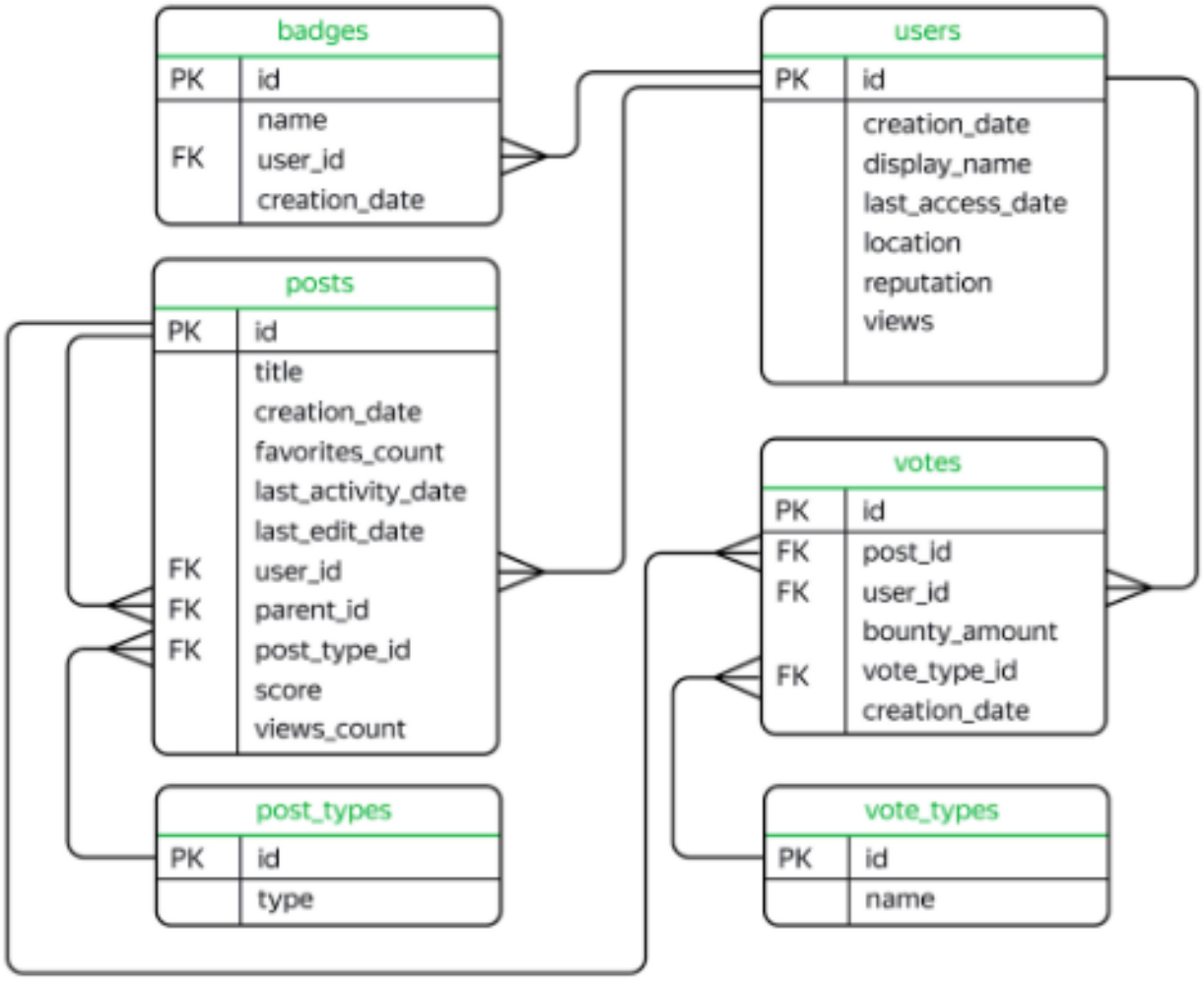
# Описание данных

Проект состоит из двух частей:

Внимательно читайте условия.

В самостоятельном проекте вы будете работать с базой данных StackOverflow — сервиса вопросов и ответов о программировании. StackOverflow похож на социальную сеть — пользователи сервиса




задают

вопросы, отвечают на посты, оставляют комментарии и ставят оценки другим ответам.

Вы будете работать с версией базы, где хранятся данные о постах за 2008 год, но в таблицах вы найдёте информацию и о более поздних оценках, которые эти посты получили.

