Курс: Основы реляционных баз данных. MySQL
Урок 8. Сложные запросы
Выполнил: Кузнецов Сергей (Факультет Geek University Python-разработки)

Домашнее задание (все запросы на JOIN):

1. Определить кто больше поставил лайков (всего) - мужчины или женщины?
2. Подсчитать общее количество лайков десяти самым молодым пользователям (сколько лайков получили 10 самых молодых пользователей).
3. Найти 10 пользователей, которые проявляют наименьшую активность в использовании социальной сети (критерии активности необходимо определить самостоятельно).
4. Составьте список пользователей users, которые осуществили хотя бы один заказ orders в интернет магазине.
5. Выведите список товаров products и разделов catalogs, который соответствует товару.
6. (по желанию) Пусть имеется таблица рейсов flights (id, from, to) и таблица городов cities (label, name). Поля from, to и label содержат английские названия городов, поле name — русское. Выведите список рейсов flights с русскими названиями городов.

---

1. Определить кто больше поставил лайков (всего) - мужчины или женщины?

Работаем на БД, подготовленной в рамках 6-го урока «lesson_06». Соединим две таблицы «likes» и «profiles» для получения тип поставившего Лайк:

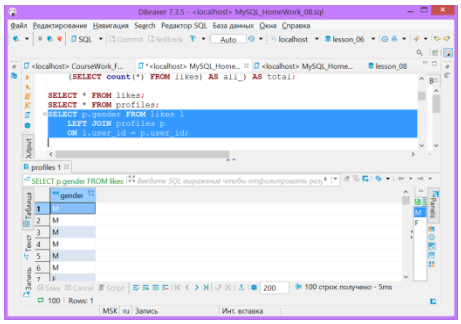
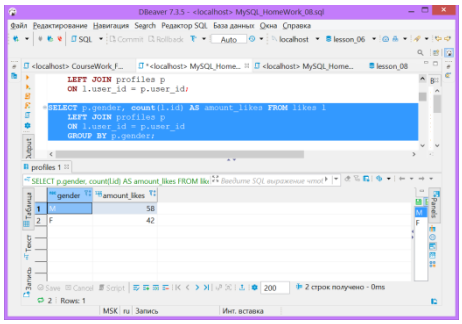
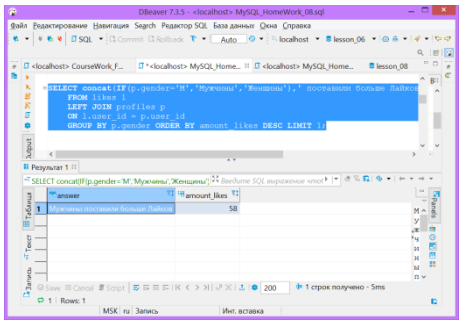
```
SELECT * FROM profiles;
SELECT p.gender FROM likes l
        LEFT JOIN profiles p
        ON l.user_id = p.user_id;
```

Произведем группировку и получим количество Лайков по каждому из полов:

```
SELECT p.gender, count(l.id) AS amount_likes FROM likes l
        LEFT JOIN profiles p
        ON l.user_id = p.user_id
        GROUP BY p.gender;
```

Отсортируем, возьмем максимальное и приведем ответ к «человеческому» виду:

```
SELECT concat(IF(p.gender='M','Мужчины','Женщины'),' поставили больше Лайков') AS answer,
        count(l.id) AS amount_likes
        FROM likes l
        LEFT JOIN profiles p
        ON l.user_id = p.user_id
        GROUP BY p.gender ORDER BY amount_likes DESC LIMIT 1;
```

2. Подсчитать общее количество лайков десяти самым молодым пользователям (сколько лайков получили 10 самых молодых пользователей).

