BNF101 Base de Données Relationnelles

Lecture et extraction des données

Requêtes Select simples

Requêtes Select avec jointures internes

LMD : SELECTION DES DONNEES

Les requêtes de sélection.

Après avoir créé des tables et peuplé ces tables avec des données, nous allons aborder les requêtes d'extraction de données.

Pour cela nous allons voir comment écrire des requêtes SQL.

Cela va nous permettre de :

- lire complètement une table,
- lire une partie d'une table,
- faire des jointures de différentes tables,
- faire des sous-requêtes imbriquées.

LMD : SELECTION DES DONNEES

Lorsqu'on désire consulter tous les attributs d'une table on n'est pas obligé de les citer tous, on peut écrire:

SELECT * **FROM** nom_de table;

Exemple :

SELECT * FROM client;

Si on désire consulter uniquement quelques lignes d'une table et ce relativement à une ou plusieurs conditions, la syntaxe peut être enrichie et devient:

SELECT nom_colonne_1,… , nom_colonne_j **FROM** nom_de table;

Exemple :

SELECT numcli, nomcli, ville FROM client;

LMD : SELECTION DES DONNEES
LES TRIS

Lorsque l'on lance la requête SELECT * FROM client :

On constate que les résultats sont triés dans l'ordre de l'insertion des lignes.

Si vous avez inséré le client dont le numcli =8 avant d'avoir inséré le client dont le numcli =1, la ligne pour le numcli=8 sera positionné avant la ligne pour le numcli=1.

On peut demander que les résultats soient affichés en étant triés par une ou plusieurs colonnes en utilisant la clause ORDER BY.

La syntaxe du select devient:

SELECT nom_colonne_1,… , nom_colonne_j **FROM** nom_de table **ORDER BY** nom_colonne **ASC ou DESC** ;

ASC ascendant et DESC POUR descendant.

LMD : SELECTION DES DONNEES
LES FILTRES

Pour ajouter un filtre à la requête de sélection, il faut ajouter le mot clé **WHERE** après le nom de la ou des tables**.**

On indique ensuite les conditions du filtre.

Exemple :

SELECT * FROM Client WHERE (conditions du filtre);

On peut répartir la consultation d'une table selon des critères en trois groupes:

1) Consultation avec condition de comparaison,

2) Consultation avec condition en sous-requête,

3) Consultation avec condition de jointure,

LMD : SELECTION DES DONNEES
LES FILTRES

1) Consultation avec condition de comparaison

Nous pouvons citer différentes formes de spécification de condition de comparaison :

- Forme 1 : opérateurs de comparaison : =, <, >

Exemple : écrire la requête permettant d'extraire les nom des produits dont le prix est compris entre 10 et 100 euros exclus (inégalité stricte).

Select nompro from PRODUIT where prixuni **>** 10 **AND** prixuni **<** 100;

- forme 2 : expression **opérateur-relationnel expression**

Les opérateurs relationnels les plus courants sont : AND, OR

- forme 3 : expression [**NOT**] **BETWEEN** expression **AND** expression,

Exemple : écrire la requête permettant d'extraire les nom des produits dont le prix est compris entre 10 et 100 euros inclus (inégalité non stricte).

Select nompro from PRODUIT where prixuni **BETWEEN** 10 **AND** 100;

BETWEEN équivaut à prixuni **>=** 10 **AND** prixuni **<=** 100;

LMD : SELECTION DES DONNEES
LES FILTRES

Exemple 2 :

écrire la requête permettant d'extraire les noms des clients parisiens :

Select nomcli from CLIENT where ville = 'paris';

nota bene: la valeur du paramètre situé entre les ' ' doit avoir la même casse que la valeur enregistrée dans la base de données.

Exemple 3 : écrire la requête permettant d'extraire les nom de produit dont le prix est supérieur à 100 euros.

Select nompro from PRODUIT where prixuni > 100;

Exemple 4 :

écrire la requête permettant d'extraire les noms des clients parisiens ou lyonnais :

Select nomcli, ville from CLIENT where ville = 'paris' **OR** ville = 'lyon';

LMD : SELECTION DES DONNEES
LES FILTRES

- forme 4 : expression [**NOT**] **IN** (liste d'expression/requête),

Exemple 5 :

écrire la requête permettant d'extraire les noms des clients parisiens ou lyonnais :

Select nomcli, ville from CLIENT where ville **IN** ('paris', 'lyon');

Remarque : Cette requête ramène les mêmes résultats que la requête de l'exemple 4 mais elle utilise la forme d'une **liste**, plus compacte.

