SQL
Занятие 3
Объединение таблиц
JOIN’ы, Множества

Бояр Владислав

Центр непрерывного образования ФКН ВШЭ 2023

JOIN

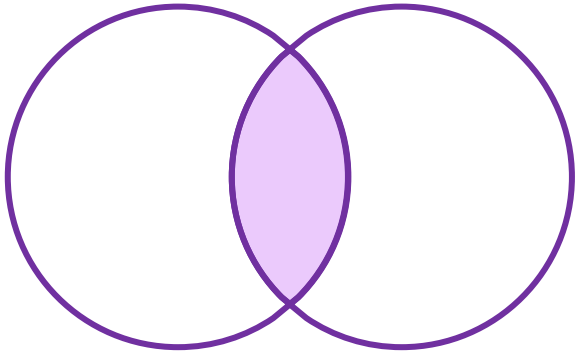
JOIN – оператор sql, позволяющий объединять данные из нескольких таблиц по заданному условию.

Общий синтаксис:

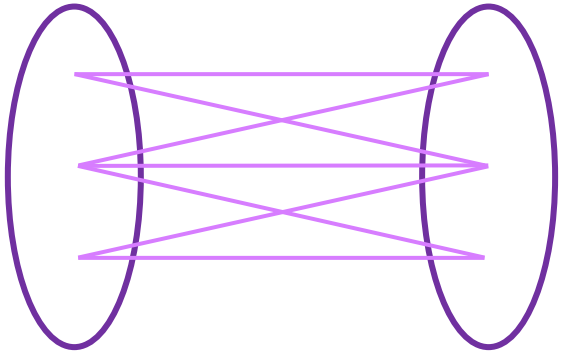
```
SELECT *
  FROM table_1 AS t1                  -- первая таблица для объединения
  JOIN                                -- тип объединения
       table_2 AS t2                  -- вторая таблица для объединения
    ON t1.column_1 = t2.column_2      -- условие объединения
```

- Вместо таблиц можно указывать подзапросы;
- Условий может быть несколько (AND, OR);
- Вместо ON можно использовать USING, когда условие равенства полей с одинаковыми именами.

