Raport optymalizacji BD sklepu “Stokrotka”

“Baza danych została zoptymalizowana!”

Informacje ogólne o przeprowadzonej optymalizacji:

- Pierwszym sposobem było ustawienie foreign key w tabelach gdzie takowe występują.
- Drugi sposób polegał na stworzeniu indeksów.
- Trzecia metoda to optymalizacja zapytań.
- **Janusz jest zadowolony. Jeff Bezos prosi o nasze CV. Sprzedaliśmy dużo ziemniaków.**

Krok 1: nałożenie foreign key na wartości nimi będące

```
ALTER TABLE produkty
ADD FOREIGN KEY (id_kategorii) REFERENCES kategorie(id_kategorii);

ALTER TABLE asortyment
ADD FOREIGN KEY (id_produktu) REFERENCES produkty(id_produktu);

ALTER TABLE asortyment
ADD FOREIGN KEY (id_sklepu) REFERENCES sklepy(id_sklepu);

ALTER TABLE sklepy
ADD FOREIGN KEY (id_miejscowosci) REFERENCES miejscowosci(id_miejscowosci)
;

ALTER TABLE sklepy
ADD FOREIGN KEY (id_kraju) REFERENCES kraje(id_kraju);

ALTER TABLE miejscowosci
ADD FOREIGN KEY (id_kraju) REFERENCES kraje(id_kraju);

ALTER TABLE zamowienia
ADD FOREIGN KEY (id_produktu, id_sklepu) REFERENCES asortyment(id_produktu,
id_sklepu);
```

Disclaimer

Projekt bazy jest bardzo istotny, gdyż wszelkie operacje typu nakładanie foreign key są bardzo kosztowne – przykład na stokrotce – wykonanie:

```
 ALTER TABLE asortyment
ADD FOREIGN KEY (id_produktu) REFERENCES produkty(id_produktu);
```

trwało około **3 godzin.**

Krok 2: analiza przydatnych indexów.

Korzystny index będzie często wykorzystywany w warunku WHERE, a dane na których się znajduje będą rzadziej aktualizowane niż selectowane. Z tego wynika następująca optymalizacja stokrotkadb:

Tabela *produkty* będzie przeszukiwana po kolumnie „nazwa”, a update’y będą wykonywane stosunkowo rzadko. Jest to pierwsze miejsce do nałożenia indexu.

Tabela **produkty** kolumna **nazwa**.

```
CREATE INDEX idx_nazwapr
ON produkty (nazwa);
```

Tabela *asortyment* będzie przeszukiwana najczęściej pod warunkiem równości id z tabelą produkty oraz osobno id sklepu porównane z tabelą sklepy. Dlatego, mimo istniejącego indexu (jednego) na kolumnach id produktu, id sklepu opłacalnym jest nałożenie osobnego indexu na id sklepu.

```
CREATE INDEX idx_idsklepu
ON asortyment (id_sklepu);
```

Disclaimer

Indeksy powodują duży przyrost zajmowanego miejsca przez bazę danych.

Uzyskana poprawa:

Krok 3: optymalizacja zapytań zgodnych ze standardami [źródło]

- Index all the predicates in JOIN, WHERE, ORDER BY and GROUP BY clauses.
- Avoid using functions in predicates.
- The index is not used by the database if there is a function on the column.
- Avoid using wildcard (%) at the beginning of a predicate.
- Avoid unnecessary columns in SELECT clause.
- Use inner join, instead of outer join if possible.
- DISTINCT and UNION should be used only if it is necessary.
- Push predicates into the OUTER JOIN clause whenever possible.
- Duplicate constant condition for different tables whenever possible

Kwerendy poprawione wedle wyżej wymienionych zasad wraz z końcowymi czasami wykonywania:

Domyślna akcja:

```
SELECT pr.nazwa,pr.cena_kg,pr.cena, asor.id_sklepu
FROM produkty AS pr
    JOIN asortyment AS asor ON pr.id_produktu=asor.id_produktu
    JOIN sklepy AS sk ON asor.id_sklepu = sk.id_sklepu
    JOIN kraje AS kr ON sk.id_kraju=kr.id_kraju
WHERE pr.nazwa = "mieso51342" AND kr.nazwa = "Polska";
```

Czas wykonania 0.000 sec / 0.000 sec

Klauzurę pr.nazwa = „” użytkownik może zmienić na zapytanie LIKE „%”

