БАЗЫ ДАННЫХ

Лекция 2.

Агрегатные функции, группирование

Подзапросы

Соединение таблиц

Таблица поставщиков (s):

n_post	name	rating	town
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

Таблица поставок (sp):

n_post	n_det	date_post	kol
S1	P1	2021-02-01	300
S1	P2	2021-04-05	200
S1	P3	2021-05-12	400
S1	P4	2021-06-15	200
S1	P5	2021-07-22	100
S1	P6	2021-08-13	100
S2	P1	2021-03-03	300
S2	P2	2021-06-12	400
S3	P2	2021-04-04	200
S4	P2	2021-03-23	200
S4	P4	2021-06-17	300
S4	P5	2021-08-22	400

Таблица товаров (p):

n_det	name	cvet	ves	town
P1	Monitor screen	Red	12000	London
P2	Computer mouse	Green	150	Paris
P3	Keyboard	Blue	400	Rome
P4	Keyboard	Red	350	London
P5	Power unit	Blue	1200	Paris
P6	Computer case	Red	7500	London

Агрегатные функции

- выполняют вычисление на наборе строк
- принимают аргумент из одного столбца
- возвращают единственное результирующее значение

Основные агрегатные функции:

Функция	**Описание**
COUNT()	Возвращает количество строк в группе.
SUM()	Возвращает сумму значений в столбце.
AVG()	Возвращает среднее арифметическое значений в столбце.
MIN()	Возвращает минимальное значение в столбце.
MAX()	Возвращает максимальное значение в столбце.

Выдать общую сумму поставок

SELECT SUM(kol) FROM sp

Агрегатные функции

- выполняют вычисление на наборе строк
- принимают аргумент из одного столбца
- возвращают единственное результирующее значение

Основные агрегатные функции:

Функция	**Описание**
COUNT()	Любые значения
SUM()	Целые, вещественные числа
AVG()	Целые, вещественные числа
MIN()	Числа, даты, строки
MAX()	Числа, даты, строки

SELECT COUNT(*) FROM sp

Группирование

Оператор **GROUP BY** группирует таблицу, представленную фразой FROM в группы таким образом, чтобы в каждой группе все строки имели одно и тоже значение поля, указанного во фразе **GROUP BY**.

Далее к каждой группе перекомпанованной таблицы (а не к каждой строке исходной таблицы) применяется фраза **SELECT**, в результате чего каждое выражение во фразе **SELECT** принимает единственное значение для группы.

Выдать для каждого поставляемого товара его номер и общий объем поставок:

```
SELECT n_det, SUM(kol) FROM SP
group by n_det
```

n_det	sum(kol)
P1	300
P2	800
P4	300
P5	400

Группирование и HAVING

Фраза **HAVING** играет ту же роль для групп, что и фраза **WHERE** для строк и используется для того, чтобы исключать группы, точно так же, как WHERE используется для исключения строк.

Выражение во фразе **HAVING** должно принимать единственное значение для группы.

Выдать номера товаров, поставляемых более чем 1 поставщиком:

```
SELECT n_det
FROM SP
GROUP BY n_det
HAVING COUNT(*) > 1
```

Порядок операций в запросе

1. **SELECT**
2. **FROM**
3. WHERE (фильтр до группирования)
4. GROUP BY
5. HAVING (фильтр после группирования)
6. ORDER BY

Подзапросы

Подзапрос – оператор **SELECT**, вложенный в спецификатор WHERE другого оператора **SELECT** (или **INSERT**, **DELETE**, **UPDATE**).

- В состав каждого подзапроса должны входить **SELECT** и **FROM**.

- Каждый подзапрос должен быть заключен в круглые скобки, чтобы указать серверу баз данных на то, что эту операцию следует выполнить первой.

Подзапросы

Подзапросы бывают коррелированными и некоррелированными.

- Подзапрос является **коррелированным**, если его значение зависит от значения, производимого внешним оператором **SELECT**, который содержит этот подзапрос.

- Любой другой вид запроса называется некоррелированным.

Позапрос

- Важное свойство коррелированного подзапроса: так как он зависит от значения результата внешнего оператора ***select***, то должен выполняться повторно по одному разу для каждого значения, производимого внешним оператором ***select***.
- Некоррелированный подзапрос выполняется только один раз.Подзапрос включается в спецификатор ***where*** оператора ***select*** с помощью следующих ключевых слов:
    - ALL
    - ANY
    - IN
    - EXISTS

Позапрос

- Важное свойство коррелированного подзапроса: так как он зависит от значения результата внешнего оператора SELECT, то должен выполняться повторно по одному разу для каждого значения, производимого внешним оператором SELECT.

