Rappels bases de données relationnelles

Mardi 26 Septembre 2006

Baptiste Mougel

UMR TETIS Cemagref-CIRAD-ENGREF

ENGREF

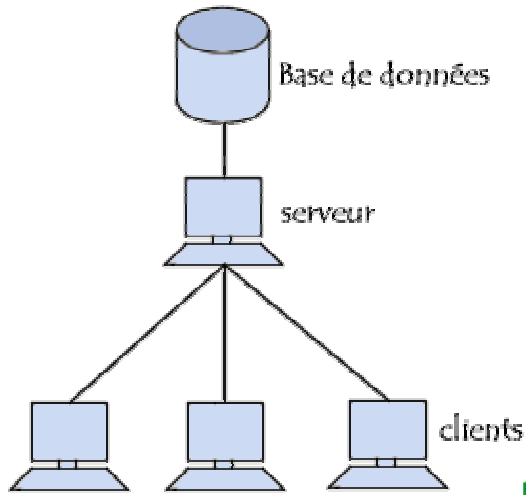
Les Bases de Données - *Introduction*

Définition :

Une base de données, ou *BD*, est un ensemble structuré et organisé permettant le stockage de grandes quantités d'informations afin d'en faciliter l'exploitation.

Utilités :

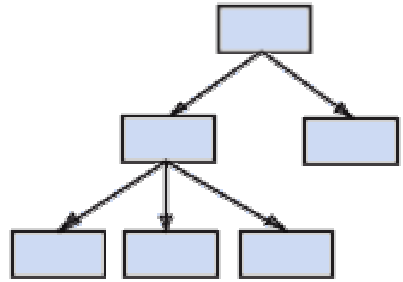
- Mise à disposition de l'information
- Aide à la décision.
- Le stockage d'informations.
- Centralisation de l'information.



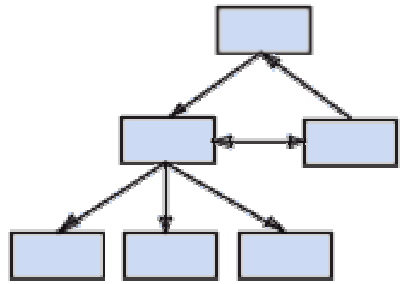
Les Bases de Données – *Les Structures*

Les Types de structures :

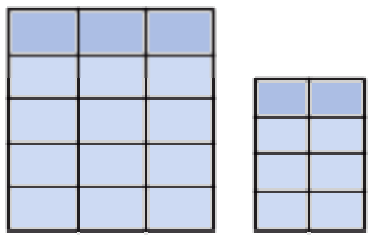
- Bases de données hiérarchiques (Nasa – Programme Apollo)

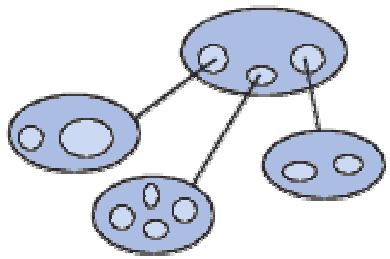


- Bases de données réseaux (Charles. W. Bachman)



- Bases de données orientées objet (ex: LDAP)
- **Bases de données relationnelles**





Les Bases de Données relationnelles – *Les SGBD*

Les *Système de Gestion de Bases de Données (SGBD):*

Outils logiciels permettant la manipulation de BD pour :

- **L'organisation des données de la BD**
  - Création de tables
  - Modifications, suppressions de tables
  - Droits utilisateurs
- **La Mise à jour de la BD**
  - Ajout, modification, suppression d'entrer
- **L'interrogation sur la BD**
  - Requêtes

DE FACON SIMPLE ET CONVIVIALE

Les Bases de Données relationnelles – Choisir *un SGBD*

- Efficacité des accès aux données
- Gestion physique & logique
- Administration centralisée des données
- Non redondance des données
- Cohérence des données
- Partage des données
- Sécurité des données
- Résistance aux pannes

SGBD du marché

Les principaux systèmes de gestion de bases de données :

- Microsoft SQL server (MsSqlSpatial)
- Microsoft Access
- Oracle (OracleSpatial)

- MySQL (Integré)

