DBS_3

GET /v3/users/:user id/badge history

Pre vybraneho pouzivatela uskutocnite analyzu jeho zıskanych odznakov (badges) a to tak, ze vo vystupe budu uvedene vsetky zıskane odznaky spolu s predchadzaj´ ucou spravou, ktoru autor napisal pred samotnym zıskanım odznaku. Ak zıskal odznak a pred danym odznakom nebola poslana ziadna sprava, tak sa vo vystupe takyto odznak nezobrazı. Ak zıskal napr. 2 odznaky a predtym bolo poslanych viacero sprav, tak vo vystupe je zobrazeny iba prvy odznak s tym, ze sa uvedie posledna sprava ktora mu predchadzala.

Volanie: /v3/users/120/badge history

SQL dopyt:

```
GET_BADGE_POSTS_HISTORY_WITH_LIMIT_QUERY = """
SELECT id, title, type, created_at, CEIL(ROW_NUMBER() OVER (ORDER BY
created_at) / 2.0) AS position
FROM(
    SELECT *, LAG(type) OVER (ORDER BY created_at) AS previous_type,
LEAD(type) OVER (ORDER BY created_at) AS next_type
    FROM(
        SELECT posts.id AS id,
            posts.title AS title,
            'post' AS type,
            TO_CHAR(posts.creationdate AT TIME ZONE 'UTC+0', 'YYYY-MM-
DD"T"HH24:MI:SS.US+00:00') AS created_at
        FROM posts
        JOIN users ON posts.owneruserid = users.id
        WHERE users.id = $1

        UNION

        SELECT badges.id AS id,
            badges.name AS title,
            'badge' AS type,
            TO_CHAR(badges.date AT TIME ZONE 'UTC+0', 'YYYY-MM-
DD"T"HH24:MI:SS.US+00:00') AS created_at
        FROM badges
        JOIN users ON badges.userid = users.id
        WHERE users.id = $1
        ORDER BY type, title
    ) as final
) as finality
WHERE (previous_type = 'post' AND type = 'badge') OR (next_type = 'badge' AND
type = 'post')
ORDER BY created_at
```

- V internom dotaze najskôr dostaneme zoznam príspevkov vytvorených konkrétnym používateľom, potom dostaneme odznaky, ktoré dostane konkrétny používateľ a spojíme ich do jednej tabuľky a zoradíme v abecednom poradí (pre správne usporiadanie odznakov, ak sú prijaté v rovnakom čase)

- Ďalej pomocou LAG a LEAD a triedením podľa času vytvorenia dostaneme typ nasledujúceho a predchádzajúceho objektu
- V externom dotaze dodatočne dostanem sériové číslo objektu v tabuľke, vydelím ho 2 a zaokrúhlim, aby som dostal párové hodnoty. A vyberiem len tie objekty, ktorých typ je odznak a pred ním je príspevok ALEBO, ktorého typ je príspevok a ďalší je odznak

GET /v3/tags/:tag/comments?count=:count

Pre zadany tag vypocıtajte pre jednotlive posty (posts), ktore maju viac ako zadany pocet komentarov (zadane v ramci API endpointu), priemernu dobu odpovede medzi jednotlivymi komentarmi v ramci daneho postu. Vo vypise uvedte ako sa jednotliva priemerna doba odpovede menila s pribudajucimi komentarmi

Volanie: /v3/tags/networking/comments?count=40

SQL dopyt:

```
GET_COMMENTS_BY_TAGS_MORE_THAN_COUNT_QUERY = """
SELECT main.post_id,
    main.title,
    main.displayname,
    main.text,
    TO_CHAR(main.post_created_at AT TIME ZONE 'UTC+0', 'YYYY-MM-
DD"T"HH24:MI:SS.US+00:00') AS post_created_at,
    TO_CHAR(main.created_at AT TIME ZONE 'UTC+0', 'YYYY-MM-
DD"T"HH24:MI:SS.US+00:00') AS created_at,
    (main.created_at - main.previous_time) AS diff,
    AVG(main.created_at - main.previous_time) OVER (PARTITION BY main.post_id
ORDER BY main.created_at ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS
avg
    FROM(
    SELECT users.displayname AS displayname, posts.id AS post_id, posts.title
AS title, comments.text AS text, comments.creationdate AS created_at,
posts.creationdate AS post_created_at, LAG(comments.creationdate, 1,
posts.creationdate) OVER (PARTITION BY posts.id ORDER BY
comments.creationdate) AS previous_time
    FROM post_tags
        JOIN posts ON post_tags.post_id = posts.id
        JOIN tags ON tags.id = post_tags.tag_id
        JOIN comments ON posts.id = comments.postid
        LEFT JOIN users ON comments.userid = users.id

    WHERE tags.tagname = $1 AND posts.commentcount > $2
    ORDER BY comments.creationdate
) main;
"""
```

