Cours de bases de données

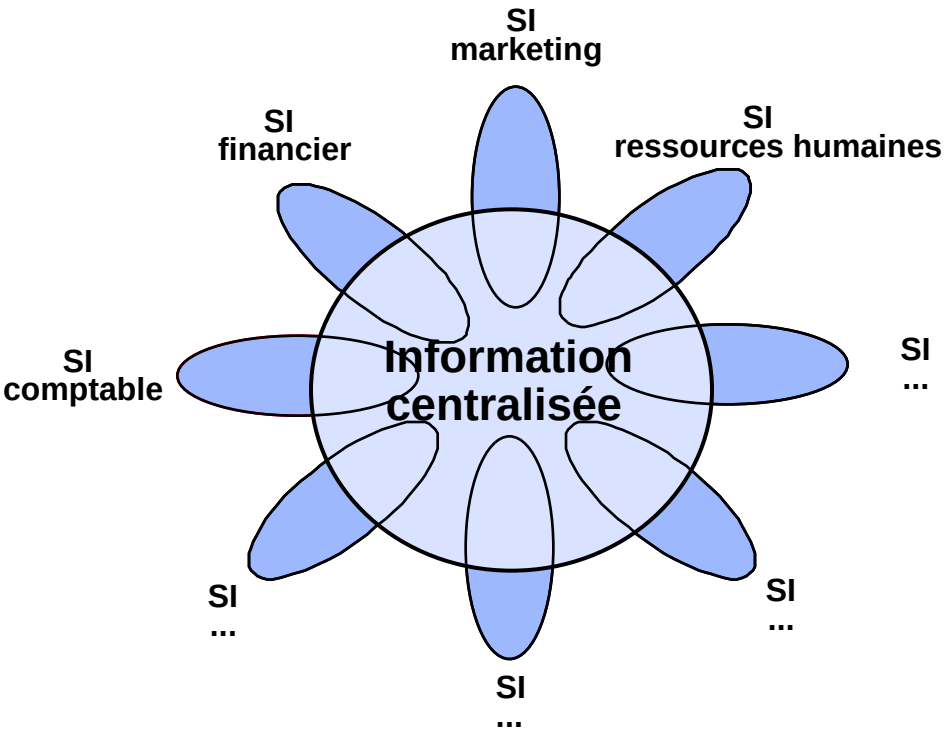
Dr Sekhri Arezki

Département des Sciences de Gestion

Université d'Oran2

Intérêt des bases de données (BD) ?

❖ Mise en œuvre et exploitation du système d'information (SI)



Objectifs du cours

- Etudier comment concevoir et réaliser une BD

  application : réalisation de bases de données avec le logiciel ACCESS

- Etudier comment accéder à des info. stockées dans une BD

  application : utilisation du langage d'interrogation graphique d'ACCESS et du langage SQL

Bibliographie

- H. Korth, A. Silberschatz
  Systèmes de Gestion de Bases de Données
  éd. McGraw-Hill

- J.M. Bourguignon
  SQL: concevoir et programmer les bases de données relationnelles
  éd. Dunod

- D. Dionisi
  L'essentiel sur Merise
  éd. Eyrolles, 1991

- Aide-mémoire ACCESS 97

Plan du cours

Chapitre 1 - Les Systèmes de Gestion de Bases de Données

Chapitre 2 - Conception de bases de données relationnelles

Chapitre 3 - Manipulation de données

Introduction - SI et base de données

❖ La conception d'un système d'information (SI) présente 2 facettes :

- conception du système d'information **organisationnel**

  quelle information doit être gérée, comment doit-elle être gérée, qui y accède et avec quels droits ?

- conception du système d'information **informatisé**

  concerne l'informatisation de l'information

  ⇨ **Base de Données**

Chapitre 1
Les systèmes de gestion de bases de données

- Bases de données (BD) et Système de Gestion de Bases de Données (SGBD)
- Fonctions d'un SGBD
- Utilisateurs des SGBD
- Marché des SGBD

Base de données et système de gestion de bases de données

❖ Base de Données (BD) :

- ensemble **structuré** de données **inter-reliées**
- exemple : SNCF

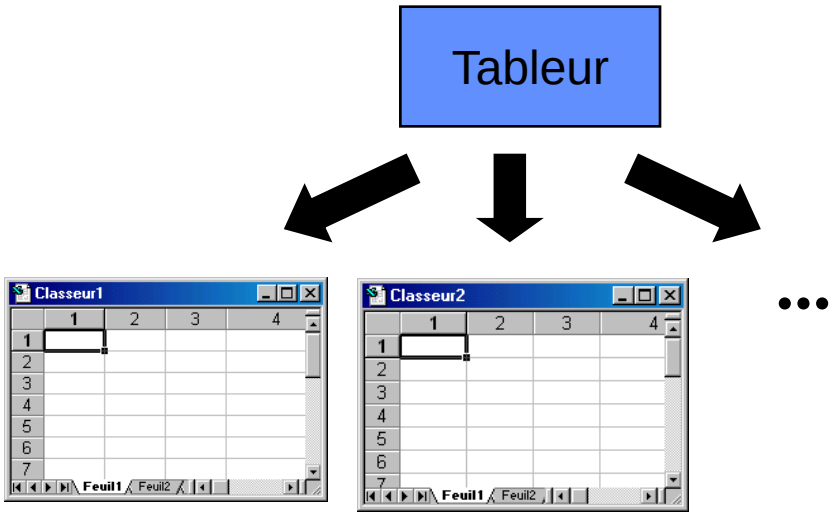
❖ Système de Gestion de Bases de Données (SGBD) :

- logiciel permettant de créer et manipuler une BD, soit directement, soit à travers des programmes d'application

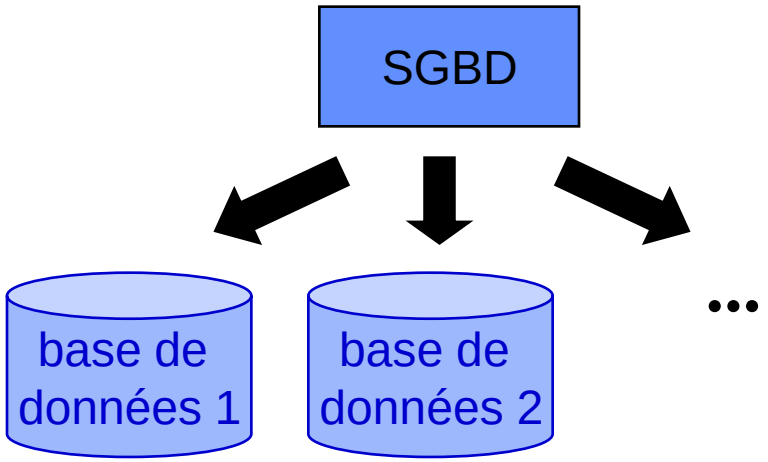
Fonctions d’un SGBD

- Autoriser l'accès partagé aux données
  - accès aux mêmes données (localisations différentes)
  - accès simultané à la base (concurrence)
- Autoriser une administration efficace des données
- Gérer la confidentialité des données
  - en fonction des utilisateurs
  - en fonction des modes d’accès (lecture, écriture)
- Assurer la sécurité des données et la reprise après panne

Tableur *vs.* SGBD



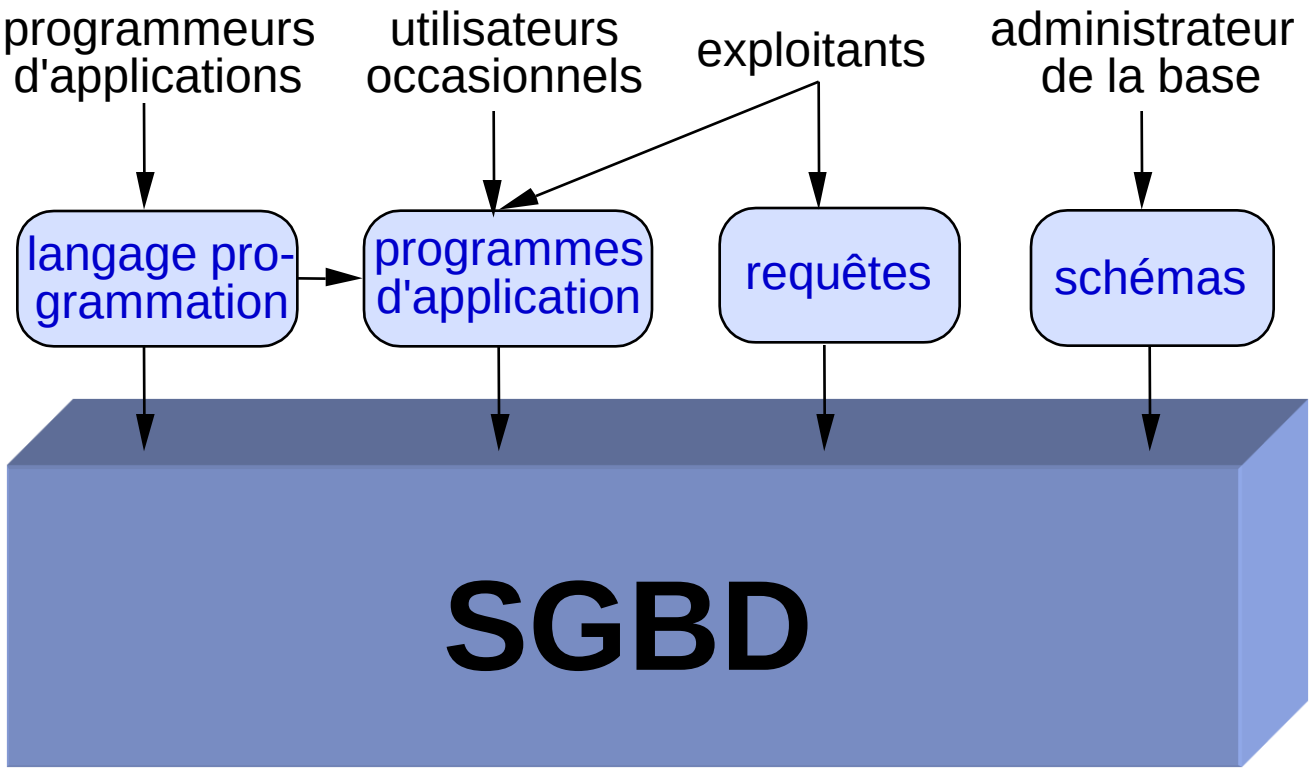
orienté vers le calcul



orienté vers la manipulation de données : requêtes, ...