- PostgreSQL (PostGIS)

Microsoft SQL server

Version actuelle : 2005 SP1

Disponibilité : Windows

Licence : commerciale

Avantages

* Administration aisée
* Une des bases les plus performantes sous Windows
* Frontaux et assistants très poussés
* Services Web
* Support XML

Microsoft SQL server

Inconvénients

* Distributions fortement liées au système d'exploitation
* Mono-plateforme (MS Windows)
* Pas d'intégration Java(orientation C#)
* Fonctionnalités s'éloignant des normes en vigueur

Microsoft Access

Version actuelle : 2003

Disponibilité : Windows

Licence : commerciale

Avantages

* **Nombreux assistants pour l'aide**
* **Un outil grand public**
* **Facilite d'utilisation**

Microsoft Access

Inconvénients

* Format de données mono-fichier
* Mono-plateforme (MS Windows)
* Trafic généré sur le réseau
* Chute de ses performances en utilisation réseau multiposte
* Chute de ses performances en multi-utilisateurs
* limite de la base a 100000 lignes

Oracle

Version actuelle : 10gR2 (10.2.0.2)

Disponibilité : Linux, Windows, Unix

Licence : commerciale, gratuite dans sa version Express

Avantages

* **Assistants performants**
* **Génération des tâches et des alarmes**
* **Pérennité de l'éditeur (40% de part de marché)**
* **Interface utilisateur extrêmement riche**
* **Services Web**
* **Support XML**

Oracle

Inconvénients

* Prix
* Fort demandeur de ressources
* Méta-modèle propriétaire, loin de la norme.
* Nécessite un expert pour la manipulation

MySql

Version actuelle : 5.0.15 (beta 5.1)

Disponibilité : Linux, Windows, MacOS X, Unix, Solaris 10

Licence : GPL et commerciale

Avantages

* Solution très courante en hébergement public
* Très bonne intégration dans l'environnement Apache/PHP
* OpenSource, bien que les critères de licence soient de plus en plus difficiles à supporter
* Facilité de déploiement et de prise en main.
* Plusieurs moteurs de stockage adaptés aux différentes problématiques.

MySql

Inconvénients

* Ne supporte qu'une faible partie des standards SQL-92
* Assez peu de richesse fonctionnelle
* Manque de robustesse avec de fortes volumétries
* Pas d'héritage de tables

PostgreSQL

Version actuelle : 8.1.4

Disponibilité : Linux, Unix, MacOS X, Windows

Licence : BSD

Avantages

* **OpenSource et gratuit**
* **Supporte la majorité du standard SQL-92**
* **Très riche, fiable et relativement performant**
* **Simple d'utilisation et d'administration**
* **Héritage de tables**

PostgreSQL

Inconvénients

* Sauvegardes peu évoluées
* Bases de taille moyenne
* Pas de services Web
* Pas de support XML

Le Modèle Relationnel

&

Langage SQL

Le Modèle Relationnel

- Organisation des données : des **tables à deux dimensions**, encore appelées *relations* et chaque ligne *n-uplet ou tuple*
- Manipulation des données : **algèbre relationnelle**
- Contrainte du modèle : l'intégrité de la base doit être vérifiée d'où la mise en place de règles
- Minimise la redondance : pour cela de nombreuse règles/méthodes existantes (ex: MERISE, Forme Normale, …)

Créer une Base de Données Relationnel

- **Présentation du modéles sur un exemple**

Nom	Nombre d'étoile	Présence d'une piscine	Présence d'internet	Commune	Code Postal
hotel du lion d'or	2	FALSE	TRUE	langnau am albis	XXXXX
chez mimile	1	FALSE	FALSE	aeugst am albis	YYYYY
hotel du grand baron	5	TRUE	TRUE	aeugst am albis	YYYYY
le fendant	3	TRUE	FALSE	hausen am albis	HHHHH
la raclette joyeuse	3	TRUE	FALSE	hausen am albis	HHHHH
la vache mauve	4	TRUE	TRUE	hausen am albis	HHHHH
les hautes cimes	4	FALSE	FALSE	mettmenstetten	XXXXX

