BASE INFO : SQL

1 ORDRE SELECT

Ordre SQL

SELECT <Liste des champs> FROM <Liste des tables concernées>
[WHERE <Condition sur les champs à lister>]
[GROUP BY <Expression des critères de regroupement>]
[HAVING <Condition sur les champs regroupés>]
[ORDER BY <Expression des critères de tri>];

Exemples

```
SELECT *
FROM CLIENT;(toutes les colonnes)
SELECT CPTCLI, NOMCLI FROM CLIENT ;(une partie des colonnes)
SELECT DISTINCT VILCLI FROM CLIENT ;(élimine les doublons)
SELECT NOMCLI AS "Nom" FROM CLIENT ;(AS renomme les colonnes)
```

Expression des restrictions (clause where)

Opérateurs de comparaison

** LIKE permet de faire des comparaisons sur des chaines grâce à des caractères, appelés caractères jokers :*

=, <>, >, >=, < ,<=	*CDPCLI* <> "*97300*"
BETWEEN	*DATFACT* BETWEEN #*1/1/2006*# And #*1/31/2006*#
IN	REFPRO IN ("P00001","P00002")
LIKE*	Comparaison de chaine de caractère : VILCLI LIKE "*OU*"
IS NULL, IS NOT NULL	AD2CLI IS NULL

Caractères jokers de l'opérateur LIKE

SQL	*ACCESS*	*Effet*
%	*	Remplace n'importe quelle séquence de caractères
_	?	Remplace n'importe quel caractère
[-]	[-]	Définit un intervalle de caractères (par exemple [a-z])

Opérateurs logiques

AND	(REFPRO="P00002" AND QTP_VENDU > 50)
OR	(REFCLI = "4110001" OR CDPCLI "97354")
NOT	REFPRO NOT IN ("P00001","P00002")

Exemples

```
SELECT * FROM CLIENT WHERE VILCLI IN
('CAYENNE','KOUROU');
SELECT * FROM CLIENT WHERE AGECLI NOT BETWEEN 30 AND 40;
SELECT * FROM CLIENT WHERE AD2CLI IS NULL;
SELECT * FROM CLIENT WHERE AD2CLI IS NOT NULL;
```

Trier des données

Clause ORDER BY SQL permet d'organiser les résultats d'une requête à l'aide de la clause ORDER BY suivie des critères de tri. Les mots clés ASC ou DESC précisent si le tri se fait de manière croissante (par défaut) ou décroissante.

Exemples

```
SELECT * FROM CLIENT ORDER BY NOMCLI;                    (tri ascendant par défaut)
SELECT * FROM CLIENT ORDER BY VILCLI,NOMCLI;             (tri par ville puis nom)
SELECT * FROM CLIENT ORDER BY SEXCLI ASC, AGECLI DESC;
```

Relier des tables : Utilisation de la clause Where Il s'agit ici d'exprimer les relations entre les tables.

Exemples

```
SELECT * FROM tab1,tab2;    (jointure sans qualification =produit cartésien)
SELECT * FROM FACTURE,VENDRE WHERE FACTURE.NFACT=VENDRE.NFACT;
SELECT * FROM FACTURE,VENDRE, PRODUIT
      WHERE FACTURE.NFACT=VENDRE.NFACT AND VENDRE.REFPRO=PRODUIT.REFPRO;
```

Regroupement des résultats Clause GROUP BY La clause GROUP BY permet de définir les critères de regroupement des données.

```
SELECT * FROM TAB1 GROUP BY COL1; (ne fonctionne pas sous Access)
SELECT VILCLI FROM CLIENT GROUP BY VILCLI;
SELECT VILCLI, CDPCLI FROM CLIENT GROUP BY VILCLI, CDPCLI;
```

Clause HAVING La clause HAVING va de pair avec la clause GROUP BY, elle permet, d'appliquer une restriction sur les regroupements créés avec la clause GROUP BY.

