CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;  
USE `mydb` ;

-- -----------------------------------------------------  
-- Table `mydb`.`Branch`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Branch` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Branch` (  
`Branch\_ID` INT NOT NULL,  
`B\_City` VARCHAR(45) NOT NULL,  
`B\_Country` VARCHAR(45) NOT NULL,  
`B\_Postcode` VARCHAR(45) NOT NULL,  
PRIMARY KEY (`Branch\_ID`))  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Employees`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Employees` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Employees` (  
`Employee\_ID` INT NOT NULL,  
`Employee\_Name` VARCHAR(45) NOT NULL,  
`Designation` VARCHAR(45) NOT NULL,  
`Branch\_info\_Branch\_ID` INT NOT NULL,  
PRIMARY KEY (`Employee\_ID`),  
INDEX `fk\_Employees\_Branch\_info\_idx` (`Branch\_info\_Branch\_ID` ASC) VISIBLE,  
CONSTRAINT `fk\_Employees\_Branch\_info`  
FOREIGN KEY (`Branch\_info\_Branch\_ID`)  
REFERENCES `mydb`.`Branch` (`Branch\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Customers`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Customers` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Customers` (  
`Customer\_ID` INT NOT NULL,  
`Customer\_Name` VARCHAR(45) NOT NULL,  
`Customer\_Postcode` VARCHAR(45) NOT NULL,  
`Email` VARCHAR(45) NOT NULL,  
`Phone\_Number` VARCHAR(45) NOT NULL,  
`Qualification` VARCHAR(45) NOT NULL,  
`Password` VARCHAR(45) NOT NULL,  
PRIMARY KEY (`Customer\_ID`))  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Programs Offered`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Programs Offered` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Programs Offered` (  
`Program\_ID` INT NOT NULL,  
`Program\_Name` VARCHAR(45) NOT NULL,  
`Description` VARCHAR(45) NULL,  
PRIMARY KEY (`Program\_ID`))  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Courses`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Courses` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Courses` (  
`Course\_ID` INT NOT NULL,  
`Course Name` VARCHAR(45) NOT NULL,  
`Fees` INT NOT NULL,  
`Course Duration` TIME NOT NULL,  
`Course Level` VARCHAR(45) NULL,  
`Prerequisites` VARCHAR(45) NULL,  
`Programs Offered\_Program\_ID` INT NOT NULL,  
PRIMARY KEY (`Course\_ID`, `Programs Offered\_Program\_ID`),  
INDEX `fk\_Courses\_Programs Offered1\_idx` (`Programs Offered\_Program\_ID` ASC) VISIBLE,  
CONSTRAINT `fk\_Courses\_Programs Offered1`  
FOREIGN KEY (`Programs Offered\_Program\_ID`)  
REFERENCES `mydb`.`Programs Offered` (`Program\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Customers\_has\_Courses`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Customers\_has\_Courses` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Customers\_has\_Courses` (  
`Customers\_Customer\_ID` INT NOT NULL,  
`Courses\_Course\_ID` INT NOT NULL,  
PRIMARY KEY (`Customers\_Customer\_ID`, `Courses\_Course\_ID`),  
INDEX `fk\_Customers\_has\_Courses\_Courses1\_idx` (`Courses\_Course\_ID` ASC) VISIBLE,  
INDEX `fk\_Customers\_has\_Courses\_Customers1\_idx` (`Customers\_Customer\_ID` ASC) VISIBLE,  
CONSTRAINT `fk\_Customers\_has\_Courses\_Customers1`  
FOREIGN KEY (`Customers\_Customer\_ID`)  
REFERENCES `mydb`.`Customers` (`Customer\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT `fk\_Customers\_has\_Courses\_Courses1`  
FOREIGN KEY (`Courses\_Course\_ID`)  
REFERENCES `mydb`.`Courses` (`Course\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Enrollment`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Enrollment` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Enrollment` (  
`Enrollment\_ID` INT NOT NULL,  
`Enrollment\_Date` DATE NOT NULL,  
`Employees\_Employee\_ID` INT NOT NULL,  
`Customers\_Customer\_ID` INT NOT NULL,  
`Customers\_has\_Courses\_Customers\_Customer\_ID` INT NOT NULL,  
`Customers\_has\_Courses\_Courses\_Course\_ID` INT NOT NULL,  
PRIMARY KEY (`Enrollment\_ID`),  
INDEX `fk\_Enrollment\_Employees1\_idx` (`Employees\_Employee\_ID` ASC) VISIBLE,  
INDEX `fk\_Enrollment\_Customers1\_idx` (`Customers\_Customer\_ID` ASC) VISIBLE,  
INDEX `fk\_Enrollment\_Customers\_has\_Courses1\_idx` (`Customers\_has\_Courses\_Customers\_Customer\_ID` ASC, `Customers\_has\_Courses\_Courses\_Course\_ID` ASC) VISIBLE,  
