République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abderrahmane Mira de Bejaia

Faculté des Sciences Exactes
Département d'Informatique

Mémoire de fin de cycle
En vue d'obtention du diplôme de master professionnel en informatique
Option : Génie logiciel

Thème

Conception et réalisation d'une base de données repartie pour la gestion d'hébergement des résidences universitaires de Bejaia

Réalisé par :

BENMOUHOUB Cherifa

BOUDJETTOU Kenza

Présente devant le jury composé de :

Président : Mr. TOUAZI Djoudi

Examinateur : Mr. SAADI Mustapha

Encadreur : Mr. SALHI Nadir

ANNEE UNIVERSITAIRE 2020– 2021

Remerciements

Nous tenons à remercier en premier lieu dieu le tout puissant qui nous a donné le courage et la patience et qui a éclairé notre chemin pour achever ce travail.

Plus particulièrement on remercie monsieur SALHI NADIR pour la confiance qu'elle nous avons accordé, pour son encadrement continu, ses remarques constructives, ses orientations et ses conseils. Nous avons l'honneur et le privilège de travailler sous votre assistance et de profiter de votre expérience

Nous tenons à adresser nos vifs remerciements aux membres du jury pour s'intéresser au sujet et avoir accepté de lire ce mémoire.

Nos sincères remerciements à tous nos professeurs du département d'informatique de la faculté des sciences exacte de Bejaïa

Enfin, nous remercions tous ce qui nous a contribués de près ou de loin à la Réalisation de ce travail.

Dédicace

Je dédie ce mémoire à

A mes très ***chers parents*** *pour m'avoir donnés le goût aux études et m'avoir apportés un grand support moral lors de la rédaction de ce mémoire et tout au long de mes études, qu'ils trouvent ici l'expression de mon profond Amour. Que Dieu, le tout puissant, vous préserve et vous procure santé et langue vie afin que je puisse à mon tour vous combler*

A ***mon mari*** *HASSEN et ma* ***petite fille*** *KHAWLA pour votre patience et inoubliables sacrifices.*
A mes ***sœurs*** *HALIMA, KANZA et SALMA et mes* ***frères*** *ZAKARI et YOUNES.*

A ma ***belle-mère*** *SAHR et mon* ***père*** *RACHID. A mes* ***beaux-frères*** *Fayçal et RAZIK et leurs a leurs femmes KAHINA et SAIDA ainsi qu'à leurs enfants, a mes belles-sœurs SOUHILA, CHAFIA et LAMIA ainsi que leurs familles.*

Je ne me permettrais surtout pas d'oublier mes très chers amis(es). A mon ***amie*** *ANAYAT AMEL, ainsi toute sa famille. A mon* ***enseignante et amie*** *TIAB AMEL.A tous les étudiants Master Informatique de l'université d'Abed Rahman mira.*

À toute personne qui m'aime
À toute personne que j'aime
À tous ceux qui cherchent le savoir

CHERIFA

Dédicace

Je dédie ce mémoire à

A mes très ***chers parents*** *pour m'avoir donnés le goût aux études et m'avoir apportés un grand support moral lors de la rédaction de ce mémoire et tout au long de mes études, qu'ils trouvent ici l'expression de mon profond Amour. Que Dieu, le tout puissant, vous préserve et vous procure santé et langue vie afin que je puisse à mon tour vous combler*

A ***mes grand parents*** *et mes* ***sœurs*** *HANAN et MANAL et mes* ***frères*** *MOUHAND et ABDESSALAM.*

A ***mes tantes****, et* ***mes oncles*** *et leurs femmes ainsi qu'à leurs enfants. Et A* ***tous*** *mes* ***cousins*** *et* ***cousines.***

Je ne me permettrais surtout pas d'oublier ***mes très chers amis(es) :*** *SARA, WASSILA, SOUMIA, KENZA, HINOUCHE, SAMIA, MERIEM, SILINA, NIHAD*

A ***mes enseignants*** *que ce soit du* ***primaire****, du* ***moyen****, du,* ***secondaire*** *ou de* ***l'enseignement supérieur****. A tous les étudiants Master Informatique de l'université d'Abed Rahman mira*

À toute personne qui m'aime
À toute personne que j'aime
À tous ceux qui cherchent le savoir

KENZA

Table de matières

Chapitre 2 Technique de conception et de gestion des bases de données réparties

Liste des figures

Liste des tableaux

Liste des abréviations

SGBD : Système de Gestion de la Base de Données

DD/D : Dictionnaire de Données Directives

SQL : Structured Query Language

BD : Base de Donné

BDR : Base de Donné Répartie

ANSI : American National Standards Institute

SPARC : Scalable Processor Architecture

SGBDR : Système de Gestion de la Base de Données Répartie

GUI : Graphical User Interface

D.O.U : Direction des Œuvres Universitaires

D.O.U.B : Direction des Œuvres Universitaires de Bejaia

UP : Processus unifié

UML : Unified Modling Language

Introduction Générale

De nos jours l'informatique, joue un rôle très important dans tous les domaines d'activités de l'humanité, ceci grâce à l'internet qui a unifié ses activités

Le monde de l'informatique évolue très rapidement, alors que son but initial, était d'offrir des services satisfaisants, du point de vue vitesse d'exécution des tâches et obtention de statistiques plus précises. Actuellement, de nouveaux besoins sont apparus, toute organisation automatisée souhaite stocker et échanger ses informations qui sont géographiquement éloignées, ce qui rend la tâche de la collecte et de traitement d'une grande quantité d'informations dispersées très délicate, de ce fait, l'amélioration des systèmes d'informations est devenue une priorité pour les gérants des entreprises.

La solution qui s'impose est de distribuer les données et les organiser dans des bases de données sur différents sites de stockage. L'ensemble de ces sites constitue un système de bases de données réparties offrant la possibilité aux utilisateurs de manipuler les différentes bases via un réseau de manière transparente, comme dans une base de données globale.

L'objectif de notre travail consiste à résoudre et remédier aux différentes difficultés rencontrées par la D.O.U, pour contrôler, coordonner et éventuellement avoir des situations permanentes et actualisées des différentes opérations effectuées par le service d'hébergement au niveau des résidences universitaires. Pour cela, nous avons conçu et mis en œuvre une base de données répartie sous Oracle pour la gestion de l'hébergement des résidences universitaires de l'université de Bejaia.

Le manuscrit est structuré en 5 chapitres. Dans le premier, nous présentons un état de l'art sur les systèmes et les bases de données réparties avec leurs avantages et inconvénient, et les principes de leurs mises en œuvre. Dans le second chapitre, nous abordons les différentes techniques de conception et de gestion des bases de données réparties ainsi que les principes de la réplication. Le troisième chapitre donne une vue sur la structure de la direction des œuvres universitaires, plus précisément la section de l'hébergement. Le quatrième chapitre présente l'analyse conceptuelle de la solution proposée basée sur le processus unifié. Les étapes et les outils nécessaires pour l'implémentation de la solution proposée et la présentation de nouveau système sont présentés dans le dernier chapitre.

En fin, nous terminons avec une conclusion générale et quelques perspectives.

Chapitre1

Généralité sur les bases de données réparties

1.1 Introduction

Dans ces dernières années le besoin d'avoir accès à l'information et aux données est considérable vue la demande effectuée par les différents systèmes. A cet effet, les moyens envisagés par la technologie de l'informatique comme la puissance des micro-ordinateurs, les performances des réseaux et la baisse des coûts du matériel informatique ont permis l'apparition de la notion de base de données réparties.

Dans ce chapitre, nous essayerons de présenter l'état de l'art des bases de données réparties et leurs objectifs.

1.2 Evolution des bases de données réparties

Les entreprises modernes, de nos jours se démarquent des autres grâce à leur capacité à traiter les informations avec fiabilité et rapidité. Ainsi, la gestion des informations prend une place prépondérante dans le développement et l'atteinte des objectifs de l'organisation.

Un système d'information renferme l'ensemble des éléments participants à la gestion, au traitement, au transport et à la diffusion de l'information au sein de l'organisation. Très concrètement, il peut être très différent d'une organisation à une autre et peut recouvrir selon les cas, tout ou une partie des éléments suivants : [1]

- Base de données de l'entreprise ;
- Progiciel de gestion intégré (Entreprise Ressources Planning) ;
- Outils de gestion de la relation client (Customer Relationship Management) ;
- Interface réseau ;
- Serveur de données et système de stockage ;
- Serveur d'application ;
- Dispositif de sécurité.

1.2.1 Système réparti

Un système réparti est un ensemble composé d'éléments reliés par un système de communication, les éléments ont des fonctions de traitement processeurs, de stockage mémoire et de relation avec le monde extérieur capteurs et actionneurs.

Les différents éléments du système ne fonctionnent pas indépendamment, mais collaborent à une ou plusieurs tâches communes. Conséquence : une partie au moins de l'état

global du système est partagée entre plusieurs éléments, sinon on aurait un fonctionnement indépendant. [1]

1.2.2 Architecture des systèmes repartis

L'appellation système réparti recouvre diverses architectures depuis les architectures client-serveur jusqu'aux architectures totalement réparties.

L'architecture client-serveur considère deux types de processeurs généralement distincts :

- Les serveurs qui offrent un service à des clients, par exemple un serveur base de données ou un serveur d'imprimante.
- Les clients qui émettent des requêtes aux serveurs pour les besoins d'une application.

Dans l'architecture client-serveur, la quasi-totalité du système de gestion de la base de données (SGBD) se trouve sur le serveur, les processeurs clients ne disposant que des mécanismes de décodage et de transmission des requêtes vers ce serveur.

Une architecture totalement répartie est une généralisation de l'architecture client-serveur, les processeurs sont autonomes dans le sens où ils peuvent disposer d'un SGBD et assurer la pleine gestion d'une base de données locale. En plus, s'ils ne disposent pas des ressources nécessaires à une application qui leur est soumise, ils déterminent la localisation des données et des traitements qui leur sont nécessaires et établissent une coopération avec les processeurs détenteurs de ces ressources. [2]

1.2.3 Objectifs des systèmes répartis [3]

Il existe six objectifs principaux :

- **Autonomie de chaque site :** chaque site est responsable de l'intégrité de ses données, de sa propre sécurité et sa gestion interne. Chaque site doit rester fonctionnel même s'il ne peut pas communiquer avec les autres sites (en cas de panne réseau par exemple).
- **Continuité de service :** le système réparti doit être tolérant aux pannes, la défaillance d'un site particulier n'affecte pas le bon fonctionnement de tout le système.
- **Transparence vis à vis de la localisation des données :** les utilisateurs ne connaissent pas le schéma d'allocation des données.
- **Indépendance vis à vis du système d'exploitation :** la base de données répartie doit être indépendante du système d'exploitation utilisé dans les différents sites.

- **Traitement des requêtes distribuées :** le support de requêtes distribuées est essentiel, ainsi que les méthodes d'optimisation des exécutions de requêtes en milieu réparti.
- **Transparence vis à vis de la fragmentation :** les utilisateurs ne connaissent pas le schéma de fragmentation des données.

1.2.4 Inconvénients des systèmes répartis

Malgré tous les avantages que les systèmes répartis présentent, cela n'exclut pas l'apparition de certains inconvénients comme la complexité des mécanismes de décomposition de requêtes, la synchronisation des traitements et le maintien de la cohérence due à la répartition de la base de données sur plusieurs sites, ainsi que le cout induit par les transmissions des données sur les réseaux locaux.

1.3 Principes des bases de données réparties

Dans chaque site on trouve un système de gestion de la base de données (SGBD) local qui gère les données stockées sur ce site, plus une copie du SGBD distribué, un Dictionnaire de Données Directives (DD/D) qui contient des informations sur la localisation des données sur le réseau.

Si une requête est émise par un utilisateur ou un programme d'application, cette dernière doit passer par le SGBD distribué qui détermine si la transaction est locale avec des données situées sur un seul site, si c'est le cas, la transaction sera orientée vers le SGBD du site concerné.

Dans le cas contraire, c'est à dire la transaction est globale avec des données stockées sur plusieurs sites, elle sera exécutée par le SGBD global en collaboration avec les SGBD locaux. [2]

1.3.1 Définition

Une base de données répartie est un assemblage de différentes bases de données mises en relation les unes avec les autres à travers un réseau d'ordinateur.

1.3.2 Exemple

On peut imaginer deux sites de production de véhicules gérants plusieurs usines et un site de gestion pour une même société.

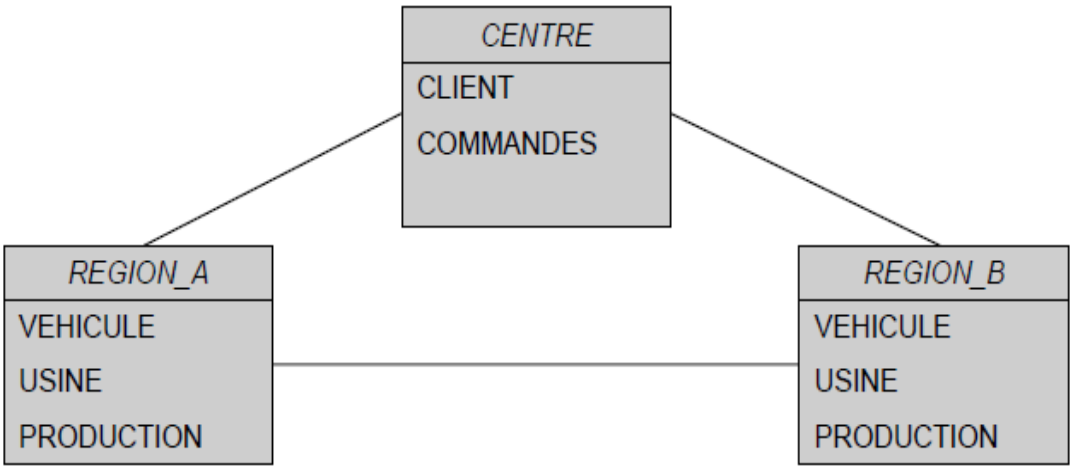

Figure 1 :Exemple de base de données repartie

.

1.3.3 Système de gestion d'une base de données répartie

Une base de données est une collection de données qui supporte les définitions, les manipulations de recherche, insertion, mise à jour et suppression de données. Ces opérations sont effectuées par un système logiciel, qui gère en général plusieurs bases de données, c'est le Système de Gestion de Bases de Données (SGBD) [4].

Le SGBD facilite le travail des utilisateurs en leur donnant l'impression que l'information est organisée comme ils le souhaitent. Il est composé de plusieurs couches [5] **:**

- le SGBD externe (*user interface handler*) : Pour interpréter les commandes utilisateurs.
- le contrôleur sémantique des données (*sémantic data controler*) : Pour vérifier qu'une requête d'un utilisateur peut être effectuée.
- le processeur de requêtes (*query processor*) : Il détermine une stratégie afin de minimiser le temps d'exécution d'une requête.
- le gestionnaire de transactions (*transaction manager*) : Il assure la coordination des différentes demandes des utilisateurs
- le gestionnaire de reprise (*recovery manager*) : Il s'occupe d'assurer la cohérence des données lorsque des pannes surviennent.
- le système de gestion des fichiers (*run-time support processor*) : Il gère le stockage physique de l'information. Il est dépendant du matériel utilisé.

