

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU



FACULTE DU GENIE ELECTRIQUE ET D'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

Mémoire de Fin d'Etudes de MASTER PROFESSIONNEL

Domaine : **Mathématiques et Informatique**

Filière : **Informatique**

Spécialité : **Ingénierie des Systèmes d'Information
(ISI)**

Présenté par
CHILI Cherifa
AKKOUCHE Malika

Thème

**Implémentation d'une solution de fédération des
bases de données homogènes.**

Cas : La scolarité de la faculté GEI

Mémoire soutenu publiquement le 27 / 09 / 2016. Devant le jury composé de :

Président : M^r AMIROUCHE M.Nabil.

Encadreur : M^{me} AMIROUCHE Fatiha.

Examineur : M^{me} BELKACEMI Lila.

Examineur : M^{elle} ILLTACHE.S

Promotion 2015/2016



Remerciements

*D'abord je tiens à remercier **ALLAH** de m'avoir accordé la force et le courage de mener à terme ce projet*

Toute notre reconnaissance et gratitude vont vers nos professeurs qui nous ont appris le savoir.

*On tiens à remercier sincèrement notre promotrice Madame **Amirouche.F** qui s'est toujours montrée à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'elle a bien voulu nous consacrer.*

On remercie les membres du jury d'avoir accepté de juger notre travail.

On exprime notre gratitude à tous les consultants rencontrés lors des recherches effectuées et qui ont accepté de répondre à nos questions avec gentillesse.

On n'oublie pas nos parents pour leur contribution, leur soutien et leur patience.

A tous nos proches qui ont toujours eu la patience de nous entendre et de nous redonner confiance et qui ont compris l'importance d'un tel travail.

Malika & Cherifa



Dédicaces

Je dédie ce modeste travail :

*À ma très chère maman : Akliouat Fatma,
Mon cher papa : Saïd, Témoignage d'affection et de grande
reconnaissance, Que Dieu les garde.*

Mon frère chouchouté : Nassim.

À la mémoire de mes grands (es) parents.

À la mémoire de mon cher oncle « Vava Mouh ».

À tous mes cousins, cousines, tantes, oncles.

*À mes chères cousines : Nacera, Fadhila,
Hamida, Sihem, Sarah.*

*À mes chères amis (es) : Kahina, Sabrina, Rafik, Lilia,
hayet, Hamid, Agniles, Saïd.*

À tous ceux qui m'aiment et que j'aime.

*À tous ceux qui ont participé de près ou de loin à la
réalisation de ce travail.*

À mon binôme Malika et à toute sa famille.

À la section MII ISI promotion 2015/2016

Cherifa



Dédicaces

Je dédie ce modeste travail à :

Mes très chers, magnifiques parents,

Que Dieu les protèges.

Mes deux adorable sœurs Ouiza , Karima.

Mon adorable frère Mohamed.

Mes nièces Ryma et Numedia et mon neveu Walid

Tous mes cousins et cousines.

Mon binôme Cherifa et à toute sa famille.

Tous ceux qui m'aiment.

Tous ceux que j'aime.

Tous mes camarades de département informatique.

Malika

Résumé

Résumé :

Résumé:

Une base de données fédérée est une collection de données hétérogènes ou homogènes capable d'inter opérer via une vue commune. L'objectif de notre travail est de remédier aux problèmes d'intégration des données entre les différents départements de la faculté de Génie Electrique et Informatique(GEI) et de donner un accès transparent au vice doyen de celle-ci afin de pouvoir bien gérer son organisation.

Ce mémoire comprend trois chapitres, le premier est purement théorique comprenant des définitions et des généralités sur les bases de données. Elle décrit aussi, les différentes méthodes d'intégration de données.

Le second chapitre est consacré pour la conception et la mise en œuvre d'une BD fédérée pour la gestion de la scolarité de la faculté GEI implémentée sous le SGBD *ORACLE 10g* version Express Edition.

Le troisième chapitre dont on présente les outils de développement de notre projet ainsi quelque interface de notre application.

On fini par une conclusion générale et quelques perspectives afin d'améliorer notre système.

Mots clé : Bases de donnée, bases de données fédérée, systèmes d'intégration, vue matérialisée, oracle 10g, SQL plus, liens base de donnée.

Abstract:

A federated database is a collection of heterogeneous or homogeneous data that can interoperate using a common view. The aim of our work is to give a solution of integration data problems exist between different departments of the (GEI) faculty, and provide seamless access to the vice dean of the latter in order to manage well its organization.

This memory consists of three chapters; the first is purely theoretical including definitions and general information on databases. It also describes the various data integration methods.

The second chapter is devoted to the design and implementation of a federated database for the management of schooling IEG faculty implemented under *ORACLE 10g* DBMS version Express Edition.

The third chapter which presents our project development tools and some interface of our application.

We ended up a general conclusion and some perspectives to improve our system.

Keywords: databases, federated databases, integration system, materialized view, oracle 10g, SQL plus, database link.

sommaire

Sommaire :

Remerciement:	
Dédécaces:	
Résumé:	
Introduction générale:.....	

Chapitre I : Les bases de données fédérées

Introduction :	1
I. Généralités sur les bases de données :.....	1
I.1 Bases de données :	1
I.1.1 Définition:	1
I.1.2 Objectifs des bases de données :.....	2
I.2 Les systèmes de gestion des bases de données (SGBD) :	2
I.2.1 Définition :	2
I.2.2 Objectifs des SGBDs :	2
I.2.3 Fonctionnalités des SGBDs :	3
I.2.4 Les SGBDs Oracle versus mysql :	3
I.3 Conception d'une base de données :	4
I.3.1 niveau conceptuel :	4
I.3.2 Niveau interne :	4
I.3.3 Niveau externe :	5
I.4 Classification des bases de données :	5
I.4.1 Classification selon le nombre d'utilisateurs :	5
I.4.2 Classification selon l'usage :	5
I.4.3 Classification selon l'emplacement :	6
I.5 Conception des bases de données réparties :	6

Sommaire :

I.5.1	Objectif :.....	6
I.5.2	La fragmentation :	6
I.5.3	Les approches de conception :.....	7
II.	Fédération des bases de données	9
II.1	Définition des bases de données fédérées (Federated DataBase) :	9
II.2	Objectif de la fédération des bases de données:.....	9
II.3	Problématique de l'intégration des données:.....	10
II.3.1	Les types de données :.....	10
II.3.2	Conflits de modèles de données :	10
II.4	Intégration des bases de données :.....	11
II.4.1	Définition :	11
II.4.2	Objectifs d'Intégration :	11
II.4.3	Processus d'intégration :	12
II.5	Classification des systèmes d'intégration :.....	12
II.5.1	Intégration matérialisée (les entrepôts de données) :.....	13
II.5.2	Intégration virtuelle (les médiateurs) :.....	14
II.5.3	Les vues :	17
II.6	Evaluation et optimisation des requêtes :.....	19
	Conclusion :	20

Chapitre II : Cas d'étude

	Introduction :	21
I.	Présentation du domaine d'étude :.....	21
I.1.	Présentation de l'organisme d'accueil :.....	21
I.1.1.	Présentation de la faculté de GEI :	21
I.1.2.	Présentation des départements de la faculté GEI :	22

Sommaire :

I.1.3. Les taches de la scolarité :	23
I.2. Planification du projet:	24
I.2.1. Problématique:	24
I.2.2. Objectifs de l'étude:	25
I.2.3. Solution proposée :	25
I.2.4. Justification du choix :	26
II. Analyse et conception :	26
II.1. Analyse	26
II.1.1. Description du sujet à étudier	27
II.1.2. Identification des acteurs	27
II.1.3. Présentation des bases de données locales.....	30
II.1.4. Démarche d'intégration suivie	30
II.2. Conception :	32
II.2.1. conception de la base de données fédérée:.....	32
II.2.2. Diagramme du schéma logique de la base de données de chaque département	32
II.2.3. Diagramme du schéma logique de la base de données fédérée	34
II.2.4. Présentation des vues matérialisées de la base de données fédérée	35
II.2.5. Création des liens.....	35
II.2.6. Les vues matérialisées créées.....	35
II.2.7. Conception de l'interface utilisateur/BDDF	36
Conclusion	37

Chapitre III : Implémentation et réalisation

Introduction :	38
I. Environnement de développement	38
I.1. Environnement logiciels	38

Sommaire :

I.1.1. Eclipse IDE.....	38
I.1.2. Oracle Database 10g express Edition	38
I.2. Les langages de programmation	38
I.2.1.Java.....	39
I.2.2.SQL.....	40
II. Implémentation de la base de données fédérée	41
II.1.Création des liens.....	42
II.2.Les vues matérialisées	44
III.Présentation de quelques interfaces	45
Conclusion	46
Conclusion générale.....	
Annexe	
Bibliographie et webiographie.....	

Liste des figures

Liste des figures

Chapitre I : Les bases de données fédérées

<i>Figure I.1 : Les bases de données.....</i>	<i>1</i>
<i>Figure I.2: L'approche descendante.....</i>	<i>8</i>
<i>Figure I.3: L'approche ascendante.....</i>	<i>9</i>
<i>Figure I.4 : processus d'intégration.....</i>	<i>12</i>
<i>Figure I.5: Approche entrepôt de données.....</i>	<i>13</i>
<i>Figure I.6 : Approche médiateur</i>	<i>15</i>
<i>Figure I.7: L'approche GAV</i>	<i>16</i>
<i>Figure I.8: L'approche Local As View (LAV).....</i>	<i>17</i>
<i>Figure I.9 : Utilisation des vues</i>	<i>18</i>
<i>Figure I.10: Utilisation des vues matérialisées.....</i>	<i>19</i>

Chapitre II : Cas d'étude

<i>Figure II.1 : Organigramme de la faculté génie électrique et informatique [2.1].</i>	<i>22</i>
<i>Figure II.2: Exemple d'organigramme d'un département de la faculté GEI [2].</i>	<i>23</i>
<i>Figure II.3: Architecture actuelle des flux d'information dans la faculté GEI</i>	<i>24</i>
<i>Figure II.4: Architecture des flux d'information dans la faculté par notre solution.....</i>	<i>26</i>
<i>Figure II.5 : Plan de représentation des schémas conceptuels.....</i>	<i>32</i>
<i>Figure II.6: Diagramme de schéma logique de la base de données Informatique</i>	<i>32</i>
<i>Figure II.7: Diagramme de schéma logique de la base de données Automatique.....</i>	<i>33</i>
<i>Figure II.8: Diagramme de schéma logique de la base de données Electronique</i>	<i>33</i>
<i>Figure II.9 : Diagramme de schéma logique de la base de données Electrotechnique</i>	<i>34</i>
<i>Figure II.10: Diagramme du schéma logique de la base de données fédérée.</i>	<i>34</i>
<i>Figure II.11: Exemple de vue matérialisée cas "vue_étudiant"</i>	<i>36</i>
<i>Figure II.12: Diagramme de contexte.....</i>	<i>36</i>
<i>Figure II.13: Diagramme de cas d'utilisation.....</i>	<i>37</i>

Liste des figures

Chapitre III : Implémentation & Réalisation

<i>Figure III.1 : L'interface de l'IDE Eclipse.....</i>	<i>37</i>
<i>Figure III.2: L'interface Oracle Express Edition 10g.....</i>	<i>37</i>
<i>Figure III.3: Exemple d'un programme java</i>	<i>39</i>
<i>FigureIII.4 :Interface de l'invite de commande SQL.....</i>	<i>40</i>
<i>Figure III.5:Exemple de lien de la base de données</i>	<i>40</i>
<i>Figure III.6:Requête de création de lien interne sous oracle</i>	<i>41</i>
<i>Figure III.7:Requête d'interogation de lien interne sous oracle</i>	<i>41</i>
<i>Figure III.8:Requête de création de lien externe sous oracle.....</i>	<i>42</i>
<i>Figure III.9:Requête d'interogation de lien externe sous oracle.....</i>	<i>42</i>
<i>Figure III.10: :Requête de création de la vue matérialisée</i>	<i>43</i>
<i>Figure III.11: Requête d'interogation de la vue matérialisée.....</i>	<i>43</i>
<i>Figure III.10: Interface de la page d'authentification</i>	<i>44</i>
<i>Figure III.11: Interface de la page d'accueil</i>	<i>44</i>
<i>Figure III.12: Interface de la page de recherche avancée</i>	<i>45</i>

Introduction générale

Introduction générale

Introduction générale

Depuis des années le monde de l'informatique se développe, alors que son but initial, au début était d'offrir des services satisfaisants, du point de vue vitesse d'exécution des tâches et obtention de statistiques plus précises. Les bases de données ont leur part de ce développement pour répondre toujours mieux aux problématiques de stockage, de normalisation, de temps de réponses et d'exécution. Actuellement, de nouveaux besoins sont apparus, toute organisation automatisée souhaite un accès transparent aux différentes bases de données qui sont géographiquement éloignées ou non, ce qui rend la tâche de la collecte et de traitement d'une grande quantité d'informations dispersées très délicate, de ce fait, la gestion des bases de données et du système d'information est devenue une priorité pour les gérants des entreprises.

La faculté GEI (Génie Electrique et Informatique) parmi les organisations qui souhaite gérer ses données dispersés sur ses quatre départements d'une façon efficace et souple.

La solution qui s'impose est la mise en place d'une base de données fédérée des différents sites de la faculté GEI (Génie Electrique et Informatique) au niveau du bureau de vice doyen, dans laquelle on implémentera les vues matérialisées sous oracle 10g.

L'objectif de ce travail est de donner au vice doyen un accès transparent aux différentes bases de données de la faculté GEI à travers un réseau d'ordinateurs. La base de données fédérée créée au niveau du bureau du vice doyen constitue la fenêtre d'accès aux différentes bases des départements de la faculté offrant une manipulation transparente pour celui-ci.

Introduction générale

Notre mémoire est structuré en trois (3) chapitres comme suit :

Dans le premier chapitre, nous présentons des généralités sur les bases de données et les bases de données fédérées, ainsi les méthodes d'intégration de ces dernières. Nous illustrons aussi leurs avantages et inconvénients, et les principes de leurs mises en œuvre.

Le second chapitre est composé de deux parties une qui commence par une présentation de l'organisme d'accueil de la scolarité de la faculté, ainsi les problèmes rencontrés, dans la deuxième partie on exposera une analyse et une conception au niveau de fédération des données et au niveau application.

Le troisième chapitre donne nous détaillons la réalisation de la solution proposée. Il présente la mise en place du nouveau système à l'aide de différents outils tels que « ORACLE ».

Nous finirons par une conclusion et quelques perspectives qui peuvent être envisagées afin d'améliorer notre système.

Chapitre I:

Les bases de données fédérées

Introduction :

De nos jours, le développement d'une nouvelle application de traitement de données fait le plus souvent appel à des données déjà mémorisées dans l'ordinateur, que ce soit dans des fichiers, ou dans une, voire plusieurs bases de données indépendantes. C'est le cas, notamment, des grandes entreprises où l'usage largement répandu de l'informatique se traduit par un développement indépendant de plusieurs bases de données pour les applications spécifiques de chaque service ou filiale. Or, les structures des entreprises évoluent, surtout dans l'environnement économique actuel. Les frontières entre services ou entre sociétés sont alors susceptibles de bouger, créant de nouveaux centres d'intérêts, demandant de nouvelles applications qui devront être construites avec des données prises ici et là, plus qu'avec de nouvelles données spécifiques. Ainsi, le développement des nouveaux systèmes d'informations repose désormais sur la capacité du système informatique de réaliser l'interopérabilité entre bases de données existantes.

I. Généralités sur les bases de données :

I.1 Bases de données :

I.1.1 Définition:

Une Base de données est un ensemble de données modélisant les objets d'une partie du monde réel et servant de support à une application informatique.

Une Base de donnée peut être aussi pour certains, une collection de fichiers reliés par des pointeurs multiples, aussi cohérents entre eux que possible, organisés de manière à répondre efficacement à une grande variété de questions.

Une Base de données est donc un ensemble structure des données accessibles par l'ordinateur pour satisfaire plusieurs utilisateurs simultanément au temps opportun.



FigureI.1: Les bases de données.

I.1.2 Objectifs des bases de données :

Une Base de données a beaucoup d'objectifs parmi lesquels nous pouvons citer :

- ✓ **Exhaustivité** : implique que l'on dispose de toutes les informations relatives au sujet donné.
- ✓ **Eviter la redondance et les incohérences des données**: implique l'unicité des informations dans la base de données. En général on essaie d'éviter la duplication des données car cela pose des problèmes de cohérence lors des mises à jour de ces données.
- ✓ **Offrir un langage de haut niveau** pour la définition et la manipulation des données.
- ✓ **Contrôler l'intégrité** entre plusieurs utilisateurs et la confidentialité des données.
- ✓ **Assurer l'indépendance** entre les données et les traitements.

I.2 Les systèmes de gestion des bases de données (SGBD) :

I.2.1 Définition :

Un système de gestion de base de données (SGBD ou DBMS : *DataBases Management System*), est un ensemble de logiciel/matériel qui permet la gestion, manipulation, création, maintenance des données et l'optimisation des performances.

I.2.2 Objectifs des SGBDs :

Les mécanismes du système de gestion de base de données visent à assurer :

- ✓ **Indépendance physique** : un SGBD offre la facilité de changer le schéma interne sans changer le programme d'application. Cela signifie que le niveau physique peut être modifié indépendamment du niveau conceptuel.
- ✓ **Indépendance logique** : ici le SGBD permet de modifier schéma conceptuel sans changer le programme d'application. Donc le niveau conceptuel peut être modifié sans remettre en cause le niveau physique.
- ✓ **La manipulation de données par des langages non procéduraux** : le SGBD doit permettre interrogation et la mise à jour de données par des langages de haut niveau spécifiant les données que l'on veut traiter (de quoi) et non pas comment y accéder.
- ✓ **La facilité d'administration** : les langages de haut niveau référencent des descriptions logiques des données (schéma externes) stockées dans le dictionnaire de données pour permettre la création et modification de la description.

- ✓ **Fiabilité des données** : le SGBD permet de vérifier les contraintes des données (intégrité référentielle, reflexes, etc.) ; gérer des transactions (atomicité des transactions) et sécurité (mot de passe, etc.) ; récupérer des données en cas de crash logiciel, OS (Operating System) ou disque.

