Langage SQL

Les fonctions

Guillaume Aubian

Université Panthéon-Assas

Rappel (1/2) : Bases de données relationnelles

Définition : une base de données est un ensemble de données stockées, partagées par plusieurs utilisateurs, et organisées selon un **modèle de données**.

Modèle relationnel :

- Les données sont organisées en **relations** (tables).
- Une table contient des **attributs** (colonnes) et des **occurrences** (lignes).

Exemple de base vols :

- **vol** (id_Vol, Ville_Depart, Ville_Arrivee, Code_Avion, Code_Pilote, Date_Decollage, Heure_Decollage, Duree_Vol, Prix_Billet)
- **pilote** (id_Pilote, Nom_Pilote, Prenom_Pilote, Date_Emb, Salaire_Mensuel)
- **avion** (id_Avion, Type_Avion, Nbre_Passagers)

Rappel (2/2) : SQL et jointures

SQL (Structured Query Language) est le langage principal des bases relationnelles :

- création et modification des données,
- interrogation des tables.

Structure générale d'une requête :

SELECT *liste d'attributs*
FROM *tables*
WHERE *conditions*

Jointures : relier des tables via des clés (ex : `vol.Code_Pilote` → `pilote.id_Pilote`).

Exemple de jointure

```
SELECT *
FROM vol AS v INNER JOIN avion AS a
ON v.Code_Avion = a.id_Avion
```

Les fonctions ensemblistes : MAX

MAX fournit la valeur maximale d'un attribut.

- L'attribut peut être un nombre ou une chaîne de caractères.

Durée maximale d'un vol Paris – San Francisco

```
SELECT MAX(v.Duree_Vol)
FROM vol AS v
WHERE v.Ville_Depart='Paris'
AND v.Ville_Arrivee='San Francisco';
```

Les fonctions ensemblistes : MIN

MIN fournit la valeur minimale d'un attribut.

- L'attribut peut être un nombre ou une chaîne de caractères.

Date à laquelle le premier pilote a été embauché

```
SELECT MIN(p.Date_Emb)
FROM pilote AS p;
```

Les fonctions ensemblistes : COUNT

COUNT permet de compter le nombre d'occurrences.

Nombre d'avions ayant décollé à 10:00

```
SELECT COUNT(*)
FROM vol AS v
WHERE v.Heure_Decollage='10:00';
```

Les fonctions ensemblistes : SUM

SUM permet d'additionner les valeurs d'un même attribut.

- L'attribut doit être numérique.

Nombre total de passagers transportés par le pilote nommé Simon

```
SELECT SUM(a.Nbre_Passagers)
FROM vol AS v INNER JOIN avion AS a
ON v.Code_Avion=a.id_Avion
INNER JOIN pilote AS p
ON v.Code_Pilote=p.id_Pilote
WHERE p.Nom_Pilote='Simon';
```

Les fonctions ensemblistes : AVG

AVG permet de calculer la moyenne des valeurs d'un même attribut.

- L'attribut doit être numérique.

Moyenne des salaires des pilotes embauchés avant le 1er janvier 2017

```
SELECT AVG(p.Salaire_Mensuel)
FROM pilote AS p
WHERE p.Date_Emb<'2017-01-01';
```

Possibilité de faire des calculs dans SELECT

On peut effectuer des calculs directement dans la clause `SELECT`.

Recette globale pour le vol V01

```
SELECT a.Nbre_Passagers*v.Prix_Billet
FROM vol AS v INNER JOIN avion AS a
ON v.Code_Avion=a.id_Avion
WHERE v.id_Vol='V01';
```

Recette globale pour tous les vols

```
SELECT SUM(a.Nbre_Passagers*v.Prix_Billet)
FROM vol AS v INNER JOIN avion AS a
ON v.Code_Avion=a.id_Avion;
```

DISTINCT

DISTINCT permet de n'obtenir qu'une seule fois chaque occurrence.

