59
2 ej.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES

CAMPUS ARAGÓN

ORACLE DATA WAREHOUSE

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A :

ROBERTO RODRÍGUEZ DÍAZ

ASESOR:

ING. DONACIANO JIMENEZ VAZQUEZ

MEXICO **1998**

TESIS CON
FALLA DE ORIGEN

267082

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

A MIS PADRES:

Alicia y Tiburcio, porque gracias a su esfuerzo y apoyo durante todo este tiempo he logrado mi superación, tanto personal como profesional. Porque ellos me han enseñado con su ejemplo a ser lo mejor que puedo ser. Por ser las personas más significativas en mi vida... Y también por ser las personas a las que más quiero.

A MI HERMANA:

Yolanda, por todo lo que hemos compartido desde la infancia. Porque juntos aprendimos a apoyarnos en todo lo que necesitábamos. Y por ese cariño que ha ido creciendo junto con nosotros.

A MI ESPOSA:

Xohitl Zoraida, porque ella me ha enseñado una nueva forma de ver la vida. Por todo el apoyo, comprensión y paciencia que me ha tenido durante el desarrollo de este trabajo. Y por ser el "Xol" que me ilumina...

A MI ASESOR:

Ing. Donaciano Jiménez Vázquez, por su valiosa colaboración y tiempo dedicado a este trabajo.

A TODAS LAS PERSONAS:

Que de alguna u otra forma intervinieron en la realización de este trabajo.

A MI ESCUELA Y PROFESORES:

Porque ahí encontré todo lo necesario para realizar mi formación profesional.

Indice

Introducción

INTRODUCCIÓN

Hasta hace unos años los sistemas transaccionales eran los encargados de soportar la información de un negocio, pero éstos sólo manejan las operaciones a un nivel muy detallado; lo cual no era muy bueno para los gerentes o personas encargadas del análisis de los datos de una empresa, ya que tenían que esperar a que el departamento de sistemas elaborara el reporte que ellos necesitaban para el análisis de su empresa, lo cual podía llevarse de días hasta semanas para que el reporte se recibiera en la forma requerida por el gerente. Por otra parte el área de sistemas tenía que "sufrir" tratando de dar formato, hacer consultas e imprimir los archivos que se generaran para poder entregar los reportes con todos los requerimientos que el gerente había solicitado.

Las personas encargadas de la toma de decisiones eran dependientes del área de sistemas, en lo que a información se refiere, ya que para poder adquirir información de las operaciones de la empresa debían recurrir a esta área. Y en ocasiones el área de sistemas no podía proporcionar los reportes requeridos por la gerencia porque existían ciertas circunstancias que no permitían elaborar los reportes con los formatos especificados por la gerencia.

Por otra parte los sistemas transaccionales sólo podían dar respuesta a preguntas como: ¿Cuántos productos se han vendido en el presente mes? ¿Cuál es el producto más caro? ¿Cuántos productos tengo en existencia? En cambio a la gerencia le interesaba contestar preguntas como: ¿Qué pasaría si se incrementa el precio a un producto X ? ¿Puedo reducir el precio de un producto sin afectar el consumo de otros? ¿ Qué pasaría si reducimos la existencia de un producto X en almacén?

Este tipo de cuestiones no podían ser contestadas por los sistemas transaccionales, así que el gerente tenía que ingeniárselas para pode realizar análisis de su negocio tomando los datos que sus sistemas transaccionales le otorgaban. Hasta que se desarrollo la idea del Data Warehouse, el cual vino a cambiar la forma de manejo de la información.

Con esta nueva herramienta, los gerentes han podido sacar el mayor provecho a su información y han logrado sacar ventajas competitivas ante sus diferentes "rivales" de negocios. Es por esto que el Data Warehouse se está convirtiendo en una poderosa herramienta para que en los negocios se puedan tomar decisiones a tiempo y en forma eficiente a través del correcto análisis de la información.

