Ministère de l'Enseignement Supérieur de la Recherche et de l'Innovation

---

Université Assane Seck de Ziguinchor

UFR des Sciences et Technologies

Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master

Mention : Informatique

Spécialité : Génie Logiciel

Sujet :

Optimisation des Performances d'un serveur de données et Conception d'une application informatique pour son automatisation

Soutenu publiquement le 14/05/2018 par

M. Mamadou Tahirou Diallo

Membres du Jury :

Pr. Alassane DIEDHIOU	Président
Dr. Serigne Diagne	Encadrant
Dr. Ibrahima DIOP	Examinateur
Dr. Khalifa GAYE	Examinateur

Sous la direction de :

Dr. Serigne DIAGNE

Sous la supervision de :

Pr. Alassane DIEDHIOU

Année académique : 2016-2017

Résumé

L'optimisation d'une BD permet aux entreprises d'obtenir un bon rendement. En effet, elle augmente le niveau de productivité des entreprises et de satisfaction de ses clients. C'est ainsi que l'optimisation des applications de BD est au cœur des problématiques soulevées par les administrateurs de bases de données, les experts et les développeurs. Elle a pour objectif de fournir, aux différents utilisateurs, un système optimal en termes de coût CPU et de mémoire, de temps de réponse, etc. L'optimisation peut être réfléchie à plusieurs niveaux : niveau applicatif (elle touche parfois la méthode de programmation), niveau base de données (on peut optimiser le modèle de données, les requêtes, l'infrastructure matériels ou logicielle) pour atteindre les meilleures performances en termes de temps de réponse et d'utilisation des ressources. Le travail effectué durant ce mémoire consiste, après avoir fait une étude sur les différentes méthodologies d'optimisation existantes, à proposer une manière d'améliorer les performances d'un serveur de données en termes de temps de réponse et de pourcentage d'utilisation du CPU et de la RAM. Pour ce faire, nous avons fait un ensemble de tests comparatifs sur deux bases de données implémentées sous le SGBDR Oracle avec différentes méthodes de gestion des mémoires, en utilisant plusieurs métriques.

Mots-clés : Base de données relationnelle, SGBDR, ORACLE, PostgreSQL, CBO, Optimisation CPU, Optimisation RAM, Temps de réponse

Abstract

Optimization enables companies to perform well. It allows companies to increase the level of productivity and satisfaction of these customers. This is how application optimization is at the heart of issues raised by database administrators, experts and developers. It aims to provide the various users with an optimal system in terms of CPU cost and memory, response time, etc. Optimization can be reflected on several levels: application level (sometimes it affects the programming method), database level (we can optimize the data model, queries, hardware or software infrastructure) to achieve the best performance in terms of response time and resource use. The work done during this dissertation consists, after doing a study on the different optimization methodologies, to propose a way to improve the performance of a relational database in terms of response time and percentage of use. CPU and RAM. To do this, we did a set of comparative tests on two databases implemented under the Oracle RDBMS with different memory management methods, using several metrics

Keywords : Relational Database, RDBMS, ORACLE, PostgreSQL, CBO, Optimization, CPU Optimization, RAM Optimization, Response Time**.**

Remerciements

Je remercie tout d'abord Notre Seigneur, Créateur (Allah), de m'avoir donné la force, le courage et la patience de bien mener ce modeste travail.

Je tiens à remercier sincèrement et chaleureusement **M. Serigne DIAGNE** qui, en tant qu'encadrant, s'est toujours montré attentif et très impliqué tout au long de la réalisation de ce mémoire. Ainsi, je lui dois beaucoup pour ses remarques pertinentes, ses suggestions, ses conseils et le temps qu'il a bien voulu me consacrer et sans qui ce mémoire n'aurait jamais vu le jour. Il représente beaucoup pour moi, un grand frère, un conseiller, une référence et surtout une idole, et le restera pour toujours.

J'adresse mes remerciements à **M. Alassane DIEDHIOU** qui, malgré son calendrier chargé a accepté de présider ce jury de soutenance. Aussi j'exprime mes sincères remerciements **M. Khalifa GAYE** et **M. Ibrahima DIOP** qui me font honneur en examinant ce travail.

Un Grand merci à **M. Youssou DIENG** Chef du département Informatique et à **M. Youssou FAYE** responsable du Master Génie Logiciel qui ont toujours œuvré pour un meilleur enseignement en Master.

Ces remerciements vont à l'endroit du corps professoral et administratif de l'UASZ surtout ceux de l'UFR Sciences et Technologies particulièrement du département Informatique, pour la richesse et la qualité de leurs enseignements et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.

Je tiens aussi à remercier **M. Alousseynou FALL.** Malgré son emploi du temps chargé, il m'a toujours ouvert sa porte pour toutes questions, ses pertinentes remarques et suggestions m'ont beaucoup aidé. Mes remerciements sont adressés à **M. Wilfried** (**DBA SQL Server**), avec qui j'ai eu à discuter sur les performances des serveurs de données.

Je n'oublierai jamais de remercier très profondément ma chère maman **Kadidiatou DIALLO** et père **Ibrahima DIALLO**, pour leur éducation, leur accompagnement, leur amour, leurs prières tout au long de mes études, sont tous pour moi**.** Je remercie également mes frères et sœurs, mes tantes, mes oncles pour leurs soutiens, leurs conseils. Je remercie mon jeune frère **Mohamadou Diang DIALLO**, pour sa compréhension, sa considération et surtout de m'avoir prêté son PC pour finaliser ce travail, sans oublier mon tuteur **Abdourahim DIALLO** et sa famille et **Ismaila TRAORE** et sa famille. Merci à vous !

A mes condisciples sans exception, à tous mes amis, et à toutes les personnes qui ont contribué à l'élaboration de ce travail.

UN IMMENSE MERCI A VOUS TOUS !!!!!!!

Dédicace

Louange à Allah le Miséricordieux

Je dédie ce modeste travail

À l'âme de mon père, que Dieu lui Garde dans son Paradis,

À ma Maman chérie adorée à qui je donnerai ma vie,

À tous mes frères, sœurs, tantes, oncles, tuteurs,

À la promo Master 2 GL 2015-2016 et amis,

À la promo L3 Info 2013-2014.

Tables des Matières

Résumé ..... **ii**
Abstract ..... **ii**
Remerciements ..... **iv**
Dédicace ..... **v**
Tables des Matières ..... **vi**
Liste des Figures ..... **x**
Liste des Tableaux ..... **xi**
Liste des Abréviations ..... **xii**
Introduction Générale **14**
1.1 - Contexte 14
1.2 - Problématique 14
1.3 - Objectifs 14
1.4 - Organisation du Rapport de mémoire 15
Première partie : Éléments fondamentaux **16**
Chapitre 1 : Les Éléments de Base **18**
Introduction 18
1.1 - Bases de données 18
1.1.1. Historique 18
1.1.2. Définition et Types de base de données 18
1.1.3. Les méthodes d'implémentation des bases de données 19
1.1.4. Les langages de requêtes 20
1.2 - Système de Gestion de Base de données (SGBD) 21
1.2.1. Historique 21
1.2.2. Rôle et types de SGBD 21
1.2.3. Exemple de SGBD propriétaire : Oracle 23
1.2.3.1. Architecture d'Oracle 12c 24
1.2.3.2. Composants Oracle 12c 24
1.2.4. Exemple de SGBD Libre : PostgreSQL 25
1.2.4.1. Architecture de PostgreSQL 25
1.2.4.2. Les composants de PostgreSQL 26
1.2.5. Synthèse 26
1.3 - Les ressources 27
Conclusion 27
Chapitre 2 : Optimisation des bases de données ..... Erreur ! Signet non défini.
Introduction 29
2.1 - Les principes d'optimisation 29
2.1.1. Optimisation d'un schéma de base de données 29
2.1.2. Optimisation des requêtes SQL 30
2.1.3. Optimisation des ressources d'un serveur de données 30

2.2 – Principes d'optimisation des SGBD étudiés 31
2.2.1. L'optimisation sous Oracle 31
2.2.1.1. L'optimisation des requêtes sous Oracle 31
2.2.1.2. L'optimisation des ressources sous Oracle 31
2.2.1.3. Les outils d'optimisation sous Oracle 32
2.2.2. L'optimisation sous PostgreSQL 32
2.2.2.1. L'optimisation des requêtes sous PostgreSQL 32
2.2.2.2. L'optimisation des ressources sous PostgreSQL 32
2.2.2.3. Les outils d'optimisation sous PostgreSQL 33
2.2.3. Synthèse 33
Conclusion 33
Deuxième partie : Application et Réalisation 34
Chapitre 3 : Dimensionnement de mémoires 36
Introduction 36
3.1 - Distribution de la mémoire Oracle 36
3.2 - Optimisation du SGA (System Global Area) 36
3.2.1. Optimisation du Shared Pool 37
3.2.1.1. Optimisation du Library cache 37
3.2.1.2. Outils de diagnostic du Library Cache 38
3.2.1.3. Optimisation du Dictionary cache 38
3.2.1.4. Outils de diagnostic du Dictionary Cache 38
3.2.2. Optimisation du Tampon de données 39
3.2.3. Outils de diagnostic du Tampon de données 39
3.2.4. Optimisation du Log Buffer 39
3.2.5. Outils de diagnostic du Redo Log Buffer 40
3.3 - Optimisation du PGA (Program Global Area) 40
3.3.1. Taille Optimale du PGA 40
3.3.2. Régler le PGA à la taille optimale. 41
3.3.3. Réduire les Passes dans le PGA 41
3.3.4. Éviter des Multi-Passes 41
3.3.5. Outils de diagnostics du PGA 41
3.4 - Recommandation de bon dimensionnement du SGA 42
3.4.1. Recommandation de bon dimensionnement du Shared Pool 42
3.4.1.1. Recommandation de bon dimensionnement du Library Cache 42
3.4.1.2. Recommandation de bon dimensionnement du Dictionary Cache 43
3.4.2. Synthèse 43
3.4.3. Recommandation de bon dimensionnement du Tampon de données 44

3.4.4. Recommandation de bon dimensionnement du Redo Log 44
3.4.5. Synthèse 44
3.5 - Recommandation de bon dimensionnement du PGA 45
Conclusion 45
Chapitre 4 : Implémentation des Recommandations 47
Introduction 47
4.1 - La Base de données Expérimentale 47
4.2 - Environnement de travail 48
4.2.1. Orcl 48
4.2.2. Orcl1 49
4.3 - Métriques 49
4.4 - Outils et Méthodes Utilisés 49
4.4.1. Plan d'Exécutions des Requêtes SQL 49
4.4.2. Consultation des vues système 50
4.4.3. Oracle Entreprise Manager Express 12c (OEM) 51
4.5 - Expérimentations et Résultats 51
4.5.1. Consultation des Plan d'exécution 51
4.5.2. Utilisation des Vues système 53
4.5.3. Comparaison des Résultats des plans d'exécutions 54
4.5.4. Résultats obtenus 55
4.5.5. Représentation graphique des Résultats obtenus 55
4.5.6. Analyse et Interprétation des Résultats obtenus 56
4.5.7. Visualisation d'OEM 56
4.6 - Étude et Comparaison de Requêtes 59
4.6.1. Exécutions des différentes Requêtes sur les BD 59
4.6.2. Résultats obtenus de l'exécution des trois Requêtes 63
4.6.3. Représentation graphique (voir Figure 21,…, Figure 23) ci-dessous 63
4.6.4. Analyse et Interprétation des résultats 63
4.6.5. Analyse et Interprétation des résultats des trois requêtes sur Orcl1 64
Conclusion 65
Troisième partie : Analyse et Conception 66
Chapitre 5 : Fonctionnalités, Spécification et Analyse, Architecture du logiciel 67
Introduction 67
5.1 - Fonctionnalités 67
5.1.1. Identification des acteurs du système 67
5.1.2. Identification des fonctionnalités 67
5.2 - Spécification et Analyse des diagrammes 68
5.2.1. Diagramme de cas d'utilisation 68
5.2.2. Analyse de l'authentification 68

5.2.3. Description de cas d'utilisation « s'authentifier » 68
5.2.4. Les activités de l'authentification 70
5.2.5. Diagramme de séquence de l'authentification 70
5.2.6. Analyse de la gestion des ressources 71
5.2.7. Description de cas d'utilisation « allouer ressources » 71
5.2.8. Les activités de la gestion des ressources 72
5.2.9. Diagramme de séquence du cas d'utilisation « allouer ressources» 72
5.2.10. Analyse du cas d'utilisation «ModifierConfig» 73
5.2.11. Description du cas d'utilisation «ModifierConfig» 73
5.2.12. Les activités de la modification d'une configuration 73
5.2.13. Le diagramme de séquence du cas « modifierconfig » 74
5.2.14. Diagramme d'activité de l'application 75
5.2.15. Diagramme de composant de l'application 75
5.3 - Architecture 76
5.3.1. Succession d'exécution des fonctionnalités 76
5.3.2. Les modules du logiciel 76
5.3.2.1. Le module gestion des ressources 76
5.3.2.2. Le module gestion des performances 77
Conclusion 77
Conclusion Générale et Perspectives 78
Bibliographie 80
Webographie 82
Annexes 83

Liste des Figures

Figure 1 : Architecture d'Oracle 12c [W4] 24
Figure 2 : structure de PostgreSQL [9] 26
Figure 3 : Vérification de la gestion de mémoires utilisées par Orcl 51
Figure 4 : Plan d'exécution de R sur Orcl 52
Figure 5 : Plan d'exécution de R sur Orcl1 52
Figure 6 : Statistique vues système 53
Figure 7 : Coût (CPU+I/O) donné par les vues d'Orcl 54
Figure 8 : Coût (CPU+I/O) donné par les vues d'Orcl1 54
Figure 9 : histogramme du temps mis par la requête sur les bd 55
Figure 10 : histogramme coût (cpu+io) des deux bases de données 55
Figure 11 : histogramme de la consommation en ram des bd 55
Figure 12 : charge d'uc sur orl et orcl1 57
Figure 13 : gestion ressources des bd Orcl et orcl1 58
Figure 14 : e/s disque sur Orcl et orcl1 58
Figure 15 : Plan d'Exécution de R1 sur Orcl 60
Figure 16 : Plan d'Exécution de R1' sur Orcl 60
Figure 17 : Plan d'Exécution de R1'' sur Orcl 61
Figure 18 : Plan d'Exécution de la requête R1 sur Orcl1 61
Figure 19 : Plan d'Exécution de la requête R1' sur Orcl1 62
Figure 20 : Plan d'Exécution de la requête R1'' sur Orcl1 62
Figure 21 : histogramme temps mis par les requêtes sur les bd 64
Figure 22 : histogramme ram utilisé par les requêtes sur les bd 64
Figure 23 : histogramme coût (cpu+io) des requêtes sur les bd 64
Figure 24 : histogramme du temps d'exécutions des requêtes sur orcl1 65
Figure 25 : histogramme ram consommée par les requête sur orcl1 65
Figure 26: DIAGRAMME DE CAS D'UTILISATION DE L'ADMINISTRATEUR SYSTEME 69
Figure 27 : DIAGRAMME DE CAS D'UTILISATION DU DBA 69
Figure 28 : DIAGRAMME D'ACTIVITE DU CAS D'UTILISATION «S'AUTHENTIFIER» 70
Figure 29 : DIAGRAMME DE SEQUENCE DU CAS D'UTILISATION «S'AUTHENTIFIER» 71
Figure 30 : DIAGRAMME D'ACTIVITE DU CAS «ALLOUER RESSOURCES» 72
Figure 31 : DIAGRAMME DE SEQUENCE DU CAS D'UTILISATION «ALLOUER RESSOURCES» 73
Figure 32 : DIAGRAMME D'ACTIVITE DU CAS « MODIFIERCONFIG » 74
Figure 33 : DIAGRAMME SEQUENCE DU CAS «MODIFIERCONFIG» 74
Figure 34 : DIAGRAMME D'ACTIVITE DU SYSTEME 75
Figure 35 : DIAGRAMME DE COMPOSANT 75
Figure 36 : Architecture du logiciel AGPSD 76
Figure 37 : script de création de la base de données drh 83

Liste des Tableaux

Tableau 1 : avantages et inconvénients des types de BD [2][3]. 19
Tableau 2 : avantages et inconvénients des types d'index 20
Tableau 3 : avantages et inconvénients des Types de SGBD 22
Tableau 4 : avantages et inconvénients du SGBDR Oracle [15] 23
Tableau 5 : avantages et inconvénients du SGBDR PostgreSQL [15] 25
Tableau 6 : Dictionnaire de données de notre base de données 47
Tableau 7 : Volumétrie de la base de données DRH 48
Tableau 8 : Dimensionnement des mémoires d'Orcl sous Oracle 48
Tableau 9 : Dimensionnement de mémoires Orcl1 sous Oracle 49
Tableau 10 : Indicateurs mesurés sur les deux bases de données 55
Tableau 11 : Résultats des trois requêtes effectuées sur Orcl et Orcl1 63
Tableau 12 : Identification des acteurs 67
Tableau 13 : Identification des fonctionnalités du système 67
Tableau 14 : Description de cas d'utilisation « s'authentifier » 69
Tableau 15 : Description de cas d'utilisation «allouer ressources » 71
Tableau 16 : Description de cas d'utilisation « modifierconfig» 73

Liste des Abréviations

AGPSD : Aide à la Gestion des Performances d'un Serveur de Données

BD : Base de Données

BDD : Base de Données Déductive

BDE : Base de Données Extensionnelle

BDO : Base de Données Objet

BDOR : Base de Données Objet Relationnelle

BDI : Base de Données Intensionnelle

BDR : Bases de Données Relationnelle

CBO: Cost Based Optimizer

CDB **:** Multitenant Container Database

CPU : Central Processing Unit

DBA : Database Administrator

DBMS : Database Management System

DWH : Datawarehouse

DSS : Decison Système Support

E/S : Entrées/Sorties

I/O : Input/Output

LRU : Least Recently Used.

OEM : Oracle Entreprise Manager 12c

OLAP : OLine Analytic Processing

OLTP : OnLine Transaction Processing

OQL : Object Query Language

ODMG : Objet Data Management Group

PDB : Pluggeable Database

PGA : Program Global Area

RAM : Random Access Memory

RBO : Rule Based Optimizer

RM : Oracle Resource Manager Express 12c

SGA : System Global Area

SGBD : Système de Gestion de Bases de Données

SGBDOO : Système de Gestion de base de Données Orienté Objet

SGBDR : Système de Gestion de Bases de Données Relationnelles

SQL : Structured Query Language

UML : Unified Modeling Langage

Introduction Générale

1.1 - Contexte

Aujourd'hui l'optimisation des applications est un vaste problème et fait souvent l'objet de débats majeurs entre experts, administrateurs de base de données (DBA) et programmeurs sur les causes générant les effets constatés (lenteur d'affichage, fort trafic réseau, saturation du serveur de données). L'amélioration des performances en termes de diminution des temps de réponse est un domaine en perpétuel croissance. Diminuer les temps de réponse des serveurs de données est une tâche quotidienne qui s'étend de la conception du modèle de données à l'exploitation du serveur.

1.2 - Problématique

La gestion d'un serveur de données est une tâche très importante dans le bon fonctionnement et la réduction des temps d'attente d'un système d'information. L'optimisation de sa gestion permettant d'allouer plus de ressources aux utilisateurs qui en ont le plus besoin est une nécessité dans le traitement des demandes des utilisateurs. Pour mieux comprendre l'approche d'optimisation, il est important de préciser la situation dans laquelle la réflexion a été élaborée. Il s'agit du cas où la durée de traitements des requêtes dépasse les délais d'attente. Aussi une consommation excessive de ressources est constatée. Ainsi, pour satisfaire les besoins en performances, les DBA ont tendance à augmenter d'une manière excessive la taille mémoire allouée. Cependant, la réduction du temps de réponse, la gestion des ressources, l'augmentation de la productivité des utilisateurs sont primordiales pour assurer le succès du fonctionnement d'un système d'information.

Ces constats nous amènent à nous poser la question suivante :

- Comment optimiser les ressources d'un serveur de données ?

1.3 - Objectifs

L'objectif visé dans ce mémoire est de proposer une méthodologie d'allocation de la mémoire pour de meilleures performances d'un serveur de données. Pour atteindre cet objectif, nous proposons des recommandations de dimensionnement des mémoires. Ces recommandations sont basées sur le temps de réponse et la consommation des ressources d'un serveur de données. Ainsi nous avons deux objectifs spécifiques :

- Optimisation des requêtes SQL sur un serveur de données,
- Optimisation des ressources (CPU, RAM) d'un serveur de données.

Dans chacun de ces cas majeurs, il est nécessaire de définir un ensemble de métriques pour l'optimisation, afin d'obtenir de meilleures performances. Dans ce mémoire nous allons explorer les principales techniques d'optimisation disponibles pour pouvoir estimer les améliorations à apporter ou proposer une méthodologie d'optimisation plus performante.

1.4 - Organisation du Rapport de mémoire

Notre mémoire est subdivisée en trois parties : dans la première partie comportant deux chapitres, nous parlons des éléments fondamentaux. Ces éléments sont d'une importance capitale par rapport au sujet abordé.

- Dans le chapitre 1, nous présentons les éléments de bases. Ce chapitre fera une étude généralisée sur les bases de données, les SGBD, les ressources d'un serveur de données. Il permettra de définir chacune de ces parties, de donner leur importance dans le fonctionnement d'un serveur de données.
- Le chapitre 2 sera consacré à l'optimisation des bases de données. Ce chapitre montre les principes fondamentaux d'optimisation d'une base de données, l'importance de l'optimisation des bases de données dans les performances d'un serveur de données, etc.

La deuxième partie comptant deux chapitres, parlent des recommandations de dimensionnement des mémoires et la pratique. Cette partie montre l'impact du dimensionnement des mémoires dans les performances d'un serveur de données. Elle présente l'implémentation d'une même base de données sous Oracle et les métriques utilisées dans la pratique pour mesurer les performances.

- Dans le chapitre 3, nous évoquons le dimensionnement des mémoires sous oracle. Il permet de connaitre la taille allouée aux mémoires sous Oracle. Ainsi, il donne des recommandations de dimensionnement des mémoires pour de meilleures performances d'un serveur de données.
- Le chapitre 4 sera consacré à la pratique. Dans ce chapitre, nous implémentons une même base de données avec les méthodes de gestion de mémoires existantes. Différents indicateurs sont étudiés sur cette base de données afin de mesurer les performances.

La troisième et dernière partie concerne l'analyse et la conception de l'application pour l'automatisation de l'optimisation d'un serveur de données et comporte 1 chapitre.

- Le chapitre 5 sera consacré à la conception de l'application. Dans ce chapitre nous donnons l'architecture, les fonctionnalités et les diagrammes de l'application.

Première partie : Éléments fondamentaux

Chapitre 1 : Les éléments de base

Chapitre 2 : Optimisation des bases de données

Chapitre 1 : Les Éléments de Base

Introduction

Dans ce chapitre introductif, nous allons faire d'abord une brève présentation des bases de données dans la Section 1.1. Il s'agira de définir les bases de données, leur historique, les types de BD, leurs méthodes d'implémentation, les langages de manipulation de ces bases de données et les index. Ensuite nous parlons dans la Section 1.2 des systèmes de gestion de bases de données tout en donnant les types de SGBD et nous étudions deux exemples de SGBD : Oracle (propriétaire), PostgreSQL (Open Source et gratuit). Enfin dans la Section 1.3, nous parlons des ressources utilisées par un serveur de données pour satisfaire les utilisateurs.

1.1 - Bases de données

1.1.1. Historique

Au début, la gestion et le traitement des données se faisaient par la méthode classique, basée sur les fichiers, à laquelle nous avons pu dégager les défauts suivants [1] :

- La redondance de données ;
- La dépendance pleine entre données et traitements ;
- Le manque de normalisation au niveau de stockage de données.

Pour remédier à ces problèmes, les bases de données sont créées en 1960 répondant aux questions suivantes : la structuration, l'accès aux données selon de multiples critères, l'intégrité des données, la relation entre les données.

1.1.2. Définition et Types de base de données

Une base de données est un ensemble d'informations, connexes de manière directe ou indirecte, enregistrée dans un dispositif informatique. Les bases de données ont connu des évolutions majeures. Chaque évolution venait répondre à un besoin spécifique sur la structuration et le stockage des données.