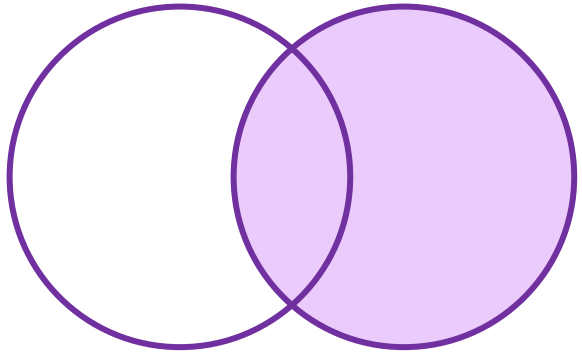
Типы JOIN



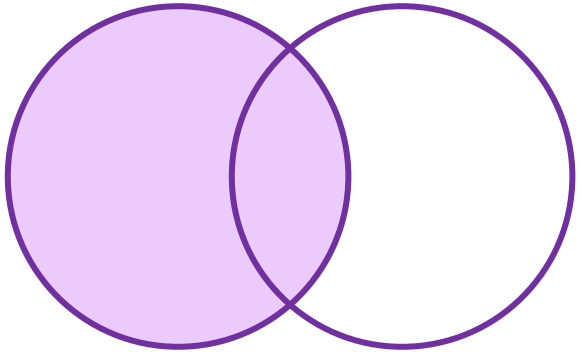
INNER JOIN



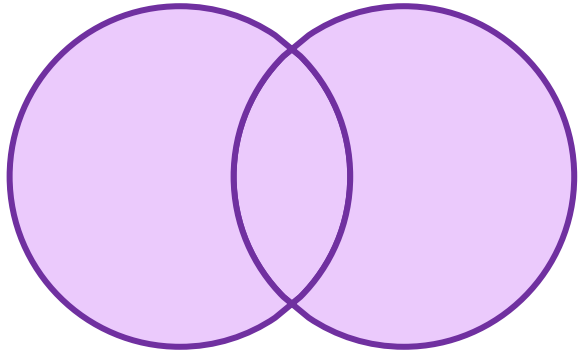
CROSS JOIN



RIGHT JOIN



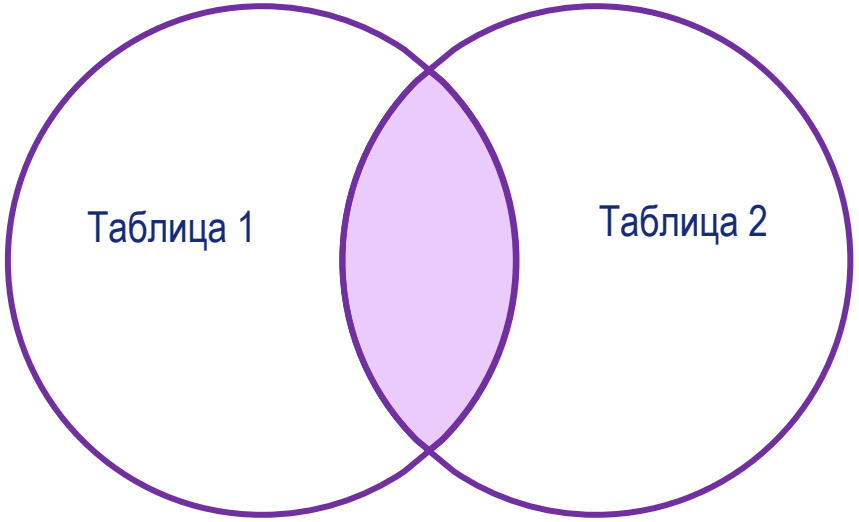
LEFT JOIN



FULL JOIN

INNER JOIN

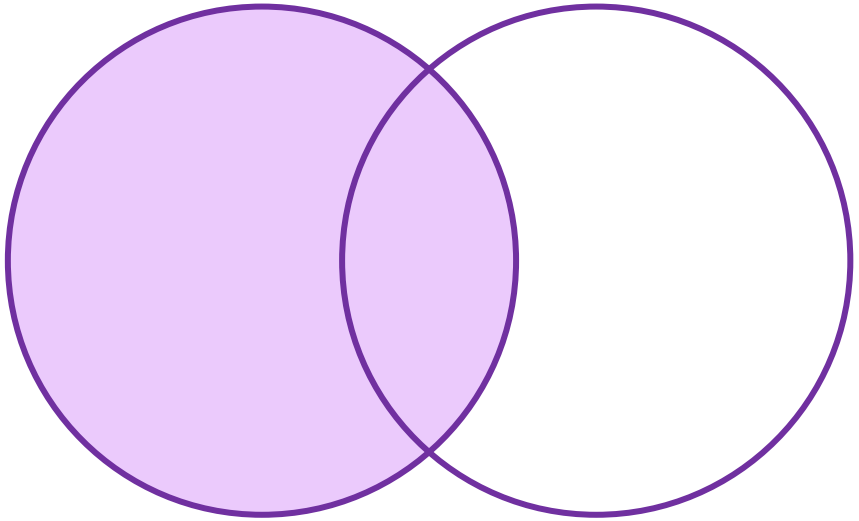
В результате объединения останутся только те пары строк объединяемых таблиц, для которых справедливо условие объединения.
Если одной строке из таблицы 1 по условию объединения соответствует N строк из таблицы 2, то в результате будет выведено N строк для каждого подобного случая.




```
SELECT *
  FROM table_1 AS t1
 INNER JOIN
       table_2 AS t2
    ON t1.column_1 = t2.column_2
```

LEFT JOIN

1. Сначала выполняется внутреннее соединение (INNER JOIN).
2. Затем в результат добавляются все строки из таблицы 1, которым не соответствуют никакие строки в таблице 2, а вместо значений полей таблицы 2 вставляются NULL.



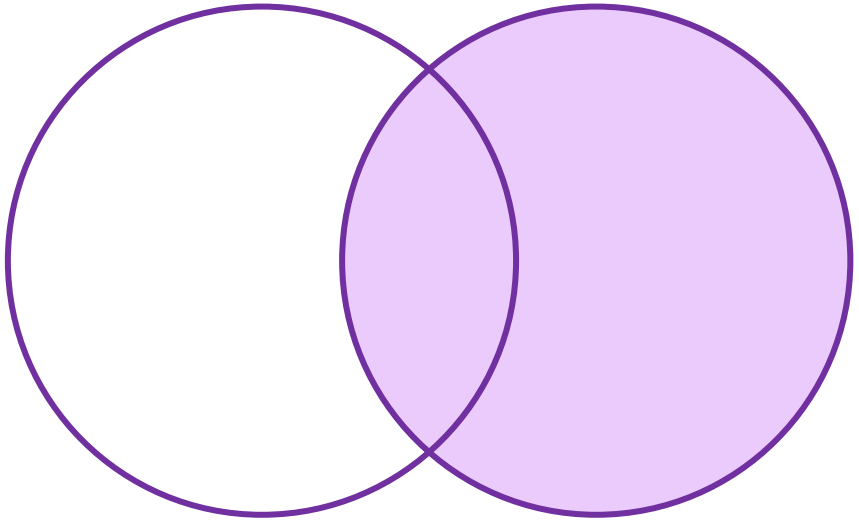
В результате объединения останутся все строки из левой таблицы и соответствующие условию объединения строки из правой таблицы. Если в правой таблице нет соответствующих значений, то будут возвращены NULL значения.

```
SELECT *
  FROM table_1 AS t1
  LEFT JOIN
       table_2 AS t2
    ON t1.column_1 = t2.column_2
```