I.2.3 Fonctionnalités des SGBDs :

Un SGBD assure plusieurs fonctionnalités qui sont :

- ✓ La description des données de bases en utilisant un langage de description de données (LDD).
- ✓ L'interrogation des données à l'aide d'un langage d'interrogation (LID).
- ✓ La mise à jour, la transformation des représentations avec un langage de manipulation des données (LMD).
- ✓ L'assurance du contrôle d'intégrité, d'occurrence et de sécurité avec le langage de contrôle de données (LCD)

Ces fonctions sont prises en charge par un informaticien appelé administrateur de base de données, parfois DBA (*DataBase Administrator*), c'est le gestionnaire responsable du bon fonctionnement de SGBD et de la mise en œuvre de la bases de données.

I.2.4 Les SGBDs Oracle versus mysql :

De nombreux SGBD sont disponibles sur le marché, partant des SGBD gratuits jusqu'aux SGBD destinés spécialement aux professionnels, comportant de plus nombreuses fonctionnalités, mais plus coûteux. Il existe bien entendu plusieurs SGBDs on s'intéresse à faire une comparaison entre Oracle et Mysql server comme le tableau ci-dessus le montre :

Chapitre I : Les bases de données fédérées

Avantages/ inconvénients	Mysql	Oracle
Avantages	➤ Solution très courante en hébergement public. ➤ Très bonne intégration dans l'environnement Apache/PHP Open Source, bien que les critères de licence soient de plus en plus difficiles à supporter. ➤ Version cluster depuis la version 4. ➤ ordonnanceur et partitionnement dès la version 5.1. ➤ Facilité de déploiement et de prise en main. ➤ Plusieurs moteurs de stockage adaptés aux différentes problématiques, configurable au niveau table. ➤ Multi plate-forme Linux, Windows, OSX, Unix...ect.	➤ Richesse fonctionnelle. ➤ Fonction d'audit évolué. ➤ Row Level Storage Security (RLSS) : permet de ne faire apparaître que certaines lignes des tables pour un utilisateur/une application donnée. ➤ Parallélisme, caches nommés, haute disponibilité. ➤ Procédures stockées en PL-Sql (langage propriétaire Oracle, orienté ADA) ou... en JAVA (depuis la version 8.1.7) ce qui peut s'avérer utile pour les équipes de développement. ➤ Assistants performants via Oracle Manager Server, possibilité de gérer en interne des tâches et des alarmes. ➤ Gestion centralisée de plusieurs instances. ➤ Concept unique de retour arrière (Flashback). ➤ Pérennité de l'éditeur : avec plus de 40 ➤ Réglages fins : dans la mesure où l'on connaît suffisamment le moteur, presque tout est paramétrable. ➤ Accès aux données système via des vues, bien plus aisément manipulable que des procédures stockées. ➤ Services Web, support XML ➤ Ordonnanceur intégré ➤ Compression des données et des sauvegardes Support technique Orion extrêmement riche et fourni.
Inconvénients	➤ Ne supporte qu'une faible partie des standards SQL-92. ➤ Support incomplet des triggers et procédures stockées. ➤ Assez peu de richesse fonctionnelle. ➤ Manque de robustesse avec de fortes volumétries. ➤ Pas d'héritage de table. ➤ Pas de vue matérialisée. ➤ Cluster par clonage de base : impact prépondérant sur la volumétrie.	➤ Prix élevé, tant au point de vue des licences que des composants matériels (RAM,CPU) à fournir pour de bonnes performances. ➤ Administration complexe, liée à la richesse fonctionnelle. ➤ Fort demandeur de ressources, ce qui n'arrange rien au point précité, Oracle est bien plus gourmand en ressource mémoire que ses concurrents, ce qui implique un investissement matériel non négligeable. ➤ Méta-modèle propriétaire, loin de la norme. ➤ Gestion des verrous mortels mal conçue (suppression d'une commande bloquante sans rollback).

Tableau 1.1 : SQL versus Oracle

I.3 Conception d'une base de données :

Pour simplifier la vision des utilisateurs, les SGBD assurent une abstraction des données stockées sur disque en distinguant trois niveaux de description de données qui sont :

I.3.1 niveau conceptuel :

C'est le niveau central correspondant à la structure canonique de données qui existent dans l'entreprise autrement dit leur structure sémantique inerrance sans souci d'implémentation en machine, représentant la vue intégrée de tous les utilisateurs. La définition du schéma conceptuel d'une entreprise nécessite un accord sur les concepts de base que modélisent les données.

I.3.2 Niveau interne :

Ce niveau correspond à la structure de stockage du schéma interne nécessite au préalable le choix d'un SGBD. Elle permet donc de décrire les données telles qu'elles sont stockées dans la machine.

I.3.3 Niveau externe :

Au niveau externe, chaque groupe de travail utilisant des données possède une description des données perçues appelée schéma externe. Cette description est effectuée selon la manière dont le groupe voit la base dans ses programmes d'application. Alors qu'au niveau conceptuel et interne les schémas décrivent toute une base de données, au niveau externe ils décrivent simplement la partie de données présentant un intérêt pour l'utilisateur ou un groupe d'utilisateurs. En conséquence, le schéma externe est souvent qualifié de vue externe. Le modèle externe utilisé est dépendant du langage de manipulation de la base de données.

I.4 Classification des bases de données :

I.4.1 Classification selon le nombre d'utilisateurs :

➤ Base de données mono utilisateur :

Un seul, un petit groupe, voire une entreprise. Une base de données de bureau est installée sur un ordinateur personnel au service d'un seul usager.

➤ **Base de données multi utilisateurs :**

Une base de données d'entreprise est installée sur un ordinateur puissant au service de centaines d'utilisateurs.

I.4.2 Classification selon l'usage :

➤ **Base de données opérationnelles OLTP :**

Les bases de données dites opérationnelles ou OLTP (OnLine Transaction Processing) sont destinées à assister des usagers à tenir l'état d'activités quotidiennes. Elles permettent en particulier de stocker sur le champ les informations relatives à chaque opération effectuée dans le cadre de l'activité: achats, ventes, réservations, paiements. Dans de telles applications l'accent est mis sur la vitesse de réponse et la capacité de traiter plusieurs opérations simultanément.

➤ **Base de données d'analyse OLAP :**

Les bases de données d'analyse dites aussi OLAP (OnLine Analytical Processing), sont composées d'informations historiques telles que des mesures sur lesquelles sont effectuées des opérations massives en vue d'obtenir des statistiques et des prévisions. Les bases de données sont souvent des entrepôts de données (anglais datawarehouse): des bases de données utilisées pour collecter des énormes quantités de données historiques de manière quotidienne depuis une base de données opérationnelle. Le contenu de la base de données est utilisé pour effectuer des analyses d'évolution temporelle et des statistiques telles que celles utilisées en management. Dans de telles applications l'accent est mis sur la capacité d'effectuer des traitements très complexes et le logiciel moteur (leSGBD) est essentiellement un moteur d'analyse.

I.4.3 Classification selon l'emplacement :

➤ **Bases de données centralisées:**

Une base de données centralisée est une base de données gérée par un seul SGBD et stockée dans sa totalité à un emplacement physique unique. Ses divers traitements sont confiés à une seule unité de traitement.

➤ Bases de données réparties :

Une base de données répartie (BDR) est un ensemble structuré et cohérent de données, stocké sur des processeurs distincts, et géré par un système de gestion de bases de données réparties (SGBDR).

I.5 Conception des bases de données réparties :

La définition du schéma de répartition est la partie la plus délicate de la phase de conception d'une BDR, car il n'existe pas de méthode miracle pour trouver la solution optimale.

I.5.1 Objectif :

L'objectif des bases de données réparties est de minimiser le nombre de transferts entre sites, les temps de transfert, le volume de données transférées, les temps moyens de traitement des requêtes, et le nombre de copies de fragments.

I.5.2 La fragmentation :

La fragmentation est le processus de décomposition d'une base de données en un ensemble de sous bases de données. Cette décomposition doit être sans perte d'information.

I.5.2.1 Types de fragmentations :

➤ La fragmentation horizontale :

C'est un découpage d'une table en sous tables par utilisation de prédicats permettant de sélectionner les lignes appartenant à chaque fragment. L'opération de fragmentation est obtenue grâce à la sélection des tuples d'une table selon un ou des critères bien précis et la reconstitution de la relation initiale se fait grâce à l'union (U) des sous-relations.

➤ La fragmentation verticale :

Elle est le découpage d'une table en sous tables par projection permettant de sélectionner les colonnes composant chaque fragment. La relation initiale doit pouvoir être recomposée par la jointure des fragments.

➤ La fragmentation hybride (mixte) :

Elle résulte de l'application successive d'opérations de fragmentation horizontale et verticale sur une relation globale.

❖ L'allocation des fragments :

Suite à la fragmentation des données, il est nécessaire de les placer sur les différentes machines. Un schéma doit être élaboré afin de déterminer la localisation de chaque fragment et sa position dans le schéma global, c'est ce qu'on appelle l'allocation.

L'affectation des fragments sur les sites est décidée en fonction de l'origine prévue des requêtes qui ont servi à la fragmentation. Le but est de placer les fragments sur les sites où ils sont les plus utilisés, et ce pour minimiser les transferts de données entre les sites.

I.5.3 Les approches de conception :

Deux approches fondamentales sont à l'origine de la conception des bases de données réparties : la conception descendante '*Top down design*' et la conception ascendante '*Bottom up design*'.

I.5.3.1 L'approche descendante :

On commence par définir un schéma conceptuel global de la base de données répartie, puis on le distribue sur les différents sites en des schémas conceptuels locaux. La répartition se fait donc en deux étapes, en première étape la fragmentation et en deuxième étape l'allocation de ces fragments aux sites.

L'approche top down est intéressante quand on part du néant. Si les BDs existent déjà, la méthode bottom up est utilisée.

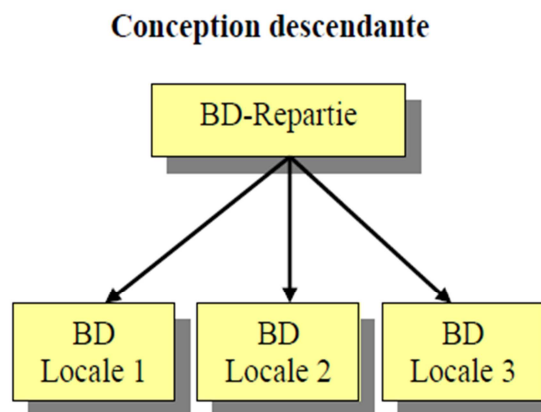


Figure I.2 : L'approche descendante

I.5.3.2 L'approche ascendante:

Cette approche se base sur le fait que la répartition est déjà faite, mais il faut réussir à intégrer les différentes BDs existantes en une seule BD globale. En d'autre terme, les schémas conceptuels locaux (*LCS*) existent et il faut réussir à les unifier dans un schéma conceptuel global (*GCS*).

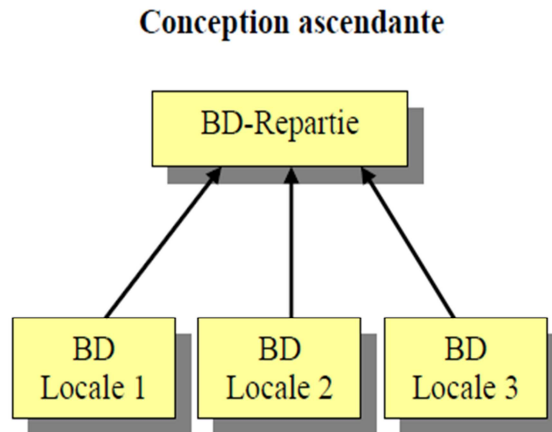


Figure I.3 : L'approche ascendante

II. Fédération des bases de données

II.1 Définition des bases de données fédérées (*Federated DataBase*) :

Une base de données fédérée est une collection de données **hétérogènes** ou **homogènes** capables d'inter opérer via une vue commune, ou les applications voient la donnée d'une manière unifiée, sans avoir recours à la formulation des requêtes multiples, et la combinaison des résultats, elle utilise l'approche ascendante.

D'une manière générale des systèmes sont dits **homogènes** s'ils ont le même modèle de données (modèle Relationnel, modèle réseau, modèle orienté objet, ...). Le même logiciel qui gère toutes les données sur tous les sites.

D'autre part, des systèmes sont dits **hétérogènes** s'ils sont différents soit dans les modèles de données, les formats, logiciels gérant les données, ou les langages d'interrogation.

✓ **Remarque :**

L'hétérogénéité est indépendante de la distribution physique des données à travers un réseau, car on peut avoir des données distribuées à travers des postes différents d'un réseau mais qui sont homogènes (c.à.d. un système distribué mais homogène). De même on peut

avoir des données stockées sur le même poste mais qui sont hétérogène (système centralisé mais hétérogène).

Le logiciel chargé de fournir le contrôle, la manipulation, et la coordination des différents composants d'une base de données fédérée est appelé un système de gestion de base de données fédérée (SGBDF).

II.2 Objectifs de la fédération des bases de données:

Les objectifs principaux des bases de données fédérées sont :

- Donner aux utilisateurs une vue unique des données implémentées sur plusieurs systèmes à priori hétérogènes (en plate-forme et SGBD).
- Faire cohabiter les différents systèmes d'une entreprise tout en leur permettant d'inter opérer.

II.3 Problématique de l'intégration des données:

L'hétérogénéité sémantique et syntaxique des sources de données, l'intégration des schémas locaux pour créer un schéma global causent des conflits, ou se situe cette hétérogénéité de source de données, et quels sont les conflits qu'elle engendre ?

II.3.1 Les sources de données :

Généralement les données manipulées par l'utilisateur proviennent de diverses sources de données (générateurs de données) et qui ont des formats différents :

- ✓ **Données structurées :**
- ✓ **Données semi structurées :**
- ✓ **Données non structurées :**

II.3.2 Conflits de modèles de données :

Ils sont dus à la diversité des modèles utilisés dans la conception des différentes bases (relationnel, relationnel étendu, objet, relationnel objet).

II.3.2.1 Conflits de schémas :

Ils regroupent :

- ✓ **Conflits structurels :** apparaissent lorsque le même élément du monde réel est perçu avec des structures différentes d'une base à une autre.
- ✓ **Conflits de types de données :** apparaissent lorsque deux bases de données utilisent deux types différents pour le même concept du monde réel.

Une solution pour ce type de conflits consiste à effectuer des conversions de type.

- ✓ **Conflits d'organisation et de description des données** : sous forme de classes et de relations.
- ✓ **Conflits de généralisation/spécialisation** : base sur la notion d'une relation généralisée ou spécialisée.
- ✓ **Conflits de représentation des formes géométriques** : (au niveau des SIG)

II.3.2.2 Conflits sémantiques :

Ils sont dus au fait que des données présentes sur plusieurs systèmes admettent des interprétations différentes en fonction du contexte local d'utilisation, ils regroupent :

- ✓ **Conflits de nommage** : Ils touchent essentiellement des entités et des attributs, et consiste à affecter des noms différents pour le même concept (problème des synonymes), ou affecter le même nom à des concepts différents (problème des homonymes). Généralement on résout ce type de conflits par renommage.
- ✓ **Conflits de codification de valeurs** : Différence d'échelles utilisées, d'unités de mesure, d'intervalles de valeurs affectées aux différents attributs, ... etc.
- ✓ **Conflits de localisation des objets spatiaux (les SIG)** : différents systèmes de projection, type de positionnement direct ou relatif, précision de localisation, ... etc.

II.4 Intégration des bases de données :

II.4.1 Définition :

Par définition les bases de données contiennent des représentations d'objets d'un monde réel avec leurs propriétés et leurs liens entre eux. Ces représentations peuvent être diverses d'une base de données à une autre (problème de l'hétérogénéité).

L'intégration des données consiste tout simplement à permettre un accès cohérent à des données : d'origine, de structuration, et de représentation différentes. Autrement dit, l'intégration des bases de données dépasse les représentations pour considérer en premier lieu ce qui est représenté (les objets) plutôt que comment il est représenté.

II.4.2 Objectifs d'Intégration :

Les principaux objectifs de l'intégration de données sont :

- Produire un schéma global (intégré) à partir de plusieurs schémas locaux.

- Représentation des méta-données qui envisagent les différentes correspondances entre le schéma global (intégré) et les schémas locaux en terme données et requêtes.
- Permettre le pilotage de l'ensemble des bases de données d'une organisation.
- Assurer un accès rapide, simple et efficace aux données traitées.
- Permettre l'analyse comparative des données et améliorer la prise de décision.
- fournir les données appropriées aux bonnes personnes pour résoudre leur questionnement professionnel.

II.4.3 Processus d'intégration :

Le cœur du problème de conception est le processus d'intégration des bases de données. Ce processus consiste à prendre en entrée un ensemble de bases de données (schémas et populations), et à produire en sortie une description unifiée des schémas initiaux (le schéma intégré) et les règles de traduction (mapping) qui vont permettre l'accès aux données existantes à partir du schéma intégré.

- ❖ **Pré-intégration** : c'est l'étape dans laquelle les schémas en entrée sont transformés de différentes manières pour les rendre plus homogènes du coté sémantique et syntaxique.
- ❖ **Recherche des correspondances** : une étape consacrée à l'identification des éléments semblables dans les schémas initiaux et à la description précise de ces liens inter-schémas;
- ❖ **Intégration** : c'est l'étape finale dans le processus d'intégration, elle permet d'unifier les types en correspondance en un schéma intégré et de produire les règles de traduction associées entre le schéma intégré et les schémas initiaux.

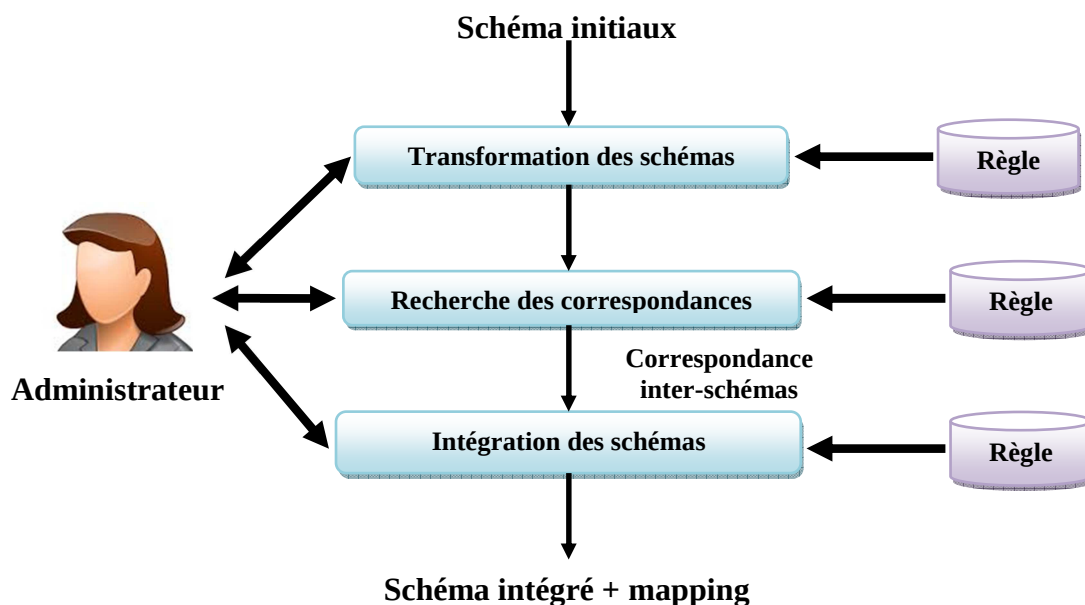


Figure I.4 : processus d'intégration

II.5 Classification des systèmes d'intégration :

Il existe plusieurs méthodes pour l'intégration de données. On cite l'approche entrepôt de données, et l'approche de médiation.

