Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté desSciences
Département d'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de Master en Informatique

Option: *Système d'Information et de Connaissances (S.I.C)*

Thème

Optimisation des bases de données

Mise en œuvre sous Oracle

Réalisé par :

- **BERRADANE Fatima Zohra**
- **MEDJAHDAOUI Rabiea**

Présenté le 22 Juin 2015 devant le jury composé de MM.

- ***Mme*** **ILES Nawel** *(Présidente)*
- ***Mr*** **MATALLAH Houcine** *(Encadreur)*
- ***Mme*** **EL YEBDRI Zeyneb** *(Examinatrice)*
- ***Mme*** **BENMANSOUR Fazilet** *(Examinatrice)*

Année universitaire : 2014-2015

REMERCIEMENTS

Nous remercions vivement Notre créateur (Allah) pour nous avoir donné de la force à accomplir ce travail.

Notre profonde gratitude à Mr Matallah houcine , d'avoir proposé ce sujet et qui s'est toujours montré à l'écoute et très impliqué tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'il a bien voulu nous consacrer et sans qui ce mémoire n'aurait jamais vu le jour.

Nous tenons à exprimer nos sincères remerciements aux membres du jury qui nous font honneur en jugeant ce travail.

Nous adressons également notre gratitude à tous les professeurs de l'université ABOUBAKR BELKAID en particulier ceux du département INFORMATIQUE.

Enfin, nous remercions toutes les personnes qui, de près ou de loin, ont contribué à l'élaboration de ce mémoire.

Dédicaces

Grâce à dieu le tout puissant qui m'a donné le privilège et la chance d'étudier et de suivre le chemin, je dédie ce modeste travail

- À l'âme de mon père, que dieu lui garde dans son paradis
- À celle qui m'a donné la vie, et a veillé mes nuits, à celle qui a vécue et a puisé sa vie que pour me voir un jour réussir, à ma très chère mère, pour son indéfectible soutien, les marques d'amour et les sacrifices.

Nul mot ne saura exprimer mon amour, ma reconnaissance envers vous. Que Dieu vous protège et vous accorde une longue vie.

- À la lumière de ma vie, mon cher mari sidi Mohamed, à qui je souhaite une longue vie pleine de bonheur et de santé;
- À mes plus belles étoiles qui puissent exister dans l'univers, mes enfants Diyaeddine, Nizar et Ouissal ;
- À mes chers frères Mohamed et Hicham pour leur amour inconditionnel ;
- À ma très chère sœur Malika et ma sœur douce Hafida et sa petit famille, qui m'ont encouragé et soutenu tout au long de mes études ;
- À toute ma famille ;
- À mes chères amies Dalila et Mansouria et toute l'équipe de la pharmacie centrale E.P.H Ghazaouet ;
- À mon binôme Rabiea et mes camarades de la promotion Master2 SIC 2014/2015
- À mon encadreur et tous mes enseignants ;
- À tous ceux qui me portent dans leurs cœurs.

Fatima Zohra Berradane

Dédicaces

Je dédie ce modeste travail :

A l'homme de ma vie, mon soutien moral
et source de joie et de bonheur, celui qui s'est toujours sacrifié
pour me voir réussir de toutes ma vie, que dieu te garde dans son
vaste paradis, à toi Mon père.
A la lumière de mes jours, la source de mes efforts, la flamme de
mon cœur, ma vie et mon bonheur, maman que j'adore.
A mes chères sœurs
A mes chers frères
A Manel, Kawtar, Aichoucha, Nessrine
A Abdel Basset, Khalil
A toutes la famille de maman ***Lahmar***
A toutes ma famille ***Medjahdaoui***
A notre encadreur Mr. MATTALAH
A mon binôme Fatima Zohra
A mes très chères amies et surtout Ibtissam, Hassna.
A tous ceux qui m'ont soutenu moralement de près ou de loin.
Enfin à toute(es) les étudiants de la promotion Master2
informatique avec lesquelles j'ai passé cette année.

Rabiea

Résumé

L'optimisation des bases de données est au cœur des problématiques soulevées par les administrateurs des bases de données. Elle a pour objectif de fournir, aux différents utilisateurs, un système optimal en termes de coût et d'usage. L'optimisation peut être réfléchie à plusieurs niveaux : au niveau conceptuel, au niveau infrastructure matérielle et au niveau exécution des requêtes pour atteindre les meilleures performances en terme de temps de réponse et utilisation de ressources.

Notre PFE consiste à appliquer un ensemble de tests comparatifs sur une base de données, en variant plusieurs paramètres dans les différentes phases de notre étude, pour aboutir au meilleur temps de réponse ou meilleur coût en fonction des ressources matérielles disponibles.

Mots clés : BDD, SGBDR, ORACLE, CBO, Optimisation

Abstract

The optimization of data bases is at the heart of the issues raised by the database administrator. Its objective is to provide, to different users, an optimal system in terms of cost and use. Optimization fear being reflected at several levels: the conceptual level, the physical infrastructure level and at query execution to achieve the best performance in terms of response time and use of resources.

Our PFE consists of applying a set of comparative test son a data base, by varying several parameters in different phases of our study, to achieve the best response time or the lowest cost in terms of material resources.

Keywords : DB, DBMS, ORACLE, CBO, Optimisation

ملخص

تحسين الاستفادة من قواعد البيانات هو في قلب القضايا التي يثيرها مدراء قواعد البيانات. هدفها هو توفير، لمختلف المستخدمين، النظام الأمثل من حيث التكلفة والاستخدام. يمكن التفكير فيها على عدة مستويات :أثناء تصميم قواعد البيانات، على مستوى الأجهزة وعند طرحنا للأسئلة لتحقيق أفضل أداء من حيث زمن الاستجابة واستخدام الأجهزة.

يسعى عملنا الى اجراء مقارنة عبرمجموعة من الاختبارات على قاعدة البيانات، من خلال تغيير العديد من العوامل في مختلف مراحل الدراسة التي قمنا بها ، من أجل تحقيق أفضل وقت استجابة أو بأقل تكلفة من حيث الأجهزة المتوفرة.

Table des matières

Introduction générale

Chapitre I : Concepts de base

Chapitre II : SGBD ORACLE

Chapitre III : Optimisation des Bases de Données

Introduction générale

Introduction

Aujourd'hui, le concept de base de données, est plus que jamais au cœur des applications. Le champ d'utilisation de bases de données s'est étendu considérablement, au début réservé à l'informatique de gestion, puis elles sont petit à petit apparues dans les autres domaines. La donnée est devenue un enjeu central pour les entreprises des différents secteurs. La question qui s'impose est comment traiter une masse d'informations toujours plus grande ? et comment peut-on avoir des bases répondant aux besoins d'évolutivité et de performance ?

Problématique

De nombreuses problématiques opposent les différents chercheurs dans le domaine des bases de données : sécurité, intégrité, mapping objet / relationnel, amélioration des performances et l'optimisation des bases de données. Justement, sur cette dernière que va porter le sujet de ce mémoire.

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 des traitements dépasse les délais permis ou encore les délais raisonnables ou d'une autre manière, les informations contenues dans la base ne sont pas envoyées rapidement au script qui les demande ou un coût élevé d'utilisation de ressources. Cependant, La réduction de la latence, l'augmentation de la productivité des utilisateurs et l'amélioration des performances de la base de données sont primordiales pour assurer le succès du fonctionnement du système.

Solution et Objectif

L'optimisation des bases de données est un sujet très vaste, elle est souvent l'objet des débats d'experts et des administrateurs de base de données(DBA) sur les causes générant les effets indésirables, elle se présente sur trois facettes :

- Optimisation du modèle de données
- Optimisation d'infrastructure
- Optimisation des requêtes

Dans chacun de ces trois cas majeurs, il est nécessaire de définir un ensemble de métriques pour l'optimisation, qui s'appuient sur différents critères.

Dans ce mémoire nous allons explorer les trois axes, en parcourant les principales techniques d'optimisation disponibles pour pouvoir estimer les améliorations apportées.

Organisation du rapport de mémoire

Ce manuscrit est entamé par une introduction générale et s'articule autour de quatre chapitres :

- Dans le premier, nous présentons quelques concepts de base en relation avec le sujet abordé.
- Le deuxième chapitre est consacré à la présentation du SGBD ORACLE.
- Le troisième chapitre décrit la notion d'optimisation de base de données en abordant les axes majeurs.
- Le quatrième chapitre sera consacré aux différentes expérimentations faites, ou nous allons proclamer les impacts de choix des solutions d'optimisation.

Nous achèverons ce mémoire par une conclusion générale et quelques perspectives.

Chapitre I

Concepts de base

I.1. Introduction

Avant d'entamer notre étude, il est nécessaire de rappeler quelques notions élémentaires qui seront utiles tout au long de ce mémoire, à savoir base de données, système de gestion de base de données, modèle relationnel, SQL, index.

I.2. Base de données

Une base de données est une entité dans laquelle est possible de stocker une grande quantité d'informations de façon structurée et avec le moins de redondance possible.

Elle permet de mettre ces informations à la disposition d'utilisateurs pour une consultation, une saisie ou bien une mise à jour, tout en s'assurant des droits accordés. Son avantage majeur est le fait qu'elle soit exploitée par plusieurs utilisateurs simultanément.

Une base de données peut être locale, centralisée dans une machine manipulée par un ou plusieurs utilisateurs, ou bien répartie, les informations sont stockées sur des machines distantes et accessibles par réseau.

Les bases de données sont apparues à la fin des années 60, à une époque où la nécessité d'un système de gestion de l'information souple se faisait ressentir.

I.3. Système de gestion de base de données (SGBD)

Le SGBD (Système de Gestion de Bases de Données) ou en anglais DBMS (Data Base Management System) est un ensemble de services ou applications logicielles permettant de gérer les bases de données.

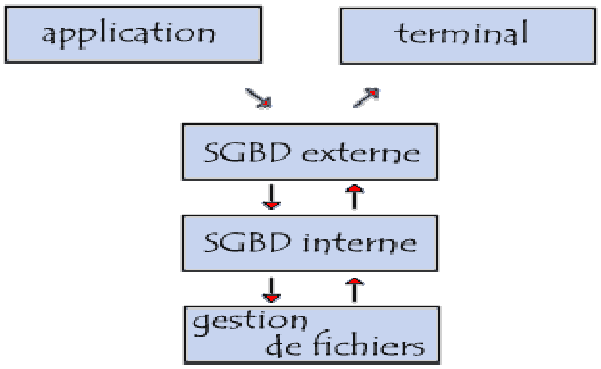
Capacités basiques d'un SGBD:

- Stockage d'un grand volume de données avec durabilité.
- Accès efficace.
- Support d'un modèle de données.
- Permettre à l'utilisateur de créer des nouvelles bases de données ainsi que de spécifier leurs schémas (utilisation de DDL).
- Permettre à l'utilisateur d'interroger et de modifier les données de la base.
- 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. [1]

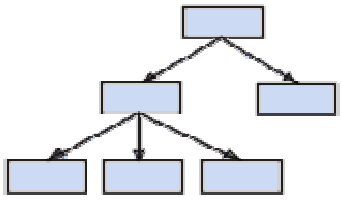
Le SGBD peut se décomposer en trois sous-systèmes :

- **Système de gestion de fichiers :** permet le stockage des informations sur un support physique.
- **SGBD interne :** gère l'ordonnancement des informations.
- **SGBD externe :** représente l'interface avec l'utilisateur.

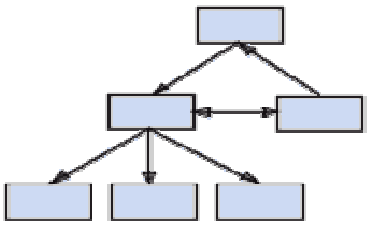


Différents modèles de SGBD ont été conçus pour supporter certaines exigences, on peut les classer comme suit:

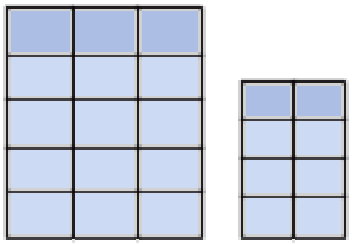
- **Modèle hiérarchique** : les données sont classées hiérarchiquement, selon une arborescence descendante. Ce modèle utilise des pointeurs entre les différents enregistrements. Il s'agit du premier modèle de SGBD.



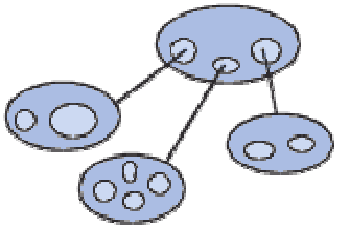
- **Modèle réseau** : comme le modèle hiérarchique ce modèle utilise des pointeurs vers des enregistrements. Toutefois la structure n'est plus forcément arborescente dans le sens descendant.



- **Modèle relationnel** (**SGBDR**, *Système de gestion de bases de données relationnelles*) : les données sont enregistrées dans des tableaux à deux dimensions (lignes et colonnes). La manipulation de ces données se fait selon la théorie mathématique des s.



- **Modèle déductif** : les données sont représentées sous forme de table, mais leur manipulation se fait par calcul de prédicats.
- **Modèle objet** (**SGBDO**, *Système de gestion de bases de données objet*) : les données sont stockées sous forme d'objets, c'est-à-dire de structures appelées *classes* présentant des données membres. Les champs sont des instances de ces classes. [2]



Parmi les SGBD les plus connus : Oracle, Informix, Sybase, Ingres, Microsoft SQL server, MySQL, PostgreSQL, SQLite, Firebird, DB2.

I.4. Modèle relationnel

Le modèle relationnel représente la base de données comme un ensemble de tables, sans préjuger de la façon dont les informations sont stockées dans la machine. Les tables constituent donc la structure logique du modèle relationnel. Au niveau physique, le système est libre d'utiliser n'importe quelle technique de stockage (fichiers séquentiels, indexage, adressage dispersé, séries de pointeurs, compression...), dès lors qu'il est possible de relier ces structures à des tables au niveau logique. Les tables ne représentent donc qu'une abstraction de l'enregistrement physique des données en mémoire. [3]

Le modèle relationnel est fondé sur la théorie mathématique de l'algèbre relationnel. Il permet de produire une représentation simple des données sous forme de

tables constituées de lignes et de colonnes. Il n'y a donc pas de pointeurs qui figent la structure de la base. Cela abouti sur une représentation souple des données dont on peut extraire des informations à travers des langages non procéduraux, développés grâce à l'algèbre linéaire, l'exemple le plus populaire est SQL (Structured Query Language ou Langage Structuré de Requêtes).

Le modèle relationnel est le modèle le plus utilisé par les SGBD actuellement disponibles sur le marché.

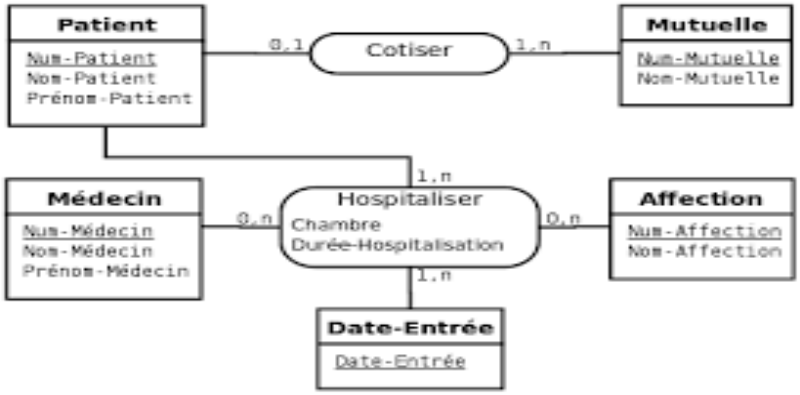


Figure I.1 : *Exemple de modèle relationnel*

I.5. SQL (Structured Query Language)

SQL est un langage pour les bases de données relationnelles. Créé en 1970 par IBM. Il est caractérisé, principalement par :

- **Normalisation** : SQL implémente le modèle relationnel.
- **Standard** : Du fait de cette normalisation, la plupart des éditeurs de SGBDR intègrent SQL à leurs produits (Oracle, Informix, Sybase, Ingres, MS SQL Server, DB2, etc.). Ainsi, les données, requêtes et applications sont assez facilement portables d'une base à une autre.
- **Non procédural** : SQL est un langage de requêtes qui permet à l'utilisateur de demander un résultat sans se préoccuper des moyens techniques pour trouver ce résultat (assertionnel). C'est l'optimiseur du SGBD (composant du moteur) qui se charge de cette tâche.
- **Universel** : SQL peut être utilisé à tous les niveaux dans la gestion d'une base de données relationnelle. Il est composé de trois sous langages :

1. Langage de Définition de Données LDD, permet la description de la structure de la base de données (tables, vues, attributs, index) : CREATE, ALTER, DROP.
2. Langage de Manipulation de Données LMD, permet la manipulation des tables et des vues avec les quatre commandes : SELECT, INSERT, DELETE, UPDATE.
3. Langage de Contrôle de Données LCD, comprend les primitives de gestion des transactions : COMMIT, ROLLBACK et des privilèges d'accès aux données : GRANT et REVOKE.

I.6. Index

Les bases de données prennent souvent des proportions importantes, voire considérables. Si une recherche d'information dans une table s'effectue de manière simplement séquentielle (c'est à dire en examinant toute la table, ou du moins tous les champs concernés, du début jusqu'à la fin), le temps d'attente peut devenir prohibitif pour l'opérateur. L'index est l'outil qui permet de résoudre ce problème.

L'index est une structure de données utilisée et entretenue par le système de gestion de base de données (SGBD) pour lui permettre de retrouver rapidement les données, en accédant aux enregistrements, selon la valeur d'un ou plusieurs champs.

L'utilisation d'un index simplifie et accélère les opérations de recherche, de tri, de jointure ou d'agrégation effectuées par le SGBD, il est de taille très inférieure à celle de la table : on peut le remettre à jour en temps réel à chaque modification de cette dernière, il peut servir à empêcher l'opérateur de créer des enregistrements dupliquées en saisissant deux fois, par erreur, les mêmes données. [4]

Types d'index

a. **L'arbre B (*B-tree*)** : les index B*Tree sont présents dans de nombreux SGBDR. Leur principe est de contenir les valeurs des clés de l'index de façon ordonnée dans une structure arborescente. Ainsi, ils permettent de trouver très rapidement une valeur précise en parcourant l'arbre de la racine vers les feuilles et non pas en recherchant la valeur dans l'ensemble des enregistrements de la table.

