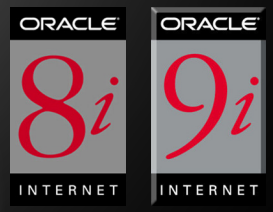
Oracle in Hindi



BccFalna.com
097994-55505

Kuldeep Chand

In this EBook I have covered **SQL** so that you can learn the Query Language and can Query various kinds of Data from the Database, created in First Step "**Database Designing Process**". This is **Standard Query Language** which is approximately same in approx. all DBMS Software like Oracle, DB2, MySQL, SQL Server, etc…

Then I have covered PL/SQL which is specially added Functionality in Oracle so that we can use Oracle with more control to develop Professional Database Applications. This feature is not available in any other DBMS Software.

And in last part I have used Visual Basic 6 to develop Frontend of a Professioanl Database Application. So that you can understand the fundamental concepts of Professional Oracle Database Application Software Development Process properly.

Oracle

8i/9i

(SQL/PLSQL)

In Hindi

■■■
Kuldeep Chand

Betalab Computer Center
Falna

Oracle 8i/9i – SQL/PLSQL in Hindi

Lead Editors: **Kuldeep Chand**

Distributed to the book trade worldwide by Betalab Computer Center, Behind of Vidhya Jyoti School, Falna Station Dist. Pali (Raj.) Pin 306116

e-mail bccfalna@gmail.com

or

visit http://www.bccfalna.com

For information on translations, please contact BetaLab Computer Center, Behind of Vidhya Jyoti School, Falna Station Dist. Pali (Raj.) Pin 306116

Phone **97994-55505**

This book is dedicated to those

who really wants to be

a

PROFESSIONAL DEVELOPER

INDEX OF CONTENTS

Table of Contents

ORACLE

THE HISTORY

Oracle – The History

Oracle दुनियां की बहुत ही बडी Software Companies में से एक है, जिसकी स्थापना 1977 में Relational Software Corporation के नाम से हुई थी। इस Company ने दुनियां के सबसे पहले Relational Database Management Software Oracle को Develop किया।

इसे Develop करने का पहला मुख्य कारण Database को नई Create की गई SQL Language के Compatible बनाए रखना था और दूसरा मुख्य कारण ये था कि वे Database Softwares को C Language में Develop करना चाहते थे, ताकि Database Software Cross Platform पर Run हो सकें। इन दोनों मुख्य Requirement को ध्यान में रखते हुए Company ने 20 साल तक इस Software को Develop किया और इस Software को Oracle का आज का रूप प्राप्त हुआ।

Oracle के पहले और दूसरे Version को Company के नाम Relational Software Incorporated (RSI) से ही Market में लाया गया था जबकि तीसरे Version को Market में लाने के साथ ही Company का नाम Change करके Oracle Corporation कर दिया गया।

हालांकि Oracle की Stability व Reliability धीरे–धीरे Improve हुई जब तक कि Oracle का पांचवा Version तैयार नहीं हुआ। इस पांचवे Version में Oracle ने जिस Architecture को Use किया, उसे आज हम Client/Server Architecture के नाम से जानते हैं। इस Version में Parallel Server Option को भी Define किया गया था। इसके बाद के छठे व सातवें Versions में High Performance, High Reliability व Greater Scalability प्राप्त करने के Trend को जारी रखा गया।

आठवें Version के साथ ही Oracle के Architecture को फिर से Modify किया गया और इसे **Network Computer Architecture** में Convert किया गया, जिसमें Oracle Database Fundamental Part के रूप में था।

हालांकि Oracle 8 के बहुत सारे Features को Oracle 7 में ही Appear कर दिया गया था, लेकिन ये नया Version Universal Database के Concept को ज्यादा Better तरीके से Implement करता था।

Universal Database एक ऐसा Database Implementation होता है, जो केवल Relational tables को ही नहीं बल्कि किसी भी प्रकार के Data को Store व Process कर सकता है। विभिन्न प्रकार के Data को Manage करने के Concept को Oracle 8 में Demonstrate किया गया, जिसमें हम विभिन्न प्रकार के Large Objects (LOB) को, Object Option के साथ Structured Objects को तथा विभिन्न प्रकार के Multimedia Objects जैसे कि Graphical, Musical व Videos को Mange कर सकते हैं। आठवें Version के बाद से इसी Trend को जारी रखा गया

है, जिसमें ज्यादा Data, ज्यादा Users व Better Performance को Manage किया जाता रहा है।

The Relational Database Architecture

Oracle के Structure को हम दो भागों में बांट कर देख सकते हैं। पहला भाग Oracle का Simple रूप है जबकि दूसरा भाग Oracle का Advance रूप है। यहां हम Oracle के Simple रूप को समझने की कोशिश करेंगे, जिसमें हम File-Based System, Client/Server System व Multi-User Architecture व Network Computing Architecture (NCA) के बारे में जानकारी प्राप्त करेंगे। विभिन्न प्रकार के Architectures के बीच के अन्तर को समझना इसलिए जरूरी है ताकि हम हमारे Application व Database की जरूरत के आधार पर इन में से किसी Architecture को Choose कर सकें या एक Architecture से दूसरे Architecture पर Switch कर सकें।

File-Based Systems

Relational Database का सबसे सरल रूप File-Based System होता है। उदाहरण के लिए Microsoft Company का Microsoft Access एक File-Based Relational Database Management System है। Access में **.mdb** Extension के नाम की एक File होती है। इसी File में Database से सम्बंधित विभिन्न प्रकार के अन्य सभी Database Elements जैसे कि Tables, Queries व Forms होते हैं। ये File किसी User के स्वयं के Computer अथवा किसी Network पर स्थित हो सकती है।

हालांकि हम सामान्यतया इस File के Data को Display करने के लिए Access के Form या Query Elements का प्रयोग करते हैं, इसके अलावा हम Visual Basic जैसे किसी Software का प्रयोग करके भी MS-Access के इस File-Based Database को Access कर सकते हैं।

जब हम Data को Store करने के लिए Back-End के रूप में किसी File-Based System जैसे कि MS-Access का प्रयोग करते हैं, तब हर Front-End Application जैसे कि Visual Basic को इस बात के लिए सावधान रहना होता है कि इस File-Based System से Data को किस प्रकार से Read करना है या इसमें Data को किस प्रकार से Write करना है साथ ही एक ही समय में एक से ज्यादा Users समान Database के Data को बिना किसी परेशानी के Use कर सकें, इसके लिए हमें Application में Locking Mechanism के लिए भी परिभाषित करना पडता है।

जब हम Back-End के रूप में Microsoft Access को तथा Front-End के रूप में Visual-Basic जैसे किसी Application Developer को Use करना चाहते हैं, तब Front-End के लिए MS-Access के Database को Access करने के लिए हमें JET Engine का प्रयोग करना पडता है। यदि हम JET के अलावा किसी अन्य तरीके से MS-Access के Database को Access करने की कोशिश करते हैं, तो हमारा Database Corrupt हो सकता है।

चलिए, पहले Locking Mechanism को समझते हैं। जब MS-Access जैसा कोई Database किसी Network पर स्थित होता है और उसे एक से ज्यादा Users Access करते हैं, तब किसी समय उस Database की किसी एक ही Table के Data को एक से ज्यादा Users Access करने के लिए Request कर सकते हैं। इस स्थिति में यदि दोनों ही Users एक साथ किसी Table के Data को Access करते हैं, तो Table के Data के Corrupt होने की सम्भावना रहती है।

इसलिए एक ऐसे तरीके का प्रयोग किया जाता है, जिसमें यदि कोई एक User किसी Network पर स्थित Database के किसी Element को Access कर रहा होता है, तो उस Element को तब तक कोई दूसरा User Access नहीं कर सकता जब तक कि पहला User उस Element को Free ना कर दे।

यानी एक ऐसी प्रक्रिया को Use किया जाता है, जिसमें पहले User के लिए ही कोई Database Element Useable होता है, किसी अन्य User के लिए वही Element जिसे पहला User Use कर रहा है, तब तक के लिए Inaccessible होता है, जब तक कि पहला User उस Element को Free नहीं कर देता। इस स्थिति में पहले User के अलावा सभी अन्य Users के लिए वह Database Element Locked रहता है। इस प्रक्रिया को **Locking Mechanism** कहा जाता है।

File-Based Databases के साथ परेशानी ये है कि हम इसे बढा नहीं सकते हैं। एक छोटे Business System के लिए MS-Access का Database Suitable होता है, लेकिन बडे System के लिए MS-Access जैसे File-Based Database को Use नहीं किया जा सकता है।

हालांकि एक File-Based Database 100MB Data के साथ Successfully Deal कर सकता है, लेकिन जब Database में Records की संख्या काफी बढने लगती है या जब एक ही Database को दर्जनों Users Access करने लगते हैं, तब एक File-Based Database की Performance काफी घट जाती है।