II.5.1 Intégration matérialisée (les entrepôts de données) :

II.5.1.1 Définition d'un entrepôt:

Un entrepôt de données (Data Warehouse) est défini comme un ensemble de données orientées sujets (classées selon leurs sujets ou centres d'intérêts), organisées, coordonnées, intégrées, variables dans le temps et non volatiles (disponible tout le temps). Ces données sont orientées vers le processus d'aide à la décision (marketing, commerce électronique, ...). En autre terme, un entrepôt de données est une réalisation particulière d'une grosse base de données (généralement une base de données fédérée) qui organise des données opérationnelles et historiques d'une entreprise afin de faciliter l'analyse et l'interrogation complexe de ces données. Alors l'entrepôt de données répond à tous les problèmes des données dispersées sur de multiple systèmes hétérogènes ; notamment l'intégration de ces données.

II.5.1.2 Approche entrepôt de données :

La démarche consiste à voir cette intégration comme la construction de bases de données réelles, appelées entrepôt de données, regroupant les informations pertinentes pour les applications considérées. L'utilisateur pose alors ses requêtes ou lance un traitement directement sur les données stockées dans l'entrepôt. Les problèmes posés par sa construction à partir de plusieurs bases de données spécialisées concernent, la définition de son schéma, son peuplement et sa mise à jour, en fonction des différentes sources d'information à partir desquelles il est construit. La figure illustre l'architecture générale d'un tel système.

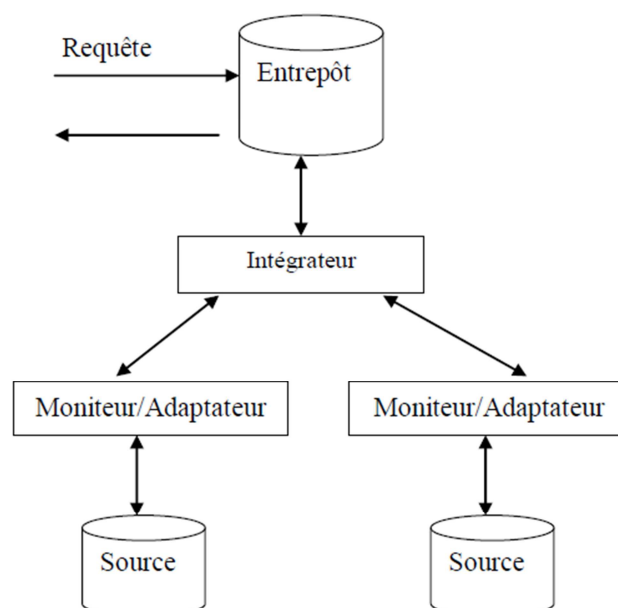


Figure I.5 : Approche entrepôt de données.

II.5.1.3 Processus d'intégration de données dans les entrepôts de données

Avant d'être stockées dans l'entrepôt, les données doivent être intégrées conformément au schéma médiateur pour donner à l'utilisateur une vue globale des informations. Nous abordons dans cette section les différents processus de l'intégration de données dans les entrepôts de données.

❖ Etapes d'intégration :

On distingue deux niveaux dans la construction des entrepôts de données. Le premier niveau correspond à la construction des sources de données opérationnelles, et de l'entrepôt de données global. Le second niveau englobe tous les entrepôts de données locaux. La raison de cette distinction est, qu'à chaque niveau, sont associées différentes étapes de traitement et différentes difficultés techniques.

Au premier niveau, le processus de construction est décomposé en quatre étapes principales, qui sont :

- ✓ l'extraction des données des sources de données opérationnelles.
- ✓ la transformation des données aux niveaux structurel et sémantique.
- ✓ l'intégration des données.
- ✓ le stockage des données intégrées dans le système cible. La figure résume l'enchaînement de ces étapes de traitement.

II.5.2 Intégration virtuelle (les médiateurs) :

II.5.2.1 Définitions :

L'approche de médiation consiste à développer une application chargée de jouer le rôle d'interface entre les bases de données et les applications utilisées par des utilisateurs. Un médiateur permet de résoudre les différents conflits entre les schémas des sources de données, il génère un plan d'exécution entre les utilisateurs et ces sources. Il simplifie, et décrit les données distribuées.

❖ Médiateur(*Mediator*) :

C'est un module logiciel recevant directement la requête d'un usager et devant la traiter. Celui-ci doit localiser l'information nécessaire pour répondre à la requête, résoudre les conflits schématiques et sémantiques, interroger les différentes sources et intégrer les résultats partiels dans une réponse homogène et cohérente.

Le rôle d'un médiateur est de collecter, nettoyer et combiner les données attendues par le système de médiation et produites par les différents adaptateurs.

Chapitre I : Les bases de données fédérées

Il permet à l'utilisateur d'interroger différentes bases de données. Pour ce faire, le médiateur récupère les requêtes des utilisateurs, les traite, les traduit, les distribue sur les bases de données, intègre les résultats des différentes bases de données et renvoie les résultats. Le médiateur utilise un schéma médiateur (vue unifiée). Cette vue est générée à partir de schémas de bases de données hétérogènes à intégrer.

❖ Adaptateur (*Wrapper*):

Un adaptateur est donc un composant qui :

- ✓ Réalise la transformation entre le modèle de données dans lequel sont représentées les données de la source et le modèle choisi au niveau médiateur.
- ✓ Effectue la transformation entre les expressions de requêtes du niveau global en expressions compréhensibles par les sources. Inversement, il traduit les réponses des sources au format du niveau global.
- ✓ Assure les traitements spécifiques non disponibles au niveau local et nécessaire pour le niveau global (augmentation du pouvoir d'expression et de traitement de la source locale).

Un adaptateur prend en compte les dimensions Distribution, Autonomie et Interopérabilité. Son objectif est d'accéder à une source, d'extraire les données appropriées et de les présenter dans un format spécifié.

La Figure montre une application utilisant un médiateur.

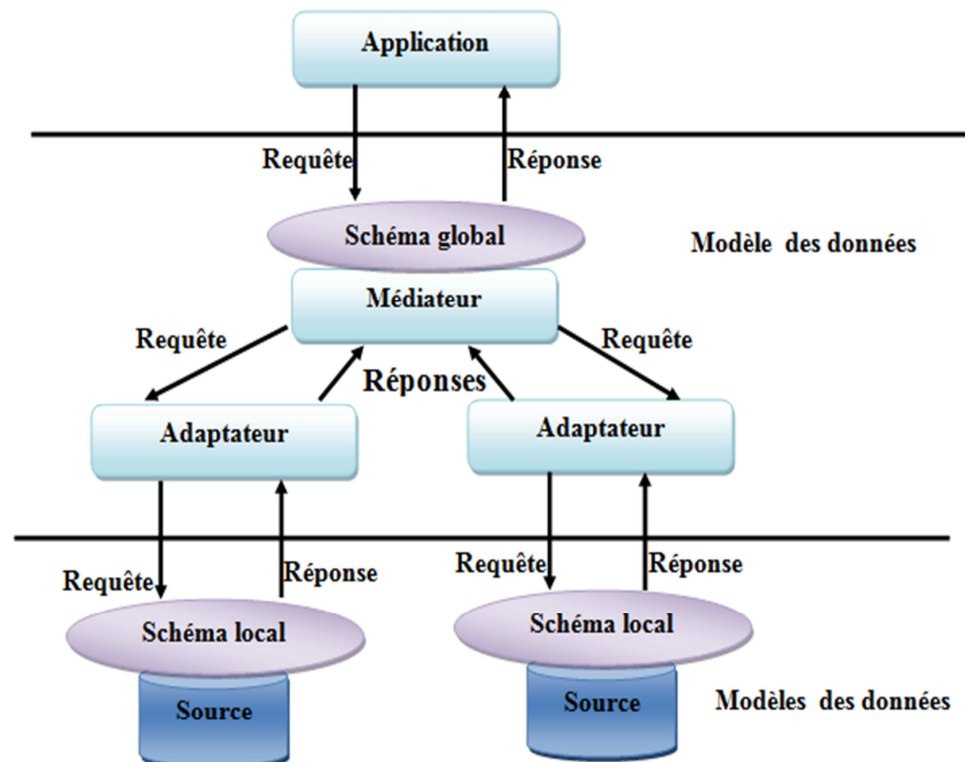


Figure I.6 : Approche médiateur.

II.5.2.2 Les approches de médiation :

Les approches de médiation sont classifiées selon le type de leur schéma médiateur en deux groupes : les approches **GAV** (global As View) où le schéma médiateur est une vue sur des schémas locaux et les approches **LAV** (Local As View), où les schémas locaux sont des vues du schéma global, ce qui signifie que les données dans chaque source doivent être définies selon le schéma global et non l'inverse.

❖ L'approche GAV :

Dans l'approche GAV le schéma médiateur est défini en fonction des schémas des sources à intégrer. Sa vue est utilisée pour réécrire les requêtes sur le schéma global en des requêtes distribuées sur des bases de données locales. Les requêtes sont alors transformées d'une requête sur la vue globale en des requêtes sur les sources locales. Son avantage est que la reformulation des requêtes est simple car chaque terme de la requête est remplacé par son équivalent dans les schémas des bases de données. L'inconvénient de cette approche réside dans l'ajout et la suppression de relations dans les schémas locaux car chaque modification entraîne la modification du schéma global et la création des interactions de la nouvelle source avec les sources existantes. Parmi les systèmes appartenant aux approches GAV, nous citons GARLIC, DISCO, TSIMMIS, MIX, et E-XMLMEDIA.

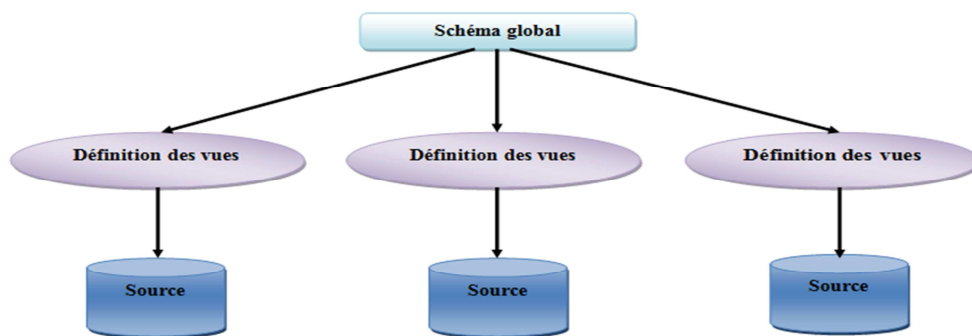


Figure I.7 : L'approche GAV

❖ LAV :

L'approche LAV consiste à décrire le contenu des sources à intégrer en fonction du schéma médiateur. Le traitement des requêtes sur le schéma global implique leur réécriture en des requêtes utilisant les vues. La requête sur le schéma global doit être reformulée suivant les schémas des sources locales, ce qui la rend complexe. Dans une telle approche l'ajout ou la suppression d'une relation est facile à traiter car elle n'affecte pas le schéma médiateur, vu

que chaque source est définie indépendamment. Parmi les systèmes appartenant aux approches LAV, nous citons Information Manifold, PICSEL et OBSERVER.

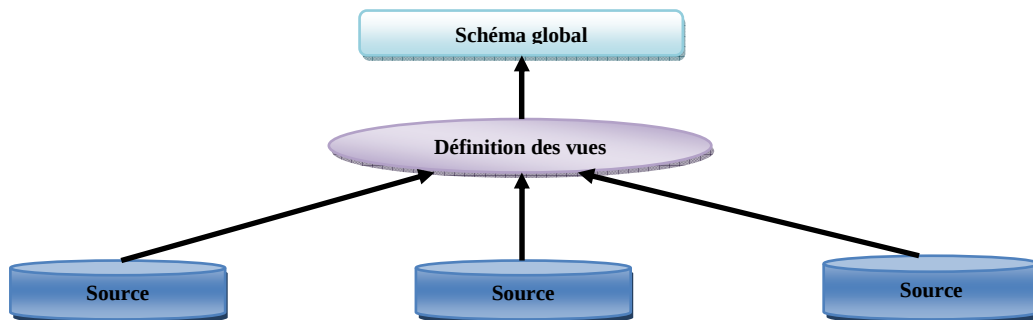


Figure I.8 : L'approche Local As View (LAV)

II.5.2.3 Processus de médiation :

Les médiateurs doivent en général se retrouver sur des nœuds différents de ceux qui contiennent les données (BD) et des postes de travail où sont utilisées en dernière instance les données (par les utilisateurs). Il existe plusieurs sortes de médiateurs accomplissant des tâches distinctes, voici les principales :

- ✓ Transformation et filtrage des BD.
- ✓ Accès et intégration de plusieurs BD.
- ✓ Traitement des données supportant un haut niveau d'abstraction et de généralisation (traitement intelligent de l'information).
- ✓ Répertoires intelligents des bases d'information comme les catalogues, les aides à l'indexation, les structures de thesaurus (ontologies).
- ✓ Gestion de l'incertitude liée aux données absentes ou incomplètes et aux données mal comprises.

II.5.3 Les vues :

Les vues corespondent à ce qu'on appelle « le niveau externe » qui reflète la partie visible de la base de données, elle apparait comme une table pour l'utilisateur.

En théorie, les utilisateurs ne devraient accéder aux informations qu'en questionnant des vues, ces dernières masquant la structure des tables interrogées. En pratique, la plupart des applications se passent de ce concept en manipulant directement les tables. La figure suivante illustre ce qui a été dit.

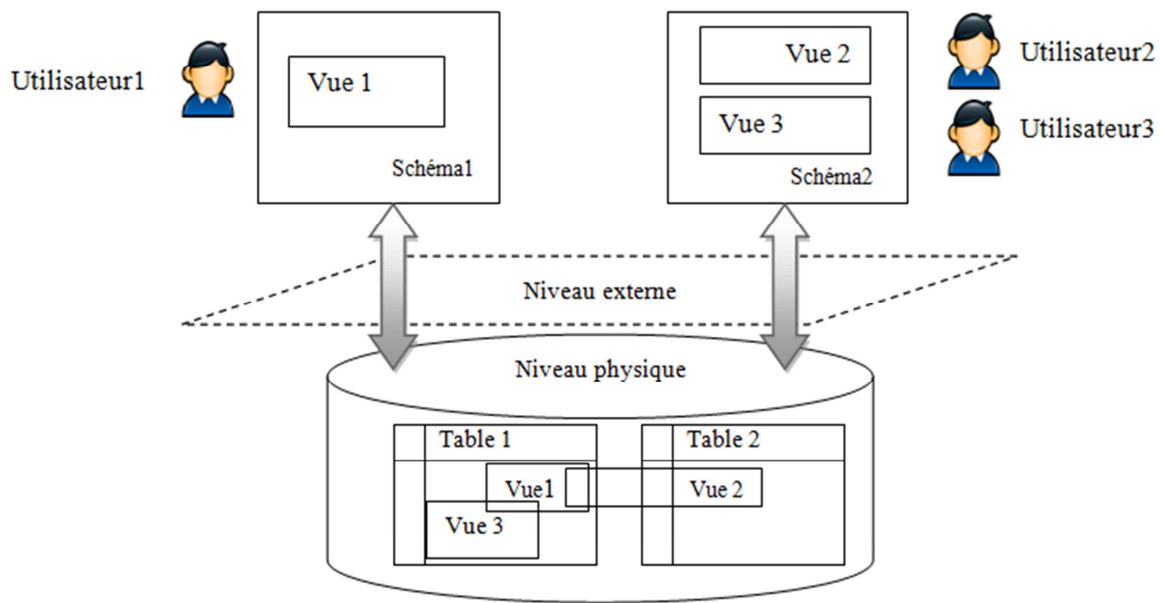


Figure I.9 : utilisation des vues

II.5.3.1 Les vues classiques :

Une vues est créée à l'aide d'une instruction `SELECT` appelée « requête de définition ». Cette requete interroge une (ou plusieurs) table(s) ou vue(s). une vue se recharge chaque fois qu'elle est interrogée, donc elle est considérée comme une table virtuelle car elle n'a pas d'existence propre, c'est-à-dire seule sa structure est stockée dans le dictionnaire. Ses données seront extraites de la mémoire à partir des tables source, à la demande.

II.5.3.2 Les vues matérialisées :

Afin de combiner les avantages des vues et des tables normales, il fallait donc créer une nouvelle présentation de données, c'est ainsi que le concept des vues matérialisées est né.

Une vue matérialisée est un objet qui permet de stocker le résultat d'une requête `select`. Là ou une vue classique se contente de stocker la requete, la vue matérialisée va sauvgarder directement les résultats (elle va donc les matérialiser), plutôt que la requête. Lorsque l'on fait une requête sur une vue matérialisée on va donc directement chercher des données dans celle-ci, sans passer par les tables d'origine et/ou une table temporaire intermédiaire. La figure ci-dessus montre le principe de fonctionnement d'une vue matérialisée.