Classeur	Base de données
de une à quelques feuilles de calcul	grand nombre de tables (plusieurs centaines)
quelques milliers de données	quelques millions de données
utilisation individuelle	utilisation en réseau
données en mémoire vive	données sur disque

Différents utilisateurs d'un SGBD



Différents utilisateurs d'un SGBD (suite)

- Programmeurs d'applications
  - informaticiens manipulant les BD grâce à des langages de programmation

    ex: développement de l'application de réservation de TGV sur Internet

- Utilisateurs occasionnels
  - personnes utilisant la BD par le biais de programmes d'application

    ex: vous, sur votre PC connecté à Internet

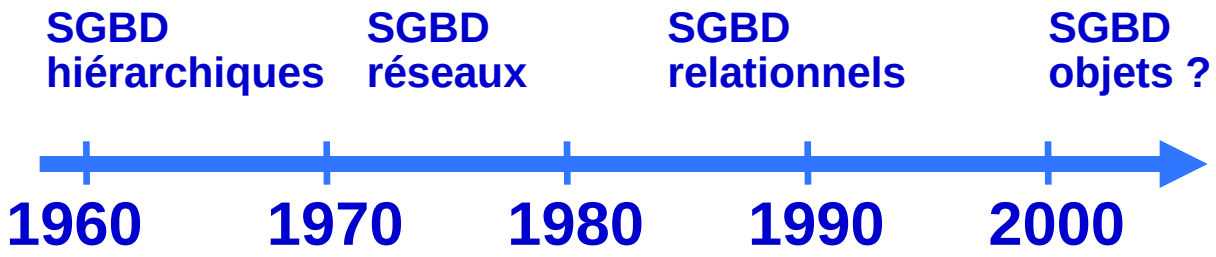
- Exploitants
  - agents utilisant et manipulant la BD par le biais de programmes d'application ou de requêtes

    ex: agent de la SNCF

- Administrateur de la base de données
  - Assure le contrôle de la BD et des programmes d'applications

Le marché des SGBD

❖ 4 générations principales de SGBD :



❖ Les SGBD relationnels dominent aujourd'hui le marché

❖ Exemples de SGBD relationnels :

- ORACLE, INGRES, DB/2, ...
- ACCESS, SQL Server, …
  - intégration dans la bureautique

Outils de gestion basés sur les SGBD

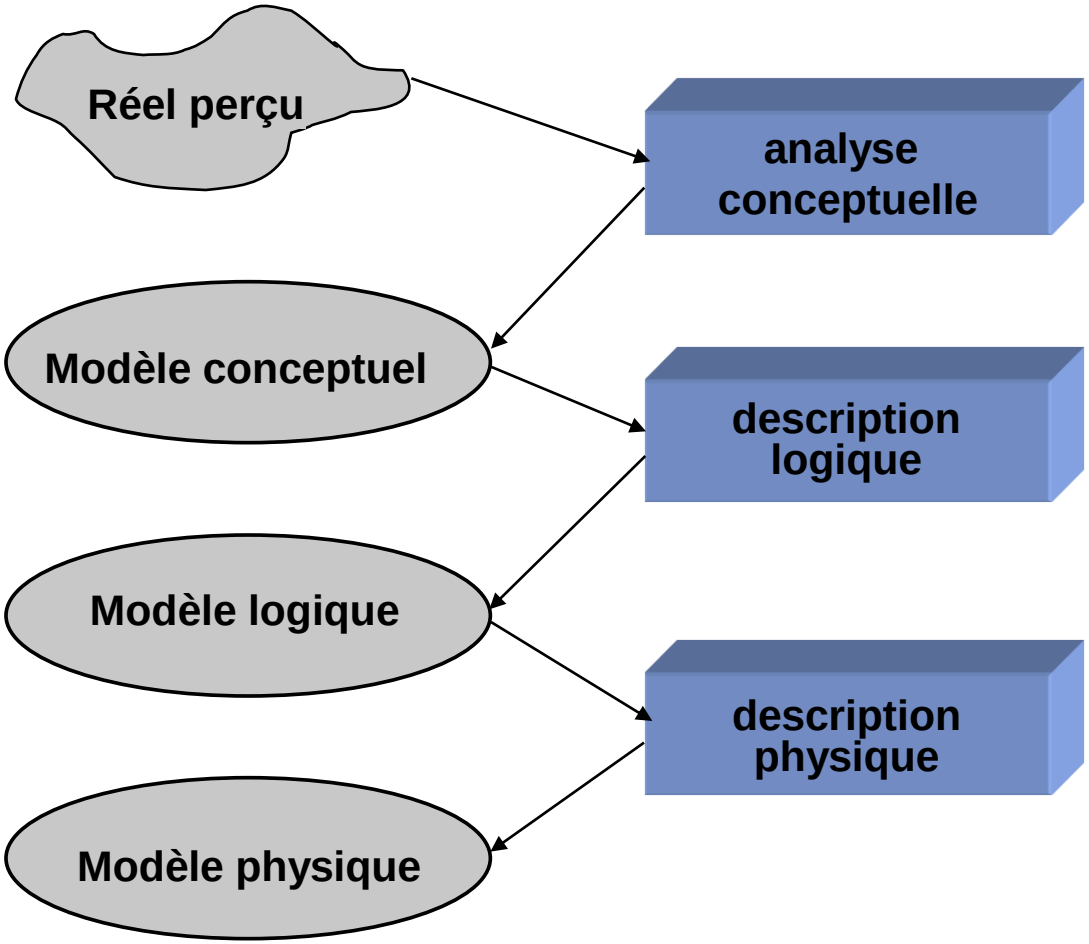
- Les ERP
- Les outils de reporting
- Datawarehouse
- Data mining

Chapitre 2
Conception de BD relationnelles

- Etapes de la conception d'une base de données
- Conception de la base de données : le modèle entité-association
- Réalisation de la base de données : le modèle relationnel
- Passage du modèle entité-association à un modèle relationnel

Etapes de la conception d'une BD

❖ 3 étapes :



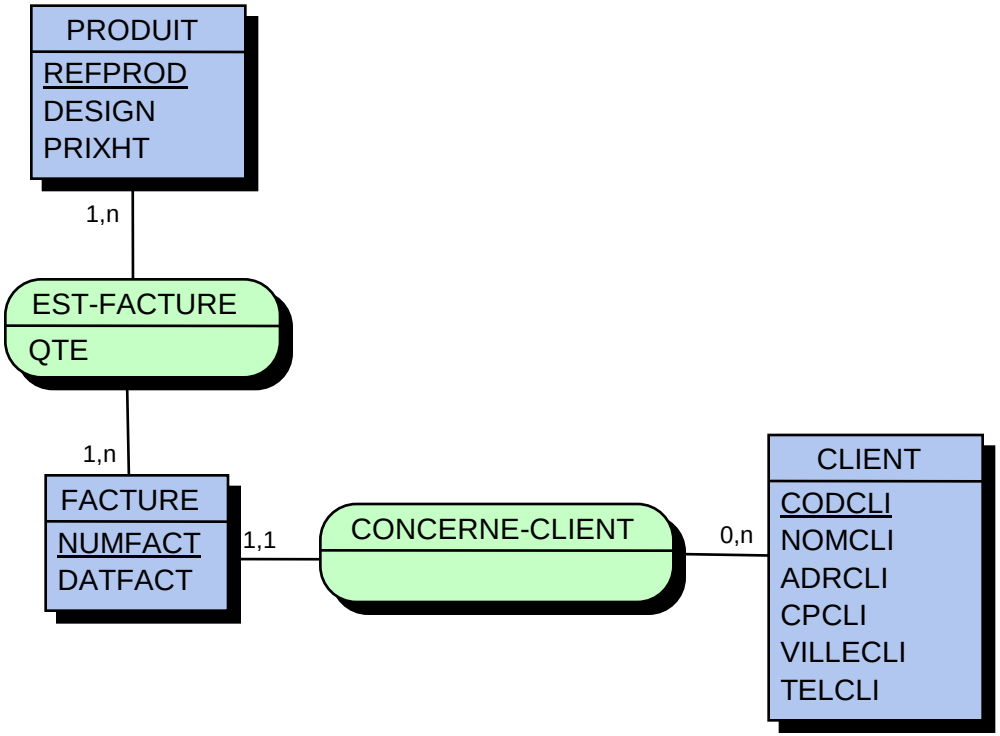
- à chaque étape correspond une vision différente des données