«Старый» запрос с подзапросами (ответ – 7):

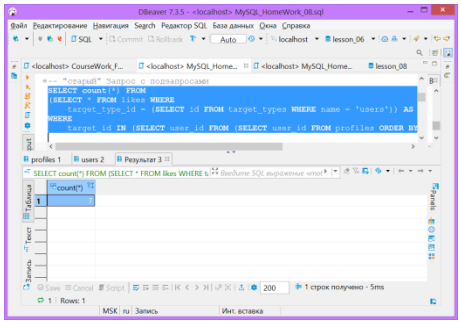
```
SELECT count(*) FROM
    (SELECT * FROM likes WHERE
        target_type_id = (SELECT id FROM target_types WHERE name = 'users')) AS ll
    WHERE
        target_id IN (SELECT user_id FROM
            (SELECT user_id FROM profiles ORDER BY birthday DESC LIMIT 10) ttt);
```

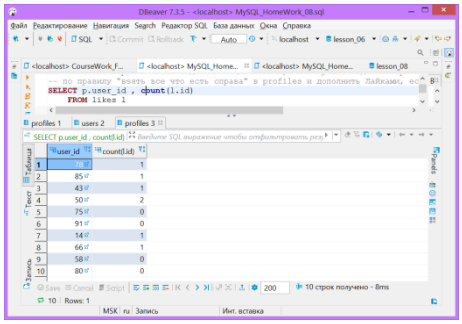
Добавим объединение с likes и target_types (порядок имеет значение). Сначала соберем все лайки для таблицы "users", потом уже объединим с profiles по правилу "взять все что есть справа" в profiles и дополнить Лайками, если есть:

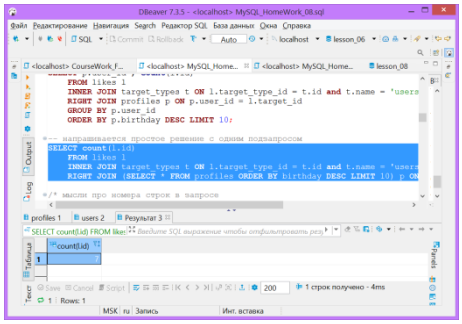
```
SELECT p.user_id , count(l.id)
    FROM likes l
    INNER JOIN target_types t ON l.target_type_id = t.id and t.name = 'users'
    RIGHT JOIN profiles p ON p.user_id = l.target_id
    GROUP BY p.user_id
    ORDER BY p.birthday DESC LIMIT 10;
```

Напрашивается простое решение с одним подзапросом *(ответ совпадает)*:

```
SELECT count(l.id)
    FROM likes l
    INNER JOIN target_types t ON l.target_type_id = t.id and t.name = 'users'
    RIGHT JOIN (SELECT * FROM profiles ORDER BY birthday DESC LIMIT 10) p ON p.user_id = l.target_id;
```

