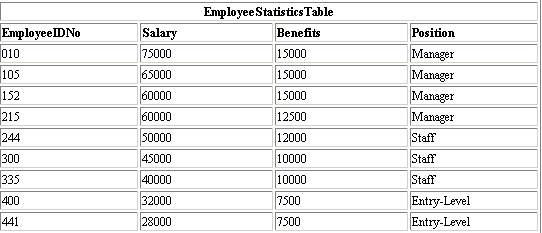
**SQL Exercises – Analytic Functions**

# Exercise 1:

You have the below table:



**1-** Develop a query that gives the smallest/minimum figure of the Benefits column, of the employees who are Managers.

**2-** Develop a query that gives the employeeidno and the maximum salary of employees by position.

**3-** Develop a query that gives the employeeidno and the maximum benefits of employee by position if the salary>40000

# Exercise 2:

You’ll see the use of the all\_sales table in the following sections.

All\_sales stores the sum of all the sales by dollar amount for a particular year, month, product type, and employee.

The all\_sales table is created as follows:

**CREATE TABLE SSDX\_TMP.ALL\_SALES**

**(**

**YEAR INTEGER NOT NULL,**

**MONTH INTEGER NOT NULL,**

**PRD\_TYPE\_ID INTEGER,**

**EMP\_ID INTEGER,**

**AMOUNT NUMBER (8, 2)**

**);**

**ALTER TABLE SSDX\_TMP.ALL\_SALES add CONSTRAINT ALL\_SALES\_PK**

**PRIMARY KEY (YEAR, MONTH, PRD\_TYPE\_ID, EMP\_ID);**

As you can see, the above table contains five columns, which are as follows:

\* Year stores the year the sales took place

\* Month stores the month the sales took place (1 to 12)

\* Prd\_type\_id stores the product\_type\_id of the product

\* Emp\_id stores the employee\_id of the employee who handled the sales

\* Amount stores the total dollar amount of the sales

The following output shows the first 13 rows in the all\_sales table:

**YEAR   MONTH PRD\_TYPE\_ID     EMP\_ID AMOUNT**

---------- ----------- -------------------- ----------- -------------

     2003          1 1          21    10034.84

     2003          2            1          21   15144.65

     2003          3            1          21     20137.83

     2003          4            1          21     25057.45

     2003          5            1          21     17214.56

     2003          6            1          21     15564.64

     2003          7            1         21     12654.84

     2003          8            1          21     17434.82

     2003          9            1          21     19854.57

     2003         10            1          21   21754.19

     2003         11            1          21   13029.73

     2003         12            1         21     10034.84

     2003          1           1          22   11034.84

**Questions:**

1) For the first three months of 2003, write the following query reports:

\* The total sum of all sales for all three months (labeled as total\_month\_amount)

\* The total sum of all sales for all product types (labeled as total\_product\_type\_amount)

2) Write a query to get the previous and next month’s sales amount

3) Write a query to get the ranking of sales by product type for 2003

# Exercise 3:

You have the following two tables:

**CREATE TABLE DEPT**

**(**

**DEPTNO NUMBER (2) NOT NULL PRIMARY KEY,**

**DNAME VARCHAR2 (14),**

**LOC VARCHAR2 (13)**

**);**

INSERT INTO DEPT (DEPTNO, DNAME, LOC) VALUES (10, 'ACCOUNTING', 'NEW YORK');

INSERT INTO DEPT (DEPTNO, DNAME, LOC) VALUES (20, 'RESEARCH', 'DALLAS');

INSERT INTO DEPT (DEPTNO, DNAME, LOC) VALUES (30, 'SALES', 'CHICAGO');

INSERT INTO DEPT (DEPTNO, DNAME, LOC) VALUES (40, 'OPERATIONS', 'BOSTON');

INSERT INTO DEPT (DEPTNO, DNAME, LOC) VALUES (50, 'KERNEL', '1');

COMMIT;

