BUAN 6320

Database Foundations for Business Analytics

Assignment 3

Problem 1

Create the following table in your database with the following schema:

Table: `Point`

```
+-------------+------+
| Column Name | Type |
+-------------+------+
| x           | int  |
+-------------+------+
x is the primary key column for this table.
Each row of this table indicates the position of a point on the X-axis.
```

Add the following data to your tables:

Input:

```
Point table:
+----+
| x  |
+----+
| -1 |
| 0  |
| 2  |
+----+
```

Write an SQL query to report the shortest distance between any two points from the Point table.

HINT: To get the distances between two points, use ABS() function since the distance is nonnegative.

The result should be:

Output:

```
+----------+
| shortest |
+----------+
| 1        |
+----------+
```

Problem 2

Create the following table in your database with the following schema:

Table: `MyNumbers`

```
+-------------+------+
| Column Name | Type |
+-------------+------+
| num         | int  |
+-------------+------+
There is no primary key for this table. It may contain duplicates.
Each row of this table contains an integer.
```

Add the following data to your tables:

Input:

```
MyNumbers table:
+-----+
| num |
+-----+
| 8   |
| 8   |
| 3   |
| 3   |
| 1   |
| 4   |
| 5   |
| 6   |
+-----+
```

Write an SQL query to report the largest single number. If there is no single number, report null.

A single number is a number that appeared only once in the MyNumbers table.

The results should be:

Output:

```
+-----+
| num |
+-----+
| 6   |
+-----+
```

Problem 3

Create the following tables in your database with the following schema:

Table: `Employee`

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| empId       | int     |
| name        | varchar |
| supervisor  | int     |
| salary      | int     |
+-------------+---------+
empId is the primary key column for this table.
Each row of this table indicates the name and the ID of an employee in
addition to their salary and the id of their manager.
```

Table: `Bonus`

```
+-------------+------+
| Column Name | Type |
+-------------+------+
| empId       | int  |
| bonus       | int  |
+-------------+------+
empId is the primary key column for this table.
empId is a foreign key to empId from the Employee table.
Each row of this table contains the id of an employee and their respective
bonus.
```

Add the following data to your tables:

Input:

```
Employee table:
+-------+--------+------------+--------+
| empId | name   | supervisor | salary |
+-------+--------+------------+--------+
| 3     | Brad   | null       | 4000   |
| 1     | John   | 3          | 1000   |
| 2     | Dan    | 3          | 2000   |
| 4     | Thomas | 3          | 4000   |
+-------+--------+------------+--------+

Bonus table:
+-------+-------+
| empId | bonus |
+-------+-------+
| 2     | 500   |
| 4     | 2000  |
+-------+-------+
```

Write an SQL query to report the name and bonus amount of each employee with a bonus less than 1000.

Return the result table in any order.

The result should be:

Output:

name	bonus
Brad	null
John	null
Dan	500

Problem 4

Create the following table in your database with the following schema:

Table: `SalesPerson`

```
+-----------------+---------+
| Column Name     | Type    |
+-----------------+---------+
| sales_id        | int     |
| name            | varchar |
| salary          | int     |
| commission_rate | int     |
| hire_date       | date    |
+-----------------+---------+
sales_id is the primary key column for this table.
Each row of this table indicates the name and the ID of a salesperson
alongside their salary, commission rate, and hire date.
```

Table: `Company`

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| com_id      | int     |
| name        | varchar |
| city        | varchar |
+-------------+---------+
com_id is the primary key column for this table.
Each row of this table indicates the name and the ID of a company and the
city in which the company is located.
```

Table: `Orders`

```
+-------------+------+
| Column Name | Type |
+-------------+------+
| order_id    | int  |
| order_date  | date |
| com_id      | int  |
| sales_id    | int  |
| amount      | int  |
+-------------+------+
order_id is the primary key column for this table.
com_id is a foreign key to com_id from the Company table.
sales_id is a foreign key to com_id from the SalesPerson table.
Each row of this table contains information about one order. This includes
the ID of the company, the ID of the salesperson, the date of the order, and
the amount paid.
```

Add the following data to your tables:

Input:
SalesPerson table:

sales_id	name	salary	commission_rate	hire_date
1	John	100000	6	4/1/2006
2	Amy	12000	5	5/1/2010
3	Mark	65000	12	12/25/2008
4	Pam	25000	25	1/1/2005
5	Alex	5000	10	2/3/2007

Company table:

com_id	name	city
1	RED	Boston
2	ORANGE	New York
3	YELLOW	Boston
4	GREEN	Austin

Orders table:

order_id	order_date	com_id	sales_id	amount
1	1/1/2014	3	4	10000
2	2/1/2014	4	5	5000
3	3/1/2014	1	1	50000
4	4/1/2014	1	4	25000

Write an SQL query to report the names of all the salespersons who did not have any orders related to the company with the name "RED".

Return the result table in any order.

The results should be:

Output:

name
Amy
Mark
Alex