- Некоррелированный подзапрос выполняется только один раз.

Позапрос

```
SELECT * FROM s WHERE n_post IN
(SELECT n_post FROM sp where kol>200)
```

```
SELECT * FROM s WHERE
(SELECT kol FROM sp where sp.n_post = s.n_post)>200
```

Позапрос

```
SELECT * FROM s WHERE n_post IN
(SELECT n_post FROM sp where kol>200)
```

Некоррелированный подзапрос, выполнится 1 раз

```
SELECT * FROM s WHERE
(SELECT kol FROM sp where sp.n_post = s.n_post)>200
```

Подзапрос

```
SELECT * FROM s WHERE n_post IN
(SELECT n_post FROM sp where kol>200)
```

Некоррелированный подзапрос, выполнится 1 раз

```
SELECT * FROM s WHERE
(SELECT kol FROM sp where sp.n_post = s.n_post)>200
```

Коррелированный подзапрос, выполнится много раз для каждого строки внешней таблицы

Некоррелированные подзапросы

Фраза ALL. Ключевое слово ALL, указываемое перед подзапросом используется для определения того, выполняется ли условие сравнения для каждого возвращаемого подзапросом значения. Если подзапрос не возвращает ни одного значения, то условие поиска считается выполненным.

Получить перечень поставщиков, рейтинг которых выше рейтинга любого лондонского поставщика.

```
SELECT x.n_post, x.name, x.rating
FROM s x WHERE x.rating > ALL
    (SELECT y.rating
    FROM s y
    WHERE y.town='London')
```

n_post	name	rating
S3	Blake	30
S5	Adams	30

Некоррелированные подзапросы

Фраза ANY. Ключевое слово ANY, указываемое перед запросом, используется для определения того, выполняется ли сравнение по крайней мере для одного значения, возвращаемого подзапросом. Если подзапрос не возвращает ни одного значения, то условие поиска считается не выполненным.

Получить перечень поставщиков, рейтинг которых выше рейтинга хотя бы одного парижского поставщика.

```
SELECT x.n_post, x.name, x.rating
FROM S x
WHERE x.rating > ANY
    (SELECT y.rating
    FROM S y
    WHERE y.town = 'Paris')
```

n_post	name	rating
S1	Smith	20
S3	Blake	30
S4	Clark	20
S5	Adams	30

Некоррелированные подзапросы

Фраза IN. Простой подзапрос.

Выдать фамилии поставщиков, поставляющих товар под номером P2.

```
SELECT name
FROM s
WHERE n_post IN
        (SELECT n_post
        FROM sp
        WHERE n_det ='P2')
```

Некоррелированные подзапросы

Подзапрос с несколькими уровнями вложенности

Выдать фамилии поставщиков, поставляющих по крайней мере один красный товар.

```
SELECT name FROM s
WHERE n_post IN
        (SELECT n_post FROM sp
        WHERE n_det IN
                (SELECT n_det FROM p
                WHERE cvet='Red'))
```

Некоррелированные подзапросы

Использование одной и той же таблицы в подзапросе внешнем запросе

Выдать номера поставщиков, поставляющих, по крайней мере, один товар, поставляемый поставщиком S2.

```
SELECT DISTINCT n_post FROM sp spx
WHERE spx.n_det IN
    (SELECT spy.n_det FROM sp spy
    WHERE spy.n_post='S2')
```

Некоррелированные подзапросы

Подзапрос с оператором сравнения, отличным от IN

Выдать номера поставщиков, находящихся в том же городе, что и поставщик S1

```
SELECT n_post
FROM S
WHERE town =
        (SELECT town
        FROM S
        WHERE n_post ='S1')
```

Коррелированные подзапросы

Выдать фамилии поставщиков, поставляющих товар P2.

```
SELECT фамилия
FROM S
WHERE 'P2’ IN
        (SELECT номер_товара
        FROM SP
        WHERE номер_поставщика= S.номер_поставщика)
```

В коррелированном подзапросе внутренний подзапрос не может быть отработан раз и навсегда, прежде чем будет отработан внешний запрос, поскольку этот внутренний подзапрос зависит от переменной, значение которой изменяется по мере того, как система проверяет различные строки таблицы, участвующие во внешнем запросе.

