

# **Introduction à ORACLE**

13/02/2010

Carina Roels - Marie Labidoire

## Contenu

### **I. Quelques définitions**

- I.1 B.D. vs. SGBD.
- I.2 Le rapport ANSI-SPARC
- I.3 Utilisation d'un SGBD.

### **II. Les différentes structures de bases de données**

- II.1 L'approche hiérarchique
- II.2 L'approche réseau
- II.3 L'approche relationnelle

### **III. Les SGBDR**

- III.1 L'architecture fonctionnelle
- III.2 L'architecture physique

### **IV. Quelques concepts importants des SGBDR**

- IV.1 Conception et normalisation
- IV.2 Normalisation vs. Optimisation
- IV.3 Le langage d'accès aux données
- IV.4 Les index
- IV.5 Les séquences
- IV.6 Les synonymes
- IV.7 Les vues
- IV.8 Les triggers
- IV.9 Les procédures stockées

13/02/2010

Carina Roels - Marie Labidoire

## **I. Quelques définitions**

### **I.1 B.D. vs. SGBD**

#### **BASE DE DONNEES (BD)**

Définition AFNOR :

**Structure de données** permettant de recevoir, de stocker et de fournir à la demande des données à de multiples utilisateurs indépendants.

Cette structure décrit les informations et les associations qui peuvent exister entre elles.

13/02/2010

Carina Roels - Marie Labidoire

## I. Quelques définitions

### I.1 B.D vs. SGBD

#### SYSTEME de GESTION de BASE de DONNEES (SGBD)

Ensemble logiciel qui

- supporte les concepts de base **d'un modèle de données** ( Hiérarchique, Réseau, Relationnel, ... ).
- permet la mise en œuvre (**la définition**) de bases de données.
- permet la **manipulation** des informations contenues dans des bases de données.
- gère **l'intégrité** des données.
- gère les **transactions** et accès concurrents.
- permet des **reprises** après panne.

13/02/2010

Carina Roels - Marie Labidoire

#### Années 60

Premier développement des bases de données.  
(Fichiers reliés par pointeurs).

#### Premiers SGBD Milieu des années 60

Séparation de la description des données et des programmes.  
SGBD **hiérarchiques** ( IMS / DL1 )  
SGBD **réseau** ou **CODASYL** ( TOTAL / IDMS / IDS2... )

#### La deuxième génération des SGBD En laboratoire 1970. Commercialisés depuis 1982.

Le modèle **relationnel** vise à faciliter les accès aux données par les utilisateurs.  
Aujourd'hui largement répandu.  
SGBDR ( ORACLE / SYBASE / INGRES / INFORMIX / DB2 / RDB )

#### La troisième génération des SGBD. En laboratoire depuis les années 80.

**Basée sur des modèles à objets** intégrant une structuration conjointe des programmes et des données (classes).  
  
SGBDO ( ONTOS / OBJECTSTORE / VERSANT / ORION / O2 particulièrement intéressants pour des applications CAO ou industrielles )

#### Et :

Le modèle **relationnel / objet** qui tire profit des 2 technologies

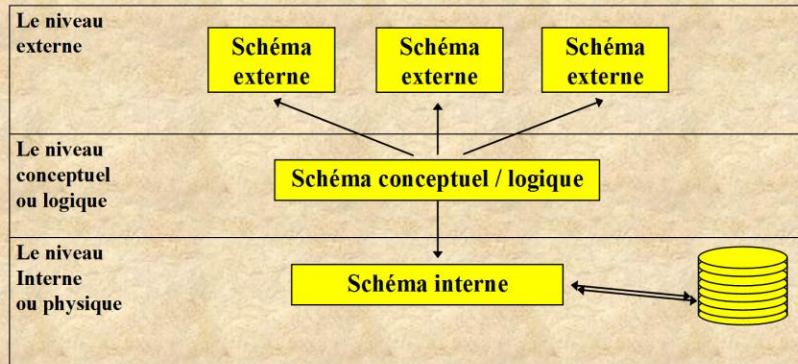


## I. Quelques définitions

### I.2 Le rapport ANSI-SPARC

#### ANSI-SPARC

Groupe de normalisation créé en 1969 pour étudier l'impact des SGBD sur les systèmes d'information et dont le résultat, publié en 1975, proposait 3 niveaux de description de données.



13/02/2010

Carina Roels - Marie Labidoire

#### Au niveau conceptuel ou logique,

l'univers réel est modélisé à l'aide des **concepts** de la **méthode** utilisée. A ce niveau on fait abstraction totale de l'utilisation des données ainsi que de leur implémentation physique.

#### Le niveau interne ou physique

décrit la façon dont les objets conceptuels seront **stockés sur la mémoire secondaire** (disque).

Le schéma décrit également la correspondance entre structures logiques de données et structures physiques. Le choix des structures de stockage est fait en tenant compte de l'utilisation qui sera faite des données (fréquence d'utilisation, sélectivité, etc. ) de façon à optimiser les accès à la base.

