**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)*** |
|  | ***drop function if exists num\_of\_students;***  ***delimiter $***  ***CREATE FUNCTION num\_of\_students(p\_id integer)***  ***RETURNS int(11)***  ***BEGIN***  ***Declare v\_count integer;***  ***SET v\_count =***  ***(select count(student\_id) from students\_courses where course\_id = p\_id and grade < 80) ;***  ***RETURN v\_count;***  ***END$***  ***delimiter ;*** |
| ***2*** | ***Create stored procedure to display the names of the absence students of a certain courses.(Absent means has no grades)*** |
|  | |  |  |  | | --- | --- | --- | | ***drop procedure if exists display\_absents;***  ***delimiter $***  ***CREATE PROCEDURE display\_absents ()***  ***BEGIN***   |  | | --- | | **Select CONCAT(s.first\_name, “ “, s.last\_name) as Name, c.course\_name, c.course\_id, IFNULL(e.grade, “Absent”)**  **From (courses c, students s) left join (students\_courses e)**  **ON e.course\_id = c.course\_id**  **AND e.student\_id = s.student\_id**  **AND e.grade = “Absent”;** |   ***END$***  ***delimiter ;***    ***Or with parameter:***  ***drop procedure if exists display\_absents2;***  ***delimiter $***  ***CREATE PROCEDURE display\_absents2 (p\_id integer)***  ***BEGIN***   |  | | --- | | **Select CONCAT(s.first\_name, “ “, s.last\_name) as Name, c.course\_name, c.course\_id, IFNULL(e.grade, “Absent”)**  **From (courses c, students s) left join (students\_courses e)**  **ON e.course\_id = c.course\_id**  **AND c.course\_id = p\_id**  **AND e.student\_id = s.student\_id**  **AND e.grade = “Absent”;** |   ***END$***  ***delimiter ;*** | |
| ***3*** | ***Create stored procedure to calculate the average grades for certain course.*** |
|  | ***drop procedure if exists average\_grades;***  ***delimiter $***  ***CREATE PROCEDURE average\_grades (p\_id integer)***  ***BEGIN***   |  | | --- | | **Select c.course\_name, avg(e.grade)**  **From courses c, students\_courses e**  **WHERE e.course\_id = c.course\_id**  **AND e.course\_id = p\_id**  **Group by c.course\_name;** |   ***END$***  ***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 IF NOT EXISTS *changes*(**  **id INT(11) NOT NULL AUTO\_INCREMENT PRIMARY KEY,**  **user VARCHAR(30),**  **action VARCHAR(40),**  **old\_grade int,**  **new\_grade int,**  **change\_date date)**  **ENGINE = INNODB;**  **DELIMITER //**  **CREATE TRIGGER grades\_change**  **AFTER UPDATE ON students\_courses**  **FOR EACH ROW**  **BEGIN**  **IF !(NEW.grade <=> OLD.grade) THEN**    **INSERT INTO *changes* (user, action, old\_grade, new\_grade, change\_date )**  **VALUES (current\_user(), “Change grade”, OLD.grade, NEW.grade, Current\_date());**  **END IF;**  **END;//**  **DELIMITER ;**  **UPDATE *students\_courses***  **SET *grade = 85***  **WHERE student\_id = 1 AND course\_id = 1;**    **UPDATE *students\_courses***  **SET *grade = 95***  **WHERE student\_id = 1 AND course\_id = 2;** | |
| ***5*** | ***Create event to delete the changes tables every 5 minute*** |
|  | select @@global.event\_scheduler;  set @@global.event\_scheduler=1;  .  .  .  .  CREATE EVENT delete\_changes  ON SCHEDULE EVERY 5 MINUTE  DO  DELETE FROM changes; |