



# SQL

СПбГУ 2020 Помыткина Т.Б.

# SQL

Structured Query Language

- язык структуризованных запросов

Это универсальный компьютерный язык

(но не "язык программирования"),

применяемый для создания, модификации и управления данными в реляционных базах данных.

SQL основывается на реляционной алгебре.

К началу 1980-х годов существовало несколько вариантов СУБД от разных производителей, и каждый из них обладал собственной реализацией языка запросов.

В 1983 году Международная организация по стандартизации (ISO) и Американский национальный институт стандартов (ANSI) приступили к разработке стандарта языка SQL.

## Стандарты SQL:

SQL1	/	ANSI 86	/	ANSI 89	/	SQL86	/	SQL87
SQL2	/	ANSI 92	/	ISO 92	/	FIPS 127-2	/	SQL 92
SQL3	/	SQL:1999						
SQL4	/	SQL:2003						
		SQL:2008						
		SQL:2011						
		SQL:2016						

$$\text{SQL} = \text{DDL} + \text{DML} + \text{DCL} + \text{TCL} + \dots$$

- операторы определения данных

Data Definition Language      DDL

- операторы манипуляции данными

Data Manipulation Language      DML

- операторы разграничения доступа к данным

Data Control Language      DCL

- операторы управления транзакциями

Transaction Control Language      TCL

DDL	Data Definition Language
DML	Data Manipulation Language

```
CREATE TABLE student (id_st INT, name VARCHAR, phone CHAR(9));
```

```
INSERT INTO student (id_st, name, phone)  
VALUES (1, 'Иванов', 1234567);
```

```
INSERT INTO student (id_st, name, phone)  
VALUES (2, 'Петров', 2345678);
```

```
SELECT name, phone FROM student;
```

```
DROP TABLE student;
```

# DML Data Manipulation Language

Добавление записи?

INSERT

Изменение записи?

UPDATE

Удаление записи?

DELETE

Выборка данных?

SELECT

# Оператор SELECT

```
SELECT [DISTINCT]
{{функция агрегирования.. | выражение для вычисления значения
[AS имя столбца] }.,...}
| { спецификатор,* }
| *
FROM { { имя таблицы [AS] [имя корреляции]
[(имя столбца.,...)]}
| { подзапрос [AS] имя корреляции
[ имя столбца.,...]}
| соединенная таблица
} .,...
[ WHERE предикат ]
GROUP BY { { [ имя таблицы |имя корреляции ].] имя столбца}.,...
[ HAVING предикат]
[ { UNION | INTERSECT | EXCEPT ) (ALL]
[ CORRESPONDING [ BY ( имя столбца.,...) ] ]
оператор select | {TABLE имя таблицы}
| конструктор значений таблицы]
[ ORDER BY { { столбец-результат [ ASC | DESC ]}.,...}
| { { положительное целое [ASC | DESC ] }.,...};
```