CONSTRAINT `fk\_Enrollment\_Employees1`  
FOREIGN KEY (`Employees\_Employee\_ID`)  
REFERENCES `mydb`.`Employees` (`Employee\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT `fk\_Enrollment\_Customers1`  
FOREIGN KEY (`Customers\_Customer\_ID`)  
REFERENCES `mydb`.`Customers` (`Customer\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT `fk\_Enrollment\_Customers\_has\_Courses1`  
FOREIGN KEY (`Customers\_has\_Courses\_Customers\_Customer\_ID` , `Customers\_has\_Courses\_Courses\_Course\_ID`)  
REFERENCES `mydb`.`Customers\_has\_Courses` (`Customers\_Customer\_ID` , `Courses\_Course\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Certification`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Certification` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Certification` (  
`Certification\_ID` INT NOT NULL,  
`Date of Issue` DATE NOT NULL,  
`Issued by` VARCHAR(45) NOT NULL,  
PRIMARY KEY (`Certification\_ID`))  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Evaluation`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Evaluation` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Evaluation` (  
`Evaluation\_ID` INT NOT NULL,  
`Assessment\_type` VARCHAR(45) NOT NULL,  
`Completion\_Time\_Hours` TIME NOT NULL,  
`Assessment\_Status` VARCHAR(45) NOT NULL,  
`Passing\_Percentage` FLOAT NOT NULL,  
`Date\_of\_Completion` DATE NOT NULL,  
`Enrollment\_Enrollment\_ID` INT NOT NULL,  
`Courses\_Course\_ID` INT NOT NULL,  
`Certification\_Certification\_ID` INT NOT NULL,  
PRIMARY KEY (`Evaluation\_ID`, `Enrollment\_Enrollment\_ID`, `Certification\_Certification\_ID`),  
INDEX `fk\_Evaluation\_Enrollment1\_idx` (`Enrollment\_Enrollment\_ID` ASC) VISIBLE,  
INDEX `fk\_Evaluation\_Courses1\_idx` (`Courses\_Course\_ID` ASC) VISIBLE,  
INDEX `fk\_Evaluation\_Certification1\_idx` (`Certification\_Certification\_ID` ASC) VISIBLE,  
CONSTRAINT `fk\_Evaluation\_Enrollment1`  
FOREIGN KEY (`Enrollment\_Enrollment\_ID`)  
REFERENCES `mydb`.`Enrollment` (`Enrollment\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT `fk\_Evaluation\_Courses1`  
FOREIGN KEY (`Courses\_Course\_ID`)  
REFERENCES `mydb`.`Courses` (`Course\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT `fk\_Evaluation\_Certification1`  
FOREIGN KEY (`Certification\_Certification\_ID`)  
REFERENCES `mydb`.`Certification` (`Certification\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Universities`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Universities` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Universities` (  
`University\_ID` INT NOT NULL,  
`University\_Name` VARCHAR(45) NOT NULL,  
`U\_City` VARCHAR(45) NOT NULL,  
`U\_Country` VARCHAR(45) NOT NULL,  
PRIMARY KEY (`University\_ID`))  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Trainers`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Trainers` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Trainers` (  
`Trainer\_ID` INT NOT NULL,  
`Trainer\_Name` VARCHAR(45) NOT NULL,  
`Experience\_in\_Yeras` INT NULL,  
`Universities\_University\_ID` INT NOT NULL,  
PRIMARY KEY (`Trainer\_ID`),  
INDEX `fk\_Trainers\_Universities1\_idx` (`Universities\_University\_ID` ASC) VISIBLE,  
CONSTRAINT `fk\_Trainers\_Universities1`  
FOREIGN KEY (`Universities\_University\_ID`)  
REFERENCES `mydb`.`Universities` (`University\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Transactions`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Transactions` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Transactions` (  
`Transaction\_ID` INT NOT NULL,  
`Account Number` VARCHAR(45) NOT NULL,  
`Transaction Date` DATE NULL,  
`Customers\_Customer\_ID` INT NOT NULL,  
`Courses\_Course\_ID` INT NOT NULL,  
PRIMARY KEY (`Customers\_Customer\_ID`),  
INDEX `fk\_Transactions\_Courses1\_idx` (`Courses\_Course\_ID` ASC) VISIBLE,  
CONSTRAINT `fk\_Transactions\_Customers1`  
FOREIGN KEY (`Customers\_Customer\_ID`)  
REFERENCES `mydb`.`Customers` (`Customer\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT `fk\_Transactions\_Courses1`  
FOREIGN KEY (`Courses\_Course\_ID`)  
REFERENCES `mydb`.`Courses` (`Course\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;

