Structure système : Utilisateurs => Réseau => SGBD => BDD

SGBD : Système de gestion de BDD. Gère : déf des données, manip donn, intégrité donn, sécurité donn, concurrence d’accès, résistance pannes, indèp physique et logique.

MERISE : Méthode d’Etude et de Réalisation Informatique pour les Systèmes d’Entreprise. Entités : objet identifiable et nommable : acteur, objet, flux. Attributs : valeur qui qualifie une entité. Identifiants (clés) : propriété qui permet d’identifier une occurrence de l’entité. Associations : lien sémantique reliant des entités. Cardinalités : quantifient le nombre d’occurrences d’une entité.

MCD : Le MCD permet de représenter les données sous forme de schéma. Il faut dans un premier construire les entités (chacune possède une clé qui permet de repérer de façon unique une donnée), puis ajouter les relations qui existent entre les entités. En plus de cela, il convient d’ajouter la cardinalité qui régit les relations.

MLD : Règle 1 : Toute entité est représentée par une relation. Chaque attribut de l'entité devient un attribut de la relation. L'identifiant est conservé en tant que clé de la relation.

Règle 2 : Toute association qui associe plus de deux entités (ternaire et au-delà) est représentée par une relation.

Règle 3 : Toute association binaire dont les cardinalités maximales sont n de chaque côté est une relation (relation dont les attributs sont les attributs clefs des entités qu'elle relie ainsi que les éventuels attributs propres à l’association).

Règle 4 : Une association de type père - fils, cardinalité maximum à n d’un côté et à 1 de l’autre, n’est pas représentée par une relation. On indique les attributs clefs de l’entité père (côté (.,n)) dans le fils (côté (.,1)).

Dépendance fonctionnelle élémentaire : Une DF élémentaire est une DF de la forme X → A où A est un attribut unique n’appartenant pas à X et où il n’existe pas X' inclus au sens strict dans X (i.e. X' Ì X) tel que X' → A.

Dépendance fonctionnelle directe : Une DF X → A est une DF directe s’il n’existe aucun attribut B tel que l’on puisse avoir X → B et B → A.

Norme 1FN : Une relation est en 1FN si, et seulement si, tout attribut contient une valeur atomique (non multiple, non composée).

Norme 2FN : Une relation est en 2FN si, et seulement si elle est en 1FN et si toutes les dépendances fonctionnelles entre la clé et les autres attributs sont élémentaires.

Norme 3FN : Une relation est en 3FN si, et seulement si elle est en 2FN et si toutes les dépendances fonctionnelles entre la clé et les autres attributs sont directes.

Norme BCNF : Une relation est en BCNF si, et seulement si elle est en 3FN et si les seules dépendances fonctionnelles élémentaires sont celles dans lesquelles une clé détermine

un attribut (et non l’inverse).

LDD :langage de déf des données : CREATE, DROP , ALTER. LMD : langage de manip des données : SELECT, UPDATE, INSERT, DELETE . LCD :langage de contrôle des données : GRANT, REVOKE. LCT : langage de contrôle des transactions : COMMIT, SAVEPOINT, ROLLBACK

CREATE TABLE <*nom*>CREATE TABLE personne( ALTER TABLE <nom\_table> DROP TABLE <nom\_table>

(<*définition colonne*> | <*définition* nom VARCHAR2(30), [<add>][<modify>][<drop>]

*contrainte*>,...) age INTEGER,

[<*spécification stockage*>] salaire NUMBER(10,2)

[<*données provenant d’une requête*>] );

**TYPES :** CHAR(n) : chaine à longueur fixe.  VARCHAR2(n) : chaine <255 char. NUMBER(p,s) : nombre décimal. INTEGER : entier long. FLOAT : réel. DATE : date calendrier.

**Contraintes :** Nullité de colonne : NULL / NOT NULL. Unicité de valeur dans une colonne : UNIQUE (colonne1, colonne2, …). Clé primaire : PRIMARY KEY (colonne1, colonne2, …). Vérification sur un ensemble de valeur : CHECK (condition). Clé étrangère : FOREIGN KEY (colonne1, colonne2, …) ; REFERENCES table [(colonne1, colonne2,…)].