**CREATE TABLE EMP**

**(**

**EMPNO NUMBER (4) NOT NULL,**

**ENAME VARCHAR2 (10),**

**JOB VARCHAR2 (9),**

**MGR NUMBER (4),**

**HIREDATE DATE,**

**SAL NUMBER (7,2),**

**COMM NUMBER (7,2),**

**DEPTNO NUMBER (2) CONSTRAINT FK\_EMP REFERENCES DEPT (DEPTNO)**

**);**

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7369, 'SMITH', 'CLERK', 7902, 800.00, NULL, 20);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7499, 'ALLEN', 'SALESMAN', 7698, 1600.00, 300.00, 30);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7521, 'WARD', 'SALESMAN', 7698, 1250.00, 500.00, 30);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7566, 'JONES', 'MANAGER', 7839, 2975.00, NULL, 20);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7654, 'MARTIN','SALESMAN', 7698, 1250.00, 1400.00, 30);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7698, 'BLAKE', 'MANAGER', 7839, 2850.00, NULL, 30);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7782, 'CLARK', 'MANAGER', 7839, 2450.00, NULL, 10);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7788, 'SCOTT', 'ANALYST', 7566, 3000.00, NULL, 20);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7839, 'KING', 'PRESIDENT',NULL, 5000.00, NULL, 10);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7844, 'TURNER','SALESMAN', 7698, 1500.00, 0.00, 30);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7876, 'ADAMS', 'CLERK', 7788, 1100.00, NULL, 20);

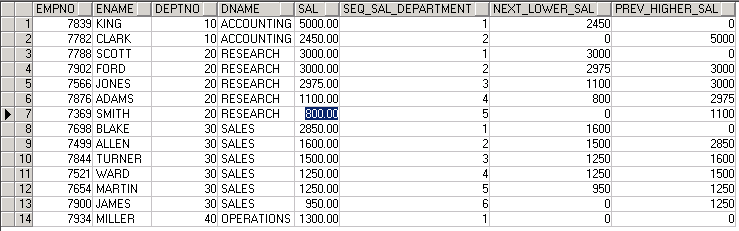
INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7900, 'JAMES', 'CLERK', 7698, 950.00, NULL, 30);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7902, 'FORD', ‘ANALYST’, 7566, 30000.00, NULL, 20);

INSERT INTO EMP (EMPNO, ENAME, JOB, MGR, SAL, COMM, DEPTNO) VALUES (7934, 'MILLER','CLERK', 7782, 1300.00, NULL, 40);

COMMIT;

**Write a query that output the below results:**



1. **SEQ\_SAL\_DEPARTMENT** is a sequence for the employee that order him depending on his salary in his department
2. **NEXT\_LOWER\_SAL** gives for each employee, the next lower salary than his salary in his department
3. **PREV\_HIGHER\_SAL** gives for each employee, the previous higher salary in his department.

# Exercise 4:

You have the below EMP table.

**Create table EMP**

**(**

**EID NUMBER,**

**ENAME VARCHAR2 (60),**

**DEPTNO NUMBER,**

**SAL NUMBER,**

**HIREDATE DATE**

**);**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) Values (1, 'CLARK', 10, 2450, to\_date('19-06-1981', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) Values (2, 'KING', 10, 5000, to\_date('01-11-1981', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (3, 'MILLER', 10, 1300, to\_date('23-01-1982', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (4, 'ADANS', 20, 1100, to\_date('01-01-1983', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (5, 'FORD', 20, 3000, to\_date('01-12-1981', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (6, 'JONES', 20, 2975, to\_date('01-04-1981', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (7, 'SCOTT', 20, 3000, to\_date('01-12-1982', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (8, 'SMITH', 20, 800, to\_date('01-12-1980', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (9, 'ALLEN', 30, 1600, to\_date('01-02-1981', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (10, 'BLAKE', 30, 2850, to\_date('01-05-1981', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (11, 'JAMES', 30, 950, to\_date('01-12-1981', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (12, 'MARTIN', 30, 1250, to\_date('01-09-1981', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (13, 'TURNER', 30, 1500, to\_date('01-09-1981', 'dd-mm-yyyy'));**

**Insert into EMP (EID, ENAME, DEPTNO, SAL, HIREDATE) values (14, 'WARD', 30, 1250, to\_date('01-02-1981', 'dd-mm-yyyy'));**

**COMMIT;**

1. Write a query that shows by department and for each employee, its hire date, how many days before was the last hire in the department and when, how many days after was the last hire and when.
2. Write a query that shows for each employee, its department and the name of the employee with the lowest salary in this department.
3. Repeat question 1 & 2 without using analytic functions.
4. Write a query that shows the top 3 salary earners in each department as columns. The query needs to return exactly 1 row per department and the row would have 4 columns. The DEPTNO, the name of the highest paid employee in the department, the name of the next highest paid, and so on.