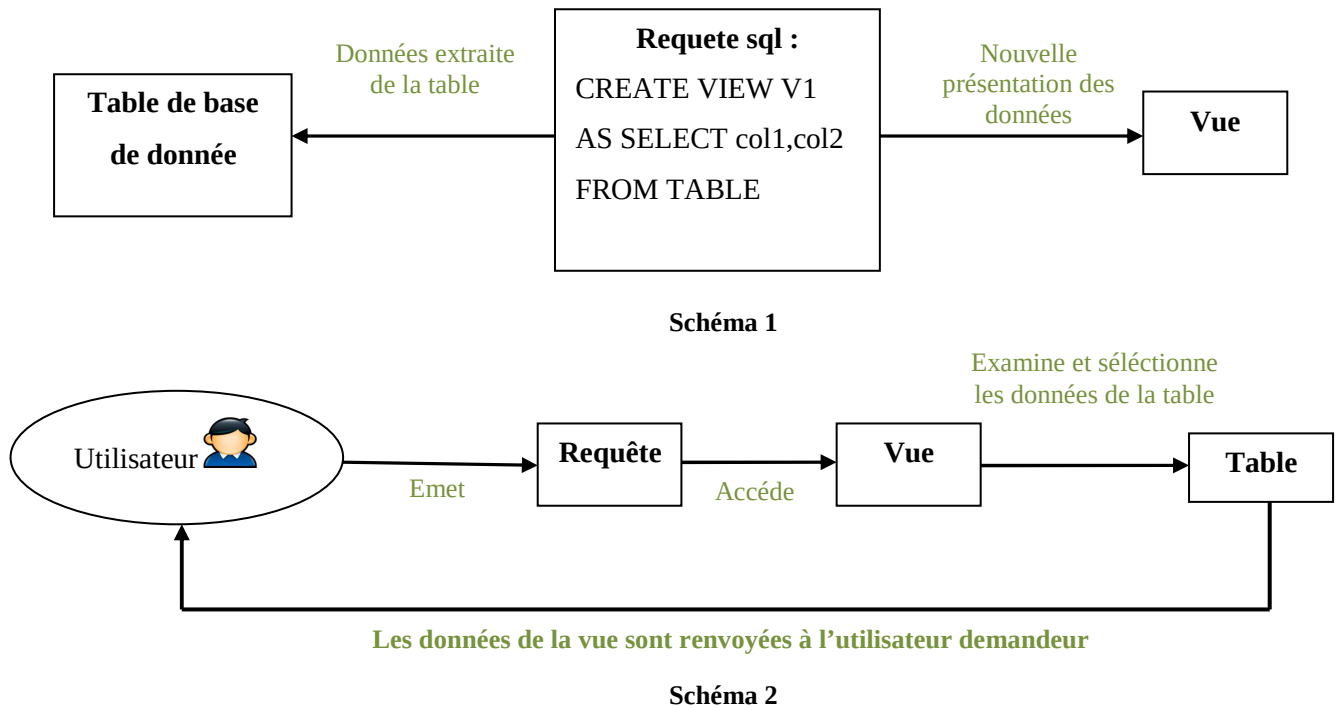


Figure I.10 : Utilisation des vues matérialisées .

- ✓ Le schéma 1 montre la création d'une vue à travers une requête sql qui permet de sélectionner des données d'une ou plusieurs tables.
- ✓ Le schéma 2 présente comment l'utilisateur accède à une vue à travers une requête sql, celle-ci est exécutée et lui renvoie les données. La demande d'une vue est identique à celle d'une table.

❖ **Avantages des vues matérialisées :**

- ✓ Permettent de gagner en performances.
- ✓ Permettent ou interdisent à un groupe d'utilisateur l'accès à certaines données.
- ✓ Gagnent de temps en évitant certains traitements lourds et répétitifs.
- ✓ Disponibilité des données.
- ✓ Intégrité des données.

II.6 Evaluation et optimisation des requêtes :

Le traitement de requêtes est un mécanisme absolument obligatoire dans l'intégration des bases de données. Il commence par une reformulation de celle-ci à partir du schéma global aux schémas à l'exportation des sources de données. Les algorithmes pour la reformulation des requêtes de changement, dépendent de la manière dont la relation entre le schéma global et les schémas locaux a été définie (comme GAV ou LAV). Lorsqu'une requête est reformulée, son optimisation

a lieu, parce que les sources de données peuvent avoir des capacités limitées pour satisfaire les requêtes.

Certaines sources de données sont en mesure de s'acquitter de requêtes SQL, d'autres sources peuvent être assez limitées dans leur capacité à satisfaire les requêtes. La vitesse du réseau de transmission de données peut changer, l'optimiseur peut ne pas reconnaître les statistiques de source de données qui sont généralement nécessaires dans l'optimisation de requêtes.

Compte tenu de ces problèmes, il est difficile d'utiliser les mêmes algorithmes d'optimisation de requêtes dans les applications d'intégration de données. C'est pourquoi les spécialistes ont proposé divers algorithmes adaptatifs, où l'exécution et l'optimisation d'une requête peuvent se produire en parallèle.

Une fois l'optimisation terminée et un plan d'exécution préparé, la requête est exécutée.

Conclusion :

L'intégration de bases de données est certainement une tâche complexe. C'est pourtant une tâche que les entreprises peuvent difficilement éviter si elles veulent mettre en route de nouvelles applications ou réorganiser le système d'information existant pour une meilleure productivité. Nous avons donné dans ce chapitre quelques définitions sur les bases de données et les SGBDs (Systèmes de Gestion de Bases de Données) qui les manipulent, Nous nous sommes intéressés aux bases de données réparties en général et aux bases de données fédérées en particulier.

Nous avons présenté les approches d'intégration et défini leurs architectures, dans le chapitre suivant nous allons détailler sur l'approche qu'on a opté dans notre mémoire.

Chapitre II:

Cas d'étude

Introduction :

Le sujet de notre projet a été effectué au niveau de la scolarité de la faculté GEI (Génie Electrique et Informatique) qui concerne la problématique de l'intégration de bases de données des différents départements qui la constitue. Avant de présenter notre solution, nous allons commencer par une présentation de l'organisme d'accueil de la scolarité de la faculté, ainsi les problèmes rencontrés. Nous exposons ensuite la solution qu'on a opté dans ce projet pour faire une intégration et interrogation de bases de données homogènes en se basant sur :

- ✓ Construction d'une vue unifiée à partir de schémas de sources de données existantes : et ça on utilisant des vues matérialisées.
- ✓ Contrôle d'accès aux informations confidentielles : présentation des approches de contrôle de donnée.

I. Présentation du domaine d'étude :

I.1. Présentation de l'organisme d'accueil :

L'université de Tizi-Ouzou a ouvert ses portes en 1977, elle avait alors le statut de centre universitaire et était implantée à Oued Aissi à 6km de sa résidence actuelle. Les modes de fonctionnement de l'université ont passés de plusieurs étapes jusqu'au 1999 où les instituts ont été dissous et érigés en facultés. après le décret exécutif No98/32 du 02/12/1998, l'article stipule la création au sein de l'université de Tizi-Ouzou des facultés :

- ✓ Faculté de médecine
- ✓ Faculté des sciences.
- ✓ Faculté de droit.
- ✓ Faculté des lettres et sciences humains.
- ✓ Faculté de génie de la construction.
- ✓ **Faculté de génie électrique et informatique (GEI).**
- ✓ Faculté des sciences biologiques et agronomiques.
- ✓ Faculté des sciences économiques et sciences de gestion.

I.1.1. Présentation de la faculté de GEI :

La faculté de G.E.I est une organisation à caractère de formation et de recherche, elle est composée de (4) départements :

- ✓ Département Automatique,
- ✓ Département Electronique,

Chapitre II : Cas d'étude

- ✓ Département Electrotechnique
- ✓ Département Informatique.

La figure ci-dessus montre l'organisation de la faculté:

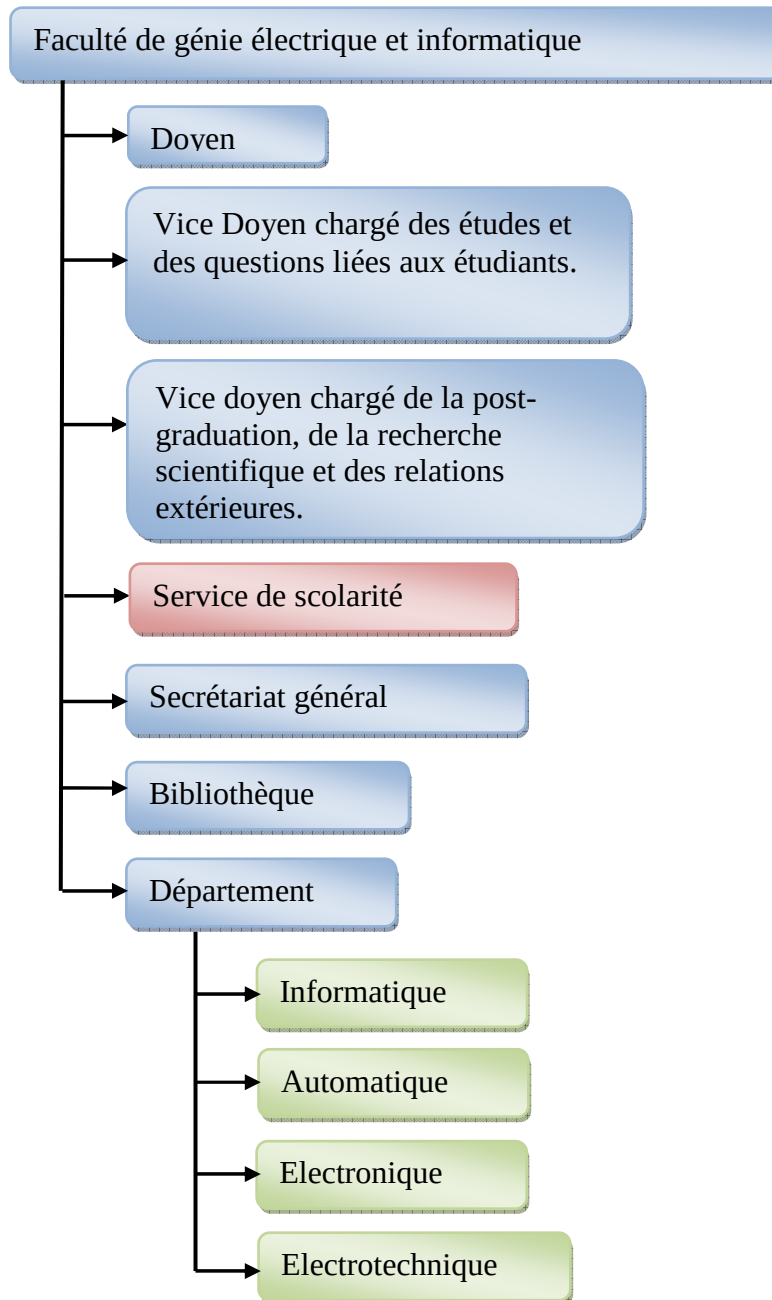


Figure II.1: Organigramme de la faculté génie électrique et informatique [2.1].

I.1.2. Présentation des départements de la faculté GEI :

Chaque département de la faculté GEI assure des tâches concernant le service de la scolarité, la figure ci-dessus présente l'organigramme d'un département en général :

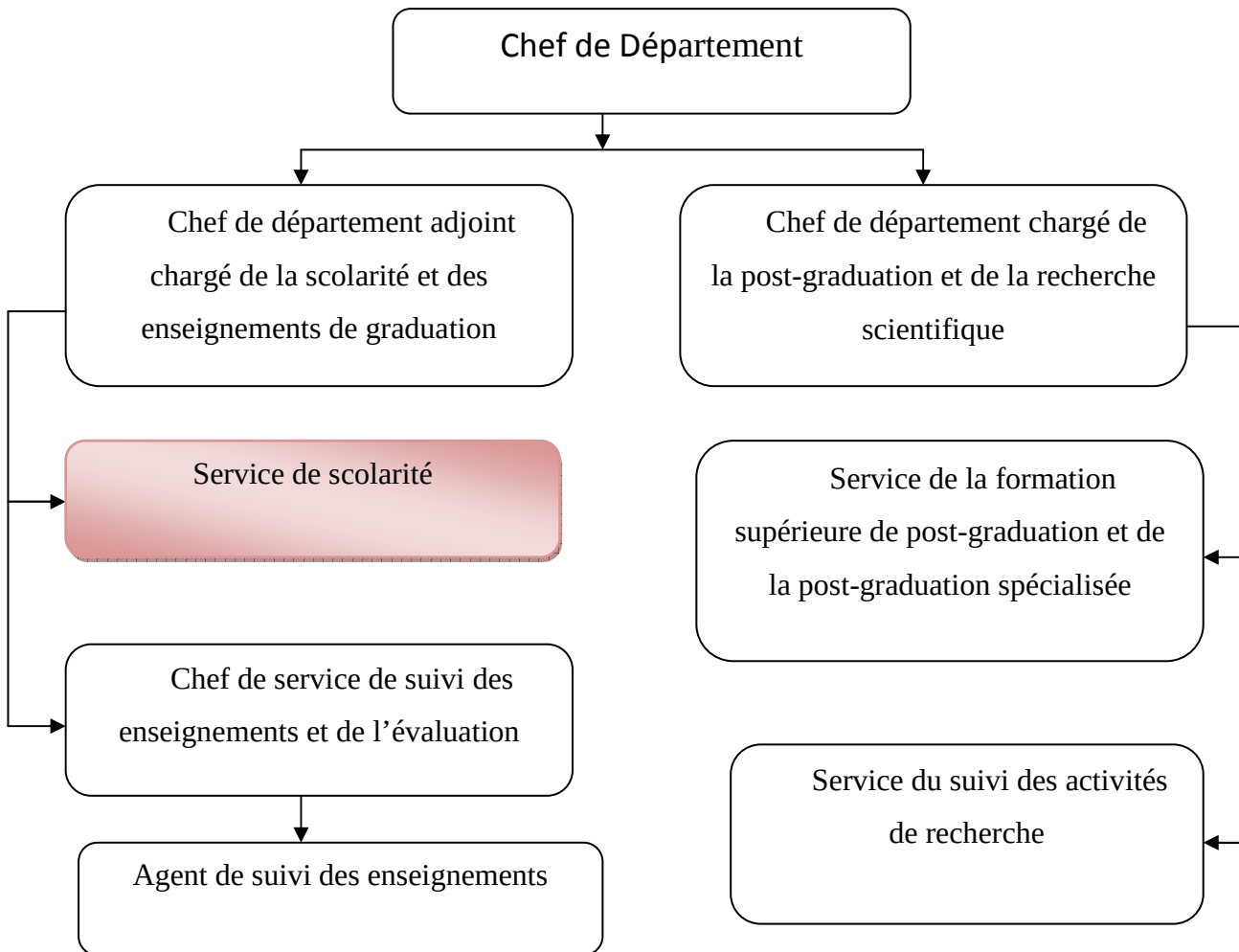


Figure II.2: Exemple d'organigramme d'un département de la faculté GEI [2].

I.1.3. Les tâches de la scolarité :

Le service de la scolarité de chaque département, assure de multiples tâches telles que :

- ✓ Les inscriptions.
- ✓ L'orientation.
- ✓ Délivrance de scolarité, des relevés de notes et des diplômes.
- ✓ Statistiques.

Ces tâches sont gérées par le personnel du département.

I.2. Planification du projet:

I.2.1. Problématique:

Après l'exploration du terrain et les interviews que nous avons réalisées avec le service de la scolarité de GEI, nous avons constaté que le système actuel de la gestion de celle-ci, présente certain nombre d'insuffisances, qui peuvent se résumer en :

- ✓ La faculté de GEI dispose d'un réseau informatique à haut débit mais très mal exploité. En fait, il n'existe aucune application ou logiciel fonctionnant sous réseau.
- ✓ Le transfert de données entre le responsable de la scolarité et les différents départements de la faculté GEI se fait manuellement.
- ✓ La non disponibilité des informations au bon moment.
- ✓ Contrôle non supervisé de cursus universitaire des étudiants.

Dans la figure qui suit ont montre l'organisation actuelle des échanges des flux d'information dans la faculté GEI c'est-à-dire l'état du système des bases de données :

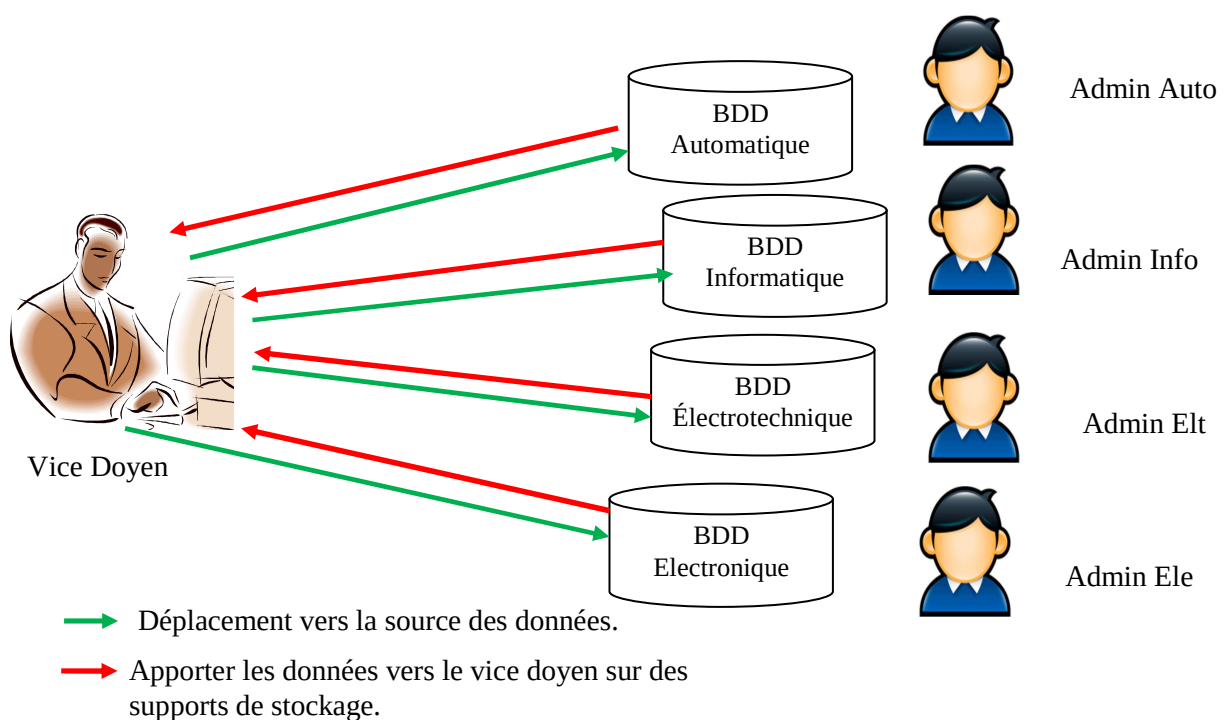


Figure II.3. Architecture actuelle des flux d'information dans la faculté GEI

I.2.2. Objectifs de l'étude:

L'objectif principal de notre étude, est de remédier aux problèmes cités ci-dessus. Pour cela, nous avons proposé une intégration des bases de données réparties sur les quatre départements de la faculté GEI, permettant de :

- ✓ Intégrer l'information dans une seule base de données pour garantir un accès transparent et cohérent.
- ✓ Augmenter la disponibilité de données et réduire le trafic réseau qui est une première possibilité d'accroître les performances du réseau en utilisant la réplication.
- ✓ Gagner du temps d'exécution des différents traitements réalisés. Par conséquent, satisfaire les besoins des utilisateurs en matière d'efficacité et de rapidité d'exécution.
- ✓ Offrir une vue uniforme et transparente des données stockées dans différentes bases de données.
- ✓ Réaliser quelques traitements à travers une interface facile à manipuler.

