Sub-queries and Nested SELECTs

We will run through some SQL practice problems that will provide hands-on experience with nested SQL SELECT statements (also known as Sub-queries).

How does a typical Nested SELECT statement syntax look?

```
SELECT column_name [, column_name ]
FROM table1 [, table2 ]
WHERE column_name OPERATOR
   (SELECT column_name [, column_name ]
   FROM table1 [, table2 ]
   WHERE condition);
```

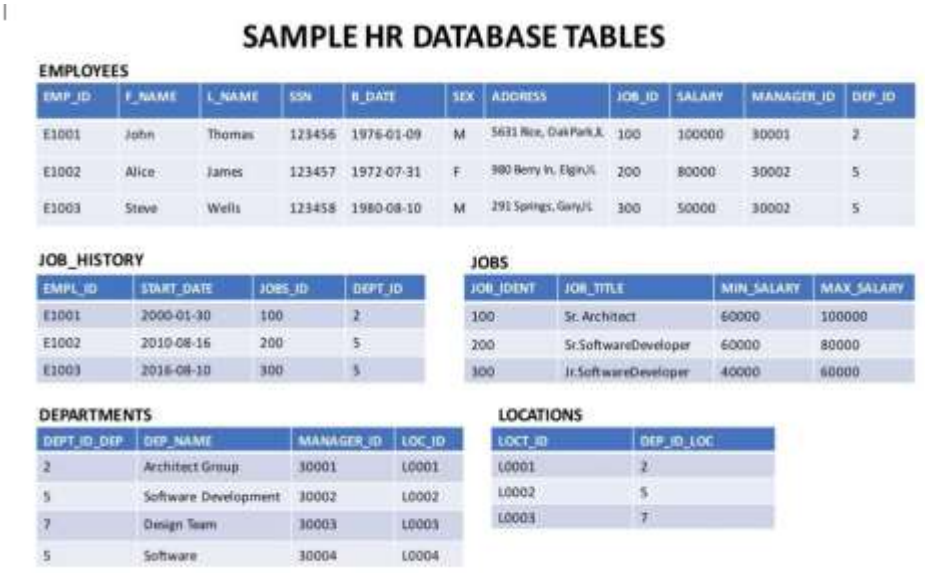
Software Used in this Lab

We will use an IBM Db2 Database. Db2 is a Relational Database Management System (RDBMS) from IBM, designed to store, analyze and retrieve data efficiently.

To complete this lab you will utilize a Db2 database service on IBM Cloud.

Database Used in this Lab

The database used in this lab is an internal database. You will be working on a sample HR database. This HR database schema consists of 5 tables called **EMPLOYEES**, **JOB_HISTORY**, **JOBS**, **DEPARTMENTS** and **LOCATIONS**. Each table has a few rows of sample data. The following diagram shows the tables for the HR database:



Objectives

- Writing SQL queries that demonstrate the necessity of using sub-queries
- Composing sub-queries in the where clause
- Building Column Expressions (i.e. sub-query in place of a column)
- Writing Table Expressions (i.e. sub-query in place of a table)

Instructions

When you approach the exercises in this lab, follow the instructions to run the queries on Db2:

- Go to the Resource List of IBM Cloud by logging in where you can find the Db2 service instance that you created in a previous lab under **Services** section. Click on the **Db2-xx service**. Next, open the Db2 Console by clicking on **Open Console** button. Click on the 3-bar menu icon in the top left corner and go to the **Run SQL** page. The Run SQL tool enables you to run SQL statements.