- Najprv si vytvoríme tabuľku (zoradenú podľa času) komentárov, ktorá bude obsahovať názov príspevku a meno komentátora, navyše dostaneme čas vytvorenia predchádzajúceho komentára (alebo ak ide o prvý komentár , potom čas vytvorenia príspevku) Vyberáme len komentáre s určitým tagom a ak sú pod príspevkom komentáre (pod ktorými bol tento komentár zanechaný) je viac ako určitý počet.

- V externej požiadavke získame rozdiel medzi predchádzajúcim a týmto komentárom odčítaním. Získame priemernú hodnotu rozdielu, počnúc prvým komentárom až po aktuálny (vydelením tohto počítadla príspevkami)

GET /v3/tags/:tagname/comments/:position?limit=:limit

Vradte komentare pre prıspevky s tagom :tagname, ktore boli vytvorene ako k-te v poradı (:position) zoradenych podla datumu vytvorenia postup s limitom :limit.

Volanie: /v3/tags/linux/comments/2?limit=1

SQL dopyt:

```
GET_K_COMMENT_TO_POSTS_BY_TAGS_WITH_LIMIT_QUERY = """
    SELECT comments.id, users.displayname, base.body, comments.text,
comments.score, ARRAY_POSITION(base.comment_ids, comments.id) AS position
    FROM comments
    LEFT JOIN users ON comments.userid = users.id
    JOIN(
        SELECT posts.id as post_id, posts.body, posts.creationdate,
ARRAY_AGG(comments.id ORDER BY comments.creationdate) AS comment_ids
        FROM tags
            JOIN post_tags ON tags.id = post_tags.tag_id
            JOIN posts ON post_tags.post_id = posts.id
            JOIN comments ON posts.id = comments.postid
        WHERE tags.tagname = $1
        GROUP BY posts.id, posts.body, posts.creationdate
        HAVING COUNT(*) >= $2
        ORDER BY posts.creationdate
        LIMIT $3
    ) base ON comments.id = base.comment_ids[$2];
"""
```

- Aby som získal K-tý komentár, najprv získam pole ID komentárov spojených s jedinečnými príspevkami, ktoré majú určitú značku, majú viac ako určitý počet zoskupených objektov (ID komentárov), zoradených podľa dátumu vytvorenia príspevku s určitým limitom na zobrazenie vhodných.
- V externom dotaze zlúčime komentáre s používateľmi, aby sme získali mená používateľov a spojili ich s výslednou filtrovanou tabuľkou. Pozíciu prijatého komentára získame aj v zozname všetkých komentárov pomocou ARRAY_POSITION.

GET/v3/posts/:postid?limit=:limit

Vystupom je zoznam o velkosti :limit vlakna pre prıspevok(post) s ID postid. Vlakno zacına prıspevkom postid a pokracuje prıspevkami, kde postid je parented zoradeny podla datumu vytvorenia od najstarsieho.

Volanie: /v3/posts/2154?limit=2

SQL dopyt:

```
GET_POSTS_AND_PARENT_POSTS_BY_POSTID_WITH_LIMIT_QUERY = """
SELECT users.displayname, posts.body, TO_CHAR(posts.creationdate AT TIME ZONE
'UTC+0', 'YYYY-MM-DD"T"HH24:MI:SS.US+00:00') AS created_at
FROM posts
JOIN users ON posts.owneruserid = users.id
WHERE posts.id = $1 OR posts.parentid = $1
ORDER BY posts.creationdate
LIMIT $2
"""
```

- Prepojte tabuľku príspevkov s tabuľkou s používateľmi, aby sa zobrazilo meno používateľa, ktorý príspevok vytvoril. Vyberáme príspevky so zvoleným ID, ako aj tie príspevky, ktorých *paretnid* sa rovná zvolenému ID.