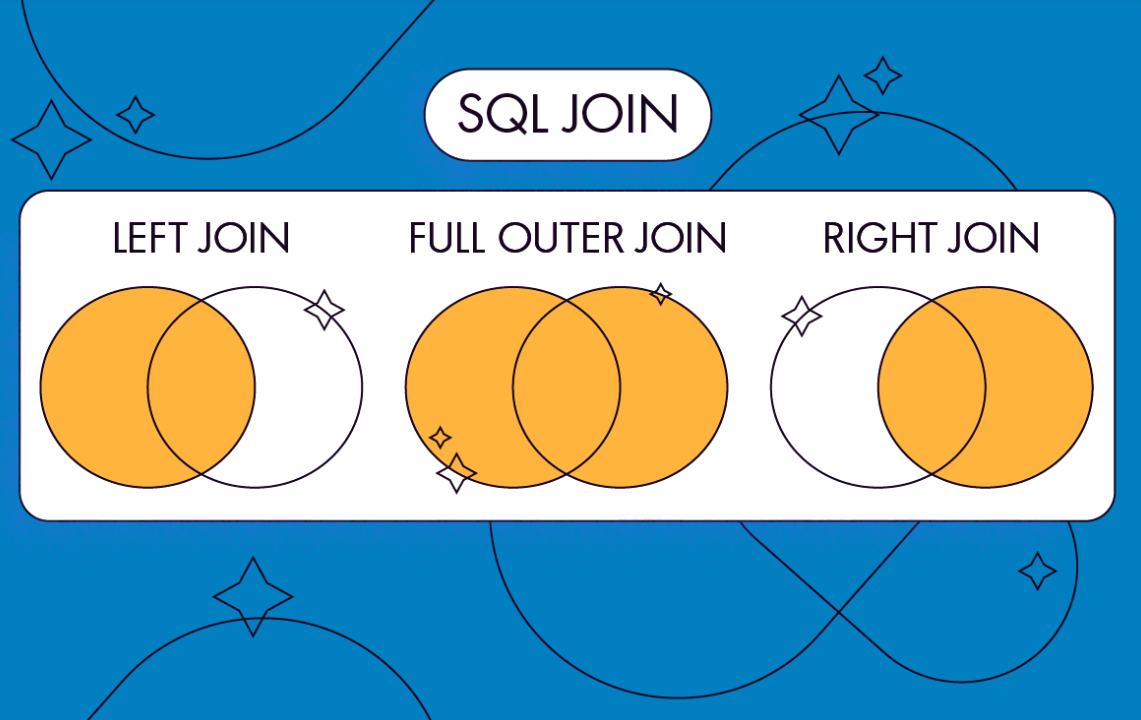
**Что такое join в SQL и как с ним работать**

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

[](https://sky.pro/media/wp-content/uploads/2022/06/image2-1.png)

SQL — Simple Query Language, то есть «простой язык запросов». Его создали, чтобы работать с реляционными базами данных. В таких базах данные представлены в виде таблиц. Зависимости между несколькими таблицами задают с помощью связующих — реляционных столбцов.

Когда запрашиваем данные из одной таблицы, работа со связующими столбцами не нужна. Но если нужно агрегировать данные из нескольких, стоит описать правила: как будут связаны строки на основе значений связующих столбцов. Тогда на помощь и приходит оператор join.

**Что такое оператор join в SQL**

Join — оператор, который используют, чтобы объединять строки из двух или более таблиц на основе связующего столбца между ними. Такой столбец еще называют ключом.