Créer une Base de Données Relationnel

Clef étrangère

Nom	Nombre d'étoile	Présence d'une piscine	Présence d'internet	id_commune
hotel du lion d'or	2	FALSE	TRUE	1
chez mimile	1	FALSE	FALSE	2
hotel du grand baron	5	TRUE	TRUE	2
le fendant	3	TRUE	FALSE	3
la raclette joyeuse	3	TRUE	FALSE	3
la vache mauve	4	TRUE	TRUE	3
les hautes cimes	4	FALSE	FALSE	4

Relation

Commune	code postal	parcellaire	id_commune
langnau am albis	xxxxxx	aaaaa	1
aeugst am albis	yyyyyy	bbbb	2
hausen am albis	hhhhh	cccc	3
mettmenstetten	xxxxxx	dddd	4

Clé Primaire

SQL - Généralités

Définition :

Structured Query Language

SQL est un langage normaliser d'interrogation de bases de données.

SQL couvre :
- la création des données,
- la manipulation des données,
- le contrôle des données.

SQL - Historique

- 1974, IBM lance le projet System/R et crée SEQUEL
- 1977, SEQUEL/2
- 1986, normalisation ANSI renome SEQUEL/2 en SQL
- 1987, Normalisation ISO pour SQL
- 1992, La norme SQL 2
- 1999, SQL:1999 (SQL 3)
- 2003, SQL:2003

SQL : Fonctionnalités

- Gestions des insertions, modifications et suppression des données
- Opérateurs arithmétiques et de comparaison
- Affichage des données
- Affectation
- Fonctions d'aggregation

SQL – les opérateurs de base

BASE :

•Manipulation des tables

- •CREATE
- • DROP
- • ALTER

• Manipulation des données

- • INSERT
- • DELETE
- • UPDATE
- • SELECT

• Gestion des privilèges utilisateurs

- • GRANT
- • REVOKE

SQL – Commandes : CREATE

Nom	Nombre d'étoile	Présence d'une piscine	Présence d'internet	Commune
hotel du lion d'or	2	FALSE	TRUE	langnau am albis
chez mimile	1	FALSE	FALSE	aeugst am albis
hotel du grand baron	5	TRUE	TRUE	aeugst am albis
le fendant	3	TRUE	FALSE	hausen am albis
la raclette joyeuse	3	TRUE	FALSE	hausen am albis
la vache mauve	4	TRUE	TRUE	hausen am albis
les hautes cimes	4	FALSE	FALSE	mettmenstetten

Syntaxe :

CREATE TABLE <nom_de_relation> (
 <nom_d'attribut> <type_de_données> [NOT NULL] [, ...]
[, PRIMARY KEY (<nom_d'attribut> [, ...])]
[, FOREIGN KEY (<nom_d'attribut>) REFERENCES <nom_de_relation> [, ...]])
[CHECK (<condition>)]

Exemple :

```
CREATE TABLE hotels(
  Nom varchar(20),   etoile int4,   piscine bool,  internet bool,
  CONSTRAINT hotels_pkey PRIMARY KEY (Nom)
)
```

SQL – Commandes : INSERT

Nom	Nombre d'étoile	Présence d'une piscine	Présence d'internet	Commune
hotel du lion d'or	2	FALSE	TRUE	langnau am albis
chez mimile	1	FALSE	FALSE	aeugst am albis
hotel du grand baron	5	TRUE	TRUE	aeugst am albis
le fendant	3	TRUE	FALSE	hausen am albis
la raclette joyeuse	3	TRUE	FALSE	hausen am albis
la vache mauve	4	TRUE	TRUE	hausen am albis
les hautes cimes	4	FALSE	FALSE	mettmenstetten

Syntaxe :

INSERT INTO <nom_de_relation > [(<liste_d'attributs>)]
VALUES (<liste_de_valeurs >)

ou

INSERT INTO <nom_de_relation> [(<liste_d'attributs>)] <expression_de_sélection>

Exemple :

INSERT INTO hotels ("hotel du lion d'or",2,False,True);

ou

INSERT INTO hotels ('Nom', 'etoile', 'piscine')
VALUES ("hotel du lion d'or",2,False);