De plus, les feuilles sont chainées entre elles. Aussi, lorsqu'il est nécessaire de parcourir l'index séquentiellement, on passe de feuille en feuille sans devoir remonter au niveau des branches.

b. **Index bitmap** : Les index bitmap sont adaptés à la manipulation de colonnes avec peu de valeurs distinctes. Ils ne sont utilisables que pour des prédicats d'égalité, mais Oracle arrive, dans certains cas, à convertir des intervalles par un ensemble d'égalité, rendant ainsi ces index opérants sur des intervalles et même sur des jointures.

Les index bitmap sont des masques de bits pour les valeurs distinctes de colonnes indexées : des ET et OU binaires permettent de faire des tests d'égalité.

c. **Table de hachage :** Une méthode de transformation simple consiste à attribuer une adresse (représentée par un nombre naturel de 1 à n) à chaque valeur de clé choisie dans une table. Cette adresse est interprétée comme un numéro de page relatif.

Chaque page contient un nombre fixe de valeurs de clé, avec ou sans les enregistrements de données correspondants. [5]

I.7. Conclusion

Après avoir abordé les principaux concepts relatives au fonctionnement de base de données, nous allons présenter dans ce qui suit, un des poids lourds du marché des SGBD, à savoir Oracle dans sa version 10g Professionnel.

Chapitre II

SGBD ORACLE

II.1. Introduction

Dans le cadre de notre étude, on a opté pour Oracle 10g Professionnel, puisqu'actuellement, c'est le SGBD le plus populaire et le leader mondialement. Il est basé sur le modèle relationnel et il offre un véritable environnement de travail doté d'un ensemble d'outils d'administration.

II.2. Les fonctionnalités d'Oracle

Oracle Database10g est le système de gestion de bases de données Oracle, permettant à plusieurs utilisateurs d'accéder simultanément aux données tout en garantissant une disponibilité et des mécanismes de récupération après incident.

Pour garantir un niveau de performance élevé, Oracle maintient la base de données grâce à des structures mémoires (sur mémoire vive) et des structures physiques (sur disque dur) et utilise des processus pour le stockage des données en mémoire ou sur disque.

L'écriture sur disque n'est effectuée qu'en cas de nécessité et sous conditions tandis que la mémoire est utilisée autant que possible étant donné que l'accès mémoire est plus rapide que l'accès disque (*gain en performances*). [6]

Oracle est un SGBD permettant d'assurer :

- La définition et la manipulation des données.
- La cohérence des données.
- La confidentialité et l'intégrité des données.
- La sauvegarde et la restauration des données.
- La gestion des accès concurrents.

II.3. Les composants d'Oracle

Oracle est constitué de nombreux modules logiciels permettant une administration graphique, de s'interfacer avec des produits divers et d'assistants de création de bases de données et de configuration de celles-ci. [7]

On peut classer les outils d'Oracle selon diverses catégories :

- Les outils d'administration
- Les outils de développement
- Les outils de communication
- Les outils de génie logiciel
- Les outils d'aide à la décision

II.3. 1. Les outils d'administration d'Oracle

Oracle est fourni avec de nombreux outils permettant de simplifier l'administration de la base de données. Parmi ces outils, les plus connus sont :

- Oracle Manager (SQL*DBA)
- NetWork Manager
- Oracle Enterprise Manager
- Import/Export : un outil permettant d'échanger des données entre deux bases Oracle

II.3. 2. Outils de développement d'Oracle

Oracle propose également de nombreux outils de développement permettant d'automatiser la création d'applications s'interfaçant avec la base de données. Ces outils de développement sont :

- Oracle Designer
- Oracle Developer
- SQL*Plus : une interface interactive permettant d'envoyer des requêtes SQL et PL/SQL à la base de données. SQL*Plus permet notamment de paramétrer l'environnement de travail (formatage des résultats, longueur d'une ligne, nombre de lignes par page, ...)
- Oracle Developper : il s'agit d'une suite de produits destinés à la conception et à la création d'applications client-serveur. Il est composé de 4 applications :
  1) Oracle Forms (anciennement SQL*Forms) : un outil permettant d'interroger la base de données de façon graphique sans connaissances préalables du langage SQL. SQL*Forms permet ainsi de développer des applications graphiques (fenêtres, formulaires, ...) permettant de sélectionner, modifier et supprimer des données dans la base.
  2) Oracle Reports (SQL*ReportWriter) : un outil permettant de réaliser des états
  3) Oracle Graphics : un outil de génération automatique de graphiques dynamiques pour présenter graphiquement des statistiques réalisées à partir des données de la base
  4) Procédure Builder : un outil permettant de développer des procédures, des fonctions et des packages [8]

II.3. 3. Outils de programmation

Oracle dispose d'un grand nombre d'interfaces (API) permettant à des programmes écrits dans divers langages de s'interfacer avec la base de données en envoyant des requêtes SQL. Ces interfaces (appelées précompilateurs) forment une famille dont le nom commence par *PRO** : [9]

- Pro*C
- Pro*Cobol
- Pro*Fortran
- Pro*Pascal
- Pro*PLI
- ...

II.4. Structure interne d'Oracle

Lorsqu'une base de données est démarrée, Oracle alloue une zone de mémoire partagée nommée SGA (System Global Area) et lance plusieurs processus Oracle en arrière-plan.

La mémoire SGA et les processus Oracle forme une instance Oracle. Lorsque l'instance démarre, Oracle l'associe à une base de données, c'est le montage de la base de données. La base de données est alors ouverte pour les utilisateurs et les applications. Plusieurs instances peuvent s'exécuter simultanément sur le même ordinateur, chacune ayant accès à sa propre base de données physique.

II.4.1. Le serveur Oracle

IL s'agit du système installé sur une machine qui va permettre la gestion de toutes les bases de données disponibles sur la machine. Le serveur oracle est basé sur une architecture multiserveur. Le serveur est responsable de traitement de toutes les activités de la base de données tel que l'exécution des instructions SQL, gestion des ressources, gestion des utilisateurs ainsi que la gestion du stockage. Pour consulter les données, l'utilisateur doit tout d'abord se connecter à un Serveur Oracle.

Un serveur Oracle = une base de données sur disque + des données chargées en mémoire ou SGA+ des processus d'arrière-plan.

Lorsque un utilisateur est connecté à une machine sur laquelle réside un serveur Oracle, deux processus supplémentaire sont invoqués : Le processus utilisateur et le

processus serveur. Une connexion spécifique entre un utilisateur et un serveur Oracle est appelé une Session.

De nombreuses sessions concurrentes d'un même utilisateur ou de plusieurs peuvent s'exécuter sur le serveur Oracle.

La figure suivante illustre l'architecture du serveur Oracle constituée des structures du stockage, les processus, et les structures mémoire. [10]

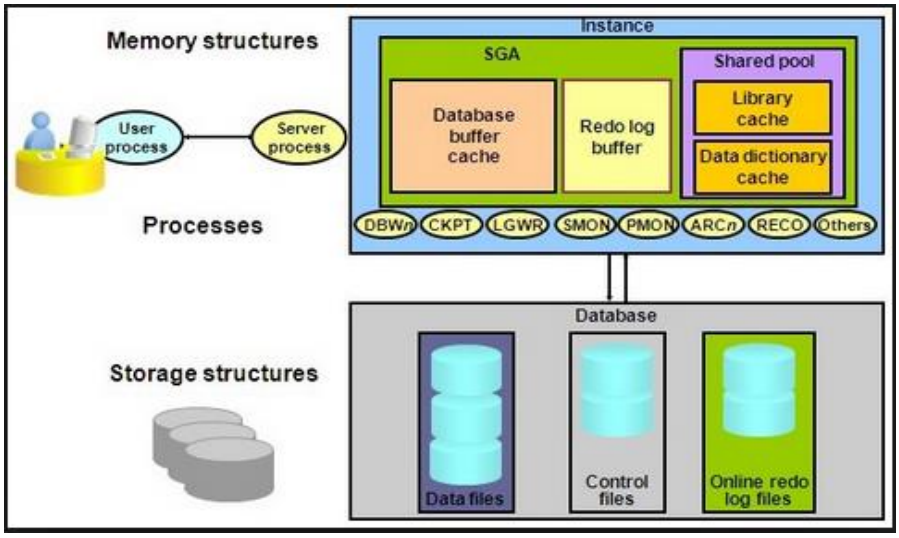


Figure II.1 : *Architecture serveur oracle*

II.4.2. Instance oracle

Chaque base de données Oracle en cours d'exécution est associée à une instance Oracle. Une instance Oracle est composée de structures mémoire, appelées mémoire SGA (System Global Area), et de processus en arrière-plan qui gèrent la plupart des opérations qui ont lieu "en coulisses" lors de l'exécution d'une instance.

Une fois qu'une instance est démarrée, la base peut être montée et les utilisateurs peuvent avoir accès aux informations présentes dans la base.

A. Structure mémoire

a. La mémoire SGA (System Global Area)

Elle est partagée par tous les processus du serveur et les processus en arrière-plan. Cette mémoire contient des informations de contrôle de données sur l'instance. Elle comprend les structures de données suivantes :

- **Buffer cache (cache de tampons)** : met en mémoire cache les blocs de données extraits de la base de données.
- **Redo log buffer (tampon de journalisation)** : met en mémoire cache les informations de journalisation jusqu'à ce qu'elles soient écrites dans les fichiers de journalisation (redo log) physiquement stockés sur disque.
- **Shared pool (zone mémoire partagée)** : met en mémoire diverses informations partagées par les utilisateurs.
- **Large pool (zone mémoire étendue)** : zone de mémoire (facultative) pour des processus volumineux (sauvegarde, récupération de données, import/export, ...).
- **Java Pool (zone de mémoire Java)** : mémoire dédié pour l'ensemble du code Java et des données de sessions dans la JVM (Machine Virtuelle Java).
- **Streams Pool** : mémoire dédié à Oracle Streams (système de réplication transactionnel Oracle). [8]

b. La mémoire PGA (Program Global Area)

Elle est propre à chaque processus du serveur et des processus en arrière-plan. Chaque processus dispose de sa mémoire PGA. La PGA stocke des informations de contrôle spécifiques à la session utilisateur (zones privées pour le traitement des curseurs, les variables attachées, des informations sur les sessions, une zone de tri ...). La mémoire PGA se libère dès qu'une session est fermée.

B. Les processus

Le fonctionnement de la base Oracle est régi par un certain nombre de processus chargés en mémoire permettant d'assurer la gestion de la base de données. On distingue généralement deux types de processus :

1) **Les processus utilisateurs** (appelés aussi user process ou noyau oracle)

Ces processus assurent la liaison entre les programmes utilisateurs, les processus de la base de données, la zone SGA et les fichiers qui composent la base. Le processus utilisateur affiche aussi l'information demandée par l'utilisateur.

Un processus utilisateur est créé pour chaque programme exécuté par un utilisateur afin de fournir l'environnement nécessaire à l'exécution de celui-ci.

2) **les processus systèmes** (oracle process)

Les processus Oracle (processus système) se classent en deux catégories :

- **Les processus serveurs** (process server) : gérant les requêtes des utilisateurs provenant des connexions à la base de données générées par des outils tels que SQL*Plus. Le processus serveur est chargé de la communication entre la SGA et le processus utilisateur. Il permet ainsi d'analyser et d'exécuter les requêtes SQL des utilisateurs, de lire les fichiers de données et de placer les blocs de données correspondants dans la SGA.
- **Les processus d'arrière-plan** (background process) : chargés d'assurer le fonctionnement interne du SGBD Oracle (gestion de la mémoire, écriture dans les fichiers, ...).

Les processus en arrière-plan les plus courants sont les suivants :

- **SMON** (System Monitor) : effectue la récupération après une panne lorsque l'instance est démarrée suite à une défaillance.
- **PMON** (Process Monitor) : nettoyage des processus utilisateur en cas d'échec.
- **DBWn** (DatabaseWriter) : écrit les blocs modifiés du cache tampon (buffer cache) de la base de données vers les fichiers de données stockés sur disque.
- **CKPT** (Checkpoint) : met à jour l'ensemble des fichiers de données et de contrôle de la base de données pour indiquer le point de reprise le plus récent.
- **LGWR** (Log Writer) : écrit les entrées de journalisation sur le disque (redologs).
- **ARCn** (Archiver) : copier les fichiers de journalisation (redologs) dans l'emplacement de stockage d'archivage en cas de changement de fichier de journalisation (archivelog). [11]

II.4.3. La base de données Oracle

Une base de données Oracle est une collection de données traitées comme une unité. Son architecture peut être considérée comme un ensemble de composants structurels interdépendants. Elle est gérée et accessible via des processus et des structures mémoire. Toutes les structures mémoires se trouvent dans la mémoire principale des ordinateurs qui composent le serveur de base de données.

Une base de données Oracle comporte des structures logiques et des structures physiques. Elle est constituée physiquement de plusieurs fichiers de données regroupés logiquement en tablespace. Chaque base de données est divisée logiquement en plusieurs tablespaces. Le fichier de données d'un tablespace peut être stocké physiquement sur toutes les technologies de stockage pris en charge par Oracle. Les

niveaux de stockages suivant sont les segments, composés d'extents, composés de bloc de données Oracle.

A partir de la version 10G d'Oracle, une base de données comporte au minimum deux fichiers de données appartenant aux tablespaces SYSTEM et SYSAUX. [12]

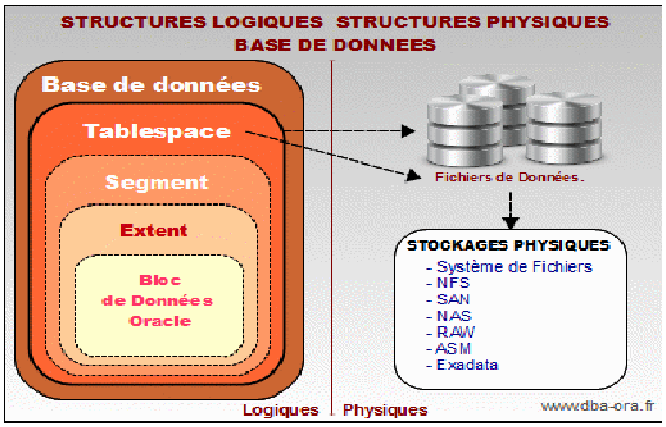


Figure II.2 : *Architecture base de données Oracle*

Une base de données est composée des fichiers suivants :

- **Control files** (fichiers de contrôle) : contiennent des données relatives à la base de données elle-même (c'est-à-dire des informations propres à la structure de base de données physique). Ces fichiers sont essentiels pour la base de données. Sans eux, il est impossible d'ouvrir les fichiers de données pour accéder aux données de la base.
- **Data files** (fichiers de données) : contiennent les données d'application ou les données utilisateur de la base.
- **Redo log** (fichiers de journalisation) : permettent la récupération d'une instance de la base de données. Si la base de données connaît une défaillance et qu'aucun fichier de données n'est perdu, l'instance peut récupérer la base de données à partir des informations de ces fichiers.
- **Init file** (fichier de paramètres) : fichier de paramètres utilisés au démarrage de la base de données.
- **Passwd** (fichier des mots de passe) : permet aux utilisateurs de se connecter à distance à la base de données et d'effectuer des tâches d'administration.
- **Archives logs** : contiennent un historique des modifications des données (informations de journalisation) générées par l'instance. A l'aide de ces fichiers et d'une sauvegarde de la base de données, vous pouvez récupérer un fichier de

données perdu. Autrement dit, les fichiers de journalisation archivés permettent la récupération des fichiers de données restaurés.

- **Fichier de sauvegardes** : utilisé en cas de défaillance physique de l'instance et/ou de la base de données.
- **Fichier trace** : Chaque processus serveur et processus en arrière-plan peut écrire dans un fichier trace associé. Lorsqu'un processus détecte une erreur interne, il réalise dans son fichier trace un dump des informations relatives à cette erreur. Certaines informations écrites dans un fichier trace sont destinées à l'administrateur de base de données, et d'autres au support technique Oracle.
- **alert logs** (fichiers d'alertes) : Il s'agit de fichiers trace spéciaux, dans lesquels sont consignées les alertes. Le fichier d'alertes d'une base de données est un journal chronologique des messages et des erreurs. Oracle recommande de consulter ces fichiers. [11]

II.5. Conclusion

Dans ce chapitre nous avons mis l'accent sur les principes de base nécessaires à la compréhension du fonctionnement du SGBDR Oracle, ce qui nous permet d'aborder le problème d'optimisation et faire le choix des solutions dans le chapitre suivant.

Chapitre III

Optimisation des Bases de Données

III.1. Introduction

L'optimisation est un aspect crucial des bases de données, son objectif est de fournir aux utilisateurs un système optimal en termes de coût et d'usage. Elle se situe à tous les niveaux, dés la conception jusqu'à la mise en œuvre.

L'optimisation d'une base de données est une activité qui peut avoir lieu :

- De façon curative, c'est-à-dire lorsque les problèmes apparaissent lors de l'implémentation ou en production.
- De façon plus préventive lors de la conception de la base. [5]

De ce fait, trois axes majeurs sont distingués.

III.2. Optimisation du modèle conceptuel de données

Lors de la conception d'un projet, une base se conçoit en faisant le choix de l'implantation optimale des données de façon à obtenir les meilleures performances.

La conception et l'implémentation d'une base de données supposent l'identification des tables de grande taille, ainsi que les processus complexes que la base de données devra effectuer.

Il est également nécessaire d'accorder une attention particulière aux performances lors de la conception de ces tables, aussi de prendre en considération l'impact sur les performances dû à l'augmentation du nombre d'utilisateurs pouvant accéder à la base de données.

III.2.1. Présentation du modèle

Les bases de données relationnelles étudiées dans ce sujet, sont fondées sur le modèle relationnel. Un des grands intérêts du modèle relationnel est le fait qu'il repartit l'ensemble des données dans différentes relations et établit des associations entre ces relations au moyen de leur clé primaire. Ce principe permet d'éviter la duplication d'information et donc d'avoir des données plus consistantes.

Le modèle relationnel a été conçu pour être utilisé avec une algèbre relationnelle qui permet d'effectuer des opérations sur les entités. Les principales sont :

- **La jointure :** Relie deux entités par leur association.
- **La sélection :** Filtre les tuples dont les attributs répondent à un prédicat.
- **La projection :** Sélectionner seulement certains attributs d'une entité.