Czas wykonania 0.000 sec / 0.000 sec

Wyszukiwanie konkretnego produktu z danej kategorii i o przybliżonej nazwie w danym państwie

```
SELECT pr.nazwa, pr.cena_kg, pr.cena
FROM produkty AS pr
    JOIN kategorie AS ka ON pr.id_kategorii=ka.id_kategorii
    JOIN asortyment AS asor ON pr.id_produktu=asor.id_produktu
    JOIN sklepy AS sk ON asor.id_sklepu=sk.id_sklepu
    JOIN kraje AS pa ON sk.id_kraju=pa.id_kraju
WHERE pr.nazwa LIKE "mieso99%" AND pa.nazwa = "Bialorus";
```

Czas wykonania: 24.297 sec

Zawężenie do konkretnego miasta:

```
SELECT pr.nazwa,pr.cena_kg,pr.cena, asor.id_sklepu
  FROM produkty AS pr
        JOIN asortyment AS asor ON pr.id_produktu=asor.id_produktu
        JOIN sklepy AS sk ON asor.id_sklepu = sk.id_sklepu
        JOIN kraje AS kr ON sk.id_kraju=kr.id_kraju
        JOIN miejscowosci AS mi ON mi.id_kraju = kr.id_kraju
WHERE pr.nazwa = "mieso51342" AND mi.nazwa = "miejscowosc5";
```

Czas trwania: 0.016 sec / 0.000 sec

Zapytanie z LIKE zamiast =

Czas wykonania: 0.000 sec / 0.000 sec

Asortyment konkretnego sklepu:

```
SELECT pr.nazwa, pr.cena_kg, pr.cena, sk.id_sklepu
FROM produkty AS pr
        JOIN asortyment AS asor ON pr.id_produktu = asor.id_produktu
        JOIN sklepy AS sk ON sk.id_sklepu = asor.id_sklepu
WHERE pr.id_kategorii = 2 AND sk.id_sklepu = 53;
```

Czas trwania 0.922 sec / 3.578 sec

Zawężenie do 3 sklepów wybranych przez użytkownika:

```
SELECT pr.nazwa,pr.cena_kg,pr.cena, asor.id_sklepu
FROM produkty AS pr
    JOIN asortyment AS asor ON pr.id_produktu=asor.id_produktu
    JOIN sklepy AS sk ON asor.id_sklepu = sk.id_sklepu
    JOIN kraje AS kr ON sk.id_kraju=kr.id_kraju
    JOIN miejscowosci AS mi ON mi.id_kraju = kr.id_kraju
WHERE pr.nazwa = "mieso51342" AND mi.nazwa IN ('miejscowosc5', 'miejscowosc32',
'miejscowosc87');
```

Czas trwania: 0.000 sec / 0.000 sec

Wyszukanie konkretnego produktu we wszystkich sklepach:

```
SELECT *
FROM produkty
    JOIN asortyment ON produkty.id_produktu=asortyment.id_produktu
WHERE produkty.nazwa = 'mieso412';
```

Czas wykonania: 0.125 sec / 0.000 sec

Wyszukaj wszystkich produktów dostępnych w sklepach w „miejscowosc1”.

```
SELECT pr.nazwa, pr.cena_kg, pr.waga, pr.cena
FROM produkty AS pr
    JOIN asortyment AS aso ON pr.id_produktu=aso.id_produktu
    JOIN sklepy AS sk ON aso.id_sklepu=sk.id_sklepu
    JOIN miejscowosci AS mi ON sk.id_miejscowosci=mi.id_miejscowosci
WHERE mi.nazwa = 'miejscowosc1';
```

Czas wykonania: 0.000 sec / 0.187 sec

Wyszukaj wszystkie produkty dostępne w sklepach w Kosowie.

SELECT pr.nazwa, pr.cena_kg, pr.waga, pr.cena FROM produkty AS pr

JOIN asortyment AS aso ON pr.id_produktu=aso.id_produktu

JOIN sklepy AS sk ON aso.id_sklepu=sk.id_sklepu

WHERE sk.id_kraju IN (SELECT id_kraju FROM kraje WHERE nazwa = 'Kosowo');

Czas wykonania: 0.031 sec / 0.360 sec

Wyszukaj produktów zaczynających się na nazwę „nabial4278” w miejscowości „,miejscowosc31”.

