Administración de Oracle: Parte II

Organiza y gestiona: Grupo Cartuja Informática

Cliente: Centro de Formación del Profesorado de Sevilla

Nº de Horas: 40 horas

Fecha de inicio: 8 de Septiembre

Fecha de finalización: 18 de Septiembre.

Docente y Autor del Documento: Javier Navarro

ÍNDICE

TEMA 1: INTRODUCCIÓN

INTRODUCCIÓN..... 1-2
OBJETIVOS DEL CURSO..... 1-2
CONSIDERACIONES..... 1-2
ARQUITECTURA BÁSICA DE ORACLE..... 1-3
ESQUEMA DE LA ARQUITECTURA DE UNA BASE DE DATOS..... 1-3
SGA: SYSTEM GLOBAL AREA O SHARED GLOBAL AREA..... 1-3
PROCESOS BACKGROUND..... 1-5
FICHEROS DE LA BASE DE DATOS..... 1-8
FALLOS EN UNA BASE DE DATOS..... 1-12
FALLOS DE SENTENCIA..... 1-12
FALLOS DE PROCESO DE USUARIO..... 1-12
FALLOS DE ERROR DE USUARIO..... 1-13
FALLOS DE LA INSTANCIA..... 1-13
FALLOS DEL MEDIO FÍSICO..... 1-14
SINCRONIZACIÓN DE UNA BASE DE DATOS..... 1-15
ROLLBACK PARALELO DE INICIO RÁPIDO..... 1-15

TEMA 2: OPCIONES DE BACKUP

INTRODUCCIÓN..... 2-2
MODO NOARCHIVELOG..... 2-2
IMPLICACIONES DEL MODO NOARCHIVELOG..... 2-3
MODO ARCHIVELOG..... 2-3
IMPLICACIONES DEL MODO ARCHIVELOG..... 2-3
OPCIONES DE ARCHIVADO DE REDO LOG ONLINE..... 2-4
DEFINICIÓN DEL DESTINO DE LOGS ARCHIVADOS..... 2-4
DUPLICACIÓN DE LOGS ARCHIVADOS..... 2-4
ACTIVADO DEL MODO ARCHIVELOG..... 2-6
PROCESO DE ARCHIVADO AUTOMÁTICO O MANUAL..... 2-6
INFORMACIÓN DEL ARCHIVADO DE REDO LOG..... 2-7

TEMA 3: BACKUP

INTRODUCCIÓN..... 3-2
BACKUP FÍSICO CON BD CERRADA..... 3-2

VENTAJAS..... 3-3
DESVENTAJAS..... 3-3
INFORMACIÓN DE ARCHIVOS DE LA BASE DE DATOS..... 3-3
COMO REALIZAR UNA COPIA DE SEGURIDAD DE BD CERRADA..... 3-4
BACKUP FÍSICO CON BD ABIERTA..... 3-4
VENTAJAS..... 3-4
DESVENTAJAS..... 3-4
COMO REALIZAR UNA COPIA DE SEGURIDAD DE BD ABIERTA..... 3-5
INFORMACIÓN SOBRE BACKUP..... 3-5
COPIA DE SEGURIDAD DEL ARCHIVO ARCHIVO DE CONTROL..... 3-6
COPIAS DE SEGURIDAD DE TABLESPACE DE SÓLO LECTURA..... 3-7

TEMA 4: RECUPERACIÓN

INTRODUCCIÓN..... 4-2
COMANDO DE RECUPERACIÓN..... 4-2
RECUPERACIÓN COMPLETA EN MODO NOARCHIVELOG..... 4-3
COMO REALIZAR UNA RECUPERACIÓN COMPLETA EN MODO NOARCHIVELOG..... 4-3
RECUPERACIÓN COMPLETA EN MODO ARCHIVELOG..... 4-4
RECUPERACIÓN CON BASE DE DATOS MONTADA..... 4-5
RECUPERACIÓN CON BD ABIERTA INICIALMENTE ABIERTA..... 4-5
RECUPERACIÓN CON BD ABIERTA INICIALMENTE CERRADA..... 4-6
RECUPERACIÓN DE ARCHIVOS SIN COPIA DE SEGURIDAD. 4-7
RECUPERACIÓN ANTE FALLO DE COPIA DE SEGURIDAD..... 4-7
PROBLEMAS CON LOS ARCHIVOS REDO LOG..... 4-8
RECUPERACIÓN DE REDO LOG INACTIVOS..... 4-8
RECUPERACIÓN INCOMPLETA..... 4-9
COMANDO DE RECUPERACIÓN INCOMPLETA..... 4-10
INSTRUCCIONES GENERALES PARA LA RECUPERACIÓN INCOMPLETA..... 4-11
RECUPERACIÓN INCOMPLETA BASADA EN EL TIEMPO..... 4-12
RECUPERACIÓN INCOMPLETA BASADA EN CANCELACIÓN.. 4-13
RECUPERACIÓN INCOMPLETA BASADA EN MODIFICACIÓN..... 4-14

TEMA 5: EXPORT/IMPORT

INTRODUCCIÓN..... 5-2
MÉTODOS DE EXPORTACIÓN..... 5-2
EXPORTACIÓN INTERACTIVA SIN PARÁMETROS..... 5-3
EXPORTACIÓN CONTROLADA POR PARÁMETEOS..... 5-4
EXPORTACIÓN CON EL PARÁMETRO PARFILE..... 5-7
MODOS DE EXPORTACIÓN..... 5-7
MODO TABLA..... 5-8
MODO USUARIO..... 5-9
MODO DE BASE DE DATOS COMPLETA..... 5-10
MÉTODOS DE IMPORTACIÓN..... 5-11
IMPORTACIÓN INTERACTIVA SIN PARÁMETROS..... 5-11
IMPORTACIÓN CONTROLADA POR PARÁMETEOS..... 5-13
IMPORTACIÓN CON EL PARÁMETRO PARFILE..... 5-15
MODOS DE IMPORTACIÓN..... 5-17
MODO TABLA..... 5-17
MODO USUARIO..... 5-18
MODO DE BASE DE DATOS COMPLETA..... 5-19

TEMA 6: RECOVERY MANAGER

INTRODUCCIÓN..... 6-2
FUNCIONES Y MEJORAS DE RECOVERY MANAGER..... 6-3
CONCEPTOS GENERALES DE RMAN..... 6-3
PAQUETES DE RECOVERY MANAGER..... 6-4
CONSIDERACIONES DE CONFIGURACIÓN DE RECOVERY MANAGER..... 6-5
CATÁLOGO DE RECUPERACIÓN..... 6-5
FASES DE RECOVERY MANAGER..... 6-6
COMPILACIÓN..... 6-6
EJECUCIÓN..... 6-7
TIPOS DE COMANDOS..... 6-7
COMANDOS STAND ALONE..... 6-7
COMANDOS DE TRABAJO..... 6-8
OTROS COMANDOS..... 6-8
MODOS DE OPERACIÓN DE RECOVERY MANAGER..... 6-9
CONEXIÓN A RECOVERY MANAGER..... 6-10
EJEMPLOS DE COMANDOS..... 6-12

TEMA 7: CATÁLOGO DE RMAN

INTRODUCCIÓN..... 7-2
CONSIDERACIONES SOBRE EL CATÁLOGO DE RECUPERACIÓN..... 7-2
CREACIÓN DEL CATÁLOGO DE RECUPERACIÓN..... 7-3
REGISTRAR BASES DE DATOS DESTINO..... 7-4
MANTENIMIENTO DEL CATÁLOGO DE RECUPERACIÓN..... 7-5
COMANDO RESYNC..... 7-5
COMANDO CATALOG..... 7-6
COMANDO CHANGE..... 7-6
INFORMACIÓN SOBRE EL CATÁLOGO DE RECUPERACIÓN..... 7-8
COMANDO RESET DATABASE..... 7-9
COPIAS DE SEGURIDAD Y RECUPERACIÓN DEL CATÁLOGO DE RECUPERACIÓN..... 7-10

TEMA 8: BACKUP CON RMAN

INTRODUCCIÓN..... 8-2
COPIA DE SEGURIDAD DE BASE DE DATOS ENTERA..... 8-3
TIPOS DE COPIAS DE SEGURIDAD..... 8-4
CANALES..... 8-5
ETIQUETAS..... 8-6
COPIAS DE IMAGEN..... 8-7
COMANDO COPY..... 8-8
PROCESO DE COPIA DE IMAGEN..... 8-8
PARALELIZACIÓN DEL PROCESO DE COPIA DE IMAGEN..... 8-9
COMO REALIZAR UNA COPIA DE TODOS LOS ARCHIVOS DE DATOS Y EL ARCHIVO DE CONTROL..... 8-10
JUEGOS DE COPIAS DE SEGURIDAD..... 8-10
CONSIDERACIONES SOBRE UN JUEGO DE COPIAS DE SEGURIDAD..... 8-11
COMANDO BACKUP..... 8-12
PARALELIZACIÓN DE JUEGOS DE COPIA DE SEGURIDAD.... 8-14
PIEZAS DE JUEGOS DE COPIAS DE SEGURIDAD..... 8-15
JUEGO DE COPIAS DE SEGURIDAD DE ARCHIVOS DE DATOS..... 8-16
JUEGO DE COPIAS DE SEGURIDAD DE LOG ARCHIVADOS... 8-17
COPIAS DE SEGURIDAD COMPLETAS..... 8-18
COPIAS DE SEGURIDAD INCREMENTALES..... 8-19
COPIAS DE SEGURIDAD INCREMENTALES ACUMULATIVAS..... 8-21
RESTRICCIONES DE JUEGOS DE COPIAS DE SEGURIDAD... 8-22
ESCENARIOS DEL JUEGO DE COPIAS DE SEGURIDAD..... 8-22

DETECCIÓN DE CORRUPCIÓN..... 8-26
VISTAS DEL DICCIONARIO DE DATOS..... 8-27

TEMA 9: RESTAURACIÓN Y RECUPERACIÓN CON RMAN

INTRODUCCIÓN..... 9-2
RESTAURACIÓN DE ARCHIVOS DE DATOS, FICHEROS DE CONTROL Y REDO LOG ARCHIVADOS..... 9-2
RESTAURACIÓN DE UNA BASE DE DATOS..... 9-2
RESTAURACIÓN DE TABLESPACES Y ARCHIVOS DE DATOS..... 9-3
RESTAURACIÓN DE ARCHIVOS DE CONTROL..... 9-5
RESTAURACIÓN DE REDO LOG ARCHIVADOS..... 9-7
RECUPERACIÓN DE ARCHIVOS DE DATOS..... 9-8
RECUPERACIÓN COMPLETA..... 9-8
RECUPERACIÓN INCOMPLETA..... 9-12
DIFERENTES ESCENARIOS DE RESTAURACIÓN Y RECUPERACIÓN..... 9-14
RESTAURACIÓN DE UNA BASE DE DATOS EN MODO NOARCHIVELOG..... 9-14
RESTAURACIÓN DE UNA BASE DE DATOS A UNA ENCARNACIÓN ANTERIOR..... 9-15
RECUPERACIÓN DE UN ARCHIVO DE DATOS SIN UN BACKUP..... 9-17

TEMA 10: TEMAS ADICIONALES

INTRODUCCIÓN..... 10-2
RECUPERACIÓN DE ARCHIVOS DE DATOS NO ESENCIALES..... 10-2
EL ARCHIVO LOG DE ALERTAS..... 10-3
LOS ARCHIVOS DE RASTREO..... 10-4
DETECCIÓN Y REPARACIÓN DE CORRUPCIÓN DE BLOQUES..... 10-4
PARÁMETROS DE INICIALIZACIÓN DB_BLOCK_CHECKSUM Y DB_BLOCK_CHECKING..... 10-4
PAQUETE DBMS_REPAIR..... 10-5
COMANDO ANALYZE VALIDATE STRUCTURE..... 10-11
UTILIDAD DBVERIFY..... 10-12
UTILIDAD LOGMINER..... 10-13
CÓMO REALIZAR UN ANÁLISIS CON LA HERRAMIENTA LOGMINER..... 10-13

Tema 1
INTRODUCCIÓN

Introducción

OBJETIVOS DEL CURSO

- Describir las estructuras y procesos de Oracle para realizar copias de seguridad y recuperaciones de la base de datos.
- Aprender los diferentes tipos de copia de seguridad y recuperaciones.
- Visión general de la utilidad Oracle Recovery Manager.

CONSIDERACIONES

Una de las principales tareas del administrador de una base de datos es mantener la disponibilidad de la misma. Y en caso de fallos será su tarea garantizar la mínima pérdida de información.

El DBA deberá realizar copias de seguridad de la base de datos para asegurarse de poder poner en marcha la base de datos ante cualquier fallo que se produzca.

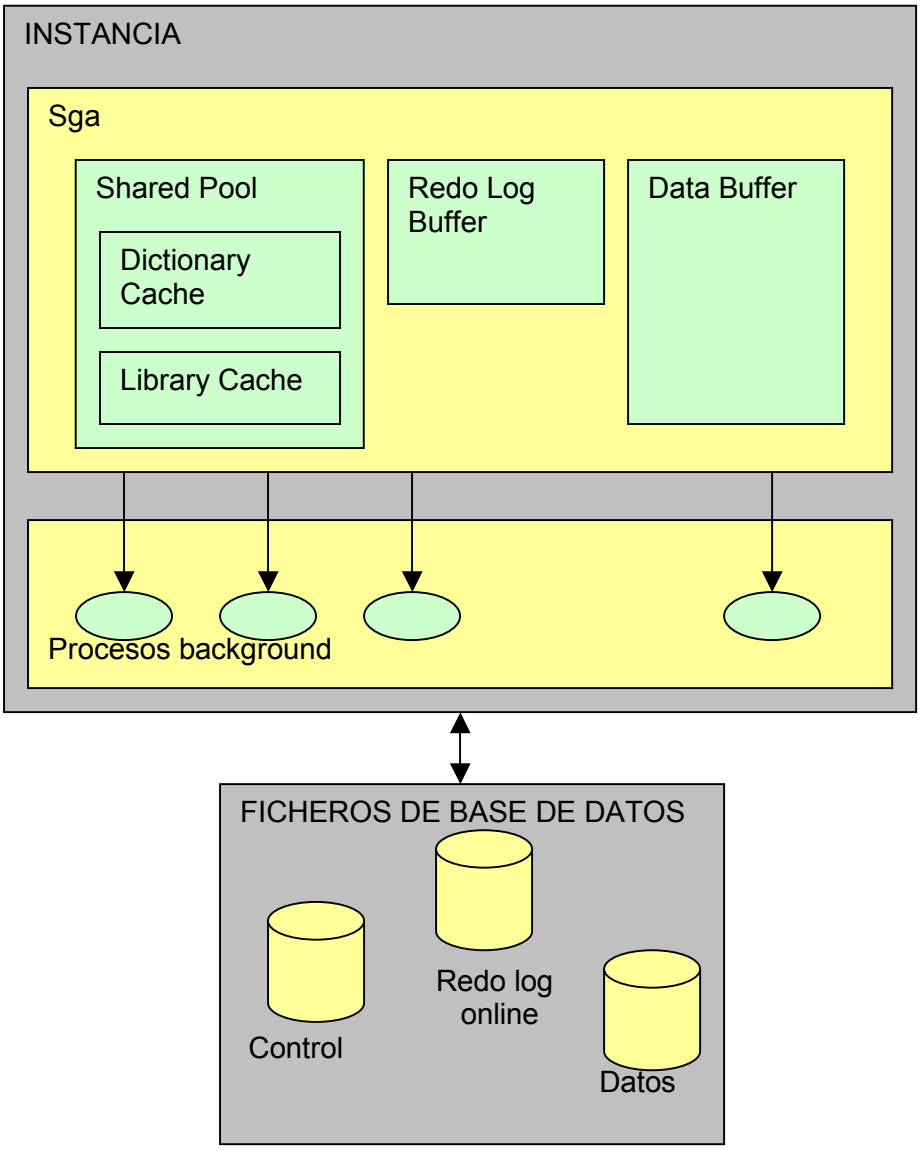
Existen dos parámetros muy importantes que debe tener en cuenta el administrador de la base de datos:

- Tiempo medio de recuperación (MTTR): El administrador de la base de datos debe procurar en la medida de lo posible minimizar el tiempo en que la base de datos no está disponible. En definitiva debe reducir lo máximo posible el tiempo de recuperación.
- Tiempo medio entre fallos (MTBF): Es otra tarea importante para el administrador proteger a la base de datos contra varios tipos de fallos. Debería configurar la base de datos de forma que no se produzcan estos fallos a menudo y así aumentar el tiempo medio entre fallos.

Además, es tarea muy importante la de revisar periódicamente la validez de la estrategia de seguridad y recuperación acordada entre el administrador de la base de datos y la dirección de la empresa.

Arquitectura Básica de Oracle

ESQUEMA DE LA ARQUITECTURA DE UNA BASE DE DATOS:



SGA: SYSTEM GLOBAL AREA o SHARED GLOBAL AREA

Es una estructura de memoria compartida, un área de memoria que contiene datos e información de control del servidor.

Oracle asigna memoria a la SGA cuando la instancia se arranca y la libera cuando se para.

Es memoria compartida, dentro de la memoria de la máquina y no paginable.

Cada instancia tiene su propia SGA.

SGA = SHARED POOL + DATABASE BUFFER CACHE + REDO LOG BUFFER.

SHARED POOL

Almacena el DICCIONARIO DE DATOS (DATA DICTIONARY CACHE) y las sentencias SQL más recientemente utilizadas (SHARED SQL o LIBRARY CACHE).

Es donde tiene lugar la fase de PARSING o ANALISIS de las sentencias SQL.

Su tamaño se define con el parámetro SHARED_POOL_SIZE del fichero de parámetros INIT.ORA y no debería ser mayor al 50% de la memoria disponible de la máquina.

DATABASE BUFFER

Almacena los datos más recientemente utilizados.

Si al hacer una petición al sistema, los datos están ya en memoria por una petición anterior, se evita una entrada a disco, por lo que el rendimiento será mejor (compartir información).

El tamaño del buffer se define en el parámetro DB_BLOCK_BUFFERS.

También contiene los BUFFER DE ROLLBACK, que almacenan la imagen anterior de los datos, y que proporcionan consistencia en lectura.

REDO LOG BUFFER.

Su funcionamiento es circular y dinámico. Son de acceso secuencial y graban todos los cambios hechos a la base de datos con el mínimo de información necesaria.

Su función es la de proporcionar seguridad, frente a una caída de la BD.

Su tamaño se define con el parámetro LOG_BUFFER.

La SGA se gestiona mediante el algoritmo LRU, las sentencias y datos menos recientemente utilizados son eliminados de la memoria para permitir la entrada de nuevos datos.

CONJUNTO GRANDE.

El conjunto es una zona opcional de memoria. La utiliza la herramienta RMAN para E/S cuando realiza copias de seguridad y recuperaciones. El tamaño de esta área de memoria se define con el parámetro LARGE_POOL_SIZE. Para utilizar esta zona de memoria hay que definir los parámetros de inicialización DBWR_IO_SLAVES o BACKUP_TAPE_IO_SLAVES.

```
SQL > SHOW  SGA;

SQL > SELECT * FROM V$SGA;
```

PROCESOS BACKGROUND.

Son procesos de sistema, encargados de ejecutar las funciones comunes que son necesarias para el servicio de respuesta a los usuarios.

Dependen de la configuración del server.

Hay cinco obligatorios:

- **PMON**

  Recupera el proceso cuando hay un fallo en el proceso usuario. Libera la cache y libera los recursos asignados a ese proceso usuario.

- **SMON**

  Recupera la instancia (lee el controlfile). Limpia los segmentos temporales no usados y recupera las transacciones ante una caída del sistema.

  Compacta el espacio libre en los ficheros de datos.

- **DBWR**

  Escribe los datos modificados del buffer de datos a los ficheros de datos.

- **LGWR**

  Graba los cambios que se registran en el buffer de redo log a los ficheros de redo log.

- **CKPT**

  Modifica el estado de la información de la BD (cabeceras de ficheros), cuando hay un checkpoint o un log switch(llenado de redo log).

Según la configuración del server, existen otros procesos background:

- **RECO**: si existe opción distribuida. Recupera las transacciones en una base de datos distribuida. Requiere el parámetro DISTRIBUITED_TRANSACTION > 0.
- **LCKn**: para realizar bloqueos en Parallel Server.
- **Pnnn** : Para realizar consultas en paralelo. (Parallel Query)
- **Dnnn**: si estamos utilizando la arquitectura MULTITHREAD.
- **SNPn**: para la gestión de réplicas, ya que refresca los snapshots. También gestiona los lob y las colas de replicación.
- **ARCn**: Este proceso archiva automáticamente los redo log online antes de que se pueda volver a

utilizar el log, con el fin de proteger todos los cambios realizados en la base de datos. Sólo actúa cuando la base de datos funciona en modo ARCHIVELOG.

```
SQL > SELECT * FROM V$PROCESS;

SQL > SELECT * FROM V$BGPROCESS;
```

FICHEROS DE LA BASE DE DATOS.

Representan el almacenamiento físico de la información. Son:

- **FICHEROS DE DATOS.**

Almacenan el Diccionario de Datos, los objetos de usuario (segmentos de tablas, índices…), y la imagen anterior de los bloques de datos que se han modificado en las transacciones (segmentos de rollback).

Están divididos en bloques.

Los bloques Oracle son la unidad mínima de E/S. (2K –4K)

Oracle formatea sus ficheros a bloques propios independientemente del SO.

El primer bloque almacena información de control, no se emplea para almacenar datos, es la CABECERA de FICHERO.

Normalmente lo que hace Oracle es una E/S multibloque basada en el parámetro MULTIBLOCK_READ_COUNT.

Para obtener información de los Ficheros de Datos en el DD, consultar la tabla DBA_DATA_FILES o bien V$DATAFILE.

- **FICHEROS DE REDO LOG.**

Graban todas las modificaciones que sufre la base de datos, para seguridad y recuperación en caso de caída.

Funcionan circularmente y se sobrescriben. Mínimo deben existir 2, aunque la instalación por defecto es de 3.

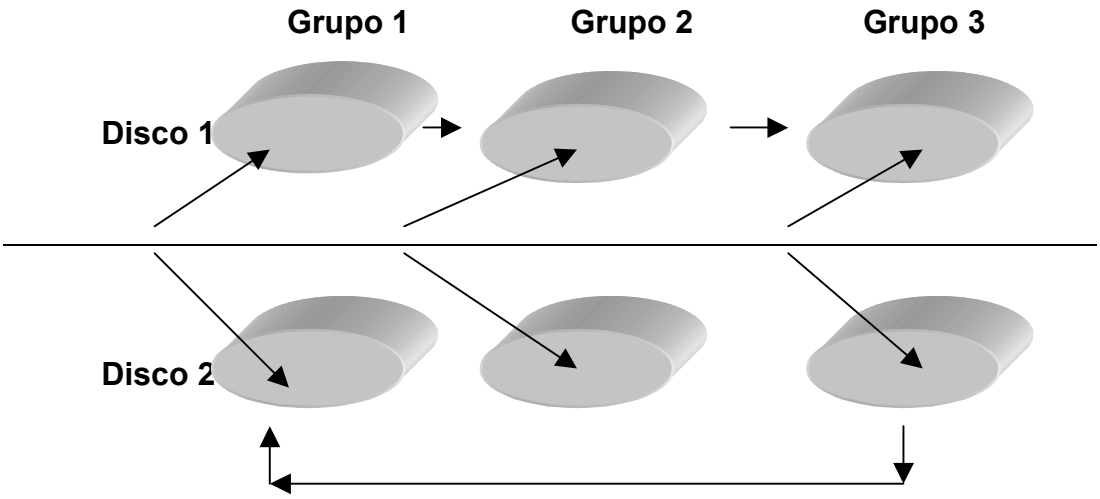
Se recomienda trabajar con ficheros de Redo Log MULTIPLEXADOS en espejo, de forma que la información es escrita en varios ficheros, a ser posible en distintos discos.

Son n grupos de n miembros a modo de espejo y en discos distintos.

Son de acceso secuencial, por lo que interesa ponerlos en dispositivos rápidos.

Se vacían al hacer COMMIT o cuando se llena el buffer a un tercio.

Encontramos información sobre ellos en V$LOGFILE y V$LOG.



- **FICHEROS DE CONTROL.**

Almacenan la información (fichero binario) para mantener y verificar la integridad de la Base de Datos: estructura, nombre y fecha de creación de la base de datos, nombre de los ficheros de datos y de los ficheros de redo log…

Guarda también información de SINCRONIZACIÓN: estado de los ficheros cuando se cerró la base de datos, tiempo de cierre…

Se identifican en el parámetro CONTROL_FILES.

También es necesario para recuperación.

Necesita un fichero, pero es mejor tener copias en discos distintos.

Los puntos críticos de fallos son el CONTROL FILE y los Ficheros de REDO LOG.

Si se pierde el fichero de redo log en uso, sólo un backup completo de la base de datos recuperaría hasta ese momento.

Si falla otro de los ficheros de redo log, Oracle lo ignora y pasaría a otro, por lo que es importante que existan 3 o 4.

