Basic Select

1. Revising the Select Query I

Query all columns 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:

CITY

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

Solution

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

2. Revising the Select Query II

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:

CITY

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

Solution

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

3. Select All

Query all columns (attributes) for every row in the **CITY** table.

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

CITY

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

Solution:

SELECT * FROM CITY;

4. Select By ID

Query all columns for a city in **CITY** with the ID `1661`.

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

CITY

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

Solution

SELECT * FROM CITY WHERE ID = 1661;

5. Japanese Cities' Attributes

Query all attributes of every Japanese city in the **CITY** table. The **COUNTRYCODE** for Japan is `JPN`.

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

CITY

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

Solution:

SELECT * FROM CITY WHERE COUNTRYCODE = 'JPN';

6. Japanese Cities' Names

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:

CITY

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

Solution:

SELECT NAME FROM CITY WHERE COUNTRYCODE = 'JPN';

7. Weather Observation Station 1

Query a list of **CITY** and **STATE** from the **STATION** table.

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

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where **LAT_N** is the northern latitude and **LONG_W** is the western longitude.

Solution:

SELECT CITY, STATE FROM STATION;

8. Weather Observation Station 3

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:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where **LAT_N** is the northern latitude and **LONG_W** is the western longitude.

Solution:

SELECT DISTINCT CITY FROM STATION WHERE ID%2=0 ORDER BY CITY ASC;

9. Weather Observation Station 4

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:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where **LAT_N** is the northern latitude and **LONG_W** is the western longitude.

For example, if there are three records in the table with **CITY** values 'New York', 'New York', 'Bengalaru', there are 2 different city names: 'New York' and 'Bengalaru'. The query returns 1, because $\text{total number of records} - \text{number of unique city names} = 3 - 2 = 1$.

Solution:

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

10. Weather Observation Station 5

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.

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

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where **LAT_N** is the northern latitude and **LONG_W** is the western longitude.

Sample Input

For example, **CITY** has four entries: **DEF**, **ABC**, **PQRS** and **WXY**.

Sample Output

```
ABC 3
PQRS 4
```

Explanation

When ordered alphabetically, the **CITY** names are listed as **ABC**, **DEF**, **PQRS**, and **WXY**, with lengths $3, 3, 4,$ and 3. The longest name is **PQRS**, but there are 3 options for shortest named city. Choose **ABC**, because it comes first alphabetically.

Note

You can write two separate queries to get the desired output. It need not be a single query.

Solution:

(select city, length(city) from station order by length(city), city limit 1)

union

(select city, length(city) from station order by length(city) DESC,city limit 1)

11. Weather Observation Station 6

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

Input Format

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

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

Solution

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;

12. Weather Observation Station 7

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

Input Format

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

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

Solution

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';

13. Weather Observation Station 8

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.

Input Format

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

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

Solution:

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;

14. Weather Observation Station 9

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

Input Format

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

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

Solution:

SELECT DISTINCT(CITY) FROM STATION

WHERE (CITY Not LIKE 'a%' or CITY Not LIKE 'A%') and (CITY Not LIKE 'E%' or CITY Not LIKE 'e%')

and (CITY Not LIKE 'I%' or CITY Not LIKE 'i%') and (CITY Not LIKE 'O%' or CITY Not LIKE 'o%')

and (CITY Not LIKE 'U%' or CITY Not LIKE 'u%')

ORDER BY CITY ASC;

15. Weather Observation Station 10

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

Input Format

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

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

Solution:

```
SELECT DISTINCT(CITY) FROM STATION
WHERE (CITY Not LIKE '%A' or CITY Not LIKE '%a') and (CITY Not LIKE '%E' or CITY Not LIKE '%e')
and (CITY Not LIKE '%I' or CITY Not LIKE '%i') and (CITY Not LIKE '%O' or CITY Not LIKE '%o')
and (CITY Not LIKE '%U' or CITY Not LIKE '%u')
ORDER BY CITY ASC;
```

16. Weather Observation Station 11

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:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

Solution:

```
SELECT DISTINCT CITY FROM STATION
WHERE CITY NOT LIKE '[aeiou]%' or CITY NOT LIKE '%[aeiou]';
```

17. Weather Observation Station 12

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:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

Solution:

```
SELECT DISTINCT CITY FROM STATION
WHERE CITY NOT LIKE '[aeiou]%' and CITY NOT LIKE '%[aeiou]';
```

18. Higher Than 75 Marks

Query the Name of any student in **STUDENTS** who scored higher than 75 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.

Input Format

Column	Type
ID	Integer
Name	String
Marks	Integer

The **STUDENTS** table is described as follows: The Name column only contains uppercase (A-Z) and lowercase (a-z) letters.

Sample Input

ID	Name	Marks
1	Ashley	81
2	Samantha	75
4	Julia	76
3	Belvet	84

Sample Output

```
Ashley
Julia
Belvet
```

Explanation

Only Ashley, Julia, and Belvet have Marks > 75. If you look at the last three characters of each of their names, there are no duplicates and 'ley' < 'lia' < 'vet'.

Solution:

select name from students where marks > 75 order by RIGHT(name,3),id

19. Employee Names

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.

Sample Input

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

Sample Output

```
Angela
Bonnie
Frank
Joe
Kimberly
Lisa
Michael
Patrick
Rose
Todd
```

Solution:

select name from employee order by name

20. Employee Salaries

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.

Solution:

select name from employee where salary > 2000 and months < 10 order by employee_id