

Les S.G.B.D.

**Conception et compréhension
de bases de données relationnelles**

Carina Roels

Contenu

1. Le modèle physique des données

- 1.1 La clé primaire
- 1.2 Les clés étrangères
- 1.3 Les tables de liaison
- 1.4 Les clés étrangères réflexives
- 1.5 Les tables en héritage
- 1.6 La clés primaire relative

2. Les tendances en entreprise

- 2.1 MERISE : le MCD
- 2.2 MERISE2 : ajouts au MCD
- 2.3 UML : diagramme de classes

3. La normalisation.

- 3.1 Qu'est-ce la normalisation ?
- 3.2 La première Forme Normale (1FN)
- 3.3 La deuxième Forme Normale (2FN)
- 3.4 La troisième Forme Normale (3FN)
- 3.5 Les règles complémentaires
- 3.6 Pourquoi normaliser ou dé-normaliser ?

4. La création de la B.D. (Oracle)

- 4.1 Table sans clé étrangère
- 4.2 Table avec clés étrangères
- 4.3 Table de liaison
- 4.4 Table avec clé étrangère réflexive
- 4.5 Tables en héritage
- 4.6 Table avec clé primaire relative

1. Le modèle physique de données

La conception d'une B.D. est réalisée à l'aide d'une méthode (ex. MERISE).

Le **modèle conceptuel de données (MCD)**

défini à l'aide de la méthode doit être traduit d'après certaines règles afin d'obtenir un **modèle physique de données (MPD)**.

Le MPD obtenu sert ensuite de base à l'écriture des instructions SQL pour créer la **Base de Données**.

1. Le modèle physique de données

1.1 La notion de clé primaire

Toute table doit avoir une clé primaire qui :

- garantit l'unicité de chaque enregistrement de la table
- est obligatoirement renseignée (NOT NULL)

CLIENT
<u>Nocli</u>
Nom
Prénom
Adr
Cpost
Ville
Taux_remise

Il est préférable de choisir des clés primaire du genre 'numéro' ou 'code' au lieu de colonnes de type 'texte' ou 'date'.

Dans l'exemple :

La colonne NOCLI étant unique pour chaque client, elle est définie comme clé primaire.

Exemple de contenu :

CLIENT

Nocli	Nom	Prénom	...
1	Dutch	Frédéric	...
2	Darel	Marie	...
3	Dutch	Paul	...

REMARQUE :

Une table ne peut avoir qu'une seule clé primaire.

La clé primaire peut éventuellement être composée de plusieurs colonnes.

1. Le modèle physique de données

1.2 La notion de clé étrangère

Si pour un enregistrement d'une table A, il n'existe qu'un seul enregistrement de la table B :

La table B doit comporter une clé étrangère vers la table A.



Dans l'exemple :

Un client peut avoir plusieurs commandes, mais une commande n'appartient qu'à un seul client : la table COMMANDE doit comporter une clé étrangère vers la table CLIENT.

Exemple de contenu :

CLIENT

<u>Nocli</u>	Nom	Prénom	...
1	Dutch	Frédéric	...
2	Darel	Marie	...
3	Dutch	Paul	...

COMMANDE

<u>Nocde</u>	Date_cde	<u>Nocli</u>
120	02/05/2005	1
121	15/05/2005	3

REMARQUE :

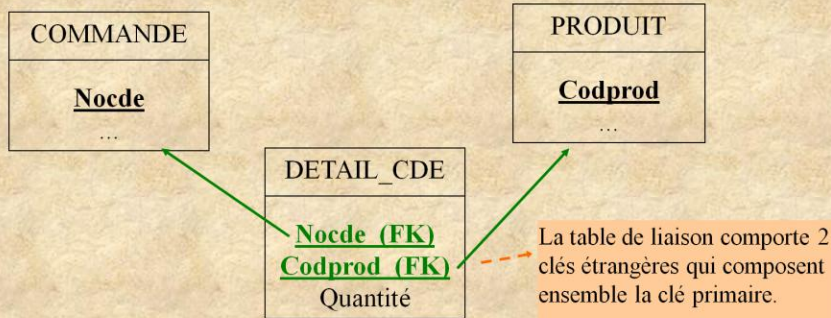
Une table peut avoir plusieurs clés étrangères vers des tables différentes.