Precisamente este trabajo está dedicado a los fundamentos de esta nueva tecnología llamada Data warehouse. Conceptos que se manejan en el entorno Data Warehouse así como las herramientas utilizadas para su explotación. Sin duda alguna esta es la nueva era de la *información* gerencial para el desarrollo de los sistemas de soporte de decisiones.

ii El uso de dos dígitos para especificar la posición es meramente estético. Algunos archivos de control grandes pueden ser difíciles de leer si hay muchas líneas con las especificaciones no alineadas. Fíjese en cómo se refiere a los campos numéricos de la tabla como "character data" (datos de tipo carácter) en el archivo de entrada. El número "23", por ejemplo, es simplemente el carácter alfanumérico "2" seguido de un "3", aunque la tabla destinataria los trate como números.

iii Fíjese en el paréntesis de cierre de la última línea de posiciones. No hay una coma al final de la última línea.

2.1.4 Ejemplo de carga con SQL*Loader

Para acabar esta sección del capítulo, he aquí un ejemplo real de sesiones de SQL*Loader. Esté ejemplo cargará datos de un archivo de entrada en la tabla CUENTAS.

EJEMPLO. Esta sesión está conforme a las especificaciones de la tabla siguiente:

Especificación	*Valor*
Archivo de entrada	Cuenta.dat
Archivo de control	Cuenta.ctl
Archivo de erróneos	Cuenta.bad
Archivo de registro	Cuentaload.out
Tabla destino	Cuenta
Estado de la tabla antes de la carga	Preservar los datos existentes

Tabla 2.1 Especificaciones para el ejemplo de SQL*Loader

```
load data
infile 'cuentas.dat'
into table cuentas append

(número_cuenta          position( 1:10)   char,
 tipo_cuenta            position(11:12)   char,
 propietario            position(13:42)   char,
 última_actividad       position(43:48)   date 'YYMMDD',
 estado                 position(49:50)   char)
```

Fíjese que los datos de entrada toman el formato de fecha como 'YYMMDD' para sus fechas. Este formato -por ejemplo, 18 enero, 1998, aparecerá como 98118- aparece en el archivo de control. A medida que los datos se van introduciendo en Oracle, se convierten al formato que tiene Oracle por defecto y se convierte en 18-JAN-98. En el siguiente listado aparece la salida de la ejecución de SQL*Loader:

```
sql ldr userid = system/manager control = account.ctl  log = account.out
SQL*Loader: Release 7.3.2.1.0 - Productión on Sat Jan 11 18:21:45 1997
Copyright  ( c ) Oracle Corporation 1979, 1994. All rights reserved.

Commit point reached - logical record count 64
Commit point reached - logical record count 128
Commit point reached - logical record count 192
Commit point reached - logical record count 256
Commit point reached - logical record count 320
Commit point reached - logical record count 384
Commit point reached - logical record count 448
Commit point reached - logical record count 512
Commit point reached - logical record count 576
Commit point reached - logical record count 640
Commit point reached - logical record count 704
Commit point reached - logical record count 768
Commit point reached - logical record count 832
Commit point reached - logical record count 896
Commit point reached - logical record count 960
```

Veamos ahora el archivo de registro:

```
SQL*Loader: Release 7.3.2.1.0 - Productión on Mon Dec 30 16:47:13 1996
Copyright  ( c ) Oracle Corporation 1979, 1994. All rights reserved.

Control File:      ./.cuenta.ctl
Data File:         ./.cuenta.dat
   Bad File:       ./.cuenta.bad
   Discard File:   none specified

(Allow all discards)
Number to load: ALL
Number to skip: 0
Errors allowed: 50
Bind array:     64 rows, maximum of 65536 bytes
Continuation:    none specified
Path used:          Convertional
```

```
Table CUENTA, loaded from every logical record.
Insert option in effect for this table: APPEND

Column Name                    Position       Len  Term  Encl  Datatype
NUMERO_CUENTA                   1:10          10                CHARACTER
TIPO_CUENTA                     11:12          2                CHARACTER
PROPIETARIO                     13:42         30                CHARACTER
ÚLTIMA_ACTIVIDAD                43:48          6                CHARACTER
ESTADO                          49:50          2                CHARACTER

Table CUENTA:
  960 Rows successfully loaded.
  0 Rows not loaded due to data errors.
  0 Rows not loaded because all WHEN clauss were failed.
  0 Rows not loaded because all fields were null.

Space allocated for bind array:                  4608 bytes (64 rows)
Space allocated for memory besides bind array:   56709 bytes
Total logical records skipped:          0
Total logical records read:           960
Total logical records rejected:         0
Total logical records discarded:        0

Run began on Mon Dec 30 16:47:13 1996
Run ended on Mon Dec 30 16:47:20 1996
Elapsed time was:     00:00:01.12
CPU time was:         00:00:00.17
```