Cette forme permet aussi d'obtenir les résultats complémentaires en ajoutant le mot clé NOT :

requête permettant d'extraire les noms des clients ni parisiens ni lyonnais :

Select nomcli, ville from CLIENT where ville **NOT IN** ('paris', 'lyon');

LMD : SELECTION DES DONNEES
LES FILTRES :REMARQUE SUR L'UTILISATION D'UNE LISTE DANS LE FILTRE.

Dans la mesure où on utilise une liste pour gérer les valeurs d'un filtre, on peut construire la liste de façon **statique**, avec des données en dur.

⇒ Cf exemple : select nomcli, ville from CLIENT WHERE ville IN

('paris', 'lyon', 'marseille');

On peut aussi construire la liste de façon **dynamique** en insérant une sous-requête select.

Exemple : on peut avoir une table qui contient des noms de ville et leur nombre d'habitants entre-autres.

Ainsi, on peut imbriquer une sous-requête pour obtenir par exemple la liste des villes de plus de 100 000 habitants.

On détaillera la syntaxe des sous-requêtes SELECT dans le prochain cours (cours 11).

LMD : SELECTION DES DONNEES
LES FILTRES

- forme 5 : colonne [**NOT**] **LIKE** 'chaine',

Exemple 6 :

Ecrire la requête qui permet d'extraire les numéros des clients dont le nom commence par 's':

SELECT NumCli **FROM** CLIENT **WHERE** NomCli **LIKE** 's%';

Exemple 7 :

Ecrire la requête qui permet d'extraire les numéros des clients dont le nom se termine par 's':

SELECT NumCli **FROM** CLIENT **WHERE** NomCli **LIKE** '%s';

Exemple 7 :

Ecrire la requête qui permet d'extraire les numéros des clients dont le nom contient la sous-chaîne par 'au':

SELECT NumCli **FROM** CLIENT **WHERE** NomCli **LIKE** '%au%';

Remarques :

il faut être vigilant sur la casse (minuscule, majuscule) des caractères que l'on saisit.

Le caractère % joue dans le langage SQL le role du caractère * dans les recherches textuelles sous linux ou sous windows.

LMD : PRODUIT CARTESIEN ET JOINTURES INTERNES.

Nous avons extrait les données d'une seule table

- avec ou sans filtre,
- avec ou sans tri et
- avec toutes les colonnes ou avec une partie des colonnes.

L'objectif de la suite de ce cours consiste à extraire les données enregistrées dans plusieurs tables en utilisant des jointures internes entre les tables.

Nous allons commencer par comparer la notion de jointure interne (produit cartésien partiel) avec la notion de produit cartésien complet.

LMD : SELECTION DES DONNEES – LE PRODUIT CARTESIEN COMPLET

Produit cartésien :

concaténation de toutes les lignes de la première table avec toutes

les lignes de la seconde table.

Exemple :

Select **client.**numcli,nomcli,numcom,**commande.**numcli, datecom from client, commande;

Cette requête ne possède pas de critère de jointure entre une table et l'autre.

Dans ce cas, le moteur SQL calcule le produit cartésien des deux ensembles, c'est-à-dire qu'à chaque ligne de la première table (client), il accole l'ensemble des lignes de la seconde (commande).

LMD : SELECTION DES DONNEES – RESULTAT DU PRODUIT CARTESIEN COMPLET

NUMCLI	NOMCLI	NUMCOM	NUMCLI	DATECOM
1	redford	1	2	22/11/06
2	spacey	1	2	22/11/06
3	roberts	1	2	22/11/06
4	benini	1	2	22/11/06
5	maurante	1	2	22/11/06
6	pausini	1	2	22/11/06
7	murphy	1	2	22/11/06
8	Eastwood	1	2	22/11/06
1	redford	2	1	15/10/06
2	spacey	2	1	15/10/06
3	roberts	2	1	15/10/06

NUMCLI	NOMCLI	NUMCOM	NUMCLI	DATECOM
4	benini	2	1	15/10/06

Le tableau de résultats est volontairement tronqué pour l'affichage dans ce slide

8	Eastwood	7	7	17/08/18

64 ligne(s) selectionnee(s).