Une clé étrangère doit avoir le même format que la clé primaire référencée.

1. Le modèle physique des données

1.3 La notion de table de liaison.

Si pour un enregistrement d'une table A, il peut exister plusieurs enregistrements de la table B ET
si pour un enregistrement de la table B, il peut exister plusieurs enregistrements de la table A :
Il faut créer une **table de liaison**.



Dans l'exemple :

Une commande peut contenir plusieurs produits.

Un produit peut concerner plusieurs commandes.

Il faut donc une table de liaison afin de mémoriser les produits concernés par chaque commande. La quantité commandée dépend du produit Et de la commande; il faut donc placer cette donnée dans la table de liaison.

Exemple de contenu :

COMMANDE		PRODUIT	DETAIL CDE		
Nocde	...	Codprod	Nocde	Codprod	Quantité
120	...	A2	120	A2	10
121	...	B5	120	B5	5
			121	A2	6

REMARQUE :

La table de liaison comporte 2 clés étrangères : une vers chaque table référencée. Afin de définir la clé primaire de cette table, 2 solutions existent :

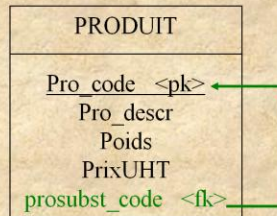
- définir les 2 clés étrangères ensemble comme clé primaire,
- définir une nouvelle colonne (ex. No_detail) en tant que clé primaire.

Etant donné qu'un produit donné ne risque pas d'être présent plusieurs fois sur la même commande, la première solution convient.

1. Le modèle physique des données

1.4 La notion de clé étrangère réflexive

Lorsqu'un enregistrement d'une table doit référencer un autre enregistrement de la même table :
il faut une **clé étrangère réflexive**.



Dans l'exemple :

Un produit peut être substitué (remplacé) par un autre produit.
Pour chaque produit, il faut donc connaître son éventuel remplaçant.
Ceci est indiqué par une clé étrangère vers un autre enregistrement de la même table.

Exemple de contenu :

Pro code	Pro descr	Poids	PrixUHT	prosubst code
M1	Marteau bois	30	11,50	M2
M2	Marteau bois	40	13	M1

Ici : le marteau bois M1 peut être remplacé par le M2 et vice versa.

REMARQUE :

Il n'est pas possible d'avoir 2 colonnes de même nom dans une table.
Il a donc fallu donner un nom différent à la colonne qui sera définie comme clé étrangère.

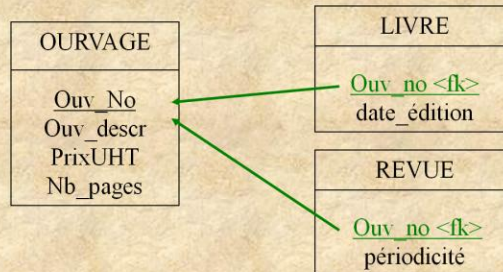
1. Le modèle physique des données

1.5 La notion de tables en héritage

Si une table comporte plusieurs groupes de données ayant :

- des données communes
- des données spécifiques

il est utile de créer des **tables en héritage** (1 par groupe)



Dans l'exemple :

Parmi les ouvrages édités, nous distinguons les revues et les livres.
Bien qu'il existe des informations valables pour tous les ouvrages, chacun des 2 types d'ouvrage possède également des informations propres.

Dans ce cas, il est utile d'avoir 3 tables distinctes :

- une table maîtresse comportant les informations communes (OUVRAGE)
- 2 tables en héritage comportant les informations spécifiques (LIVRE et REVUE)

Exemple de contenu :

OUVRAGE

Ouvr No	Ouvr descr	PrixUHT	Nb Pages
1	L'Echo	4,5	20
2	Le chemin	8,90	86

LIVRE

Ouvr no	Date édition
2	25/03/2002

REVUE

Ouvr no	Périodicité
1	semaine

REMARQUE :

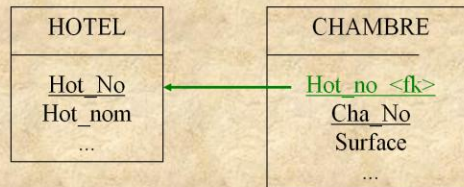
Les tables en héritage comportent :

- une clé primaire qui est en même temps clé étrangère vers la table maîtresse
- des informations spécifiques

1. Le modèle physique des données

1.6 La notion de clé primaire relative.

Si l'unicité d'une clé primaire ne peut être garantie QUE si l'on y ajoute une clé étrangère, il y a une **clé primaire relative**.