Предположим, что у нас есть таблица заказов — Orders:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerID** | **OrderDate** |
| 304101 | 21 | 10-05-2021 |
| 304102 | 34 | 20-06-2021 |
| 304103 | 22 | 25-07-2021 |

И таблица клиентов — Customers:

|  |  |  |
| --- | --- | --- |
| **CustomerID** | **CustomerName** | **ContactName** |
| 21 | Балалайка Сервис | Иван Иванов |
| 22 | Рога и копыта | Семён Семёнов |
| 23 | Редиска Менеджмент | Пётр Петров |

Столбец CustomerID в таблице заказов соотносится со столбцом CustomerID в таблице клиентов. То есть он — связующий двух таблиц. Чтобы узнать, когда, какой клиент и какой заказ оформил, составьте запрос:

|  |  |
| --- | --- |
|  | **SELECT** Orders.OrderID, Customers.CustomerName, Orders.OrderDate  **FROM** Orders  JOIN Customers  **ON** Orders.CustomerID=Customers.CustomerID; |

Результат запроса будет выглядеть так:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerName** | **OrderDate** |
| 304101 | Балалайка Сервис | 10-05-2021 |
| 304103 | Редиска Менеджмент | 25-07-2021 |

*Общий синтаксис оператора join:*

JOIN <Название таблицы для присоединения> ON <Условие присоединения на основе связующих столбцов>

Соединять можно и больше двух таблиц: к запросу добавьте еще один оператор join. Например, в дополнение к предыдущим двум таблицам у нас есть таблица продавцов — Managers:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **ManagerName** | **ContactDate** |
| 304101 | Артём Лапин | 05-05-2021 |
| 304102 | Егор Орлов | 15-06-2021 |
| 304103 | Евгений Соколов | 20-07-2021 |

Таблица продавцов связана с таблицей заказов столбцом OrderID. Чтобы в дополнение к предыдущему запросу узнать, какой продавец обслуживал заказ, составьте следующий запрос:

|  |  |
| --- | --- |
|  | **SELECT** Orders.OrderID, Customers.CustomerName, Orders.OrderDate, Managers.ManagerName  **FROM** Orders  JOIN Customers  **ON** Orders.CustomerID=Customers.CustomerID  JOIN Managers  **ON** Orders.OrderId=Managers.OrderId |

Результат:

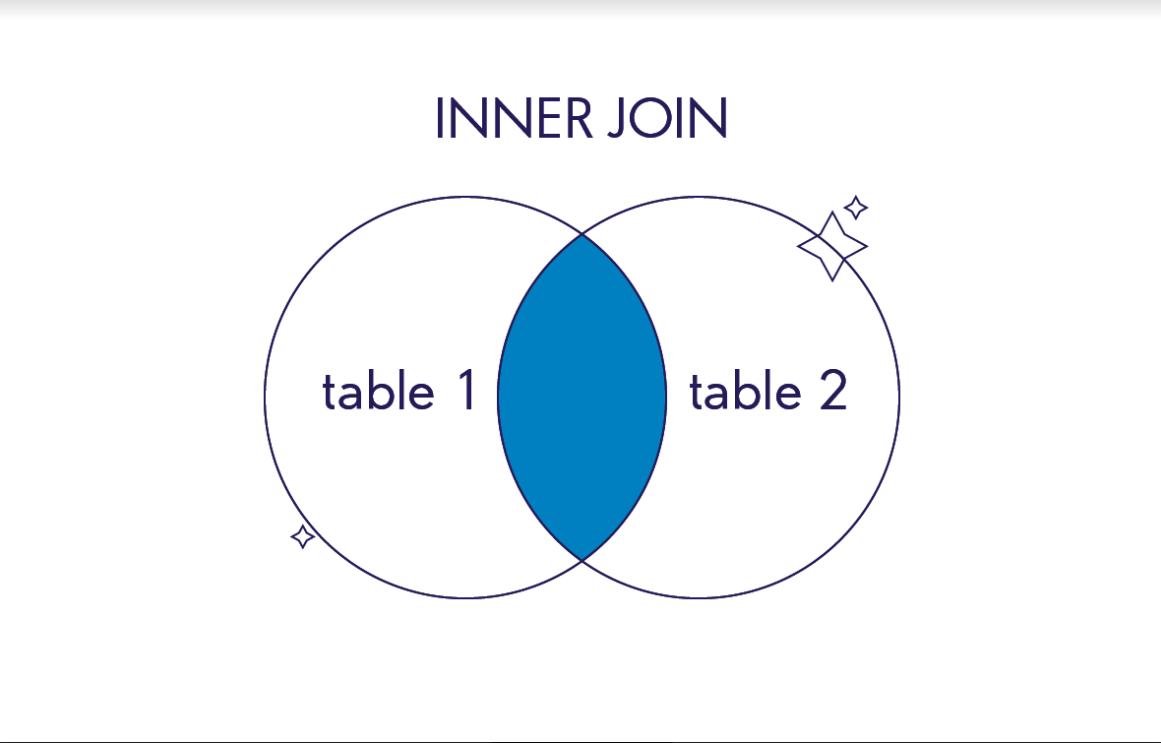
|  |  |  |  |
| --- | --- | --- | --- |
| **OrderID** | **CustomerName** | **OrderDate** | **ManagerName** |
| 304101 | Балалайка Сервис | 10-05-2021 | Артём Лапин |
| 304103 | Редиска Менеджмент | 25-07-2021 | Евгений Соколов |