Изучите ER-диаграмму базы:

Теперь познакомьтесь с данными таблиц.

`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 типа 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 |

## ПРОЕКТ ЧАСТЬ 1

1.Найдите количество вопросов, которые набрали больше 300 очков или как минимум 100 раз были добавлены в «Закладки». Подсказка. Чтобы отфильтровать данные, используйте логические операторы `AND` и `OR`. Обратите внимание на приоритет выполнения этих операторов.

```
SELECT COUNT(id)
FROM stackoverflow.posts
WHERE (favorites_count >= 100 OR score > 300) -- условие набрали больше 300 очков или как минимум 100 раз были добавлены в «Закладки»
AND post_type_id = 1; -- тип вопрос или ответ id=1 это вопрос
```

Результат

| count |
|---|
| 1355 |

2.Сколько в среднем в день задавали вопросов с 1 по 18 ноября 2008 включительно? Результат округлите до целого числа. Подсказка/ Делая срез, обратите внимание на тип данных поля с датой поста. Сгруппируйте записи по дням и найдите количество вопросов в день, а после посчитайте среднее.

```
WITH questions AS
(SELECT DISTINCT DATE_TRUNC('day', creation_date)::date,
      COUNT(p.id) OVER (PARTITION BY DATE_TRUNC('day', creation_date)::date)
FROM stackoverflow.posts AS p
LEFT JOIN stackoverflow.post_types AS pt
      ON p.post_type_id = pt.id
WHERE pt.type = 'Question' AND (DATE_TRUNC('day', creation_date)::date BETWEEN '2008-11-01'
AND '2008-11-18')
)
SELECT ROUND(AVG(count))::int
FROM questions;
```

Результат

| round |
|---|
| 383 |

3.Сколько пользователей получили значки сразу в день регистрации? Выведите количество уникальных пользователей.Подсказка. Данные о пользователях хранит таблица `users`, а данные о значках — таблица `badges`. Соедините их.

```
SELECT COUNT(DISTINCT(u.id))
FROM stackoverflow.users u
JOIN stackoverflow.badges b
ON u.id = b.user_id
WHERE CAST(DATE_TRUNC('day', u.creation_date) AS date) = CAST(DATE_TRUNC('day', b.creation_date) AS date);
```

| count |
|---|
| 7047 |

4.Сколько уникальных постов пользователя с именем Joel Coehoorn получили хотя бы один голос? Подсказка Присоедините несколько таблиц, чтобы собрать все нужные данные.

```
SELECT COUNT(DISTINCT(p.id)) -- считаем уникальные id постов из posts
FROM stackoverflow.users AS u
JOIN stackoverflow.posts p ON u.id = p.user_id --innerджойним таблицу posts к users  остаются только те p.id которые  имеют посты в posts
RIGHT JOIN stackoverflow.votes AS v ON p.id = v.post_id -- к votes джойним. получается отсекаем тех у кого нет войсов
WHERE u.display_name = 'Joel Coehoorn';  -- условие с именем Joel Coehoorn
```

Результат

| 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`.
Подсказка Чтобы пронумеровать записи в обратном порядке, используйте оконную функцию.

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

| id | name | rank |
|---|---|---|
| 1 | AcceptedByOriginator | 15 |
| 2 | UpMod | 14 |

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. Подсказка.Таблицы `vote_types` и `users` не связаны напрямую, поэтому вам понадобится присоединить несколько таблиц. Не забудьте добавить условие.

```
SELECT v.user_id AS user_votes,
       COUNT (v.id) AS votes_cnt
FROM stackoverflow.votes AS v
JOIN stackoverflow.vote_types AS vt
ON vt.id = v.vote_type_id
WHERE name = 'Close'
GROUP BY user_votes
ORDER BY votes_cnt DESC,
         user_votes DESC
LIMIT 10;
```

| user_votes | votes_cnt |
|---|---|
| 20646 | 36 |
| 14728 | 36 |

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.Подсказка.Чтобы назначить места в рейтинге, воспользуйтесь оконной функцией ранжирования.

```
SELECT user_id,
       COUNT (id),
       DENSE_RANK () OVER (ORDER BY COUNT (id) DESC)
FROM stackoverflow.badges
WHERE creation_date::date BETWEEN '2008-11-15' AND '2008-12-15'
GROUP BY user_id
ORDER BY COUNT (id) DESC,
         user_id
LIMIT 10;
```

| user_id | count | dense_rank |
|---|---|---|
| 22656 | 149 | 1 |
| 34509 | 45 | 2 |

