https://github.com/chhayac/SQL-hackerrank-problems/blob/master/basic-select.md

Query the **ALL** field for all American cities in the **CITY** table with populations larger than `100000`. The CountryCode for America is `USA`.
The **CITY** table is described as follows:

```
SELECT *
FROM CITY
WHERE COUNTRYCODE = 'USA' AND POPULATION > 100000
```

Query the **NAME** field for all American cities in the **CITY** table with populations larger than `120000`. The CountryCode for America is `USA`.
The **CITY** table is described as follows:

```
SELECT NAME
FROM CITY
WHERE COUNTRYCODE = 'USA' AND POPULATION > 120000
```

Query all columns (attributes) for every row in the **CITY** table.
The **CITY** table is described as follows:
SELECT *
FROM CITY

Query all columns for a city in **CITY** with the ID `1661`.
The **CITY** table is described as follows:

SELECT *
FROM CITY
WHERE ID = 1661

Query all attributes of every Japanese city in the **CITY** table.
The **COUNTRYCODE** for Japan is `JPN`.
The **CITY** table is described as follows:
SELECT *
FROM CITY
WHERE COUNTRYCODE = 'JPN'

Query the names of all the Japanese cities in the **CITY** table.
The **COUNTRYCODE** for Japan is `JPN`.
The **CITY** table is described as follows:
SELECT NAME

```
FROM CITY
WHERE COUNTRYCODE = 'JPN'
```

Query a list of **CITY** and **STATE** from the **STATION** table.
The **STATION** table is described as follows:

```
SELECT CITY, STATE
FROM STATION
```

Query a list of **CITY** names from **STATION** for cities that have an even **ID** number. Print the results in any order, but exclude duplicates from the answer.
The **STATION** table is described as follows:

```
SELECT DISTINCT CITY
FROM STATION
WHERE ID%2 = 0
```

Find the difference between the total number of **CITY** entries in the table and the number of distinct **CITY** entries in the table.
The **STATION** table is described as follows:

```
SELECT COUNT(CITY)  - COUNT(DISTINCT CITY)
FROM STATION
```

Query the two cities in **STATION** with the shortest and longest CITY names, as well as their respective lengths (i.e.: number of characters in the name). If there is more than one smallest or largest city, choose the one that comes first when ordered alphabetically.

```
Declare @Small int
Declare @Large int
select @Small = Min(Len(City)) from Station
select @Large = Max(Len(City)) from Station
select Top 1 City as SmallestCityName,Len(City) as Minimumlength from
Station where Len(City) = @Small Order by City Asc
select Top 1 City as LargestCityName,Len(City) as MaximumLength from
Station where Len(City) = @Large Order by City Asc
```

Query the list of CITY names starting with vowels (i.e., `a`, `e`, `i`, `o`, or `u`) from **STATION**. Your result cannot contain duplicates.

```
SELECT DISTINCT(CITY) FROM STATION WHERE CITY LIKE 'A%' OR
CITY LIKE 'E%' OR CITY LIKE 'I%' OR CITY LIKE 'O%'
OR CITY LIKE 'U%' ORDER BY CITY ASC;
```

Query the list of CITY names ending with vowels (a, e, i, o, u) from **STATION**. Your result cannot contain duplicates.

```
SELECT DISTINCT(CITY) FROM STATION WHERE CITY LIKE '%A' OR
CITY LIKE '%E' OR CITY LIKE '%I' OR CITY LIKE '%O'
OR CITY LIKE '%U' ORDER BY CITY ASC;
```

Query the list of CITY names from **STATION** which have vowels (i.e., a, e, i, o, and u) as both their first and last characters. Your result cannot contain duplicates.

```
SELECT DISTINCT CITY
FROM STATION
WHERE (CITY LIKE 'A%' OR CITY LIKE 'E%' OR CITY LIKE 'I%' OR CITY
LIKE 'O%' OR CITY LIKE 'U%') AND (CITY LIKE '%a' OR CITY LIKE '%e'
OR CITY LIKE '%i' OR CITY LIKE '%o' OR CITY LIKE '%u') order by
CITY;
```

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

```
SELECT DISTINCT CITY FROM STATION
WHERE upper(SUBSTR(CITY,1,1)) NOT IN ('A','E','I','O','U') AND lower(SUBSTR(CITY,1,1))
NOT IN
('a','e','i','o','u');
```

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

```
SELECT DISTINCT CITY
FROM STATION
WHERE UPPER(SUBSTR(CITY, LENGTH(CITY), 1)) NOT IN ('A','E','I','O','U') AND
LOWER(SUBSTR(CITY, LENGTH(CITY),1)) NOT IN ('a','e','i','o','u');
```

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.

```
SELECT DISTINCT CITY FROM STATION WHERE
LOWER(SUBSTR(CITY,1,1)) NOT IN ('a','e','i','o','u') OR
LOWER(SUBSTR(CITY, LENGTH(CITY),1)) NOT IN ('a','e','i','o','u');
```

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.

```
SELECT DISTINCT CITY FROM STATION WHERE
LOWER(SUBSTR(CITY,1,1)) NOT IN ('a','e','i','o','u') AND
LOWER(SUBSTR(CITY, LENGTH(CITY),1)) NOT IN ('a','e','i','o','u');
```

Query the Name of any student in **STUDENTS** who scored higher than  Marks. Order your output by the last three characters of each name. If two or more students both have names ending in the same last three characters (i.e.: Bobby, Robby, etc.), secondary sort them by ascending ID.

```
SELECT NAME
FROM STUDENTS
WHERE MARKS > 75
ORDER BY SUBSTR(NAME, LENGTH(NAME)-2, 3), ID;
```

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

```
SELECT NAME
FROM EMPLOYEE
ORDER BY NAME;
```

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.

```
SELECT NAME
FROM EMPLOYEE
WHERE SALARY > 2000 AND MONTHS < 10
ORDER BY EMPLOYEE_ID ASC;
```