**Внутреннее соединение INNER JOIN**

Если использовать оператор INNER JOIN, в результат запроса попадут только те записи, для которых выполняется условие объединения. Еще одно условие — записи должны быть в обеих таблицах. В финальный результат из примера выше не попали записи с CustomerID=23 и OrderID=304102: для них нет соответствия в таблицах.

*Общий синтаксис запроса INNER JOIN:*

|  |  |
| --- | --- |
|  | **SELECT** column\_name(s)  **FROM** table1  **INNER** JOIN table2  **ON** table1.column\_name = table2.column\_name; |



*Иллюстрация работы INNER JOIN*

Слово INNER в запросе можно опускать, тогда общий синтаксис запроса будет выглядеть так:

|  |  |
| --- | --- |
|  | **SELECT** column\_name(s)  **FROM** table1  **INNER** JOIN table2  **ON** table1.column\_name = table2.column\_name; |

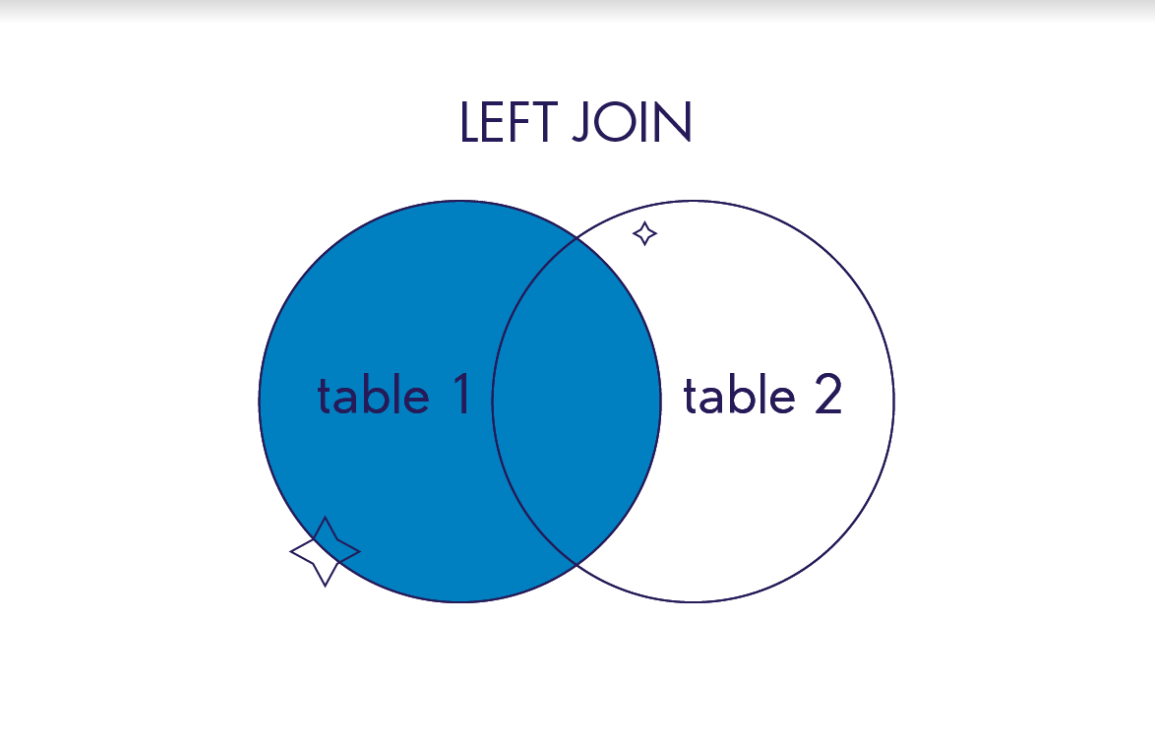
**Внешние соединения OUTER JOIN**

Если использовать внешнее соединение, то в результат запроса попадут не только записи с совпадениями в обеих таблицах, но и записи одной из таблиц целиком. Этим внешнее соединение отличается от внутреннего.

Указание таблицы, из которой нужно выбрать все записи без фильтрации, называется направлением соединения.