#### Le niveau externe

correspond aux vues que vont avoir **les utilisateurs (voire, les applications)** des données. Ces différentes vues sont décrites à l'aide de schémas externes, appelés également sous-schémas. Chaque schéma traduit un type d'utilisation des données.

⇒  
accroître le degré d'indépendance entre  
les données et les traitements.

**Indépendance physique :**

L'utilisation des données est indépendante de l'organisation physique (support de stockage et méthodes d'accès).

**Indépendance logique :**

L'utilisation des données est indépendante de la structure logique globale.

**Indépendance par rapport aux stratégies d'accès**

La stratégie d'accès doit être définie, autant que possible, par le SGBD et non par l'application.

13/02/2010

Carina Roels - Marie Labidoire

**Indépendance physique :**

La modification de l'organisation physique des données ne doit pas entraîner des modifications dans les programmes accédant à ces données.

**- Indépendance logique :**

Une modification du schéma conceptuel ne doit pas entraîner la modification des programmes; une modification de certains schémas externes est utile dans ce cas.

**Indépendance par rapport aux stratégies d'accès :**

Intervient au moment de l'utilisation des données.

Un programme ne doit pas préciser comment accéder à une donnée, mais quelle donnée il souhaite manipuler.

Le SGBD doit décider du meilleur chemin d'accès aux données.

## Fonctions du SGBD / niveau

### Niveau conceptuel / logique

- Un langage de DDL et de DML.
- La gestion de la confidentialité.
- Le maintien de l'intégrité des données.

### Niveau externe

- Vues
- Outils de développement , d'aide, de saisie, de définition d'états, interfaces ...

### Niveau interne / physique

- Gestion de la mémoire secondaire (disque, fichiers ).
- Gestion du schéma, des index.
- Le partage de données/gestion des concurrences d'accès.
- La reprise sur pannes (fiabilité).
- La distribution des données/gestion de l'interopérabilité.

13/02/2010

Carina Roels - Marie Labidoire

### **NOTE :**

Le développeur a une vision (compréhension) du niveau conceptuel ou logique, pas du niveau physique.

Les applications sont écrites pour accéder à des objets du niveau logique ; il est inutile de savoir où et comment ces objets sont stockés sur la mémoire secondaire.

Ceci est géré par le SGBD.

## **I. Quelques définitions**

### **I.3 Utilisation d'un SGBD.**

#### **Que doit-on comprendre et savoir utiliser pour travailler avec un SGBD ?**

- La définition du schéma de données
- Les opérations sur les données : recherche, mise à jour, ...
- L'optimisation des performances, par le réglage de l'organisation physique des données ou par des règles d'écriture des accès aux données
- Le partage des données entre plusieurs utilisateurs, grâce au mécanisme de transaction

13/02/2010

Carina Roels - Marie Labidoire



## II. Les différentes structures de bases de données

### Illustration sur un exemple

#### PRODUIT

CodP	NomP	Couleur	Poids	Ville
P1	Tenaille	Rouge	12	Brest
P2	Marteau	Vert	17	Paris
P3	Tournevis	Bleu	17	Lille
P4	Tournevis	Rouge	14	Paris

#### FOURNISSEUR

CodeF	Nom	Ville
F1	Senard	Brest
F2	Jardin	Paris
F3	Bourdin	Paris

13/02/2010

Carina Roels - Marie Labidoire

#### PRIX\_ACHAT

CodeF	CodP	Prix
F1	P1	5,50
F1	P2	4
F1	P3	6
F2	P1	5
F2	P2	6
F3	P2	4

### REMARQUE :

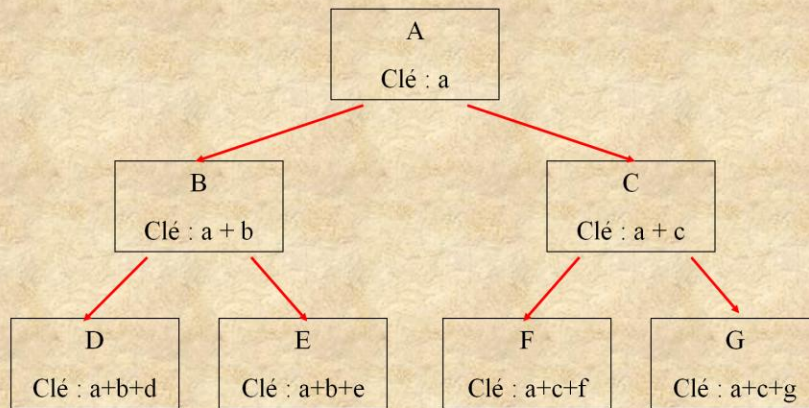
Si nous voulions un exemple complet et entièrement satisfaisant en termes de conception, nous devrions introduire des informations complémentaires.

Toutefois, l'exemple présenté ci-dessus a été fait de façon volontairement simpliste.

Ceci dans un but d'illustration claire et simple.