SELECT VILCLI, CDPCLI FROM CLIENT GROUP BY VILCLI, CDPCLI HAVING CDPCLI > "97300";

SQL :

SELECT NFACT, AVG(QTP_VENDU) AS MOYENNE FROM VENDRE GROUP BY NFACT HAVING **MOYENNE** >10;

ACCESS :

SELECT NFACT, AVG(QTP_VENDU) AS MOYENNE FROM VENDRE GROUP BY NFACT HAVING **AVG(QTP_VENDU)**>10;

Expression des manipulations de données

Effectuer des calculs

Calcul	Exemple
Ajouter un nombre à un champ	QTP_VENDU + 5
Soustraire un nombre d'un champ	QTP_VENDU - 5
Multiplier un champ par un nombre	PUVPRO * 1.2
Diviser un champ par un nombre	QTP_VENDU / 2
Calculer avec des champs	(QTP_VENDU * PUVPRO) / 1.05

Manipulation de chaînes de caractères (Access)

Manipulation	Exemple
Concaténer deux chaînes	CDPCLI & VILCLI
Extraire une chaine de caractères SUBSTRING(champ, début, longueur)	SUBSTRING(CDPCLI, 3, 2)
Extraire la partie gauche d'une chaine LEFT(champ, longueur)	LEFT(CDPCLI, 2)
Extraire la partie droite d'une chaine RIGHT(champ, longueur)	RIGHT(CDPCLI, 3)
Mettre une chaine en minuscule	LCASE(VILCLI)
Mettre une chaine en majuscule	UCASE(VILCLI)
Longueur d'une chaine de caractères	DATALENGTH(AD2CLI)

Exemples

SELECT DESPRO, PUVPRO*1.2 AS [PRIX TTC] FROM PRODUIT;

SELECT * FROM CLIENT WHERE LEFT(CDPCLI) <> "973");

Les fonctions statistiques

Principe

Il est souvent utile d'effectuer des statistiques sur les données, c'est l'objet des fonctions d'agrégation. Ces opérations, mentionnées dans la clause SELECT, effectuent des regroupements selon les critères de la clause GROUP BY. En cas d'absence de clause GROUP BY, le regroupement est effectué sur la totalité des données.

Fonctions d'agrégation

Fonction	Opération effectuée
AVG	moyenne
COUNT	nombre d'éléments
MAX	maximum
MIN	minimum
SUM	somme

Exemples

SELECT COUNT(*)FROM CLIENT;

SELECT COUNT(*) AS [NOMBRE DE CLIENTS],
LCASE(VILCLI) FROM CLIENT GROUP BY VILCLI;

SQL :

SELECT NFACT, AVG(QTP_VENDU) AS MOYENNE FROM VENDRE GROUP BY NFACT HAVING **MOYENNE** >10;

ACCESS :

SELECT NFACT, AVG(QTP_VENDU) AS MOYENNE FROM VENDRE GROUP BY NFACT HAVING **AVG(QTP_VENDU)**>10;

Requête imbriquées SQL

Principe

Une requête SQL renvoie une liste, cette liste peut être utilisée afin d'effectuer des comparaisons comme avec l'opérateur IN :

```
SELECT * FROM CLIENT WHERE VILCLI IN ('CAYENNE','KOUROU');
```

La liste entre parenthèse sera alors remplacée par un ordre SELECT.

Opérateurs utilisables

ANY	Test si une ou plusieurs lignes du résultat d'une sous-requête répondent à la condition spécifiée
ALL	Test si toutes les lignes du résultat d'une sous-requête répondent à la condition spécifiée.
IN	Test si un élément est présent dans les lignes du résultat d'une sous-requête.
NOT IN	Test si un élément n'est pas présent dans les lignes du résultat d'une sous-requête

```
SELECT * FROM TAB1 WHERE PRIX > (SELECT MIN(PRIX) FROM TAB2);
SELECT * FROM TAB1 WHERE NOM NOT IN (SELECT NOM FROM TAB2);
SELECT * FROM TAB1 WHERE PRIX > ALL (SELECT PRIX FROM TAB2);      (SUP. A TTES LES VALEURS)
SELECT * FROM TAB1 WHERE PRIX > ANY (SELECT PRIX FROM TAB2) ;     (SUP. A AU MOINS 1)
```