-- -----------------------------------------------------  
-- Table `mydb`.`Courses\_has\_Trainers1`  
-- -----------------------------------------------------  
DROP TABLE IF EXISTS `mydb`.`Courses\_has\_Trainers1` ;

CREATE TABLE IF NOT EXISTS `mydb`.`Courses\_has\_Trainers1` (  
`Courses\_Course\_ID` INT NOT NULL,  
`Trainers\_Trainer\_ID` INT NOT NULL,  
PRIMARY KEY (`Courses\_Course\_ID`, `Trainers\_Trainer\_ID`),  
INDEX `fk\_Courses\_has\_Trainers1\_Trainers1\_idx` (`Trainers\_Trainer\_ID` ASC) VISIBLE,  
INDEX `fk\_Courses\_has\_Trainers1\_Courses1\_idx` (`Courses\_Course\_ID` ASC) VISIBLE,  
CONSTRAINT `fk\_Courses\_has\_Trainers1\_Courses1`  
FOREIGN KEY (`Courses\_Course\_ID`)  
REFERENCES `mydb`.`Courses` (`Course\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT `fk\_Courses\_has\_Trainers1\_Trainers1`  
FOREIGN KEY (`Trainers\_Trainer\_ID`)  
REFERENCES `mydb`.`Trainers` (`Trainer\_ID`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;  
SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;  
SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

DELIMITER $  
  
CREATE PROCEDURE Sign\_Up (   
 IN Customer\_First\_Name varchar(45), Customer\_Last\_Name varchar(45),   
 Customer\_Postcode varchar(45), Email varchar(45), Phone\_Number Varchar(45),   
 Qualification varchar(45), Customer\_Password varchar(45), Customer\_city varchar(45)  
)  
BEGIN  
 INSERT INTO Customers(Customer\_First\_Name, Customer\_Last\_Name, Customer\_Postcode, Email, Phone\_Number, Qualification, Customer\_Password, Customer\_city)  
 VALUES (Customer\_First\_Name, Customer\_Last\_Name, Customer\_Postcode, Email, Phone\_Number, Qualification, Customer\_Password, Customer\_city);  
END$   
DELIMITER ;  
  
CALL Sign\_Up("Jaymin", "Jack", "DD1JS", "MartinJaymin@gmail.com", "076458633", "Graduate", "8hfdffr", "Dundee");  
  
DELIMITER $  
  
CREATE PROCEDURE AboutUpdateProfile (   
 IN C\_Postcode varchar(45), IN C\_City varchar(45), IN C\_Password varchar(45),   
 IN C\_Email varchar(45), IN C\_Phone\_Number varchar(45), IN C\_Qualification varchar(45)  
)  
BEGIN  
 UPDATE Customers SET Customer\_Postcode = C\_Postcode, Customer\_city = C\_City, Phone\_Number = C\_Phone\_Number, Qualification = C\_Qualification, Customer\_Password = C\_Password  
 WHERE Email = C\_Email;  
END$   
DELIMITER ;  
  
CALL AboutUpdateProfile("EH1 1BE", "Edinburgh", "ghr74gd", "MartinJaymin@gmail.com", "076458633", "Graduate");  
  
DELIMITER $  
  
CREATE PROCEDURE OurCourses()  
BEGIN  
 SELECT InCourse.Course\_name, InCourse.Fees, InCourse.Course\_Duration, InCourse.Course\_Level, InCourse.Prerequisites, InPrograms.Program\_Name  
 FROM Courses AS InCourse  
 LEFT JOIN Programs\_Offered AS InPrograms ON InCourse.Programs\_Offered\_Program\_ID = InPrograms.Program\_ID;  
END$  
DELIMITER ;  
  
CALL OurCourses();  
  
DELIMITER $  
  
CREATE PROCEDURE selectcourse(IN corse\_name varchar(45))  
BEGIN  
 SELECT Co.Course\_ID, Co.Course\_Name, Co.Fees, P.Program\_Name, U.University\_Name, T.Trainer\_First\_Name, T.Trainer\_Last\_Name  
 FROM Courses AS Co  
 INNER JOIN Programs\_Offered AS P ON Co.Programs\_Offered\_Program\_ID = P.Program\_ID  
 INNER JOIN Courses\_has\_Trainers1 ON Co.Course\_ID = Courses\_Course\_ID  
 INNER JOIN Trainers AS T ON Trainer\_ID = Trainers\_Trainer\_ID  
 INNER JOIN Universities AS U ON University\_ID = Universities\_University\_ID  
 WHERE Co.Course\_Name = corse\_name;  
END$  
DELIMITER ;  
  