## II. Les différentes structures de bases de données.

### II.1 L'approche hiérarchique



13/02/2010

Carina Roels - Marie Labidoire

### Les concepts de base :

**Structures d'arbre :**

Racine, nœud, feuille

**Parcours :**

De haut en bas, de gauche à droite

**Contrainte :**

Uniquement contrainte d'appartenance (1:n)

**Le rôle du segment :**

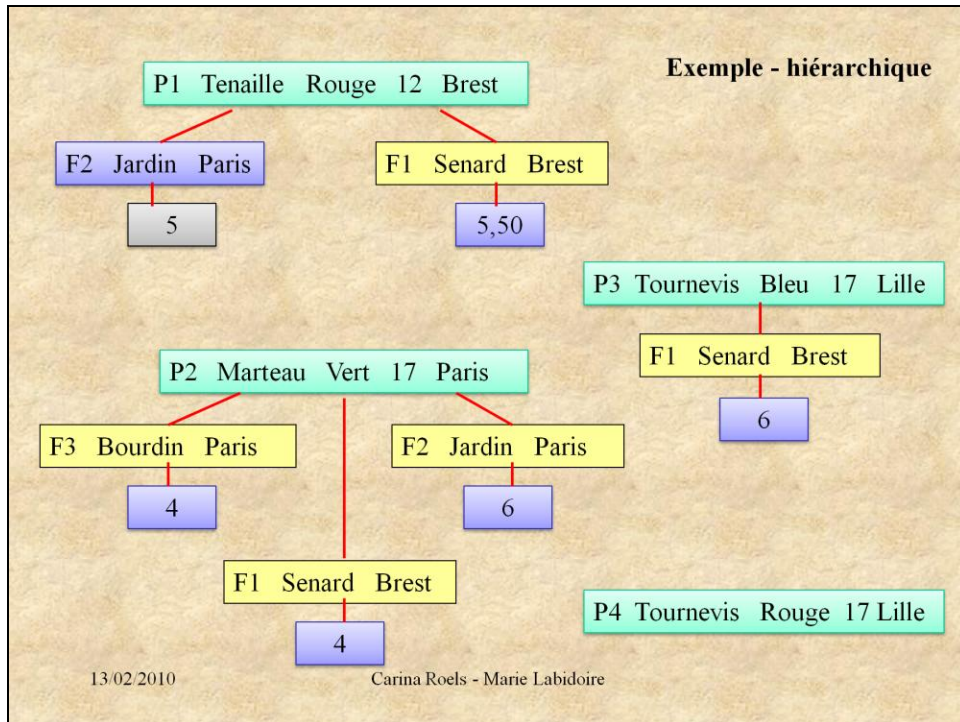
Père, fils, père et fils

**La clé d'un segment :**

concaténation des clés des segments qui se trouvent sur le chemin depuis la racine.

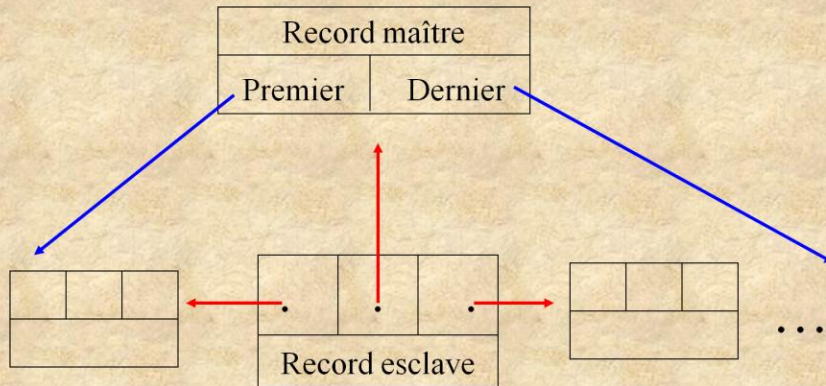
### Conséquences sur les structures :

- Redondance
- La suppression d'un nœud entraîne la suppression en cascade de tous les nœuds fils.
- Impossibilité d'ajouter un nœud fils avant de créer un père.
- Difficulté pour remonter dans la structure.



## II. Les différentes structures de bases de données.

### II.2 L'approche réseau (CODASYL)



13/02/2010

Carina Roels - Marie Labidoire

### Les concepts de base

**Structures :** Record maître, record esclave

**Parcours :** Toutes directions

**Le rôle d'un record :** entité de données (ensemble d'attributs regroupés en une entité logique, constituant l'unité d'échange entre la BD et les applications). Le record est maître, esclave ou les 2.

