Big data management
M2 LID 2022-23

Denormalization and query execution performance on RDBMS

The goal of this session is to understand the impact on query execution performance of the denormalization process. We will be using the PostgreSQL system.

1 Dataset

You download the CIS_bdpm and CISCIP_bdpm text files. They contain information on drugs being sold in France. The fields elements in the CIS file are :

- Code CIS (Code Identifiant de Spécialité)
- Dénomination du médicament
- Forme pharmaceutique
- Voies d'administration (avec un séparateur ';' entre chaque valeur quand il y en a plusieurs)
- Statut administratif de l'autorisation de mise sur le marché (AMM)
- Type de procédure d'autorisation de mise sur le marché (AMM)
- Etat de commercialisation
- Date d'AMM (format JJ/MM/AAAA)
- StatutBdm : valeurs possibles : 'Alerte' ou 'Warning disponibilité' (icône grise)
- Numéro de l'autorisation européenne
- Titulaire(s) : S'il y a plusieurs titulaires, les valeurs seront séparées par des ';'
- Surveillance renforcée: valeurs 'Oui' ou 'Non'

Considering the CISCIP file, the fields are :

- Code CIS
- Code CIP7 (Code Identifiant de Présentation à 7 chiffres)
- Libellé de la présentation
- Statut administratif de la présentation
- Etat de commercialisation de la présentation tel que déclaré par le titulaire de l'AMM
- Date de la déclaration de commercialisation (format JJ/MM/AAAA)
- Code CIP13 (Code Identifiant de Présentation à 13 chiffres)
- Agrément aux collectivités ('oui', 'non' ou 'inconnu')
- Taux de remboursement (avec un séparateur ';' entre chaque valeur quand il y en a plusieurs)
- Prix du médicament en euro
- Texte présentant les indications ouvrant droit au remboursement par l'assurance maladie s'il y a plusieurs taux de remboursement pour la même présentation

2 Transform and load the data into PostgreSQL

Create a database on PostgreSQL, create tables. Upload the data of the TSV files into the corresponding table.

3 Querying

We would like to study the impact of various approaches on query performance. We are interested in two realistic queries. The first one is not selective since it retrieves over 2000 records from the database. The second one is highly selective since it is retrieving a single record:

Q1: select cip7, cc.cis, denom from bdpm_cis c join bdpm_ciscip cc on cc.cis=c.cis where cip7¡3000000;

Q2: select cip7, cc.cis, denom from bdpm_cis c join bdpm_ciscip cc on cc.cis=c.cis where cip7=3000064;

3.1 No denormalization, no indexes

Using explain analysis, note the performance of each query. Execute several times the Q1, do you see any difference in terms of execution time? Why would it be fair to consider an average of several runs (3 to 5) starting from the 2nd run?

3.2 No denormalization, indexes

Create primary key indexes on the two tables. Which column(s) did you select? Run queries Q1 and Q2, note their performance in the same context of 'hot runs'.

3.3 Denormalization, no indexes

Given that Q1 and Q2 are popular in our application, it is worth to use a denormalization approach. In fact, we will copy and extend one of the two tables. Which one? In production setting, that new table would replace the one you are extending. So that would leave you with two tables. Once your selection is validated, create and populate that table. Modify the two queries such that they run on this single table (thus eliminating a join). Run the performance test in the same context as previously.

3.4 Denormalization, indexes

Create a primary key index on the denormalized table. Run the modified queries.

3.5 Conclusion

Based on the query execution evaluation you have just performed, what do you conclude. Which is the most efficient approach.

4 Limitations of denormalization

Do you see any drawback/limitation to this denormalization approach?

5 Views

Starting in PostgreSQL version 10, materialized views are available (https://www.postgresql.org/docs/10/rules-materializedviews.html). So two forms of views can be used in PostgreSQL.

5.1 View creation

Create standard (virtual) and materialized views that could be used to answer our two queries.

5.2 querying the views

Evaluate the performances of the two queries over these views. What do you think? What do you propose to improve the query execution performance? Test your approach.