Généralement pour mettre en place des schémas de bases de données performants, il est important d'utiliser les bonnes pratiques de modélisation et de conception de base de données ainsi que le choix de la méthodologie appropriée. Pour ce faire, il faut prendre en considération les points suivants :

- Choix approprié des types des champs : il faut choisir les variables les plus adaptées au besoin (exemple pour le stockage d'un nombre ne dépassant pas les 10 chiffres, il est plus judicieux d'utiliser un type TINYINT). Utiliser les champs les plus petits possible permet de charger en mémoire plus de colonnes.
- Utilisation de champs à longueur fixe : l'utilisation de longueurs prédéterminées permet d'optimiser l'accès aux colonnes car leurs positions sont prédéfinies. Ceci implique diminuer l'utilisation de VARCHAR, de TEXT et de BLOB.
- Faire un bon indexage des tables : L'index représente une structure de données facile et rapide à parcourir. Si une recherche est fréquente et qu'elle inclut une ou plusieurs colonnes, il faut créer l'index correspondant pour optimiser le temps de réponse.

D'après le Schéma Conceptuel de Données (SCD), obtenu à la phase d'analyse (un ensemble d'entités et d'associations ou un ensemble de classes selon le formalisme utilisé), la transformation en schéma relationnel (Schéma Logique de Données SLD) permet l'implantation du SCD dans une base de données relationnelle, et l'exploitation de cette dernière par le SGBD et les modules de programmation.

Pour s'assurer de la bonne conception du SLD, et du non redondance de ses données, nous devons appliquer le processus de normalisation.

III.2.2. Normalisation

La normalisation des modèles de données permet de vérifier la robustesse de leur conception pour améliorer la modélisation (et donc obtenir une meilleure représentation) et faciliter la mémorisation des données en évitant la redondance et les problèmes sous-jacents de mise à jour ou de cohérence. La normalisation s'applique à toutes les entités et aux relations porteuses de propriétés.

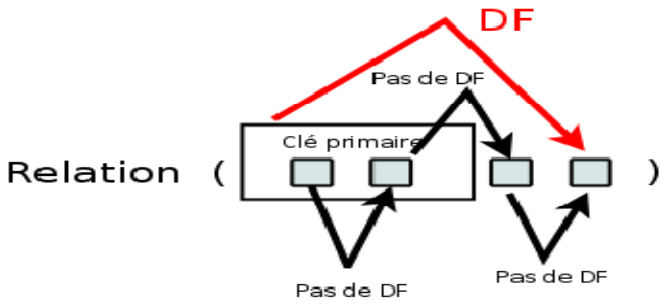


Figure III.1 : *Format d'une relation bien conçue*

Pour qu'un modèle relationnel soit normalisé, il faut qu'il respecte certaines contraintes appelées les formes normales, qui s'appuient sur les dépendances fonctionnelles entre attributs.

A. 1° forme normale

Une relation est normalisée en première forme normale si:

1) Elle possède une clé identifiant de manière unique et stable pour chaque ligne.
2) Chaque attribut est monovalué (ne peut avoir qu'une seule valeur par ligne).
3) Aucun attribut n'est décomposable en plusieurs attributs significatifs.

B. 2° forme normale

Une relation R est en deuxième forme normale si et seulement si:

1) Elle est en 1FN
2) Tout attribut non clé est totalement dépendant de toute la clé.

Autrement dit, aucun des attributs ne dépend que d'une partie de la clé.

La 2FN n'est à vérifier que pour les relations ayant une clé composée. Une relation en 1FN n'ayant qu'un seul attribut clé est toujours en 2FN

C. 3° forme normale

Une relation est en 3° forme normale si et seulement si:

1) Elle est en 2° forme normale
2) Tout attribut doit dépendre directement de la clé, c'est-à-dire qu'aucun attribut ne doit dépendre de la clé par transitivité.

Autrement dit, aucun attribut ne doit dépendre d'un autre attribut non clé.

D. Forme normale de Boyce Codd (BCNF)

La forme normale de Boyce Codd est une version plus contraignante de la troisième forme normale.

Une relation est en forme normale de Boyce Codd (BCNF), si et seulement si :

1) Elle est en troisième forme normale (3NF).
2) Tous les attributs ne faisant pas partie de ses clés dépendent exclusivement des clés candidates.

E. Autres formes de normalisation

Il existe une cinquième forme normale encore appelée PJ/NF (*Project/Join Normal Form* : forme normale par projection/jointure), et une sixième forme normale, mais elles sont rarement prises en compte en pratique.

Normalisation d'un exemple de table

Les étapes suivantes démontrent le processus de normalisation d'une table d'étudiants fictive.

- Table non normalisée :

N° d'étudiant	Conseiller	Salle-Conseiller	Cours1	Cours2	Cours3
1022	Ali	412	101-07	143-01	159-02
4123	Omar	216	201-01	211-02	214-01

- Première forme normale : aucune répétition de groupe.

Les tables ne doivent avoir que deux dimensions. Un étudiant ayant plusieurs cours, trois cours doivent être répertoriés dans une table distincte. Les champs Cours1, Cours2 et Cours3 des enregistrements ci-dessus sont l'indication d'un problème conceptuel.

Créer une autre table en première forme normale en éliminant le groupe répété (N° de cours), tel qu'illustré ci-dessous :

N° d'étudiant	Conseiller	Salle-Conseiller	N° de cours
1022	Ali	412	101-07
1022	Ali	412	143-01
1022	Ali	412	159-02
4123	Omar	216	201-01
4123	Omar	216	211-02
4123	Omar	216	214-01

- Deuxième forme normale : Eliminer les données redondantes.

Notant les valeurs multiples de N° de cours pour chaque valeur de N° d'étudiant dans la table ci-dessus. N° de cours ne dépend pas fonctionnellement de N° d'étudiant (clé primaire) ; cette relation n'est donc pas en deuxième forme normale.

Les deux tables suivantes démontrent la deuxième forme normale :

Étudiants

N° d'étudiant	Conseiller	Salle-Conseiller
1022	Ali	412
4123	Omar	216

Inscription

N° d'étudiant	N° de cours
1022	101-07
1022	143-01
1022	159-02
4123	201-01
4123	211-02
4123	214-01

– Troisième forme normale : éliminer les données qui ne dépendent pas de la clé

Dans le dernier exemple, *Salle-Conseiller* (le numéro du bureau du conseiller) dépend de manière fonctionnelle de l'attribut *Conseiller*. La solution consiste à déplacer cet attribut de la table Étudiants à la table *Faculté*, tel qu'illustré ci-dessous :

Étudiants

N° d'étudiant	Conseiller
1022	Ali
4123	Omar

Faculté

Nom	Salle	Département
Ali	412	42
Omar	216	42

III.2.3. Dénormalisation

Contrairement à la normalisation qui conduit à la fragmentation des données dans plusieurs tables, la dénormalisation consiste à regrouper plusieurs tables liées par des

références, en une seule table, en réalisant statiquement les opérations de jointure adéquates.

L'objectif de la dénormalisation est d'améliorer les performances de la base de données en recherche sur les tables considérées, en implémentant les jointures plutôt qu'en les calculant.

La dénormalisation est par définition un facteur de redondance. A ce titre elle doit être utilisée à bon escient et des moyens doivent être mis en œuvre pour contrôler la redondance créée.

Un schéma doit être dénormalisé lorsque les performances de certaines recherches sont insuffisantes et que cette insuffisance à pour cause des jointures.

Bien que la normalisation soit le plus souvent considérée comme le but de la modélisation de base de données, la dénormalisation, c'est-à-dire la duplication délibérée de certaines données afin d'accélérer l'extraction des données, peut s'avérer utile dans certains cas :

- Lorsque les requêtes les plus importantes portent sur des données réparties sur plusieurs tables.
- Lorsque des calculs doivent être effectués sur une ou plusieurs colonnes avant que la requête ne renvoie une réponse.
- Si les tables doivent être consultées de différentes manières par différents utilisateurs lors d'une même période.
- Si certaines tables sont très fréquemment utilisées.

La dénormalisation peut être effectuée de différentes façons :

- **Partitionnement horizontal :** utilisé pour diviser une table en plusieurs tables contenant les mêmes colonnes, mais moins de lignes.
- **Partitionnement vertical :** utilisé pour diviser une table en plusieurs tables contenant le même nombre de lignes, mais moins de colonnes.
- **Fusion de tables :** permet de fusionner des tables afin d'éliminer la jointure entre elles.
- **Dénormalisation de colonne :** permet de répéter une colonne dans plusieurs tables afin d'éviter d'avoir à créer des jointures entre les tables. [9]

III.3. Optimisation d'infrastructure

III.3.1. Optimisation de l'exécution du SGBDR

Plusieurs paramètres sont disponibles pour adapter le comportement des SGBDR.

A. Ajustement de la mémoire utilisable

La plupart des SGBDR intègrent un paramètre permettant de configurer la quantité maximale de mémoire qu'ils peuvent utiliser. Il faut veiller à ce que ces paramètres n'étouffent pas le serveur, mais il faut que le SGBDR utilise au mieux la mémoire disponible sur le serveur.

– Sous Oracle 10g, on utilise le paramètre **SGA_TARGET** qui fait la gestion automatique de la mémoire, alors il définit la quantité de mémoire disponible pour le SGA.

B. Répartition des fichiers

Repartir les fichiers de données, de contrôle et de journalisation, sur des disques différents améliore généralement les performances en lecture et en écriture. Pour les fichiers de contrôle Oracle, cette pratique est recommandée pour des raisons de fiabilité.

Repartir les données et les indexes sur des tablespaces qui sont sur des disques différents est une bonne pratique assez répandue. [5]

III.3.2. Optimisation matérielle

Un des moyens d'amélioration des performances est d'améliorer les ressources matérielles utilisées pour faire tourner le SGBDR.

Soumise à une forte charge, une de ces ressources est susceptible de limiter la performance de l'ensemble. Les outils de surveillance des infrastructures de production permettent de déterminer quelle ressource est liée à la dégradation des performances.
Parmi les ressources matérielles, il y a :

- CPU
- RAM
- Disques durs
- Réseau

Afin de déterminer des pistes de solutions à étudier, il convient de mesurer l'utilisation des différentes ressources citées ci-dessus afin de déterminer quelles sont celles qui sont exploitées de manière trop intense.

III.4. Optimisation des requêtes

Dans le but de fournir l'algorithme d'accès à la base de données pour répondre à une requête exprimée en langage assertionnel, la requête passe par plusieurs étapes allant de l'analyse jusqu'à l'exécution. En comprenant les mécanismes internes, nous pouvons contrôler le temps qu'Oracle passe à évaluer l'ordre de jointure des tables, et améliorer les performances des requêtes en général.

Oracle permet d'analyser les requêtes et de connaître le temps et le plan d'exécution, ces informations permettent de définir ce qui ralentit l'exécution des requêtes afin de les optimiser. Une fois que les requêtes lentes détectées, lancer la commande EXPLAIN permet de comprendre l'exécution et donc connaître ou intervenir pour optimiser.
L'optimisation des requêtes consiste à :

- maximiser le parallélisme entre les entrées/sorties
- minimiser :
  - le nombre d'E/S
  - la taille mémoire nécessaire à l'exécution
  - le temps unité centrale

Les bases Oracle possèdent des paramètres qui limitent le nombre d'options à évaluer pour construire le plan d'exécution d'une requête. Bien définis, ils permettent d'améliorer sensiblement les performances générales de la base. [13]

III.4.1. Préparation de l'instruction SQL

Lorsqu'une instruction SQL entre dans la mémoire cache de la bibliothèque Oracle, les étapes suivantes se succèdent avant que l'instruction ne soit prête à être exécutée :

- **Contrôle de la syntaxe** : le système vérifie que l'instruction SQL est correcte en termes d'orthographe et d'ordre des mots.
- **Analyse sémantique** : Oracle vérifie toutes les tables et les noms de colonnes ainsi que les privilèges accordés par rapport au dictionnaire de données.
- **Vérification du schéma stocké (stored outline)** : Oracle vérifie dans le dictionnaire des données s'il existe un schéma stocké pour l'instruction SQL.
- **Génération du plan d'exécution** : Oracle utilise des algorithmes et des statistiques d'optimisation basés sur le coût dans le dictionnaire de données pour déterminer le plan d'exécution optimal.
- **Création du code binaire**: Oracle génère un exécutable binaire en fonction du plan d'exécution.

III.4.2. Le plan d'exécution

Le plan d'accès aux données accompagné des opérations permettant de les manipuler forment le plan d'exécution de l'ordre SQL. Une fois généré par l'optimiseur (un module particulier du code Oracle), ce dernier est stocké dans le cache LIBRARY. Par la suite, l'optimiseur détermine le meilleur plan pour chaque requête.

A. Optimiseur

Le module optimiseur du serveur Oracle peut opérer sous deux modes (exclusifs) : le mode REGLE ou le mode COUT (respectivement RBO et CBO).

- **Le Mode REGLE** (RBO, Rule Based Optimizer) est basé sur des règles, ce mode de fonctionnement n'utilise pas les statistiques des tables et des index.
- **Le mode COUT** (CBO, Cost Based Optimizer) est basé sur l'estimation des coûts d'exécution des opérations des plans d'exécution. Pour une requête donnée, le SGBDR établit plusieurs plans d'exécution possibles, et le CBO estime pour chacun d'eux le coût d'exécution et choisit le moins élevé. [5]

B. Paramètres de l'optimiseur

L'optimiseur peut également être influencé d'une manière globale par les paramètres suivants :

1) OPTIMIZER_GOAL = CHOOSE : Sélectionne le mode RBO si aucune statistique n'est présente ou le mode CBO si des statistiques, sur au moins une table, existent.
2) OPTIMIZER_GOAL = RULE : Force le mode RBO même si des statistiques existent.
3) OPTIMIZER_GOAL = ALL_ROWS : Passe en mode CBO et s'arrange pour optimiser les opérations de lectures de beaucoup d'enregistrements (BATCH). Typiquement, des accès du type MERGE-JOINS sont préférés.
4) OPTIMIZER_GOAL = FIRST_ROWS : Passe en mode CBO et optimise le temps pour retourner le premier enregistrement (NESTED LOOPS par exemple). [14]

Malgré toutes ces règles, l'optimiseur peut ne pas choisir le plan d'exécution optimal. Une méthode avancée consiste à utiliser des ordres appelés **hint** qui se placent dans la requête sous forme de commentaires afin d'obtenir les meilleures performances possibles :

C. Hints

Sont des conseils sous forme d'instructions que nous pouvons insérer dans nos ordres SQL pour influencer l'optimiseur. Ils peuvent être indiqués dans des SELECT, INSERT et UPDATE, et spécifiés juste après un commentaire avec le signe '+' : /*+ ou --+.

Exemple/syntaxe générique :

SELECT /_+ LE HINT*/colonne (s) FROM table (s) WHERE . . .

Les HINTS les plus couramment utilisés sont :

- **RULE** : optimisation syntaxique
- **FULL** (table) : Parcours de toute la table (sans utiliser l'index)
- **INDEX** (table, index) : Force l'utilisation de l'index de la table
- **NO_INDEX** (table, index) : Désactivation de l'index de la table
- **ORDERED** : Force l'ordre de jointure des tables, telles qu'elles apparaissent dans la clause From
- **FIRST_ROWS** : Force Oracle à choisir le plan d'exécution qui retourne les n premières lignes de manière la plus efficace
- **PARALLEL** (table, n) : Spécification des n serveurs concurrent pouvant être utilisés pour une requête etc. [15]

III.4.3. L'accès aux données

L'accès aux enregistrements d'une table peut s'effectuer :

- directement dans la table relativement à l'adresse physique (identification du fichier, bloc dans le fichier et ligne dans le bloc) de la ligne (ROWID).
- par un parcours d'un cluster indexé (cluster scan) où les lignes ayant même valeur pour la clé du cluster sont stockées dans le même bloc.
- par un parcours d'un cluster par hachage (hash scan) où les lignes ayant même valeur de hachage sont stockées dans le même bloc.
- par un parcours d'un index (index scan) pour retrouver les lignes grâce à une ou plusieurs colonnes de l'index, l'index peut être unique (le parcours de la table d'index retourne une seule valeur ROWID) ou multiple (le parcours de la table d'index retourne une ou plusieurs valeurs ROWID).
- par un parcours séquentiel de toutes les données directement recherchées dans la table (full table scan)

L'expérience montre que 80% des problèmes de performances, sont résolus, par une optimisation des requêtes SQL. [15]

III.5. Conclusion

La section précédente consistait à une présentation des grands axes d'optimisation de base de données et quelques méthodes et outils employés avec leurs cadres d'utilisation. Dans ce qui suit, nous allons appliquer les différentes méthodes d'optimisation, vues précédemment, pour démontrer leurs impacts sur les performances.

Chapitre IV

Etude comparative

IV.1. Introduction

Dans ce chapitre nous allons aborder la partie pratique de notre projet, pour cela nous allons créer trois bases de données avec différents schémas en utilisant l'outil d'Oracle 10g professionnel « Oracle Manager Entreprise », puis ensuite nous allons appliquer quelques méthodes sur les trois bases pour obtenir des résultats permettant de faire une comparaison selon les pistes d'optimisation établis dans les chapitres précédents.

IV.2. Présentation de la base de données et les différents schémas proposés

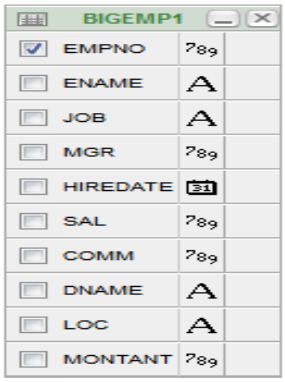
La base sur laquelle on va faire les différents tests, est une base d'employés contenant des renseignements sur ces derniers, les départements d'affectation, ainsi que les montants d'augmentations des salaires.

EMP (EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPNO, DNAME, LOC, MONTANT)

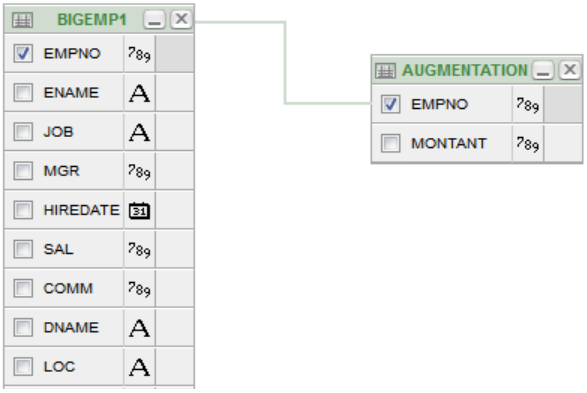
EMPNO : numéro d'employé, ENOM : nom de l'employé, JOB : profession, MGR *:* numéro d'employé responsable direct, HIREDATE : date d'embauche, SAL : salaire, COMM : commission (un employé peut ne pas avoir de commission), DEPNO : numéro de département auquel est affecté l'employé, DNAME : nom du département, LOC : la localisation du département, MONTANT : le montant d'augmentation que peut avoir un employé.