- **Les bases de données relationnelles** : ce modèle a été prononcé pour la première fois en 1970 par Edgar F.Codd dans sa thèse de mathématique sur l'algèbre relationnelle. Ce modèle est caractérisé par des relations ou tables liées entre elles grâce à des clés étrangères [2].
- **Les bases de données objets** : apparues en 1990, les BDO permettent de gérer les structures de données complexes, en profitant de la puissance de modélisation des modèles objets et de la puissance de stockage des BD classiques [2].

- **Les bases de données relationnelles objet** : elles sont nées du double constat de la puissance nouvelle promise par les BDO et de leur insuffisance pour répondre aux exigences de l'industrie des BD classiques [2].

TABLEAU 1 : AVANTAGES ET INCONVENIENTS DES TYPES DE BD [2][3].

Types de BD	Avantages	Inconvénients
BDR	- Approche formellement définie ; - Fiables et performantes ; - BDR sont plus répandues ; - Modèle simple ; - SGBD inclut des outils de performances, de gestion de requêtes, d'optimisation, etc.	- Structure très limité pour représenter les types complexes ; - Types de données disponibles sont limités et non extensibles ; - Sémantique insuffisante ;
BDO	- Les principes d'encapsulation et d'abstraction du modèle objet permettent de mieux séparer les BD de leurs applications ; - Héritage d'opération et de structure facilite la réutilisation de type de données ; - Possibilités de définir des données abstraites ;	- Gestion des pannes peu prise en compte ; - Gestion de la persistance et de la coexistence des objets en mémoire et sur disque complexe ; - Complexité de gérer les transactions concurrentes; - Système peu fiable
BDOR	- Partir des BDR et rajout du modèle objet ; - Extension du langage SQL (SQL 3); - Encapsulation des données des tables ;	- Pas de syntaxe commune de la part des éditeurs de SGBD (IBM, Oracle); - Complexité de faire un retour du modèle objet vers le modèle relationnel

1.1.3. Les méthodes d'implémentation des bases de données

Les bases de données peuvent être implémentées différemment selon les objectifs visés, les volumes de données, la taille de l'entreprise, etc.

- Une BD centralisée : **c**'est-à-dire que les informations sont stockées dans une seule machine manipulée par un ou plusieurs utilisateurs [4].

- Une BD répartie : les différentes parties sont stockées sur des sites généralement géographiquement distants, reliés par un réseau [4][5].

1.1.4. Les langages de requêtes

Un langage de requête est un langage informatique utilisé pour accéder aux données d'une BD. Parmi ces langages, nous pouvons citer :

- Le Langage SQL est un langage utilisé pour manipuler les bases de données relationnelles. Il a été développé par IBM dans les années 70. SQL3 est l'une des évolutions de SQL qui tire le bénéfice des SGBDR et des SGBDOO. Il permet de gérer des données complexes dans le cadre de SGBD objet relationnel [w1].
- Le Langage OQL est un langage de requêtes standard pour manipuler les BDO définis par ODMG. À la différence de SQL dans les BDR, le langage OQL utilise des variables pour effectuer ses requêtes.

1.1.5. Les Index

Un index est une structure de données stockée avec la base de données. Il permet d'extraire rapidement les données et améliorer les performances des requêtes SQL dans une base de données. Différents types d'index existe.

- **Les B-arbre ou B-Tree :** Ils sont présents dans de nombreux SGBDR. La clé d'un index B-Tree subit une réorganisation dynamique à la suite d'une insertion ou mise à jour.
- **Les index Bitmap** : Un index bitmap est une structure de données définie dans un SGBD et très utilisée pour les opérations de sélection de gros volumes de données. Ces index sont utilisés pour les Applications de type DWH.
- **Table de Hachage** : C'est une méthode de transformation simple consistant à attribuer une adresse (représentée par un nombre naturel de 1 à n) à chaque valeur de clé choisie dans une table. L'accès à un élément se fait par l'intermédiaire d'une fonction de hachage.

TABLEAU 2 : AVANTAGES ET INCONVENIENTS DES TYPES D'INDEX

Types d'Index	Avantages	Inconvénients
B-Tree	- Arbre équilibré : toutes les feuilles de l'arbre ont la même profondeur ; - Balancement automatique; - Lecture séquentielle rapide et accès rapide ;	- Consomme une place importante dans la BD; - Taille de l'index pouvant être important ; - Ralentis les opérations d'insertion, de modification

	- Facilité d'adaptation à une variété de requêtes ; - Utilisé par la majorité des SGBD du marché ;	- Les valeurs nulles ne sont pas indexées
Bitmap	- Très pratique pour les applications DSS; - Accélère les opérations de sélections ; - Utilise peu d'espace disque; - Possibilité d'indexer les valeurs nulles; - Supporte les requêtes ad hoc en DSS;	- Pas adapté aux applications OLTP; - Ralentis les opérations d'insertion, de modification et de suppressions ; - Utilisé par peu de SGBD ;
Table hachage	- Accélère les opérations d'insertion; - Accélère les opérations de suppression; - Rapide dans une sélection avec une inégalité;	- Sélection ordonnée; - Ne fonctionne pas avec une inégalité; - Perte de performance en cas de multiples duplicatas;

1.2 - Système de Gestion de Base de données (SGBD)

1.2.1. Historique

Les volumes des données des bases de données évoluant très rapidement dans le temps. Nous assistons à une problématique posée autour de la mise en place d'un outil permettant de :

- Créer et manipuler des bases de données efficaces;
- Rendre robustes les BD;
- Gérer l'accès à une BD;
- Sécuriser les BD.

C'est en ce sens qu'en 1961, les SGBD ont vu le jour pour résoudre à ces problèmes. Depuis cette année à nos jours, différents modèles de SGBD existaient (voir section 1.2.2).

1.2.2. Rôle et types de SGBD

Un SGBD (Système de Gestion de Base de données) est un système permettant de gérer une base de données. Les fonctionnalités essentielles d'un SGBD sont :

- Stockage d'un grand volume de données avec durabilité;

- Accès efficace;
- Support d'un modèle de données;
- Sécurité de la base de données;
- Contrôler les accès concurrents;
- Le SGBD doit être capable de faire des transformations entre chaque niveau, de manière à transformer une requête exprimée au niveau externe en requête du niveau conceptuel puis du niveau physique [6].

Les SGDB ont connu des succès très importants dans le besoin de support de certaines exigences des utilisateurs, parmi ces évolutions nous avons : **Modèle hiérarchique** (créé dans les années 1960, c'est le plus ancien modèle de données), **Modèle Réseau, Modèle relationnel, Modèle déductif (**Une base de données déductive (**BDD**) est constituée de : **BDE (**Ensemble des faits connus dans la BD relationnelle**)** et **BDI** (Ensemble des faits ou règles déduits) [7]), **Modèle Objet** (fondé sur la notion d'objet et de la programmation orientée objet [w2]).

TABLEAU 3 : AVANTAGES ET INCONVENIENTS DES TYPES DE SGBD

Types SGBD	**Avantages**	**Inconvénients**
Modèle hiérarchique	- Compréhensible pour les développeurs COBOL avec les occurrences; - Rapide dans une sélection avec une inégalité; - Simplicité relative à l'implémentation ; - Adéquation parfaite à une entreprise à structure arborescente ;	- Accès lent pour chercher le père d'un fils nommé; - Pas de sécurité : la suppression d'un père entraine celle de ses fils; - Les accès se font uniquement depuis la racine; - Pas d'interface utilisateur simple ;
Modèle Réseau	- Résolution des problèmes d'ajout, de modification et de suppression de fils ; - Sécurité des données ; - Accès rapide à l'information	- Langage navigationnel : parcours des SET ; - Gestion des pointeurs physique à la charge du SGBD
Modèle relationnel	- Modèle simple : BD=ensembles de tables ; - Un langage commun et universel (SQL) ; - Technologie plus répandue ;	- Pas adapté aux environnements distribués requis par de volumes gigantesques de données; - Très limité pour le big-data ;

	- Efficace pour les applications de gestion classique ; - Une méthodologie de conception de schéma : normalisation ;	- Difficile pour modéliser les systèmes complexes ;
Modèle Objet	- Identité d'objet : favorise le partage de données et supporte les collections ; - Facilite l'encapsulation des données ; - Possibilité de définir les opérations abstraites (polymorphisme : augmentant la productivité des développeurs d'applications);	- Complexe à implémenter; - Peu compatible avec les SGBDR;

1.2.3. Exemple de SGBD propriétaire : Oracle

Oracle a été développé par Larry Ellison, Bob Miner et Ed Oates. La version 12cR2 d'Oracle est sortie le 19 septembre 2016, développée en C, Java, C++, [w3]. Oracle 12cR2 permet aux clients d'améliorer la qualité et les performances de leurs applications, gagner du temps avec une disponibilité maximale. Il simplifie la consolidation de la base en gérant des centaines de milliers de bases de données comme s'il n'y en avait qu'une.

TABLEAU 4 : AVANTAGES ET INCONVENIENTS DU SGBDR ORACLE [8]

Avantages	**Inconvénients**
- Richesse fonctionnelle ; - Fonction d'audit très évolué ; - Assistants performants via Oracle Entreprise Manager, possibilité de gérer en interne des tâches et des alertes ; - Gestion centralisée de plusieurs instances; - Gestion d'un millier de PDB avec une seule instance; - Très avancé en OLTP et en DSS ;	- Prix élevé, tant au point de vue des licences que des composants matériels (RAM, CPU) à fournir pour de bonne performance ; - Administration complexe, liée à la richesse fonctionnelle; - Gourmand en ressources et en espace disque par exemple près de 700 Ko/utilisateur; - Pas de type auto-incrément ;

1.2.3.1. Architecture d'Oracle 12c

Pour faire face à l'avènement du **Cloud** et satisfaire ses clients, Oracle a mis en œuvre une nouvelle architecture «**multitenant»**, qui a vu le jour en 2013. Oracle 12c a valu 2500 années hommes de développement selon Tom Kytes Architecte Sénior chez Oracle.

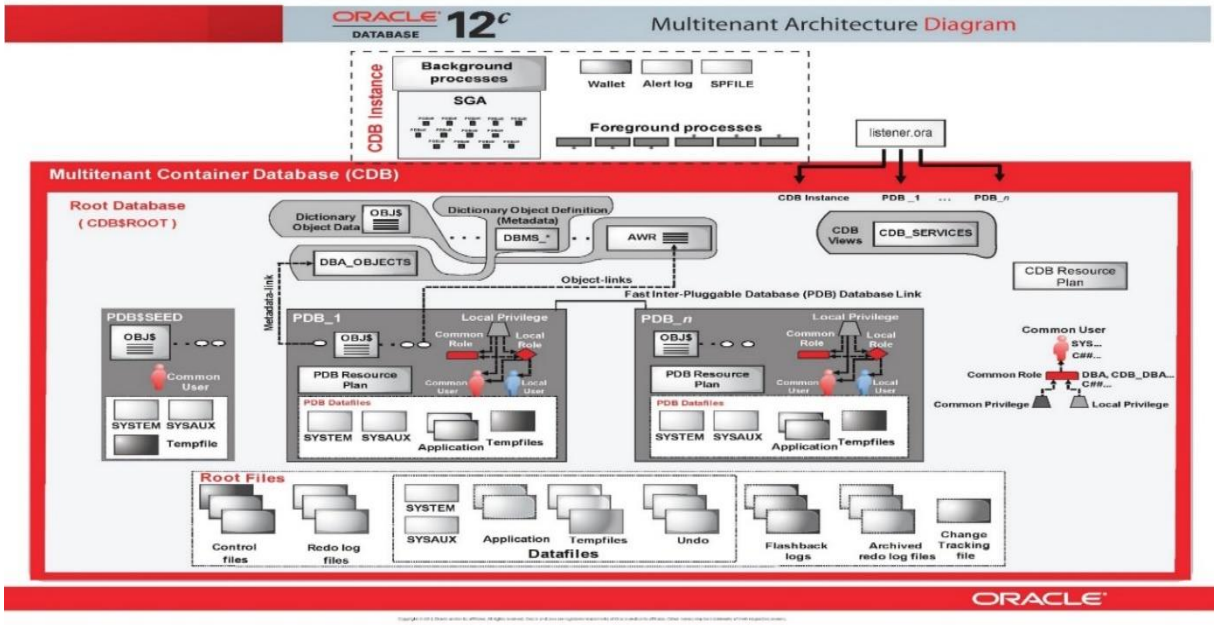


FIGURE 1 : ARCHITECTURE D'ORACLE 12C [W4]

Cette architecture nous fait voir les éléments suivants :

- **CDB :** c'est la base de données conteneur mutualisée, pouvant gérer jusqu'à 4056 bases de données insérées (PDB). Un CDB contient un ensemble de schémas, d'objets et de structures connexes. Chaque conteneur possède un identifiant unique et un nom;
- **Root :** Communément appelé **Conteneur Racine** «CDB$ROOT». Elle contient le dictionnaire de données qui englobe certaines informations sur les bases de données insérées PDB;
- **Le modèle de type base de données** insérée PDB, appelé «PDB$SEED», est utilisé pour la création d'autres bases de données insérées PDB;
- **PDB :** elle fonctionne de manière autonome et est vue comme une BD normale aux yeux des utilisateurs. La PDB contient ses propres tablespaces «SYSTEM», «SYSAUX» et «TEMP» et son propre dictionnaire de données. Une PDB est utilisé pour stocker des données spécifiques à une application. Son arrêt n'a aucun effet sur les autres PDB.

1.2.3.2. Composants Oracle 12c

Un serveur de BD Oracle est composé de fichiers, de processus et de mémoires.

- **Les Fichiers** : ce sont les fichiers, de contrôles, de journalisations, de données.

- **Les processus** : **les processus serveur** (ils gèrent les requêtes des utilisateurs lors des connexions à la BD), **les processus d'arrière-plan (**chargés d'assurer le fonctionnement interne d'Oracle).
- **Les mémoires Oracle** : parmi ces mémoires, les plus importantes sont :
  - **SGA** : il englobe diverses zones mémoires contenant des données et les informations de contrôles d'une instance Oracle. Elle est partagée par tous les processus du serveur et les processus en arrière-plan,
  - **PGA** : est une zone mémoire allouée au démarrage de la BD Oracle et est partagée aux utilisateurs. Il existe un PGA pour chaque processus serveur et le processus d'arrière-plan.

1.2.4. Exemple de SGBD Libre : PostgreSQL

PostgreSQL a été développé à l'université de Californie au département des sciences informatiques de Berkeley. PostgreSQL 10.2 est sorti le 06 février 2018, développé en C. Cette nouvelle version vient avec une numérotation exprimée sur 2 nombres contrairement aux autres versions (comme PostgreSQL 9.6.3). Il est largement reconnu pour son comportement stable, proche d'Oracle [W5].

TABLEAU 5 : AVANTAGES ET INCONVENIENTS DU SGBDR POSTGRESQL [8]

Avantages	**Inconvénients**
- Open Source et gratuit ; - Fiable et relativement performant, tout en restant simple à l'utilisation ; - Supporte la majorité du standard SQL-92 et possède un certain nombre d'extensions (java, Ruby, PL-SQL); - Administration simple; - Pas gourmand en ressources comparées à Oracle;	- La modification du fichier de sécurité pg_hba_conf nécessite un reboot pour être prise en compte; - Pas possibilité de requêter sur plusieurs BD : chaque BD nécessite sa connexion; - Pas de fonction d'agrégat OLAP; - Très limité pour de grandes BD ;

1.2.4.1. Architecture de PostgreSQL

Au regard de la **Figure 2**, nous voyons que PostgreSQL présente une architecture très simple.

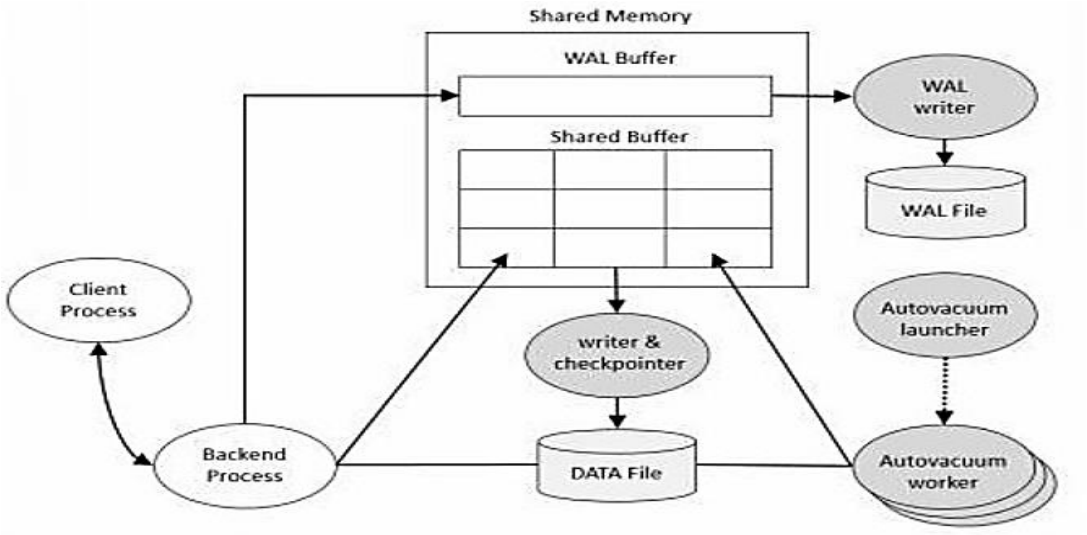


FIGURE 2 : STRUCTURE DE POSTGRESQL [9]

1.2.4.2. Les composants de PostgreSQL

Il se compose de mémoire partagée, de quelques processus d'arrière-plan et fichiers de données.

- **La mémoire partagée** : Cette mémoire est composée :
  - **Shared Buffer :** son but est de minimiser les entrées et sorties disque,
  - **WAL Buffer :** c'est un tampon qui stocke temporairement les modifications apportées dans la base de données. Les contenus stockés dans le tampon WAL sont écrits dans le fichier WAL à un moment prédéterminé.
- **Types de processus PostgreSQL :** PostgreSQL a quatre types de processus que sont : p**rocessus postmaster (**processus parent de tous les processus), p**rocessus de Fond (**chargé des opérations dans PostgreSQL), p**rocessus Backend (**exécutant la requête du processus utilisateur), p**rocessus Client (**processus en arrière-plan).
- **Fichier de données :** c'est l'ensemble qui constitue les données de la base de données.

1.2.5. Synthèse

Les deux SGBD étudiés présentent des structures différentes. Chacun de ses SGBD a des avantages et des inconvénients. Vu leurs architectures, nous voyons qu'Oracle permet à ses clients de stocker dans le **cloud** leur application critique, afin d'améliorer leur qualité et leur performance. Cette architecture permet de gérer des centaines de BD contrairement à PostgreSQL. Elle permet d'avoir un gain de temps de réponse considérable, la possibilité de gérer plusieurs requêtes sur les BD, alors que pour PostgreSQL, chaque BD nécessite sa propre connexion.

Au constat de tous ses avantages, ses architectures, nous remarquons qu'Oracle gère mieux les ressources et les performances devant PostgreSQL, ce qui est en parfaite adéquation par rapport le sujet traité, d'où notre choix d'Oracle dans sa version 12cR2.

1.3 - Les ressources

Les ressources d'un serveur de données sont les matériels, les logiciels connectés à ce serveur de données. Parmi ces ressources ceux qui nous intéressent sont :

- Le processeur (CPU) : est une puce permettant de faire des opérations arithmétiques et logiques plus ou moins complexes sur des données [10]. La puissance d'un processeur dépend : du nombre de cœurs du processeur, de la taille de sa mémoire cache, de l'architecture interne du processeur et de sa fréquence;
- La mémoire vive (RAM) : dont le rôle est d'être une zone de travail et de stockage temporaire. Les critères permettant d'évaluer la puissance d'une RAM sont : sa vitesse (Mb/s) de communication avec le CPU, sa fréquence (MHZ) de fonctionnement.

Conclusion

Dans ce chapitre nous avons d'abord défini les bases de données tout en donnant leur historique, leurs types et leurs méthodes d'implémentation. Ensuite nous avons présenté les SGBD, en donnant leur historique, leurs types. Nous avons étudié deux exemples de SGBD très utilisé dans le monde informatique : Oracle (propriétaire), PostgreSQL (open source et gratuit). Enfin nous avons terminé ce chapitre par parler des ressources utilisées par un serveur de données dans son fonctionnement.

Dans le chapitre 2 de notre première partie, nous allons parler de l'optimisation des bases de données. Ainsi, nous donnons quelques principes d'optimisations et leurs importances dans les performances d'un serveur de données.

Chapitre 2 : Optimisation des bases de données

Introduction

L'optimisation peut se définir comme étant la recherche de la meilleure manière possible pour réaliser une activité en utilisant le moins de ressources possible tout en diminuant les temps de réponse [11]. Ainsi l'optimisation d'une BD est un aspect crucial dans le fonctionnement d'un système d'information. Son objectif est de fournir aux utilisateurs un système optimal en termes de coût (CPU et RAM), de gain en performance, de rapidité dans le traitement des requêtes. Elle peut se situer à tous les niveaux, de la conception à l'exploitation. Dans ce chapitre, nous allons nous intéresser d'abord aux principes d'optimisations dans la Section 2.1. Dans la Section 2.2, nous montrons l'importance de l'optimisation dans les performances d'un serveur de données. Enfin dans la Section 2.3, nous présentons les techniques proposées par les deux SGBD étudiés au chapitre 1 pour optimiser les performances.

2.1 - Les principes d'optimisation

La question est la suivante : comment optimise-t-on ? L'optimisation est un processus itératif. Pour une base de données l'optimisation est définie [12] :

- De façon préventive : c'est-à-dire dès la conception du schéma de base de données;
- De façon curative : c'est-à-dire dès que la base de données rencontre des problèmes en production ou en implémentation.

Avant de répondre à la question posée, on voit bien que l'optimisation peut se faire au niveau : schéma de base de données, requêtes et infrastructure matérielles [12] [13].

2.1.1. Optimisation d'un schéma de base de données

Il est nécessaire d'accorder une attention particulière aux performances lors de la conception d'un schéma de base de données. Pour avoir un schéma de base de données optimal, il faut :

- Respecter les 3 formes normales. Aussi tenir compte des formes normales de Boyce Codd, les 4é et 5é forme normale;
- Dénormaliser le schéma si nécessaire.

L'impact de l'optimisation d'un schéma est de :

- Réduire l'espace disque utilisé par une base de données, en normalisant;
- Diminuer le temps d'exécution des requêtes d'un serveur de données en dénormalisant.

2.1.2. Optimisation des requêtes SQL

Une application bien conçue peut se heurter à des problèmes de performances si les requêtes SQL utilisées sont mal construites. Ainsi pour optimiser une requête il faut [12] [14]:

- Mesurer la requête : c'est-à-dire recueillir les résultats de l'exécution de la requête, pour pouvoir voir si la requête est optimale;
- Écrire des requêtes simples;
- Éviter des sélections comme SELECT * From Etudiant … en indiquant les attributs de la sélection comme suit SELECT nom, prenom,…, From Etudiant;
- Éviter des insertions ligne par ligne en mettant les requêtes dans des transactions;
- Utiliser des index.

Une autre technique d'optimiser les requêtes consiste à utiliser les hints. Ils permettent d'obliger l'optimiseur dans le choix d'un plan d'exécution. Leur syntaxe générique est : SELECT/_+NomHints*/colonne(s) From table(s) Where…,

L'optimisation d'une requête à un impact significatif sur l'amélioration des performances d'un serveur de données. En effet une requête optimisée permet de [6] [14] :

- Diminuer le temps de réponse d'un serveur de données;
- Réduire la consommation excessive de mémoire;
- Diminuer la consommation du CPU;
- Limiter le nombre E/S disque.

2.1.3. Optimisation des ressources d'un serveur de données

Il s'agira ici de parler de [12]:

- L'Optimisation de la mémoire RAM ;
- L'Optimisation du CPU.