**SELECT [DISTINCT] *выражение*, ...**

**FROM *имя\_табл*, ...**

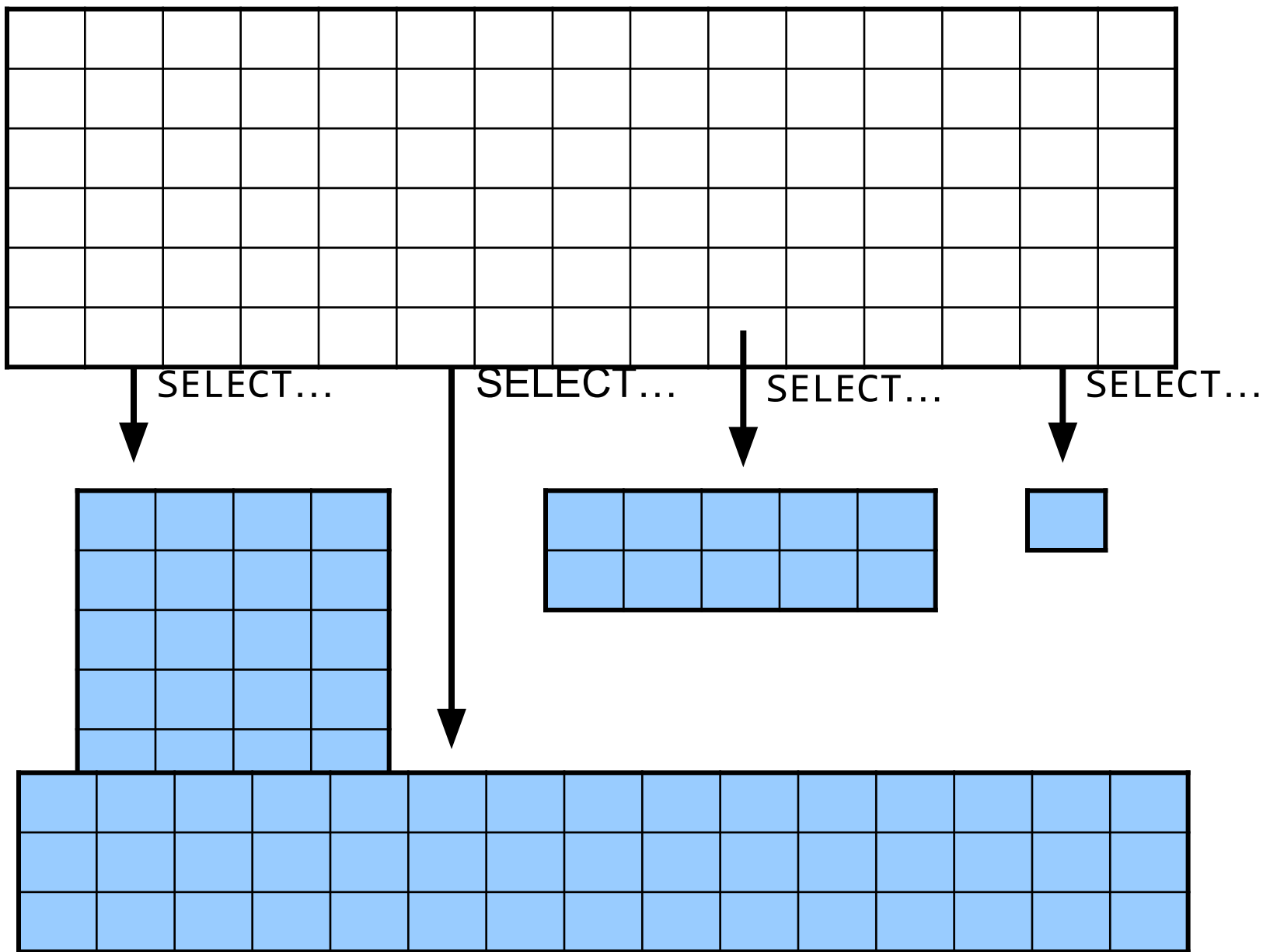
**[WHERE *условие*]**

**[GROUP BY *имя\_столбца* | *ном\_столбца*, ...]**

**[HAVING *условие*,]**

**[ORDER BY *имя\_столбца* | *ном\_столбца* [ASC|DESC],  
...];**





student

<i>id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>	<i>grant_st</i>
1	Иванов	123-45-67	м	700
2	Петрова	234-56-78	ж	700
3	Сидоров	345-67-89	м	700
4	Алексеев	456-78-90	м	1000
...	...	...	...	...



<i>name</i>
Иванов
Петрова
Сидоров
Алексеев
...



<i>id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>
1	Иванов	123-45-67	м
2	Петрова	234-56-78	ж



<i>phone</i>
234-56-78

student

<i>Id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>	<i>grant_st</i>
1	Иванов	123-45-67	м	700
2	Петрова	234-56-78	ж	700
3	Сидоров	345-67-89	м	700
4	Алексеев	456-78-90	м	1000
...	...	...	...	...



SELECT name FROM student;



SELECT name, phone FROM student;



SELECT \* FROM student;

<i>name</i>	<i>name</i>	<i>phone</i>	<i>Id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>	<i>grant_st</i>
Иванов	Иван		1	Иванов	123-45-67	м	700
Петров	Петр		2	Петрова	234-56-78	ж	700
Сидоро	Сидор		3	Сидоров	345-67-89	м	700
Алексее	Алекс		4	Алексеев	456-78-90	м	1000
...	...		...	...	...	...	...

student

<i>Id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>	<i>grant_st</i>
1	Иванов	123-45-67	М	700
2	Петрова	234-56-78	Ж	700
3	Сидоров	345-67-89	М	700
4	Алексеев	456-78-90	М	1000
...	...	...	...	...

SELECT sex FROM student;

↓

SELECT ALL sex FROM student;

<i>sex</i>
М
Ж
М
М
...

SELECT DISTINCT sex FROM student;

<i>sex</i>
М
Ж

SELECT DISTINCT sex, grant\_st  
FROM student;

<i>sex</i>	<i>grant_st</i>
М	700
Ж	700
М	1000
...	

student

<i>Id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>	<i>grant_st</i>
1	Иванов	123-45-67	м	700
2	Петрова	234-56-78	ж	700
3	Сидоров	345-67-89	м	700
4	Алексеев	456-78-90	м	1000
...	...	...	...	...

SELECT grant\_st - 100 FROM student;

<i>grant_st</i>
600
600
600
900
...

SELECT DISTINCT grant\_st - 100 FROM student;

<i>grant_st</i>
600
900

student

<i>Id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>	<i>grant_st</i>
1	Иванов	123-45-67	м	700
2	Петрова	234-56-78	ж	700
3	Сидоров	345-67-89	м	700
4	Алексеев	456-78-90	м	1000
...	...	...	...	...