Sachant que notre objectif principal est purement pédagogique, nous allons nous contenter d'atteindre les objectifs cités dessus en ajoutant :

I.2.3. Solution proposée :

La solution qu'on a adoptée dans ce projet pour atteindre les objectifs souhaités est la mise en place d'une base de données fédérée (entrepôt) des différents sites de la faculté GEI au niveau du bureau de vice doyen, dans laquelle on implémentera les vues matérialisées sous oracle 10g.

Notre application contient quatre (4) sources de données (BDD département informatique, BDD département automatique, BDD département électronique, BDD département électrotechnique), que nous allons intégrer vers une base de données globale. Dont l'objectif est de donner au vice doyen (utilisateur qui en accès aux bases de données de la faculté) une vue globale sur les données stockées sur les sites locaux comme s'il accède à une seule base de données. La figure suivante illustre l'échange des différents flux d'information dans la faculté GEI.

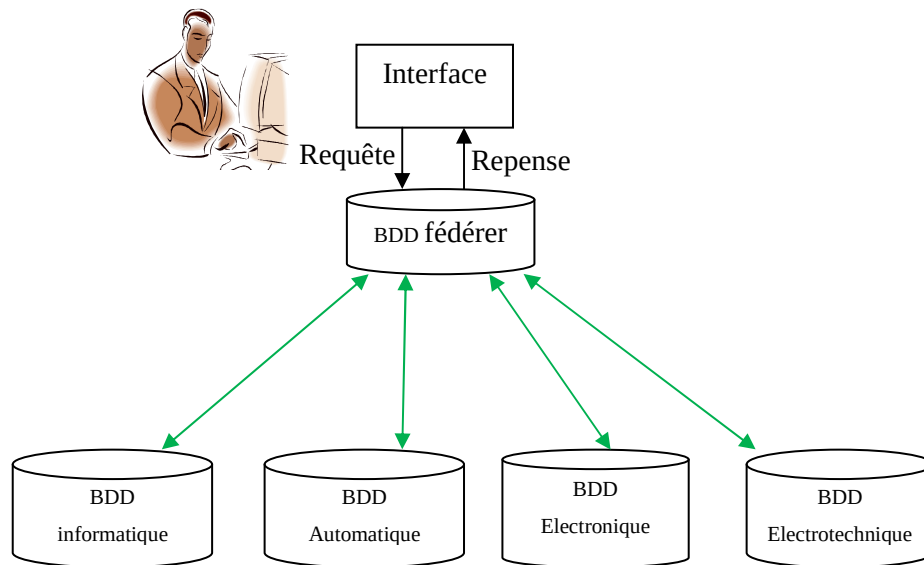


Figure II.4. Architecture des flux d'information dans la faculté GEI par notre solution

I.2.4. Justification du choix :

La création des vues matérialisées (*materialized views*) au niveau du bureau du vice doyen, nous permet d'avoir une localisation physique des données au niveau de ce dernier et un rafraîchissement complet des données (par exemple, tous les jours à 10 h du matin). Donc, on favorise les accès locaux qui ont pour but de réduire le trafic réseau.

Comme le doyen a besoin de toutes les données de tous les départements, la création des vues (*views*) au niveau de ce dernier, lui permet d'avoir accès à toutes les données en toute transparence.

Les accès distants au niveau du bureau du doyen permettent à ce dernier d'effectuer le contrôle de cursus universitaire.

II. Analyse et conception :

II.1 Analyse :

Cette phase comprend l'identification des besoins du système, les acteurs participant, leurs interactions avec le système ainsi que les cas d'utilisation.

II.1.1 Description du sujet à étudier :

Dans notre projet on s'est intéressé aux bases de données des étudiants et leurs cursus universitaire depuis L2 (2eme année de licence) jusqu'à M2 (2eme année de master) des différents départements de la faculté GEI.

Chaque département possède une base de données comportant un ensemble de table dans notre cas nous allons travaillés sur trois (3) tables (table_etudiant, table_module, table_note), qu'on a supposé car l'objectif de notre travail consiste à l'intégration des différentes bases de données et non à la gestion de la scolarité elle même :

II.1.2 Identification des acteurs :

Dans notre sujet on distingue 5 acteurs qui sont :

➤ Quatre (4) administrateurs des différents départements :

Ils assurent les taches suivantes :

- ◆ Mise à jours des bases de données (modifier, supprimer, ajouter...).

➤ Vice doyen de la faculté :

Etant donner que notre solution est pour le service de la pédagogie, nous focalisons notre intérêt sur le vice doyen qui de son coté assure plusieurs taches telles que :

- ◆ Extraire et visualiser les données à travers le système fédéré.
- ◆ Suivre le cursus des étudiants de la faculté.
- ◆ Rechercher des anomalies pour intervenir dans le bien des étudiants.

Nous s'intéresserons juste à celles citées dessus :

1. Extraire et visualiser les données à travers le système fédéré.
2. Affichage de tous les étudiants de la faculté.
3. Affichage des étudiants de chaque département.

Chapitre II : Cas d'étude

II.1.3 Présentation des bases de données locales :

Nous allons décrire ci-dessus les schémas de chaque base de données des quatre départements qu'on a supposé nous-mêmes pour présenter notre solution de fédération :

➤ Département Informatique :

◆ Table étudiant :

Champs	désignation	Type	Clé
Id_inf	Identifiant étudiant	VARCHAR (10)	Primaire
Nom_inf	Nom étudiant	VARCHAR (10)	
Prénom_inf	Prénom étudiant	VARCHAR (10)	
Sexe_inf	Sexe étudiant	VARCHAR (4)	
Adr_inf	Adresse étudiant	VARCHAR (10)	
D_nais_inf	Date de naissance étudiant	DATE	
Ann_bac_inf	Année d'obtention du bac	NUMBER(4)	
Moy_bac_inf	Moyenne du bac	NUMBER(4)	
W_bac_inf	Wilaya du bac	VARCHAR(10)	
Option_inf	Spécialité d'étude	VARCHAR(10)	
Niv_act_inf	Niveau d'étude pour l'année universitaire mentionnée	VARCHAR(10)	
Ann_uni_inf	Année universitaire	NUMBER(4)	

◆ Table module :

champ	Spécification	Type	clé
Cod_mod_inf	Code module	VARCHAR(6)	primaire
Desc_mod_inf	Désignation module	VARCHAR(15)	
Coef_mod_inf	Coefficient module	NUMBER(1)	

➤ Département automatique :

◆ Table étudiant :

Champs	désignation	Type	Clé
Id_aut	Identifiant étudiant	VARCHAR (10)	Primaire
Nom_aut	Nom étudiant	VARCHAR (10)	
Prénom_aut	Prénom étudiant	VARCHAR (10)	
Sexe_aut	Sexe étudiant	VARCHAR (1)	
Adr_aut	Adresse étudiant	VARCHAR (10)	
D_nais_aut	Date de naissance étudiant	DATE	
Opt	Spécialité d'étude	VARCHAR(10)	
Niv_act_aut	Niveau d'étude pour l'année universitaire	VARCHAR(10)	
Ann_uni_aut	Année universitaire	VARCHAR2(4)	

◆ Table module :

champ	Spécification	Type	clé
Cod_mod_aut	Code module	VARCHAR(6)	primaire
Desc_mod_aut	Désignation module	VARCHAR(15)	
Coef_mod_aut	Coefficient module	NUMBER(1)	

Chapitre II : Cas d'étude

➤ Département électronique :

◆ Table étudiant :

Champ	Signification	Type	Clé
Id_ele	Identifiant étudiant	VARCHAR(5)	Primaire
Nom_ele	Nom étudiant	VARCHAR(10)	
Prénom_ele	Prénom étudiant	VARCHAR(10)	
Sex_ele	Sexe étudiant	VARCHAR(10)	
Adr_ele	Adresse étudiant	VARCHAR(20)	
Dat_nais_ele	Date de naissance étudiant	DATE	
Ann_bac_ele	Année de bac	NUMBER(4)	
W_bac_ele	Wilaya de bac	VARCHAR(15)	
Option_ele	Option	VARCHAR(10)	
Niv_act_ele	Niveau actuel	VARCHAR(10)	
Ann_uni_ele	Année universitaire	NUMBER(4)	

◆ Table module :

Champ	Signification	Type	Clé
Cod_mod_ele	Code de module	VARCHAR(6)	Primaire
Desc_mod_ele	Description de module	VARCHAR(10)	
Coef_mod_ele	Coefficient de module	NUMBER(1)	

➤ Département électrotechnique :

◆ Table étudiant :

Champ	Signification	Type	Clé
Id_elt	Identifiant étudiant	VARCHAR(5)	Primaire
Nom_elt	Nom étudiant	VARCHAR(10)	
Prénom_elt	Prénom étudiant	VARCHAR(10)	
Sex_elt	Sexe étudiant	VARCHAR(10)	
Adr_elt	Adresse étudiant	VARCHAR(10)	
Dat_nais_elt	Date de naissance étudiant	DATE	
Ann_bac_elt	Année de bac	NUMBER(4)	
Moy_bac_elt	Moyen étudiant	NUMBER(4)	
Option_elt	Option	VARCHAR(10)	
Niv_act_elt	Niveau actuel	VARCHAR(10)	
Ann_uni_elt	Année universitaire	NUMBER(4)	

◆ Table module :

Champ	Signification	Type	Clé
Cod_mod_elt	Code de module	VARCHAR(6)	Primaire
Desc_mod_elt	Description de module	VARCHAR(10)	
Coef_mod_elt	Coefficient de module	NUMBER(1)	

II.1.4 Démarche d'intégration suivie :

Le processus d'intégration est le cœur du problème de ce projet. Nous avons opté la démarche suivante :

❖ **Pré-intégration** : c'est l'étape dans laquelle les tâches suivantes sont faites :

- ◆ Recherche des correspondances entre les diverses tables des différentes bases de données locales : c'est-à-dire on analyse les trois tables des quatre départements puis on mentionne les attributs communes ainsi voir leurs types.
- ◆ Résolution des conflits :
 - ✓ Conflits de noms : dans la table `etudiant_info` du département informatique et `etudiant_elt` du département électrotechnique les noms des attributs sont différents, exemple : `id_info` et `id_elt`, ce conflit se résout par la création des attributs de la vue matérialisée.
 - ✓ Conflits de nombre d'attribut : dans les tables, `etudiant_info` du département informatique, `etudiant_ele` du département électronique, `etudiant_elt` du département électrotechnique et `etudiant_aut` du département automatique le nombre d'attribut est différent on prend le nombre maximal des attributs, pour les tables qui ont moins d'attribut on leur ajoute des champs nuls.
 - ✓ Conflits de type : dans la table `etudiant_aut` l'attribut `ann_uni_aut` est de type « VARCHAR2 » et dans les autres tables est de type « NUMBER », on a résolu ce problème avec le « CAST ».

❖ **Intégration** :

- ◆ Création des liens entre la base de données fédérée et les quatre bases de données des différents départements :
 - ✓ Lien de la base de données fédérée vers la base de données de département informatique qui s'appelle « lieninfo ».
 - ✓ Lien de la base de données fédérée vers la base de données de département automatique qui s'appelle « lienauto ».
 - ✓ Lien de la base de données fédérée vers la base de données de département électronique qui s'appelle « lienele ».

Chapitre II : Cas d'étude

- ✓ Lien de la base de données fédérée vers la base de données de département électrotechnique qui s'appelle « lienelt ».
- ◆ Création des différentes vues matérialisées au niveau de la base de données fédérée :
 - ✓ « VUE_ETUDIANT » vue matérialisée sur les tables « etudiant_info », « etudiant_eut », « etudiant_ele », « etudiant_elt »,
 - ✓ « VUE_MODULE » vue matérialisée sur les tables « module_info », « module_aut », « module_ele », « module_elt »,
 - ✓ « VUE_NOTE » vue matérialisée sur les tables « note_info », « note_aut », « note_ele », « note_elt ».

II.2 Conception :

II.2.1 Conception de la base de données fédérée :

Nous allons présenter les schémas conceptuels locaux des bases locales, par la suite on présente les vues matérialisées créées ainsi les liens entre les différents sites, pour compléter le schéma conceptuel global de la base de données fédérée comme le montre la figure suivante :

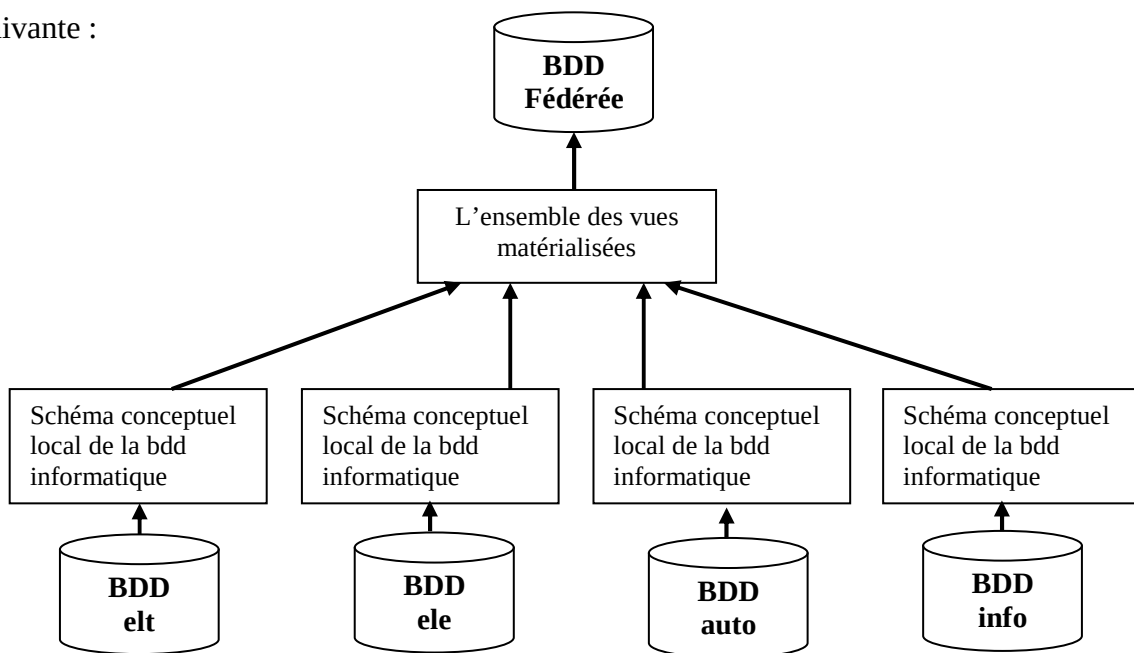


Figure II.5 Plan de représentation des schémas conceptuels

II.2.2 Diagramme du schéma conceptuel de la base de données de chaque département :

✓ Informatique :

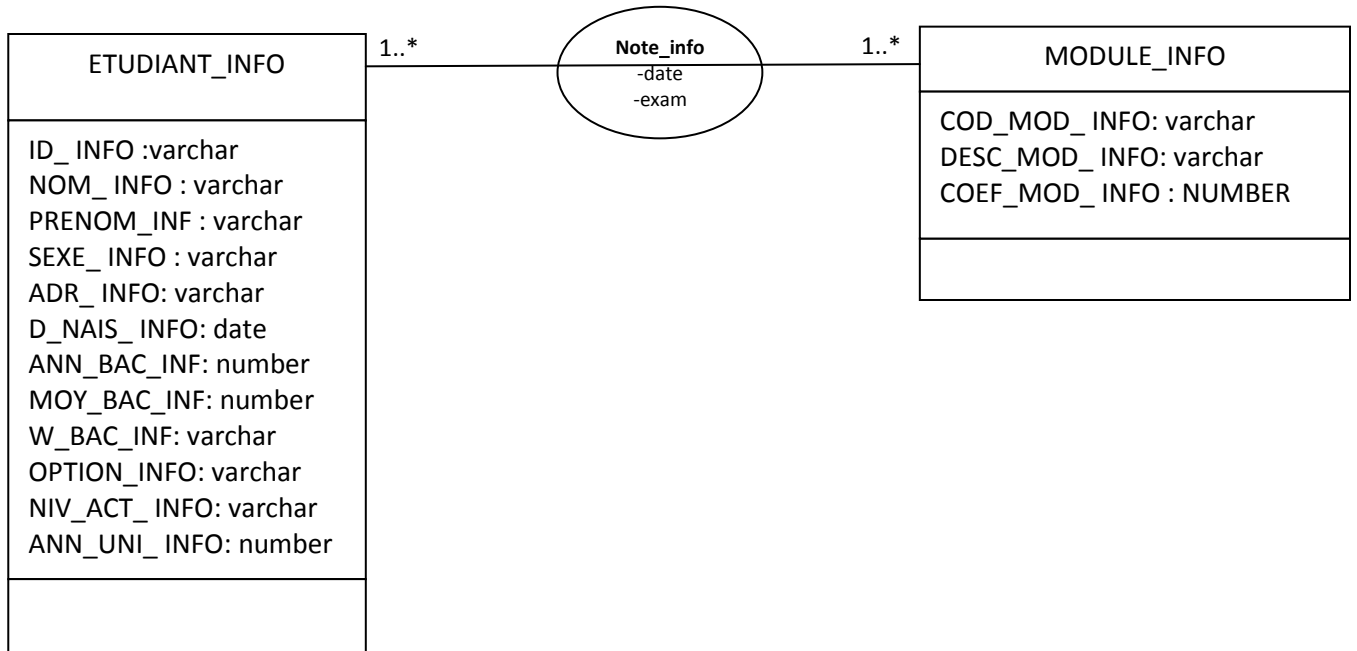


Figure II.6 : Diagramme de schéma conceptuel de la base de données Informatique

✓ Automatique :

