**Lab Programs – 4 AND 5 (On Aggregate Functions and Nested / Sub Queries)**

Title: Implement nested sub queries (Single Row Query / Multiple Row Queries)

While Creating Table Declare constraints where ever required.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **EMPLOYEE\_ID** | **FIRST\_NAME** | **LAST\_NAME** | **EMAIL** | **PHONE\_NUMBER** | **HIRE\_DATE** | **JOB\_ID** | **SALARY** | **COMMISSION\_PCT** | **MANAGER\_ID** | **DEPARTMENT\_ID** |
| 100 | Steven | King | SKING | 515.123.4567 | 1987-06-17 | AD\_VP | 24000 | 0 | 0 | 90 |
| 101 | Neena | Kochhar | NKOCHHAR | 515.123.4568 | 1987-06-18 | AD\_VP | 17000 | 0 | 100 | 90 |
| 107 | Diana | Lorentz | DLORENTZ | 590.423.5567 | 1987-06-24 | IT\_PROG | 4200 | 0 | 103 | 60 |
| 108 | Nancy | Greenberg | NGREENBE | 515.124.4569 | 1987-06-25 | FI\_MGR | 12000 | 0 | 101 | 100 |
| 114 | Den | Raphaely | DRAPHEAL | 515.127.4561 | 1987-07-01 | CLK | 11000 | 0 | 114 | 30 |

|  |  |  |  |
| --- | --- | --- | --- |
| **DEPARTMENT\_ID** | **DEPARTMENT\_NAME** | **MANAGER\_ID** | **LOCATION\_ID** |
| 90 | VP | 0 | 1000 |
| 100 | Finance | 101 | 1400 |
| 30 | Clerk | 114 | 1200 |
| 60 | IT | 103 | 1100 |
| 70 | Press | 110 | 1300 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LOCATION\_ID** | **STREET\_ADDRESS** | **POSTAL\_CODE** | **CITY** | **STATE\_PROVINCE** | **COUNTRY\_ID** |
| 1000 | Avenue | 989 | Roma |  | IT |
| 1100 | Rashti Road | 10934 | Venice |  | IT |
| 1200 | Shinju Rd | 1689 | Tokyo | Tokyo Prefecture | JP |
| 1300 | Kamiya Rd | 6823 | Hiroshima |  | JP |
| 1400 | Jabberwocky Rd | 26192 | Southlake | Texas | US |

1. Create Table Employees, using constraints on required columns (PK, NOT NULL, CHECK, FK) –

create database Employees;

Use Employees;

/\* LOCATION\*/

CREATE TABLE IF NOT EXISTS LOCATION (

LOCATION\_ID int NOT NULL PRIMARY KEY,

ADRESS varchar(20) DEFAULT NULL,

POSTAL\_CODE int NOT NULL,

CITY VARCHAR(20) NOT NULL, STATE\_PROVINCE varchar(20) DEFAULT NULL,

COUNTRY\_ID varchar(20) NOT NULL);

Desc LOCATION;

/\* Department\*/

CREATE TABLE IF NOT EXISTS Department (

DEPARTMENT\_ID int NOT NULL PRIMARY KEY,

DEPARTMENT\_NAME varchar(20) DEFAULT NULL,

MANAGER\_ID int UNIQUE NOT NULL ,

LOCATION\_ID int DEFAULT NULL,

foreign key (LOCATION\_ID) REFERENCES LOCATION (LOCATION\_ID));

Desc Department;

/\* employees\*/

CREATE TABLE employees (

EMPLOYEE\_ID decimal(6,0) NOT NULL PRIMARY KEY,

FIRST\_NAME varchar(20) DEFAULT NULL,

LAST\_NAME varchar(25) NOT NULL,

EMAIL varchar(25) NOT NULL,

PHONE\_NUMBER varchar(20) DEFAULT NULL,

HIRE\_DATE date NOT NULL,

JOB\_ID varchar(10) NOT NULL,

SALARY decimal(8,2) DEFAULT NULL,

COMMISSION\_PCT decimal(2,2) DEFAULT NULL,

MANAGER\_ID int DEFAULT NULL ,

DEPARTMENT\_ID int,

FOREIGN KEY(DEPARTMENT\_ID) REFERENCES Department (DEPARTMENT\_ID));