Коррелированные подзапросы

Коррелированный подзапрос с использованием одной и той же таблицы

Выдать номера товаров, поставляемых более чем одним поставщиком.

```
SELECT DISTINCT spx.n_det
FROM sp spx
WHERE spx.n_det in
        (SELECT spy.n_det
        FROM sp spy
        WHERE spy.n_post<>spx.n_post)
```

Оператор EXISTS

Выдать фамилии поставщиков, поставляющих товара P2.

```
SELECT name
FROM s
WHERE EXISTS

    (SELECT * from sp
    WHERE sp.n_post = s.n_post and n_det = 'P2')
```

Использование функций в подзапросе

Выдать поставщиков с рейтингом меньшим, чем максимальный в таблице

```
SELECT name
FROM s
WHERE rating <
    (SELECT MAX(rating) FROM s)
```

Выдать информацию о всех поставщиках, у которых рейтинг больше, чем средний для конкретно их города

```
SELECT * FROM s sx
WHERE rating >
    (SELECT AVG(rating)
    FROM s sy
    WHERE sy.town=sx.town)
```

Соединение таблиц

Классическая реляционная алгебра Кодда включает девять реляционных операций, последовательное применение которых позволяет реализовать выборку любых данных. Три из этих операции так или иначе связаны с соединением таблиц:

- операция взятия декартова произведения;
- операция соединения (соответствующая ей в стандарте ANSI операция носит название операции внутреннего соединение);
- операция эквисоединения.

Операция взятия декартова произведения содержит все возможные комбинации конкатенаций кортежей (строк) из соединяемых таблиц.

Операция соединения представляет собой соединение кортежей соединяемых таблиц по указанному условию соединения. Строки, которые не удовлетворяют условиям соединения, отбрасываются.

Операция эквисоединения является частным случаем операции соединение по условию равенства атрибутов.

Кроме этого существует практически важное расширение операции эквисоединения – естественное соединение.

Внутреннее соединение (INNER JOIN) используется, когда нужно включить все строки из обеих таблиц, удовлетворяющие условию объединения.

Внутреннее соединение имеет место и тогда, когда в предложении WHERE сравниваются значения полей из разных таблиц. В этом случае строится декартово произведение строк первой и второй таблиц, а из полученного набора данных отбираются записи, удовлетворяющие условиям объединения.

В условиях объединения могут участвовать поля, относящиеся к одному и тому же типу данных и содержащие один и тот же вид данных, но они не обязательно должны иметь одинаковые имена.

Блоки данных из двух таблиц объединяются, как только в указанных полях будут найдены совпадающие значения.

Если в предложении FROM перечислено несколько таблиц и при этом не употребляется спецификация JOIN, а для указания соответствия полей из таблиц используется условие в предложении WHERE, то некоторые реляционные СУБД (например, Access) оптимизируют выполнение запроса, интерпретируя его как соединение.

Если перечислять ряд таблиц или запросов и не указывать условия объединения, в качестве результирующей таблицы будет выбрано декартово (прямое) произведение всех таблиц.

Естественным соединением (NATURAL JOIN) называется соединение по эквивалентности двух отношений R и S, выполненное по всем общим атрибутам, из результатов которого исключается по одному экземпляру каждого общего атрибута.

Внешнее соединение может сохранить строки, для которых не находится соответствия в другой таблице. В этом случае недостающие поля заполняются значениями NULL. Решение о том, войдет ли такая строка в результирующий набор, зависит от того, в какой из соединяемых таблиц отсутствуют данные, и от типа внешнего соединения. Существуют три разновидности внешних соединений:

- Левое внешнее соединение. Всегда содержит как минимум один экземпляр каждой строки из таблицы, указанной слева от ключевого слова JOIN. Отсутствующие поля из правой таблицы заполняются значениями NULL.
- Правое внешнее соединение. Всегда содержит как минимум один экземпляр каждой строки из таблицы, указанной справа от ключевого слова JOIN. Отсутствующие поля из левой таблицы заполняются значениями NULL.
- Полное внешнее соединение. Всегда содержит как минимум один экземпляр каждой строки каждой из соединяемых таблиц. Отсутствующие поля в записях результирующего набора заполняются значениями NULL.

Для построения соединений стандарт ANSI предусматривает следующую конструкцию спецификаторов ***from*** и ***join***:

FROM *источник1* [*Natural*] *тип соединения* **JOIN** *источник2* [*on* условие [,...] | *Using* (поле1 [,...])]

- *Источник1*. Первый из соединяемых наборов данных (имя таблицы или подзапрос).
- *[Natural]*. Два набора данных соединяются по равным значениям одноименных полей. Конструкции *On* и *Using* в этом случае недопустимы.