Modèle conceptuel

- Permet d'analyser et de modéliser les données indépendamment :
  - d'une organisation particulière des données (relationnel, objets, ...)
  - d'un SGBD particulier

- Dans le cours, le modèle conceptuel de type **entité-association** est étudié

exemple :



Modèle logique

- Description des données utilisant un des modèles suivants : hiérarchique, réseau, relationnel, objet, …
  - c'est à ce niveau qu'est fait le choix du SGBD

- Dans le cours, le modèle logique de type **relationnel** est étudié

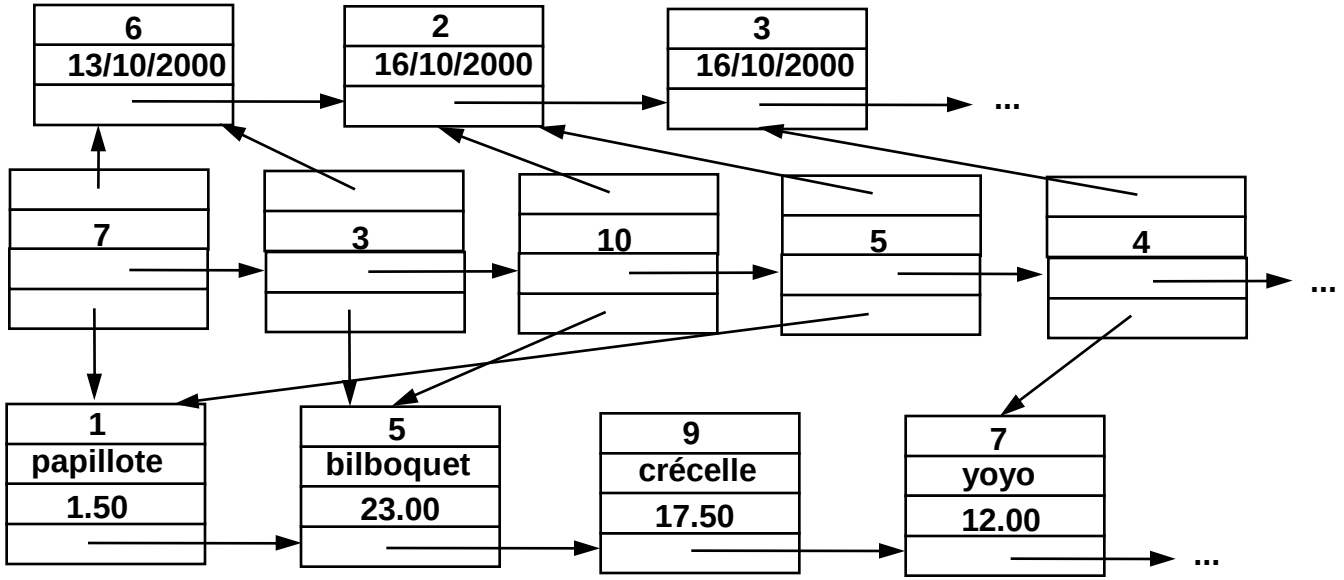
exemple:

FACTURE	NUMFACT	DATFACT
	6	13/10/2000
	2	16/10/2000
	3	16/10/2000

EST-FACTURE	NUMFACT	REFPROD	QTE
	6	1	7
	6	5	3
	2	5	10
	2	1	5
	3	7	4

Modèle physique (ou interne)

- Modèle définissant stockage et organisation des données:
  - stockage des données sur les fichiers (nom, organisation, localisation, …)
  - stockages des enregistrements dans les fichiers (champs, longueur, …)
  - définition des accès aux articles (index, contraintes d'intégrité, …)
- pris en charge (pour l'essentiel) par le SGBD
- joue un rôle important au niveau des performances



Conception de BD: le modèle entité-association

❖ **Modèle entité-association** :

- c'est un modèle **conceptuel de données** (MCD), c'est-à-dire une représentation abstraite des données indépendante :
  - de l'organisation des données
  - du SGBD utilisé
- utilise une représentation graphique des données : bon outil de communication entre les concepteurs et les utilisateurs finaux
- technique de conception très utilisée dans les méthodes actuelles d'analyse de SI: MERISE, ...
- peut être implanté avec un SGBD hiérarchique, réseau ou relationnel

❖ Principe :

données regroupées en classes **d'entités** et liées par des **associations**

Entité et classe d’entités

- **Entité** : objet discernable parmi d’autres objets
  - peut être concret ou abstrait
  - exemple : le produit de référence AX-37667, la facture n° 6765

- **Classe d’entités** : ensemble d’entités similaires pouvant être regroupées
  - exemples : les produits, les factures, ...
  - chaque classe d'entités possède un nom : PRODUIT, FACTURE, …

Attribut et identifiant

❖ **Attribut d'une classe d'entités** : caractéristique des entités d'une classe

- chaque attribut porte un nom
- chaque attribut possède une valeur dans un domaine

pour une entité donnée, un attribut possède une **et une seule** valeur

- exemples :

pour la classe PRODUIT :

REFPROD (chaîne de car.)
DESIGN (chaîne de car.)
PRIXHT (réel)

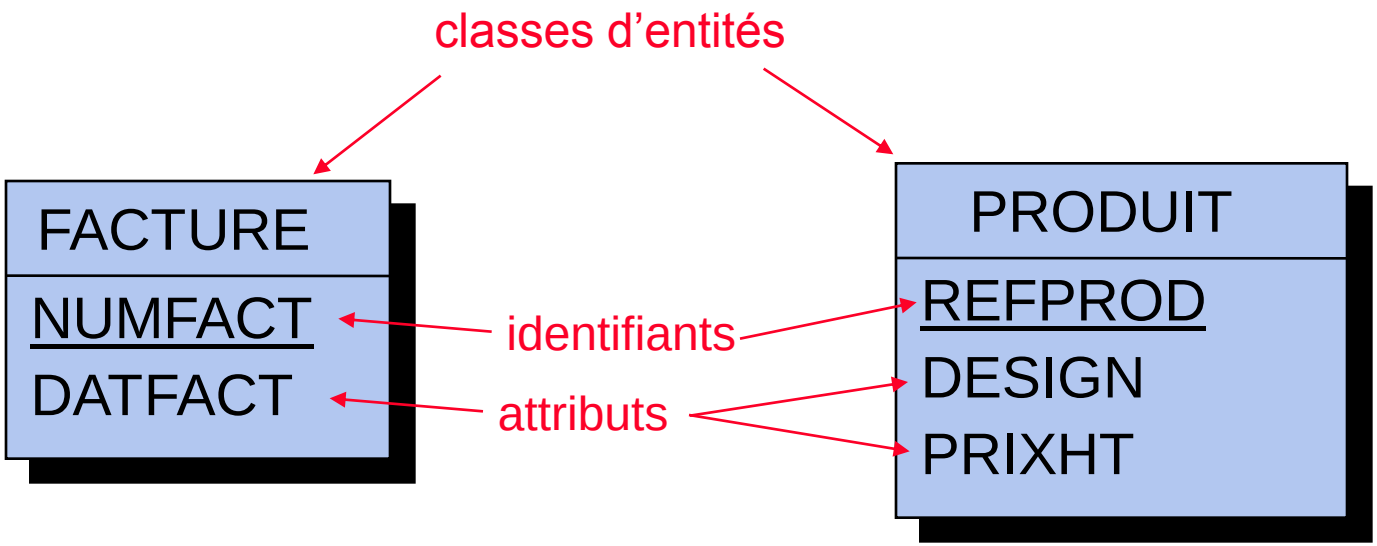
pour la classe FACTURE :

NUMFACT (entier)
DATFACT (date)

❖ **Identifiant (ou clé) d'une classe d'entités** : ensemble minimal d'attributs déterminant de manière unique une entité dans la classe