**LEFT OUTER JOIN / LEFT JOIN**

В финальный результат такого соединения попадут все записи из левой, первой таблицы. Даже если не будет ни одного совпадения с правой. И записи из второй таблицы, для которых выполняется условие объединения.



*Иллюстрация работы LEFT JOIN*

***Синтаксис:***

|  |  |
| --- | --- |
|  | **SELECT** column\_name(s)  **FROM** table1  LEFT JOIN table2  **ON** table1.column\_name = table2.column\_name; |

***Пример:***

Таблица Orders:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerID** | **OrderDate** |
| 304101 | 21 | 10-05-2021 |
| 304102 | 34 | 20-06-2021 |
| 304103 | 22 | 25-07-2021 |

Таблица Customers:

|  |  |  |
| --- | --- | --- |
| **CustomerID** | **CustomerName** | **ContactName** |
| 21 | Балалайка Сервис | Иван Иванов |
| 22 | Рога и копыта | Семён Семёнов |
| 23 | Редиска Менеджмент | Пётр Петров |

Запрос:

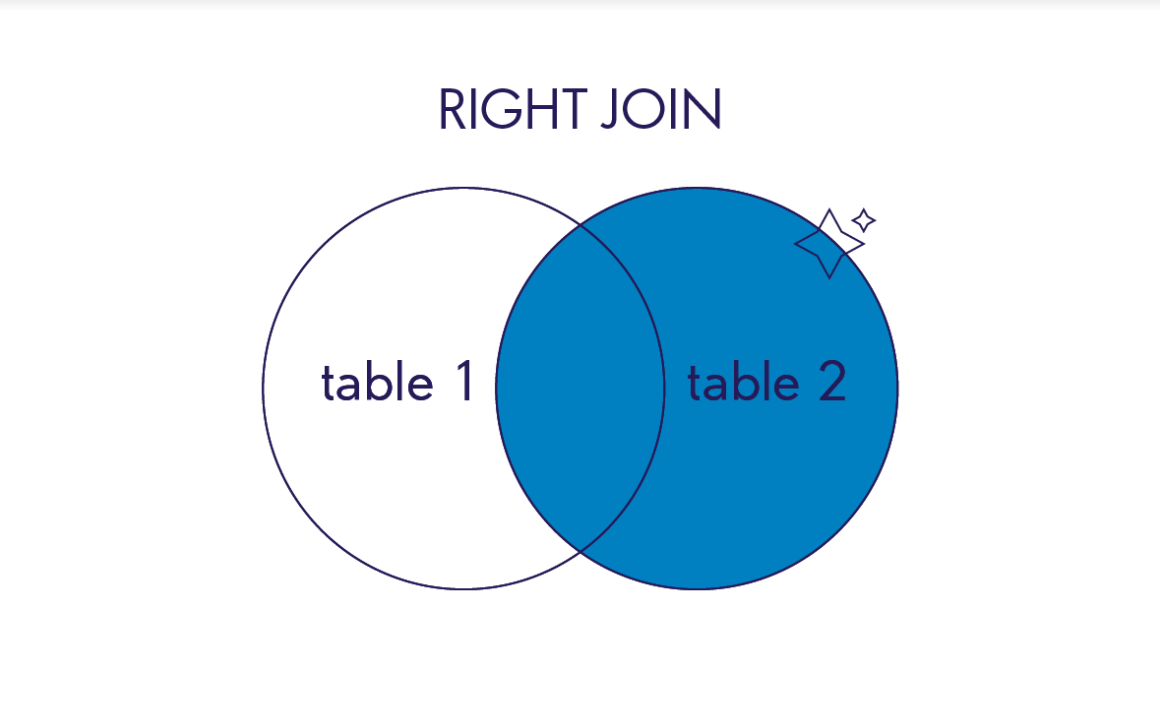
|  |  |
| --- | --- |
|  | **SELECT** Orders.OrderID, Customers.CustomerName, Orders.OrderDate  **FROM** Orders  LEFT JOIN Customers  **ON** Orders.CustomerID=Customers.CustomerID; |