Тип соединения **Возможные виды соединений:**

1. [Inner] - внутреннее соединение;
2. Left [Outer] - левое внешнее соединение;
3. Right [Outer] - правое внешнее соединение;
4. Full [Outer] - полное внешнее соединение;
5. Cross – декартово произведение;

- *Источник2*. Второй из соединяемых наборов данных (имя таблицы или подзапрос).
- *On условие [,...]*. Отношение между источниками - критерий, аналогичный тому, который задается в конструкции *Where*.
- *Using (поле1 [,...])*. Одноименные поля источников, по совпадающим значениям которых производится соединение. В отличии от *Natural* позволяет ограничиться некоторыми одноименными полями, тогда как *Natural* производит соединение по всем одноименным полям.

FROM *источник1* [*Natural*] *тип соединения* **JOIN** *источник2* [*on* условие [,...] | *Using* (поле1 [,...])]

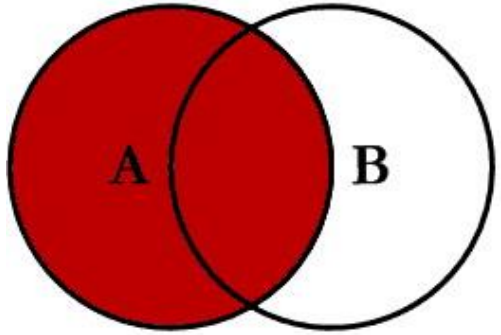
- *Источник1*. Первый из соединяемых наборов данных (имя таблицы или подзапрос).
- *[Natural]*. Два набора данных соединяются по равным значениям одноименных полей. Конструкции *On* и *Using* в этом случае недопустимы.

  Тип соединения: Inner / Left [Outer] / Right [Outer] / Full [Outer] / Cross.

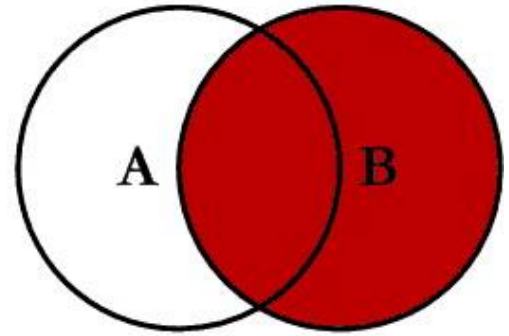
- *Источник2*. Второй из соединяемых наборов данных (имя таблицы или подзапрос).
- *On условие [,...]*. Отношение между источниками - критерий, аналогичный тому, который задается в конструкции *Where*.
- *Using (поле1 [,...])*. Одноименные поля источников, по совпадающим значениям которых производится соединение. В отличии от *Natural* позволяет ограничиться некоторыми одноименными полями, тогда как *Natural* производит соединение по всем одноименным полям.

SQL JOINS

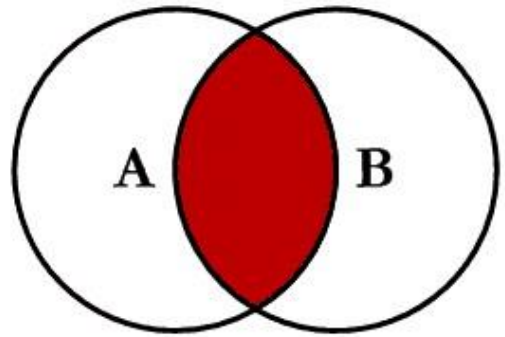
(упрощенно, но наглядно)



