Zaawansowane Bazy Danych 2020
zadanie 1

Kamil Dubil
kd370826@students.mimuw.edu.pl

8 listopada 2020

Wygenerowane dane umieściłem w katalogu `data/`.

1 Wersja JSONowa

Tabele z danymi:

```
DROP TABLE IF EXISTS jsonb_audience, jsonb_targets;

CREATE TABLE jsonb_audience (
    date date NOT NULL,
    content jsonb NOT NULL
);
CREATE TABLE jsonb_targets (
    content jsonb NOT NULL
);
```

Wczytanie danych z plików:

```
COPY jsonb_audience(content, date) FROM PROGRAM 'find /home/kd/zbd/zad1/
    data/ -type f -name "audience*json" -exec cat {} \; -exec echo -en "\t
    " \; -exec sh -c "basename {} | cut -c10-19" \;';
COPY jsonb_targets(content) FROM '/home/kd/zbd/zad1/data/targets.json';
```

Zapytanie:

```
WITH audience AS (
    SELECT DISTINCT
        date,
        (jb->'person_id')::int person_id,
        jb->>'demography' demography,
        regexp_split_to_table(jb->>'contacts', '') contact
    FROM (
        SELECT
            date,
            jsonb_array_elements(content) jb
        FROM
            jsonb_audience
        ) ta
    WHERE
        jb->>'contacts' != ''
), targets AS (
    SELECT
        (jb->'target')::int id,
        replace(jb->>'definition', ' ', '_') definition
    FROM (
        SELECT
            jsonb_array_elements(content) jb
        FROM
            jsonb_targets
        ) tt
)
```

```
SELECT
    audience.date dzien,
    targets.id grupa,
    audience.contact reklama,
    count(*) osob
FROM
    audience
JOIN
    targets
ON
    audience.demography LIKE targets.definition
GROUP BY
    dzien,
    grupa,
    reklama
ORDER BY
    dzien,
    grupa,
    reklama;
```

Wynik:

```
   dzien    | grupa | reklama | osob
------------+-------+---------+------
 2019-01-01 |     0 | B       |    1
 2019-01-01 |     0 | C       |    2
 2019-01-01 |     0 | E       |    2
 ...
 2019-01-30 |     9 | X       |    1
 2019-01-30 |     9 | Y       |    2
 2019-01-30 |     9 | Z       |    3
(5721 rows)
```

Statystyki wykonania (fragment wyniku `EXPLAIN ANALYZE`):

```
 Planning Time: 0.353 ms
 JIT:
   Functions: 23
   Options: Inlining true, Optimization true, Expressions true, Deforming
       true
   Timing: Generation 2.252 ms, Inlining 75.790 ms, Optimization 166.603
      ms, Emission 108.365 ms, Total 353.010 ms
 Execution Time: 426.394 ms
```

2 Wersja SQLowa

Tymczasowe tabele z danymi:

```
DROP TABLE IF EXISTS jsonb_audience, jsonb_targets;

CREATE TABLE jsonb_audience (
    date date NOT NULL,
    content jsonb NOT NULL
);
CREATE TABLE jsonb_targets (
    content jsonb NOT NULL
);
```

Wczytanie danych z plików:

```
COPY jsonb_audience(content, date) FROM PROGRAM 'find /home/kd/zbd/zad1/
    data/ -type f -name "audience*json" -exec cat {} \; -exec echo -en "\t
    " \; -exec sh -c "basename {} | cut -c10-19" \;';
COPY jsonb_targets(content) FROM '/home/kd/zbd/zad1/data/targets.json';
```

Właściwe tabele i przeniesienie do nich danych:

```
DROP TABLE IF EXISTS audience, targets;

CREATE TABLE audience (
    date date NOT NULL,
    person_id int NOT NULL,
    demography text NOT NULL,
    contact char NOT NULL,
    PRIMARY KEY (date, person_id, demography, contact)
);
CREATE TABLE targets (
    id int PRIMARY KEY,
    definition text NOT NULL
);

INSERT INTO audience (
    SELECT DISTINCT
        date,
        (jb->'person_id')::int person_id,
        jb->>'demography' demography,
        regexp_split_to_table(jb->>'contacts', '') contact
    FROM (
        SELECT
            date,
            jsonb_array_elements(content) jb
        FROM
            jsonb_audience
        ) ta
    WHERE
        jb->>'contacts' != ''
);

INSERT INTO targets (
    SELECT
        (jb->'target')::int id,
        replace(jb->>'definition', ' ', '_') definition
    FROM (
        SELECT
            jsonb_array_elements(content) jb
        FROM
            jsonb_targets
        ) tt
);

DROP TABLE IF EXISTS jsonb_audience, jsonb_targets;
```

Zapytanie:

```
SELECT
    audience.date dzien,
    targets.id grupa,
    audience.contact reklama,
    count(*) osob
FROM
    audience
JOIN
    targets
ON
    audience.demography LIKE targets.definition
GROUP BY
    dzien,
    grupa,
    reklama
ORDER BY
    dzien,
    grupa,
    reklama;
```

Wynik:

```
   dzien    | grupa | reklama | osob
------------+-------+---------+------
 2019-01-01 |     0 | B       |    1
 2019-01-01 |     0 | C       |    2
 2019-01-01 |     0 | E       |    2
 ...
 2019-01-30 |     9 | X       |    1
 2019-01-30 |     9 | Y       |    2
 2019-01-30 |     9 | Z       |    3
(5721 rows)
```

Statystyki wykonania (fragment wyniku `EXPLAIN ANALYZE`):

```
 Planning Time: 0.125 ms
 Execution Time: 29.539 ms
```

3 Podsumowanie

Wykonanie zapytania w wersji SQLowej jest istotnie szybsze. Jeśli doliczymy czas przenoszenia danych do właściwych tabel, to wynik jest dużo gorszy, ale nadal lepszy od wersji JSONowej — `Execution Time` dla `INSERT INTO audience`: `288.620 ms`, dla `INSERT INTO targets`: `0.206 ms`, co łącznie daje `318.365 ms`.