Aujourd'hui, tout SGBD populaire est relationnel, c'est le cas d'Oracle, MySQL ou Microsoft SQL Server pour ne citer qu'eux. Pour un environnement distribué, on parle de SGBD réparti. Ce dernier garde les mêmes propriétés que le relationnel centralisé, mais ajoute

une gestion des bases de données distribuées (réparties). On notera l'existence des SGBD parallèles pour les architectures de même nom [4].

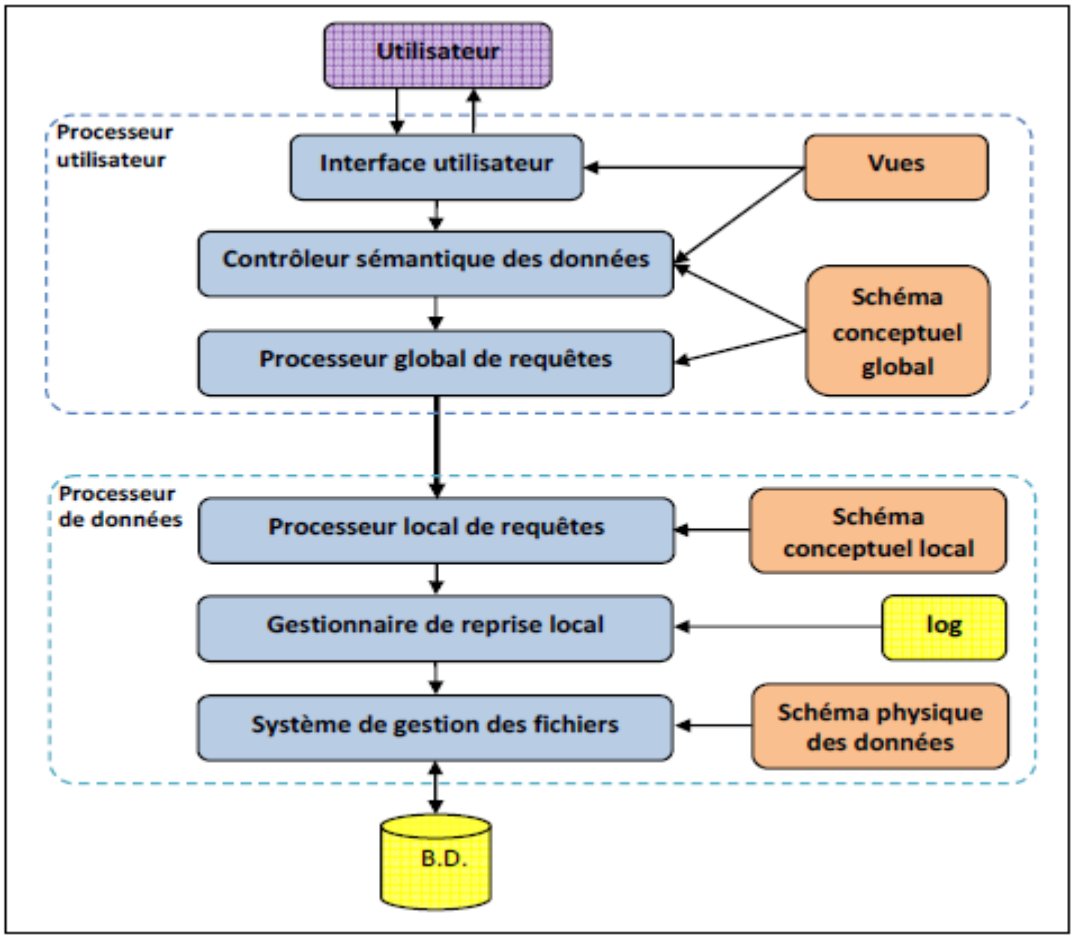

Figure 2 Architecture d'un SGBD réparti

1.3.4 Concepts de bases

1.3.4.1 Schéma local

Une base de données locale comporte un schéma géré par le SGBD local. Lors de la constitution d'une base de données répartie, chaque base de données locale rend visible une partie de la base aux sites clients [2].

1.3.4.2 Schéma global

Le schéma global permet de définir l'ensemble des types de données de la base. Il ignore les concepts d'implémentation. Il n'est pas forcément matérialisé, chaque base locale en implémente une partie [3].

La figure, ci-dessous, résume l'architecture globale d'une BD répartie avec les deux types de schémas :

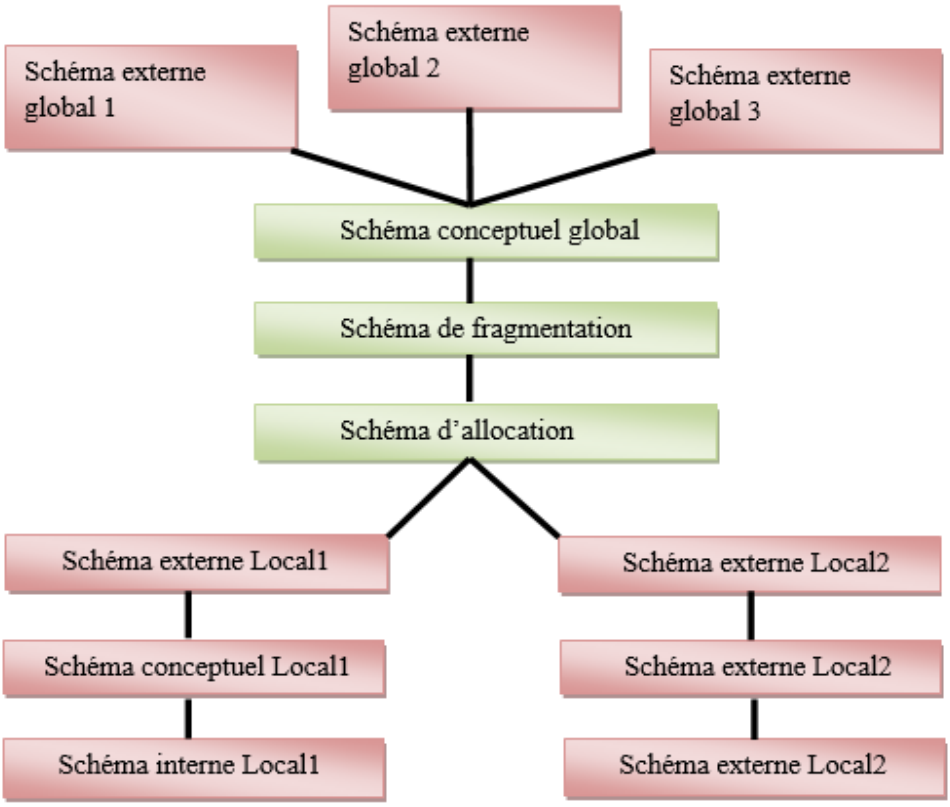

Figure 3 : Architecture d'une base de données repartie [6]

Comme toute BD, une BDR est décrite dans un dictionnaire de données sous la forme de schémas globaux distincts conformément à l'architecture ANSI/SPARC : [7]

- ✓ **Le schéma externe** : le niveau externe décrit les données sous forme de vues, chacune d'elles étant adaptée à une classe particulière d'utilisateurs ; un schéma externe, élaboré à partir du schéma conceptuel, peut naturellement mixer des données stockées dans différentes bases ;
- ✓ **Le schéma interne** : le niveau interne global n'a pas d'existence réelle mais fait place à des schémas internes locaux, répartis sur différents sites. Ces schémas correspondent à la description de l'organisation physique de la base, notamment la spécification de la fragmentation des données et la localisation de ces fragments ;
- ✓ **Le schéma conceptuel** : Dans ce schéma les données sont représentées sans prendre en compte des contraintes techniques ou de mise en forme ; toutes les données sont décrites dans ce schéma en utilisant un modèle de données, indépendamment de leur localisation dans le système réparti.

L'utilisateur accède aux données réparties à travers ces différents schémas en utilisant le langage *SQL*.

1.3.5 Décomposition et optimisation des requêtes

Un traitement réparti fait appel à des données gérées par des SGBD distincts. Ce traitement contient donc des requêtes qui correspondent à un ensemble d'opérations de recherche et de mises à jour sur des données de la BDR, formulées à partir d'un schéma externe global. Le SGBDR contrôle et analyse chaque requête puis la décompose en opérations locales afin d'être exécutées par les SGBD concernés. [2]

L'optimisation est donc indissociable de la requête car elle entre en jeu à tous les stades du traitement de la requête. Au niveau de la décomposition, l'optimisation permet de simplifier la requête et cela après avoir éliminé les sous requêtes inutiles ou bien répétées plusieurs fois.La figure ci-dessous, illustre le plan d'exécution répartie d'une requête :

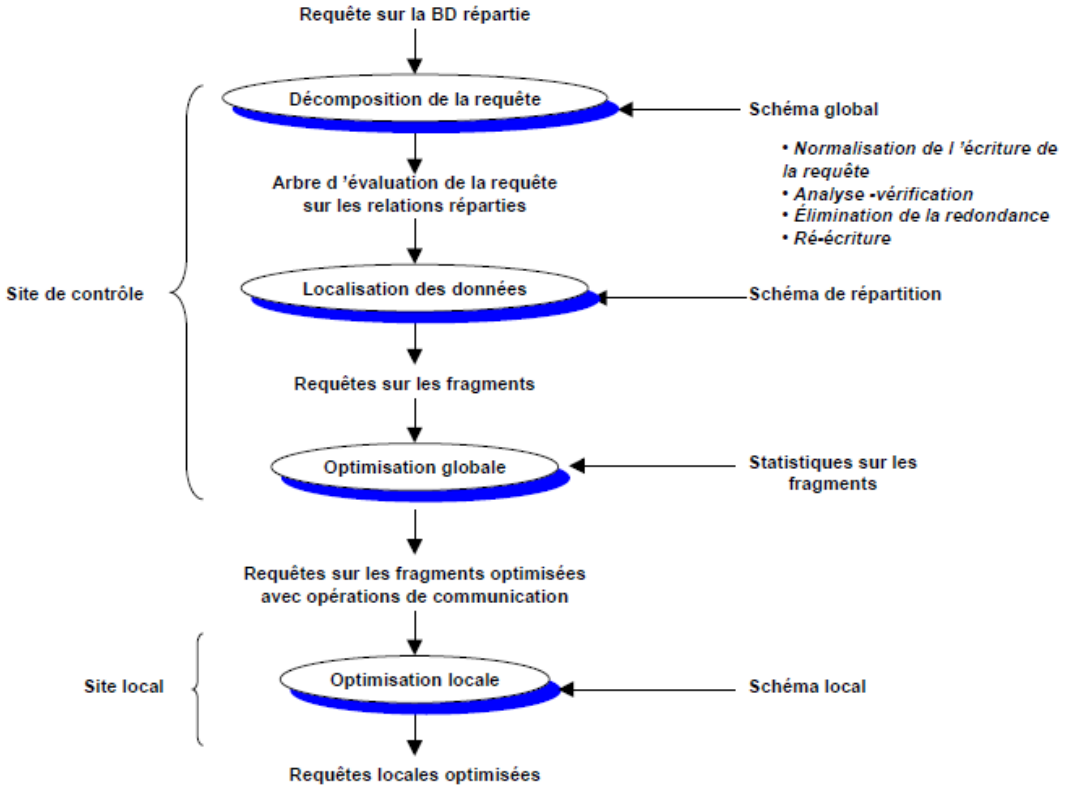

Figure 4 : Décomposition et optimisation d'une requête [5].

Le contrôle de l'intégrité des données est un des outils les plus importants d'une base de données assurant que les données ne soient pas modifiées ou détruites de façon illicite et

limitant les risques d'erreurs et de malveillance. Selon les contraintes qui ont été définies sur les relations et les attributs, différentes anomalies peuvent se déclencher dès qu'un accès aux données est effectué. L'intégrité des données se réfère à leurs cohérences par rapport à ce qui a été défini au départ [5].

1.3.6 Exécution répartie

Après que les requêtes sont décomposées en opérations locales par les SGBDR, elles seront exécutées par les SGBD concernés.

1.4 Utilisation d'une base de données repartie

Au niveau de la BDR, la transparence est un principe fondamental qui apparaît dans la localisation, le partitionnement et la duplication [8].

1.4.1 Transparence de la localisation

Les utilisateurs accèdent à la base de données soit directement par le schéma conceptuel soit indirectement au travers de vues externes. Mais ils n'ont en aucun cas les moyens d'accéder aux schémas locaux ni de préciser le site.

1.4.2 Transparence de partitionnement

Les utilisateurs n'ont pas à connaître les partitionnements de la base de données. Les utilisateurs ne doivent pas savoir si telle information est fractionnée, et ne doivent donc pas se préoccuper de la réunifier. C'est le système qui gère les partitionnements et les modifier en fonction de ses besoins, et c'est donc lui qui doit rechercher toutes les partitions et les intégrer en une seule information logique présenté à l'utilisateur.

1.4.3 Transparence de duplication

Enfin, les utilisateurs n'ont pas à savoir si plusieurs copies d'une même information sont disponibles. C'est le principe de transparence de duplication. La conséquence directe est que lors de la modification d'une information, c'est le système qui doit se préoccuper de mettre à jour toutes les copies.

1.5 La répartition des bases de données

1.5.1 But de répartition des données [9]

Les bases de données réparties ont une architecture plus adaptée à l'organisation des entreprises décentralisées.

- ✓ **Plus de fiabilité** : les bases de données réparties ont souvent des données répliquées. La panne d'un site n'est pas très importante pour l'utilisateur, qui s'adressera à un autre site.
- ✓ **Meilleures performances** : réduire le trafic sur le réseau est une possibilité d'accroître les performances. Le but de la répartition des données est de les rapprocher de l'endroit où elles sont accédées. Répartir une base de données sur plusieurs sites permet de répartir la charge sur les processeurs et sur les entrées/sorties.
- ✓ **Faciliter l'accroissement** : l'accroissement se fait par l'ajout de machines sur le réseau

1.5.2 Les inconvénients [9]

- ✓ Coût : la distribution entraîne des coûts supplémentaires en termes de communication, et en gestion des communications (hardware et software à installer pour gérer les communications et la distribution).
- ✓ Problème de concurrence.
- ✓ Sécurité : la sécurité est un problème plus complexe dans le cas des bases de données réparties que dans le cas des bases de données centralisées.

1.6 Architecture des bases de données réparties

1.6.1 Autonomie

L'autonomie se rapporte au degré avec lequel une des bases locales peut travailler indépendamment des autres. Ces dernières peuvent avoir trois types d'alternatives dans l'autonomie. [5]

- ✓ **L'intégration totale :** une image unique de la base de données globale est offerte aux différents utilisateurs. Un SGBD contrôle de bout en bout une requête d'un utilisateur même si la requête met en jeu les différentes bases locales et donc différents SGBD locaux.
- ✓ **La semi-autonomie** : Les SGBD locaux peuvent opérer indépendamment mais ils participent à une collection de bases qui coopèrent afin de partager leurs données.

- ✓ **L'isolation total :** Cette méthode pose de nombreux problèmes car une base locale ne connaît ni l'existence des autres bases ni la façon de se communiquer avec elles.