उदाहरण के लिए यदि हम किसी File-Based Database के किसी Table के किसी Column में Stored 1 लाख Records में से सबसे बडी Value को प्राप्त करना चाहें, तो ये File-Based System सभी Records को Network से प्राप्त करेगा, फिर उन्हें Locally Process करेगा और Required Value को खोजेगा। एक लाख Records को Network से Retrieve करने में एक File-Based System को बहुत ही ज्यादा समय लगेगा और हमें बहुत ही ज्यादा देर तक Required Result के लिए Wait करना पडेगा।

Client/Server Architecture

जब किसी Database को बहुत सारे Users Use करते हैं और Database में बहुत सारा Data Store करना होता है, तब हम File-Based Approach को Use नहीं कर सकते हैं, क्योंकि उस स्थिति में File Based System की Performance काफी कम हो जाती है।

एक File-Based System से Performance प्राप्त करने के लिए हमारे पास एक बहुत ही ज्यादा Speed वाला Computer होना चाहिए और एक बहुत ही तेज Speed का Network होना चाहिए। जबकि हमेंशा ऐसा सम्भव नहीं होता है। आज भी Network की Speed बहुत ज्यादा तेज नहीं है और ज्यादा तेज Speed वाले Networks के लिए बहुत ज्यादा खर्चा करना पडता है। इसलिए इस समस्या का दूसरा समाधान Client/Server Architecture में है और यही वह स्थिति भी है, जहां Oracle महत्वपूर्ण भूमिका निभाता है।

एक Client/Server Architecture में Client व Server दोनों होते हैं। Client Computers ज्यादा Powerful नहीं होते हैं और इन पर Visual Basic जैसा कोई Client Application Stored होता है। Server एक ज्यादा Powerful Computer होता है, जिस पर Database Server को Store किया जाता है साथ ही System से सम्बंधित सभी महंगे Equipments को इस Server के साथ ही Attach किया जाता है।

हम हमारे Client Software को Visual Basic में Develop करेंगे, हालांकि Clients को विभिन्न प्रकार की Programming Languages में, विभिन्न Hardware Platform तथा Operating Systems के लिए Develop किया जा सकता है। इसी तरह से हम हमारे Server Software को Oracle में Develop करेंगे, जिसे लगभग सभी प्रकार के Operating Systems व Hardware पर Place किया जा सकता है।

Client/Server Architecture द्वारा प्राप्त होने वाली ज्यादातर Advantages Server में ही Exist होती हैं, जो Server को Data पर विभिन्न प्रकार की Processing को Apply करने की Capabilities Provide करती हैं। आज के PC इतने Powerful हैं जो Data को विभिन्न तरीकों से Process करके उन्हें विभिन्न प्रकार की Graphical Form में Display कर सकते हैं।

Client/Server Architecture में किसी Database Application को दो हिस्सों में Develop किया जाता है। Server Part Data पर विभिन्न प्रकार की Processing करने व विभिन्न प्रकार की Information Generate करने का काम करता है जबकि Client Part Server से Generate होने वाले विभिन्न प्रकार की Information को विभिन्न प्रकार के Format में Display करने व Output Generate करने का काम करता है। दोनों ही Part एक दूसरे से स्वतंत्र रूप से काम करते हैं।

Multi-Tire Architecture

Client/Server Architecture में भी जब Database के Data बढते हैं, तब कुछ परेशानियां पैदा होती हैं और इस Situation के कई Solutions हैं। Multi-Tier Software में एक ही Software के कई Tiers होते हैं और हर Tier एक Specific काम को अच्छे तरीके से पूरा करता है। Tiers को सामान्यतया तीन भागों में बांटा जाता हैः

1 पहला Client या GUI Tier,
2 दूसरा Middle या Business Tier व
3 तीसरा Data Tier.

Multi-Tier Architecture को सामान्यतया **Three-Tier Architecture** भी कहा जाता है। Multi-Tier Approach में Visual Basic जैसे Frontend से सम्बंधित Softwares को Client का Role Play करता है, जैसा कि Client/Server Architecture में करता है, हालांकि Multi-Tier Architecture में Visual Basic को Client/Server Architecture की तुलना में बहुत ही कम Data Processing का काम करना पडता है।

Multi-Tier Architecture में Visual Basic का मुख्य काम Data को विभिन्न प्रकार से Display करना ही होता है। लेकिन Visual Basic Middle Tier में बहुत ही महत्वपूर्ण Role Play करता है, जहां पर विभिन्न प्रकार की Processing व Business Rules को Capture किया जाता है।

Visual Basic Middle Tier में ये महत्वपूर्ण Role इसलिए Play कर सकता है, क्योंकि Visual Basic 6 में हम ActiveX Components Create कर सकते हैं, जो कि एक Transaction-Processing Environment जैसे कि Microsoft Transaction Server (MTS) के Under में Server पर Run हो सकता है। Multi-Tier Architecture में हम निम्न सुविधाओं को प्राप्त करते हैं, जिन्हें Provide करने में एक Client/Server Architecture Fail हो जाता हैः

1 Create किए गए Applications को Deploy करना व Update रखना सरल होता है। ये Architecture उस स्थिति में काफी महत्वपूर्ण साबित होता है, जब हम हमारे Application को Web पर Use करने के लिए बढाना चाहते हैं।
2 इस Architecture की वजह से जब भी Business के Business Rules Change होते हैं, हमें केवल Client Tier को ही Modify करना पडता है।
3 विभिन्न प्रकार के Business Rules को Middle Tier में Encapsulate करने के कारण कोई भी अन्य Application, जो कि इन Business Rules को Access करना चाहता है, वह Shared Components Create करके, एक ही Client Application के विभिन्न Business Rules को अन्य Client Applications में Access कर सकता है।
4 Business Layer इस बात के लिए निश्चित करता है कि Data की Security Standard नियमों पर ही आधारित है इसलिए Data पूरी तरह से Stable हैं।
5 Multi-Tier Architecture के कारण Application Scalable होता है।

Network Computing Architecture

Oracle के Multi-Tier Design को ही Network Computing Architecture (NCA) भी कहा जाता है। इस Architecture में Multi-Tier Architecture के सभी Features शामिल होते हैं। फिर भी NCA में Client की तुलना में मुख्य महत्व Middle Tier व Back-End Tiers का ही होता है। NCA में Multi-Tier Architecture के तीनों Layers या तीन से अधिक Layers हो सकते हैं।

Clients

तीनों ही Tiers को निम्नानुसार तीन Categories में Describe किया गया है, जिसे हम **Thin Client** कह सकते हैं। इस Architecture में Client एक Universal Thin Client होता है, जो निम्न में से कोई भी हो सकता हैः

- A Traditional Web Browser
- A Java-Based Client
- A Network Computer

Thin Client बनाने का मुख्य Purpose ये हैं कि कोई भी Application किसी भी User तक पहुंच सके, फिर वह User चाहे जो Operating Software Use करता हो या चाहे जिस Software के Through वह इस Application को Use करे।

Application Server

Application Server एक Special प्रकार का Software Piece होता है, जो कि Middle Tier को Efficient व Scalable बनाने के लिए Develop किया जाता है। किसी भी Middle Tier Software को Develop करने का एक मकसद ये भी होता है कि Server से आने वाली किसी भी Request को पूरा करने के लिए ये Middle Tier अपने किसी भी Code, Object या Component को Server के लिए Available करे।

Application Server एक Flexible Design होता है, जिसे Oracle में **Cartridges** कहा जाता है। Cartridges वे Products होते हैं जो Server Software के Top पर उसी तरह से Run हो सकते हैं, जिस तरह से हमारे Web Browser में **Plug-Ins** Run होते हैं। Cartridge का प्रयोग करके हम मुख्यतः Base Application Software को ही अपने स्वयं के Code Statements द्वारा Extend करते हैं। Cartridge की एक सबसे बडी विशेषता ये है कि हमारे Cartridge Code में यदि कोई Error हो तो हमारा Application Server काम करना बन्द नहीं करता है।

Universal Data Server

Data Server Layer को Design Philosophy के कारण अक्सर Universal Data Server के नाम से भी पुकारा जाता है क्योंकि ऐसे Data Server किसी भी प्रकार के Data को Handle करने में सक्षम होते हैं। Oracle 8 में हम 4 GB तक का Data Handle कर सकते हैं। Oracle की इस विशेषता के कारण हम Graphics व Video की Information को Oracle के Database में Store कर सकते हैं।

ORACLE

ARCHITECTURE

Oracle - Architecture

Oracle Database को हम Database व Instance दो रूपों में देख सकते हैं। Technically एक Oracle Database उन Files का एक Physical Collection होता है, जो Database में Exist होते हैं। लेकिन स्वयं Database अपने स्तर पर कुछ नहीं होता है, क्योंकि Database से Directly Interact करने का User के पास कोई तरीका नहीं होता है।