- exemples : REFPROD, NUMFACT

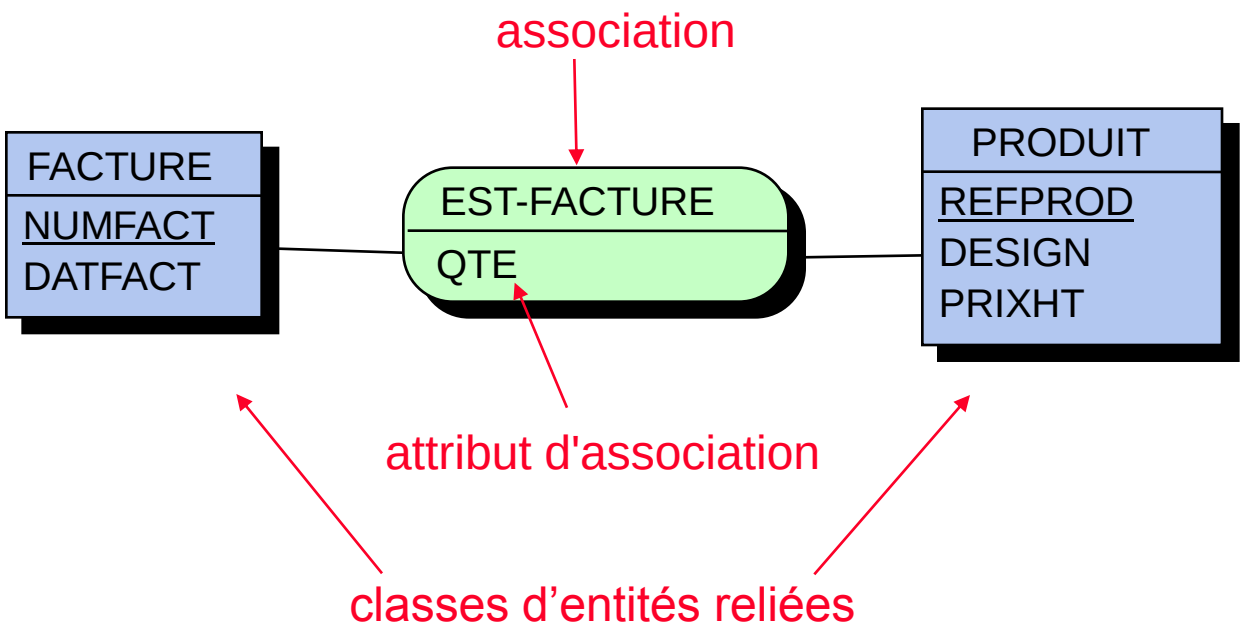
Représentation graphique d'une classe d'entités



Association

- **Association** : relie plusieurs classes d'entités (deux ou plus)
  - porte un nom
  - exemple : l'association EST-FACTURÉ entre les classes PRODUIT et FACTURE matérialise le fait que les produits sont facturés sur des factures
  - peut avoir des attributs (ex : quantité facturée, ...)

- Représentation graphique d'une association :



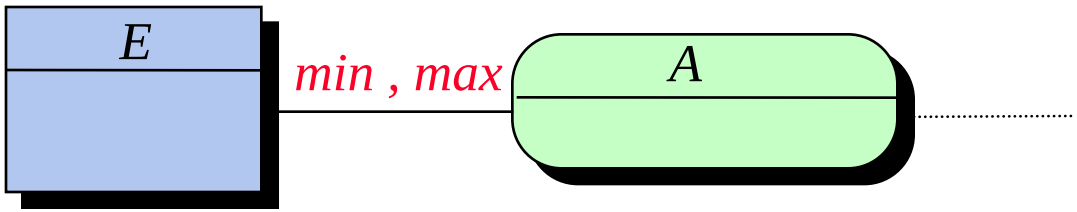
Cardinalité d'une association

- Cardinalité d'une association A vis-à-vis d'une classe d'entités E :

  nombre minimum et maximum de fois où une entité **donnée** de la classe E peut apparaître dans l'association A

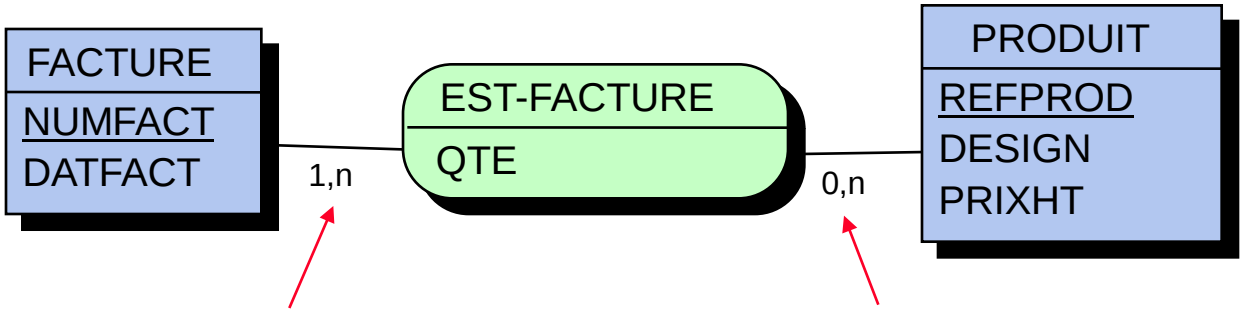
- Cardinalité mininimum:
  - 0 : il peut exister des entités de E qui n'apparaissent pas dans A
  - 1 : toute entité de E apparaît au moins une fois dans A

- Cardinalité maximum:
  - 1 : toute entité de E apparaît au plus une fois dans A
  - n : il peut exister des entités de E apparaissant plusieurs fois dans A

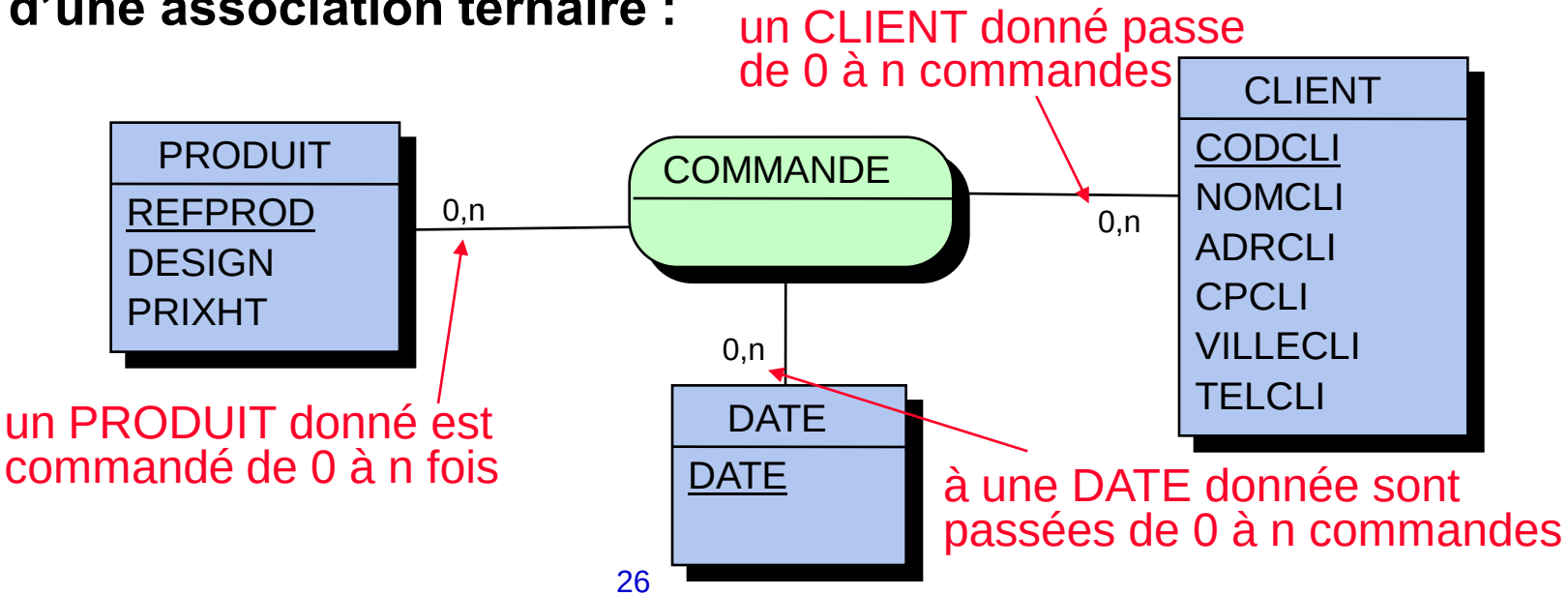


Cardinalité d'une association : exemples

❖ **Cas d'une association binaire :**



❖ **Cas d'une association ternaire :**



Construction d'un modèle entité-association

- Pour construire un modèle entité-association, on procède :
  - en analysant et critiquant l'existant (documents papier, ...)
  - en analysant les besoins des utilisateurs ou futurs utilisateurs
  - en étudiant les éventuels logiciels existants

- La construction d'un modèle entité-association s'appuie sur deux représentations complémentaires :
  - le dictionnaire des données
  - le graphe de dépendances fonctionnelles

Dictionnaire des données

❖ Inventaire des données manipulées :

Attribut	Signification	Domaine
...	...	...
...	...	...
...	...	...

- mettre seulement les **données élémentaires**, c'est-à-dire les attributs ne pouvant pas être obtenus par calcul

❖ Exemple :

Attribut	**Signification**	**Domaine**
REFPROD	Référence du produit	Chaîne(12)
DESIGN	Désignation du produit	Chaîne(30)
PRIXHT	Prix unitaire HT	réel
NUMFACT	Numéro de la facture.	entier
DATFACT	Date de la facture.	Date/heure
QTE	Quantité facturée	entier
...	...	...