Dans l'exemple :

Un hôtel est composé de chambres. Une chambre donnée n'existe que dans un seul hôtel.

Le numéro de chambre n'étant pas unique (il peut exister une chambre N° 1 dans différents hôtels), il faut combiner les colonnes HOT_NO et CHA_NO pour former la clé primaire.

Exemple de contenu :

<u>HOTEL</u>		<u>CHAMBRE</u>		
<u>Hot_No</u>	<u>Hot_nom</u>	<u>Hot_no</u>	<u>Cha_no</u>	<u>surface</u>
1	Ibis	1	1	12
2	F1	1	2	15
		2	1	20
		...		

REMARQUE :

La clé primaire relative comporte :

- une clé étrangère vers une autre table
- une donnée supplémentaire

2. Les tendances en entreprise

Utilisation d'ateliers de génie logiciel (AGL)
basés sur une méthode.

Les outils CASE propriétaires

Exemple ORACLE*Case :

- Case*dictionary
- Case*designer
- Case*generator

méthode 'maison'

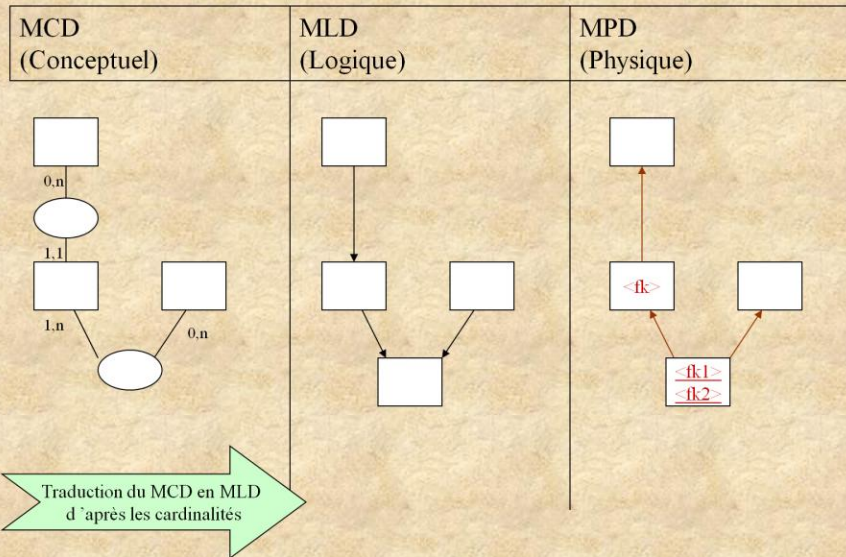
Les outils CASE tiers

- PowerDesignor
- Win*Design
- Rational Rose
- ...

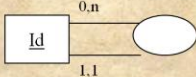
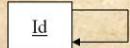

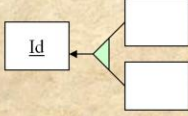
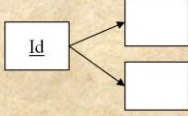
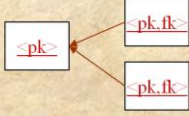



MERISE / MERISE2 / UML

UML

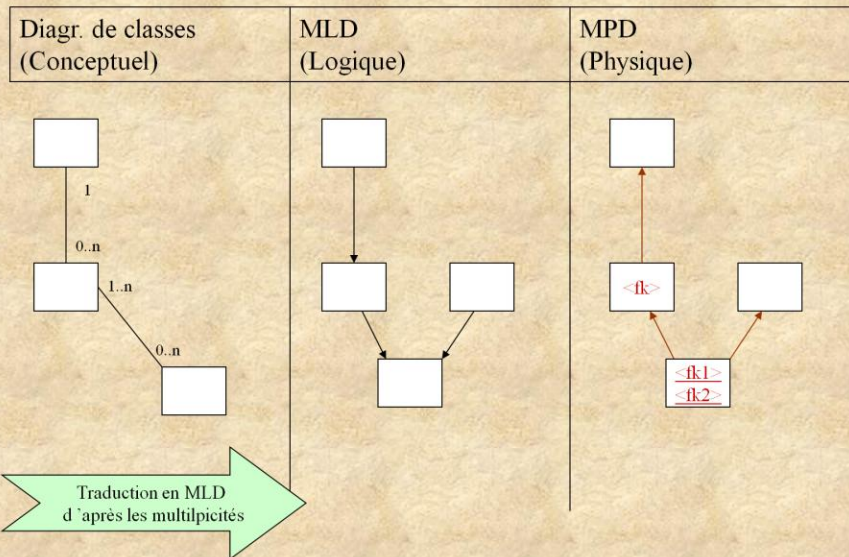
2.1 MERISE : Le modèle conceptuel de données



