1. Query all columns for all American cities in the **CITY** table with populations larger than 100000. The **CountryCode** for America is USA.

Ans: select \* from city where countrycode='USA' and population>100000;

1. Query the **NAME** field for all American cities in the **CITY** table with populations larger than 120000. The CountryCode for America is USA.

ans: select name from city where population>120000 and countrycode='usa';

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

Select \* from city;

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

Ans: select \* from city where id=1661;

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

Ans:select \* from city where countrycode='JPN';

6) Query the names of all the Japanese cities in the **CITY** table. The **COUNTRYCODE** for Japan is JPN.

select name from city where countrycode='JPN';

whether observation1

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

Select city,name from station;

1. Weather Observation Station 2

Query the following two values from the **STATION** table:

1. The sum of all values in *LAT\_N* rounded to a scale of  decimal places.
2. The sum of all values in *LONG\_W* rounded to a scale of  decimal places.

select round(sum(lat\_n),2),round(sum(long\_w),2)from station;

1. 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.

select distinct(city) from station where ID%2=0;

1. 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.

select count(city)-count(distinct(city)) from station;

1. 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.

Whether 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.

Ans: select distinct city from station where left(city,1) in ('a','e','i','o','u') ;

SELECT DISTINCT(city) FROM station WHERE SUBSTR(city,1,1) IN ('a','e','i','o','u');

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%';

1. 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

select distinct city from station where right(city,1) in ('a','e','i','o','u');.

1. 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.

select distinct city from station where left(city,1) in ('a','e','i','o','u') and right(city,1) in ('a','e','i','o','u') ;

SELECT distinct city

FROM station

WHERE city LIKE '[aeiouAEIOU]%[aeiouAEIOU]';

1. Weather Observation Station 9

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 left(city,1) not in ('a','e','i','o','u');

SELECT DISTINCT CITY FROM STATION

WHERE CITY REGEXP '^[^aeiou].\*';

1. Weather Observation Station 10

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 city regexp '[^aeiou]$';

1. 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.

SELECT DISTINCT city

FROM station

WHERE city REGEXP '^[^aeiouAEIOU]|[^aeiouAEIOU]$'

1. 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.

select distinct city from station where city regexp '^[^aeiuo]'and city regexp '[^aeiuo]$';

1. Weather Observation Station 13

Query the sum of Northern Latitudes (LAT\_N) from **STATION** having values greater than  and less than . Truncate your answer to  decimal places.

SELECT round(sum(lat\_n),4) as su from station where lat\_n>38.7880 and lat\_n<137.2345;

1. Weather Observation Station 14

Query the greatest value of the Northern Latitudes (LAT\_N) from **STATION** that is less than . Truncate your answer to  decimal places.

select round(max(lat\_n),4) from station where lat\_n<137.2345;

1. Weather Observation Station 15

Query the Western Longitude (LONG\_W) for the largest Northern Latitude (LAT\_N) in **STATION** that is less than . Round your answer to  decimal places.

SELECT ROUND(LONG\_W, 4) FROM STATION WHERE LAT\_N =

(SELECT MAX(LAT\_N) FROM STATION WHERE LAT\_N < 137.2345);

1. Weather Observation Station 16

Query the smallest Northern Latitude (LAT\_N) from **STATION** that is greater than . Round your answer to  decimal places.

select round(lat\_n,4) from station where lat\_n=(select min(lat\_n) from station where lat\_n>38.7780);

1. Weather Observation Station 17

Query the Western Longitude (LONG\_W)where the smallest Northern Latitude (LAT\_N) in **STATION** is greater than . Round your answer to  decimal places.

select round(long\_w,4) from station where lat\_n=(select min(lat\_n) from station where lat\_n>38.7780);

1. Higher Than 75 Marks

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.

Ans:select name from station where marks>75 order by substr(name,length(name-2),3) ,id asc

1. Employee Names

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 asc;

1. 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  per month who have been employees for less than  months. Sort your result by ascending employee\_id.;

select name from employee where salary>2000 and months<10 order by employee\_id asc;

1. Revising Aggregations - The Count Function

Query a count of the number of cities in **CITY** having a Population larger than .

select count(id) from city where population>100000;

1. Revising Aggregations - The Sum Function

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

select sum(population) from city where district="california";

1. Revising Aggregations - Averages

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

select avg(population) from city where district='california';

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

select round(avg(population)) from city ;

1. Japan Population

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

select sum(population) from city where countrycode='JPN';

1. Population Density Difference

select max(population)-min(population) from city;

1. Top Earners

We define an employee's total earnings to be their monthly  worked, and the maximum total earnings to be the maximum total earnings for any employee in the **Employee** table. Write a query to find the maximum total earnings for all employees as well as the total number of employees who have maximum total earnings. Then print these values as  space-separated integers.

SELECT (months\*salary) as earnings,

COUNT(\*) FROM Employee

GROUP BY earnings

ORDER BY earnings DESC LIMIT 1;

1. Population Census

Given the **CITY** and **COUNTRY** tables, query the sum of the populations of all cities where the CONTINENT is 'Asia'.

select sum(c.population) from city c join country co on c.countrycode=co.code where co.continent='asia';

1. African Cities

Given the **CITY** and **COUNTRY** tables, query the names of all cities where the CONTINENT is 'Africa'.

select c.name from city c join country co on c.countrycode=co.code where co.continent='africa';