Desc employees;

1. Insert values as in Location, Department and Employees table respectively.

/\*Location \*/

insert into LOCATION values (1000,'Avenue','989','Roma','Italy','IT');

insert into LOCATION values (1100,'Rashti ','10934','Venice','Italy','IT');

insert into LOCATION values (1200,'Shinju ','1689','Tokyo','Tokyo','JP');

insert into LOCATION values (1400,'Jabberwocky ','26192','Southlake','Michigan','US');

INSERT INTO LOCATION VALUES (1300, 'Shinju Rd', '6823', 'Tokoy', 'Texus', 'JP');

/\*Department\*/

INSERT INTO Department VALUES(90,'VP',0,1000);

INSERT INTO Department VALUES(100,'Finance',101,1400);

INSERT INTO Department VALUES(30,'Clerk ',114,1200);

INSERT INTO Department VALUES(60,'IT',103,1100);

INSERT INTO Department VALUES(70,'Press',110,1300);

/\* Employees\*/

INSERT INTO employees VALUES(100,'Steven','King','SKING','456789','1987-06-17','AD\_VP',24000,0,'0',90);

INSERT INTO employees VALUES(101,'Neena','Kochhar','NKOCHHAR','23646','1987-06-18','AD\_VP',17000,0,100,90);

INSERT INTO employees VALUES(107,'Diana','Lorentz','DLORENTZ','15863','1987-06-24','IT\_PROG',4200,0,103,60);

INSERT INTO employees VALUES(108,'Nancy','Greenberg','NGREENBE','79546','1987-06-25','FI\_MGR',12000,0,101,100);

INSERT INTO employees VALUES(114,'Den','Raphaely','DRAPHEAL','79556','1988-06-25','CLK',11000,0,114,30);

1. Write a SQL query to find those employees who receive a higher salary than the employee with ID 114. Return first name, last name

Solution –

SELECT first\_name, last\_name

FROM employees

WHERE salary >

( SELECT salary

FROM employees

WHERE employee\_id=114

);

1. Write a SQL query to find those employees whose salary matches the lowest salary of any of the departments. Return first name, last name and department ID.

Solution –

SELECT first\_name, last\_name, salary, department\_id FROM employees

WHERE salary IN

( SELECT MIN(salary)

FROM employees

GROUP BY department\_id

);

1. write a SQL query to find those employees who earn more than the average salary. Return employee ID, first name, last name

Solution –

SELECT employee\_id, first\_name,last\_name

FROM employees

WHERE salary >

( SELECT AVG(salary)

FROM employees

);

1. write a SQL query to find all those employees who work in the IT department. Return department ID, name (first), job ID and department name.

Solution –

SELECT e.department\_id, e.first\_name, e.job\_id , d.department\_name

FROM employees e , department d

WHERE e.department\_id = d.department\_id

AND d.department\_name = 'IT';

1. write a SQL query to find those employees whose salary falls within the range of the smallest salary and 10000. Return all the fields.

SELECT \*

FROM employees

WHERE salary BETWEEN

(SELECT MIN(salary)

FROM employees) AND 10000;

1. write a SQL query to find those employees who do not work in the departments where managers’ IDs are between 100 and 110 (Begin and end values are included.). Return all the fields of the employees.

SELECT \*

FROM employees

WHERE department\_id NOT IN

(SELECT department\_id

FROM department

WHERE manager\_id BETWEEN 100 AND 110);

1. write a SQL query to find those employees who get second-highest salary. Return all the fields of the employees.

