Lab1: Practical Assignment no.1

Title: Analytical functions.

Theory:

SQL is a standard language for storing, manipulating and retrieving data in databases.

Structured Query Language, abbreviated as SQL is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). It is particularly useful in handling structured data, i.e. data incorporating relations among entities and variables.

SQL Analytical function:

As it goes by the name, these are some special functions using which we can execute analytic queries on the dataset and obtain useful results. In comparison to standard SQL queries, sometimes it becomes necessary for the Data Analysts to deep dive into the data more and obtain insights from analytic perspectives

Table emp-

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7934	MILLER	CLERK	7782	23-JAN-82	1300	-	10
7782	CLARK	MANAGER	7839	09-JUN-81	2450	-	10
7788	SCOTT	ANALYST	7566	19-APR-87	3000	-	20
7876	ADAMS	CLERK	7788	23-MAY-87	1100	-	20
7900	JAMES	CLERK	7698	03-DEC-81	950	-	30
7566	JONES	MANAGER	7839	02-APR-81	2975	-	20
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7839	KING	PRESIDENT	-	17-NOV-81	5000	-	10
7698	BLAKE	MANAGER	7839	01-MAY-81	2850	-	30
7902	FORD	ANALYST	7566	03-DEC-81	3000	-	20
7369	SMITH	CLERK	7902	17-DEC-80	800	-	20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30

Table dept-

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
30	SALES	CHICAGO
20	RESEARCH	DALLAS
40	OPERATIONS	BOSTON

1. Display empno, ename, sal from emp table and give numbers to each row.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  SAL
FROM emp;
```

Output:

ROW_NUMBER	ENAME	EMPNO	SAL
1	ADAMS	7876	1100
2	ALLEN	7499	1600
3	BLAKE	7698	2850
4	CLARK	7782	2450
5	FORD	7902	3000
6	JAMES	7900	950
7	JONES	7566	2975
8	KING	7839	5000
9	MARTIN	7654	1250
10	MILLER	7934	1300
11	SCOTT	7788	3000
12	SMITH	7369	800
13	TURNER	7844	1500
14	WARD	7521	1250

Download CSV

2. Display empno, ename, sal and give numbers to each row in ascending order of
salary.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY ename ASC) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  SAL
FROM emp;
```

Output:

ROW_NUMBER	ENAME	EMPNO	SAL
1	ADAMS	7876	1100
2	ALLEN	7499	1600
3	BLAKE	7698	2850
4	CLARK	7782	2450
5	FORD	7902	3000
6	JAMES	7900	950
7	JONES	7566	2975
8	KING	7839	5000
9	MARTIN	7654	1250
10	MILLER	7934	1300
11	SCOTT	7788	3000
12	SMITH	7369	800
13	TURNER	7844	1500
14	WARD	7521	1250

3. Assign the ranks to employees in ascending order of salary.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  SAL,
  RANK() OVER(ORDER BY SAL ASC) Rank
FROM emp
ORDER BY Rank;
```

Output:

ROW_NUMBER	ENAME	EMPNO	SAL	RANK
12	SMITH	7369	800	1
6	JAMES	7900	950	2
1	ADAMS	7876	1100	3
9	MARTIN	7654	1250	4
14	WARD	7521	1250	4
10	MILLER	7934	1300	6
13	TURNER	7844	1500	7
2	ALLEN	7499	1600	8
4	CLARK	7782	2450	9
3	BLAKE	7698	2850	10
7	JONES	7566	2975	11
5	FORD	7902	3000	12
11	SCOTT	7788	3000	12
8	KING	7839	5000	14

4. Assign the ranks to employees in ascending order of salary using dense rank and point out the difference.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
```

```
  ENAME,
  EMPNO,
  SAL,
  DENSE_RANK() OVER(ORDER BY SAL ASC) Rank
FROM emp
ORDER BY Rank;
```

Output:

ROW_NUMBER	ENAME	EMPNO	SAL	RANK
12	SMITH	7369	800	1
6	JAMES	7900	950	2
1	ADAMS	7876	1100	3
9	MARTIN	7654	1250	4
14	WARD	7521	1250	4
10	MILLER	7934	1300	5
13	TURNER	7844	1500	6
2	ALLEN	7499	1600	7
4	CLARK	7782	2450	8
3	BLAKE	7698	2850	9
7	JONES	7566	2975	10
5	FORD	7902	3000	11
11	SCOTT	7788	3000	11
8	KING	7839	5000	12

5. Assign the ranks to employees in ascending order of salary but display records in descending order of salary.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  SAL,
  DENSE_RANK() OVER(ORDER BY SAL ASC) Rank
FROM emp
```

ORDER BY Rank DESC;

Output:

ROW_NUMBER	ENAME	EMPNO	SAL	RANK
8	KING	7839	5000	12
11	SCOTT	7788	3000	11
5	FORD	7902	3000	11
7	JONES	7566	2975	10
3	BLAKE	7698	2850	9
4	CLARK	7782	2450	8
2	ALLEN	7499	1600	7
13	TURNER	7844	1500	6
10	MILLER	7934	1300	5
14	WARD	7521	1250	4
9	MARTIN	7654	1250	4
1	ADAMS	7876	1100	3
6	JAMES	7900	950	2
12	SMITH	7369	800	1

6. Assign the rank to emp table rows in the ascending order of department and salary.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  DENSE_RANK() OVER(ORDER BY DEPTNO , SAL ASC) Rank
FROM emp
ORDER BY Rank;
```

Output:

ROW_NUMBER	ENAME	EMPNO	DEPTNO	SAL	RANK
10	MILLER	7934	10	1300	1
4	CLARK	7782	10	2450	2
8	KING	7839	10	5000	3
12	SMITH	7369	20	800	4
1	ADAMS	7876	20	1100	5
7	JONES	7566	20	2975	6
5	FORD	7902	20	3000	7
11	SCOTT	7788	20	3000	7
6	JAMES	7900	30	950	8
14	WARD	7521	30	1250	9
9	MARTIN	7654	30	1250	9
13	TURNER	7844	30	1500	10
2	ALLEN	7499	30	1600	11
3	BLAKE	7698	30	2850	12

7. Assign the rank to emp table rows in the ascending order of department and descending order of salary.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  DENSE_RANK() OVER(ORDER BY DEPTNO ASC, SAL DESC) Rank
FROM emp
ORDER BY Rank;
```

Output:

ROW_NUMBER	ENAME	EMPNO	DEPTNO	SAL	RANK
8	KING	7839	10	5000	1
4	CLARK	7782	10	2450	2
10	MILLER	7934	10	1300	3
11	SCOTT	7788	20	3000	4
5	FORD	7902	20	3000	4
7	JONES	7566	20	2975	5
1	ADAMS	7876	20	1100	6
12	SMITH	7369	20	800	7
3	BLAKE	7698	30	2850	8
2	ALLEN	7499	30	1600	9
13	TURNER	7844	30	1500	10
14	WARD	7521	30	1250	11
9	MARTIN	7654	30	1250	11
6	JAMES	7900	30	950	12

8. Calculate the ranks of employee for each department according to sal.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  RANK() OVER(PARTITION BY DEPTNO ORDER BY SAL DESC) Rank
FROM emp
ORDER BY DEPTNO;Output:
```

Output:

ROW_NUMBER	ENAME	EMPNO	DEPTNO	SAL	RANK
8	KING	7839	10	5000	1
4	CLARK	7782	10	2450	2
10	MILLER	7934	10	1300	3
11	SCOTT	7788	20	3000	1
5	FORD	7902	20	3000	1
7	JONES	7566	20	2975	3
1	ADAMS	7876	20	1100	4
12	SMITH	7369	20	800	5
3	BLAKE	7698	30	2850	1
2	ALLEN	7499	30	1600	2
13	TURNER	7844	30	1500	3
14	WARD	7521	30	1250	4
9	MARTIN	7654	30	1250	4
6	JAMES	7900	30	950	6

