Concordia University
COMP 353 –Summer 2018
Sample Solution -Assignment # 4

1) **Find the count of departments, region name(s) and cities for the department(s) that have more than 500 employees.**

```
SELECT COUNTRY_NAME, CITY, COUNT(DEPARTMENT_ID)
FROM COUNTRIES JOIN LOCATIONS USING (COUNTRY_ID) JOIN DEPARTMENTS USING
(LOCATION_ID)
WHERE DEPARTMENT_ID IN
    (SELECT DEPARTMENT_ID FROM EMPLOYEES
         GROUP BY DEPARTMENT_ID
         HAVING COUNT(DEPARTMENT_ID)>500)
GROUP BY COUNTRY_NAME, CITY;
```

2) **For a department in which the max salary is greater than 100000 for employees who worked in the past, set the manager name as 'Picard'.**

Ans.

```
UPDATE EMPLOYEES
SET FIRST_NAME= 'Picard' WHERE DEPARTMENT_ID IN
(SELECT DEPARTMENT_ID FROM EMPLOYEES
 WHERE EMPLOYEE_ID IN (SELECT EMPLOYEE_ID FROM JOB_HISTORY)
 GROUP BY DEPARTMENT_ID
 HAVING MAX(SALARY) >10000);
```

3) **Find month and year which witnessed lowest count of employees joining a department located in 'Vancouver'.**

```
SELECT YEAR(E.HIRE_DATE) AS "YEAR", MONTH(E.HIRE_DATE) AS "MONTH", D.DEP_ID
FROM EMPLOYEES E
JOIN (DEPARTMENTS D, LOCATIONS L)
    ON D.DEP_ID = E.DEP_ID
    AND D.LOCATION_ID = L.LOCATION_ID
WHERE L.CITY = "VANCOUVER"
HAVING COUNT(E.EMP_ID) = (
    SELECT MIN(COUNT(E.EMP_ID))
    FROM EMPLOYEES
    JOIN (DEPARTMENTS, LOCATIONS)
        ON DEPARTMENTS.DEP_ID = EMPLOYEES.DEP_ID
        AND DEPARTMENTS.LOCATION_ID = LOCATIONS.LOCATION_ID
   WHERE LOCATIONS.CITY = "VANCOUVER"
);
```

4) **With the help of schema find the year which witnessed maximum number of employee intake.**

```
SELECT EXTRACT (YEAR FROM Joining_Date) FROM (
SELECT COUNT(Emp_ID), Joining_Date
FROM Employee_History
GROUP BY Joining_Date
HAVING MAX(Emp_ID)) AS JY;
```

This Stored procedure answers 4 & 5 as well

```
declare

      v_year  number(4);
      v_c     number(2);
begin
      select  to_char(hire_date,'yyyy') into v_year
      from  employees
      group by to_char(hire_date,'yyyy')
      having count(*) =
             ( select  max( count(*))
               from  employees
               group by to_char(hire_date,'yyyy'));

      dbms_output.put_line('Year : ' || v_year);

      for month in 1 .. 12
      loop
          select  count(*) into v_c
          from employees
          where  to_char(hire_date,'mm') = month and
to_char(hire_date,'yyyy') = v_year;

          dbms_output.put_line('Month : ' || to_char(month) || ' Employees :
' || to_char(v_c));

     end loop;

end;
```

5) **For the year in query-4, find how many joined in each month in that specific year.**

```
SELECT MONTH(E.HIRE_DATE) AS "MONTH", COUNT(E.HIRE_DATE) AS "EMPLOYEE COUNT"
FROM EMPLOYEES E
WHERE YEAR(E.HIRE_DATE) = (
    SELECT YEAR(E.HIRE_DATE)
    FROM EMPLOYEES E
    HAVING COUNT(E.EMP_ID) = (
        SELECT MAX(COUNT(E.EMP_ID))
        FROM EMPLOYEES
    )
)
GROUP BY MONTH(E.HIRE_DATE);
```

[PART-2]

6) **Create a trigger to ensure that a salary of an employee cannot exceed the salary of his/her manager. If the employee does not have a manager, then his/her salary cannot be more than 10% of the highest salary in the database.**

```
CREATE TRIGGER SALARY_UPDATE_UNDER_MANAGER_TRIGGER
BEFORE UPDATE ON EMPLOYEES
FOR EACH ROW
BEGIN
    DECLARE maxSalary INT;

        IF (NEW.MANAGER_ID IS NULL) THEN
        SET maxSalary = (1.1 * (SELECT MAX(SALARY) FROM EMPLOYEES));
               IF (NEW.SALARY > maxSalary) THEN
            SET NEW.SALARY = maxSalary;
        END IF;

    ELSE
        SET maxSalary = (SELECT SALARY FROM EMPLOYEES WHERE EMP_ID =
NEW.MANAGER_ID);
        IF (NEW.SALARY > maxSalary) THEN
            SET NEW.SALARY = maxSalary;
        END IF;
        END IF;
END;
```

7) For changes in the job of an employee, updated details provided below must be written to Employee History:
Hire date of the employee for start date, old job ID, old department ID, Employee ID, todays' system date for end date. In case a row is already present for employee job history then the start date must be the end date of that (row +1).

```
create or replace trigger trg_log_job_change
after update of job_id
on employees
for each row
declare
    v_enddate   date;
    v_startdate date;
begin
   -- find out whether the employee has any row in job_history table
   select max(end_date) into v_enddate
   from job_history
   where employee_id = :old.employee_id;

   if v_enddate is null then
      v_startdate := :old.hire_date;
   else
      v_startdate := v_enddate + 1;
   end if;

   insert into  job_history values (:old.employee_id, v_startdate, sysdate,
:old.job_id, :old.department_id);
end;
```

Note: Before testing the above trigger, you need to disable UPDATE_JOB_HISTORY trigger, which is already present in HR account, as it does the same.

8) **Make a Trigger to ensure that the salary of the employee is never decreased while working in an organization.**

```
create or replace trigger  trg_employees_salary_check
before update
on employees
for each row
begin
   if  :old.salary > :new.salary then
         raise_application_error(-20111,'Sorry! Salary can not be
decreased!');
   end if;
end;
```

9)**Create a trigger to ensure that an increase of salary for an employee is conform with the following rules: If experience is more than 8 years, increase salary by max 20%; If experience is greater than 3 years, increase salary by max of 10%; Otherwise a max increase of 5%.**

//Employee Considered here is with Employee_ID=115//

```
declare
    v_exp  number(2);
    v_hike number(5,2);
begin
    select  floor((sysdate-hire_date) / 365 ) into v_exp
    from employees
    where employee_id = 115;

    v_hike := 1.05;

    case
      when  v_exp > 10 then
            v_hike := 1.20;
      when  v_exp > 5  then
            v_hike := 1.10;
    end case;

    update employees set salary = salary * v_hike
    where employee_id = 115;
end;
```

10) **Create a trigger to ensure that Min_salary cannot exceed Max_salary for any job.**

/* To signal a generic SQLSTATE value, use '45000', which means “unhandled user-defined exception.” */

```
CREATE TRIGGER MIN_SALARY_UPDATE_TRIGGER
BEFORE UPDATE ON JOBS
FOR EACH ROW
BEGIN
        IF (NEW.MIN_SALARY > NEW.MAX_SALARY) THEN
                SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Min_salary cannot be more than Max_salary';
        END IF;
END;
```