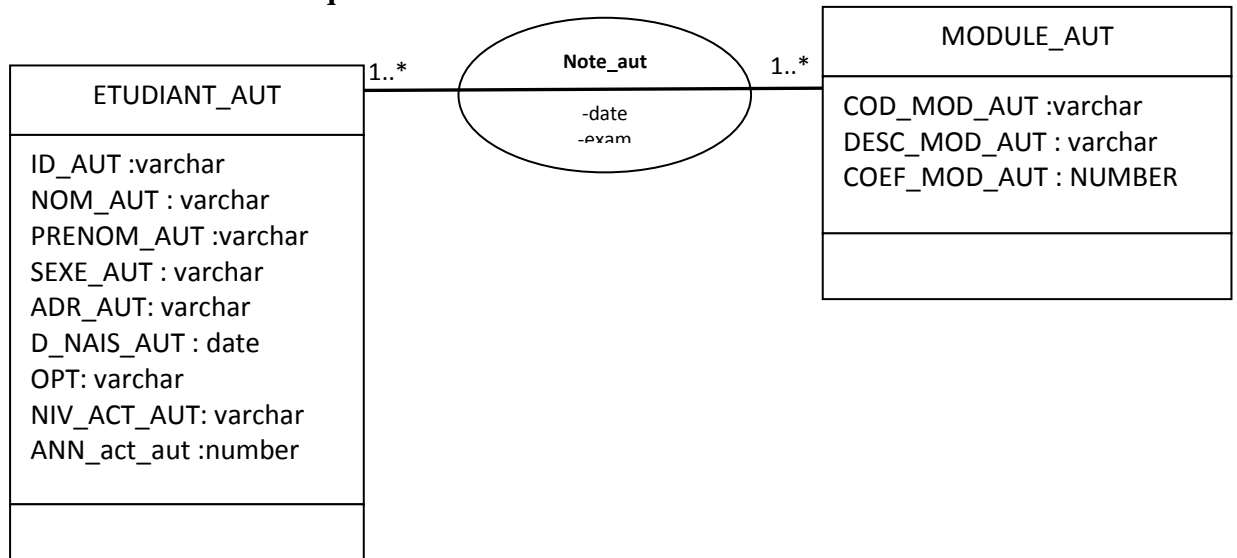


Figure II.7 : Diagramme de schéma conceptuel de la base de données Automatique

✓ Electronique :

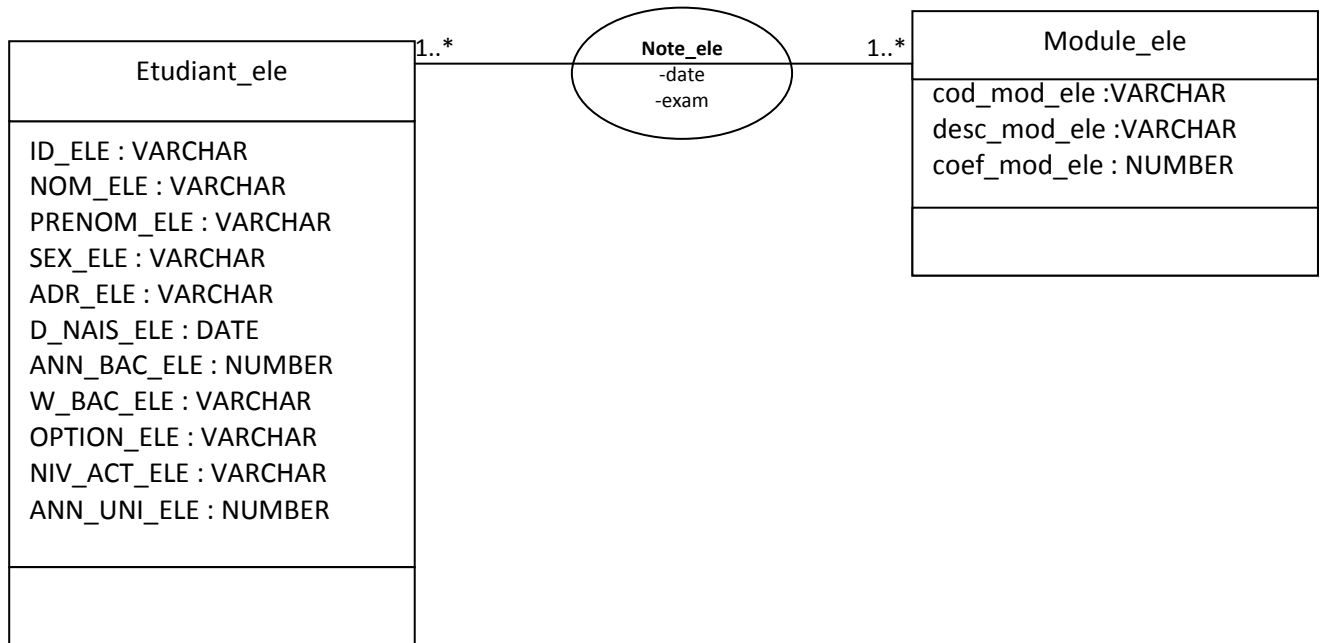


Figure II.8 : Diagramme de schéma conceptuel de la base de données électronique

✓ Electrotechnique :

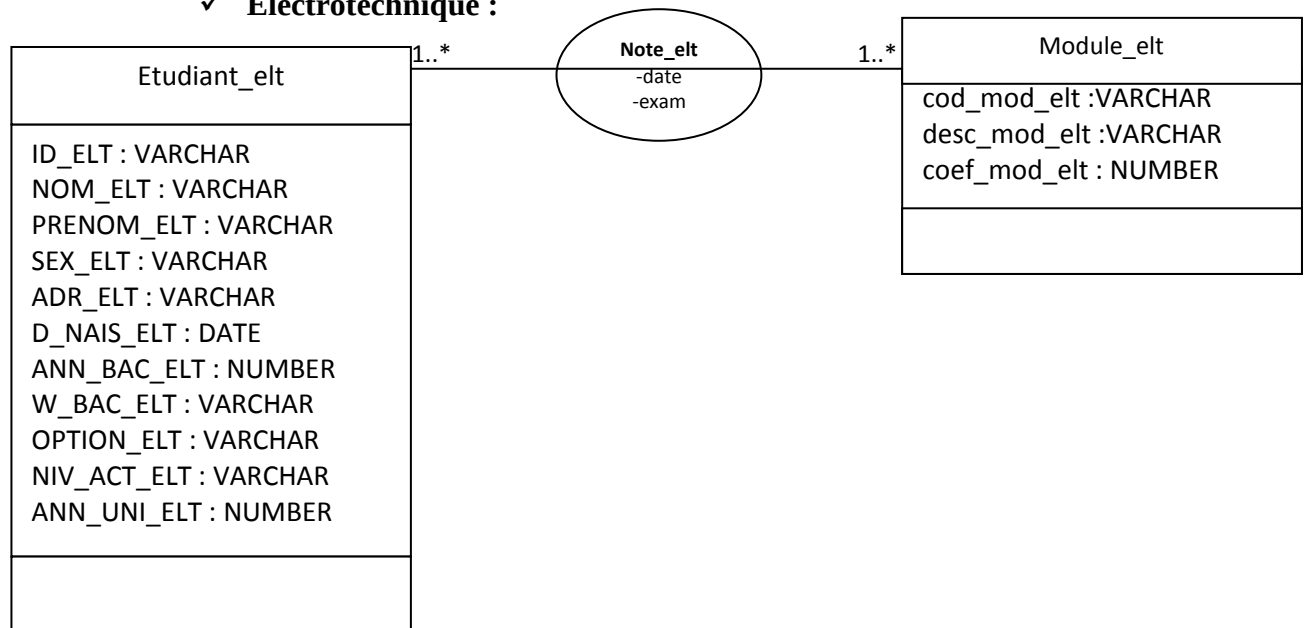


Figure II.8 : Diagramme de schéma conceptuel de la base de données électrotechnique

Chapitre II : Cas d'étude

✓ Diagramme du schéma logique de la base de données fédérée

II.2.3 Diagramme du schéma conceptuel de la base de données fédérée :

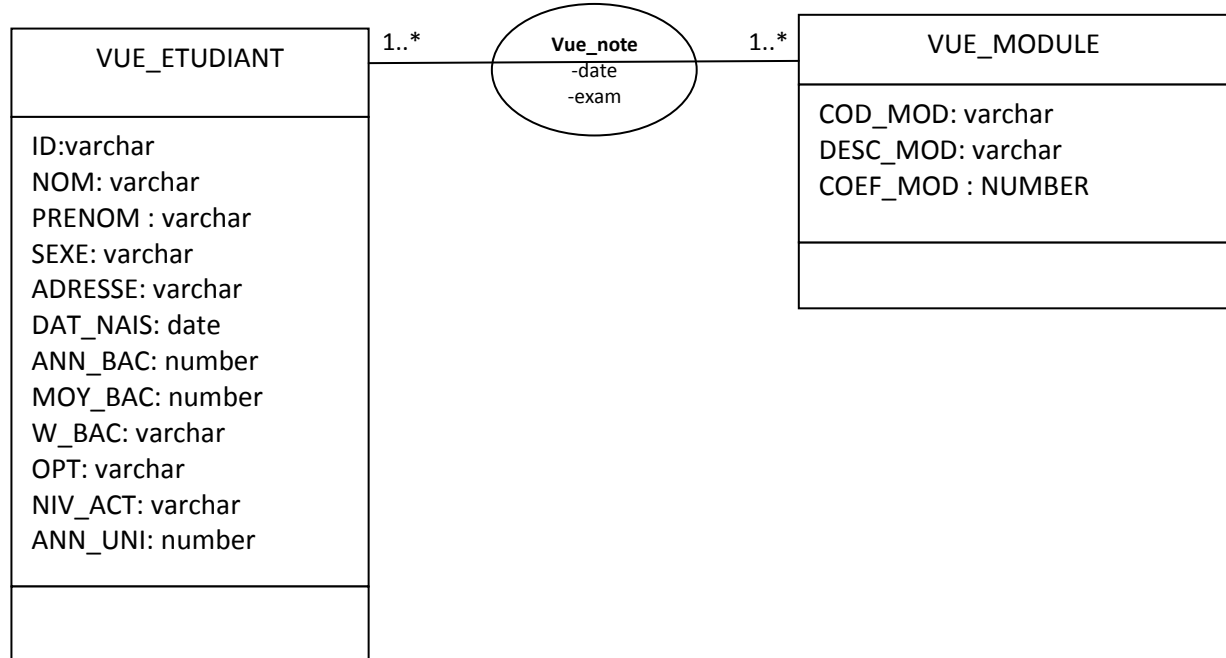


Figure II.10 : Diagramme du schéma conceptuel de la base de données fédérée.

II.2.4 Présentation des vues matérialisées de la base de données fédérée :

♦ Table étudiant :

Champ	Signification	Type	Clé
Id	Identifiant étudiant	VARCHAR(10)	Primaire
Nom	Nom étudiant	VARCHAR(10)	
Prénom	Prénom étudiant	VARCHAR(10)	
Sexe	Sexe étudiant	VARCHAR(10)	
Adr	Adresse étudiant	VARCHAR(20)	
Dat_nais	Date de naissance étudiant	DATE	
Ann_bac	Année de bac	NUMBER(4)	
Moy_bac	Moyen étudiant	NUMBER(4)	
W_bac	Wilaya de bac	VARCHAR15)	
Opt	Option	VARCHAR(10)	
Niv_act	Niveau actuel	VARCHAR(10)	
Ann_act	Année universitaire	NUMBER(4)	

♦ Table module :

Champ	Signification	Type	Clé
id_mod_f	Code de module	VARCHAR(6)	Primaire
Desc_mod_f	Description de module	VARCHAR(10)	
Coef_mod_f	Coefficient de module	NUMBER(1)	

II.2.5 Création des liens :

Pour permettre un accès polyvalent et transparent du vice doyen à la base de données fédérée, on a procédé à la création des liens (*DATABASE LINK*) entre les différents sites existants.

II.2.6 Les vues matérialisées créées :

La fédération des données se fait en utilisant des vues matérialisée, entre les différentes tables des diverses bases de données.

On citera dessus un exemple de création de vue matérialisée « vue_etudiant » :

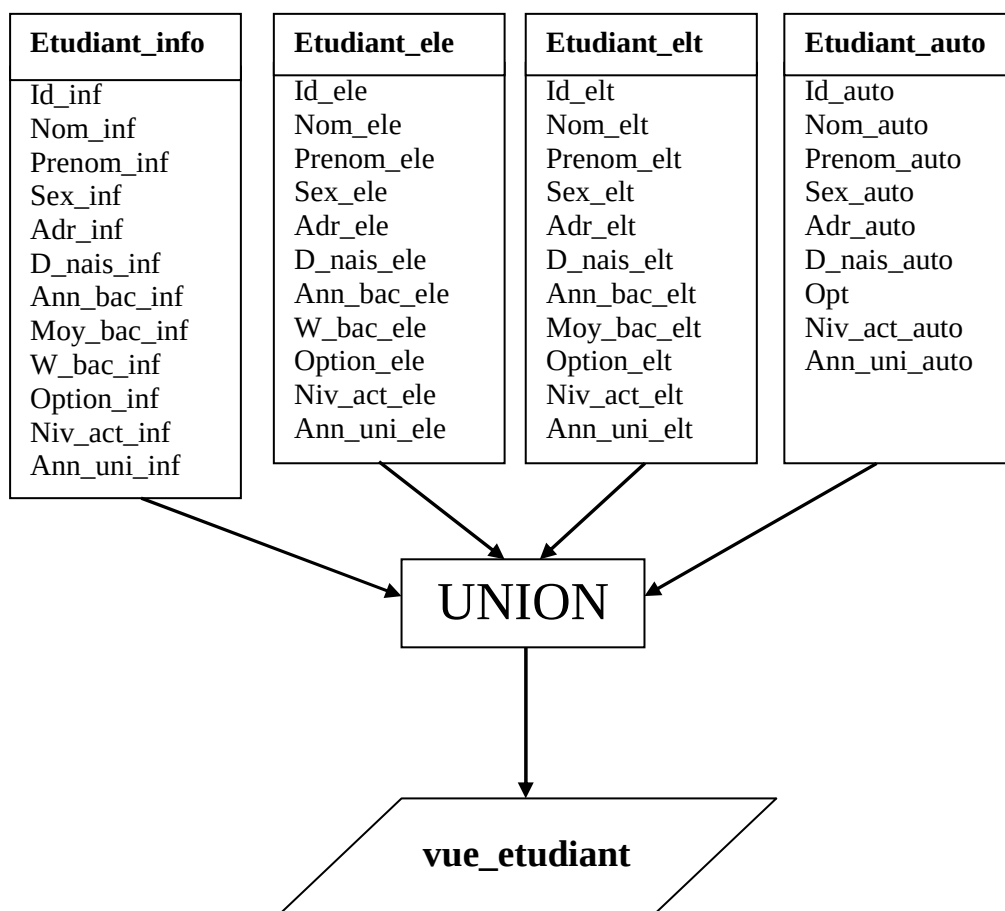


Figure II.11. Exemple de vue matérialisée cas « vue_etudiant »

II.2.7 Conception de l'interface utilisateur/BDDF :

✓ Diagramme de cas d'utilisation globale :

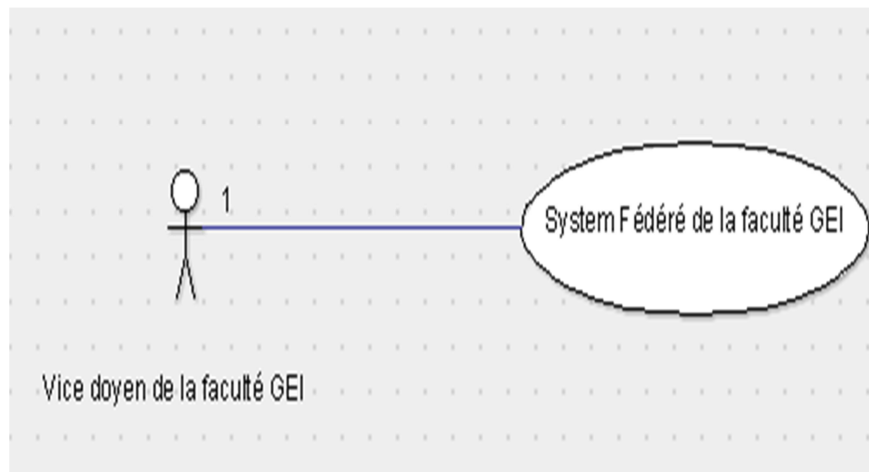


Figure II.12 : Diagramme de contexte

✓ Diagramme de cas d'utilisation :

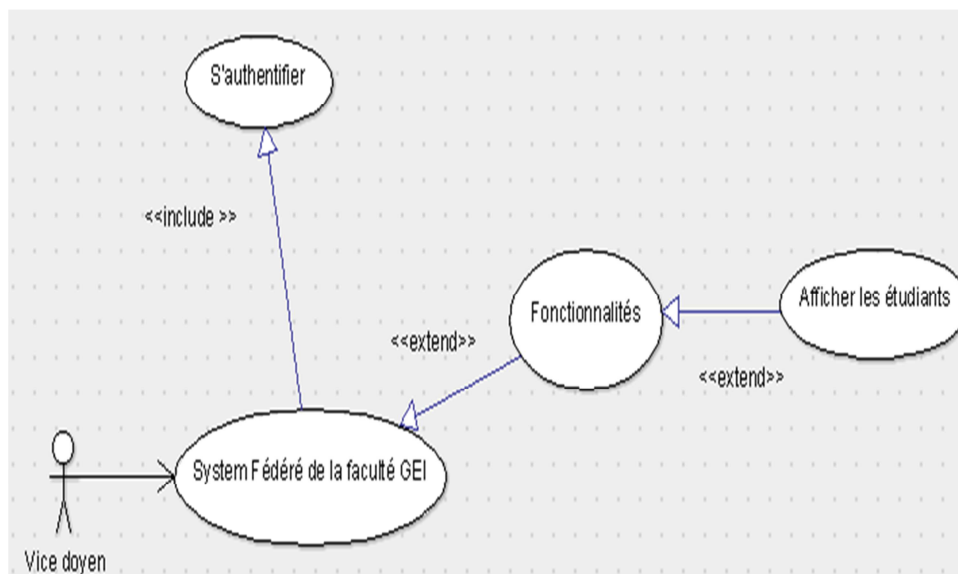


Figure II.13 : Diagramme de cas d'utilisation

Conclusion :

Nous avons présenté dans ce chapitre le processus d'analyse et conception en deux niveaux, le niveau applicatif et le niveau de données. On a commencé par le niveau de données qui est le but de notre projet, ou on a exposé la démarche qu'on a suivie pour réaliser l'intégration des données au niveau du bureau du vice doyen. Puis on est passé au niveau applicatif ou on a illustré quelques fonctionnalités qui accèdent à la base de données fédérée.

