Проект: Продвинутый SQL

Задача проекта: проанализировать базу данных StackOverflow — сервис вопросов и ответов о программировании. StackOverflow похож на социальную сеть — пользователи сервиса задают вопросы, отвечают на посты, оставляют комментарии и ставят оценки другим ответам.

Описание данных:

ER-диаграмма

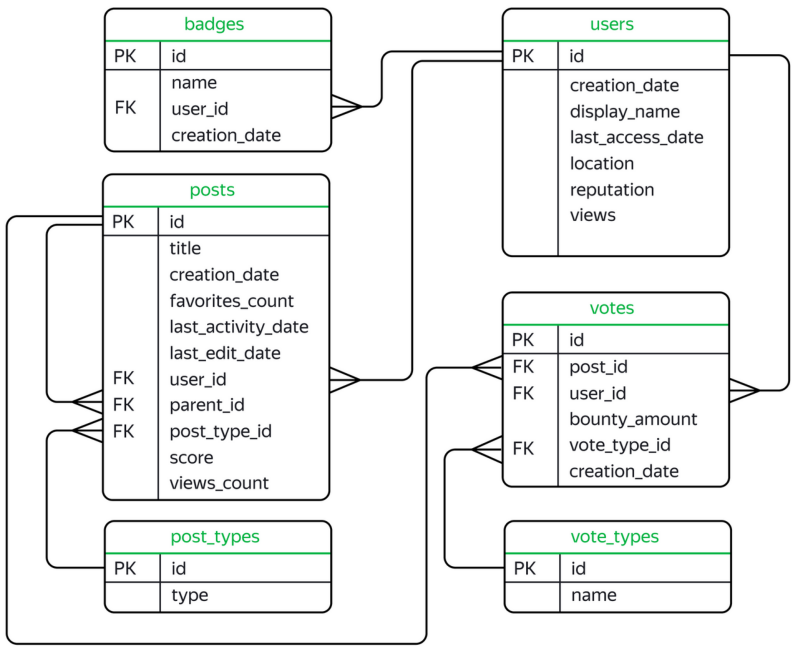

Таблица `stackoverflow.badges`

Хранит информацию о значках, которые присуждаются за разные достижения. Например, пользователь, правильно ответивший на большое количество вопросов про PostgreSQL, может получить значок postgresql.

Поле	Описание
id	Идентификатор значка, первичный ключ таблицы
name	Название значка
user_id	Идентификатор пользователя, которому присвоили значок, внешний ключ, отсылающий к таблице `users`
creation_date	Дата присвоения значка

Таблица `stackoverflow.post_types`

Содержит информацию о типе постов. Их может быть два:

- `Question` — пост с вопросом;
- `Answer` — пост с ответом.

Поле	Описание
id	Идентификатор поста, первичный ключ таблицы
type	Тип поста

Таблица `stackoverflow.posts`

Содержит информацию о постах.

Поле	Описание
id	Идентификатор поста, первичный ключ таблицы
title	Заголовок поста
creation_date	Дата создания поста
favorites_count	Число, которое показывает, сколько раз пост добавили в «Закладки»
last_activity_date	Дата последнего действия в посте, например комментария
last_edit_date	Дата последнего изменения поста
user_id	Идентификатор пользователя, который создал пост, внешний ключ к таблице `users`
parent_id	Если пост написали в ответ на другую публикацию, в это поле попадёт идентификатор поста с вопросом
post_type_id	Идентификатор типа поста, внешний ключ к таблице `post_types`
score	Количество очков, которое набрал пост

views_count	Количество просмотров

Таблица `stackoverflow.users`

Содержит информацию о пользователях.