CREATE TABLE personne( CREATE TABLE personne(

id NUMBER CONSTRAINT pk\_per PRIMARY KEY, code VARCHAR2(4),

nom VARCHAR2(20), nom VARCHAR2(20),

salaire NUMBER(12,2) CONSTRAINT const\_sal NOT NULL, salaire NUMBER(12,2),

codepost NUMBER(5) CHECK(codepost BETWEEN codepost CHAR(5),

00001 AND 99999) CONSTRAINT pk\_personne PRIMARY KEY (code,nom)

o Insert

INSERT INTO <nom\_table>

VALUES (<expr>[,<expr>...]); // Attention à l’ordre des colonnes

INSERT INTO <nom\_table> (<col1> [, … ])

VALUES (<expr1> [, … ]);

o Update

UPDATE <nom\_table>

SET <col>=<expr>[,<col>=<expr>…]

[WHERE <predicat>]

o Delete

DELETE FROM <tab> [WHERE <predicat>]

o Select

SELECT <col1> [, … ] FROM <tab> [WHERE <predicat>]

o Les vues

Simplicité des requêtes performances : la traduction de la requête peut être longue

Simplicité structurelle restriction des màj : possibilité de màj qu’à partir de vues simples

Isolation des modifs

Intégrité des données

CREATE VIEW <nom\_vue> [(<col1>,<col2>…)] AS

SELECT …

Où le select est le résultat de n’importe quelles opérations définies précédemment. On peut en plus ajouter des vérifications d’intégrité dans le cas où un utilisateur voudrait ajouter une donnée dans cette table sans passer par le select => il pourrait y avoir des données qui ne correspondent pas aux conditions. Pour ce faire, on ajoute après chaque prédicat le mo clé WITH CHECK OPTION CONSTRAINT <Nom\_Contrainte>.

DROP VIEW <nom\_vue>

o Les index

CREATE INDEX nom\_index ON nom\_table

(nom\_col [ASC|DESC], nom\_col…,);

DROP INDEX nom\_ind;

o Les privilèges

GRANT <privilège> ON <table> TO

<utilisateur> [with grant option]

REVOKE <privilège> ON <table> FROM

<utilisateur2>

Les joitures externes en SQL

SELECT ... FROM

<table gauche> LEFT | RIGHT | FULL [OUTER] JOIN

<table droite> Attention : La clause WHERE est une condition sur une table et la clause ON est une condition de jointure.

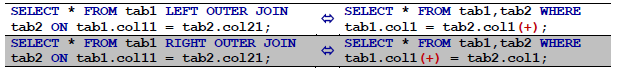
[ LEFT | RIGHT | FULL [OUTER] JOIN Les résultats sont complètements différents. Voici les schéma respectivement des jointures LEFT, RIGHT et FULL :

<table droite> … ]

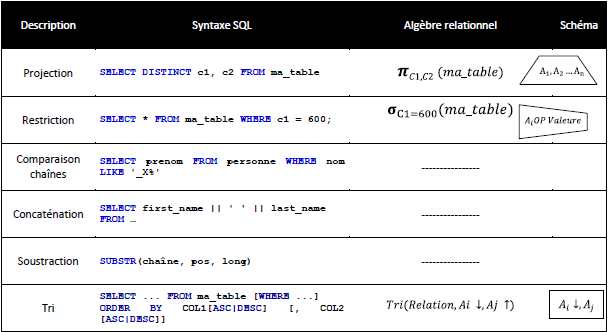
ON 

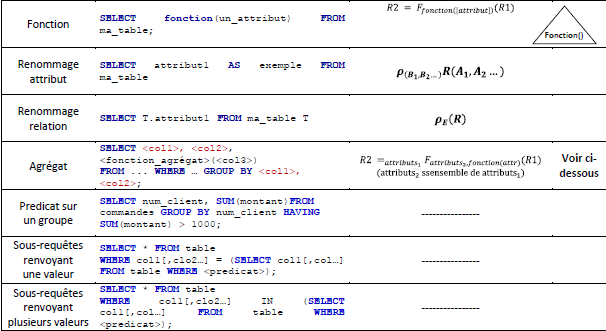
<condition de jointure>

WHERE … ; Remarque : A la place des jointures externes, on peut utiliser les pivots.