CALL SearchCourses('Marketing Analytics');  
  
DELIMITER $  
  
CREATE PROCEDURE BuyNow(  
 IN Account\_Number varchar(45), IN Transaction\_Date date, IN Amount\_paid varchar(45), IN Customer\_ID INT  
)  
BEGIN  
 INSERT INTO Transactions (Account\_Number, Transaction\_Date, Amount\_paid, Customers\_Customer\_ID)  
 VALUES (Account\_Number, Transaction\_Date, Amount\_paid, Customer\_ID);  
END$  
DELIMITER ;  
  
CALL BuyNow('56432', '2023-8-11', '200', '300145');  
  
DELIMITER $  
  
CREATE PROCEDURE checkpayment(IN Customer\_ID INT)  
BEGIN  
 SELECT C.Customer\_ID, C.Customer\_First\_Name, C.Customer\_Last\_Name, T.Amount\_paid  
 FROM Customers AS C  
 INNER JOIN Transactions AS T ON C.Customer\_ID = T.Customers\_Customer\_ID  
 WHERE C.Customer\_ID = Customer\_ID;  
END$  
DELIMITER ;  
  
CALL checkpayment('300145');  
  
CREATE PROCEDURE enrollstudent(  
 IN EnrollmentDate DATE, IN CustomerID INT, IN BranchID INT  
)  
BEGIN  
 INSERT INTO Enrollment (Enrollment\_Date, Customers\_Customer\_ID, Branch\_Branch\_ID)  
 VALUES (EnrollmentDate, CustomerID, BranchID);  
  
 IF ROW\_COUNT() > 0 THEN  
 SELECT 'Enrollment inserted successfully' AS Status;  
 ELSE  
 SELECT 'Failed to insert enrollment' AS Status;  
 END IF;  
END//  
  
DELIMITER ;  
CALL enrollstudent('2023-11-12', '300145', '7');  
  
DELIMITER $  
  
CREATE PROCEDURE Refund(IN Customer\_ID INT)  
BEGIN  
 DELETE FROM Transactions WHERE Customers\_Customer\_ID = Customer\_ID;  
END$  
DELIMITER ;  
  
CALL Refund('300145');  
  
DELIMITER //  
  
CREATE PROCEDURE ManagerInsertEmployee(  
 IN FirstName VARCHAR(50), IN LastName VARCHAR(50), IN Designation VARCHAR(50), IN BranchID INT  
)  
BEGIN  
 INSERT INTO Employees (Employee\_First\_Name, Employee\_Last\_Name, Designation, Branch\_info\_Branch\_ID)  
 VALUES (FirstName, LastName, Designation, BranchID);  
  
 SELECT 'Employee inserted successfully' AS Status;  
END//  
DELIMITER ;  
  
CALL ManagerInsertEmployee('Leo', 'hao', 'HR', '7');  
  
CREATE PROCEDURE ManagerDeleteEmployee(  
 IN Employee\_First\_Name varchar(45)  
)  
BEGIN  
 DELETE FROM Employees WHERE Employee\_First\_Name = Employee\_First\_Name;  
  
 SELECT 'Employee deleted successfully' AS Status;  
END//  
  
DELIMITER ;  
CALL ManagerDeleteEmployee('Lao');  
  
DELIMITER $$  
  
CREATE PROCEDURE InsertTrainer(  
 IN TrainerFirstName VARCHAR(45), IN TrainerLastName VARCHAR(45), IN ExperienceinYears INT, IN UniversitiesUniversityID INT  
)  
BEGIN  
 INSERT INTO Trainers (Trainer\_First\_Name, Trainer\_last\_Name, Experience\_in\_Years, Universities\_University\_ID)  
 VALUES (TrainerFirstName, TrainerLastName, ExperienceinYears, UniversitiesUniversityID);  
END $$  
  
DELIMITER $$  
  
CALL InsertTrainer('Madeeha', 'Khan', 4, 25301);  
  