Результат:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerName** | **OrderDate** |
| 304101 | Балалайка Сервис | 10-05-2021 |
| 304102 | null | 20-06-2021 |
| 304103 | Редиска Менеджмент | 25-07-2021 |

**RIGHT OUTER JOIN / RIGHT JOIN**

В финальный результат этого соединения попадут все записи из правой, второй таблицы. Даже если не будет ни одного совпадения с левой. И записи из первой таблицы, для которых выполняется условие объединения.



*Иллюстрация работы RIGHT JOIN*

***Синтаксис:***

|  |  |
| --- | --- |
|  | **SELECT** column\_name(s)  **FROM** table1  RIGHT JOIN table2  **ON** table1.column\_name = table2.column\_name; |

***Пример:***

Таблица Orders:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerID** | **OrderDate** |
| 304101 | 21 | 10-05-2021 |
| 304102 | 34 | 20-06-2021 |
| 304103 | 22 | 25-07-2021 |

Таблица Customers:

|  |  |  |
| --- | --- | --- |
| **CustomerID** | **CustomerName** | **ContactName** |
| 21 | Балалайка Сервис | Иван Иванов |
| 22 | Рога и копыта | Семён Семёнов |
| 23 | Редиска Менеджмент | Пётр Петров |

|  |  |
| --- | --- |
|  | **SELECT** Orders.OrderID, Customers.CustomerName, Orders.OrderDate  **FROM** Orders  RIGHT JOIN Customers  **ON** Orders.CustomerID=Customers.CustomerID; |

