SQL Cheatsheet

SQL

SQL (Structured Query Language) is a language used to talk to relational databases.

Sample Data

Locations where Olympics have been hosted:

olympic_hosts					
id	**city**	**year**	**season**	**continent**	**country_id**
1	Paris	2024	Summer	Europe	78
2	Beijing	2022	Winter	Asia	46
3	Tokyo	2020	Summer	Asia	109

Countries with their capitals:

countries		
c_id	**country**	**capital**
1	Abkhazia	Sukhumi
2	Afghanistan	Kabul

QUERIES

Retrieve all columns from a single table:

```
SELECT *
FROM olympic_hosts;
```

Retrieve specific columns only:

```
SELECT city, year, continent
FROM olympic_hosts;
```

ALIAS

Change the column name in the results:

```
SELECT city AS host_city, year
FROM olympic_hosts;
```

FILTERS

Retrieve all Olympics hosted in Australia:

```
SELECT *
FROM olympic_hosts
WHERE continent = 'Australia';
```

Retrieve all Olympics not hosted in Europe:

```
SELECT *
FROM olympic_hosts
WHERE continent <> 'Europe';
```

Retrieve 3 Olympics hosted in North America:

```
SELECT *
FROM olympic_hosts
WHERE continent = 'North America'
LIMIT 3;
```

Retrieve the Olympics hosted in either South America or Australia:

```
SELECT city, continent
FROM olympic_hosts
WHERE continent = 'South America'
  OR continent = 'Australia';
```

FILTERS (Numeric)

Retrieve the Olympics hosted in Europe after 1980:

```
SELECT *
FROM olympic_hosts
WHERE continent = 'Europe'
  AND year > 1980;
```

Retrieve the Olympics held between 2000 and 2015:

```
SELECT city, year
FROM olympic_hosts
WHERE year BETWEEN 2000 AND 2015;
```

FILTERS (Text)

Retrieve the Olympics hosted in Asia or Europe:

```
SELECT city, continent, year
FROM olympic_hosts
WHERE continent IN ('Asia',
  'Europe');
```

Retrieve the Olympic host cities starting with the letter S:

```
SELECT city, continent, year
FROM olympic_hosts
WHERE city LIKE 'S%';
```

Retrieve the Olympics not hosted in the Americas:

```
SELECT city, continent, year
FROM olympic_hosts
WHERE continent NOT LIKE
'%America';
```

AGGREGATE FUNCTIONS

Aggregate functions calculate a result using multiple records: COUNT(), SUM(), AVG(), MIN(), and MAX().

Retrieve the number of Olympics hosted in Asia:

```
SELECT COUNT(*)
FROM olympic_hosts
WHERE continent = 'Asia';
```

Retrieve the number of continents that have hosted the Olympics:

```
SELECT COUNT(DISTINCT continent)
FROM olympic_hosts;
```

Retrieve the average year that Australia hosted the Olympics:

```
SELECT AVG(year) AS avg_year
FROM olympic_hosts
WHERE continent = 'Australia';
```

Retrieve the average year that Asia hosted the Olympics with rounding:

```
SELECT ROUND(AVG(year))
FROM olympic_hosts
WHERE continent = 'Asia';
```

Retrieve the earliest and most recent years that the Olympics where hosted in North America:

```
SELECT MIN(year), MAX(year)
FROM olympic_hosts
WHERE continent = 'North America';
```

SORTING

Retrieve the Olympics hosted in Europe from oldest to most recent:

```
SELECT city, continent, year
FROM olympic_hosts
WHERE continent = 'Europe'
ORDER BY year ASC;
```

Retrieve the Olympics hosted in Asia from most recent to oldest:

```
SELECT city, continent, year
FROM olympic_hosts
WHERE continent = 'Asia'
ORDER BY year DESC;
```

GROUPING

Retrieve the number of Olympics hosted on each continent:

```
SELECT continent, COUNT(*)
FROM olympic_hosts
GROUP BY continent;
```

Retrieve the continents that have hosted more than 3 Olympics:

```
SELECT continent, COUNT(*)
FROM olympic_hosts
GROUP BY continent
HAVING COUNT(year) > 3;
```

DATABASE KEYS

The **primary key** is typically used to uniquely identify each record (i.e. row) in a table:

```
CREATE TABLE countries(
    c_id SERIAL PRIMARY KEY,
    country VARCHAR(255) NOT NULL,
    capital VARCHAR(255) NOT NULL
);
```

A **foreign key** establishes a relationship between tables:

```
CREATE TABLE olympics_hosts(
    id SERIAL PRIMARY KEY,
    city VARCHAR(255) NOT NULL,
    …
    country_id INT,
    CONSTRAINT fk_country FOREIGN
      KEY(country_id) REFERENCES
      countries(c_id)
);
```

JOINS

Join operations in SQL combine rows from two or more tables based on a related column.

