|  |
| --- |
| FIIT STU |
| DBS zadanie 3 |
| Dokumentácia Databázové systémy |

|  |
| --- |
| Norbert Matuška  3-23-2024 |

Contents

[GET /v3/tags/:tag/comments?count=:count 2](#_Toc162273252)

[Example output pre tag ´networking´ a count > 40 4](#_Toc162273253)

[GET /v3/tags/:tagname/comments/:position?limit=:limit 5](#_Toc162273254)

[Example output pre tagname ´linux´, position 2 a limit 1 6](#_Toc162273255)

[GET /v3/posts/:postid?limit=:limit 7](#_Toc162273256)

[Example output pre id 2154 a limit 2 7](#_Toc162273257)

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

cur.execute("""  
SELECT  
 TO\_CHAR('00:00:00'::time + (comments\_data.sum\_diff / comments\_data.count\_diff) \* '1 second'::interval, 'HH24:MI:SS.US') AS avg,  
 comments\_data.created\_at,  
 comments\_data.diff,  
 comments\_data.displayname,  
 comments\_data.post\_created\_at,  
 comments\_data.post\_id,  
 comments\_data.text,  
 comments\_data.title  
FROM (  
 SELECT  
 post\_data.post\_id,  
 post\_data.title,  
 post\_data.displayname,  
 post\_data.text,  
 post\_data.post\_created\_at,  
 post\_data.created\_at,  
 TO\_CHAR(post\_data.created\_at - LAG(post\_data.created\_at) OVER (  
 PARTITION BY post\_data.post\_id ORDER BY post\_data.created\_at  
 ), 'HH24:MI:SS.US') AS diff,  
 SUM(post\_data.diff) OVER (  
 PARTITION BY post\_data.post\_id ORDER BY post\_data.created\_at  
 ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW  
 ) as sum\_diff,  
 COUNT(post\_data.diff) OVER (  
 PARTITION BY post\_data.post\_id ORDER BY post\_data.created\_at  
 ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW  
 ) as count\_diff  
 FROM (  
 SELECT  
 posts.id AS post\_id,  
 posts.title,  
 users.displayname,  
 comments.text,  
 posts.creationdate AS post\_created\_at,  
 comments.creationdate AS created\_at,  
 EXTRACT(EPOCH FROM (comments.creationdate - LAG(comments.creationdate) OVER (  
 PARTITION BY comments.postid ORDER BY comments.creationdate  
 ))) AS diff  
 FROM  
 comments  
 JOIN  
 posts ON comments.postid = posts.id  
 JOIN  
 users ON comments.userid = users.id  
 JOIN  
 post\_tags ON posts.id = post\_tags.post\_id  
 JOIN  
 tags ON post\_tags.tag\_id = tags.id  
 WHERE  
 tags.tagname = %s  
 ) AS post\_data  
 JOIN (  
 SELECT  
 postid  
 FROM  
 comments  
 GROUP BY  
 postid  
 HAVING  
 COUNT(\*) > %s  
 ) AS popular\_posts ON post\_data.post\_id = popular\_posts.postid  
 WHERE  
 post\_data.diff IS NOT NULL  
) AS comments\_data  
ORDER BY  
 comments\_data.post\_id, comments\_data.created\_at;  
""", (tag, count))

Subequery post\_data joinne viacero tables aby sme mohli vyhľadávať podľa špecifického tagname a vypočítame časový rozdiel pomocou funkcie LAG. Táto window function môže accessovať dáta z predošlého riadka. Používame partition cez ID aby sme počítali každý post ako separátny subset dát. Ďalej v subquery popular posts filtrujeme posty, ktoré majú aspoň x počet komentárov. Na základe post\_data si tiež vypočítavam sum\_diff a count\_diff pre každý príspevok až po aktuálny riadok (ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW). Nakoniec vypočítam avg pre komentáre každého postu (sum\_diff/countdiff) na formát HH24:MI:SS.US.

## Example output pre tag ´networking´ a count > 40

A screenshot of a computer screen

Description automatically generated

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

cur.execute("""  
SELECT  
 sub.body,  
 sub.displayname,  
 sub.id,  
 sub.position,  
 sub.score,  
 sub.text,  
 sub.creationdate  
FROM (  
 SELECT  
 p.id as post\_id,  
 c.id as id,  
 u.displayname,  
 p.body as body,  
 c.text as text,  
 c.score,  
 p.creationdate,  
 ROW\_NUMBER() OVER (PARTITION BY p.id ORDER BY c.creationdate ASC) as position  
 FROM  
 comments c  
 JOIN  
 posts p ON p.id = c.postid  
 JOIN  
 users u ON u.id = c.userid  
 JOIN  
 post\_tags pt ON p.id = pt.post\_id  
 JOIN  
 tags t ON pt.tag\_id = t.id  
 WHERE  
 t.tagname = %s  
) sub  
WHERE  
 sub.position = %s  
ORDER BY  
 sub.creationdate ASC  
LIMIT %s;  
""", (tag\_name, position, limit))

Pomocou ROW\_NUMBER() v subquery sub, si očíslujem komentáre zoradené chronologicky podľa creationdate a reštartuje sa poďla každého unikátneho postu (PARTITION BY p.id). Outer query len vyfiltruje komentáre a posty ktorých position je práve tá ktorá prišla do požiadavky.

## Example output pre tagname ´linux´, position 2 a limit 1

A screenshot of a computer

Description automatically generated

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

cur.execute("""  
SELECT  
 u.displayname,  
 p.body,  
 p.creationdate as created\_at  
FROM  
 posts p  
JOIN  
 users u ON p.owneruserid = u.id  
WHERE  
 p.id = %s OR p.parentid = %s  
ORDER BY  
 p.creationdate ASC  
LIMIT %s;  
""", (post\_id, post\_id, limit))

Keďže zoradím posty podľa creationdate, môžem si byť istý, že na začiatku bude originálny post a ďalej budú nasledovať subposty.

## Example output pre id 2154 a limit 2

A screenshot of a computer program

Description automatically generated