El tamaño de los ficheros de redo log depende mucho de cada situación. Si es muy grande, en un recovery, tardaría mucho tiempo en arrancar.

Oracle recomienda que se les dé un tamaño que permita que cada redo log tarde en llenarse unos 15 ó 20 minutos.

Se pueden añadir redo log on line (flexibilidad de Oracle).

Respecto al control file es interesante tener una copia guardada en otro dispositivo. En caso de fallo, modificando en el init.ora el fichero de control con el debe arrancar, leería ese control file.

- **OTROS FICHEROS.**

  - **FICHERO DE PARÁMETROS INITsid.ORA.**

    Define las características de la instancia y se lee cuando esta arranca, configurándola según el valor de los parámetros.

  - **FICHERO ALERTsid.LOG.**

Contiene errores internos, redo log, checkpoint, DDL, modificaciones del init.ora, corrupciones…

Por defecto se encuentra en $ORACLE_HOME / RDBMS / log.

Podemos cambiar su destino con el parámetro BACKGROUND_DUMP_DEST del fichero init.ora.

- **FICHEROS DE TRAZAS.**

Para seguimiento a nivel de aplicaciones. Es generado por el proceso servidor. Son ficheros TCR

Se puede activar por sesión con:

```
ALTER SESSION SET SQL_TRACE = TRUE;
```

Su destino se puede modificar en USER_DUMP_DEST del init.ora.

- **FICHERO DE PASSWORD.**

Valida los usuarios de la base de datos. Hay que emplear la utilidad ORA8PWD, del núcleo para crearlo.

- **FICHERO O ARCHIVO DE REDO LOG.**

Implica que la base de datos trabaje en modo ARCHIVELOG.

Permite almacenar offline copias de los ficheros de Redo Log.

Cuando en online el fichero de redolog debe sobrescribirse, se hace una copia en los archivos.

En caso de caída se puede recuperar la base de datos a un estado consistente y completo hasta el momento del fallo.

Fallos en una base de datos

En una base de datos pueden ocurrir diferentes fallos. Según el origen de estos fallos, pueden clasificarse en diferentes categorías:

Fallos de sentencia

Un fallo de sentencia se puede producir por los siguientes motivos:

- Error lógico de la aplicación.
- Violación de restricciones de integridad.
- Insuficientes privilegios del usuario.
- Insuficiente espacio libre en un tablespace.

Ante uno de estos errores, el DBA podrá realizar una de estas acciones:

- Corregir la aplicación para que realice la operación correctamente.
- Modificar la sentencia y volver a lanzarla.
- Proporcionar los privilegios necesarios.
- Añadir espacio a los tablespaces necesarios.

Fallos de proceso de usuario

Un proceso de usuario es otro proceso que conforma la arquitectura básica de Oracle. Se crea cuando un usuario realiza una conexión a la base de datos a través de cualquier herramienta. Proporciona la interfaz que permite al usuario lanzar comandos contra la base de datos.

El proceso de usuario puede fallar por los siguientes motivos:

- La sesión del usuario ha terminado de forma anormal.
- El usuario realiza una desconexión anormal.
- El programa de usuario provoca una excepción que termina la sesión.

El DBA pocas veces podrá actuar ante un fallo de proceso de usuario. Generalmente el proceso background PMON se encarga de realizar una limpieza tras el fallo de un proceso de usuario. Deshace la transacción del proceso terminado anormalmente, libera los recursos y bloqueos que pudiera tener el proceso que ha fallado.

Fallos por error de usuario

Un fallo de este tipo ocurre cuando:

- El usuario borra o trunca una tabla accidentalmente.
- El usuario elimina las filas necesarias de una tabla.
- El usuario validó los cambios que llevaba realizados y detectó que había un error en los mismos.

Ante este tipo de fallos, la acción del DBA es necesaria y en todo momento debe procurar minimizar errores de usuario asegurándose que estos están formados adecuadamente y conocen las implicaciones de disponibilidad e integridad de la base de datos.

Las soluciones a errores de usuario pueden lleva a realizar una recuperación a partir de una copia de seguridad válida de la base de datos.

Fallos de la instancia

Hay muchas situaciones en las que puede producirse un error de la instancia:

- Problemas hardware.
- Cortes de fluido electrico.
- Fallo de algún proceso background.

La recuperación de la instancia debe realizarla el DBA iniciando la instancia de nuevo con la sentencia STARTUP y analizar posteriormente la causa del error en el archivo de alerta. El estado nuevo de la instancia será el inmediatamente anterior consistente al fallo. Es el proceso SMON el que se encarga de realizar la recuperación durante el inicio de la instancia. Ahora bien, los usuarios deberán volver a introducir los datos que no hubieran validado antes de que se produjera el error.

Las fases que se realizan para recuperar la instancia cuando el DBA lanza la sentencia STARTUP son las siguientes:

1.- Determinar si existen datos no validados.

2.- El proceso DBWR escribe los datos validados y no validados en los archivos de datos. En definitiva, aplicar todos los cambios registrados en los archivos de redo log a los ficheros de datos. Esta es la fase de roll forward.

3.- Para eliminar datos no validados, se realiza una fase de rollback. Para ello se utilizan los segmentos de rollback rellenados antes de producirse el error. Durante esta fase la base de datos está ya disponible a excepción de los bloques que participan en el rollback. Esto se debe al rollback paralelo de inicio rápido presentado en la versión Oracle 7.1.3.

Al finalizar el rollback, en los discos sólo residen datos validados y los ficheros de datos están sincronizados.

Fallos del medio físico

Un fallo en el medio físico se produce cuando existe un problema físico al leer o escribir en un archivo de la base de datos. Este tipo de fallos es el más grave y normalmente tendrá que intervenir el administrador de la base de datos para solucionarlo.

La recuperación dependerá del método con que se realizara la copia de seguridad y de si se dispone o no de archivos redo log archivados para recuperar todos los datos validados hasta el momento del fallo.

Sincronización de una base de datos

Una base de datos debe estar sincronizada para poder abrirse. Es decir, los archivos de datos, redo log y control deben estar sincronizados. Si no estuvieran sincronizados, sería necesaria una recuperación de la base de datos.

Todos los archivos de datos deben tener el mismo número de punto de control, con excepción de los archivos de tablespaces offline o de sólo lectura.

La sincronización se basará en los pun tos de control y secuencia de redo log actual.

Los archivos redo log online recuperan transacciones validadas y no validadas para sincronizar los archivos.

Se solicitarán los redo log archivados para realizar la recuperación.

Rollback paralelo de inicio rápido

El rollback paralelo de inicio rápido aparece en la versión de Oracle 7.3.1. Con el rollback paralelo de inicio rápido se permite trabajar a los usarios una vez terminada la fase de roll forward durante una recuperación de la base de datos. La idea es que la base de datos esté disponible a excepción claro de los bloques que vayan a participar en el rollback.

A partir de la versión 8i, Oracle permite que el proceso SMON que se encarga de la recuperación, utilice esclavos de la consulta paralela para completar la operación de rollback. El rollback paralelo se inicia automáticamente cuando SMON detecta que la transacción ha generado un número elevado de bloques de rollback.

El rollback paralelo de inicio rápido se activa con el parámetro FAST_START_PARALLEL_ROLLBACK. Los valores para este parámetro pueden ser:

- FALSE: Es el valor por defecto e indica que sólo se utilice un servidor para la recuperación paralela.
- LOW: Se utilizan 2*CPU_COUNT servidores de recuperación paralela.
- HIGH: Se utilizan 4*CPU_COUNT servidores de recuperación paralela.

Tema 2

OPCIONES DE BACKUP

Introducción

En este tema vamos a repasar los conceptos principales de archivado de la base de datos.

En primer lugar se repasan los diferentes modos en los que puede trabajar la base de datos (NOARCHIVELOG y ARCHIVELOG) y las implicaciones que conllevan en una recuperación de la base de datos.

A continuación veremos todos los aspectos relativos al archivado de los ficheros de redo log online.

Modo NOARCHIVELOG

Ya debemos saber que todo cambio que se produce en la base de datos se registran en los ficheros de redo log online. La finalidad de los estos archivos es permitir una recuperación automática de las transacciones en caso de producirse un fallo en la base de datos.

Podemos configurar la base de datos para que trabaje en modo NOARCHIVELOG (que es el modo por defecto) o bien en modo ARCHIVELOG.

Cuando decidimos configurar la base de datos para que funcione en modo NOARCHIVELOG, los ficheros de redo log online no se archivan en los denominados ficheros redo log archivados. Esto implica que en caso de necesitar una recuperación podemos perder transacciones. Esto se debe a que los archivos de redo log online se reciclan automáticamente y se van sobrescribiendo quedando disponibles para la recuperación sólo las transacciones más recientes.

Los archivos redo log online se utilizan de forma circular y se pueden sobrescribir una vez realizado un punto de control.

En este modo, cuando los ficheros log online son sobrescritos, la recuperación de la base de datos sólo será posible hasta la última copia de seguridad completa.

IMPLICACIONES DEL MODO NOARCHIVELOG

Sólo se pueden realizar copias de seguridad con la base de datos cerrada. No permite copias de seguridad online.

Una copia de seguridad deberá incluir el juego completo de ficheros de datos, de redo log y archivos de control.

En caso de necesitar una recuperación, se perderán todos los datos desde la última copia de seguridad completa.

Si un tablespace deja de funcionar debido a un fallo, no se podrá seguir utilizando la base de datos hasta eliminar el tablespace o restaurar la base de datos.

Para restaurar la base de datos, se deberán restaurar todos los archivos de datos, el archivo de control y los archivos redo log.

Modo ARCHIVELOG

Cuando configuramos la base de datos en modo ARCHIVELOG, los ficheros de redo log online se van archivando en ficheros redo log archivados.

Por este motivo, en una base de datos funcionando en modo archivelog, un fichero redo log online no podrá sobrescribirse hasta que se realice un punto de control y se archive el contenido del redo log mediante el proceso en background ARCn.

IMPLICACIONES DEL MODO ARCHIVELOG

La base de datos no perderá ningún dato ni modificación en caso de producirse un error.

Se pueden realizar copias de seguridad mientras la base de datos está online.

Cuando un tablespace deja de funcionar, el resto de la base de datos sigue disponible ya que podemos recuperar tablespace mientras la base de datos está abierta.

Puede ser necesario añadir más grupos de archivos redo log para asegurar que el archivado de los mismos se realiza con tiempo suficiente para no hacer esperar a la base de datos en caso de tener que ser reutilizados.

Podemos restaurar archivos dañados y utilizar los redo log archivados mientras la base de datos está online u offline.

Podemos restaurar la base de datos hasta un punto específico en el tiempo.

Podemos restaurar la base de datos hasta el final de un log archivado que se especifique.

Podemos recuperar la base de datos hasta un número de secuencia SCN específico.

OPCIONES DE ARCHIVADO DE REDO LOG ONLINE

DEFINICIÓN DEL DESTINO DE LOGS ARCHIVADOS

Podemos definir el destino de los archivos redo log archivados a través del parámetro de inicialización LOG_ARCHIVE_DEST.

Además podemos también especificar un formato para los nombres de los archivos nombrados con el parámetro de inicialización LOG_ARCHIVE_FORMAT.

DUPLICACIÓN DE LOGS ARCHIVADOS

Podemos tener los archivos log archivados duplicados para aumentar la seguridad en caso de fallo. Para activar la duplicación de archivos log archivados debemos especificar el parámetro LOG_ARCHIVE_DUPLEX_DEST, donde indicamos la dirección donde se van a almacenar los duplicados de los archivos de logs archivados.

Para la multiplexación de los archivos de logs archivados también disponemos de otra posibilidad y son los parámetros LOG_ARCHIVE_DEST_n. Estos son parámetros dinámicos que permiten especificar hasta cinco destinos como máximo mediante el sufijo n.

El uso de dos parámetros LOG_ARCHIVE_DEST_n equivale a utilizar los parámetros LOG_ARCHIVE_DEST y LOG_ARCHIVE_DUPLEX_DEST. Es recomendable utilizar los parámetros LOG_ARCHIVE_DEST_n ya que permiten la archivación en ubicaciones remotas.

No se pueden utilizar conjuntamente los dos métodos.

Cuando utilizamos la opción de los parámetros LOG_ARCHIVE_DEST_n podemos especificar los destinos como obligatorios u opcionales. Si elegimos un destino MANDATORY, el archivado en este destino debe terminar forzosamente y de manera correcta antes de sobrescribir el redo log online.

Si elegimos la opción OPCIONAL, que es la opción por defecto, no obligamos a que el archivado haya terminado correctamente para volver a utilizar el redo log online.

Disponemos también de otro parámetro, LOG_ARCHIVE_MIN_SUCCED _DEST, que especifica el límite inferior de destinos obligatorios locales. Es decir, si asignamos un valor 3 a este parámetro, no se volverá a utilizar el redo log online hasta que se hayan realizado las tres copias. Ahora bien, este parámetro está directamente relacionado con los valores MANDATORY u OPTIONAL de los parámetros LOG_ARCHIVE_DEST_n. Si el valor de LOG_ARCHIVE_MIN_SUCCED_DEST es menor que el número de destinos obligatorios, será ignorado este parámetro. Si por el contrario es mayor, el número de destinos locales para el archivado de redo log online deberá ser al menos igual al valor de este parámetro.

El estado de un destino de archivo, por defecto está ENABLE. Esto indica que es un destino de archivo utilizable y viene especificado en los parámetros LOG_ARCHIVE_DEST_STATE_n. Podemos modificar el valor de estos parámetros a un valor DEFER para hacer temporalmente un destino inutilizable. Esto es útil cuando se detecta un error en la escritura en un destino determinado.

Para modificar el valor dinámicamente, hay que lanzar la sentencia:

```
ALTER SYSTEM SET LOG_ARCHIVE_DEST_STATE_1 = DEFER;
```

ACTIVADO DEL MODO ARCHIVELOG

Para cambiar la base a modo ARCHIVELOG, el administrador deberá seguir los siguientes pasos:

1.- Preparar el sistema operativo creando los directorios de destino de los archivos redo log archivados.

2.- Modificar el archivo INIT.ora, estableciendo el parámetro LOG_ARCHIVE_START a TRUE (si queremos una archivación automática, que es la opción recomendada) y los parámetros relativos a los destinos de los redo log archivados.

3.- Cerrar la base de datos.

4.- Iniciar la base de datos en modo MOUNT:

```
STARTUP MOUNT;
```

5.- Activar el modo ARCHIVELOG:

```
ALTER DATABASE ARCHIVELOG;
```

6.- Abrir la base de datos:

```
ALTER DATABASE OPEN;
```

PROCESO DE ARCHIVADO AUTOMÁTICO O MANUAL

Al definir la base de datos en modo ARCHIVELOG, hay que realizar el archivado de los ficheros de redo log online. Este proceso de archivado puede realizarse de forma manual o automática. Si definimos el parámetro LOG_ARCHIVE_START a FALSE, deberá ser el administrador el que realice de forma manual el archivado de los redo log online.

En cambio, si se define el parámetro a valor TRUE, se activará el proceso background ARCn para que realice las copias de los ficheros. Esta es la opción recomendada.

Por defecto se activa un único proceso ARC0, pero se pueden activar más procesos mediante el parámetro LOG_ARCHIVE_MAX_PROCESES. El máximo número de procesos permitido es 10. Esto permite tener un número dinámico de procesos de archivado utilizando la sentencia ALTER SYSTEM SET.

Para activar el archivado automático en una instancia ya abierta en modo ARCHIVELOG habrá que lanzar la siguiente sentencia:

```
ALTER SYSTEM ARCHIVE LOG START TO ruta;
```

Para realizar el archivado manual de los redo log online hay que utilizar la siguiente sentencia:

ALTER SYSTEM ARCHIVE LOG opcion;

Donde 'opcion' puede ser:

SÉQUENCE n: Archiva el grupo de redo log online que tienen el identificador de secuencia n.

LOGFILE nombre: Archiva el grupo de redo log online identificado por el nombre de alguno de sus miembros.

GROUP n: Archiva el grupo de redo log online identificado con el número de grupo n.

NEXT: Archiva el grupo redo log online más antiguo sin archivar.

ALL: Archiva todos los grupos de redo log.

INFORMACIÓN DEL ARCHIVADO DE REDO LOG

V$ARCHIVED_LOG: Muestra información del archivado de redo log que contiene el fichero de control.

V$ARCHIVE_DEST: Muestra información relativa al destino de los archivados de redo log.

V$DATABASE: Muestra el estado actual del archivado.

V$ARCHIVE_PROCESS: Muestra información de los procesos ARCn.

Además de estas vistas dinámicas, podemos utilizar el comando ARCHIVE LOG LIST en el prompt de SQL*Plus que nos proporciona la información más relevante del archivado.

```
SQL> ARCHIVE LOG LIST
Modo log de la base de datos          Modo de No Archivado
Archivado automático                  Desactivado
Destino del archivo                   C:\Oracle\Ora81\RDBMS
Secuencia de log online más antigua   120
Secuencia de log actual               122
```

Tema 3

BACKUP

Introducción

En este tema vamos a introducir los diferentes métodos de copias de seguridad para una base de datos. Las opciones que se desarrollan a lo largo del tema se muestran en el siguiente esquema:

- BACKUP
  - Físico
    - BD cerrada
    - BD abierta
      - A nivel de tablespace
      - A nivel de ficheros
  - Lógico (export/import)

Nota: El backup lógico se verá en una lección posterior.

Backup físico con BD cerrada

Una copia de seguridad con la base de datos cerrada consiste en realizar una copia desde el sistema operativo de todos los archivos que conforman la base de datos, archivo de control, archivos de datos, archivo de parámetros, archivo de contraseñas.

Cuando se realiza una copia de seguridad con la base de datos cerrada, no es necesario incluir copias de los archivos de redo log online siempre y cuando se haya cerrado la base de datos en modo normal, transaccional o inmediato.

Este método de copia de seguridad se puede realizar tanto si la base de datos está en modo ARCHIVELOG como en NOARCHIVELOG. La única diferencia estará en que si la base de datos está en modo ARCHIVELOG también habrá

que realizar copias desde el sistema operativo de los ficheros redo log archivados.

VENTAJAS

- Es una operación sencilla. Sólo debe seguir los siguientes pasos:
  - Cerrar la base de datos.
  - Copiar desde el sistema operativo los archivos.
  - Abrir la base de datos.
- Se puede automatizar el proceso de backup mediante un script del sistema operativo.
- Toda la información archivada es consistente hasta el punto en el que se cerró la base de datos.

DESVENTAJAS

- La base de datos estará inactiva durante el proceso de copia de archivos.
- El tiempo de inactividad depende del número de archivos que conforman la base de datos. El tiempo puede ser considerable.

INFORMACIÓN DE ARCHIVOS DE LA BASE DE DATOS

Para Realizar la copia de seguridad de la base de datos desde el sistema operativo, tendremos que conocer el nombre y ubicación de los archivos de los que vamos a realizar el backup. Para obtener esta información disponemos de las siguientes vistas dinámicas:

V$DATAFILE: Contiene información de los ficheros de datos.

V$CONTROLFILE: Contiene información de los archivos de control.

V$LOGFILE: Contiene información de los archivos redo log online.

COMO REALIZAR UNA COPIA DE SEGURIDAD DE BD CERRADA

1.- Obtenga la información con las vistas anteriores de la ubicación y nombre de los archivos que vaya a incluir en la copia de seguridad. Asegúrese de mantener la información de las ubicaciones de los archivos para facilitar la recuperación si fuera necesaria.

2.- Pare la base de datos utilizando una de las siguientes opciones: NORMAL, TRANSACTIONAL o IMMEDIATE.

3.- Realice la copia de los archivos mediante un comando del sistema operativo.

4.- Abra la base de datos.

Backup físico con BD abierta

El backup físico con la base de datos abierta permite realizar copias de seguridad de archivos de la base de datos mientras ésta permanece activa. Esto es útil en empresas con una actividad continua de 24h.

Es imprescindible para realizar una copia física con la base de datos abierta, que ésta trabaje en modo ARCHIVELOG. Y asegurarse de que se realiza correctamente el archivado de los ficheros de redo log online ya sea de forma manual o automática.

VENTAJAS

- La base de datos sigue disponible para los usuarios mientras se realiza la copia.
- Puede realizarse la copia de seguridad a nivel de tablespace o a nivel de archivo.

DESVENTAJAS

- EL proceso de backup en este modo es más complejo que en el modo de base de datos cerrada. Requiere más experiencia del administrador.
- El uso de este método de copia de seguridad puede afectar al tamaño o número de los redo log online. Cuando hacemos la copia de un archivo de datos, éste queda bloqueado y será necesario mantener una imagen completa de los bloques de datos en el redo log ya que no será suficiente con la información de fila.

COMO REALIZAR UNA COPIA DE SEGURIDAD DE BD ABIERTA

1.- Mediante el comando ALTER TABLESPACE ... BEGIN BACKUP, ponemos al tablespace indicado en modo de copia de seguridad.

2.- Con un comando del sistema operativo realizamos la copia de los archivos del tablespace que se requieran.

3.- Mediante el comando ALTER TABLESPACE ... END BACKUP, finalizamos el modo de copia de seguridad del tablespace.

4.- Lanzar el comando ALTER SYSTEM SWITCH LOGFILE. Este comando sincroniza los archivos de datos a los números de secuencia actuales. Mientras el tablespace está en modo de copia, no se varían los números de secuencia en los archivos de datos.

INFORMACIÓN SOBRE BACKUP

V$BACKUP: Esta vista nos permite saber el estado de todos los archivos de datos de la base de datos. Podemos averiguar si un archivo está en modo de copia de seguridad o no.

```
SQL> SELECT * FROM V$BACKUP;

    FILE# STATUS            CHANGE# TIME
---------- ------------------ ---------- --------
         1 NOT ACTIVE              0
         2 NOT ACTIVE              0
         3 NOT ACTIVE        1791587 04/09/02
         4 NOT ACTIVE              0
         5 NOT ACTIVE              0
         6 NOT ACTIVE              0
         7 NOT ACTIVE              0
         8 NOT ACTIVE              0
         9 NOT ACTIVE        1791803 04/09/02
        10 NOT ACTIVE        1791803 04/09/02

10 filas seleccionadas.
```

COPIA DE SEGURIDAD DEL ARCHIVO DE CONTROL

La mejor forma de realizar copias de seguridad del archivo de control es a través de su multiplexación en el archivo de parámetro INIT.ORA antes de inicializar la base de datos. Si la base de datos está abierta, lo mejor seguiría siendo pararla y modificar este archivo de parámetros y una vez modificado volver a abrir la base de datos. Pero, también disponemos de la opción de realizar una copia de seguridad del archivo de control con la base de datos abierta. Esto se realiza con la sentencia:

```
ALTER DATABASE BACKUP CONTROLFILE TO dirección;
```

También podemos realizar una traza del archivo de control actual. Esta traza crea un archivo con la información resumen de todos los archivos que conforman la base de datos que será la información necesaria en caso de tener que volver a crear el archivo de control. El comando para realizar la traza del archivo de control es el siguiente:

ALTER DATABASE BACKUP CONTROLFILE TO TRACE;

El contenido de un archivo de traza sería similar a:

ORA03204.TRC - Bloc de notas

Archivo Edición Formato Ver Ayuda

```
# The following commands will create a new control file and use it
# to open the database.
# Data used by the recovery manager will be lost. Additional logs may
# be required for media recovery of offline data files. Use this
# only if the current version of all online logs are available.
STARTUP NOMOUNT
CREATE CONTROLFILE REUSE DATABASE "AVALON" NORESETLOGS ARCHIVELOG
    MAXLOGFILES 32
    MAXLOGMEMBERS 2
    MAXDATAFILES 32
    MAXINSTANCES 16
    MAXLOGHISTORY 1815
LOGFILE
  GROUP 1 'C:\ORACLE\ORADATA\AVALON\REDO03.LOG'  SIZE 1M,
  GROUP 2 'C:\ORACLE\ORADATA\AVALON\REDO02.LOG'  SIZE 1M,
  GROUP 3 'C:\ORACLE\ORADATA\AVALON\REDO01.LOG'  SIZE 1M
DATAFILE
  'C:\ORACLE\ORADATA\AVALON\SYSTEM01.DBF',
  'C:\ORACLE\ORADATA\AVALON\RBS01.DBF',
  'C:\ORACLE\ORADATA\AVALON\USERS01.DBF',
  'C:\ORACLE\ORADATA\AVALON\TEMP01.DBF',
  'C:\ORACLE\ORADATA\AVALON\TOOLS01.DBF',
  'C:\ORACLE\ORADATA\AVALON\INDX01.DBF',
  'C:\ORACLE\ORADATA\AVALON\DR01.DBF',
  'C:\ORACLE\ORADATA\AVALON\OEM_REPOSITORY.ORA',
  'C:\ORACLE\ORADATA\AVALON\BEA1.ORA',
  'C:\ORACLE\ORADATA\AVALON\BEA2.ORA'
CHARACTER SET WE8ISO8859P1
```

COPIAS DE SEGURIDAD DE TABLESPACE DE SÓLO LECTURA

Cuando tenemos un tablespace en modo de sólo lectura, la única operación que necesita para realizar un backup del mismo es la copia desde el sistema operativo de los ficheros de datos que conforman el tablespace. En caso de realizar un backup posterior de la base de datos completa no será necesario incluir estos tablespaces.