SELECT \*

FROM employees

WHERE employee\_id IN

(SELECT employee\_id

FROM employees

WHERE salary =

(SELECT MAX(salary)

FROM employees

WHERE salary <

(SELECT MAX(salary)

FROM employees)));

1. write a SQL query to find those employees who earn more than the average salary and work in the same department as an employee whose first name contains the letter 'S'. Return employee ID, first name and salary.

SELECT employee\_id, first\_name , salary

FROM employees

WHERE salary >

(SELECT AVG (salary)

FROM employees )

AND department\_id IN

( SELECT department\_id

FROM employees

WHERE first\_name LIKE '%S%');

1. write a SQL query to calculate total salary of the departments where at least one employee works. Return department ID, total salary

SELECT department.department\_id, result1.total\_amt

FROM department,

( SELECT employees.department\_id, SUM(employees.salary) total\_amt

FROM employees

GROUP BY department\_id) result1

WHERE result1.department\_id = department.department\_id;

**Lab Program 5 - 5 (On Aggregate Functions and Nested / Sub Queries)**

1. Write a query to display the employee id, name ( first name and last name ) and the job id column with a modified title SALESMAN for those employees whose job title is ST\_MAN and DEVELOPER for whose job title is IT\_PROG

SELECT employee\_id, first\_name, last\_name,

CASE job\_id

WHEN ' FI\_MGR' THEN ‘MANAGER’

WHEN 'IT\_PROG' THEN 'DEVELOPER'

ELSE job\_id

END AS designation, salary

FROM employees;

1. write a SQL query to find those employees whose salaries exceed 50% of their department's total salary bill. Return first name, last name.

SELECT e1.first\_name, e1.last\_name

FROM employees e1

WHERE salary >

( SELECT (SUM(salary))\*.5

FROM employees e2

WHERE e1.department\_id=e2.department\_id);

1. write a SQL query to find those employees who manage a department. Return all the fields of employees table.

SELECT \*

FROM employees

WHERE employee\_id=ANY

( SELECT manager\_id FROM department );

1. write a SQL query to find those managers who supervise four or more employees. Return manager name, department ID.

SELECT first\_name || ' ' || last\_name AS Manager\_name,department\_id

FROM employees

WHERE employee\_id IN

(SELECT manager\_id

FROM employees

GROUP BY manager\_id

HAVING COUNT(\*)>=1);

1. write a SQL query to find the first name, last name, department, city, and state province for each employee.

SELECT E.first\_name,E.last\_name,

D.department\_name, L.city, L.state\_province

FROM employees E

JOIN department D

ON E.department\_id = D.department\_id

JOIN location L

ON D.location\_id = L.location\_id;

1. write a SQL query to find the employees and their managers. These managers do not work under any manager. Return the first name of the employee and manager.

SELECT E.first\_name AS "Employee Name",

M.first\_name AS "Manager"

FROM employees E

LEFT OUTER JOIN employees M

ON E.manager\_id = M.employee\_id;

1. write a SQL query to calculate the average salary, the number of employees receiving commissions in that department. Return department name, average salary and number of employees

SELECT department\_name, AVG(salary), COUNT(commission\_pct)

FROM department

JOIN employees USING (department\_id)

GROUP BY department\_name;

1. write a SQL query to find the department name, department ID, and number of employees in each department.

SELECT d.department\_name,

e.\*

FROM department d

JOIN

(SELECT count(employee\_id),

department\_id

FROM employees

GROUP BY department\_id) e USING (department\_id);

**Practice –**

1. write a SQL query to find those employees who earn more than the average salary. Sort the result-set in descending order by salary. Return first name, last name, salary, and department ID.

SELECT first\_name, last\_name , salary, department\_id

FROM employees

WHERE salary > (

SELECT AVG(salary)

FROM employees )

ORDER BY salary DESC;

1. write a SQL query to find the first name, last name, department number, and department name for each employee

SELECT E.first\_name , E.last\_name ,

E.department\_id , D.department\_name

FROM employees E

JOIN department D

ON E.department\_id = D.department\_id;