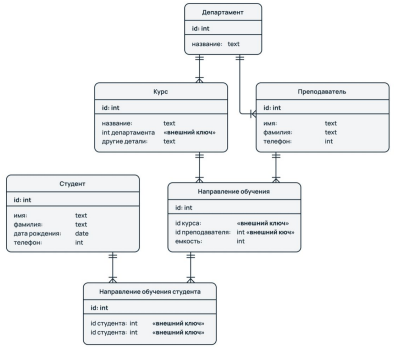
SQL. Теория

Соболева
Татьяна

Введение

База данных — это набор сведений об объектах, структурированный определенным образом.

Одним из наиболее часто встречающихся видов БД являются реляционные БД, в которых взаимоотношения объектов моделируются с помощью внешних ключей. В каждой таблице БД необходимо наличие **первичного ключа** – так именуют поле или набор полей, однозначно идентифицирующий каждый экземпляр объекта или запись



Структура запроса

Классический запрос состоит из 6 операторов, 4 из которых необязательны, но используются часто.

SELECT — выбирает отдельные столбцы или всю таблицу целиком (обязательный);

FROM — из какой таблицы получить данные (обязательный);

WHERE — условие, по которому SQL выбирает данные;

GROUP BY — столбец, по которому мы будут группироваться данные;

HAVING — условие, по которому сгруппированные данные будут отфильтрованы;

ORDER BY — столбец, по которому данные будут отсортированы;

```
SELECT
   id
FROM
   persons
where gender = 'M'
group by 1
```

Создание таблиц

Обычно у нас уже есть база данных, в которой можно создать таблицу. Таблица создается с помощью команды **create table** , однако, если планируете ее пересоздавать, то придется ее удалять перед созданием. При создании необходимо указывать типы данных.

```
Drop table if exists persons;
Create table persons as (
    name varchar(100),
    income integer,
    gender varchar(100)
);
```

Новые колонки можно добавлять с помощью команды **alter table**

```
ALTER TABLE persons
ADD age integer;
```

Создание таблиц

Также можно добавлять записи в таблицу с помощью **insert**

```
INSERT INTO persons
VALUES
('Андрей',100000,'M',40);
```

И удалять ненужные строки с помощью **delete**

Агрегация

Агрегация относится к таким операциям, когда больший набор строк свёртывается в меньший. Типичные агрегатные функции - COUNT, MIN, MAX, SUM и AVG.

Функция **sum** возвращает сумму значений заданного поля, а функция **count** - общее число записей

```
SELECT client_id, count(distinct transaction_id)
FROM
   transaction
GROUP BY 1;
```

Where, having true

В **HAVING** и только в нём можно писать условия по агрегатным функциям (SUM, COUNT, MAX, MIN и т. д.).

Главное отличие **HAVING** от **WHERE** в том, что в **HAVING** можно наложить условия на результаты группировки, потому что порядок исполнения запроса устроен таким образом, что на этапе, когда выполняется **WHERE**, ещё нет групп, а **HAVING** выполняется уже после формирования групп.

```
SELECT client_id, count(distinct transaction_id)
FROM
   transaction
WHERE
   type = 'pay'
GROUP BY 1
HAVING count(distinct transaction_id) > 10;
```

