**Hacker Rank Functions in SQL problem solution,**

1. Query the list of *CITY* names from **STATION** that either do not start with vowels or do not end with vowels. Your result cannot contain duplicates.

**Input Format**

The **STATION** table is described as follows:



where *LAT\_N* is the northern latitude and *LONG\_W* is the western longitude.

ANSWER:

Select distinct city from station

Where

Not(city like ‘A%’ or city like ‘E%’ or city like ‘I%’ or city like ‘O%’ or city like ‘U%’) or

Not(city like ‘%A’ or city like ‘%E’ or city like ‘%I’ or city like ‘%O’ or city like ‘%U’);

2. Query the list of *CITY* names from **STATION** that *do not start* with vowels and *do not end* with vowels. Your result cannot contain duplicates.

**Input Format**

The **STATION** table is described as follows:



where *LAT\_N* is the northern latitude and *LONG\_W* is the western longitude.

ANSWER:

Select distinct city from station

Where

Not(city like ‘A%’ or city like ‘E%’ or city like ‘I%’ or city like ‘O%’ or city like ‘U%’) and

Not(city like ‘%A’ or city like ‘%E’ or city like ‘%I’ or city like ‘%O’ or city like ‘%U’);

3 . Write a query that prints a list of employee names (i.e.: the *name* attribute) from the **Employee** table in alphabetical order.

**Input Format**

The **Employee** table containing employee data for a company is described as follows:

| **Column** | **Type** |
| --- | --- |
| employee\_id | Integer |
| name | String |
| months | Integer |
| salary | Integer |

where *employee\_id* is an employee’s ID number, *name* is their name, *months* is the total number of months they’ve been working for the company, and *salary* is their monthly salary.

| **employee\_id** | **name** | **months** | **salary** |
| --- | --- | --- | --- |
| 12228 | Rose | 15 | 1968 |
| 33645 | Angela | 1 | 3443 |
| 45692 | Frank | 17 | 1608 |
| 56118 | Patrick | 7 | 1345 |
| 59725 | Lisa | 11 | 2330 |
| 74197 | Kimberly | 16 | 4372 |
| 78454 | Bonnie | 8 | 1771 |
| 83565 | Michael | 6 | 2017 |
| 98607 | Todd | 5 | 3396 |
| 99989 | Joe | 9 | 3573 |

ANSWER:

**SELECT** name

**FROM** employee

**ORDER BY** name;

4 . Write a query that prints a list of employee names (i.e.: the *name* attribute) for employees in **Employee** having a salary greater than **$2000** per month who have been employees for less than **10** months. Sort your result by ascending *employee\_id*.

**Input Format**

The **Employee** table containing employee data for a company is described as follows:

| **Column** | **Type** |
| --- | --- |
| employee\_id | Integer |
| name | String |
| months | Integer |
| salary | Integer |

where *employee\_id* is an employee’s ID number, *name* is their name, *months* is the total number of months they’ve been working for the company, and *salary* is the their monthly salary.

Explanation

*Angela* has been an employee for **1** month and earns **$3443** per month.  
*Michael* has been an employee for **6** months and earns **$2017** per month.  
*Todd* has been an employee for **5** months and earns **$3396** per month.  
*Joe* has been an employee for **9** months and earns **$3573** per month.  
We order our output by ascending *employee\_id*.

ANSWER:

**select** name **from** employee

**where** salary > 2000 and months <10

**order By** employee\_id;

5 . Query a *count* of the number of cities in **CITY** having a *Population* larger than **100,000**.

**Input Format**

The **CITY** table is described as follows:

| **Field** | **Type** |
| --- | --- |
| ID | NUMBER |
| NAME | VARCHAR2(17) |
| COUNTRYCODE | VARCHAR2(3) |
| DISTRICT | VARCHAR2(20) |
| POPULATION | NUMBER |

ANSWER:

**SELECT** **COUNT**(\*)

**FROM** **CITY**

**WHERE** **POPULATION** > 100000;

6 . Query the total population of all cities in **CITY** where *District* is **California**.

**Input Format**

The **CITY** table is described as follows:

| **Field** | **Type** |
| --- | --- |
| ID | NUMBER |
| NAME | VARCHAR2(17) |
| COUNTRYCODE | VARCHAR2(3) |
| DISTRICT | VARCHAR2(20) |
| POPULATION | NUMBER |

ANSWER:

**select** sum(population)

**from** city

**where** district ='California';

7 . Query the average population of all cities in **CITY** where *District* is **California**.

**Input Format**

The **CITY** is described as follows:

| **Field** | **Type** |
| --- | --- |
| ID | NUMBER |
| NAME | VARCHAR2(17) |
| COUNTRYCODE | VARCHAR2(3) |
| DISTRICT | VARCHAR2(20) |
| POPULATION | NUMBER |

ANSWER:

**select** avg(population)

**from** city

**where** district='California';

8 . Query the average population for all cities in **CITY**, rounded *down* to the nearest integer.

**Input Format**

The **CITY** is described as follows:

| **Field** | **Type** |
| --- | --- |
| ID | NUMBER |
| NAME | VARCHAR2(17) |
| COUNTRYCODE | VARCHAR2(3) |
| DISTRICT | VARCHAR2(20) |
| POPULATION | NUMBER |

ANSWER:

**select** floor(avg(population)) **from** city;

(or)

**select** round(avg(population),0) **from** city;

9 . Query the sum of the populations for all Japanese cities in **CITY**. The *COUNTRYCODE* for Japan is **JPN**.

**Input Format**

The **CITY** table is described as follows:

| **Field** | **Type** |
| --- | --- |
| ID | NUMBER |
| NAME | VARCHAR2(17) |
| COUNTRYCODE | VARCHAR2(3) |
| DISTRICT | VARCHAR2(20) |
| POPULATION | NUMBER |

ANSWER:

**select** sum(population)

**from** city

**where** countrycode='JPN';

10 . Query the difference between the maximum and minimum populations in **CITY**.

**Input Format**

The **CITY** table is described as follows:

| **Field** | **Type** |
| --- | --- |
| ID | NUMBER |
| NAME | VARCHAR2(17) |
| COUNTRYCODE | VARCHAR2(3) |
| DISTRICT | VARCHAR2(20) |
| POPULATION | NUMBER |

ANSWER:

**select** max(population) - min(population)

**from** city;