जबकि Instance Oracle का एक Running Database होता है, जिसमें Memory Structure होता है और Associated Process होते हैं, जो इन Memory Structures से Interact होते हैं व Memory Structures को Manipulate करते हैं।

जो लोग Oracle Relational Database Management Systems (RDBMS) पर काम करते हैं, वे अक्सर Instance व Database को Interchangeably Use करते हैं। इन दोनों के बीच का मुख्य अन्तर ये है कि एक Physical Database File से कई Instances Connect हो सकते हैं।

जब किसी Oracle Instance को Start करने के लिए कोई Script Run होता है, तब विभिन्न प्रकार के Processes को Start करने के लिए Oracle उन्हें Required Memory Space Allocate करता है।

इस तरीके में हमारे पास एक ही Server पर एक से ज्यादा Database Instance हो सकते हैं, जो कि एक दूसरे से स्वतंत्र रूप से Run होते हैं। Oracle Parallel Server में Same Single Data Files के समूह के साथ एक से ज्यादा Instances Mount हो सकते हैं। Oracle को समझने के लिए हमें Memory, Server Processes व Data Files तीनों Concepts को समझना होता है।

Memory Structures

System Global Area (SGA) जिसे कई बार Shared Global Area भी कहते हैं, Oracle का Main Memory Component होता है। Oracle एक Memory Based Database है, जिसका मतलब ये है कि Data, Locks व अन्य Elements Memory में Hold होते हैं। SGA की Size कभी भी Server Machine की Physical Available Memory से ज्यादा नहीं होती है, अन्यथा Virtual Memory Paging होने लगता है, जिससे Oracle Database की Performance कम हो जाती है।

जब किसी Computer में ज्यादा Physical Memory नहीं होती है, तब बडे Programs व बहुत सारे Programs को एक साथ चलाने के लिए Operating System में Virtual memory Model को Use किया जाता है।

इस Model में जो Application Program Active होता है, उसके Data तो Main Memory में रहते हैं और जो Program Inactive होते हैं, उन्हें Hard Disk पर भेज दिया जाता है। Hard Disk में जितना Free Space होता है, उसे Operating System द्वारा Virtual Memory की तरह Use कर लिया जाता है।

Main Memory में जो भी Application Store होता है, वह Memory के विभिन्न हिस्सों में Store होता है। Memory के विभिन्न हिस्सों को Page कहा जाता है और Main Memory के Data को Disk की Virtual Memory में भेजने की प्रक्रिया को **Paging** कहा जाता है।

सामान्यतया Memory Pages की Size 4KB या 8KB होती है। पहले जो Operating Systems बने थे, उनमें पूरे Application को ही Virtual Memory में भेज दिया जाता था। इस प्रक्रिया को **Swapping** कहा जाता था। इसलिए आज के नए Operating System में भी जिस File के Data को Main Memory से Virtual Memory में भेजा जाता है, उसे **Swap File** ही कहते हैं।

Oracle के System Global Area (SGA) Memory में निम्न Components होते हैं:

- **Database Buffer Cache**
- **Redo Log Buffer**
- **Shared Pool Area (Which is made up of the following components:**
  - **Library Cache (Which includes a Shared SQL Area)**
  - **Data Dictionary Cache (also known as the Row Cache)**

जब कोई Oracle Instance Start होता है, तब Oracle उसके SGA को Memory Allocate करता है। Oracle स्वयं ही इसके विभिन्न Components की Memory को Manage करता है और तब तक किसी भी Component की Memory को Release नहीं करता है, जब तक कि Oracle के उस Instance को Shut Down नहीं कर दिया जाता। साथ ही Allocate की जाने वाली Memory Dynamically Increase नहीं होती है।

Memory Allocation को Change करने के लिए, हमें Oracle के Instance को Shut Down करना पडता है, उसके बाद Initialization Parameters में Change करना होता है और फिर से Oracle के Instance को Start करना होता है।

The Database Buffer Cache

ये SGA की Memory का वह हिस्सा होता है, जहां Database की किसी Data File से Read किया गया Data Store होता है। यही Data किसी Visual Basic Client Application द्वारा Access होता है। ये Cache Buffers से बनता है, जिसकी Individual Size उस Physical

Database Files के किसी Database Block के बराबर होती है, जिसमें Data को Manage किया जा रहा होता है।

यदि इस Memory में कोई Data ना हो, तो User के Client Application की Request पर Generate होने वाला Data इसी Buffer में Store होता है। जब हम Database से Connect होते हैं, तब Oracle एक Server Process Create करता है, जिसे Shadow Process भी कहते हैं। ये Process Application की Request को हमारी जरूरत के आधार पर Handle करता है।

Cache में दो Lists होती हैं जो Buffer को Manage करती हैः जिन्हें **Least Recently Used (LRU) List** व **Dirty List** कहते है। Least Recently Used List सबसे ज्यादा महत्वपूर्ण होती है। **SELECT** जैसे किसी Command से जितनी बार भी Data Buffer Access होता है, ये Data इस List के Top पर पहुंच जाता है।

जबकि जो Buffer Access नहीं होता है, वह इस List के Bottom में Move हो जाता है। इस List में हमेंशा अन्तिम बार Access किया गया Data Stored रहता है। इस Buffer को Create करने का मुख्य Purpose ये होता है कि किसी Data के लिए Physical Disk Reading Operation को कम किया जा सके।

Oracle का दूसरा List उन Data Buffers की जानकारी को Hold करता है, जिन्हें Change किया गया है। जब भी Client Application किसी ऐसे Data की Request करता है, जो कि Database Buffer Cache में नहीं होता है, तो Oracle LRU List को Free Buffer के लिए Search करता है। यदि Search के दौरान Dirty List Buffer प्राप्त होता है, तो Server Process उन्हें Dirty List में Move कर देता है।

जब कुछ निश्चित संख्या में Buffers को Scan कर लिया जाता है और कोई भी Free Buffer प्राप्त नहीं होता है, तो Database Writer Process (DBWR) कुछ Dirty Buffers को Disk पर Write करता है, जो उन्हें Free कर देते हैं। वे Buffers जो कि Dirty नहीं होते हैं, वे किसी भी समय List से Move Out हो सकते हैं और वे Buffers जो कि LRU List में Bottom पर होते हैं, वे सबसे पहले Free हो सकते हैं।

Redo Log Buffer

Redo Log Buffer Memory का वह Area होता है, जो Database में किए जाने वाले सभी General Changes को Hold करता है। Database में किए जाने वाले विभिन्न Changes को Redo Log Entries कहते हैं। ये Datablocks की Copy नहीं होते हैं बल्कि ये वे जानकारियां होती हैं, जिनकी जरूरत Datablocks को फिर से Reconstruct करने के लिए पडती है।

जब Database के Changes को Capture किया जाता है, तब इन Entries को जितना हो सकता है उतना छोटा रखा जाता है। इनका प्रयोग Database के Crash होने पर Recovery के

लिए किया जाता है। Buffer को Memory में Circular Area के रूप में Use किया जाता है, इसलिए Buffer Memory के अन्त पर पहुंच कर फिर से Memory की शुरूआत से Data Holding का काम करने लगता है।

Redo Log Entries को Storage में Permanently Write या Overwrite किया जाता है। Log Writer Process (LGWR) इस Buffer को Monitor करता है और जब भी जरूरत होती है, इसके Contents को Ftush करता रहता है। Server Processes Redo Log Buffer में Writing को Control करता है और LGWR Redo Log Entries के आधार पर Changes को Buffer में Write करता है।

Shared Pool

Shared Pool SGA Memory का एक ऐसा Area होता है, जिसमें उस Memory का हिस्सा होता है, जिसे विभिन्न Users द्वारा Share किया जा सकता है। इसमें बहुत सारे छोटे–छोटे Memory Areas होते हैं, जिन्हें अग्रानुसार समझाया गया है:

Library Cache

Library Cache में ऐसे बहुत सारे Components होते हैं, जिन्हें विभिन्न Database Users द्वारा Share किया जाता है। इसका Shared SQL Area सबसे महत्वपूर्ण Component होता है। Shared SQL Area में हर SQL Statement की Detail के साथ ही SQL Statement के Execute होते समय के Execution Plan की भी Detail होती है, जिसे **Parse Tree** कहा जाता है।

Shared Area में वे Identical SQL Statement Hold होता है, जिसे एक से ज्यादा Users Share करते हैं। इस Statements का सभी Users के लिए बिल्कुल Identical होना जरूरी होता है साथ ही ये SQL Statements समान Object से ही Refer होने भी जरूरी होते हैं।

इसका फायदा ये होता है कि यदि एक से ज्यादा Users एक समान SQL Statements को Execute कर रहे होते हैं, तो एक ही Object से Refer होने के कारण Oracle एक ही SQL Statement को Hold करता है, जिससे Database की Performance Increase हो जाती है।

