SQL

SQL, или *язык структурированных запросов*, - это язык взаимодействия с базами данных. Он позволяет выбирать конкретные данные и создавать сложные отчеты. SQL является универсальным языком данных. Он используется практически во всех технологиях обработки данных.

ВЫБОРКА ДАННЫХ

COUNTRY

id	name	population	area
1	France	66600000	640680
2	Germany	80700000	357000
...	...	...	...

CITY

id	name	country_id	population	rating
1	Paris	1	2243000	5
2	Berlin	2	3460000	3
...	...	...	...	...

ЗАПРОС К ОДНОЙ ТАБЛИЦЕ

Выбрать все колонки из таблицы стран

```
SELECT *
FROM country;
```

Выбрать id и название из таблицы городов

```
SELECT id, name
FROM city;
```

Выбрать названия городов, отсортировав по столбцу рейтинга в порядке позрастания

```
SELECT name
FROM city
ORDER BY rating [ASC];
```

Выбрать названия городов, отсортировав по столбцу рейтинга в порядке убывания

```
SELECT name
FROM city
ORDER BY rating DESC;
```

ПСЕВДОНИМЫ

КОЛОНКИ

```
SELECT name AS city_name
FROM city;
```

ТАБЛИЦЫ

```
SELECT co.name, ci.name
FROM city AS ci
JOIN country AS co
  ON ci.country_id = co.id;
```

ФИЛЬТРАЦИЯ

ОПЕРАТОРЫ СРАВНЕНИЯ

Выбор названия городов, у которых рейтинг выше 3:

```
SELECT name
FROM city
WHERE rating > 3;
```

Выбор названия городов, которые не Берлин и не Мадрид

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

ТЕКСТОВЫЕ ОПЕРАТОРЫ

Выбор названия городов, которые начинаются на "P" или заканчиваются на "s":

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

Выбор названия городов, которые начинаются с любой буквы, за которой следует "ублин" (например, Дублин в Ирландии или Люблин в Польше):

```
SELECT name
FROM city
WHERE name LIKE '_ublin';
```

ДРУГИЕ ОПЕРАТОРЫ

Выбрать названия городов с населением от 500 тысяч до 5 миллионов человек:

```
SELECT name
FROM city
WHERE population BETWEEN 500000 AND 5000000;
```

Выбор названия городов, в которых рейтинг не null:

```
SELECT name
FROM city
WHERE rating IS NOT NULL;
```

Выбор названия городов, которые находятся в странах с id 1, 4, 7 или 8:

```
SELECT name
FROM city
WHERE country_id IN (1, 4, 7, 8);
```

ЗАПРОС ИЗ НЕСКОЛЬКИХ ТАБЛИЦ

INNER JOIN

JOIN (или именно **INNER JOIN**) возвращает строки, которые имеют совпадающие значения в обеих таблицах.

```
SELECT city.name, country.name
FROM city
[INNER] JOIN country
  ON city.country_id = country.id;
```

CITY			COUNTRY	
id	name	country_id	id	name
1	Paris	1	1	France
2	Berlin	2	2	Germany
3	Warsaw	4	3	Iceland

FULL JOIN

FULL JOIN (или именно **FULL OUTER JOIN**) возвращает все строки из обеих таблиц – если во второй таблице нет совпадающей строки, возвращаются нулевые значения.

```
SELECT city.name, country.name
FROM city
FULL [OUTER] JOIN country
  ON city.country_id = country.id;
```

CITY			COUNTRY	
id	name	country_id	id	name
1	Paris	1	1	France
2	Berlin	2	2	Germany
3	Warsaw	4	NULL	NULL
NULL	NULL	NULL	3	Iceland

LEFT JOIN

LEFT JOIN возвращает все строки из левой таблицы с соответствующими строками из правой таблицы. Если совпадающей строки нет, в качестве значений из второй таблицы возвращаются **NULL**.

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

CITY			COUNTRY	
id	name	country_id	id	name
1	Paris	1	1	France
2	Berlin	2	2	Germany
3	Warsaw	4	NULL	NULL

CROSS JOIN

CROSS JOIN возвращает все возможные комбинации строк из обеих таблиц. Доступно два синтаксиса.

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

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

CITY			COUNTRY	
id	name	country_id	id	name
1	Paris	1	1	France
1	Paris	1	2	Germany
2	Berlin	2	1	France
2	Berlin	2	2	Germany

RIGHT JOIN

RIGHT JOIN возвращает все строки из правой таблицы с соответствующими строками из левой таблицы. Если совпадающей строки нет, в качестве значений из левой таблицы возвращаются **NULL**.

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

CITY			COUNTRY	
id	name	country_id	id	name
1	Paris	1	1	France
2	Berlin	2	2	Germany
NULL	NULL	NULL	3	Iceland

NATURAL JOIN

NATURAL JOIN объединит таблицы по всем столбцам с одинаковым именем.

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

CITY			COUNTRY	
country_id	id	name	name	id
6	6	San Marino	San Marino	6
7	7	Vatican City	Vatican City	7
5	9	Greece	Greece	9
10	11	Monaco	Monaco	10

NATURAL JOIN использовало эти столбцы для сопастовления строк **city.id, city.name, country.id, country.name**

NATURAL JOIN очень редко использовалось на практике

АГРЕГИРОВАНИЕ И ГРУППИРОВКА

GROUP BY группирует вместе строки, которые имеют одинаковые значения в указанных столбцах. Происходит подсчёт количества таких значений.

