Administración de Oracle

Sumario

O dicionario de datos

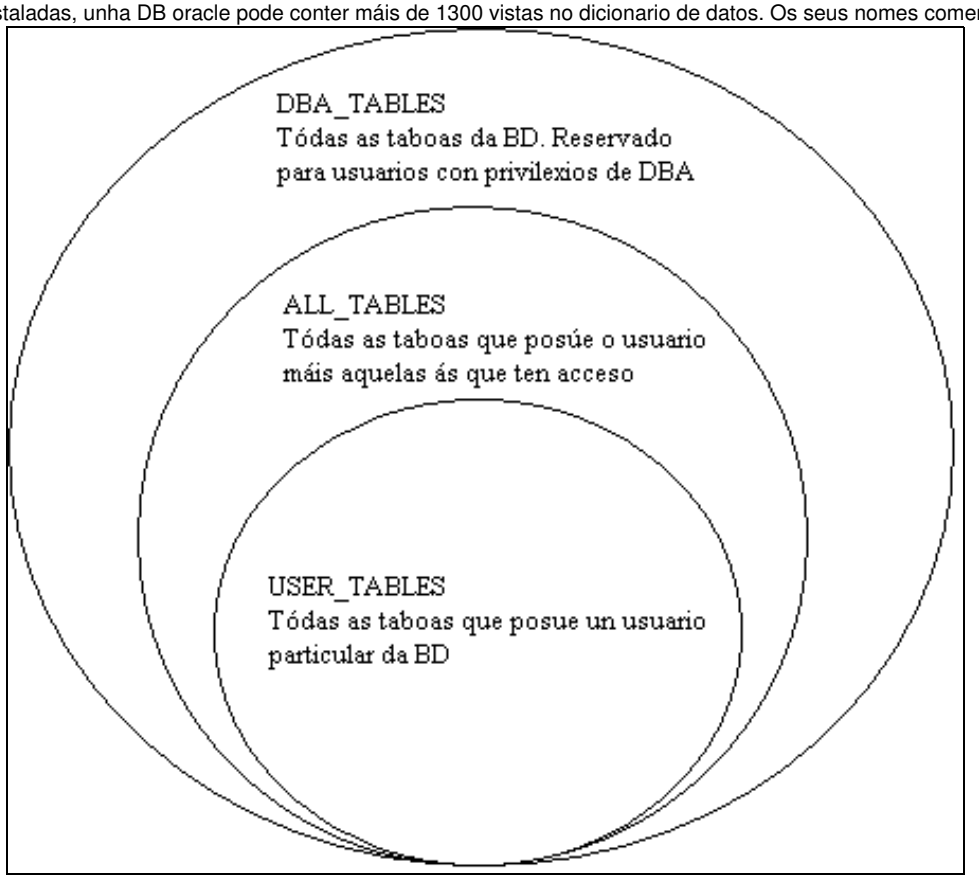
Vistas do dicionario de datos

É un conxunto de táboas de referencia de so lectura que conterán información sobre a propia BD. O dicionario de datos conterá información acerca de:

- A definición de todos os obxectos dentro da BD (táboas, índices, vistas, sinónimos, secuencias, etc)
- Cando espacio esta asignado a cada obxecto
- Valores por defecto de cada columna
- Información das restricións de integridade
- Información sobre os usuario da BD
- Privilexios e roles de cada usuario
- Información sobre auditoría
- Outra información de caracter xeral

O propietario do dicionario de datos será o usuario SYS, pero estas táboas non deberían modificarse nunca directamente xa que podemos comprometer a integridade dos datos.

Dependendo das características instaladas, unha DB oracle pode conter máis de 1300 vistas no dicionario de datos. Os seus nomes comenzarán sempre por DBA_, ALL_, y USER_.



Exemplos de vistas do dicionario de datos

Vista	Descrición
DBA_TABLES	Mostra os nomes e a información sobre o almacenamiento físico de todas as taboas da base de datos
DBA_USERS	Mostra información acerca de todos os usuarios da BD
DBA_VIEWS	Mostra información de todas as vistas da BD
DBA_TAB_COLUMNS	Mostra os nomes e tipos de datos de todas as columnas de cada táboa na BD

Vista dinámicas de rendemento

Oracle manterá un conxunto virtual de táboas que rexistrarán a actividade da BD. Non son verdadeiras táboas, e non deberían ser accedidas pola maioría dos usuarios. O propietario destas vistas tamén será o usuario SYS.

Dependendo das características instaladas pode haber ata 350 vistas de rendemento. A maior parte de elas comenzarán con V$. A continuación mostramos algunhas como exemplo:

Exemplos de vistas dinámicas de rendemento

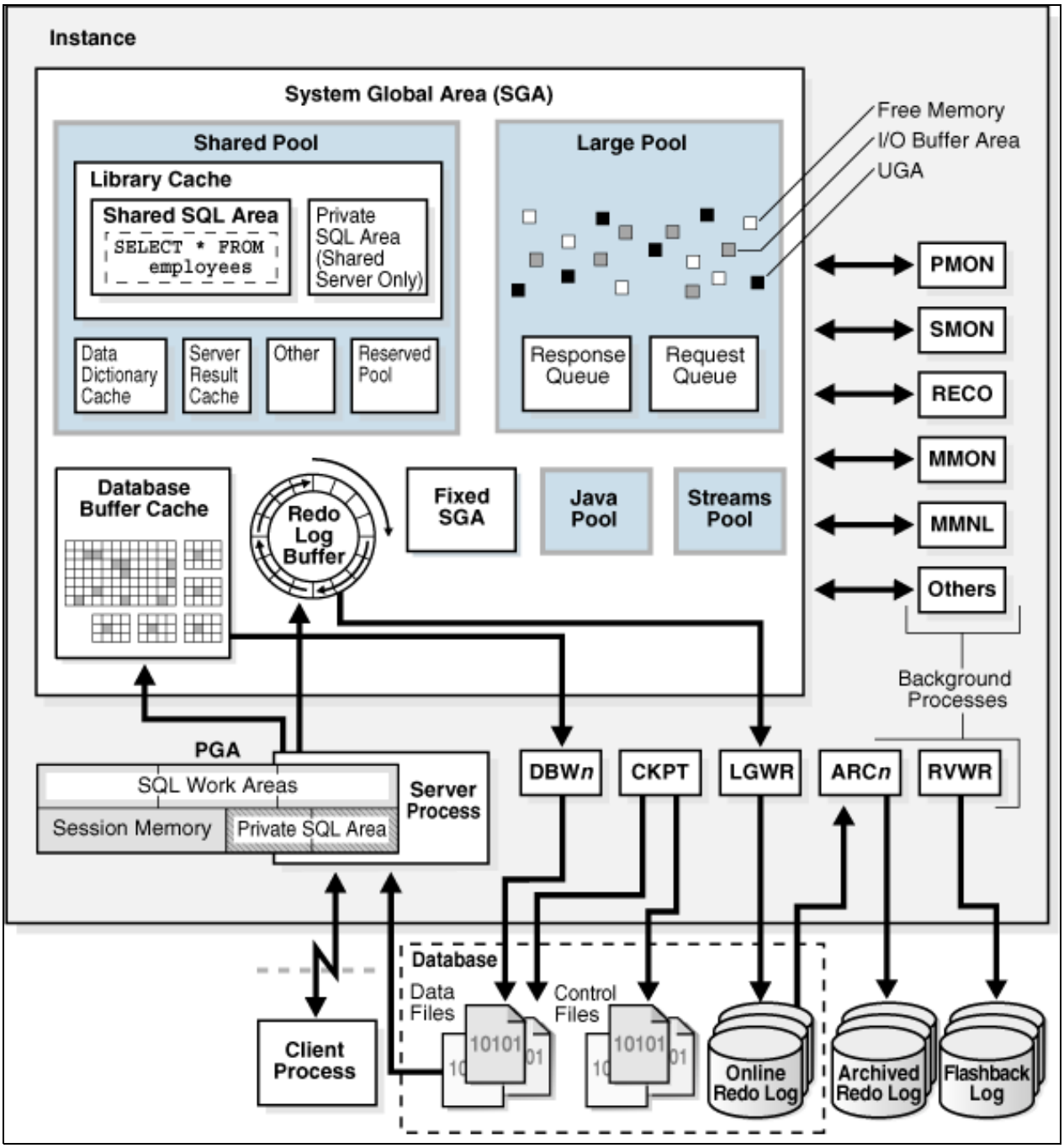
Vista	Descrición
V$DATABASE	Conten información acerca da BD: nome, cando foi creada, etc
V$VERSION	Mostra a versión do software usado
V$OPTION	Compoñentes opcionais instalados na BD
V$SQL	Mostra información acerca das sentencias SQL que os usuarios están executando.

Arquitetura Oracle

A arquitectura dun servidor oracle pode dividirse en 3 categorías:

- Procesos de usuario.
- Estruturas lóxicas na memoria as que en conxunto podemos chamar instancia.
- Estruturas físicas en disco as que en conxunto podemos chamar base de datos.

A seguinte figura mostra todas as partes dunha instancia e unha base de datos oracle.