1.6.2 Relation entre machine

On distingue deux architectures principales dans les bases de données réparties :

- **Architecture client-serveur**

Dans cette architecture applicative, les programmes sont répartis en processus clients et serveurs qui communiquent à travers des requêtes et des réponses. Sur la machine cliente, les utilisateurs disposent d'une interface. Sur les serveurs, la gestion des bases de données est effectuée (analyse, optimisation des requêtes, et répartition). On peut distinguer deux types de clients :

- **Client lourd :** l'utilisateur est obligé de se connecter explicitement à tous les serveurs dont il a besoin pour la requête qu'il veut formuler [8].
- **Client léger** : l'utilisateur ne se connecte qu'à la base de données via un unique serveur. Le SGBDR se charge alors de gérer les différentes connexions que nécessitera la requête de l'utilisateur. Donc, il offre plus de transparence [8].

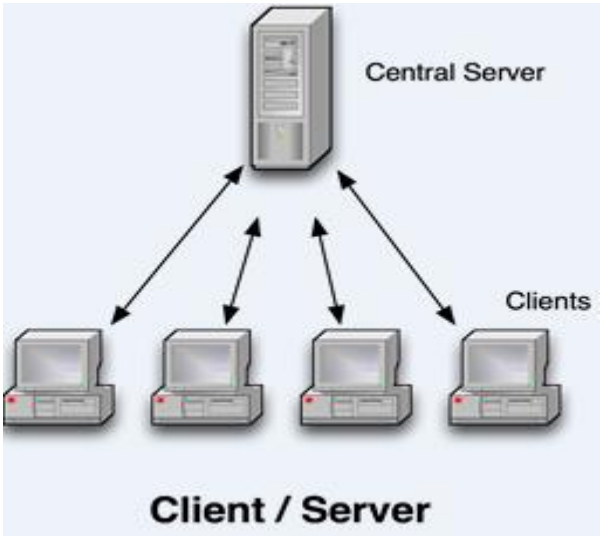

Figure 5 : Architecture client /serveur

- **Architecture Peer To Peer**

On désigne par ce terme un type de communication pour lequel toutes les machines ont une importance équivalente. Il n'ya pas de machine qui a une importance hiérarchique

par rapport aux autres. C'est aussi ce qu'on peut désigner comme l'architecture totalement répartie [5].

Chacune de ces architectures possède des avantages et des inconvénients. Le client serveur avec sa structure plus hiérarchique est très sensible aux problèmes de panne des serveurs, bloquant les utilisateurs (les clients), en revanche la prise de décision des serveurs est rapide.

Pour l'architecture peer-to-peer, comme les machines sont strictement équivalentes une Panne d'une machine ne fait que rendre le système un peut plus lent, mais cette architecture engendre énormément de communications pour toute décision.

1.6.3 Hétérogénéité

L'hétérogénéité peut intervenir à plusieurs niveaux. On peut penser aux problèmes matériels, aux protocoles réseaux, mais ce qui nous importe ici, ce sont les problèmes spécifiques aux bases de données. L'hétérogénéité peut exister au niveau de la représentation des données, au niveau du langage de requête ou au niveau du modèle des différentes bases de données (bases de données relationnelles, base de données objet).

1.7 Conclusion

Les bases de données réparties constituent un domaine important pour la gestion des informations stockées sur différent sites.

Après avoir présenté les base de donnée réparties du côté fonctionnelle. En outre, nous avons défini les principes de base de données et du système de gestion ainsi que les différentes architectures de cette dernière. Le chapitre suivant sera consacré à définir les techniques de conceptions et de gestion de base de données réparties.

Chapitre2

Technique de conception et de gestion des bases de données réparties

2.1 Introduction

Avant de concevoir une base de données répartie, il est nécessaire de bien comprendre les étapes de conception, car différentes méthodes de conception existent et chacune d'elles nous offre une approche très différente de l'autre.

Dans ce chapitre, nous présenterons les techniques utilisées pour la conception des bases de données réparties afin d'assurer le bon fonctionnement des données.

2.2 Conception d'une base de données répartie

La conception d'une base de données répartie se fait à partir d'un schéma de répartition qui se base sur des critères techniques et organisationnelles avec pour objectif de minimiser le nombre de transferts entre sites, les temps des transferts, le volume des données transférés, le temps moyen de traitement des requêtes, le nombre de copies de fragments, etc… [9].

2.2.1 Méthode 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*'.

2.2.1.1 Conception ascendante

L'approche se base sur le fait que la répartition est déjà faite, mais il faut réussir à intégrer les différentes BD existantes en une seule BD globale. En d'autres termes, les schémas conceptuels locaux existent et il faut réussir à les unifier dans un schéma conceptuel global [1].

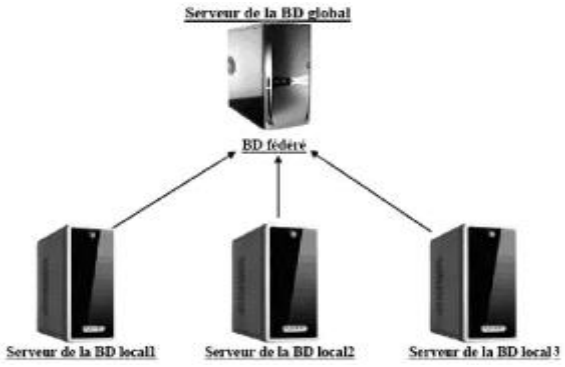

Figure 6 : Architecture de la conception ascendante

2.2.1.2 Conception 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 [5].

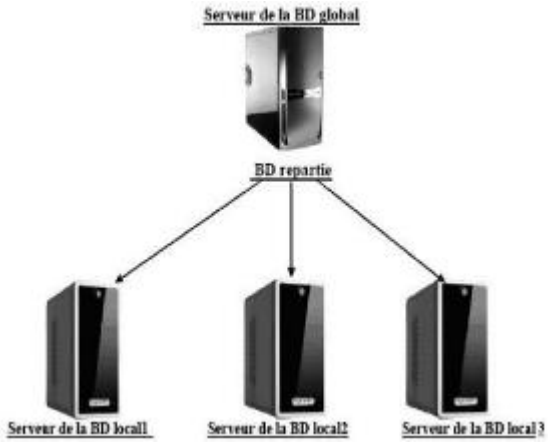

Figure 7 : Architecture de la Conception descendante

2.2.2 La fragmentation

2.2.2.1 Définition

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. La fragmentation peut être coûteuse s'il existe des applications qui possèdent des besoins opposés [7].

2.2.2.2 Les stratégies de fragmentation

- **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 [9].

Exemple

Client

nclient	Nom	ville
C1	Dupond	Paris
C2	Martin	Lyon
C3	Martin	Paris
C4	Smith	Lille

Client1

nclient	Nom	ville
C1	Dupond	Paris
C3	Martin	paris

Client2

nclient	nom	ville
C2	Martin	Lyon
C4	Smith	Lille

Client1 = Client where ville ="Paris" Client2 = Client where ville ≠″Paris"

Reconstruction Client = Client1 U Client2

Figure 8 : La fragmentation horizontale

- **La fragmentation verticale :**c'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 [7]

Exemple

Cde

Ncde	Nclient	Produit	Qté
D1	C1	P1	10
D2	C2	P2	20
D3	C3	P3	5
D4	C4	P4	10

Cde1

Ncde	nclient
D1	C1
D2	C2
D3	C3
D4	C4

Cde2

Ncde	produit	Qté
D1	P1	10
D2	P2	20
D3	P3	5
D4	P4	10

Cde1 = Cde (ncde, nclient) Cde2 = Cde (ncde, produit, qté)

Reconstruction Cde = [ncde, nclient, produit, qté] where Cde1.ncde = Cde2.ncde

Figure 9 : La fragmentation verticale

- **La fragmentation mixte** : elle résulte de l'application successive d'opérations de fragmentation horizontale et verticale sur une relation globale.

2.2.2.3 Objectif de la fragmentation

L'objectif de la fragmentation nous mène à parler des avantages de la distribution : où l'utilisation de petits fragments permet de faire tourner plus de processus simultanément, ce qui fait une meilleure utilisation des capacités du réseau d'ordinateurs. Le temps de recherche dans la BD diminue par rapport aux données stockées, ce qui augmente les performances de recherche et la disponibilité des données [10].

2.2.2.4 Les problèmes de la fragmentation [11]

La fragmentation peut causer certains problèmes, cités comme suit :

- ✓ La fragmentation peut être coûteuse s'il existe des applications qui possèdent des besoins opposés. On est en quelque sorte dans le cas d'une exclusion mutuelle qui empêche une fragmentation correcte. Par ailleurs, la vérification des dépendances sur différents sites peut être une opération très longue.
- ✓ La génération des fragments disjoints est un problème difficile.
- ✓ L'accès multiples aux fragments nécessite des opérations de jointure et d'union.
- ✓ La migration des données (conséquence d'une mauvaise fragmentation horizontale).

2.2.2.5 les règles de fragmentation [11]

Afin de bien implémenter la fragmentation, le concepteur de la BD doit appliquer certaines règles, à savoir :

- ✓ **La complétude :** pour toute donnée d'une relation *R*, il existe un fragment *Ri* de la relation *R* qui possède cette donnée.
- ✓ **La reconstruction** : pour toute relation décomposée en un ensemble de fragments *Ri*, il existe une opération de reconstruction. Pour les fragmentations horizontales,

l'opération de reconstruction est une union. Pour les fragmentations verticales c'est la jointure.

- ✓ **La Disjonction** : assure que les fragments d'une relation sont disjoints deux à deux, sauf dans le cas de la fragmentation verticale pour la clé primaire qui doit être présente dans l'ensemble des fragments issus d'une relation.

2.2.2.6 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 [8].

2.3 Techniques de répartition avancées

Dans le cas où la méthode classique de fragmentation-allocation ne s'avère pas satisfaisante, des techniques plus puissantes mais plus complexes à mettre en œuvre doivent être envisagées l'allocation avec duplication de fragments, l'allocation dynamique des fragments ou même la fragmentation dynamique.

2.3.1 Allocation avec duplication

Certains fragments peuvent être dupliqués sur plusieurs sites (éventuellement sur tous les sites) ce qui procure l'avantage d'améliorer les performances en termes de temps d'exécution des requêtes (en évitant certains transferts de données).Elle permet aussi une meilleure disponibilité des informations (connues de plusieurs sites), et une meilleure fiabilité contre les pannes. Par contre, l'inconvénient majeur est que les mises à jour doivent être effectuées sur toutes les copies d'une même donnée [5].

2.3.2 Allocation dynamique

Avec cette technique, l'allocation d'un fragment peut changer en cours d'utilisation de la BDR. Ça peut être le cas suite à une requête par exemple. Dans ce cas, le schéma d'allocation et les schémas locaux doivent être tenus à jour. Cette technique est une alternative à la duplication qui se révèle plus efficace lorsque la base de données est sujette à de nombreuses mises à jour [5].

2.3.3 Fragmentation dynamique

Dans le cas où le site d'allocation peut changer dynamiquement, il est possible que deux fragments complémentaires (verticalement ou horizontalement) se retrouvent sur le

même site. Il est alors normal de les fusionner. A l'inverse, si une partie d'un fragment est appelée sur un autre site, il peut être intéressant de décomposer ce fragment et de ne faire migrer que la partie concernée. Ces modifications du schéma de fragmentation se répercutent sur le schéma d'allocation et sur les schémas locaux [5].

2.4 La réplication

La réplication consiste à copier les informations d'une base de données sur une autre, afin de réduire la quantité de données transmises sur le réseau, et améliorer par conséquent les performances des requêtes. Plusieurs options de réplication peuvent être envisagées [9].

2.4.1 Objectifs de la réplication de donnée

- ✓ Disponibilité des données
- ✓ Respect de l'intégrité des données
- ✓ Optimisation des accès
- ✓ Administration centralisée
- ✓ Gestionnaires de données hétérogènes
- ✓ Autonomie locale

2.4.2 Principe de la réplication de données

Le principe de la réplication, qui met en jeu au minimum deux SGBD, est assez simple et se déroule en trois étapes : [12]

1. La base maître reçoit un ordre de mise à jour (INSERT, UPDATE ou DELETE).
2. Les modifications faites sur les données sont détectées et stockées dans un fichier ou une file d'attente en vue de leur propagation.
3. Le processus de réplication prend en charge la propagation des modifications à faire sur une seconde base dite esclave. Il peut bien entendu y avoir plus d'une base esclave.

2.4.3 les types de réplication

2.4.3.1 Réplication asymétrique

La réplication asymétrique distingue un site maître appelé site primaire, chargé de centraliser les mises à jour. Il est le seul autorisé à mettre à jour les données, et chargé de diffuser les mises à jour aux copies dites secondaires [7].

Le plus gros problème de la gestion asymétrique est la panne du site primaire. Dans ce cas, il faut choisir un remplaçant si l'on veut continuer les mises à jour. On aboutit alors à une

technique asymétrique mobile dans laquelle le site primaire change dynamiquement. On distingue l'asymétrie synchrone et l'asymétrie asynchrone :

- ✓ **Réplication asymétrique synchrone :** elle utilise un site primaire qui pousse les mises à jour en temps réel vers un ou plusieurs sites secondaires. La table répliquée est immédiatement mise à jour pour chaque modification par utilisation de trigger sur la table maître. [3]

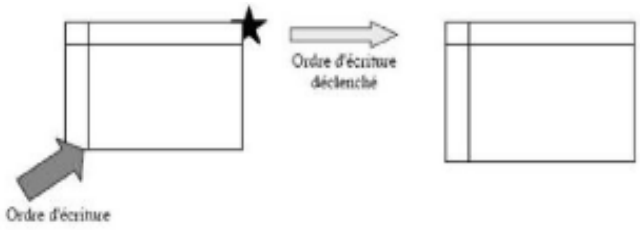

Figure 10 : Réplication asymétrique synchrone

- ✓ **Réplication asymétrique asynchrone :** elle pousse les mises à jour en temps différé via une file persistante. Les mises à jour seront exécutées ultérieurement, à partir d'un déclencheur externe, l'horloge par exemple. [3]

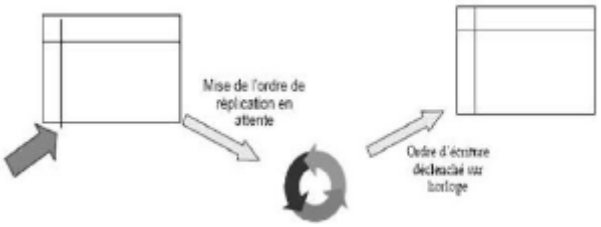

Figure 11 : Réplication asymétrique asynchrone

2.4.3.2 Réplication symétrique

A l'opposé de la réplication précédente, la réplication symétrique ne privilégie aucune copie, c'est-à-dire chaque copie peut être mise à jour à tout instant et assure la diffusion des mises à jour aux autres copies. [3]

Cette technique pose problème de la concurrence d'accès risquant de faire diverger les copies. Une technique globale de résolution de conflits doit être mise en oeuvre. On distingue la symétrie synchrone et la symétrie asynchrone :

- ✓ **Réplication symétrique synchrone :** Lors de la réplication symétrique synchrone, il n'y a pas de table maître. L'utilisation de trigger sur chaque table doit différencier une mise à jour client à répercuter d'une mise à jour par réplication. Cette technique nécessite l'utilisation de jeton. [3]

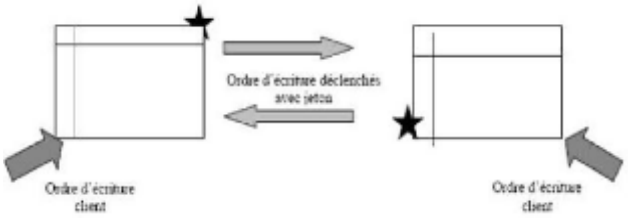

Figure 12 : Réplication symétrique synchrone

- ✓ **Réplication symétrique asynchrone :** Dans ce cas, la mise à jour des tables répliquées est différée. Cette technique risque de provoquer des incohérences de données.

