MySQL Labs

**MySQL (Day3):**

**insert into students\_courses**

**values**

**(1,4,60,NULL),**

**(2,1,NULL,NULL),**

**(2,4,75,NULL),**

**(3,1,NULL,NULL),**

**(3,2,NULL,NULL),**

**(3,3,75,NULL);**

|  |  |
| --- | --- |
| *1* | ***Create function to calculate the number of students who get grade less than 80 in a certain exam (course id will be sent as a parameter)*** |
|  | DELIMITER $  CREATE FUNCTION calc\_no\_students(ID INT)  -> RETURNS INT  -> DETERMINISTIC  -> BEGIN  -> DECLARE count INT DEFAULT 0;  -> SELECT COUNT(\*) INTO count  -> FROM students\_courses  -> WHERE course\_id = ID  -> AND grade < 80;  -> RETURN count;  -> END$  DELIMITER ;  SELECT calc\_no\_students(4); |
| *2* | ***Create stored procedure to display the names of the absence students of a certain courses.(Absent means has no grades)*** |
|  | *DELIMITER $*  *CREATE PROCEDURE names\_of\_abs()*  *-> BEGIN*  *-> SELECT CONCAT(first\_name, ' ', last\_name) AS Name, grade*  *-> FROM students JOIN students\_courses*  *-> ON students.student\_id = students\_courses.student\_id*  *-> WHERE grade IS NULL;*  *-> END$*  *DELIMITER ;*  *CALL names\_of\_abs()*    *SELECT \* FROM students join students\_courses on students.student\_id = students\_courses.student\_id;* |
| *3* | ***Create stored procedure to calculate the average grades for certain course.*** |
|  | *DELIMITER $*  *CREATE PROCEDURE calc\_avg\_grades(c\_id INT)*  *-> BEGIN*  *-> SELECT*  *-> AVG(grade) AS avg\_grade*  *-> FROM students\_courses*  *-> WHERE course\_id = c\_id;*  *-> END$*  *DELIMITER ;* |
| *4* | ***Create trigger to keep track the changes(updates) of the grades in the studnets\_courses table***  ***( create changes table with the following fields:***  ***id int primary key ,***  ***user varchar(30),***  ***action varchar(40),***  ***old\_grade int,***  ***new\_grade int,***  ***change\_date date).***  ***Test the trigger by updating grade int the “Students\_courses” table***  ***Confirm that the row is added in the” change\_table”*** |
|  | **CREATE TABLE changes (**  **-> id INT AUTO INCREMENT PRIMARY KEY,**  **-> user VARCHAR(30),**  **-> action VARCHAR(40),**  **-> old\_grade INT,**  **-> new\_grade INT,**  **-> change\_date date);**    **CREATE TRIGGER grade\_update\_trigger**  **-> AFTER UPDATE ON students\_courses**  **-> FOR EACH ROW**  **-> BEGIN**  **-> IF OLD.grade <> NEW.grade THEN**  **-> INSERT INTO changes**  **-> VALUES (NULL, USER(), 'Grade Update', OLD.grade, NEW.grade, CURDATE());**  **-> END IF;**  **-> END$** |
| *5* | ***Create event to delete the changes tables every 5 minute*** |
|  | CREATE EVENT delete\_changes  -> ON SCHEDULE EVERY 5 MINUTE  -> DO  -> DELETE FROM changes; |