L'étude consiste à comparer entre les résultats obtenus par trois bases de données avec des schémas conceptuels différents :

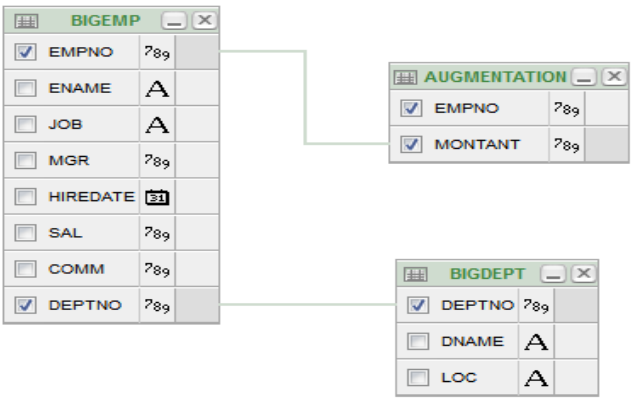
- Le premier schéma : la base de données est constituée d'une seule table contenant tous les attributs de la base, en utilisant les types **number** et **varchar2**.



– Le deuxième schéma : la base de données contenant deux tables Employé et Augmentation en utilisant le type **char** pour les champs ENAME, JOB, DNAME, LOC et **integer** pour les champs EMPNO et MGR.



– Le troisième schéma : On applique la notion de normalisation avec précaution dans le typage. Le schéma relationnel de la base de données obtenu se compose de trois tables.



IV.3. Installation du SGBD et création des bases de données

IV.3.1. Installation d'Oracle professionnel 10g

Oracle Enterprise Manager est un ensemble d'outils utilisant une interface graphique pour simplifier la gestion des différents objets de la base de données. Parmi les outils les plus utilisés **O**racle **E**nterprise **M**anager **D**atabase **C**ontrol**,** une application installée en local sur chaque serveur de base de données qui fournit une interface Web centralisée permettant de gérer tout l'environnement Oracle de l'entreprise.

Sous Windows, l'installation doit se faire par un utilisateur du groupe administrateurs système. Ce compte sera ajouté à un groupe système **ORA_DBA** créé automatiquement durant l'installation d'Oracle Database.

***Etape 1*:** Lancement du "D:\Oracle10g\OracleServeur\db\Disk1\setup.exe" pour démarrer "**Oracle Universal Installer**"

Choix de l'option "**Installation avancée**" ainsi que le mot de passe dans l'écran qui suit, les autres valeurs choisis par défaut:

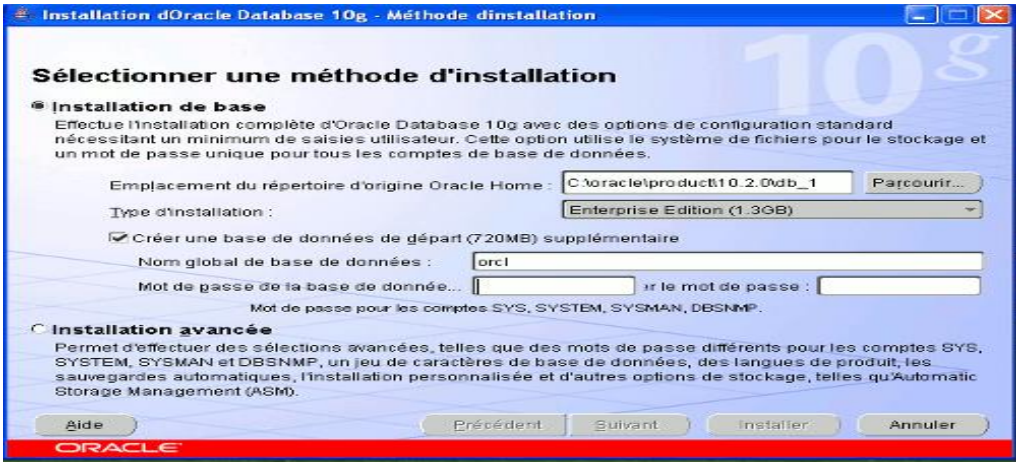


Figure IV.1 : *Méthode d'installation d'Oracle 10g*

En cliquant sur suivant, Oracle va procéder à une vérification des prérequis :

Figure IV.2 : *vérifications de prérequis propre au produit*

Etape 2 : Un écran résumant les paramètres d'installation est affiché :

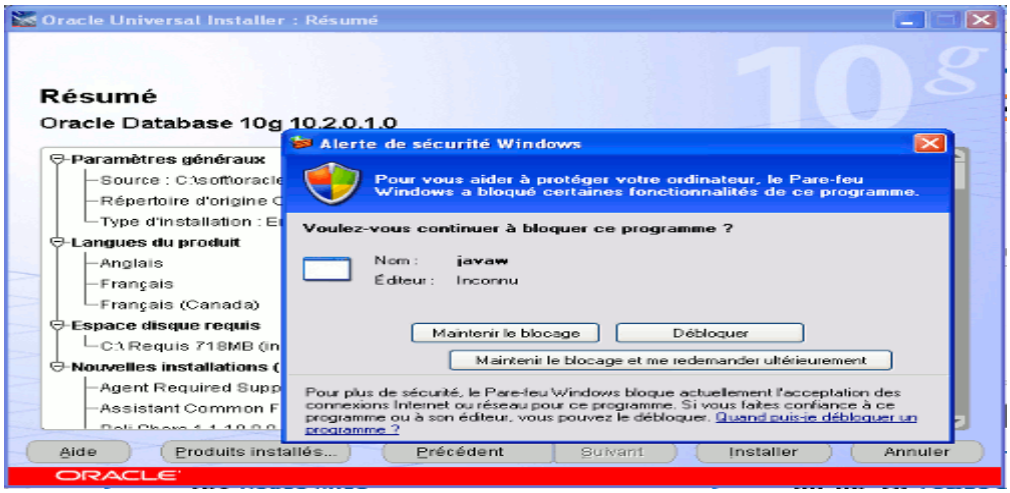


Figure IV.3 : *les paramètres d'installation d'Oracle 10g*

Autorisation des composants java à s'exécuter en cliquant sur "**Débloquer**" puis sur "Installer".

***Etape 3*:** A ce niveau l'installation proprement dite est démarrée :

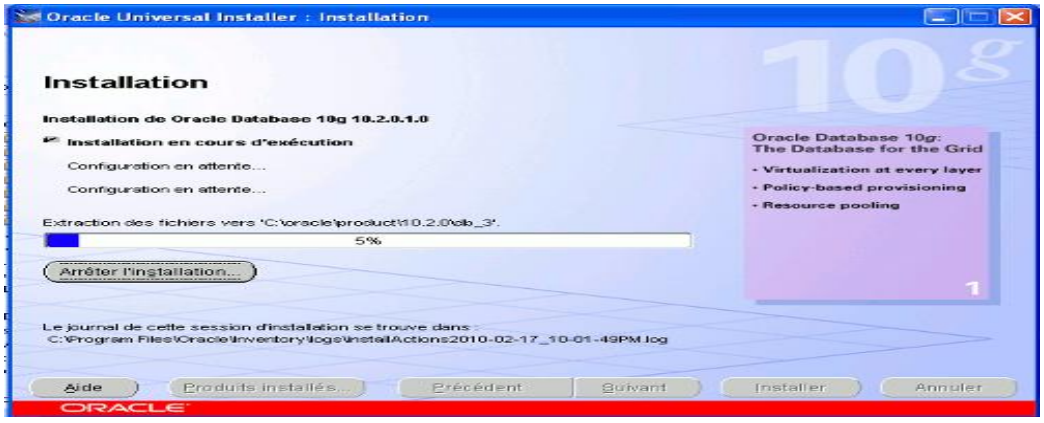


Figure IV.4 : *Démarrage de L'installation*

A la fin, l'assistant de configuration affiche le statut "Succès" :

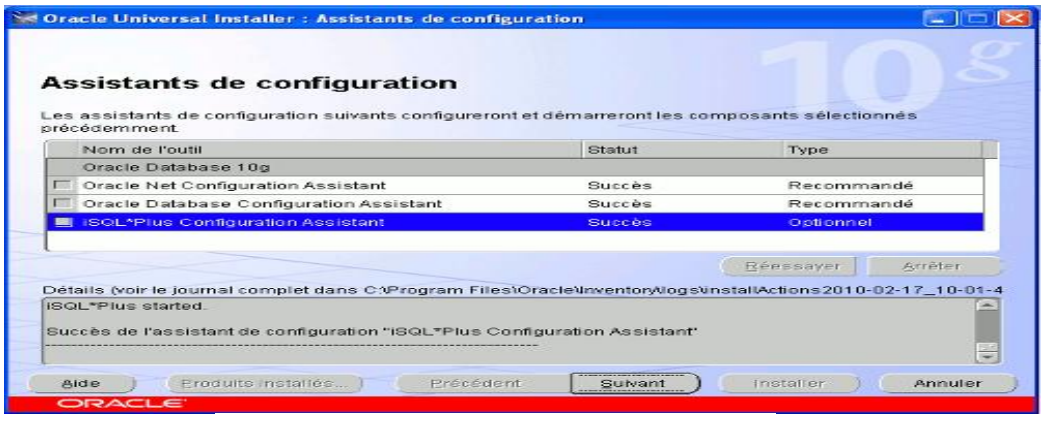


Figure IV.5 : *Assistant de configuration*

IV.3.2. Création et configuration d'une base de données

Pour créer une nouvelle base de données, l'utilitaire Oracle Database Configuration Assistant est accessible depuis le menu ***Démarrer > Programmes > Oracle ï OraDb10g home1 > Outils de Conýguration et de migration > Assistant Configuration de base de donnée.***

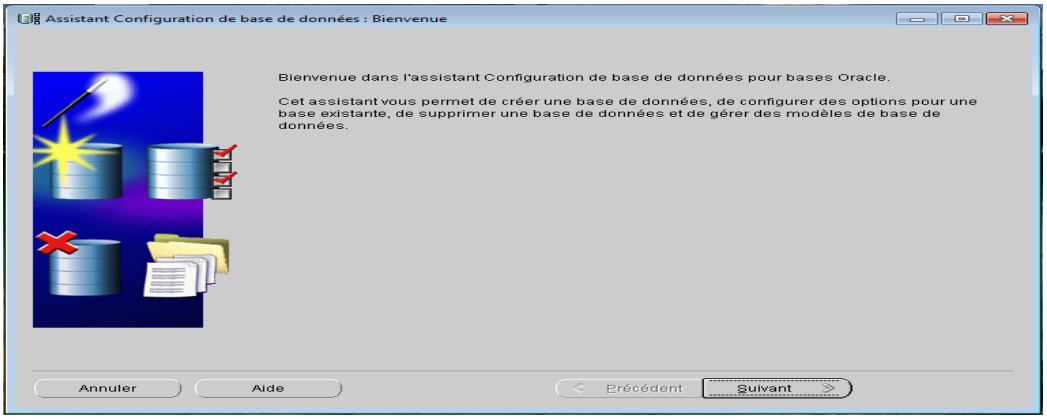


Figure IV.6 : *Écran d'accueil d'Oracle Database Configuration Assistant*

Plusieurs options sont proposées, nous choisissons la création d'une nouvelle base Oracle 10g qui passe par 12 phases. Nous laissons les réglages par défaut durant tous les étapes. A la fin de cet assistant, la création de notre base se lance.

Etapes de création de base de données :

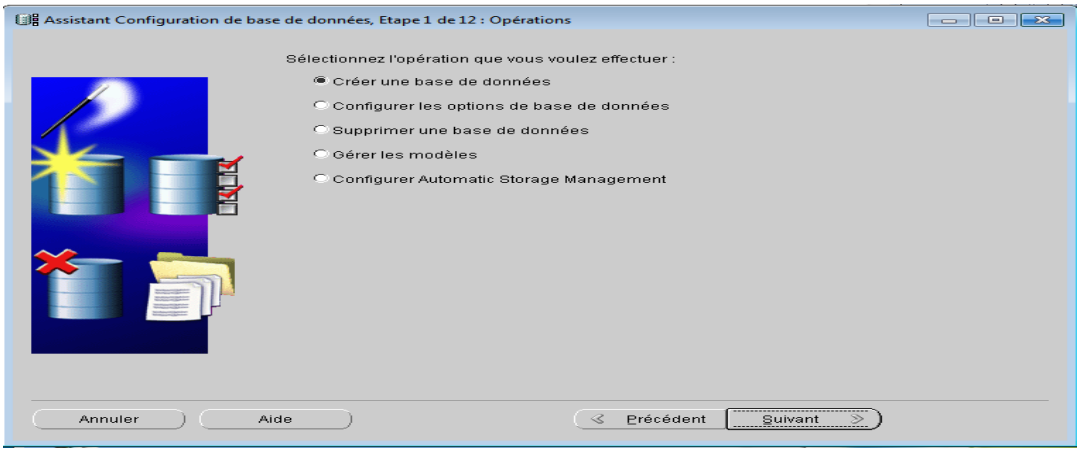


Figure IV.7 : *1^{ére} étape de création de BDD*

Plusieurs options sont proposées : Créer une nouvelle base de données, Configurer une base de données existante, comme on peut aussi Supprimer.

On va dérouler toutes les étapes puis cliquer sur quitter pour terminer et valider la création.

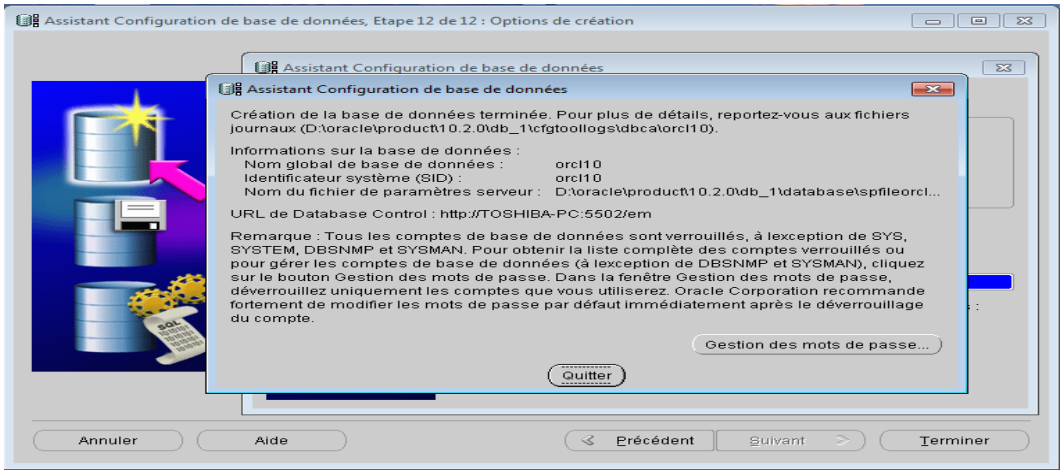


Figure IV.8 : *Fin de création de base*

De la même manière, nous avons créé les trois bases : orcl1, orcl2 et orcl3 qui vont subir les différents tests afin d'illustrer les concepts que nous avons présenté dans les chapitres précédents.

IV.3.3. Création des tables

Pour mettre en évidence l'intérêt des optimisations, les bases de données doivent être simples mais de taille importante. Ces bases contiennent de grandes tables fondées sur des petites tables mais dupliquées 100 000 fois.