Figure13 : Réplication symétrique asynchrone

2.4.4 vue matérialisée

Une vue matérialisée est un moyen simple de créer une vue physique d'une table. Elle correspond à une photo instantanée des données au moment de l'exécution de la requête. Le résultat de la requête est physiquement stocké dans la base de données.

Les vues matérialisées peuvent porter sur des tables, mais aussi des vues ou des vues matérialisées.

2.4.4.1 Objectifs des vues matérialisées

L'utilisation des vues matérialisées permet l'amélioration des performances d'accès et la réduction du trafic sur le réseau, elles sont mises à jour périodiquement, ce qui les rendent efficaces [13].

2.4.4.2 Mise à jour des vues matérialisées

Afin d'assurer une certaine cohérence des données, il faut mettre à jour les vues matérialisées et les tables périodiquement. Il existe trois façons de mises à jour qui sont la régénération complète, rapide et forcée :

- **Rafraîchissement complet :** Il va ré-exécuter la requête basée sur la table de base et remplace l'ensemble des données de la vue matérialisées par les données obtenues et ceci même si la table de base n'a pas été modifiée, selon le volume de données qui satisfait la requête, ce rafraîchissement peut être gourmant en ressources. [14]
- **Rafraîchissement incrémental :** Ce type de rafraîchissement est particulièrement efficace si les tables de base sont relativement peu modifiées. On considère que si plus de la moitié des lignes sont modifiées un rafraîchissement complet sera plus efficace. [14]
- **Rafraîchissement forcé :** Dans ce type de rafraîchissement, lorsqu'une régénération rapide n'est pas possible, alors une régénération complète est exécutée.

2.4.5 les avantages de la réplication

Les avantages de la réplication sont assez nombreux, selon le type on trouve :

_l'allégement du trafic réseau en répartissant la charge sur divers sites. Par conséquent, l'accès aux données est très rapide.

_l'amélioration des performances des requêtes.

_ La résistance aux pannes par l'augmentation de la disponibilité des données.

2.5 Gestion des données réparties

Le problème de la fragmentation avec ou sans duplication concerne principalement les mises à jour tandis que le problème des coûts des communications concernent surtout les requêtes.

2.5.1 Mise à jour des données distantes

2.5.1.1 Requête répartie en écriture [3]

Lorsque la mise à jour des données est répercutée sur plusieurs fragments, la requête de mise à jour doit être sûre que chaque requête locale peut être effectuée avant de valider l'ensemble. Ce protocole de validation se nomme *COMMIT* à deux phases et garantit le tout ou rien dans la base répartie. Le principal inconvénient de cette validation est sa lourdeur.

2.5.1.2 Requête répartie en lecture

Lors de l'exécution d'une requête en lecture, la base de données répartie va décomposer la requête globale en sous requêtes locales à l'aide des métadonnées de distribution.

2.5.2 Contraintes déclaratives [3]

2.5.2.1 Contraintes locales

Les contraintes locales sont des contraintes placées sur un seul site. Ces contraintes sont donc stockées dans le dictionnaire de chaque site.

2.5.2.2 Contraintes globales

Les contraintes globales doivent être placées sur la relation globale. Il est important de comprendre qu'il n'est pas possible de matérialiser simplement une contrainte globale. D'un point de vue simpliste, nous dirons qu'il n'est pas possible de créer des contraintes sur des vues, mais il est plus important de comprendre qu'une contrainte globale doit être placée dans plusieurs dictionnaires. Le schéma global n'étant pas physiquement implémenté, il n'est pas possible de mettre en place ces contraintes de manière déclaratives.

2.6 Conclusion

Pour une meilleure conception des bases de données, la distribution des données sur différents sites est la solution idéale.

Dans ce chapitre, nous avons présenté les bases de données réparties de côté conceptuelle où nous avons défini les différentes approches de conception (ascendante, descendante) et les types de fragmentations ainsi que les méthodes de réplications des données. Dans le chapitre suivant, nous allons entamer la présentation de l'organisme d'accueil.

Chapitre3

Etude préliminaire du système

3.1 Introduction

Notre projet consiste à concevoir et à réaliser une base de données répartie pour la gestion de l'hébergement des résidences universitaires. Pour ce faire, il s'avère nécessaire de présenter l'organisme d'accueil qui est le service d'hébergement au niveau de la direction des œuvres universitaires (*D.O.U*) de Bejaia ainsi que les résidences universitaires afin de comprendre les activités principales qu'elles exercent.

3.2 Présentation de la D.O.U

Placée sous la tutelle de l'office national des œuvres universitaires et du ministère de l'enseignement supérieur et de la recherche scientifique. Elle est créée en vertu de l'arrêté interministériel du 22/12/2004 portant la création des directions des œuvres universitaires. La direction est chargée de la gestion des Résidences Universitaires de la wilaya en assurant au profit des étudiants différentes prestations de services en matière d'hébergement, transport, restauration, activités scientifiques, culturelles et sportives, prévention sanitaire, et bourse.

3.3 Missions et activités de la D.O.U

Sa principale mission est de promouvoir la politique de l'état visant à améliorer la vie quotidienne de l'étudiant universitaire à travers un ensemble de services tels le soutien financier, l'hébergement, le transport, la restauration, et tout cela dans un cadre agréable à l'intérieur des Résidences Universitaires grâce à un riche programme d'activités culturelles, scientifiques et sportives, sans oublier bien sûr une protection sanitaire assurée par une équipe médicale et paramédicale professionnelle . Elle assure plus précisément les activités suivantes :

- La prise en charge totale en matière d'application de la politique nationale des œuvres universitaires
- Le contrôle et la coordination entre les résidences universitaires.
- De veiller à l'amélioration des conditions de vie de l'étudiant à l'intérieur de la résidence universitaire.
- La gestion des bourses.
- L'élaboration du plan de transport des résidences universitaires et le suivi de son exécution.
- La promotion des activités scientifiques, culturelles, sportives et loisirs.
- Assurer hygiène et sécurité.

- L'accueil et l'orientation des nouveaux bacheliers

3.4 Organigramme de la Direction des Œuvres Universitaires de béjaia

L'organigramme de la direction des œuvres universitaires est présenté dans le tableau suivant :

Direction des œuvres universitaires

Département du Contrôle et de la Coordination	**Département des bourses**	**Département des ressources humaines**	**Département des finances et des marches public**
Service du transport	Service d'attribution des bourses	Service de la gestion des carrières	Service du budget et de comptabilité
Service de la restauration	Service de renouvellement	Service de la formation et de perfectionnement	Service de marche publique
Service de l'hébergement			Service du suivi des opérations de construction et de l'équipement
Service des activités culturelles scientifique et sportive			

Tableau 1: Organigramme de la D.O.U.B

3.5 L'organisation de la D.O.U

La direction des œuvres universitaires de béjaia est composée de 04 départements selon le schéma suivant :

1. **Département Contrôle et Coordination**

Département Contrôle et Coordination comporte quatre Services :

- Service Activités Culturelles, Sportives et Scientifiques.
- Service Hébergement.
- Service Restauration.
- Service Transport.

Ce département est chargé de :

- Elaborer les plans de transport universitaire concernant les résidences universitaires rattachées à la DOU et suivre leur mise en œuvre.
- Suivre, contrôler et coordonner les activités d'œuvre universitaires assurées par les résidences universitaires rattachées à la DOU.
- Proposer toute mesure de rationalisation de l'utilisation des moyens humains, matériels et financiers consacrés aux activités des œuvres universitaires.
- Examiner les programmes d'activités scientifiques, culturelles et sportives et veiller au suivi de leur application après leur approbation par le directeur de la direction.

2. Le Département des Bourses :

Le département des bourses comporte Deux Services :

- Service Attribution des Bourses.
- Service Renouvellement des Bourses.

Il est chargé de :

- Assurer le traitement et le suivi des dossiers des étudiants bénéficiaires de bourses.
- Assurer le paiement régulier des bourses.
- Assurer le traitement et la prise en charge des bourses des étudiants étrangers.

3. Département des Finances et des Marchés Publics

Ce département comporte trois Services :

- Service de budget et de la comptabilité.
- Service des marchés publics.
- Service du suivi des opérations de construction et des équipements.

Il est chargé de :

- Suivre et gérer les traitements du personnel relevant de la direction des œuvres universitaires.
- Prise en charge et le suivi de l'exécution des marchés publics relatifs au transport et à la restauration.
- Suivi des opérations de construction et d'équipement des résidences universitaires, et ce en collaboration avec les services compétents.

4. Département des Ressources Humaines

Ce département deux Services :

- Service de la gestion des carrières.

- Service de la formation et du perfectionnement.

Il est chargé de :

- Gérer la carrière des personnels relevant de la direction des œuvres universitaires.
- Assurer la mise en œuvre des plans de formation et du perfectionnement des personnels relevant de la direction des œuvres universitaires.

3.6 Les résidences de la direction des œuvres universitaires

La D.O.U. de Bejaia est composée de 8 résidences :

- ✓ Résidence TargaOuzamour.
- ✓ Résidence Ihaddadene.
- ✓ Résidence 1000 lits.
- ✓ Résidence 17 Octobre 1961.
- ✓ Résidence Aamriw.
- ✓ Résidence Ireyahen.
- ✓ Résidence nouvelle pépinière.
- ✓ Résidence Berchiches.

Chaque résidence est dotée de plusieurs structures d'accompagnement.

3.6.1 Activités principales des résidences

Les résidences universitaires sont des établissements publics à caractère administratif sous tutelle de l'office national des œuvres universitaire, dépendant du ministère de l'enseignement supérieur et de la recherche scientifique, à vocation de prestation de service, les activités principales de ces établissements sont :

- ➢ Assurer l'hébergement et le suivi des dossiers des étudiants (mission de service d'hébergement).
- ➢ Assurer la restauration des résidant (mission de service restauration)
- ➢ Assurer la prévention sanitaire et le suivi médical des résidants.
- ➢ Amélioration du bien-être des étudiants par l'animation d'activités culturelles et de communication), (mission de service des activités culturelles et sportives).
- ➢ Assurer le transport des étudiants.

- Assurer la gestion administrative de l'établissement, ainsi que la coordination de différents services.

3.6.2 Organigramme d'une résidence

La figure ci-dessous représente l'organigramme d'une résidence :

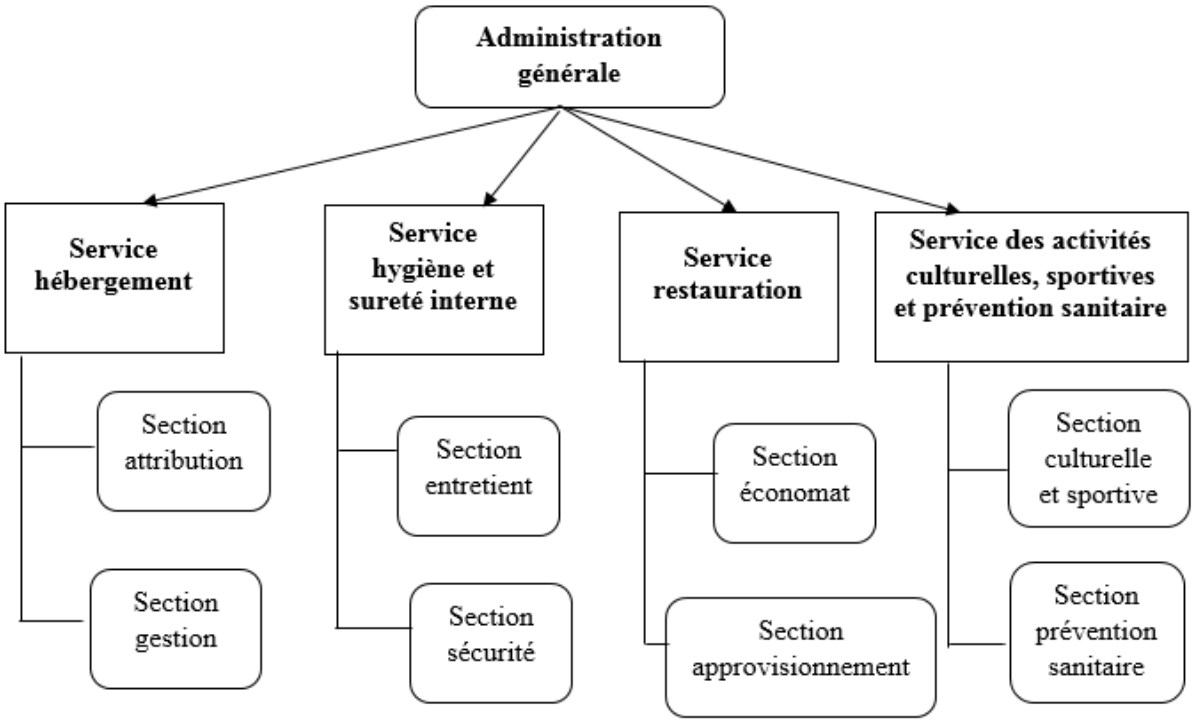

Figure 14 : Organigramme d'une résidence

3.7 Analyse du système

Le service d'hébergement effectue plusieurs tâches qui se récapitulent comme suit :

- ✓ Effectuer les inscriptions des nouveaux bacheliers reçus. la réinscription des anciens étudiants se fait au niveau de chaque résidence.
- ✓ Etablissement des listes globales des étudiants et leur répartition par résidence, et par bloc.
- ✓ Etablissement des statistiques sur l'état d'hébergement des résidences (nombre de places libres, les abandons, ... etc.).
- ✓ Contrôle des dossiers.

Parmi les problèmes qui existent après l'analyse de ce système, on peut citer :

- ✓ Le transfert de données entre la D.O.U et les différentes résidences se fait manuellement ;
- ✓ La répartition des étudiants d'une manière arbitraire, ce qui engendre l'augmentation de l'enveloppe budgétaire alloue au transport.
- ✓ La perte de temps ;
- ✓ non disponibilité des informations au bon moment ;
- ✓ La D.O.U dispose d'un réseau informatique à haut débit mais très mal exploité. En fait, il n'existe aucune application ou logiciel fonctionnant sous réseau ;
- ✓ L'inconsistance et l'incohérence des informations.

3.8 Présentation de la solution

Afin de remédier aux problèmes cités ci-dessus. Nous avons proposé de concevoir et de réaliser une base de de données répartie en se basant sur la fragmentation horizontale et le développement d'une application de gestion de l'hébergement universitaire permettant de :

- ✓ Automatiser les transferts de données ;
- ✓ Affecter les étudiants aux différentes résidences selon le critère de spécialité et nombre de place.
- ✓ Assurer un accès facile et rapide aux donnée ;
- ✓ 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.
- ✓ Gain financière, en réduisant le nombre de bus de transport universitaire.

3.9 Conclusion

Apres avoir présenté l'organigramme d'accueil et différentes tâches accomplies par le service de l'hébergement nous avons remarqué que ce service souffre de plusieurs problèmes liés particulièrement à une mauvaise gestion de l'hébergement, à savoir principalement, le traitement manuel et la centralisation des données au niveau de la D.O.U. Le chapitre suivant sera consacrée à la partie analyse et conception de notre application.

