Assignment

Feb19/ DBT/ 011

Database Technologies

Diploma in Advance Computing

February 2019

**Sub-queries with joins.**

USE ***student\_phone, student\_address, faculty\_phone, faculty\_address, batch\_students, course\_batches, student\_qualifications, faculty\_qualifications, course\_modules, modules, faculty, student, course*** relation to solve the following queries.

|  |
| --- |
| 1. Display all student who have taken admission in more than 2 batches. |
| select student.\* from student, batch\_students where student.ID = batch\_students.studentID and batch\_students.studentID in (select studentID from batch\_students group by studentID having count(studentID) > 2); |
|  |
| 1. Display the student detail who have joined the same batch of the student ‘saleel’. |
| select student.\* from student, batch\_students where student.ID = batch\_students.studentid and batchID in (select batchID from student, batch\_students where student.ID = batch\_students.studentID and namefirst='saleel'); |
|  |
| 1. Display all courses where least number of student have taken the admission. |
| select course.\* from batch\_students, course\_batches, course where batch\_students.batchid = course\_batches.id and course\_batches.courseID = course.ID and batch\_students.studentid in (select studentID from batch\_students group by studentID having count(studentID ) = (select min(r1) from (select studentID, count(\*) R1 from batch\_students group by studentID) x)); |
|  |
| 1. Display course*\_batches* detailswhereno student has taken the admission. |
| select \* from course\_batches where not exists (select batchID from batch\_students where course\_batches.ID = batch\_students.batchID); |
|  |
| 1. Get all courses where no modules are defined in course\_modules table. |
| select \* from course where not exists (select id1 from (select distinct courseID id1 from modules, course\_modules where course\_modules.moduleID = modules.id) x where course.id = x.id1); |
|  |
| 1. Display course*\_batches* detailswherestudent has taken the admission. |
| select \* from course\_batches where exists (select batchID from batch\_students where course\_batches.ID = batch\_students.batchID) |
|  |
| 1. Display all students whose marks of ‘BE’ is more than ‘ULKA’. |
| select \* from student, student\_qualifications where student.ID = student\_qualifications.studentID and name='BE' and marks >(select marks from student, student\_qualifications where  student.ID = student\_qualifications.studentID and namefirst='ulka' and name='BE'); |
|  |
| 1. Display all students whose marks are more than ‘saleel’ marks in 10th std. |
| select \* from student, student\_qualifications where student.id = student\_qualifications.studentID and marks > (select marks from student, student\_qualifications where student.id = student\_qualifications.studentID and student.namefirst = 'saleel' and name=10) and student\_qualifications.name=10; |
|  |
| 1. Display students whose DOB is as same as ‘kaushal’ |
| select \* from student where DOB = (select DOB from student where namefirst = 'kaushal'); |
|  |
| 1. Display all employee details who have three or more phone numbers. |
| select \* from student where id = (select studentID from student\_phone group by studentid having count(\*)>3); |
|  |
| 1. Display marks for the studentID 1 and 7. (Note: the marks must be displayed side by side). |
| select (select marks from student\_qualifications where student\_qualifications.name='BE' and studentid = 1) as "StudentID 1", (select marks from student\_qualifications where student\_qualifications.name='BE' and studentid = 7) as "StudentID 7"; |
|  |
| 1. Display marks for the studentID 1 and 7 also fine out the difference of marks between them.   (Note: the marks and difference between the marks must be displayed side by side) |
| select (select marks from student\_qualifications where student\_qualifications.name='BE' and studentid = 1) as "StudentID 1", (select marks from student\_qualifications where student\_qualifications.name='BE' and studentid = 7) as "StudentID 7", abs((select marks from student\_qualifications where student\_qualifications.name='BE' and studentid = 1) - (select marks from student\_qualifications where student\_qualifications.name='BE' and studentid = 7)) as "Marks Difference"; |
|  |
| 1. Display all student who are not joined any of the batch. |
| select \* from student where not exists (select \* from batch\_students where student.ID = batch\_students.studentID); |
|  |
| 1. Display all course\_batches details who are starting on the same day as ‘Batch1’. |
| select \* from course\_batches where starton = (select starton from course\_batches where course\_batches.name = 'batch1'); |
|  |
| 1. Display all employees’ commission details for the currentjob who are getting commission. |
|  |
|  |
| 1. Display all employee details who have three or more emailid. |
|  |
|  |
| 1. Get lowest salary of employee working in current job? |
|  |
|  |
| 1. Get lowest salary of employee working in current job? |
|  |
|  |
| 1. Get average salary of employee working in current job? |
|  |
|  |
| 1. Get sum salary of employee working in current job? |
|  |
|  |
| 1. Get highest *commission* of employee for the current job. |
|  |
|  |
| 1. Get second highest *salary* of employee for the current job. |
|  |
|  |
| 1. Print *salary* of all employees in the given format 3000\*\*\*\*\* for the current job. |
|  |
|  |
| 1. Get the *(department name, and count)*, than how many employees are working in which department for the current job. |
|  |
|  |
| 1. Display the name of stream where maximum number of employees has taken admission in ‘BE’. |
|  |
|  |
| 1. Count how many employee are not getting commission for the current job. |
|  |
|  |
| 1. Count how many employee are getting commission for the current job. |
|  |
|  |
| 1. Display all commission details of those employees who are not getting commission for the current job. |
|  |
|  |