Queries

1. query to find people who are absent in each course on a *particular date* and display course_id,name,finger_id,course_name (FOR SENDING MSG TO ABSENTEES) NOTE: Change the date clause depending on the current system date, currently initialized a default value

```
SELECT * FROM (SELECT DISTINCT course_id,finger_id FROM (SELECT
course_id,finger_id FROM (SELECT course_id FROM (SELECT fiSELECT * FROM
(SELECT DISTINCT course_id,finger_id FROM (SELECT course_id,finger_id FROM
(SELECT course_id FROM (SELECT
finger_id,course_id,attendance,cast(timestamp as date) as date FROM
course_attendance) AS a WHERE date='2018-04-06’) AS b NATURAL JOIN
(SELECT * FROM course_taken)AS c ) AS x WHERE NOT EXISTS (SELECT
DISTINCT course_id,finger_id FROM (SELECT course_id,finger_id FROM (SELECT
finger_id,course_id,attendance,cast(timestamp as date) as date FROM
course_attendance) AS d WHERE date='2018-04-06’) AS e WHERE
x.course_id=course_id AND x.finger_id=finger_id)) AS y NATURAL JOIN (SELECT
name,s_id,finger_id FROM student) AS z NATURAL JOIN (SELECT
course_id,course_name FROM faculty) AS w
```

2.This shows that attendance of student out of total lecs for a course (FOR STUDENT PAGE) NOTE: accept username from app that will be the id for which this info has to be displayed. So, a where clause has to be added

```
SELECT * FROM (SELECT course_id,course_name,finger_id, total_attendance,lecs
FROM (SELECT course_id,finger_id,COUNT(attendance) as total_attendance
FROM course_attendance GROUP BY course_id,finger_id) AS a NATURAL JOIN
(SELECT * FROM faculty) AS b) AS k NATURAL JOIN (SELECT name,s_id,finger_id
FROM student) AS l
```

3.These are the students whose entry is not present for a particular course that means the attendance is 0 for that course out of the total lecs (FOR STUDENT PAGE)

```
SELECT* FROM (SELECT * FROM (SELECT course_id,finger_id FROM
course_taken AS x WHERE NOT EXISTS(SELECT
course_id,course_name,finger_id, total_attendance,lecs FROM (SELECT
course_id,finger_id,COUNT(attendance) as total_attendance FROM
course_attendance GROUP BY course_id,finger_id) AS a NATURAL JOIN (SELECT
* FROM faculty) AS b WHERE x.course_id=course_id and x.finger_id=finger_id))
AS y NATURAL JOIN (SELECT course_name,course_id,lecs FROM faculty) AS z) AS
k NATURAL JOIN (SELECT name,s_id,finger_id FROM student) AS l
```

4.Shows *date wise* attendance in the course to faculty
(FOR FACULTY page) NOTE: to be displayed depending on username of faculty fetched from app so a where clause needs o be added for that

```
SELECT * FROM (SELECT
course_id,course_name,f_id,name,total_attendance,students,date FROM (SELECT
course_id,cast(timestamp as date) as date,COUNT(attendance) as
total_attendance FROM course_attendance GROUP BY course_id,cast(timestamp
as date)) AS a NATURAL JOIN (SELECT * FROM faculty) AS b) AS k
```

5. Trigger - Now when attendance is taken for a course in faculty table the lecs should be incremented corresponding to that course id
(Note: has been already added, no need to write php for this)

```
CREATE DEFINER=`id5262205_riya`@`%` TRIGGER `updateLecs` BEFORE
INSERT ON `course_attendance` FOR EACH ROW BEGIN IF (cast(NEW.timestamp
as date) Not IN (SELECT DISTINCT cast(timestamp as date) FROM
course_attendance)) THEN UPDATE faculty SET lecs = lecs + 1 WHERE course_id
= NEW.course_id ; END IF; END
```