Chapitre4

Analyse et conception

4.1 Introduction

Dans ce chapitre, nous suivrons les étapes nécessaires pour concevoir une base de données repartie pour la gestion d'hébergement des résidences universitaires. Pour cela, dans cette conception, nous avons opté pour la méthode UP (processus unifié), cette dernière qui utilise UML (Unified Modling Language) comme langage de modélisation qui offre une stabilité et une flexibilité marquante grâce à l'utilisation des diagrammes.

4.2 Processus unifié [15]

Un processus unifié est un processus de développement logiciel construit sur UML ; il est itératif et incrémental, centré sur l'architecture, conduit par les cas d'utilisation et piloté par les risques.

La gestion d'un tel processus est organisée suivant les 4 phases suivantes : préétude (*inception*), élaboration, construction et transition.

Ses activités de développement sont définies par 5 disciplines fondamentales qui décrivent la capture des besoins, l'analyse et la conception, l'implémentation, le test et le déploiement.

4.3 Analyse et spécification des besoins

La phase consiste à qualifier les besoins fonctionnels et d'analyser l'existant afin de fournir le cahier des charges des spécifications techniques détaillées. Elle contient principalement les spécifications fonctionnelles détaillées et les spécifications techniques générales en adéquation avec la cible du client.

4.3.1 Spécification des besoins [16]

UP distingue deux types de besoins :

- ✓ les besoins fonctionnels qui conduisent à l'élaboration des cas d'utilisation.
- ✓ les besoins non fonctionnels (techniques) qui aboutissent à la rédaction d'une matrice des exigences de ce système pour sa réalisation et son bon fonctionnement.

4.3.1.1 Besoins fonctionnels

- ✓ **Authentification** : les fonctionnaires de l'application doivent s'authentifier pour accéder à leurs interfaces.
- ✓ **Dirigeant :** l'application donne au dirigeant la possibilité de faire les tâches suivantes : consulter, ajout, modification des étudiants.

- ✓ **Administrateur** : l'application donne à l'administrateur la possibilité de faire les tâches suivantes :
  - ajout, modification des résidences.
  - Consulter les informations relatives aux étudient.

4.3.1.2 Besoins non fonctionnels

- Existence d'un lien dynamique entre les serveurs de données basé sur le système de base de données répartie.
- L'application doit optimiser les traitements et l'utilisation de ses ressources (autonomie batterie, processeur, mémoire disponible….).

4.3.2 Analyse des besoins

Après l'étude des besoins fonctionnels cités précédemment, on peut passer à l'étape de formalisation de ces besoins à l'aide de diagramme de contexte et de diagramme de cas d'utilisation.

4.3.2.1 Diagramme de contexte

L'objectif du digramme de contexte consiste à représenter le système à modéliser et les différents acteurs qui interagissent avec ce system.

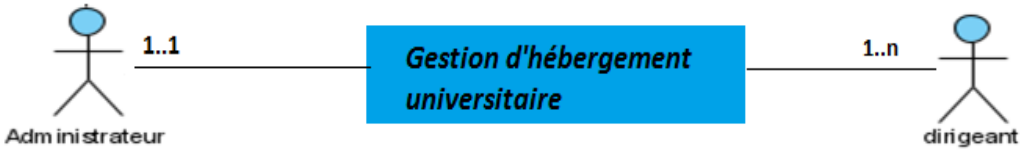

Figure 15 : Diagrammes de contexte

4.3.2.2 Digramme de cas d'utilisation

a) Identification des cas d'utilisation

Un cas d'utilisation correspond à un certain nombre d'actions que le système devra exécuter en réponse à un besoin d'un acteur. Un cas d'utilisation doit produire un résultat observable pour un ou plusieurs acteurs ou parties prenantes du système.

Liste des cas d'utilisation	**Désignation**
- S'authentifier	L'administrateur et le dirigeant doivent s'authentifier avant d'accéder à leur espace personnel.
- Ajouter des étudiants	L'administrateur ajoute de nouveaux étudiants au system.

- Modifier des étudiants	Lorsque les informations concernant les étudiants changent, le système permet leur modification.
- Ajouter des résidences	Administrateur a la possibilité d'ajouté une nouvelle résidence dans le système.
- Modifier des résidences	Lorsque les informations concernant résidence changent, le système permet leur modification
- Consulter les informations relatives aux étudiants	L'administrateur a le droit de consulter toutes les informations concernant les étudiants

Tableau 2 : Liste des cas d'utilisation

b) Diagramme des cas d'utilisation

✓ Diagramme de cas d'utilisation : **«** Authentification**»**

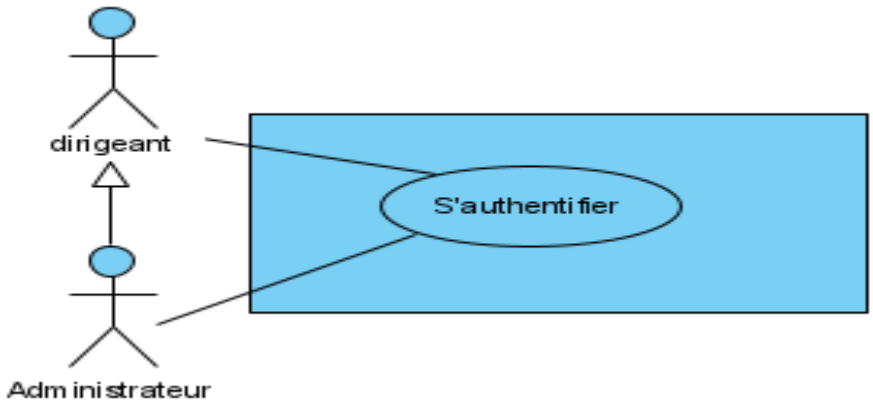

Figure 16 : Diagramme de cas d'utilisation « Gestion d'authentification »

- l'administrateur et le dirigeant du système saisissent dans une interface d'authentification un login et un mot de passe.
- Si les informations sont correctes le système lui donne accès à son espace personnel, sinon, le système affiche « erreur » et retourne à l'interface d'authentification.

✓ Diagramme de cas d'utilisation : **«** Gestion des étudiants**»**

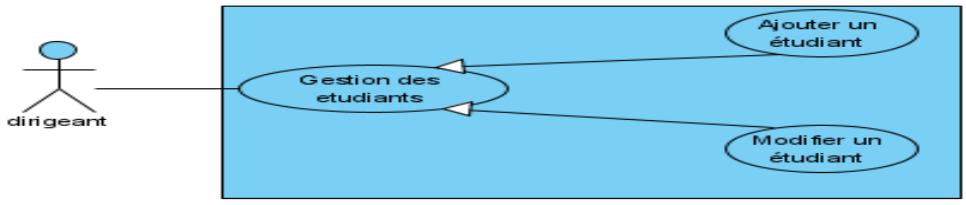

Figure 17 : Diagramme de cas d'utilisation « Gestion des étudiants)

- Après l'authentification du dirigeant avec sucée.
- Le système doit lui renvoyer l'interface dans laquelle il peut gérer les étudiants (ajout, modification).
- Si les données sont correctes, il les enregistre dans sa base, puis un message de confirmation sera envoyé.
- Sinon, il affiche un message d'erreur.

✓ Diagramme de cas d'utilisation : **«** Gestion des résidences**»**

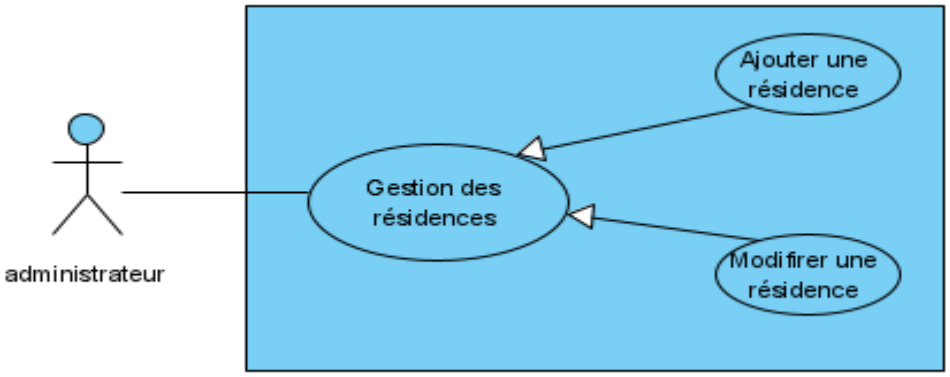

Figure 18 : Diagramme de cas d'utilisation « Gestion des résidences»

- Après l'authentification de l'administrateur avec sucée.
- Le système doit renvoyer à l'administrateur l'interface dans laquelle il peut gérer les résidences (ajout, modification).
- Si les données sont correctes, il les enregistre dans sa base, puis il affiche un message de confirmation sera affiché.
- Sinon, il affiche un message d'erreur.

✓ Diagramme de cas d'utilisation : **«** Consulter les information relative aux étudiants**»**

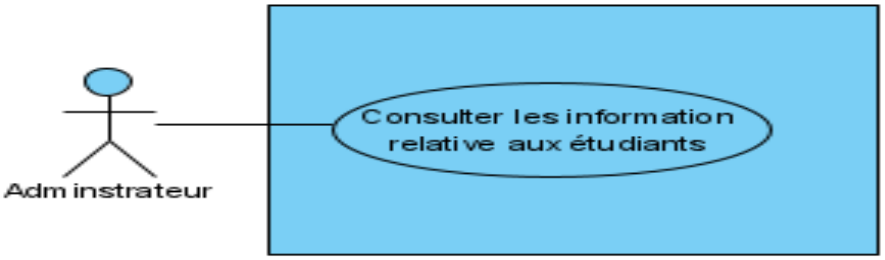

Figure 19 : Diagramme de cas d'utilisation «Consulter les informations relatives aux étudiants»

- Après l'authentification de l'administrateur avec sucée.
- Le système doit renvoyer à l'administrateur l'interface dans laquelle il peut consulter les informations relatives aux étudiants.

- **Digramme de cas d'utilisation général :**

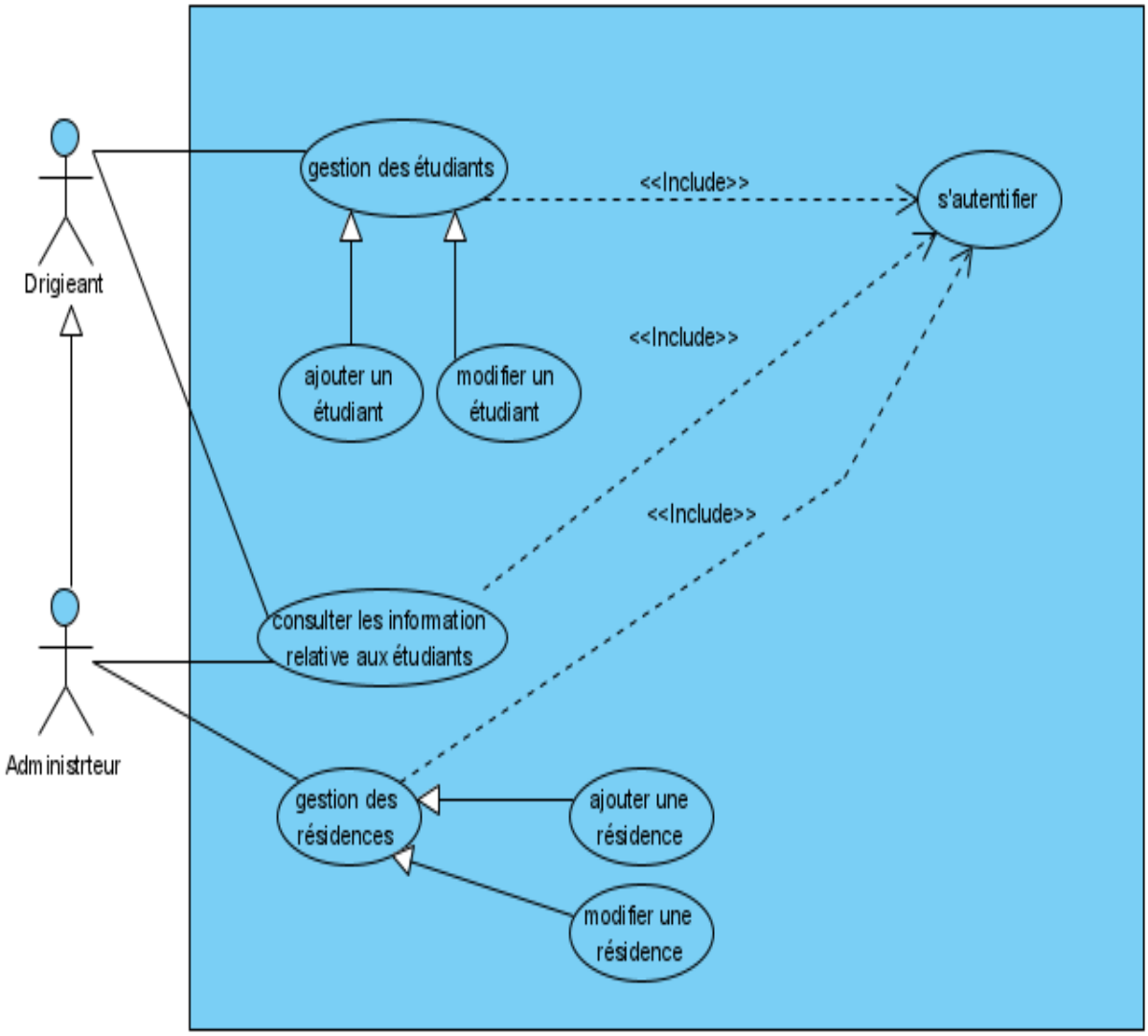

Figure 20 : Diagramme de cas d'utilisation

4.3.2.3 Digramme de séquence de cas d'utilisation :

Le diagramme de séquence est une représentation séquentielle du déroulement des traitements et des interactions entre les éléments du system et/ou de ses acteurs [17].