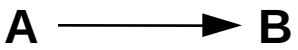
- les attributs calculés sont obtenus par programmation - ex : PRIXTTC

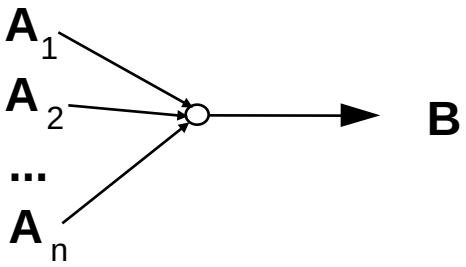
Dépendances fonctionnelles (DF)

- Dépendance fonctionnelle d'un attribut A vers un attribut B : la connaissance de la valeur de A détermine une valeur **unique** de B

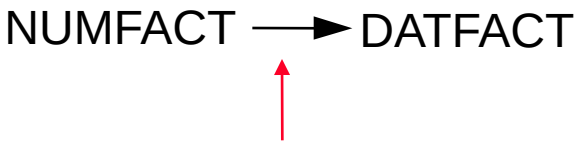
- Généralisation aux cas de plusieurs attributs : la connaissance des attributs A_1, A_2,...,A_n détermine une valeur **unique** de B

- Représentation graphique :

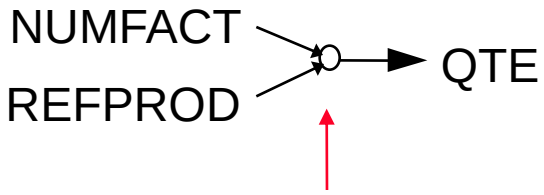




- Exemples :



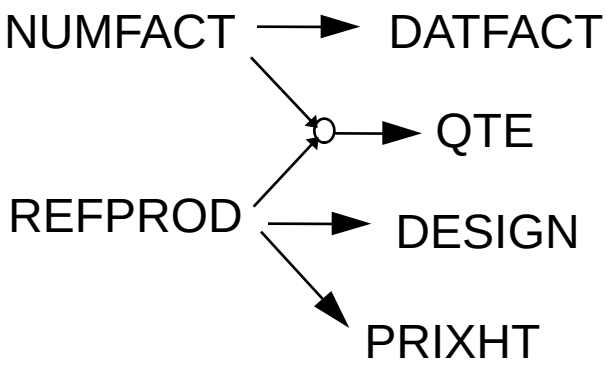
un n° de facture détermine une et une seule date de facture



un n° de facture et une ref. de produit déterminent une et une seule quantité facturée

Graphe de dépendances fonctionnelles

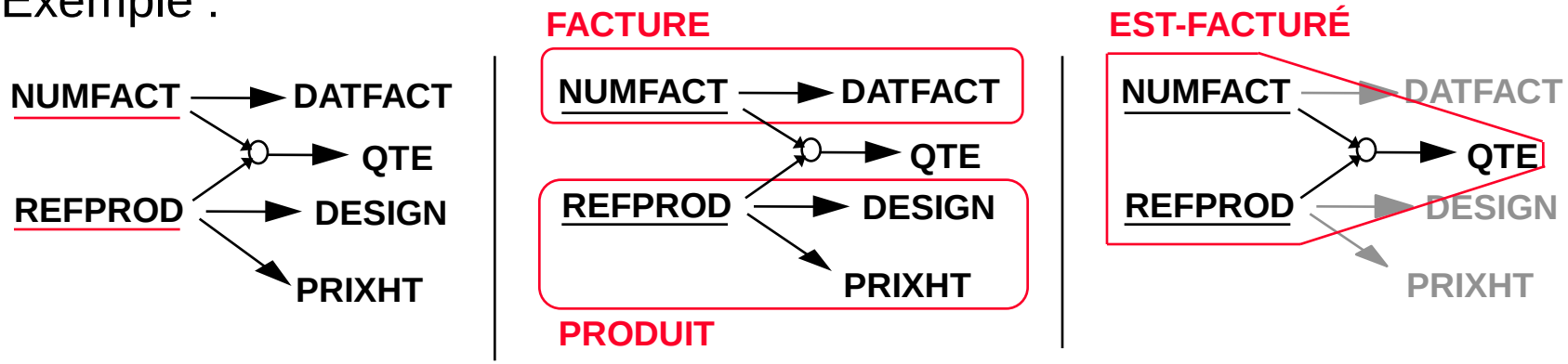
- **Graphe de dépendances fonctionnelles** : graphe dans lequel on représente l'ensemble des DF

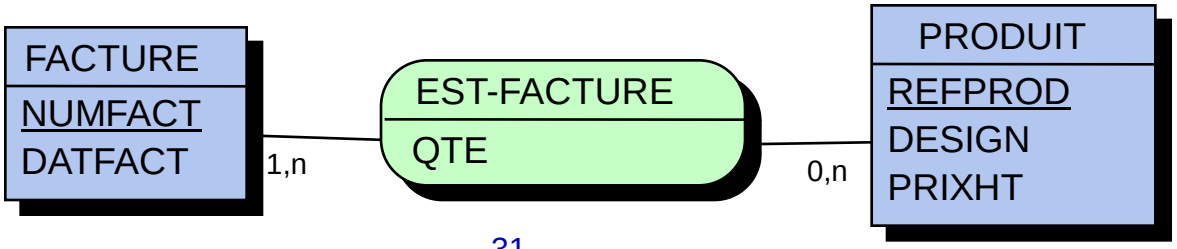


Graphe de DF et modèle entité-association

- Le graphe de DF facilite la construction du modèle entité-association :
  1. déterminer les identifiants
  2. déterminer les classes d'entités : identifiants et attributs qui en dépendent directement et uniquement d'eux
  3. déterminer les associations : construites à partir DF restantes

- Exemple :



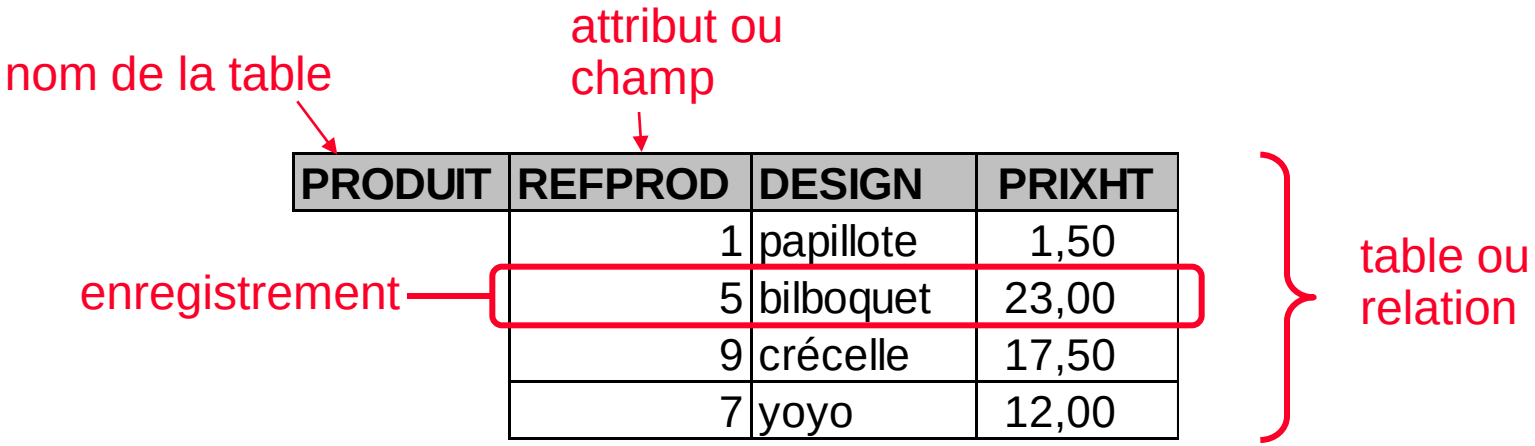


Modèle logique des données : le modèle relationnel

- **Principe du modèle relationnel :** toutes les données sont stockées dans des tables

- **Relation** ou **table** : ensemble des enregistrements issus d'une classe d'entités ou d'une association

- **Attribut** (ou **champ**) d'une relation :
  - matérialise un attribut d'une classe d'entités ou d'une association
  - caractérisé par un nom et un domaine de valeurs



☞ ordre des enregistrements sans importance

BD relationnelle

- Une BD relationnelle est une collection de tables
- Exemple :

PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillote	1,50
	5	bilboquet	23,00
	9	crécelle	17,50
	7	yoyo	12,00

FACTURE	NUMFACT	DATFACT
	6	13/10/2000
	2	16/10/2000
	3	16/10/2000

EST-FACTURE	NUMFACT	REFPROD	QTE
	6	1	7
	6	5	3
	2	5	10
	2	1	5
	3	7	4

Modèle relationnel et SGBD relationnel

❖ Les SGBD relationnels permettent de construire et manipuler des relations :

- le **schéma** (ou la **structure**) des relations : nom de relation + liste des attributs
- le contenu des relations (enregistrements)

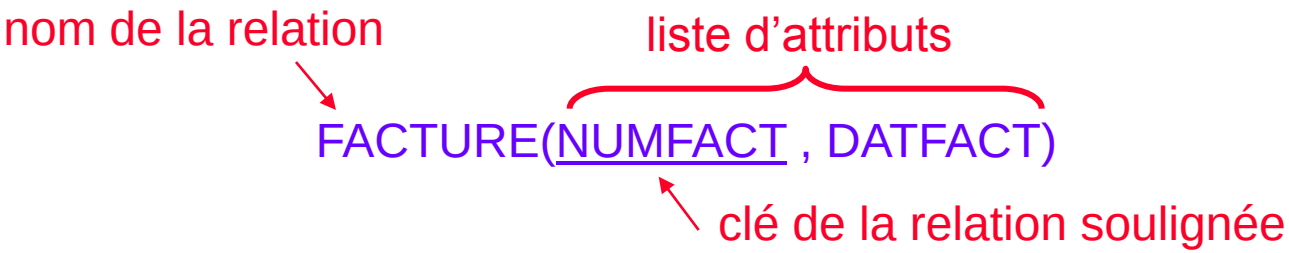
❖ Le modèle relationnel exige que chaque relation ait une **clé**