Data Dictionary Cache

SGA का Data Dictionary Cache उन Tables की Information Hold करता है, जिन्हें Use किया जा रहा है। ये Table Name, Table Column Name व Column Data Types को Store करता है। इस स्थिति में जब भी किसी SQL को Parse करने की जरूरत होती है, SQL से सम्बंधित Information पहले से ही उपलब्ध रहती है। ये सभी Memory Areas व Processes के लिए Available रहता है और ये पूरी तरह से Performance से सम्बंधित होता है। Data

Dictionary Cache के हिस्सों को भी उसी तरह से Ftush किया जाता है, जिस तरह से Database Buffer Cache को किया जाता है।

Additional Memory Areas

इन सभी Memory Areas के अलावा दो और Memory Areas होते हैं, जो कि निम्नानुसार हैं:

1. **Program Global Area** या **PGA,** जो कि Server Processes के लिए Control Information को Store करता है।
2. **Sort Areas** जो कि Memory-Based Sorts के लिए Use होता है।

Processes

Oracle को जिन Memory Structures की जरूरत होती है, उन्हें समझने के बाद अब हमें उन Processes को समझना है, जिनकी जरूरत इन विभिन्न Memories को Manage करने के लिए Oracle को होती है। ये Processes Memory Structures के साथ मिलकर Oracle के मुख्य Concept यानी Oracle Instance को परिभाषित करते हैं।

एक Oracle Instance के साथ बहुत सारे Background Processes Associated होते हैं और ये सभी Database में अलग–अलग तरीके से अपना Role Play करते हैं। मुख्य रूप से चार Processes Mandatory होते हैं, जो कि निम्नानुसार हैं:

1. The Database Writer (DBWR),
2. The Log Writer (LGWR),
3. The Process Monitor (PMON),
4. The System Monitor (SMON),

इन चारों Processes के बारे में हम आगे विस्तार से जानकारी प्राप्त करेंगे। इन चार Processes के अलावा हम एक और Process यानी **Archive Process (ARCH)** के बारे में भी जानकारी प्राप्त करेंगे, क्योंकि इसकी जानकारी के बिना हम Disk Failure की स्थिति में Database को पूरी तरह से Recover नहीं कर सकते हैं और अन्त में हम **Listener Processes** के बारे में जानेंगे, जो कि किसी Network User जैसे कि Visual Basic जैसे किसी Client Program को Database से Connect होने की सुविधा प्रदान करता है। किसी भी Oracle Instance में मुख्य रूप से निम्न Processes Involved होते हैं:

Process Name : ARCH

Description : ये Process Redo Logs को Archive करने के लिए Disk पर या किसी अन्य External Device पर Copy कर देता है।

Process Name : CKPT

Description : ये Process Checkpoint Event के समय Data File के Header को Synchronization Number के साथ Update करता है। ये Process Optional होता है। यदि हम इस Process को Use ना करें, तो LGWR Process इस काम को पूरा करता है।

Process Name : Dnnn

Description : ये Dispatcher Process एक या एक से अधिक User Processes के लिए Database के Shared Access को Control करता है। ये Process तब Enabled हो जाता है, जब Multithreaded Server Option को Use किया जाता है। ये उस Server Processes को Replace कर देता है, जो User Processes के आधार पर SQL Request को Handle करता है।

Process Name : DBWR

Description : ये Process उस Data को File में Write करता है, जिसे Database Cache में Change किया गया होता है, ताकि Request किए गए नए Data को SGA में Hold करने के लिए Room Create किया जा सके। इसका प्रयोग Transaction को Commit करने के लिए नहीं किया जाता है।

Process Name : LCKn

Description : ये Process केवल Parallel Server Option के लिए Enabled किया जाता है।

Process Name : PMON

Description : जब Server Process Failure की स्थिति होती है, तब Process Monitor Transaction को Rollback करने का काम करता है। ये Process इस बात को निश्चित करता है कि जिस Transaction के लिए Database के Resources को Lock किया गया था, वे Transaction के Fail होने की स्थिति में Released हो जाएंगे।

Process Name : RECO

Description : ये Recover Process Networking Failure के बाद Distributed Transaction को Clean कर देता है।

Process Name : Snnn

Description : ये Shared Server Process केवल Multithreaded Server Option में Use होता है। ये Dispatcher Processes द्वारा Request किए गए SQL को Process करने का काम करता है।

Process Name : SMON
Description : Crash Recovery की स्थिति में SMON Database Changes को Last Checkpoint Event तक Replay कर देता है। ये Changes Online Redo Log Files में Store हो जाते हैं।

Process Name : SNPn
Description : ये Automatic Snapshot Refresh Process, Distributed Database Configuration में Master Database व अन्य Databases के Changes को Propagates करने के लिए Use होता है। हम इन्हें हमारे स्वयं के कामों के लिए भी Use कर सकते हैं।

Oracle Instances की सबसे ज्यादा Important Processes DBWR, LGWR, PMON व SMON हैं। यदि इनमें से कोई भी Service Fail हो जाए, तो Oracle Instance Fail हो जाता है। हालांकि ARCH एक Optional Process है, लेकिन फिर भी इस Process के बिना Data Archive नहीं किया जा सकता है और Failure के बाद Recovery का Chance बहुत ही कम होता है।

Windows NT पर Oracle के उपरोक्त सभी Processes एक Single Service द्वारा Active होते हैं। Service एक Executable Process होता है, जो कि Windows NT वाले Computer पर Install होता है और उस स्थिति में भी Run होता रहता है, जब कोई भी User Logged On नहीं होता है। वह Oracle Service जो कि इन सभी Threads को NT पर Run करता है, **OracleServicesid** कहलाता है, जहां **sid** Run होने वाले Oracle Instance का नाम होता है।

हम कई और Oracle Services को देख सकते हैं, जो कि उस स्थिति में स्वयं ही Automatically Start हो जाते हैं, जब Windows NT और **OracleTNSListener** को Restart किया जाता है।

OracleTNSListener एक ऐसा Service होता है जो Listener Process को Start करता है। ये Process Network पर स्थित विभिन्न Users को Database से Connect करता है। Server पर स्थित स्वयं **SQL *Plus Session** जैसे Native Connection के लिए Connection को पूरा करने के लिए किसी Listener Process की जरूरत नहीं होती है।

Database Writer (DBWR)

हम Database में जो भी Change करते हैं, वे सभी Changes Initially Memory में होते हैं। बाद में जरूरत होने पर Oracle स्वयं ही इन Changes को Data Files में Update करता है। Data File का Updation Oracle स्वयं ही Database Writer Process द्वारा Handle करता है। ये ही वह मुख्य Process होता है जो System Global Area के Data को Data Files में Write करता है। यदि User जिस Data के लिए Oracle से Request कर रहा है, वह Data Database

Buffer Cache में ना हो, तो Oracle स्वयं ही उस Data को पहले Database Buffer Cache में Hold करता है और उसके बाद User की Request को पूरा करता है।

Database Writer Process Database Buffer Cache से Dirty Blocks को Database Files में Write करता है। जब ये Buffer INSERT, UPDATE या DELETE जैसे किसी SQL Statement के Execution के कारण Change होते हैं, इस Buffer को **Dirty Block** के रूप में Mark कर दिया जाता है।

जब Buffer में Dirty Blocks की संख्या Dirty List में एक मान तक पहुंच जाती है, तब Database Writer Least Recently Used List को Use करके ये पता लगाता है कि वह Most Suitable Buffer कौनसा है, जिसे Data Files में Write करके उस Buffer को Free किया जा सकता है। क्योंकि Oracle स्वयं का File Structure Use करता है, इसलिए Database Writer कई Data Blocks को एक ही समय में Data Files में Write कर सकता है, जिसे Multi-Block Write कहा जाता है।

फिर भी चूंकि Oracle सभी Data को Data Files में Continuously Write नहीं करता है, इसलिए उस स्थिति में कुछ Data Memory में ही रह सकता है, Crash की स्थिति में इस Memory का Data, Data File में Write नहीं हो पाता है। इस Problem को Log Writer द्वारा Solve किया जाता है।

Log Writer (LGWR)

हम हमारे Database के Data में जो भी Changes करते हैं, वह Change न केवल Database Buffer Cache में होता है, बल्कि उस Change की Entry Redo Log Buffer में भी होती है। यदि हमारा Oracle Instance Crash हो जाता है, तो Buffer की Entries Lost हो जाती हैं।

Crash के बाद Recovery को Enable करने के लिए इन Entries को External-Disk Files में Store किया जाना जरूरी होता है। Log Writer Process इन Entries को Redo Log Buffer File से लेकर एक या एक से अधिक Online Redo Log Files में Write करने का काम करता है।

चूंकि जब भी हम हमारे Database में Change करते हैं, उस Change की Entry Redo Log Buffer में भी होती है, इसलिए Crash होने की स्थिति में Oracle इन Redo Log Buffer Files का प्रयोग करके Recovery का काम करता है।