Tema 4
RECUPERACIÓN

Introducción

En este tema vamos a introducir los diferentes tipos de recuperación que podemos aplicar a una base de datos que la necesita.

- RECUPERACIÓN
  - Completa
    - Modo NOARCHIVELOG
    - Modo ARCHIVELOG
  - Incompleta (Sólo en modo ARCHIVELOG)
  - Lógica (import)

Nota: La recuperación lógica se verá en una lección posterior.

Comando de recuperación

Cuando necesitemos realizar una recuperación de la base de datos por algún motivo determinado, tendremos que utilizar alguno de los siguientes comandos de recuperación:

RECOVER [AUTOMATIC] DATABASE: Para recuperaciones de bases de datos cerradas.

RECOVER [AUTOMATIC] TABLESPACE <numero> | <nombre>: Para recuperaciones con base de datos abierta.

RECOVER [AUTOMATIC] DATAFILE <numero> | <nombre>: Para recuperaciones con la base de datos abierta o cerrada.

Recuperación completa en modo NOARCHIVELOG

Una recuperación completa de la base de datos consiste en restaurar los archivos necesarios hasta el momento en que se ha producido el error.

Cuando tenemos la base de datos en modo NOARCHIVELOG, ante una recuperación necesaria por un fallo del medio físico, tendremos las siguientes ventajas:

- Es fácil de llevar a cabo. Sólo será necesario restaurar los archivos copiados en la copia de seguridad.
- El tiempo transcurrido para la recuperación es el que utilice el sistema operativo en la copia de archivos en su posición original.

Pero el modo NOARCHIVELOG implica también una serie de desventajas:

- Todos los datos cambiados desde la última copia de seguridad, se perderían siempre y cuando se hayan sobrescrito los archivos de redo log online.
- En la misma situación, si se han sobrescrito los archivos redo log online, será necesario restaurar la base de datos completa aunque sólo se haya perdido un archivo.

CÓMO REALIZAR UNA RECUPERACIÓN COMPLETA EN MODO NOARCHIVELOG

Hay que diferenciar dos casos en la recuperación de la base de datos en modo NOARCHIVELOG:

1.- Si se ha perdido un solo archivo y no se han sobrescrito los redo log online:

a.) Lo único que hay que realizar es restaurar la copia del archivo perdido en su ubicación original (copiando desde el sistema operativo el fichero). Al arrancar la base de datos de nuevo, no pide que se realice recuperación porque no es necesario la aplicación de redo log al archivo restaurado.

2.- Si se ha perdido un solo archivo y se han sobrescrito los archivos redo log online:

a.) En este caso, al intentar arrancar la base de datos una vez restaurado el archivo desde el sistema operativo, nos pedirá que realicemos la recuperación del archivo en cuestión ya que estarán descoordinados los números de secuencia de los redo log y el archivo restaurado. Al lanzar la operación de recover datafile nos exigirá aplicar los archive log necesarios, que no existen por tener la base de datos en modo NOARCHIVELOG. Será imposible realizar la recuperación completa hasta el momento del fallo.

b.) Habrá que proceder a una restauración de toda la base de datos a partir del último backup realizado y por lo tanto se perderán todos los cambios

realizados desde esta copia de seguridad. Debe cerrar la base de datos, restaurar todos los archivos de la base de datos desde el sistema operativo y volver a arrancar la instancia.

Recuperación completa en modo ARCHIVELOG

Antes de ver los diferentes casos que podemos encontrarnos ante una recuperación de la base de datos en modo ARCHIVELOG veamos las vistas que nos van a proporcionar in formación necesaria para realizar correctamente la operación de recuperación:

- V$RECOVER_FILE: Vista que muestra información sobre los archivos que necesitan recuperación.
- DBA_DATA_FILES: Las columnas file_id (identificador de archivo), file_name (nombre del archivo), tablespace_name (nombre del tablespace al que pertenece el archivo) y status (estado del archivo: AVAILABLE si está disponible) son las que pueden interesar para obtener la información que buscamos.
- V$DATAFILE: Las columnas más importantes de esta vista son: file# (identificador del archivo), name (nombre del archivo) y status (estado del archivo: RECOVER si necesita recuperación).
- V$DATAFILE_HEADER: De esta vista la columna más importante es la columna STATUS ya que nos dirá si está ONLINE u OFFLINE el archivo en caso de que el estado en V$DATAFILE sea RECOVER.

Cuando la base de datos trabaja en modo ARCHIVELOG, podemos encontrarnos en cuatro situaciones diferentes ante una recuperación de la base de datos:

RECUPERACIÓN CON BASE DE DATOS MONTADA

Esta recuperación será necesaria cuando se produzca un error en un archivo de sistema o de rollback ya que en este caso la recuperación sería imposible con la base de datos abierta.

Cuando se detecte el error en uno de estos archivos deberemos :

1.- Restaurar el archivo oportuno desde la copia de seguridad desde el sistema operativo.

2.- Montar la base de datos: STARTUP MOUNT

3.- Recuperar el fichero con el comando RECOVER DATAFILE...

4.- Abrir la base de datos: ALTER DATABASE OPEN

RECUPERACIÓN CON BD ABIERTA INICIALMENTE ABIERTA

Este tipo de recuperación debe realizarse cuando se detecta mientras funciona la base de datos que un archivo que no pertenece al tablespace system ni rollback necesita recuperación. Ante esta situación, los pasos que debe realizar son los siguientes:

1.- Comprobar a qué tablespace pertenece el archivo a recuperar.

2.- Comprobar si el archivo en cuestión ya está en estado offline. En caso negativo ponerlo en dicho estado mediante el comando:

ALTER DATABASE DATAFILE '...' OFFLINE;

3.- Restaurar el archivo desde el sistema operativo.

4.- Recuperar el archivo mediante el comando RECOVER DATAFILE '...'

5.- Volver a situar el archivo o tablespace en estado online mediante los comandos:

ALTER DATABASE DATAFILE '...' ONLINE;

ALTER TABLESPACE nombre ONLINE;

RECUPERACIÓN CON BD ABIERTA INICIALMENTE CERRADA

Esta recuperación será necesaria cuando se ha parado la base de datos y se ha producido un error en un archivo que no es de sistema ni de rollback.

1.- Montar la base de datos.

2.- Comprobar en v$datafile que el archivo esté offline, si no es así ponerlo en dicho estado con: ALTER DATABASE DATAFILE '...' OFFLINE;

3.- Abrir la base de datos: ALTER DATABASE OPEN;

4.- Restaurar el archivo desde el sistema operativo.

5.- Indicar a la base de datos si es necesario el cambio de ubicación: ALTER DATABASE RENAME FILE '...' TO '...'

6.- Recuperar el archivo con el comando RECOVER DATAFILE.

7.- Cambiar el estado del archivo a ONLINE.

RECUPERACIÓN DE ARCHIVOS SIN COPIA DE SEGURIDAD

El que no haya copia de seguridad de un tablespace hay que evitarlo procurando realizar un backup del tablespace que se acaba de crear.

Pero ante una situación de este tipo:

1.- Si la base de datos está cerrada: Montar la base de datos, desactivar el tablespace o archivo y abrir la base de datos.

Si la base de datos está abierta, debemos incluir la opción IMMEDIATE si decidimos desactivar el tablespace: ALTER TABLESPACE nombre OFFLINE IMMEDIATE;

2.- Volver a crear el archivo: ALTER DATABASE CREATE DATAFILE '...' AS '...'

3.- Recuperqar el archivo con RECOVER TABLESPACE o DATAFILE según la opción seguida.

4.- Activar el tablespace con ALTER TABLESPACE nombre ONLINE

Recuperación ante fallo de copia de seguridad

Mientras se realiza una copia de seguridad online puede producirse un fallo que provoque el cierre de la base de datos. Cuando ocurre esto, los archivos y tablespaces que estuvieran en modo de seguridad lo seguirán estando al intentar arrancar la base de datos, y quedarán desincronizados con los demás archivos de la base de datos.

La copia de seguridad que estaba en proceso deberá rechazarse y volver a realizar una copia correcta cuando se levante la base de datos.

Para ello, al arrancar la base de datos, que se quedará en estado mount al detectar la desincronización de los archivos, debemos realizar la sincronización de los archivos que estaban en modo de copia de seguridad, obligando a que dejen este estado de copia con el siguiente comando:

ALTER DATABASE DATAFILE '...' END BACKUP;

El comando ALTER TABLESPACE ... END BACKUP no estará disponible en el estado mount de la base de datos.

Después de realizar esta operación de finalización de modo de copia de seguridad, sólo tendrá que abrir la base de datos y proceder a realizar de nuevo la copia de seguridad.

Problemas con los archivos redo log

RECUPERACIÓN DE REDO LOG INACTIVOS

Para determinar el estado de los grupos y miembros de redo log online y verificar si es necesaria la recuperación de alguno de ellos, disponemos de la vista V$LOG.

```
SQL> SELECT * FROM V$LOG;

GROUP#   THREAD# SEQUENCE#    BYTES   MEMBERS ARC STATUS   FIRST_CHANGE# FIRST_TI

---------- ---------- --------------    --------- ---------- ----- ---------- --------------------- ----------
```

1	1	205	1048576	1	YES	INACTIVE	175533 09/09/02
2	1	206	1048576	1	YES	INACTIVE	175534 09/09/02
3	1	207	1048576	1	NO	CURRENT	195538 09/09/02

Esta vista nos proporciona información de cual es el grupo de redo log online actual, si el grupo ha sido o no archivado, números de secuencia...

La columna FIRST_CHANGE mostrará un 0 si hay algún problema en el grupo.

Si detectamos algún error en un grupo no actual habrá que recuperar el mismo realizando los siguientes pasos:

1.- Borrar el grupo de redo log online con el comando ALTER DATABASE DROP LOGFILE GROUP n;

Este comando dará un error si la base de datos sólo cuenta con dos grupos de redo log online.

2.- Crear un nuevo grupo de redo log online: ALTER DATABASE ADD LOGFILE GROUP n '...' SIZE n;

3.- Si no se pudo borrar en el paso anterior el grupo erróneo, borrarlo ahora.

Recuperación incompleta

Una recuperación incompleta es aquella que se realiza hasta un momento anterior en el tiempo al momento del fallo. Este tipo de recuperaciones sólo podrá realizarse si la base de datos funciona en modo ARCHIVELOG.

En las recuperaciones incompletas siempre se producirá la pérdida de datos, los correspondientes a transacciones realizadas después del tiempo de recuperación. Será necesario que los usuarios vuelvan a introducir esos datos manualmente para su recuperación.

Para realizar una recuperación incompleta deben cumplirse los siguientes requisitos:

- Tener una copia de seguridad válida de todos los archivos de la base de datos.
- Tener los log archivados desde la copia de seguridad hasta el momento de la recuperación.

Alguna de las causas que pueden exigir una recuperación incompleta son las siguientes:

- Cuando falla una recuperación completa por pérdidas de redo log archivados.
- Errores de usuario como la eliminación de tablas no deseadas.
- Pérdidas de redo log online que no han sido archivados junto con un archivo de datos.

Existen diferentes tipos de recuperación incompleta:

- Recuperación basada en el tiempo.
- Recuperación basada en cancelación.
- Recuperación basada en modificación.

COMANDO DE RECUPERACIÓN INCOMPLETA

El comando que debemos utilizar para realizar cualquiera de los tipos de recuperación incompleta es el comando RECOVER. Su sintaxis es la que se muestra a continuación:

RECOVER [AUTOMATIC] DATABASE opciones

Donde opciones podrá ser:

until time 'YYYY-MM-DD:HH:MI:SS' – Para recuperaciones basadas en el tiempo.

until cancel – Para recuperaciones basadas en cancelaciones.

until change <n> – Para recuperaciones basadas en modificaciones.

INSTRUCCIONES GENERALES PARA LA RECUPERACIÓN INCOMPLETA

Los pasos de forma general para realizar cualquier recuperación incompleta son:

1.- Realizar una copia de seguridad de la base de datos completa con la base de datos cerrada incluyendo los redo log y los archivos de control antes de iniciar la recuperación.

2.- Restaurar todos los archivos de la base de datos de la copia de seguridad. Con esto estamos haciendo retroceder a la base de datos hasta el momento en que se realizó la copia de seguridad.

3.- Montar la base de datos y realizar la recuperación oportuna.

4.- Abrir la base de datos utilizando la opción RESETLOGS.

5.- Es aconsejable realizar una copia de seguridad con la base de datos cerrada después de realizar la recuperación.

Es importante que una vez realizada la recuperación, realicemos una copia de seguridad de los redo log archivados y después los eliminemos de su ubicación por defecto. Esto es para evitar la mezcla de archivos redo log archivados de diferentes vidas de la base de datos ya que con la operación RESETLOGS de la recuperación se resetean los números de secuencia de los logs y podrían llegar a sobrescribirse en un momento posterior.

Podemos obtener información de la recuperación en el fichero de alerta.

RECUPERACIÓN INCOMPLETA BASADA EN EL TIEMPO

Supongamos que un usuario ha eliminado una tabla de la base de datos necesaria.

Habrá que recuperar la base de datos hasta un momento anterior a la eliminación de la tabla siguiendo los siguientes pasos:

1.- Cerrar la base de datos con cualquier opción excepto ABORT.

2.- Montar la base de datos.

3.- Restaure desde el sistema operativo los archivos de datos de la última copia de seguridad.

4.- Recuperar la base de datos con el siguiente comando:

```
RECOVER DATABASE UNTIL TIME 'YYYY-MM-DD:HH:MI:SS';
```

Para conocer el instante en el que debemos recuperar la base de datos, podemos utilizar la herramienta LogMiner o la vista V$LOG_HISTORY.

5.- Abrir la base de datos con la opción RESETLOGS para sincronizar los archivos de datos con los archivos de control y redo log.

6.- Ahora puede comprobar si el error se ha solucionado. En este caso ver que se ha recuperado la tabla eliminada por el usuario.

7.- Si la recuperación ha finalizado correctamente debería realizar una copia de seguridad de la base de datos.

8.- Informar a los usuarios de que introduzcan de nuevo los datos desde la hora en que se realiza la recuperación.

RECUPERACIÓN INCOMPLETA BASADA EN CANCELACIÓN

Supongamos que detectamos que un archivo redo log no fue archivado, por lo tanto no podremos realizar una recuperación completa de la base de datos. Habrá que realizar una recuperación incompleta hasta el redo log archivado anterior al archivo perdido.

Para ver que archivo redo se ha perdido, podemos consultar la vista V$LOG_HISTORY.

Pasos:

1.- Cerrar la base de datos con una opción diferente a ABORT.

2.- Montar la base de datos.

3.- Restaurar desde el sistema operativo los archivos de datos de la copia de seguridad.

4.- Recuperar la base de datos hasta la secuencia perdida con el comando:

```
RECOVER DATABASE UNTIL CANCEL;
```

Este comando irá pidiendo la confirmación para aplicar los archivos de redo log archivados. Cuando lleguemos a la secuencia perdida deberemos indicar la opción CANCEL.

5.- Abrir la base de datos con la opción RESETLOGS.

6.- Si la recuperación ha finalizado correctamente debería realizar una copia de seguridad de la base de datos.

7.- Informar a los usuarios de que introduzcan de nuevo los datos desde la hora en que se realiza la recuperación.

RECUPERACIÓN INCOMPLETA BASADA EN MODIFICACIÓN

Esta opción es otra posibilidad de realizar una recuperación incompleta cuando detectamos que un archivo redo log no fue archivado, por lo tanto no podremos realizar una recuperación completa de la base de datos. Habrá que realizar una recuperación incompleta hasta la última modificación del redo log archivado anterior al perdido.

Una vez conocemos el número de secuencia del redo log archivado perdido, para averiguar el número de cambio hasta donde vamos a recuperar debemos lanzar la siguiente sentencia:

```
SELECT FIRST_CHANGE#-1 FROM V$LOG_HISTORY

WHERE SEQUENCE#=<n_log_perdido>;
```

Este número de cambio es el número correspondiente al último cambio incluido en el redo log archivado con número de secuencia anterior al perdido.

Pasos:

1.- Cerrar la base de datos con una opción diferente a ABORT.

2.- Montar la base de datos.

3.- Restaurar desde el sistema operativo los archivos de datos de la copia de seguridad.

4.- Recuperar la base de datos hasta la secuencia perdida con el comando:

```
RECOVER DATABASE UNTIL CHANGE <n>;
```

5.- Abrir la base de datos con la opción RESETLOGS.

6.- Si la recuperación ha finalizado correctamente debería realizar una copia de seguridad de la base de datos.

7.- Informar a los usuarios de que introduzcan de nuevo los datos desde la hora en que se realiza la recuperación.

Tema 5
EXPORT/IMPORT
SQL*LOADER

Introducción

Las utilidades EXPORT e IMPORT permiten realizar copias y recuperaciones lógicas fiables de los datos Oracle. La utilidad export hace una copia de los datos y de las estructuras de los mismos en un archivo del sistema operativo. La utilidad import lee los archivos creados con la utilidad export y coloca los datos y las estructuras en archivos de la base de datos.

Estas utilidades se usan principalmente de las siguientes maneras:

- Como parte de procesos de recuperación y backup lógicos.
- Para transferir datos entre diferentes bases de datos Oracle.
- Para transferir datos de un usuario de un tablespace a otro.
- Para reconstruir una base de datos existente.

Las utilidades export e import se comportan de una manera muy similar en los siguientes casos:

- Ambas pueden ejecutarse interactivamente o pueden leer los parámetros desde un archivo.
- Ambas aceptan parámetros posicionales y nominales.
- Trabajan con copias de datos en sólo lectura y con estructuras Oracle.

La utilidad SQL*Loader tiene como uso principal la carga de datos externos a Oracle, mediante ficheros planos.

Métodos de exportación

Hay diferentes formas para invocar la utilidad EXPORT:

- Exportación interactiva sin parámetros.
- Exportación controlada por parámetros.
- Exportación con el parámetro parfile.

EXPORTACIÓN INTERACTIVA SIN PARÁMETROS

Cuando se ejecuta la utilidad export interactivamente, Oracle le presenta una lista de preguntas. Las respuestas a dichas preguntas afectarán al contenido del archivo de exportación.

```
Símbolo del sistema
C:\>exp

Export: Release 8.1.6.0.0 - Production on Vie Sep 13 13:03:58 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Nombre de usuario: beatriz
Clave:

Conectado a: Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production
Introduzca tama±o del buffer de b·squeda del vector: 4096 10240

Exportar a fichero: EXPDAT.DMP > c:\exp_1.dmp

(2)U(suarios), o (3)T(ablas): (2)U > t

Exportar datos de la tabla (sÝ/no): sÝ > s

Comprimir extensiones (sÝ/no): sÝ >

Exportaci¾n realizada en el juego de caracteres WE8ISO8859P1 y el juego de carac
teres NCHAR WE8ISO8859P15
servidor utiliza el juego de caracteres WE8ISO8859P15 (posible conversi¾n de jue
go de caracteres)

Exportando las tablas especificadas a travús de la Ruta Convencional ...
Tabla(T) o Partici¾n(T:P) a exportar: (RETORNO para salir) > prueba1

. exportando la tabla                          PRUEBA1        548 filas exportadas

Tabla(T) o Partici¾n(T:P) a exportar: (RETORNO para salir) >

La exportaci¾n ha terminado con Úxito y sin avisos.

C:\>_
```

Al lanzar desde el sistema operativo el comando exp, se activa el modo interactivo incluyendo una serie de preguntas a las que deberá responder para realizar la exportación. A continuación se muestra un listado de cada pregunta y su significado:

1.- Nombre de usuario: Debe indicar el nombre del usuario que va a realizar la exportación.

2.- Clave: Introducir la contraseña del usuario.

3.- Introduzca el tamaño del buffer: Deberá indicar el tamaño de memoria que se utilizará como área de trabajo. El valor suele estar comprendido entre 10240 (10k) y 10485760 (10M).

4.- Exportar a fichero: Puede cambiar el nombre por defecto del archivo de exportación.

5.- (1)B(ase de datos completa), (2)U(suarios), o (3)T(ablas): Deberá elegir la opción B, U o T en función del tipo de exportación que vaya a realizar.

6.- Exportar datos de la tabla (si/no): Si elige la opción de exportar tabla, la siguiente pregunta es para decidir si se exportan o no los datos de la tabla.

7.- Comprimir extensiones (si/no): Esta pregunta permite que las órdenes create table del archivo de exportación, comprueben si hay espacio suficiente para almacenar todos los datos de la tabla.

8.- Tabla(T) o Partición(T:P) a exportar: (RETORNO para salir): Indicar en esta pregunta el nombre de la tabla a exportar o bien pulsar intro para terminar.

EXPORTACIÓN CONTROLADA POR PARÁMETROS

Este método de exportación se ejecuta con la orden exp junto a una serie de parámetros que se lanzan desde la línea de comandos. El formato de la orden es el siguiente:

```
exp palabra_clave1=valor1 palabra_clave2=valor2....
```

donde palabra_claven y valorn se sustituirá por un parámetro y valor de la siguiente lista:

Palabra clave	Significado	Valor por defecto
USERID	Nombre de usuario y contraseña del usuario que lanza la utilidad.	
BUFFER	Tamaño del buffer en bytes. El valor suele estar comprendido entre 10240 (10k) y 10485760 (10M).	4096
FILE	Nombre del archivo de exportación.	Expdat.dmp
COMPRESS	Permite que las órdenes create table del archivo de exportación, comprueben si hay espacio suficiente para almacenar todos los datos de la tabla.	Sí.
GRANTS	Escribe órdenes grant en el archivo de exportación.	Sí.
INDEXES	Escribe órdenes para la creación de índices en el archivo de exportación.	Sí.
TRIGGERS	Escribe órdenes para la creación de triggers en el archivo de exportación.	Sí.

ROWS	Exporta los datos de las filas de las tablas.	Sí.
CONSTRAINTS	Escribe órdenes para la creación de integridad en el archivo de exportación.	Sí.
LOG	Genera la e/s por pantalla de export a un fichero del sistema operativo.	
FULL	Escribe las órdenes necesarias en el fichero de exportación para exportar la base de datos completa.	No.
OWNER	Proporciona una lista de usuarios cuyos objetos deben incluirse en el fichero de exportación.	
TABLES	Proporciona una lista de tablas cuya estructura y datos deben escribirse en el archivo de exportación.	
INCTYPE	Especifica el tipo de exportación incremental.	
PARFILE	Especifica el nombre de un archivo que contiene valores para los parámetros de exportación.	
STATISTICS	Escribe órdenes ANALYZE en el archivo de exportación.	Estimate.

Puede obtener ayuda de todos los parámetros desde la línea de comandos lanzando la siguiente orden:

```
exp help=y
```

Ejemplo de exportación controlada por parámetros:

```
C:\>exp USERID=beatriz/beatriz FILE=c:\exp1.dmp TABLES=prueba2 LOG=c:\log2

Export: Release 8.1.6.0.0 - Production on Lun Sep 16 10:10:54 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Conectado a: Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production
Exportaci¾n realizada en el juego de caracteres WE8ISO8859P1 y el juego de carac
teres NCHAR WE8ISO8859P15
servidor utiliza el juego de caracteres WE8ISO8859P15 (posible conversi¾n de jue
go de caracteres)

Exportando las tablas especificadas a travús de la Ruta Convencional ...
. exportando la tabla                         PRUEBA2          1 filas exportadas

La exportaci¾n ha terminado con úxito y sin avisos.

C:\>
```

EXPORTACIÓN CON EL PARÁMETRO PARFILE

Ya hemos visto que existe un parámetro PARFILE que se utiliza para especificar un archivo que contiene los parámetros y sus valores para realizar una exportación.

Para utilizar este método habría que lanzar el siguiente comando exp:

```
exp parfile=nombre_fichero
```

Donde nombre_fichero es el nombre del fichero (junto con extensión) que contiene los parámetros y podría ser algo similar a:

```
USERID=beatriz/beatriz
FILE=c:\exp1.dmp
TABLES=prueba2
LOG=c:\log2
```

Modos de exportación

La utilidad export se puede ejecutar en tres modos diferentes dependiendo de lo que queramos conseguir:

- Modo tabla.
- Modo usuario.
- Modo base de datos completa.

MODO TABLA

Este modo se utiliza por ejemplo cuando queremos crear una copia de una tabla que pertenece a un usuario1 en el área de un usuario2. Se especifican los nombres de las tablas que se van a exportar.

Ejemplo:

```
Símbolo del sistema
C:\>exp USERID=beatriz/beatriz TABLES=(prueba1,prueba2) FILE=c:\exp1.dmp

Export: Release 8.1.6.0.0 - Production on Lun Sep 16 10:29:23 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Conectado a: Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production
Exportaci¾n realizada en el juego de caracteres WE8ISO8859P1 y el juego de carac
teres NCHAR WE8ISO8859P15
servidor utiliza el juego de caracteres WE8ISO8859P15 (posible conversi¾n de jue
go de caracteres)

Exportando las tablas especificadas a travús de la Ruta Convencional ...
. exportando la tabla                         PRUEBA1        548 filas exportadas

. exportando la tabla                         PRUEBA2          1 filas exportadas

La exportaci¾n ha terminado con Úxito y sin avisos.

C:\>
```

El ejemplo realiza una exportación en modo tabla que incluye dos tablas en la operación.

