**2.11 Практика по темам «Группировка результатов, оператор GROUP BY» и «Объединение таблиц, JOIN-ы»**

Описание

В этом курсе мы предусмотрели опциональные мини-задания для самостоятельной практики. Их не нужно сдавать на проверку. Нажав на кнопку внизу страницы, вы найдёте алгоритм и советы по выполнению этого задания с перечнем использованных команд. Постарайтесь решить задание самостоятельно, но если возникнет вопрос, то вы всегда сможете посмотреть в эту подсказку.

В предыдущих видео вы узнали, как выводить уникальные записи по одному или нескольким полям, а также как группировать записи.

В этом задании вам необходимо выполнить несколько SQL-запросов к базе данных shop.sql:

1. Выведите ID статусов заказов, которые вообще встречаются в таблице order.
2. Выведите первые пять ID товаров и количество заказов, в которых они встречаются.
3. Выведите двумя способами все уникальные переходы статусов заказов.
4. Выведите имена статусов заказов и количества заказов, находящихся в этих статусах, в порядке возрастания количества.
5. Выведите имена пользователей, заказавших хотя бы раз «пуэр с молоком».
6. Выведите ID и названия товаров, которые никогда не заказывались.

Эталонное решение

1. Вывести ID статусов заказов, которые вообще встречаются в таблице order.

Результат выполнения запроса:



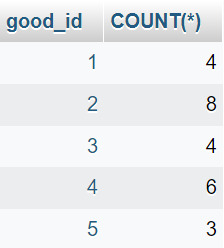
Запрос:

SELECT DISTINCT `status\_id`  
FROM `order`;

1. Вывести первые пять ID товаров и количество заказов, в которых они встречаются.

**Комментарий:** для выполнения этого задания используйте базу `order2good`. Здесь нужно использовать группировку и вывод общего количества элементов в группе.

Результат выполнения запроса:



Запрос:

SELECT  
 `good\_id`,  
  COUNT(\*)  
FROM `order2good`  
GROUP BY `good\_id`  
LIMIT 5;

1. Вывести двумя способами все уникальные переходы статусов заказов.

**Комментарий:** здесь понадобится таблица `order\_status\_change`. Все уникальные пары можно вернуть как с помощью оператора GROUP BY, так и с помощью DISTINCT.

Результат выполнения запроса:

Запрос:

SELECT  
  `src\_status\_id`,  
  `dst\_status\_id`  
FROM `order\_status\_change`  
GROUP BY `src\_status\_id`, `dst\_status\_id`;

И синонимичный запрос:

SELECT DISTINCT  
  `src\_status\_id`,  
  `dst\_status\_id`  
FROM `order\_status\_change`;

1. Вывести имена статусов заказов и количества заказов, находящихся в этих статусах, в порядке возрастания количества.

**Комментарий:** используйте таблицы `order` и `order\_status`. Здесь нужно воспользоваться тем, что можно обращаться к присвоенным именам для новых полей.

Результат выполнения запроса:

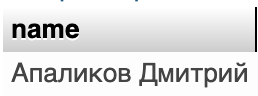
Запрос:

SELECT  
 os.`name`,  
 COUNT(\*) `count`  
FROM `order` o  
JOIN `order\_status` os ON  
 os.id = o.status\_id  
GROUP BY os.`name`  
ORDER BY `count`;

1. Вывести имена пользователей, заказавших хотя бы раз «пуэр с молоком».

**Комментарий:** используйте базы `user`, `order`, `order2good` и `good`. Сначала свяжите пользователей с заказами через первый JOIN. Второй используйте для связи с промежуточной таблицей, которая объединяет идентификаторы заказов и товаров. Третьим JOIN’ом свяжите заказы с товарами. Получив для каждого пользователя товары, которые они брали, отфильтруйте пользователей по тем, кто покупал «пуэр с молоком».

Результат выполнения запроса:



Запрос:

SELECT u.`name`  
FROM `user` u  
JOIN `order` o ON o.user\_id = u.id  
JOIN `order2good` o2g ON o2g.order\_id = o.id  
JOIN `good` g ON g.id = o2g.good\_id  
WHERE g.`name` LIKE '%пуэр с молоком%';

1. Вывести ID и названия товаров, которые никогда не заказывались.

**Комментарий:** используйте базы `good` и `order2good`. Посмотрите, какие товары при связке с промежуточной таблицей соответствия дают NULL в LEFT JOIN. LEFT JOIN оставит все элементы из левой таблицы (той, к которой присоединяете), в отличие от обычного JOIN (можете это проверить по общему количеству элементов в запросе). NULL как раз и означает, что товар никогда не покупался, потому что ему нет соответствия в каком-либо заказе.

Результат выполнения запроса:



Запрос:

SELECT  
  `id`,  
 g.`name`  
FROM `good` g  
LEFT JOIN `order2good` o2g ON  
 o2g.good\_id = g.id  
WHERE o2g.order\_id IS NULL;