DELIMITER $$  
  
CREATE PROCEDURE ViewCourses(IN Trainer\_ID INT)  
BEGIN  
 SELECT cour.Course\_Name, cour.Course\_Duration, cour.Course\_Level, cour.Prerequisites, t.Trainer\_First\_Name  
 FROM Courses\_has\_Trainers1 ct  
 INNER JOIN Courses cour ON ct.Courses\_Course\_ID = cour.Course\_ID  
 INNER JOIN Trainers t ON t.Trainer\_ID = ct.Trainers\_Trainer\_ID  
 WHERE t.Trainer\_ID = Trainer\_ID;  
END $$  
  
DELIMITER ;  
  
CALL ViewCourses(56715);  
  
DELIMITER $$  
  
CREATE PROCEDURE updateEvaluation(  
 IN EvaluationID INT, IN AssessmentType VARCHAR(45), IN CompletionTimeHours TIME, IN AssessmentStatus VARCHAR(45),  
 IN PassingPercentage FLOAT, IN DateofCompletion DATE  
)  
BEGIN  
 UPDATE Evaluation  
 SET Assessment\_Type = AssessmentType, Completion\_Time\_Hours = CompletionTimeHours, Assessment\_Status = AssessmentStatus,  
 Passing\_Percentage = PassingPercentage, Date\_of\_Completion = DateofCompletion  
 WHERE Evaluation\_ID = EvaluationID;  
END $$  
  
DELIMITER ;  
  
CALL updateEvaluation(300115, 'Group Assignment', '12:00:00', 'PASS', 50, '2023-11-30');  
  
DELIMITER $$  
  
CREATE PROCEDURE ViewEvaluationMetrics(IN trainerID INT)  
BEGIN  
 SELECT COUNT(\*) AS evaluations,  
 ((SELECT COUNT(\*) FROM Evaluation WHERE Assessment\_Status = 'PASS') / COUNT(\*) \* 100) AS Total\_Pass,  
 ((SELECT COUNT(\*) FROM Evaluation WHERE Assessment\_Status = 'FAIL') / COUNT(\*) \* 100) AS Total\_Fail,  
 ((SELECT COUNT(\*) FROM Evaluation WHERE Assessment\_Status = 'PENDING') / COUNT(\*) \* 100) AS Total\_Pending  
 FROM Evaluation;  
END $$  
  
DELIMITER ;  
  
CALL ViewEvaluationMetrics(56701);  
  
DELIMITER //  
  
CREATE PROCEDURE ViewEmployees()  
BEGIN  
 SELECT e.Branch\_info\_Branch\_ID AS Branch\_ID, e.Designation, COUNT(\*) AS Employee\_Count  
 FROM Employees e  
 GROUP BY e.Branch\_info\_Branch\_ID;  
END//  
  
CALL ViewEmployees;  
  
DELIMITER //  
  
CREATE PROCEDURE TotalRevenue()  
BEGIN  
 SELECT DATE\_FORMAT(Transaction\_Date, '%Y-%m') AS Month, SUM(Amount\_Paid) AS Total\_Amount\_Paid  
 FROM Transactions  
 GROUP BY DATE\_FORMAT(Transaction\_Date, '%Y-%m');  
END//  
  
DELIMITER ;  
  
CALL TotalRevenue;  
  
DELIMITER //  
  
CREATE PROCEDURE ViewTotalCustomersByCourse()  
BEGIN  
 CREATE TEMPORARY TABLE IF NOT EXISTS temp\_results (Courses VARCHAR(255), Customer INT);  
  
 INSERT INTO temp\_results (Courses, Customer)  
 SELECT c.Course\_Name, COUNT(chc.Customers\_Customer\_ID) AS Total\_Customers  
 FROM Customers\_has\_Courses chc  
 JOIN Courses c ON chc.Courses\_Course\_ID = c.Course\_ID  
 GROUP BY c.Course\_Name;  
  
 SELECT \* FROM temp\_results ORDER BY Customer DESC;  
  
 DROP TEMPORARY TABLE IF EXISTS temp\_results;  
END//  
  
DELIMITER ;  
  
CALL ViewTotalCustomersByCourse;  
  
DELIMITER //  
  
CREATE PROCEDURE UpdateBranchInformation(  
 IN p\_Branch\_ID INT, IN p\_B\_City VARCHAR(255), IN p\_B\_Country VARCHAR(255), IN p\_B\_Postcode VARCHAR(255)  
)  
BEGIN  
 UPDATE Branch  
 SET B\_City = p\_B\_City, B\_Country = p\_B\_Country, B\_Postcode = p\_B\_Postcode  
 WHERE Branch\_ID = p\_Branch\_ID;  
  
 SELECT 'Branch information updated successfully.' AS 'Result';  
END //  
  
DELIMITER ;  
  
CALL UpdateBranchInformation('1', 'Glasgow', 'UK', 'DD1');