यदि हम Checkpoint Process (CKPT) को Enable नहीं करते हैं, तो Log Writer Data File के Header की Updating Checkpoint Event की स्थिति में स्वयं करता है। Log Writer Process निम्न स्थितियों में Redo Log Buffer के Contents को Write करता हैः

1 जब Database Transaction Commit होता है।

2 जब Redo Log Buffer एक तिहाई भर जाता है।
3 Checkpoint Event की स्थिति में।

यदि उपरोक्त में से कोई भी स्थिति ना हो, तो Log Writer हर तीन सेकण्ड के अन्तराल पर Data को Disk Files में Write करता रहता है।

System Monitor (SMON)

SMON मुख्य रूप से निम्न कामों को पूरा करता है:

1. यदि Previous Database Shutdown में एक System-Wide Checkpoint Include ना हुआ हो, तो ये Process System Recovery का काम करता है।
2. Data File में Free Spaces के Adjacent Extents को Combine करता है। इस Action को Tablespace Level पर **PCTINCREASE = 0** Setting द्वारा Turn Off किया जा सकता है, जो कि Tablespace के Default Storage Parameter में होता है।

Process Monitor (PMON)

जब Server Process Fail होता है, तब Process Monitor SGA को Clean कर देता है। विशेष रूप से PMON Failed Session के Transaction को Roll Back कर देता है और Transaction से Associated किसी भी Resource को Lock कर देता है।

Archiever (ARCH)

ये Process हालांकि Mandatory नहीं होता है, फिर भी हम इसके बिना किसी Database को Media Failure जैसे कि Hard Disk के Damage होने की स्थिति में पूरी तरह से Recover नहीं कर सकते हैं।

हालांकि LGWR Process Data को Redo Log Buffer से Disk पर Write करता है, लेकिन इन Disk Files की Size व संख्या Limited होती है। इन्हें Circular Resources की तरह Use किया जाता है, ताकि जब Memory पूरी तरह से Fill हो जाए, तब LGWR फिर से शुरू से Memory में Data Hold कर सके।

जब ऐसा होता है, तो पिछला Data नए Data से Over Write हो जाता है और पुराना Data Lost हो जाता है। इस Setup को **NOARCHIVELOG Mode** कहा जाता है, जो कि Oracle का Default Mode होता है। Archive Process Data को Online Redo Logs File से Destination Directory में Write करता है।

Online Redo Log Files की Overwriting को Avoid करने के लिए हमें Archiver Process को Start करना पडता है। इसके बाद जितनी बार भी File Fills Up हो जाती है, ये Process Data को Archive Area में Copy कर देता है। हमें इस बात के लिए भी Ensure होना होता है कि Database ARCHIVELOG Mode में है।

Archiver Process कुछ हद तक Control Files को Up-To-Date रखने के लिए भी जिम्मेदार होता है, हालांकि ये काम कुछ अन्य Processes जैसे कि LGWR के साथ Shared होता है, जो कि Checkpoints व Log Sequence Information के साथ Control Files को Update करते हैं।

इनके अलावा Server Processes भी उस स्थिति में Control Files को Update करते हैं, जब Table Space को **ALTER DATABASE** Command द्वारा Add या Alter किया जाता है। यदि Archive Destination भर जाता है और Archiving को Enabled किया गया होता है, तो Oracle Online Redo Logs को Overwrite नहीं करता है, बल्कि अक्सर Hang हो जाता है।

Server Processes

Server Processes को इस तरह के नाम इसलिए दिए गए हैं, क्योंकि Server पर स्थित ये ही वे Processes हैं, जो User Requests के साथ Deal करने के लिए Design किए गए हैं। फिर भी अक्सर इन्हें गलत तरीके से User Processes के रूप में Identify किया जाता है। जबकि वास्तव में Visual Basic जैसे Actual Applications ही User Processes होते हैं। हर User Application जैसे कि Visual Basic Program द्वारा कोई Session Create करने पर इस Client की Request को Handle करने के लिए Oracle में Server Processes Create किया जाता है। ये Setup ही Users के लिए सबसे Basic स्तर का Setup होता है। Oracle को ज्यादा Users के बीच Scalable बनाने के लिए हम Multi-Threaded Server Option को Use कर सकते हैं, जो कि Dispatcher Process के साथ Shared Server Processes को Use करते हुए User की Request को पूरा करता है। एक Server Process के मुख्यतः निम्न काम होते हैं:

1. SQL Statements को Parse करना और उन्हें Execute करना।
2. Generate होने वाले Resultset को User के Client Program पर Return करना।
3. जब Data की जरूरत हो और Required Data, Data Buffers Cache में उपलब्ध ना हो, तब Data Buffer में Data Blocks को Read करना।
4. Data में किए गए Changes को Redo Log Entries के रूप में Redo Log Buffer में Write करना।

Listener Process

Listener Processes कोई Required Background Process नहीं होता है, लेकिन यदि कोई Database को Network द्वारा Access करना चाहता है, तो इस Process की जरूरत होती है। हम इस Process को Start करके Network से आने वाली Connection Requests को "**Listen**" करते हैं। ये Web Server के HTTP Listener की तरह ही काम करता है। Connection स्थापित करने के बाद Listener Process, User व Oracle के बीच Communication को Handle करने के लिए Server Processes Create करता है।

Database Files

हालांकि एक Oracle Instance **Memory Structure** व उन **Processes** से बना होता है, जो Memory Structures को Manage करते हैं, फिर भी Physical Database Files वे Files होती है, जो System को Useable बनाती हैं। ये Files निम्न कारणों से Database से Associated सभी प्रकार के Data को Hold करती हैं:

1. Database को ये Allow करने के लिए कि वह Physical Memory में Store हो सकने वाले Data की क्षमता से ज्यादा Data के साथ Dealing कर सके। एक बात ध्यान रखें कि Operating System के Swapping व Paging को किसी भी कीमत पर Avoid किया जाना होता है, ताकि Oracle की Performance पर कोई विपरीत असर ना पडे।

2. Database Transaction की Recovery को Allow करने के लिए, फिर चाहे वह Recovery Failure Point से हो या किसी Previous Point से हो।

किसी Real-Life Database में ये Files ही सबसे ज्यादा महत्वपूर्ण होती हैं, क्योंकि इसी के आधार पर पूरा System काम करता है। Oracle में मुख्य रूप से चार तरह की Database Files होती हैं:

1. **Control Files**
2. **Initialization (Parameter) Files**
3. **Online Redo Log Files**
4. **Data Files**

ये चारों ही Files Oracle को ठीक तरीके से Run करने के लिए जरूरी व महत्वपूर्ण होती हैं।

Control Files

ये एक छोटी सी File होती है, जो Current Database Structure को Describe करती है। हम इस File को एक Online Database Header File के रूप में देख सकते हैं। ये एक Binary File होती है, इसलिए हमें इस File को Edit करने की कोशिश नहीं करनी चाहिए। इस File को Oracle Instance द्वारा Start-Up के समय Read किया जाता है और इस File को तब तक Up-To-

Date रखा जाता है, जब तक कि Oracle के Instance को Shut Down नहीं किया जाता। इस File के निम्न Purpose होते हैं:

1 Database का नाम रखने के लिए।
2 Database Files व Log Files को Identify करने के लिए।
3 Recovery के लिए जरूरी जानकारियों जैसे कि Checkpoints आदि को Synchronize करने के लिए, जिनका प्रयोग Database की Recovery के लिए किया जाता है।

जितनी बार भी Database Structure Change किया जाता है, उदाहरण के लिए जब हम किसी Table को Create या Drop करते हैं या किसी Log File को Add करते हैं, तब इस Physical Change के साथ ही Control File भी Update हो जाती है। Oracle में Default रूप से केवल एक Control File को Configure किया जाता है, लेकिन हमें कम से कम दो Control Files को Configure करके रखना चाहिए, ताकि Crash की स्थिति में Recovery निश्चित हो।

Parameter Files

Parameter File में Oracle Instance के Start-Up Options से सम्बंधित सभी Information होती हैं। इसे तब Read किया जाता है जब Database Start होता है और ये Microsoft Windows की Initialization File **(*.ini)** Files के समान ही होती है, जिसके आधार पर एक Oracle Instance Start होता है। हम इस File को Edit कर सकते हैं और Parameters को Manually Set कर सकते हैं। यदि हम हमारी Settings को Change करते हैं, तो उन Settings का Effect तब तक नहीं होता है, जब तक कि हम Oracle Instance को Restart ना करें।

एक Typical Parameter File को आगे दर्शाया गया है। By Default इसका नाम **init*SID*.ora** होता है, जबकि ***SID*** Start किए जाने वाले Instance का नाम है, जिसे हम Oracle के Installation के समय Specify करते हैं।

