

```
git add README.md
```

Add all files

```
git add .
```

Commit changes

```
git commit -m "Initial project setup"
```

Check commit history

```
git log
```

Create branch

```
git branch feature-update
```

Switch branch

```
git checkout feature-update
```

Pull latest changes

```
git pull origin main
```

Push changes

```
git push origin main
```

### 2. Upload single document to GitHub using bash

File name: Lab1\_GitCommands.txt

```
git init
```

```
git add Lab1_GitCommands.txt
```

```
git commit -m "Added Lab1 Git Commands document"
```

```
git branch -M main
```

```
git remote add origin https://github.com/pprince/University-Lab-Work.git
```

```
git push -u origin main
```

### 3. Upload and update a folder in GitHub

Folder name: SQL\_Practice

```
git add SQL_Practice/
```

```
git commit -m "Added SQL Practice folder with session files"
```

```
git push origin main
```

After making changes inside folder:

```
git add SQL_Practice/  
git commit -m "Updated SQL Practice exercises"  
git push origin main
```

4. Create branch and push/commit one folder in Eclipse

Branch name: eclipse-feature

```
git branch eclipse-feature  
git checkout eclipse-feature
```

Folder name: Eclipse\_SQL\_Project

```
git add Eclipse_SQL_Project/  
git commit -m "Added Eclipse SQL project folder"  
git push origin eclipse-feature
```

5. Create branch and push/commit one folder in VS Code

Branch name: vscode-feature

```
git branch vscode-feature  
git checkout vscode-feature
```

Folder name: VSCode\_Database\_Project

```
git add VSCode_Database_Project/  
git commit -m "Added VS Code database project folder"  
git push origin vscode-feature
```

---

## SESSION 2 – SQL

1. Create database

```
CREATE DATABASE university_management;  
USE university_management;
```

2. Create tables

```
CREATE TABLE Student (  
vtu_number VARCHAR(15) PRIMARY KEY,  
name VARCHAR(100),  
email VARCHAR(100),  
phone VARCHAR(15),  
department VARCHAR(50)  
);
```

```
CREATE TABLE Course (  
course_code VARCHAR(10) PRIMARY KEY,  
course_name VARCHAR(100),  
faculty_id INT,  
student_id VARCHAR(15),
```

```
faculty_email VARCHAR(100)
);
```

### 3. Insert minimum 5 records

```
INSERT INTO Student VALUES
```

```
('1RV21CS001','Rahul Sharma','rahul.sharma@gmail.com','9876543210','CSE'),
('1RV21EC002','Anjali Rao','anjali.rao@gmail.com','9876543211','ECE'),
('1RV21CS003','Vikram Singh','vikram.singh@gmail.com','9876543212','CSE'),
('1RV21ME004','Sneha Patel','sneha.patel@gmail.com','9876543213','ME'),
('1RV21CS005','Amit Kumar','amit.kumar@gmail.com','9876543214','CSE');
```

```
INSERT INTO Course VALUES
```

```
('CS101','Database Management System',101,'1RV21CS001','faculty.dbms@rvce.edu.in'),
('CS102','Operating Systems',102,'1RV21EC002','faculty.os@rvce.edu.in'),
('CS103','Computer Networks',103,'1RV21CS003','faculty.cn@rvce.edu.in'),
('ME201','Thermodynamics',104,'1RV21ME004','faculty.thermo@rvce.edu.in'),
('CS104','Artificial Intelligence',105,'1RV21CS005','faculty.ai@rvce.edu.in');
```

### 4. Select records using different clauses

```
SELECT * FROM Student;
SELECT name, department FROM Student;
SELECT * FROM Student WHERE department='CSE';
SELECT * FROM Student ORDER BY name ASC;
SELECT * FROM Student WHERE department='CSE' AND name LIKE 'A%';
```

---

## SESSION 3 – SQL

### 1. Display all records

```
SELECT * FROM Student;
SELECT * FROM Course;
```

### 2. Aggregate functions

```
SELECT COUNT() AS total_students FROM Student;
SELECT department, COUNT() AS dept_count FROM Student GROUP BY department;
```

### 3. Sort by VTU number

```
SELECT * FROM Student ORDER BY vtu_number ASC;
SELECT * FROM Student ORDER BY vtu_number DESC;
```

### 4. Display students from CSE department

```
SELECT * FROM Student WHERE department='CSE';
```

### 5. Join VTU number with course and faculty

```
SELECT s.vtu_number, s.name, c.course_name, c.faculty_email
FROM Student s
JOIN Course c ON s.vtu_number = c.student_id;
```

---

## SESSION 4 – SQL

1. Count students per course

```
SELECT course_name, COUNT(student_id) AS total_students
FROM Course
GROUP BY course_name;
```

2. Insert faculty courses into new table

```
CREATE TABLE Faculty_Course_Assignments AS
SELECT faculty_id, course_name
FROM Course;
```

3. Select top 5 students from CSE and update phone with country code

```
SELECT * FROM Student
WHERE department='CSE'
LIMIT 5;
```

```
UPDATE Student
SET phone = CONCAT('+91', phone)
WHERE department='CSE'
LIMIT 5;
```

---

## SESSION 5 – SQL

1. Initiate and commit transaction

```
START TRANSACTION;
INSERT INTO Student VALUES ('1RV21CS006','Ravi
Verma','ravi.verma@gmail.com','9876543220','CSE');
COMMIT;
```

2. Savepoint and rollback

```
START TRANSACTION;
SAVEPOINT before_update;
UPDATE Student SET phone='9999999999' WHERE vtu_number='1RV21CS001';
ROLLBACK TO before_update;
COMMIT;
```

3. User-defined function for mathematical calculation

```
DELIMITER $$

CREATE FUNCTION multiply_numbers(a INT, b INT)
RETURNS INT
DETERMINISTIC
BEGIN
```

```
RETURN a * b;  
END$$
```

```
DELIMITER ;
```

4. User-defined function returning student count by department

```
DELIMITER $$
```

```
CREATE FUNCTION get_department_student_count(dept VARCHAR(50))  
RETURNS INT  
DETERMINISTIC  
BEGIN  
DECLARE total INT;  
SELECT COUNT(*) INTO total  
FROM Student  
WHERE department = dept;  
RETURN total;  
END$$
```

```
DELIMITER ;
```