Parece que todo ha ido bien. Las 960 líneas han sido cargadas. En caso de que hubiera ocurrido algún problema, el listado describiría la misma sesión de carga, pero con algunos datos erróneos:

```
Record 64: Rejected - Error on table CUENTAS, column ÚLTIMA_ACTIVIDAD.
ORA-01839: date not valid for month specified
  Record 203: Rejected - Error on table CUENTAS, column NUMERO_CUENTA.
ORA-01722: invalid number

Table CUENTA:
  958 Rows successfully loaded.
  2 Rows not loaded due to data errors.
  0 Rows not loaded because all WHEN clauss were failed.
  0 Rows not loaded because all fields were null.
```

```
Space allocated for bind array:                 4608 bytes (64 rows)
Space allocated for memory besides bind array:  56709 bytes
Total logical records skipped:          0
Total logical records read:           960
Total logical records rejected:         2
Total logical records discarded:        0
```

El registro 64 ha sido rechazado porque tiene un mes con el número 15; el registro 203 también lo fue porque se esperaba que el número de cuenta fuera numérico, pero contenía al menos un carácter no numérico. Las filas erróneas han sido escritas en cuenta.bad, que aparece en el siguiente listado. Los datos erróneos están escritos en negrita:

```
0067897782 RRDatabase Technologies                961529TT
889GHG8777GGlan Abramson Systems                  960529TT
```

Lo más importante de este archivo de erróneos es que los datos erróneos se pueden arreglar y usarlos más tarde como entrada para otra sesión de SQL*Loader. Al escribir las filas erróneas en un archivo de erróneos, proporciona dos grandes ventajas:

1. No se pueden escribir datos erróneos en la base de datos.

2. Los datos erróneos se pueden arreglar y alimentar al sistema con ellos en alguna sesión futura, proporcionando al operador un alto grado de chequeo de la completitud.

2.1.5 Carga Paralela y directa.

La tecnología de carga paralela de Oracle es una característica fundamental del cargador de datos a la hora de poblar el data warehouse. Esta tecnología se enorgullece de un potencial de carga superior a 100 gigabytes por hora. Antes de la versión 7.2, la carga paralela necesitaba la Parallel Query Option (opción de carga paralela) del Oracle7 server. Con la versión 7.3, ésta viene con el producto base. La construcción **create table unrecoverable as**; se puede usar para crear el data warehouse accediendo a los datos usando sentencias SQL estándar. Cuando se traen los datos de orígenes externos y estos son datos de tipo texto, se utiliza el SQL*Loader de Oracle para insertar los datos en el repositorio del DSS.

SQL*Loader, normalmente, lee los datos de entrada y se los da al motor de SQL*Loader de Oracle, quien los pone en la base de datos usando una sentencia SQL INSERT. Puede

que esté familiarizado con la creación de filas en la base de datos usando la siguiente sintaxis:

```
insert into suma_cuenta values (129,'12-DEC-98', 19291000, 249, 'VRP');
```

Cuando se trabaja con una herramienta de cara al usuario como Oracle Forms, ésta sólo pasa la sentencia SQL a Oracle para que lo procese. Si un usuario tiene una pantalla donde se crean nuevas filas y hay un botón de *Salvar*, cuando se pulsa éste, se formatean los datos y se pasan a Oracle como una sentencia insert. Usar la paralelización del SQL*Loader junto con la opción de modo de carga directa, hace que la carga sea más rápida que cuando se utilizan los mecanismos de carga convencionales. Supongamos que hay tres conjuntos de datos que hay que cargar en un data warehouse; todos ellos van en la misma tabla. Cuando la carga se termine, la tabla destino tendrá los datos combinados de los tres archivos de entrada. El siguiente listado muestra la forma de realizar esto:

```
sql userid = /  control = acct1.ctl   direct = true parallel = true
sql userid = /  control = acct2.ctl   direct = true parallel = true
sql userid = /  control = acct3.ctl   direct = true parallel = true
```