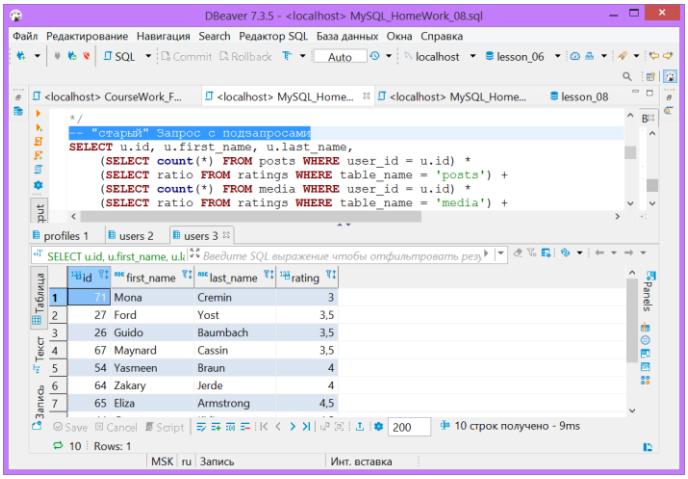






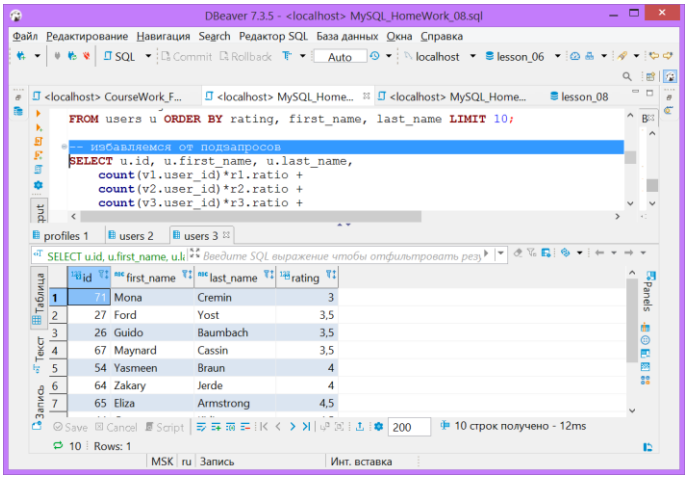
3. *Найти 10 пользователей, которые проявляют наименьшую активность в использовании социальной сети (критерии активности необходимо определить самостоятельно).*

Есть «старое» решение с хранимыми процедурами и альтернативное из подзапросов:




```
SELECT u.id, u.first_name, u.last_name,
        (SELECT count(*) FROM posts WHERE user_id = u.id) *
        (SELECT ratio FROM ratings WHERE table_name = 'posts') +
        (SELECT count(*) FROM media WHERE user_id = u.id) *
        (SELECT ratio FROM ratings WHERE table_name = 'media') +
        (SELECT count(*) FROM friendship WHERE user_id = u.id) *
        (SELECT ratio FROM ratings WHERE table_name = 'friendship'
AND table_col_name = 'user_id') +
        (SELECT count(*) FROM friendship WHERE friend_id = u.id) *
        (SELECT ratio FROM ratings WHERE table_name = 'friendship'
AND table_col_name = 'friend_id') +
        (SELECT count(*) FROM communities WHERE owner_id = u.id)
*
        (SELECT ratio FROM ratings WHERE table_name =
'communities') +
        (SELECT count(*) FROM communities_users WHERE user_id =
u.id) *
        (SELECT ratio FROM ratings WHERE table_name = 'communities_users') +
        (SELECT count(*) FROM likes WHERE user_id = u.id) *
        (SELECT ratio FROM ratings WHERE table_name = 'likes')
        AS rating
FROM users u ORDER BY rating, first_name, last_name LIMIT 10;
```

Избавляемся от подзапросов:




```
SELECT u.id, u.first_name, u.last_name,
        count(v1.user_id)*r1.ratio +
        count(v2.user_id)*r2.ratio +
        count(v3.user_id)*r3.ratio +
        count(v4.user_id)*r4.ratio +
        count(v5.owner_id)*r5.ratio +
        count(v6.user_id)*r6.ratio +
        count(v7.user_id)*r7.ratio
                AS `rating`
        FROM users u
        LEFT JOIN posts v1 ON v1.user_id = u.id
        JOIN ratings r1 ON r1.table_name = 'posts'
        LEFT JOIN media v2 ON v2.user_id = u.id
        JOIN ratings r2 ON r2.table_name = 'media'
        LEFT JOIN friendship v3 ON v3.user_id = u.id
        JOIN ratings r3 ON r3.table_name = 'friendship' AND r3.table_col_name = 'user_id'
        LEFT JOIN friendship v4 ON v4.friend_id = u.id
        JOIN ratings r4 ON r4.table_name = 'friendship' AND r4.table_col_name = 'friend_id'
        LEFT JOIN communities v5 ON v5.owner_id = u.id
        JOIN ratings r5 ON r5.table_name = 'communities'
        LEFT JOIN communities_users v6 ON v6.user_id = u.id
        JOIN ratings r6 ON r6.table_name = 'communities_users'
        LEFT JOIN likes v7 ON v7.user_id = u.id
        JOIN ratings r7 ON r7.table_name = 'likes'
        GROUP BY u.id
        ORDER BY rating, first_name, last_name LIMIT 10;
```

Все работает ничуть не хуже.