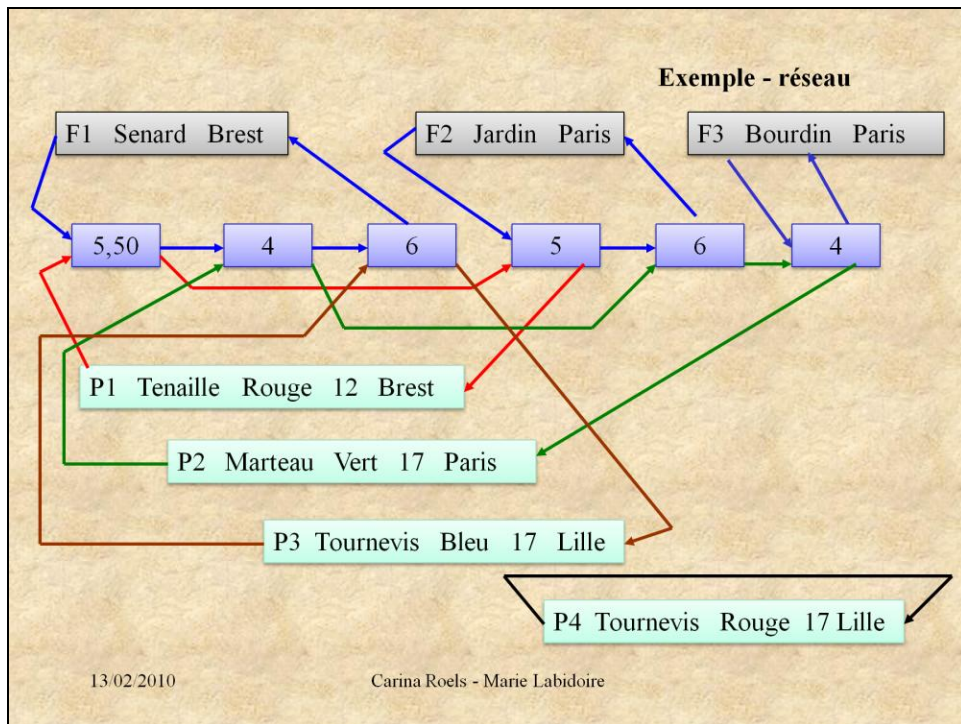
**Le rôle du lien :** ensemble de pointeurs qui permet de naviguer dans tous les sens. Selon que le record est maître ou esclave, il est doté :

- pour le rôle de maître : de 2 pointeurs (1er et dernier esclave)
- pour le rôle d'esclave : de 3 pointeurs (le maître, le précédent, le suivant).

### **Conséquences sur les structures :**

- un record possède plusieurs pointeurs (en fonction des rôles qu'il joue)
- la navigation au travers des pointeurs est parfois longue
- les pointeurs ont un coût de stockage et de gestion





## II. Les différentes structures de bases de données.

### II.3 L'approche relationnelle

<u>Col1</u>	Col2	Col3	...

Colonne, attribut  
prenant éventuellement ses valeurs  
dans un domaine.

Ligne, tuple, row, n-  
uplet, enregistrement

La clé primaire de la table est soulignée.

La clé primaire peut être simple (1 colonne) ou composée (plusieurs colonnes).

13/02/2010

Carina Roels - Marie Labidoire

### Les concepts de base :

**Domaine :** ensemble de valeurs caractérisé par un nom.

**Table / relation :** tableau à 2 dimensions : lignes et colonnes

**Clé candidate :** attribut ou ensemble d'attributs dont la connaissance des valeurs permet d'identifier de façon unique chaque ligne de la table.

**Clé primaire :** Il y a une clé primaire par table, choisie parmi les clés candidates.

**Clé étrangère :** attribut ou ensemble d'attributs d'une table qui correspond à une clé primaire d'une autre table. Une table peut contenir plusieurs clés étrangères.

**Valeur nulle :** valeur conventionnelle qui doit représenter une information inconnue ou non existante.

## II. Les différentes structures de bases de données.

### II.3 L'approche relationnelle (suite)

#### **Contrainte d'intégrité :**

Prédicat qui doit vérifier un sous ensemble de la base de données afin que l'on puisse considérer la base de données comme cohérente.

- **Contrainte de domaine**
- **Contrainte déclarative**
- **Contrainte référentielle**
- **Contrainte d'entité**

13/02/2010

Carina Roels - Marie Labidoire

**Contrainte de domaine**  
donnée,

contrôle syntaxique et sémantique d'une en faisant référence au type de définition du domaine.

**Contrainte déclarative**

contrainte imposée sur des attributs (valeur null, valeur par défaut, clé primaire, validité des valeurs ... )

**Contrainte référentielle**

la valeur d'un attribut d'une table existe comme valeur de clé primaire dans une autre table (clé étrangère -> clé primaire).

**Contrainte d'entité**

Toute table possède une clé primaire.  
Toute colonne participant à la clé primaire doit être non nulle.