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. Подсказка Используйте оконную функцию и укажите поле, по которому сформировать окна.

```
SELECT title, user_id, score,
       ROUND(AVG(score) OVER (PARTITION BY user_id))::int
FROM stackoverflow.posts
WHERE title IS NOT NULL
      AND score != 0;
```

| title | user_id | score | round |
|---|---|---|---|
| Diagnosing Deadlocks in SQL Server 2005 | 1 | 82 | 573 |
| How do I calculate someone's age in C#? | 1 | 1743 | 573 |

9. Отобразите заголовки постов, которые были написаны пользователями, получившими более 1000 значков. Посты без заголовков не должны попасть в список. Подсказка.Это задание лучше выполнить по частям. Сформируйте список пользователей, которые заработали больше 1000 значков. С помощью этого списка можно отфильтровать записи в основном запросе.

```
SELECT title
FROM stackoverflow.posts
WHERE title IS NOT NULL AND
      user_id IN (
            SELECT user_id
            FROM stackoverflow.badges
            GROUP BY user_id
            HAVING COUNT(id) > 1000) ;
```

| title |
|---|
| What's the strangest corner case you've seen in C# or .NET? |
| What's the hardest or most misunderstood aspect of LINQ? |
| What are the correct version numbers for C#? |
| Project management to go with GitHub |

10.Напишите запрос, который выгрузит данные о пользователях из Канады (англ. Canada). 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 количеством просмотров меньше либо равным нулю не должны войти в итоговую таблицу. Подсказка. Чтобы создать категории пользователей, используйте оператор `CASE`. Отфильтруйте данные по стране, пользуясь оператором `LIKE`. Данные неидеальны: перед словом для поиска и после него могут встречаться лишние пробелы.

```
SELECT id,
       views,
       CASE
         WHEN views < 100 THEN 3
         WHEN views >= 100 AND views < 350 THEN 2
         ELSE 1
       END AS group
FROM stackoverflow.users
WHERE location LIKE '%Canada%' AND views != 0
ORDER BY views DESC;
```

| id | views | group |
|---|---|---|
| 3153 | 21991 | 1 |
| 7552 | 10981 | 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. Подсказка.Добавьте предыдущий запрос в подзапрос. Посчитайте максимальное количество просмотров по категориям. В список должны попасть пользователи, у которых число просмотров равно максимальному значению.

```
WITH us_users AS
(SELECT id AS user_id,
       views AS views_cnt,
       CASE
         WHEN views < 100 THEN 3
         WHEN views >= 100 AND views < 350  THEN 2
         ELSE 1
       END AS groups
FROM stackoverflow.users
WHERE location LIKE '%Canada%' AND views != 0
)

SELECT user_id,
      groups,
      views_cnt
FROM (
        SELECT user_id,
              views_cnt,
              groups,
              MAX(views_cnt) OVER (PARTITION BY groups
ORDER BY views_cnt DESC) AS max_views
           FROM us_users
       ) AS max_us
WHERE views_cnt =  max_views
ORDER BY views_cnt DESC, user_id;
```

| user_id | groups | views_cnt |
|---|---|---|
| 3153 | 1 | 21991 |
| 46981 | 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. Подсказка. Для подсчёта суммы с накоплением вам понадобится оконная функция. Не забудьте отфильтровать таблицу по месяцу и году.

```
WITH t1 AS
(
SELECT CAST(DATE_TRUNC('day', creation_date) AS date) AS days,
       COUNT(id) AS users_cnt
FROM stackoverflow.users
GROUP BY CAST(DATE_TRUNC('day', creation_date) AS date)
ORDER BY CAST(DATE_TRUNC('day', creation_date) AS date)
)

SELECT RANK() OVER (ORDER BY days),
       users_cnt,
       SUM(users_cnt) OVER (ORDER BY days)::int AS cum
FROM t1
WHERE CAST(DATE_TRUNC('day', days) AS date) BETWEEN '2008-11-01' AND '2008-11-30';
```

| rank | users_cnt | cum |
|---|---|---|
| 1 | 34 | 34 |
| 2 | 48 | 82 |

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. Подсказка. Для каждого пользователя найдите время создания первого поста с помощью оконной функции ранжирования. Если от этого времени отнять дату регистрации пользователя, получится нужный интервал. Не меняйте тип данных поля `creation_date`.

```
WITH p AS
(
SELECT user_id,
      creation_date,
      RANK() OVER (PARTITION BY user_id ORDER BY creation_date) AS first_pub
FROM stackoverflow.posts
ORDER BY user_id
)

SELECT user_id,
      p.creation_date - u.creation_date AS delta
FROM p
JOIN stackoverflow.users AS u
ON p.user_id = u.id
WHERE first_pub = 1;
```

| user_id | delta |
|---|---|
| 1 | 9:18:29 |
| 2 | 14:37:03 |
| 3 | 3 days, 16:17:09 |