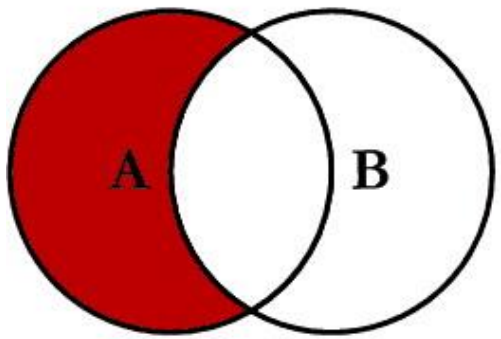

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



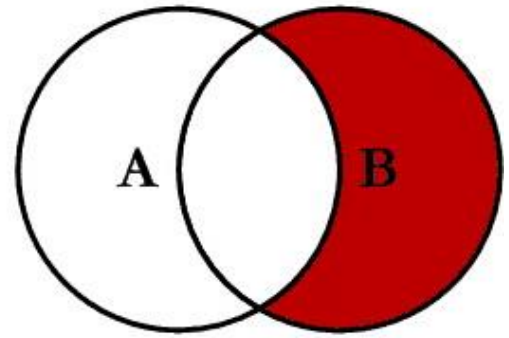

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



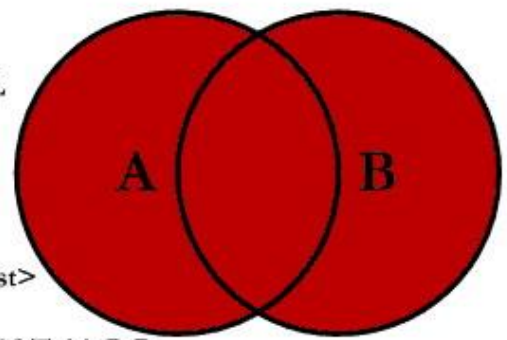

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



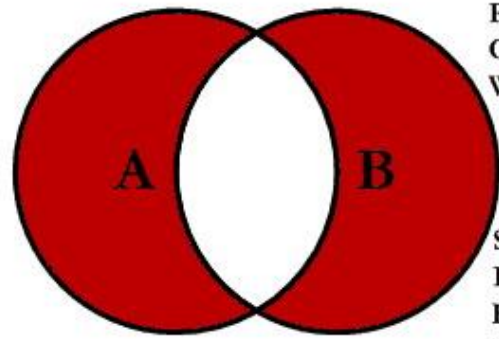

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```



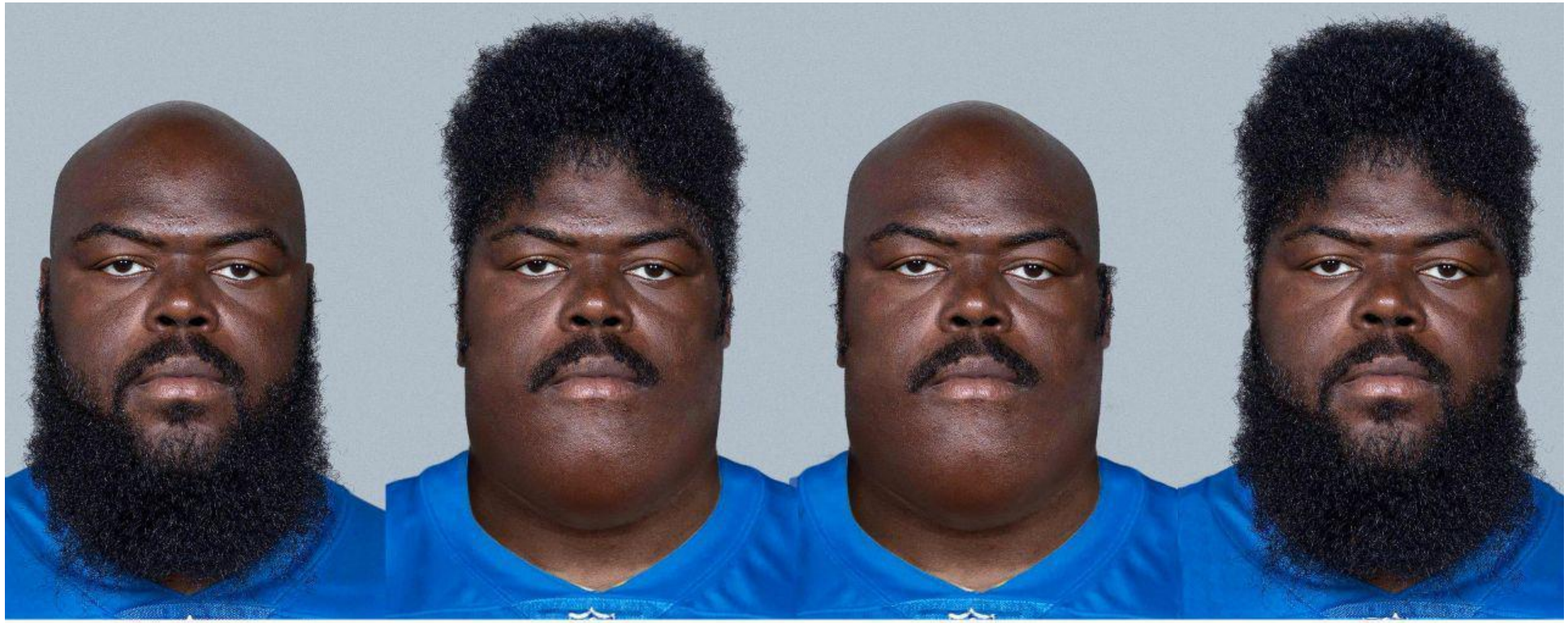

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```


LEFT JOIN
RIGHT JOIN
INNER JOIN
FULL OUTER JOIN

Пример.Простое декартово произведение.

Выдать информацию обо всех возможных парах поставщик - товар.