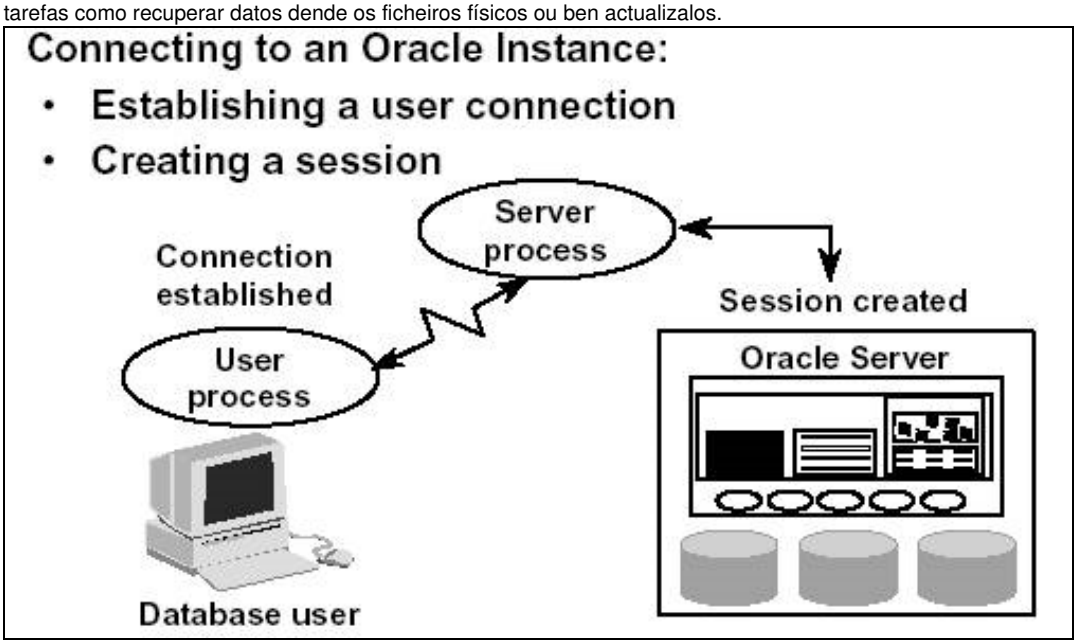

Instance
System Global Area (SGA)
Shared Pool
Library Cache
Shared SQL Area
SELECT * FROM employees
Private SQL Area (Shared Server Only)
Data Dictionary Cache
Server Result Cache
Other
Reserved Pool
Large Pool
Response Queue
Request Queue
Free Memory
I/O Buffer Area
UGA
PMON
SMON
RECO
MMON
MMNL
Others
Background Processes
Database Buffer Cache
Redo Log Buffer
Fixed SGA
Java Pool
Streams Pool
PGA
SQL Work Areas
Session Memory
Private SQL Area
Server Process
DBWn
CKPT
LGWR
ARCn
RVWR
Database
Data Files
Control Files
Client Process
Online Redo Log
Archived Redo Log
Flashback Log

A continuación describiremos polo miúdo cada un dos compoñentes que aparecen no gráfico

Procesos de usuario

Cada vez que un usuario executa unha aplicación Oracle levanta un proceso de usuario que dará soporte á conexión do usuario coa instancia. Dependendo da arquitectura da aplicación, estes procesos existirán ou ben no PC do propio usuario ou ben nunha capa intermedia (tipicamente un servidor de aplicacións). O proceso de usuario será o encargado de iniciar a conexión coa instancia. A iniciación e o mantemento da comunicación entre o proceso de usuario e a instancia chámase conexión. Unha vez que a conexión se establece crearase unha sesión na instancia para darlle soporte.

Unha vez creada a sesión levantarase un proceso de servidor no servidor de BD. Este proceso será o responsable de realizar en nome do usuario tarefas como recuperar datos dende os ficheiros físicos ou ben actualizalos.



Además dos procesos de usuario e de servidor asociados cunha conexión, tamén se crea unha estrutura de memoria adicional por cada usuario chamada PGA(Program Global Area) onde se almacenará información específica da sesión como variables bind ou variables de sesión. Cada proceso de servidor ten asociada unha PGA.

Estruturas lóxicas

Oracle utiliza unha serie de estruturas lóxicas para manexar os ficheiros físicos. Éstas inclúen espacios de táboas, segmentos, extensións e bloques que permitirán a Oracle controlar o uso de espacio físico que ten asignado.

Un esquema é unha estrutura lóxica que agrupa unha serie de obxectos (táboas, índices, paquetes, etc) relacionados de algunha maneira. O uso de esquemas facilita a organización dos datos.

Bloques de datos

Un bloque de datos é un conxunto de bytes de espacio en disco. Todo o espacio que Oracle asigna ou desasigna é sempre en termos de bloques de datos.

É o compoñente lóxico máis pequeno ó que normalmente chamamos bloque Oracle. Cada un deles estará formado por 1 ou máis bloques de disco do SO (que normalmente oscilan entre 512b e 2Kb).

O tamaño do bloque por defecto establecese no momento da creación da BD ainda que poidan convivir tamaños diferentes dentro da mesma base de datos. Tamaños normais de bloque son 2Kb, 4Kb, 8Kb, 16Kb e 32Kb.

O parámetro que establece o tamaño de bloque Oracle é DB_BLOCK_SIZE (dentro do ficheiro *init.ora*).

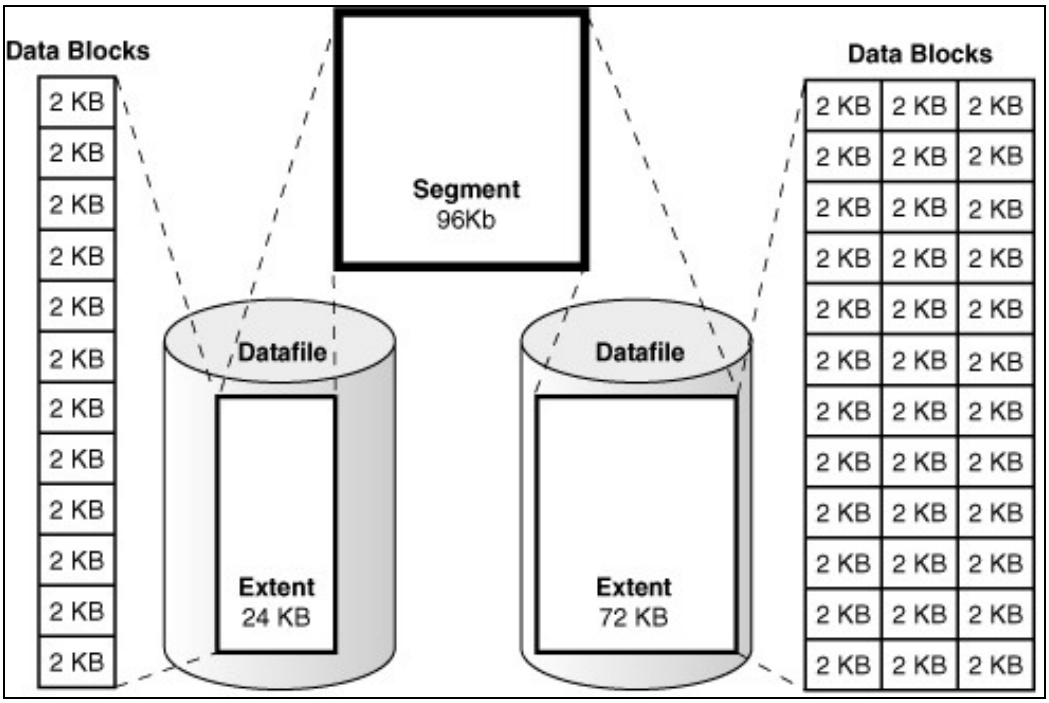
Extensións

Unha extensión é unha sucesión de bloques Oracle contiguos. Cada vez que creamos un obxecto dentro da BD (táboa, índice, etc) asignase unha cantidade de espacio inicial chamada extensión inicial; tamén debemos especificar o tamaño da seguintes extensión a asignar.

Podemos consultar a información sobre as extensións na táboa DBA_EXTENTS

Segmentos

Un segmento é un conxunto de extensións asignados a un obxecto da BD, como unha táboa, un índice, etc. Cada obxecto que consume espacio físico estará almacenado nun único segmento que a súa vez pertencerá unicamente a un espacio de táboas, dentro do cal pode haber varias extensións distribuidas ao longo de varios ficheiros de datos, tal e como mostra a seguinte figura:



A seguinte táboa lista os tipos de segmentos que podemos atopar dentro dunha BD.

Exemplos de segmentos

Tipo segmento	Descrición
Táboa	Almacena os datos nunha estrutura de filas e columnas
Índice	Mellora o rendemento de acceso ás taboas
Rollback	Mantén a consistencia de lectura durante as transacciones e realiza a recuperación das mesmas.
Partición	Divide unha táboa en partes máis pequenas e manexables de forma que se poida mellorar o rendemento.

Podemos consultar a información sobre os segmentos da BD na táboa DBA_SEGMENTS

Tablespaces

Unha BD Oracle estará lóxicamente dividida en *tablespaces*, sendo cada un deles unha entidade lóxica formada por 1 ou varios ficheiros de datos. Normalmente deberíanse manter as táboas relacionadas dentro do mesmo espacio de táboas, en tanto que éstes actúan como contedores para segmentos lóxicos tales como táboas.

Cando a BD se queda sen espacio é preciso crear máis *tablespaces* con novos ficheiros de datos asignados a eles, ou ben asignar máis ficheiros de datos aos *tablespaces* xa existentes.

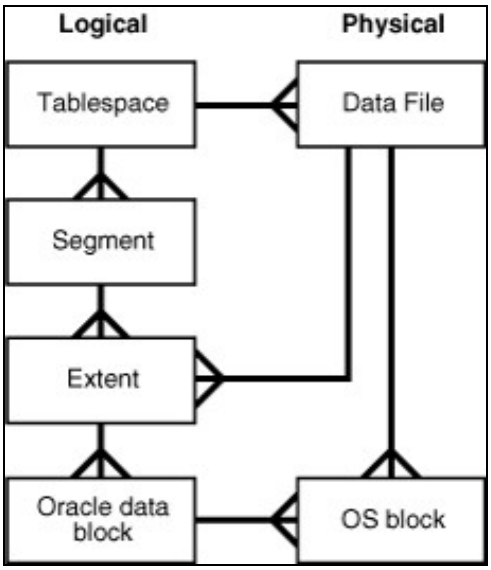
Non existen regras acerca do número de tablespaces que debería haber nunha BD, aínda que os seguintes 5 son os que por defecto aparecen

normalmente nunha BD, pero sería posible unha BD con só os 2 primeiros:

- SYSTEM
- SYSAUX
- UNDO
- TEMP
- USERS

Podemos consultar os tablespaces da BD na táboa DBA_TABLESPACES e os ficheiros físicos na táboa DBA_DATA_FILES

O siguientes diagramas mostran a relación entre as distintas estruturas lóxicas:



Estruturas físicas

Ao conxunto de ficheiros físicos utilizados por Oracle normalmente chamarémoslle Base de Datos (en oposición a instancia Oracle). A táboa seguinte mostra unha lista de ficheiros Oracle:

Ficheiros Oracle

Tipo de ficheiro	Información que contén
Control	Localizacións dos outros ficheiros físicos, o nome da BD, o tamaño de bloque, o xogo de carácteres, e a información de recuperación. Son necesarios para abrir a BD.
Datos	Datos almacenados na BD. Tamén contén metadatos.
Redo Log	Grava todos os cambios que se fan na BD. Utilizase nas recuperacións/restauracións.
Ficheiros de parámetros (PFILE ou SPFILE)	Configura os parámetros da SGA, características instaladas e procesos background.
Archived Log	Contén unha copia dos redo log, utilizados na recuperación.
Contrasinais	Ficheiro opcional utilizado para almacenar os nomes dos usuarios con privilexios SYSDBA e SYSOPER.
Oracle Net	Configuran o listener da BD e a conectividade cliente-base de datos.

