отношение "многие ко многим" между двумя таблицами всегда происходит с помощью введения третьей таблицы

для того, чтобы выбрать данные из таблицы, надо воспользоваться командой SELECT,

\*- позволяет выбрать все данные из таблицы, из всех колонок

если указывать наименования столбцов в таблицах, будут они и выгружаться,

выгружать можно без дублирования (только уникальные объекты) повторяющихся объектов через команду

SELECT DISTINCT \_\_\_\_

from\_\_\_\_\_

выгружать можно несколько столбцов, если выгружать - как тут - города и страны, то наиболее полный список будет в приоритете,

при этом более узкий может уже содержать неуникальные объекты, то есть расширяется

"Seattle" "USA"

"Kirkland" "USA"

"London" "UK"

"Redmond" "USA"

"Tacoma" "USA"

можно делать **подсчет выведенных строк**

допустим, есть куча заказов, как вывести их количество? можно вывести все заказы, перемотать вниз и посмотреть,

но зачастую нужно получить конкретное число программно, чтобы потом его использовать

делается через запрос

SELECT COUNT(\*)

from \_\_\_

можно объединить запросы, чтобы посчитать количество стран, в которых работают работники

SELECT COUNT (DISTINCT country)

FROM employees

**фильтрация WHERE**

если надо наложить **фильтрацию** по **условию**

SELECT \_\_\_\_

FROM table

WHERE условие

a = b, a > b, a < b, a >= b, a <= b, a <> b или a!=b , два последних - одинаковые

Выбрал всю строку, т.к. Берлин один

SELECT \*

FROM customers

WHERE city = 'Berlin'

и еще пример

SELECT first\_name, last\_name, home\_phone, city

FROM employees

WHERE city = 'London'

**несколько условий, мб не только два, а и больше**

select \*

from orders

WHERE order\_date > '1998-01-01' and order\_date < '1998-01-31'

select \*

from customers

where country = 'USA' or country = 'UK'

когда идет сложное условие, надо это условие разграничивать семантически, по смыслу то есть, чтобы не было каких-либо проблем с выгрузкой данных

в примере ниже страну отправки надо было заключить в скобки, чтобы интерпретатор правильно все понимал и не допускал ошибок

select \*

from orders

WHERE order\_date > '1998-01-01' AND (ship\_country = 'USA' OR ship\_country = 'UK' OR ship\_country = 'Germany')

чтобы получить интервал, мы можем как использовать и старые операторы, и там проставить диапазон, например, больше 50 и меньше 100, так и использовать оператор BETWEEN, при этом здесь идет включение и верхней границы, и нижней, то есть BETWEEN это >= ------ <=

SELECT \*

FROM orders

WHERE freight between '10' AND '15'

SELECT \*

FROM orders

WHERE order\_date between '1998-01-01' AND '1998-01-31'

Если в условиях нужно осуществить выборку по нескольким одинаковым условиям, типа выбрать всех, у кого город – и тут идет 5 городов или больше, то перебирать их через OR не очень красиво, есть оператор IN

Если надо выбрать всех, кроме каких-то – есть оператор NOT IN

SELECT \*

FROM customers

WHERE country IN ('UK', 'Germany', 'Mexico') – выбирает только 'UK', 'Germany', 'Mexico'

С числами также

SELECT \*

FROM products

WHERE category\_id NOT IN (1,3,5) – выбирает все, кроме 1, 3, 5

Если надо **отсортировать**, например, список стран, куда отправляются посылки, надо использовать ORDER BY,

Он идет после FROM и WHERE. По умолчанию сортировка идет по возрастанию, в запросе неявно указывается ASC – ассендинг, типа возрастание, DESC – убывание, его уже надо прописывать

SELECT DISTINCT(country)

FROM customers

ORDER BY country

Сортировка может идти в нескольких столбиках одновременно, допустим, мы выгружаем список стран, куда отправляем посылки, и сразу же выгружаем список городов

SELECT DISTINCT country, city

FROM customers

ORDER BY country, city

**Есть такая тема как поиск всякого**

% - этот символ подразумевает под собой любое количество символов, будь то цифра или же буква и тд

\_ - это символ, который подразумевает под собой только один любой символ,

Пример,

LIKE ‘U%’ – строка, которая начинается с U

LIKE ‘%a’ – строка, которая кончается на а

LIKE ‘%John%’ – строка, которая содержит в себе имя Джон

LIKE ‘J%n’ – тут типа начинается на Дж, кончается на н

LIKE ‘\_oh\_’ – ну тут понятно

LIKE ‘\_oh%’ – начинается с одного символа, а кончается уже любым количеством символов

SELECT last\_name, first\_name

FROM employees

WHERE last\_name LIKE '%n'

