

Задания на БД vk:

1. Проанализировать какие запросы могут выполняться наиболее часто в процессе работы приложения и добавить необходимые индексы.

Если взять за точку отсчета мобильное приложение или сайт, то самым часто используемым запросом будет таблица постов которая складывается в новости или ленту новостей. Значит мы должны навесить основной индекс на «created_at,user_id,id». Далее идет таблица лайков в которую как правило передается списком уже загруженные идентификаторы постов и следовательно индекс надо навесить на post_id, но у нас есть таблица агрегации и количество лайков к посту мы возьмем оттуда «like_post_id». Индексы на остальное мы навешиваем но это не часто используемые таблицы, так как пользователи и группы это кешируемые данные и хранимые в базе приложений и обновляемые только при изменении.

Скрипт с индексами в приложении

2. Задание на денормализацию

Разобраться как построен и работает следующий запрос

Очень странный запрос

Видимо хотели посчитать общее количество лайков для 10 самых старых пользователей

```
SELECT SUM(count) as overall FROM (
```

#Так как в верху агрегат , то снизу нет никакого смысла в конкатенации и вычисления возраста –

#это только заставляя сервер избыточно вычислять

```
SELECT
```

```
CONCAT(u.firstname, ' ', u.lastname) as user,
```

```
count(l.id) as count,
```

```
TIMESTAMPDIFF(YEAR, p.birthday, NOW()) AS age
```

#Пользователь должен быть вычислен предварительно и передан через IN списком

```
FROM users AS u
```

```
INNER JOIN profiles AS p
```

```
ON p.user_id = u.id
```

```
LEFT JOIN media as m
```

```
ON m.user_id = u.id
```

```
LEFT JOIN messages as t
```

```
ON t.from_user_id = u.id
```

```
LEFT JOIN
```

```
likes AS l
```

```
ON l.post_id = u.id AND l.like_type_id = 2
```

```
OR l.post_id = m.id AND l.like_type_id = 1
```

```
OR l.post_id = t.id AND l.like_type_id = 3
```

```
GROUP BY u.id
```

```
ORDER BY p.birthday DESC
```

```
LIMIT 10) AS likes;
```