**Que. 1**: 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:

A screenshot of a computer

AI-generated content may be incorrect.

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

**Que.2**: 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:

A screenshot of a computer

AI-generated content may be incorrect.

**Ans**: select name from city where countrycode = 'USA' and population > 120000

**Que. 3:** Query all columns (attributes) for every row in the CITY table.

The CITY table is described as follows:

A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select \* from city

**Que. 4:** Query all columns for a city in CITY with the *ID* 1661.

The CITY table is described as follows:

A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select \* from city where ID = 1661

**Que. 5**: Query all attributes of every Japanese city in the **CITY** table. The **COUNTRYCODE** for Japan is JPN.

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

A screenshot of a computer

AI-generated content may be incorrect.

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

**Que. 6:** 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:

A screenshot of a computer

AI-generated content may be incorrect.

**Ans**: select name from city where countrycode = 'JPN'

**Que. 7**: Query a list of **CITY** and **STATE** from the **STATION** table.  
The **STATION** table is described as follows:

A table with text and numbers

AI-generated content may be incorrect.

**Ans**: select city, state from station

**Que.8**: 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:

A table with text and numbers

AI-generated content may be incorrect.

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

**Ans**: select distinct city from station where ID % 2 = 0

**Que.9**: 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:

A table with text and numbers

AI-generated content may be incorrect.

**Ans:** select count(city) - count(distinct city) from station

**Que: 10** Write a query that prints a list of employee names (i.e.: the *name* attribute) from the **Employee** table in alphabetical order.

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

A table of numbers with black text

AI-generated content may be incorrect.

**Ans:** select name from employee order by name asc

**Que: 11** 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*.

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

A table of numbers with black text

AI-generated content may be incorrect.

**Ans:** select name from employee

where salary > $2000 and months < 10

order by employee\_id asc

**Que: 12** Given the **CITY** and **COUNTRY** tables, query the sum of the populations of all cities where the *CONTINENT* is *'Asia'*. **Note:** *CITY.CountryCode* and *COUNTRY.Code* are matching key columns.

The **CITY** and **COUNTRY** tables are described as follows:

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select sum(c.population) from city as c

join country as ctry

on c.countrycode = ctry.code

where ctry.continent = 'Asia'

**Que: 13** Given the **CITY** and **COUNTRY** tables, query the names of all cities where the *CONTINENT* is *'Africa'*. **Note:** *CITY.CountryCode* and *COUNTRY.Code* are matching key columns.

The **CITY** and **COUNTRY** tables are described as follows:

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select c.name from city as c

join country as ctry

on c.countrycode = ctry.code

where ctry.continent = 'Africa'

**Que: 14** Given the **CITY** and **COUNTRY** tables, query the names of all the continents (*COUNTRY.Continent*) and their respective average city populations (*CITY.Population*) rounded *down* to the nearest integer. **Note:** *CITY.CountryCode* and *COUNTRY.Code* are matching key columns.

The **CITY** and **COUNTRY** tables are described as follows:

A screenshot of a computer

AI-generated content may be incorrect. A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select ctry.continent, avg(c.population) from country as ctry

join city as c

on ctry.code = c.countrycode

group by ctry.continent

**Que: 15** 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*.

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

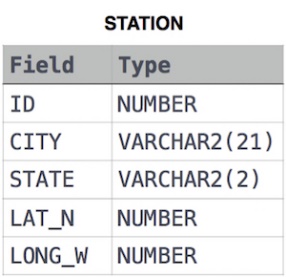
A table of numbers with text

AI-generated content may be incorrect.

**Ans:** select name from students where marks > 75

order by right(name,3), ID asc

**Que:16** 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. The **STATION** table is described as follows:



**Ans:** select distinct city from station

where left(city,1) not in ('A', 'E', 'I', 'O', 'U') and

right(city,1) not in ('A', 'E', 'I', 'O', 'U')

**Que:17** Query a *count* of the number of cities in **CITY** having a *Population* larger than  100000.

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

A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select count(name)from city where population > 100000

**Que:18** Query the total population of all cities in **CITY** where *District* is **California**.

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

A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select sum(population) from city where district = 'California'

**Que: 19** Query the average population of all cities in **CITY** where *District* is **California**.

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

A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select avg(population) from city where district = 'California'

**Que: 20** Query the average population for all cities in **CITY**, rounded *down* to the nearest integer.

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

A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select floor(avg(population)) from city

**Que: 21** Query the sum of the populations for all Japanese cities in **CITY**. The *COUNTRYCODE* for Japan is **JPN**. The **CITY** table is described as follows:

A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select sum(population) from city where countrycode = 'JPN'

**Que: 22** Query the difference between the maximum and minimum populations in **CITY**.

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

A screenshot of a computer

AI-generated content may be incorrect.

**Ans:** select max(population) - min(population) from city

Que: 23 Query the list of *CITY* names starting with vowels (i.e., a, e, i, o, or u) from **STATION**. Your result *cannot* contain duplicates. The **STATION** table is described as follows:

A table with text and numbers

AI-generated content may be incorrect.

Ans: 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%'

Que:24 Query the list of *CITY* names ending with vowels (a, e, i, o, u) from **STATION**. Your result *cannot* contain duplicates. The **STATION** table is described as follows:

A table with text and numbers

AI-generated content may be incorrect.

Ans: 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'

**Que: 25** 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. The **STATION** table is described as follows:

A table with text and numbers

AI-generated content may be incorrect.

**Ans:** 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')

**Que: 26** Query the list of *CITY* names from **STATION** that *do not start* with vowels. Your result cannot contain duplicates. The **STATION** table is described as follows:

A table with text and numbers

AI-generated content may be incorrect.

**Ans:** select distinct city from station

where left(city, 1) NOT IN ('A', 'E', 'I', 'O', 'U')

**Que: 27** Query the list of *CITY* names from **STATION** that *do not end* with vowels. Your result cannot contain duplicates. The **STATION** table is described as follows:

A table with text and numbers

AI-generated content may be incorrect.

**Ans:** select distinct city from station

where right(city, 1) NOT IN ('A', 'E', 'I', 'O', 'U')

**Que: 28** 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. The **STATION** table is described as follows:

A table with text and numbers

AI-generated content may be incorrect.

**Ans:** select distinct city from station

where left(city, 1) NOT IN ('A', 'E', 'I', 'O', 'U') OR

right(city, 1) NOT IN ('A', 'E', 'I', 'O', 'U')

Que: 29