L'optimisation des infrastructures matérielles d'un serveur de données permet de [12]:

- Réduire les accès disque en optimisant la RAM;
- Faciliter l'équilibrage des charges en optimisant le CPU;
- Faciliter le traitement et l'exécution des instructions;
- Diminuer le temps de réponse des serveurs de données;
- Exécuter de très grand nombre de processus;
- Accélérer l'accès aux données de la base régulièrement accédées;
- Optimiser la communication entre les processus et la base de données;
- Mettre les processus en mémoire

2.2 – Principes d'optimisation des SGBD étudiés

2.2.1. L'optimisation sous Oracle

2.2.1.1. L'optimisation des requêtes sous Oracle

Depuis sa version 7, Oracle est pourvu de deux optimiseurs de requêtes que sont : **le RBO** introduit en version 6 et le **CBO** introduit en version 7.

- **Optimiseur à Règle (RBO) :** il repose sur l'utilisation d'un ensemble de règles pour déterminer le plan d'exécution d'une requête. Ce type d'optimiseur est utilisé uniquement par Oracle depuis sa version 7 [15]. Le RBO donne les conditions nécessaires d'utilisation d'un index et comment un index sera utilisé.
- **Optimiseur à base de Coût (CBO) :** c'est une technique d'optimisation basée sur l'estimation des statistiques et coûts d'exécutions des requêtes. Les ressources utilisées pour évaluer le coût sont : Le Temps CPU, le nombre d'I/O disque et la taille de mémoire vive (RAM) nécessaire [12].

2.2.1.2. L'optimisation des ressources sous Oracle

Les techniques utilisées par Oracle pour optimiser les ressources sont :

- **Gestion de la mémoire sous Oracle :** Pour une optimisation de la mémoire, Oracle utilise différents paramètres selon la méthode de gestion de mémoire souhaitée.
  - **La gestion automatique de la mémoire :** dans ce cas, Oracle distribue automatiquement la taille de la mémoire entre le SGA et le PGA. Ainsi **memory_target** spécifie la taille de mémoire cible et **memory_max_target** spécifie une limite de la taille cible;
  - **la gestion manuelle de la mémoire** : le dimensionnement des mémoires est à la charge du DBA. C'est au DBA de définir les tailles de mémoires SGA et PGA.
- **Oracle Resource Manager 12c :** RM permet de gérer l'allocation des ressources aux PDB. Pour ce faire RM utilise des **Plans de ressources** (spécifient comment les ressources sont allouées aux différents groupes de consommateurs), des **Directives** (contrôlent l'allocation de ressources CPU, la limitation de nombre de sessions utilisateurs actifs, la taille mémoire RAM, les serveurs d'exécution parallèle) et les **groupes de consommateurs** (regroupent un ensemble de sessions d'utilisateurs en fonction de leurs besoins de traitement) pour optimiser les ressources.

2.2.1.3. Les outils d'optimisation sous Oracle

Oracle fournit un ensemble d'utilitaires facilitant l'optimisation de requêtes SQL. Parmi eux nous pouvons citer :

- **SQL Tuning Advisor :** c'est un outil de conseils qui permet d'analyser une requête et d'en dégager des propositions d'amélioration. Il est accessible depuis la console web, depuis la feuille de calcul SQL, depuis les écrans de suivi de la performance**;**
- **Explain plan :** c'est un utilitaire en mode ligne de commande permettant d'afficher le plan d'exécution d'une requête SQL.

2.2.2. L'optimisation sous PostgreSQL

2.2.2.1. L'optimisation des requêtes sous PostgreSQL

L'optimiseur de requêtes de PostgreSQL encore appelé le **Planner** se base sur **les coûts (CBO)** pour estimer les performances d'une requête. Il calcule le **coût** de chaque plan. Pour cela, il dispose d'informations sur les données (des statistiques), sur la configuration et d'un ensemble de règles inscrites en dur. Une fois tous les **coûts** calculés, il sélectionne le plan qui a le plus petit coût. Le CBO cherche toujours le plan d'exécution optimal en termes de consommation de ressources (CPU, RAM) et de temps de réponse d'un serveur de données.

2.2.2.2. L'optimisation des ressources sous PostgreSQL

Les techniques utilisées par PostgreSQL pour optimiser les ressources sont :

- **Optimisation de la RAM sous PostgreSQL :** il est intimement lié à la taille de mémoire physique disponible. PostgreSQL utilise différents paramètres pour optimiser sa mémoire afin d'obtenir de meilleures performances. Parmi eux nous avons [16][13] :
  - **shared_buffers :** il permet d'initialiser la taille de mémoire que le serveur de données utilise comme mémoire partagée. La valeur par défaut de ce paramètre peut être automatiquement abaissée si la configuration du noyau ne la supporte pas. Une valeur significative est généralement nécessaire pour de bonnes performances. Ce paramètre ne peut être configuré qu'au lancement du serveur,
  - **work_mem :** il indique la taille de mémoire que les opérations de tri internes et les tables de hachages peuvent utiliser avant de basculer sur des fichiers disque temporaires.
- **Optimisation du CPU sous PostgreSQL :** PostgreSQL est un système multiprocessus. Chaque connexion d'un client est gérée par un processus, responsable de l'exécution des requêtes et du renvoi des données au client. Par conséquent, chaque requête exécutée est traitée par un cœur de processeur. Le choix du processeur se fait suivant le type d'utilisation du serveur :

- Une majorité de petites requêtes en très grande quantité : privilégier le nombre de cœurs;
- Une majorité de grosses requêtes en très petite quantité : privilégier la puissance processeurs.

2.2.2.3. Les outils d'optimisation sous PostgreSQL

- **pgbench** est un outil pour réaliser des tests de performance. Il est capable de créer son propre environnement de test permettant de tester une configuration PostgreSQL en termes de performance, en termes de charge;
- **explain analyze** permet d'afficher le plan d'exécution d'une requête (algorithmes utilisés, accès effectués, jointures utilisées), avec à chaque fois une estimation du temps que l'opération prendra, et le temps qu'elle a réellement pris [16].

2.2.3. Synthèse

Les deux SGBD proposent des techniques d'optimisation à plusieurs niveaux : requêtes, matériels. Ils présentent des outils d'optimisations diverses. L'optimiseur Oracle peut se baser sur les règles ou coûts alors que pour PostgreSQL, son optimiseur se base sur les coûts. Au regard de tout, on voit qu'Oracle permet de mieux optimiser que PostgreSQL, il gère mieux les requêtes parallèles, les ressources (RAM, CPU), et présente des d'outils d'optimisation très avancés et évolués, qui s'adaptent à une multitude de technologies comparée à PostgreSQL.

Conclusion

Dans ce chapitre, nous avons parlé de l'optimisation d'une base de données. Nous avons présenté quelques méthodes d'optimisation avant de donner l'importance de l'optimisation dans les performances d'un serveur de données. Enfin, nous avons montré quelques techniques d'optimisations utilisées sous Oracle et sous PostgreSQL. En effet, on voit bien que l'optimisation d'une base de données est d'une importance capitale dans les performances d'un serveur de données.

Dans la suite de notre travail (partie 2), nous utilisons les bases de données relationnelles, car elles répondent le plus à notre besoin. Pour les langages de requêtes, nous utilisons SQL, car permettant de manipuler les BDR. L'implémentation de la base de données relationnelle se fera sur le SGBDR Oracle 12cR2. Il offre un véritable environnement de travail adapté à une multitude de technologies et présente un ensemble d'outils d'administration, de gestion des performances, de surveillances de BD très variés et très évolués. En plus, Oracle 12cR2 est l'un des leaders mondiaux des SGBD rencontrés sur le marché et l'un des plus populaires [w6].

Deuxième partie : Application et Réalisation

Chapitre 3 : Dimensionnement des mémoires

Chapitre 4 : Implémentation des Recommandations

Chapitre 3 : Dimensionnement de mémoires

Introduction

La mémoire est très sollicitée dans le fonctionnement d'un serveur de données. La mémoire Oracle est partagée entre le SGA et le PGA. Ainsi, on peut définir le dimensionnement de la mémoire Oracle comme étant l'allocation d'espace mémoire à chacun d'eux. Dans ce chapitre nous allons faire, dans la Section 3.1, un rappel sur la distribution de la mémoire entre le SGA et le PGA, puis, dans la Section 3.2, une présentation des méthodes d'optimisations de mémoires sous Oracle et enfin, dans la Section 3.3 des recommandations de dimensionnement de mémoire pour de meilleures performances.

3.1 - Distribution de la mémoire Oracle

Une base de données peut fonctionner soit, en environnement OLTP, en environnement DSS ou OLAP, en environnement OLTP et DSS.

OLTP (OnLine Transaction Processing) est le modèle utilisé par les SGBD. Le mode de travail est transactionnel. L'objectif est de pouvoir insérer, modifier des enregistrements et interroger rapidement en toute sécurité la base de données [w7]. DSS (Decision Support System) ou OLAP (On Line Analytical Processing) est le modèle utilisé par les datawarehouses. Ce système travaille en lecture seulement. Le but est de consulter d'importantes masses de données [w7]. Ce mode de fonctionnement est observé dans les applications décisionnelles ou de reportoring.

La distribution de la mémoire Oracle diffère selon les environnements dans lesquels la base de données fonctionne. En OLTP, Oracle recommande de conserver 20% de la mémoire au PGA et 80% de la mémoire au SGA [13]. Dans un système DSS, 50% à 70% de la mémoire sont données au PGA et le restent au SGA [17].

Dans la Section suivante, nous allons parler de l'optimisation des mémoires sous Oracle. Il s'agira de faire une présentation des méthodes d'optimisations du SGA et du PGA.

3.2 - Optimisation du SGA (System Global Area)

C'est une zone mémoire regroupant un ensemble de mémoires partagées. Elle comprend en grand partie du : Shared Pool, Log Buffer et Cache de base de données. Initialement, le SGA est réparti comme suit [18][w8] :

- 50% de Cache de base de données,

- 40% de Shared Pool,
- 10% de Redo Log Buffer.

La taille de la mémoire SGA est configurée à l'aide du paramètre sga_target qui représente la taille cible et du paramètre sga_max_target qui est la taille limite. Pour connaitre l'espace libre dans le SGA **:** select * from v$sgastat where name = 'free memory'.

3.2.1. Optimisation du Shared Pool

Il contient les instructions SQL, les procédures stockées et les informations spécifiques du dictionnaire de données. Le Shared pool est géré par l'algorithme LRU. Cet algorithme fonctionne comme les pointures, chaque pointeur fait référence à un bloc de données dans la cache. La taille du Shared pool est définit par le paramètre shared_pool_size. Ainsi, pour optimiser le Shared pool, il est nécessaire d'optimiser les mémoires suivantes : Library Cache, Dictionary Cache.

3.2.1.1. Optimisation du Library cache

Il contient le code SQL et les plans d'exécutions associés, les blocs PL/SQL, les classes Java à partagés entre utilisateurs. Ainsi, pour optimiser le Library Cache, nous avons principalement deux objectifs à atteindre.

- **Réduire l'absence de données :** on parle d'absence de données dans le Library cache lorsque les requêtes exécutées par un utilisateur ne sont pas mises en cache mémoire. Cette absence de données oblige le serveur Oracle de faire une analyse répétitive des SQL déjà exécutés. Par exemple :
  - Si une instruction SQL est lancée alors qu'elle n'existe pas dans la zone SQL partagée du Library cache, Oracle analyse l'ordre et alloue une zone SQL partagée à cette instruction;
  - Si un appel SQL est lancé alors que la zone SQL partagée est libérée du Library Cache pour faire place à un autre ordre, Oracle analyse implicitement la requête SQL et lui alloue une nouvelle zone SQL partagée;
  - Si l'objet d'un schéma est référencé dans un ordre SQL et qu'on le modifie par la suite, la zone SQL partagée devient invalide.

La réduction d'absence de données permet de minimiser les analyses répétitives des ordres SQL déjà exécutés par un utilisateur, car les requêtes sont mises en cache à chaque exécution. Aussi le besoin en taille mémoire est réduit, car les utilisateurs ou les applications utilisent le même groupe d'instruction SQL

- **réduire la fragmentation :** on parle de fragmentation de données lorsque les données ne sont pas positionnées physiquement les unes à la suite des autres dans la mémoire (fragmentation externe) ou quand les blocs de données ne sont pas remplis complètement (fragmentation interne) [19]. La lecture des données est infiniment plus rapide lorsqu'ils sont contigus. Cela s'explique du fait que la tête de lecture de la mémoire (ou disque) ne se déplace pas après chaque donnée lue sur disque. Pour réduire la fragmentation, il faut :
  - Allouer un espace réservée dans la zone Shared pool afin d'avoir suffisamment d'espace contigu pour les besoins importants en mémoire ;
  - Compacter les données pour que les blocs de données soient remplis ;
  - Maintenir en mémoire les objets volumineux les plus fréquemment demandés ;
  - Utiliser de petites fonctions PL/SQL au lieu de larges blocs PL/SQL anonymes.

La réduction de la fragmentation permet de diminuer l'espace occupé par les objets et de réduire le temps d'accès à ces données sur le serveur de données.

3.2.1.2. Outils de diagnostic du Library Cache

Ils permettent de monitorer, d'obtenir des conseils sur l'utilisation du Library cache. Parmi ces vues dynamiques, nous pouvons noter :

V$librarycache : elle contient l'ensemble des statistiques sur les performances et l'activité du cache de la bibliothèque de données.

V$db_object_cache : liste les objets de la base de données mis en cache dans la bibliothèque. Ces objets peuvent être des tables, index, les packages, synonymes, etc.

V$library_cache_memory : permet de consulter la taille de mémoire allouée aux objets du Library Cache.

3.2.1.3. Optimisation du Dictionary cache

Il stocke les copies des blocs de données lus à partir des fichiers de données. Ces blocs de données contiennent des informations de références sur la base de données, ses structures et ses utilisateurs. Le Dictionary cache est très solliciter par le serveur Oracle lors des analyses de requêtes SQL.

Pour optimiser cette zone de mémoire, il est conseillé d'examiner l'activité du Dictionary Cache après un certain temps de fonctionnement du dictionnaire. LRU est au cœur du maintien des données, de la libération d'espace inutilisé et de l'optimisation de cette mémoire cache. Ainsi, l'optimisation du Dictionary cache permet de diminuer la consommation inutile de ressources. Cela permet de libérer de l'espace pour d'autres requêtes SQL.

3.2.1.4. Outils de diagnostic du Dictionary Cache

C'est l'ensemble des vues dynamiques qui donnent des statistiques, des conseils d'utilisation de cette mémoire. Parmi ces vues nous allons parler de la vue principale V$rowcache. Cette vue permet d'afficher les statistiques et l'activité du dictionnaire de données. Chaque ligne de cette vue montre les statistiques d'un cache donné. La description de cette vue indique un certain nombre de colonnes importantes, parmi eux nous avons :

Parameter : indique le nom du paramètre déterminant le nombre d'entrées dans le cache du dictionnaire de données.

Gets : affiche le nombre total de demandes d'information sur une catégorie d'éléments du dictionnaire de données

Getmisses : cumul des demandes manquées sur l'élément (c'est-à-dire les requêtes qui échouent sur ces catégories).

3.2.2. Optimisation du Tampon de données

La taille de cette mémoire est définie par db_cache_size. Pour de nombreux types d'opérations, Oracle utilise le cache tampon de données pour stocker les blocs de données lus à partir du disque. Lors de la configuration d'une nouvelle instance de base de données, il est impossible de connaitre la taille correcte pour le cache de mémoire tampon.

Pour utiliser efficacement le cache de tampon de données, la syntonisation des instructions SQL est nécessaire. Pour atteindre cet objectif, une vérification des instructions SQL souvent exécutées et des instructions SQL exécutant de nombreuses données mise en mémoire tampon est nécessaire. La syntonisation SQL permet d'éviter la consommation inutile de ressources. De plus, un tampon optimisé permet de minimiser les accès disque qui sont couteux.

3.2.3. Outils de diagnostic du Tampon de données

C'est l'ensemble des vues dynamiques qui permettent d'obtenir d'informations sur la configuration du tampon de données. Parmi ces vues, nous pouvons noter :

V$bd_cache_advice : cette vue affiche les taux d'échec simulés pour une plage de tailles de cache tampon potentielles. Cette vue assiste le dimensionnement du cache en fournissant des informations indiquant le nombre de lectures physiques pour chaque taille potentielle de cache.

V$sysstat : c'est une vue de performance permettant d'afficher les statistiques cumulatives à l'échelle de l'instance depuis le démarrage de l'instance. Elle aide à connaitre le taux d'accès du tampon de données. Ce taux représente la fréquence à laquelle un bloc demandé a été trouvé dans le cache du tampon sans nécessiter l'accès au disque. Il est utilisé pour vérifier les entrées et sorties physiques prévues par la vue V$db_cache_size.

3.2.4. Optimisation du Log Buffer

Il contient les enregistrements des images avant et après des données en cours de modification. Le tampon redo log est circulaire et stocke les enregistrements dans les entrées redo. Les entrées Redo sont principalement utilisées pour la récupération de la base de données si nécessaire. Le tampon de journal redo log est dimensionné à l'aide du paramètre log_buffer.

Pour optimiser le redo log, les processus du serveur génèrent des données redo dans le tampon de journal puisqu'elles apportent des modifications aux blocs de données dans le tampon. Le processus LGWR écrit par la suite les entrées du tampon de journalisation de redo dans le journal redo en ligne. Ainsi, une bonne optimisation de cette mémoire permet de diminuer le temps d'écriture du LGWR dans le fichier redo-log, améliorant les performances du serveur de données. Aussi un débit élevé est constaté en optimisant le log buffer, permettant les applications OLTP de s'exécuter rapidement.

3.2.5. Outils de diagnostic du Redo Log Buffer

C'est l'ensemble des paramètres, statistiques indiquant soit, des événements d'attentes, le dimensionnement, la configuration, liées au log buffer. Parmi ces vues nous allons parler de la vue principale V$Parameter.

V$Parameter : cette vue indique la valeur actuelle du log buffer.

Nous donnons la description de quelques colonnes de la vue V$sysstat donnant les statistiques pertinentes du cache redo log.

Redo buffer allocation retries: il indique le nombre de fois où un processus utilisateur a attendu avant d'écrire dans le redo log buffer.

Redo log space wait time : il donne le temps cumulé attendu par tous les processeurs dans le tampon journal.

3.3 - Optimisation du PGA (Program Global Area)

Le PGA est scindé en deux zones : les zones de travails et celles privées. Il est créé par Oracle lorsqu'un processus serveur est démarré. L'accès au PGA est exclusif à un processus du serveur et il est lu et écrit uniquement par un code Oracle agissant en son nom. Les paramètres pga_aggregate_target et pga_aggregate_limit_target donnent respectivement la taille cible et une limite du PGA. Une zone SQL privée contient des données telles que les informations de liaison et des structures de mémoire d'exécution. Chaque session qui émet une instruction SQL possède une zone SQL privée. La mémoire de session est la mémoire allouée pour contenir les variables d'une session et d'autres informations liées à la session.

3.3.1. Taille Optimale du PGA

Lors du dimensionnement du PGA, l'objectif est de s'assurer que la majorité des zones de travails fonctionnent avec une taille optimale supérieure ou égale à 90% du PGA dans les systèmes OLTP purs et moins de 10% du PGA est donné aux autres zones de travail restantes [w9]. Une importante taille donnée aux zones de travail peut améliorer considérablement les performances au détriment d'une consommation élevée de mémoire. D'où il faut faire un compromis entre la taille allouée aux zones de travails et mémoire PGA.

L'optimisation du PGA consiste à optimiser chacune de ces deux parties : les zones de travails et les zones SQL privées. Cependant nous nous intéressons à l'optimisation des zones de travails, qui ont plus d'influences sur les performances du PGA comparé aux zones SQL privées. Une bonne optimisation des zones de travail améliore les performances du PGA à 90%. Pour optimiser les zones de travails, il faut [w10] **:**

3.3.2. Régler le PGA à la taille optimale.

C'est la taille mémoire permettant de créer autant de zones d'exécutions que de données entrantes. Elle est la taille idéale et suffisante pour les zones de travail.

3.3.3. Réduire les Passes dans le PGA

C'est quand les zones de travail sont configurées à une taille inférieure à la taille optimale, entrainant une légère augmentation du temps d'exécution d'un serveur de données sous Oracle.

3.3.4. Éviter des Multi-Passes

Dans une situation de Multi-Passes plusieurs passages sur les données d'entrées sont nécessaires augmentant nettement le temps d'exécution d'un serveur de données sous Oracle, car la taille de la zone de travail est trop petite.

3.3.5. Outils de diagnostics du PGA

Dans cette partie nous allons donner quelques vues dynamiques permettant de donner des statistiques, des conseils, sur l'utilisation de la mémoire PGA. Parmi ces vues nous avons :
V$pgastat : indique les statistiques sur l'utilisation du PGA au niveau de l'instance.
V$sql_workarea_histogram : montre le nombre de zones de travails qui fonctionne, avec une taille optimale, en mode passe-unique, en mode multi-passe.
Donnons quelques colonnes indiquant des statistiques pertinentes sur l'utilisation de la mémoire PGA. Ces colonnes appartiennent à la vue V$pgastat.

Total pga allocated : fait référence à la taille de mémoire allouée au PGA actuellement.

Cache hit percentage : indique le pourcentage de zones de travails exécuté avec une taille optimale.

Après avoir présenté les méthodes d'optimisations des mémoires sous Oracle. La question que nous nous posons est la suivante : quel est le bon dimensionnement de ces mémoires ?

3.4 - Recommandation de bon dimensionnement du SGA

Pour parler de bon dimensionnement du SGA, il est nécessaire de donner les recommandations de bon dimensionnement du : Shared Pool, Cache tampon, Log buffer.

3.4.1. Recommandation de bon dimensionnement du Shared Pool

Le bon dimensionnement du Shared Pool dépend du dimensionnement du Library cache, du Dictionary cache.

3.4.1.1. Recommandation de bon dimensionnement du Library Cache

D'abord il est important de connaitre la taille de mémoire partageable utilisée, ensuite vérifier les valeurs indiquées par des colonnes de la vue V$librarycache. Les résultats des trois requêtes ci-dessous sont en octets et ses résultats varient selon les bases de données**.**

Calcule de la mémoire partageable utilisée :

- Pour des objets stockés tels que les packages, fonctions et procédures;

```
Select sum (sharable_mem)
From v$db_object_cache
Where type in ('Package','Package Body','Function','Procedure');
```

```
SUM(SHARABLE_MEM)
-----------------
           443421
```

- Pour les ordres SQL, interroger V$sqlarea après une certaine période d'exécution de la base de données. Le choix de cette période est arbitraire et dépend des DBA, nous avons choisi de prendre après 5 heures d'exécutions de la base de données. Pour les ordres SQL fréquemment émis, utiliser la requête ci-dessous bien que celle-ci ne prenne pas en compte le SQL dynamique :

```
Select sum (sharable_mem)
From v$sqlarea
Where executions > 5;
```

```
SUM (SHARABLE_MEM)
------------------
           1557221
```

- Il est nécessaire de prévoir 250 octets par utilisateur et par curseur partagé dans le Shared pool. Ceci peut être testé aux heures de forte activité de la base de données.

```
Select sum (250*users_opening
From v$sqlarea;
```

```
SUM (250*USERS_OPENING)
----------
      7000
```

Nous citons quelques colonnes indiquant des statistiques pertinentes de la vue V$librarycache.

Reloads : c'est le nombre de fois qu'une requête mise en cache a été rechargée dû à une invalidation. Dans ce cas Oracle effectue une nouvelle analyse pour ces requêtes,

Invalidations : représente le nombre de fois qu'une requête SQL a été invalidée dans la mémoire, provoquant des rechargements,

Pins : c'est le nombre de fois qu'une demande sur un élément a été effectuée [w11],

Pinhits : représente le nombre de fois qu'une requête SQL exécuté sur un élément a été trouvé en mémoire cache [w11].

Le Taux d'accès au Library cache ou Hit Ratio du Library cache : représente le rapport qu'une demande SQL a été trouvée en mémoire sur le nombre de demandes SQL sur cet élément (**R2 Annexes**). Plus que ce taux est élevé moins sont les requêtes SQL invalidées.

Pour un meilleur dimensionnement du Library cache, il faut que :

- La taille de la mémoire partageable utilisée de la Library cache soit aussi grande que la somme des trois valeurs ci-dessus. Cependant il est important d'ajouter une petite taille à cette somme pour l'exécution des requêtes SQL dynamique, cette taille varie selon la taille de mémoire RAM qu'on dispose;
- **Reloads** soit le plus proche possible de zéro pour avoir une taille optimale [w12];
- **Invalidations** aussi doit être proche de zéro afin d'obtenir une taille optimale;
- **Le Taux** d'accès du Library cache doit être supérieur ou égal à 90% (**R2 Annexes**) [w13].