Union : R1 U R2 : SELECT <attribut> FROM table UNION SELECT Intersection : R1 Π R2 : SELECT <attribut> FROM table INTERSECT SELECT





Opérations de jointures :

o Voici une liste des opérateurs de comparaison pour le ‘where’ : = , <> , <= , >= , < , >

o Voici une liste des opérateurs logiques pour le ‘where’ : AND , OR , NOT

o Test sur la nullité d’un attribut : IS NULL , IS NOT NULL

o Voici une liste des meta-caractères existants en SQL : \_ (un seul caractère) , % (0 ou n caractères)

o Voici une liste des opérations sur les chaînes de caractère : UPPER(MAJASCULE) , LOWER(minuscule), TO\_NUMBER(Chaîne → nombre), TO\_CHAR(nb,masque)

o Conversion d’une date : TO\_DATE(chaîne,format) , TO\_CHAR(date,format)

o Liste des opérateurs acceptés pour les sous-requêtes renvoyant une valeur : = , <> , <= , >= , < , >

o Liste des opérateurs acceptés pour les sous-requêtes renvoyant plusieurs valeurs : IN , NOT IN , ALL , ANY , = , <> , <= , >= , < , >

Remarque 2 : Lorsque l’on utilise une sous-requête, au lieu de faire une comparaison, on peut faire un test sur la liste : SELECT T1.ID FROM table2 T2 WHERE EXISTS | NOT EXISTS (SELECT T2.ID FROM table2 T2 WHERE T2.ID = T1.ID)

Transactions : Une transaction est un ensemble ordonné d’opérations modifiant des données (LMD) qu’un SGBD effectuera parfaitement et complètement ou pas du tout. Une opération d’une transaction est donc auchoix de tupe INSERT, UPDATE ou DELETE.

Une transaction vérifie les 4 Ptés suivantes :

Atomicité : une transaction doit être complètement validée (COMMIT) ou complètement annulée (ROLLBACK)

Cohérence : une transaction ne peut pas laisser la base de données dans un état incohérent.(Si une panne se produit pendant la validation ou l’annulation d’une transaction, lors du redémarrage la fin de la transaction est effectuée si cela est possible, ou la BDD est remise dans l’état avant la transaction).

Isolation : une transaction ne peut voir aucune autre transaction en cours d’exécution. Lors de l’exécution d’un ordre select**, le SGBD affiche** : toutes les données déjà validées lors de la transaction précédente, les données créées ou màj dans la transaction courante, les données détruites lors d’autres transactions en cours. **Le SGBD n’affiche pas** : les données détruites définitivement lors de transactions validées, les données détruites lors de la transaction courante, les données créées ou màj dans les autres transactions en cours.

Durabilité : après que le client a été informé du succès de la transaction, les résultats de celle-ci ne disparaitront pas.

Découpage d’une transaction : On peut poser des jalons : savepoint nom\_point Ont peut annuler un ordre de sous ensemble avec rollback to savepoint nom\_point. Toutes les opérations créées depuis le jalon sont annulées. Un jalon ne disparait quà la fin de la transaction. Chaque màj d’une ligne d’une table provoque le verrouillage de la ligne.

Règles de restructuration :

1. Commutativité des jointures et Associativité des jointures

2. Fusion des projections

3. Regroupement des restrictions

4. Quasi-commutativité des restrictions et projections

5. Quasi-commutativité des restrictions et jointures

6. Commutativité des restrictions et des unions, intersections ou différences.

7. Quasi-commutativité des projections et des jointures

8. Commutativité des projections avec les unions.