Ficheiros de control

Son compoñentes criticos dentro da BD xa que almacenan información que non está dispoñible en ningún outro sitio. A información que conteñen inclúe:

- O nome da BD
- Nome, ubicacións, e tamaños dos ficheiros de datos e dos ficheiros redo log.

- Información utilizada para recuperar a base de datos en caso de fallo de disco ou erro de usuario.

Os ficheiros de control créanse no momento de creación da BD, especificándose a súa ubucación no parámetro *control_files* no PFILE. Debido a súa importancia serán multiplexados a a varias ubicacións. Utilizase o proceso CKPT para actualizar cada copia dos ficheiros de control, mantendoas así todas sincronizadas. Podemos consultar a vista V$CONTROLFILE para comprobar os seus nomes e ubicacións.

Vistas do dicionario de datos asociadas cos ficheiros de control

Ficheiros de datos

Son ficheiros físicos que almacenan os datos contidos nas táboas. Son estruturas físicas que dan soporte aos *tablespaces* (permiten agrupar segmentos loxicamente). Por cada *tablespace* debe existir polo menos 1 ficheiro de datos, inda que pode haber máis.

Cando un usuario executa unha consulta sobre unha táboa, o proceso servidor de usuario copia os datos necesarios dende os ficheiros de datos oportunos ata o *Buffer Cache* na SGA. Se o usuario realizou algunha transacción sobre eses datos, o proceso escritor (DBWn) escribe os bloques de datos modificados de volta aos ficheiros de datos.

Vistas do dicionario de datos asociadas con ficheiros de datos

Ficheiros de redo log

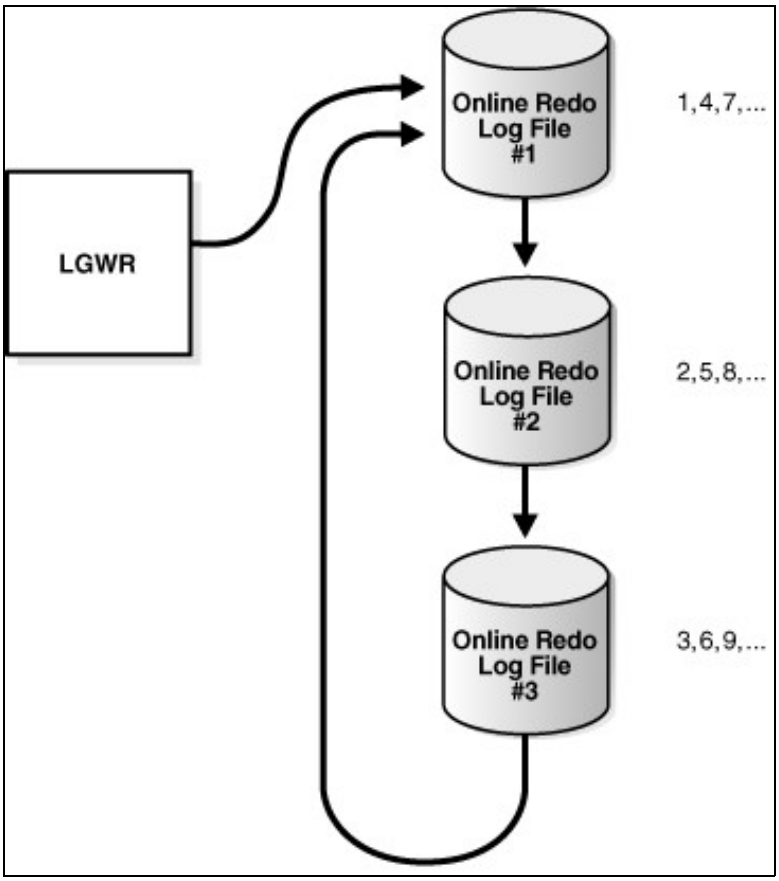
Cada vez que un usuario realiza unha transacción, a información necesaria para repetila en caso de fallo grávase automáticamente no Buffer Redo Log. Os contidos de este Buffer son escritos cada certo tempo aos ficheiros Redo Log polo proceso LGWR.

Debido a súa importancia nunha base de datos, estes ficheiros de redo, normalmente están replicados a diferentes ubicacións. Cada conxunto de copias de cada un dos ficheiros chámase grupo de redo log, e cada un dos ficheiros dentro do grupo chámase membro. Oracle escribirá automáticamente a todos os membros do grupo para mantelos sincronizados. Cada grupo debe estar composto de 1 ou máis ficheiros e unha BD debe ter como mínimo 2 grupos con 1 ficheiro cada un deles que serán utilizados dun xeito circular.

O proceso LGWR escribirá no grupo de redo log activo cada vez que:

- Cada 3 segundos
- Un usuario fai un commit
- O buffer redo log encheuse a 1/3 da súa capacidade
- O buffer redo log contén 1MB de información para escribir
- Cada vez que ocurre un checkpoint

O proceso LGWR escribirá no ficheiro redo log activo ata que este se encha, momento no cal se cambiará a o seguinte ficheiro redo log ata que se encha tamén para pasar ao seguinte, e así sucesivamente ata chegar a primeiro de novo. Os ficheiros redo log funcionan por tanto dun xeito circular. Podemos consultar a vista dinámica V$LOG para ver que grupo está activo en cada momento.



Cando a BD funciona en modo *ARCHIVELOG*, os ficheiros redo log, unha vez que se enchen, serán copiados a un ubicación secundaria antes de que poidan ser reutilizados polo LGWR. Será o proceso ARCn o encargado de estas copias.

Vistas do dicionario de datos asociados cos ficheiros redo log

Parámetros de inicialización

Tanto se a instancia opera con xestión da SGA manual ou automática determínase nos parámetros de inicialización. Hai dous tipos de parámetros: ficheiros de parámetros (PFILES) e ficheiros de parámetros de servidor (SPFILES). Podemos utilizar calquera deles para configurar as opcions da instancia, incluindo o tamaño da SGA, o tipo de xestión. Sen embargo existen certas diferencias entre os dous tipos que mostramos na seguinte táboa:

Compoñentes SGA

PFILE	SPFILE
Ficheiro de texto editable	Ficheiro binario que non pode ser editado directamente
Cando se fan cambios a instancia debe reiniciarse para que surtan efecto	A maior parte dos cambios pódense facer dinámicamente mentres a instancia se está executando
O seu nome será *initORACLE_SID.ora*	O seu nome será *spfileORACLE_SID.ora*
Pódese crear a partires do *SPFILE* usando o comando *create pfile from spfile*	Pódese crear a partires do ficheiro *PFILE* usando o comando *create spfile from pfile*

Existen máis de 250 parámetros documentados nun ficheiro PFILE ou SPFILE, que se dividirán en 2 categorias: básicos e avanzados. Oracle recomenda modificar manualmente só os parámetros básicos, uns 30. Estos son algúns dos parámetros de inicialización de Oracle 11g R2 máis comúns.

Instancia oracle

Unha instancia está formada pola estrutura de memoria principal de Oracle, chamada Área Global de Sistema (SGA) e varios procesos de Oracle en segundo plano. É coa SGA con que comunican os procesos de servidor cando un usuario accede aos datos. Tamén serve para a transferencia de

información entre usuarios así como a información estrutural máis frecuentemente utilizada.

Área Global de Sistema (SGA)

Está formada por 3 compoñentes obrigatorios e 3 opcionais

Compoñentes SGA

Compoñente	Descrición	Tipo de compoñente
Shared Pool	Cachea as sentencias SQL máis recentemente utilizadas na BD	Obrigatorio
Buffer Cache	Cachea os datos máis recentemente accedidos pola BD	Obrigatorio
Redo Log	Almacena información sobre transaccións en caso de ter que recuperar	Obrigatorio
Java Pool	Cachea os obxectos Java máis recentemente utilizados cando se usa a opción Oracle JVM	Opcional
Large Pool	Cachea datos para operacións masivas tales como as de RMAN	Opcional
Streams Log	Cache os datos da cola de mesaxes cando se utilizan a opción de colas avanzadas	Opcional

Oracle utilizará o algoritmo LRU (least recently used) para xestionar os contidos do Shared Pool e do Buffer Cache. Os compoñentes da SGA poden ser xestionados manual (debemos especificar o tamaño de cada compoñente) ou automáticamente (especificamos o tamaño da SGA e Oracle axusta automáticamente o tamaño de cada compoñente). Tanto se utilizamos unha ou outra forma de xestión a SGA dividirase en trozos máis pequenos chamados granulos. A asignación dinámica de memoria farase en termos de granulos. Estes gránulos serán asignados ou desasignados do *Buffer Cache*, *Shared Pool*, *Large Pool* ou *Java Pool* según as necesidades de cada unha destas áreas.

Procesos oracle en segundo plano

Existen varios tipos de procesos *background* e cada un realizará un traballo diferente na xestión da instancia. 5 deles son obrigatorios, o resto son opcionais e dependen da configuración.

Procesos oracle obrigatorios

Nome do proceso	Proceso do SO	Descrición
Monitor do sistema	SMON	Realiza a recuperación da instancia despois dun fallo, agrupa espacio libre na BD e xestiona o espacio usado nas ordenacións.
Monitor de procesos	PMON	Encargarse de limpar conexións fallidas coa BD.
Escritor da BD	DBWn	Escribe os bloques modificados da BD dende o buffer cache na SGA ata os ficheiros de datos en disco.
Escritor de Log	LOGWR	Escribe a información de recuperación de transaccións dende o Buffer de Redo Log na SGA ata os ficheiros Redo Log en disco.
Checkpoint	CKPT	Actualiza os ficheiros da BD despois dun evento Checkpoint.

Procesos oracle opcionais

Nome do proceso	Proceso do SO	Descrición
Archiver	ARCn	Copia a información de recuperación de transaccións escrita en disco polo LGWR ata os Redo Logs online e ata unha ubicación secundaria en disco, que se usarán en caso ter que facer unha recuperación.
Recuperador	RECO	Recupera transacción fallidas distribuidas a través de multiples BD cando usamos Oracle Distribuido.
Monitor da cola de traballos	CJQn	Aignar traballos ao procesos na cola de taballos cando usamos a Planificación de Traballos Oracle.