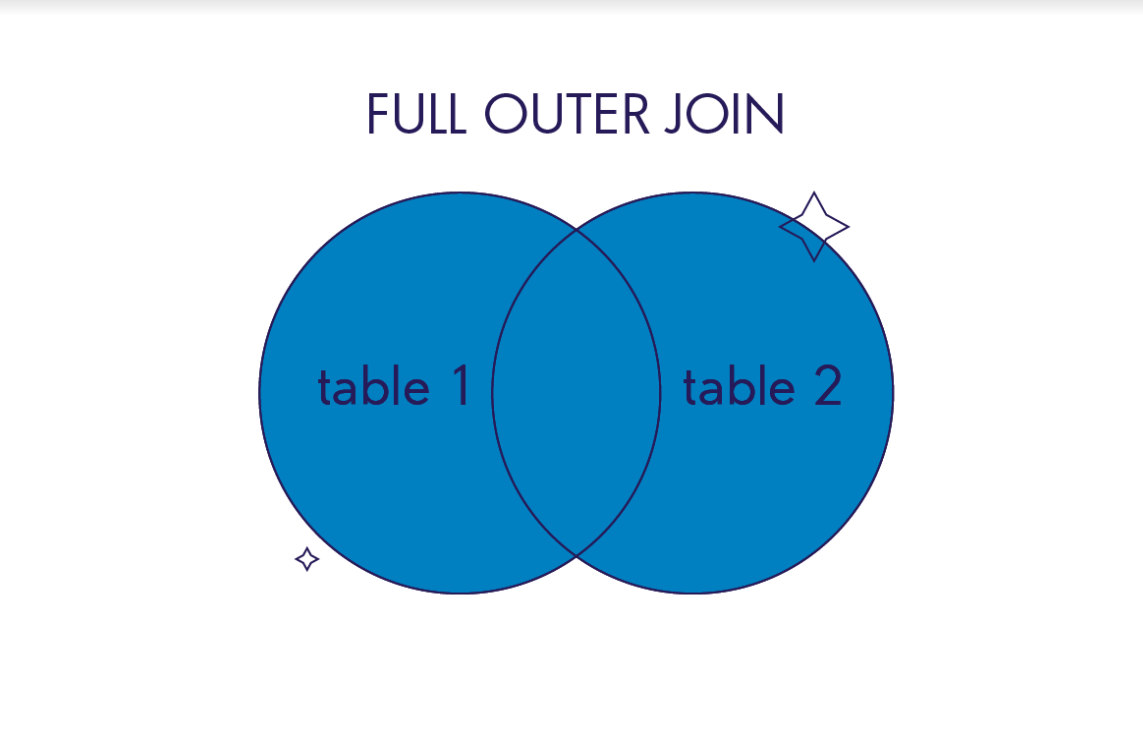
Запрос:

Результат:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerName** | **OrderDate** |
| 304101 | Балалайка Сервис | 10-05-2021 |
| null | Рога и копыта | null |
| 304103 | Редиска Менеджмент | 25-07-2021 |

**FULL OUTER JOIN / FULL JOIN**

В финальный результат такого соединения попадут все записи из обеих таблиц. Независимо от того, выполняется условие объединения или нет.



*Иллюстрация работы FULL JOIN*

|  |  |
| --- | --- |
|  | **SELECT** column\_name(s)  **FROM** table1  **FULL** JOIN table2  **ON** table1.column\_name = table2.column\_name; |

***Синтаксис:***

***Пример:***

Таблица Orders

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerID** | **OrderDate** |
| 304101 | 21 | 10-05-2021 |
| 304102 | 34 | 20-06-2021 |
| 304103 | 22 | 25-07-2021 |

Таблица Customers:

|  |  |  |
| --- | --- | --- |
| **CustomerID** | **CustomerName** | **ContactName** |
| 21 | Балалайка Сервис | Иван Иванов |
| 22 | Рога и копыта | Семён Семёнов |
| 23 | Редиска Менеджмент | Пётр Петров |

Запрос:

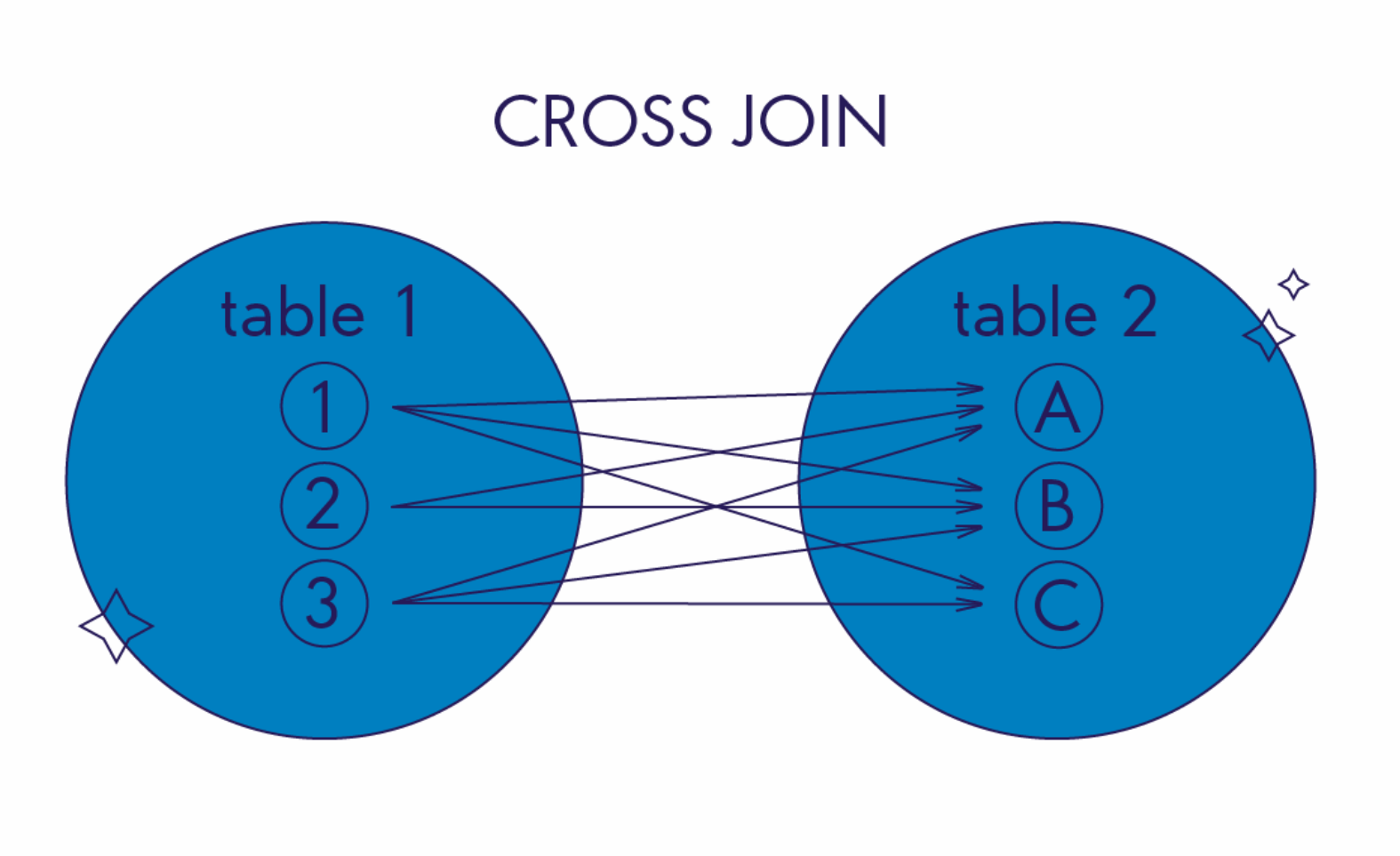
|  |  |
| --- | --- |
|  | **SELECT** Orders.OrderID, Customers.CustomerName, Orders.OrderDate  **FROM** Orders  **FULL** JOIN Customers  **ON** Orders.CustomerID=Customers.CustomerID; |

Результат:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerName** | **OrderDate** |
| 304101 | Балалайка Сервис | 10-05-2021 |
| 304102 | null | 20-06-2021 |
| 304103 | Редиска Менеджмент | 25-07-2021 |
| null | Рога и копыта | null |

**Перекрестное соединение CROSS JOIN**

Этот оператор отличается от предыдущих операторов соединения: ему не нужно задавать условие объединения (ON table1.column\_name = table2.column\_name). Записи в таблице с результатами — это результат объединения каждой записи из левой таблицы с записями из правой. Такое действие называют декартовым произведением.