Поле	Описание
id	Идентификатор пользователя, первичный ключ таблицы
creation_date	Дата регистрации пользователя
display_name	Имя пользователя
last_access_date	Дата последнего входа
location	Местоположение
reputation	Очки репутации, которые получают за хорошие вопросы и полезные ответы
views	Число просмотров профиля пользователя

Таблица `stackoverflow.vote_types`

Содержит информацию о типах голосов. Голос — это метка, которую пользователи ставят посту. Типов бывает несколько:

- `UpMod` — такую отметку получают посты с вопросами или ответами, которые пользователи посчитали уместными и полезными.
- `DownMod` — такую отметку получают посты, которые показались пользователям наименее полезными.
- `Close` — такую метку ставят опытные пользователи сервиса, если заданный вопрос нужно доработать или он вообще не подходит для платформы.
- `Offensive` — такую метку могут поставить, если пользователь ответил на вопрос в грубой и оскорбительной манере, например, указав на неопытность автора поста.
- `Spam` — такую метку ставят в случае, если пост пользователя выглядит откровенной рекламой.

Поле	Описание
id	Идентификатор типа голоса, первичный ключ

name	Название метки

Таблица `stackoverflow.votes`

Содержит информацию о голосах за посты.

Поле	Описание
id	Идентификатор голоса, первичный ключ
post_id	Идентификатор поста, внешний ключ к таблице `posts`
user_id	Идентификатор пользователя, который поставил посту голос, внешний ключ к таблице `users`
bounty_amount	Сумма вознаграждения, которое назначают, чтобы привлечь внимание к посту
vote_type_id	Идентификатор типа голоса, внешний ключ к таблице `vote_types`
creation_date	Дата назначения голоса

Задания (Первая часть)

1. Найдите количество вопросов, которые набрали больше 300 очков или как минимум 100 раз были добавлены в «Закладки».

```
SELECT COUNT(post_type_id)
FROM stackoverflow.posts p
JOIN stackoverflow.post_types pt ON p.post_type_id=pt.id
WHERE type = 'Question' AND (score > 300 OR favorites_count >= 100);
```

count

1355

2. Сколько в среднем в день задавали вопросов с 1 по 18 ноября 2008 включительно? Результат округлите до целого числа.

```
WITH profile AS
(SELECT p.id,
```

```
        DATE_TRUNC('day', p.creation_date)::date as day
FROM stackoverflow.posts p
JOIN stackoverflow.post_types pt ON p.post_type_id=pt.id
WHERE (p.creation_date::date BETWEEN '2008.11.01' AND '2008.11.18') AND pt.type
= 'Question'),
counts AS
(SELECT COUNT(id) as count,
        day
FROM profile pr
GROUP BY day)
SELECT ROUND(AVG(count))
FROM counts
```

round

383

3. Сколько пользователей получили значки сразу в день регистрации? Выведите количество уникальных пользователей.

```
SELECT COUNT(DISTINCT u.id)
FROM stackoverflow.users u
JOIN stackoverflow.badges b ON u.id=b.user_id
WHERE u.creation_date::date = b.creation_date::date;
```

count

7047

4. Сколько уникальных постов пользователя с именем Joel Coehoorn получили хотя бы один голос?

```
SELECT COUNT(DISTINCT p.id)
FROM stackoverflow.posts p
```

```
JOIN stackoverflow.users u ON p.user_id=u.id
JOIN stackoverflow.votes v ON p.id=v.user_id
WHERE u.display_name='Joel Coehoorn';
```

count
12

5. Выгрузите все поля таблицы `vote_types`. Добавьте к таблице поле `rank`, в которое войдут номера записей в обратном порядке. Таблица должна быть отсортирована по полю `id`.

```
SELECT *,
        ROW_NUMBER() OVER(ORDER BY id DESC) as rank
FROM stackoverflow.vote_types
ORDER BY rank DESC;
```

id	name	rank
1	AcceptedByOriginator	15
2	UpMod	14
3	DownMod	13
…	…	…

6. Отберите 10 пользователей, которые поставили больше всего голосов типа `Close`. Отобразите таблицу из двух полей: идентификатором пользователя и количеством голосов. Отсортируйте данные сначала по убыванию количества голосов, потом по убыванию значения идентификатора пользователя.

```
SELECT u.id,
        COUNT(v.vote_type_id) as counts
FROM stackoverflow.users u
JOIN stackoverflow.votes v ON u.id=v.user_id
```

```
JOIN stackoverflow.vote_types vt ON v.vote_type_id=vt.id
WHERE vt.name = 'Close'
GROUP BY u.id
ORDER BY counts DESC, u.id DESC
LIMIT 10;
```

id	counts
20646	36
14728	36
27163	29
…	…

7. Отберите 10 пользователей по количеству значков, полученных в период с 15 ноября по 15 декабря 2008 года включительно.
   Отобразите несколько полей:

- идентификатор пользователя;
- число значков;
- место в рейтинге — чем больше значков, тем выше рейтинг.