Cola de traballos	Jnnn	Executa os traballos planificados.
Monitor da cola	QMNn	Monitoriza as mensaxes na cola de traballos cando utilizamos as Colas Avanzadas Oracle.
Esclavo de consultas paralelas	Qnnn	Utilizado para executar porcións de consultas moi grandes cando están activadas as consultas paralelas.
Dispatcher	Dnnn	Asigna as peticións dos usuarios a unha cola para ser asignados aos procesos de servidor compartidos cando esta característica esta activada.
Servidor compartido	Snnn	Proceso de servidor compartido por varios usuarios.
Xestor de memoria	MMAN	Xestiona o tamaño de cada un dos compoñentes da SGA cando se activa a xestión automática da memoria compartida.
Monitor de memoria	MMON	Recolecta e analiza as estatísticas usadas polo *Automatic Workload Repository*.
Monitor de memoria lixeiro	MMNL	Recolecta e analiza as estatísticas usadas polo *Automatic Workload Repository*.
Escritor de recuperación	RVWR	Escribe información de recuperación en disco cando esta activada a opción de recuperación Flashback.
Escritor de seguemento de cambios	CTWR	Mantén un seguemento dos bloques da BD que cambiaron cando esta activada a opción de Xestión de Recuperación Incremental.

Nun sistema Unix podemos ver os procesos anteriores mediante un `ps -ef | grep PROD`

Algúns obxectos oracle

Aparte dos obxectos de BD habituais (táboas, índices, funcións, disparadores, etc) a continuación relacionamos unha serie de obxectos especiais dispoñibles en oracle.

Particións

O particionamento permite descompoñer táboas e índices moi grandes en partes máis pequenas e manexables chamadas particións. Cada partición será un obxecto independente co seu propio nome e opcionalmente unhas características de almacenamento propias.

As particións son útiles nos seguintes casos:

- Incrementan a dispoñibilidade: algunha partición pode estar non dispoñible e aínda podemos acceder as outras
- Maior facilidade para administrar os obxectos do esquema: Un obxecto particionado pódese manexar como un todo ou como as súas partes. As sentencias DDL poden manipular as particións ou o obxecto completo.
- Reduce os conflitos a recursos compartidos en sistemas OLTP: Por exemplo un DML pódese distribuir en varios segmentos en lugar de nun so.
- Mellora o rendemento das consultas en datos *warehouse*: As BD *warehouse* normalmente manexan obxectos moi grandes, de xeito que o seu particionamento mellorará o rendemento das consultas.

Unha partición dunha táboa ou dun índice debe conter o mesmo conxunto de atributos lóxicos, tales como os nomes das columnas, tipos de datos e restricións. Sen embargo particións diferentes poden ter atributos físicos diferentes, tales como os tablespaces nos que están almacenadas.

A clave da partición estará formada por 1 ou máis columnas que determinarán que filas van en cada partición. Cada fila debe ser asignada a unha soa partición dun xeito únivoco.

Estratexias de partición

- Partición de rango: asignanse filas a particións baseándose nun rango de valores da clave de partición. É o método máis común.
- Partición de lista: úsase unha lista de valores como a clave de partición. Controlamos o xeito en que filas individuais se asignan a particións.
- Partición hash: asignanse as filas as distintas particións baseándose nun algoritmo de hash que a BD aplica á clave de partición indicada polo usuario. O destino da fila ven indicado pola función interna de hash.

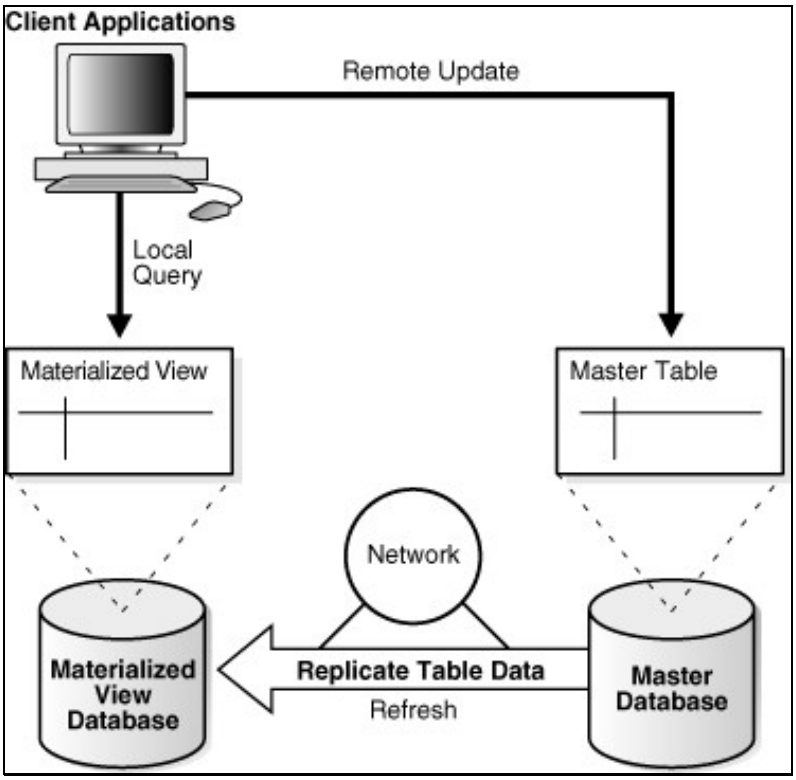
Ver documentación sobre particións

Vista materializadas

Un vista materializada é unha vista almacenada ou materializada previamente como un obxecto do esquema. Estas vistas serán útiles nos seguintes casos:

- En datawarehouse para almacenar datos procedentes de funcións agregadas tales como sumas ou medias.
- Para replicar, a vista conterá unha copia completa ou parcial da táboa nun momento determinado.
- En entornos móbiles, podemos usar vistas materializadas para descargar subconxuntos de datos dende un servidor central a clientes móbiles, con refrescos periódicos dende o servidor e propagación de cambios dende os clientes ao servidor.

O seguinte diagrama ilustra unha vista materializada nun BD baseada nunha táboa master noutra BD. Os cambios na táboa master replicanse a BD coa vista materializada.



Ver documentación sobre vistas materializadas

Columnas virtuais

En oracle 11g podemos utilizar este tipo de columnas derivadas de outra(s) mediante a utilización dunha expresión SQL ou PL/SQL. A diferencia das columnas normais, non serán almacenadas no disco. A BD calculará a columna virtual dinámicamente cando se fai a consulta. Podemos utilizalas tanto en DML como en DDL. Tamén se poden definir índices e recopilar estatísticas sobre elas.

Clusters

Os *clusters* son 2 ou máis táboas que teñen columnas comúns e almacenan datos relacionados no mesmo bloque. Cando varias táboas están nun cluster, un único bloque poderá conter filas de múltiples táboas. Por exemplo, un bloque pode conter filas das táboas *empregados* e *departamentos* (normalmente accedidas xuntas) en lugar de datos dunha soa táboa.

A clave do cluster será a columna ou columnas que as táboas teñen en común. No caso das táboas *empregados* e *departamentos* podería ser a columna *id_departamento*. Debemos especificar a clave do cluster no momento da súa creación e cando creamos cada unha das táboas que formarán parte del.

O valor da clave do cluster é o valor da columna para un conxunto particular de filas. Todos os datos que conteñen o mesmo valor da clave de cluster, por exemplo id_departamento=20, serán almacenados xuntos.

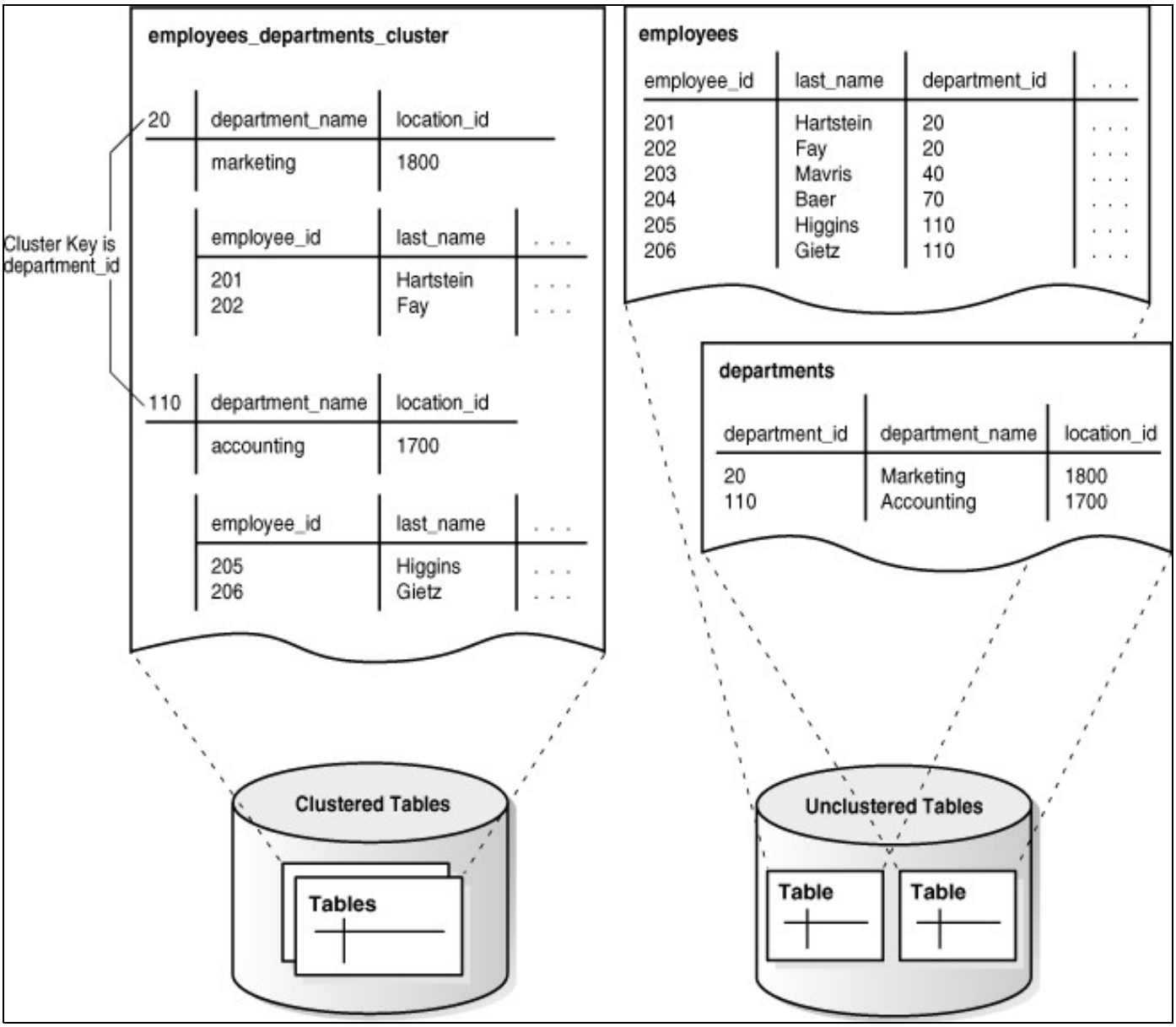
Os clusters ofrecen as seguintes vantaxes:

- O E/S a disco redúcese para os *joins*
- O tempo de acceso mellora para os *joins*
- Require menos espacio de almacenamento posto que a columna común almacénase só unha vez.