*Иллюстрация работы CROSS JOIN*

|  |  |
| --- | --- |
|  | **SELECT** column\_name(s)  **FROM** table1  CROSS JOIN table2; |

***Синтаксис:***

***Пример:***

Таблица Orders:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerID** | **OrderDate** |
| 304101 | 21 | 10-05-2021 |
| 304102 | 34 | 20-06-2021 |
| 304103 | 22 | 25-07-2021 |

Таблица Customers:

|  |  |  |
| --- | --- | --- |
| **CustomerID** | **CustomerName** | **ContactName** |
| 21 | Балалайка Сервис | Иван Иванов |
| 22 | Рога и копыта | Семён Семёнов |
| 23 | Редиска Менеджмент | Пётр Петров |

Запрос:

|  |  |
| --- | --- |
|  | SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate  FROM Orders  CROSS JOIN Customers; |

Результат:

|  |  |  |
| --- | --- | --- |
| **OrderID** | **CustomerName** | **OrderDate** |
| 304101 | Балалайка Сервис | 10-05-2021 |
| 304101 | Рога и копыта | 10-05-2021 |
| 304101 | Редиска Менеджмент | 10-05-2021 |
| 304102 | Балалайка Сервис | 20-06-2021 |
| 304102 | Рога и копыта | 20-06-2021 |
| 304102 | Редиска Менеджмент | 20-06-2021 |
| 304103 | Балалайка Сервис | 25-07-2021 |
| 304103 | Рога и копыта | 25-07-2021 |
| 304103 | Редиска Менеджмент | 25-07-2021 |

**Соединение SELF JOIN**

Его используют, когда в запросе нужно соединить несколько записей из одной и той же таблицы.

В [SQL](https://sky.pro/media/chto-takoe-sql/) нет отдельного оператора, чтобы описать SELF JOIN соединения. Поэтому, чтобы описать соединения данных из одной и той же таблицы, воспользуйтесь операторами JOIN или WHERE.

Учтите, что в одном запросе нельзя дважды использовать имя одной и той же таблицы: иначе запрос вернет ошибку. Поэтому, чтобы выполнить соединение таблицы SQL с самой собой, в запросе ей присваивают два разных временных имени — алиаса.

***Синтаксис соединения SELF JOIN при использовании оператора JOIN:***

|  |  |
| --- | --- |
|  | **SELECT** column\_name(s)  **FROM** table1 a1  JOIN table1 a2  **ON** a1.column\_name = a2.column\_name; |

Оператор JOIN может быть любым: используйте LEFT JOIN, RIGHT JOIN. Результат будет таким же, как когда объединяли две разные таблицы.

|  |  |
| --- | --- |
|  | **SELECT** column\_name(s)  **FROM** table1 a1, table1 a2  **WHERE** a1.common\_col\_name = a2.common\_col\_name; |

***Синтаксис соединения SELF JOIN при использовании оператора WHERE:***

***Пример:***

Талица Students:

|  |  |  |  |
| --- | --- | --- | --- |
| **StudentID** | **Name** | **CourseID** | **Duration** |
| 1 | Артём | 1 | 3 |
| 2 | Пётр | 2 | 4 |
| 1 | Артём | 2 | 4 |
| 3 | Борис | 3 | 2 |
| 2 | Ирина | 3 | 5 |

Запрос с оператором WHERE:

|  |  |
| --- | --- |
|  | **SELECT**  s1.StudentID, s1.**Name**  **FROM** Students **AS** s1, Students s2  **WHERE** s1.StudentID = s2.StudentID  AND s1.CourseID <> s2.CourseID; |

Результат:

|  |  |
| --- | --- |
| **StudentID** | **Name** |
| 1 | Артём |
| 2 | Ирина |
| 1 | Артём |
| 2 | Пётр |

|  |  |
| --- | --- |
|  | **SELECT**  s1.StudentID, s1.**Name**  **FROM** Students s1  JOIN Students s2  **ON** s1.StudentID = s2.StudentID  AND s1.CourseID <> s2.CourseID  **GROUP** **BY** StudentID; |

Запрос с оператором JOIN:

Результат:

|  |  |
| --- | --- |
| **StudentID** | **Name** |
| 1 | Артём |
| 2 | Ирина |