3.4.1.2. Recommandation de bon dimensionnement du Dictionary Cache

L'idéal pour un bon dimensionnement du Dictionary cache, il est important que :

- Le hit ratio du Dictionary cache soit inférieur à 10% ou 15% (**R3 Annexes**) [w12][w13]. Une taille mémoire suffisante doit être accordée à cette partie afin de minimiser les risques de requêtes SQL en échecs, etc.

3.4.2. Synthèse

Représentant 40% du SGA, l'optimisation du Shared Pool est primordiale dans l'amélioration globale des performances d'un serveur de données sous Oracle. L'optimisation du Library cache à un niveau acceptable de Hit ratio garantit également un Hit ratio acceptable dans le cache du Dictionnaire de données. Ce que nous pouvons retenir est :

- Si V$SHARED_POOL_ADVICE donne un résultat satisfaisant;
- Si les statistiques du Library Cache sont acceptables;
- Si le Taux du Dictionary cache est inférieur aux valeurs de référence (10% ou 15%)

Alors le pool partagé est correctement dimensionné.

3.4.3. Recommandation de bon dimensionnement du Tampon de données

Pour obtenir un bon dimensionnement du Tampon de base de données, il faut **:**

- Si le taux d'accès au cache (**R1 Annexes**) est faible (<< 90%) et que la BD doit éviter d'effectuer des analyses de tables complètes, augmenter la taille du cache de la mémoire tampon [20];
- Si le taux d'accès du cache des tampons est élevé (>>90%), le cache de la mémoire tampon est assez important pour stocker les données les plus fréquemment consultées. Si tel est le cas et que la mémoire est requise pour une autre structure de mémoire, réduire la taille du cache tampon [20];
- Syntoniser les requêtes pour éviter l'utilisation intitule de ressources dans le Cache de base de données.

3.4.4. Recommandation de bon dimensionnement du Redo Log

Le bon dimensionnement du log buffer dépend de la nature des applications clientes tournantes sur le serveur de données. En effet le bon dimensionnement se repose sur le respect des conditions suivantes **:**

- Si les applications effectuent des insertions, des modifications, des suppressions de gros volumes de données, il faut accorder une taille importante au redo log afin d'avoir un débit élevé entrainant une diminution du temps d'exécution d'un serveur de données;
- Il faut s'assurer que la valeur du paramètre **redo buffer allocation retries** (**Annexe R1**) est toujours proche de zéro.

3.4.5. Synthèse

L'optimisation du SGA est capitale dans la gestion des performances de la mémoire d'un serveur de données sous Oracle. La consultation des vues dynamiques et des statistiques permet d'avoir la taille allouée au SGA et les conseils d'utilisation du SGA. De plus, il est nécessaire de voir les statistiques indiquées par les composants du SGA. Pour cela il faut que :

- la mémoire partagée (Shared Pool) présente des statistiques acceptables;
- le taux de réussite du tampon de données est élevé;
- le log buffer ait une taille de mémoire importante pour garantir un débit élevé.

Ainsi, si ces composants (Shared Pool, Cache de BD, Log Buffer) sont bien dimensionnés alors le SGA est optimisé. Occupant la plus grande partie de la mémoire, un SGA bien dimensionner permet une amélioration considérable des performances d'un serveur de données sous Oracle.

3.5 - Recommandation de bon dimensionnement du PGA

Pour un bon dimensionnement du PGA, il faut s'assurer :

- Cache hit percentage soit le plus proche possible de 100%. Un ratio élevé permet d'éviter les échanges entre la mémoire vers le disque;
- Une importance taille mémoire est réservé aux zones de travails.

Conclusion

Dans ce chapitre nous avons parlé d'abord de la distribution de la mémoire Oracle entre SGA et PGA, ensuite pour chaque mémoire nous avons vu les techniques d'optimisation, des outils d'optimisation existants et enfin nous avons terminé par une recommandation de bon dimensionnement. Un bon dimensionnement des mémoires d'un serveur de données sous Oracle permet une amélioration significative des performances.

Pour montrer que les recommandations faites dans ce chapitre permettent d'améliorer les performances d'un serveur de données sous Oracle, le chapitre 4 est consacré à la partie pratique. Dans ce chapitre (chapitre 4), nous allons voir nettement l'importance du bon dimensionnement des mémoires Oracle. Ainsi, nous implémentons deux bases de données dont l'une utilise les recommandations proposées et l'autre la gestion automatique des mémoires proposée par Oracle.

Chapitre 4 : Implémentation des Recommandations

Introduction

Dans ce chapitre nous allons aborder la partie pratique de notre mémoire. Pour cela nous allons implémenter la même base de données de deux manières différentes selon les méthodes de gestion de la mémoire. Ainsi, dans la Section 4.1 nous allons faire une présentation de la base de données expérimentale, puis, dans la Section 4.2 de l'environnement de travail. Nous avons donné dans la Section 4.3, les métriques de mesures des performances. Ces métriques sont : Temps d'exécution des requêtes, la taille mémoire RAM nécessaire, le pourcentage du CPU consommé. Les outils utilisés sont décrits à la Section 4.4. Parmi, les outils utilisés, nous avons : plan d'exécution, vues systèmes, Oracle entreprise manager. Enfin nous avons terminé par faire des expérimentations et résultats à la Section 4.5, et des études de quelques requêtes et résultats à la Section 4.6. Le serveur utilisé pour l'expérimentation est Oracle 12cR2.

4.1 - La Base de données Expérimentale

Pour les tests nous avons utilisé la base de données d'une société constituée de trois tables [w14]. Ces tables sont : Augmentation, Bigemp et Bigdept qui contiennent respectivement des informations sur l'augmentation des salaires des employées, des informations sur les employées et les départements dans lesquels ces employées sont affectées. Cette base de données est nommée DRH et présente le schéma suivant :

AUGMENTATION (EmpNo, Montant);

BIGDEPT (DeptNo, Dname, Loc);

BIGEMP (EmpNo, Ename, Job, Mgr, Hiredate, Sal, Comm, DeptNo).

Le tableau ci-dessous présente le dictionnaire de données de la base de données DRH

TABLEAU 6 : DICTIONNAIRE DE DONNEES DE NOTRE BASE DE DONNEES

Attributs	Types	Descriptions
Empno	Number	Le numéro de l'employé
Montant	Number(7,2)	Le montant augmenté à l'employé
Deptno	Number	Le numéro du département où travail l'employé
Dname	Varchar2(14)	Le nom du service de l'employé
Loc	Varchar2(13)	Le lieu où se trouve le service de l'employé
Ename	Varchar2(10)	Le nom de l'employé
Job	Varchar2(9)	La fonction de l'employé
Mgr	Number	Le numéro de l'employé responsable direct d'un service

Hiredate	Date	La date d'embauche de l'employé
Sal	Number(7,2)	Le salaire de l'employé
Comm	Number(7,2)	La commission de l'employé

Le **Tableau 7** donne le nombre d'enregistrements de chaque table de notre base de données DRH que nous avons utilisée dans la pratique.

TABLEAU 7 : VOLUMETRIE DE LA BASE DE DONNEES DRH

Nom de Tables	**Nombre d'enregistrements**
BigEmp	25839
Bigdept	25785
Augmentation	25650

4.2 - Environnement de travail

Après téléchargement d'Oracle 12cR2 [w15], pour l'installer sous Windows, il faut [w16] :

- Un Système d'Exploitation (Windows 7, 8, 8.1, 10) de 64 bits uniquement;
- Architecture Système : Processeur (AMD64 et Intel EM64T);
- 10Go d'espace libre sur le disque de l'installeur;
- 2Go de RAM sur le système.

L'infrastructure matérielle sur laquelle les différents tests sont effectués est :

- Processeur AMD Athlone™ 64 X2 Dual-Core Processor TK-55 1.80 GHz;
- Mémoire installée (RAM) 3Go;
- OS: Windows 10 Professionnel 64 bits.

Nous avons l'implémentation de la même base de données DRH : Orcl et Orcl1.

Pour rappel les bases de données (Orcl et Orcl1) fonctionnent en environnement OLTP.

4.2.1. Orcl

Orcl est l'implémentation de la base de données DRH sous Oracle. Elle utilise la gestion automatique de la mémoire. Nous avons noté les dimensionnements suivants (voir **Tableau 8)**

TABLEAU 8 : DIMENSIONNEMENT DES MEMOIRES D'ORCL SOUS ORACLE

Mémoires	**Tailles Allouées (Mo)**
Shared Pool	320
Cache du Tampon	336
Pool java	16
Large Pool	32
PGA	162

4.2.2. Orcl1

Orcl1 est l'implémentation de la base de données DRH sous Oracle. Elle utilise les recommandations de gestion de mémoires étudiées au chapitre 3. Pour ce faire, nous avons attribué des tailles de mémoires comme suit (voir le **Tableau 9**)

TABLEAU 9 : DIMENSIONNEMENT DE MEMOIRES ORCL1 SOUS ORACLE

Mémoires	**Tailles Allouées (Mo)**
Shared Pool	370
Cache du Tampon	386
Pool java	10
Large Pool	40
PGA	220

4.3 - Métriques

Pour mesurer les performances apportées par les différentes méthodes de gestion de mémoires, il nous a fallu étudier quelques indicateurs et critères d'évaluations. Parmi les critères d'évaluation, nous nous sommes intéressées :

- Au temps de réponse d'un serveur de données;
- À la taille de mémoire RAM nécessaire;
- Au pourcentage de consommation du CPU.

Pour obtenir ces indicateurs, il est indispensable d'utiliser des outils et des méthodes d'exploitations.

4.4 - Outils et Méthodes Utilisés

L'étude des statistiques renseigne l'état d'une base de données et les ressources utilisées. Il existe plusieurs méthodes et outils utilisés pour mesurer les performances d'une base de données, nous avons utilisé quelques-uns : Plan d'exécution, Vues système, Oracle Entreprise Manager 12c (OEM).

4.4.1. Plan d'Exécutions des Requêtes SQL

Le plan d'exécution permet de voir les méthodes utilisées et le temps mis par un serveur pour exécuter une requête donnée. Dans un plan d'exécution nous retrouvons les chemins utilisés par l'optimiseur (les jointures, les index, les tables, les algorithmes), la taille de mémoire RAM utilisée, le pourcentage de consommation du CPU et le temps mis pour retourner les résultats. Dans le plan d'exécution d'une requête, l'indicateur le plus considéré est le temps de réponse du serveur.

4.4.2. Consultation des vues système

Cela consiste à récupérer les statistiques à partir des vues système. Elles sont utilisées pour calculer la consommation de ressources CPU (le coût (CPU)) et les entrées et sorties disques (coût (I/O) disque). Ainsi, les formules permettant de retrouver les coûts (CPU et IO) sont les suivantes :

Cette première équation représente l'équation pour estimer le coût (CPU+IO) [18] :

1) **Cost = (((#SRDS * sreadtm) + (#MRDS * mreadtm) + (#CPUCycles / cpuspeed)) / Sreadtm)**

Ou

#SRDS représente le nombre de lectures E/S en mode mono bloc

Sreadtm est le temps nécessaire à la lecture d'un bloc

#MDRS est le nombre de lectures multi blocs

Mreadtm le temps de lecture multi blocs

#CPUCycles le nombre de cycle CPU

Cpuspeed est le nombre de cycles CPU par seconde

L'équation ci-dessous fournit les composants nécessaires pour résoudre l'équation (1)

2)
Sreadtm = (ioseektm + (db_block_size / iotfrspeed))
Mreadtm = (ioseektm + (MBRC * db_block_size) / (iotfrspeed))
Cpuspeed = cpuspeedNW

Ou

Ioseektm représente le temps de recherche d'un bloc sur disque en ms, par défaut 10

Iotfrspeed la vitesse de transfert de données en bytes par ms, par défaut 4096

CpuspeedNW le nombre de moyen de cycles CPU par seconde, dépend du système

MBRC est le nombre de blocs lus en moyenne dans une opération E/S en mode multi blocs

Avec des transformations effectuées à l'équation (1) et l'application de (2) sur (1), nous pouvons finalement retenir que :

(3)
COST = IO_COST + CPU_COST
CPU_COST = Cpu_Cycles / (Cpuspeed * Sreadtm)

Cpu_cycles = le nombre des cycles CPU : colonne cpu_cost dans plan_table

Sreadtm = ioseektm + (db_block_size / iotfrspeed)

4.4.3. Oracle Entreprise Manager Express 12c (OEM)

C'est une interface Web fournie afin de faciliter l'administration de la base de données Oracle aux DBA. Cet outil donne une vision plus claire de l'activité de l'instance. Elle montre un résumé de : gestion des performances, gestion des mémoires, etc. Elle est accessible via un navigateur. Pour notre cas voici le chemin d'accès est : https://localhost:5500/em/login et la connexion est faite via l'utilisateur **SYS** avec le rôle **DBA.**

4.5 - Expérimentations et Résultats

Nous vérifions la méthode de gestion de mémoire utilisée par Orcl via SQL*Plus. C'est-à-dire voir si le SGA est déverrouillé et si la taille mémoire est attribuée au paramètre **memory_target** (taille cible de l'instance).

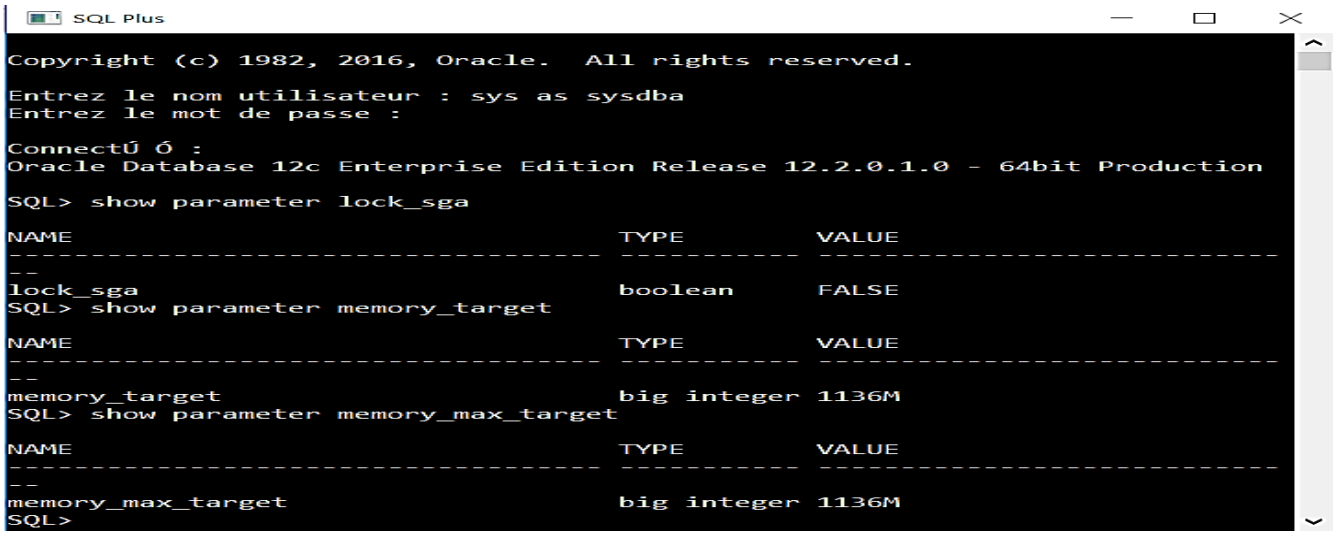


FIGURE 3 : VERIFICATION DE LA GESTION DE MEMOIRES UTILISEES PAR ORCL

4.5.1. Consultation des Plan d'exécution

Pratique: Prenons une requête donnant la liste des employés, leur travail, département affecté, avec un montant d'augmentation inférieur à 4000. Nous avons nommé R cette requête pour la suite. Elle est exécutée sur les deux bases de données (Orcl, Orcl1) pour voir les différents plans d'exécution donnée par cette requête sur les bases de données.

```
R: explain plan for
   Select ename, job, dname, montant from bigemp e, bigdept d, augmentation a
   Where e.empno=a.empno and e.deptno=d.deptno and montant < 4000;
```

Sur Orcl (**Figure 4**), la requête R donne le plan d'exécution suivant :

Analyse du résultat :

Après avoir exécuté la requête R, nous avons un temps de réponse égal à **03.17**secondes et le type d'optimisation utilisé à savoir le plan adaptatif utilisé. Ensuite nous notons :

- une double jointure de type hachage est effectuée par l'optimiseur à la ligne **1** et **3**,
- Les opérations **4** et **5** montrent que l'optimiseur a fait un balayage complet (Full Access) sur les tables Bigemp et Augmentation suivie d'une jointure de hachage entre ces deux tables pour extraire des informations,
- L'opération **2** indique que l'accès aux informations de la table Bigdept est fait par un balayage complet de table puis l'optimiseur effectue une seconde (2ème) jointure par hachage entre cette table et le résultat de la première (1ère) jointure,
- Et enfin en **0**, on voit que l'optimiseur a effectué une opération de sélection des attributs demandés.

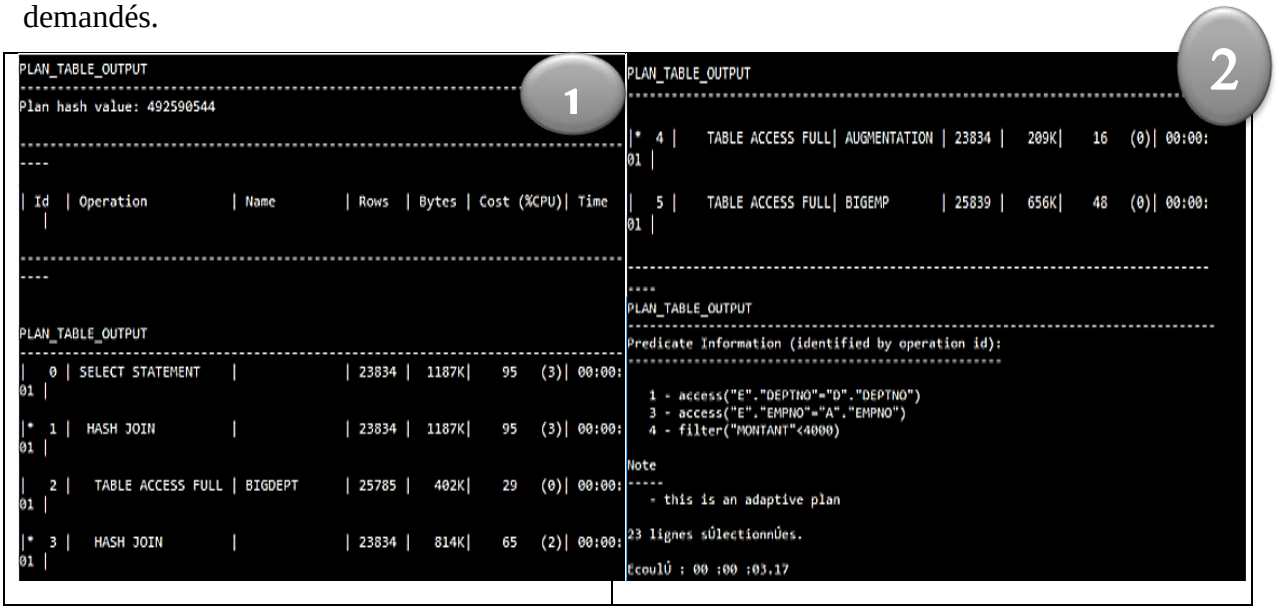


FIGURE 4 : PLAN D'EXECUTION DE R SUR ORCL

La requête R donne le plan d'exécution suivant sur Orcl1 (voir **Figure 5**) :

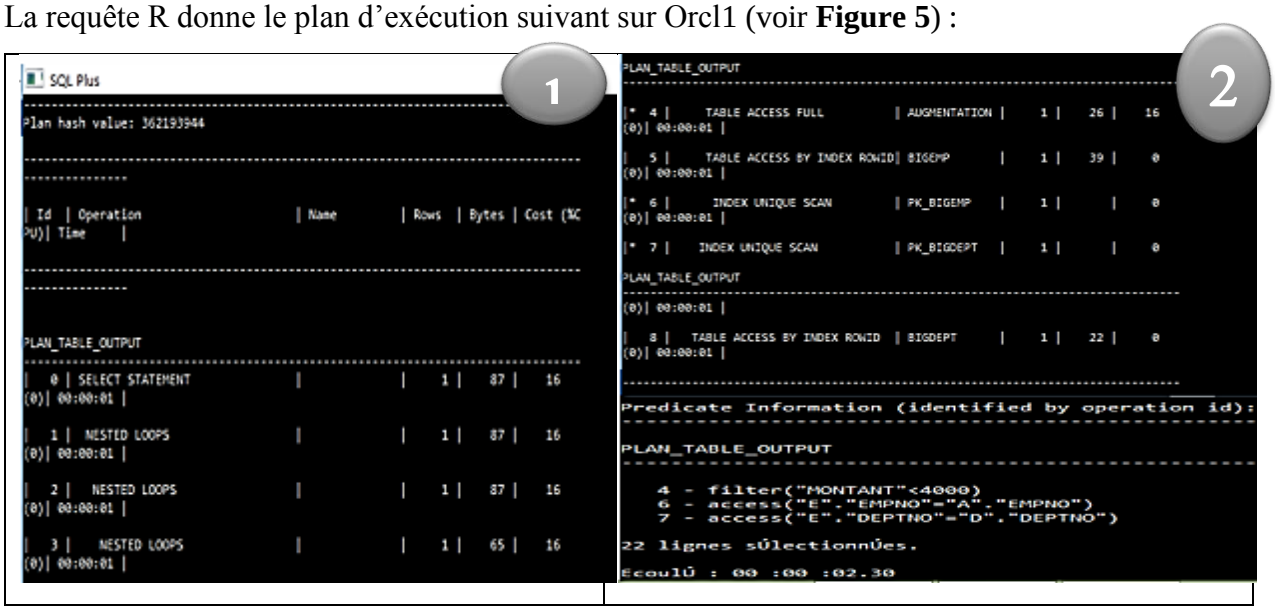


FIGURE 5 : PLAN D'EXECUTION DE R SUR ORCL1

Lecture du Plan d'Exécution Orcl1 :

Après exécution de la requête R (**Figure 5**), nous constatons un temps de réponse égal à **02.30** secondes. Ensuite nous avons noté :

- En **1**,**2** et **3,** nous voyons que l'optimiseur a utilisé trois boucles imbriquées (Nested Loops) pour trier le résultat,
- Les opérations **8** et **5** montrent que l'optimiseur a fait un accès par index (ROWID) pour extraire les informations des tables Bigemp et Bigdept,
- Les opérations **7** et **6** montrent l'opération par laquelle l'optimiseur a effectué sur l'index pour faire l'accès aux informations enregistrées des tables Bigemp et Bigdept,
- En **4**, l'extraction des informations contenues de la table Augmentation est faite par un balayage complet (Full Access),
- Enfin en 0, la sélection des attributs faisant la demande.

4.5.2. Utilisation des Vues système

Les valeurs des statistiques sont obtenues en lançant les requêtes sur les vues suivantes :

```
SQL> select sname,pname,pval1 from sys.aux_stats$
  2  where sname='SYSSTATS_MAIN';

SNAME                          PNAME                               PVAL1
------------------------------ ------------------------------ -----------
SYSSTATS_MAIN                  CPUSPEED
SYSSTATS_MAIN                  CPUSPEEDNW                     1723,18339
SYSSTATS_MAIN                  IOSEEKTIM                              10
SYSSTATS_MAIN                  IOTFRSPEED                           4096
SYSSTATS_MAIN                  MAXTHR
SYSSTATS_MAIN                  MBRC
SYSSTATS_MAIN                  MREADTIM
SYSSTATS_MAIN                  SLAVETHR
SYSSTATS_MAIN                  SREADTIM

9 lignes sÚlectionnÚes.

SQL> select value from v$parameter where name='db_block_size';

VALUE
--------------------------------------------------------------------------------
8192
```

FIGURE 6 : STATISTIQUE VUES SYSTEME

Sreadtm = ioseektm + (db_block_size / iotfrspeed)

= 10 + (8192 / 4096) = **12**

Cherchons le coût (CPU) et coût (I/O) disque en exécutant ces requetés sur les deux bases de données (Orcl et Orcl1). Les **Figure 7** et **Figure 8** permettent respectivement de calculer le coût (CPU+IO) donné par les bases Orcl et Orcl1. Pour avoir le coût (IO_COST), il suffit de lire la valeur affichée sur la colonne **IO_COST** des différentes figures.