As táboas en cluster non son axeitadas para:

- Táboas que se modifican con frecuencia
- Táboas que sobre as que se fan *full scans* con frecuencia

A seguinte figura mostra o cluster das táboas departamentos e empregados. A BD almacenará as filas do departamento 20 xuntas, as do departamento 110 xuntas, etc. Se as táboas non estivesen en cluster non poderíamos asegurar que as filas relacionadas estivesen xuntas.



Documentación sobre clusters

Tipos de datos

A continuación relacionamos os tipos de datos dispoñibles en Oracle:

Tipos caracter

- **CHAR(tamaño [byte|char]), NCHAR(tamaño):** Tipo fixo que sempre almacena o numero de bytes especificado, engadindo espacios a dereita se fose preciso. O tamaño é en bytes se non incluimos a palabra reservada char. A variante NCHAR utiliza o xogo de carácteres Unicode e o tamaño é sempre en caracteres.
- **VARCHAR(tamaño [byte|char], VARCHAR2(tamaño [byte|char]), NVARCHAR2(tamaño):** Tipo de tamaño variable. A diferencia do CHAR so se almacenan a cantidade de datos utilizada. NVACHAR2 é o mesmo para Unicode. O tipo preferido e VARCHAR2.
- **LONG:** Mantido por razón de compatibilidade. Está desaconsellado o seu uso. Almacena datos alfanuméricos de lónxitude variable ata 2Gb. Existen moitas limitación cando usamos este tipo de datos: as táboas non se poden particionar, os tipos LONG non se poden usar en subconsultas, só unhas poucas funcións se poden usar con este tipo. O tipo preferido é CLOB se os datos non caben nun VARCHAR2.

Tipos numéricos

- **NUMBER[(precision[, escala])]:** Almacena cero, numeros positivos e negativos. A precisión é o número de díxitos (por defecto 38).
- **BINARY_FLOAT e BINARY_DOUBLE:** Almacenan números de precisión simple e doble.

Tipos de data

- **DATE:** Almacena datas cunha granularidade de 1 seg.
- **TIMESTAMP[(precision)]:** Almacena datas e horas cunha granularidade de subsegundos. A precisión por defecto e 6, e pode variar de 0 a 9
- **TIMESTAMP[(precision)] WITH TIMEZONE:** Extentendo o tipo TIMESTAMP engadindolle a zona horaria.
- **INTERVAR YEAR[(precision)] TO MONTH:** Almacena un periodo de tempo en anos e meses.
- **INTERVAL DAY[(precision)] TO SECOND[(s_precision)]:** Alamacena un período de tempo en días, horas, miutos e segundos.

Tipos LOB

- **CLOB:** Almacena datos alfanuméricos de lonxitude variable.
- **NCLOB:** Almacena datos alfanuméricos de lonxitude variable usando Unicode.
- **BLOB:** Datos binarios de lonxitude variable.
- **BFILE:** Almacena datos binarios de lonxitude variable fora da BD. Os bfiles están limitados a 4Gb de datos, incluso menos en algún SO.

Tipos ROWID

- **ROWID:** Almacena enderezos físicos de 64 bytes para as filas dunha táboa. O ROWID incorpora o ID do obxecto, un número relativo do ficheiro de físico de datos, o número de bloque e a posición da fila dentro do bloque.
- **UROWID:** O mesmo que ROWID para as táboas organizadas como índices.

Tipos binarios

- **RAW(tamaño):** Almacena datos sen estrutura de ata 2000 bytes de tamaño.
- **LONG RAW:** Almacena datos sen estrutura de ata 2Gb bytes de tamaño. No é aconsellable, debería usarse BLOB no seu lugar.

Xestión de usuarios

Cada usuario debería ter unha conta, xa que contas compartidas son difíciles de auditar, aparte de ofrecer un nivel de seguridade escaso. Oracle ofrece 3 métodos de autenticación

Usuarios autenticados por password

Cando un usuario tenta conectar a BD verifica que o usuario é válido e o contrasinal é o mesmo que o almacenado na BD para es usuario.

```
CREATE USER ramiro IDENTIFIED BY contrasinal;
```

Usuarios autenticados externamente

A BD verifica que o usuario é correcto, pero confia a súa autenticación ao SO. Por tanto este tipo de usuarios non validarán o contrasinal contra a BD.

```
CREATE USER ops$ramiro IDENTIFIED EXTERNALLY;
```

Usuario autenticados globalmente

A BD verifica que o usuario é válido e pasa a información da conexión a opción de seguridade avanzada correspondente para realizar a auténticación. Entre as opción dispoñibles temos certificados X.509, Kerberos ou RADIUS. A sintaxe para crear un usuario autenticado globalmente depende da opción de seguridade escollida, pero sempre se utilizará IDENTIFIED GLOBALLY. A continuación mostramos un exemplo:

```
CREATE USER ramiro IDENTIFIED GLOBALLY AS 'cn=ramiro, OU=tier2, O=security, C=US';
```

A cada usuario asignaselle un espacio de táboas por defecto. Se non o asignamos explicitamente no momento de creación de usuario, Oracle farao por nos. Usaremos o modificador DEFAULT TABLESPACE.

```
CREATE USER ramiro IDENTIFIED BY ramiro DEFAULT TABLESPACE users;
```

Tamén podemos asignar espacios de táboas temporais, onde se almacenarán os segmentos temporais. Son aqueles creados durante operacións masivas de ordeamento tales como ORDER BY, GROUP BY, SELECT DISTICNT, MERGE JOIN ou CREATE INDEX. A continuación vemos un exemplo:

```
CREATE USER ramiro IDENTIFIED BY ramiro DEFAULT TABLESPACE users TEMPORARY TABLESPACE temp;
```

Asignar un perfil a un usuario

Todos os usuarios teñen asignado un perfil, que non é máis que un nome para un conxunto de limites no uso dos recursos da BD asociados a ese usuario. Cada usuario poderá ter asignado únicamente un perfil. O perfil por defecto será default. Para explícitamente asignar un perfil a un usuario incluiremos a palabra reservada PROFILE.

```
CREATE USER ramiro IDENTIFIED BY ramiro DEFAULT TABLESPACE users TEMPORARY TABLESPACE temp PROFILE resource_profile;
```

Conceder e revocar privilexios

Os privilexios permitiran a un usuario acceder aos obxectos da BD ou executar procedementos almacenados propiedade de outros usuarios. Tamén permitirán a un usuario realizar operacións a nivel de sistema tales como conectar coa BD, crear unha táboa, ou alterar a BD. Os privilexios asignaranse co comando GRANT a un usuario, ao usuario especial PUBLIC (asigna a todos os usuarios os permisos asignados a public) ou a un rol e revocaranse co comando REVOKE. Oracle xestiona tres tipos de privilexios:

Privilexios de obxectos: Permisos a obxectos dentro dun esquema tales como táboas, vistas, secuencias, procedementos e paquetes.

- ***Privilexios sobre táboas***: SELECT, INSERT, UPDATE, DELETE, ALTER, DEBUG, INDEX, REFERENCES
- ***Privilexios sobre vistas***: SELECT, INSERT, UPDATE, DELETE, DEBUG, REFERENCES
- ***Privilexios sobre secuencias***: SELECT, ALTER
- ***Privilexios sobre funcións, procedementos, paquetes a obxectos Java***: DEBUG, EXECUTE

Privilexios de sistema: Permisos sobre operacións a nivel de base de datos, tales como conectarse, crear outros usuarios, alterar a base de datos ou consumir unha cantidade ilimitada do espacio da táboas

- ***Privilexios sobre a base de datos***: ALTER DATABASE, ALTER SYSTEM, AUDIT SYSTEM, AUDIT ANY
- ***Privilexios sobre o depurador***: DEBUG CONNECT SESSION, DEBUG ANY PROCEDURE
- ***Privilexios sobre índices***: CREATE ANY INDEX, ALTER ANY INDEX, DROP ANY INDEX
- ***Privilexios sobre o planificador de traballos***: CREATE JOB, CREATE ANY JOB, EXECUTE ANY PROGRAM, EXECUTE ANY CLASS, MANAGE SCHEDULER

- ***Privilexios sobre procedementos***: CREATE PROCEDURE, CREATE ANY PROCEDURE, ALTER ANY PROCEDURE, DROP ANY PROCEDURE, EXECUTE ANY PROCEDURE
- ***Privilexios sobre perfiles***: CREATE PROFILE, ALTER PROFILE, DROP PROFILE
- ***Privilexios sobre roles***: CREATE ROLE, ALTER ANY ROLE, DROP ANY ROLE, GRANT ANY ROLE
- ***Privilexios sobre secuencias***: CREATE SEQUENCE, CREATE ANY SEQUENCE, ALTER ANY SEQUENCE, DROP ANY SEQUENCE, SELECT ANY SEQUENCE
- ***Privilexios sobre sesións***: CREATE SESSION, ALTER SESSION, ALTER RESOURCE COST, RESTRICTED SESSION
- ***Privilexios sobre sinónimos***: CREATE SYNONYM, CREATE ANY SYNONYM, CREATE PUBLIC SYNONYM, DROP ANY SYNONYM, DROP PUBLIC SYNONYM
- ***Privilexios sobre táboas***: CREATE TABLE, CREATE ANY TABLE, ALTER ANY TABLE, DROP ANY TABLE, COMMENT ANY TABLE, SELECT ANY TABLE, INSERT ANY TABLE, UPDATE ANY TABLE, DELETE ANY TABLE, LOCK ANY TABLE, FLASHBACK ANY TABLE
- ***Privilexios sobre espacios de táboas***: CREATE TABLESPACE, ALTER TABLESPACE, DROP TABLESPACE, MANAGE TABLESPACE, UNLIMITED TABLESPACE
- ***Privilexios sobre disparadores***: CREATE TRIGGER, CREATE ANY TRIGGER, ALTER ANY TRIGGER, DROP ANY TRIGGER, ADMINISTER DATABASE TRIGGER
- ***Privilexios sobre usuarios***: CREATE USER, ALTER USER, DROP USER
- ***Privilexios sobre vistas***: CREATE VIEW, CREATE ANY VIEW, DROP ANY VIEW, COMMENT ANY TABLE, FLASHBACK ANY TABLE
- ***Outros privilexios***: ANALYZE ANY, GRANT ANY OBJECT PRIVILEGE, GRANT ANY PRIVILEGE, GRANT ANY ROLE, SELECT ANY DICTIONARY, SYSDBA, SYSOPER

Privilexios de rol: Privilexios sobre obxectos ou sobre o sistema que o usuario obten mediante un rol. Os roles son ferramentas que simplifican a administración de grupos de privilexios. Os usuarios aos que se lle concede un rol herdan todos os seus privilexios. Concédense privilexios a un rol, que a súa vez pode ser otorgado a outros roles ou a un usuario.