9. For the above query use dense_rank() and point out the difference.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  DENSE_RANK() OVER(PARTITION BY DEPTNO ORDER BY SAL DESC) Rank
FROM emp
ORDER BY DEPTNO;
```

Output:

ROW_NUMBER	ENAME	EMPNO	DEPTNO	SAL	RANK
8	KING	7839	10	5000	1
4	CLARK	7782	10	2450	2
10	MILLER	7934	10	1300	3
11	SCOTT	7788	20	3000	1
5	FORD	7902	20	3000	1
7	JONES	7566	20	2975	2
1	ADAMS	7876	20	1100	3
12	SMITH	7369	20	800	4
3	BLAKE	7698	30	2850	1
2	ALLEN	7499	30	1600	2
13	TURNER	7844	30	1500	3
14	WARD	7521	30	1250	4
9	MARTIN	7654	30	1250	4
6	JAMES	7900	30	950	5

Here we can see the difference is that DENSE_RANK() provides unique rank to each departments.

10. Calculate the ranks of employee for each department & display only top 2 high salaried employees for each of them.

Script:

```
SELECT ENAME,DEPTNO,SAL,RANK FROM

(SELECT  ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  DENSE_RANK() OVER(PARTITION BY DEPTNO ORDER BY SAL DESC) as Rank
FROM emp)
```

WHERE RANK<=2;

Output:

ENAME	DEPTNO	SAL	RANK
KING	10	5000	1
CLARK	10	2450	2
SCOTT	20	3000	1
FORD	20	3000	1
JONES	20	2975	2
BLAKE	30	2850	1
ALLEN	30	1600	2

11. Find out top 3 low salaried employees for each department.

Script:

```
SELECT ENAME,DEPTNO,SAL,RANK FROM

(SELECT  ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  DENSE_RANK() OVER(PARTITION BY DEPTNO ORDER BY SAL ASC) as Rank
FROM emp)
WHERE RANK<=3;
```

Output:

ENAME	DEPTNO	SAL	RANK
MILLER	10	1300	1
CLARK	10	2450	2
KING	10	5000	3
SMITH	20	800	1
ADAMS	20	1100	2
JONES	20	2975	3
JAMES	30	950	1
MARTIN	30	1250	2
WARD	30	1250	2
TURNER	30	1500	3

12. Find out top 2 low salaried employees.

Script:

```
SELECT ENAME,DEPTNO,SAL,RANK FROM

(SELECT  ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  DENSE_RANK() OVER(ORDER BY SAL ASC) as Rank
FROM emp)
WHERE RANK<=2;
```

Output:

ENAME	DEPTNO	SAL	RANK
SMITH	20	800	1
JAMES	30	950	2

13. Find information of employee who is having lowest sal in each department.

Script:

```
SELECT ENAME,DEPTNO,SAL,RANK FROM


(SELECT  ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  DENSE_RANK() OVER(PARTITION BY DEPTNO ORDER BY SAL ASC) as Rank
FROM emp)
WHERE RANK=1;
```

Output:

ENAME	DEPTNO	SAL	RANK
MILLER	10	1300	1
SMITH	20	800	1
JAMES	30	950	1

14. Assign row numbers in desc order of salary. Display the records in asc order of commission and null values of commission should come last.

Script:

```
SELECT ROW_NUMBER() OVER(ORDER BY SAL DESC) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  COMM,
  DENSE_RANK() OVER(PARTITION BY DEPTNO ORDER BY SAL DESC) Rank
FROM emp
ORDER BY COMM ASC;
```

Output:

ROW_NUMBER	ENAME	EMPNO	DEPTNO	SAL	COMM	RANK
8	TURNER	7844	30	1500	0	3
7	ALLEN	7499	30	1600	300	2
11	WARD	7521	30	1250	500	4
10	MARTIN	7654	30	1250	1400	4
5	BLAKE	7698	30	2850	-	1
6	CLARK	7782	10	2450	-	2
14	SMITH	7369	20	800	-	4
2	FORD	7902	20	3000	-	1
9	MILLER	7934	10	1300	-	3
12	ADAMS	7876	20	1100	-	3
13	JAMES	7900	30	950	-	5
4	JONES	7566	20	2975	-	2
3	SCOTT	7788	20	3000	-	1
1	KING	7839	10	5000	-	1

15. Display empno, ename, sal, comm., in desc order of comm. And replace all null values of comm. by 8888.

Script:

```
UPDATE emp
SET COMM=8888
WHERE COMM IS NULL;


SELECT
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  COMM

FROM emp
ORDER BY COMM DESC;
```

Output:

ENAME	EMPNO	DEPTNO	SAL	COMM
KING	7839	10	5000	8888
BLAKE	7698	30	2850	8888
CLARK	7782	10	2450	8888
JONES	7566	20	2975	8888
SCOTT	7788	20	3000	8888
FORD	7902	20	3000	8888
MILLER	7934	10	1300	8888
SMITH	7369	20	800	8888
JAMES	7900	30	950	8888
ADAMS	7876	20	1100	8888
MARTIN	7654	30	1250	1400
WARD	7521	30	1250	500
ALLEN	7499	30	1600	300
TURNER	7844	30	1500	0

16. Display empname, job & sal. Give ranking to sal job wise.

Script:

```
SELECT
  ENAME,
  JOB,
  SAL,
  DENSE_RANK() OVER(ORDER BY SAL DESC) Rank
FROM emp
```

Output:

ENAME	JOB	SAL	RANK
KING	PRESIDENT	5000	1
SCOTT	ANALYST	3000	2
FORD	ANALYST	3000	2
JONES	MANAGER	2975	3
BLAKE	MANAGER	2850	4
CLARK	MANAGER	2450	5
ALLEN	SALESMAN	1600	6
TURNER	SALESMAN	1500	7
MILLER	CLERK	1300	8
MARTIN	SALESMAN	1250	9
WARD	SALESMAN	1250	9
ADAMS	CLERK	1100	10
JAMES	CLERK	950	11
SMITH	CLERK	800	12

17. Display the details of all salesman using dense rank on ascending order of salary.

Script:

```
SELECT
```

```
EMPNO,
MGR,
COMM,
DEPTNO,
  ENAME,
  JOB,
  SAL,
      DENSE_RANK() OVER(ORDER BY SAL ASC) Rank
    FROM emp
    WHERE JOB='SALESMAN';
```

Output:

EMPNO	MGR	COMM	DEPTNO	ENAME	JOB	SAL	RANK
7521	7698	500	30	WARD	SALESMAN	1250	1
7654	7698	1400	30	MARTIN	SALESMAN	1250	1
7844	7698	0	30	TURNER	SALESMAN	1500	2
7499	7698	300	30	ALLEN	SALESMAN	1600	3

18. Display first 5 records of employee in descending order of salary.

Script:

```
SELECT * FROM
(SELECT  ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  DENSE_RANK() OVER(ORDER BY SAL DESC) as Rank
FROM emp)
WHERE ROW_NUMBER<=5;
```

Output:

ROW_NUMBER	ENAME	EMPNO	DEPTNO	SAL	RANK
1	ADAMS	7876	20	1100	10
2	ALLEN	7499	30	1600	6
3	BLAKE	7698	30	2850	4
4	CLARK	7782	10	2450	5
5	FORD	7902	20	3000	2

19. Display first 5 records of employee in ascending order of salary, replace null values of comm. by zero

Script:

```
UPDATE emp
SET COMM=0
WHERE COMM IS NULL;
```

```
SELECT * FROM
(SELECT  ROW_NUMBER() OVER(ORDER BY ename) AS ROW_NUMBER,
  ENAME,
  EMPNO,
  DEPTNO,
  SAL,
  COMM,
  DENSE_RANK() OVER(ORDER BY SAL ASC) as Rank
```

FROM emp)

WHERE ROW_NUMBER<=5;

Output:

ROW_NUMBER	ENAME	EMPNO	DEPTNO	SAL	COMM	RANK
1	ADAMS	7876	20	1100	0	3
2	ALLEN	7499	30	1600	300	7
3	BLAKE	7698	30	2850	0	9
4	CLARK	7782	10	2450	0	8
5	FORD	7902	20	3000	0	11

20. Create weather table with fields month, year and avgtemp. Values could be:

Month	Year	Avgtemp
1	2012	14.5
2	2012	34.5

Put atleast 12 records for 4 different years.

a) Use rank function to display the information of weather in order of hottest to coolest month year to year.

b) Find the hottest month of every year

Script:

```
CREATE TABLE WEATHER (

MONTH   NUMBER,

YEAR    NUMBER,

AVGTEMP  FLOAT

);

INSERT INTO WEATHER VALUES (1,2012,14.5);

INSERT INTO WEATHER VALUES (2,2012,15.5);
```

```
INSERT INTO WEATHER VALUES (3,2011,13.5);
INSERT INTO WEATHER VALUES (4,2012,15.5);
INSERT INTO WEATHER VALUES (5,2013,13.5);
INSERT INTO WEATHER VALUES (6,2014,15.5);
INSERT INTO WEATHER VALUES (7,2015,16.5);
INSERT INTO WEATHER VALUES (8,2016,15.5);
INSERT INTO WEATHER VALUES (9,2017,13.5);
INSERT INTO WEATHER VALUES (10,2018,25.5);
INSERT INTO WEATHER VALUES (11,2019,23.5);
INSERT INTO WEATHER VALUES (12,2012,23.5);
```

a.

```
SELECT
  MONTH,
  YEAR,
  AVGTEMP

FROM WEATHER
 ORDER BY AVGTEMP DESC;
```

Output:

MONTH	YEAR	AVGTEMP
10	2018	25.5
11	2019	23.5
12	2012	23.5
7	2015	16.5
6	2014	15.5
4	2012	15.5
8	2016	15.5
2	2012	15.5
1	2012	14.5
5	2013	13.5
3	2011	13.5
9	2017	13.5

b.

Script:

```
SELECT MONTH,YEAR,AVGTEMP FROM WEATHER WHERE AVGTEMP IN (SELECT
MAX(AVGTEMP) FROM WEATHER GROUP BY YEAR) ORDER BY YEAR ASC
```

Output:

MONTH	YEAR	AVGTEMP
7	2001	25.5
9	2001	25.5
9	2002	25.5
7	2002	25.5
9	2003	25.5
7	2003	25.5

2. Analytical Functions.

1. Write a query for finding highest and lowest salary of each department.

Script:

SELECT DEPTNO,MAX(SAL),MIN(SAL) FROM EMP GROUP BY DEPTNO ORDER BY DEPTNO;

Output:

DEPTNO	MAX(SAL)	MIN(SAL)
10	5000	1300
20	3000	800
30	2850	950

2. Write a query to find information of employees who were hired first in each department.

   Script:

   SELECT ENAME,DEPTNO,HIREDATE FROM EMP WHERE HIREDATE IN (SELECT MIN(HIREDATE) FROM EMP GROUP BY DEPTNO)

Output:

ENAME	DEPTNO	HIREDATE
CLARK	10	09-JUN-81
SMITH	20	17-DEC-80
ALLEN	30	20-FEB-81

3. Write a query which returns the salary from previous row; give the column name as sal_prev. Calculate the difference between sal of current row and that of previous row.

   Script:

SELECT SAL,FIRST_VALUE(sal) OVER (ORDER BY sal ROWS BETWEEN 1 PRECEDING AND CURRENT ROW) AS sal_prev ,SAL - FIRST_VALUE(sal) OVER (ORDER BY sal ROWS BETWEEN 1 PRECEDING AND CURRENT ROW) AS sal_DIFF FROM EMP;

Output:

SAL	SAL_PREV	SAL_DIFF
800	800	0
950	800	150
1100	950	150
1250	1100	150
1250	1250	0
1300	1250	50
1500	1300	200
1600	1500	100
2450	1600	850
2850	2450	400
2975	2850	125
3000	2975	25
3000	3000	0
5000	3000	2000

4. Write a query which returns a salary from next row, name it as sal_next and calculate the diff between the sal of current and following row.

   Script:

SELECT SAL,FIRST_VALUE(sal) OVER (ORDER BY sal ROWS BETWEEN CURRENT ROW AND 1 FOLLOWING) AS sal_next ,SAL - FIRST_VALUE(sal) OVER (ORDER BY sal ROWS BETWEEN 1 PRECEDING AND CURRENT ROW) AS sal_diff FROM EMP;

Output:

SAL	SAL_NEXT	SAL_DIFF
800	800	0
950	950	150
1100	1100	150
1250	1250	150
1250	1250	0
1300	1300	50
1500	1500	200
1600	1600	100
2450	2450	850
2850	2850	400
2975	2975	125
3000	3000	25
3000	3000	0
5000	5000	2000

5. Create a table with fields pro_id, order date, quantity. Insert at least 6 records into it.

   Script:

```
create table PRODUCT(
  PRO_ID      number(2,0),
  ORDERDATE       date,
  QTY         varchar2(13),
  constraint PK_PRODUCT primary key (PRO_ID)
)
```

Output:

TABLE PRODUCT

Column	Null?	Type
PRO_ID	NOT NULL	NUMBER(2,0)
ORDERDATE	-	DATE
QTY	-	VARCHAR2(13)

Script:

```
INSERT INTO PRODUCT VALUES(1,to_date('1-5-1981','dd-mm-yyyy'),2);
INSERT INTO PRODUCT VALUES(2,to_date('1-5-1981','dd-mm-yyyy'),2);
INSERT INTO PRODUCT VALUES(3,to_date('1-6-1981','dd-mm-yyyy'),4);
INSERT INTO PRODUCT VALUES(4,to_date('1-8-1981','dd-mm-yyyy'),5);
INSERT INTO PRODUCT VALUES(5,to_date('1-9-1981','dd-mm-yyyy'),2);
INSERT INTO PRODUCT VALUES(6,to_date('1-10-1981','dd-mm-yyyy'),065);
INSERT INTO PRODUCT VALUES(7,to_date('1-1-1982','dd-mm-yyyy'),82);
```

Output:

PRO_ID	ORDERDATE	QTY
1	01-MAY-81	2
2	01-MAY-81	2
3	01-JUN-81	4
4	01-AUG-81	5
5	01-SEP-81	2
6	01-OCT-81	65
7	01-JAN-82	82

6. Create a table student with roll no, stud_name, total_marks scored in last semester. Insert at least 6 records in it. Find out difference between different rank holders in class.

   Script:

```
create table STUDENT(
  ROLL_NO      number(2,0),
  STUDENT_NAME        varchar2(13),
  MARKS        number(3,0),
  constraint PK_STUDENT primary key (ROLL_NO)
)
```

Output:

TABLE STUDENT

Column	Null?	Type
ROLL_NO	NOT NULL	NUMBER(2,0)
STUDENT_NAME	-	VARCHAR2(13)
MARKS	-	NUMBER(3,0)

Download CSV

Script:

```
INSERT INTO STUDENT VALUES (1,'Ashvin',99);
INSERT INTO STUDENT VALUES (2,'Alex',70);
INSERT INTO STUDENT VALUES (3,'Carl',55);
INSERT INTO STUDENT VALUES (4,'Johnson',22);
INSERT INTO STUDENT VALUES (5,'Martin',0);
INSERT INTO STUDENT VALUES (6,'Jake',58);
INSERT INTO STUDENT VALUES (7,'Paul',94);
```

Output:

ROLL_NO	STUDENT_NAME	MARKS
1	Ashvin	99
2	Alex	70
3	Carl	55
4	Johnson	22
5	Martin	0
6	Jake	58
7	Paul	94

Script:

```
select student_name, marks ,FIRST_VALUE(marks) OVER (ORDER BY marks ROWS BETWEEN 1 PRECEDING AND CURRENT ROW) as marks_prev, marks - FIRST_VALUE(marks) OVER (ORDER BY marks ROWS BETWEEN 1 PRECEDING AND CURRENT ROW) AS marks_diff  from student;
```

Output:

STUDENT_NAME	MARKS	MARKS_PREV	MARKS_DIFF
Martin	0	0	0
Johnson	22	0	22
Carl	55	22	33
Jake	58	55	3
Alex	70	58	12
Paul	94	70	24
Ashvin	99	94	5