Пользователям, которые набрали одинаковое количество значков, присвойте одно и то же место в рейтинге.

Отсортируйте записи по количеству значков по убыванию, а затем по возрастанию значения идентификатора пользователя.

```
WITH profile AS
(SELECT u.id,
            COUNT(b.id) as counts
 FROM stackoverflow.users u
 JOIN stackoverflow.badges b ON u.id=b.user_id
 WHERE b.creation_date::date BETWEEN '2008.11.15' AND '2008.12.15'
 GROUP BY u.id)
```

```
SELECT *,
          DENSE_RANK() OVER(ORDER BY counts DESC)
FROM profile
```

```
ORDER BY counts DESC, id
LIMIT 10;
```

id	counts	dense_rank
22656	149	1
34509	45	2
1288	40	3
…	…	…

8. Сколько в среднем очков получает пост каждого пользователя?

Сформируйте таблицу из следующих полей:

- заголовок поста;
- идентификатор пользователя;
- число очков поста;
- среднее число очков пользователя за пост, округлённое до целого числа.

Не учитывайте посты без заголовка, а также те, что набрали ноль очков.

```
SELECT p.title,
        u.id,
        p.score,
        ROUND(AVG(p.score) OVER(PARTITION BY u.id))
FROM stackoverflow.posts p
JOIN stackoverflow.users u ON p.user_id=u.id
WHERE p.score!= 0 AND p.title != '';
```

title	id	score	round
Diagnosing Deadlocks in SQL Server 2005	1	82	573
How do I calculate someone's age in C#?	1	1743	573
Why doesn't IE7 copy <pre><code> blocks to the clipboard correctly?	1	37	573

…	…	…	…

9. Отобразите заголовки постов, которые были написаны пользователями, получившими более 1000 значков. Посты без заголовков не должны попасть в список.

```
WITH profile as
(SELECT u.id as id_pr,
        COUNT(b.id) as counts
 FROM stackoverflow.badges b
 JOIN stackoverflow.users u ON u.id = b.user_id
 GROUP BY u.id)
```

```
SELECT p.title
FROM stackoverflow.posts p
JOIN profile pr ON pr.id_pr=p.user_id
WHERE counts > 1000 AND p.title != '';
```

title
What's the hardest or most misunderstood aspect of LINQ?
What's the strangest corner case you've seen in C# or .NET?
Project management to go with GitHub
…

10. Напишите запрос, который выгрузит данные о пользователях из США (англ. United States). Разделите пользователей на три группы в зависимости от количества просмотров их профилей:

- пользователям с числом просмотров больше либо равным 350 присвойте группу `1`;
- пользователям с числом просмотров меньше 350, но больше либо равно 100 — группу `2`;
- пользователям с числом просмотров меньше 100 — группу `3`.

Отобразите в итоговой таблице идентификатор пользователя, количество просмотров профиля и группу. Пользователи с нулевым количеством просмотров не должны войти в итоговую таблицу.

```
SELECT id,
       views,
       CASE
          WHEN views >= 350 THEN 1
          WHEN views < 350 AND views >= 100 THEN 2
          WHEN views < 100 THEN 3
       END
FROM stackoverflow.users
WHERE location LIKE '%United States%' and views != 0;
```

id	views	case
3	24396	1
13	35414	1
23	757	1
…	…	…

11. Дополните предыдущий запрос. Отобразите лидеров каждой группы — пользователей, которые набрали максимальное число просмотров в своей группе. Выведите поля с идентификатором пользователя, группой и количеством просмотров. Отсортируйте таблицу по убыванию просмотров, а затем по возрастанию значения идентификатора.

```
SELECT DISTINCT id,
   CASE
      WHEN views>=350 THEN 1
      WHEN views>=100 THEN 2
      ELSE 3
   END groupss, views
FROM stackoverflow.users
WHERE views IN (SELECT max(views) mv
```

```
                    FROM (SELECT id, views,
                                CASE
                                   WHEN views>=350 THEN 1
                                   WHEN views>=100 THEN 2
                                   ELSE 3
                                END groupss
FROM stackoverflow.users
WHERE views!=0 AND location like '%United States%') q1
GROUP BY groupss) AND location like '%United States%'
ORDER BY 3 desc, 1
```

id	groupss	views
16587	1	62813
9094	2	349
9585	2	349
…	…	…

12. Посчитайте ежедневный прирост новых пользователей в ноябре 2008 года. Сформируйте таблицу с полями:

- номер дня;
- число пользователей, зарегистрированных в этот день;
- сумму пользователей с накоплением.

```
WITH profile as
(SELECT EXTRACT(DAY FROM creation_date::date) as dt,
            COUNT(id) as counts
FROM stackoverflow.users
WHERE creation_date::date BETWEEN '2008.11.01' AND '2008.11.30'
GROUP BY dt)
```

```
SELECT dt,
            counts,
            SUM(counts) OVER(ORDER BY dt)
FROM profile;
```

dt	counts	sum
1	34	34
2	48	82
3	75	157
…	…	…

13. Для каждого пользователя, который написал хотя бы один пост, найдите интервал между регистрацией и временем создания первого поста. Отобразите:

- идентификатор пользователя;
- разницу во времени между регистрацией и первым постом.

```
WITH profile as
(SELECT p.user_id,
            u.creation_date,
            MIN(p.creation_date) OVER(PARTITION BY p.user_id ORDER BY
p.creation_date) as mins
 FROM stackoverflow.posts p
 JOIN stackoverflow.users u ON p.user_id=u.id)
```

```
SELECT DISTINCT user_id,
           pr.mins - pr.creation_date as date
FROM profile pr
```

user_id	date
27088	22 days, 10:32:25
4666	4 days, 13:51:01
43473	0:00:00
…	…

Задания (Вторая часть)

1. Выведите общую сумму просмотров постов за каждый месяц 2008 года. Если данных за какой-либо месяц в базе нет, такой месяц можно пропустить. Результат отсортируйте по убыванию общего количества просмотров.

```
SELECT DATE_TRUNC('month', creation_date)::date,
           SUM(views_count)
FROM stackoverflow.posts
WHERE creation_date::date BETWEEN '2008.01.01' AND '2008.12.31'
GROUP BY DATE_TRUNC('month', creation_date)::date
ORDER BY SUM(views_count) DESC;
```

date_trunc	sum
2008-09-01	452928568
2008-10-01	365400138
2008-11-01	221759651
…	…

2. Выведите имена самых активных пользователей, которые в первый месяц после регистрации (включая день регистрации) дали больше 100 ответов. Вопросы, которые задавали пользователи, не учитывайте. Для каждого имени пользователя выведите количество уникальных значений `user_id`. Отсортируйте результат по полю с именами в лексикографическом порядке.

```
SELECT u.display_name,
           COUNT(DISTINCT u.id)
FROM stackoverflow.users u
JOIN stackoverflow.posts p ON u.id=p.user_id
JOIN stackoverflow.post_types pt ON p.post_type_id=pt.id
WHERE pt.type = 'Answer' AND (p.creation_date::date >= u.creation_date::date AND
p.creation_date::date <= u.creation_date::date + interval '1 month')
GROUP BY u.display_name
HAVING count(p.id) > 100
ORDER BY 1;
```

display_name	count
1800 INFORMATION	1
Adam Bellaire	1

Adam Davis	1
…	…

3. Выведите количество постов за 2008 год по месяцам. Отберите посты от пользователей, которые зарегистрировались в сентябре 2008 года и сделали хотя бы один пост в декабре того же года. Отсортируйте таблицу по значению месяца по убыванию.

```
SELECT DISTINCT
        DATE_TRUNC ('month', p.creation_date)::date dt,
        COUNT(p.id) OVER (PARTITION BY DATE_TRUNC ('month',
p.creation_date)::date)
FROM stackoverflow.posts p
JOIN stackoverflow.users u ON p.user_id = u.id
WHERE u.id IN (SELECT u.id
                FROM stackoverflow.users u
                JOIN stackoverflow.posts p ON u.id = p.user_id
                WHERE u.creation_date::date BETWEEN '2008-09-01' AND '2008-09-
30'
                    AND p.creation_date::date BETWEEN '2008-12-01' AND '2008-12-
31')
ORDER BY dt DESC;
```

dt	count
2008-12-01	17641
2008-11-01	18294
2008-10-01	27171
…	…

4. Используя данные о постах, выведите несколько полей:

- идентификатор пользователя, который написал пост;
- дата создания поста;
- количество просмотров у текущего поста;

- сумму просмотров постов автора с накоплением.

Данные в таблице должны быть отсортированы по возрастанию идентификаторов пользователей, а данные об одном и том же пользователе — по возрастанию даты создания поста.

```
SELECT p.user_id,
            p.creation_date,
            p.views_count,
            SUM(views_count) OVER(PARTITION BY p.user_id ORDER BY
p.creation_date)
FROM stackoverflow.posts p
ORDER BY 1
```

user_id	creation_date	views_count	sum
1	2008-07-31 23:41:00	480476	480476
1	2008-07-31 23:55:38	136033	616509
1	2008-07-31 23:56:41	0	616509
…	…	…	…

5. Сколько в среднем дней в период с 1 по 7 декабря 2008 года включительно пользователи взаимодействовали с платформой? Для каждого пользователя отберите дни, в которые он или она опубликовали хотя бы один пост. Нужно получить одно целое число — не забудьте округлить результат.

```
WITH profile AS
(SELECT user_id,
             COUNT(DISTINCT creation_date::date) counts
 FROM stackoverflow.posts
 WHERE creation_date::date BETWEEN '2008.12.01' AND '2008.12.07'
 GROUP BY user_id)
```

```
SELECT ROUND(AVG(counts))
FROM profile
```

round

2

6. На сколько процентов менялось количество постов ежемесячно с 1 сентября по 31 декабря 2008 года? Отобразите таблицу со следующими полями:

- номер месяца;
- количество постов за месяц;
- процент, который показывает, насколько изменилось количество постов в текущем месяце по сравнению с предыдущим.

Если постов стало меньше, значение процента должно быть отрицательным, если больше — положительным. Округлите значение процента до двух знаков после запятой.

Напомним, что при делении одного целого числа на другое в PostgreSQL в результате получится целое число, округлённое до ближайшего целого вниз. Чтобы этого избежать, переведите делимое в тип `numeric`.

```
WITH a AS
(SELECT EXTRACT(MONTH FROM creation_date) as month,
            COUNT(id) as count_id
 FROM stackoverflow.posts
 WHERE creation_date::date BETWEEN '2008.09.01' AND '2008.12.31'
 GROUP BY month),
```

```
b AS
(SELECT *,
            LAG(count_id) OVER (ORDER BY month) AS previous_cnt,
            ((count_id::numeric / LAG(count_id) OVER (ORDER BY month)) - 1) * 100 AS
growth
 FROM a)
```

```
SELECT month, count_id, ROUND(growth,2)
FROM b;
```

month	count_id	round

9	70371	
10	63102	-10.33
11	46975	-25.56
12	44592	-5.07

7. Выгрузите данные активности пользователя, который опубликовал больше всего постов за всё время. Выведите данные за октябрь 2008 года в таком виде:

- номер недели;
- дата и время последнего поста, опубликованного на этой неделе.

```
WITH profile AS
(SELECT user_id,
        COUNT(id),
        EXTRACT(WEEK FROM creation_date::date) as week,
        creation_date
 FROM stackoverflow.posts
 WHERE user_id = 22656 AND creation_date::date BETWEEN '2008.10.01' AND
'2008.10.31'
 GROUP BY user_id, week, creation_date)
```

```
SELECT DISTINCT week,
LAST_VALUE(creation_date) OVER(ORDER BY week)
FROM profile;
```

week	last_value
43	2008-10-26 21:44:36
40	2008-10-05 09:00:58
44	2008-10-31 22:16:01
42	2008-10-19 06:49:30
41	2008-10-12 21:22:23