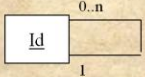
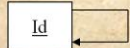

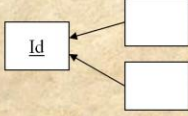
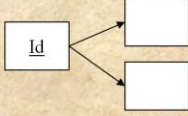
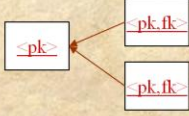
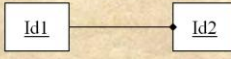


2.2 MERISE2 : ajout des cas particuliers au niveau conceptuel

MCD (Conceptuel)	MLD (Logique)	MPD (Physique)
<p>La dépendance réflexive</p> 		
<p>L'héritage</p> 		
<p>L'identifiant relatif</p> 		

2.3 UML : Le diagramme de classes



2.3 UML : les cas particuliers

Diagr. de classes (Conceptuel)	MLD (Logique)	MPD (Physique)
<p>La dépendance réflexive</p> 		
<p>L'héritage</p> 		
<p>L'identifiant relatif (agrégation, composition)</p> 		

3. La conception d'une BD et la normalisation.

3.1 Qu'est-ce la normalisation ?

La normalisation est un concept inventé pour la conception de bases de données relationnelles.

La normalisation est un processus réversible exécuté par étape, afin de remplacer une table par plusieurs tables qui répondent à certaines règles.

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

3. La conception d'une BD et la normalisation.

3.2 La première forme normale (1FN)

Une table est en première forme normale à condition,
et UNIQUEMENT à condition :

- **que toutes les informations soient atomiques,**
- **qu'elles dépendent de la clé primaire**
(Dépendance fonctionnelle entre l'identifiant et les autres informations).

Autrement dit :

Chaque information doit être élémentaire (non décomposable).
Il faut enlever les groupes répétitifs.

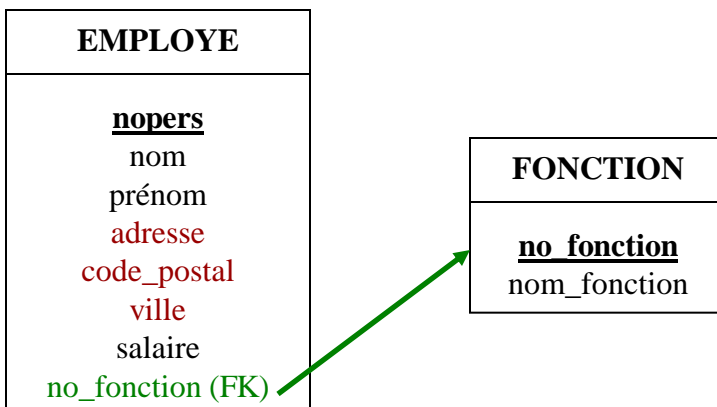
Exemple :

EMPLOYE
<u>nopers</u> nom prénom adresse salaire fonction

La table PERSONNE ne respecte pas la 1FN pour 2 raisons :

- la colonne ADRESSE n'est pas atomique. Elle devrait être décomposée en : ADRESSE, CODE POSTAL et VILLE
- la colonne FONCTION ne dépend pas de la clé primaire no_pers. Elle a une existence propre. Les valeurs de cette colonne sont prédéfinies et devraient être mémorisées dans une table à part.