Cpu_cost = cpu_cycles / (cpuspeed * sreadtm)

= 8718198 / (1723183,39 * 12) = 0.4216≈ 0.42

Io_cost = 48

Cost = io_cost + cpu_cost

= 48 + 0.42 = 48.42 ≈**48.42**

```
SQL> explain plan set statement_id='BigEmpSansSysStat' for select * from bigemp;

ExplicitÚ.

SQL> select * from table(dbms_xplan.display);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Plan hash value: 2368838273

-----------------------------------------------------------------------------
| Id  | Operation          | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |        | 25839 |  1034K|    48   (0)| 00:00:01 |
|   1 |  TABLE ACCESS FULL| BIGEMP | 25839 |  1034K|    48   (0)| 00:00:01 |
-----------------------------------------------------------------------------

8 lignes sÚlectionnÚes.

SQL> select id,cost,cpu_cost,io_cost from plan_table
  2  where statement_id='BigEmpSansSysStat';

        ID       COST   CPU_COST    IO_COST
---------- ---------- ---------- ----------
         0         48    8718198         48
         1         48    8718198         48
```

FIGURE 7 : COUT (CPU+I/O) DONNE PAR LES VUES D'ORCL

```
SQL> explain plan set statement_id='BigEmpSansSysStat' for select * from bigemp;

ExplicitÚ.

SQL> select * from table(dbms_xplan.display);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Plan hash value: 2368838273

-----------------------------------------------------------------------------
| Id  | Operation          | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |        |     1 |    87 |    48   (0)| 00:00:01 |
|   1 |  TABLE ACCESS FULL| BIGEMP |     1 |    87 |    48   (0)| 00:00:01 |
-----------------------------------------------------------------------------

8 lignes sÚlectionnÚes.

SQL> select id,cost,cpu_cost,io_cost from plan_table
  2  where statement_id='BigEmpSansSysStat';

        ID       COST   CPU_COST    IO_COST
---------- ---------- ---------- ----------
         0         48    1224888         48
         1         48    1224888         48
```

FIGURE 8 : COUT (CPU+I/O) DONNE PAR LES VUES D'ORCL1

Cpu_cost = cpu_cycles / (cpuspeed * sreadtm)

= 1224888 / (1723183,39 * 12) = 0.0592 ≈ **0.06**

Io_cost = **48**

Cost = io_cost + cpu_cost

= 48 + 0.06 = 48.06 ≈ **48.06**

4.5.3. Comparaison des Résultats des plans d'exécutions

Pour ce faire nous allons présenter nos résultats obtenus sous forme de tableaux. Ensuite nous représentons ces résultats dans des graphes. Enfin nous donnons des analyses et interprétations. La taille de mémoire RAM utilisée est calculée en faisant la somme des valeurs présentes dans la colonne **Bytes** des résultats des requêtes effectuée sur Orcl et Orcl1**.** Pour la conversion des résultats de **bytes** en **Mo** voir **Annexes**

4.5.4. Résultats obtenus

Le **Tableau 10** donne le résumé du temps mis par la requête R pour retourner les résultats, la taille de mémoire RAM utilisée par R et le pourcentage de consommation du CPU, sur les deux bases de données.

TABLEAU 10 : INDICATEURS MESURES SUR LES DEUX BASES DE DONNEES

Paramètres \ BD	Orcl	Orcl1
Temps (s)	00 :00 :03.17	00 :00 :02.30
RAM (Mo)	4,35	0,000393
Coût (CPU+IO)	48,42	48,06

4.5.5. Représentation graphique des Résultats obtenus

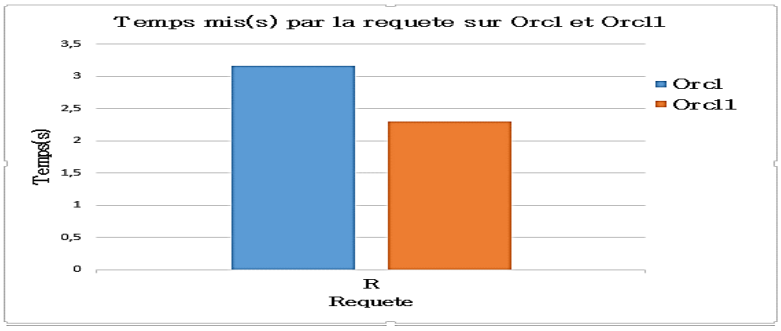


FIGURE 9 : HISTOGRAMME DU TEMPS MIS PAR LA REQUETE SUR LES BD

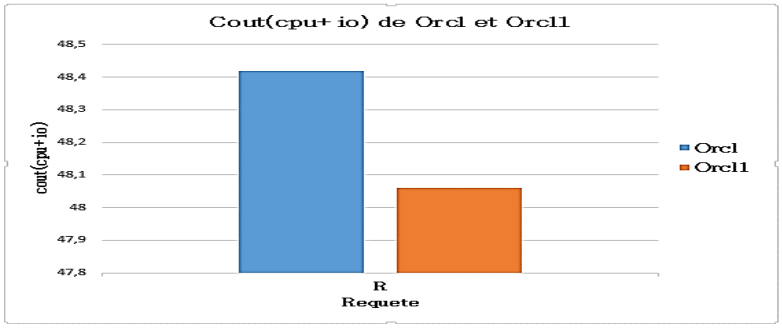


FIGURE 10 : HISTOGRAMME COUT (CPU+IO) DES DEUX BASES DE DONNEES

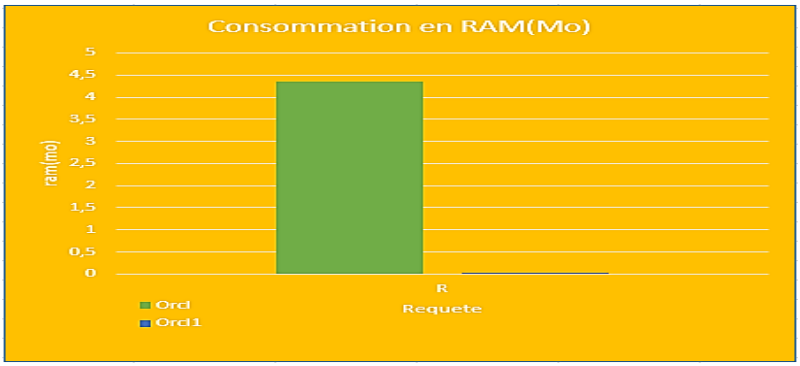


FIGURE 11 : HISTOGRAMME DE LA CONSOMMATION EN RAM DES BD

Nous avons choisi des histogrammes, car simple et facile pour ressortir les différences et à interpréter. Dans la section suivante, nous commençons l'analyse et l'interprétation des histogrammes.

4.5.6. Analyse et Interprétation des Résultats obtenus

En vue des trois Histogrammes (**Figure 9…Figure 11),** nous voyons clairement que la base de données Orcl1 présente un temps de réponse inférieur à la base de données Orcl, aussi elle a un coût (CPU+IO) moins important par rapport à Orcl. Orcl consomme une taille de mémoire RAM élevée comparée à la base de données Orcl1.

- La différence du temps d'exécution constatée peut s'expliquer du fait que sur Orcl, l'optimiseur a effectué une analyse complète des trois tables qui est lent alors que sur Orcl1, l'optimiseur a utilisé l'index pour retrouver très rapidement les informations.

- Le coût (CPU+IO) élevé d'Orcl s'explique par l'utilisation de HASH JOIN en 1 et 3 sur le plan d'exécution. Cela a valu à l'optimiseur respectivement une consommation en ressources égale à 65 et 95. Sur Orcl, l'optimiseur a parcouru la totalité des enregistrements des trois tables (Full Access) ce qui a entrainé un coût élevé comparé à Orcl1 qui utilise des index pour accéder aux informations recherchées. Aussi ce résultat s'explique du fait que le temps mis sur Orcl est supérieur à celui mis sur Orcl1 pour exécuter la requête.

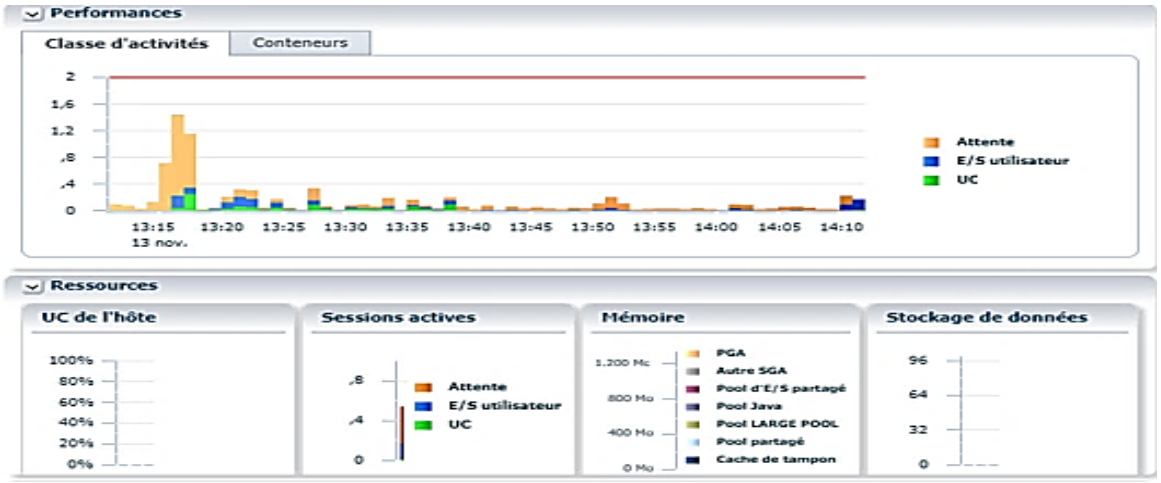
- La consommation excessive de RAM par Orcl par rapport à Orcl1 s'explique par l'utilisation de la jointure HASH JOIN. Dans une jointure de hachage, la base de données effectue un balayage complet de table, puis génère une table de hachage dans la RAM et recherche ensuite les lignes respectant la condition à une autre table (la grande table). Alors que les jointures imbriquées (NESTEED LOOP) se servent des index pour trier les résultats.

- Ce résultat est dû aussi à l'allocation de tailles mémoires des deux bases de données. Pour Orcl nous avons noté le dimensionnement suivant : Shared pool = 320Mo, cache de mémoire tampon=336Mo, et pour Orcl1 on a : Shared pool=370Mo, cache tampon=386Mo. L'augmentation de la taille du Shared pool permet d'éviter l'analyse répétitive d'une requête donnée, cela facilite une réponse très rapide aux utilisateurs. Une taille importante du cache tampon permet de garder plus de données dans le cache, cela permet d'éviter les accès disques qui sont couteux. Pour obtenir de bonnes performances au niveau de l'instance il convient d'agrandir ces mémoires afin de limiter les échanges entre mémoires vives (RAM) et disque.

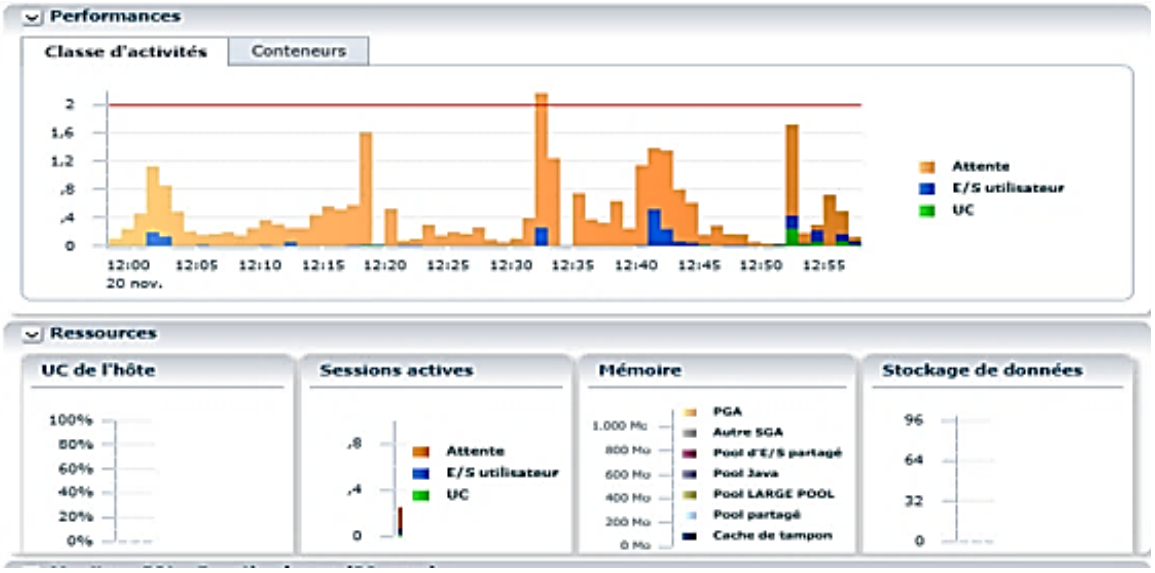
4.5.7. Visualisation d'OEM

Les deux écrans ci-dessous fournissent des informations générales sur l'état des bases de données Orcl et Orcl1. Ils montrent une description globale des performances, la charge des

bases de données, la consommation de ressources (UC de l'hôte, Session active, Mémoires), le monitoring SQL, et le statut global de la base de données (voir **Figure 12**).



a. Orcl



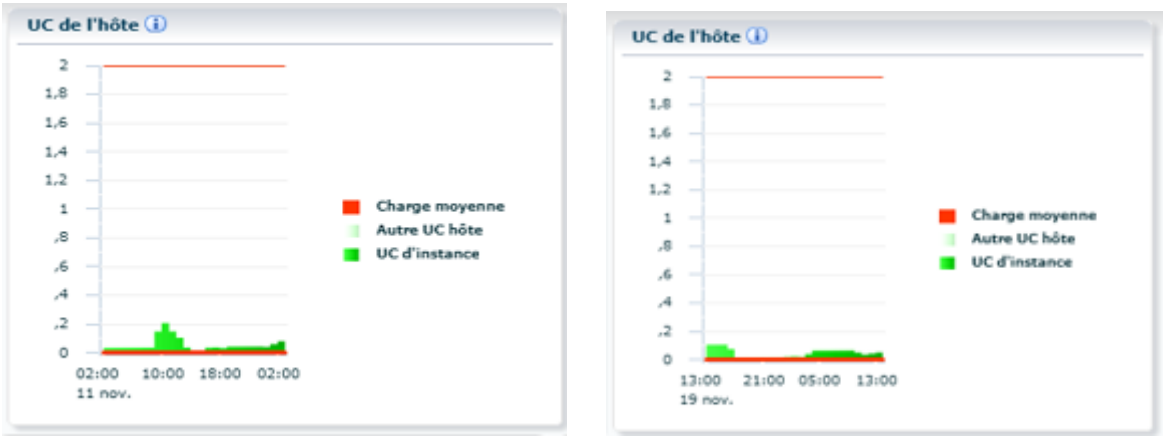
b. Orcl1

FIGURE 12 : CHARGE D'UC SUR ORL ET ORCL1

Analyse et Interprétation du résultat observé sur l'utilisation de l'UC :

- La charge de l'UC au niveau Orcl (**Figure 12.a**) est légèrement plus élevée que celle Orcl1 (**Figure 12.b**). Cela peut s'explique du fait que Orcl utilise plus d'UC comparés à Orcl1. Plus que l'UC a une faible valeur plus qu'il est disponible pour traiter les demandes des instances. D'où l'importance consommation de RAM d'Orcl comparé à Orcl1.

Nous passons aux écrans donnant une vision claire sur la gestion des ressources au niveau des bases de données Orcl et Orcl1 (voir **Figure 13**)



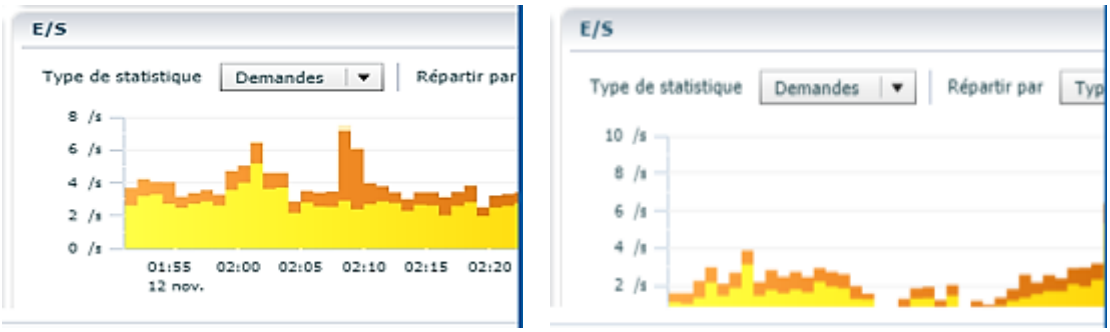
a. Orcl **b. Orcl1**

FIGURE 13 : GESTION RESSOURCES DES BD ORCL ET ORCL1

<u>Analyse et Interprétation du résultat observé sur la gestion des ressources</u> :

- Le graphe UC de l'Hôte montre que la BD Orcl (**Figure 13.a**) a un pourcentage d'UC d'instance utilisée par les processus de premier plan et d'arrière-plan supérieur à celui d'Orcl1 (**Figure 13.b**), ceci explique une légère dégradation des performances observées au niveau d'Orcl. Ce qui justifie que la base de données Orcl consomme plus de ressources avec un Coût (CPU+IO) important comparé à Orcl1.

Enfin, nous terminons par donnée l'analyse et l'interprétation des courbes de gestion de performances des deux de données (Orcl et Orlc1) sur la **Figure 14.**



a. Orcl **b. Orcl1**

FIGURE 14 : E/S DISQUE SUR ORCL ET ORCL1

<u>Analyse et Interprétation du résultat observé sur la gestion des performances</u> :

- En regardant les courbes de gestion de performances des bases de données : Orcl et Orcl1, nous voyons que les E/S d'Orcl (**Figure 14.a**) sont plus importantes que les E/S d'Orcl1 (**Figure 14.b**). Ceci s'explique du fait qu'Orcl effectue plus d'accès disque par seconde comparée à Orcl d'où le temps de réponse d'Orcl1 inférieur à celui d'Orcl.

4.6 - Étude et Comparaison de Requêtes

Dans cette partie nous allons exécuter une même requête écrite de trois manières différentes sur les deux bases de données et utiliser les mêmes indicateurs que précédemment pour mesurer les performances. L'objectif est de voir quel est l'impact de l'écriture des requêtes sur les performances d'un serveur de données.

4.6.1. Exécutions des différentes Requêtes sur les BD

Pour ce faire nous prenons une requête donnant la liste des employés appartenant au département 'SALES' dont l'augmentation est inférieure à 4000. Cette requête est nommée R1 pour la suite. Ces deux requêtes ne nomment R1' et R1''. Ensuite nous avons exécuté ces trois requêtes (R1, R1', R2'') sur Orcl et Orcl1.

```
R1: explain plan for select * from bigemp e, bigdept d, augmentation a
      Where e.deptno = d.deptno and dname = 'SALES'
      And e.empno = a.empno and montant < 4000;
```

```
R1': explain plan for Select * from bigemp where empno in
       (select empno from augmentation where montant < 4000)
       and deptno in (select deptno from bigdept where dname = 'SALES') ;
```

```
R1'': explain plan for select ename, job, sal from bigemp where
        empno in (select empno from augmentation where montant < 4000)
        and deptno in (select deptno from bigdept where dname =  'SALES) ;
```

L'exécution de la requête R1 sur Orcl, donne le plan d'exécution (**Figure 16**) suivant :

-Temps (s) = 08.63s

-RAM = (1698+….+ 1034)=5.80=6Mo de la colonne **Bytes**

- Coût (CPU+IO) = 348

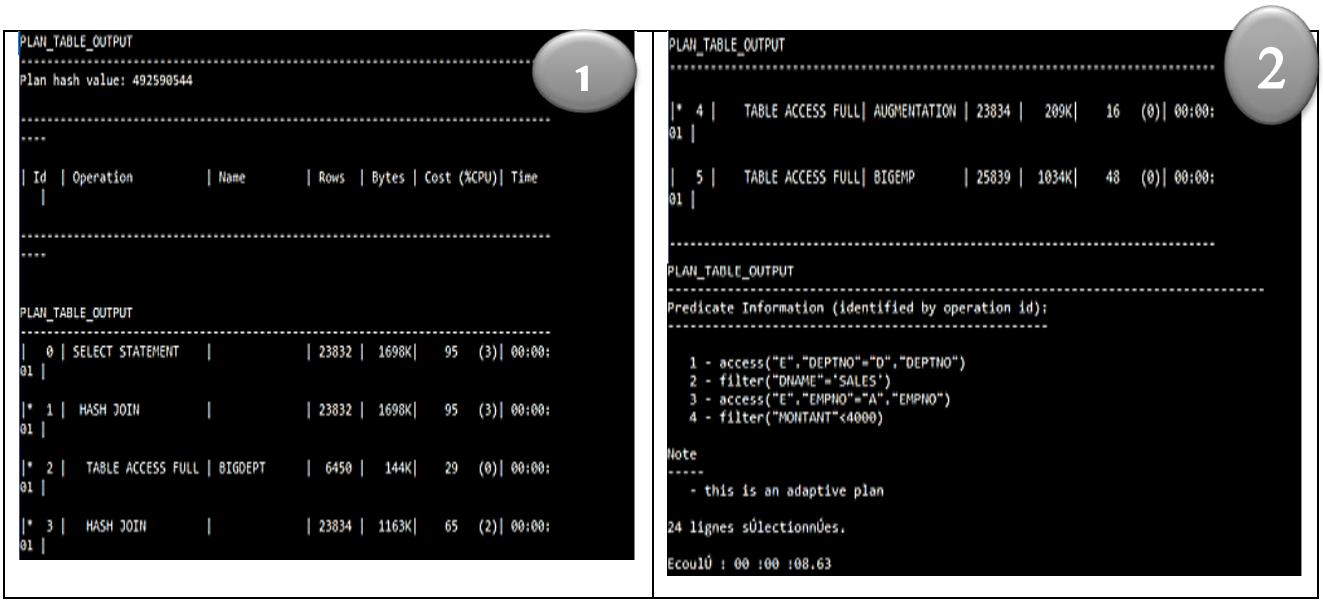


FIGURE 15 : PLAN D'EXECUTION DE R1 SUR ORCL

La requête R1' donne le plan d'exécution (**Figure 17**) suivant sur Orcl :