यदि हमने हमारे SID का नाम SCT रखा हो, तो हमें इस File को Open करने के लिए initSCT.ora नाम की File को "**..\Oracle\Ora81\DATABASE**" Path से Open करना चाहिए। जब हम इस File को Notepad का प्रयोग करके Open करते हैं, तब हमें इस File में निम्नानुसार Statement दिखाई देता है:

```
IFILE='C:\Oracle\admin\SCT\pfile\init.ora'
```

Database की पूरी Initialization Information को हम **init.ora** नाम की File से प्राप्त कर सकते हैं, जो कि हमें "**C:\Oracle\admin\SCT\pfile**" Path पर प्राप्त होती है। इस File में Data निम्नानुसार हो सकते हैं:

```
#
```

```
# Copyright (c) 1991, 1998 by Oracle Corporation
#
###########################################################################
# Example INIT.ORA file
#
# This file is provided by Oracle Corporation to help you customize
# your RDBMS installation for your site.  Important system parameters
# are discussed, and example settings given.
#
# Some parameter settings are generic to any size installation.
# For parameters that require different values in different size
# installations, three scenarios have been provided: SMALL, MEDIUM
# and LARGE.  Any parameter that needs to be tuned according to
# installation size will have three settings, each one commented
# according to installation size.
#
# Use the following table to approximate the SGA size needed for the
# three scenarios provided in this file:
#
#                -------Installation/Database Size------
#                 SMALL         MEDIUM         LARGE
#  Block        2K    4500K          6800K         17000K
#  Size         4K    5500K          8800K         21000K
#
# To set up a database that multiple instances will be using, place
# all instance-specific parameters in one file, and then have all
# of these files point to a master file using the IFILE command.
# This way, when you change a public
# parameter, it will automatically change on all instances.  This is
# necessary, since all instances must run with the same value for many
# parameters. For example, if you choose to use private rollback segments,
# these must be specified in different files, but since all gc_*
# parameters must be the same on all instances, they should be in one file.
#
# INSTRUCTIONS: Edit this file and the other INIT files it calls for
# your site, either by using the values provided here or by providing
# your own.  Then place an IFILE= line into each instance-specific
# INIT file that points at this file.
#
# NOTE: Parameter values suggested in this file are based on conservative
# estimates for computer memory availability. You should adjust values upward
# for modern machines.
#
```

```
###########################################################################

db_name = "SCT"

db_domain = Domain

instance_name = SCT

service_names = SCT.Domain

db_files = 1024

control_files =
      ("C:\Oracle\oradata\SID\control01.ctl",
      "C:\Oracle\oradata\SID\control02.ctl",
      "C:\Oracle\oradata\SID\control03.ctl")

open_cursors = 100
max_enabled_roles = 30
db_file_multiblock_read_count = 8

db_block_buffers = 2048

shared_pool_size = 4194304

large_pool_size = 614400
java_pool_size = 0

log_checkpoint_interval = 10000
log_checkpoint_timeout = 1800

processes = 50

parallel_max_servers = 5

log_buffer = 32768

#audit_trail = true  # if you want auditing
#timed_statistics = true  # if you want timed statistics
max_dump_file_size = 10240  # limit trace file size to 5M each

# Uncommenting the line below will cause automatic archiving if archiving has
# been enabled using ALTER DATABASE ARCHIVELOG.
```

```
# log_archive_start = true
# log_archive_dest_1 = "location=C:\Oracle\oradata\SCT\archive"
# log_archive_format = %%ORACLE_SCT%%T%TS%S.ARC

# If using private rollback segments, place lines of the following
# form in each of your instance-specific init.ora files:
#rollback_segments = ( RBS0, RBS1, RBS2, RBS3, RBS4, RBS5, RBS6 )

# Global Naming -- enforce that a dblink has same name as the db it connects to
global_names = true

# Uncomment the following line if you wish to enable the Oracle Trace product
# to trace server activity.  This enables scheduling of server collections
# from the Oracle Enterprise Manager Console.
# Also, if the oracle_trace_collection_name parameter is non-null,
# every session will write to the named collection, as well as enabling you
# to schedule future collections from the console.
# oracle_trace_enable = true

oracle_trace_collection_name = ""
# define directories to store trace and alert files
background_dump_dest = C:\Oracle\admin\SCT\bdump
#Uncomment this parameter to enable resource management for your database.
#The SYSTEM_PLAN is provided by default with the database.
#Change the plan name if you have created your own resource plan.# resource_manager_plan = system_plan
user_dump_dest = C:\Oracle\admin\SCT\udump

db_block_size = 8192

remote_login_passwordfile = exclusive

os_authent_prefix = ""

distributed_transactions = 500
compatible = 8.0.5
sort_area_size = 65536
sort_area_retained_size = 65536
```

Online Redo Log Files

हालांकि Data में किया जाने वाला कोई भी Change Memory के Redo Log Buffer में होता है, फिर भी Instance के Fail होने की स्थिति में Backup लेना जरूरी होता है। इस काम को Log

Writer Process द्वारा पूरा किया जाता है, जो कि Redo Log Buffers से Data को Read करके SGA के Online Redo Log Files में Store कर देता है। ये Redo Log Files फिर से Circular Storage Areas की तरह Treat होती हैं और यदि Database **ARCHIVELOG** Mode में ना हो, तो ये Continuously Overwrite होती रहती हैं।

Oracle में Default रूप से दो Log Files होती हैं, जिन्हे हम **redo_1a** व **redo_2a** नाम दे सकते हैं। इन्हें Separate Groups का Member माना जा सकता है, जो कि हमारे Case में हर Group में केवल एक File या Member के रूप में है।

वास्तव में हमें Performance को बनाए रखने के लिए कम से कम एक और Group की जरूरत रहती है, इसलिए हम एक और Group Create कर रहे हैं, जिसमें **redo_3a** नाम की एक Member File है। अब Log Writer redo_1a, redo_2a व redo_3a Log Files में Data को Write करता है।

इसके बाद ये फिर से redo_1a में Writing करने लगता है, और इसी समय इसके Data को Archive Destination पर Copy कर दिया जाता है। जो Group Write किया जा रहा होता है, उसे **Current Group** कहते हैं।

इस तरह से सारांश में कहें तो Log Writer पहले एक Log Group में Data को Write करता है फिर क्रम से आगे के Groups में Data को Write करता है। हर Log Group में एक या एक से ज्यादा Members हो सकते हैं और एक Group के सभी Members Different Disks पर Store होने चाहिए, ताकि Crash की स्थिति में किसी भी अन्य Disk से Data को Recover किया जा सके।

Important Redo Log Files की Losing के Risk को कम करने के लिए, जो कि मुख्य रूप से Disk Crash होने की स्थिति में Database की Recovery के लिए जिम्मेदार होती हैं, निम्न Guidelines का पालन किया जाता है:

1. हर Group में एक से ज्यादा File को Add करना चाहिए, जो कि Data को Mirror करे। हम redo_1a, redo_2a व redo_3a को तीनों Groups के नए Members के रूप में Add कर सकते हैं।
2. ये बात निश्चित कर लेनी चाहिए कि सभी Mirrored Files को Separate Disks पर Store किया गया हो, ताकि Failure की स्थिति में हमारे पास एक से ज्यादा स्थानों पर Backup हो।
3. हमें ये निश्चित कर लेना चाहिए कि Archiving को Enable किया गया है।
4. इस बात को निश्चित कर लेना चाहिए कि विभिन्न Redo Logs Different Disks पर हों, ताकि Database के Performance पर प्रभाव ना पडे।

Data Files

ये Physical Files होती हैं, जहां पर Data को Store किया जाता है और इसे हम Data का Physical Representation भी कह सकते हैं। हमें इन Files का Backup समय–समय पर लेते रहना चाहिए।

हर वह Data File जिसका Extension सामान्यतया **.ora** या **.dbf** होता है, वह Native Operating System File Structure के आधार पर Data Blocks का एक समूह होता है। इस Base Level के उपर Oracle अपने Logical Structure को Impose करता है। इस तरह से Oracle विभिन्न Platforms पर अपने Database को Run करता है, जो कि बहुत ही मामूली रूप से Native Operating System पर निर्भर होता है।

Logical Structure

Oracle अपना Logical Database Structure उस Disk Space से बनाता है, जो Operating System Oracle को प्रदान करता है। ध्यान रखें कि Database शब्द वास्तव में Oracle Database के File Part पर Apply होता है ना कि Memory या Process Based Part पर।

Oracle अपने Logical Structure को Tablespaces के आधार पर Maintain करता है और हर Tablespace के लिए Files को Resource की तरह Use करता है। Oracle के Logical Structure के विभिन्न हिस्सों को हम निम्नानुसार समझ सकते हैं:

Database

Database Data का एक Total Collection होता है, जो कि एक Separate Unit की तरह होता है। Physically ये **Data Files** की एक Series होती है, जबकि Logically ये **Tablespaces** का Group होता है।

