EMPLOYEE DATA ANALYSIS USING HIVE

==========

Input Dataset:

Users.txt
Locations.txt

HIVE Commands:

```
hive> CREATE DATABASE IF NOT EXISTS employeedb;

hive> USE employeedb;

hive> CREATE TABLE IF NOT EXISTS users
(
id INT,
name STRING,
salary INT,
unit STRING
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
;

hive> CREATE TABLE IF NOT EXISTS locations
(
id INT,
location STRING
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
;


hive> LOAD DATA LOCAL INPATH '/home/cloudera/Downloads/users.txt'
```

```
INTO TABLE users;

hive> LOAD DATA LOCAL INPATH '/home/cloudera/Downloads/locations.txt'
INTO TABLE locations;

hive> SELECT * FROM users;

hive> SELECT * FROM locations;

hive> DESCRIBE users;

hive> DESCRIBE locations;
```

Problem Statement 1:
Getting maximum salary across all the units

```
hive> SELECT unit, MAX(salary) FROM users GROUP BY unit;
```

Output:

```
DNA   300
ECS   600
FCS   500
FSI    900
```

Problem Statement 2:
Getting list of employees who have maximum salary across all the units

---

```
--Not possible with GROUP BY

hive> SELECT id, name, salary, rank FROM
    ( SELECT id, name, salary, rank() OVER (PARTITION BY unit ORDER BY salary
DESC) AS rank FROM users) temp WHERE rank = 1;
```

OutPut:

```
3       Yadav 300    1
8       Haya  600    1
4       Sunil  500    1
```

10 Chhaya 900 1

Problem Statement 3:
RANK according to salary

--Skips intermediate numbers in case of a tie.

```
hive> SELECT rank() OVER (ORDER BY salary), id, name, salary, unit FROM users;
```

Output:

```
1   5   Kranti  100   FCS
1   1   Amit    100   DNA
3   9   Swara   200   ECS
3   6   Mahoor        200   FCS
3   2   Sumit   200   DNA
6   3   Yadav   300   DNA
7   12  Rashi   500   FSI
7   7   Kohinoor      500   ECS
7   4   Sunil   500   FCS
10  8   Haya    600   ECS
11  11  Sam     700   FSI
12  10  Chhaya        900   FSI
```

Problem Statement 4:
DENSE_RANK according to salary

--Doesn't skip intermediate numbers in case of a tie.

```
hive> SELECT dense_rank() OVER (ORDER BY salary), id, name, salary, unit FROM
users;
```

Output:

```
1   5   Kranti  100   FCS
1   1   Amit    100   DNA
2   9   Swara   200   ECS
```

2	6	Mahoor	200	FCS
2	2	Sumit	200	DNA
3	3	Yadav	300	DNA
4	12	Rashi	500	FSI
4	7	Kohinoor	500	ECS
4	4	Sunil	500	FCS
5	8	Haya	600	ECS
6	11	Sam	700	FSI
7	10	Chhaya	900	FSI

Problem Statement 5:
DENSE_RANK according to salary for every unit

```
hive> SELECT dense_rank() OVER (PARTITION BY unit ORDER BY salary DESC) AS
rank, id, name, salary, unit
FROM users;
```

Output:

1	3	Yadav	300	DNA
2	2	Sumit	200	DNA
3	1	Amit	100	DNA
1	8	Haya	600	ECS
2	7	Kohinoor	500	ECS
3	9	Swara	200	ECS
1	4	Sunil	500	FCS
2	6	Mahoor	200	FCS
3	5	Kranti	100	FCS
1	10	Chhaya	900	FSI
2	11	Sam	700	FSI
3	12	Rashi	500	FSI

Problem Statement 6:
Top 2 highest paid employees for every unit

```
hive> SELECT name, salary, unit, rank FROM (SELECT dense_rank() OVER
(PARTITION BY unit ORDER BY salary DESC) AS rank, id, name, salary, unit FROM
users) temp WHERE rank <= 2;
```

Output:

```
Yadav 300    DNA   1
Sumit 200    DNA   2
Haya  600    ECS   1
Kohinoor     500   ECS   2
Sunil 500    FCS   1
Mahoor       200   FCS   2
Chhaya       900   FSI   1
Sam   700    FSI   2
```

Problem Statement 7:
Getting current name and salary alongwith next higher salary in the same unit

---

```
hive> SELECT name, salary, LEAD(salary) OVER (PARTITION BY unit ORDER BY
salary) FROM users;
```

Output:

```
Amit   100    200
Sumit  200    300
Yadav  300    NULL
Swara  200    500
Kohinoor      500    600
Haya   600    NULL
Kranti 100    200
Mahoor        200    500
Sunil  500    NULL
Rashi  500    700
Sam    700    900
Chhaya        900    NULL
```

Problem Statement 8:

Getting current name and salary along with next to next higher salary in the same unit

---

hive> SELECT name, salary, LEAD(salary, 2) OVER (PARTITION BY unit ORDER BY salary) FROM users;

Output:

```
Amit    100     300
Sumit   200     NULL
Yadav   300     NULL
Swara   200     600
Kohinoor        500     NUL
Haya    600     NULL
Kranti  100     500
Mahoor          200     NULL
Sunil   500     NULL
Rashi   500     900
Sam     700     NULL
Chhaya          900     NULL
```

Problem Statement 9:
Getting current name and salary along with next to next higher salary in the same unit replacing NULL with -1

---

hive> SELECT name, salary, LEAD(salary, 2, -1) OVER (PARTITION BY unit ORDER BY salary) FROM users;

Output:

```
Amit    100     300
Sumit   200     -1
Yadav   300     -1
Swara   200     600
Kohinoor        500     -1
Haya    600     -1
Kranti  100     500
Mahoor          200     -1
```

Sunil 500 -1
Rashi 500 900
Sam 700 -1
Chhaya 900 -1

Problem Statement 10:
Getting current name and salary along with the closest lower salary

---

```
hive> SELECT name, salary, LAG(salary) OVER (PARTITION BY unit ORDER BY
salary) FROM users;
```

Output:

Amit 100 NULL
Sumit 200 100
Yadav 300 200
Swara 200 NULL
Kohinoor 500 200
Haya 600 500
Kranti 100 NULL
Mahoor 200 100
Sunil 500 200
Rashi 500 NULL
Sam 700 500
Chhaya 900 700

Problem Statement 11:
Getting current name and salary along

with next to next lower salary in the same unit replacing NULL with -1

```
hive> SELECT name, salary, LAG(salary,2,-1) OVER (PARTITION BY unit ORDER BY
salary) FROM users;
```

Output:

Amit 100 -1

```
Sumit  200    -1
Yadav 300    100
Swara 200    -1
Kohinoor      500    -1
Haya   600    200
Kranti 100    -1
Mahoor        200    -1
Sunil   500    100
Rashi  500    -1
Sam    700    -1
Chhaya        900    500
```

Problem Statement 12:
Getting maximum salary in organization.

```
hive> SELECT MAX(salary) FROM users ;
```

Output:

```
900
```

Problem Statement 13:
Getting minimum salary in organization.

```
hive> SELECT Min(salary) FROM users ;
```

Output:

```
100
```

Problem Statement 14:
Getting average salary across all the units.

```
hive> SELECT unit, AVG(salary) FROM users GROUP BY unit;
```

Output:

```
DNA    200.0
```

ECS 433.33333333
FCS 266.66666666
FSI 700.0