Solution :



3. La conception d'une BD et la normalisation.

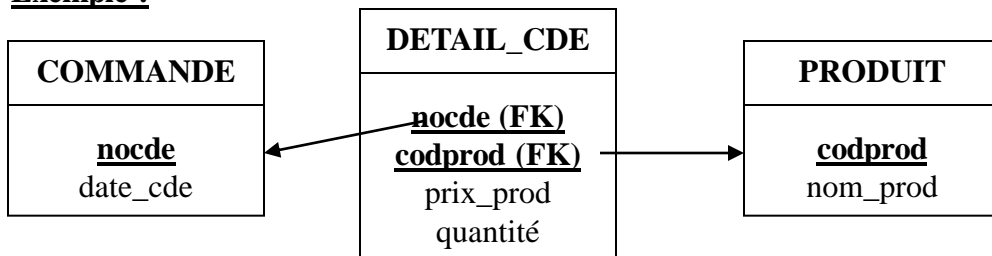
3.3 La deuxième forme normale (2FN)

Une table est en deuxième forme normale (2FN)
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 les données qui ne dépendent QUE d'une partie de la clé primaire.

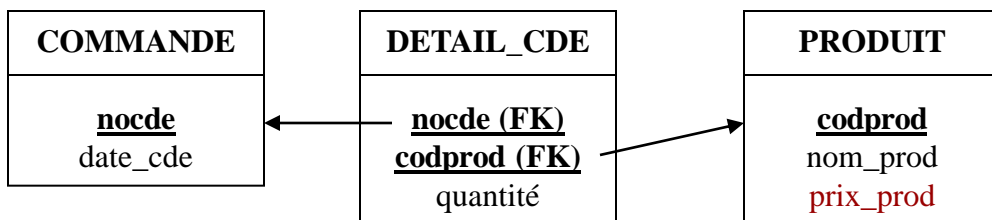
Exemple :



La table DETAIL_CDE ne respecte pas la 2FN :

La colonne PRIX_PROD ne dépend pas du NOCDE et du CODPROD. Cela signifierait que le prix d'un produit dépend de la commande (ce qui est illégal)!

Solution :



3. La conception d'une BD et la normalisation.

3.4 La troisième forme normale (3FN)

Une table est en troisième forme normale (3FN)
si elle est en deuxième forme normale ET
si tous les attributs sont UNIQUEMENT dépendants de la clé primaire.

Autrement dit :

Il faut enlever les données qui dépendent d'une donnée qui n'est pas la
clé primaire de la table.

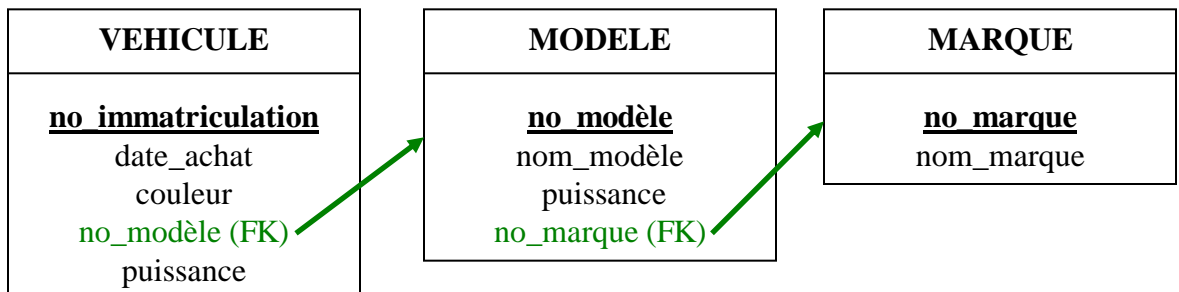
Exemple :

VEHICULE
<u>no immatriculation</u> date_achat couleur marque modèle puissance

La table VEHICULE ne respecte pas la 3FN :
La colonne PUISSANCE ne dépend pas du
NO_IMMATRICULATION, mais plutôt du
modèle.

Par ailleurs, la 2FN n'est pas respectée non
plus :
le MODELE et la MARQUE ont une existence
propre et ne dépendent donc pas du
NO_IMMATRICULATION.

Solution :



3. La conception d'une BD et la normalisation.

3.5 Les règles complémentaires

Les règles complémentaires

la règle de BOYCE/CODD

la quatrième et cinquième forme normale (4FN - 5FN)

Dans la recherche et dans la littérature concernant les bases de données, on parle de ces étapes complémentaires. Ce sont toutefois des concepts qui ont très peu de retombées pratiques.

En général, normaliser jusqu'au 3FN est plus que suffisant.