MODO USUARIO

En una exportación en modo usuario, se exportan todos los objetos del esquema de usuario incluyendo vistas, triggers, índices, sinónimos, procedimientos...

Este modo se puede utilizar si queremos transportar los datos de un usuario de un tablespace a otro diferente.

Ejemplo:

```
Símbolo del sistema
C:\>exp USERID=system/manager OWNER=beatriz

Export: Release 8.1.6.0.0 - Production on Lun Sep 16 10:35:23 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Conectado a: Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production
Exportaci¾n realizada en el juego de caracteres WE8ISO8859P1 y el juego
teres NCHAR WE8ISO8859P15
servidor utiliza el juego de caracteres WE8ISO8859P15 (posible conversi¾
go de caracteres)

Exportando los usuarios especificados ...
. exportando acciones y objetos de procedimiento pre-esquema
. exportando nombres de biblioteca de funciones ajenas para el usuario B
. exportando definiciones de tipos de objetos para el usuario BEATRIZ
Exportando los objetos de BEATRIZ  ...
. exportando enlaces con la base de datos
. exportando n·meros de secuencia
. exportando definiciones de agrupamiento (cluster)
. exportando las tablas de BEATRIZ a travÚs de la Ruta Convencional ...
. exportando la tabla                        PRUEBA1        548 filas ex

. exportando la tabla                        PRUEBA2          1 filas ex

. exportando sin¾nimos
. exportando vistas
. exportando procedimientos almacenados
```

En el ejemplo se ha exportado el esquema completo de un determinado usuario.

MODO DE BASE DE DATOS COMPLETA

Cuando se hace una exportación completa de la base de datos, todas las instrucciones de creación de estructuras de la base de datos (archivos de datos, tablespace...) y los datos de usuarios se escriben en un archivo de exportación para cada usuario excepto para el usuario SYS.

Ejemplo:

```
Símbolo del sistema

C:\>exp USERID=system/manager FULL=y

Export: Release 8.1.6.0.0 - Production on Lun Sep 16 10:40:21 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Conectado a: Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production
Exportaci¾n realizada en el juego de caracteres WE8ISO8859P1 y el juego
teres NCHAR WE8ISO8859P15
servidor utiliza el juego de caracteres WE8ISO8859P15 (posible conversi¾
go de caracteres)

Exportando toda la base de datos ...
. exportando definiciones de tablespace
. exportando perfiles
. exportando definiciones de usuario
. exportando roles
. exportando costes de recursos
. exportando definiciones de segmentos de rollback (reconstrucci¾n)
. exportando enlaces con la base de datos
. exportando n·meros de secuencia
. exportando alias de directorios
. exportando namespaces de contexto
. exportando nombres de bibliotecas de funciones ajenas
. exportando definiciones de tipos de objetos
. exportando acciones y objetos de procedimiento del sistema
. exportando acciones y objetos de procedimiento pre-esquema
. exportando definiciones de agrupamiento (cluster)
. exportando las tablas de SYSTEM a travús de la Ruta Convencional ...
. exportando la tabla                    DEF$_AQCALL          0 filas ex
```

Nota: Las exportaciones en modo usuario y base de datos completa suelen realizarse utilizando una cuenta de DBA con privilegios.

Métodos de importación

Los métodos para lanzar la utilidad IMPORT son los mismos que los de la utilidad EXPORT:

- Importación interactiva sin parámetros.
- Importación controlada por parámetros.
- Importación con el parámetro parfile.

IMPORTACIÓN INTERACTIVA SIN PARÁMETROS

Cuando se ejecuta la utilidad IMPORT interactivamente, Oracle le presenta una lista de preguntas que deberá ir respondiendo:

```
Símbolo del sistema
C:\>imp

Import: Release 8.1.6.0.0 - Production on Lun Sep 16 11:05:11 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.

Nombre de usuario: system
Clave:

Conectado a: Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production

Fichero de importaci¾n: EXPDAT.DMP >

Introduzca el tama±o del buffer de inserci¾n (mÝn. 8192) 30720) 10240

Fichero de exportaci¾n creado por EXPORT:V08.01.06 a travús de la ruta co
nal

Aviso: los objetos fueron exportados por BEATRIZ, no por usted

importaci¾n realizada en el juego de caracteres WE8ISO8859P1 y el juego d
teres NCHAR WE8ISO8859P15
el servidor de importaci¾n utiliza el juego de caracteres WE8ISO8859P15 (
 conversi¾n de juego de caracteres)
Listar solamente el contenido del fichero de importaci¾n (sÝ/no): no >

Ignorar error de creaci¾n debido a que el objeto ya existe (sÝ/no): no >

Importar permisos (sÝ/no): sÝ >

Importar los datos de la tabla (sÝ/no): sÝ >

Importar todo el fichero de exportaci¾n (sÝ/no): no > s

. importando objetos de BEATRIZ en SYSTEM
. importando la tabla                    "PRUEBA1"        548 filas imp
```

Al lanzar desde el sistema operativo el comando imp, se activa el modo interactivo incluyendo una serie de preguntas a las que deberá responder para realizar la importación. A continuación se muestra un listado de cada pregunta y su significado:

1.- Nombre de usuario: Debe indicar el nombre del usuario que va a realizar la importación.

2.- Clave: Introducir la contraseña del usuario.

3.- Fichero de importación: Nombre del fichero que debe leer la utilidad import. Por defecto es EXPDAT.DMP. Este fichero se habrá creado con la utilidad EXPORT.

4.- Introduzca el tamaño del buffer: Deberá indicar el tamaño de memoria que se utilizará como área de trabajo. El valor suele estar comprendido entre 10240 (10k) y 10485760 (10M).

5.- Listar solamente el contenido del fichero de importación: Si la respuesta es Sí, se enumerarán las órdenes SQL escritas en el archivo de importación. Si responde No, IMPORT almacenará los datos y sus definiciones en la base de datos.

6.- Ignorar error de creación debido a que el objeto ya existe: Si la respuesta es Sí, Oracle ignora que el objeto ya exista en la base de datos y almacena los datos. Si la respuesta es No, Oracle informa del error y pasa al siguiente objeto.

7.- Importar permisos: Si es afirmativo, se ejecutarán las órdenes GRANT del archivo de importación después de haber importado el objeto.

8.- Importar los datos de la tabla: Si la respuesta es afirmativa Oracle almacenará los datos en las tablas. Si es negativa, sólo lanzará la creación de los objetos.

9.- Importar todo el fichero de exportación: Si la respuesta es afirmativa, la importación comienza inmediatamente y realiza la importación de todos los objetos incluidos en el archivo. Si la respuesta es negativa, Oracle irá preguntando si desea o no importar el objeto.

IMPORTACIÓN CONTROLADA POR PARÁMETROS

Este método de importación se ejecuta con la orden imp junto a una serie de parámetros que se lanzan desde la línea de comandos. El formato de la orden es el siguiente:

```
imp palabra_clave1=valor1 palabra_clave2=valor2....
```

donde palabra_claven y valorn se sustituirá por un parámetro y valor de la siguiente lista:

Palabra clave	Significado	Valor por defecto
USERID	Nombre de usuario y contraseña del usuario que lanza la utilidad.	
BUFFER	Tamaño del buffer en bytes. El valor suele estar comprendido entre 10240 (10k) y 10485760 (10M).	4096
FILE	Nombre del archivo del que se lee.	Expdat.dmp
SHOW	Permite que se realice la importación o simplemente que se muestre el contenido del archivo de exportación.	No.
IGNORE	Si se especifica No, informa de error cuando intentamos crear un objeto que ya existe. Si especificamos Sí, se ignora el error y se	No.

	introducen los datos.	
GRANTS	Ejecuta órdenes grant en el archivo de exportación.	Sí.
INDEXES	Ejecuta órdenes para la creación de índices en el archivo de exportación.	Sí.
ROWS	Importa las filas de las tablas.	Sí.
LOG	Genera la e/s por pantalla de import a un fichero del sistema operativo.	
INDEXFILE	Permite escribir las instrucciones CREATE TABLE o INDEX en un archivo del sistema operativo.	
CHARSET	Especifica el conjunto de caracteres de los datos del archivo de importación.	El valor del parámetro NLS_LANG del INIT.ora
FULL	Controla si se escriben las órdenes del fichero de importación de creación de tablas, archivos de datos...	No.
FROMUSER	Especifica el propietario de los datos que contiene el archivo de importación.	
TOUSER	Especifica el usuario en el que debería realizarse la importación.	
TABLES	Lista de tablas para importar.	
INCTYPE	Especifica el tipo de exportación incremental que se está leyendo.	
COMMIT	Notifica a Oracle si confirma las transacciones o no después de importar las filas de una tabla.	No.
PARFILE	Especifica el nombre de un archivo que contiene valores para los parámetros de importación.	

Puede obtener ayuda de todos los parámetros desde la línea de comandos lanzando la siguiente orden:

```
imp help=y
```

Ejemplo de importación controlada por parámetros:

```
Símbolo del sistema
C:\>imp system/manager fromuser=beatriz touser=system

Import: Release 8.1.6.0.0 - Production on Lun Sep 16 12:47:44 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Conectado a: Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production

Fichero de exportaci¾n creado por EXPORT:V08.01.06 a travús de la ruta co
nal

Aviso: los objetos fueron exportados por BEATRIZ, no por usted

importaci¾n realizada en el juego de caracteres WE8ISO8859P1 y el juego d
teres NCHAR WE8ISO8859P15
el servidor de importaci¾n utiliza el juego de caracteres WE8ISO8859P15 (
 conversi¾n de juego de caracteres)
. importando la tabla                        "PRUEBA1"        548 filas imp

La importaci¾n ha terminado con úxito y sin avisos.

C:\>
```

IMPORTACIÓN CON EL PARÁMETRO PARFILE

También se puede realizar la importación utilizando un archivo de parámetros al igual que la exportación.

Para utilizar este método habría que lanzar el siguiente comando imp:

```
imp parfile=nombre_fichero
```

Donde nombre_fichero es el nombre del fichero (junto con extensión) que contiene los parámetros y podría ser algo similar a:

```
parametros - Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
USERID=beatriz/beatriz
FILE=c:\exp1.dmp
TABLES=prueba2
LOG=c:\log2
```

Al igual que la utilidad export, la importación se puede ejecutar en tres modos diferentes dependiendo de lo que queramos conseguir:

- Modo tabla.
- Modo usuario.
- Modo base de datos completa.

MODO TABLA

Cuando se utiliza la importación en modo tabla, se debe indicar a Oracle los nombres de las tablas que hay que importar.

Ejemplo:

```
Símbolo del sistema
C:\>imp userid=system/manager tables=prueba1

Import: Release 8.1.6.0.0 - Production on Lun Sep 16 12:55:48 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Conectado a: Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production

Fichero de exportaci¾n creado por EXPORT:V08.01.06 a travús de la ruta co
nal

Aviso: los objetos fueron exportados por BEATRIZ, no por usted

importaci¾n realizada en el juego de caracteres WE8ISO8859P1 y el juego d
teres NCHAR WE8ISO8859P15
el servidor de importaci¾n utiliza el juego de caracteres WE8ISO8859P15 (
 conversi¾n de juego de caracteres)
. importando objetos de BEATRIZ en SYSTEM
. importando la tabla                        "PRUEBA1"        548 filas imp

La importaci¾n ha terminado con Úxito y sin avisos.
```

El ejemplo realiza la importación de la tabla prueba1 en el esquema de usuario SYSTEM. Al no especificar el parámetro TOUSER, se toma como esquema destino el especificado USERID.

MODO USUARIO

En una importación en modo usuario, se importan todos los objetos del esquema de usuario incluyendo vistas, triggers, índices, sinónimos, procedimientos...

Este modo se puede utilizar si queremos transportar los datos de un usuario de un tablespace a otro diferente.

Ejemplo:

```
Símbolo del sistema
C:\>imp userid=system/manager fromuser=beatriz touser=usu2

Import: Release 8.1.6.0.0 - Production on Lun Sep 16 13:18:17 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Conectado a: Oracle8i Enterprise Edition Release 8.1.6.0.0 - Production
With the Partitioning option
JServer Release 8.1.6.0.0 - Production

Fichero de exportaci¾n creado por EXPORT:V08.01.06 a travús de la ruta convenc
nal
importaci¾n realizada en el juego de caracteres WE8ISO8859P1 y el juego de car
teres NCHAR WE8ISO8859P15
el servidor de importaci¾n utiliza el juego de caracteres WE8ISO8859P15 (posib
 conversi¾n de juego de caracteres)
. importando objetos de BEATRIZ en USU2
. importando la tabla                        "PRUEBA1"        548 filas importad

. importando la tabla                        "PRUEBA2"          1 filas importad

La importaci¾n ha terminado con úxito y sin avisos.

C:\>
```

En el ejemplo se ha importado el esquema completo de un determinado usuario.

MODO DE BASE DE DATOS COMPLETA

Una importación completa de la base de datos se realiza en dos fases. En la primera fase se ejecutan todas las instrucciones de creación de archivos de datos, tablespaces, segmentos de rollback, etc. Cuando se completa esta fase ya disponemos de la estructura completa de la base de datos. La segunda fase importaría los objetos de los usuarios a los tablespaces correspondientes.

Ejemplo: El siguiente comando realizaría una importación completa de la base de datos.

```
imp userid=system/manager full=y
```

Nota: Las importaciones en modo de base de datos completa suelen realizarse utilizando una cuenta de DBA con privilegios.

CARGA DE DATOS DE FUENTES EXTERNAS A ORACLE (SQL*Loader)

La utilidad SQL*Loader permite cargar los datos en una base ORACLE a partir de archivos externos a la misma.

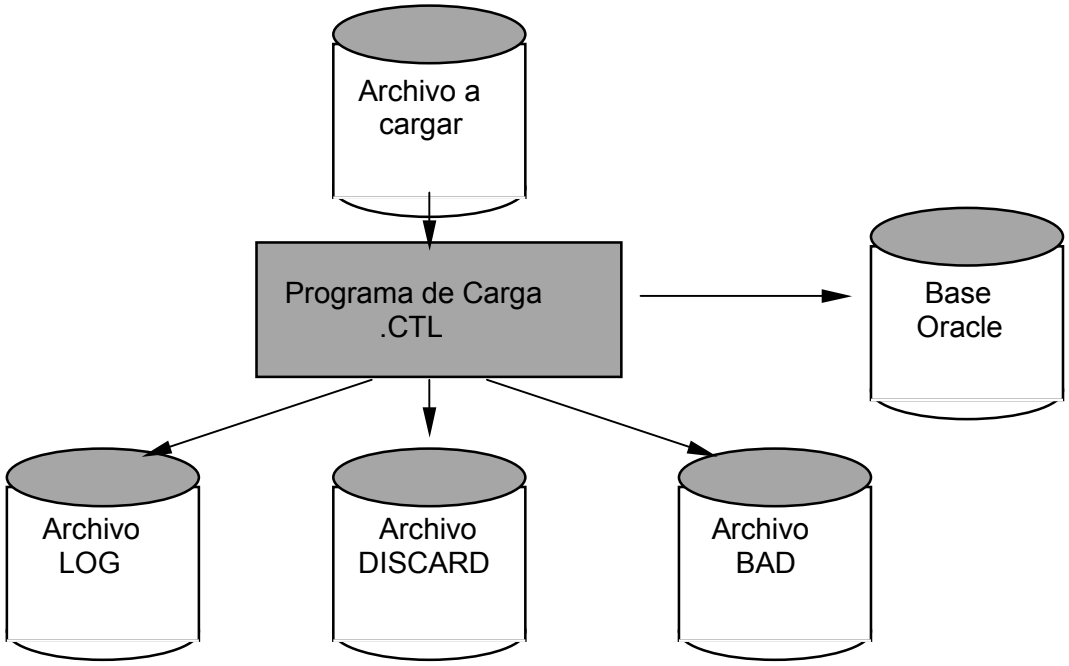
Para cargar los datos en una base hace falta escribir un programa con SQL*Loader, llamado programa de control (<nombre>.ctl), en el que se indica el nombre de los archivos externos, el formato de sus registros, las tablas y las columnas en las que se van a cargar los datos.

Se puede igualmente definir las condiciones que deben satisfacer los registros a cargar.

En un mismo programa de control, es posible cargar desde varios archivos externos hacia una o varias tablas de la base.

La carga genera:

- Una o varias tablas cargadas.
- Un archivo *LOG* que escribe el desarrollo de la operación de carga: número de registros leídos, cargados, rechazados, etc.
- Un archivo *BAD* conteniendo los registros rechazados, por tanto, no cargados por incompatibilidad de tipo con la base.
- Un archivo *DISCARD* que contiene los registros no cargados, porque no verifican las condiciones del archivo de control.



Veamos a continuación un ejemplo de programa de control de carga desde un archivo PEDIDOS.DAT hacia las tres tablas:

PEDIDO(num_ped, date_ped, num_cl)

CLIENTE(num_cl, nom_cl)
LINEA_PED(num_linea, num_ped, num_art, cant)

La estructura de un registro del archivo a cargar es la siguiente:

num_pedido, num_cl, montante_ped, num_cl, nom_cl, num_linea, num_ped, num_art, cant

Ejemplo de programa de carga:

```
LOAD DATA
INFILE PEDIDO.DAT
BADFILE PEDIDO.BAD
DICARDFILE PEDIDO.DIS
APPEND
INTO TABLE PEDIDO
(numped    POSITION(1:5) INTEGER EXTERNAL,
date_ped   POSITION(6:12) CHAR,
num_cl     POSITION(18:22) INTEGER EXTERNAL)

INTO TABLE CLIENTE
(num_cl    POSITION(18:22) INTEGER EXTERNAL,
nom_cl     POSITION(23:52) CHAR,

INTO TABLE LINEA_PED
when num_art != ' '
(num_linea SEQUENCE(MAX),
num_ped    POSITION(1:5) INTEGER EXTERNAL,
num_art    POSITION(53:57) INTEGER EXTERNAL,
cant       POSITION(58:61) INTEGER EXTERNAL)
```

El programa se puede ejecutar por la utilidad SQLLOAD con varias opciones de ejecución.

Ejemplo:

SQLLOAD CONTROL=PGM.CTL, LOG=PEDIDO.LOG, BAD=PEDIDO.BAD, DISCARD=PEDIDO.DIS, USERID=SYSTEM/MANAGER, ERRORS=50

"ERRORS=50" indica que se debe parar la carga cuando se sobrepasen 50 registros rechazados.

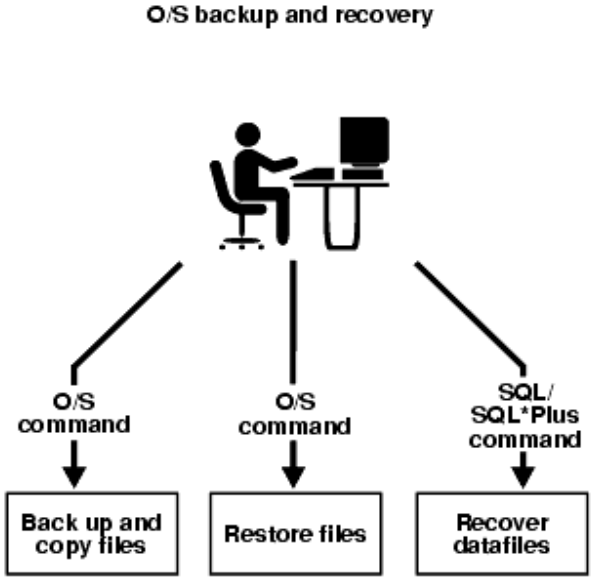
Tema 6

RECOVERY MANAGER

Introducción

RMAN o Recovery Manager es una de las herramienta que proporciona Oracle para realizar o ayudar a realizar las tareas de backup y recuperación de bases de datos a los administradores de bases de datos.

RMAN proporciona una serie de utilidades o funciones que no se tienen cuando se realizan los backups o recuperaciones desde el sistema operativo.



Funciones y mejoras de Recovery Manager

Entre otras funciones, RMAN proporciona las siguientes utilidades:

- Podemos almacenar en archivos de comandos o scripts las operaciones que se ejecutan más a menudo.
- Se pueden realizar copias incrementales y acumulativas.
- Se puede determinar el tiempo y el tamaño de las copias de seguridad.
- Se puede verificar la validez de las copias de seguridad realizadas.
- Se pueden detectar corrupciones en los bloques.

Además, RMAN proporciona las siguientes mejoras:

Permite la paralelización de las operaciones de copia de seguridad, restauración y recuperación.

No genera información redo log adicional durante la realización de las copias de seguridad online.

Conceptos generales de RMAN

- **Procesos**: Los procesos que utiliza RMAN son procesos que se conectan a la base de datos destino para realizar las operaciones de backup y recuperación.

- **Base de datos destino**: Es la base de datos con la que trabaja RMAN. RMAN trabaja principalmente con los archivos de control de las bases de datos destino para obtener información sobre la estructuras de las bases de datos.
- **Canal**: RMAN requiere un enlace con la base de datos destino para poder realizar sus funciones. Este enlace es lo que se conoce como canal.
- **Metadatos**: Son los datos con los que trabaja RMAN para las operaciones realizadas.
- **Catálogo**: Es el lugar donde se almacenan los metadatos. No es obligatorio y en caso de no existir, la información estará en los archivos de control. Es recomendable utilizar el catálogo de RMAN.
- **Base de datos auxiliar**: Es un duplicado de la base de datos destino.
- **Juego de copias de seguridad**: Son los archivos de salida después de realizar una copia de seguridad con la herramienta RMAN. Cada archivo del juego de copia de seguridad se denomina pieza de copia de seguridad.
- **Copias de imagen**: RMAN permite realizar copias de los archivos de una base de datos en otra ubicación. Éstos archivos pueden servir como copias de seguridad. Incluye bloques vacios.

PAQUETES DE RECOVERY MANAGER

RMAN utiliza dos paquetes para realizar sus operaciones:

- DBMS_RCVCAT
- DBMS_RCVMAN
- DBMS_BACKUP_RESTORE

Estos paquetes están disponibles en la base de datos en cualquier momento. No es necesario que la base de datos esté abierta para poder utilizarlos.

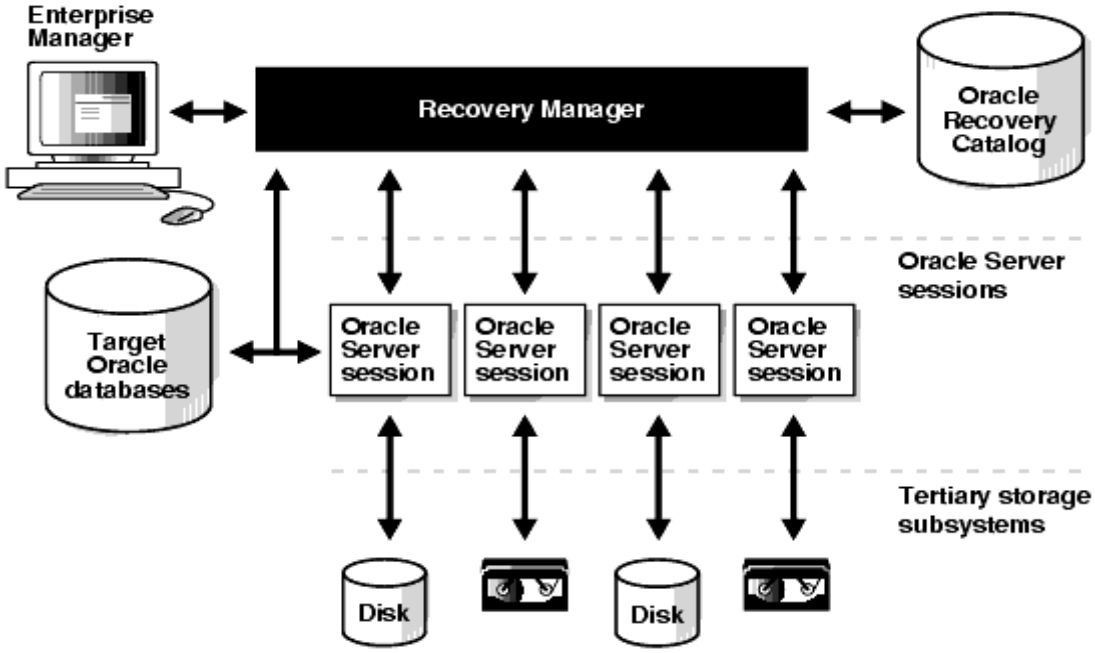
El paquete DBMS_BACKUP_RESTORE se crea con los archivos dbmsbkrs.sql y prvtbkrs.plb. Se utiliza como interfaz entre el sistema operativo y Oracle para crear, restaurar y recuperar copias de seguridad.