Types d'avions

```
SELECT DISTINCT a.Type_Avion
FROM avion AS a;
```

ORDER BY

ORDER BY permet d'ordonner par ordre croissant ou décroissant (`DESC`).

Type des avions des vols au départ de Londres (ordre alphabétique)

```
SELECT DISTINCT a.Type_Avion, a.Nbre_Passagers
FROM avion AS a INNER JOIN vol AS v
ON a.id_Avion=v.Code_Avion
WHERE v.Ville_Depart='Londres'
ORDER BY a.Type_Avion;
```

Nom et date d'embauche des pilotes (ordre décroissant)

```
SELECT p.Nom_Pilote, p.Date_Emb
FROM pilote AS p
ORDER BY p.Date_Emb DESC;
```

Tri sur plusieurs clés

- La 1ère clé de tri est `Ville_Depart` (ordre décroissant).
- La 2ème clé de tri est `Duree_Vol` (ordre croissant).
- Pour une même valeur de `Ville_Depart`, les occurrences sont triées selon `Duree_Vol`.

Ville de départ (DESC) et durée des vols (ASC)

```
SELECT v.Ville_Depart, v.Duree_Vol
FROM vol AS v
ORDER BY v.Ville_Depart DESC, v.Duree_Vol;
```

GROUP BY

GROUP BY permet de faire des regroupements.

Nombre total de passagers transportés sur chaque type d'avion

```
SELECT a.Type_Avion, SUM(a.Nbre_Passagers)
FROM vol AS v INNER JOIN avion AS a
ON v.Code_Avion=a.id_Avion
GROUP BY a.Type_Avion;
```

HAVING

HAVING est souvent utilisé avec `GROUP BY` pour spécifier des caractéristiques du groupement.

Passagers par ville de départ si le 1er vol est après 9:00

```
SELECT v.Ville_Depart, SUM(a.Nbre_Passagers)
FROM vol AS v INNER JOIN avion AS a
ON v.Code_Avion=a.id_Avion
GROUP BY v.Ville_Depart
HAVING MIN(v.Heure_Decollage) > '09:00';
```

Remarque sur HAVING

Il n'est pas obligatoire d'utiliser la clause `HAVING` quand la caractéristique du groupement peut s'exprimer avec une condition simple.

Prix moyen des billets pour chaque trajet (sauf Madrid)

```
SELECT v.Ville_Depart, v.Ville_Arrivee, AVG(v.Prix_Billet)
FROM vol AS v
WHERE v.Ville_Arrivee <> 'Madrid'
GROUP BY v.Ville_Depart, v.Ville_Arrivee;
```

BETWEEN

BETWEEN a AND b permet de tester l'appartenance à un intervalle (bornes comprises).

Vols dont la durée est comprise entre 50 et 200 minutes

```
SELECT DISTINCT v.Ville_Depart, v.Ville_Arrivee
FROM vol AS v
WHERE v.Duree_Vol BETWEEN 50 AND 200;
```

LIKE

LIKE permet d'utiliser des jokers dans des chaînes de caractères.

- _ représente un unique caractère
- % représente une chaîne de longueur quelconque

Numéro des vols effectués sur un avion de type AirBus

```
SELECT v.id_Vol
FROM vol AS v INNER JOIN avion AS a
ON v.Code_Avion=a.id_Avion
WHERE a.Type_Avion LIKE 'AirBus%';
```

Numéro des vols qui ont décollé à xx:00

(L'attribut `Heure_Decollage` est une chaîne de caractères)

```
SELECT v.id_Vol
FROM vol AS v
WHERE v.Heure_Decollage LIKE '__:00';
```

IS NULL / IS NOT NULL

IS NULL et **IS NOT NULL** permettent de vérifier si un attribut est renseigné ou non.

Nom des pilotes dont on ne connaît pas la date d'embauche

```
SELECT p.Nom_Pilote
FROM pilote AS p
WHERE p.Date_Emb IS NULL;
```

Fin