Select S.*, P.*

from S

Cross Join P

n_post	name	rating	town	n_det	name	cvet	ves	town
S1	Smith	20	London	P1	Monitor screen	Red	12000	London
S2	Jones	10	Paris	P1	Monitor screen	Red	12000	London
S3	Blake	30	Paris	P1	Monitor screen	Red	12000	London
S4	Clark	20	London	P1	Monitor screen	Red	12000	London
S5	Adams	30	Athens	P1	Monitor screen	Red	12000	London
S1	Smith	20	London	P2	Computer mouse	Green	150	Paris
S2	Jones	10	Paris	P2	Computer mouse	Green	150	Paris
S3	Blake	30	Paris	P2	Computer mouse	Green	150	Paris
S4	Clark	20	London	P2	Computer mouse	Green	150	Paris
S5	Adams	30	Athens	P2	Computer mouse	Green	150	Paris
S1	Smith	20	London	P3	Keyboard	Blue	400	Rome
S2	Jones	10	Paris	P3	Keyboard	Blue	400	Rome
S3	Blake	30	Paris	P3	Keyboard	Blue	400	Rome
S4	Clark	20	London	P3	Keyboard	Blue	400	Rome
S5	Adams	30	Athens	P3	Keyboard	Blue	400	Rome
S1	Smith	20	London	P4	Keyboard	Red	350	London
S2	Jones	10	Paris	P4	Keyboard	Red	350	London
S3	Blake	30	Paris	P4	Keyboard	Red	350	London
S4	Clark	20	London	P4	Keyboard	Red	350	London
S5	Adams	30	Athens	P4	Keyboard	Red	350	London
S1	Smith	20	London	P5	Power unit	Blue	1200	Paris
S2	Jones	10	Paris	P5	Power unit	Blue	1200	Paris
S3	Blake	30	Paris	P5	Power unit	Blue	1200	Paris
S4	Clark	20	London	P5	Power unit	Blue	1200	Paris
S5	Adams	30	Athens	P5	Power unit	Blue	1200	Paris
S1	Smith	20	London	P6	Computer case	Red	7500	London
S2	Jones	10	Paris	P6	Computer case	Red	7500	London
S3	Blake	30	Paris	P6	Computer case	Red	7500	London
S4	Clark	20	London	P6	Computer case	Red	7500	London
S5	Adams	30	Athens	P6	Computer case	Red	7500	London

Замечание: тот же результат может быть получен запросом

Select * from S, P

Результат:

n_post	name	rating	town	n_det	name	cvet	ves	town
S1	Smith	20	London	P1	Monitor screen	Red	12000	London
S2	Jones	10	Paris	P1	Monitor screen	Red	12000	London
S3	Blake	30	Paris	P1	Monitor screen	Red	12000	London
S4	Clark	20	London	P1	Monitor screen	Red	12000	London
S5	Adams	30	Athens	P1	Monitor screen	Red	12000	London
S1	Smith	20	London	P2	Computer mouse	Green	150	Paris
S2	Jones	10	Paris	P2	Computer mouse	Green	150	Paris
S3	Blake	30	Paris	P2	Computer mouse	Green	150	Paris
S4	Clark	20	London	P2	Computer mouse	Green	150	Paris
S5	Adams	30	Athens	P2	Computer mouse	Green	150	Paris
S1	Smith	20	London	P3	Keyboard	Blue	400	Rome
S2	Jones	10	Paris	P3	Keyboard	Blue	400	Rome
S3	Blake	30	Paris	P3	Keyboard	Blue	400	Rome
S4	Clark	20	London	P3	Keyboard	Blue	400	Rome
S5	Adams	30	Athens	P3	Keyboard	Blue	400	Rome
S1	Smith	20	London	P4	Keyboard	Red	350	London
S2	Jones	10	Paris	P4	Keyboard	Red	350	London
S3	Blake	30	Paris	P4	Keyboard	Red	350	London
S4	Clark	20	London	P4	Keyboard	Red	350	London
S5	Adams	30	Athens	P4	Keyboard	Red	350	London
S1	Smith	20	London	P5	Power unit	Blue	1200	Paris
S2	Jones	10	Paris	P5	Power unit	Blue	1200	Paris
S3	Blake	30	Paris	P5	Power unit	Blue	1200	Paris
S4	Clark	20	London	P5	Power unit	Blue	1200	Paris
S5	Adams	30	Athens	P5	Power unit	Blue	1200	Paris
S1	Smith	20	London	P6	Computer case	Red	7500	London
S2	Jones	10	Paris	P6	Computer case	Red	7500	London
S3	Blake	30	Paris	P6	Computer case	Red	7500	London
S4	Clark	20	London	P6	Computer case	Red	7500	London
S5	Adams	30	Athens	P6	Computer case	Red	7500	London

В отличие от предыдущего запроса этот запрос написан с отклонением от стандарта ANSI, но он более точно отражает смысл операции взятия декартова произведения.

