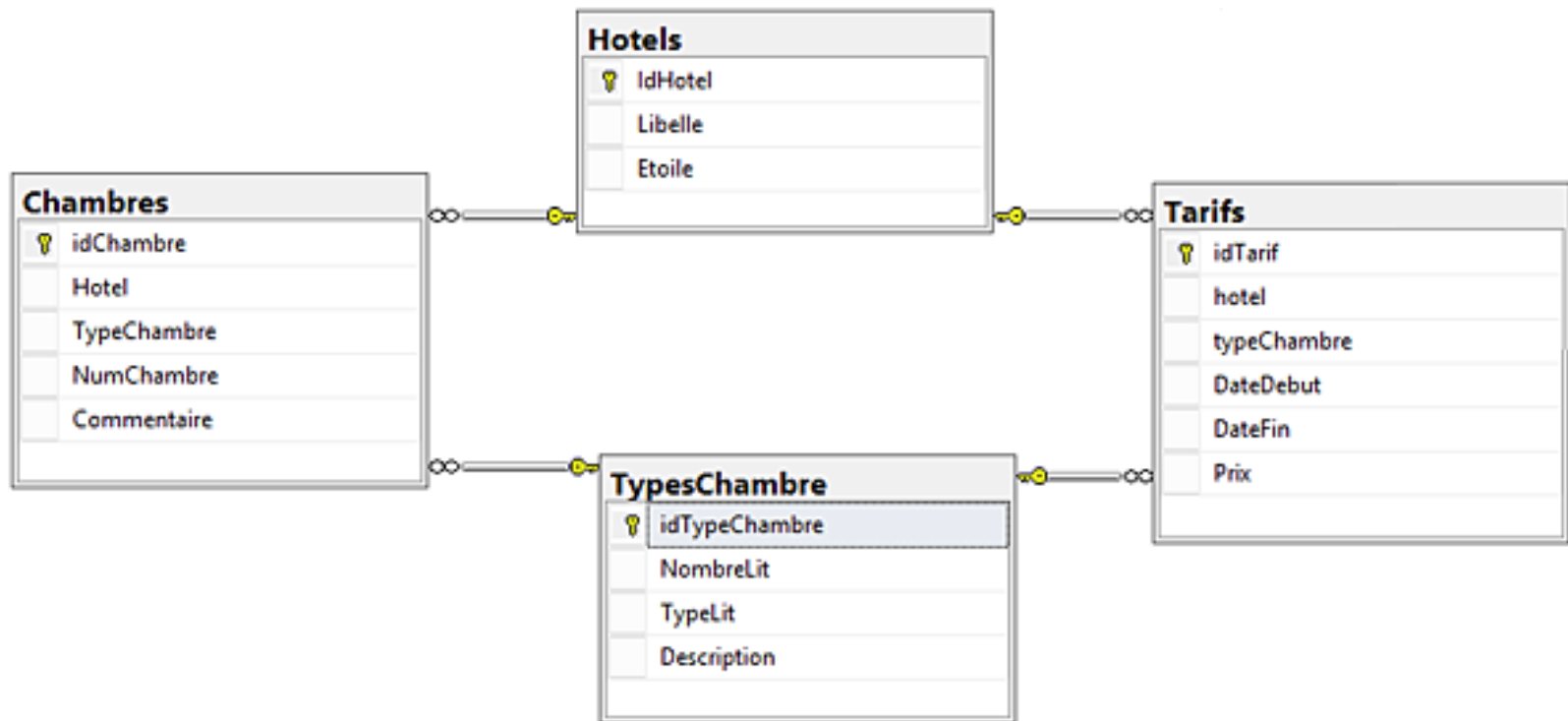


La sélection de données

Le SELECT est l'ordre le plus important et le plus utilisé en SQL. Avec cet ordre, nous pouvons ramener des lignes d'une ou plusieurs tables mais également transformer des données par l'utilisation de fonction ou encore réaliser des calculs.



Téléchargez le script de création de la BD

| idTypeChambre | NombreLit | TypeLit | Description |
|---------------|-----------|------------|--|
| 1 | 1 | lit simple | 1 lit simple avec douche |
| 2 | 2 | lit simple | 2 lits simples avec douche |
| 3 | 2 | lit simple | 2 lits simples avec douche et WC séparés |
| 4 | 1 | lit double | 1 lit double avec douche |
| 5 | 1 | lit double | 1 lit double avec douche et WC séparés |
| 6 | 1 | lit double | 1 lit double avec bain et WC séparés |
| 7 | 1 | lit XL | 1 lit double large avec bain et WC séparés |

utilisation la plus courante consiste à sélectionner des lignes dans une table comme ceci :

```
SELECT NombreLit, Description FROM  
TypesChambre;
```

Nous aurions pu écrire :

```
SELECT * FROM TypesChambre;
```

Si certaines colonnes ont le même nom mais appartiennent à des tables différentes, il faudra ajouter le nom de la table devant la colonne afin que le système sache quelle colonne prendre.