✓ **Diagramme de séquence de cas d'utilisation «Authentification» :**

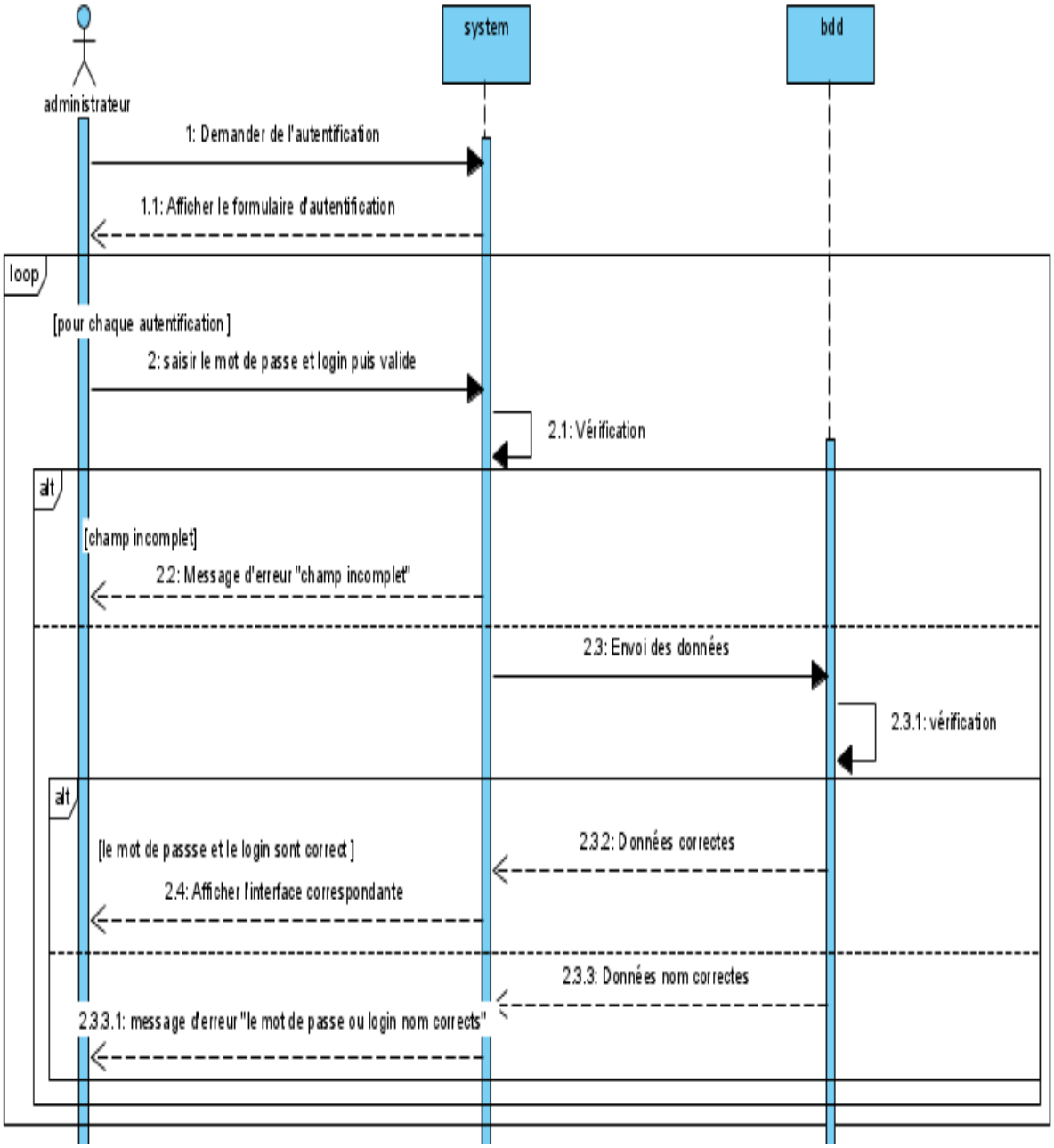

Figure 21 : Diagramme de séquence du cas d'utilisation «S'authentifier»

- ✓ **Diagramme de séquence du cas d'utilisation consulter les informations relative aux étudiants**

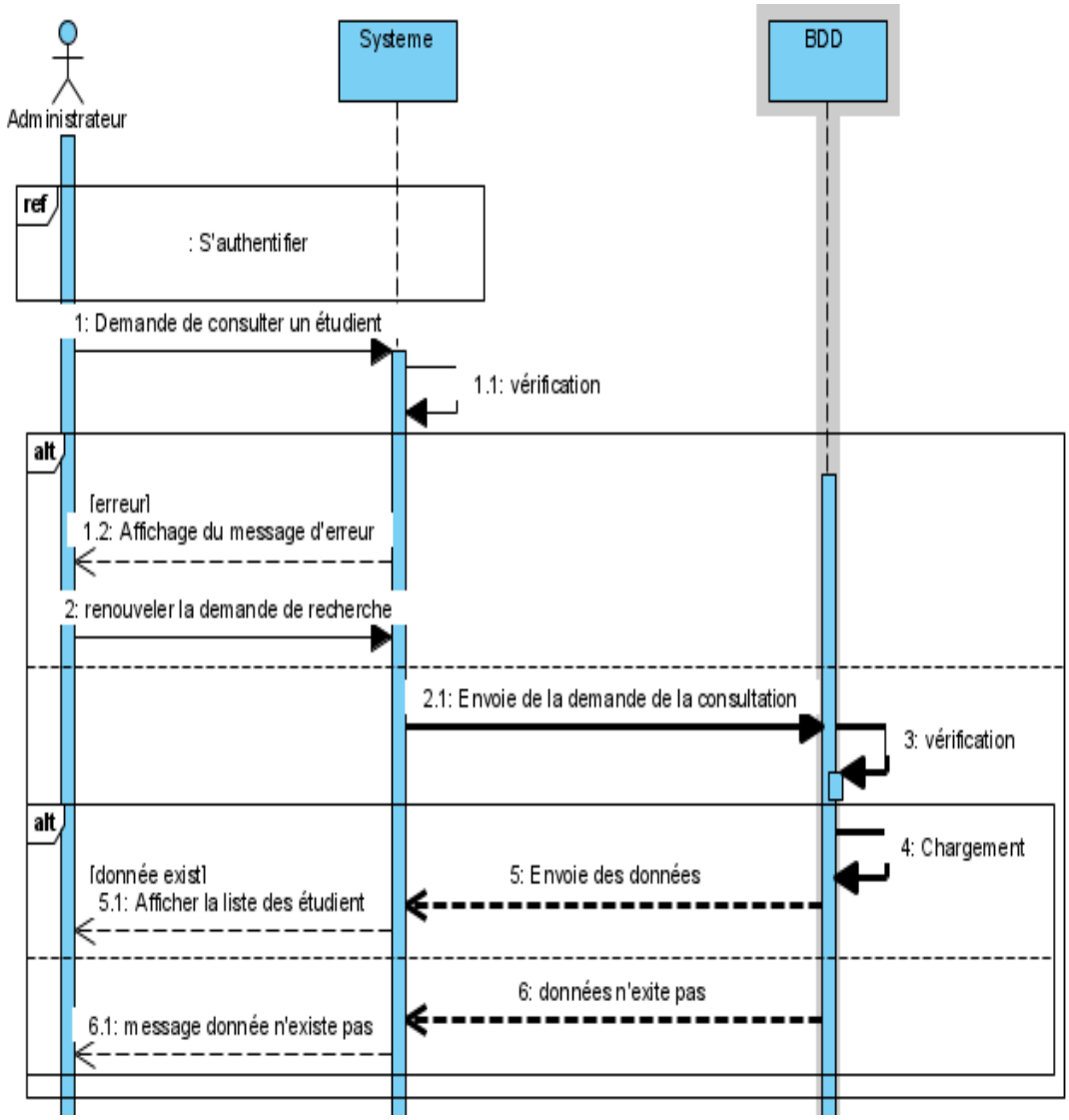

Figure 22 : Diagramme de séquence du cas d'utilisation «consulter les informations relative aux étudiants,»

✓ **Digramme de séquence de cas utilisation ajout, modification étudiant**

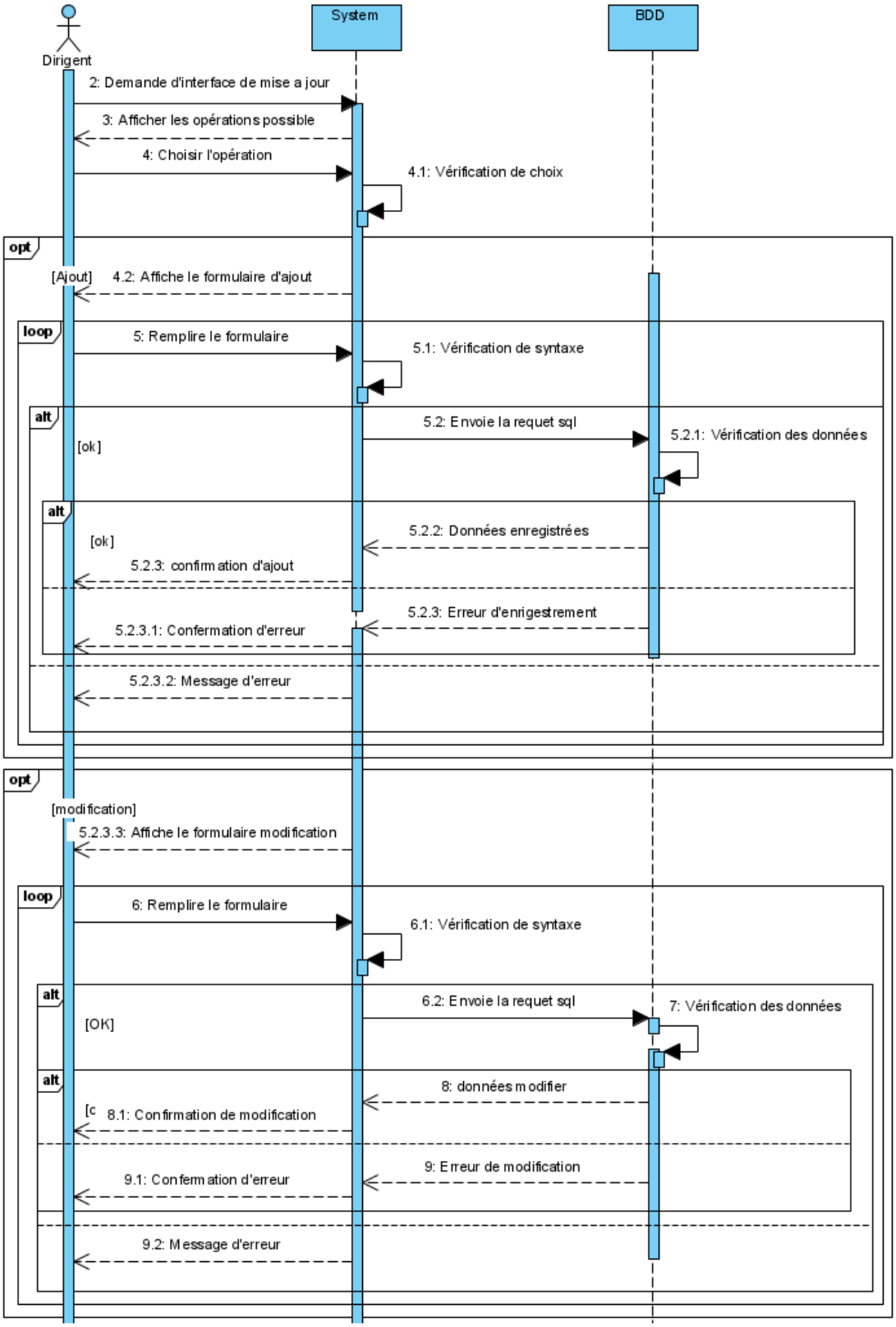

Figure 23 : Diagramme de séquence du cas d'utilisation «gestion des étudiant»

4.3 Conception

Dans la conception nous présentons les diagrammes de classe qui représentent de manière générale la structure statique d'un system, en terme de classe et de relation entre ces classe [18].

4.4.1 Digramme de classe

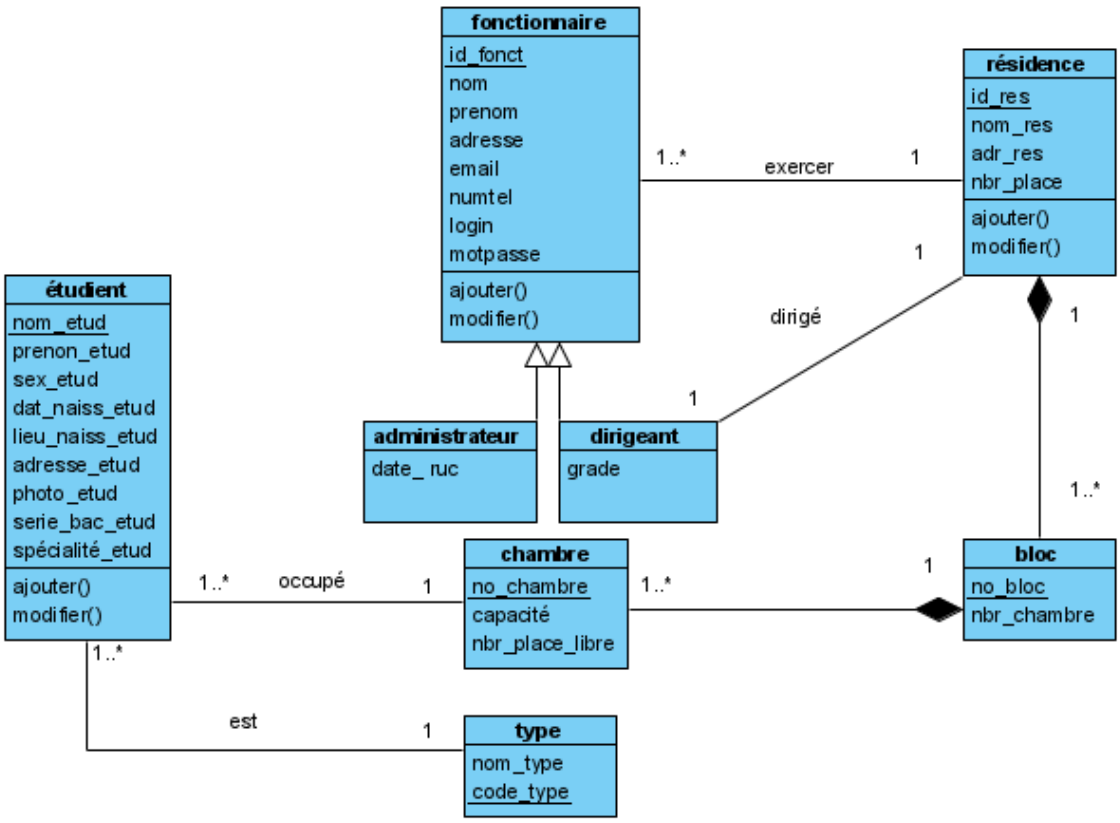

Figure 24 : Diagramme de classe

4.4.2 Passage du digramme de classe au modèle relationnel

Le model relationnel est le modèle logique des données qui correspond à l'organisation des données relationnel.il est composé de relations qui sont décrites par des attributs. Pour décrire une relation, on indique son nom, suivi du nom de, ses attributs entre parenthèses. L'identifiant d'une relation est composé d'un ou plusieurs attributs qui forment la clé primaire. Une relation peut faire une référence à une autre en utilisant une clé étrangère qui correspond à la clé primaire de la relation référenciée et caractérisé par '#'. [19]

4.4.3 Le modèle relationnelle associe à notre diagramme de classe

_ Administrateur (id_fonct, nom, prénom, adresse, email, numtel, login, motpasse, dat_ruc) ;

_ Dirigeant (id_fonct_d, nom_d, prénom_d, adresse_d, email_e_d, numtel_d, login_d, motpasse_d, grade) ;

_ Résidence (id_res, nom_res, adr_res,nbr_palace, #id_fonct_d) ;

_ Etudiant (id_etud, nom_etud, prenom_etud, sex_etud,date_naiss_etud, lieu_nais_etud, photo_etud, serie_bac_etud, adresse_etud, spécialité_etud,#no_chambre,#code_typ) ;

_ Bloc (no_bloc, nbr_chambre, #id_res);

_ Chambre (no_chambre, capacité, nbr_place_libre, #no_bloc) ;

_ Type (code_type,nom_type) ;

4.5 Conclusion

Dans ce chapitre nous avons déterminé les différents acteurs interagissant avec le système et identifié les différents cas d'utilisation associés à chacun, nous avons par la suite élaboré les diagrammes de séquence ou nous avons représenté graphiquement la relation entre les acteurs et le system de façon chronologique, et vers la fin nous avons mis en place le diagramme de classe d'où nous avons tiré le schéma de la base de donnée de notre système. Et nous avons essayé de répondre au mieux aux besoins du service d'hébergement d'une façon à éviter les anomalies et les problèmes constatées durant notre étude.