Exemple - relationnel

**PRODUIT**

<u>CodP</u>	NomP	Couleur	Poids	Ville
P1	Tenaille	Rouge	12	Brest
P2	Marteau	Vert	17	Paris
P3	Tournevis	Bleu	17	Lille
P4	Tournevis	Rouge	14	Paris

**FOURNISSEUR**

<u>CodeF</u>	Nom	Ville
F1	Senard	Brest
F2	Jardin	Paris
F3	Bourdin	Paris

**PRIX\_ACHAT**

<u>CodeF</u>	<u>CodP</u>	Prix
F1	P1	5,50
F1	P2	4
F1	P3	6
F2	P1	5
F2	P2	6
F3	P2	4

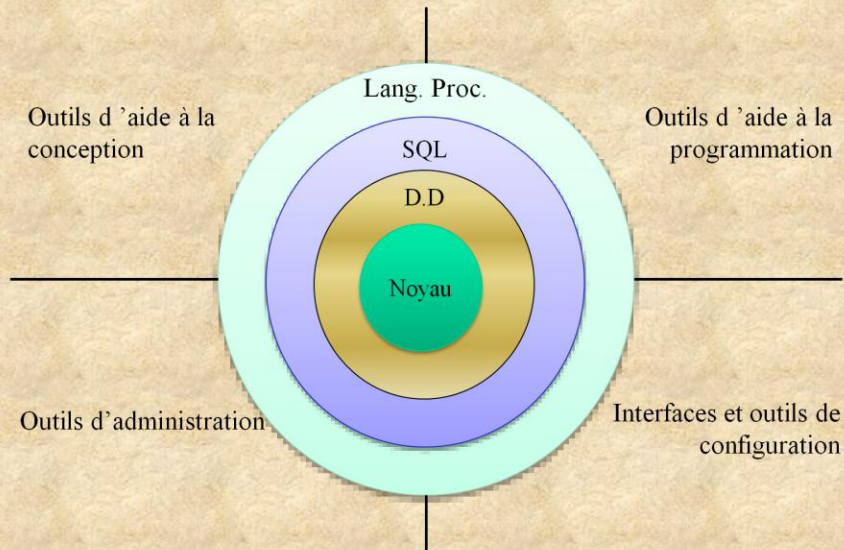
13/02/2010

Carina Roels - Marie Labidoire



### III. Les SGBDR

#### III.1. L'architecture fonctionnelle



13/02/2010

Carina Roels - Marie Labidoire

#### **Un SGBDR :**

multi bases, multi-utilisateurs et multisessions

Structure d'administration du SGBD basée sur un dictionnaire de données (D.D.) intégré, organisé lui-même comme une base de données.

Utilisation du SGBD à travers différents outils. Exemple, pour Oracle :

**Outils d'aide à la conception :** Oracle Designor

**Outils d'aide à la programmation :**

- SQL\*Plus (interface conversationnelle)
- Pro\*C, Pro\*Java,... (interfaces de programmation)
- Sql\*Report (générateur de rapports)
- Oracle developer (générateur d'application)

**Outils d'administration :**

- Oracle Entreprise Manager
- les utilitaires EXP/IMP (export/import)
- SQL\*Load (utilitaire de chargement de données)

**Interfaces et outils de configuration :**

- SQL\*Net (configuration client/serveur)

Etc.

### III. Les SGBDR

#### III.2. L'architecture physique - B.D. et Instance

- Une base de données Oracle se compose de :
  - Fichiers : données, redo-log, contrôle, ...
  - Exécutables pour faire fonctionner la B.D.
  - Mémoire (une zone partagée : SGA, une zone privée pour chaque utilisateur connecté).
- On parle de *base de données* pour qualifier l'ensemble des fichiers qui la composent.
- Une *instance* regroupe la base de données, la zone de mémoire allouée et les exécutables assurant le fonctionnement de la base.

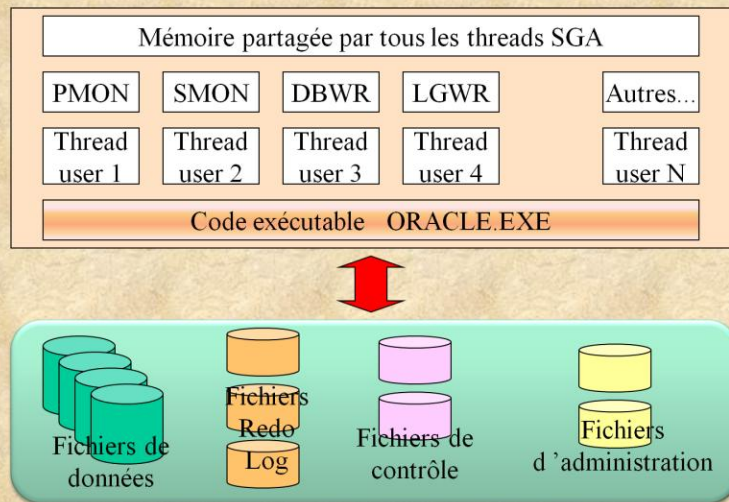
*Une instance est une base de données « en action »*

13/02/2010

Carina Roels - Marie Labidoire

### III. Les SGBDR

#### III.2. L'architecture physique - B.D. et Instance



13/02/2010

Carina Roels - Marie Labidoire

Le serveur Oracle fonctionne sous Windows comme un exécutable unique ayant de multiples threads.

Les threads internes au fonctionnement d'Oracle sont les équivalents des processus sous Unix.

L'exécutable fonctionne sous la forme d'un service dans les environnements Microsoft.