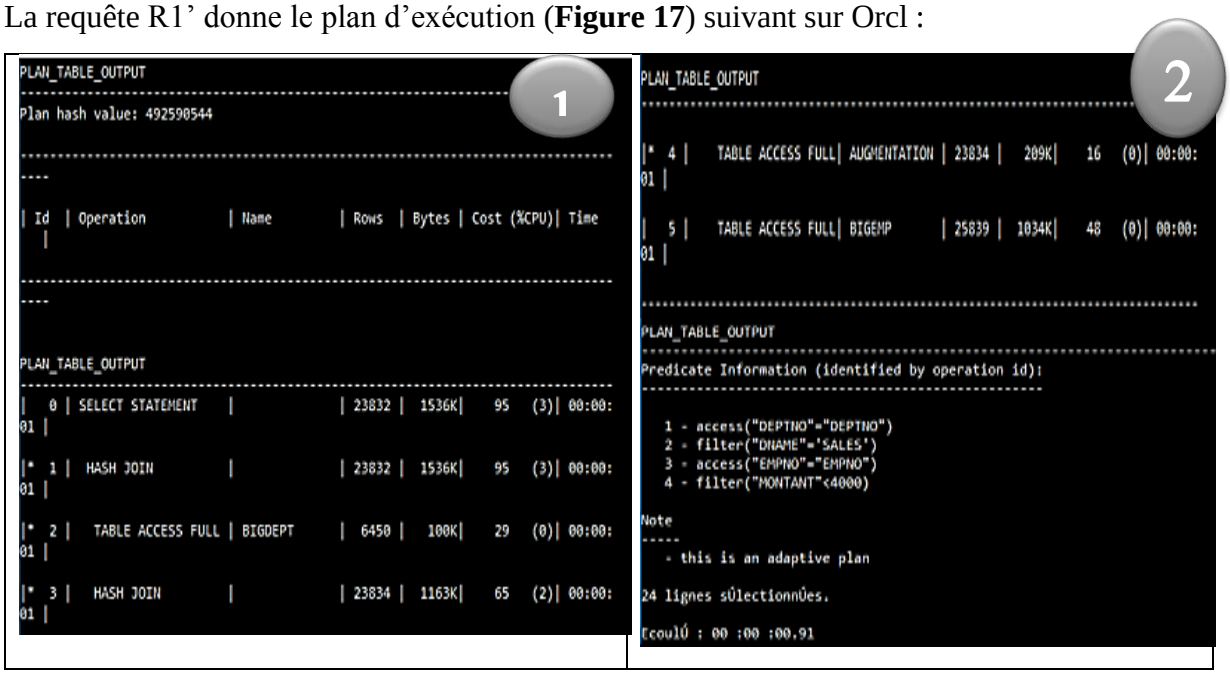


FIGURE 16 : PLAN D'EXECUTION DE R1' SUR ORCL

-Temps=0.91s

-RAM = (1536+…+1034)=5.44=5Mo de la colonne **Bytes**

- Coût (CPU+IO) = 348

Pour terminer sur Orcl, R1'' donne le plan d'exécution (**Figure 18**) suivant :

-Temps=0.89s

-RAM = (1288+…+757)=4,42=4Mo de la colonne **Bytes**

- Coût (CPU+IO) =348

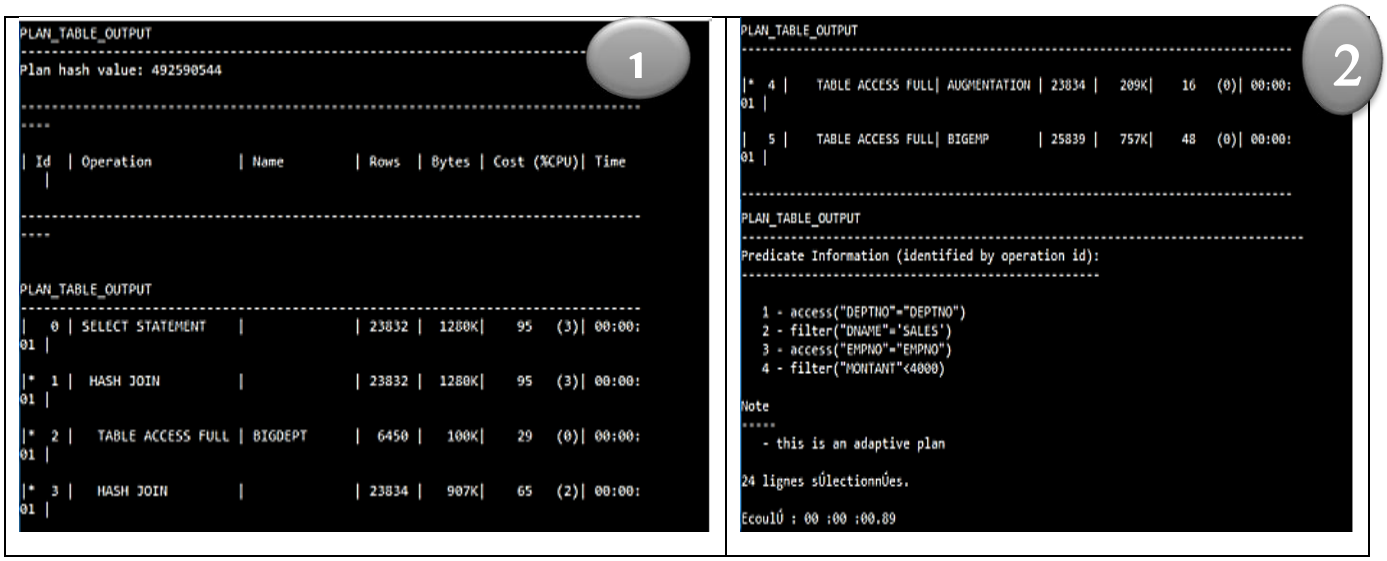


FIGURE 17 : PLAN D'EXECUTION DE R1'' SUR ORCL

Nous allons reprendre les mêmes requêtes que précédemment, sur Orcl1.

R1 donne le plan d'exécution (**Figure 19**) suivant :

-Temps = 00.30s

-RAM = 685bytes=0,000653266=0,000653Mo de la colonne **Bytes**

- Coût (CPU+IO) =80

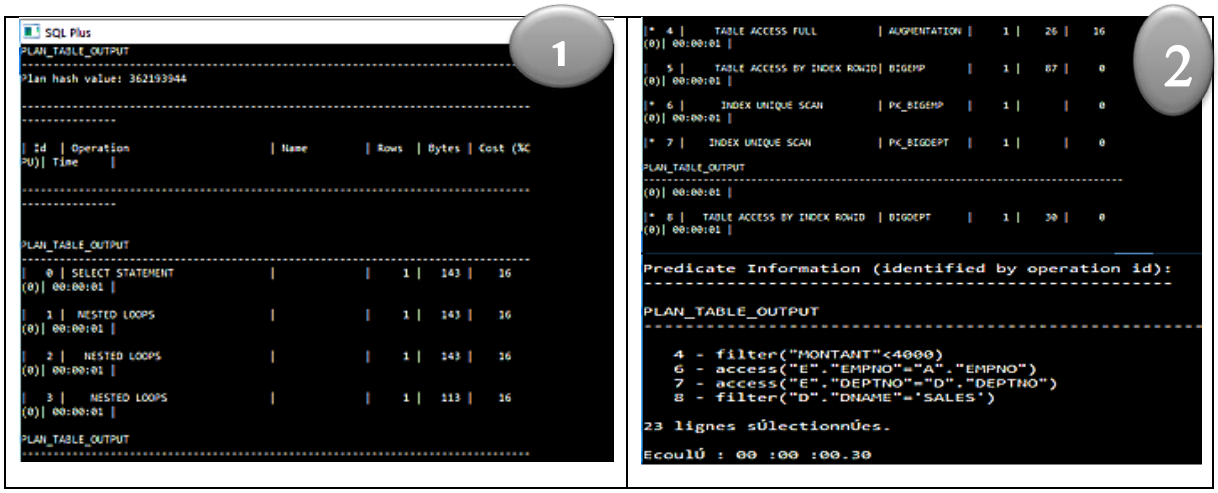


FIGURE 18 : PLAN D'EXECUTION DE LA REQUETE R1 SUR ORCL1

Sur Orcl1, la requête R1' donne le plan d'exécution (**Figure 20**) suivant :

-Temps = 00.28s

-RAM = 653Bytes=0,000622749=0,000622Mo

- Coût (CPU+IO) = 80

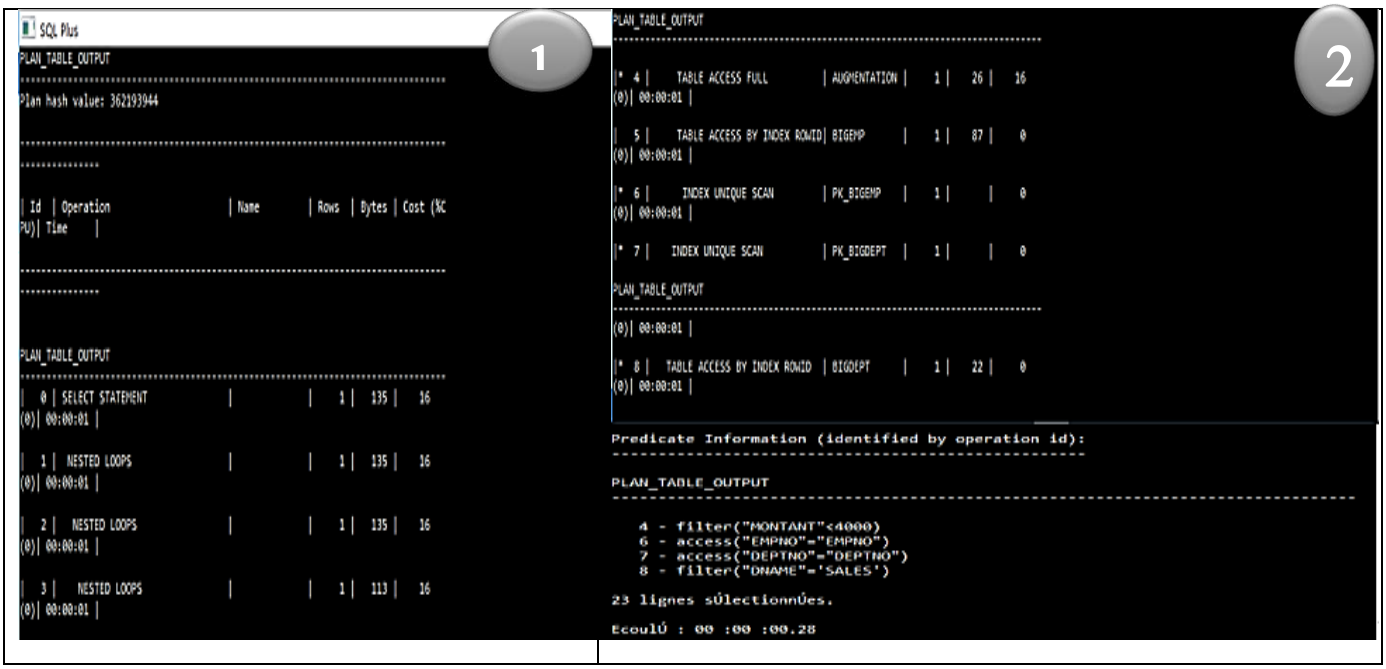


FIGURE 19 : PLAN D'EXECUTION DE LA REQUETE R1' SUR ORCL1

Enfin la requête R1'', donne le plan d'exécution (**Figure 21**) suivant :

-Temps = 00.25s

-RAM = 478Bytes=0,000455856=0,000455Mo

- Coût (CPU+IO) = 80

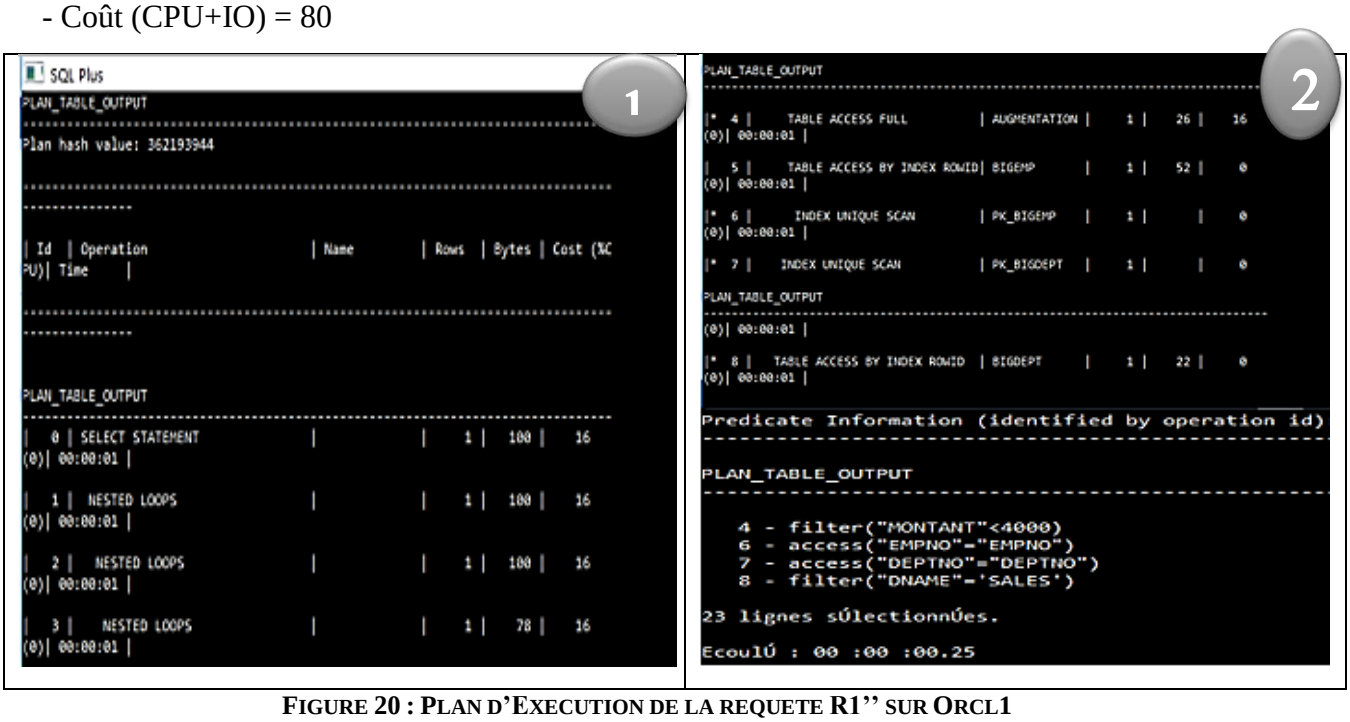


FIGURE 20 : PLAN D'EXECUTION DE LA REQUETE R1'' SUR ORCL1

4.6.2. Résultats obtenus de l'exécution des trois Requêtes

TABLEAU 11 : RESULTATS DES TROIS REQUETES EFFECTUEES SUR ORCL ET ORCL1

Requêtes	Paramètres étudiés \ BD	Orcl	Orcl1
R1	Temps en s	00 :00 :08.63	00 :00 :0.30
	Consommation RAM (Mo)	6	0,000653
	Cout (CPU+IO)	348	80
R1'	Temps (s)	00 :00 :0.91	00 :00 :0.28
	Consommation RAM (Mo)	5	0,000622
	Cout (CPU+IO)	348	80
R1''	Temps (s)	00 :00 :0.89	00 :00 :0.25
	Consommation RAM (Mo)	4	0,000455
	Cout (CPU+IO)	348	80

4.6.3. Représentation graphique (voir **Figure 21**,…, **Figure 23**) ci-dessous

4.6.4. Analyse et Interprétation des résultats

Après observation des trois histogrammes, nous constatons que les mêmes requêtes exécutées sur les deux bases de données (Orcl, Orcl1) donnent des temps de réponse plus faible sur Orcl1. Sur la base de données Orcl1 ont un temps de réponse moins élevé que celles exécutées sur Orcl. Les requêtes exécutées sur Orcl1 consomment moins de mémoire RAM et ont un Coût (CPU+IO) meilleur comparé aux requêtes exécutées sur Orcl. Ceci peut s'expliquer du fait de la spécification du dimensionnement des mémoires, dans Orcl1 nous avons accordé une taille de mémoire importante aux tampons de base de données permettant de diminuer les accès disque qui sont couteux, à la mémoire partagée par rapport de ce que le système les attribués dans Orcl. Au niveau consommation RAM, ce résultat peut s'expliquer du fait de l'utilisation de la jointure de hachage (HASH JOIN) par l'optimiseur au niveau de la base de données Orcl, et la jointure de boucle imbriquée (NESTED LOOP) au niveau d'Orcl1. La jointure HASH JOIN de par son fonctionnement utilise plus de RAM car génère une table dans la RAM qui servira de référence, ce qui entraine une consommation plus élevée de RAM comparée à la jointure imbriquée qui cherche des éléments suivant un sous-ensemble.

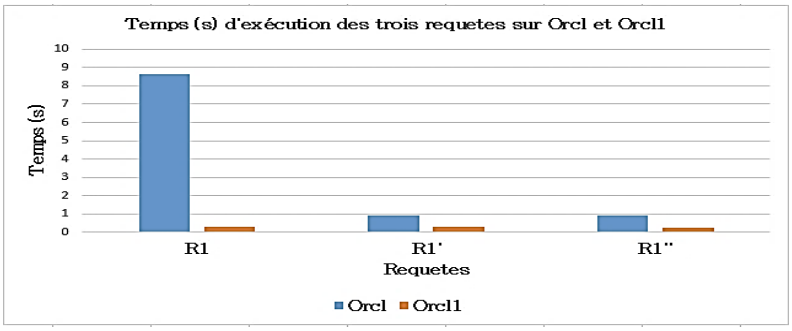


FIGURE 21 : HISTOGRAMME TEMPS MIS PAR LES REQUETES SUR LES BD

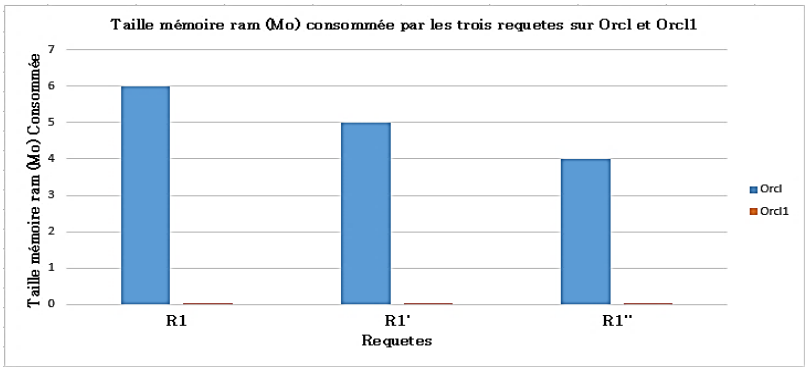


FIGURE 22 : HISTOGRAMME RAM UTILISE PAR LES REQUETES SUR LES BD

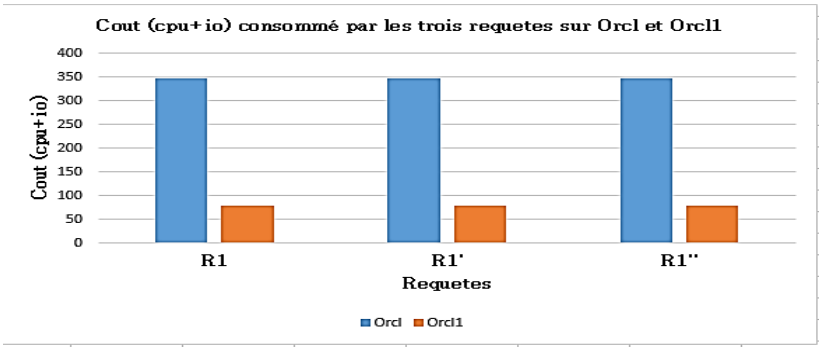


FIGURE 23 : HISTOGRAMME COUT (CPU+IO) DES REQUETES SUR LES BD

4.6.5. Analyse et Interprétation des résultats des trois requêtes sur Orcl1

Avec les résultats obtenus du **Tableau 11** de la section 4.6.2, nous allons donner une analyse et une interprétation des **Figure 24** et **Figure 25.**

Après observation des deux histogrammes ci-dessous, nous constatons que la requête R1'' a le meilleur temps de réponse et consomme moins de RAM que les deux autres requêtes SQL (R1 et R1'), ceci peut s'expliquer du fait de la spécification des attributs de la sélection permettant d'éviter de parcourir toute la table BIGEMP; de plus la réécriture de la requête en modifiant la clause **Where**. Cette requête est plus optimale que les deux autres requêtes. Cette requête R1'' permet de bien montrer l'impact de l'écriture des requêtes sur les performances des serveurs de bases de données. Plus les requêtes sont males conçues plus les performances sont faibles.

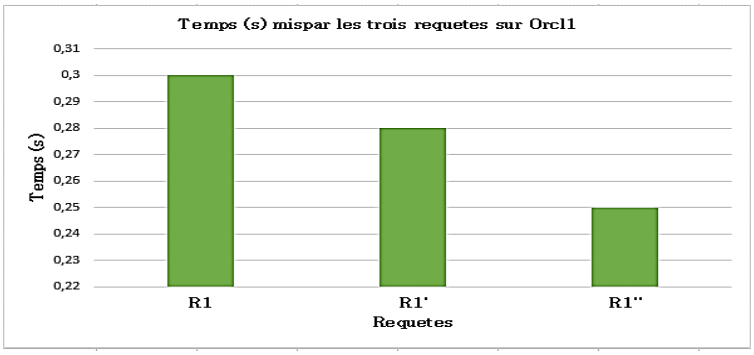


FIGURE 24 : HISTOGRAMME DU TEMPS D'EXECUTIONS DES REQUETES SUR ORCL1

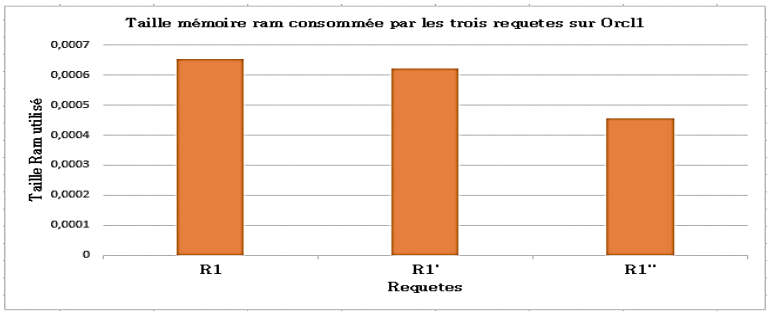


FIGURE 25 : HISTOGRAMME RAM CONSOMMEE PAR LES REQUETE SUR ORCL1

Conclusion

Dans ce chapitre nous avons fait des tests pratiques qui ont fourni différents résultats. L'étude est faite sur deux méthodes d'implémentation de la même base de données sous Oracle 12c Release 2. La première base de données (Orcl) utilise la gestion automatique de la mémoire et l'autre (Orcl1) utilise la gestion manuelle de la mémoire basée nos recommandations de bon dimensionnement. Les outils utilisés pour l'étude sont : les plans d'exécutions, les vues Systèmes et Oracle Entreprise Manager Express 12c. Pour mesurer les performances, nous avons utilisé les indicateurs suivants : Temps d'exécution, Taille de mémoire RAM (taille en Mo) nécessaire et Coût (CPU+IO). Les résultats sont beaucoup plus clairs sur le plan d'exécution des requêtes que sur l'interface web Oracle Entreprise Manager Express 12c.

À la sortie de ce chapitre, nous voyons que la gestion manuelle des mémoires sous Oracle consomme moins de ressources (RAM et CPU), avec un temps de réponse moins important que la gestion automatique des mémoires sous Oracle. Bien que cette méthode présente des résultats plus satisfaisants, elle demande un niveau de compétence élevé de la part des DBA.

Troisième partie : Analyse et Conception

Chapitre 5 : Le logiciel AGPSD

Chapitre 5 : Le logiciel AGPSD

Introduction

Ce chapitre fait l'ébauche d'un logiciel d'aide à l'amélioration des performances d'un serveur de données. Ce logiciel est nommé **A**ide à la **G**estion des **P**erformances d'un **S**erveur de **D**onnées (AGPSD). Ainsi, nous parlons dans la Section 5.1 des Fonctionnalités de AGPSD. La spécification et l'analyse du logiciel sont données à la Section 5.2. Son architecture est décrite dans la Section 5.3.

5.1 - Fonctionnalités

Nous commençons par identifier les acteurs du système avant de donner les différentes fonctionnalités du logiciel AGPSD.

5.1.1. Identification des acteurs du système

Un acteur désigne une entité qui interagit avec le système. Il joue un rôle interne ou externe. Un acteur peut être principal ou secondaire. Pour le système étudié, les acteurs sont recensés dans le tableau ci-dessous.

TABLEAU 12 : IDENTIFICATION DES ACTEURS

Acteurs	Rôles
Administrateur de base de données (DBA)	C'est la personne chargée de la gestion intégrale des bases de données fonctionnant sur un serveur de données
Administrateur Système (AdminSys)	Il est chargé de la gestion des ressources (CPU, RAM) et des utilisateurs sur le serveur de données

5.1.2. Identification des fonctionnalités

Les besoins fonctionnels représentent les actions que le système doit exécuter, il ne devient opérationnel que s'il les satisfait. Dans le **Tableau 13,** nous avons identifié un ensemble de fonctionnalités. La fonctionnalité gérer les ressources englobe la gestion CPU et RAM d'un serveur de données. La fonctionnalité gérer RAM signifie la gestion de la taille de mémoire RAM reçue par le DBA pour le fonctionnement de ses bases de données.