- **clé** (ou **clé primaire**) d'une relation : sous-ensemble minimum d'attributs d'une relation qui détermine les autres

  exemple - pour FACTURE : NUMFACT

- correspond à la notion d'identifiant dans un modèle entité-association

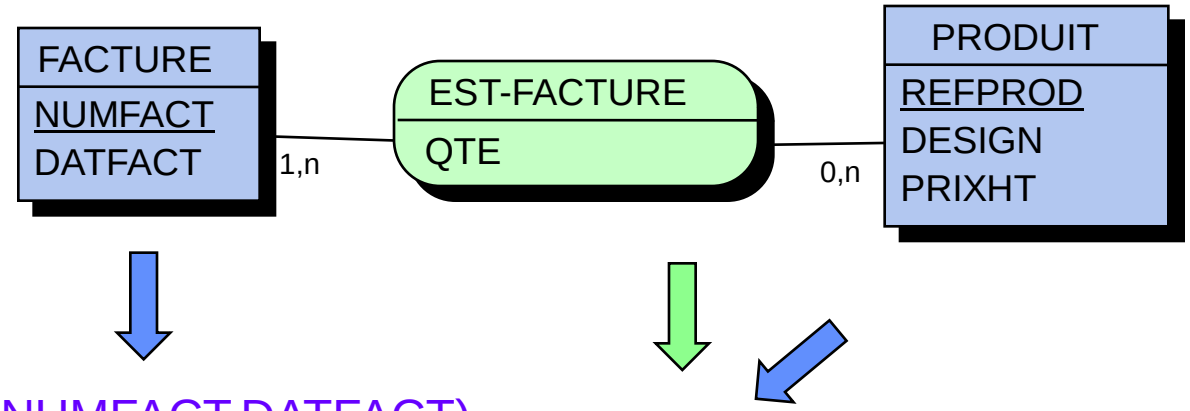
❖ Le schéma d'une relation est souvent noté : FACTURE(NUMFACT , DATFACT)



Passage du modèle entité-association au relationnel

1. A chaque classe d'entités correspond une relation :
   - nom : nom de la classe d'entités
   - attributs : attributs de la classe d'entités
   - clé : identifiant de la classe d'entités

2. A chaque association correspond une relation :
   - nom : nom de l'association
   - attributs : identifiants des classes d'entités reliées + attributs de l'association
   - clé : ensemble des identifiants des classes d'entités reliées



FACTURE(NUMFACT,DATFACT)
PRODUIT(REFPROD,DESIGN,PRIXHT)
EST-FACTURE(NUMFACT,REFPROD,QTE)

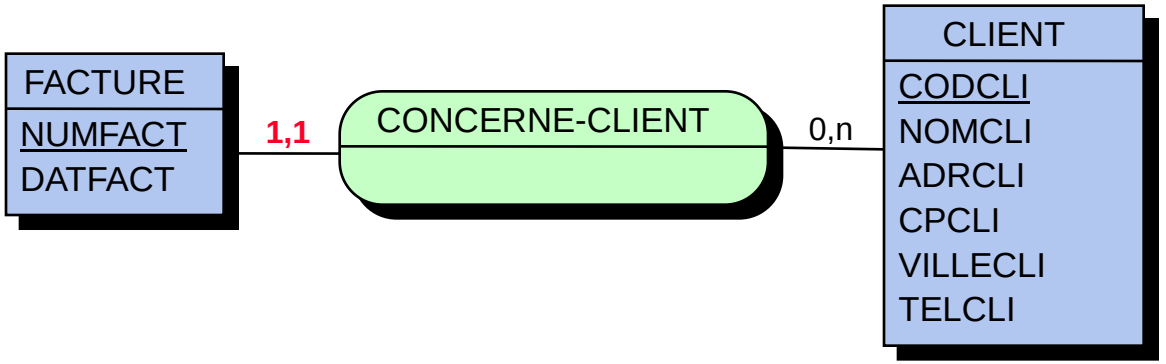
Passage du modèle entité-assoc. au relationnel (suite)

3. Simplifier le modèle relationnel:

les associations binaires ayant une cardinalité 0,1 ou 1,1 vis-à-vis d'une classe d'entités peuvent être supprimées en déplaçant l'identifiant relié

☞ clé étrangère

exemple :



normalement :

FACTURE(NUMFACT,DATFACT)

CLIENT(CODCLI,NOMCLI,ADRCLI,CPCLI,VILLECLI,TELCLI)

CONCERNE-CLIENT(NUMFACT,CODCLI)

après simplification :

FACTURE(NUMFACT,DATFACT,CODCLI)

CLIENT(CODCLI,NOMCLI,ADRCLI,CPCLI,VILLECLI,TELCLI)

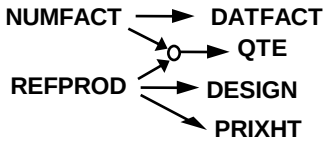
~~**CONCERNE-CLIENT(NUMFACT,CODCLI)**~~

Résumé de la démarche de la réalisation d'une BD

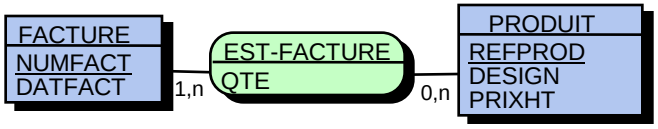
1. Dico. des données

Attribut	Signification	Domaine
REFPROD	Référence du produit	Chaîne(12)
DESIGN	Désignation du produit	Chaîne(30)
PRIXHT	Prix unitaire HT	réel
...	...	...

2. Graphe des DF



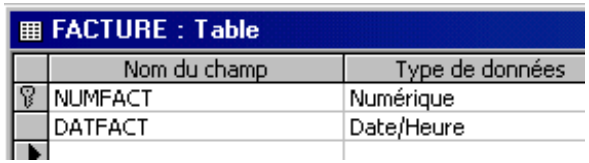
3. Modèle entité-association

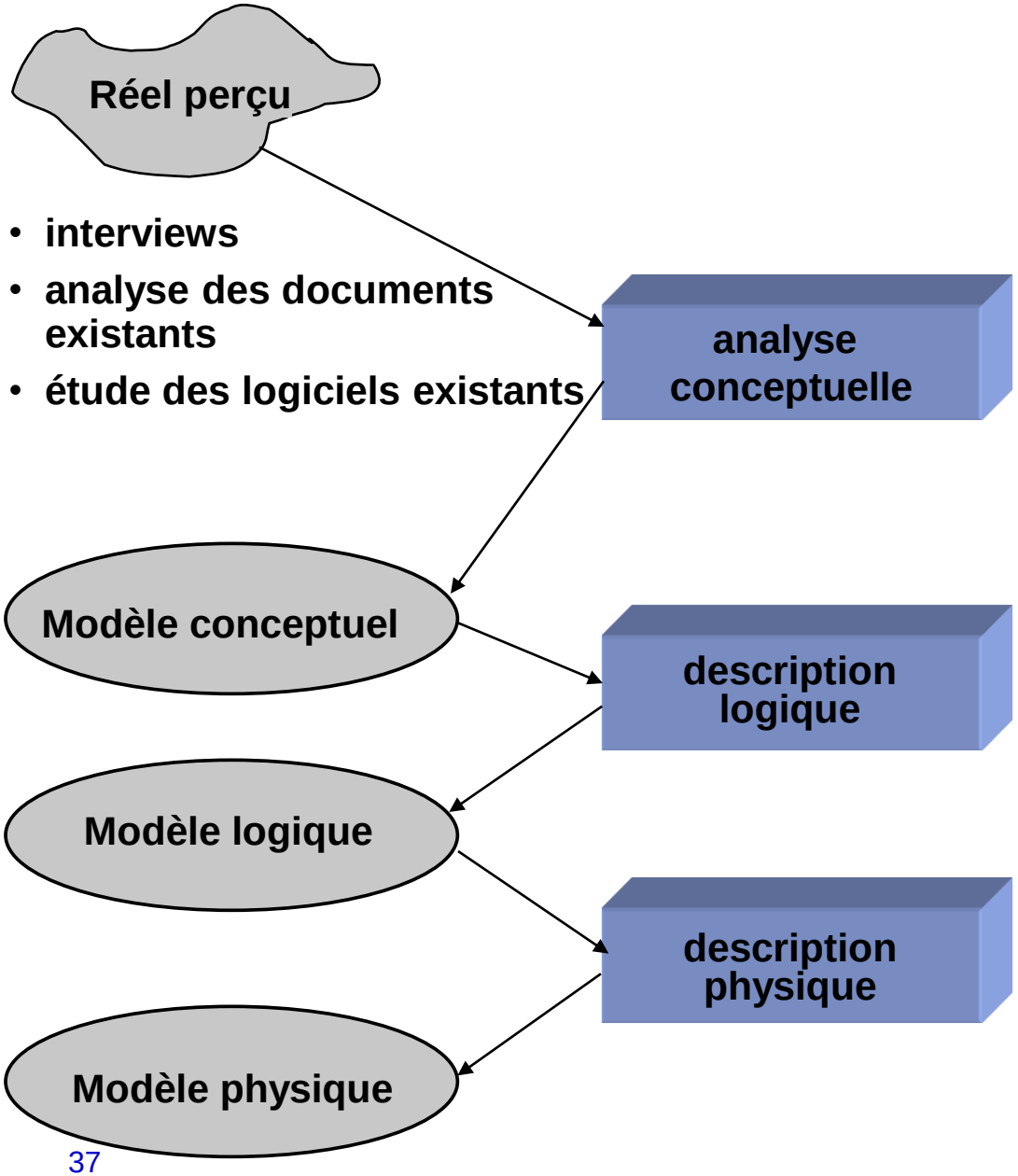


4. Schéma de la BD

FACTURE(NUMFACT,DATFACT)
PRODUIT(REFPROD,DESIGN,PRIXHT)
EST-FACTURE(NUMFACT,REFPROD,QTE)