#### Les différents threads d'Oracle :

- **PMON** : process Monitor
  - **SMON** : System Monitor
  - **DBWR** : DataBase Writer
  - **LGWR** : Log Writer
  - **ARCH** : Archiver
  - **CKPT** : Checkpoint dédié
- + d'autres threads optionnels

## IV. Quelques concepts importants des SGBDR

### IV.1 Conception et normalisation

- Conception à l'aide d'une méthode ou d'un formalisme (Dépendances fonctionnelles, MERISE, UML, ...)
- Vérification à l'aide des règles de normalisation (Formes Normales)

**1FN :** \_\_atomicité des informations  
dépendance des informations de la clé primaire

**2FN :** dépendance TOTALE des informations de la clé primaire

**3FN :** dépendance UNIQUE des informations de la clé primaire

13/02/2010

Carina Roels - Marie Labidoire

**La normalisation** est un processus réversible, exécuté par étape, et qui remplace une table par plusieurs tables qui répondent à certaines règles.

Les tables ont, d'une étape à une autre, une structure plus simple et homogène.

#### **1FN :**

Une table est en première forme normale à condition que toutes les informations soient atomiques (non décomposables), et qu'elles dépendent de la clé primaire (dépendance fonctionnelle entre la clé primaire et les autres informations).

#### **2FN :**

Une table est en deuxième forme normale si elle est en première forme normale Et si tous les attributs sont **TOTALEMENT** dépendants de la clé primaire.

**Autrement dit :** ceci concerne uniquement les tables avec une clé primaire composée! Il faut enlever de la table, les informations qui ne dépendent que d'une partie de la clé primaire.

#### **3FN :**

Une table est en troisième forme normale si elle est en deuxième forme normale ET si tous les attributs dépendent **UNIQUEMENT** de la clé primaire.

**Autrement dit :** Il faut enlever de la table, les informations qui dépendent d'une donnée qui n'est pas la clé primaire.



## IV. Quelques concepts importants des SGBDR

### IV.2 Normalisation vs. Optimisation

#### Les règles complémentaires de la normalisation

- la règle de BOYCE/CODD (FNBC)
- la quatrième et cinquième forme normale (4FN - 5FN)

La normalisation :	Limiter la redondance. Manipulation aisée des données ( ajouts, retraits, modifications de données sans créer d'anomalies ).
--------------------	---

La dé-normalisation :	Optimisation des traitements. Redondance contrôlée.
-----------------------	--

13/02/2010

Carina Roels - Marie Labidoire

L'introduction d'un certain niveau de redondance est une technique d'optimisation des requêtes les plus souvent utilisées. Cette redondance est dite ' contrôlée ' ou ' calculée ', car elle tient compte des besoins des modules de traitement et leur exigence en temps de réponse.

La redondance calculée peut être réalisée de deux manières :

#### - le stockage de données déductibles :

mémoriser les résultats des requêtes les plus fréquentes

mémoriser les résultats de calculs complexes

*par exemple : stocker le montant total dans une table COMMANDE.*

#### - la dénormalisation :

passer une table qui est en 3FN à la 2FN ou même à la 1FN.

Le but étant de réduire le nombre de jointures qui peuvent être coûteuses en performance.

*Par exemple : intégrer le taux de TVA dans une table PRODUIT.*

## **IV. Quelques concepts importants des SGBDR**

### **IV.3 Le langage d'accès aux données**

Le langage SQL (Structured Query Language)

Langage non procédural

Contient :

- LDD (langage de définition de données)
- LMD (langage de manipulation de données)
- LCD (langage de contrôle des données)

13/02/2010

Carina Roels - Marie Labidoire

## IV. Quelques concepts importants des SGBDR

### IV.4 Les index

- Objet de la B.D. regroupant :
  - les différents valeurs d'une colonne (ou de plusieurs colonnes combinées)
  - les RID des lignes correspondant aux valeurs
- Utilisés et mis à jour automatiquement par le serveur Oracle
- Créés explicitement ou automatiquement
- Structure par défaut en B-tree (d'autres structures possibles)

#### Avantages :

- Accélère les recherches
- Unicité de ligne ( si index unique )

#### Inconvénients :

- Ralentit les mises à jour

13/02/2010

Carina Roels - Marie Labidoire

Le serveur Oracle crée automatiquement un index unique sur :

- la clé primaire de la table
- sur une colonne avec une contrainte d'unicité

#### Création d'un index :

*Create index i\_prod\_code on produit (codp);*

#### Suppression d'un index :

*Drop index i\_prod\_code;*

Vues du dictionnaire qui permettent de connaître les index et les colonnes qui les composent :

*user\_indexes*

*user\_ind\_columns*

#### Démarche d'indexation 'standard' :

Table de la B.D. « brute »