TABLEAU 13 : IDENTIFICATION DES FONCTIONNALITES DU SYSTEME

Fonctionnalité (s) d'un système	Acteur (s)
S'authentifier	DBA, AdminSys
Gérer ressources (attribuer, consulter, modifier, supprimer)	AdminSys

Gérer UsersServeur (ajouter, lister modifier, supprimer)	AdminSys
Gérer RAM (allouer, consulter, modifier)	DBA
Gérer UsersBD (ajouter, modifier, supprimer, lister les utilisateurs)	DBA
Consulter pourcentage (%CPU, %RAM)	DBA
Gérer Configuration (ajouter, modifier, supprimer, afficher)	DBA
Gérer Requêtes (Afficherplan, Modifier)	DBA

UsersServeur signifie les utilisateurs d'un serveur de données et UsersBD représente les utilisateurs d'une base de données.

5.2 - Spécification et Analyse des diagrammes

Dans cette partie, nous allons donner la spécification et l'analyse des différents diagrammes permettant d'implémenter les fonctionnalités décrites dans la section 5.1.2. Nous parlons dans la Section 5.2.1 des diagrammes de cas d'utilisation. Pour chaque cas d'utilisation étudié, nous donnons respectivement sa description, son diagramme de séquence et d'activité correspondant. La Section 5.2.14 présente le diagramme d'activité du système. Enfin le diagramme de composant est présenté dans la Section 5.2.15.

5.2.1. Diagramme de cas d'utilisation

Les diagrammes de cas d'utilisation sont des diagrammes UML utilisés pour donner une vision globale du comportement fonctionnel d'un système. Un cas d'utilisation représente une unité discrète d'interaction entre un utilisateur et un système. Les **Figure 26** et **Figure 27**, représentent respectivement, le diagramme de cas d'utilisation de l'AdminSys, le diagramme de cas d'utilisation du DBA.

Dans la suite, nous donnons la description de chaque cas d'utilisation étudié, suivi de son diagramme d'activité et son diagramme de séquence et enfin nous donnons le diagramme d'activité et le diagramme de composant du logiciel.

5.2.2. Analyse de l'authentification

Nous commençons par décrire le cas d'utilisation « s'authentifier » ensuite le diagramme d'activité de l'authentification et enfin le diagramme de séquence.

5.2.3. Description de cas d'utilisation « s'authentifier »

Le **Tableau 14** permet de décrire le cas d'utilisation « s'authentifier ».

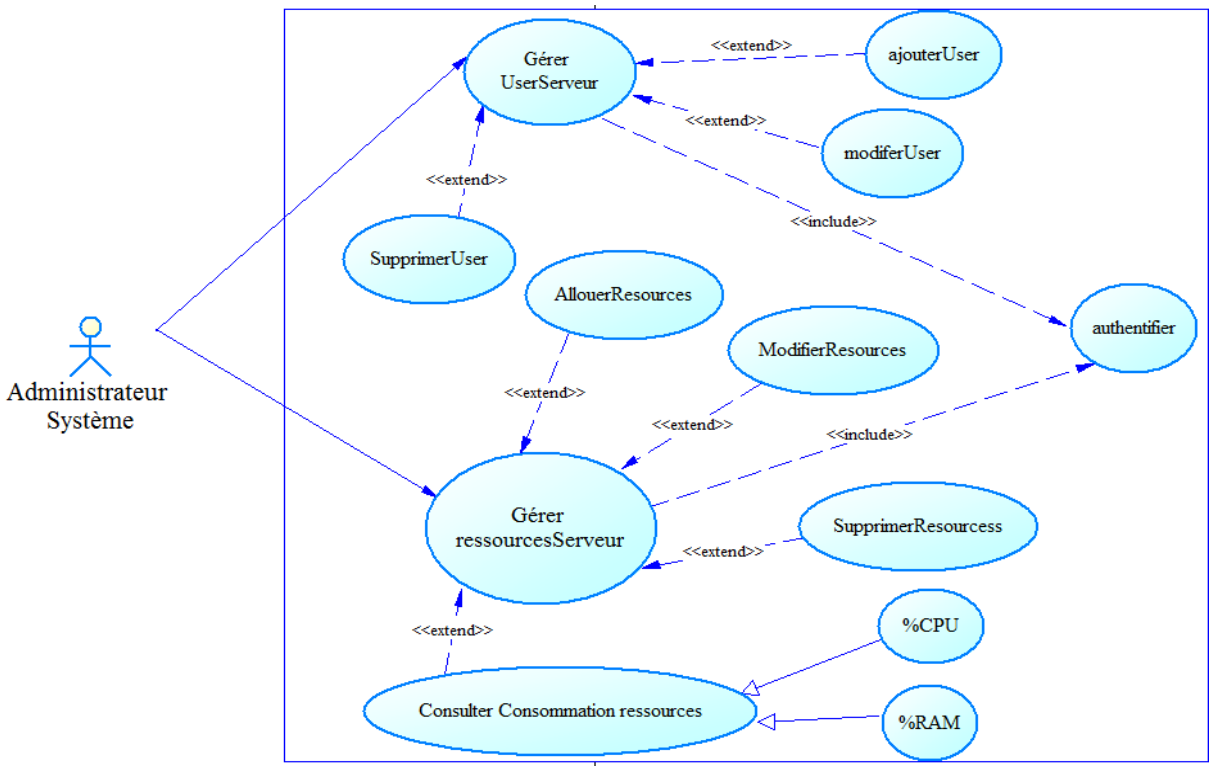


FIGURE 26: DIAGRAMME DE CAS D'UTILISATION DE L'ADMINISTRATEUR SYSTEME

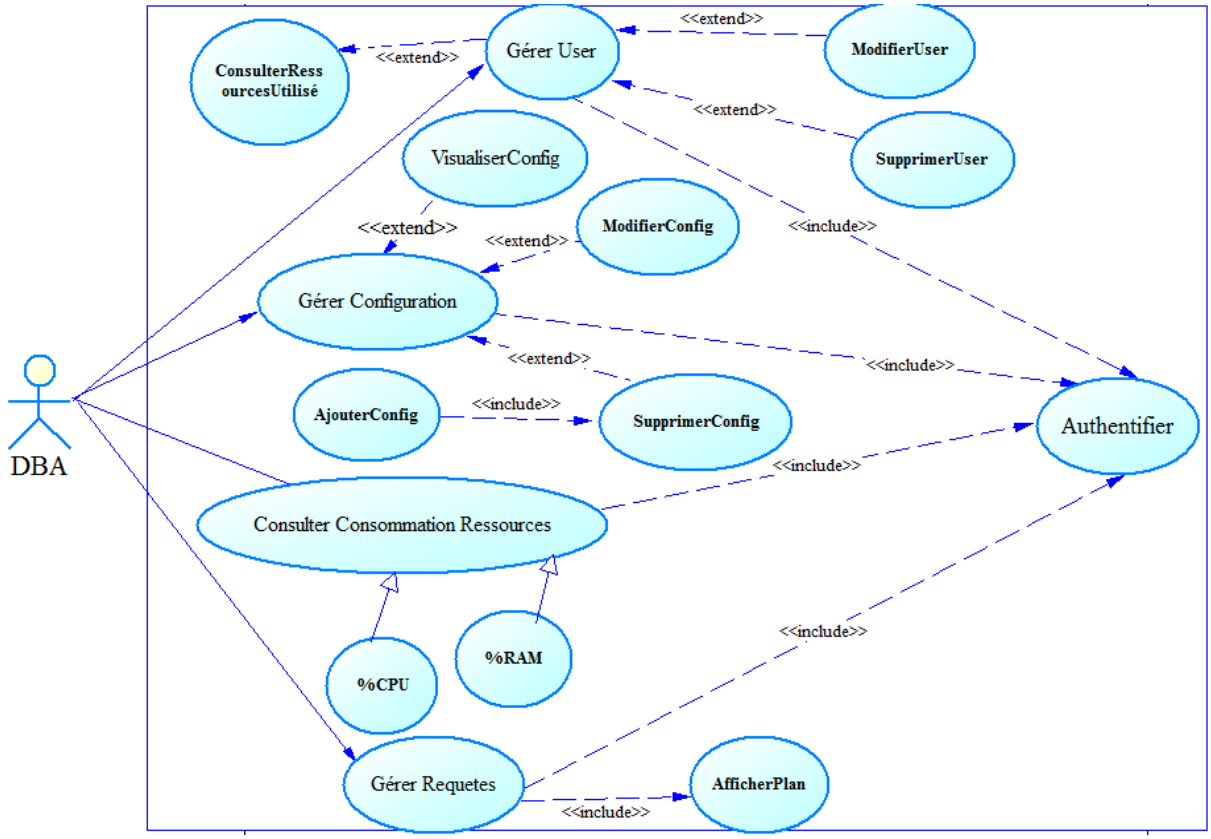


FIGURE 27 : DIAGRAMME DE CAS D'UTILISATION DU DBA

TABLEAU 14 : DESCRIPTION DE CAS D'UTILISATION « S'AUTHENTIFIER »

Description de cas d'utilisation « s'authentifier »

Titre	S'authentifier
Résumé	Permet de vérifier l'accès au système
Acteur (s)	DBA, AdminSys
Pré condition	Avoir un compte d'utilisateur
Scénario nominal	✓ L'utilisateur saisit son identifiant et son mot de passe ✓ Le système vérifie les informations saisies ✓ Le système récupère le profil de l'utilisateur
Post condition	Accéder la page d'accueil avec son profil de l'utilisateur
Exception	Saisie d'un identifiant ou d'un mot de passe incorrecte

5.2.4. Les activités de l'authentification

Le cas d'utilisation « s'authentifier » est composé de plusieurs activités. Nous commençons par **saisir un identifiant et un mot de passe**. Une fois que cette activité est exécutée, le système procède à une vérification. Si les données saisies sont incorrectes, **un message d'erreur s'affiche** et l'activité «**saisir un identifiant et un mot de passe»** doit être reprise. Par ailleurs, si les champs sont bien renseignés, elle débouche sur **l'accès à la page d'accueil**. L'ensemble de ces enchaînements sont décrits dans le diagramme d'activité de la **Figure 28**.

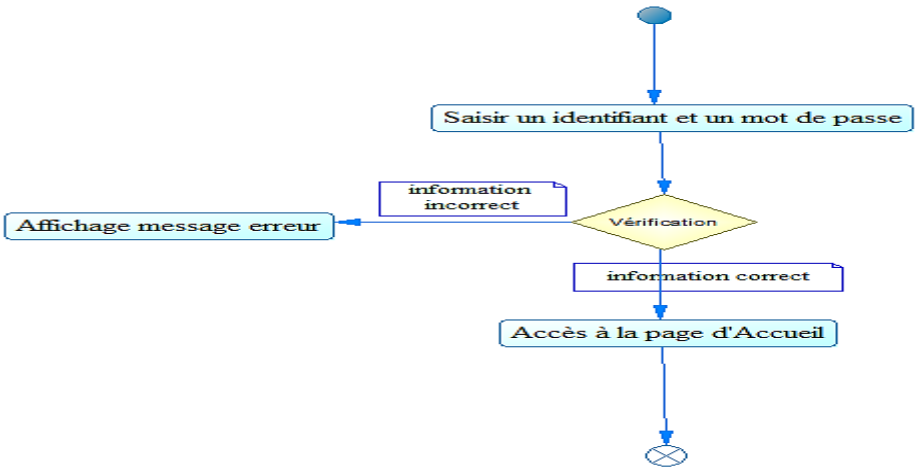


FIGURE 28 : DIAGRAMME D'ACTIVITE DU CAS D'UTILISATION «S'AUTHENTIFIER»

5.2.5. Diagramme de séquence de l'authentification

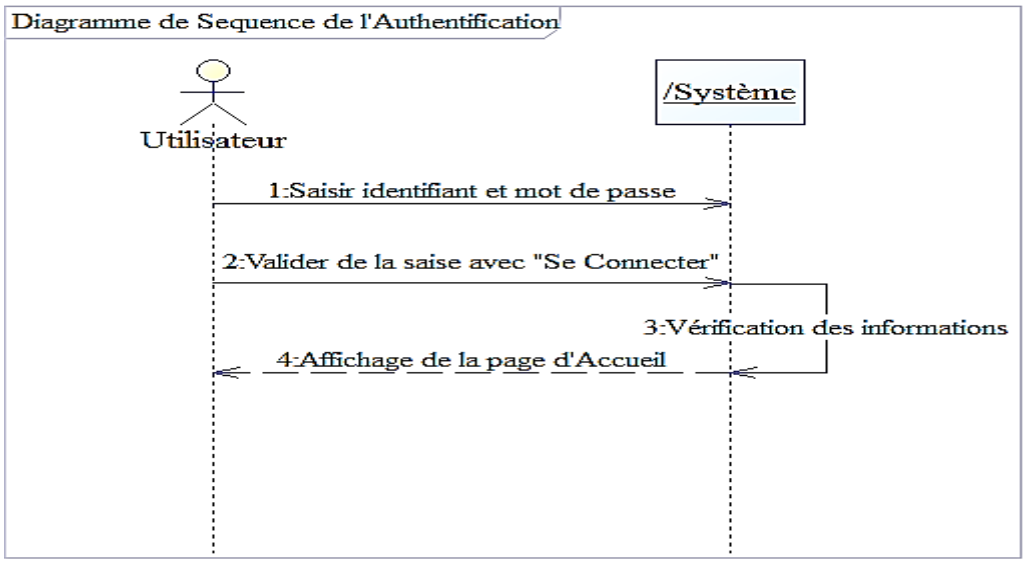


FIGURE 29 : DIAGRAMME DE SEQUENCE DU CAS D'UTILISATION «S'AUTHENTIFIER»

Pour le diagramme d'activité, plusieurs alternatives sont notées. Ce qui n'est pas le cas pour le diagramme de séquence qui schématise un seul scénario. Dans notre cas nous représentons le scénario nominal. L'utilisateur saisit son identifiant et son mot de passe. Le système procède à une vérification. Une fois que cela est fait, le système affiche la page d'accueil. La **Figure 29** illustre le diagramme de séquence du cas d'utilisation de l'authentification.

5.2.6. Analyse de la gestion des ressources

Le cas d'utilisation «Allouer ressources» équivaut à allouer le CPU et la taille mémoire RAM totale d'un serveur de données. Dans cette partie nous donnons la description du cas d'utilisation «Allouer ressources ». En plus de la description, nous détaillons les diagrammes d'activité et de séquence du cas « Allouer ressources ».

5.2.7. Description de cas d'utilisation « allouer ressources »

TABLEAU 15 : DESCRIPTION DE CAS D'UTILISATION «ALLOUER RESSOURCES »

Description de cas d'utilisation «allouer ressources »	
Titre	Allouer ressources
Résumé	Permet d'allouer des ressources (CPU, RAM) à un utilisateur d'un serveur de données
Acteur	AdminSys
Pré condition	Authentification
Scénario nominal	✓ L'utilisateur clic sur le bouton attribuer ressources pour attribuer des ressources à un utilisateur donné, ✓ Le système affiche un formulaire de confirmation, ✓ Le système actualise la page après validation,

	✓ Il archive les données de la base de données.
Post condition	Message de confirmation de l'attribution de ressources

5.2.8. Les activités de la gestion des ressources

Pour allouer des ressources, l'AdminSys accède à la page d'accueil et sélectionne l'onglet «Gérer ressources» et puis choisit l'option « allouer ressources » sur cette option un formulaire s'ouvre. Il recherche l'utilisateur à attribuer des ressources sur une liste déroulante. Ce dernier doit impérativement exister dans la base de données. Si l'utilisateur apparait, l'AdminSys renseigne les valeurs de CPU et de RAM à attribuer à cet utilisateur. Après renseignement des valeurs requises, il clique sur le bouton «Attribuer», une boîte de dialogue s'affiche pour confirmer ou annuler l'attribution. Si on clique sur « Oui » les ressources sont attribuées à l'utilisateur sinon rien n'est attribué. Si l'utilisateur recherché n'est pas trouvé, le processus créer un utilisateur peut être déclenché. (Voir **Figure 30**)

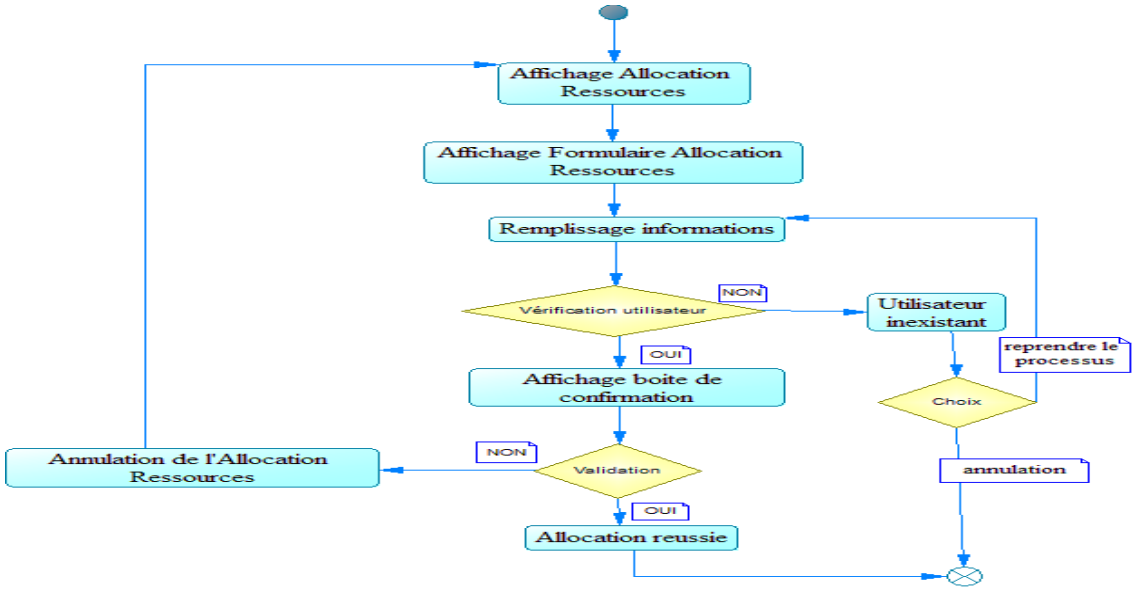


FIGURE 30 : DIAGRAMME D'ACTIVITE DU CAS «ALLOUER RESSOURCES»

5.2.9. Diagramme de séquence du cas d'utilisation « allouer ressources»

Après authentification, l'utilisateur décide de gérer les ressources. Une fois l'option Ressources choisie, le système envoie un formulaire de confirmation. Après avoir confirmé par « Oui » l'attribution de ressources à cet utilisateur est effective par le système. (Voir **Figure 31)**

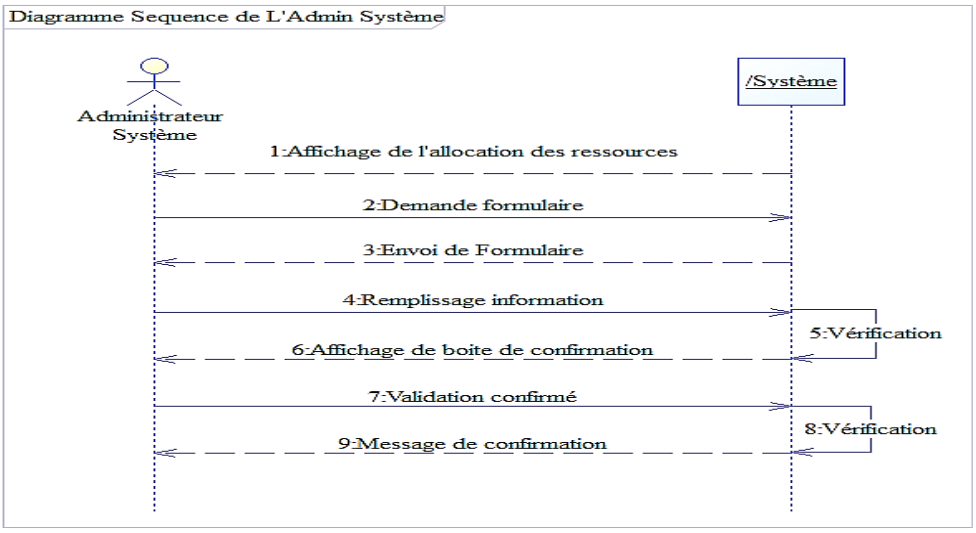


FIGURE 31 : DIAGRAMME DE SEQUENCE DU CAS D'UTILISATION «ALLOUER RESSOURCES»

5.2.10. Analyse du cas d'utilisation «ModifierConfig»

Le cas d'utilisation «ModifierConfig» consiste à modifier le dimensionnement des mémoires d'un serveur de données. La description du cas d'utilisation «ModifierConfig» est donnée en 5.2.11. Les diagrammes d'activités et de séquences sont donnés respectivement en 5.2.12 et 5.2.13.

5.2.11. Description du cas d'utilisation «ModifierConfig»

TABLEAU 16 : DESCRIPTION DE CAS D'UTILISATION « MODIFIERCONFIG»

Description de cas d'utilisation «Modifierconfig »	
Titre	Modifier une configuration d'une BD
Résumé	Permet de modifier le dimensionnement des mémoires d'une BD
Acteur (s)	DBA
Pré condition	Authentification
Scénario nominal	✓ L'utilisateur visualise la configuration actuelle ✓ L'utilisateur clique sur le bouton modifié qui se trouve sur la liste des configurations afin de les ✓ Il remplit les champs actifs à modifier puis valide ✓ Le système modifie puis enregistre dans la base de données
Post condition	Message de confirmation de la modification
Exception	Annulation de la modification

5.2.12. Les activités de la modification d'une configuration

Pour effectuer la modification de la configuration d'une BD, le DBA accède à la page d'accueil et choisit l'option «Gérer Configuration», ensuite l'option «ModifierConfig» et une

liste de configuration apparait. La configuration doit exister auparavant. Tous les champs à modifier sont actifs. Le DBA effectue la modification souhaitée puis valide (voir **Figure 32**).

5.2.13. Le diagramme de séquence du cas « modifierconfig »

Une fois la page d'accueil, l'utilisateur choisit successivement les options «Gérer configuration » et «ModifierConfig ». Sur la même ligne, l'utilisateur décide de modifier la configuration d'une BD. Les champs concernés seront activés. Après avoir rempli les champs, il valide la modification. (Voir **Figure 33**)

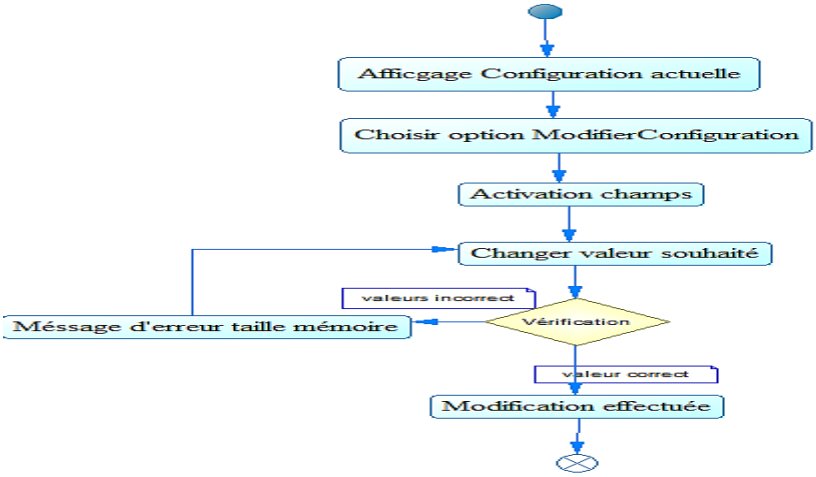


FIGURE 32 : DIAGRAMME D'ACTIVITE DU CAS « MODIFIERCONFIG »

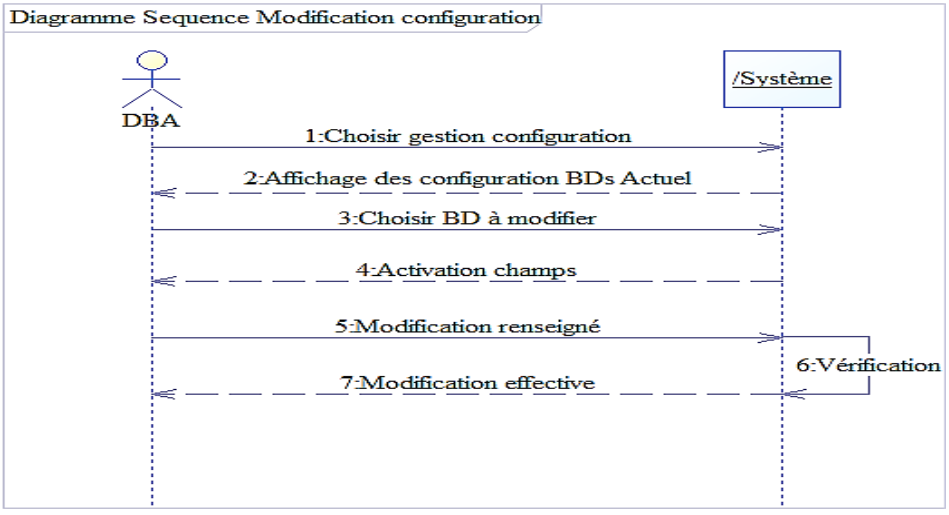


FIGURE 33 : DIAGRAMME SEQUENCE DU CAS «MODIFIERCONFIG»

Les informations échangées sont traitées par les modules suivant les activités représentées sur le diagramme d'activités décrit dans la Section 5.2.14.