La connexion aux bases de données est faite par l'utilisateur « SYSTEM » avec les mots de passe correspondants choisis lors de la création (« orcl1 » pour la 1^{ére} base, « orcl2 » pour la 2^{éme}, « orcl3 » pour la 3^{éme}) par le biais de l'outil SQL*Plus qui est accessible depuis le menu ***Démarrer > Programmes > Oracle ï OraDb10g home1 > Développement dâapplication > SQL Plus.***

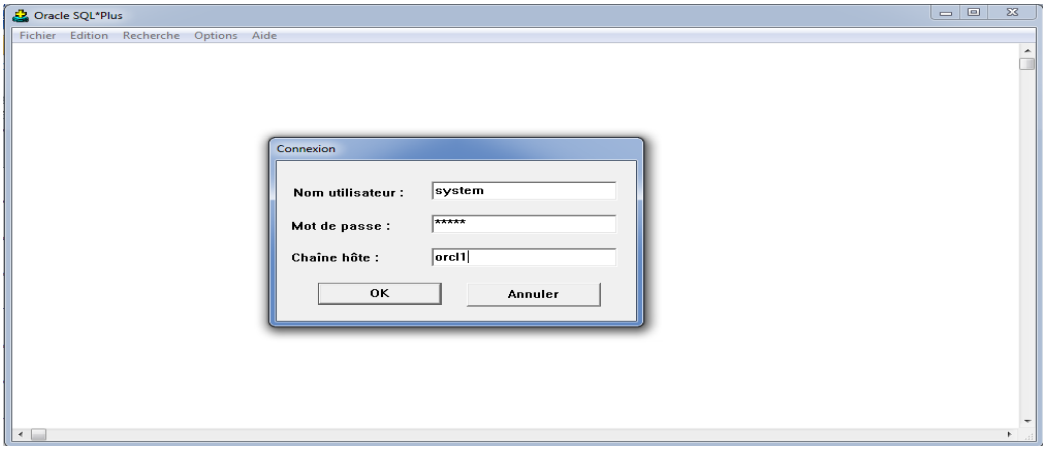


Figure IV.9 : *Connexion de la base avec SQL-PLUS*

Pour construire les petites tables nous avons utilisé le script suivant :

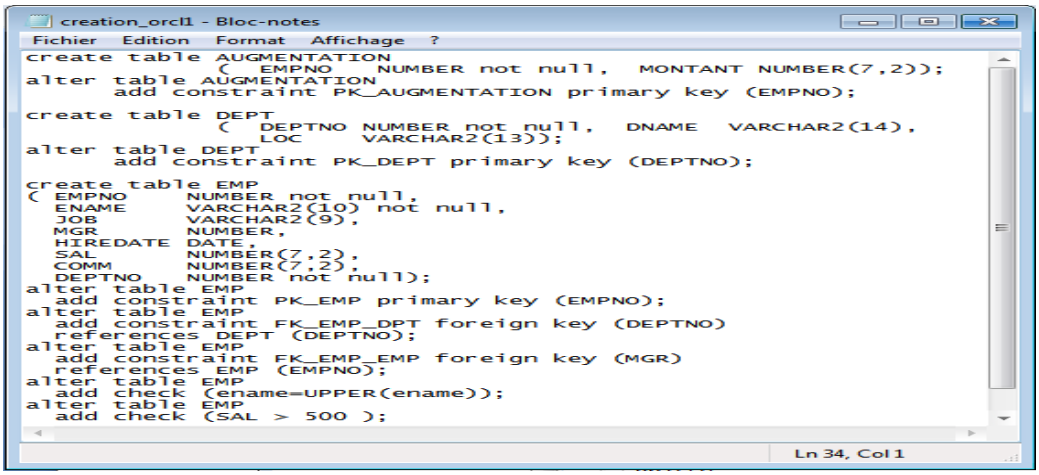


Figure IV.10 : *Le script de création des tables d'ORCL1*

Puis nous avons créé et remplit les grandes tables à l'aide de script suivant [5] :

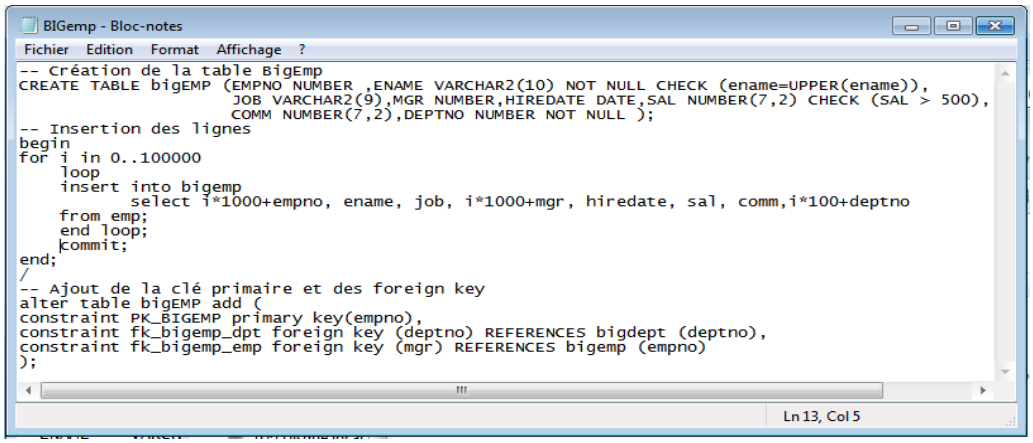


Figure IV.11 : *Script de remplissage des grandes tables*

De la même manière, nous avons procéder à la création des autres bases de données en respectant les schémas préalablement conçus.

IV.4. Résultats expérimentaux

IV.4.1. Choix des mesures de performance

Pour mesurer les améliorations apportées par les différentes méthodes d'optimisation, il est nécessaire de fixer les indicateurs et les critères à évaluer. Les plus importants critères utilisés sont :

- Le temps de réponse
- La consommation mémoire
- La consommation CPU
- Le nombre d'Entrées /Sorties disque

IV.4.2. Méthodes et outils utilisées pour collecte d'information

Plusieurs statistiques fournissent des indications sur le taux de charge de la base de données ainsi que sur les ressources utilisées. Beaucoup de méthodes sont utilisées pour mesurer la consommation de ces ressources, nous allons présenter essentiellement celles qu'on a appliquées.

A. Consultation des plans d'exécution

Consiste à afficher les statistiques relatives à l'exécution d'une même requête exécutée sur les trois bases de données, nous obtenons les résultats selon le plan d'exécution utilisé. L'optimiseur basé sur les coûts (CBO) prend en considération les influences externes lorsqu'il détermine le meilleur plan d'exécution d'une requête SQL. Puisqu'un ensemble de processus d'arrières plans sont exécutés dès la création de l'instance de la base de données, le CBO est capable de prendre en compte les coûts des E/S disques externes et des cycles CPU pour chaque opération SQL.

Calculons d'abord Le temps de réponse qui est l'indicateur le plus ressenti par l'utilisateur, c'est la mesure la plus pratique, il est obtenu par la commande ***set timing on;***

Prenant exemple de requête : La liste des employés avec leurs job, département, et le montant d'augmentation ne dépassant pas 2000 ;

En activant le mode AUTOTRACE, nous avons obtenus les plans d'exécutions suivants :

ORCL1

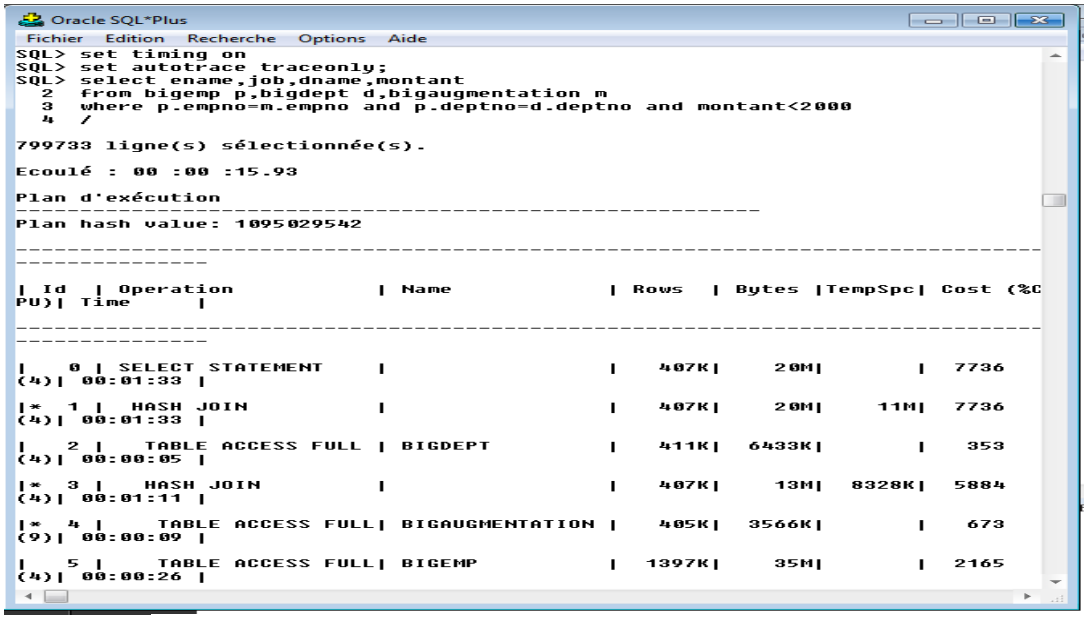


Figure IV.12 : *Plan d'exécution d'une requête de la base ORCL1*

Lecture du Plan dœexécution

Après la dernière ligne de la requête, on obtient le nombre de lignes sélectionnées, puis le plan d'exécution contenant les valeurs estimées. On peut lire ce plan de la manière suivante :

- L'optimiseur a fait double jointure sous forme de double hachage avec une allocation de place temporaire allouée par le système pour les jointures par hachage de 19M représenté par la colonne TempSpc.
- Les opérations 4 et 5 font l'extraction des lignes des tables BIGEMP et BIGAUGMENTATION en utilisant un balayage de table complet, après l'optimiseur fait une jointure par hachage entre les deux tables précédentes.
- L'opération 2 signifie l'extraction des lignes de la table BIGDEPT en utilisant un balayage de table complet, puis l'optimiseur fait une 2^{éme} jointure par hachage entre cette table et le résultat de la 1^{ére} jointure.
- L'optimiseur termine par l'opération 0, une sélection des attributs.

ORCL2

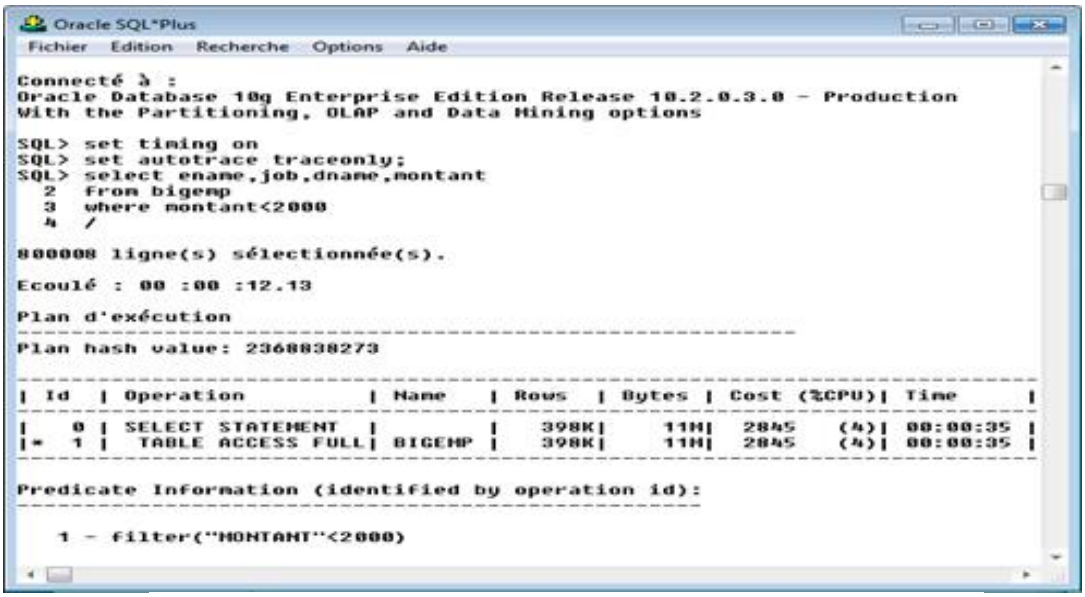


Figure IV.13 : *Plan d'exécution d'une requête de la base ORCL2*

Lecture du Plan dœexécution

- L'optimiseur extrait toutes les lignes de la table BIGEMP en utilisant un balayage de table complet, après il réalise une opération de sélection, ces deux opérations consomme 22M de la RAM.

ORCL3

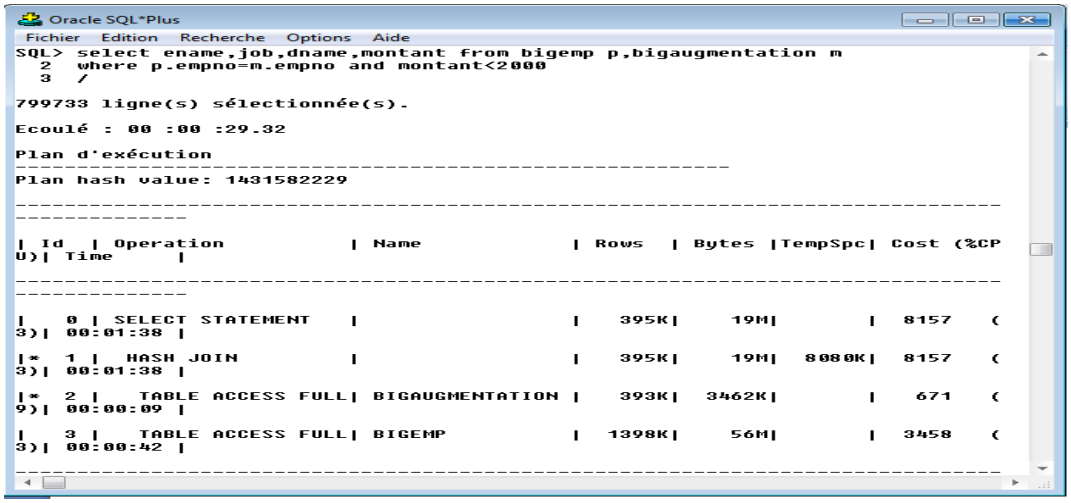


Figure IV.14 : *Plan d'exécution d'une requête de la base ORCL3*

Lecture du Plan dœexécution

L'optimiseur extrait les données par un balayage de table complet (table BIGEMP opération (3)), puis un balayage de table complet (table BIGAUGMENTATION opération (2)), puis, il réalise une jointure par hachage qui consomme 19K de la RAM (opération(1)), il termine par une opération de sélection.

B. Utilisation des vues

Une deuxième méthode consiste à récupérer les statistiques à partir des vues systèmes. Ces dernières sont utilisées pour obtenir les coûts de CPU et I/O disque.

Voici les formules suivantes :

cost = io_cost + cpu_cost

cpu_cost = cpu_cycles / (cpuspeed * sreadtm)

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

cpuspeed = cpuspeedNW

sreadtm = ioseektm + db_block_size / iotfrspeed

- ioseektime : temps de recherche d'un bloc sur le disque en ms, valeur par défaut 10
- iotfrspeed : la vitesse de transfert de données en bytes par ms, valeur par défaut 4096
- cpuspeedNW : le nombre moyen de cycles CPU par seconde, valeur par défaut qui dépend du système.

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

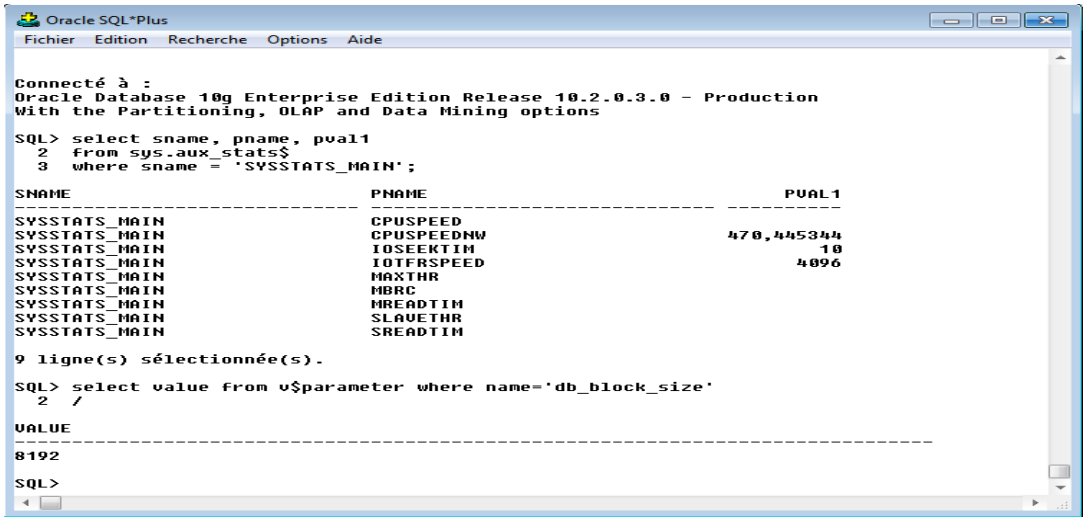


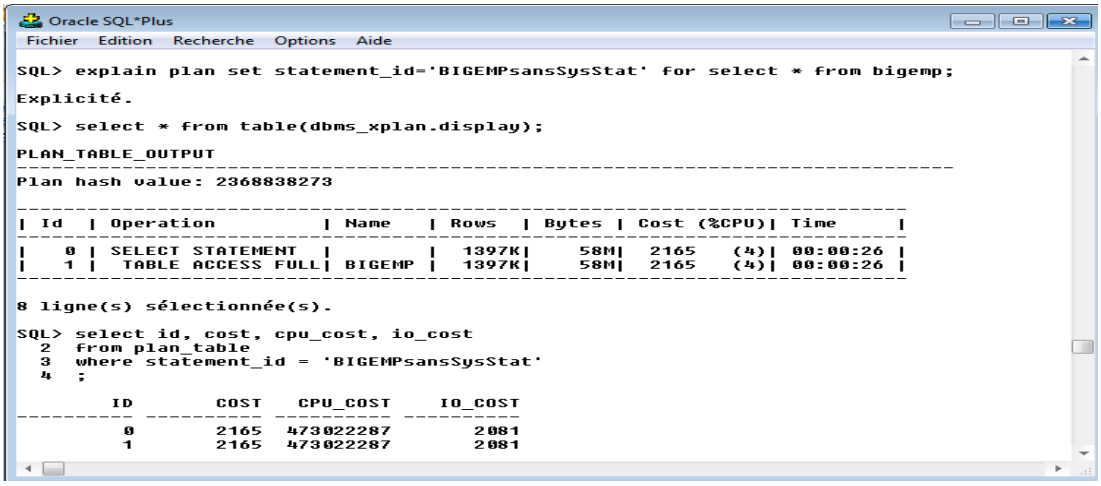
Figure IV.15 : *Statistiques obtenues par les vues*

`sreadtm = ioseektm + db_block_size / iotfrspeed`

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

Pour avoir le coût I/O et le coût de CPU nous exécutons les requêtes suivantes :

ORCL1



FigureIV.16 : *Coût CPU / IO obtenus par les vues d'ORCL1*

cpu_cost = cpu_cycles / (cpuspeed * sreadtm)

= 473022287 / (470445.344 * 12) = 83.784 ≈ **84**

io_cost = **2081**

cost = io_cost + cpu_cost

= 2081 + 84 = **2165**

ORCL2

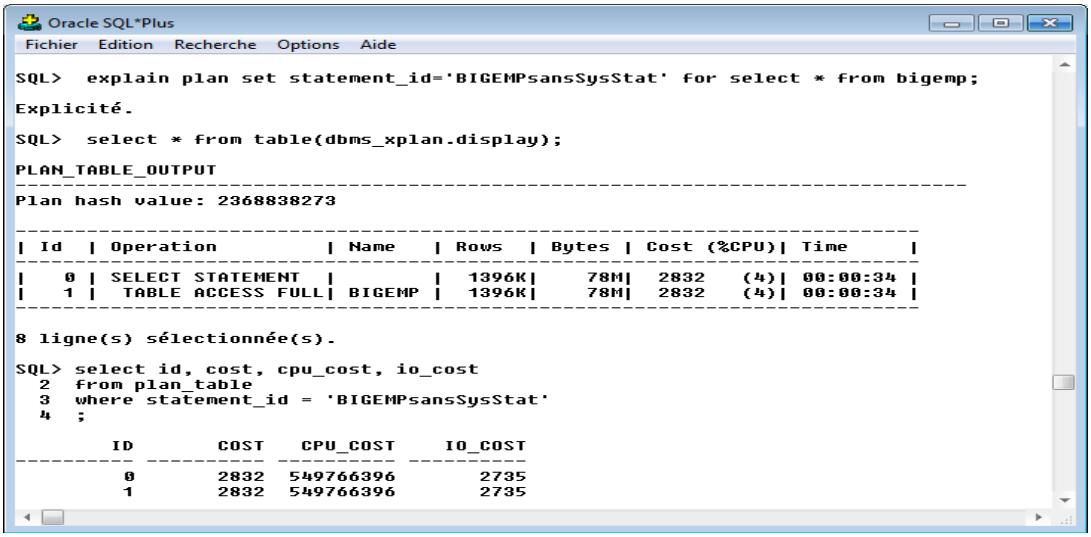


Figure IV.17 *: Coût CPU / IO obtenus par les vues d'ORCL2*

cpu_cost = cpu_cycles / (cpuspeed * sreadtm)