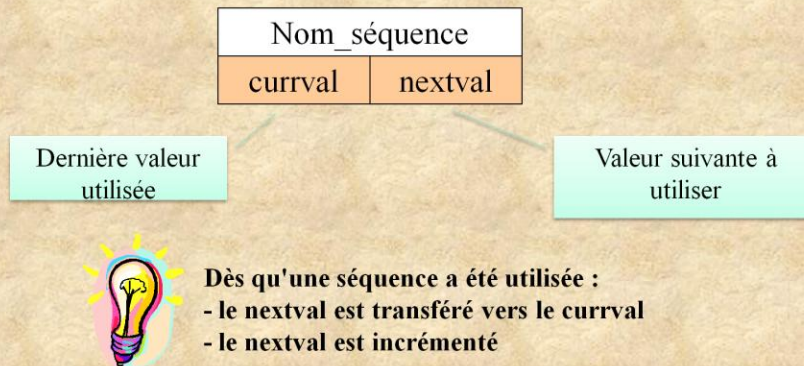
- ❶ Réflexion sur les accès par clé primaire
- ❷ Etude des requêtes fréquentes (simples)
- ❸ Etude des requêtes fréquentes '(jointures)

+ réflexions sur d'autres formes d'organisation des données

## IV. Quelques concepts importants des SGBDR

### IV. 5 Les séquences

Séquence : utilisé afin d'incrémenter la valeur d'une colonne (ex. clé primaire)



13/02/2010

Carina Roels - Marie Labidoire

#### Création d'une séquence :

```
create sequence          s_produit  
start with 1  
increment by 1  
maxvalue 9999  
nocycle;
```

#### Utilisation d'une séquence dans un insert :

```
insert into produit (codep, nomp) values( s_produit.nextval, ' nouveau ');
```

#### Vues du dictionnaire qui permettent de connaître les séquences :

```
user_sequences
```

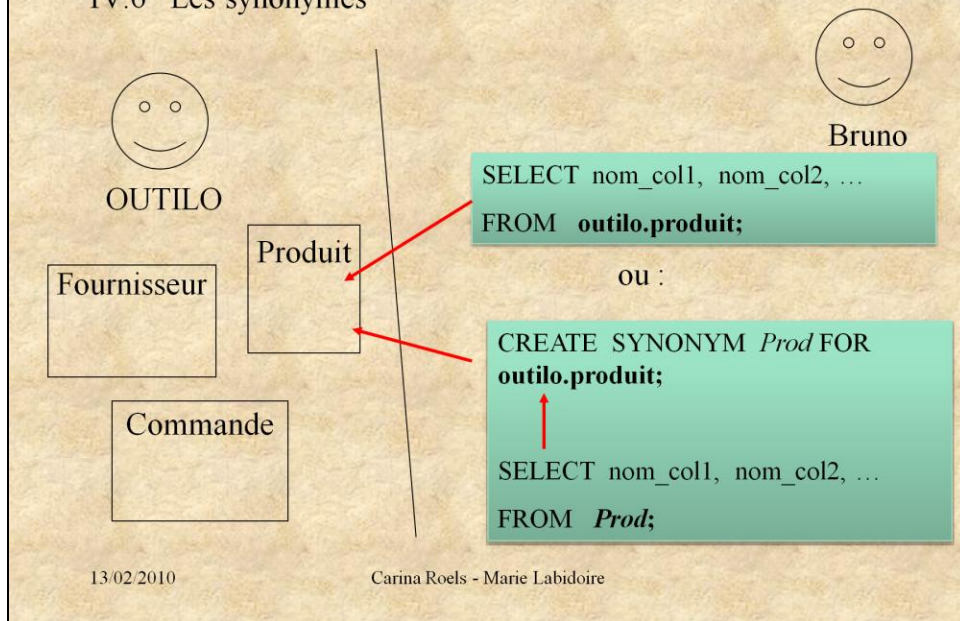
Il peut y avoir des trous dans la numérotation, dû à un rollback ou à un arrêt du système.

Une séquence n'est pas liée à une table. Elle peut donc servir pour des tables différentes, ce qui peut générer également des interruptions dans la numérotation.



## IV. Quelques concepts importants des SGBDR

### IV.6 Les synonymes



#### Création d'un synonyme :

*Create [public] synonym prod for outilo.produit;*

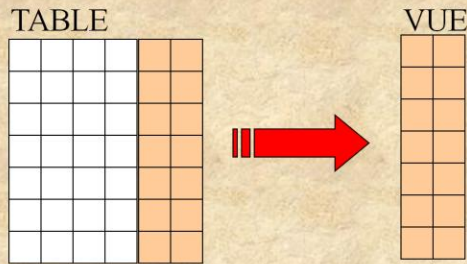
Si le mot `public` est utilisé, le synonyme est accessible à tous les utilisateurs.