RIGHT JOIN

Аналог левого объединения.

1. Сначала выполняется внутреннее соединение (INNER JOIN).
2. Затем в результат добавляются все строки из таблицы 2, которым не соответствуют никакие строки в таблице 1, а вместо значений полей таблицы 1 вставляются NULL.

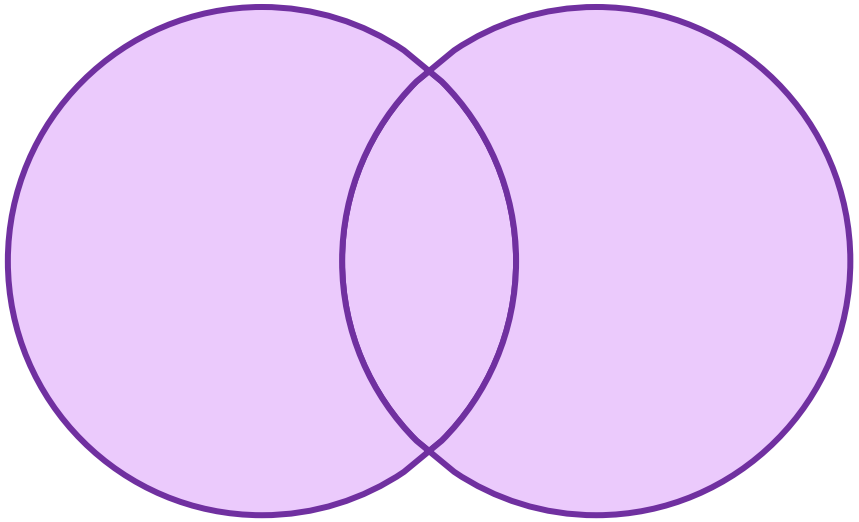


В результате объединения останутся все строки из правой таблицы и соответствующие условию объединения строки из левой таблицы. Если в левой таблице нет соответствующих значений, то будут возвращены NULL значения.

```
SELECT *
  FROM table_1 AS t1
 RIGHT JOIN
       table_2 AS t2
    ON t1.column_1 = t2.column_2
```

FULL JOIN

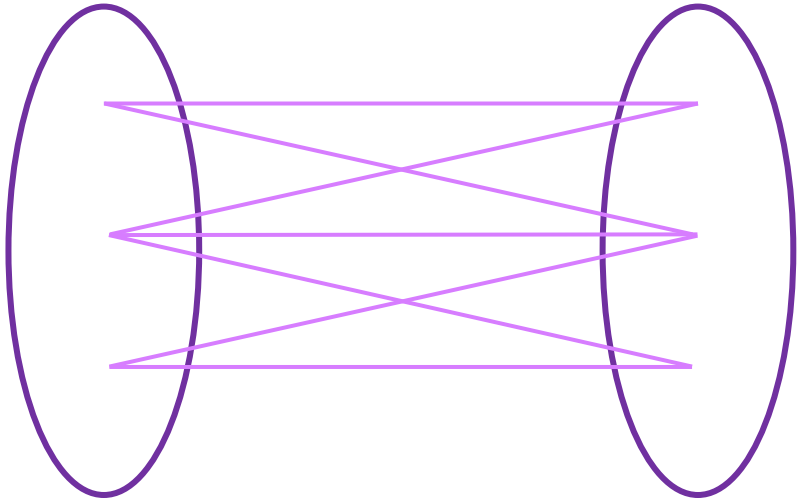
1. Сначала выполняется внутреннее соединение (INNER JOIN).
2. Затем в результат добавляются все строки из таблицы 1, которым не соответствуют никакие строки в таблице 2, а вместо значений полей таблицы 2 вставляются NULL.
3. И наконец, в результат включаются все строки из таблицы 2, которым не соответствуют никакие строки в T1, а вместо значений полей T1 вставляются NULL.



```
SELECT *
  FROM table_1 AS t1
  FULL JOIN
       table_2 AS t2
    ON t1.column_1 = t2.column_2
```