Dans cet extrait de tableau de résultats, la commande de la table commande dont le numcom est égal à 1 est multipliée (associée) à chaque ligne (enregistrement) de la table client.
Ici, seule la ligne qui concerne le client dont le numcli est égale à 2 correspond à la commande passée par ce client => cf correspondance entre la valeur de la **clé primaire client.numcli** et la valeur de la **clé étrangère commande.numcli**.

LMD : SELECTION DES DONNEES – LES JOINTURES INTERNES

Jointure :

lien entre 2 tables disposant d'au moins une colonne commune (sémantiquement). On associe à chaque ligne de la première table la ligne de la seconde table qui possède la colonne commune.

Une jointure est un lien entre 2 tables disposant d'une ou plusieurs colonnes communes sémantiquement.

Exemple : valeur de la clé primaire = valeur des clés étrangères associées.

L'opération de jointure consistera à créer une table temporaire composé des lignes satisfaisant la condition de jointure.

LMD : SELECTION DES DONNEES – LES JOINTURES INTERNES

Exemple :

Pour connaître les clients (numéro client, nom, adresse) qui ont passé au moins une commande, on est amené à utiliser le lien entre les tables Client et Commandes puis d'en extraire seulement les lignes satisfaisant la condition suivante :

SELECT **Client.**NumCli, Client.NomCli, Client.AdresseCli, Commande.Numcom, **Commande.**Numcli, Commande.DateCom FROM Client, Commande **WHERE Client.NumCli = Commande.NumCli**;

Autre syntaxe :

SELECT Client.NumCli, Client.NomCli, Client.AdresseCli, Commande.Numcom, Commande.Numcli, Commande.DateCom FROM Client **INNER JOIN** Commande **ON Client.NumCli = Commande.NumCli**;

Dans cette forme, la clause WHERE disparaît au profit de la clause INNER JOIN ON.

LMD : SELECTION DES DONNEES – RESULTAT DE LA JOINTURE INTERNE

NUMCLI	NOMCLI	NUMCOM	NUMCLI	DATECOM
1	redford	2	1	15/10/06
1	redford	3	1	18/09/18
2	spacey	1	2	22/11/06
3	roberts	4	3	17/11/18
3	roberts	6	3	13/08/18
5	maurante	5	5	13/08/16
7	murphy	7	7	17/08/18
8	Eastwood	8	8	16/11/18

8 ligne(s) selectionnee(s).

On obtient la liste des clients qui ont réellement passé des commandes.

La clé primaire numcli dans la table client a la même valeur que la clé étrangère numcli dans la table commande.

LMD : SELECTION DES DONNEES – LES JOINTURES INTERNES

Remarque :

Dans la mesure où on retrouve la colonne numcli dans la table client et dans la table commande, il faut la préfixer par le nom de sa table.

Client.numcli et commande.numcli.

Par soucis d'homogénéité et pour éviter toute ambiguité, il est préférable de préfixer toutes les colonnes par le nom de leur table d'appartenance.

Dans ce cours, on se limite au cas des equi-jointures (signe = entre les 2 tables).

LMD : SELECTION DES DONNEES – LES JOINTURES INTERNES

Exercices sur les jointures :

Lister tous les **clients Parisiens** qui ont passé une commande entre les dates du premier janvier 1993 et aujourd'hui.

Pour cette requête, on affichera les colonnes suivantes :

Numéro du client, nom du client, la ville du client, le numéro de la commande, la date de la commande.

Lister tous **les produits** qui n'ont pas été commandés entre le premier janvier 2018 et aujourd'hui.

Pour cette requête, on affichera les colonnes suivantes :

Numéro du produit, nom du produit.

LMD : SELECTION DES DONNEES – LES JOINTURES INTERNES

Correction des exercices sur les jointures :

Lister tous les clients Parisien qui ont passé une commande entre les dates du premier janvier 1993 et aujourd'hui :

```
SELECT Client.NumCli, Client.NomCli, Client.ville, Commande.DateCom, commande.NumCom FROM Client, Commande
WHERE Client.NumCli=Commande.NumCli AND Client.Ville= 'paris'
AND Commande.DateCom BETWEEN '01/01/1993' AND CURRENT_DATE;
```

Nota bene : il faut bien respecter le format 'jj/mm/aaaa'.