При написании запросов следует придерживаться стандарта ANSI, позволяющего формировать более читабельные запросы.

Таблица поставщиков (s):

n_post	name	rating	town
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

Таблица поставок (sp):

n_post	n_det	date_post	kol
S1	P1	2021-02-01	300
S1	P2	2021-04-05	200
S1	P3	2021-05-12	400
S1	P4	2021-06-15	200
S1	P5	2021-07-22	100
S1	P6	2021-08-13	100
S2	P1	2021-03-03	300
S2	P2	2021-06-12	400
S3	P2	2021-04-04	200
S4	P2	2021-03-23	200
S4	P4	2021-06-17	300
S4	P5	2021-08-22	400

Таблица товаров (p):

n_det	name	cvet	ves	town
P1	Monitor screen	Red	12000	London
P2	Computer mouse	Green	150	Paris
P3	Keyboard	Blue	400	Rome
P4	Keyboard	Red	350	London
P5	Power unit	Blue	1200	Paris
P6	Computer case	Red	7500	London

Пример. Простое эквисоединение.

Выдать все комбинации информации о поставщиках

и товарах, расположенных в одном городе.

```
Select S.n_post, p.n_det, rating
from S
Cross Join P
where S.town = P.town
```

n_post	n_det	rating
S1	P1	20
S4	P1	20
S2	P2	10
S3	P2	30
S1	P4	20
S4	P4	20
S2	P5	10
S3	P5	30
S1	P6	20
S4	P6	20

```
Select S.*, p.*
from S
Cross Join P
where S.town = P.town
```

n_post	name	rating	town	n_det	name	cvet	ves	town
S1	Smith	20	London	P1	Monitor screen	Red	12000	London
S4	Clark	20	London	P1	Monitor screen	Red	12000	London
S2	Jones	10	Paris	P2	Computer mouse	Green	150	Paris
S3	Blake	30	Paris	P2	Computer mouse	Green	150	Paris
S1	Smith	20	London	P4	Keyboard	Red	350	London
S4	Clark	20	London	P4	Keyboard	Red	350	London
S2	Jones	10	Paris	P5	Power unit	Blue	1200	Paris
S3	Blake	30	Paris	P5	Power unit	Blue	1200	Paris
S1	Smith	20	London	P6	Computer case	Red	7500	London
S4	Clark	20	London	P6	Computer case	Red	7500	London

Замечание: тот же результат может быть получен запросом с использованием конструкции операции внутреннего соединения в стандарте ANSI

```
Select S.n_post, P.n_det, rating
from S
Inner Join P on S.town=P.town
```

n_post	n_det	rating
S1	P1	20
S4	P1	20
S2	P2	10
S3	P2	30
S1	P4	20
S4	P4	20
S2	P5	10
S3	P5	30
S1	P6	20
S4	P6	20

или запросом, написанным с отклонением от стандарта ANSI

```
Select S.n_post, P.n_det, rating
from S, P
where S. town =P. town
```

Пример. Соединение таблиц с дополнительным условием.

Выдать все комбинации информации о поставщиках и товарах, расположенных в одном городе, кроме поставщиков с рейтингом = 20.

```
Select S.n_post, p.n_det, rating
from S
Cross Join P
where S.town = P.town and S.rating < > 20
```

n_post	n_det	rating
S2	P2	10
S3	P2	30
S2	P5	10
S3	P5	30

Пример. Соединение таблицы с ней самой.

Выдать все пары поставщиков из одного города.

```
Select one.n_post, two.n_post
from S one
Cross Join S two
where one.town = two.town  and one.n_post <
two.n_post
```

n_post	n_post
S2	S3
S1	S4

Пример. Внутреннее соединение.

Выдать для каждой поставки номер поставщика,

его фамилию и количество товаров.

```
Select S.n_post, S.name, SP.kol
from SP
inner join S on S.n_post=SP.n_post
```

n_post	name	kol
S1	Smith	300
S1	Smith	200
S1	Smith	400
S1	Smith	200
S1	Smith	100
S1	Smith	100
S2	Jones	300
S2	Jones	400
S3	Blake	200
S4	Clark	200
S4	Clark	300
S4	Clark	400

Пример. Соединение трех таблиц.

Выдать все пары названий городов, таких, что какой-либо поставщик, находящийся в первом из этих городов, поставляет товар, хранимый в другом городе.

```
Select distinct S.town, P.town
from SP
inner join S on S.n_post = SP.n_post
inner join P on P.n_det = SP.n_det
```

town	town
London	London
London	Paris
London	Rome
Paris	London
Paris	Paris

Пример. Простое внешнее соединение двух таблиц.