D'autant plus que dans certains cas, on sera contraint de dé-normaliser ... , pour des raisons d'optimisation.

3. La conception d'une BD et la normalisation.

3.6 Pourquoi normaliser ou dé-normaliser ?

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.

- Le stockage de données déductibles :
 - mémoriser les résultats des requêtes les plus fréquentes
 - mémoriser les résultats de calculs complexes
- passer une table qui est en 3FN à la 2FN ou même à la 1FN.

REMARQUE :

La dé-normalisation est une parmi de nombreuses solutions d 'optimisation.

Cette solution pourrait être envisagée **après une réflexion** concernant :

- les applications et les accès aux données
- les besoins en espace de stockage
- la gestion de l 'espace de stockage par le SGBD
- les autres solutions d 'optimisation

4. La création de la B.D. (Oracle).

4.1 Une table sans clés étrangères.

create table CLIENT

(

CLI_NO NUMBER(6) not null,

CLI_NOM VARCHAR(35) not null,

CLI_PREN VARCHAR(35) not null,

CLI_NRUE NUMBER(3) ,

CLI_RUE VARCHAR(50) ,

CLI_CPOST CHAR(6) ,

CLI_VILLE VARCHAR(50) ,

CLI_TEL CHAR(10) ,

constraint pk_client primary key (CLI_NO)

);

Format de la donnée,
parmi ceux acceptés
par le SGBD utilisé

Not null :

- obligatoire pour la clé primaire
- possible pour les autres colonnes

Nom de la
contrainte

Type de
contrainte

Indication de la colonne qui
constitue la clé primaire

4. La création de la B.D. (Oracle).

4.2 Une table avec clés étrangères

```
create table COMMANDE
(
  CDE_NO          NUMBER(6)      not null,
  CLI_NO          NUMBER(6)      not null,
  TCD_CODE        CHAR(4)        not null,
  CDE_DATCDE      DATE           ,
  CDE_DATREC      DATE           ,
  CDE_COMMENT     VARCHAR(35)    ,
  constraint pk_commande primary key (CDE_NO),
  constraint fk_cde_cli foreign key (CLI_NO)
    references CLIENT (CLI_NO),
  constraint fk_cde_tcde foreign key (TCD_CODE)
    references TYPE_COMMANDE (TCD_CODE)
);
```

Nom de la
contrainte

Table(colonne) vers laquelle cette
clé étrangère fait référence

Type de
contrainte

Colonne de la table
en cours de création
qui sera clé
étrangère

4. La création de la B.D. (Oracle).

4.3 Une table de liaison

```
create table DETAIL_CDE
(
  CDE_NO      NUMBER(6)      not null,
  PRO_CODE    CHAR(5)        not null,
  QUANTITE    INTEGER
  ,
  constraint pk_detail primary key (CDE_NO, PRO_CODE)
  constraint fk_detail_cde foreign key (CDE_NO)
    references COMMANDE (CDE_NO),
  constraint fk_detail_prod foreign key (PRO_CODE)
    references PRODUIT (PRO_CODE),
);
```

Clé primaire
composée de 2
colonnes

Chaque colonne de la clé
primaire est également clé
étrangère vers une table
spécifique.

4. La création de la B.D. (Oracle).

4.4 Une table avec clé étrangère réflexive

```
create table PRODUIT
(
  PRO_CODE          char(5)          NOT NULL,
  PRO_DESCR         varchar(50)      NOT NULL,
  POIDS             number(4,2)       ,
  PRI XUHT          number(6,2)      NOT NULL,
  PROSUBST_CODE     char(5)          ,
  constraint pk_produit primary key (PRO_CODE)
  constraint fk_prod_prod foreign key (PROSUBST_CODE)
    references PRODUIT (PRO_CODE)
);
```