Tablespace

Tablespace Storage का एक Logical Unit होता है, जिसे किसी विशेष Purpose के लिए DBA द्वारा Setup किया जाता है। **SYSTEM** Tablespace Oracle के लिए सबसे जरूरी Tablespace होता है और Oracle स्वयं ही इसे Automatically Create करता है। Users इस Tablespace का प्रयोग **Dictionary Information** व **System Definitions** के लिए करता है।

हमें Sorting के काम के लिए Temporary Memory Area में **TEMPORARY** नाम का एक Tablespace भी Add करना चाहिए। हमें कम से कम एक और Tablespace Create करना

होता है, जिसका प्रयोग User Data के लिए किया जाना होता है। यदि हम User Data के लिए एक अलग Tablespace Create नहीं करते हैं, तो हमें SYSTEM Tablespace को ही User Data Hold करने के लिए Use करना पडता है।

हमें सामान्यतया सभी Applications के लिए एक Tablespace Create करना चाहिए और हर Index के लिए भी एक अलग Tablespace Create करना चाहिए। हमें Rollback Segments के Data को Hold करने के लिए भी एक Tablespace Create करना चाहिए।

इस Special Tablespace को Create करने का कारण ये है कि यदि किसी Tablespace में Online Rollback Segment के Data हों, तो उस Tablespace को Offline में Use नहीं किया जा सकता है। इसलिए Rollback Segments को सभी अन्य Segments से अलग यानी **Isolated** रखा जाना जरूरी होता है।

हर Tablespace में Physically एक या एक से ज्यादा Data Files होती हैं, जिन्हें हम उस स्थिति में समान Disk पर Hold करके रख सकते हैं, जब हमारे पास अधिक Disks ना हों। Logically एक Tablespace Segments का एक Group होता है, जो कि Database के Internal Structures को Organize करने का काम करता है। Tablespaces को Offline भी बनाया जा सकता है या इसे बाकी के Database को बिना प्रभावित किए हुए स्वतंत्र रूप से **Backed Up** भी किया जा सकता है। फिर भी SYSTEM Tablespace को किसी भी हालत में Offline नहीं बनाया जा सकता।

Segments

हर Segment एक अमुक प्रकार के Database Structure को Represent करता है। हर Segment Extents के एक समूह का बना होता है। Segments कई प्रकार के होते हैं, जिन्हें निम्नानुसार समझा जा सकता हैः

1 *Data Segments*
Oracle के Database में हर Non-Clustered Table के लिए एक Data Segment होता है।

2 *Cluster Segments*
हर Cluster के लिए एक Cluster Segment होता है। ये Cluster उन एक या एक से ज्यादा Tables को Hold करने का काम करता है, जिन्हें Database Designer ने आपस में Physically नजदीक रखने के लिए Decide किया है, ताकि Tables की Performance को Improve किया जा सके।

3 *Rollback Segments*
Oracle में हमेंशा कम से कम एक या अधिक Rollback Segment होता है, जो कि Oracle के **SYSTEM** Tablespace में होता है। इन Segments को इस Information को Hold करने के लिए Use किया जाता है, यदि जरूरत हो तो किसी Transaction को किस प्रकार से Roll Back किया जा सकता है।

4 *Index Segments*
इनका प्रयोग किसी Individual Indexes को Hold करने के लिए किया जाता है।

5 *Temporary Segments*
इस Area को Temporary Work Area के रूप में Use किया जाता है, जिसमें Sorting जैसे किसी काम को पूरा किया जाता है। ये Segments सामान्यतया Temporary Type के Tablespace में उपस्थित होते हैं।

हमें हमेंशा एक Temporary Tablespace को Create करना चाहिए और इसे **CREATE USER** Command में Specify करना चाहिए। यदि हम स्वयं ये Segment Create नहीं करते हैं, तो Oracle स्वयं SYSTEM नाम के Tablespace के Segments को Use करके इससे सम्बंधित काम को पूरा करता है।

Extents

Extent किसी Data File के Data Blocks का एक Contiguous Group होता है, जो किसी Particular Type की Information को Hold करने का काम करता है। Extents आपस में Logically Group होकर एक Segment के रूप में Identify होते हैं।

Database Blocks

Database Blocks किसी Database के Logical Storage का Lowest Level होता है। हम Database Blocks का प्रयोग करके Schema Objects जैसे कि Table, View आदि को Create करते समय उनके लिए Storage Requirement को Define करते हैं। हर Logical Database Block को एक या एक से ज्यादा Physical Operating System Blocks द्वारा Represent किया जाता है।

ORACLE

DATA CONCURRENCY

&

CONSISTENCY

Data Concurrency and Data Consistency

जब कभी भी किसी Database को एक से ज्यादा Users Use करते हैं, तब ये जानना जरूरी होता है कि जब दो Users समान Table के समान Row को एक ही समय पर Access करना चाहते हैं, तब Oracle उन्हें कैसे Manage करता है। ये बात Access या Oracle किसी भी Database पर लागू होती है। एक Database इस Multi-User Issue को कितनी अच्छी तरह से Deal करता है, ये बात एक Developer के लिए बहुत ही महत्वपूर्ण होती है। हालांकि Oracle इस समस्या को स्वयं ही Handle करता है और हमें इस सम्बंध में किसी प्रकार की कोई चिन्ता करने की जरूरत नहीं होती है। Multi-Users Issue के साथ निम्न Oracle Functions Deal करते हैं:

Data Concurrency

Different Users द्वारा समान Data को Update करने की प्रक्रिया को Manage करने की Ability को **Data Concurrency** कहा जाता है। Oracle में ये Management एक विशेष Locking Schema पर आधारित होता है। यदि हम किसी Table के Row को Update कर रहे होते हैं, तो Oracle उस Row को Lock कर देता है और कोई अन्य User तब तक उस Data को Access नहीं कर सकता है, जब तक कि उस Row के साथ हमारा Transaction Complete नहीं हो जाता है।

Data Consistency

किसी **SELECT** Statement द्वारा एक Consistence Result प्राप्त करने की क्षमता को **Data Consistency** कहा जाता है। इसका मतलब ये होता है कि यदि हम किसी Data को Use कर रहे हैं तो किसी दूसरे User द्वारा उसी Data को Update करने का प्रभाव हमारे Result पर नहीं पडता है। इस प्रक्रिया में वह Change भी शामिल होता है, जिसे किसी अन्य User ने Database में Apply तो किया है, लेकिन उस Change को अभी तक Commit नहीं किया है। इस Function को **Statement Level Read Consistency** कहा जाता है।

Locking Strategies

जब कोई User किसी Data को Change करना चाहता है, तब Oracle Affect होने वाले Rows पर एक Lock लगा देता है। ऐसा करने के कारण कोई भी दूसरा User Same Rows को जब तक के लिए Modify नहीं कर पाता है, जब तक कि पहला User उन Affected Rows के साथ अपना Transaction Complete नहीं कर लेता है। इन Affected Rows को ***Destructive Interference*** कहा जाता है।

इसके अलावा Oracle Table पर एक **Share Lock** भी Place करता है। ये Lock किसी भी अन्य User को Data Definition Language (DDL) Statements द्वारा Database के Structure को Alter करने या किसी Table को Drop करने से रोक देता है।

Oracle किसी Table में Row Level पर Locking करने में सक्षम है, जबकि कई अन्य Databases केवल **Page-Level** Locking की सुविधा ही Provide करते हैं, जिसमें बहुत सारे Rows Lock हो जाते हैं, फिर चाहे एक ही Row को Update क्यों ना किया जा रहा हो।

Consistency Achievement

Oracle में Consistency को एक बहुत ही Clear तरीके से प्राप्त किया जाता है। जब हम Oracle को कोई **SELECT** Statement Submit करते हैं, तो Oracle Request के Start Time को Note कर लेता है और उसके बाद Data को Retrieve करता है। SELECT Statement के Start होने के बाद यदि कोई Row Change होता है, तो Consistency Rule का Violation होता है।

इस समस्या के समाधान के लिए Oracle Changed Data को फिर से Rollback करके उसी स्थिति में Store कर देता है, जिस स्थिति में Data **Rollback Segment** में पहले था। इस तरह से Rollback Segment में पिछला Data ही Stored रहता है। जब SELECT Statement Changed Row पर पहुंचता है, Oracle Directly उसे Rollback Segment पर भेज देता है, ताकि पिछला Record ही प्राप्त हो।

Schemas

Developers की नजर में Database एक ऐसा System होता है, जो User को विभिन्न प्रकार के Database Objects Provide करता है। Developer का View Schemas पर आधारित होता है। Oracle में जितने भी Users होते हैं, उन सभी का **एक** Schema होता है, जो कि Database के विभिन्न Objects को Group करने का एक तरीका होता है, ताकि हर User अपने Database Objects को अन्य Users से सुरक्षित रख सके।