CONSIDERACIONES DE CONFIGURACIÓN DE RECOVERY MANAGER

- Recursos del sistema: La mayoría del trabajo realizado por RMAN se realiza mediante procesos de Oracle Server. Se pueden realizar en paralelo para aumentar la capacidad de procesamiento.

- Conjunto de usuarios que realizan operaciones privilegiadas: Hay que determinar de antemano el grupo de usuarios que realizan operaciones privilegiadas.
- Operaciones remotas: Será necesario utilizar un archivo de contraseñas para conectarse a la base de datos destino y realizar operaciones privilegiadas.
- Uso del catálogo de recuperación: El utilizar el catálogo de recuperación permite tener disponibles más funciones automatizadas de copia y recuperación. Ahora bien implicará disponer de mayor capacidad de almacenamiento y más mantenimiento.

Catálogo de recuperación



Ya hemos comentado que los metadatos con los que trabaja RMAN pueden ser almacenados en un catálogo de recuperación. Es más, es aconsejable utilizar dicho catálogo.

Incluso algunas funciones de Recovery Manager sólo estarán disponibles utilizando el catálogo de recuperación.

El mantenimiento del catálogo se realiza automáticamente por RMAN, es por esto por lo que cada vez que conecte con RMAN deberá realizar un proceso de refresco.

El contenido del catálogo de recuperación es el siguiente:

- Juegos de copias de seguridad y piezas de copia de seguridad. Contiene información sobre el nombre y la hora del juego de copias de seguridad.
- Copias del archivo de datos: Son copias físicas de un archivo de datos. Contiene el nombre y la hora de las copias de archivos.
- Redo log archivados y copias de los mismos: Mantiene un registro de los log archivados y de las copias realizadas por RMAN.
- Estructura física de la base de datos de destino: Información similar a la del archivo de control.

Fases de Recovery Manager

RMAN procesa los comandos con los que trabaja en dos fases: compilación y ejecución.

COMPILACIÓN

Durante la fase de compilación, Recovery Manager lleva a cabo dos operaciones:

- La traducción de términos: La mayoría de los comandos RMAN requieren la identificación de objetos sobre los que opera el comando en cuestión. La traducción de términos es la operación en la que RMAN traduce lo que se ha especificado en el comando, en una lista de entidades. Estas entidades pueden ser ficheros de datos, archivos redo log, copias de ficheros...
- Construcción de pasos de trabajo PL/SQL: RMAN lleva a cabo sus tareas generando dinámicamente uno o más programas PL/SQL y ejecutándolos. RMAN contiene una copia del motor PL/SQL y ejecuta los programas PL/SQL internamente durante la fase de ejecución.

EJECUCIÓN

Durante la fase de ejecución, RMAN planifica y ejecuta los programas PL/SQL generados en la fase de compilación. RMAN asigna un programa PL/SQL a cada canal reservado (sesión de servidor). Los canales ejecutan los programas PL/SQL concurrentemente. Por ejemplo, si tienes reservados tres canales entonces RMAN ejecutará tres programas PL/SQL al mismo tiempo.

Tipos de comandos

Recovery Manager utiliza dos tipos de comandos:

- Comandos stand-alone.
- Comandos de trabajo.

Los comandos stand-alone, con excepción de change, crosscheck y delete, van independientes y en cambio los comandos de trabajo se deben ejecutar con el comando run.

COMANDOS STAND-ALONE

Algunos de los comandos stand-alone son los siguientes:

- Catalog.
- Change.
- Create catalog, drop catalog, upgrade catalog.
- Create script, delete script, replace script.
- Crosscheck.
- Delete expired backupset.
- List.
- Report.

Alguno de estos comandos no son estrictamente stand-alone ya que deben ir precedidos por el comando:

```
allocate channel for maintenance
```

COMANDOS DE TRABAJO

Todos los comandos de trabajo deben ir precedidos del comando run. Algunos comandos de trabajo son:

- Allocate channel.
- Backup.
- Copy.
- Duplicate.
- Recover.
- Restore.
- Switch.

Dentro del comando run, los comandos de trabajo se ejecutan secuencialmente. Si algún comando del bloque falla, los comandos siguientes no se ejecutan.

OTROS COMANDOS

La mayoría de los comandos son stand-alone o comandos de trabajo. Si intentamos lanzar un comando de trabajo fuera de un bloque run o si intentamos emitir un comando stand-alone dentro de un bloque run, RMAN generará un mensaje de error.

Sin embargo, hay algunos comandos que pueden funcionar como comandos stand-alone o como comandos de trabajo. Estos son:

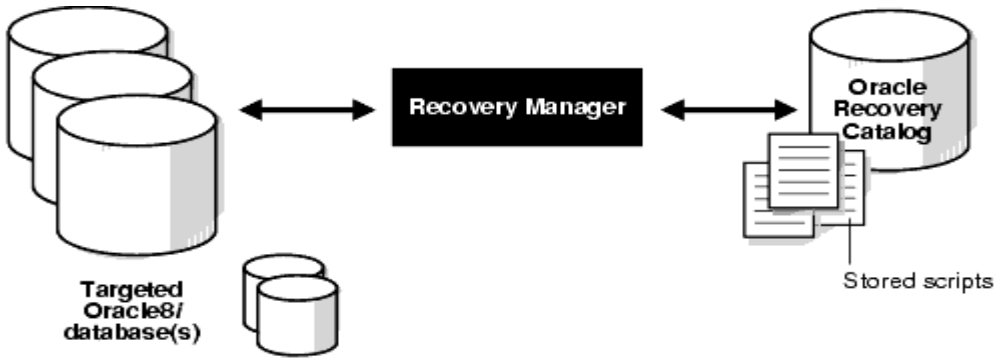
- @
- @@
- Host.
- Send.
- Shutdown.
- Startup.
- Sql.

Modos de operación de Recovery Manager

Recovery Manager es una utilidad que se lanza desde la línea de comando y tiene dos modos de trabajo:

- Interactivo: Para trabajar en el modo interactivo después de iniciar la herramienta RMAN se lanzarán los comandos interactivamente.
- Batch: Puede escribir los comandos RMAN en un archivo y después especificar su nombre en la línea de comandos para ejecutarlo. El resultado de la ejecución podrá ser también almacenado en un fichero log de salida si se especifica. RMAN analizará el archivo de comandos al completo antes de compilar o ejecutar los comandos.

Dispone además de la posibilidad de utilizar scripts que se almacenan en el catálogo de Recovery Manager.



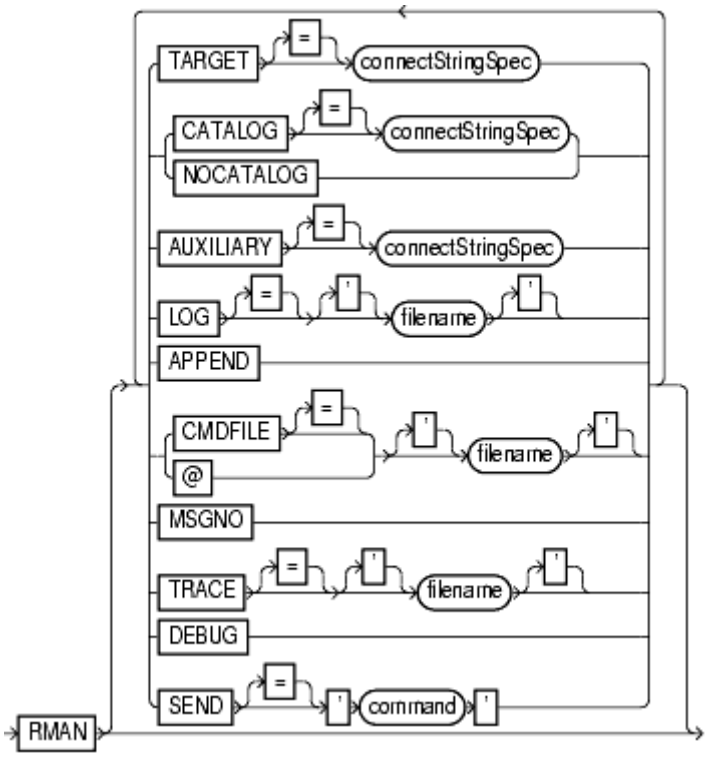
Los scripts almacenados tienen las siguientes características:

- Se crean con el comando de trabajo create script.
- Se almacenan en el catálogo de recuperación.
- Permite planificar y ejecutar una serie de comandos para realizar copias de seguridad, restauración y recuperación.
- Minimiza errores de sintaxis.
- Se ejecutan mediante el comando run.
- Se pueden sustituir con el comando replace script.

Para determinar los scripts almacenados en el catálogo de recuperación utilizaremos las vistas RC_STORED_SCRIPT y RC_STORED_SCRIPT_LINE.

Conexión a Recovery Manager

Para realizar la conexión a RMAN debemos lanzar desde la línea de comandos del sistema operativo el comando RMAN seguido de las opciones requeridas:



donde las opciones disponibles son:

- **Target**: especifica una cadena de conexión a la base de datos. Ejemplo:

target *sys/change_on_install@inst1*.

- **Catalog**: especifica una cadena de conexión a la base de datos que contiene el catálogo de recuperación. Ejemplo:

  catalog *rman/rman@inst2.*

- **Nocatalog**: Indica que vamos a usar RMAN sin catálogo de recuperación.
- **Auxiliary**: especifica una cadena de conexión a una base de datos auxiliar. Ejemplo:

  auxiliary *sys/change_on_install@dupdb*.

- **Log filename**: especifica el archivo donde se almacenará la salida de RMAN.
- **Append**: Provoca que una nueva salida de RMAN se añada al fichero de salida. Si no se especifica esta opciópn, el fichero log se sobrescribirá.
- **Cmdfile filename**: ejecuta un archivo de comandos que contiene una lista de comandos RMAN definidos por el usuario. El contenido del archivo de comandos debe ser idéntico a los comandos introducidos en el prompt de RMAN.
- **@filename**: es equivalente a la opción cmdfile.
- **Msgno**: Visualiza números de mensajes a la salida del comando list.
- **Trace filename**: Especifica el nombre del fichero en el que almacenará información de depuración. Deberás especificar esta opción junto con la opción debug.
- **Debug**: Activa la característica de depuración.

Ejemplos de comandos

Ejemplo1: Asignación de canal. Antes de lanzar comandos de backup y recuperación debe realizar la asignación de canal. Un canal establece una conexión entre RMAN y la base de datos destino o auxiliar. Esto se lleva a cabo iniciando una sesión de servidor en la instancia. Cada canal se corresponde normalmente con un dispositivo de salida.

```
run {
    allocate channel dev1 type disk;
}
```

Los dispositivos de salida disponibles se pueden consultar en la vista V$BACKUP_DEVICE.

Ejemplo2: El comando stand-alone REPORT ayuda a listar la información del catálogo de recuperación.

Report schema; – Lista la estructura de la base de datos.

Report obsolete; – Lista las copias de seguridad que se pueden eliminar porque están obsoletas.

Report need backup incremental n; – Identifica los archivos de datos que necesitan una copia de seguridad en función de 'n'. 'n' indica el número de copias de seguridad incrementales que se deberían restaurar.

Report need backup days n; – Identifica los archivos de datos que necesitan una copia de seguridad en función de 'n'. 'n' indica el número de días máximos desde la última copia de seguridad completa o incremental.

Report need backup redundancy n; – Identifica los archivos de datos que necesitan una copia de seguridad en función de 'n'. 'n' indica el número de copias mínimo de redundancia.

Ejemplo3: El comando stand-alone LIST genera informes de la siguiente información:

List copy of tablespace "SYSTEM"; – Muestra una lista de las copias de seguridad de cualquier fichero de datos que sea miembro del tablespace indicado.

List backup of datafile ".../archivo_datos"; – Muestra una lista de las copias de seguridad que contengan el fichero de datos indicado.

List incarnation of database PRUEBA; – Muestra una lista de las encarnaciones de la base de datos indicada.

Ejemplo4: El comando RUN se utiliza para lanzar todos los comandos de trabajo.

Run { execute script nombre; } – Ejecuta el script almacenado con el nombre especificado.

Run { host "comando s.o."; } – Ejecuta un comando del sistema operativo.

Run { sql "sentencia sql"; } – Ejecuta una sentencia SQL indicada.

Tema 7
CATÁLOGO DE RMAN

Introducción

Recovery Manager permite trabajar con o sin catálogo de recuperación. Se recomienda el uso del catálogo de recuperación. Cuando trabajamos con RMAN con catálogo de recuperación disponemos de una mayor variedad de funciones para realizar las operaciones de backup y recuperación.

RMAN realiza el mantenimiento del catálogo a través de las siguientes operaciones:

- Creación del catálogo.
- Registro del catálogo con la base de datos destino.
- Resincronización con el archivo de control de la base de datos destino.
- Volver una base de datos a una encarnación anterior.
- Modificación de la información de los archivos de backup y restauración.
- Realización de una operación de backup o recuperación (se verán en los temas siguientes).

CONSIDERACIONES SOBRE EL CATÁLOGO DE RECUPERACIÓN

En cuanto al almacenamiento debemos tener en cuenta los siguientes aspectos:

- No crear el catálogo de recuperación en la base de datos destino.
- Si está en el mismo sistema de la base de datos destino, almacénelo en una base de datos diferente y en discos distintos.
- Si hay más de una base de datos destino, lo mejor sería crear una base de datos independiente sólo para el catálogo.
- Debería incluir en la estrategia de copias de seguridad el catálogo de recuperación.
-

En cuanto a las funciones debemos tener en cuanta los siguientes aspectos:

- Utilizar archivos de comandos almacenados.
- Mantener un registro adecuado de las operaciones de backup y recuperación realizadas.

El tamaño del catálogo de recuperación dependerá de los siguientes factores:

- Número de bases de datos destino.
- Número y tamaño de los archivos de comandos almacenados en el catálogo.
- Número de archive log.

Creación del catálogo de recuperación

Para crear el catálogo de recuperación debe seguir los pasos que se proponen a continuación:

1.- Conectarse a la base de datos del catálogo y crear un tablespace para el mismo.

```
SQL > create tablespace nombre_ts datafile 'espec_fichero'
    size n
    default_storage_clause;
```

2.- Crear un usuario de Recovery Manager.

```
SQL > create user nombre_us identified by clave
    default tablespace nombre_ts
        temporary tablespace nombre_tst
        quota unlimited on nombre_ts;
```

3.- Otorgar los permisos al usuario para que pueda mantener el catálogo de recuperación.

```
SQL > grant recovery_catalog_owner to nombre_us;
```

4.- Asegúrese de que los segmentos de rollback y el tablespace tienen suficiente espacio.

5.- Ejecute el comando RMAN desde el intérprete de comandos. Conectarse para crear el catálogo de recuperación:

```
c:\> rman catalog nombre_us/clave@bd
rman> create catalog tablespace nombre_ts;
rman> exit;
```

6.- Si se detecta algún error, deberá borrar todos los objetos creados y volverlos a crear desde un principio.

Registrar bases de datos destino

Para registrar las bases de datos destino debe seguir los siguientes pasos:

1.- Conectarse a la base de destino:

```
c:\> rman target sys/oracle@db1
```

2.- Ya dentro de RMAN, conectarse al catálogo de recuperación:

```
rman > connect catalog nombre_us/clave@bd
```

3.- Registrar la base de datos:

```
rman > register database;
```

4.- Si la base de datos destino tenía ya realizadas copias de seguridad antes de empezar a utilizar RMAN, debería registrar estas copias de seguridad en el catálogo de recuperación usando el comando CATALOG.

Mantenimiento del catálogo de recuperación

Una vez creado el catálogo, se pueden realizar diferentes operaciones sobre el mismo. La actualización del catálogo puede hacerse de forma automática o manual. La forma automática se realiza con el comando RESYNC y obtiene información directamente del archivo de control de la base de datos destino. Los comandos CHANGE y CATALOG son los comandos que permiten la actualización manual del catálogo de recuperación y por lo tanto su mantenimiento.

COMANDO RESYNC

El comando RESYNC de RMAN se utiliza para sincronizar el catálogo de recuperación con la base de datos destino. Esta resincronización del catálogo de recuperación se realiza de forma automática si tenemos el catálogo disponible (abierto) y emitimos comandos como backup, copy, duplicate, list, recover, report, restore y switch.

Sin embargo, si al lanzar estos comandos el catálogo no estuviera disponible, deberíamos abrir la base de datos del catálogo en algún momento después y realizar una resincronización manualmente con el comando RESYNC:

```
rman > resync catalog;
```

Cuando se emite un comando RESYNC se actualiza la siguiente información:

- Registros de cambios de log: Se crean cuando se produce un cambio de log. RMAN utiliza esta información para saber qué archive logs puede encontrar.
- Registros de copias de logs archivados.
- Registros de historial de copias de seguridad: Asociados con juegos de copias de seguridad, piezas de copias de seguridad, miembros del juego de copias de seguridad y copias de archivos.

COMANDO CATALOG

El comando CATALOG de RMAN se utiliza para agregar información al catálogo de recuperación sobre copias de archivos de datos, sobre logs archivados o sobre copias del archivo de control cuando:

- Han sido creados antes de empezar a trabajar con RMAN.
- Han sido creados en el sistema operativo en vez de con RMAN.
- Los archivos deben tener el mismo número de encarnación que la base de datos.

Ejemplo: Imaginemos que ya tenemos realizada una copia de un archivo de datos en el sistema operativo y queremos añadir esta información al catálogo:

1.- Nos conectamos a la herramienta con la base de datos destino y el catálogo:

```
c:\> rman target system/manager@prueba catalog rman/rman@db
```

2.- Lanzamos el comando catalog desde el prompt de la herramienta:

```
rman > catalog datafilecopy 'espec_fichero';
```

COMANDO CHANGE

El comando CHANGE permite realizar diferentes operaciones de mantenimiento del catálogo de recuperación:

- Marcar como disponible o no una pieza de copia de seguridad o redo log archivado.
- Eliminar del sistema operativo una pieza de copia de seguridad o un redo log archivado actualizando el registro del catálogo de recuperación.
- Verificar si están disponibles las piezas de copia de seguridad o redo logs archivados.

OPCIONES UNAVAILABLE/AVAILABLE

Con la opción unavailable podemos marcar archivos como no disponibles. Un archivo marcado como no disponible no se utilizará en una restauración. La opción available por el contrario se utiliza para marcar archivos como válidos y por lo tanto serán utilizados en restauraciones posteriores.

```
rman > change datafilecopy '/oracle/backup/cf_c.f'
  unavailable;
rman > change datafilecopy '/oracle/backup/cf_c.f'
  available;
```

Con el comando LIST COPY podemos comprobar el estado de las copias de los archivos de datos para confirmar si están o no disponibles.

```
rman > list copy;
```

OPCIÓN DELETE

La opción DELETE elimina un archivo físico de un disco y catálogo. Necesita un canal para realizar esta operación. Es necesario establecer un canal antes de utilizar esta opción del comando change.

```
rman > allocate channel for delete type disk;
rman > change datafilecopy 'espec_fichero' delete;
rman > release channel;
```

El comando release channel es para liberar el canal asignado.

La opción delete también puede utilizarse para eliminar un registro del catálogo de recuperación de una copia de seguridad de un archivo sin tocar la copia física.

```
rman > allocate channel for maintenance type disk;
rman > change datafilecopy 'espec_fichero' uncatalog;
rman > release channel;
```

El comando DELETE EXPIRED se utiliza para eliminar registros de backups en el catálogo de recuperación que están vencidos.

```
rman > allocate channel for delete type disk;
rman > delete expired backup;
rman > release channel;
```

Se pueden eliminar también registros del catálogo de una encarnación anterior de una base de datos:

```
rman > allocate channel for maintenance type disk;
```

rman > change backupset n1, n2, n3 delete;

rman > release channel;

rman > exit;

c:\ sqlplus rman/rman@db

Obtener valores de DBINC_KEY de la encarnación de la vista RC_DATABASE_INCARNATION.

Borrar registros de la tabla DBINC que coincidan con DBINC_KEY.

Información sobre el catálogo de recuperación

Principalmente se utilizan dos comandos de RMAN para analizar y obtener información del catálogo de recuperación: LIST y REPORT.

El comando LIST se utiliza para visualizar información del catálogo de recuperación.

El comando REPORT se utiliza para analizar diferentes aspectos de las copias de operaciones de copia, restauración y recuperación.

Otra fuente importante de información son las vistas que se crearon al realizar la creación del catálogo de recuperación. Puede acceder a ellas desde SQL*Plus.

Ejemplos del comando LIST:

list backup of tablespace nombre_ts completed before 'fecha'; (Lista de copias de archivos del tablespace indicado antes de la fecha)

list backup of database device type disk; (Lista de todos los juegos de copia de seguridad)

list incarnation of database; (Lista de encarnaciones registradas)

Ejemplos del comando REPORT:

Report need backup days 3 tablespace nombre_ts; (Informe de los archivos del tablespace indicado de los que no se ha realizado copia desde hace 3 ó mas dias)

Report need backup redundancy 3 datafile 1,2; (Informa qué archivos entre el 1 y el 2 necesitan copias de seguridad porque no existen tres o más copias)

Report unrecoverable database device type 'sb_tape'; (Informe de todos los archivos de cinta que necesitan una nueva copia de seguridad porque existen cambios no registrados desde la última copia de seguridad)

Report obsolete redundancy 2 device type disk; (Informa sobre las copias de seguridad y copias en disco que están obsoletas porque ya existen dos copias más recientes)

Comando RESET DATABASE

Cuando una base de datos destino se abre con la opción RESETLOGS, antes de volver a utilizar RMAN con dicha base de datos, hay que indicarle que la base de datos ha sido reseteada a una nueva encarnación.

El comando de RMAN, RESET DATABASE hace que se cree un nuevo registro de encarnación en el catálogo de recuperación y éste será el identificador de la encarnación actual. Por lo tanto, todas las operaciones de backup y archivado de redo logs quedarán asociadas a la nueva encarnación.

Copias de seguridad y recuperación del catálogo de recuperación

Ya comentamos en un apartado anterior la importancia de incluir en la estrategia de seguridad el catálogo de recuperación.

Puede utilizar cualquier método de copia de seguridad hasta ahora vistos:

- Puede exportar la base de datos que contiene el catálogo si no es muy grande.
- Puede elegir si por el contrario es una base de datos grande, exportar sólo el esquema del catálogo.
- Si el catálogo está almacenado en un tablespace independiente (opción recomendada) y la base de datos está en modo ARCHIVELOG, podrá hacer copias de seguridad online del tablespace.
- Puede utilizar la herramienta RMAN para hacer una copia de la base de datos completa que contiene el catálogo de recuperación.

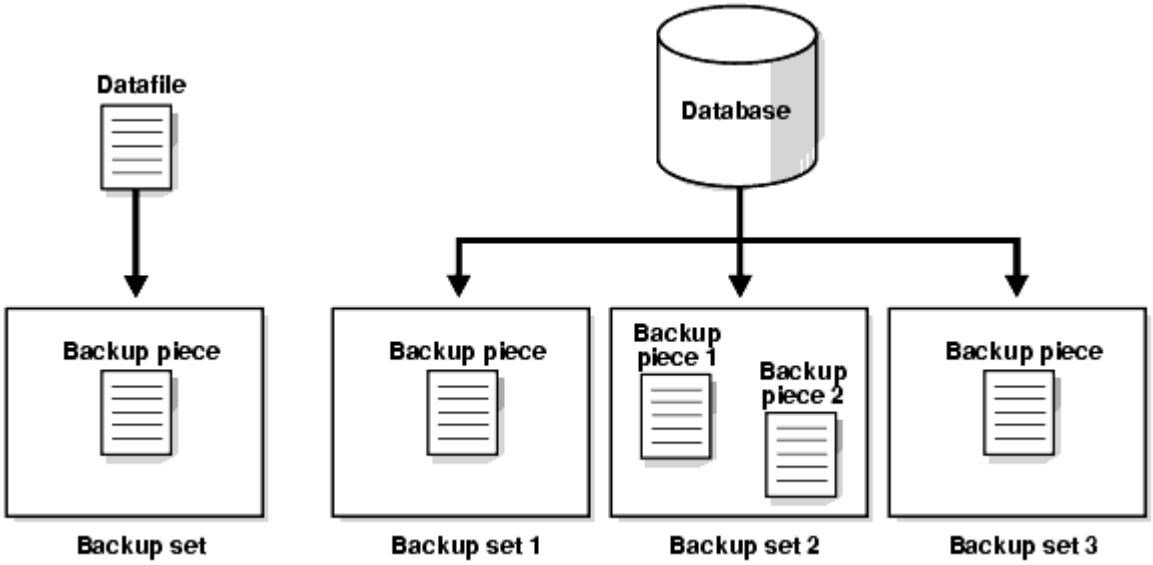
En cuanto a la recuperación del catálogo de recuperación en caso de fallo, dependerá de la estrategia de seguridad empleada y del tipo de fallo producido. Lo que es importante es realizar una sincronización del catálogo con el archivo de control de las bases de datos destino después de recuperarlo y tal vez necesite eliminar registros no deseados en el catálogo de recuperación.

Tema 8

BACKUP CON RMAN

Introducción

En este tema vamos a tratar todos los aspectos relacionados con la realización de backups desde la herramienta Recovery Manager. Veremos los distintos tipos de backups soportados por la herramienta así como la realización de los mismos.