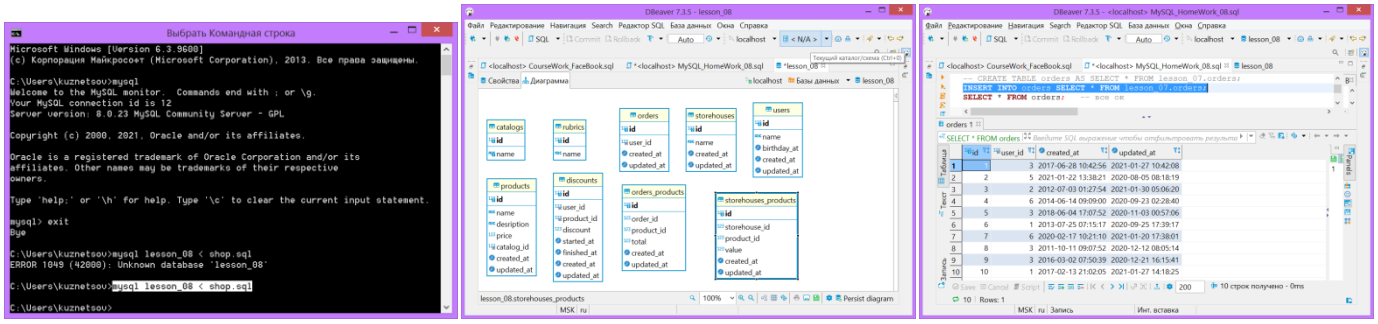
P.S. Дальше можно не смотреть, это все было представлено в ДЗ №7

4. *Составьте список пользователей users, которые осуществили хотя бы один заказ orders в интернет магазине.*

Создаем БД «*CREATE DATABASE lesson_08;*». В принципе, можно все и в БД «lesson_07», но потренируемся в копировании БД. Прогружаем в нее прилагаемый дамп «*mysql lesson_07 < shop.sql*».

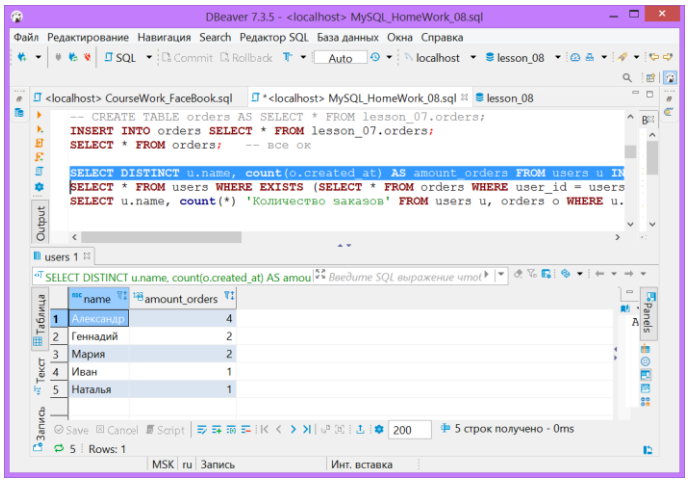
Убедимся, что база прогрузилась (просмотрим диаграмму в DBeaver)

Помня, что в таблице «users» данные присутствуют, а вот таблицу «orders» - пустая, не будем заполнять ее через http://filldb.info/, а скопируем из «lesson_07 - «*INSERT INTO orders SELECT * FROM lesson_07.orders;*» Учитывая то, что пользователей 7 и нужно показать, что не все делали заказ, заполним «orders» 10-ю записями – логичнее это было сделать вручную…



Решением задачи может быть несколько вариантов запросов, но нас интересует только на JOIN:

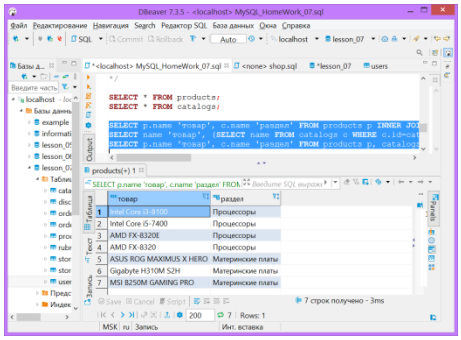
SELECT DISTINCT u.name, count(o.created_at) AS amount_orders
FROM users u INNER JOIN orders o ON u.id=o.user_id
GROUP BY u.id ORDER BY amount_orders DESC;



5. Выведите список товаров products и разделов catalogs, который соответствует товару.