Il n'est pas obligatoire de mettre le nom mais pour une question de lisibilité et de maintenance.

```
SELECT Hotels.Libelle  
, Hotels.Etoile  
, Chambres.NumChambre  
, TypesChambre.Description  
FROM Chambres, Hotels, TypesChambre;
```

Les options DISTINCT et ALL

Par défaut, lors de l'exécution d'un SELECT, toutes les lignes sont ramenées (l'option ALL est automatique). Si l'on veut supprimer les doublons, il faut ajouter l'ordre DISTINCT.

```
SELECT NombreLit, TypeLit  
FROM TypesChambre;
```

| NombreLit | TypeLit |
|-----------|------------|
| 1 | lit simple |
| 2 | lit simple |
| 2 | lit simple |
| 1 | lit double |
| 1 | lit double |

```
SELECT DISTINCT NombreLit, TypeLit FROM  
TypesChambre;
```

| NombreLit | TypeLit |
|------------------|----------------|
| 1 | lit double |
| 1 | lit simple |
| 1 | lit XL |
| 2 | lit simple |

!La clause DISTINCT ne peut pas être utilisée avec des opérateurs de regroupement (voir le GROUP BY). En effet, les opérateurs de type COUNT ou SUM éliminent automatiquement les doublons. !

ORDER BY : un résultat trié sur certaines colonnes.

Doit être unique et toujours la dernière clause de la requête.

Le tri par défaut est ascendant, noté ASC (du plus petit au plus grand). Il est possible d'indiquer que l'on désire réaliser le tri en descendant en notant DESC.


ORDER BY <colonne 1> [ASC|DESC], <colonne 2> [ASC|DESC]...


```
SELECT hotel  
, typeChambre , DateDebut , prix  
FROM Tarifs  
ORDER BY DateDebut, prix DESC;
```


| Hotel | typeChambre | DateDebut | prix |
|-------|-------------|------------|--------|
| 1 | 7 | 01/04/2017 | 103,49 |
| 4 | 7 | 01/04/2017 | 103,49 |
| 4 | 6 | 01/04/2017 | 91,99 |
| 1 | 6 | 01/04/2017 | 91,99 |
| 1 | 5 | 01/04/2017 | 80,49 |
| 4 | 3 | 01/04/2017 | 80,49 |

Les options LIMIT, OFFSET

Les 10 premières lignes :

```
SELECT idTarif, hotel, typechambre, dateDebut, DateFin, Prix  
FROM   
Tarifs LIMIT 10;
```

Les 10 dernières lignes :

```
SELECT idTarif, hotel, typechambre, dateDebut, DateFin, Prix  
FROM   
Tarifs ORDER BY idTarif DESC LIMIT 10;
```

Les 10 premières lignes à partir de la cinquième ligne :

```
SELECT idTarif, hotel, typechambre, dateDebut, DateFin, Prix  
FROM Tarifs LIMIT 10 OFFSET 5;
```

ou :

```
SELECT idTarif, hotel, typechambre, dateDebut, DateFin, Prix  
FROM Tarifs LIMIT 5, 10;
```

Alias (pas le film!)

Dans une requête SQL qui comporte plusieurs tables, il est souhaitable d'attribuer un diminutif à chaque nom de table, que l'on appelle un alias, toujours dans l'optique de rendre les requêtes plus lisibles pour tous les programmeurs.

Cet alias peut aussi être utilisé pour un nom de colonne, mais également pour un résultat de fonction ou de SELECT imbriqués.

Trois syntaxes pour l'alias :

Il suit directement l'élément qu'il va supplanter.

```
SELECT NombreLit, Description FROM  
TypesChambre TYP;
```

Il peut être précédé de l'ordre AS.

```
SELECT NombreLit, Description FROM  
TypesChambre AS TYP
```

L'alias est aussi utilisé lorsque deux colonnes ont le même nom dans deux tables différentes. Il faut préciser au système l'origine de la colonne.

```
SELECT CH.NumChambre  
      , TYP.NombreLit  
      , T.Prix  
FROM Chambres CH  
      , TypesChambre TYP  
      , Tarifs T  
WHERE CH.TypeChambre = TYP.idTypeChambre  
AND T.TypeChambre = TYP.idTypeChambre;
```

La clause de restriction WHERE

```
SELECT idChambre, Hotel, NumChambre FROM  
Chambres WHERE TypeChambre = 3;
```

C'est simple !

```
SELECT CH.idChambre, CH.Hotel,  
CH.NumChambre FROM Chambres CH  
WHERE CH.TypeChambre IN (3, 6)  
AND idChambre > 5  
AND EXISTS  
    (SELECT Libelle FROM Hotels HT WHERE  
HT.idHotel = CH.Hotel);
```

Comment on traduit ça ? c'est du wolof ?