TP 5 : Le tri

Rappel du contexte d'étude "magasin" L'entreprise que nous allons étudier est un commerce de proximité de type "supérette" organisé en rayons contenant des articles de diverses marques. Lors du passage en caisse, on conserve la date, le détail des articles achetés ainsi que le montant total qui a été réglé pour cet achat. On conserve également les informations concernant les fournisseurs et les livraisons. Pour chaque livraison, on mémorise le fournisseur ainsi que le ou les lots qui ont été livrés. Chaque lot concerne un seul article. Un lot est identifié par son numéro de lot et par le code-barres de l'article qu'il concerne. Un article possède un identifiant unique, le codebarres. C'est son appartenance à un lot qui permet de retrouver les informations qui le concernent (dates de fabrication, de livraison et de péremption). On peut d'ailleurs voir sur les photos suivantes quatre articles ayant le même codebarres, mais des numéros de lot différents et donc des dates de péremption différentes.

Les tris

SQL nous offre la possibilité de trier les résultats d'une requête grâce à l'instruction "ORDER BY". Par défaut, l'instruction SELECT nous retourne les enregistrements dans l'ordre dans lequel ils apparaissent dans la table.

" Pour les tris nous utilisons la clause « ORDER BY » qui nous permet d'organiser les enregistrements en s'appuyant sur n'importe quel type de champ :
- Ordre alphabétique : si le champ est basé sur des chaînes de caractères ;
- Ordre chronologique : si le champ est basé sur des dates ;
- Ordre incrémental : si le champ est basé sur des nombres ;
- Ordre de vérité : si le champ est basé sur des valeurs booléennes.

Par exemple, on souhaite obtenir la liste des fournisseurs triés par ordre alphabétique. Cela signifie que l'on souhaite tous les champs et tous les enregistrements de la table "fournisseur", mais cela veut également dire que l'on souhaite effectuer un tri "croissant" sur le champ raisonSociale.

En langage SQL ceci nous donne :

```
SELECT *
FROM fournisseur
ORDER BY raisonSociale
```

Testez la requête

Le tri par ordre croissant se fait grâce à la clause "ASC" qui est normalement positionnée après le champ à trier.
Le type de tri par défaut, activé par l'instruction "ORDER BY", est "ASC", ce qui fait que cette clause est facultative.

```
SELECT *
FROM fournisseur
ORDER BY raisonSociale ASC
```

Testez la requête

" ASC " permet de préciser que l'on souhaite effectuer un tri "croissant":
-si c'est un champ de type "chaînes de caractères", l'affichage se fera de "A" à "Z" -si c'est un champ de type "date", l'affichage commencera par les dates les plus
anciennes pour se terminer avec les plus récentes
-si c'est un champ de type "numérique", l'affichage commencera par les nombres les plus "petits" pour se terminer avec les plus grands si c'est un champ de type "booléen", l'affichage commencera par les valeurs "fausses" (false en anglais) pour se terminer par les valeurs "vraies" (true).

On souhaite à présent obtenir la liste des lots, mais on souhaite également que l'affichage commence par les livraisons les plus récentes.

Cela signifie que l'on souhaite effectuer un tri "décroissant" (date les plus récente en premier) :
Dans la requête SQL, on peut voir que la clause "DESC" a été ajoutée après le champ à trier :

```
SELECT *
FROM lot
ORDER BY dateLivraison DESC
```

Testez la requête

" DESC " nous permet de préciser que l'on souhaite effectuer un tri "décroissant", ce qui revient à faire l'inverse de "ASC".

Il est possible d'effectuer plusieurs tris consécutifs.
Par exemple, on souhaite afficher les articles triés par rayon par ordre croissant (idRayon), mais également triés par prix, les moins chers en premier .

En SQL , on peut voir que le tri porte d'abord sur "idRayon" et ensuite sur "prix"

```
SELECT *
FROM article
ORDER BY idRayon, prix
```

Testez la requête

L'ordre de tri sera bien entendu croissant ("ASC"), rappelons-le puisque c'est l'action par défaut de l'instruction "ORDER BY".
La requête précédente peut donc s'écrire de la manière suivante :

```
SELECT *
FROM article
ORDER BY idRayon ASC, prix ASC
```

Testez la requête

Enfin, il est possible de combiner les deux ordres de tri.
Par exemple, on souhaite obtenir la liste des fournisseurs en commençant par les plus éloignés (distancekm) et en triant les noms par ordre alphabétique (sans ce tri, "Surgel'2000" est placé avant "Livre boite") :

Dans la requéte SQL, on peut voir que le tri porte d'abord sur "distanceKm" et que ce champ sera trié dans l'ordre décroissant ("DESC") et ensuite que le tri porte sur "prix" et que ce champ sera trié par ordre croissant (par défaut) :

```
SELECT *
FROM fournisseur
ORDER BY distanceKm DESC, raisonSociale
```

Testez la requête

Exercice :

Réaliser les requêtes permettant d'obtenir les informations ci-dessous

- la liste des marques par ordre alphabétique
- la liste des articles en affichant les articles "bio" en premier
- la liste des marques avec les articles correspondants. Les noms de marques doivent être triés par ordre alphabétique inverse et les libellés d'articles par ordre alphabétique
- la liste des lots avec le nom des fournisseurs et le libellé des articles en commençant par la livraison la plus ancienne. Les articles seront triés par ordre alphabétique