Решением задачи может быть несколько вариантов запросов:

```
SELECT p.name 'товар', c.name 'раздел' FROM products p INNER JOIN catalogs c ON catalog_id=c.id;
SELECT name 'товар', (SELECT name FROM catalogs c WHERE c.id=catalog_id) 'раздел' FROM products;
SELECT p.name 'товар', c.name 'раздел' FROM products p, catalogs c WHERE catalog_id=c.id;
```

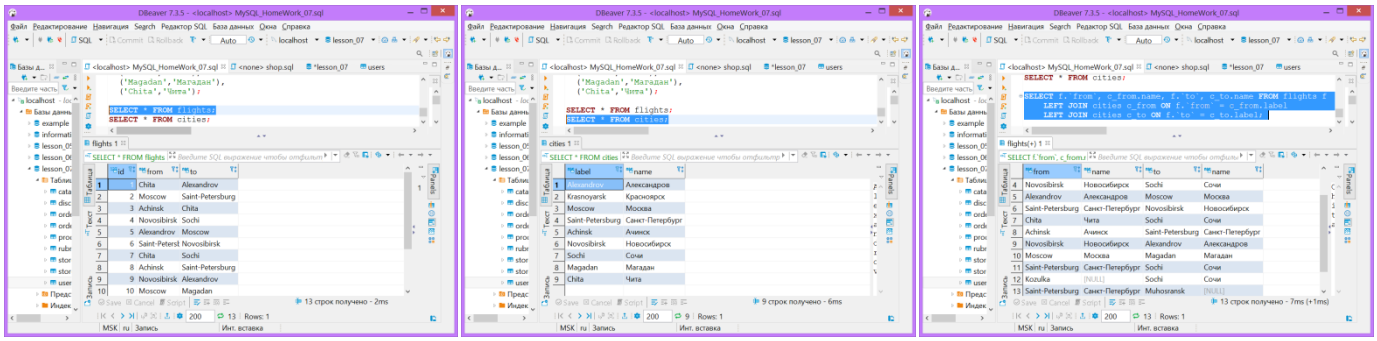


6. (по желанию) Пусть имеется таблица рейсов flights (id, from, to) и таблица городов cities (label, name). Поля from, to и label содержат английские названия городов, поле name — русское. Выведите список рейсов flights с русскими названиями городов.

Создаем и заполняем таблицы значениями. Не забываем, что служебные слова «from» , «to» в именах столбцов заключаем в обратные апострофы «`».

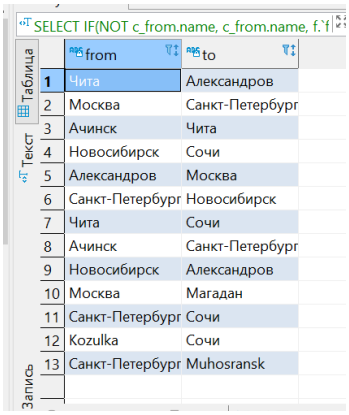
Оформляем запрос вывода списка рейсов с русскими названиями. Не забудем, что таблица городов может содержать не все названия и воспользуемся «*LEFT JOIN*». Обратите внимание на строчки с рейсами «Muhosransk» и «Kozulka»

```
SELECT f.`from`, c_from.name, f.`to`, c_to.name FROM flights f
        LEFT JOIN cities c_from ON f.`from` = c_from.label
        LEFT JOIN cities c_to ON f.`to` = c_to.label;
```



Немного модифицируем запрос, чтобы в случае отсутствия в справочнике городов перевода, система оставляла английское название:

```
SELECT IF(NOT c_from.name, c_from.name, f.`from`) `from`, IF(NOT c_to.name, c_to.name,f.`to`) `to` FROM flights f
        LEFT JOIN cities c_from ON f.`from` = c_from.label
        LEFT JOIN cities c_to ON f.`to` = c_to.label;
```



	from	to
1	Чита	Александров
2	Москва	Санкт-Петербург
3	Ачинск	Чита
4	Новосибирск	Сочи
5	Александров	Москва
6	Санкт-Петербург	Новосибирск
7	Чита	Сочи
8	Ачинск	Санкт-Петербург
9	Новосибирск	Александров
10	Москва	Магадан
11	Санкт-Петербург	Сочи
12	Kozulka	Сочи
13	Санкт-Петербург	Muhosransk