Оконные функции

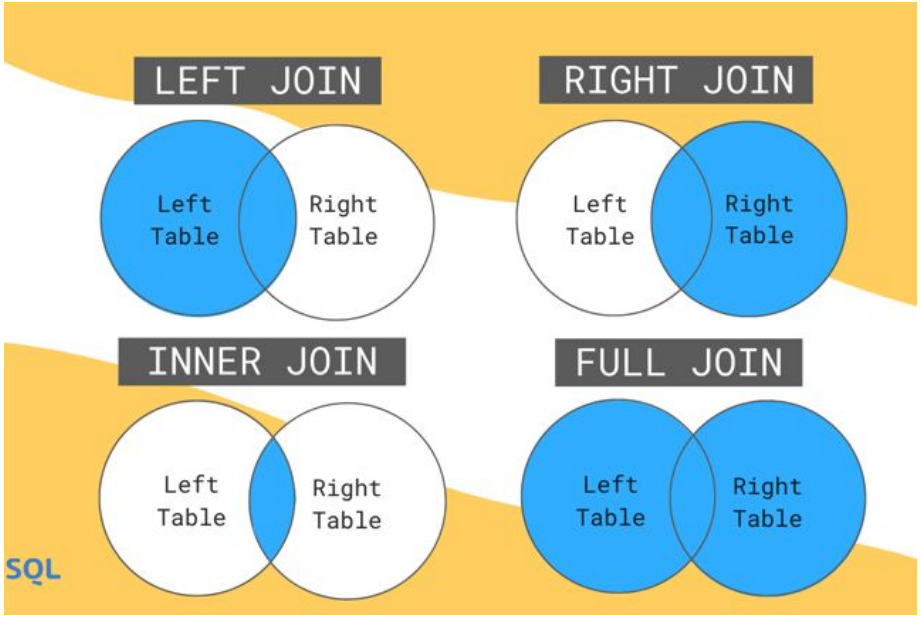
Оконная функция в SQL - функция, которая работает с выделенным набором строк (окном, партицией) и выполняет вычисление для этого набора строк в отдельном столбце. Оконные функции позволяют не сокращают количество строк в запросе, а позволяют считать все в отдельной колонке.

Партиции (окна из набора строк) - это набор строк, указанный для оконной функции по одному из столбцов или группе столбцов таблицы.

```
SELECT name, subject, grade,
sum(grade) over (partition by name) as sum_grade,
avg(grade) over (partition by name) as avg_grade,
count(grade) over (partition by name) as count_grade,
min(grade) over (partition by name) as min_grade,
max(grade) over (partition by name) as max_grade
FROM student_grades;
```

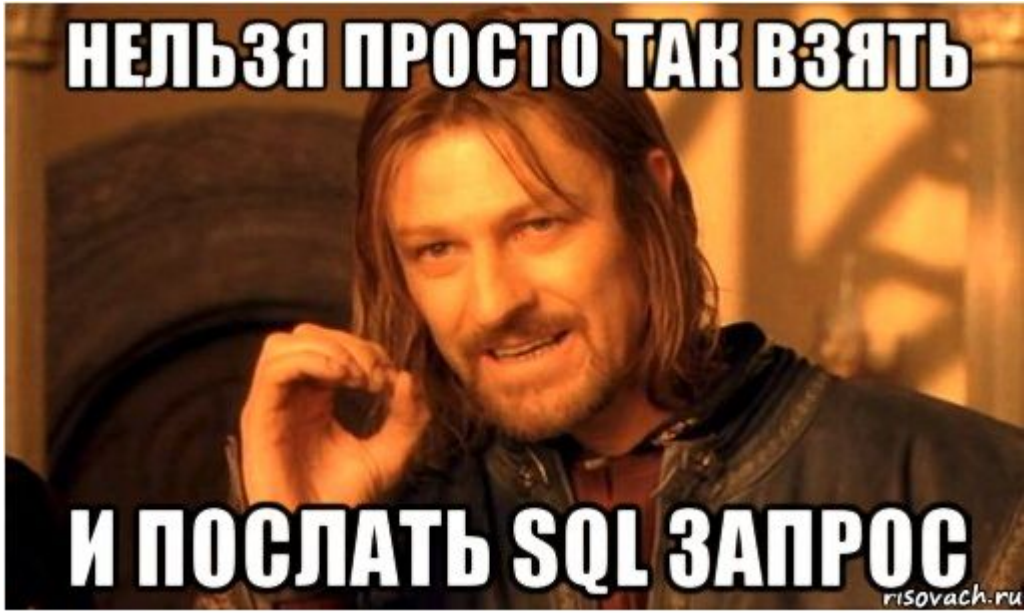
Join

Join — оператор для объединения данных из нескольких таблиц с общим ключом.




```
SELECT *
FROM products as t1
INNER JOIN brand_popularity as t2
ON t1.product=t2.brand
```

Спасибо за внимание!



Путь аналитика не ждет