**LIMIT** – вывод ограниченного количества элементов выводимого списка

SELECT product\_name, unit\_price

FROM products

WHERE discontinued <> 1

ORDER BY unit\_price

LIMIT 10

**Проверка на NULL**

Если надо проверить, есть ли какое-то значение в конкретном поле или оно пустое, то делаем так:

SELECT ship\_city, ship\_region, ship\_country

FROM orders

WHERE ship\_region IS NULL

Тут ищутся все записи, где регион отправки не заполнен, а если надо такие, где обязательно не пустое, то

SELECT ship\_city, ship\_region, ship\_country

FROM orders

WHERE ship\_region IS NOT NULL

**Группировка по определенным значениям**

Здесь выбираем страну отправки, подчитываем количество товаров больше 50 кг и группируем по странам

И сортируем по количеству посылок по возрастанию

SELECT ship\_country, COUNT (\*)

FROM orders

WHERE freight > 50

GROUP BY ship\_country

ORDER BY COUNT(\*)

Группировка – пока я понимаю так, что если мы хотим группировать по определенным значениям, то это мы и выбираем в основном селекте

Вот еще пример сортировки

Тут какая тема, если например 20 товаров в 10 категориях, надо посчитать, сколько товаров всего в каждой из десяти категорий, для этого мы говорим, что выбираем категории и там суммируем товары в продаже из таблицы с продуктами, а потом группируем их по той же самой категории, (можно выбирать и другие еще столбики, а группировать также по категориям, тут как угодно), и в конечном счете сортируем также по категориям, чтобы видеть по порядку каждую

SELECT category\_id, SUM(units\_in\_stock)

FROM products

GROUP BY category\_id

ORDER BY category\_id

уже знаем, что для фильтрации запроса используем слово WHERE, но есть еще постфильтрация, это о чем:

после накладывания фильтра и группировки нам может потребоваться фильтрация еще раз, в этом случае надо прописать структурную часть HAVING - это т. н. вторичный фильтр

пример, посчитать сумму, на которую продается товар в отдельной категории и вывести те, где сумма будет более 5000

сначала мы выбираем категории и суммируем количество товара на стоимость, но берем только те, которые есть в продаже, далее всю эту тему мы группируем по категориям, и суммы разных продуктов собираются в свою категорию, после этого применяем постфильтр, который берет только те категории товаров, где общая сумма больше 5000, ну и сортирует их по категориям опять

SELECT category\_id, SUM(unit\_price \* units\_in\_stock)

FROM products

WHERE discontinued <> 1

GROUP BY category\_id

HAVING SUM(unit\_price \* units\_in\_stock) > 5000

ORDER BY category\_id

**объединения, пересечения и исключения - операции на множествах**

задача: вывести все страны, из которых у нас заказчики и работники, это две разные таблички, соединить их можно с помощью UNION. Юнион объединяет результаты и при этом убирает повторения. Если мы хотим, чтобы повторения были, надо использовать UNION ALL

SELECT country

FROM customers

UNION

SELECT country

FROM employees

ORDER BY country

Следующая операция – пересечение. Например, нам надо выбрать все страны, где находятся и заказчики, и доставщики продукции. Будут выбраны только они, остальные будут пропущены

SELECT country

FROM customers

INTERSECT

SELECT country

FROM suppliers

ORDER BY country

Следующая операция – исключение. Например, нам надо взять все страны, в которых находятся одни, но не находятся другие. EXCEPT берет из первого только те страны, которых нет во втором.

SELECT country

FROM customers

EXCEPT

SELECT country

FROM suppliers

ORDER BY country

Решения задач, остановился на 5

select order\_id, customer\_id, freight, ship\_country

from orders

where ship\_country like 'N%'

ORDER BY freight

LIMIT 10

SELECT first\_name, last\_name, home\_phone, region

FROM employees

WHERE region IS NULL

SELECT COUNT(order\_id)

FROM orders

WHERE ship\_region IS NULL

SELECT country, COUNT(\*)

FROM suppliers

GROUP BY country

ORDER BY COUNT(\*) DESC

6

select ship\_country, SUM(freight)

from orders

where ship\_region is not null

group by ship\_country

having SUM(freight) > 2750

order by SUM(freight)

7

SELECT DISTINCT(country)

FROM customers

UNION

SELECT DISTINCT(country)

FROM suppliers

ORDER BY country

8

SELECT country

FROM customers

INTERSECT

SELECT country

FROM suppliers

ORDER BY country

9

SELECT country

FROM customers

INTERSECT

SELECT country

FROM suppliers

EXCEPT

SELECT country

FROM EMPLOYEES

ORDER BY country