Crear un rol

```
CREATE ROL rol_op;
```

Pódese asocial un contrasinal cun rol

```
SET ROLE rol_op IDENTIFIED BY pass;
```

Os roles pódense activar mediante o comando `SET ROLE nome_rol` para activar o rol nome_rol. Coa palabra reservada ALL activamos todos os roles asignados e ase usuario. Se un role ten un contrasinal asociado debemos proporcionarlo cando o activamos

```
SET ROLE rol_op IDENTIFIED BY "pass";
```

Os roles otorgados a túa sesión pódense consultar na táboa session_roles. Para identificar os roles otorgados o teu usuario ou a PUBLIC executa a seguinte sql

```
select granted_role from user_role_privs where username in (user, 'PUBLIC');
```

O seguinte comando activa todos os roles salvo a lista role_list

```
SET ROLE ALL EXCEPT role_list
```

Asignar cuotas de disco aos usuarios

Cando creamos un usuario podemos asignarlle unha cuota dentro dun espacio de táboas para limitar a cantidade de espacio que poden consumir. Cada usuario, por defecto non ten cuota, é por iso que precisamos asignarlle unha no momento da creación ou posteriormente.

```
CREATE USER chip IDENTIFIED BY "Seek!r3t" QUOTA 100M ON USERS;

ALTER USER bart QUOTA UNLIMTED ON USERS;
```

Limitar os recursos mediante o uso de perfís

Os perfís permitirán establecer os límites dos recursos, como por exemplo o tempo de CPU, a memoria, o número de lecturas lóxicas por sesión, etc. Activamos os limites a recursos con perfís mediante un parámetro da BD:

```
ALTER SYSTEM SET resource_limit = TRUE SCOPE = BOTH;
```

Para asignar limites sobre os recursos a perfís utilizaremos os comandos CREATE PROFILE ou ALTER PROFILE. Estas sentencias soportan as seguintes palabras reservadas para limitar recursos: CONNECT_TIME,CPU_PER_CALL,CPU_PER_SESSION,IDLE_TIME,LOGICAL_READS_PER_CALL,LOGICAL_READS_PER_SESSION,PRIVATE_SGA,SE

```
CREATE PROFILE agent LIMIT CONNECT_TIME 10;

ALTER PROFILE data_analyst LIMIT CONNECT_TIME UNLIMITED;
```

Xestión de consistencia e de concurrencia

Cada vez que un proceso ou un usuario modifica datos na BD, Oracle gravará os antigos valores como datos *undo* o que permitirá facer *rollback*, consultas de lectura consistente, ou consultas *flashback*, ou recuperacións en caso de fallo. Os datos *undo* almacenaránse nos espacios de táboas **UNDO**.

Aínda que pode haber varios espacios de táboas **UNDO**, somentes un pode estar activo ao mesmo tempo. Os segmentos dentro de este espacio de taboas conterán os datos para desfacer transaccións, e medrarán ou reduciranse según a demanda, actuando todos eles como un buffer circular. Unha transacción pode residir unicamente nun segmento de *rollback*.

Os segmentos **undo** tamén coñecidos como segmentos **rollback**, son similares a outros tipos de segmentos (de táboa, de índices, etc); están formados por extensións que a súa vez están formados por bloques de datos contiguos. Sen embargo este tipo de segmentos só poderán almacenarse no espacio de táboas **undo** e serán sempre propiedade do SYS independentemente de quen sexa o propietario da transacción. Podemos ver o nomes dos segmentos **undo** activos na vista V$ROLLNAME. Tamén podemos consultar a vista V$TRANSACTION para ver a relación entre transaccións e segmentos **undo**.

Os segmentos de *rollback* son os encargados da consistencia de lectura. Se un usario está facendo cambios nunha táboa, e outro usuario, ao mesmo tempo fai unha consulta sobre esa táboa este verá os valores antigos ata que o primeiro usuario faga un **COMMIT**.

Monitorizar, configurar e administrar o espacio de táboas undo

A partires de Oracle9i, a xestión dos tablespaces **undo** son relativamente automáticos, sen embargo existirán 2 situación que requerirán a intervención do DBA:

- *ORA-01650: Unable to extend rollback segment*: Quedamos sen espacio nos segmentos de *rollback*
- *ORA-01555: Snapshot too old*: Consultas de longa duración son incapaces de manter a consistencia de lectura.

Os parámetros de configuración básicos que permitirán a configuración do espacio de táboas undo son os seguintes:

UNDO_MANAGEMENT indica a forma de xestión dos datos **undo**, ou ben manualmente mediante segmentos de **rollback** ou ben automáticamente mediante un único tablespace **undo**.

UNDO_TABLESPACE indica o nome do tablespace **undo** utilizado para garantir a consitencia de lectura e o rollback de transaccións.

UNDO_RETENTION especifia, en segundos, canto tempo se debe manter a información para desfacer os cambios despois de que se fixera COMMIT sobre os mesmos. A modificación deste parámetro terá un gran efecto sobre o tamaño do tablespace **undo**.

RETENTION GUARANTEE cando este parámetro está activado como mínimo garantese o tempo indicado polo parámetro UNDO_RETENTION. Exemplo de como modificar este parámetro. ALTER TABLESPACE UNDOTBS1 RETENTION GUARANTEE.

Monitorizar bloqueos e resolver os seus conflitos

Un conflito aparece cando 2 usuarios simultáneos tratan de escribir o mesmo(s) rexistro(s). Cando varios usuarios solititan un bloqueo, o primeiro en chegar é o que o obtén, e o resto pasan a unha cola FIFO esperando polo recurso solititado. As sesións en espera quedan como colgados a non ser que especifiquemos a opción NOWAIT no bloqueo. Ao final da transacción, cando emitimos un COMMIT ou un ROLLBACK, todos os bloqueos de esa sesión son liberados.

Podemos bloquear unha táboa mediante o comando LOCK TABLE utilizando algún dos modos que detalla a seguinte táboa:

Modos de bloqueo Oracle

Modo	Descrición
ROW SHARE	Permite o acceso concurrente a unha táboa bloqueada, pero prohibe a outros usuarios bloquear a táboa enteira para acceso exclusivo
ROW EXCLUSIVE	O mesmo que ROW SHARE, pero tamén prohibe bloquear en modo SHARE. Este é o tipo de bloqueo por defecto cos comandos SQL DML estándar (UPDATE, INSERT, DELETE)
SHARE	Permite consultas concurrentes pero prohibe DMLs na táboa; requírese este modo cando facemos un CREATE INDEX.
SHARE ROW EXCLUSIVE	Permite a consulta a outros usuarios pero prohibe o bloqueo en modo SHARE ou a modificación de filas.
EXCLUSIVE	O modo máis restritivo; permite consultas sobre a táboa pero prohibe calquera DML por parte de outros usuarios, así como calquera outro modo de bloqueo. Este modo é obrigatorio cando facemos un DROP TABLE.

Tamén podemos obter bloqueos sobre filas individuais usando a sentencia SELECT... FOR UPDATE.

Un abrazo mortal é un tipo de conflito de bloqueo no que 2 (ou máis) usuarios esperan por recursos bloqueados polo outro usuario. O usuario 1 espera por un recurso bloqueado polo usuario 2 que a súa vez espera por un recurso bloqueado polo usuario 1. Este tipo de bloqueos requiren intervención externa. O erro oracle é *ORA-00060: Deadlock detected while waiting for resource*

Niveis de aillamento para transaccións definidos por ANSI/ISO SQL

- SERIALIZABLE: Este nivel é o máis restritivo e di que as transacción deben ter unha completa independencia entre elas, como se fosen executadas dun xeito secuncial. Non se poden dar ningún tipo de anomalia.
- REPEATABLE READ: Os datos obtidos por unha sentencia SELECT non poden ser modificados; sen embargo se o SELECT conten unha clausula WHERE poden aparecer filas fantasmas. Neste nivel a transacción obten bloqueos de lectura, pero non obten bloqueos de rango.

Transacción 1	Transacción 2

`/* Consulta 1 */`
`SELECT * FROM usuarios`
`WHERE idade BETWEEN 10 AND 30;`	

	`/* Consulta 2 */`
	`INSERT INTO usuarios VALUES ( 3, 'Bob', 27 );`
	`COMMIT;`

`/* Consulta 1 */`
`SELECT * FROM usuarios`
`WHERE idade BETWEEN 10 AND 30;`	

- READ COMMITTED: Os datos recuperados por unha transacción puideron ser modificados por outra transacción. Poden ocurrir lecturas non repetibles; algúns datos recuperados por unha sentencia SELECT puideron ser modificados por outra transacción confirmada. Neste nivel de aillamento os bloqueos de lectura liberanse inmediatamente despois da consulta, mentras que os bloqueos de escritura manteñense ata o final da transacción.

Transacción 1	Transacción 2
`/* Consulta 1 */` `SELECT * FROM usuarios WHERE id = 1;`	

	`/* Consulta 2 */`
	`UPDATE usuarios SET idade= 21 WHERE id = 1;`
	`COMMIT;`

Transacción 1	Transacción 2

`/* Consulta 1 */`
`SELECT * FROM usuarios WHERE id = 1;`
`COMMIT;`	

- READ UNCOMMITTED: Permite as lecturas sucias. Unha transacción pode ver cambios de outra transacción antes de que esta última faga o commit.