#### Suppression d'un synonyme :

*Drop [public] synonym prod;*

## IV. Quelques concepts importants des SGBDR

### IV.7 Les vues



**UNE VUE =  
UNE TABLE VIRTUELLE EXTRAITE  
d'UNE OU PLUSIEURS TABLES EXISTANTES.**

**Il est possible d'effectuer des sélections à travers  
une vue, comme s'il s'agissait d'une table.**

#### Utilité :

- l'indépendance logique des données (schéma externes)
- la réduction de la visibilité de certaines données (limiter les colonnes et/ou les lignes visibles)

13/02/2010

Carina Roels - Marie Labidoire

#### Création d'une vue:

```
Create view v_cdes_F1 as  
select * from comande where fournisseur = ' F1 ';
```

#### Suppression d'une vue :

```
Drop view v_cdes_F1;
```

#### Types de vues :

- réalisant une réduction de visibilité sur certaines colonnes
- réalisant une réduction de visibilité sur certaines lignes
- réalisant une réduction de visibilité sur certaines colonnes et lignes
- statiques (contenant une condition endur)
- dynamiques (contenant une condition qui s'appuie sur le compte de l'utilisateur connecté).

## IV. Quelques concepts importants des SGBDR

### IV.8 Les triggers

Objet de la B.D. contenant :

un ensemble de requêtes SQL, incluses ou non dans un langage procédural, **déclenché** suite à un **événement** qui s'est produit dans la base de données.

Evénement :

- une opération sur une donnée (une mise à jour ,...)
- le passage à 'VRAI' d'une condition sur la valeur de données

Utilité :

- Renforcer des autorisations complexes.
- Renforcer les contrôles d'intégrité lors de certaines mis à jours.  
( contrôle de l'intégrité sélectif par rapport à l'action menée)
- Génération automatique de certaines valeurs de colonnes,  
lors de mis à jours.
- Maintenir la réplication des tables.

13/02/2010

Carina Roels - Marie Labidoire

Un trigger se déclenche sur des actions modifiant les données de la table sur laquelle le trigger est défini. Le code s'exécute quelque soit l'utilisateur ou l'application ayant déclenché le trigger.

Un trigger peut être déclenché AVANT ou APRES l'action.

Exemple de trigger (en PL/SQL - ORACLE) :

Supposons qu'il existe dans notre base de données, une table  
GRILLE\_SALAIRE( Fonction, Salmin, Salmax).

Le trigger suivant sera déclenché : soit lors de la création d'un employé  
soit lors de la mise à jour des colonnes salaire et/ou fonction d'un employé.  
Lorsqu'il s'agit d'une création ou d'une mise à jour en masse (plusieurs lignes affectées), le trigger sera exécuté pour chaque ligne affectée.

```
CREATE TRIGGER verif_salaire
BEFORE INSERT or UPDATE OF salaire, fonction ON employe
FOR EACH ROW WHEN ( new.fonction != 'PRESIDENT' )
DECLARE
    psalmin    NUMBER;
    psalmax    NUMBER;
BEGIN
    /* Vérification que le salaire de l'employé soit comprise */
    /* entre les salaires minimum et maximum pour la nouvelle fonction */
    /* Ici : le code correspondant ..... */
END;
```

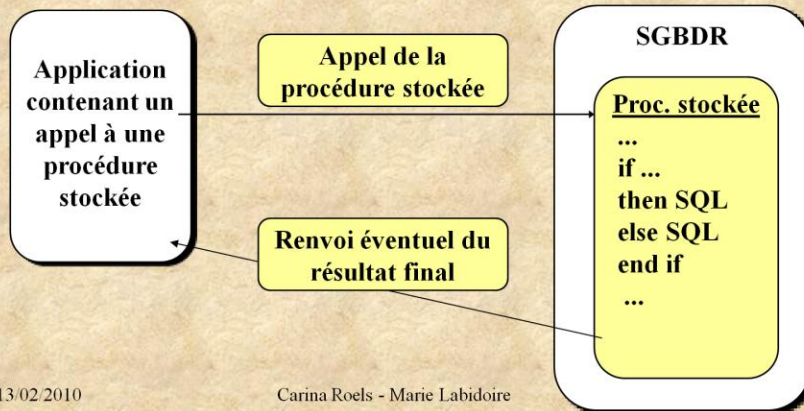
/

## IV. Quelques concepts importants des SGBDR

### IV.9 Les procédures / fonctions stockées

Objet de la B.D. contenant :

Une fonction ou procédure écrite en langage procédural et en SQL.



13/02/2010

Carina Roels - Marie Labidoire

Une procédure ou fonction stockée peut être « compilée » : les résultats des phases 'parsing' et 'détermination du chemin d'accès' sont stockés avec la procédure. Ceci améliore sensiblement les performances lors de l'exécution.

#### Exemple de création d'une procédure stockée :

```
CREATE PROCEDURE rapport_emp_proj
    ( cpro IN CHAR, noemp IN NUMBER, nbheures IN NUMBER )
AS
BEGIN
    INSERT INTO assignments(codproj, no_emp, heures, datrap)
        values( cpro, noemp, nbheures, sysdate);
END rapport_emp_proj;
/
```

#### Exemple de création d'une fonction stockée :

```
CREATE FUNCTION calcul_coutproj( cpro IN CHAR )
RETURN NUMBER AS
    cout      NUMBER;
BEGIN
    SELECT    SUM( cout )    INTO cout
        FROM  assignments WHERE codproj = cpro
    RETURN( cout );
END calcul_coutproj;
/
```