>> Le SGBD ORACLE renvoie l'erreur (ORA-01861: le litteral ne concorde pas avec le format chaine de caractères) si le format de date n'est pas respecté.

Ou bien 2ème syntaxe pour le format de date.

```
SELECT Client.NumCli, Client.NomCli, Client.ville, Commande.DateCom, commande.NumCom FROM Client, Commande
WHERE Client.NumCli=Commande.NumCli AND Client.Ville= 'paris'
AND Commande.DateCom BETWEEN to_date('1993/01/01', 'yyyy/mm/dd') AND CURRENT_DATE;
```

LMD : SELECTION DES DONNEES – LES JOINTURES INTERNES

Correction des exercices sur les jointures :

Lister tous les clients Parisien qui ont passé une commande entre les dates du premier janvier 1993 et aujourd'hui :

syntaxe avec la clause INNER JOIN.

SELECT Client.NumCli, Client.NomCli, Client.ville, Commande.DateCom, commande.NumCom FROM Client **INNER JOIN** Commande

ON Client.NumCli=Commande.NumCli **WHERE** Client.Ville= 'paris'

AND Commande.DateCom BETWEEN to_date('1993/01/01', 'yyyy/mm/dd') AND CURRENT_DATE;

LMD : SELECTION DES DONNEES – LES JOINTURES INTERNES

Correction des exercices sur les jointures :

Lister tous les produits qui n'ont pas été commandés entre le premier janvier 2018 et aujourd'hui :

```
SELECT Produit.NumPro, Produit.NomPro FROM Produit, LigneCom, Commande
WHERE Commande.NumCom=LigneCom.NumCom AND Produit.NumPro = LigneCom.NumPro
AND Commande.DateCom NOT BETWEEN '01/01/2018' AND CURRENT_DATE;
```

Nota bene : il faut bien respecter le format 'jj/mm/aaaa'.

Ou bien 2ème syntaxe pour le format de date :

```
SELECT Produit.NumPro, Produit.NomPro FROM Produit, LigneCom, Commande
WHERE Commande.NumCom=LigneCom.NumCom AND Produit.NumPro = LigneCom.NumPro
AND Commande.DateCom NOT BETWEEN to_date('2018/01/01', 'yyyy/mm/dd') AND CURRENT_DATE;
```

Remarque : ans cette syntaxe, l'ordre des jointures importe peu; on peut aussi écrire :

```
WHERE Produit.NumPro = LigneCom.NumPro
AND Commande.NumCom=LigneCom.NumCom
```

LMD : SELECTION DES DONNEES – LES JOINTURES INTERNES

Correction des exercices sur les jointures :

Lister tous les produits qui n'ont pas été commandés entre le premier janvier 2018 et aujourd'hui :

Syntaxe avec la clause INNER JOIN :

SELECT Produit.NumPro, Produit.NomPro **FROM** Produit

INNER JOIN lignecom ON Produit.NumPro = LigneCom.NumPro

INNER JOIN commande ON Commande.NumCom=LigneCom.NumCom

WHERE Commande.DateCom NOT BETWEEN to_date('2018/01/01', 'yyyy/mm/dd') **AND** CURRENT_DATE;

Nota Bene :

- **Dans cette syntaxe, l'ordre des lignes INNER JOIN est sensible : on doit respecter la séquence FROM produit >> ligneCom >> commande.**
- **Le mot clé INNER JOIN doit être indiqué autant de fois qu'il y a de jointure(s) interne(s).**
- **Il ne faut pas ajouter l'opérateur AND entre chaque INNER JOIN.**

LMD : SELECTION DES DONNEES – LES JOINTURES INTERNES

Correction des exercices sur les jointures (suite) :

Remarque :

on peut faire plus simple et éviter d'écrire toujours le nom de la table en indiquant des alias (ci-dessous en gras).

SELECT **P.**NumPro, **P.**NomPro FROM Produit **P**, LigneCom **LC**, Commande **C** WHERE **C.**NumCom=**LC.**NumCom AND **P.**NumPro = **LC.**NumPro AND **C**.DateCom **NOT** BETWEEN '01/01/2018' AND CURRENT_DATE;