La colonne PROSUBST_CODE
est définie comme clé étrangère

Elle fait référence à la table
PRODUIT dans sa clé primaire
PRO_CODE

4. La création de la B.D. (Oracle).

4.5 Des tables en héritage

```
Create table OUVRAGE
( OUV_NO          NUMBER(3)      NOT NULL,
  OUV_DESCR       VARCHAR(50)    NOT NULL,
  PRIXUHT         NUMBER(4,2)    NOT NULL,
  NB_PAGES        NUMBER(3)      ,
  constraint PK_OUVRAGE primary key (OUV_NO) );
```

```
Create table LIVRE
( OUV_NO          NUMBER(3)      NOT NULL,
  DATE_EDITION    DATE          ,
  constraint PK_OUVRAGE primary key (OUV_NO),
  constraint FK_LIVRE_OUVR foreign key (OUV_NO)
    references OUVRAGE (OUV_NO) );
```

```
Create table REVUE
( OUV_NO          NUMBER(3)      NOT NULL,
  PERIODICITE     varchar(10)    ,
  constraint PK_OUVRAGE primary key (OUV_NO),
  constraint FK_REVUE_OUVR foreign key (OUV_NO)
    references OUVRAGE (OUV_NO) );
```

4. La création de la B.D. (Oracle).

4.6 Une table avec clé primaire relative

```
Create table HOTEL
( HOT_NO      NUMBER(3)      NOT NULL,
  HOT_NOM     VARCHAR(50)    NOT NULL,
  ...
  constraint PK_HOTEL primary key (HOT_NO)
);
```

```
Create table CHAMBRE
( HOT_NO      NUMBER(3)      NOT NULL,
  CHA_NO      NUMBER(3)      NOT NULL,
  ...
  constraint PK_CHAMBRE primary key (HOT_NO, CHA_NO) ,
  constraint FK_CHA_HOTEL foreign key (HOT_NO)
    references HOTEL (HOT_NO)
);
```

4. La création de la B.D. (Oracle).

4.7 L 'ordre des créations



Les tables doivent être créées dans un certain ordre :

- tables sans clés étrangères
- tables avec clés étrangères vers les tables déjà existantes



Astuce :

Créer **TOUTES** les tables, sans indication de clé étrangère
(Cf. instruction de création d 'une table ` simple `).
Puis, ajouter les indications de clé étrangère :

`alter table` **COMMANDE**

```
add constraint fk_commande_cli foreign key (CLI_NO)  
references CLIENT (CLI_NO);
```

Toutes les instructions devraient être écrites dans un fichier texte (Ex. Crebase.sql).

Le fichier sera ensuite exécuté par l 'intermédiaire de l 'utilitaire SQL*Plus d 'Oracle.

4. La création de la B.D. (Oracle).

4.8 Lancer les instructions

1. Lancer l'utilitaire SQL*Plus

1ère connexion avec un compte ' administrateur B.D. ' :

utilisateur : **SYSTEM**
mot de passe : **xxxx**
chaîne hôte : _____

Le mot de passe connu dans la B.D. pour le compte SYSTEM

Ne rien indiquer : utilisation de la B.D. locale

2. Créer un compte sous lequel les tables devront être créées :

```
SQL> create user COURS identified by COURS;  
SQL> grant connect, resource to COURS;
```

3. Se connecter à SQL*Plus avec le compte créé :

```
SQL> connect COURS/COURS
```

4. Exécuter le script SQL (fichier contenant les instructions) :

```
SQL> @c:\cours\sql\crebase.sql
```

1. Lancer l'utilitaire SQL*Plus,

en se connectant avec un compte ' administrateur B.D. ' .

Ce compte sera utilisé UNIQUEMENT pour des tâches d'administration (création de comptes, attribution de droits, etc.)

2. Créer un compte utilisateur.

Le compte SYSTEM est propriétaire d'un certain nombre de tables ' système '. Il est préférable de créer les tables supplémentaires sous un autre compte. Une B.D. peut contenir des tables sous différents comptes. Chaque compte ayant créé des tables est appelé un **schéma**.

Un compte doit recevoir quelques privilèges (droits)

afin de pouvoir utiliser la B.D. :

CONNECT = le droit de se connecter à Oracle

RESOURCE = le droit de créer des objets dans la B.D.

3. Se connecter à SQL*Plus avec le compte créé.

Ceci peut se faire de 2 manières :

- sans quitter SQL*Plus, comme indiqué dans le diapositive
- Quitter SQL*Plus. Relancer SQL*Plus en se connectant avec le nouveau compte

4. Exécuter le script SQL.

Après le caractère @, on indique le chemin et le nom du fichier contenant les instructions

SQL. Si le chemin contient des espaces, il faut mettre le tout entre " .

exemple : @ "c:\Mes documents\oracle\crebase.sql"