Exercise:

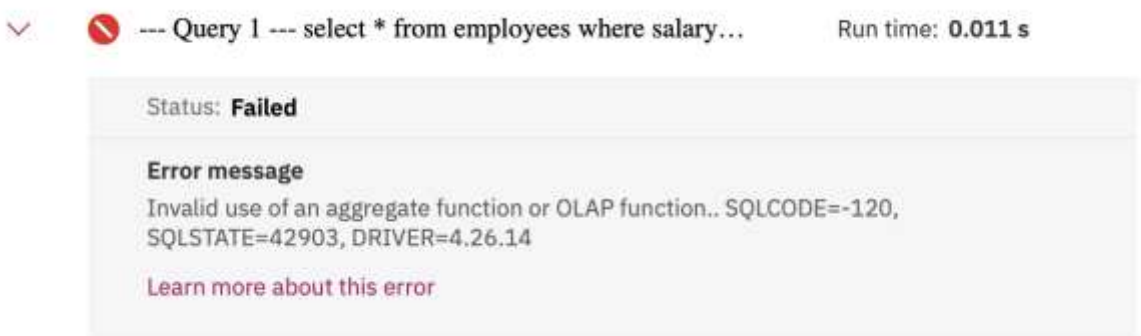
1. Problem:

   Execute a failing query (i.e. one which gives an error) to retrieve all employees records whose salary is lower than the average salary.

   Solution:

   ```
   select *
   from employees
   where salary < AVG(salary);
   ```

   Output:



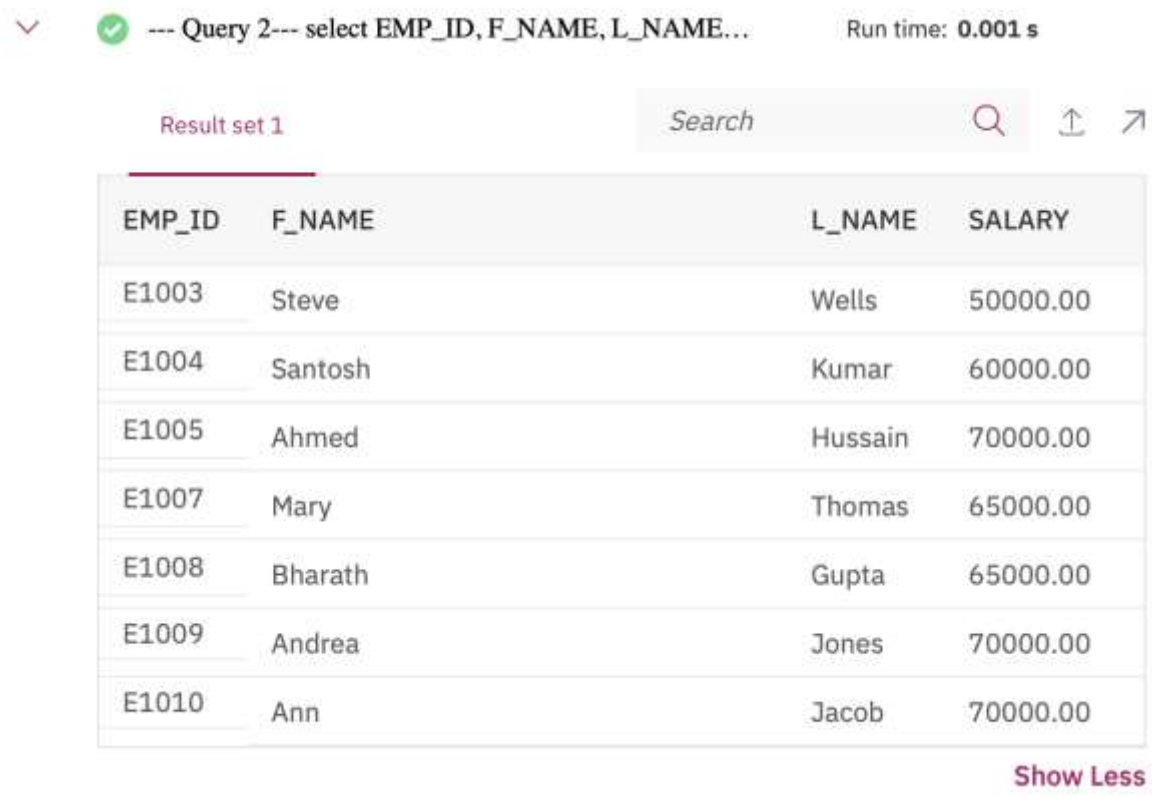
2. Problem:

Execute a working query using a sub-select to retrieve all employees records whose salary is lower than the average salary.