Le chapitre suivant sera consacré à la présentation de l'architecture de notre système, les outils et l'environnement de développement de notre système.

Chapitre5

Réalisation

5.1 Introduction

Apres avoir présenté l'état de l'art sur les bases de données et le cadre d'étude, nous entamons la partie pratique. Nous essayons au mieux d'utiliser les données, les informations recueillies et les connaissances acquises tout au long des chapitres précédents pour implémenter notre solution.

En première partie, nous expliquons la démarche suivie et les étapes nécessaires pour la création du réseau entre les différents ordinateurs et la configuration d'Oracle coté client et serveur.

Dans la seconde partie, nous allons présenter la solution proposée, et qui devra répondre aux exigences de la gestion de l'hébergement des résidences universitaire en détaillant les étapes qui la composent.

5.2 Structure générale de la solution proposée

La solution consiste à créer trois bases de données qui ont le même schéma, deux pour les résidences universitaires et une pour la D.O.U. Après la sauvegarde des données sur le serveur de la D.O.U les données seront fragmentées et envoyé au serveur auquel elles correspondent, ça veut dire que la base de données globale sera distribuée sur les deux autres bases selon la filière de l'étudiant et le nombre de place dans chaque résidence.

La figure ci-dessous, illustre la structure générale de la solution proposée :

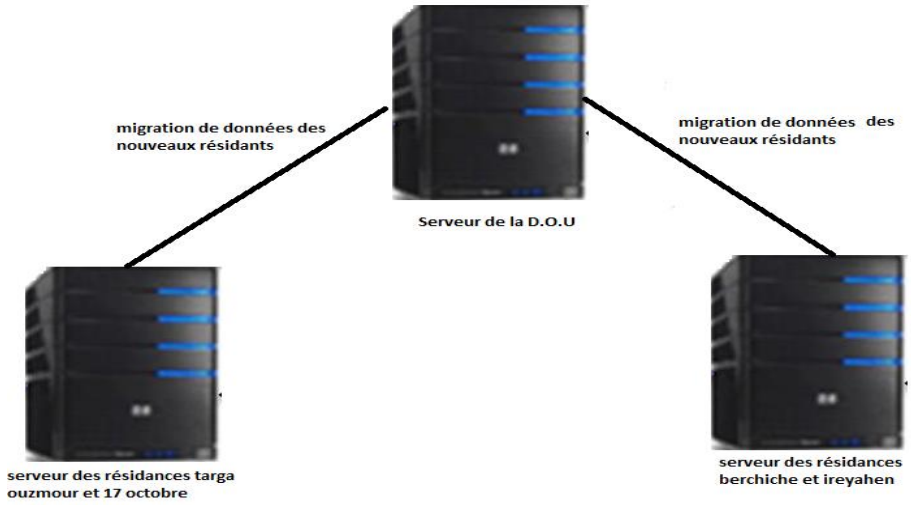

Figure 25 : Structure générale de la solution proposée

5.3 Présentation de l'environnement du travail

Pour réaliser notre système réparti, nous avons utilisé différents outils. Comme outils de développement, on a choisi NetBeans avec comme langage de programmation Java et le système de gestion de base de données Oracle 11g.

- **NetBeans :** NetBeans est un projet open source fondé par Sun Microsystems. L'IDE NetBeans est un environnement de développement permettant d'écrire, compiler, déboguer et déployer des programmes. Il est écrit en Java, mais peut supporter n'importe quel langage de programmation. Il y a également un grand nombre de modules pour étendre l'IDE NetBeans. L'IDE NetBeans est un produit gratuit, sans aucune restriction quant à son usage **[20].**
- **Java :** Java est un langage Objet permettant le développement d'applications complètes s'appuyant sur les structures de données classiques (tableaux, fichier) et utilisant abondamment l'allocation dynamique de mémoire pour créer des objets en mémoire. La notion de structure qui est un ensemble de données décrivant une entité (un objet en Java), est remplacée par la notion de classe au sens de la programmation Objet. Le langage Java permet également la définition d'interfaces graphiques (GUI : Graphical User Interface) facilitant le développement d'application interactives et permettant à l'utilisateur de piloter son programme dans un ordre non imposé par le logiciel [21].
- **Oracle** est un SGBD (système de gestion de bases de données) édité par la société du même nom (Oracle Corporation), leader mondial des bases de données. Il fonctionne de façon relativement identique sur tout type d'ordinateur. Ce qui fait que les connaissances acquises sur une plate-forme sont utilisables sur une autre et que les utilisateurs et développeurs Oracle expérimentés constituent une ressource très demandée.

Oracle est un SGBD permettant d'assurer [22] :

- La définition et la manipulation des données ;
- La cohérence des données ;
- La confidentialité des données ;
- L'intégrité des données ;
- La sauvegarde et la restauration des données ;

- La gestion des accès concurrents.

5.4 Les configurations nécessaires

5.4.1 Configuration du réseau

Les ordinateurs dont nous disposons vont être reliés grâce à un réseau sans fil '*Ad hoc*' afin qu'ils puissent se connecter à distance à la base ORACLE. Pour cela, il faut définir le même domaine de travail pour l'ensemble des ordinateurs (*WORKGROUP* ou autre), désactiver leurs Pare-feu et ajouter un nouveau réseau.

Pour ajouter un nouveau réseau il faut aller au « panneau de configuration »/ « réseau et internet » / « centre réseau et partage »/ « configurer une nouvelle connexion ou un nouveau réseau » / « Configurer un nouveau réseau sans fil ad hoc ».

On spécifie le nom de réseau ainsi que la clé de sécurité comme indique l'image suivante :

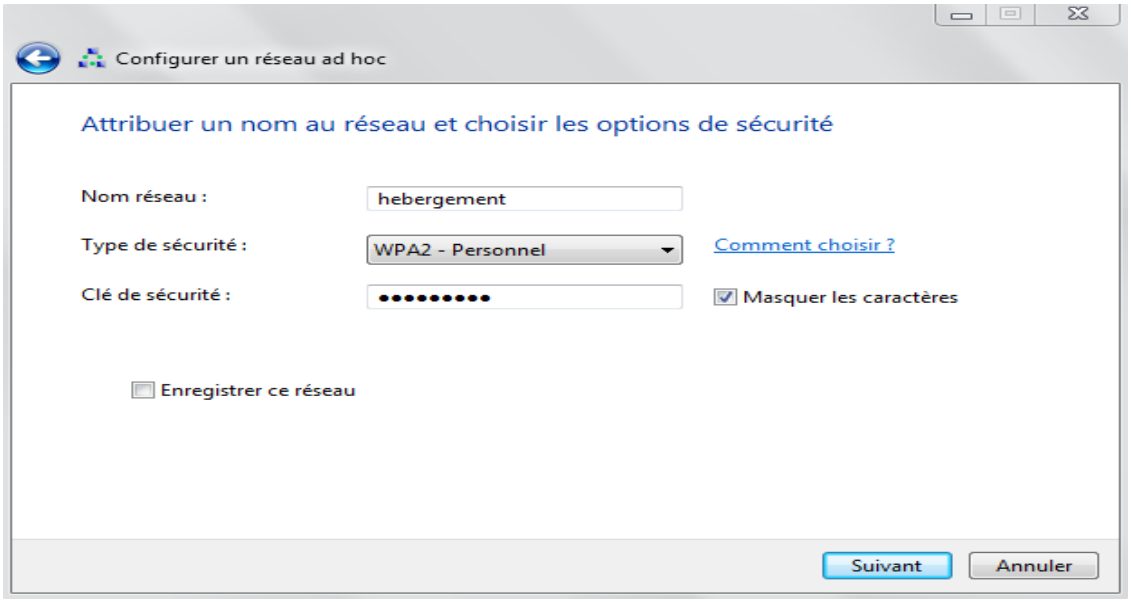

Figure 16 : Configuration d'un réseau ad hoc

5.4.2 Configuration ORACLE

Nous installons Oracle 11g Server sur le poste D.O.U et sur chaque serveur des résidences. Tandis qu'Oracle 11 g Client sera installé uniquement sur chaque poste client.

Ensuite, nous configurons la couche réseau grâce au Net Configuration Assistant pour que les serveurs et les clients puissent se communiquer.

- **Côté Serveur**

Une fois oracle installé, et les bases de données sont créés, on va configurer le « listener »pour permettre au client de se connecter à distance t à la base de données .Son fichier de paramètre se trouve dans ORACLE_HOME/Network/Admin, On va utiliser « Oracle Net Manager »pour le configurer.

La figure suivante représente le contenu de ce fichier :

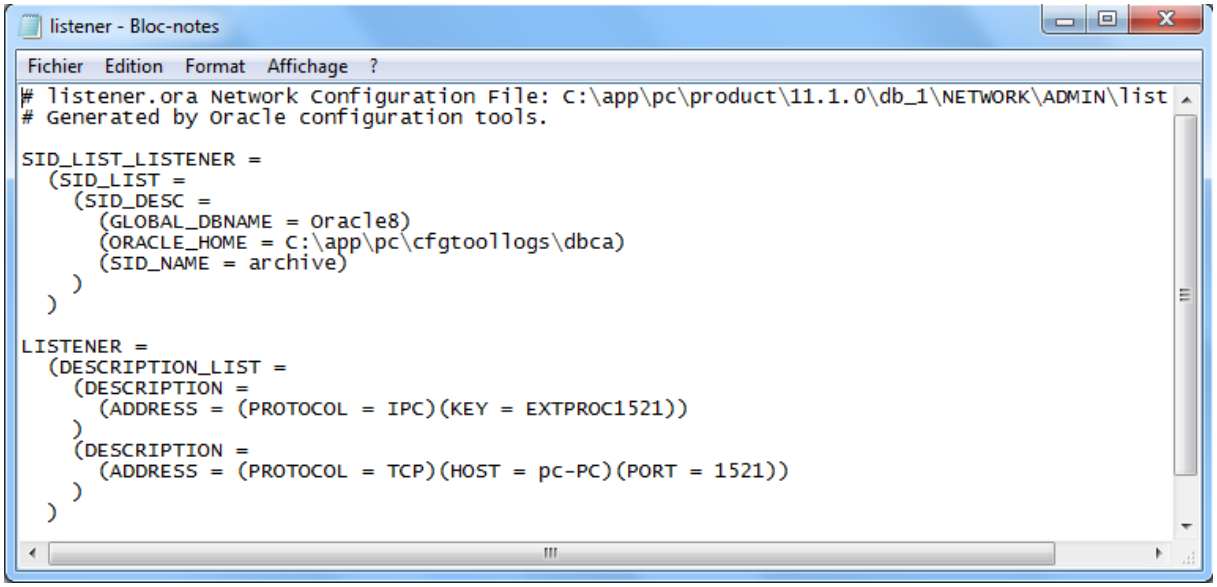

Figure 27 : Le fichier de configuration du Listener Oracle

- **Côté client :**

Afin de configurer le client il faut :

- Sélectionner les méthodes de résolution de résolution de nom utilisable par le client.
- Configurer les méthodes de résolution de nom sélectionné.

Les méthodes de résolution de noms utilisable par le client sont stockées dans le fichiersqlnet.ora. Il faut en plus définir un ou plusieurs noms de services réseau dans le fichier tnsnames.ora. On utilise aussi oracle net manager pour les configurer.la figure suivante représente la configuration de ces deux fichiers :

.

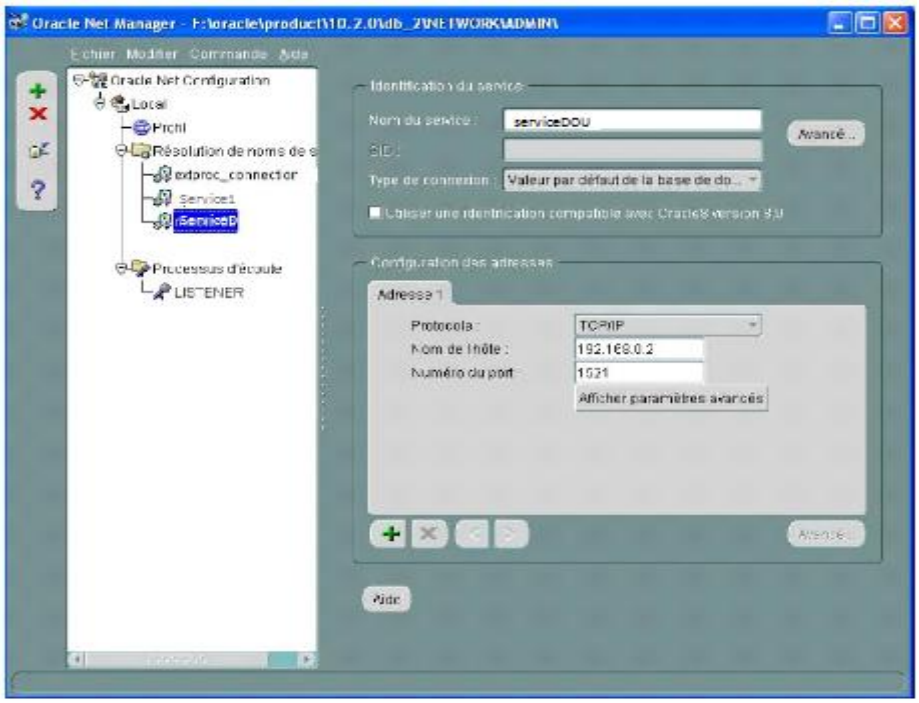

Figure 28 : Oracle Net Manager

5.5 Implémentation de la base de données

- **Création des utilisateurs et des tables**

Pour les trois bases ‘DOU’ ,’residanceA’ et ‘residanceT’ on va créer trois utilisateurs UserDOU’,’userA‘ et ‘userT’ .

Comme exemple on va créer l’utilisateur « userDOU »

```
CREATE USER  userDOU
IDENTIFIED BY user-DOU;
GRANT all privilleges TO userDOU;
```

Après on va créer les tables nécessaires (Administrateur, Dirigeant, Résidence, Etudiant, Bloc, Chambre, Type).

- **Création les liens entre les bases de données**

Pour la création d’un lien entre la 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 User IDENTIFIED BY password USING connect_string [9].

- **Entre la DOU et résidenceT**

Create database link *LienDOU_RT*

connect to *userT*

identified by *user_T*

Using *SERVICE1*

- **Entre la DOU et résidenceA**

Create database link *LienDOU_RA*

connect to *userA*

identified by *user_A*

Using *SERVICE2*

5.6 Présentation de quelques interfaces de l'application

Pour la réalisation de notre système on va utiliser oracle 11g ainsi que NetBeans avec sa version 8.0.2 la bibliothèque Swing (Java).

Pour la connexion entre le NetBeans et l'oracle 11g on utilise la bibliothèque ojdbc14.jar

5.6.1 Structure générale

Nous présentons là-dessus la structure générale du fonctionnement de notre application :

- ✓ Accueil :
  - Administration
    - Interface d'authentification administrateur
      - Gestion des résidences
        - ❖ Rechercher
        - ❖ Ajouter une résidence
        - ❖ Modifier une résidence
      - Consulter la liste des étudiants
  - Gestion d'hébergement
    - Interface d'authentification dirigeant
      - Gestion des étudiants
        - ❖ Rechercher
        - ❖ Ajouter un étudiant
        - ❖ Modifier un étudiant
      - Consulter la liste des étudiants

5.6.2 Les interfaces de l'application

Nous allons présenter dans ce qui suit les différentes interfaces de notre application.

