MU5IN852

Bases de Données Large Echelle

Rappels sur SQL

octobre 2021

hubert.naacke@lip6.fr

Motivations
SQL est un langage « durable »

- Fondation théorique solide
- Pérenne face à l'évolution des solutions big data
  - SQL pour l'accès aux SGBD relationnels « classiques »
  - SQL supporté dans les systèmes récents issus du NOSQL
    - exple Google Spanner
- Générique : cas d'usage très variés
  - ETL, préparation de données
  - BI / E-commerce
  - Stream
- Langage standardisé et largement adopté
  - Langage pivot entre un DSL applicatif et un système de gestion de données.
    - Exple : TAO de Facebook : accès dédié « graphe social » en SQL

Motivations :
SQL est un langage efficace

- Efficacité : traitement plus rapide
  - Une requête SQL peut être optimisée automatiquement
  - Accélère le traitement d'une requête sans la modifier
- SQL est un langage déclaratif. Avantages :
  - Indépendant de l'organisation réelle des données
  - Indépendant des algorithmes qui implantent les opérations de l'algèbre relationnelle
- Transformation SQL --> programme
  - Nombreuses opportunités pour optimiser une requête SQL
    - Reformulation logique en une requête équivalente
    - Invocation des primitives de calcul des processeurs modernes
    - Gestion dédiée des ressources cpu, mémoire

Schéma relationnel

- Description des données selon le modèle relationnel
  - Un schéma relationnel est un ensemble de relations
  - Une relation représente un ensemble de tuples
  - Un tuple a plusieurs attributs, chacun avec un nom et un type
- Clé d'une relation
  - Une clé sert à identifier chaque tuple d'une relation
    - Une clé est composée d'un ou plusieurs attributs
- Référence entre relations
  - Lien entre entités : une relation peut faire référence d'autres relations

  →Une clé étrangère fait référence à la clé primaire d'une relation
- Contraintes logiques
  - domaine d'un attribut, valeur non nulle, unicité, prédicat global

Requêtes SQL

- Mots-clés
  - SELECT … FROM … peut être complété de :
  - WHERE … GROUP BY … ORDER BY …
- Principales opérations
  - Projection
  - Sélection
  - Agrégation
  - Regroupement
  - Jointure
  - Opérations sur des ensembles :
    - union, intersection, différence
  - Tri

Projection et valeurs distinctes

- Schéma: **Visite** (photoID, personID, ville, pays, note)

- Projection : liste des attributs à garder dans le résultat

  Select personID, ville
  from Visite
- Projection sans doubles
  - Select distinct personID, ville from Visite
  - Le résultat ne contient pas 2 nuplets avec la même personne et la même ville
- Renommage dans une projection

  Select ville as city
  from Visite
- Appliquer une fonction sur un attribut : valeur--> valeur
  - Permet de générer des nouveaux attributs

  Select upper(pays) as PAYS from Visite

Sélection : WHERE

Schéma: **Visite** (photoID, personID, ville, pays, note)

- Sélection = filtre multi critères exprimé dans la clause where

  Select *
  from Visite
  where *prédicat*

- Prédicat simple avec les opérateurs =, <, >, <>, like

  Select * from Visite where note = 5
  Select * from Visite where ville like 'New%'

- Prédicat composé avec des connecteurs logiques
  - and, or, not, in, between

  Where (note between 2 and 4)
  and (pays in ('France', 'Italie') or ville='Oslo')

Agrégation

Schéma: **Visite** (photoID, personID, ville, pays, note)

- Fonction d'agrégation :
  - f: ensemble de valeurs --> valeur

  Select f(attribut)
  From Visite
  - Fonctions prédéfinies :
    - Ensemble de nombres --> Nombre
      - max(), min(), sum(), avg()
    - Ensemble de tuples --> Nombre
      - count(*)
- Fonctions ad-hoc définies par l'utilisateur
- Exprimer plusieurs agrégations

  Select min(note), max(note), avg(note), count(distinct pays)
  From Visite

Regroupement : GROUP BY

Schéma: **Visite** (photoID, personID, ville, pays, note)

- Regroupement toujours suivi d'agrégations
  - Découper une relation en plusieurs groupes disjoints
  - Chaque groupe produit un et un seul tuple du résultat
  - Les attributs définissant le regroupement peuvent être projetés dans le résultat
  - Les autres attributs doivent être agrégés
    - sauf si on sait qu'ils dépendent d'un attribut du regroupement

```
Select pays, count(*) as nbreVisite
From Visite
Group by pays
```

Regroupement : exemples

- Groupe composé de plusieurs attributs

```
Select personID, pays, count(*) as nbreVisite
From Visite
Group by personID, pays
```

- Un regroupement sans agrégation = une projection sans doubles

```
Select personID, pays
From Visite
Group by personID, pays
```

équivalent à

```
Select distinct personID, pays
From Visite
```

- Un regroupement avec attributs ‘redondants’

VisiteDetail (photoID, personID, profession, ville, pays, note)

```
Select personID, profession, count(*)
From VisiteDetail
Group by personID, profession
```