Transacción 1	Transacción 2
`/* Consulta 1 */` `SELECT * FROM usuarios WHERE id = 1;`	

	`/* Consulta 2 */`
	`UPDATE usuarios SET idade = 21 WHERE id = 1;`
	`/* Aquí non se fai commit */`

`/* Consulta 1 */` `SELECT * FROM usuarios WHERE id = 1;`	
	`ROLLBACK;`

A seguintes táboas mostran a relación entre os niveis de aillamento e as distintas anomalias e os tipos de bloqueos

Aillamento Vs Anomalías

Nivel de aillamento	Lecturas sucias	Lecturas non repetibles	Fantasmas
Read uncommited	X	X	X
Read Committed	-	X	X
Repeatable Read	-	-	X
Serializable	-	-	-

Aillamento Vs Bloqueos

	Range Lock	Read Lock	Write Lock
Read Uncommited	X	X	X
Read Committed	X	X	V
Repeatable Read	X	V	V
Serializable	V	V	V

Niveis de aillamento en Oracle

- **Read committed:** cada consulta executada por unha transacción ve únicamente os cambios confirmados antes de que a consulta -non a transacción- comence. Nunha transacción read-committed, ocurre un conflito de escritura cando unha transacción intenta modificar unha fila modificada en non confirmada por outra transacción (chamada transacción bloqueante). Unha anomalía típica deste nivel de aillamento é o *update perdido*
- **Serializable:** unha transacción ve únicamente os cambios confirmados no momento de comenzar a transacción -non a consulta- e os cambios feitos pola mesma transacción. Deste xeito, calquera fila lida dentro da transacción devoverá sempre o mesmo resultado se se rele mentres dure a transacción. Por tanto, cambios feitos por outras transaccións non serán visibles. Oracle permitirá a unha transacción serializable facer cambios sobre unha fila modificada se ésta foi confirmada antes de que a transacción comence. Devolverase un erro (*ORA-08177: Cannot serialize access for this transaction*) cando unha transacción serializable intenta modificar ou borrar unha fila modificada por outra transacción e confirmada despois de que a primeira transacción comenzase.
- **So lectura:** É similar ao nivel serializable, pero as transacción de so lectura non permiten modificar os datos dentro de sí mesmas a non ser que o usuario sexa SYS. Deste xeito, nunha transacción de só lectura non se pode dar o ORA-08177. Son útiles cando é preciso xerar informes consistentes co momento de comenzo da transacción. Con este tipo de transacción podemos obter o erro *snapshot too old* cando o período de retención *undo* non é suficiente para garantir a consistencia de lectura durante toda a duración da transacción.

Por defecto o nivel de aillamento é *read committed*.

Acceso concurrente e consistencia de datos

Copias de respaldo e recuperacións da BD

Os *backups* son unhas das tarefas máis imporantes dun DBA. A dispoñibilidade definida no SLA (service-level agreements) e o máximo tempo que o sistema pode estar caído definirán a método de respaldo/restauración escollido. Atendendo a estratexia temos dispoñibles:

- Total: Inclúe todos os ficheiros da BD a polo menos un ficheiro de control. O ficheiros de redo log nunca se copiarán; restaurar ficheiros redo log reemplazando os actuais sempre resultará nunha perda de datos.
- Parcial: Inclúe cero ou máis tablespaces, que a súa vez incluirán cero ou máis ficheiros de datos; o ficheiro de control será opcional nunha recuperación parcial.

Atendendo ao tipo:

- Completo: Incluirá todos os bloques de todos os ficheiros de datos.
- Incremental: Só incluirá aqueles bloques de datos modificados dende o último backup completo

Dependendo do modo:

- Consistente: Tamén coñecido como *backup offline*. Realizase mentres a BD está pechada. Estes backups son consitentes porque o SCN nos ficheiros de control coincide co SCN na cabeceira de cada ficheiro de datos.
- Inconsistente: Tamén chamado *backup online*, realizase mentres a BD está aberta e dispoñible ao usuarios. Será inconsistente porque os SCN dos ficheiros de control no coincidirán coas cabeceiras dos ficheiros de datos.

Os elementos implicados nos procesos de respaldo e recuperación son:

- Os ficheiros de control que manterán un lista dos ficheiros da BD xunto cun rexistro de cal foi o backup máis recente (só se usamos a ferramnto RMAN).
- O proceso background *checkpoint* (CKPT)
- O proceso background *database writer* (DBWn)
- Os ficheiros redo log e/ou ficheiros archive redo log se o fallo foi máis serio
- Area flash recovery

A ferramenta principalmente utilizada para facer copias de respaldo é RMAN, e podemos utilizala para facer calquera tipo de backup dos anteriormente relacionados.

Os ficheiros dos que RMAN fará backup son:

- Datafiles, ficheiros de control e SPFILE
- Redo logs arquivados
- Outros backups creados con RMAN

Os backups de RMAN pódense almacenar como:

- Imaxes de copia(*image copies*): Duplicado bit por bit do datafile, será idéntica á copia feita mediante o SO. As imaxes de copia creadas con RMAN rexistraranse no seu repositorio, a diferencia das creadas co SO.
- Conxuntos de backup (*backup sets*): É un estrutura lóxica contendo datos dos datafiles, ficheiros de control, SPFILE, e redo logs arquivados. Tamén se pode incluir un conxunto de backup noutro conxunto. Será un ou máis ficheiros con formato específico de RMAN.

Un backup completo dun datafile é un backup que inclue cada bloque de datos usado dentro do datafile. Se se crea como unha imaxe de copia, reproducirase o contido do datafile enteiro. Cando o copiamos a un backupset os bloques non usados saltanse.

Un backup incremental dun datafile captura unicamente aqueles bloques modificados dende un momento determinado, normalmente cando se fixo o anterior backup. Os backups incrementais serán sempre backupsets. RMAN pode crear únicamente backups incrementais de datafiles, nunca de ficheiros redolog ou outros ficheiros.

Se a BD está en modo ARCHIVELOG, podemos facer backups (completos ou incrementais) coa BD aberta; se está en modo NOARCHIVELOG só poderemos facer backups incrementais despois dun peche ordeado.

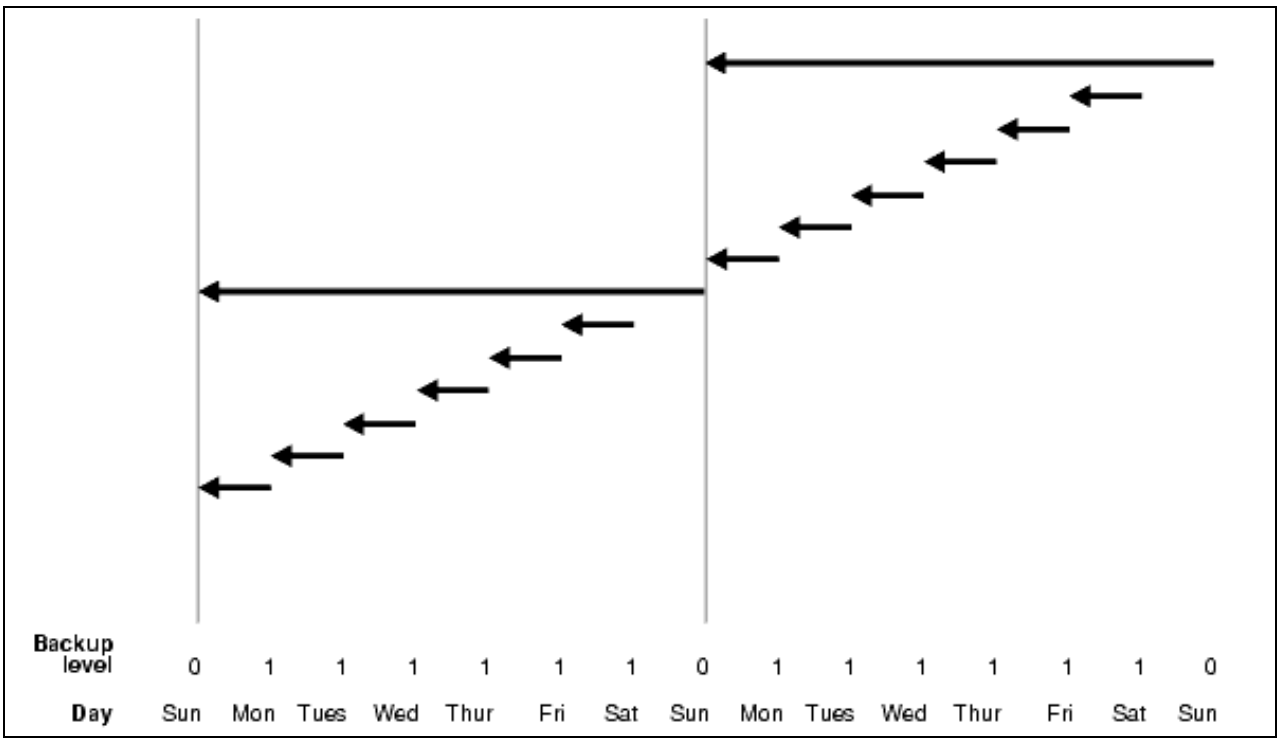
Nivel 0 e nivel 1 dos backups incrementais: Os backups incrementais poden ser de nivel 1 ou 0. Un nivel 0 fará unha copia completa da BD; a única diferencia entre un backup nivel 0 e un backup completo é que este último non pode estar incluido dentro dunha estratexía incremental.