En un primer paso diremos que los backups pueden clasificarse en:

- Backup completo (full) o backup incremental.
- Con base de datos abierta o cerrada.
- Backup consistente o inconsistente.

Un **full backup** es un backup de todos los bloques de datos usados en los archivos de datos. Es diferente a **un backup de la base de datos entera**, el cual incluye todos los archivos de datos y el archivo de control de la base de datos destino. Cuando se realiza un proceso de copia de seguridad completa (full backup), un proceso Oracle Server lee el archivo entero y copia todos los bloques de datos saltándose los bloques del archivo que nunca se han utilizado.

Un **backup incremental** es un backup de archivos de datos que sólo incluyen los bloques que hayan cambiado desde un backup incremental previo.

Un **backup con base de datos abierta** es un backup de cualquier parte de la base de datos destino que se realiza con la base de datos abierta. Cuando

realice un backup online de un tablespace no utilice el comando ALTER TABLESPACE BEGÍN BACKUP, RMAN utiliza un mecanismo diferente que garantiza la consistencia.

Un bac**kup con base de datos cerrada** es aquel que se realiza de cualquier parte de la base destino cuando ésta está en estado MOUNT y no OPEN.

Un **backup consistente** es aquel que se realiza con la base de datos montada pero no abierta. Además la base de datos se cerró en modo NORMAL, TRANSACTIONAL o IMMEDIATE. Un backup consistente puede ser restaurado sin recuperación.

Un **backup inconsistente** es aquel que se realiza de cualquier parte de la base de datos destino cuando ésta está abierta, tuvo un error antes de montarla o se hizo un shutdown con la opción ABORT antes de montarla. Un backup inconsistente necesitará recuperación para poder tener una base de datos consistente.

Copia de seguridad de base de datos entera

Ya hemos comentado anteriormente que una copia de seguridad de base de datos entera consiste en una copia de seguridad de todos los archivos de datos y del archivo de control. Pueden realizarse con la base de datos abierta o cerrada y es el método que más se utiliza en copias de seguridad.

Cuando realizamos una copia de seguridad de base de datos entera cuando la base de datos está cerrada con las opciones NORMAL o IMMEDIATE se conoce como copia de seguridad consistente. Todas las cabeceras de los archivos de datos están sincronizadas con el archivo de control y cuando se restauran completamente no es necesaria la recuperación. Esta es la única opción para restaurar y recuperar una base de datos que se utiliza en modo NOARCHIVELOG, copia entera y consistente.

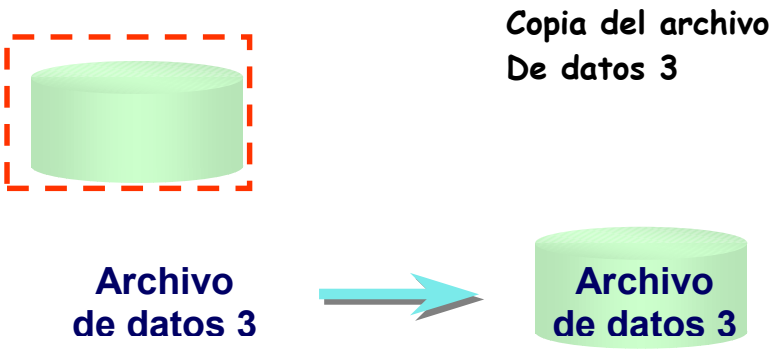
Cuando la base de datos está abierta, es probable que las cabeceras de los archivos de datos no sean consistentes con el archivo de control. También ocurre cuando la base de datos se cerró con modo ABORT. Estas copias de seguridad inconsistentes necesitarán recuperación para dejar a la base de datos en estado consistente. Además este tipo de copias de seguridad sólo pueden realizarse con bases de datos que funcionen en modo ARCHIVELOG.

Tipos de copias de seguridad

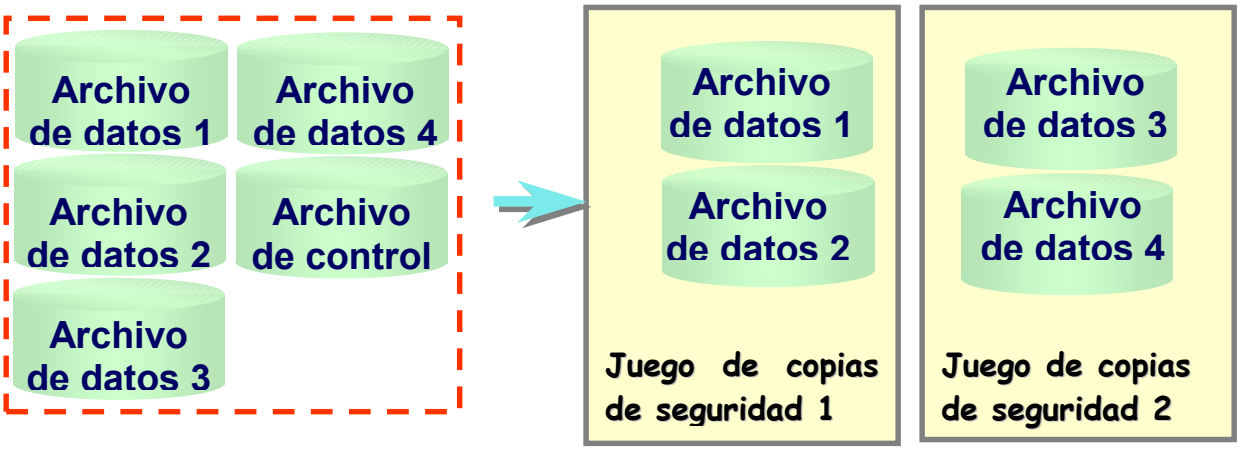
Se distinguen dos tipos principales de copia de seguridad en RMAN:

- Copias de imagen: Copias de un archivo de datos o un log archivado. La copia de imagen del archivo incluye todos los bloques del archivo incluso

los que no han sido utilizados. Sólo puede incluir un archivo y una operación de copia no se puede multiplexar.



- Juegos de copia de seguridad: Pueden incluir uno o más archivos de datos o log archivados.

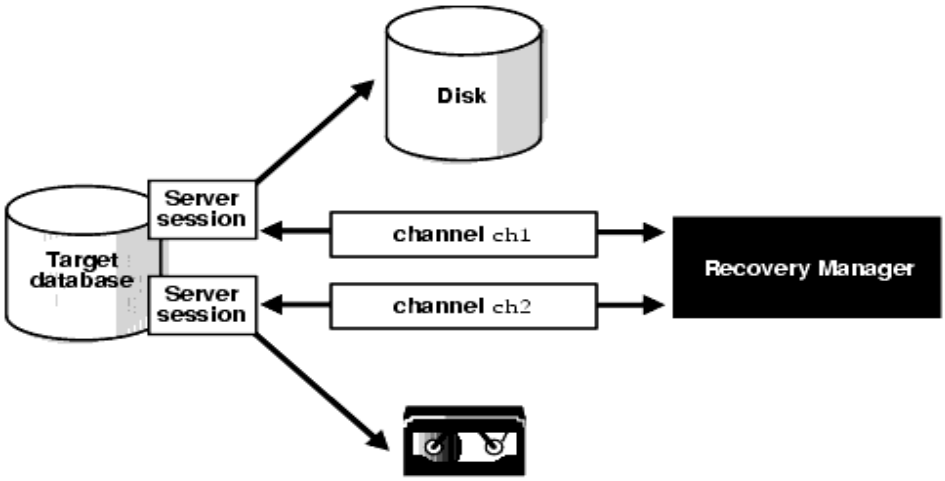


La salida de un juego de copia de seguridad podrá ser uno o más archivos y podrá realizarse de dos maneras diferentes:

- Copia de seguridad completa: Se pueden incluir uno o más archivos. Se incluyen todos los bloques que contengan datos de los archivos especificados para la copia de seguridad.
- Copia de seguridad incremental: Copia de archivos de datos que incluye únicamente los bloques que han cambiado desde que se realizó la última copia de seguridad incremental. Requieren un nivel base (o nivel incremental cero), que incluirán todos los bloques que contengan datos de los archivos especificados.

Canales

Los canales se utilizan para la comunicación entre Oracle Server y el sistema operativo. Los comandos de Recovery Manager, BACKUP, RESTORE y RECOVER necesitan al menos un canal para poder ejecutarse.



La asignación de canal inicia un proceso de servidor en la base de datos destino.

El número de canales que se asigne será el grado máximo de paralelismo que se puede utilizar en la operación.

Para ver los dispositivos que se soportan puede consultar la vista V$BACKUP_DEVICE, aunque el dispositivo DISK no aparecerá por estar siempre disponible.

Ejemplo de asignación de canal:

```
RMAN> run {
2>    allocate channel d1 type disk
3>    format = '/.../backup/user23.bak';
4>    backup datafile '/.../data/user.dbf';}
```

Podrá imponer límites en las operaciones de BACKUP y COPY en la especificación de parámetros del comando ALLOCATE CHANNEL.

set limit channel nombre_c read rate = numero (Limita el número de buffers que se leen por segundo en cada archivo.

set limit channel nombre_c kbytes = numero (Limita el tamaño del archivo de piezas de seguridad creado por un canal)

set limit channel nombre_c maxopenfiles = numero (Limita el número de archivos abiertos simultáneamente en una copia de seguridad)

Etiquetas

Una etiqueta es un nombre significativo que se puede asignar a un juego de copia de seguridad o a una copia de imagen. El uso de etiquetas proporciona una serie de ventajas como las que se comentan a continuación:

- Proporcionan una referencia útil a una recopilación de copias de archivos o a un juego de copias de seguridad.
- Pueden utilizarse en el comando LIST para encontrar fácilmente copias de seguridad de archivos.
- También se pueden utilizar en los comandos RESTORE y SWITCH.
- La misma etiqueta puede hacer referencia a más de un juego de copias de seguridad o a varias copias de archivo. Recovery Manager utilizará siempre el archivo más actual referenciado.

El comando para la asignación de etiquetas es el comando TAG.

Ejemplo:

```
rman> list copy of datafile 2 tag etiqueta1;
```

El ejemplo lista las copias del archivo de datos 2 con la etiqueta 'etiqueta1'.

Copias de imagen

Para crear una copia de imagen se utiliza el comando COPY de Recovery Manager. Se utiliza una sesión Oracle Server para realizar la operación.

Las características más importantes de una copia de imagen son las siguientes:

- Una copia de imagen sólo se puede escribir en disco, no directamente en cinta. Cuando se trate de un archivo de gran tamaño, la copia puede tardar bastante tiempo pero la recuperación será muy rápida.
- Se pueden utilizar directamente y no es necesario almacenarla de nuevo.
- En una copia de imagen se incluyen todos los bloques de datos sin excluir los que no contengan datos.
- Una copia de imagen puede ser parte de una copia de seguridad completa o incremental de nivel 0 ya que una copia de archivo incluye siempre todos los bloques. Opción LEVEL 0.
- No es posible utilizarla en niveles superiores al cero en copias de seguridad incrementales.

COMANDO COPY

El comando COPY es el comando de RMAN que crea una copia de imagen de una archivo. El archivo de salida se escribirá siempre en disco. En ocasiones es mejor utilizar copias de imagen de archivos ya que no utilizan el formato especial de RMAN y por lo tanto podrá ser utilizado directamente.

Con el comando COPY puede realizar copias de archivos de datos (actuales o copias), logs archivados y del archivo de control (actual o copia).

Durante la operación de copia, el proceso Oracle Server realiza una detección de bloques corruptos. Esta comprobación puede evitarse con la opción NOCHECKSUM.

Otra opción del comando COPY es la opción CHECK LOGICAL que se utiliza para comprobar si después de pasar la comprobación física, existe corrupción lógica en el bloque, por ejemplo en una entrada de índice. Si se detecta corrupción lógica, el bloque se registrará en el log de alertas y en el archivo de rastreo del proceso del servidor. Si el número de bloques corruptos llega a un umbral establecido en la cláusula MAXCORRUPT el proceso de copia finaliza sin rellenar las vistas.

Ejemplo:

```
rman> run {
  2>      allocate channel dev1 type disk;
  3>      copy datafile 'espec_fichero' to 'destino',
  4>       datafilecopy 'espec_copia' to 'destino',
  5>       archivelog 'espec_fichero' to 'destino',
  6>           current controlfile to 'destino',
  7>           controlfilecopy 'espec_copia' to 'destino';
  8>  }
```

PROCESO DE COPIA DE IMAGEN

Al realizar una operación de copia, Recovery Manager utiliza el siguiente proceso:

1.- Un proceso de servidor sólo opera con un archivo a la vez.

2.- No se salta ningún bloque del archivo.

3.- Se comprueban los bloques corruptos.

4.- Se calcula una suma de control para verificar la integridad.

5.- Se escribe una cabecera para el nuevo archivo.

6.- Se registra una entrada de la copia en el archivo de control y el catálogo de recuperación.

PARALELIZACIÓN DEL PROCESO DE COPIA DE IMAGEN

Cuando tenemos varios comandos COPY, por defecto se ejecutarán en serie. Podemos paralelizar la operación asignado varios canales e incluyendo más de un archivo en un único comando COPY.

Ejemplo:

```
rman> run {
2>      allocate channel c1 type disk;
3>     allocate channel c2 type disk;
4>     allocate channel c3 type disk;
5>      copy datafile 1 to 'destino',
6>        datafile 2 to 'destino';
7>     copy datafile 3 to 'destino';
8>  }
```

Se reservan tres canales c1, c2, c3. Con el primer comando COPY se utilizarán los dos primeros, habrá dos procesos servidor activos. Hasta que no finalice el primer comando COPY no empezará el segundo, que utilizará el primer canal liberado al terminar el primer comando COPY. Luego se habrá quedado el canal c3 sin utilizar.

CÓMO REALIZAR UNA COPIA DE TODOS LOS ARCHIVOS DE DATOS Y EL ARCHIVO DE CONTROL

1.- Conéctese a la base de datos destino y al catálogo:

```
c:\> rman target system/manager@prueba catalog rman/rman@db
```

2.- Obtenga un listado de los archivos de datos de la base de datos:

```
rman> report schema;
```

3.- Utilice el comando COPY para realizar las copias de los archivos:

```
rman> run {
2>          allocate channel c1 type disk;
3>          allocate channel c2 type disk;
4>          ...
5>          copy
```

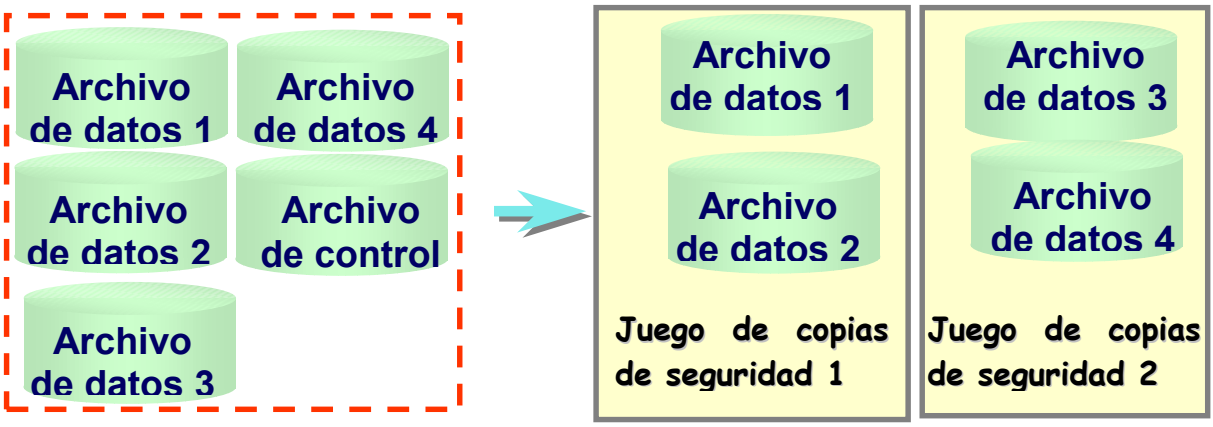
```
6>                  datafile 1 to 'destino',
7>                  ...,
8>                  current controlfile to 'destino';
9>          release channel c1;
10>         ...
11>         }
```

4.- Utilice el comando LIST para comprobar si la copia está catalogada.

```
rman> list copy;
```

Juegos de copias de seguridad

Un juego de copias de seguridad consiste en uno o más archivos físicos almacenados en disco o cinta en un formato especial de Oracle. Un juego de copias de seguridad puede contener uno o varios archivos de la base de datos.



Los juegos de copias de seguridad serán de dos tipos en función de su contenido:

- Archivo de datos: Puede contener archivos de datos y de control, no archived logs.
- Log archivado: Puede contener únicamente log archivados.

El archivo de control se puede incluir en un juego de copias de seguridad de forma explícita o implícita. Para hacerlo de la primera forma hay que incluir la opción INCLUDE CONTROL FILE en la operación de backup. La forma implícita es hacer una copia de seguridad del archivo de datos del sistema.

CONSIDERACIONES SOBRE UN JUEGO DE COPIAS DE SEGURIDAD

- Contiene uno o más archivos físicos denominados piezas de copia de seguridad.
- Se puede controlar el número de archivos de datos que contiene con el parámetro FILESPERSET.
- Los juegos de copias de seguridad pueden escribirse en disco o cinta.
- Puede limitarse el tamaño de cada pieza de un juego de copias de seguridad.
- Para extraer los archivos de un juego de copias de seguridad es necesaria una operación de restauración.
- Los juegos de copias de seguridad de logs archivados no pueden ser incrementales. Los juegos de copiass de seguridad de archivos de datos pueden ser completos o incrementales.
- Un juego de copias de seguridad se comprime mediante la no inclusión de bloques vacíos de archivos de datos.

COMANDO BACKUP

Para realizar una operación de backup en Recovery Manager se utiliza el comando BACKUP dentro de un comando RUN. Una operación de backup será interrumpida si se reciben errores de E/S durante la ejecución de la operación.

Antes de realizar una operación de backup debe realizar:

- Montar o abrir la base de datos destino. Esto influirá en que el juego de copias de seguridad sea consistente o inconsistente. Cuando la base de datos va a estar abierta ésta deberá funcionar en modo ARCHIVELOG.
- Utilizar un archivo de control actual.
- Debe utilizar un comando run donde incluir el comando BACKUP.
- Antes de utilizar el comando backup debe asignar un canal para la operación.
- Decidir un nombre único para cada pieza de copia de seguridad.

Opciones del comando BACKUP:

- ‘full’: La sesión del proceso de servidor copiará todos los bloques en el juego de copias de seguridad, sólo saltará los bloques de archivos de datos que nunca hayan sido utilizados. No se saltan bloques cuando realizamos copias de seguridad de log archivados o de archivos de control.

- ‘incremental level n’: La sesión copia todos los bloques de datos que han cambiado desde la última copia de seguridad incremental n. ‘n’ debe ser un entero del 1 al 4 y Recovery Manager comprobará que exista una copia de seguridad de nivel 0 de cada archivo del comando backup. RMAN no soporta la realización de copias de seguridad incrementales de archivos de control, log archivados o juegos de copias de seguridad.

- ‘filesperset n’: Especifica el máximo número de fichero de entrada por juego de copias. Se compara con el número de archivos de entrada por canal y se elige el menor de los dos. Si no se especifica filesperset, se toma como valor de n: 64.

- ‘skip’: Para excluir archivos de datos o log archivados del juego de copias de seguridad. Se utiliza junto con alguno de estos parámetros: offline (excluye los archivos de datos offline del juego), readonly (excluye los archivos de datos de tablespaces de sólo lectura) o inaccesible (excluye los archivos de datos o log archivados que no se pueden leer debido a errores de E/S).

- ‘setsize n’: Especifica el tamaño máximo de un juego de copias de seguridad en unidades de 1024bytes.

- ‘diskratio n’: Indica a RMAN que solamente asigne a los juegos de copias de seguridad los archivos distribuidos en el número de unidades especificado.

- ‘delete input’: Elimina los archivos de entrada una vez que se ha creado correctamente el juego de copias de seguridad. Es útil para realizar copias de seguridad de logs archivados o de copias de archivos de datos.

- ‘include current controlfile’: Crea una instantánea del control actual y la incluye en el juego de copias de seguridad.

- ‘format’: Formato para el nombre de la salida. Los siguientes parámetros podrán utilizarse para especificar el formato:
    - ‘%c’: especifica el número de copia de una pieza de backup dentro de un juego de piezas de backup duplicadas. Si no emites antes el comando SET DUPLEX, el valor será 1 para juegos de copias regulares.
    - ‘%p’: especifica el número de pieza de backup dentro de un juego de copias de seguridad. El valor comienza en 1 para cada juego de copias de seguridad y se va incrementando por cada pieza creada.

- ‘%s’: especifica el número del juego de copias de seguridad. Es un contador del archivo de control que se incrementa en cada juego de copias realizado. Comienza en 1 y es único durante la vida de un archivo de control. Si se restaura un backup del archivo de control puede haber valores duplicados. Sin embargo, la sentencia CREATE CONTROLFILE inicializa el contador a 1.
- ‘%d’: especifica el nombre de la base de datos.
- ‘%n’: especifica el nombre de la base de datos añadiéndole al mismo por la derecha caracteres ‘x’ hasta llegar a una longitud de 8 caracteres.
- ‘%t’: especifica el timestamp del backup. Es un valor de 4 bytes.

PARALELIZACIÓN DE JUEGOS DE COPIA DE SEGURIDAD

La paralelización de los juegos de copias de seguridad se consigue:

- Con la asignación de varios canales.
- Con la especificación de muchos archivos para realizar la copia de seguridad.
- Con la especificación de la opción FILESPERSET en el comando BACKUP.

Ejemplo: Supongamos que queremos realizar una copia de seguridad de nueve archivos. Los archivos de datos han sido asignados a diferentes juegos de copias.

```
rman> run {
2>      allocate channel c1 type disk;
3>      allocate channel c2 type disk;
4>      allocate channel c3 type disk;
5>      backup
6>              incremental level = 0
7>              format 'ruta/df_%d_%s_%p.bak'
8>              (datafile 1, 4, 5 channel c1 tag = DF1)
9>              (datafile 2, 3, 9 channel c2 tag = DF2)
10>             (datafile 6, 7, 8 channel c3 tag = DF3);
11>     }
```

Como habrá tres archivos por copia y hay tres canales disponibles, no es necesario especificar el parámetro FILESPERSET, se escribirán tres juegos de

copias de seguridad y cada uno incluirá bloques de tres archivos de datos. Se utilizan tres canales para escribir en paralelo.

PIEZAS DE JUEGOS DE COPIAS DE SEGURIDAD

Cuando realizamos un juego de copias de seguridad, lo normal es que sólo contenga una pieza de copia de seguridad a no ser que ejecutemos anteriormente el comando SET LIMIT.

Una pieza de copia de seguridad es un archivo físico que puede contener uno o más archivos de datos Oracle o log archivados. Un juego de copias de seguridad es un juego completo de piezas de una copia de seguridad completa o incremental.

Ejemplo:

```
rman> run {
2>      allocate channel c1 type disk;
3>      set limit channel c1 kbytes 2097152;
4>      backup
5>              format 'df_%t_%s_%p' filesperset 3
6>              (tablespace nombre_ts);
7>      }
```

En el ejemplo se realiza una copia del tablespace nombre_ts en disco. Hemos definido un tamaño máximo de archivo de 2GB. El archivo de salida si supera el tamaño especificado se dividirá en más piezas de seguridad. Cada pieza tendrá bloques intercalados de tres archivos.

JUEGO DE COPIAS DE SEGURIDAD DE ARCHIVOS DE DATOS

Cuando realizamos un juego de copias de seguridad de archivos de datos con RMAN, la herramienta sigue los siguientes pasos:

1.- Asigna buffers de memoria para cada archivo del juego.

2.- La copia de seguridad de los archivos se realiza en orden descendente en tamaño en un canal.

3.- Se realizan puntos de control en los archivos del juego y se copia el bloque de cabecera de cada archivo.

4.- Antes de incluir un bloque en la copia de seguridad se comprobará: si es una copia incremental se mira el SCN del bloque para determinar si es válido

para inclusión. Si es una copia de seguridad completa o de nivel 0, se mira si el bloque ha tenido datos alguna vez.

5.- Si se detectan bloques corruptos, se guarda información en el archivo de control y se puede consultar dicha información en la vista V$BACKUP_CORRUPTION una vez finalizada la copia de seguridad.

6.- Se realiza un cálculo del total de control.

7.- Cuando se llena el buffer de salida se envía al dispositivo de salida.

JUEGO DE COPIAS DE SEGURIDAD DE LOGS ARCHIVADOS

Un juego de copias de seguridad de log archivados no puede incluir ni archivos de datos ni de control. Además, deben ser siempre copias completas.

El proceso que se sigue para realizar un juego de copias de seguridad de logs archivados es el siguiente:

1.- Los logs archivados se ordenan por canal y tamaño.

2.- Los números de secuencia se sustituyen por un número de bloque relativo a la pieza de seguridad.

3.- No se saltan bloques ni se hacen comprobaciones incrementales.