= 549766396 / (470445.344 * 12) = 97.348 ≈ **97**

io_cost = **2735**

cost = io_cost + cpu_cost

= 2735 + 97 = **2832**

ORCL3

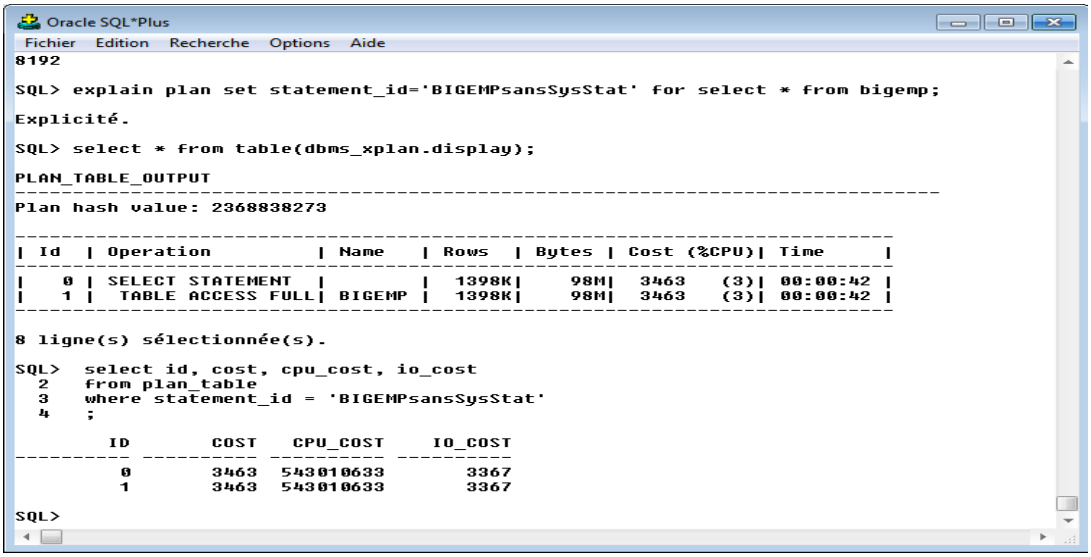


Figure IV.18 : Coût CPU / IO obtenus par les vues d'ORCL3

cpu_cost = cpu_cycles / (cpuspeed * sreadtm)

= 543010633 / (470445.344 * 12) = 96.187 ≈ **96**

io_cost = **3367**

cost = io_cost + cpu_cost

= 3367 + 96 = **3463**

C. Compteurs de performances Windows

Au travers du moniteur de performance sous Windows, nous pouvons consulter des compteurs de performances qui nous permettent d'accéder facilement à un ensemble de paramètres aidant à la surveillance de l'usage des ressources matérielles de notre serveur.

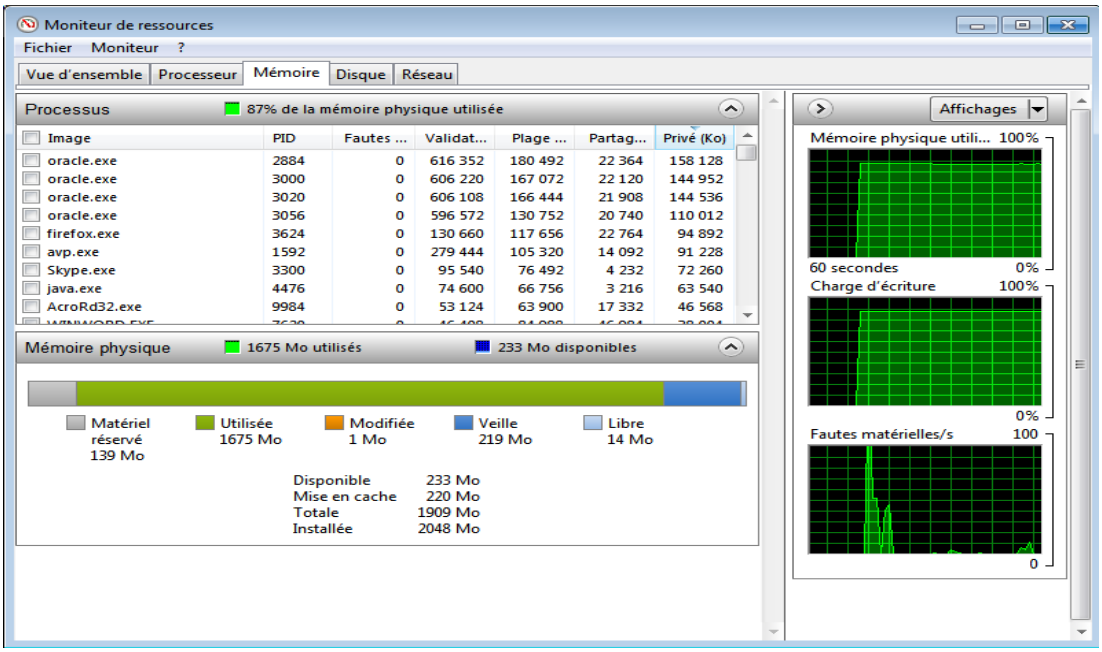


Figure IV.19 : *Moniteur de ressources de performance sous Windows (**Mémoire**)*

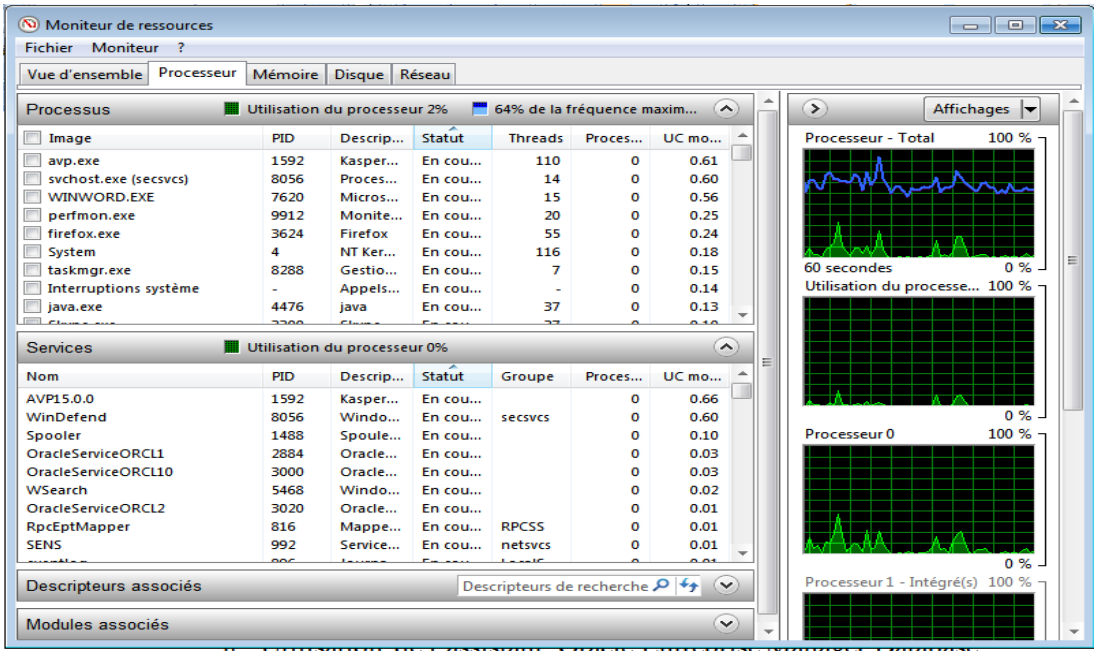


Figure IV.20 : *Moniteur de ressources de performance sous Windows(**CPU**)*

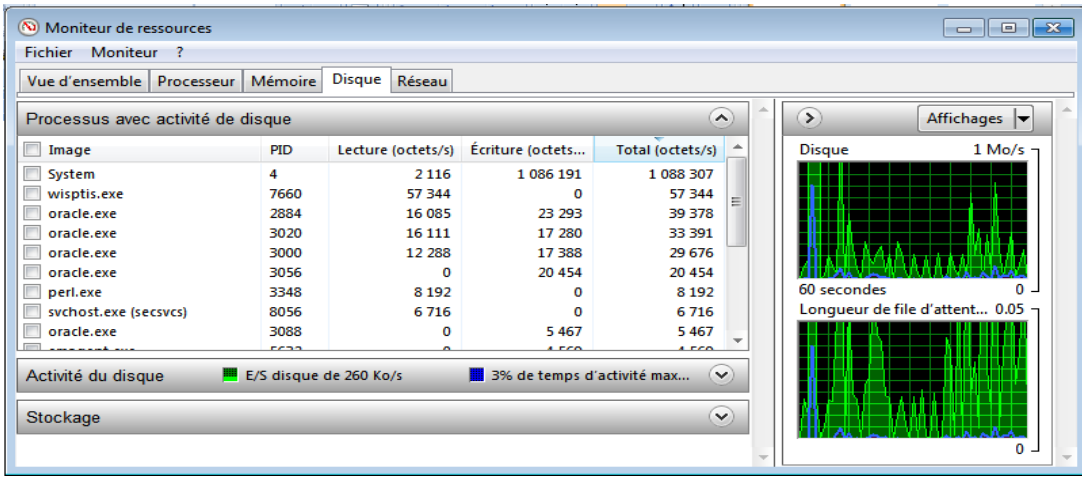


Figure IV.21 : *Moniteur de ressources de performance sous Windows(**E/S Disque**)*

D. Utilisation de l'assistant Oracle Entreprise Manager Database Control (OEMDC)

Une autre méthode consiste à utiliser L'outil Oracle OEMDC qui complète les observations effectués au niveau du système d'exploitation avec les pages relatives à l'activité de l'instance en se connectant à la base de données depuis ***OraDb10g home1***

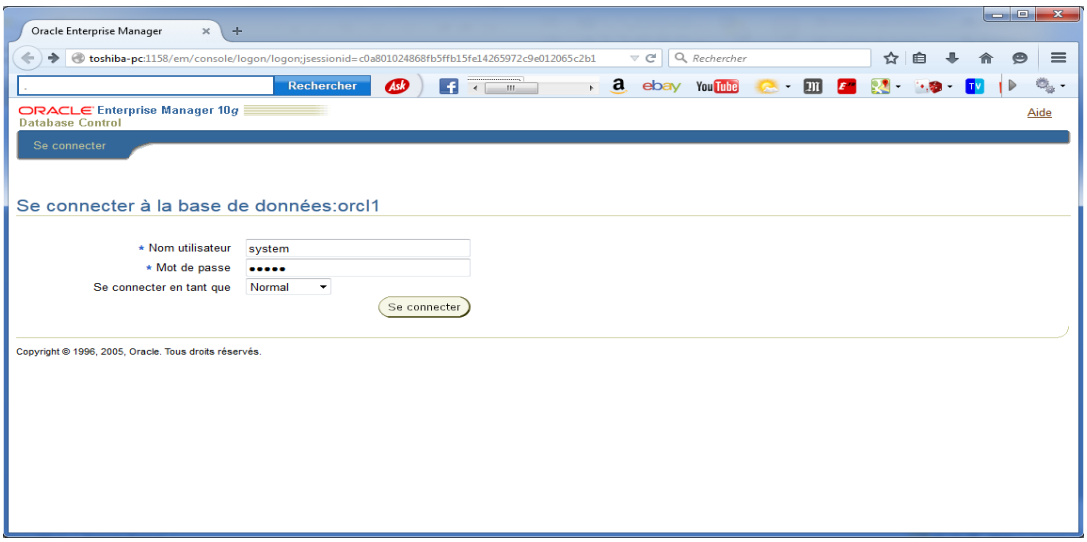


Figure IV.22 : *Connexion à la base de données sous OEMDBC*

Sur l'écran suivant nous avons la page hôte de l'UC qui fournit des graphiques de synthèses :

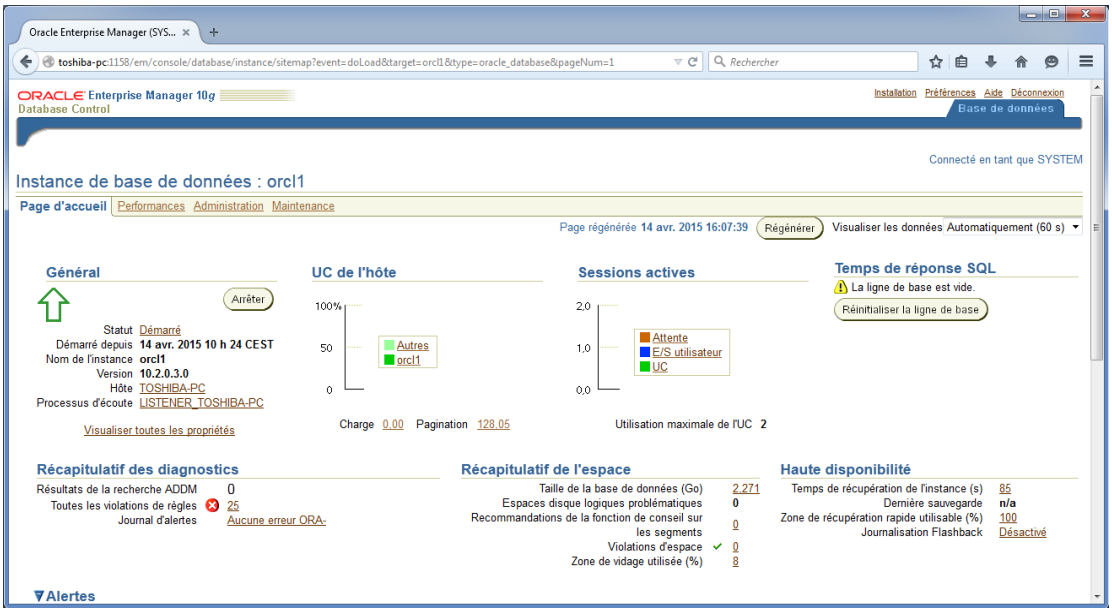


Figure IV.23 : *Page d'Accueil d'OEMDBC*

Sur l'onglet **performance** de la base de données dans l'assistant Oracle Entreprise Manager, nous obtenons L'écran qui permet de visualiser l'utilisation de l'UC, la mémoire et les E/S disques.

ORCL1

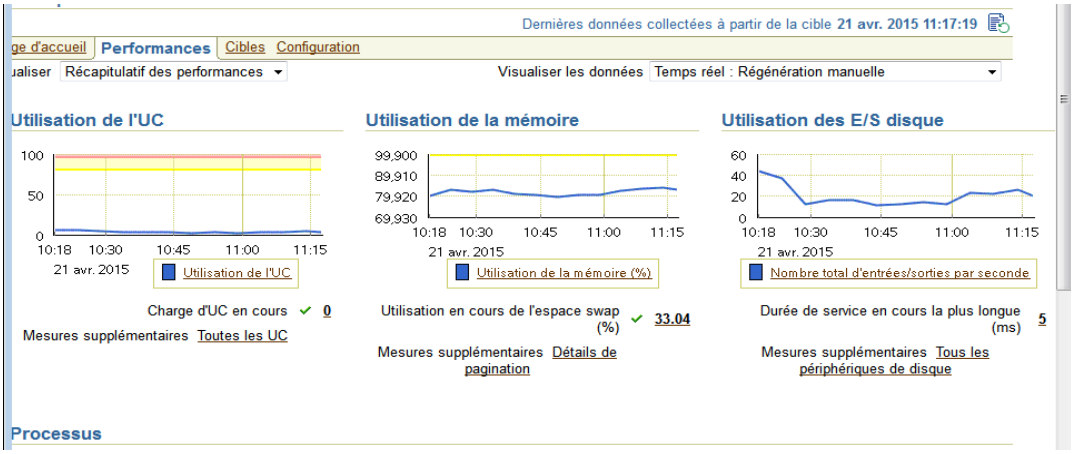


Figure IV.24 : *Les courbes d'utilisation d'UC, RAM, E/S d'ORCL1*

ORCL 2

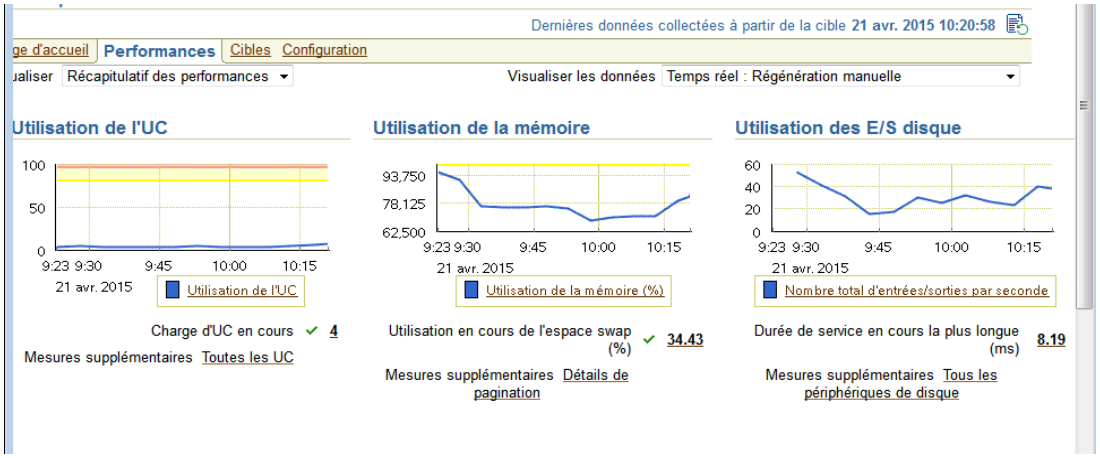


Figure IV.25 : *Les courbes d'utilisation d'UC, RAM, E/S d'ORCL2*

ORCL 3

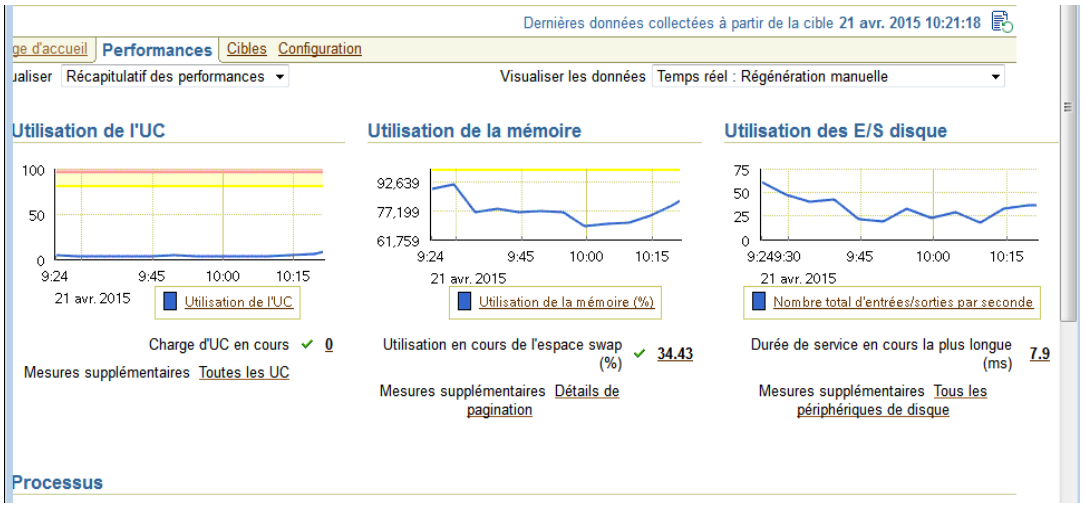


Figure IV.26 : *Les courbes d'utilisation d'UC, RAM, E/S d'ORCL3*

IV.4.3. Comparaison des résultats des différents schémas

A. Présentation des résultats

Paramètres mesurées	***Orcl1***	***Orcl2***	***Orcl3***
Temps Ecoulé (s)	00 :00 :15.93	00 :00 :12.13	00 :00 :29.32
Stockage de données (octet)	261 084 672	177 209 344	252 706 816
RAM (Mo)	97	22	97
Coût CPU	84	97	96
Coût I/O	2081	2735	3367