Dentro dun backup incremental de nivel 1 distinguimos os seguintes tipos:

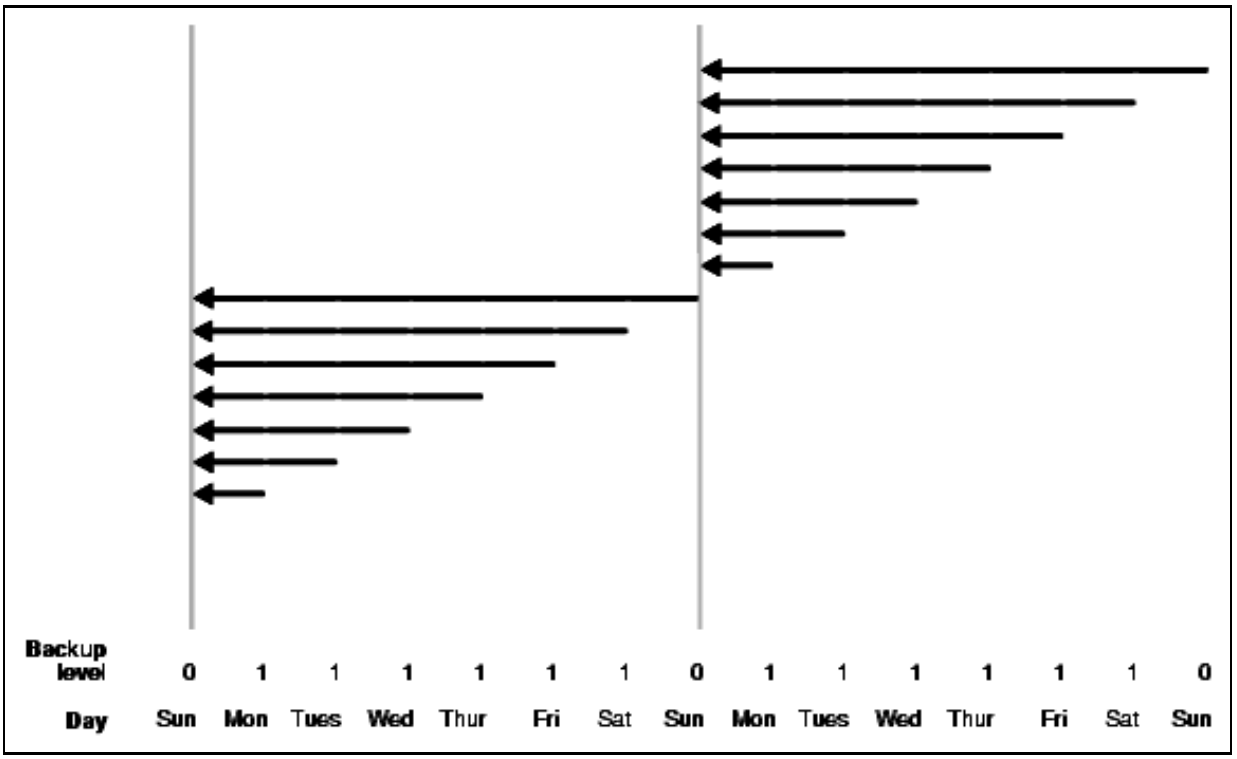
- Backup diferencial: copia todos os bloques modificados dende o último backup incremental de nivel 1 ou 0.
- Backup acumulativo: copia todos os bloques modificados dende o último backup de nivel 0.

Por defecto chamamos backup incremental a un backup diferencial.

RMAN> BACKUP INCREMENTAL LEVEL 1 DATABASE;



BACKUP INCREMENTAL LEVEL 1 CUMULATIVE DATABASE; # bloques cambiados dende o nivel 0



E partiríamos dun backup de nivel 0

BACKUP INCREMENTAL LEVEL 0 DATABASE;

Ver conceptos de Backup con RMAN

Índices

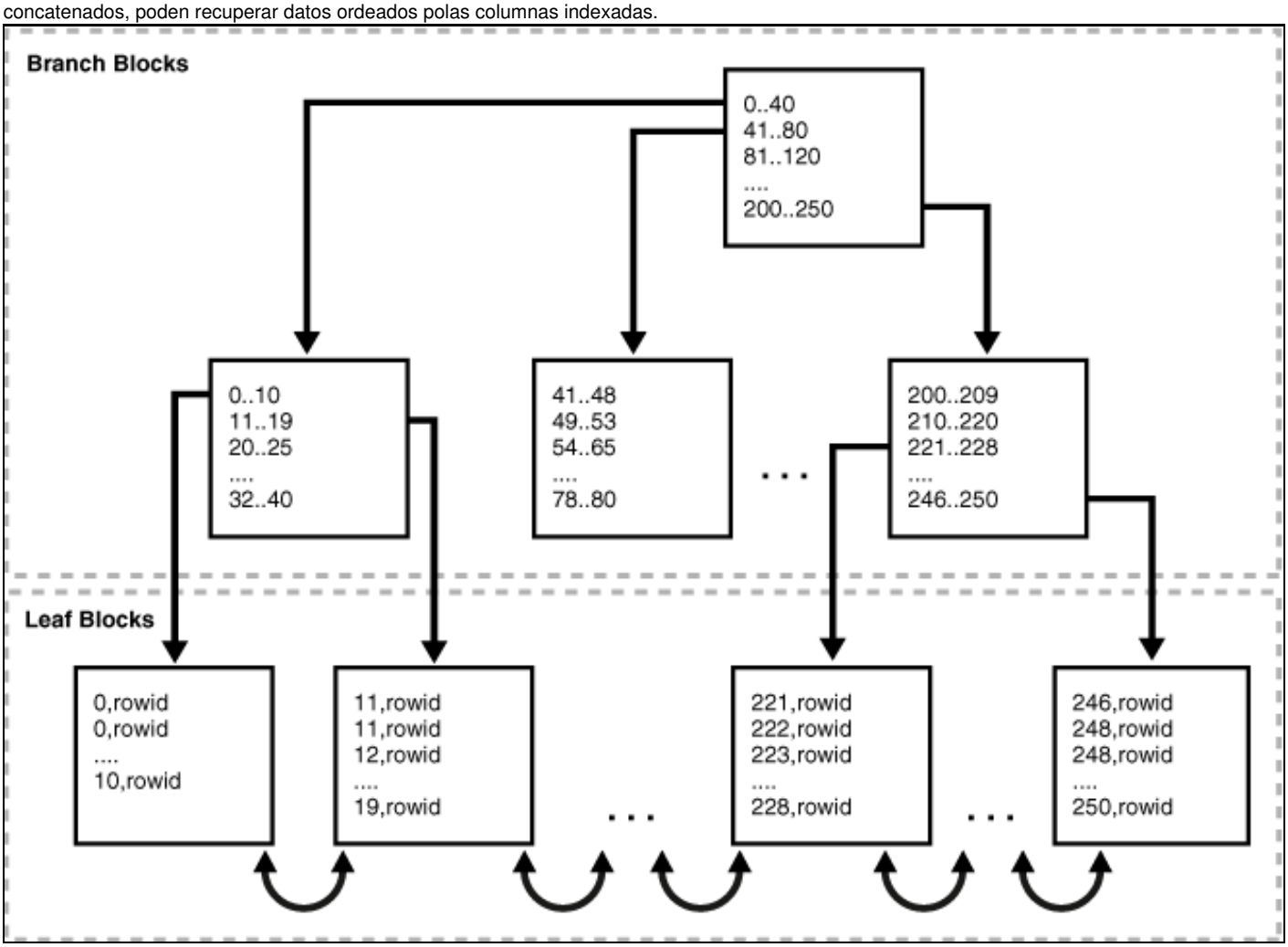
Os índices son estruturas de datos opcionais constuidas sobre táboas e permitirán mellorar a velocidade de acceso aos datos. Os índices son un dos métodos para reducir a tráfico I/O a disco. En xeral deberíamos crear un índice cando se da algunha das seguintes condicións:

- Consultase frecuentemente sobre as columnas indexadas devolvendo un pequeno porcentaxe do número total de filas da táboa.
- Existe unha restrición de integridade referencial sobre as columnas indexadas. O índice é un medio de evitar bloqueos sobre a táboa filla cando modificamos a clave primaria da nai ou borramos un dos seus rexistros.
- Temos unha restrición de clave única sobre a táboa e queremos especificar todas as opcións do índice.

Podemos dividir os índices Oracle según as seguintes categorías:

Índices B-Tree

Estos serán os índices por defecto, e teñen un bo rendemento para claves primarias e consultas altamente selectivas. Utilizados como índices concatenados, poden recuperar datos ordeados polas columnas indexadas.



Dentro dos índices B-Tree temos os seguintes subtipos:

- **Táboas organizadas por índices**: Os datos estarán dentro do propio índice. En lugar de ter un apuntador a o bloque da datos que contén esa fila, a propia fila está almacenada nas follas do índice.
- **Índices de clave inversa**: Os bytes da clave do índice están invertidos, por exemplo, 21 almacenarase com o 12 (en hexadecimal). Isto distribúe dun xeito máis uniforme o índice a través dos bloques de datos. Especialmente útil en accesos secuenciais con RAC.
- **Índices descendentes**: Almacena os datos do índice (sobre unha ou varias columnas) nunha orde descendente.
- **Índices B-Tree en cluster**: Utilizanse para táboas en cluster. En lugar de apuntar a unha fila, a clave apunta ao bloque que contén a fila relacionada coa clave do cluster.

Indices de bitmap

Utiliza un mapa de bits para apuntar a multiples filas a diferencia dos B-Tree que apuntan a unha única fila. Exemplo de índice Bitmap:

```
SQL> SELECT cust_id, cust_last_name, cust_marital_status, cust_gender
 2  FROM   sh.customers
 3  WHERE  ROWNUM < 8 ORDER BY cust_id;
  CUST_ID CUST_LAST_ CUST_MAR C
   ---------- ---------- -------- -
        1 Kessel              M
        2 Koch                F
        3 Emmerson            M
        4 Hardy               M
        5 Gowen               M
        6 Charles    single   F
        7 Ingram     single   F
```

As columnas *cust_marital_status* e *cust_gender* teñen unha cardinalidade (número de valores diferentes) baixa e por tanto son axeitados para un índice bitmap

Examplo de bitmap

Valor	Fila1	Fila2	Fila3	Fila4	Fila5	Fila6	Fila7
M	1	0	1	1	1	0	0
F	0	1	0	0	0	1	1

Un función de mapeo converte cada bit dentro do bitmap nun **ROWID** da táboa *customers*.

As veces os bitmap pódense construir para un join entre 2 ou máis táboas. Para cada valor na columna dunha das táboas, o índice almacena os **ROWIDs** das filas correspondentes da outra táboa.

Índices baseados en funcións

Estes índices traballarán con columnas transformadas mediante unha función (por exemplo UPPER, ou ben unha expresión). O tipo de índice pode ser B-Tree ou Bitmap.

A función utilizada para construir o índice pode ser una expresión aritmética ou unha expresión que conteña unha función SQL, unha función PL/SQL, etc. Por exemplo unha función que sume os valores de 2 columnas.

Índices de dominio da aplicación

Son índices específicos adaptados para unha aplicación utilizados normalmente nos seguintes casos:

- Acomodar os índices a tipos de datos complexos como documentos, datos espaciais, imaxes, etc
- Utilizar técnicas de indexado especializadas

Ver conceptos xerais sobre índices Oracle

Enlaces de interese

Documentacion Oracle 11g Release 2

Tutorial Oracle 10g Express Edition