SQL

Note: Keep in mind that SQL isn't case sensitive

Database name = Sample Data

Table 1 = Country

id	name	population	area
1	france	66600000	640680
2	Germany	80700000	357000
3	Iceland	376248	103125

Table 2 = City

id	name	country_id	population	rating
1	Paris	1	2243000	5
2	Berlin	2	3460000	3
3	Warsaw	4	1863056	6

◉ Select Operation

→   SELECT * FROM country

id	name	population	area
1	france	66600000	640680
2	Germany	80700000	357000
3	Iceland	376248	103125

→   SELECT * FROM city

id	name	country_id	population	rating
1	Paris	1	2243000	5
2	Berlin	2	3460000	3
3	Warsaw	4	1863056	6

⦿ Selecting according to ascending order

→ SELECT name
→ FROM city
→ ORDER BY rating ASC;

name
Berlin
Paris
Warsaw

⦿ Selecting according to descending order

→ SELECT name
→ FROM city
→ ORDER by DESC;

name
Warsaw
Paris
Berlin

⦿ ALIASES
✧ Columns

SELECT name as city_name
from city;

city_name
Paris
Berlin
Warsaw

✧ Tables

```
SELECT co.name, ci.name
FROM city as ci
JOIN country as co
on ci.country_id = co.id;
```

name	name
france	Paris
Germany	Berlin

⦿ Filtering the OUTPUT
➢ Comparison operators

- fetching the city having rating >4

```
SELECT name
FROM city
WHERE rating>4;
```

name
Paris
Warsaw

- fetching name of cities that are neither Berlin nor Madrid

```
SELECT name
FROM city
WHERE name != 'berlin'
AND name != 'Madrid';
```

←T→	name
☐ ✏ Edit ⧉ Copy ⊖ Delete	Paris
☐ ✏ Edit ⧉ Copy ⊖ Delete	Warsaw

➢ Text Operators
- Fetching the name of cities whose name start with 'P'

```
SELECT name
FROM city
WHERE name LIKE 'P%';
```

←T→	name
☐ ✏ Edit ⧉ Copy ⊖ Delete	Paris

- Fetching the name of cities whose name ends with 'N'

```
SELECT name
FROM city
WHERE name LIKE '%n';
```

←T→	name
☐ ✏ Edit ⧉ Copy ⊖ Delete	Berlin

- Fetching the name of cities whose name starts with 'P' and ends with 's'

```
SELECT name
FROM city
WHERE name LIKE 'p%'
or name LIKE '%s';
```

←T→	name
☐ ✏ Edit ⧉ Copy ⊖ Delete	Paris

- Fetching the name of cities that start with some letter followed by some (e.g.: _aris in Paris , __rmany, so on)
```
SELECT name
FROM city
WHERE name LIKE '__ris';
```

←T→			name
☐ 🖉 Edit ᴶ⬩ Copy ⊖ Delete			Paris

➢ Other operators

• Fetching the name of country whose area is greater than 300,000 km^2.

```
SELECT name
FROM country
WHERE area > 300000;
```

←T→			name
☐ 🖉 Edit ᴶ⬩ Copy ⊖ Delete			france
☐ 🖉 Edit ᴶ⬩ Copy ⊖ Delete			Germany

• Fetching the name of country whose population greater than 1,000,000

```
SELECT name
FROM country
WHERE population > 1000000;
```

←T→			name
☐ 🖉 Edit ᴶ⬩ Copy ⊖ Delete			france
☐ 🖉 Edit ᴶ⬩ Copy ⊖ Delete			Germany

• Fetching the name of cities whose population is between 2,000,000 and 10,000,000.

```
SELECT name
FROM city
WHERE population BETWEEN 2000000 and 10000000;
```

←T→			name
☐ 🖉 Edit ᴶ⬩ Copy ⊖ Delete			Paris
☐ 🖉 Edit ᴶ⬩ Copy ⊖ Delete			Berlin

- Fetching the name of cities whose rating is not null.

```
SELECT name
FROM city
WHERE rating is not null;
```

←T→				name
☐	🖉 Edit	⯒ Copy	⊖ Delete	Paris
☐	🖉 Edit	⯒ Copy	⊖ Delete	Berlin
☐	🖉 Edit	⯒ Copy	⊖ Delete	Warsaw

- fetching the name of cities with the help of IDs

```
SELECT name
FROM city
WHERE country_id IN (1,3);
```

←T→				name
☐	🖉 Edit	⯒ Copy	⊖ Delete	Paris

⦿ Querying multiple tables
➢ INNER JOIN
returns the matching values from both the tables

```
SELECT city.name, country.name from city
INNER JOIN country
ON city.country_id = country.id;
```

name	name
Paris	france
Berlin	Germany

➢ LEFT JOIN
returns all rows from the left table with the corresponding values from right table. if there's extra rows in left, and not in right then in right It will be null.

```
SELECT city.name, country.name
FROM city
```

```
LEFT JOIN country
ON city.country_id = country.id;
```

name	name
Paris	france
Berlin	Germany
Warsaw	*NULL*

- ➤ **RIGHT JOIN**

  Returns all rows from the right table with corresponding from the left table. If there's extra rows in right, and not in left then in left It will be null.

  ```
  SELECT city.name, country.name
  FROM city RIGHT JOIN country
  ON city.country_id = country.id;
  ```

name	name
Paris	france
Berlin	Germany
NULL	Iceland

- ➤ **FULL JOIN**

  returns all the values from the both the tables and if there's no value for the corresponding cell, it returns null.

  ```
  SELECT country.id,country.name,city.name
  FROM country
  LEFT JOIN city
  ON country.id = city.country_id

  UNION

  SELECT country.id,country.name,city.name
  FROM city
  LEFT JOIN country
  ON city.country_id = country.id;
  ```

id	name	name
1	france	Paris
2	Germany	Berlin
3	Iceland	*NULL*
NULL	*NULL*	Warsaw

➢ CROSS JOIN

```
SELECT city.name, country.name
FROM city
CROSS JOIN country;
```

name	name
Paris	france
Berlin	france
Warsaw	france
Paris	Germany
Berlin	Germany
Warsaw	Germany
Paris	Iceland
Berlin	Iceland
Warsaw	Iceland