- ✓ **La fenêtre d'accueil :** C'est la première interface de l'application apparaissant lors de l'exécution elle permet d'accéder via deux boutons, à deux interfaces d'authentification, un pour la gestion d'hébergement et un pour l'administration.

Figure 29 : Interface Accueil

- **Gestion d'hébergements**

Nous commençons par présenter les différentes interfaces du côté gestion d'hébergement. En premier lieu, nous aurons une première fenêtre Authentification gestion d'hébergements **;** après la saisie du nom d'utilisateur, du mot de passe, l'utilisateur pourra accéder à l'espace gestion d'hébergement.

Figure 30 : Interface authentification dirigeant

Lorsque le dirigeant s'est connecté, il sera dirigé vers l'interface gestion d'hébergement
S

Figure 31 : Interface gestion d'hébergement

Pour gérer les étudiants, le dirigent doit cliquer sur le bouton « gestion des étudiants» ; il sera redirigé vers la fenêtre suivante :

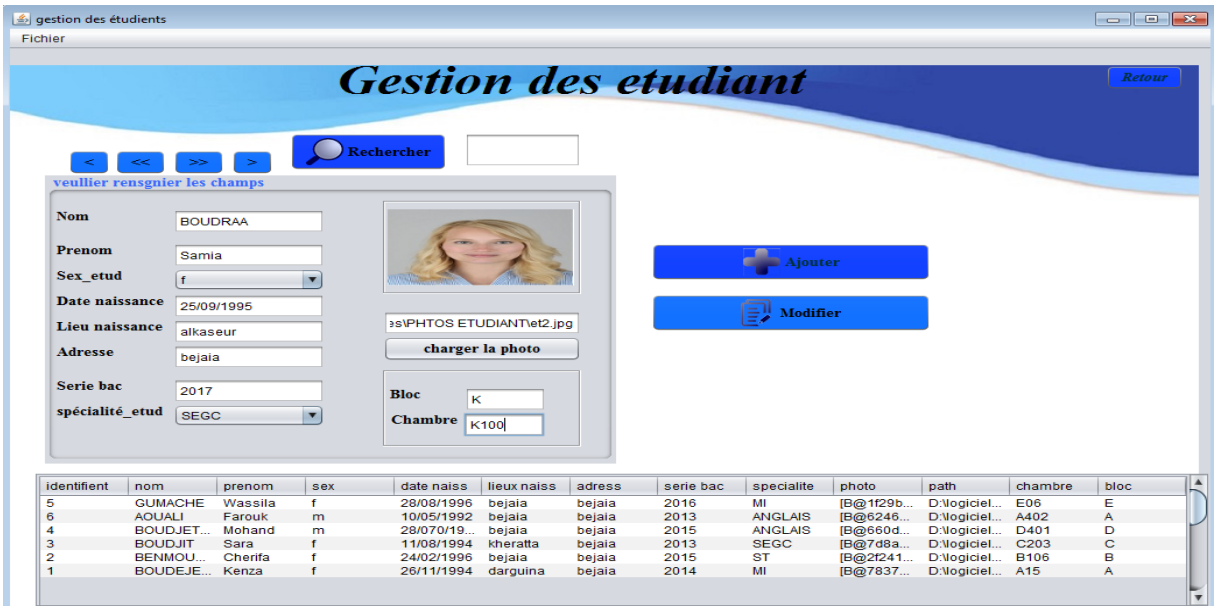

Figure 32 : Interface gestion des étudiants

Cette interface permettra au dirigeant de :

- Rechercher un étudiant : la recherche se fait en entrant le nom de l'étudient.
- Modifier un étudiant.
- Ajouter un étudiant.

Le deuxième bouton de l'interface de gestion d'hébergements permet de consulter toutes les informations relatives aux étudiants :

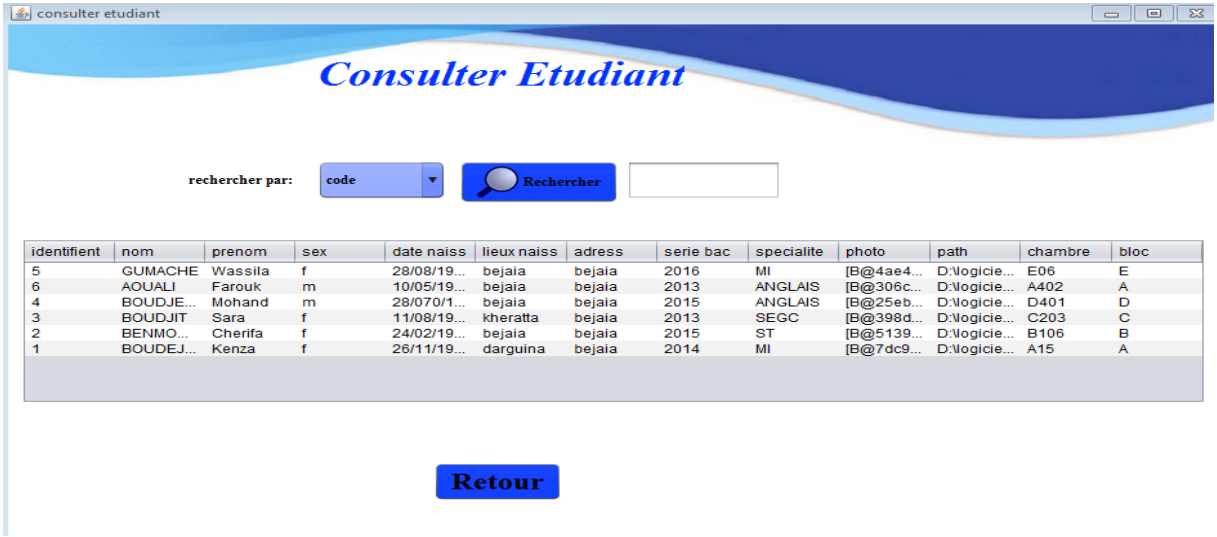

Figure 33 : Interface gestion des étudiants

La recherche d'un étudiant peut se faire par code, nom, chambre ou par bloc.

- **Administration**

La première fenêtre du côté administratif est celle de l'authentification.

Figure 34 : Interface authentification administrateur

Après que l'administrateur s'est connecté, il sera orienté vers l'espace administratif suivant :

Figure 35 : Espace administrateur

Dans cet espace l'administrateur pourra consulter la liste des étudiants et gérer les résidences. Pour gérer les résidences, l'administrateur doit cliquer sur le bouton « gestion des résidences» il sera redirigé vers la fenêtre suivante :

Figure 36 : Interface gestion résidence

✓ **Interface ajouter résidence**

Permet à l'administrateur d'ajouter de nouvelles résidences

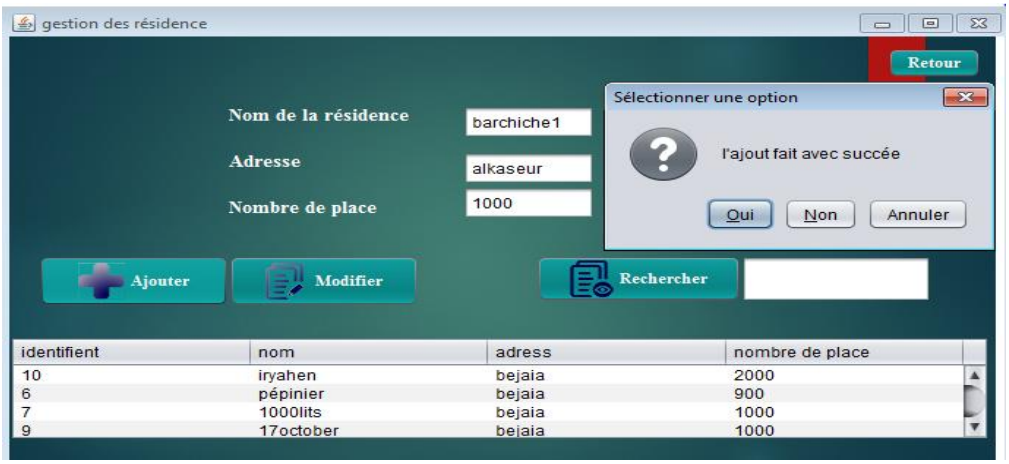

Figure 37 : Figure interface ajouter une résidence

✓ Interface modifier résidence

Permet à l'administrateur de modifier une résidence

Figure 38 : Figure interface modifié une résidence

5.7 Conclusion

Dans ce dernier chapitre, nous avons présenté les différentes parties nécessaires à la réalisation d'une base de données répartie pour la gestion de l'hébergement des résidences universitaires, à savoir la création de la base de données, la présentation des différents outils utilisés pour réaliser notre solution, ainsi que l'illustration des différentes interfaces composants notre application.

Conclusion générale et perspectives

Une base de données repartie est une collection de données logiquement unies et physiquement réparties sur plusieurs machines interconnectées par un réseau de communication. Elle permet d'intégrer et de partager des données gérées par des systèmes de gestion des bases de données réparties.

L'objectif que nous avons visé dans ce projet est la réalisation d'une base de données repartie en exploitant le réseau local existant. A l'aide d'un outil de répartition à savoir la fragmentation horizontale qui nous a permet d'échanger les informations entre la direction et les résidences, notre effort a été focalisé sur le service d'hébergement de la direction des œuvres universitaires. Pour cela, nous avons choisi ORACLE 11g comme SGBD. Ce choix est justifié par sa puissance et son efficacité, en terme de sécurité, volume de données traitées, ... etc.

Pour ce faire, nous avons commencé par la création deux bases de données pour les deux résidences, et une troisième base de données pour la D.O.U (Direction des Œuvres Universitaires), et nous avons construit les interfaces graphiques permettant de réagir avec cette base de données

Ce travail nous a permis de rentrer dans le domaine des systèmes d'information et base de données, de vivre une expérience réelle, de se rapprocher de la réalité de la vie professionnelle bien précisément la gestion documentaire et de bien maîtriser le langage de modélisation UML, et l'environnement de développement JAVA et le SGBD Oracle.

Enfin nous souhaitons que les résultats obtenus soient très encourageants et les expérimentations, réalisées sur un échantillon groupe d'étudiants, en présence des utilisateurs soient satisfaisantes. Les gains en fiabilité et en performance sont constatés très rapidement au niveau de la Direction des Œuvres Universitaires. La solution proposée peut mettre une fin définitive aux dysfonctionnements liées à la double saisie des dossiers qui se fait par la direction dans un premier temps, puis par la résidence correspondante dans un second temps, ce qui est bien évidemment très embarrassant soit pour les responsables ou pour les utilisateurs directs du système.

En perspectives nous espérons la mise en place très rapide des mécanismes de sécurité, par exemple un protocole d'authentification avancée ou un crypto-système pour les transferts de données à distance.

Bibliographie

[1] : Sacha KARKOWIAK. Introduction aux systèmes et applications réparties. Université Joseph FOURRIER de Grenoble, 2005.

[2] : Claude CHRISMENT, Genvième PUJOLLE and Gilles ZURFLUH. Base de données réparties. Techniques de l'ingénieur, Décembre 1993.

[3] : EDDY Meylan. Bases de données réparties. Haute école de gestion de Neuchâtel ,2003.

[4] : Witlod LITWIN .support de cours de base de données, intitulé : introduction et concept géneraux.2009-2010.

[5] : Vincent DESFONTAINSES. Introduction aux bases de données réparties. Université de Technologie de Compiègne, Septembre2000.

[6]: Cédric BAUDET and Eddy MEYLAN. Bases de données réparties. Bachelor of science en informatique de gestion, Haute Ecole de Gestion ARC Neuchâtel Suisse, Octobre 2007.

[7] : Jérôme GUIGNARD. Méthode de conception de BDR. 2004.

[8] : Stefano SPACCAPIETRA. Base de données réparties. Ecole Polytechnique Fédérale de Lausanne, Mars 1998.

[9] : Moussa RIM. Systèmes de Gestion de Bases de Données Réparties & Mécanismes de Répartition avec Oracle. Ecole Supérieure de Technologie et d'Informatique à Carthage, 2005.

[1]: T. Ozsu, P. Valduriez, Principles of Distributed DB Systems.

[11] : Mme S.MECHID Université M'Hamed Bougara de Boumerdes. 2011/2012 Base de Données Réparties (BDR) Master2 : Systèmes informatiques distribués.

[12]: Stephan SCHILDKNECHT and Grégoire LEJEUNE. Introduction à la réplication de bases de données, Février 2006.

[13] : Didier DELEGLISE. Manipulation des vues matérialisées, Juin 2008.

[14] : Pascal WOSCICHOWSKI. La réplication sous ORACLE. Ecole Supérieure d'Informatique, Promotion SUPINFO 2005.

[15] : Pascal ROQUES, Franck VALLÉE UML en actison Deuxième édition 2003.

[16] : Joseph Gabay, David Gabay UML 2, analyse et conception 2008.

[17] : Gilles ROY, Conception de base de données avec UML, Presses de l'université de Québec, Edition, 2009.

[18] : Annick LASSUS, méthode d'analyse orientée objet UML, cours, 2000-2001.

[19] : Pascal ROQUES et Franck VALLÉE, processus développement UML par action de l'analyse des besoins à la conception.

[20] : Hyacinthe MENIET, NetBeans. Trustonme, 2011.

[21] : Michel DIVAY, La programmation Objet en Java, 2006.

[22] : R. Bizoi, Oracle 10g Administration .

Résumé

Une base de données répartie (BDR) est une base de données dont les différentes parties sont stockées sur des sites géographiquement distants, reliés par un réseau. La réunion de ces parties forme la base de données répartie. Notre projet consiste à appliquer la technique de répartition de données entre la Direction des Œuvres Universitaires (D.O.U) et ses résidences universitaires. La première partie est purement théorique qui comprend une présentation des systèmes repartis et des bases de données réparties leurs avantages et inconvénient, en plus les différentes méthodes de conception et les technique de répartition des données dans ces systèmes. La seconde partie est consacrée pour la pratique ou nous avons détaillé la solution proposée. La solution développé a pour objectif de rendre les bases de données des résidences universitaires disponibles au niveau de la direction ainsi qu'accélérer les accès DOU-Résidences et éviter au maximum les tâches redondantes effectuées au niveau de la direction ainsi qu'au niveau des cités.

Mots clés **:** BDD, BDR, SGBD, SGBDR, Répartition de la base de données, fragmentation horizontale, oracle.

Abstract

A distributed database (BDR) is a database whose different parts are stored on geographically distant sites, linked by a network. The combination of these parts forms the distributed database. Our project consists in applying the technique of data distribution between the Department of University Works (D.O.U) and its university residences. The first part is purely theoretical which includes a presentation of distributed systems and distributed databases their advantages and disadvantages, in addition to the different design methods and techniques for distributing data in these systems. The second part is devoted to practice where we have detailed the proposed solution. The objective of the solution developed is to make the databases of university residences available at the management level as well as to accelerate the DOU-Residences access and to avoid as much as possible the redundant tasks carried out at the management level as well as at the city level.

Key words: BDD, BDR, SGBD, SGBDR, Database distribution, horizontal fragmentation, oracle.