Solution:

```
select EMP_ID, F_NAME, L_NAME, SALARY
from employees
where SALARY < (select AVG(SALARY)
                from employees);
```

Output:



EMP_ID	F_NAME	L_NAME	SALARY
E1003	Steve	Wells	50000.00
E1004	Santosh	Kumar	60000.00
E1005	Ahmed	Hussain	70000.00
E1007	Mary	Thomas	65000.00
E1008	Bharath	Gupta	65000.00
E1009	Andrea	Jones	70000.00
E1010	Ann	Jacob	70000.00

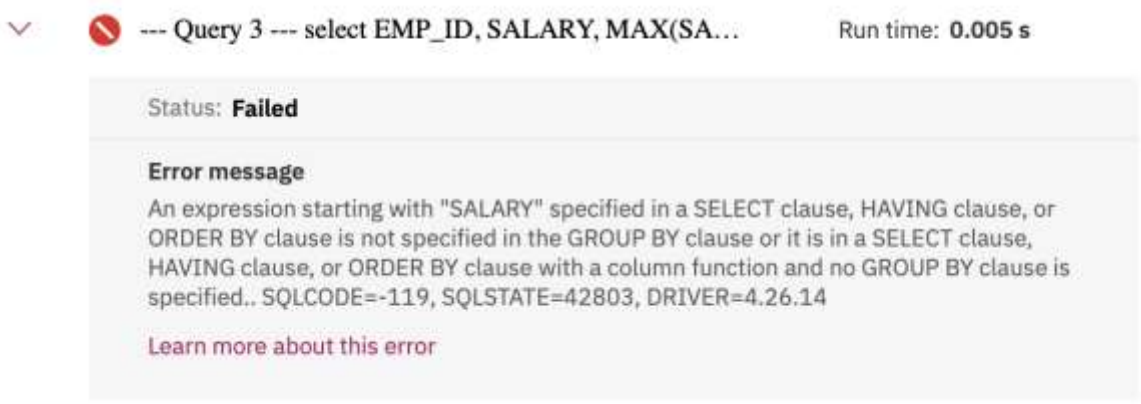
3. Problem:

   Execute a failing query (i.e. one which gives an error) to retrieve all employees records with EMP_ID, SALARY and maximum salary as MAX_SALARY in every row.

   Solution:

```
select EMP_ID, SALARY, MAX(SALARY) AS MAX_SALARY
from employees;
```

   Output:



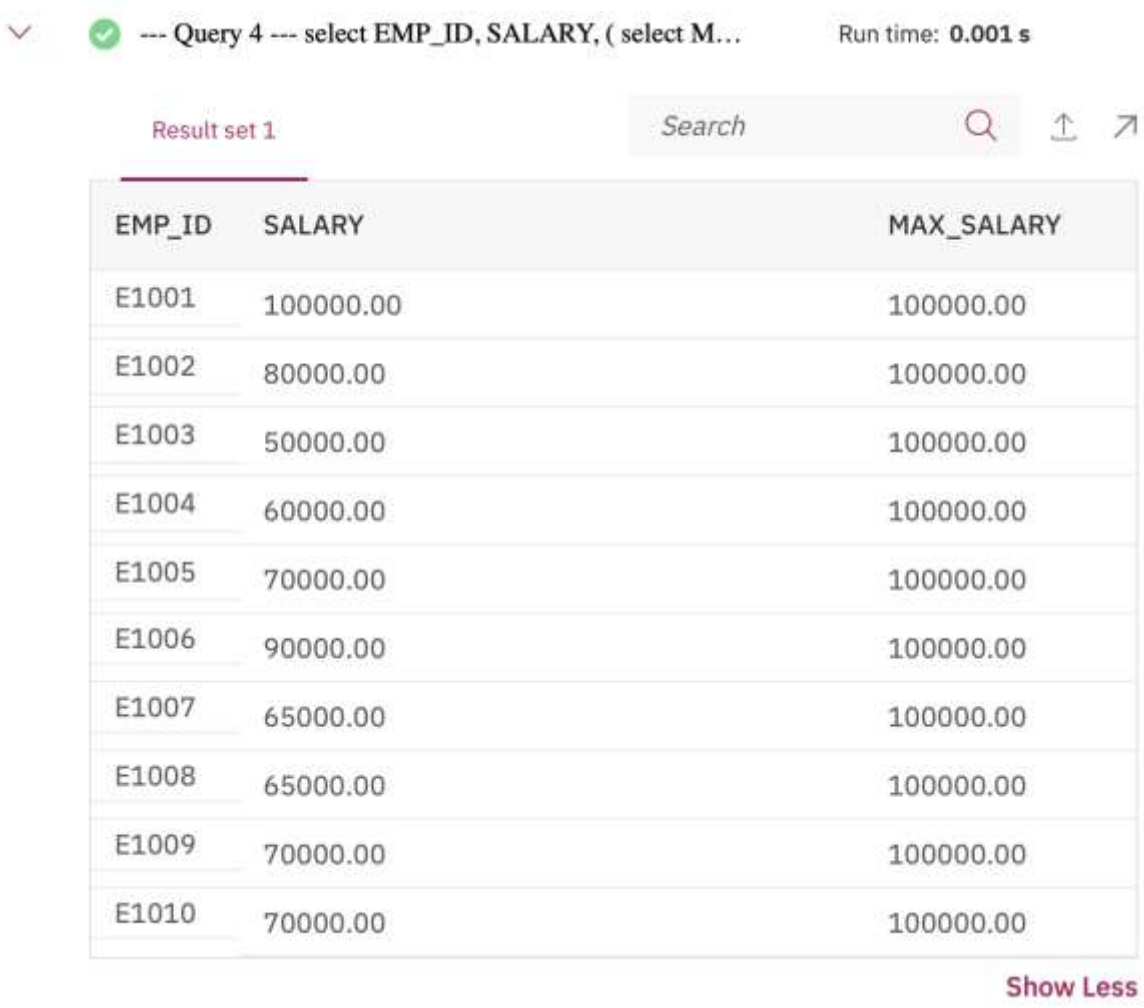
4. Problem:

Execute a Column Expression that retrieves all employees records with EMP_ID, SALARY and maximum salary as MAX_SALARY in every row.

Solution:

```
select EMP_ID, SALARY, ( select MAX(SALARY) from employees ) AS MAX_SALARY
from employees;
```

Output:



EMP_ID	SALARY	MAX_SALARY
E1001	100000.00	100000.00
E1002	80000.00	100000.00
E1003	50000.00	100000.00
E1004	60000.00	100000.00
E1005	70000.00	100000.00
E1006	90000.00	100000.00
E1007	65000.00	100000.00
E1008	65000.00	100000.00
E1009	70000.00	100000.00
E1010	70000.00	100000.00