SELECT name, 2000 FROM student;

<i>name</i>	<i>2000</i>
Иванов	2000
Петрова	2000
Сидоров	2000
Алексеев	2000
...	...

SELECT name, 2000 as grants  
FROM student;

<i>name</i>	<i>grants</i>
Иванов	2000
Петрова	2000
...	...

## Функции агрегирования:

COUNT(*)	количество записей
COUNT( DISTINCT expression)	количество различных записей
AVG([ DISTINCT ] expression)	среднее значение [среди различных]
MAX(expression)	максимальное значение
MIN(expression)	минимальное значение
SUM([ DISTINCT ] expression)	сумма значений [различных]

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

340

```
SELECT COUNT(DISTINCT sex) FROM student;
```

2

```
SELECT AVG(grant_st) FROM student;
```

775

```
SELECT AVG(DISTINCT grant_st ) FROM student;
```

850

```
SELECT MAX(grant_st) FROM student;
```

1000

## Фильтры:

### 1. *expression* (=, >, <, AND, OR)

```
SELECT * FROM student WHERE name = 'Величко' AND sex='м';
```

*expression* [ NOT ] BETWEEN *expression* AND *expression*

```
SELECT name FROM student WHERE id_st NOT BETWEEN 3 AND 6;
```

### 2. *field-name* IS [ NOT ] NULL

```
SELECT name FROM student WHERE phone IS NULL;
```

### 3. *field-name* [ NOT ] LIKE 'string' [ ESCAPE 'character' ]

```
SELECT name, phone FROM student WHERE phone LIKE '_45%';
```

### 4. *expression* [ NOT ] IN ( *value-list* | *SELECT-statement* )

```
SELECT name FROM student WHERE grant_st - 100 IN (900, 1400);
```



student

<i>Id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>	<i>grant_st</i>
1	Иванов	123-45-67	м	700
2	Петрова	234-56-78	ж	700
3	Сидоров	345-67-89	м	700
4	Алексеев	456-78-90	м	1000
...	...	...	...	...

```
SELECT * FROM student WHERE name = 'Иванов' OR name = 'Попова';
```

```
SELECT * FROM student WHERE name = 'Попова'  
      AND (phone = '111-11-67' OR phone = '+7 911 1111167');
```

```
SELECT * FROM student WHERE phone IS NULL AND sex = 'ж';
```

```
SELECT name, phone FROM student WHERE phone NOT LIKE '+7%';
```

```
SELECT name FROM student WHERE phone  
      NOT IN ('111-11-11', '333-33-33', '555-55-55');
```

## Замечание о NULL:

- NULL - это отсутствие значения
- NULL не принадлежит ни одному типу данных
- NULL не равно ни FALSE, ни пустой строке, ни нулю
- сравнение NULL с любым значением даст NULL
- NULL не равно NULL

Сравним:

```
SELECT name, phone FROM student WHERE phone IS NULL;
```

```
SELECT name, phone FROM student WHERE phone = NULL;
```

!!! Существуют специфические СУБД, в которых NULL м.б. равно NULL.

# Сортировки:

```
SELECT * FROM student  
ORDER BY sex;
```

<i>id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>
2	Петрова	234-56-78	ж
4	Павлова	456-44-55	ж
1	Иванов	123-45-67	м
3	Сидоров	345-67-89	м
5	Сачков	333-66-77	м

```
SELECT * FROM student  
ORDER BY sex, name;
```

<i>id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>
1	Иванов	123-45-67	м
5	Сачков	333-66-77	м
3	Сидоров	345-67-89	м
4	Павлова	456-44-55	ж
2	Петрова	234-56-78	ж

<i>id_st</i>	<i>name</i>	<i>phone</i>	<i>sex</i>
4	Павлова	456-44-55	ж
2	Петрова	234-56-78	ж
1	Иванов	123-45-67	м
5	Сачков	333-66-77	м
3	Сидоров	345-67-89	м

```
SELECT * FROM student  
ORDER BY sex DESC, name;
```

## Напишите следующие запросы к таблице Pet:

1. Данные на Партизана.
2. Клички и породы всех питомцев с сортировкой по возрасту.
3. Питомцы, имеющие хоть какое-нибудь описание.
4. Средний возраст пуделей.
5. Количество владельцев.

Table - dbo.Pet		Summary					
	Pet_ID	Nick	Breed	Age	Description	Pet_Type_ID	Owner_ID
	1	Bobik	unknown	3		1	1
	2	Musia		12		2	1
	3	Katok		2	crazy guy	2	1
	4	Apelsin	poodle	5		1	2
	5	Partizan	rottweiler	5	very big	2	2
	6	Daniel	spaniel	14		1	3
	7	Model		5		3	4
	8	Markiz	poodle	1		1	5
	9	Zombi	Siamese	7	wild	2	6