Users

बिना एक **User Name** व **Password** के, हम Oracle Database में Log On नहीं हो सकते हैं। Oracle अपने हर User व उसके Encrypted Password का ध्यान रखता है। जब हम एक नया User Create करते हैं, उस User से Associated एक Schema Create होता है और User उस Schema के किसी भी Object को Access करने में सक्षम होता है। फिर भी एक नया

User Database के किन Objects को Access कर सकने में सक्षम होगा, उस User को इस बात से सम्बंधित कुछ अधिकार दिए जाते हैं। हर नया User अपनी उन Limitations के अन्तर्गत ही किसी Database को Access करने में सक्षम होता है। हर User के लिए Oracle में एक Session Create करना व उसे Tablespace के कुछ हिस्से को Access करने की सुविधा देना **Privileges Allow** करना कहलाता है। Tablespace के इस Allowed Memory Space को **Quota** कहा जाता है।

Tables

Tables किसी भी Data के Fundamental Storage Unit होते हैं। इनमें Rows व Columns होते हैं। Schema के हर Table का एक Unique नाम होता है। Table के हर Column का भी एक Unique नाम होता है जो कि एक Data Type जैसे कि NUMBER या DATE से Associated होता है। Tablespace में जितनी भी Tables होती हैं, वे एक या एक से ज्यादा Data Files के बीच बिखरी हुई हो सकती हैं। Table के Extents को Locate करने के लिए **dba_extents** व **dba_data_files** Views का प्रयोग किया जा सकता है।

Indexes

Indexes एक दूसरा Schema Object होता है, जो अपने स्वयं के Segment में Store होता है। Indexes Create करने का Most Common तरीका **B-Trees** को Use करना होता है।

Clusters

Clusters विभिन्न Database Tables को Store करने का एक Alternative तरीका होता है। Clusters का प्रयोग करके हम Data के Disk पर Physically Store होने के तरीके को Control कर सकते हैं, ताकि हम Database की Performance को बढा सकें और Data के Streams को उस तरीके से प्राप्त कर सकें, जिस तरीके से Data Index File में Store होते हैं।

जब हम Cluster Segment Create करते हैं, तब हम Cluster Column को Identify करते हैं। जब हम Tables Create करते हैं, तब हम ये Indicate करते हैं कि हम उन Tables को किस Cluster में रखना चाहते हैं। जब हम किसी Table को किसी Cluster में Add करते हैं, तो Cluster व Table दोनों में एक समान Columns होने जरूरी होते हैं। Table की Rows वास्तव में Physically Cluster Index Order में Store होते हैं, इसलिए उन्हें Hashing Algorithm के आधार पर भी Store किया जा सकता है।

हम किसी Cluster में कई Tables Store कर सकते हैं, लेकिन हमें ऐसा तभी करना चाहिए, जब वे सभी Tables एक साथ Use की जाती हों। सामान्यतया Clusters का प्रयोग नहीं किया जाता है, क्योंकि Performance का फायदा केवल तभी होता है, जब Clusters में स्थित Tables Static हों।

Sequence Generators

ये एक ऐसा Built-In तरीका होता है, जिसका प्रयोग Unique Sequential Numbers Create करने के लिए किया जाता है। इसका मुख्य Purpose Tables को Unique Keys Provide करना होता है, क्योंकि Oracle में **Autonumber** जैसी कोई सुविधा नहीं होती है, जैसी की MS-Access में होती है। इसका प्रयोग बिना किसी परेशानी के एक से ज्यादा Users द्वारा किया जा सकता है, जबकि किसी दूसरे तरीके से ये Number Create करने पर Locking से सम्बंधित समस्या का सामना करना पड सकता है।

Procedures

Oracle हमें ऐसे Procedural Codes लिखने की सुविधा देता है, जिन्हें हम Database में Store करके रख सकते हैं। इन Procedures को PL/SQL Language में लिखा जाता है। ये हमें Procedures, Functions व Database Triggers लिखने की सुविधा देता है। Triggers वे Code Blocks होते हैं, जो Table पर किसी Action के Perform होने पर Automatically Execute हो जाते हैं।

Views

ये एक ऐसा Predefined तरीका होता है, जिसके द्वारा हम उन एक या एक से अधिक Joined Tables के Data को देख सकते हैं, जिन्हें ***Base Tables*** कहते हैं। ये MS-Access की Query के समान होते हैं। Views का प्रयोग Base Tables की Information को Hide करने के लिए किया जा सकता है, जिसके द्वारा Users को Base Tables के केवल कुछ Columns को ही Access करने की सुविधा Provide किया जाता है।

इसका एक सबसे बडा फायदा ये है कि यदि Base Tables में नए Files Create किए जाए, तो भी Views को Modify करने की जरूरत नहीं होती है। वे बिना Modify किए हुए भी ठीक उसी प्रकार से काम करते हैं, जिस तरह से Base Tables में नए Columns को Add करने से पहले करते थे। इनका प्रयोग करके कई Tables को Join करने की Complexity को भी Hide किया जा सकता है।

हम Views का प्रयोग भी ठीक उसी तरह से कर सकते हैं, जिस तरह से Base Tables का करते हैं। यानी हम Views पर भी उसी तरह से SELECT, INSERT, UPDATE व DELETE Commands को Apply कर सकते हैं, जिस तरह से Base Tables पर करते हैं, हालांकि ऐसा करने पर हमें कुछ Restrictions को ध्यान में रखना होता है।

Synonyms

Synonym किसी भी Schema Object का एक दूसरा नाम यानी Alias होता है। Oracle में हम किसी Table, View, Procedure या Sequence को एक दूसरा Alias भी प्रदान कर सकते हैं। Synonyms Public या Private हो सकते हैं और इनका प्रयोग Schema के किसी Object के Names को Security के कारणों से Hide करने के लिए किया जा सकता है। Synonyms को Microsoft का **ODBC** Support नहीं करता है, इसलिए इसे हम सामान्य तरीके से Visual Basic जैसे किसी Client Software में Use नहीं कर सकते हैं।

हालांकि Oracle की सभी Internal Details को पूरी तरह से समझाना काफी मुश्किल है। साथ ही जब हम Oracle को Practically Development के लिए Use करते हैं, तब हमें इन Details की जरूरत भी नहीं होती है। फिर भी हमें इस बात की General जानकारी होनी चाहिए कि Oracle क्या और कैसे कर रहा है। Oracle के काम करने की पूरी प्रक्रिया को हम निम्नानुसार समझ सकते हैं:

1 सबसे पहले DBA Database के Start-Up Procedure को Initiate करता है, जो कि
   - Initialization File से Parameters को Read करता है।
   - SGA को Memory Allocate करता है।
   - Required Processes को Start करता है।
   - Control Files को Open व Read करता है।
   - Database की Data Files को General Access के लिए Open करता है।
2 फिर DBA एक Listener Process Start करता है और User-Connection के लिए Request करता है।
3 एक Visual Basic Connection Database से ODBC Network या OLE Oracle Object द्वारा Database से Connection स्थापित करता है।
4 फिर Listener Process एक Server Process को User के SQL Requests को Handle करने के लिए Dispatch करता है।

5 Visual Basic Application एक SQL Statement को Database पर Pass करता है।
6 SQL Statements के लिए Shared Pool का एक Area Allocate किया जाता है।
7 यदि Database Buffer Cache में कोई Data ना हो, तो Required Data को इसमें Pull किया जाता है।

8 Memory में किए गए किसी भी Change व Store किए गए किसी भी Change की Entry Redo Log Buffer में की जाती है।

9 Control फिर से Visual Basic Client Application को एक Appropriate Result के साथ Return होता है।

10 जब कोई Criteria मेल करता है, तब Database Writer Process Data Changes को फिर से Disk पर Write कर देता है।

11 जब Changes को Commit किया जाता है, तब Log Writer इन Changes को Redo Log Files में Write कर देता है।

इस तरह से एक Visual Basic Client व Oracle Server के बीच का Communication Perform होता है।

How to Get Complete PDF EBook

आप **Online Order** करके **Online** या **Offline** Payment करते हुए इस Complete EBook को तुरन्त Download कर सकते हैं।

Order करने और पुस्तक को Online/Offline Payment करते हुए खरीदने की पूरी प्रक्रिया की विस्तृत जानकारी प्राप्त करने के लिए आप **BccFalna.com** के निम्न Menu Options को Check Visit कर सकते हैं।

How to Make Order

How to Order?

How to Buy Online

How to Pay Online using PayUMoney

How to Pay Online using Instamojo

How to Pay Online using CCAvenue

How to Buy Offline

How to Pay Offline

Bank A/c Details

जबिक हमारे Old Buyers के **Reviews** भी देख सकते हैं ताकि आप इस बात का निर्णय ले सकें कि हमारे Buyers हमारे PDF EBooks से कितने Satisfied हैं और यदि आप एक से अधिक EBooks खरीदते हैं, तो **Extra Discount** की Details भी Menubar से प्राप्त कर सकते हैं।