CROSS JOIN

- Возвращает декартово произведение строк из обеих таблиц.
- Другими словами, происходит соединение каждой строки первой таблицы с каждой строкой второй таблицы.
- Для этого типа объединения не нужно указывать условия.
- Если исходные таблицы содержали M и N строк, то результирующая будет содержать M * N строк.



```sql
SELECT *
  FROM table_1 AS t1
  CROSS JOIN
       table_2 AS t2
```

```sql
-- ИЛИ

SELECT *
  FROM table_1, table_2
```

Операции над множествами

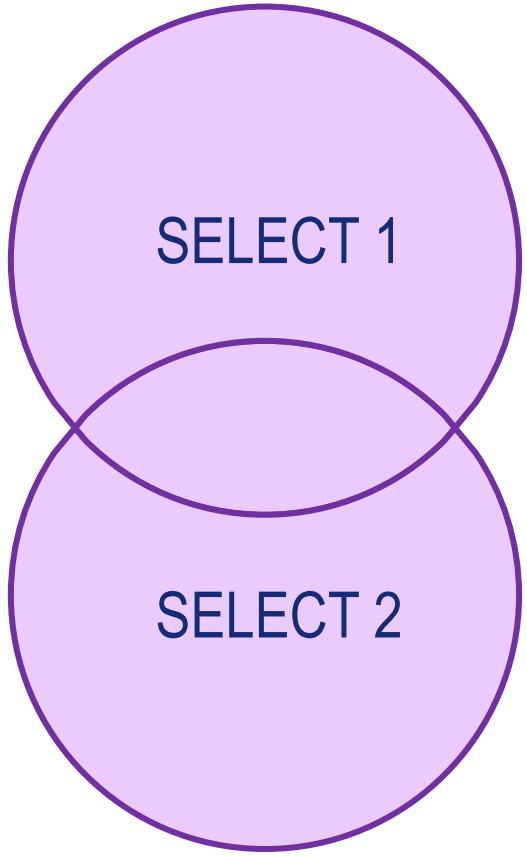
Результаты двух запросов можно обработать, используя операции над множествами:

- UNION (объединение);
- INTERSECT (пересечение);
- EXCEPT (вычитание).

UNION ALL (объединение)

- Объединяет таблицы по вертикали;
- Работает без условий объединения;
- Кол-во колонок в объединяемых таблицах должно обязательно совпадать;
- Типы данных объединяемых полей должны быть совместимы (varchar ~ text).
- Полезно, когда необходимо собрать информацию из различных источников;

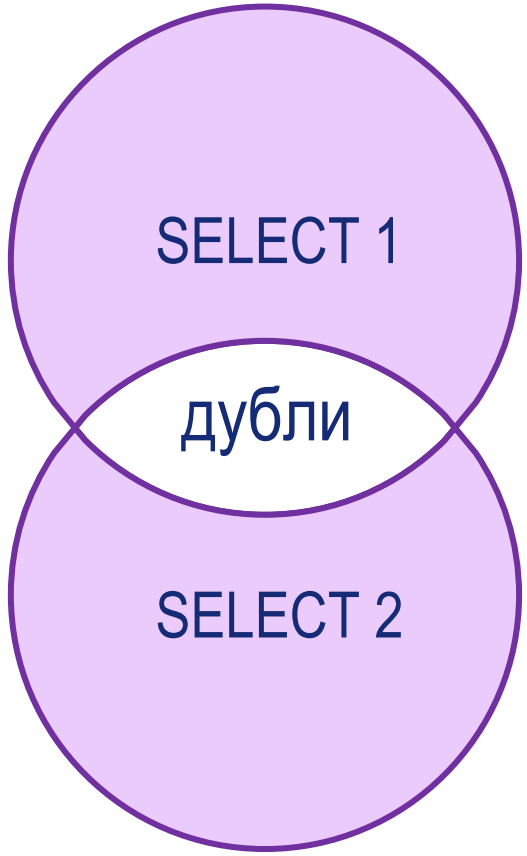
```
SELECT ...
 UNION ALL
SELECT ...
```



UNION (объединение)

- Объединяет таблицы по вертикали, при этом выполняется операция DISTINCT - произойдёт удаление дублей;
- После дедубликации останется одна из одинаковых строк;
- Значительно «тяжелее» операции UNION ALL из-за выполнения DISTINCT;
- Рекомендуется использовать только если предполагается наличие дублей в данных и их необходимо удалить на этапе объединения.