# Задания (Вторая часть)

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

Подсказка. Используйте функцию для усечения даты, а затем сгруппируйте и отсортируйте данные.

```
SELECT DATE_TRUNC('month', creation_date)::date AS month_date,
       SUM(views_count) AS total_views
FROM stackoverflow.posts
WHERE EXTRACT(YEAR FROM creation_date) = 2008
GROUP BY DATE_TRUNC('month', creation_date)
ORDER BY SUM(views_count) DESC;
```

| month_date | total_views |
|---|---|
| 2008-09-01 | 452928568 |
| 2008-10-01 | 365400138 |
| 2008-11-01 | 221759651 |
| 2008-12-01 | 197792841 |
| 2008-08-01 | 131367083 |
| 2008-07-01 | 669895 |

Обратите внимание, что данные отличаются. Возможно, повышенная активность в сентябре и октябре связана с началом учебного года. Малая активность в июле может свидетельствовать о неполноте данных.

*ВЫВОДЫ:Данные за разные месяцы отличаются. Наименьшее количество просмотров в июле связано с тем, что публичный доступ к сайту был открыт в конце июля 2008 года. Рост числа просмотров к сентябрю связан с ростом популярности сайта. Падение количества просмотров с сентября - меньше уникальных ситуаций возникает у программистов, с ними уже кто-то сталкивался и они решены ранее.*

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.
Подсказка
Вам нужно присоединить несколько таблиц — изучите внимательнее описание базы. Чтобы добавить промежуток времени к дате, используйте ключевое слово `INTERVAL`, например, так: `<дата> + INTERVAL '1 year 2 months 3 days'`.

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

| display_name | count |
|---|---|
| 1800 INFORMATION | 1 |
| Adam Bellaire | 1 |
| Adam Davis | 1 |
| Adam Liss | 1 |
| aku | 1 |
| Alan | 8 |
| Amy B | 1 |
| anjanb | 1 |
| Ben Hoffstein | 1 |
| Brian | 15 |

Кажется, что одному имени пользователя должен соответствовать один `user_id`. Но это не так: многим популярным именам вроде `Alan`, `Dan` или `Chris` соответствует несколько значений `user_id`. Данные лучше не анализировать по имени, иначе результаты будут некорректными.
*ВЫВОДЫ:*
*Какие аномалии наблюдаются в данных? О чём они говорят?*

```
anomalies = part_2_question_2.loc[part_2_question_2['count'] > 1].sort_values(by='count', ascending=False)  \
.reset_index(drop=True)
anomalies.style.hide_index()
anomalies['count'].sum()
```

*255*
*В данных наблюдаются следующие аномалии - количество user_id, которые превышают значение 1. Это говорит о том, что есть пользователи с одинаковыми именами (display_name). По данным таблицы видны случаи когда на 19 имен пользователей приходится 255 уникальных user_id.*

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.
Подсказка. Сначала найдите идентификаторы пользователей, которые зарегистрировались в сентябре 2008 года и оставили хотя бы один пост в декабре. Затем используйте результат для среза и посчитайте посты по месяцам.

```
WITH users AS
(
SELECT u.id
FROM stackoverflow.posts AS p
JOIN stackoverflow.users AS u
ON p.user_id=u.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')
GROUP BY u.id
)

SELECT DATE_TRUNC('month', p.creation_date)::date AS month,
       COUNT(p.id)
FROM stackoverflow.posts AS p
WHERE p.user_id IN
(
SELECT *
FROM users
)
      AND DATE_TRUNC('year', p.creation_date)::date = '2008-01-01'
GROUP BY DATE_TRUNC('month', p.creation_date)::date
ORDER BY DATE_TRUNC('month', p.creation_date)::date DESC;
```

| month | count |
|---|---|
| 2008-12-01 | 17641 |
| 2008-11-01 | 18294 |
| 2008-10-01 | 27171 |
| 2008-09-01 | 24870 |
| 2008-08-01 | 32 |

В итоговой таблице встречаются аномальные значения: пользователи, зарегистрированные в сентябре, были активны и в августе. Возможно, это ошибка в данных.
ВЫВОДЫ:Замечено ,что существуют аномальные значения постов в августе. В условии сказано, что пользователи зарегистрировались в сентябре, поэтому посты от них не могли появиться в августе, скорее всего присутствует техническая ошибка.