5.2.14. Diagramme d'activité de l'application

Ce diagramme permet d'implémenter les cas d'utilisations étudiés en prenant en compte les séquences (voir **Figure 34**). L'implémentation des fonctionnalités est effectuée par deux modules. Ces modules contiennent des sous-modules dont les relations sont représentées sur la figure de la Section 5.2.15.

5.2.15. Diagramme de composant de l'application

Le diagramme de composant (voir **Figure 35**) donne les différents modules du logiciel avec leurs sous-modules. Il montre également les relations entre les sous-modules d'un même module, mais aussi celles entre deux modules différents.

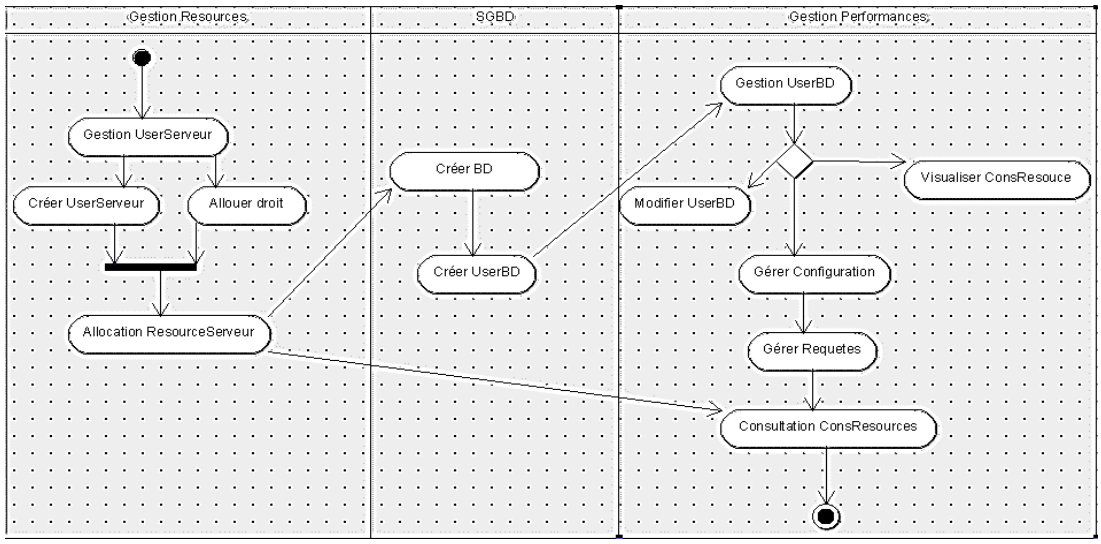


FIGURE 34 : DIAGRAMME D'ACTIVITE DU SYSTEME

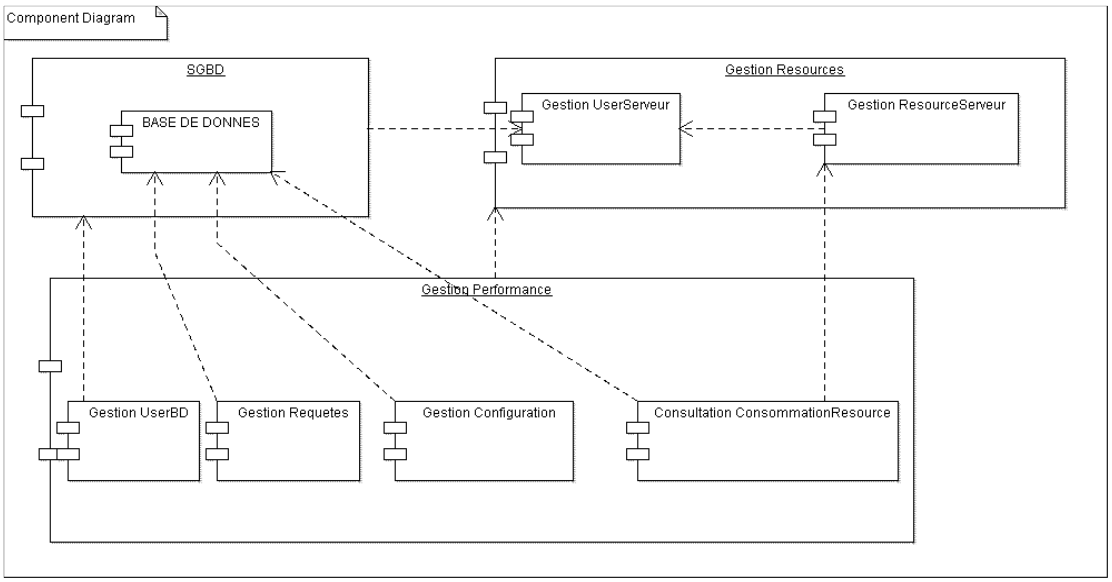


FIGURE 35 : DIAGRAMME DE COMPOSANT

5.3 - Architecture

Les fonctionnalités identifiées dans le **Tableau 13** sont implémentées suivant l'architecture donnée sur la **Figure 36** de la section 5.3.1. Ces fonctionnalités sont exécutées par deux principaux modules décrits dans la section 5.3.2.

5.3.1. Succession d'exécution des fonctionnalités

Pour passer de la gestion des ressources à la gestion des performances, l'utilisation des fonctionnalités qu'offre le logiciel se fait dans un ordre précis. La **Figure 36** donne l'architecture globale du logiciel AGPSD avec les différentes fonctionnalités et leur ordre d'exécution. Pour parvenir à l'implémentation des fonctionnalités, AGPSD est subdivisé en deux modules décrits dans la section suivante.

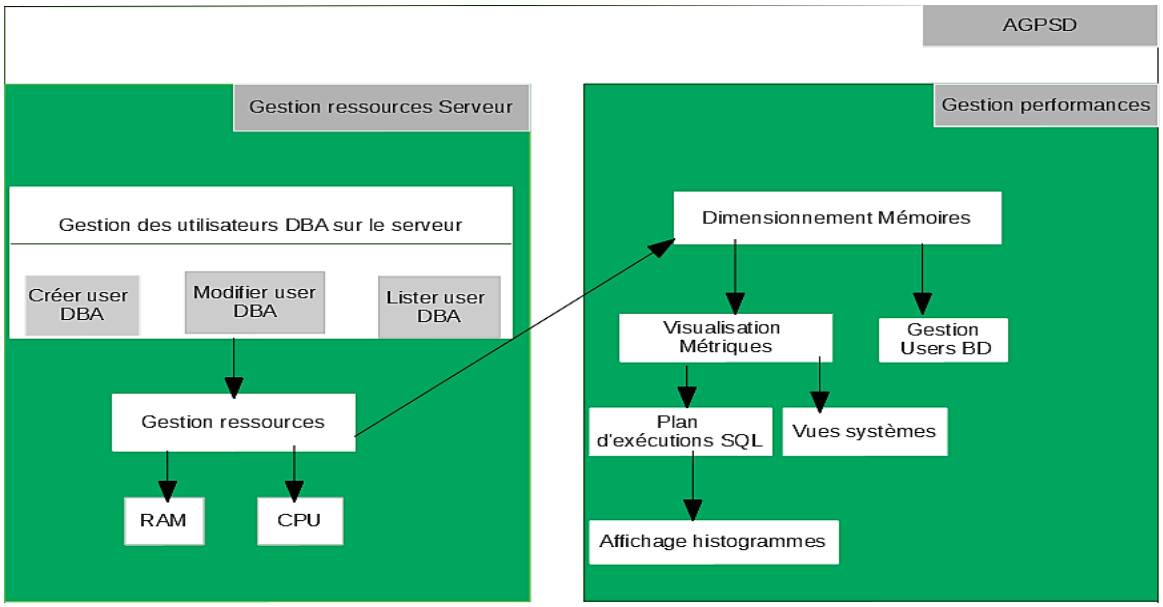


FIGURE 36 : ARCHITECTURE DU LOGICIEL AGPSD

5.3.2. Les modules du logiciel

Vu les fonctionnalités identifiées dans la section 5.1.2, nous notons l'existence de deux modules : **gestion des ressources**, **gestion des performances**. Parlons dans la Section 5.3.2.1 de la description de la gestion des ressources, dans la Section 5.3.2.2 de la gestion des performances.

5.3.2.1. Le module gestion des ressources

Ce module est le premier à être exécuté dans la gestion des performances d'un serveur de données. Il gère la création des utilisateurs d'un serveur de données en implémentant les fonctionnalités (ajouter, modifier, lister, supprimer). Il gère également les ressources d'un

serveur en implémentant les fonctionnalités (allouer ressources, modifier allocation, consulter consommation ressources (%CPU et %Ram)).

5.3.2.2. Le module gestion des performances

Ce module est d'une importance capitale dans l'amélioration des performances d'un serveur de données. Il implémente les fonctionnalités gestion des ressources, gestion configuration, consultation consommation ressources, gestion des requêtes. Les fonctions permettant de mener à bien ces fonctionnalités sont :

- Modification des utilisateurs de BD;
- Suppression des utilisateurs d'une BD;
- Visualisation des ressources consommées par un utilisateur;
- Visualisation de la configuration ou dimensionnement d'une BD;
- Modification de la configuration ou le dimensionnement d'une BD;
- Suppression la configuration d'une BD;
- Ajout la configuration d'une BD;
- Consultation (%RAM, % CPU);
- Proposition de plan d'exécution.

Conclusion

Dans ce chapitre, nous avons présenté les fonctionnalités, la spécification, l'analyse et l'architecture du logiciel AGPSD. Nous avons étudié différents diagrammes UML permettant sa réalisation. Ainsi nous avons proposé deux diagrammes de cas d'utilisation. Ensuite nous avons donné la description de quelques cas d'utilisation, tout en donnant pour chaque cas d'utilisation le diagramme d'activité et de séquence correspondant. Puis un diagramme d'activité donnant le déroulement de l'exécution des cas d'utilisation, en prenant en compte les informations échangées, est représenté. Pour terminer, le chapitre, un diagramme de composants, permettant de voir les interactions entre les modules du logiciel, entre les sous-modules d'un même module et entre les sous-modules de différents modules est, donné.

Conclusion Générale et Perspectives

Dans ce mémoire nous avons présenté l'optimisation des performances d'un serveur de données et l'ébauche d'une application pour son automatisation. Ainsi l'optimisation des bases de données est une tâche cruciale dans l'administration d'un serveur de données. La gestion des performances d'un serveur de données fait l'objet de plusieurs axes de recherches.

Notre objectif général est de répondre à la question comment obtenir un bon dimensionnement des mémoires pour de meilleures performances ? Pour cela nous avons comparé les deux méthodes de gestion de mémoire sous Oracle. Ainsi nous avons implémenté la même base de données DRH de deux manières : Orcl et Orcl1. Orcl utilise la gestion automatique de mémoire alors qu'Orcl1 utilise la gestion manuelle basée sur les recommandations proposées au chapitre 3. Les indicateurs choisis pour mesurer les performances des bases de données (Orcl et Orcl1) sont : le Temps d'exécution, la Consommation de taille de mémoire RAM et le Coût (CPU+IO). Les outils et méthodes utilisés pour mesurer ces indicateurs sont : plan d'exécution, vues systèmes, oracle entreprise manager. L'implémentation des bases de données est faite sous Oracle Entreprise 12cR2 qui est un des SGBD sur le marché. Pour l'automatisation de la gestion des performances d'un serveur de données, nous avons fait la conception d'une application informatique. Pour cette application, nous avons donné : les fonctionnalités, une architecture et des diagrammes UML pour son implémentation.

D'après l'étude expérimentale, le bon dimensionnement de mémoire en utilisant la gestion manuelle est d'une grande importance sur la gestion des performances des serveurs de données. L'exécution d'une même requête sur les deux implémentations montre une différence très significative en temps de réponse en raison des différentes reformulations de la même requête.

Nous pouvons dire que la gestion manuelle de la mémoire donne plus de performances en faisant un bon dimensionnement des mémoires et en ayant une bonne maitrise des outils d'administration. Bien que cette méthode de gestion de mémoires présente des résultats satisfaisants, elle demande un niveau de compétence élevé de la part des DBA.

Le travail effectué dans ce mémoire a apporté d'autres perspectives et appelle à relever d'autres défis. Ces perspectives sont une suite logique du travail effectué dans ce mémoire alors que d'autres sont une amélioration des méthodes de gestion des mémoires. Ainsi nous souhaitons étendre notre étude comme suite :

1. Développement de l'application pour l'automatisation de la gestion des performances des serveurs de données.

Ainsi, la suite envisagée est de :

- Compléter l'analyse et la conception de l'application. Pour cela, nous donnerons tous les modules et sous-modules possibles, une architecture complète, une conception poussée et toutes les fonctionnalités en rapport avec la gestion d'un serveur de données.
- Développer entièrement l'application pour la gestion des performances des serveurs de données. Ainsi, nous implémenterons toutes les interfaces nécessaires.
- Étendre l'application pour pouvoir gérer : les utilisateurs suivant leurs privilèges et activités, les requêtes suivant leurs priorités et leurs fréquences.

Bibliographie

[1] B.F. Zahra - L. Khadidja : « Gestion commerciale répartie : cas de l'Algérie Télécom » Mémoire de Master en Informatique Université Abou Bakr Belkaid-Tlemcen, Option Système d'Informatique et de Connaissances (S.I.C), Soutenu le 01 Juillet 2013

[2] S. Diagne : « Modélisation sémantique conceptuelle pour l'ingénierie de performances comportementales de produits complexes » Thèse de doctorat INSA de Strasbourg, Spécialité Génie Informatique-Génie Mécanique, Soutenu le 7 Juillet 2015.

[3] I. Diop : « Base de données avancées » Cours Master1 génie logiciel 2014/2015 Université Assane Seck de Ziguinchor.

[4] S. Diagne : « Equilibrage de charge dans les systèmes distribués, modèle de performance dans le cas des systèmes transactionnels » Mémoire de DEA d'Informatique Université Cheikh Anta Diop de Dakar, Option Performance des réseaux informatiques, Soutenu le 25 Aout 2007

[5] B. Hanane : « Suivi et gestion répartie des clients d'une banque : cas de la banque BADR » Mémoire de Master en Informatique Université Abou Bakr Belkaid-Tlemcen, Option Système d'Informatique et de Connaissances (S.I.C), Soutenu le 19 Septembre 2013

[6] F. Zohra - M. Rabiea : « Optimisation des bases de données mises en œuvre sous oracle » Mémoire Master Informatique Université Abou Bakr Belkaid–Tlemcen Faculté des Sciences, Système d'Information et de Connaissances (S.I.C), Soutenu le 22 Juin 2015

[7] E.M. Khadidja - M. Meriem : « Étude de sécurité en base de données avec une application pour le contrôle d'accès » Mémoire de Master en Informatique Université Abou Bakr Belkaid-Tlemcen, Option Système d'Informatique et de Connaissances (S.I.C), Soutenu le 29 Septembre 2011

[8] Fadace : « Quel SGBD choisir ?» article publié le Publié le 1er mars 2003 - Mis à jour le 8 octobre 2017 http://fadace.developpez.com/sgbdcmp/#LI

[9] C. Chauhan : « [https://severalnines.com/blog/become-postgresql-dba-understanding-architecture] » article publié 02 Octobre 2017 sur le blog become-postgresql-dba

[10] F. Gaud : « Étude et amélioration de la performance des serveurs de données pour les architectures multi-cœurs » Thèse de doctorat Université de Grenoble, Spécialité Informatique, Soutenu le 02 Décembre 2010.

[11] M. Diallo - S. Diagne : « Définition de l'optimisation », donné le 31 Mars 2018

[12] L. Navarro : « Optimisation base de données » Livre, Pearson Education France, ISBN: 978-2-7440-4156-3

[13] J. Rouhaud : « Performances PostgreSQL au niveau du système» Notes de Cours

[14] Y. Lembachar : « Bases de données » Support de cours Ecole Marocaine des Sciences de l'Ingénieur

[15] G. Mopolo - Moké : « optimisation de requêtes sql sous oracle » Cours MS BDP CERAM SOPHIA ANTIPOLIS 2005 / 2006

[16] A. Battais : «Optimisation PostgreSQL» Support de Cours à Géosciences Rennes-CNRS

[17] T. Kyte (Expert Architecte Oracle): « Tuning PGA : Partie2 » article publié le 16 mars 2016 http://oracleinaction.com/tune-pga-ii/ :

[18] T. Khaled : « Fonctionnement et Tuning de la SGA sous Oracle » Cours Base de données avancées

[19] B. Vesan : «Oracle et SQL Server : La Fragmentation» Article publié, le 18 juin 2013 sur http://blog.capdata.fr/index.php/oracle-et-sql-server-la-fragmentation/

[20] A. Pires : « Top 8 init.ora Paramètres affectant les performances » https://xanpires.wordpress.com/2012/05/20/top-8-init-ora-parameters-affecting-performance/, Article Publié le 20 Mai 2012

[21] D. Burleson : «Suivre les coûts d'UC et d'E/S avec Oracle» Article publié, le 02 Septembre 2015 sur http://www.dba-oracle.com/art_builder_cpu_io.htm

[22] L. Bellatreche : « Bases de données réparties » Support de Cours Master de,(Ensma)

[23] M. Elazab : « Dimensionnement du pool partagé », publié jeudi 5 juillet 2012, https://mohamedelazab.blogspot.sn/2012/07/sizing-shared-pool.html

[24] F. Pachot : « 12cR2 DBCA, gestion automatique de la mémoire et – databaseType », publié le 03Avril 2017 https://blog.dbi-services.com/author/franck-pachot/page/8/

[25]. D. Nougier : « Système de Gestion de Base de données : Administration Avancé Oracle Database » Support de Cours de Daphné Nougier.

[26]. https://www.dbacentral.com/2017/03/oracle-12c-release-2-database-creation.html

[27] J. Zahir : «Contributions à l'optimisation des requêtes sql par des techniques de Data-Mining : recommandation des plans d'exécution, prédiction du temps d'exécution et estimation de sélectivité » Strasbourg, Spécialité Génie Informatique-Génie Mécanique, Soutenu le 7 Juillet 2015.

Webographie

[W1] https://www.pairform.fr/doc/10/20/52/web/co/4_1__SQL3__c_est_quoi.html

[W2] http://www.commentcamarche.net

[W3] https://fr.wikipedia.org/wiki/Oracle_Database

[W4].http://www.oracle.com/webfolder/technetwork/tutorials/obe/db/12c/r1/poster/OUTPUT_poster/pdf/Multitenant%20Architecture.pdf

[W5]. https://docs.postgresql.fr/10/tutorial-advanced.html

[W6]. www.db-engine.com

[w7]. http://www-igm.univ-mlv.fr/~dr/XPOSE2005/entrepot/sgbd.html

[w8]. http://www.sqlpac.com/referentiel/docs/oracle-techniques-optimisation-sga.html

[w9]. https://docs.oracle.com/database/121/TGDBA/tune_pga.htm#TGDBA363

[w10]. https://docs.oracle.com/database/122/

[w11].https://docs.oracle.com/database/121/REFRN/GUID-3BE0F434-A392-42C9-BD77-FD2A7B72D6DB.htm#REFRN30115

[w12].https://docs.oracle.com/cloud/latest/db121/TGDBA/tune_shared_pool.htm#TGDBA578

[w13]. http://sgstocks.tripod.com/oracle-004.htm

[w14]. http://www.altidev.com/livres.php.

[w15]. http://www.oracle.com/technetwork/database/enterprise-edition/downloads/oracle12c-windows-3633015.html

[w16]. https://docs.oracle.com/en/database/oracle/oracle-database/12.2/ntdbi/oracle-database-minimum-hardware-requirements.html#GUID-BF7AD499-451F-4EAE-803F-219F0628A879

[w17].http://www.sqlpac.com/referentiel/docs/oracle-optimisation-shared-pool-sga.htm#.Wj_b_vCgK00

[w18]. http://www.remote-dba.net/t_rac_redo_log_buffers.htm

[w19].http://oraclehariprasathdba.blogspot.sn/2013/02/how-does-one-tune-redo-log-buffer.html

Annexes

R1: **Select round ((1 - (phys.value / (db. value + cons.value))) * 100, 2) "Cache Hit Ratio" From v$sysstat phys, v$sysstat db, v$sysstat cons Where phys.name = 'physical reads' And db.name = 'db block gets' And cons.name = 'consistent gets';** Elle Donne le pourcentage du hit cache ratio du tampon de base de données

R2: **Select ((Sum (pins-reloads)) / Sum (pins)) * 100** "Library Cache Hit Ratio"

From v$librarycache; Cettte requête donne le Hit ratio (taux d'accès) de la bibliothèque de données (Library Cache).

R3: **Select (Sum (getmisses)) / Sum (gets) *100** "row cache"

From v$rowcache; donne le taux (hit ratio) d'accès au dictionnaire de données (Dictionary Cache).

R4 : **Select name, value From v$sysstat Where name in ('db block gets from cache', 'consistent gets from cache', 'physical reads cache') ;**

R5: **Select name, value from v$ sysstat Where name in ('redo buffer allocation retries', 'redo entries').** Cette requête donne le nombre d'entrées et le nombre d'attente dans le redo log buffer.

Les formules de Conversions

1 octet = 8 bits (à ne pas confondre avec byte (Anglais)= octet (Français).)

1 kilooctet (ko) = 1 024 octets

1 mégaoctet (Mo) = 1 024 ko = 1 048 576 octets

1 gigaoctet (Go) = 1 024 Mo = 1 073 741 824 octets

1 téraoctet (To) = 1 024 Go = 1 099 511 627 776 octets

1 pétaoctet (Po) = 1 024 To = 1 125 899 906 842 624 octets

```
CREATE TABLE AUGMENTATION( EMPNO NUMBER not null, MONTANT NUMBER(7,2));
Alter table AUGMENTATION ADD constraint pk_Augmentation primary key(EMPNO);
CREATE TABLE BIGDEPT(DEPTNO NUMBER not null, DNAME VARCHAR2(14),
    LOC VARCHAR2(13));
Alter table BIGDEPT ADD constraint
pk_BIGDEPT primary key(DEPTNO);
Create index IS_BIGDEPT_LOC on BIGDEPT(LOC,DEPTNO);
CREATE TABLE BIGEMP( EMPNO NUMBER not null, ENAME VARCHAR2(10) not null,
    JOB VARCHAR2(9),
    MGR NUMBER,
    HIREDATE DATE,
    SAL NUMBER(7,2),
    COMM NUMBER(7,2),
    DEPTNO NUMBER not null);
Alter table BIGEMP ADD constraint pk_BIGEMP primary key(EMPNO);
Alter table BIGEMP ADD constraint fk_BIGEMP_DPT
foreign key(DEPTNO) references BIGDEPT(DEPTNO);
Alter table BIGEMP ADD check(ename=UPPER(ename));
Alter table BIGEMP ADD check (SAL > 500);
```

FIGURE 37 : SCRIPT DE CREATION DE LA BASE DE DONNEES DRH