SQL - Algèbre relationnelle

T1

A	B
1	3
5	7

T2

C	D
98	74
6	63

SELECT | A,C,D | FROM | T1,T2 | WHERE | B=3 |

Résultat

C	D	A	A
98	74	1	1
6	63	5	5
46	85		7
20	9		5
	5		

SQL – La commande SELECT

SELECTION ET PROJECTION

Nom des hôtels 3 étoile ?

```
•SELECT  nom,
 FROM hotels
 WHERE etoile = 3;
```

*Projection (* = toutes colonnes)*

Sélection selon une clause

Nom
le fendant
la raclette joyeuse

Qu 'elles sont les nom des commune ou il y a un hotel ?

```
• SELECT DISTINCT commune
 FROM hotels
 ORDER BY commune;
```

Résultat avec occurences uniques

Tri

Commune
langnau am albis
aeugst am albis
hausen am albis
mettmenstetten

SQL – La commande SELECT

JOINTURE

Qu 'elle est le nom est adresse des hôtels situer dans une ville de plus de 1000 habitants?

```
SELECT  h.nom, h.adresse, c.code_postal, c.nom
  FROM hotels h, commune c                    Produit cartésien
  WHERE h.id = c.id                           Condition de jointure
  AND c.population > 1000;                    Sélection complémentaire
```

Nom	Adresse	Code Postal	Nom
hotel du lion d'or	Adresse1	XXXXX	langnau am albis
chez mimile	Adresse2	YYYYY	aeugst am albis
otel du grand baror	Adresse3	YYYYY	aeugst am albis
le fendant	Adresse4	HHHHH	hausen am albis
la raclette joyeuse	Adresse5	HHHHH	hausen am albis
la vache mauve	Adresse6	HHHHH	hausen am albis
les hautes cimes	Adresse7	XXXXX	mettmenstetten

SQL – La commande SELECT

AGREGATION

Pour chaque culture on souhaite la surface cultiver et la taille de la plus grand culture

SELECT culture,SUM(surface),MAX(surface) *Fonctions de groupe*

FROM parcelle

GROUP BY culture *Champ de rupture=culture*

Les requêtes imbriquées

IN= Permet de tester la présence d'une valeur particulière dans un ensemble.
SELECT Telephone FROM ABONNE WHERE Nom IN (SELECT Nom FROM AUTEUR)

NOT IN= Permet de tester l'absence d'une valeur particulière dans un ensemble.
SELECT Telephone FROM ABONNE WHERE Nom NOT IN (SELECT Nom FROM AUTEUR)

EXISTS= Retourne "VRAI" si une requête imbriquée retourne au moins une ligne.
SELECT NumAbo FROM ABONNE WHERE NOT EXISTS (PRET)

Les prédicats

BETWEEN= Teste l'appartenance d'une valeur à un intervalle.

SELECT Nom, Prenom FROM PERSONNEL WHERE Salaire BETWEEN 10000 and 12000

LIKE= Permet de faire une recherche “approximative”.

SELECT Nom FROM ABONNE WHERE CodeP LIKE '38---' OR Ville LIKE '%ISERE%'

IS NULL= Permet de tester si un champ a été affecté.

SELECT Nom FROM ABONNE WHERE Telephone IS NULL

Les clauses

GROUP BY = Application de fonction agégats à des collections d'enregistrements reliées sémentiquement.

HAVING = Cette clause ne s'emploie qu'avec un "GROUP BY". Exprime une condition sur le groupe d'enregistrement associé à chaque valeur du groupage.
SELECT NumAbo, count(*) FROM PRET GROUP BY NumAbo HAVING DatePret <= '22/05/00'

ORDER BY = Permet l'ordonnancement du résultat avant l'affichage. (DESC ou ASC)

DISTINCT = Elimine les doublons avant d'utiliser une fonction agrégat.

Les fonctions agrégats :

COUNT = Dénombre les lignes sélectionnées.

SUM = Additionne les valeurs de type numérique.

MIN = Retourne la valeur minimale d'une colonne de type caractère ou numérique.

MAX = Retourne la valeur maximale d'une colonne de type caractère ou numérique.

AVG = Calcule la moyenne d'une colonne de type numérique.

Question ?

Merci de votre Attention