Пример левого внешнего соединения.

```
Select S.n_post, S.name, SP.kol
from S
left outer join SP on (S.n_post=SP.n_post)
```

n_post	name	kol
S1	Smith	300
S1	Smith	200
S1	Smith	400
S1	Smith	200
S1	Smith	100
S1	Smith	100
S2	Jones	300
S2	Jones	400
S3	Blake	200
S4	Clark	200
S4	Clark	300
S4	Clark	400
S5	Adams	*NULL*

Добавление ключевого слова *outer* перед именем таблицы SP превращает ее в подчиненную таблицу.

Результатом этого внешнего соединения будет получение сведений обо всех поставщиках, независимо от того, делали ли они поставки.

Полное внешнее соединение даст аналогичный результат, правое - результат, аналогичный внутреннему соединению.

Объединение

Объединяемые оператором UNION или UNION ALL таблицы должны быть совместны по объединению:

- иметь одинаковое число столбцов;
- соответствующие столбцы должны иметь одинаковые типы.

Любое число предложений SELECT может быть соединено оператором UNION.

При использовании оператора UNION избыточные дубликаты записей (где все столбцы в результатах одинаковы) исключаются из результата объединения, а при использовании оператора UNION ALL объединение происходит без удаления дубликатов.

Пример

Выдать номера товаров, которые имеют вес более 16 фунтов, либо поставляются поставщиком S2.

```
Select номер_товара
from P
where вес>16
union
Select номер_товара
from SP
where номер_поставщика='S2’
```

Результат:

Номер_товара
P1
P2
P3
P6

Таблица поставщиков (s):

n_post	name	rating	town
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

Таблица товаров (p):

n_det	name	cvet	ves	town
P1	Monitor screen	Red	12000	London
P2	Computer mouse	Green	150	Paris
P3	Keyboard	Blue	400	Rome
P4	Keyboard	Red	350	London
P5	Power unit	Blue	1200	Paris
P6	Computer case	Red	7500	London

Таблица поставок (sp):

n_post	n_det	date_post	kol
S1	P1	2021-02-01	300
S1	P2	2021-04-05	200
S1	P3	2021-05-12	400
S1	P4	2021-06-15	200
S1	P5	2021-07-22	100
S1	P6	2021-08-13	100
S2	P1	2021-03-03	300
S2	P2	2021-06-12	400
S3	P2	2021-04-04	200
S4	P2	2021-03-23	200
S4	P4	2021-06-17	300
S4	P5	2021-08-22	400

Пример. Внешнее соединение внутреннего соединения с третьей таблицей.

Для получении внешнего соединения будем использовать представление (***View***). Представление можно рассматривать как хранимый запрос, на основании которого создается объект базы данных. Этот объект схож с таблицей, но в его содержимом динамически отражаются только те записи, которые были заданы при создании.

Представление создается командой ***Create view***:

Create view имя_представления ***as*** запрос

Представление удаляется командой ***Drop view***.

Create view z1 as

select sp.n_post, sp.n_det, p.name, p.cvet, p.ves

from SP

join P on SP.n_det = P.n_det

where cvet in ('Red', 'Green');

Первый оператор выполняет внутреннее соединение таблиц SP и P

n_post	n_det	name	cvet	ves
S1	P1	Monitor screen	Red	12000
S1	P2	Computer mouse	Green	150
S1	P4	Keyboard	Red	350
S1	P6	Computer case	Red	7500
S2	P1	Monitor screen	Red	12000
S2	P2	Computer mouse	Green	150
S3	P2	Computer mouse	Green	150
S4	P2	Computer mouse	Green	150
S4	P4	Keyboard	Red	350

Select S.n_post, S.name, z1.n_det, z1.name,

z1.cvet, z1.ves from s

left join z1 on (s.n_post = z1.n_post)

Затем оператором Select выполняется внешнее соединение как комбинирование информации, полученной в первом запросе, с данными из главной таблицы S

n_post	name	n_det	name	cvet	ves
S1	Smith	P1	Monitor screen	Red	12000
S1	Smith	P2	Computer mouse	Green	150
S1	Smith	P4	Keyboard	Red	350
S1	Smith	P6	Computer case	Red	7500
S2	Jones	P1	Monitor screen	Red	12000
S2	Jones	P2	Computer mouse	Green	150
S3	Blake	P2	Computer mouse	Green	150
S4	Clark	P2	Computer mouse	Green	150
S4	Clark	P4	Keyboard	Red	350
S5	Adams	*NULL*	*NULL*	*NULL*	*NULL*