4.- Si se detecta alguna corrupción finaliza el proceso de copia de seguridad.

Ejemplo:

```
rman> run {
2>      allocate channel c1 type disk;
3>      backup
4>              filesperset 3
5>              format 'cadena'
6>              (archivelog from logseq = X until logseq = Y
7>               delete input);
8>      }
```

El ejemplo realiza una copia de seguridad de logs archivados desde el número de secuencia X hasta el número de secuencia Y (ambos incluidos). Una vez realizadas las copias de seguridad los log archivados se eliminarán de disco los ficheros de entrada por incluir la opción delete input.

Ejemplo:

```
rman> run {
2>      allocate channel c1 type disk;
3>      backup archivelog all delete input;
4>      }
```

El ejemplo realiza una copia de todos los log archivados eliminándolos una vez realizada la copia de seguridad.

COPIAS DE SEGURIDAD COMPLETAS

Una copia de seguridad completa es una copia de seguridad de uno o más archivos de datos que contiene todos los bloques. Una copia de seguridad completa puede contener archivos de datos, copias de imagen, logs archivados o archivos de control.

COPIAS DE SEGURIDAD INCREMENTALES

Una copia de seguridad incremental de nivel mayor o igual a 0 es una copia de seguridad de un archivo de control o de datos que sólo contiene los bloques que han sido modificados desde la copia de seguridad incremental anterior.

A continuación se listan algunas de las características de las copias incrementales:

- Una copia de seguridad incremental de nivel n, donde n es mayor que 0, copia todos los bloques modificados desde la copia de seguridad incremental anterior (nivel menor o igual que n).
- Antes debe crearse una copia de seguridad de nivel 0 o una copia de imagen. Después se podrán efectuar las copias de seguridad incrementales basadas en los cambios realizados en esas copias de nivel 0.
- Las copias de seguridad incrementales son más rápidas que las copias de seguridad de nivel 0. El número de bloques que se escriben es menor.
- Son no acumulativas por defecto.

Regla empírica:

- Si hay muchas actualizaciones en bloques diferentes, realice copias de seguridad de nivel 0.
- Si hay muchas actualizaciones en pocos bloques, realice copias de seguridad de nivel mayor que 0.

En una copia incremental hay que tener cuenta dos números importantes:

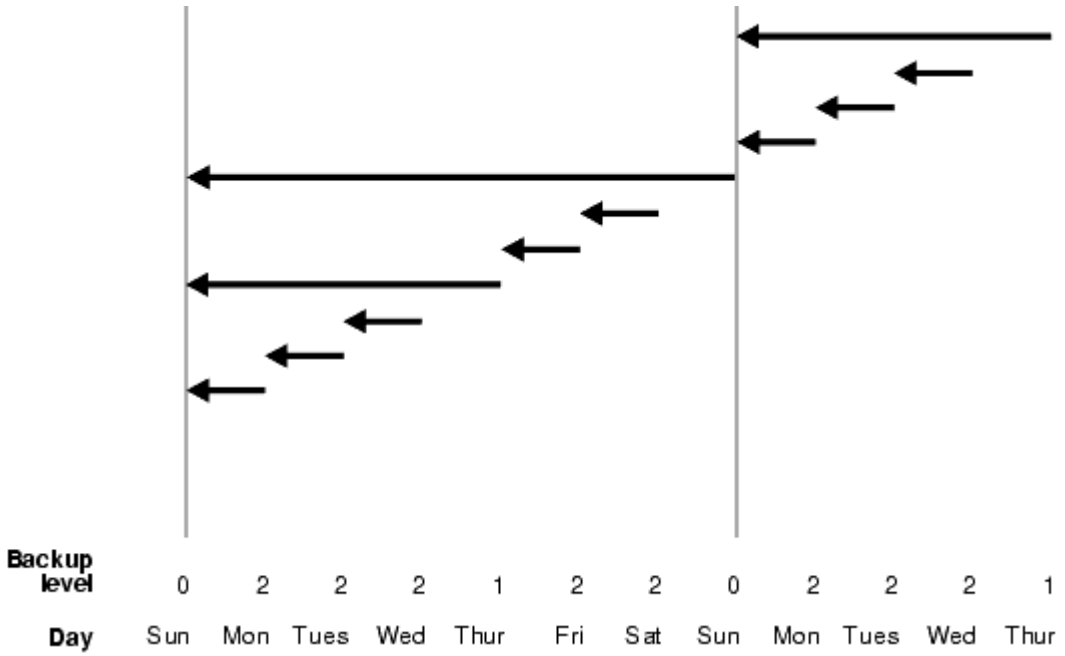
- Checkpoint_change#: El SCN del punto de control del archivo de datos en el momento en el que se inició la copia de seguridad.
- Incremental_change#: También se llama SCN de inicio incremental; será el SCN de punto de control del archivo de datos tal como estaba en la copia de seguridad incremental anterior (en la que se basa esta copia de seguridad).

Para un nivel incremental n (donde n es mayor que 0) todos los bloques van a incluir un SCN mayor o igual que incremental_change#.

Copias de seguridad incremental de varios niveles

Se pueden crear diferentes niveles de copia de seguridad incrementales. RMAN te permite definir hasta cinco niveles, del 0 al 4.

Estas copias de varios niveles facilitan las operaciones de recuperación ya que sólo se necesita una copia incremental de un nivel específico.



Ejemplo:

```
rman> run {
2>      allocate channel c1 type disk;
3>      backup
4>              incremental level = 0
```

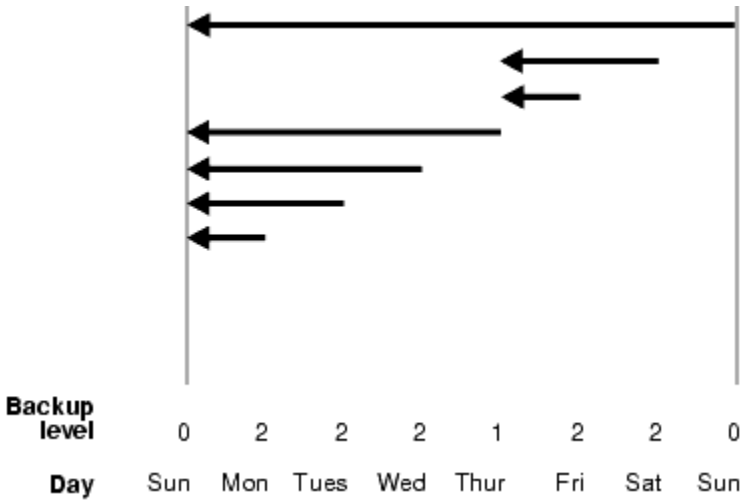
```
5>          (database);
6>     }
```

COPIAS DE SEGURIDAD INCREMENTALES ACUMULATIVAS

Una copia de seguridad incremental acumulativa necesita más especio porque incluye la información ya registrada mediante una copia de seguridad incremental anterior del mismo nivel. Ahora bien, en caso de recuperación reduce el número de copias de seguridad incrementales necesarias.

Algunas características de las copias de seguridad acumulativas son las siguientes:

- Una copia de seguridad acumulativa de nivel n con n mayor que 0, copia todos los bloques modificados desde la copia de seguridad incremental anterior de nivel menor que n.
- Las copias incrementales acumulativas copian bloques incluidos en copias de seguridad anteriores del mismo nivel. Pueden tardar más que las incrementales.
- Las copias de seguridad acumulativas aumentarán la velocidad de la recuperación en caso de ser necesaria, ya que el número de copias de seguridad que se necesita aplicar es menor.



Ejemplo:

```
rman> run {
2>     allocate channel c1 type disk;
3>     backup
4>          incremental level = 2 cumulative
```

```
5>              (database);
6>      }
```

RESTRICCIONES DE JUEGOS DE COPIAS DE SEGURIDAD

Antes de realizar una copia de seguridad con Recovery Manager debe tener en cuenta los siguientes aspectos:

- La base de datos destino deberá estar montada o abierta.
- No se soportan las copias de redo log online.
- Si la base de datos está en modo NOARCHIVELOG, sólo podrán hacerse copias consistentes. La base de datos destino debe estar montada y con una parada sin la opción ABORT.
- La herramienta RMAN no puede realizar copias de seguridad de archivos de contraseñas ni de parámetros.
- Si utiliza catálogo de recuperación, la instancia del mismo debe estar abierta.

ESCENARIOS DEL JUEGO DE COPIAS DE SEGURIDAD

COPIA DE SEGURIDAD MULTIPLEXADA EN MODO NOARCHIVELOG

1.- Comprobar que el directorio destino en el que se va a hacer la copia de seguridad está disponible y tiene espacio suficiente.

2.- Cerrar la base de datos destino con la opción NORMAL o IMMEDIATE.

3.- Montar la base de datos destino.

4.- En la herramienta RMAN reserve varios canales y configure una cadena de formato para los canales.

5.- Ejecute el comando backup correspondiente. Al estar la base de datos destino en modo NOARCHIVELOG no se pueden utilizar las copias incrementales, hay que utilizar la opción de copia completa.

6.- Verificar que la copia se ha realizado y ha sido catalogada.

7.- Abrir la base de datos destino para su uso normal.

Ejemplo:

```
C:\> rman target / catalog sysman/sysman@db
...
```

```
rman> shutdown immediate
...
rman> startup mount
...
rman> run {
2>      allocate channel c1 type disk format 'cadena_formato';
3>      allocate channel c2 type disk format 'cadena_formato_2';
4>      backup database filesperset 3;
5>      release channel c1;
6>      release channel c2;
7>      }
...
rman> list backup
```

Una copia de seguridad incremental sólo es posible cuando la base de datos está en modo ARCHIVELOG.

1.- Comprobar que la base de datos está en modo ARCHIVELOG en SQL*Plus.

2.- Conectarse con la base de datos destino y catálogo al iniciar la herramienta RMAN:

```
c:\> rman target / catalog sysman/sysman@db
```

3.- Ejecutar el comando de backup para realizar la copia incremental. Deberá tener una copia incremental de nivel 0 antes de realizar una de nivel superior.

```
rman> run {
2>      allocate channel c1 type disk;
3>      backup database incremental level 0;
4>      release channel c1;
5>      }
```

COPIA DE SEGURIDAD DE TABLESPACE

Ejemplo:

```
rman> run {
```

```
2>      allocate channel c1 type disk;
3>      backup incremental level 0 tablespace SYSTEM;
4>      release channel c1;
5>      }
```

COPIA DE SEGURIDAD DE LOGS ARCHIVADOS

Ejemplo: Realizar una copia de seguridad en disco y eliminar los archivos de entrada de la copia:

```
rman> run {
2>      allocate channel c1 type disk;
3>      backup archivelog from time 'SYSDATE-1' delete input;
4>      release channel c1;
5>      }
```

SEGURIDAD MEDIANTE ARCHIVOS DE COMANDOS ALMACENADOS

Recordemos que RMAN nos permite la utilización de archivos almacenados en el catálogo para la ejecución de comandos. En el archivo se incluirán todos los comandos a ejecutar.

Ejemplo:

1.- Creamos un archivo de texto backupcompleto.rmn que incluya la sentencia de creación de script almacenado. El contenido de este fichero debería ser:

```
create script scriptbackupcompleto {
allocate channel c1 type disk format 'cadena_formato';
allocate channel c2 type disk format 'cadena_formato_2';
backup filesperset 4 tag=SEG1 (database);
}
```

2.- Arrancamos la herramienta RMAN con la opción cmdfile para ejecutar el fichero de texto y crear de esta forma el script que se almacenará en el catálogo:

```
c:\> rman target / catalog sysman/sysman@db
cmdfile backupcompleto.rmn log salida.log
```

3.- Ahora podremos ejecutar este archivo almacenado cada vez que lo necesitemos:

```
rman> run { execute script scriptbackupcompleto; }
```

DETECCIÓN DE CORRUPCIÓN

RMAN detecta cuando se generan archivos de copia de seguridad no utilizables y puede prohibir la realización de dichas copias.

La información sobre bloques de archivos de datos corruptos se detecta durante la operación de copia de seguridad. Además se registra en el archivo de control y en el log de alertas.

Para ver los bloques de datos corruptos utilice la vista V$BACKUP_CORRUPTION si se trata de juegos de copias de seguridad o en V$COPY_CORRUPTION si se trata de copias de imagen.

El servidor identifica los bloques corruptos pero los incluye de igual forma en la copia de seguridad. Se puede establecer un número máximo de corrupciones en una copia de seguridad con el comando SET MAXCORRUPT.

Ejemplo:

```
rman> run {
2>      allocate channel c1 type disk;
3>      set maxcorrupt for datafile 'especific_fichero' to 2;
4>      copy datafile 'especific_fichero'
5>      to 'destino';
6>      release channel c1;
7>      }
```

VISTAS DEL DICCIONARIO DE DATOS

Las vistas del diccionario de datos de Recovery Manager que se utilizan para consultar el archivo de control son las siguientes:

V$ARCHIVED_LOG: Muestra información sobre los archivos que se han creado y eliminado en la base de datos, así como aquellos de los que se ha realizado copia de seguridad.

V$BACKUP_CORRUPTION: Muestra en qué archivos se han detectado bloques corruptos durante la realización de un juego de copias de seguridad.

V$COPY_CORRUPTION: Muestra qué bloques se han detectado corruptos durante una copia de imagen.

V$BACKUP_DATAFILE: Le permite determinar el número de bloques en cada archivo de datos para crear juegos de copias de seguridad del mismo tamaño. También puede calcular el número de bloques corruptos de un archivo de datos.

V$BACKUP_REDOLOG: Muestra los logs archivados almacenados en los juegos de copias de seguridad.

V$BACKUP_SET: Muestra los juegos de copias de seguridad que se han creado.

V$BACKUP_PIECE: Muestra las piezas de copia de seguridad de los juegos de copias de seguridad.

Tema 9

RESTAURACIÓN Y RECUPERACIÓN CON RMAN

Introducción

En este capítulo vamos a ver los aspectos más relevantes de las operaciones de restauración y recuperación con la herramienta Recovery Manager.

Trataremos operaciones como:

- Restauración de archivos de datos, ficheros de control y redo logs archivados.
- Recuperación de archivos de datos.
- Diferentes escenarios de restauración y recuperación.

Restauración de archivos de datos, ficheros de control y redo logs archivados

El comando para realizar restauraciones de archivos de datos, ficheros de control o redo logs archivados es el comando RESTORE.

Antes de realizar cualquier operación de restauración y recuperación debe asegurarse de tener copias de seguridad y conocer su ubicación.

Podrá restaurar archivos a su posición original o a una ubicación diferente. Para esta última opción la nueva ubicación la indicaremos con el comando set newname.

Antes de ver la restauración independiente de archivos de la base de datos veremos como podemos realizar una restauración de una base de datos completa.

RESTAURACIÓN DE UNA BASE DE DATOS

Para restaurar una base de datos a su ubicación original, la base de datos debería estar parada o los archivos de datos offline.

Los pasos a seguir para restaurar una base de datos completa son:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- Si la base de datos estuviera abierta, realizar un shutdown y dejarla en estado mount:

```
rman > shutdown immediate
rman > startup mount
```

3.- Restaurar la base de datos:

```
rman > run {
        allocate channel c1 type disk;
        allocate channel c2 type disk;
        restore database;
        }
```

RESTAURACIÓN DE TABLESPACES Y ARCHIVOS DE DATOS

Si un archivo de datos se ha perdido o está corrupto, pero sin embargo el disco donde estaba ubicado está accesible, podemos restaurarlo en su localización original. Tendremos que poner el tablespace en estado offline y a continuación emitir un comando de restauración de tablespace. Sin embargo, si la ubicación original está inaccesible, entonces tendremos que poner el tablespace offline y restaurar los archivos de datos asociados al tablespace a una nueva localización. Antes de la restauración a una nueva localización, tendremos que lanzar el comando SET NEWNAME. Este comando hace que los archivos restaurados sean considerados como copias de archivos de datros. Lanzando después de la restauración el comando SWITCH hará que los archivos restaurados sean considerados como los actuales archivos de datos. (Incluso elimina las entradas de copias de estos archivos en el catálogo de recuperación y el archivo de control).

Restauración de un tablespace a su ubicación anterior

Estos son los pasos que debe seguir para restaurar un tablespace a su posición original:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- Hay que colocar el tablespace en estado offline y restaurarlo:

```
rman > run {
        allocate channel c1 type disk;
        sql 'ALTER TABLESPACE n_ts OFFLINE TEMPORARY';
```

```
restore tablespace n_ts;
sql 'ALTER TABLESPACE n_ts ONLINE';
}
```

Restauración de un tablespace a una nueva ubicación

A continuación se muestran los pasos que debería seguir para restuarar un tablespace a una ubicación nueva:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- Hay que colocar el tablespace en estado offline, especificar la nueva ubicación para el tablespace, restaurar los archivos de datos a la nueva ubicación y reflejar dicho cambio en el archivo de control:

```
rman > run {
        allocate channel c1 type disk;
        sql 'ALTER TABLESPACE n_ts OFFLINE TEMPORARY';
        set newname for datafile 'c:\...\tbs.dbf' to 'd:\...\tbs.dbf';
        restore tablespace n_ts;
        switch datafile all;
        sql 'ALTER TABLESPACE n_ts ONLINE';
        }
```

RESTAURACIÓN DE ARCHIVOS DE CONTROL

Cuando el archivo de control de la base de datos está dañado y no tenemos copias del mismo, debemos restaurarlo desde algún backup.

Al igual que los archivos de un tablespace, el archivo de control puede ser restaurado a su ubicación hasta el momento del fallo o bien a una nueva locaclización. La diferencia entre ambas situaciones es que la réplica del archivo de control se hace de forma automática en el primer caso y habrá que indicarlo en el segundo.

Restauración del archivo de control a la ubicación anterior

Estos son los pasos que debe seguir para restaurar el archivo de control a su posición original:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- Hay que poner la base de datos en estado NOMOUNT:

```
rman > startup nomount;
```

3.- A continuación restaurar el archivo de control:

```
rman > run {
        allocate channel c1 type disk;
        restore controlfile;
        alter database open;
        }
```

La herramienta RMAN multiplexará automáticamente el archivo de control a los demás destinos del parámetro CONTROL_FILES del fichero de parámetros.

En el caso de que debamos restaurar el archivo de control de una fecha anterior, debe lanzar antes de la asignación de canales el comando SET UNTIL TIME. En este caso también es posible que necesite realizar recuperación de los archivos de datos de la base de datos.

Restauración del archivo de control a una nueva ubicación

Estos son los pasos que debe seguir para restaurar el archivo de control a una nueva ubicación:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- Hay que poner la base de datos en estado MOUNT:

```
rman > startup mount;
```

3.- A continuación restaurar el archivo de control:

```
rman > run {
        allocate channel c1 type disk;
        restore controlfile to 'd:\...\control1.ctl';
        shutdown immediate;
```

```
replicate controlfile from 'd:\...\control1.ctl';
startup;
}
```

La herramienta RMAN no multiplexa automáticamente el archivo de control y hay que indicarlo explícitamente con el comando REPLICATE. En el caso de que debamos restaurar el archivo de control de una fecha anterior, debe lanzar antes de la asignación de canales el comando SET UNTIL TIME. En este caso también es posible que necesite realizar recuperación de los archivos de datos de la base de datos.

RESTAURACIÓN DE REDO LOGS ARCHIVADOS

La herramienta RMAN también permite la restauración de los archivos de redo logs archivados. La operación de restauración, igual que ocurre con el fichero de control, podrá realizarse con la ubicación y formato indicados en los parámetros del fichero de parámetros o bien especificar un nuevo destino para los ficheros.

Los pasos a seguir para restaurar los archivos redo logs archivados son:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- Hay que poner la base de datos en estado MOUNT:

```
rman > startup mount;
```

3.- A continuación restauramos los archivos redo logs archivados:

```
rman > run {
        allocate channel c1 type disk;
        restore archivelog all;
        alter database open;
        }
```

Si queremos modificar el destino de los archivos restaurados tendremos que lanzar antes de asignar el canal, el siguiente comando:

```
Set archivelog destination to 'c:\destino_nuevo';
```

Recuperación de archivos de datos

La operación de recuperación consiste en la aplicación de redo logs o backups incrementales *a un archivo restaurado* para actualizarlo hasta un momento en el tiempo.

RECUPERACIÓN COMPLETA

Para realizar una recuperación completa de la base de datos seguiremos un procedimiento u otro en función de si el archivo de control está disponible o no.

Recuperación de la base de datos cuando el archivo de control no está dañado:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- Hay que poner la base de datos en estado MOUNT (si es necesario realizar un shutdown immediate):

```
rman > startup mount;
```

3.- A continuación restauramos y recuperamos la base de datos:

```
rman > run {
        allocate channel c1 type disk;
        restore database;
        recover database;
        alter database open;
        }
```

Comprueba los mensajes de salida de la herramienta RMAN para asegurarte que la recuperación ha sido correcta.

Recuperación de la base de datos cuando el archivo de control está dañado:

En este caso hay que utilizar un backup del archivo de control. Si utilizas catálogo de recuperación, RMAN actualiza automáticamente el archivo de control.

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- Hay que poner la base de datos en estado NOMOUNT:

```
rman > startup nomount;
```

3.- A continuación restauramos el archivo de control y restauramos y recuperamos la base de datos con los siguientes pasos:

```
rman > run {
        allocate channel c1 type disk;
        restore controlfile;
        alter database mount;
        restore database;
        recover database;
        alter database open resetlogs;
        }
```

4.- Debemos resetear la base de datos:

```
rman > reset database;
```

Es aconsejable después de una recuperación de la base de datos, realizar un backup de la misma.

Recuperación de un tablespace accesible con la base de datos cerrada:

Para realizar la recuperación de un tablespace completo cuando la base de datos está cerrada debe seguir los siguientes pasos:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- A continuación restauramos y recuperamos el tablespace con los siguientes pasos:

```
rman > run {
        allocate channel c1 type disk;
        restore tablespace n_ts;
        recover tablespace n_ts;
        }
```

Recuperación de un tablespace inaccesible con la base de datos cerrada:

Para realizar la recuperación de un tablespace inaccesible completo cuando la base de datos está cerrada debe seguir los siguientes pasos:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- A continuación restauramos y recuperamos el tablespace con los siguientes pasos:

```
rman > run {
        allocate channel c1 type disk;
        set newname for datafile 'c:\..\data1.dbf' to 'd:\..\data1.dbf';
        restore tablespace n_ts;
        switch datafile all;
        recover tablespace n_ts;
        }
```

Recuperación de un tablespace accesible con la base de datos abierta:

Para realizar la recuperación de un tablespace completo cuando la base de datos está abierta debe seguir los siguientes pasos:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- A continuación restauramos y recuperamos el tablespace con los siguientes pasos:

```
rman > run {
        sql 'ALTER TABLESPACE n_ts OFFLINE TEMPORARY';
        allocate channel c1 type disk;
        restore tablespace n_ts;
        recover tablespace n_ts;
        sql 'ALTER TABLESPACE n_ts ONLINE';
        }
```

Recuperación de un tablespace inaccesible con la base de datos abierta:

Para realizar la recuperación de un tablespace completo inaccesible cuando la base de datos está abierta debe seguir los siguientes pasos:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino. Opcionalmente nos conectaremos al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- A continuación restauramos y recuperamos el tablespace con los siguientes pasos:

```
rman > run {
        sql 'ALTER TABLESPACE n_ts OFFLINE IMMEDIATE';
        allocate channel c1 type disk;
        set newname for datafile 'c:\..\data1.dbf' to 'd:\..\data1.dbf';
        restore tablespace n_ts;
        switch datafile all;
        recover tablespace n_ts;
        sql 'ALTER TABLESPACE n_ts ONLINE';
        }
```

RECUPERACIÓN INCOMPLETA

Recordemos que una recuperación incompleta de la base de datos consiste en recuperar la base de datos completa hasta un punto anterior en el tiempo, hasta un SCN determinado o hasta un número de secuencia de log determinado. Cuando la recuperación es de la base de datos completa, la recuperación incompleta también se denomina DBPITR (database point-in-time recovery).

La principal diferencia con una recuperación completa es que al realizar una recuperación incompleta tenemos que utilizar la opción RESETLOGS para abrir la base de datos.

La forma más sencilla de llevar a cabo una recuperación incompleta de la base de datos en RMAN es lanzar un comando SET UNTIL antes de lanzar los comandos RESTORE, SWITCH o RECOVER dentro de un comando RUN. De esta forma, el comando SET UNTIL afecta a cualquiera de estos comandos que vayan detrás de él.

Recuperación incompleta con catálogo de recuperación

La base de datos debe estar cerrada para realizar una recuperación incompleta. Además es importante determinar el valor de las variables de entorno NLS_LANG y NLS_DATE_FORMAT para que la recuperación se realice correctamente.