Show Less

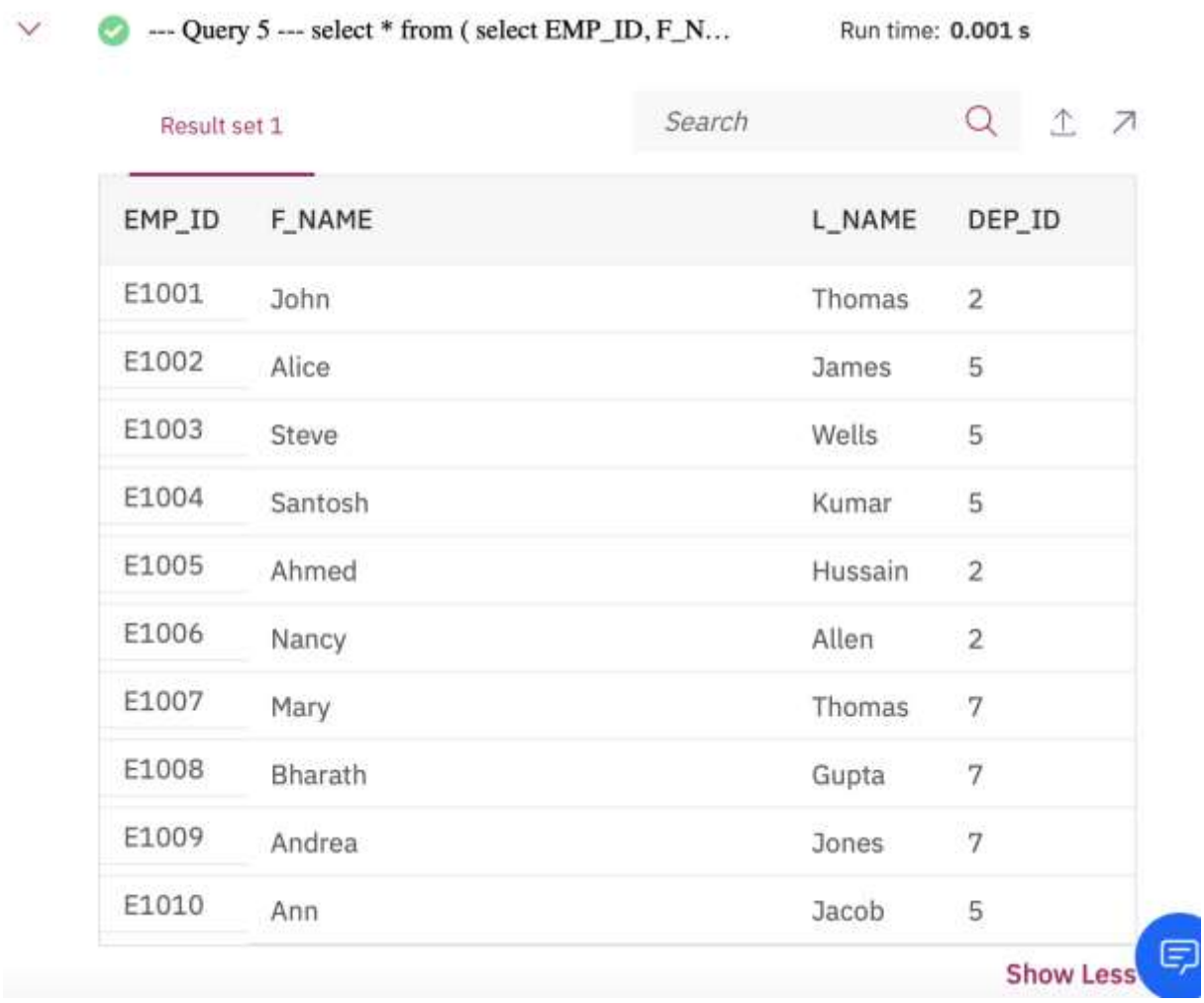
5. Problem:

Execute a Table Expression for the EMPLOYEES table that excludes columns with sensitive employee data (i.e. does not include columns: SSN, B_DATE, SEX, ADDRESS, SALARY).

Solution:

```
select * from ( select EMP_ID, F_NAME, L_NAME, DEP_ID from employees) AS EMP4ALL;
```

Output:



EMP_ID	F_NAME	L_NAME	DEP_ID
E1001	John	Thomas	2
E1002	Alice	James	5
E1003	Steve	Wells	5
E1004	Santosh	Kumar	5
E1005	Ahmed	Hussain	2
E1006	Nancy	Allen	2
E1007	Mary	Thomas	7
E1008	Bharath	Gupta	7
E1009	Andrea	Jones	7
E1010	Ann	Jacob	5