5. Mise en œuvre sur le SGBD





Chapitre 3

Manipulation de données en SQL

- Manipulation de données
- Le langage SQL

Manipulation de données

- Différentes opérations :
  - recherche d'information
  - insertion de données
  - mise à jour de données
  - suppression de données
- Langage de manipulation de données : langage permettant d’interroger et de modifier le contenu d'une BD

Exemple de BD relationnelle

PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillotes	12,00
	5	bilboquet	23,00
	9	crécelle	17,50
	7	yoyo	12,00

FACTURE	NUMFACT	DATFACT
	6	13/10/2000
	2	16/10/2000
	3	16/10/2000

EST-FACTURE	NUMFACT	REFPROD	QTE
	6	1	7
	6	5	3
	2	5	10
	2	1	5
	3	7	4

ANC-PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillotes	11,00
	9	crécelle	17,00
	4	sifflet	6,00
	5	bilboquet	23,00
	2	cerceau	35,00

Le langage SQL

- Langage créé par IBM, devenu un standard pour la manipulation de données

- SQL comporte 4 mots-clés principaux :
  - SELECT : recherche
  - INSERT : ajout
  - UPDATE : mise à jour
  - DELETE : suppression

La recherche de données : SELECT

PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillotes	12,00
	5	bilboquet	23,00
	9	crécelle	17,50
	7	yoyo	12,00

❖ Syntaxe d'une requête SELECT :

SELECT $champ_1$, ... , $champ_n$ ← champs présents dans le résultat de la requête

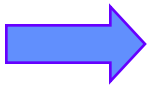
FROM $table_1$, ... , $table_m$ ← tables (ou requêtes) utilisées

optionnel { WHERE *condition* ← condition devant être vérifiée par un enreg. pour figurer dans le résultat

ORDER BY $champ_i$, ... , $champ_j$ ← ordre de présentation des enreg. du résultat de la requête

❖ Exemple :

```
SELECT REFROD, DESIGN
FROM    PRODUIT
WHERE  PRIXHT < 20
ORDER BY DESIGN
```



	REFPROD	DESIGN
	9	crécelle
	1	papillotes
	7	yoyo

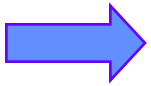
la clause SELECT

PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillotes	12,00
	5	bilboquet	23,00
	9	crécelle	17,50
	7	yoyo	12,00

❖ SELECT * : retourne tous les champs

- exemple :

```
SELECT *
FROM   PRODUIT
WHERE  PRIXHT < 20
```

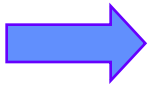


	REFPROD	DESIGN	PRIXHT
	1	papillotes	12,00
	9	crécelle	17,50
	7	yoyo	12,00

❖ SELECT DISTINCT supprime les valeurs identiques

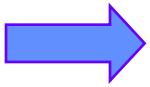
- exemple :

```
SELECT DISTINCT PRIXHT
FROM   PRODUIT
WHERE  PRIXHT < 20
```



	PRIXHT
	12,00
	17,50

```
SELECT DISTINCT REFPROD, PRIXHT
FROM   PRODUIT
WHERE  PRIXHT < 20
```



	REFPROD	PRIXHT
	1	12,00
	9	17,50
	7	12,00

car (1 , 12,00) ≠ (7 , 12,00)

La clause WHERE

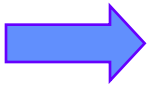
PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillotes	12,00
	5	bilboquet	23,00
	9	crécelle	17,50
	7	yoyo	12,00

❖ Sélectionne les enregistrements vérifiant une condition, composée de :

- noms de champs
- constantes num. (ex: 2.3), chaînes (ex: "Dupont"), dates (ex: #21/10/2000#)
- opérateurs (=, <>, <, <=, >, >=, +, -, *, /, …) et fonctions (sin, log, …)
- opérateurs logiques : OR, AND, NOT

❖ Exemple : désignation des produits dont le prix est entre 15 et 20 F

```
SELECT DESIGN
FROM   PRODUIT
WHERE  PRIXHT >= 15
AND    PRIXHT <= 20
```



DESIGN
crécelle

⚠ Utiliser des parenthèses en cas d'utilisation conjointe de AND et OR

- exemple : désign. des produits dont le prix est entre 15 et 20 F ou dont la réf. est supérieure 2

```
SELECT DESIGN
FROM   PRODUIT
WHERE  (PRIXHT >= 15
AND    PRIXHT <= 20)
OR     REFPROD > 2
```

La clause WHERE (suite)

PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillotes	12,00
	5	bilboquet	23,00
	9	crécelle	17,50
	7	yoyo	12,00

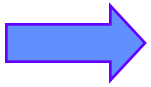
❖ Autres opérateurs dans la clause WHERE :

- comparaison avec un motif : LIKE "*motif*"
  où *motif* contient:

  ? un caractère quelconque
  ***** une suite quelconque de caractères (éventuellement vide)
  # un chiffre quelconque

exemple : désignation produits dont le nom commence par 'c'

```
SELECT DESIGN
FROM   PRODUIT
WHERE  DESIGN LIKE "c*"
```

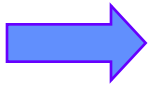


DESIGN
crécelle

- opérateur IS NULL signifiant que le champ n'est pas rempli

exemple : réf. des produits dont la désignation n'est pas remplie

```
SELECT REFPROD
FROM   PRODUIT
WHERE  DESIGN IS NULL
```



REFPROD

La clause FROM

PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillotes	12,00
	5	bilboquet	23,00
	9	crécelle	17,50
	7	yoyo	12,00

EST-FACTURE	NUMFACT	REFPROD	QTE
	6	1	7
	6	5	3
	2	5	10
	2	1	5
	3	7	4

- Lorsque plusieurs tables sont utilisées dans une requête, il faut les "joindre" sur leur(s) champ(s) commun(s)
  - exemple : désignation des produits facturés

```
SELECT DESIGN
FROM   EST-FACTURE, PRODUIT
WHERE  EST-FACTURE.REFPROD = PRODUIT.REFPROD
```

} **jointure**

champ REFPROD de la table EST-FACTURE

champ REFPROD de la table PRODUIT

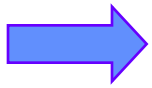
EST-FACTURE(NUMFACT, REFPROD, QTE)

PRODUIT(REFPROD, DESIGN, PRIXHT)

EST-FACTURE PRODUIT

	NUMFACT	REFPROD	QTE	REFPROD	DESIGN	PRIXHT
	6	1	7	1	papillotes	1.50
	6	5	3	5	bilboquet	23.00
	2	5	10	5	bilboquet	23.00
	2	1	5	1	papillotes	1.50
	3	7	4	7	yoyo	12.00

=



	DESIGN
	papillotes
	bilboquet
	bilboquet
	papillotes

La clause FROM (suite)

- Lorsque deux tables n'ont pas de champ commun, il faut utiliser les tables intermédiaires pour effectuer la jointure
  - exemple : désignation des produits facturés après le 14/10/2000

PRODUIT — FACTURE

FACTURE(NUMFACT, DATFACT)

↕ ?

PRODUIT(REFPROD, DESIGN, PRIXHT)

→

FACTURE(NUMFACT , DATFACT)

EST-FACTURE(NUMFACT, REFPROD, QTE)

PRODUIT(REFPROD , DESIGN , PRIXHT)

```
SELECT DESIGN
FROM   FACTURE, EST-FACTURE, PRODUIT
WHERE  EST-FACTURE.REFPROD = PRODUIT.REFPROD
AND    EST-FACTURE.NUMFACT = FACTURE.NUMFACT
AND    DATFACT > #14/10/2000#
```

} **jointures**

☞ utiliser le modèle entité-association pour trouver les tables intermédiaires

La clause FROM (fin)

- Une requête peut être basée sur une (ou plusieurs) autres requête(s) :

  SELECT ...
  FROM ***requête***$_1$, ...
  ...