Tableau IV.1 : Paramètres mesurées des trois bases

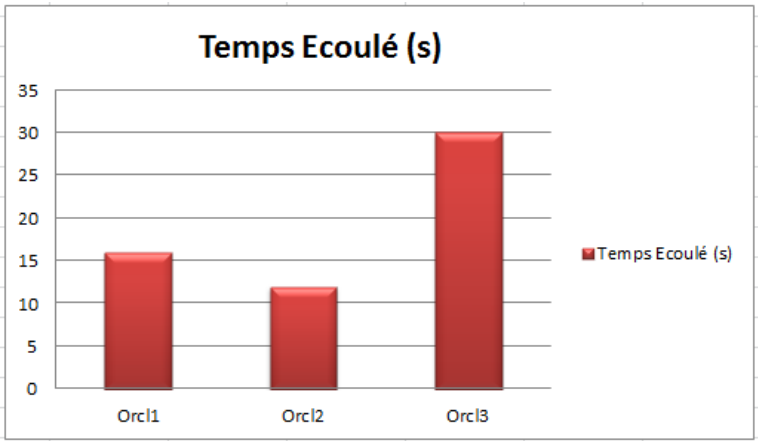


Figure IV.27 : *Histogramme temps écoulés des trois bases*

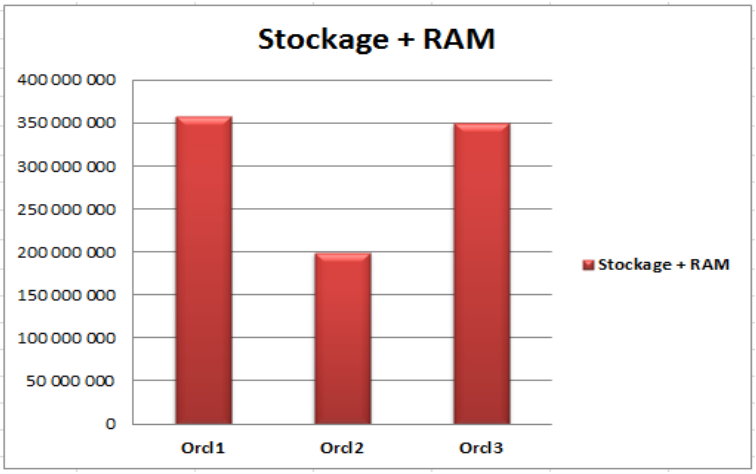


Figure IV.28 : *Histogramme Stockage + RAM (octets) des trois*

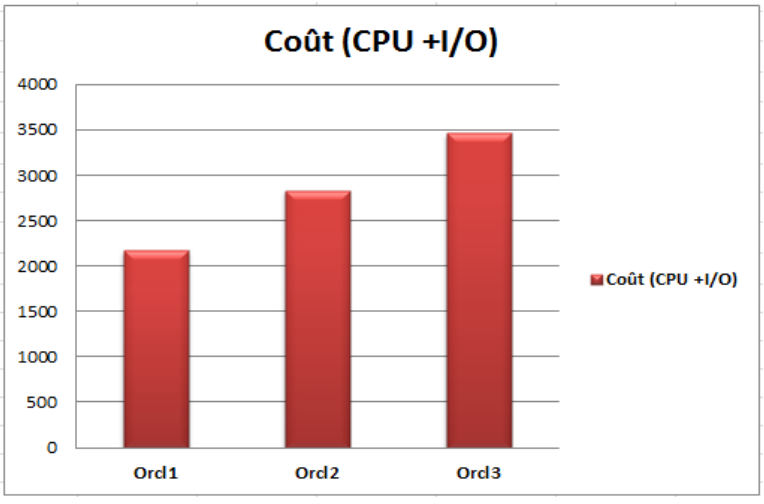


Figure IV.29 : *Histogramme Coût CPU + I/O des trois bases*

B. Analyse et interprétation des résultats

Vu les trois histogrammes **[*Figure IV.27, 28,29*],** nous remarquons que la base ORCL3 est moins performante par rapport aux autres bases, cela justifie que le non-respect des formes normales et le mauvais choix des types de données atténue les performances du système.

Passons à la comparaison d'ORCL1 avec ORCL2 :

Théoriquement, on suppose que mettre toutes les informations dans une même table rendra la base de données plus rapide puisque on n'a pas besoin de jointures, justement c'est ce qui est remarqué sur l'histogramme de temps écoulé **[*FigureIV.27*]**, la plus petite valeur revient à la base de données ORCL2, mais cela n'est pas suffisant pour juger sa performance, car le viol de la notion de normalisation conduira tôt ou tard à des tables obèses. Cependant dès qu'on passe à des grands volumes de données, la base normalisée, constituée de plusieurs tables reprend l'avantage, et plus la base accroîtra, plus l'écart sera important. Et malgré le coût des jointures conduites par le processus de normalisation, qui influe sur la consommation de la RAM, une requête avec plusieurs prédicats ou conditions s'exécutera avec un moindre coût de CPU et d'I/O disque, comme le montre l'histogramme coût CPU+ I/O **[*FigureIV.29*]**.

Comparant maintenant les résultats des graphes d'OEMDC:

- La valeur Charge représente le nombre moyen de processus en attente de l'UC, plus cette valeur est faible plus l'UC est disponible.
- La valeur de la Pagination représente le nombre de pages transférées hors mémoire par seconde, c'est le résultat d'un manque de pages disponibles en mémoire, ORCL1 présente la meilleure valeur.
- Les performances d'un système se dégradent lorsque la mémoire physique devient insuffisante, ce qui est influencé sur l'espace Swap qui doit rester aussi faible que possible. Les **[*Figures IV.24, 25,26*]** Montrent que la valeur de l'utilisation de l'espace Swap pour la base ORCL1 est la plus faible.

Pour en conclure, on peut confirmer que la base de données ORCL1 est la plus optimisée par rapport aux autres.

IV.4.4. Comparaison des résultats des différentes infrastructures matérielles

Dans cette partie, pour réaliser nos séries de tests, nous avons appliqué les mêmes méthodes citées au début du chapitre sur trois différentes infrastructures matérielles, en

utilisant comme base de données, la meilleure parmi les trois comparées dans la partie précédente, à savoir la base ***ORCL1***.

Les infrastructures matérielles sur lesquelles les différents tests sont effectués sont :

PC1 **:** Processeur intel 4 Processeur 2.16 GHz
Mémoire installée (RAM) 2 GO
OS : Windows 7 Titan

PC2 **:** Processeur intel 2 Processeur 1.20 GHz
Mémoire installée (RAM) 3 GO
OS : Windows 7

PC3 **:** Processeur intel 4 Processeur 3.16 GHz
Mémoire installée (RAM) 2 GO
OS : Windows XP

A. Présentation des résultats

Paramètres mesurées	Orcl1_pc1	Orcl1_pc2	Orcl1_pc3
Temps Ecoulé (s)	00 :00 :15.93	00 :00 :37.72	00 :00 :08.73
Stockage de données (octet)	261 084 672	389 382 336	197 482 176
RAM (Mo)	97	94	98
Coût CPU	84	208	78
Coût I/O	2081	5617	2006

Tableau IV.2 : Paramètres mesurées sur les trois machines

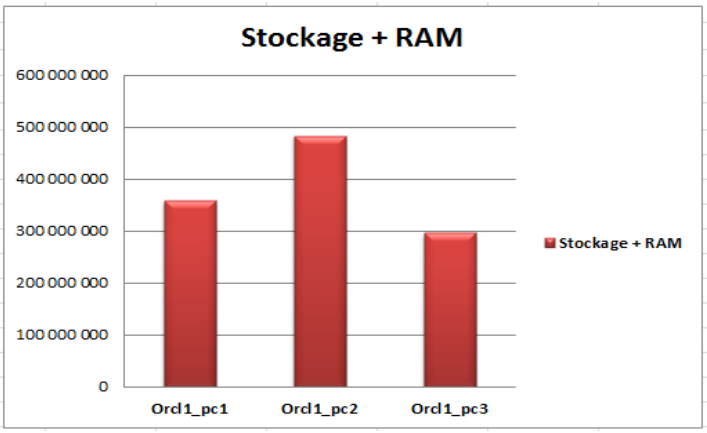


Figure IV.30 : *Histogramme Stockage + RAM (octets) sur les trois machines*

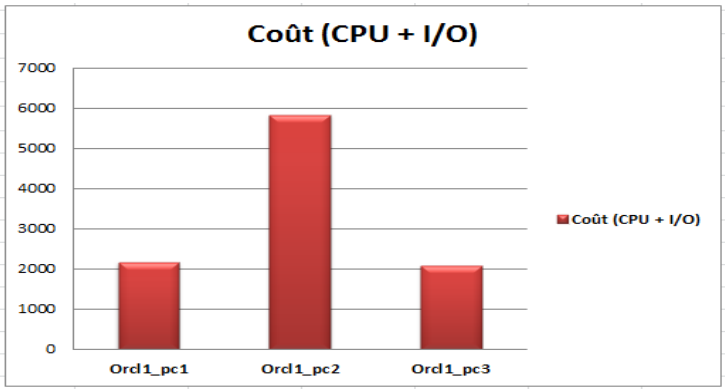


Figure IV.31 : *Histogramme Coût CPU + I/O sur les trois machines*

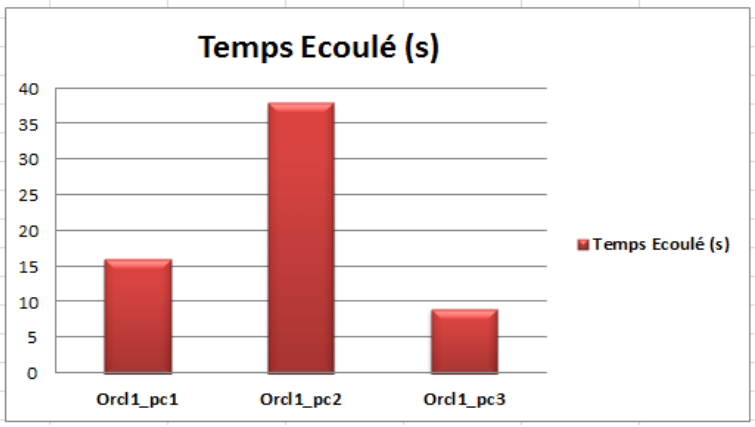


Figure IV.32 : *Histogramme temps écoulé sur les trois machines*

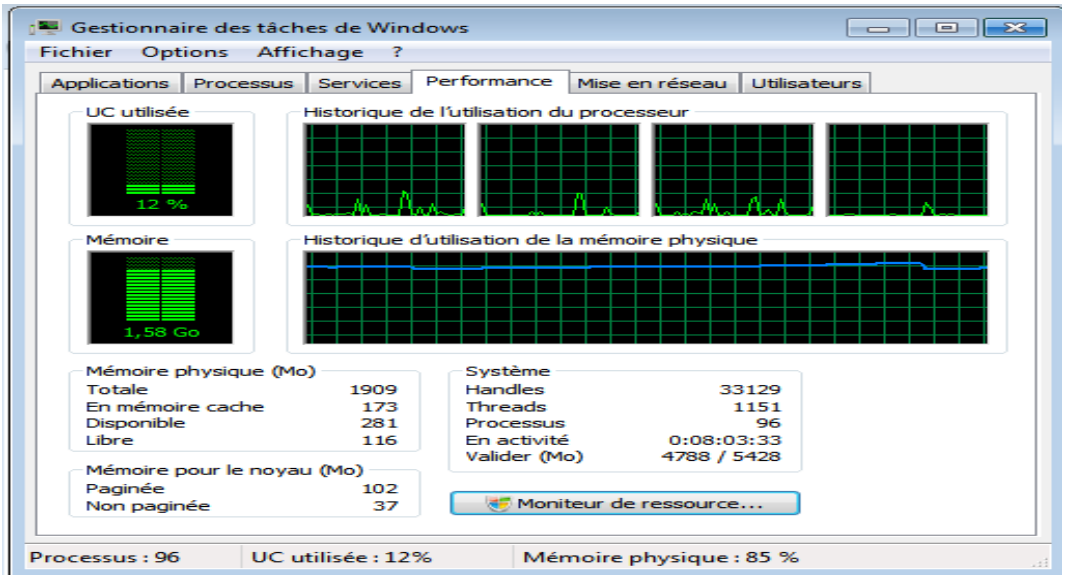


Figure IV.33 : *Gestionnaire des tâches de Windows pc1*

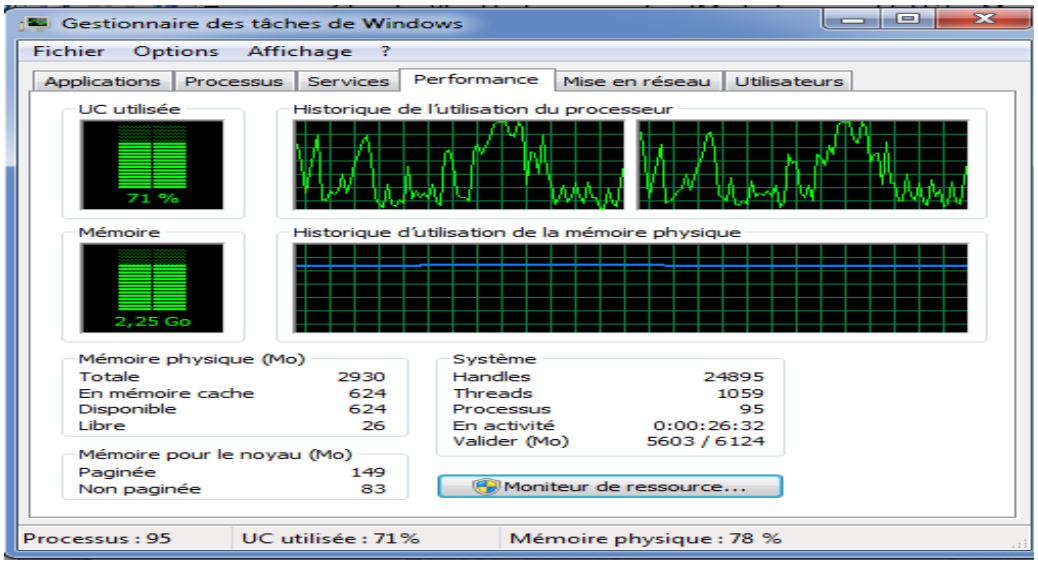


Figure IV.34 : *Gestionnaire des tâches de Windows pc2*

B. Analyse et interprétation des résultats

Le choix des ressources matérielles peut avoir une influence sur le comportement d'une base de données. Les statistiques du système nous éclairent sur l'utilisation de ces ressources ainsi que sur le comportement du système lui-même, grâce à ces informations nous pouvons déterminer le plus efficace parmi plusieurs choix.

Notre test effectué sur trois différentes infrastructures matérielles montre que le meilleur choix est celui du 3ème PC parmi les trois **[*Figure IV.30, 31, 32*]**

Comme Le CPU est un point de passage pour toutes les opérations, son amélioration a toujours un impact positif, ce que nous observons clairement dans la figure [***Figure IV.31***], de même pour l'extension de la RAM qui est souvent une amélioration matérielle efficace.