Los pasos que debe seguir para realizar la recuperación incompleta de la base de datos completa son los siguientes:

1.- Arrancar la herramienta RMAN y conectarnos a la base de datos destino y al catálogo de recuperación:

```
c:\ rman target system/manager@db catalog rman/rman@cat
```

2.- Si la base de datos está abierta debemos pararla y dejarla en estado MOUNT:

```
rman > shutdown immediate;

rman > startup mount;
```

3.- Tendremos ahora que determinar hasta qué hora vamos a recuperar la base de datos o hasta que SCN o hasta que número de sequencia de log. Una vez determinado este parámetro podemos lanzar la recuperación de la base de datos:

```
rman > run {
        set until time 'DD-MM-YYYY';
        # set until scn 2222;
        # set until logseq 10 thread 1;
        allocate channel c1 type 'sbt_tape';
        restore database;
        recover database;
        alter database open resetlogs;
        }
```

4.- A continuación debe resetear la base de datos:

```
rman > reset database;
```

5.- Finalmente debe realizar un backup de la base de datos. Como la base de datos ha pasado a una nueva encarnación, los backups realizados antes de la apertura con resetlogs no serán válidos.

```
rman > run {
        allocate channel c1 type 'sbt_tape';
        backup database;
        }
```

Variará el comando SET UNTIL en función del tipo de recuperación que vayamos a realizar. El SCN puede determinarse con un comando list de RMAN

y el número de secuencia de log y thread lo puede obtener de la vista del diccionario de datos V$LOG_HISTORY.

Diferentes escenarios de restauración y recuperación

RESTAURACIÓN DE UNA BASE DE DATOS EN MODO NOARCHIVELOG

Para realizar la restauración, RMAN utilizará el catálogo de recuperación o el archivo de control para recoger la información que necesite para llevar a cabo la operación.

Al ser una recuperación de una base de datos que funciona en modo NOARCHIVELOG será necesario restaurar todos los archivos de datos.

Ejemplo: Tenemos en disco una copia completa de la base de datos. El archivo de control no está dañado.

```
C:\> rman target system/manager@db nocatalog
rman> startup mount
rman> run {
2>      allocate channel c1 type disk;
3>      restore database;
4>      recover database;
5>      sql 'alter database open';
6>      release channel c1;
7>      }
```

Consideraciones a la hora de restaurar una base de datos en modo NOARCHIVELOG:

- La base de datos destino debe estar montada para restaurar los archivos de datos.
- Todos los archivos de datos deben pertenecer a la misma copia de seguridad.
- Puede necesitar lanzar el comando ALTER DATABASE OPEN RESETLOGS si se ha restaurado una copia del archivo de control.
- Después de realizar una apertura con la opción RESETLOGS es necesaria una copia de seguridad completa de la base de datos.

- Si las copias de seguridad se realizaron con RMAN sólo se pueden recuperar con esta herramienta.

RESTAURACIÓN DE UNA BASE DE DATOS A UNA ENCARNACIÓN ANTERIOR

Puede ser necesario en alguna ocasión situar a la base de datos en una encarnación actual a la anterior.

Los pasos a seguir para realizar esta operación es la siguiente:

1.- Cerrar la base de datos y cambiar la ubicación del archivo de control y los archivos de datos actuales:

```
rman> shutdown immediate;
```

2.- Arrancar la base de datos en estado NOMOUNT.

3.- Restablezca la base de datos a una encarnación anterior:

```
rman> reset database to incarnation n;
```

4.- Restaurar el archivo de control a partir de la encarnación anterior:

```
rman> run {
2>      allocate channel c1 type disk;
3>      restore controlfile;
4>      release channel c1;
5>      }
```

5.- Montar la base de datos.

6.- Restaurar la base de datos:

```
rman> run {
2>      allocate channel c1 type disk;
3>      restore database;
4>      release channel c1;
5>      }
```

7.- Recuperar la base de datos y abrirla con la opción RESETLOGS:

```
rman> run {
2>      allocate channel c1 type disk;
3>      recover database;
```

```
4>      sql 'alter database open resetlogs';
5>      release channel c1;
6>      }
```

RECUPERACIÓN DE UN ARCHIVO DE DATOS SIN UN BACKUP

En realidad no recuperamos el archivo de datos, lo que hacemos es recuperar los datos perdidos y crear un nuevo fichero de datos con el mismo nombre que el fichero de datos perdido.

Los pasos a seguir para realizar esta operación es la siguiente:

1.- Ejecute los siguientes pasos con la herramienta RMAN:

```
rman> run {
        allocate channel c1 type disk;
        sql 'ALTER DATABASE DATAFILE "c:\..\data1.dbf" OFFLINE';
        sql 'ALTER DATABASE CREATE DATAFILE "c:\..\data1.dbf"';
        recover datafile 'c:\..\data1.dbf';
        sql 'ALTER DATABASE DATAFILE "c:\..\data1.dbf" ONLINE';
        }
```

Tema 10

TEMAS ADICIONALES

Introducción

En este capítulo vamos a tratar algunos aspectos que son de bastante importancia para operaciones de backup y recuperación como: archivos de alerta, archivos de rastreo y además algunas utilidades Oracle para ayudar al administrador en tareas de este tipo.

Recuperación de archivos de datos no esenciales

Cuando un tablespace está dañado y se trate del tablespace TEMP o INDEX la mejor opción es volver a crearlos en vez de intentar realizar una recuperación de los mismos, es más rápido.

Para recuperar uno de estos tablespaces deberá seguir los siguientes pasos:

1.- La base de datos debe estar en estado MOUNT.

```
SQL> alter database mount;
```

2.- Consulte las vistas v$recover_file y v$datafile para obtener los nombres de los archivos a recuperar:

```
SQL> select * from v$recover_file;
```

3.- Desconectar el archivo de la base de datos para borrarlo:

```
SQL> alter database datafile '...' offline drop;
```

4.- Abra la base de datos:

```
SQL> alter database open;
```

5.- Borre el tablespace oportuno:

```
SQL> drop tablespace temp;
```

6.- Crear el tablespace de nuevo con el mismo nombre que anotó de la vista v$recover_file.

```
SQL> create tablespace temp datafile '...' size 500k;
```

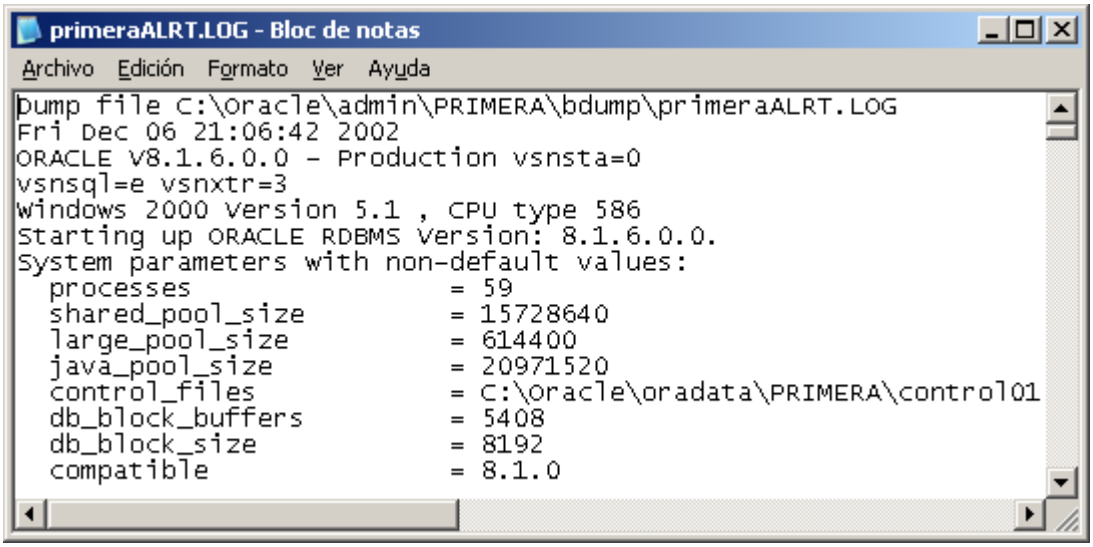
El archivo Log de Alertas

El archivo log de alertas es creado y mantenido por el servidor Oracle. Si existe algún problema con la base de datos, es el primer lugar donde el Administrador debe buscar información sobre el error, ya que es en este archivo deonde se almacena información de todo lo que afecta a la instancia.

Por ejemplo, se almacena información referente a todos los inicios y cierres de la instancia, se registra cualquier operación DDL que se realice y también guarda información sobre los errores y eventos en la base de datos.

Es aconsejable realizar una copia de este archivo periódicamente y depurarlo para evitar que crezca demasiado.

El archivo de alertas se encuentra en el directorio especificado en el parámetro BACKGROUND_DUMP_DEST donde también se encuentran los archivos de rastreo de los procesos en segundo plano. Generalmente el nombre del archivo de alertas se forma con el nombre de la base de datos más ALRT.log



Los archivos de rastreo

Cuando uno de los procesos en segundo plano detecta un error, genera un archivo de rastreo con información sobre dicho error.

La ubicación de estos archivos de rastreo es el directorio especificado en el parámetro de inicialización BACKGROUND_DUMP_DEST. La misma ubicación que el archivo de alertas de la base de datos.

Normalmente el nombre del archivo de rastreo contiene el nombre del proceso en segundo plano que lo creó para poder identificarlo y la extensión de estos archivos será TRC.

También es importante, como para el archivo de alerta, que el administrador de la base de datos revise estos ficheros periódicamente para comprobar si hay errores y depurarlos si fueran demasiado grandes.

Detección y reparación de corrupción de bloques

Existen diferentes formas de detectar bloques corruptos en la base de datos:

- Parámetros de inicialización DB_BLOCK_CHECKSUM y DB_BLOCK_CHECKING
- Paquete DBMS_REPAIR
- Comando ANALYZE VALIDATE STRUCTURE
- Utilidad DBVERIFY

PARÁMETROS DE INICIALIZACIÓN DB_BLOCK_CHECKSUM Y DB_BLOCK_CHECKING

Si asignamos valor TRUE en el archivo de parámetros de inicialización al parámetro DB_BLOCK_CHECKSUM, se calculará una suma de control para cada bloque y que se almacena en la cabecera del mismo. La siguiente vez que se lea un bloque de datos o un bloque de log,m se utilizará la suma de control para detectar si hay o no corrupción.

En caso de detectar alguna corrupción, se devolverá el mensaje ORA-01578 y se escribirá información sobre el bloque corrupto en un archivo de rastreo.

Cuando definimos este parámetro a TRUE, estaremos sobrecargando el rendimiento de la base de datos, por ello es sólo aconsejable en algunas circunstancias especiales.

Sin embargo, el poner a TRUE el parámetro DB_BLOCK_CHECKING implica un impacto mínimo en el rendimiento. En este caso no se comprueban los bloques en cada lectura sino en los cambios. En caso de detectar un bloque corrupto, genera el error ORA-1578 y lo escribe en el fichero de alertas.

PAQUETE DBMS_REPAIR

En primer lugar tendremos que crear este paquete incorporado en Oracle. Para ello tendremos que lanzar como usuario SYS los archivos de comando: dbmsrpr.sql y prvtrpr.plb. Estos archivos se encuentran en \ORACLE_HOME\RDBMS\ADMIN.

El paquete DBMS_REPAIR se utiliza para identificar corrupciones en bloques de tablas, particiones o índices. Genera una lista de los bloques que están corruptos.

Será necesario crear una tabla para almacenar los resultados antes de utilizar el paquete. El propio paquete proporciona un procedimiento para crear esta tabla de resultados, el procedimiento ADMIN_TABLES. Los parámetros de este procedimiento son los siguientes:

TABLE_NAME	Será el nombre para la tabla que almacenará los resultados de la utilidad

	de la utilidad.
TABLE_TYPE	Este parámetro puede tomar los siguientes valores: - REPAIR_TABLE: Si se va a comprobar una tabla. - ORPHAN_TABLE: Si se va a comprobar un índice
ACTION	EL tercer parámetro podrá tomar uno de los siguientes valores: - CREATE_ACTION: Para cuando se está creando la tabla de resultados - PURGE_ACTION: Para eliminar los datos relacionados con objetos no existentes - DROP_ACTION: Para borrar la tabla
TABLESPACE	Nombre del tablespace cuando estemos creando la tabla.

Ejemplo: Para crear una tabla de resultados se podría lanzar el siguiente comando:

```
BEGIN
 dbms_repair.admin_tables('RESULT_BEA',
                                DBMS_REPAIR.REPAIR_TABLE,
                                DBMS_REPAIR.CREATE_ACTION,
                                'TS_1');
END;
/
```

La sentencia del ejemplo crearía la tabla RESULT_BEA más una vista con prefijo DBA_, es decir la vista: DBA_RESULT_BEA.

Para comprobar si hay corrupción en una tabla, partición o índice se utilizará el procedimiento CHECK_OBJECT. Este procedimiento comprobará todos los bloques del objeto a analizar y rellenará la tabla de reparación que se indique especificando las direcciones de los bloques corruptos.

Los parámetros del procedimiento CHECK_OBJECT son:

SCHEMA_NAME	Nombre del esquema que contiene el objeto que se va a analizar.

OBJECT_NAME	Nombre del objeto que se va a analizar.
PARTITION_NAME	Nombre de la partición a analizar si sólo es una.
OBJECT_TYPE	Podrá tomar los valores: - TABLE_OBJECT: si se comprueba una tabla. - INDEX_OBJECT: si se comprueba un índice.
REPAIR_TABLE_NAME	Nombre de la tabla que almacena el resultado del procedimiento.
FLAGS	No se utiliza.
RELATIVE_FNO	Número del archivo relativo cuando se comprueba un rango de bloques.
BLOCK_START	Número de bloque inicial cuando se comprueba un rango de bloques.
BLOCK_END	Número de bloque final cuando se comprueba un rango de bloques.
CORRUPT_COUNT	Una variable de salida para almacenar el núemero de bloques corruptos.

Ejemplo: Para analizar la tabla EMP del usuario BEA.

```
VARIABLE corcnt NUMBER
BEGIN
  Dbms_repair.check_object('BEA', 'EMP',
                                    REPAIR_TABLE_NAME=> 'REPAIR_BEA',
                                    Corrupt_count=>:corcnt);
END;
/
```

El ejemplo analiza los bloques corruptos de la tabla EMP del esquema BEA. Una vez ejecutado podemos consultar la variable CORCNT para ver el número

de bloques corruptos del objeto analizado y consultar la tabla de reparación REPAIR_BEA para ver el detalle de los bloques corruptos si los huebiera.

Este procedimiento únicamente da información de los bloques corruptos pero no los marca como tal (la columna MARKED_CORRUPT de la tabla de reparación tendrá valor FALSE). La forma de actuar en caso de que huebiera algún bloque corrupto sería intentar extraer la información posible de ese bloque y posteriormente marcarlo como corrupto para que el bloque sea saltado completamente la siguiente vez que se intente acceder a él.

El procedimiento FIX_CORRUPT_BLOCKS se encarga de marcar los bloques corruptos. Utiliza la tabla de reparación para realizar su objetivo, cambiando después el valor de la columna MARKED_CORRUPT a TRUE y añadiendo la hora de marcado en la columna FIX_TIMESTAMP. Los parámetros para este procedimientos son:

SCHEMA_NAME	Nombre del esquema que contiene el objeto que se está tratando.
OBJECT_NAME	Nombre del objeto que se va a tratar.
PARTITION_NAME	Nombre de la partición a tratar.
OBJECT_TYPE	Podrá tomar los valores: - TABLE_OBJECT: si se comprueba una tabla. - INDEX_OBJECT: si se comprueba un índice.
REPAIR_TABLE_NAME	Nombre de la tabla que almacena el resultado del procedimiento.
FLAGS	No se utiliza.
FIX_COUNT	Una variable de salida para almacenar el número de bloques corruptos marcados.

Ejemplo:

```
VARIABLE mrccnt NUMBER
BEGIN
  Dbms_repair.fix_corrupt_blocks('BEA', 'EMP',
                                 REPAIR_TABLE_NAME=> 'REPAIR_BEA',
                                 fix_count=>:mrccnt);
END;
```

Una vez marcados los bloques corruptos habría que activar el salto de bloques corruptos con el procedimiento SKIP_CORRUPT_BLOCKS. Este procedimiento tiene los siguientes parámetros:

SCHEMA_NAME	Nombre del esquema al que pertenece el objeto tratado.
OBJECT_NAME	Nombre del objeto.
OBJECT_TYPE	Los valores pueden ser: - TABLE_OBJECT: Si es una tabla. - INDEX_OBJECT: Si es un índice.
FLAGS	Los valores pueden ser: - SKIP_FLAGS: Activar el salto de bloques corruptos. - NOSKIP_FLAG: Desactivar el salto de bloques corruptos (se generan errores ORA-1578)

Ejemplo:

```
BEGIN
  Dbms_repair.skip_corrupt_blocks('BEA', 'EMP');
END;
/
```

Podemos obtener información de si está o no activado el salto de bloques corruptos de una tabla en la vista DBA_TABLES con la columna SKIP_CORRUPT.

Cuando marcamos los bloques corruptos en una tabla sabemos que esos bloques son inaccesibles. Puede ocurrir que la tabal tenga asociado algún

índice en la base de datos y este índice podrá hacer referencia a alguno de estos bloques corruptos. Deberá regenerar los índices de una tabla después de marcar bloques corruptos.

El paquete DBMS_REPAIR contiene procedimientos para identificar índices que apuntan a bloques marcados como corruptos de una tabla y realizar reconstrucciones de estos índices. Sin embargo, la mejor opción para los índices es reconstruirlos completamente como se dijo en el párrafo anterior.

COMANDO ANALYZE VALIDATE STRUCTURE

Este comando se puede utilizar para validar estructuras de la base de datos. El comando actuará de una forma u otra en función del tipo de objeto que se analice.

En el caso de tablas, el comando realiza la verificación de la integridad de cada bloque de datos y de cada fila de la tabla.

En el caso de tablas particionadas, se verifica que las filas pertenezcan a la partición correcta. Si no es así, el identificador de la fila se almacenará en la tabla INVALID_ROWS o en otra diferente utilizando la cláusula INTO. La tabla INVALID_ROWS se crea lanzando el script utlvalid.sql que se encuentra en el directorio \ORACLE_HOME\RDBMS\ADMIN.

En el caso de índices, verifica la integridad de cada bloque de datos del índice y comprueba si existe corrupción de bloque.

En cualquiera de los casos, Oracle sacará un mensaje de confirmación de validación en caso de que haya sido correcta o un mensaje de error en el caso de haber detectado corrupción en la estructura del objeto.

Ejemplos:

```
ANALYZE INDEX n_indice VALIDATE STRUCTURE;

ANALYZE TABLE n_table VALIDATE STRUCTURE CASCADE;
```

La opción CASCADE que se añade al validar una tabla es opcional y lo que indica es que a la vez que se valida la tabla, se validen los índices asociados a ella.

UTILIDAD DBVERIFY

Es una utilidad externa a la base de datos. Permite a los administradores de la base de datos comprobar la integridad de los bloques de los archivos de datos de la base de datos. Sólo los archivos de datos.

Al ser una utilidad externa el impacto sobre las actividades de la base de datos es mínimo.

El ejecutable de la utilidad DBVERIFY se llama dbv y se encuentra en el directorio BIN en el ORACLE_HOME.

Es muy útil la herramienta DBVERIFY para verificar antes de una restauración que las copias que se van a utilizar de los archivos de datos son correctas.

Los parámetros de DBVERIFY son:

- FILE: Nombre del archivo a analizar.
- START: Dirección del bloque inicial de verificación.
- END: Dirección del bloque final de verificación.
- BLOCKSIZE: Sólo será necesario cuando el tamaño del bloque de datos sea mayor de 2kb.
- LOGFILE: Especifica el archivo donde se almacenará la información de salida.
- HELP: Muestra ayuda por pantalla.
- PARFILE: Especifica el nombre del archivo de parámetros a utilizar.

Ejemplo:

```
Símbolo del sistema
C:\Oracle\Ora81\BIN>dbv file='c:\oracle\oradata\primera\users01.dbf' blocksize=8
192

DBVERIFY: Release 8.1.6.0.0 - Production on Mar Feb 4 13:44:18 2003

(c) Copyright 1999 Oracle Corporation.  All rights reserved.

DBVERIFY - Iniciando verificaci¾n : FILE = c:\oracle\oradata\primera\users01.dbf


DBVERIFY - Verificaci¾n finalizada

Total PØginas Examinadas      : 2560
Total PØg. Procesadas (Datos): 9
Total PØg. con Fallos (Datos): 0
Total PØg. Procesadas (Indice): 7
Total PØg. con Fallos (Indice): 0
Total PØg. Procesadas (Otras): 19
Total PØginas Vacýas          : 2525
Total PØg. Marcadas Corruptas: 0
Total PØginas de Entrada      : 0

C:\Oracle\Ora81\BIN>
```

Utilidad LOGMINER

La utilidad logminer le permite realizar análisis sobre la información de archivos de redo log. Podrá generar un listado de sentencias SQL que se han lanzado sobre la base de datos y de aquellas sentencias que permiten dar marcha atrás en dichas operaciones. Para esta información utiliza una vista V$LOGMNR_CONTENTS y las dos columnas SQL_REDO y SQL_UNDO.

CÓMO REALIZAR UN ANÁLISIS CON LA HERRAMIENTA LOGMINER

Antes de comenzar el análisis de la información de los archivos de redo log, tendremos que crear un archivo de diccionario para la herramienta. Para crear este archivo tendremos que inicializar el valor del parámetro UTL_FILE_DIR en el init.ora de la base de datos. En el parámetro indicaremos el directorio o la ubicación del archivo de diccionario. A continuación deberá utiliza el procedimiento BUILD del paquete DBMS_LOGMNR_D para crear el fichero de diccionario como se muestra en el siguiente ejemplo:

```
BEGIN
  dbms_logmnr_d.build('prilgmdict.ora',
                      'c:\oracle\admin\primera\logminer');
END;
/
```

El procedimiento crea el archivo de dicccionario indicado como primer parámetro en la ubicación especificada en el segundo.

El siguiente paso es establecer una lista de los archivos de redo log que se van a analizar. Pueden ser tanto redo log archivados como redo logs online. Para crear esta lista de archivos tenemos el procedimiento ADD_LOGFILE del paquete DBMS_LOGMNR que lo debemos utilizar como se especifica a continuación:

```
BEGIN
  dbms_logmnr.add_logfile('c:\..\log1.arc',
                          DBMS_LOGMNR.NEW);
  dbms_logmnr.add_logfile('c:\..\log2.arc',
                          DBMS_LOGMNR.ADDFILE);
END;
/
```

El parámetro DBMS_LOGMNR.NEW se utiliza para desechar archivos suministrados anteriormente y comenzar una lista nueva de archivos a analizar.

El parámetro DBMS_LOGMNR.ADDFILE se utiliza para añadir archivos a la lista creada con el parámetro anterior.

Una vez creada la lista de archivos a analizar hay que comenzar el análisis de la información de los archivos de redo log. Para ello disponemos del procedimiento START_LOGMNR del paquete DBMS_LOGMNR. Es importante determinar el rango en el que se va a realizar el análisis para asegurarnos que estén incluidos los archivos correctos.

Este procedimiento admite cinco parámetros:

STARTSCN	Límite inferior del rango SCN que se va a anlaizar.
ENDSCN	Límite superior del rango SCN que se va analizar.
STARTTIME	Un valor de fecha que indica el límite inferior del rango de tiempos del análisis.
ENDTIME	Un valor de fecha que indica el límite superior del rango de tiempos de análisis.
DICTFILENAME	Nombre del archivo de diccionario de LOGMINER

Ejemplo:

```
BEGIN
dbms_logmnr.start_logmnr(dictfilename=>'...\prilgmdict.ora',
   starttime=>to_date('04/02/2003:03PM','DD/MM/YYYY:HHPM'),
   endtime=>to_date('04/02/2003:04PM','DD/MM/YYYY:HHPM'));
END;
```

Una vez terminado el análisis, se habrá rellenado la vista V$LOGMNR_CONTENTS con la información obtenida por la herramienta. Alunas de las columnas de interés de esta vista son las siguientes:

- SCN: Número de cambio del sistema, para realizar una recuperación incompleta basada en el SCN.
- TIMESTAMP: Registro de hora, información exacta para realizar una recuperación hasta un punto en el tiempo.
- USERNAME: Nombre del usuario que ejecutó la sentencia.

- SQL_REDO: La sentencia que se ejecutó.
- SQL_UNDO: Cómo deshacer la sentencia ejecutada.

También puede consultar otras vistas que incluyen otra información referente al análisis. Por ejemplo, la vista V$LOGMNR_LOGS contiene información sobre los archivos de redo incluidos en el último análisis. La vista V$LOGMNR_PARAMETERS contiene información sobre los parámetros que se han utilizado en el análisis y la vista V$LOGMNR_DICTIONARY que contiene información sobre el archivo de diccionario de la herramienta LOGMINER.

El último paso en la utilización de la herramnienta LOGMINER es la liberación de recursos utilizados en un análisis. Esta operación debería realizarla ya después de haber consultado la información, ya que el procedimiento de liberación de recursos suprime dicha información. El procedimiento que realiza esta liberación de recursos es el procedimiento END_LOGMNR del paquete DBMS_LOGMNR.

Ejemplo:

```
BEGIN
  dbms_logmnr.end_logmnr;
END;
/
```