Le chapitre suivant sera donc la traduction de cette conception en un ensemble de codes qui nous a permis de faire l'intégration (niveau données). En ce qui concerne la partie applicative on présentera quelques interfaces.

Chapitre III:

*Implémentation
et
réalisation*

Introduction :

Dans ce chapitre nous nous intéressons à la réalisation de notre application. Il est question donc de présenter le support matériel et logiciel qui nous a servi d'appui pour le développement de notre application tout en focalisant notre attention sur les outils utilisés, par la suite nous allons présenter notre application qui couvre la création et l'implémentation de notre base de données fédérée à partir de quatre sites et ainsi que la présentation de quelques interfaces.

I. Environnement de développement :

I.1. Environnement logiciels :

I.1.1. Eclipse IDE :

Eclipse est un environnement de développement intégré (IDE) Java, développé par IBM extensible, universel et polyvalent, permet de créer des projets de développement mettant en œuvre n'importe quel langage de programmation comme Java, C/C++, Python, PERL, Ruby, etc. Il supporte de nombreux outils de développement de haut niveau très complets. Eclipse est un outil puissant, gratuit, libre.

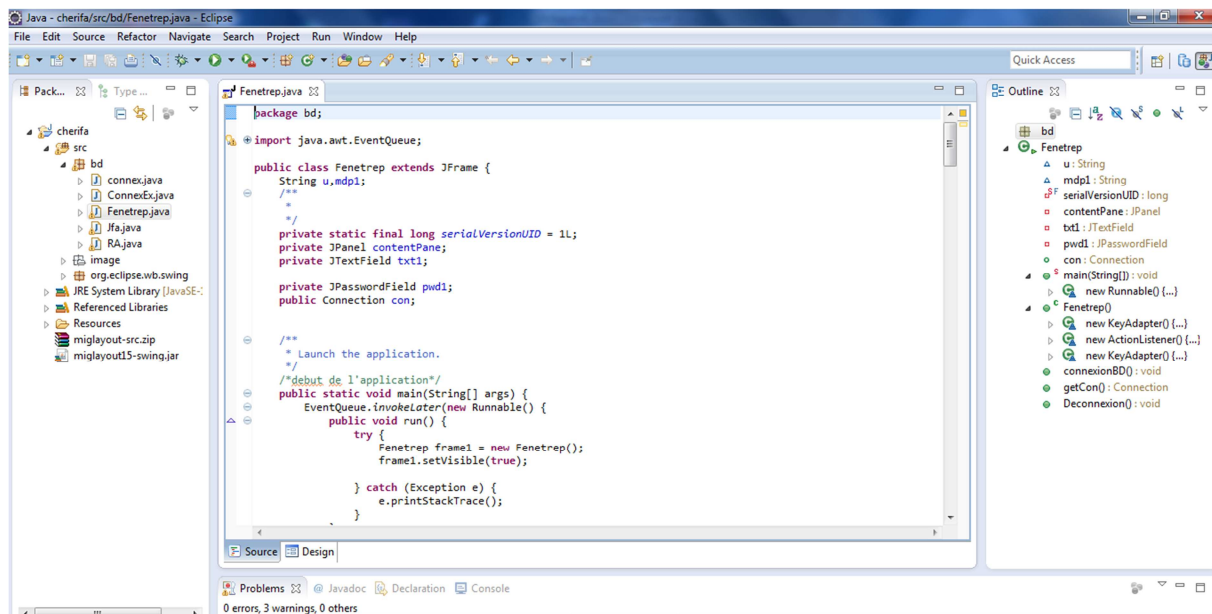


Figure III.1. Interface de l'IDE Eclipse

1.1.2. Oracle Database 10g express Edition:

Oracle 10 g se démarque des autres gestionnaires de bases de données par son côté administration très développé (Gestion des utilisateurs, des profils, des rôles et privilèges, des tablespaces) et aussi de part son architecture complexe qui repose sur la notion d'instance et qui assure un traitement rapide, sécurisé, efficace des données. Aussi, Oracle possède son propre langage de définition de procédures SQL (Structured Query Language), le PL/SQL qui est assez simple a utilisé.

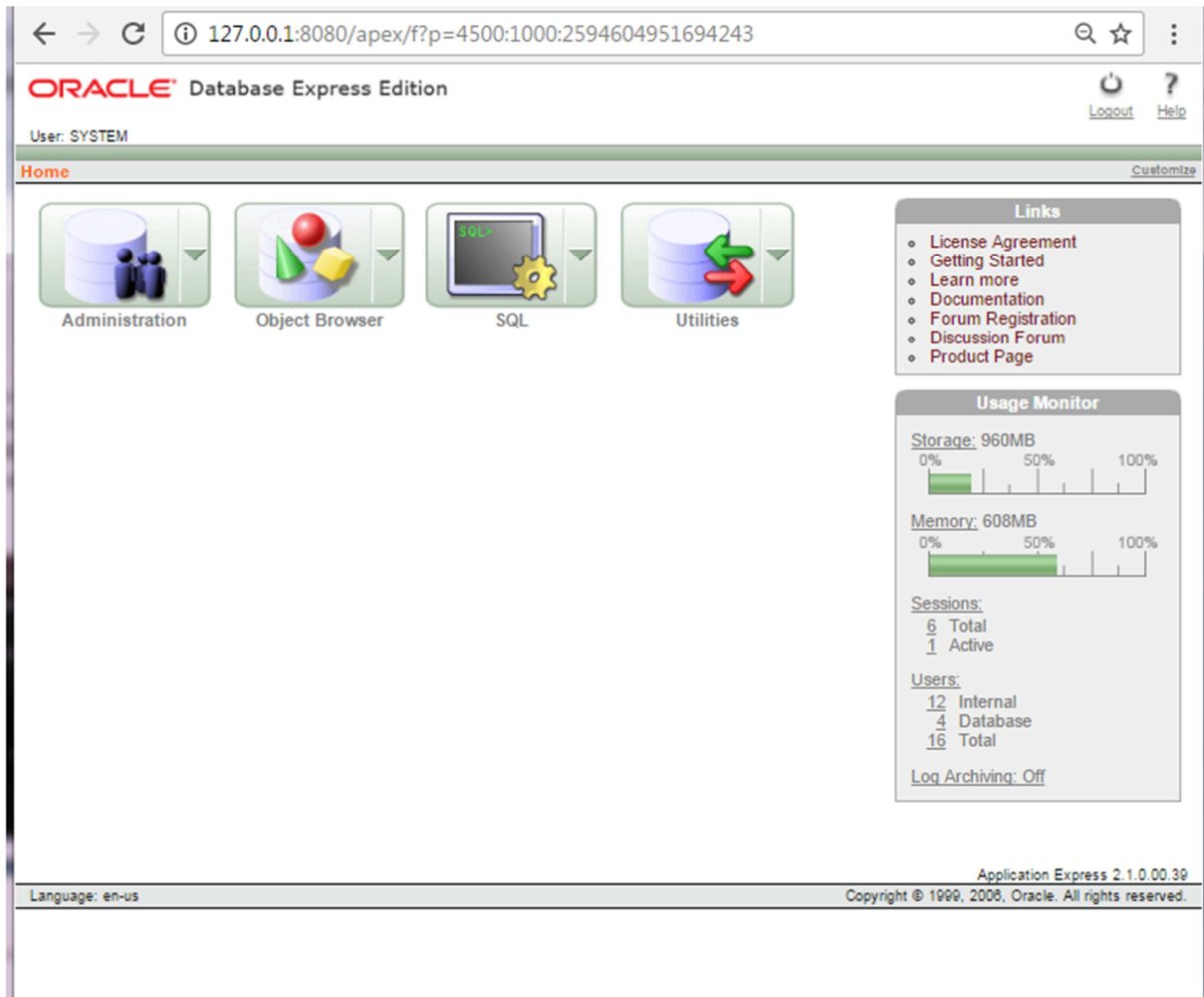


Figure III.2. Interface Oracle 10g Express Edition

I.2. Les langages de programmation :

I.2.1. Java :

Java est un langage de programmation informatique orienté objet créé par *James Gosling* et *Patrick Naughton* de Sun Microsystems. Mais c'est également un environnement d'exécution. Java peut être séparée en deux parties. D'une part, le

programme écrit en langage Java et d'autre part, une machine virtuelle (JVM) qui va se charger de l'exécution du programme Java. C'est cette plate forme qui garanti la portabilité de Java. Il suffit qu'un système ait une machine virtuelle Java pour que tout programme écrit en ce langage puisse fonctionner.

```
package bd;

import java.awt.EventQueue;

public class Fenetrep extends JFrame {
    String u,mdp1;
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTextField txt1;

    private JPasswordField pwd1;
    public Connection con;

    /**
     * Launch the application.
     */
    /*debut de l'application*/
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Fenetrep frame1 = new Fenetrep();
                    frame1.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

Figure III.3. Exemple d'un programme JAVA

I.2.2. SQL :

Le langage SQL (Structured Query Language) peut être considéré comme un langage d'accès normalisé aux bases de données. Il est aujourd'hui supporté par la plupart des produits commerciaux que ce soit par les systèmes de gestion de bases de données micro tel qu'Access ou par les produits plus professionnels tels qu'Oracle ou Sybase. Il a fait l'objet de plusieurs normes ANSI/ISO dont la plus répandue aujourd'hui est la norme SQL2 qui a été définie en 1992. Le succès du

Chapitre III : Implémentation & réalisation

langage SQL est dû essentiellement à sa simplicité et au fait qu'il s'appuie sur le schéma conceptuel pour énoncer des requêtes en laissant le SGBD responsable de la stratégie d'exécution. Le langage SQL propose un langage de requêtes ensembliste.

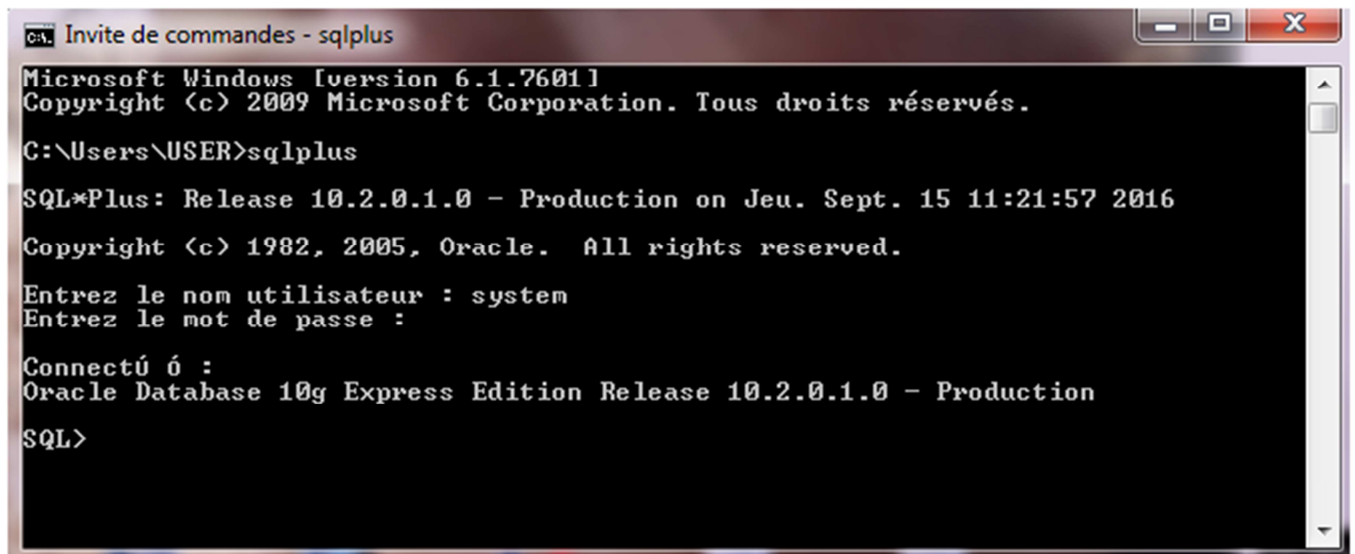


Figure III.4. Interface de l'invite de commandes SQL

II. Implémentation de la base de données fédérée :

Dans notre projet on a procédé par les configurations suivantes :

➤ **Configuration réseau :**

On a utilisé dans notre projet trois ordinateurs sous le même réseau

- ✓ « PC1 » aura l'adresse 192.168.8.x.
- ✓ « PC2 » aura l'adresse 192.168.8.y.
- ✓ « PC3 » aura l'adresse 192.168.8.z.

➤ **Configuration Oracle :**

On a créé cinq(5) bases de données, deux(02) dans un ordinateur « PC1 », Deux(2) dans un autre ordinateur « PC2 », et la base de données fédérée dans la machine « PC3 ».

◆ **Dans le « PC1 » :**

- ✓ La base de données du département informatique « depinfo »
- ✓ La base de données du département automatique « depauto »

◆ **Dans le « PC2 » :**

- ✓ La base de données du département électrotechnique « depet ».

✓ La base de données du département électronique « depele ».

◆ **Dans le « PC3 » :**

✓ La base de données fédérée « system ».

II.1 Création des liens :

Un lien de la base de données est une connexion entre deux serveurs de base de données physique qui permet à l'utilisateur d'y accéder comme une seule base de données logique.

Un lien de base de données est un chemin unidirectionnel d'un serveur à un autre. En effet, un utilisateur connecté à une BD A, peut utiliser un lien stocké dans la BD A pour accéder à la BD distante B et vice versa.

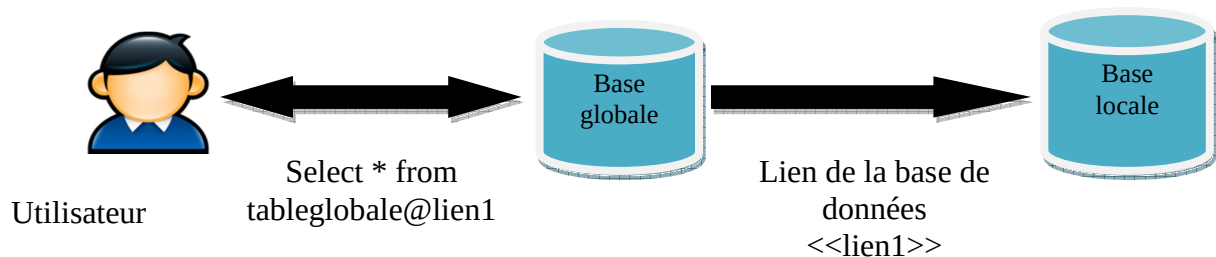


Figure III.5. Exemple de lien de la base de données

➤ Les liens internes de base de données sous Oracle :

◆ **Requête de création de lien :**

En créant un lien de BD, on doit indiquer le nom du compte auquel on se connecte, et le mot de passe de ce compte (depinfo/depinfo).

```
SQL> create database link lieninfo connect to depinfo identified by depinfo;
Lien de base de données créé.
```

Figure III.6. Requête de création de lien interne sous oracle

◆ **Requête d'interrogation avec le lien :**

Requête pour interroger une base de données interne reliée par « lien_ext_info »

```
SQL> show user
USER est "SYSTEM"
SQL> select * from etudiant_info@lieninfo;

ID_ETUD_IN NOM_ETUD_I PRENOM_ETU ADRESSE_ET DATE_N_I ANNEE_EN_COUR_INF NIU_E
-----
INFO2      MAKOUDI      NASSIMA  BOUARFA   12/01/87   2016 L2
028        name        prenom    hahh      03/07/83   2016 M2
Info1      AMMANI      HAYET    IFEKRANE  08/03/93   2016 M2
```

Figure III.7. Requête d'interrogation de lien interne sous oracle

➤ Les liens externes de base de données sous Oracle :

En créant un lien externe de BD, on doit indiquer le nom du compte auquel on se connecte, et le mot de passe de ce compte (depele/depele) en indiquant l'adresse IP.

◆ Requête de création de lien :

```
SQL> create database link lien_ext_ele connect to depele identified by depele
2 using '(<DESCRIPTION =
3      <ADDRESS_LIST =
4      <ADDRESS = <PROTOCOL = TCP><HOST = 192.168.8.101><PORT = 1521>>
5      )
6      <CONNECT_DATA =
7      <SERVICE_NAME = XE>
8      )
9      >'
10 ;
Lien de base de données créé.
```

Figure III.8. Requête de création de lien externe sous oracle

Dans notre cas on crée deux liens externes :

- ✓ Lien « lien_ext_info » de la base de données« system » vers la base de données « depinfo ».
- ✓ Lien « lien_ext_auto » de la base de données« system » vers la base de données « depauto ».
- ✓ Lien « lien_ext_ele » de la base de données« system » vers la base de données « depele ».
- ✓ Lien « lien_ext_elt » de la base de données« system » vers la base de données « depelt ».

◆ Requête d'interrogation avec le lien :

Requête pour interroger une base de données à distance reliée par lien_ext_ele

```
SQL> select nom_ele, prenom_ele, niv_act_ele, ann_uni_ele from etudiant_ele@lien_ext_ele;
NOM_ELE      PRENOM_ELE  NIV_ACT_EL  ANN_UNI_ELE
-----
xx           yy          m2          2016
kacel        lounas      m1          2016
haroun       lilia       13          2016
amani        hayet       m2          2016
amari        hakim       m1          2016
SQL>
```

```
SQL> select nom_elt, prenom_elt, niv_act_elt, ann_uni_elt from etudiant_elt@lien_ext_elt;
NOM_ELTT      PRENOM_ELTT  NIV_ACT_EL  ANN_UNI_ELTT
-----
akkouche      malika       m2          2016
ttt           yyy          m1          2016
SQL>
```

Figure III.9. Requête d'interrogation des liens externes sous oracle

II.2 Les vues matérialisées :

Pour la création des vues matérialiser on utilise les requêtes suivantes:

- Requête de création de la vue matérialisée avec les mis a jour :