# Exercise 5:

You have the below SALES table:

**CREATE TABLE SALES**

**(**

**SALE\_ID INTEGER,**

**PRODUCT\_ID INTEGER,**

**YEAR INTEGER,**

**Quantity INTEGER,**

**PRICE INTEGER**

**);**

**INSERT INTO SALES VALUES ( 1, 100, 2008, 10, 5000);**

**INSERT INTO SALES VALUES ( 2, 100, 2009, 12, 5000);**

**INSERT INTO SALES VALUES ( 3, 100, 2010, 25, 5000);**

**INSERT INTO SALES VALUES ( 4, 100, 2011, 16, 5000);**

**INSERT INTO SALES VALUES ( 5, 100, 2012, 8, 5000);**

**INSERT INTO SALES VALUES ( 6, 200, 2010, 10, 9000);**

**INSERT INTO SALES VALUES ( 7, 200, 2011, 15, 9000);**

**INSERT INTO SALES VALUES ( 8, 200, 2012, 20, 9000);**

**INSERT INTO SALES VALUES ( 9, 200, 2008, 13, 9000);**

**INSERT INTO SALES VALUES ( 10,200, 2009, 14, 9000);**

**INSERT INTO SALES VALUES ( 11, 300, 2010, 20, 7000);**

**INSERT INTO SALES VALUES ( 12, 300, 2011, 18, 7000);**

**INSERT INTO SALES VALUES ( 13, 300, 2012, 20, 7000);**

**INSERT INTO SALES VALUES ( 14, 300, 2008, 17, 7000);**

**INSERT INTO SALES VALUES ( 15, 300, 2009, 19, 7000);**

**COMMIT;**

**Questions:**

1. Write a query to find the number of products sold in each year.

2. Write a SQL query to find the total sales (QUANTITY) of each product.

3. Write a SQL query to find the cumulative sum of sales (QUANTITY) of each product.

4. Write a SQL query to find the sum of sales of current row and previous 2 rows in a product group.

5. Write a SQL query to find the minimum sales of a product without using the group by clause.

6. Write a SQL query to display the quantities sold for each product, separated by semi-colon and in ascending order by quantities.

# Exercise 6:

You have the following table:

EMPNO ENAME JOB SAL DEPT

------- ---------- --------- ---------- ----------

7369 SMITH CLERK 800 10

7900 JAMES CLERK 950 10

7876 ADAMS CLERK 1100 10

7521 WARD SALESMAN 1250 10

7654 MARTIN SALESMAN 1250 10

7934 MILLER CLERK 1300 20

7844 TURNER SALESMAN 1500 20

7499 ALLEN SALESMAN 1600 20

7782 CLARK MANAGER 2450 20

7698 BLAKE MANAGER 2850 20

7566 JONES MANAGER 2975 20

7788 SCOTT ANALYST 3000 30

7902 FORD ANALYST 3000 30

7839 KING PRESIDENT 5000 30

**Questions Using Analytic Functions:**

**1.** Return the average of salaries of all employees.

**2.** Return the departments ID, the employee name, and the average of salaries in each department.

**3.** Return the departments ID, the employee name, and the first higher salary reported in each department.

**4.** Return the department ID, employee names, and previous salaries (PREV\_SAL), with the difference between salaries and previous salaries (SALARY\_DIFF) ordered by ascending salaries.

# Exercise 7:

Regular expression exercises:

1. replace in a syntax more than one space with one ex: " Rabih     Melko  12  f"  with " Rabih Melko 12 f"
2. remove all alphabetic characters from a string to turn it to number
3. using a regular expression determine the value between the two dots " 19258.5587.357"
4. using a regular expression detect whether a string contain alphabetic values or not
5. how many customers’ ID exist in following list:"14215::62345::653623::86537::24764::73234::546::6477::12"