CITY

id	name	country_id
1	Paris	1
101	Marseille	1
102	Lyon	1
2	Berlin	2
103	Hamburg	2
104	Munich	2
3	Warsaw	4
105	Cracow	4

→

CITY

country_id	count
1	3
2	3
4	2

АГРЕГАТНЫЕ ФУНКЦИИ

- **avg**(expr) – усредняет значение строк внутри группы
- **count**(expr) – подсчитывает значения строк внутри группы
- **max**(expr) – максимальное значение в группе
- **min**(expr) – минимальное значение в группе
- **sum**(expr) – сумма значений в группе

ПРИМЕРЫ ЗАПРОСОВ

Найти количество городов

```
SELECT COUNT(*)
FROM city;
```

Найти количество городов с ненулевыми рейтингами:

```
SELECT COUNT(rating)
FROM city;
```

Найти количество индивидуальных id стран:

```
SELECT COUNT(DISTINCT country_id)
FROM city;
```

Найти наибольшее и наименьшее значение популяции страны

```
SELECT MIN(population), MAX(population)
FROM country;
```

Найти общую численность населения городов в конкретных странах

```
SELECT country_id, SUM(population)
FROM city
GROUP BY country_id;
```

Найти ср. рейтинг городов в странах, у которых ср. показатель превышает 3,0:

```
SELECT country_id, AVG(rating)
FROM city
GROUP BY country_id
HAVING AVG(rating) > 3.0;
```

ПОДЗАПРОСЫ

Подзапрос - это запрос, который вложен в другой запрос или в другой подзапрос.
Существуют различные типы подзапросов.

ЕДИНСТВЕННОЕ ЗНАЧЕНИЕ

Простейший подзапрос возвращает ровно один столбец и одну строку. Его можно использовать с операторами сравнения =, <, <=, > или >=.

Этот запрос позволяет найти города с тем же рейтингом, что и Париж:

```
SELECT name FROM city
WHERE rating = (
    SELECT rating
    FROM city
    WHERE name = 'Paris'
);
```

НЕСКОЛЬКО ЗНАЧЕНИЙ

Подзапрос также может возвращать несколько столбцов или строк. Такие подзапросы могут использоваться с операторами IN, EXISTS, ALL или ANY.

Этот запрос позволяет найти города в странах с населением > 20 мил-ов человек:

```
SELECT name
FROM city
WHERE country_id IN (
    SELECT country_id
    FROM country
    WHERE population > 20000000
);
```

КОРРЕЛИРОВАННЫЙ ПОДЗАПРОС

Коррелированный подзапрос ссылается на таблицы, представленные во внешнем запросе. Коррелированный подзапрос зависит от внешнего запроса. Он не может выполняться независимо от внешнего запроса.

Коррелированный подзапрос ссылается на таблицы, представленные во внешнем запросе.

```
SELECT *
FROM city main_city
WHERE population > (
    SELECT AVG(population)
    FROM city average_city
    WHERE average_city.country_id = main_city.country_id
);
```

Этот запрос позволяет найти страны, в которых есть хотя бы один город:

```
SELECT name
FROM country
WHERE EXISTS (
    SELECT *
    FROM city
    WHERE country_id = country.id
);
```

SET ОПЕРАЦИИ

Операции Set используются для объединения результатов двух или более запросов в один результат. Объединенные запросы должны возвращать одинаковое количество столбцов и совместимые типы данных. Имена соответствующих столбцов могут отличаться.

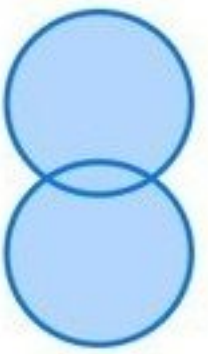
CYCLING		
id	name	country
1	YK	DE
2	ZG	DE
3	WT	PL
...	...	...

SKATING		
id	name	country
1	YK	DE
2	DF	DE
3	AK	PL
...	...	...

UNION (ОБЪЕДИНЕНИЕ)

UNION объединяет результаты двух результирующих наборов и удаляет дубликаты. UNION ALL не удаляет повторяющиеся строки. Этот запрос отображает немецких велосипедистов вместе с немецкими фигуристами:

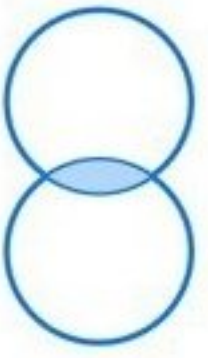
```
SELECT name
FROM cycling
WHERE country = 'DE'
UNION / UNION ALL
SELECT name
FROM skating
WHERE country = 'DE';
```



INTERSECT (ПЕРЕСЕЧЕНИЕ)

INTERSECT возвращает те строки, которые присутствуют в обоих наборах результатов. В этом запросе отображаются немецкие велосипедисты, которые являются и немецкими фигуристами:

```
SELECT name
FROM cycling
WHERE country = 'DE'
INTERSECT
SELECT name
FROM skating
WHERE country = 'DE';
```



EXCEPT (ИСКЛЮЧЕНИЕ)

UNION объединяет результаты двух результирующих наборов и удаляет дубликаты. UNION ALL не удаляет повторяющиеся строки.
Этот запрос отображает немецких велосипедистов вместе с немецкими фигуристами:

```
SELECT name
FROM cycling
WHERE country = 'DE'
EXCEPT / MINUS
SELECT name
FROM skating
WHERE country = 'DE';
```