INNER JOIN

Retrieve only the rows where there is a match in both tables:

```
SELECT olympic_hosts.id,
  olympic_hosts.city,
  olympic_hosts.year,
  countries.country
FROM olympic_hosts
JOIN countries
ON olympic_hosts.country_id =
  countries.c_id
WHERE olympic_hosts.continent =
  'North America';
```

id	city	year	Country
15	Atlanta	1996	United States
21	Los Angeles	1984	United States
24	Lake Placid	1980	United States

LEFT JOIN

Retrieve all rows from left table and matching rows from right table:

```
SELECT olympic_hosts.id,
  olympic_hosts.city,
  olympic_hosts.year,
  countries.country
FROM olympic_hosts
LEFT JOIN countries
ON olympic_hosts.country_id =
  countries.c_id
WHERE olympic_hosts.continent =
  'North America’;
```

id	city	year	Country
8	Vancouver	2010	NULL
12	Salt Lake City	2002	NULL
15	Atlanta	1996	United States

RIGHT JOIN

Retrieve all rows from the right table and the matching rows from the left table:

```
SELECT olympic_hosts.id,
  olympic_hosts.city,
  olympic_hosts.year,
  countries.country
FROM olympic_hosts
RIGHT JOIN countries
ON olympic_hosts.country_id =
  countries.c_id;
```

id	city	year	Country
...	...	...	...
53	Paris	1900	France
54	Athens	1896	Greece
NULL	NULL	NULL	Sri Lanka
NULL	NULL	NULL	New Zealand
...	...	...	...

FULL OUTER JOIN

Retrieve all rows where there is a match in either the left or right table:

```
SELECT olympic_hosts.id,
  olympic_hosts.city,
  olympic_hosts.year,
  countries.country
FROM olympic_hosts
FULL OUTER JOIN countries
ON olympic_hosts.country_id =
  countries.c_id;
```

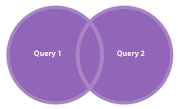
id	city	year	Country
...	...	...	...
8	Vancouver	2010	NULL
9	Beijing	2008	China
...	...	...	...
NULL	NULL	NULL	Sweden
NULL	NULL	NULL	Bermuda
...	...	...	...

SET OPERATORS

Set operations combine the results of two or more SELECT statements.

For all set operations, the number of selected columns and their respective data types must be identical.

UNION



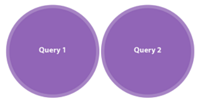
Combines the results of two or more SELECT statements without duplicates.

Retrieve all the cities that hosted the Olympics or are capitals:

```
SELECT olympic_hosts.city
FROM olympic_hosts
UNION
SELECT countries.capital
FROM countries;
```

city
Tokyo
Kuala Lampur
...

UNION ALL

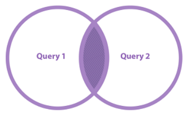


Keeps the duplicate values that UNION does not:

```
SELECT olympic_hosts.city
FROM olympic_hosts
UNION ALL
SELECT countries.capital
FROM countries;
```

city
Tokyo
Tokyo
Kuala Lampur
...

INTERSECT

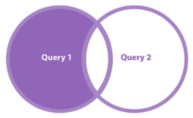


Retrieve the cities that have hosted the Olympics -AND- are capitals:

```
SELECT olympic_hosts.city
FROM olympic_hosts
INTERSECT
SELECT countries.capital
FROM countries;
```

city
Stockholm
Tokyo
Oslo
...

EXCEPT



Retrieve the cities that have hosted the Olympics that are not capitals:

```
SELECT olympic_hosts.city
FROM olympic_hosts
EXCEPT
SELECT countries.capital
FROM countries;
```

city
Sydney
St. Louis
...

SUB-QUERIES

SQL queries can be nested within another query.

Retrieve the cities that hosted the Olympics between 1960 and 1969 that are also the capitals of their countries:

```
SELECT c_id, country, capital
FROM countries
WHERE capital IN (
    SELECT city
    FROM olympic_hosts
    WHERE year BETWEEN 1960 AND
      1969
);
```

c_id	country	Capital
106	Italy	Rome
108	Japan	Tokyo
136	Mexico	Mexico City