SELECT pr.nazwa, pr.cena_kg, pr.waga, pr.cena FROM produkty AS pr

JOIN asortyment AS aso ON pr.id_produktu=aso.id_produktu

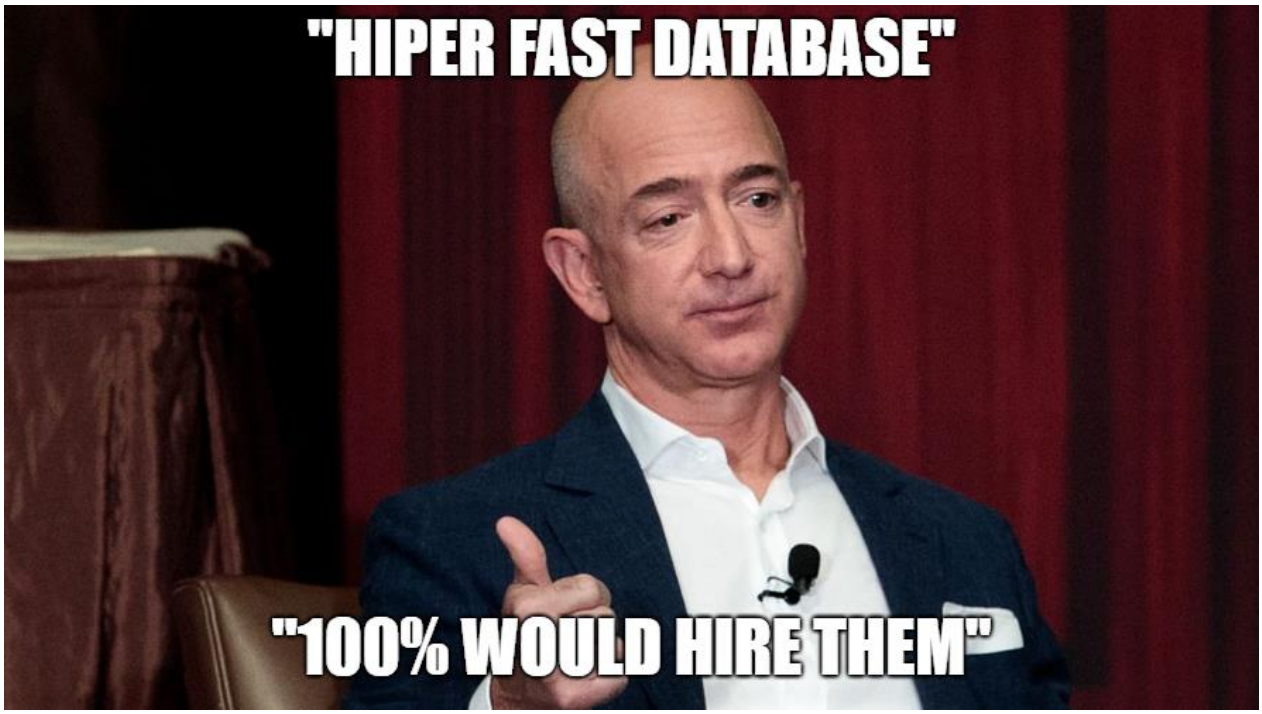
JOIN sklepy AS sk ON aso.id_sklepu=sk.id_sklepu

WHERE sk.id_miejscowosci IN (SELECT mi.id_miejscowosci FROM miejscowosci AS mi WHERE mi.nazwa = 'miejscowosc31') AND pr.nazwa LIKE 'nabial4278%';

Czas trwania: 0.062 sec / 0.000 sec

Wnioski

- Optymalizacja jest najskuteczniejsza gdy przeprowadzana od samego początku – pomysłu, projektu bazy danych.
- Im później tym bardziej kosztownie. I dłużej.
- Dobra analiza i wykrycie ścieżek krytycznych to połowa sukcesu – druga to odpowiednia optymalizacja.
- Foreign Key przyśpiesza kwerendy.
- Indeks w odpowiednim miejscu bardzo przyśpiesza kwerendy – szczególnie gdy jest wykorzystany po słowie WHERE.
- Baza przed optymalizacją – 1.7GB, po – 3.5 GB. Mamy 2 krotny przyrost zajmowanej pamięci, jednak prędkość wykonywania zapytań wzrosła ok. 160/0.001 krotnie, co przy kosztach pamięci masowych porównanych do kosztów mocy obliczeniowej jest operacją wysoce opłacalną.
- W przyszłości dobrym rozwiązaniem byłoby zastosować rozproszoną bazę danych – po jednej na sklep.



Autorzy: Michał Podgórny & Jakub Sroka