SELECT (Part 2)

GROUP BY

- Когда агрегатная функция присутствует без полей, то подразумевается группа из всех данных таблицы:
- SELECT count(*) FROM contact;

- Для указания некоторой группы используется оператор GROUP BY

- SELECT count(*) FROM contact GROUP BY first_name;
- SELECT first_name, count(*) FROM contact GROUP BY first_name;
- SELECT first_name, age, count(*) FROM contact GROUP BY first_name, age;

- ***При использовании любой агрегатной функции и нескольких полей в SELECT все эти поля должны быть перечислены в секции GROUP BY!***

HAVING

- Если нужно отфильтровать выборку по результату групповой функции, оператор WHERE в таком случае не применим, так как он умеет фильтровать по записям до группировки строк в группы, а фильтровать уже нужно по группе, т.е. данный запрос не корректный:

- SELECT first_name, count(*) FROM contact WHERE count(*) > 1 GROUP BY first_name;

- Для решения данной задачи нужно использовать оператор HAVING

- SELECT first_name, count(*) FROM contact GROUP BY first_name **HAVING** count(*) > 1;
- SELECT first_name, count(*) FROM contact GROUP BY first_name **HAVING** first_name LIKE 'И%';
- SELECT first_name, count(*) FROM contact **WHERE** first_name LIKE 'И%' GROUP BY first_name;

ORDER BY

- SELECT * FROM contact;
- SELECT * FROM contact **ORDER BY** last_name;
- SELECT * FROM contact **ORDER BY** last_name **ASC**;
- SELECT * FROM contact **ORDER BY** last_name **DESC**;
- SELECT * FROM contact **ORDER BY** last_name, first_name, middle_name;
- SELECT * FROM contact **ORDER BY** last_name, first_name **DESC**, middle_name **DESC**;
- SELECT first_name, count(*) as **cnt** FROM contact GROUP BY first_name **ORDER BY cnt DESC**;
- SELECT * FROM contact **ORDER BY** middle_name;
- SELECT * FROM contact **ORDER BY** middle_name **NULLS FIRST**;
- SELECT * FROM contact **ORDER BY** middle_name **NULLS LAST**;

LIMIT / OFFSET

- Для ограничения вывода SELECT выборки используется оператор LIMIT, для задания отступа – оператор OFFSET. (Эти операторы не входят в стандарт SQL и являются специфичными для PostgreSQL сервера СУБД).
- Именно эти операторы нужно использовать для реализации постраничного отображения данных.

- SELECT * FROM contact LIMIT 2;
- SELECT * FROM contact LIMIT 2 OFFSET 2;

- SELECT * from T WHERE **ROWNUM <= 10 (Oracle)**
- SELECT **TOP 10** * FROM T ORDER BY col (**MS SQL Server**)

Декартово произведение

- SELECT * FROM contact c, contact_value cv, category cat, contact_type ct;

- **SELECT** cat.name as category, c.first_name, c.middle_name, c.last_name, c.age, ct.protocol, ct.name as contact_type, cv.value **FROM** contact c, contact_value cv, category cat, contact_type ct **WHERE** c.id_category=cat.id AND cv.id_contact = c.id AND cv.id_contact_type = ct.id;

category	first_name	middle_name	last_name	age	protocol	contact_type	value
Друзья	Иван	Иванович	Иванов	20	tel:	Mobile Phone	+38(050) 123-45-67
Друзья	Иван	Иванович	Иванов	20	mailto:	Email	ivanov@gmail.com
Друзья	Иван	Иванович	Иванов	20	skype:	Skype	ivan.ivanov
Друзья	Петр	Null	Петров	18	tel:	Mobile Phone	+38(066) 159-35-87
Коллеги	Ирина	Викторовна	Николаева	22	tel:	Phone	365-69-69
Коллеги	Ирина	Викторовна	Николаева	22	tel:	Mobile Phone	+38(063) 122-22-33
Коллеги	Ирина	Викторовна	Николаева	22	skype:	Skype	irina.nikolaeva

- SELECT c1.id, c2.id FROM **contact** c1, **contact** c2 WHERE c1.first_name = c2.first_name AND c1.age <> c2.age

Порядок выполнения

```
SELECT
  cat.name as category, c.age, count(*) as cnt
  FROM contact c, contact_value cv, category cat, contact_type ct
  WHERE c.id_category=cat.id AND cv.id_contact = c.id
  GROUP BY category, age
  HAVING count(*) > 4
  ORDER BY cnt ASC, category DESC
  LIMIT 1
  OFFSET 1;
```

Порядок следования в операторе SELECT важен! Именно в таком порядке выполняются обработка команд.
P.S. Теперь понятно почему WHERE не работает с агрегатными функциями!

https://www.postgresql.org/docs/9.5/static/sql-select.html

Выводы

1. **SELECT** — оператор DML языка SQL, возвращающий набор данных (выборку) из базы данных, удовлетворяющих заданному условию;
2. Оператор SELECT позволяет реализовать *почти* все операции реляционной алгебры: Объединение, Пересечение, Вычитание, **Декартово произведение, Выборка (ограничение), Проекция, Соединение** (Переименование атрибутов → ALTER);
3. Между SELECT и FROM указываются проекция столбцов, которые необходимо вывести в запросе. * - означает все столбцы;
4. Оператор DISTINCT удаляет дублирующие строки из выборки;
5. Между SELECT и FROM можно указать динамические столбцы, константы и выражения;
6. С помощью оператора AS можно задать псевдоним для столбца или таблицы;
7. Для реализации операций объединение, пересечение, вычитание используются соответственно операторы UNION, INTERSECT, EXCEPT;
8. Оператор WHERE реализует операцию выборка и позволяет указать правило по которому следует выбрать строки из всего множества, т.е. правила фильтрации строк;
9. Агрегатные функции позволяют выполнять операции агрегации над данными таблицы;
10. Оператор GROUP BY позволяет добавить новые границы группы по указанному полю для агрегатных функций;
11. Оператор HAVING позволяет фильтровать выборку по результату агрегатной функции (Оператор WHERE не пригоден для этого);
12. Оператор ORDER BY задает порядок сортировки для выборки;
13. Оператор LIMIT ограничивает размер выводимой информации;
14. Оператор OFFSET задает начало вывода информации;
15. Порядок следования в операторе SELECT важен! Именно в таком порядке выполняются обработка команд