IV.4.5. Comparaison des résultats des différentes requêtes

A. Présentation des résultats

Nous proposons une requête contenant des jointures avec une sélection de tous les attributs, pour avoir la liste des employés, appartenant au département 'SALES', qui ont eu une augmentation inférieure à 2000 :

Oracle nous a retourné le résultat suivant :

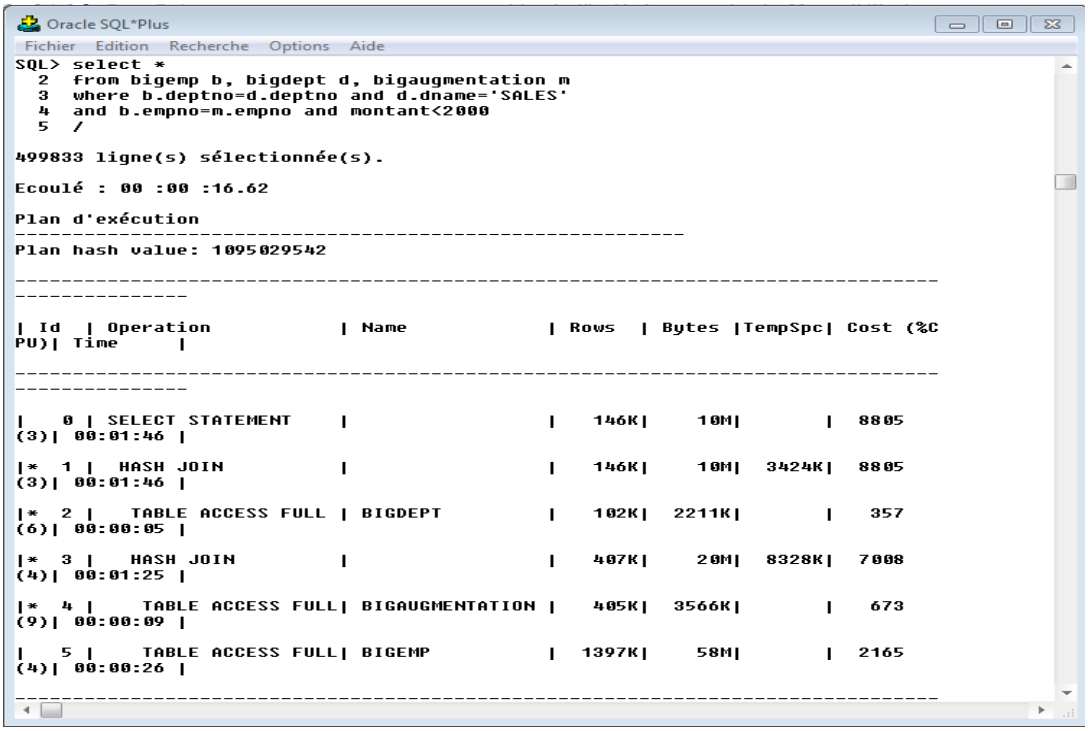


Figure IV.35 : *Plan d'exécution de la 1ère requête*

- Temps de réponse = 16.62 ≈**17 s**
- RAM = **103 Mo**
- Coût (CPU+I/O) = **27813**

Nous avons essayé d'optimiser cette requête en introduisant l'index et en modifiant les clauses **Where** comme suit :

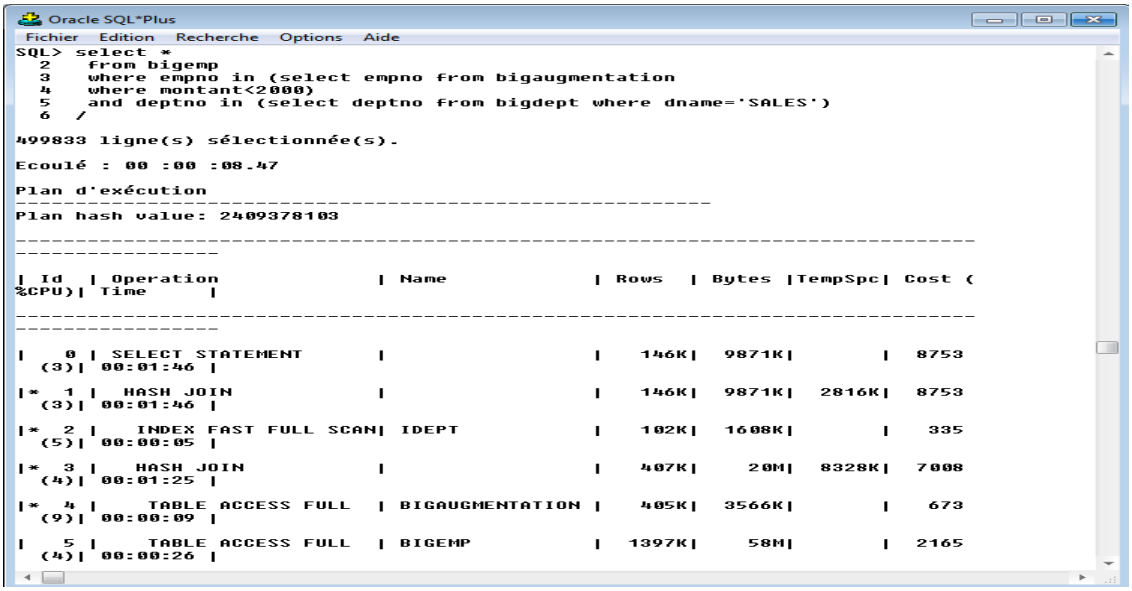


Figure IV.36 : *Plan d'exécution de la 2éme requête*

- Temps de réponse = 08.47 ≈ **8s**
- RAM = **100 Mo**
- Coût (CPU+I/O) = **27687**

Et comme nous n'avons pas besoin de tous les attributs, nous allons sélectionner ceux qui nous intéressent :

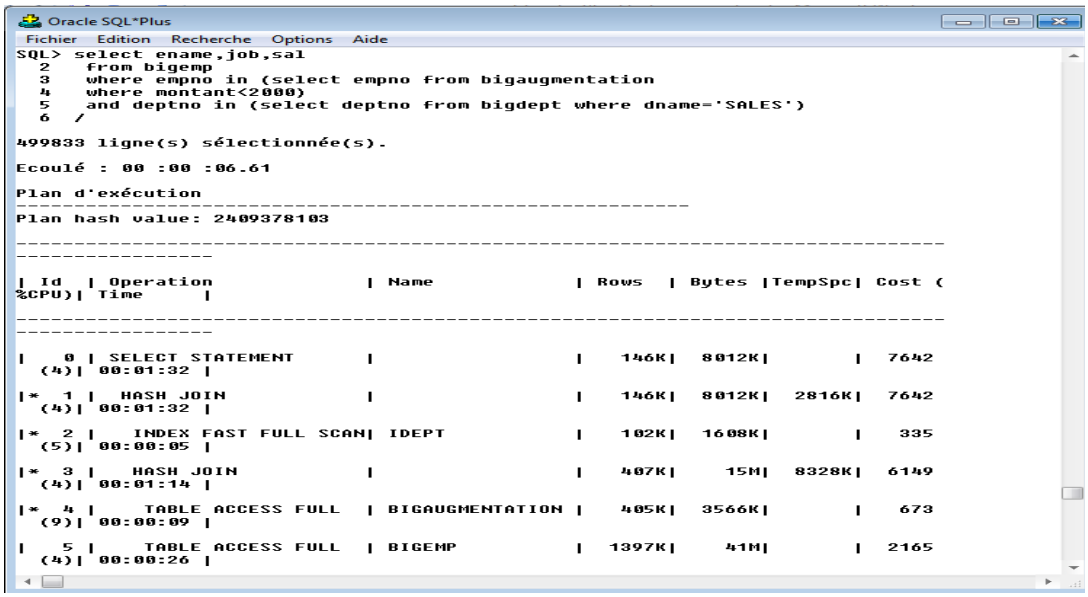


Figure IV.37 : *Plan d'exécution de la 3éme requête*

- Temps de réponse = 06.61 ≈ **7s**
- RAM = **76 Mo**
- Coût (CPU+I/O) = **24606**

Récapitulons les résultats des *trois requêtes* dans ce tableau :

Paramètres	**1ére Requête**	**2éme Requête**	**3éme Requête**
Temps écoulé (s)	00 :00 :16.62	00 :00 :08.47	00 :00 :06.61
RAM (Mo)	103	100	76
Coût (CPU+I/O)	27813	27687	24606

Tableau IV.3 : Résultats des trois requêtes

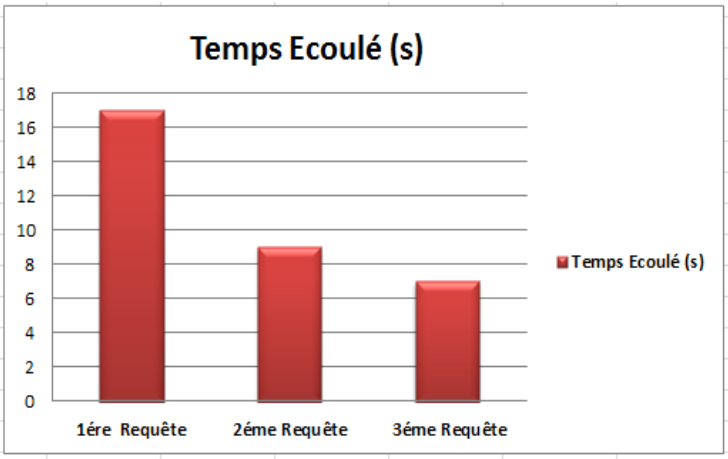


Figure IV.38 : *Histogramme temps écoulé des trois requêtes*

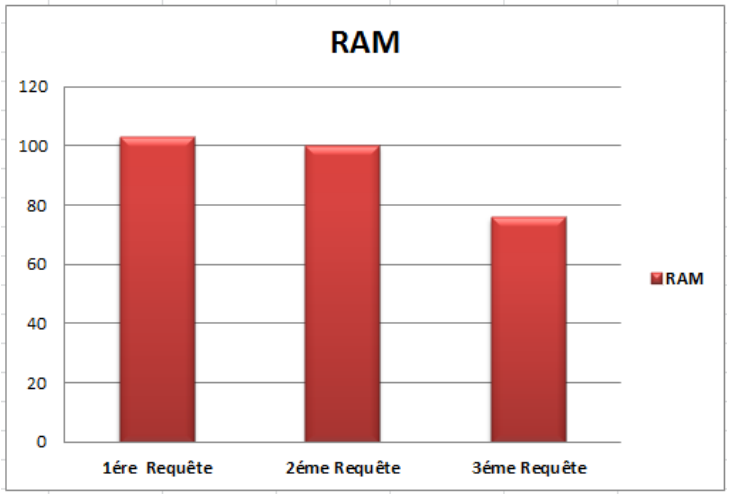


Figure IV.39 : *Histogramme RAM(Mo) des trois requêtes*

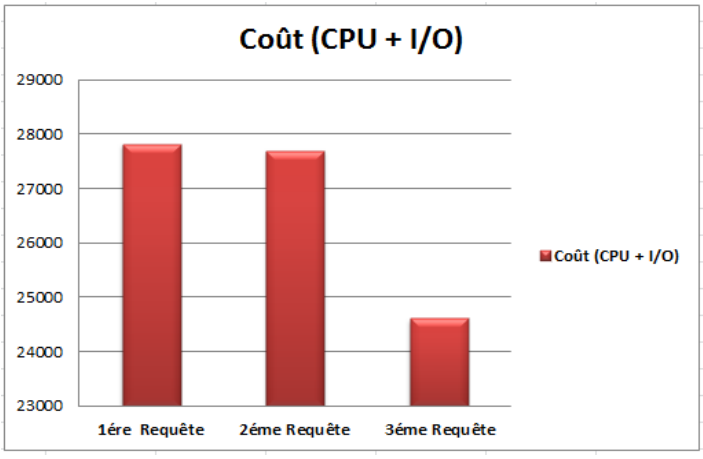


Figure IV.40 : *Histogramme Coût CPU + I/O des trois requêtes*

B. Analyse et interprétation des résultats

A ce stade, l'optimisation consiste à trouver les expressions équivalentes à la même requête, évaluer les coûts et choisir "la meilleure".

Nous remarquons que la 3^{éme} requête est la requête optimale car elle présente les meilleurs valeurs de temps de réponse, RAM, et Coût , ça s'explique du fait que dans cette dernière, l'optimiseur utilise l'index qu'on a créé, permettant de répondre à la requête sans parcourir toute la table BIGDEPT qui exige un temps considérable, ainsi que la réécriture de la requête en modifiant la clause **where** et les attributs sélectionnés ont contribués positivement à l'optimisation de la requête.

IV.5. Conclusion

Dans ce dernier chapitre nous avons accompli une série de tests qui ont généré un ensemble de résultats expérimentaux. Un rapprochement des différents résultats obtenus à tous les niveaux a été établi. Oracle 10g Entreprise Manager a été l'environnement et le champ sur lequel l'étude a été menée. Trois bases de données ont été créées et confrontées, le modèle optimal a été implanté sur trois machines de configurations différentes pour obtenir d'autres résultats expérimentaux. En fin, trois formulations différentes de la même requête dans la base de données optimale sur la meilleure machines ont été évaluées et comparées.

Conclusion générale

Conclusion générale

L'optimisation d'une base de données est un processus répétitif d'amélioration qui démarre dès la conception de la base de données, et ne s'arrête que lorsque la base cesse d'être utilisée. L'amélioration des bases de données inclut plusieurs axes de recherche. Notre étude a accentué sur les méthodes d'optimisation les plus répandues.

L'objectif du travail effectué consistait à une étude comparative selon trois grands axes d'optimisation, à savoir le schéma conceptuel, l'infrastructure matérielle, les requêtes. Nous traduisons en termes de coût les actions menées par les utilisateurs lors d'une interrogation de la base de données afin de répondre à un besoin exprimé. L'étude a été élaboré sur Oracle Database 10*g* professionnel qui offre une avalanche d'outils et mécanismes pour déterminer le meilleur plan.

Il est clairement montré que le bon choix du matériel et la bonne conception (schéma et typage de base de données) sont très bénéfiques en termes de performance. En ce qui concerne les requêtes, on constate, également, une différence significative en temps de réponse en raison des différentes manières de formulation de la même requête.

Avant de conclure, on peut admettre que l'étude qui a été menée, nous a permis d'approfondir nos compétences d'administration des bases de données et d'appliquer concrètement les différents mécanismes utilisés pour l'optimisation des bases de données dans un environnement idéal d'Oracle professionnel.

En conclusion, on peut affirmer qu'une base de données performante commence par une bonne conception, un environnement matériel adéquat en faisant les bons choix et en maitrisant les outils d'administration.

En perspectives, nous projetons d'étendre l'étude en se basant sur d'autres techniques d'optimisation, en utilisant les « **hints** » qui sont employés pour orienter l'optimiseur dans le choix du plan d'exécution pour compenser l'erreur d'estimation. Le parallélisme employé pour les traitements des données, la gestion de stockage et d'autres solutions d'ordre physique comme l'optimisation des paramètres internes des SGBD sont des pistes de recherche très prometteuses.

Références Bibliographiques

[1] Mirian Halfeld-Ferrari, BD, disponible sur le site http://www.univ-orleans.fr/lifo/Members/Mirian.Halfeld/ BD

[2] http://www.commentcamarche.net , consulté en février/2015

[3] Rim chaabane , Base de données-seance 4 , disponible sur le lien : www.ai.univ-paris8.fr/~lysop/bd/seance4-ModeleRel.pdf

[4] J.C. Sohm, Les bases de données relationnelles, chapitre 2, Tutoriel cerig.efpg.impg.fr septembre/2005

[5] Laurent Navarro, Optimisation base de données, Pearson Education France, ISBN: 978-2-7440-4156-3

[6] Cours et tutoriels DBA Oracle, disponible sur tuto-dba-oracle.com

[7] http://www.oracle.com , consulté en mars 2015

[8] http://www.oradoc.com/ora816/server.816/a76989/ch4g23.gif Article de Tittom

[9] http://www.Commentcamarche.net/Oracle-introduction-au-SGBD-Oracle, consulté en mars 2015

[10] www.OracleDataBaseIssue.com consulté en mars 2015

[11] Sebastien mahieu, Structure et architecture Oracle, support de cours

[12] www.dba.ora.fr consulté en février 2015

[13] Donald Burleson, ORACLE High-Performance SQL-Tuning , Mars 2003

[14] Oracle France - Support OR, Support de cours Référence : ntsql001

[15] **T**hierry Hamon, Bases de Données Avancées-Optimisation, support de cours Institut Galilée, Université paris 13

Liste des Figures

Liste des Tableaux

Liste des abréviations

BDD : Base de Données

DBA : Administrateur de Base de Données

DBMS : Database management system

SGBD : Système de gestion de bases de données

SGBDR : Système de gestion de bases de données relationnelles

SQL : Structured Query Language ou Langage Structuré de Requêtes

LDD : Langage de Définition de Données

LMD : Langage de Manipulation de Données

LCD : Langage de Contrôle de Données

SCD : Schéma Conceptuel de Données

SLD : Schéma Logique de Données

OEMDC : Oracle Entreprise Manager Database Control

SGA: System Global Area

PGA*:* Program Global Area

RBO: Rule Based Optimizer

CBO: Cost Based Optimizer

E/S: Entrées/Sorties Disque

I/O: Input/Output Disk

RAM: Random Access Memory

UC: Unité Centrale

Résumé

L'optimisation des bases de données est au cœur des problématiques soulevées par les administrateurs des bases de données. Elle a pour objectif de fournir, aux différents utilisateurs, un système optimal en termes de coût et d'usage. L'optimisation peut être réfléchie à plusieurs niveaux : au niveau conceptuel, au niveau infrastructure matérielle et au niveau exécution des requêtes pour atteindre les meilleures performances en terme de temps de réponse et utilisation de ressources.

Notre PFE consiste à appliquer un ensemble de tests comparatifs sur une base de données, en variant plusieurs paramètres dans les différentes phases de notre étude, pour aboutir au meilleur temps de réponse ou meilleur coût en fonction des ressources matérielles disponibles.

Mots clés : BDD, SGBDR, ORACLE, CBO, Optimisation

Abstract

The optimization of data bases is at the heart of the issues raised by the database administrator. Its objective is to provide, to different users, an optimal system in terms of cost and use. Optimization fear being reflected at several levels: the conceptual level, the physical infrastructure level and at query execution to achieve the best performance in terms of response time and use of resources.

Our PFE consists of applying a set of comparative test son a data base, by varying several parameters in different phases of our study, to achieve the best response time or the lowest cost in terms of material resources.

Keywords : DB, DBMS, ORACLE, CBO, Optimisation

ملخص

تحسين الاستفادة من قواعد البيانات هو في قلب القضايا التي يثيرها مدراء قواعد البيانات. هدفها هو توفير، لمختلف المستخدمين، النظام الأمثل من حيث التكلفة والاستخدام. يمكن التفكير فيها على عدة مستويات :أثناء تصميم قواعد البيانات، على مستوى الأجهزة وعند طرحنا للأسئلة لتحقيق أفضل أداء من حيث زمن الاستجابة واستخدام الأجهزة.

يسعى عملنا الى اجراء مقارنة عبرمجموعة من الاختبارات على قاعدة البيانات، من خلال تغيير العديد من العوامل في مختلف مراحل الدراسة التي قمنا بها ، من أجل تحقيق أفضل وقت استجابة أو بأقل تكلفة من حيث الأجهزة المتوفرة.