4.Используя данные о постах, выведите несколько полей: идентификатор пользователя, который написал пост; дата создания поста; количество просмотров у текущего поста; сумма 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.
Подсказка. Для подсчёта суммы с накоплением используйте оконную функцию.

```
SELECT user_id, creation_date, views_count,
    SUM (views_count) OVER (PARTITION BY user_id ORDER BY creation_date) AS cumulative_count
FROM stackoverflow.posts
ORDER BY user_id, creation_date;
```

В теории все расчёты с оконной функцией можно выполнить и без неё. Но размер запроса имеет значение.

| user_id | creation_date | views_count | cumulative_count |
|---|---|---|---|
| 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 |
| 1 | 2008-08-04 02:45:08 | 0 | 616509 |
| 1 | 2008-08-04 04:31:03 | 0 | 616509 |
| 1 | 2008-08-04 08:04:42 | 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. Подсказка. Посчитайте, сколько активных дней было у каждого пользователя. Добавьте данные в общее табличное выражение и используйте в основном запросе.

```
WITH temp AS
(
SELECT user_id,
      COUNT(DISTINCT DATE_TRUNC('day', creation_date)::date)
FROM stackoverflow.posts
WHERE creation_date::date BETWEEN '2008-12-01' AND '2008-12-07'
GROUP BY user_id
)
SELECT ROUND(AVG(count))::int AS result
FROM temp;
```

Попробуйте проанализировать результат: какие выводы можно сделать?
Проанализировав итоговую таблицу можно сделать вывод, что за период с 1 по 7 декабря 2008 года пользователи взаимодействовали с платформой в среднем 2 дня.

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`. Подсказка. Эту задачу стоит декомпозировать. Сформируйте запрос, который отобразит номер месяца и количество постов. Затем можно использовать оконную функцию, которая вернёт значение за предыдущий месяц, и посчитать процент.
Не забудьте сравнить количество постов в сентябре со значением предыдущего месяца. Вы получите `NULL` — это нормально.

```
WITH temp AS
(
SELECT EXTRACT(MONTH FROM creation_date)::int AS creation_month,
      COUNT(id) AS posts_count
FROM stackoverflow.posts
WHERE EXTRACT(MONTH FROM creation_date)::int BETWEEN 9 AND 12
GROUP BY creation_month
)
```

```
SELECT *,
    ROUND((posts_count::numeric/LAG(posts_count) OVER()-1)*100, 2)
FROM temp;
```

| creation_month | posts_count | round |
|---|---|---|
| 9 | 70371 | |
| 10 | 63102 | -10.33 |
| 11 | 46975 | -25.56 |
| 12 | 44592 | -5.07 |

7. Найдите пользователя, который опубликовал больше всего постов за всё время с момента регистрации. Выведите данные его активности за октябрь 2008 года в таком виде:
    номер недели;
    дата и время последнего поста, опубликованного на этой неделе.

Подсказка

Декомпозируйте задачу:
    Найдите пользователя, который опубликовал больше всего постов.
    Найдите дату и время создания каждого поста этого пользователя и номер недели.
    Отобразите данные только о последних постах пользователя. Для этого можно использовать оконную функцию.

```
WITH us AS
(SELECT user_id,
    COUNT(id)
FROM stackoverflow.posts
GROUP BY 1
ORDER BY 2 DESC
LIMIT 1)
SELECT EXTRACT(WEEK FROM p.creation_date)::int week_creation,
    MAX(p.creation_date) creation_date
FROM us u
JOIN stackoverflow.posts p ON u.user_id=p.user_id
WHERE DATE_TRUNC('month', p.creation_date)::date = '2008-10-01'
GROUP BY 1
```

| week_creation | creation_date |
|---|---|
| 40 | 2008-10-05 09:00:58 |
| 41 | 2008-10-12 21:22:23 |
| 42 | 2008-10-19 06:49:30 |