```
SQL> CREATE MATERIALIZED VIEW v_etudiant
2 REFRESH COMPLETE START WITH sysdate+0/216000 NEXT sysdate + 2/21600
3 AS select
4 id_inf as id,
5 nom_inf as nom,
6 prenom_inf as prenom,
7 sexe_inf as sexe,
8 adr_inf as adresse,
9 d_nais_inf as dat_nais,
10 ann_bac_inf as ann_bac,
11 moy_bac_inf as moy_bac,
12 w_bac_inf as w_bac,
13 option_inf as opt,
14 niv_act_inf as niv_act,
15 ann_uni_inf as ann_uni from etudiant_info@lieninfo
16 union
17 select
18 id_aut as id,
19 nom_aut as nom,
20 prenom_aut as prenom,
21 sexe_aut as sexe,
22 adr_aut as adresse,
23 d_nais_aut as dat_nais,
24 to_number(null) as ann_bac,
25 to_number(null) as moy_bac,
26 to_char(null) as w_bac,
27 opt as opt,
28 niv_act_aut as niv_act,
29 cast(ann_act_aut as number(4)) as ann_uni from etudiant_aut@lienauto
30 union
31 select id_ele as id,nom_ele as nom,prenom_ele as prenom,
32 sex_ele as sexe,adr_ele as adresse,d_nais_ele as dat_nais,
33 ann_bac_ele as ann_bac,to_number(null) as "moy_bac",
34 w_bac_ele as w_bac,option_ele as opt,niv_act_ele as niv_act,
35 ann_uni_ele as ann_uni from etudiant_ele@lien_ext_ele
36 union
37 select id_elt as id,
38 nom_elt as nom,
39 prenom_elt as prenom,
40 sex_elt as sexe,adr_elt as adresse,
41 d_nais_elt as dat_nais,ann_bac_elt as ann_bac,
42 moy_bac_elt as "moy_bac",to_char(null) as w_bac,option_elt as opt,
43 niv_act_elt as niv_act,ann_uni_elt as ann_uni from etudiant_elt@lien_ext_elt
44 /
```

Figure III.10. Requête de création de la vue matérialisée

- Requête d'interrogation de la vue matérialisée :

```
SQL> select nom, prenom,niv_act from v_etudiant;
```

NOM	PRENOM	NIU_ACT
xx	yy	m2
chili	CHERIFA	M2
ttt	yyy	m1
akkouche	malika	m2

```
SQL> select nom, prenom,adresse,niv_act,ann_uni from v_etudiant;
```

NOM	PRENOM	ADRESSE	NIU_ACT	ANN_UNI
xx	yy	aa	m2	2016
chili	CHERIFA		M2	
ttt	yyy	mat	m1	2016
akkouche	malika	maatkas	m2	2016

Figure III.11. Requête d'interrogation de la vue matérialisée

III. Présentation de quelques interfaces :

➤ Page d'authentification :



Figure III.10. Interface de la page d'authentification.

➤ Page d'accueil :



Figure III.11. Interface de la page d'accueil

➤ Page de recherche avancée :

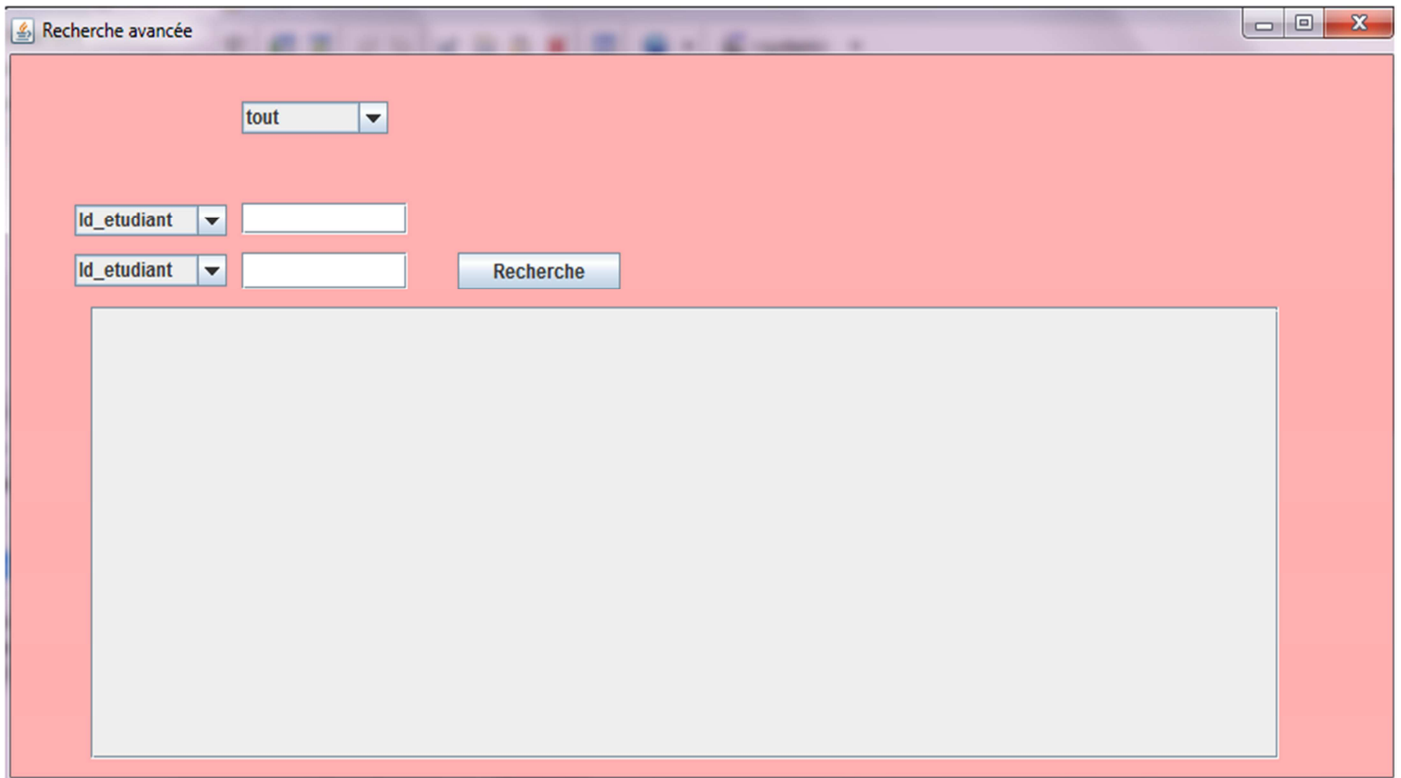


Figure III.12. Interface de la page de recherche avancée

Conclusion .

Dans ce chapitre on a présenté en premier lieu les outils et les langages de programmation utilisés, ensuite on est passé à l'implémentation de base de données globale et pour terminer on a présenté quelque interface graphique de notre application sur le system fédéré.

Conclusion générale

Conclusion générale :

Conclusion générale

Par ce que l'informatique devient du jour en jour un support incontestable de la quasi-totalité des domaines en offrant des services susceptibles d'améliorés et de faciliter la vie quotidienne, le thème de notre projet et purement pédagogique, il s'est porté sur la fédération des bases de données homogènes cas « scalarité de la faculté GEI ».

Le point de départ de la réalisation de ce projet était une récolte des informations nécessaires pour dresser un état de l'existant et présenter un aperçu sur la problématique. Par la suite, nous nous sommes intéressés à l'analyse et la spécification des besoins qui nous ont permis de distinguer les différents acteurs interagissant avec la solution visée. Puis la phase de conception dans laquelle nous avons fixé la structure globale de la solution.

Le dernier volet de notre projet était la partie réalisation qui a été consacrée à la présentation des outils de travail et des interfaces les plus significatives de l'application.

L'apport de ce travail a été d'une importance très considérable, en effet, il nous a permis de suivre une méthodologie de travail bien étudiée, de travailler en équipe dans des conditions difficiles, d'approfondir nos connaissances dans la manipulation de plusieurs logiciels.

Notre solution n'est pas unique même n'est pas la plus performante, car il existe des logiciels beaucoup couteux pour réaliser la fédération des bases de données tel que talend. Pour notre solution, nous avons visés la simplicité ainsi un coût minimal.

Enfin, espérant que ce travail sera de grands intérêts et un guide pour les nouvelles promotions.

Bibliographie

Bibliographie et Webographie :

Références bibliographiques :

- ◆ Apprendre SQL avec MySQL - Christian Soutou, Eyrolles 2006.
- ◆ SQL pour Oracle- application avec java, php et xml –Christian Soutou 3ème édition. EYROLLE.
- ◆ Bases de données (George Gardarin) Edition Eyrolles 2004.
- ◆ Cours IED Université de Cergy-Pontoise Master Informatique M1
- ◆ Oracle Database 2 Day Developer's Guide (11.2) December 2014, par Sheila Moore.
- ◆ Thèse « Une approche Hybride d'intégration de sources de données hétérogènes dans les datawarehouses » Par LAHMAR Fatima épouse BOULÇANE 2011.
- ◆ Thèse « définition d'un système générique de partage de données entre système existants » Par Mohammed Salah BOUMEDIENE, 2005.
- ◆ Thèse « Conception et réalisation d'une application web pour la gestion du prêt entre bibliothèques » par ZENDAOUI Fairouz, DRICI Cherifa,2010.

Webographie :

<http://www.oracle.com>.

<http://oracle.developpez.com>.

<http://oracle-base.com>.

<http://www.ibm.com>.

<http://www.commentcamarche.net/>.

<http://www.youtube.com/>.

Annexe

Annexe :

Quelques utilitaires sous oracle :

❖ Les liens sous oracle :

Pour interroger une BD distante, il faut créer un lien de base de données.

Un lien de base de données est un chemin unidirectionnel d'un serveur à un autre.

Lorsqu'un lien est référencé par une instruction SQL, Oracle ouvre une session dans la base distante et y exécute l'instruction. La session demeure ouverte au cas où elle serait de nouveau nécessaire.

En créant un lien de BD, on doit indiquer le nom du compte auquel on se connecte, le mot de passe de ce compte, et le nom de service associé à la base distante. En l'absence d'un nom de compte, Oracle utilise le nom et le mot de passe du compte local pour la connexion à la base distante.

```
CREATE [SHARED|PUBLIC|PRIVATE] DATABASE LINK NomLien
CONNECT TO .....
CURRENT_USER
User IDENTIFIED BY password
USING connect_string
```

Un lien est soit privé ou public. Seul l'utilisateur qui a créé un lien privé peut l'utiliser, alors qu'un lien public est utilisé par tous les utilisateurs de la base de données. Le lien partagé est propre à la configuration de serveur multithreaded.

La clause CONNECT TO active une session vers la base distante.

La clause CURRENT_USER crée un lien BD pour l'utilisateur courant. L'utilisateur doit disposer d'un compte valide dans la base distante.

La clause USING *connect_string* spécifie le nom de service d'une base distante. Les noms de service d'instances sont stockés dans le fichier de configuration intitulé *tnsnames.ora*. Ce fichier spécifie l'hôte, le port, et l'instance associés à chaque nom de service.

Des informations sur les liens de BD publics et privés, figurent respectivement dans les vues du dictionnaire de données : DBA_DB_LINKS et USER_DB_LINKS.

Après avoir créé les liens de bases de données, plusieurs objets : les vues, les synonymes et les procédures stockées, peuvent servir à cacher la distribution des données aux utilisateurs.

Annexe :

❖ Les vues :

Les vues peuvent fournir une transparence par rapport aux tables locales et distantes. Par exemple, supposons que la table *Employé* est sur une BD locale et la table *Département* est sur une BD distante. Pour rendre ces tables transparentes aux utilisateurs. Nous pouvons créer une vue dans la BD locale qui fait la jointure des données locales et distantes. Les utilisateurs accédant à cette vue n'ont pas besoin de savoir où les données sont stockées.

❖ Les synonymes :

Les synonymes sont des noms simples qui permettent d'identifier de façon unique dans un système distribué les objets qu'ils nomment. Les synonymes peuvent être créés pour différents objets : Tables, Types, Views, Snapshots, Procédures, Fonctions, Packages. Ils figurent dans le dictionnaire de données.

CREATE [PUBLIC] nom-synonyme
FOR [schéma.]nom-objet[@nom-lien-BD].

❖ Les Snapshot:

Cette option utilise des *snapshots* pour répliquer les données depuis une source maître vers plusieurs cibles. Les *snapshots* peuvent être en lecture seule (ang. *read-only*) ou mis à jour (ang. *updateable*). Avant de créer un *snapshot*, il faut d'abord créer un lien vers la base de données source.

Deux types de *snapshots* peuvent être créés : simples et complexes. Un *snapshot* simple ne contient pas de clause distinct, group by, connect by, de jointure multitables ou d'opérations set.

Le *snapshot* suivant est défini de façon à extraire les données maîtres et renouveler l'opération 7 jours plus tard.

```
CREATE SNAPSHOT LocalEmp
TABLESPACE data2
STORAGE (INITIAL 100K next 100K PCTINCREASE 0)
REFRESH FAST
START with SysDate
NEXT SysDate+7
AS SELECT * FROM Employes@RH_Lien;
```

Un REFRESH FAST utilise un *snapshot log*, pour actualiser le *snapshot*. Ce fichier se trouve sur le même site que la table maître. Dans le *snapshot log*, sont stockées les modifications intervenues sur la table maître. Ainsi, pour chaque mise à jour, seules

Annexe :

les modifications qui sont envoyées, et non l'ensemble des données. Par contre, un REFRESH COMPLETE est obligatoire pour les *snapshots* complexes.

Le *snapshot log* est à créer avant le *snapshot*:

```
CREATE SNAPSHOT LOG ON Employes
TABLESPACE DATA
STORAGE (INITIAL 10K NEXT 10K PCTINCREASE 0)
PCTFREE 5 PCTUSED 90;
```

❖ Les vues matérialisées :

Une vue matérialisée peut apporter plusieurs avantages au niveau performances.

Selon la complexité de la requête, on peut la remplir avec des changements incrémentiels, à l'aide du journal de vues matérialisées (MATERIALIZED VIEW LOG), au lieu de la recréer.

A l'inverse des *snapshots*, les vues matérialisées peuvent être utilisées directement par l'optimiseur, afin de modifier les chemins d'exécution des requêtes. Pour ce, il faut disposer du privilège QUERY REWRITE pour pouvoir réécrire la requête, et que QUERY_REWRITE_ENABLED soit TRUE (ALTER SESSION SET QUERY_REWRITE_ENABLED = TRUE).

Une vue matérialisée crée une table locale pour stocker les données, et une vue qui y accède.

➤ Exemple:

```
CREATE MATERIALIZED VIEW Ventes-par-Mois
TABLESPACE DATA_AGG
REFRESH COMPLETE
START WITH sysdate
NEXT sysdate+1
ENABLE QUERY REWRITE
AS
SELECT mois, SUM(montant)
FROM Ventes
GROUP BY mois;
```

La clause ENABLE QUERY REWRITE permet à l'optimiseur de rediriger les requêtes émises sur la table vers la vue matérialisée s'il le juge approprié.

La clause NEVER REFRESH empêche tout type d'actualisation de la vue matérialisée.

Les vues DBMS_MVIEW, ALL_MVIEW_ANALYSIS détaillent les caractéristiques des vues matérialisées créées.

❖ Modalités de rafraîchissement

➤ Méthodes de refresh :

Il existe 3 méthodes : FAST, COMPLETE ou FORCE

- ◆ **FAST REFRESH** (incrémental) c'est la méthode la plus efficace, elle utilise des journaux spécifiques traçant les modifications de la table maître : les LOGs. (Voir l'instruction CREATE MATERIALIZED VIEW LOG...)
- ◆ **COMPLETE** effectue le refresh complet en exécutant le SELECT de définition de la MV.
- ◆ **FORCE** effectue un FAST si possible, sinon un COMPLETE. C'est la méthode par défaut.
- ✓ **Rappel** : le fait que la VM soit initialisée avec des valeurs ou rafraîchie ou non n'est pas du tout lié à la méthode (y compris FORCE!) cela dépend respectivement de l'option 'build' pour l'initialisation et du mode, synchrone ou non pour le rafraîchissement...
méthode complète c'est la méthode par défaut. Toutes les lignes de la table maître sont (re) synchronisées. Ça peut donc coûter du temps mais ça fait les économies de place de la table log...

➤ Modes de refresh :

Globalement synchrone (répercuté en temps réel) ou asynchrone (mis dans une queue et traité en différé), mais il existe en fait 3 modes, 3 'fréquences' de refresh :

- ✓ **synchrone, sur commit** : clause ON COMMIT
par définition ne nécessite pas de commande REFRESH c'est la mise à jour COMMITée de la table maître qui déclenche automatiquement le REFRESH.
- ✓ **asynchrone, à la demande** : clause ON DEMAND. C'est le mode par défaut.
On utilise la procédure standard DBMS_MVIEW.refresh qui est publique (DBMS_MVIEW est un synonyme public sur SYS.DBMS_SNAPSHOT et il existe un GRANT EXECUTE TO PUBLIC ON SYS.DBMS_SNAPSHOT).
Ne nécessite donc pas de droit particulier pour être exécuté (pour le rafraîchissement d'une vue matérialisée dans un autre schéma voir les
- ✓ **asynchrone, cyclique** : clauses START WITH et NEXT, qui précise une date de début et une période de temps intermédiaire on peut changer de fréquence de refresh avec la commande ALTER :

Annexe :

```
SQL> ALTER MATERIALIZED VIEW mv1  
refresh complete  
start with sysdate next sysdate + n
```

Cette commande sans la clause NEXT, ne peut pas être dévoyée pour provoquer un refresh immédiat d'une vue non à jour.

❖ Oracle en réseau

- Oracle Net services fournit des solutions de connectivité dans des environnements distribués. Il est composé de:

1. Oracle Net.
2. Modules d'écoute/listeners.

Le fichier de configuration LISTENER.ORA contient :

- ✓ Son nom, par défaut LISTENER
 - ✓ Son adresse (HOST et PORT) : (ADDRESS= (PROTOCOL = TCP) (HOST = localhost) (PORT = 1521)
 - ✓ Les SIDs (Service ID) des BD guettées
3. Oracle Connection Manager.
 4. Outils de configuration et de gestion : Oracle Net Configuration Assistant,

Oracle Net Manager.

- Transparent Network Substrate(TNS) est une sous-couche d'Oracle Net qui reçoit les requêtes et émet des ouvertures ou fermetures de session, envoie les requêtes et reçoit des réponses.

Lorsque les hôtes qui supportent les bases de données Oracle sont connectés via un réseau, les bases peuvent communiquer via Net8 d'Oracle (précédemment appelé SQL*Net).

- Passerelle transparente (ang. Oracle Transparent Gateway) C'est la possibilité d'accéder à des objets non Oracle. La passerelle est exécutée sur l'hôte source qui contient la base à exploiter.