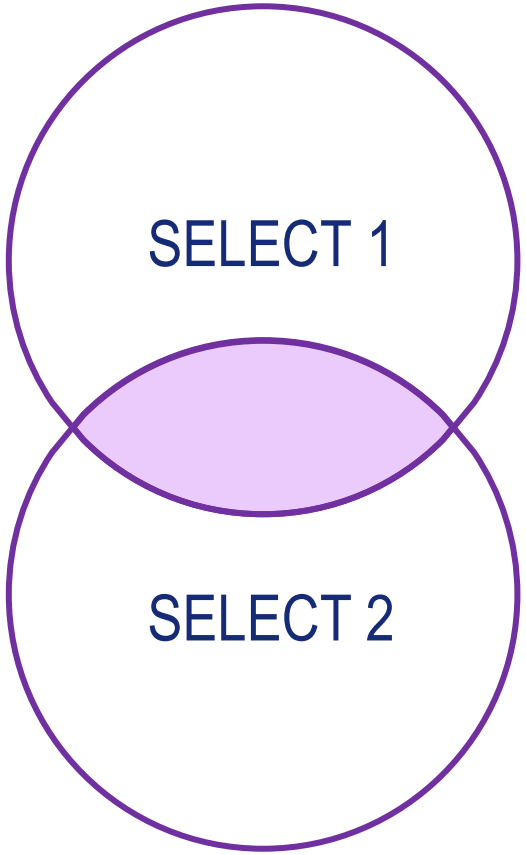
```
SELECT ...
 UNION
SELECT ...
```



INTERSECT (пересечение)

- Возвращает только общие строки, которые присутствовали в обоих SELECT-запросах.

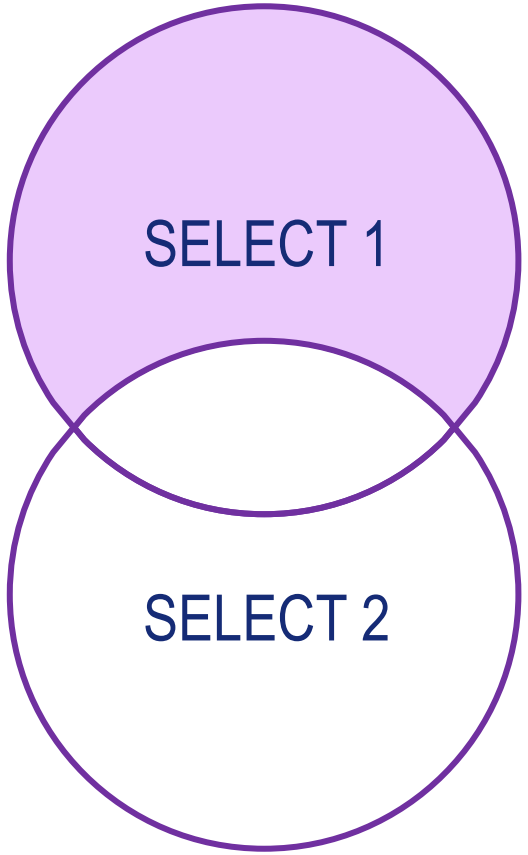
```
SELECT ...
INTERSECT [ALL]
SELECT ...
```



EXCEPT (вычитание)

- Возвращает уникальные строки, которые присутствуют в первом SELECT-запросе, но отсутствуют во втором.

```
SELECT ...
EXCEPT [ALL]
SELECT ...
```



Практика

Практика

1. Вывести идентификаторы перелётов за апрель 2017 года с указанием города вылета;
2. Вывести идентификаторы перелётов за апрель 2017 года с указанием города вылета и прибытия на русском языке;
3. Вывести: идентификатор и номер полёта, название аэропорта и город вылета, планируемое время вылета, код самолёта и его модель, номера билетов и информацию о пассажирах для каждого билета: имя пассажира, email, номер телефона, стоимость билета, класс перелёта и номер места;
4. Вывести такие же поля для первых пяти рейсов в мае 2017 года, в которых присутствовал класс Business;
5. Вывести список рейсов с информацией о модели самолета и количестве проданных билетов за 2016 год;
6. Вывести имена 10 пассажиров, которые чаще остальных летали из Москвы в Новосибирск в 2016 году, у которых заполнен адрес электронной почты и номер телефона;
7. Вывести день недели вылета для всех рейсов.