- Intérêt :
  - écrire des requêtes qui peuvent être réutilisées dans d'autres requêtes
  - décomposer une requête complexe à écrire en requêtes plus simples

- Exemple : désignation des produits ayant été facturés, dont le prix a augmenté depuis l'ancien catalogue (table ANC-PRODUIT)

R1

```
SELECT   REFPROD,PRIXHT
FROM     EST-FACTURE, PRODUIT
WHERE    EST-FACTURE.REFPROD
         = PRODUIT.REFPROD
```

```
SELECT REFPROD
FROM   R1, ANC-PRODUIT
WHERE  R1.REFPROD > ANC-PRODUIT.REFPROD
AND    R1.PRIXHT > ANC-PRODUIT.PRIXHT
```

seuls les champs présents dans la clause SELECT peuvent être réutilisés dans une autre requête

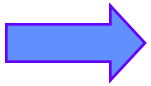
La clause ORDER BY

❖ Effectue le tri du résultat sur un ou plusieurs champs :

SELECT ...
FROM …
WHERE …
ORDER BY $champ_1$, … ASC ou DESC
optionnel

❖ Exemple : produits de moins de 20 F triés par désignation croissante

```
SELECT REFROD, DESIGN
FROM    PRODUIT
WHERE  PRIXHT < 20
ORDER BY DESIGN
```



	REFPROD	DESIGN
	9	crécelle
	1	papillotes
	7	yoyo

- par défaut, l'ordre est croissant; pour l'ordre décroissant, utiliser DESC
  ex: ORDER BY DESIGN DESC

- s'il y a plusieurs critères de tri, le plus à gauche est le critère primaire; en cas d'ex-aequo, le critère secondaire est pris en compte, etc.
  ex: ORDER BY DESIGN, REF

Calculs dans les requêtes

- Il est possible d'effectuer des calculs sur un ensemble d'enregistrements : nombre d'enregistrements, somme sur un champ, valeur min. ou max., ...

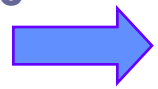
```
SELECT opération (champ)
FROM …
WHERE …
...
```

PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillotes	12,00
	5	bilboquet	23,00
	9	crécelle	17,50
	7	yoyo	12,00

- Principales opérations :
  - COUNT : nombre d'enregistrements
  - SUM : somme des valeurs du champ (numérique) sur un ensemble d'enreg.
  - AVG : moyenne des valeurs du champ (num.) sur un ensemble d'enreg.
  - MIN, MAX : valeur min. et max. dans l'ensemble d'enregistrements
  - opérations arithmétiques : +, -, *, /, ...

- Exemples:
  - réf. des produits avec leur prix TTC (TVA= 20,6%)

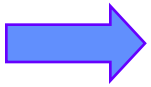
```
SELECT   REFPROD, PRIXHT * 1.206
FROM     PRODUIT
```



	REFPROD	
	1	1,81
	5	27,74
	9	21,11
	7	14,47

  - prix le plus élevé parmi les produits

```
SELECT   MAX(PRIXHT)
FROM     PRODUIT
```



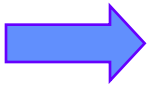
	23,00

Calculs dans les requêtes (suite)

PRODUIT	REFPROD	DESIGN	PRIXHT
	1	papillotes	12,00
	5	bilboquet	23,00
	9	crécelle	17,50
	7	yoyo	12,00

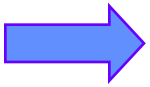
- Ne pas confondre COUNT (nombre d'enregistrements) et SUM (somme sur un champ) :

```
SELECT   COUNT(PRIXHT)
FROM     PRODUIT
WHERE    PRIXHT < 20
```



	3

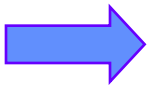
```
SELECT   SUM(PRIXHT)
FROM     PRODUIT
WHERE    PRIXHT < 20
```



	41,50

- Pour nommer une colonne en vue de réutilisation dans une autres requête : utiliser AS *nom de colonne* dans SELECT

```
SELECT   SUM(PRIXHT) AS TOTAL
FROM     PRODUIT
WHERE    PRIXHT < 20
```



	TOTAL
	41,50

Regroupements d'enregistrements

- **Regroupement** : possibilité de grouper des enregistrements sur un champ dont la valeur est identique
  - exemple : regroupement sur REFPROD dans EST-FACTURE

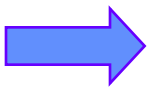
EST-FACTURE	NUMFACT	REFPROD	QTE
	6	1	7
	6	5	3
	2	5	10
	2	1	5
	3	7	4

- Regrouper des enregistrements :

```
SELECT
FROM …
WHERE …
GROUP BY champ1,...
```

- Une fois un regroupement effectué, il est possible d'effectuer des opérations sur chaque groupe
  - exemple: quantités facturées par produit

```
SELECT    REFPROD, SUM(QTE) AS QTETOT
FROM      EST-FACTURE
GROUP BY  REFPROD
```



	REFPROD	QTETOT
	1	12
	5	13
	7	4

Opérations ensemblistes en SQL

- Union de deux relations : UNION
- Intersection de deux relations : INTER
- Différence de deux relations : MINUS

Union de deux relations : UNION

- L'union de deux relations de même schéma permet de retrouver les enregistrements qui sont dans **au moins une** des deux relations

```
SELECT...
FROM …
WHERE …
UNION
SELECT ...
FROM …
WHERE …
```

- Exemple: réf. et désign. des produits dans PRODUIT ou ANC-PRODUIT

```
SELECT   REFPROD, DESIGN
FROM     PRODUIT
UNION
SELECT   REFPROD, DESIGN
FROM     ANC-PRODUIT
```

	REFPROD	DESIGN
	1	papillotes
	5	bilboquet
	9	crécelle
	7	yoyo

	REFPROD	DESIGN
	1	papillotes
	9	crécelle
	4	sifflet
	5	bilboquet
	2	cerceau

	REFPROD	DESIGN
	1	papillotes
	5	bilboquet
	9	crécelle
	7	yoyo
	4	sifflet
	2	cerceau

Intersection de deux relations : INTER

- L'intersection de deux relations de même schéma permet de retrouver les enregistrements qui sont dans **les deux** relations

```
SELECT...
FROM …
WHERE …
INTER
SELECT ...
FROM …
WHERE …
```

- Exemple: réf. et désign. des produits dans PRODUIT et ANC-PRODUIT

```
SELECT   REFPROD, DESIGN
FROM     PRODUIT
INTER
SELECT   REFPROD, DESIGN
FROM     ANC-PRODUIT
```

	REFPROD	DESIGN
	1	papillotes
	5	bilboquet
	9	crécelle
	7	yoyo

	REFPROD	DESIGN
	1	papillotes
	9	crécelle
	4	sifflet
	5	bilboquet
	2	cerceau

	REFPROD	DESIGN
	1	papillotes
	5	bilboquet
	9	crécelle

Remarque : l'opérateur INTER n'existe pas en ACCESS

Différence de deux relations : MINUS

- La différence de deux relations de même schéma est l'ens. des enreg. qui sont dans la première relation mais pas dans la seconde

```
SELECT...
FROM …
WHERE …
MINUS
SELECT ...
FROM …
WHERE …
```

- Exemple: réf. et désign. des produits dans PRODUIT mais pas dans ANC-PRODUIT

```
SELECT   REFPROD, DESIGN
FROM     PRODUIT
MINUS
SELECT   REFPROD, DESIGN
FROM     ANC-PRODUIT
```

	REFPROD	DESIGN
	1	papillotes
	5	bilboquet
	9	crécelle
	7	yoyo

	REFPROD	DESIGN
	1	papillotes
	9	crécelle
	4	sifflet
	5	bilboquet
	2	cerceau

	REFPROD	DESIGN
	7	yoyo

Remarque : l'opérateur MINUS n'existe pas en ACCESS

Autres opérations de manipulation de données en SQL

- Insertion d'enregistrements : INSERT
- Suppression d'enregistrements : DELETE
- Mise à jour d'enregistrements : UPDATE

Insertion d'enregistrements : INSERT

- Insertion d'enregistrements s'effectue de la manière suivante :

  INSERT INTO ***table*** (***champ***$_1$, … , ***champ***$_n$)
  VALUES (***valeur-champ***$_1$, … , ***valeur-champ***$_n$)

- Exemple : ajouter le produit de ref. n° 19 et de désignation "patins à roulettes" dont le prix HT est 156 F

  INSERT INTO PRODUIT (REFPROD, DESIGN, PRIXHT)
  VALUES (19," patins à roulettes", 156) VALUES)

  ou, lorsque tous les champs sont concernés :

  INSERT INTO PRODUIT
  VALUES (19," patins à roulettes", 156) VALUES)

- Pour insérer des enregistrements provenant d'une requête :

  INSERT INTO PRODUIT
  SELECT ...

Suppression d'enregistrements : DELETE

❖ La suppression d'enregistrements s'effectue de la manière suivante :

```
DELETE
FROM    champ1, … , champn
WHERE   condition
```

❖ Exemple : supprimer toutes les factures émises après le 16/10/2000

```
DELETE
FROM    FACTURE
WHERE   DATFACT > #16/10/2000#
```

- Les enregistrements sont irrémédiablement supprimés
- Vérifier la cohérence des données avec les tables connexes

Mise à jour d'enregistrements : UPDATE

❖ La mise à jour d'enregistrements s'effectue de la manière suivante :

UPDATE ***table***
SET ***champ*$_1$** = ***valeur*$_1$** , … , ***champ*$_n$** = ***valeur*$_n$**
WHERE ***condition***

❖ Exemple : remplacer la référence de produit 4 par 7 dans les factures

```
UPDATE  EST-FACTURE
SET     REFPROD = 7
WHERE   REFPROD = 4
```

- Les enregistrements sont irrémédiablement mis à jour
- Vérifier la cohérence des données avec les tables connexes