Las sesiones múltiples realizan la carga con dos profundas diferencias respecto a la carga convencional. Estas dos diferencias (paralela y directa) hacen de SQL*Loader una parte muy valiosa en el proceso de carga de datos en un DSS.

1. Los procesos que realizan la carga lo hacen de forma concurrente y el trabajo de carga se reparte entre ellos. Al repartir el trabajo distribuyéndolo entre más de un proceso, se hace más rápido y se consumen menos recursos. La paralelización permite, además, cargar en la misma tabla al mismo tiempo.

2. Con la opción de carga directa, SQL*Loader reúne, y después da formato, los datos en la memoria en la misma estructura en que se almacenaran en la tabla de base de datos. La carga convencional (que usa sentencias SQL*Loader para cada columna consume más tiempo y recursos.

2.2 Import y Export.

Estos son dos productos hermanos que permiten copiar datos de Oracle (Export) en un archivo binario comprimido legible sólo por Oracle y copiar los datos de vuelta desde este archivo a una o más tablas de Oracle (Import). Export e Import han estado ahí desde el

principio; son una parte necesaria de la estrategia para la realización de copias de seguridad de la mayoría de las instalaciones de Oracle.

Export e Import se pueden usar como parte de la estrategia de migración a la hora de transferir datos desde los sistemas operacionales al data warehouse de Oracle. Primero, veamos los tres modos de operación para Export e Import; después, los tres métodos dentro de los que opera.

2.2.1 Modos de operación.

Export e Import operan en uno de los tres modos. Cada modo se usa en diferentes situaciones dependiendo de qué se deba de hacer. Estos modos se enumeran en la tabla 2.2 se aplican tanto para Import como para Export.

Modo	*Descripción*
Toda la base de Datos	Exporta/importa toda la base de datos, incluyendo todos los datos y las definiciones de estructuras y los archivos de soporte que usa la base de datos cuando funciona.
Usuario	Exporta/importa los datos y las correspondientes definiciones de datos de uno o más usuarios.
Tabla	Exporta/importa los datos y las correspondientes definiciones de datos de una o más tablas correspondientes a uno o más usuarios.

Tabla 2.2 Modos de Operación de Export e Import.

Los modos usuario y tabla, se usarán generalmente cuando se extraigan datos de uno o más repositorios operacionales para insértalos dentro del data warehouse. Hay que destacar algunos aspectos de importancia a la hora de usar estos tres modos:

1. El propietario de las tablas en la base de datos origen no tiene por qué ser el mismo que en la base de datos destino. Usando dos parámetros que se detallan en la sección de Import, se pueden transferir datos de un usuario a otro.

2. Las tablas pertenecientes a más de un usuario se pueden especificar cuando se ejecute Import.

3. Cuando se importen tablas y no existan actualmente, hay código en el archivo de Export para crearlas antes de que se inserten datos en ellas.

Hay tres formas de llamar a Export resumidas en la Tabla 2.3. veamos ahora Export e Import y los métodos con los que se ejecutan.

Modo	*Especificación*
1. Diálogo interactivo	Hace que Export entre en un diálogo con el operador preguntándole una serie de cuestiones y recibiendo respuestas.
2. Parámetros en línea de comandos	Llama Export pasándole una serie de parámetros y valores en línea de comandos.
3. Archivo de parámetros	Llama a. Export y le proporciona un valor para la palabra clave parfile=, que nombra a un archivo en el cual se deben leer las palabras clave y sus valores.

Tabla 2.3. Métodos de operación de Export e Import.

2.2.2 Export.

El comando para llamar a Export, en la mayoría de los entornos, es:

```
exp {parametro1=valor1, parametro2=valor2,... parametron=valorn}
```

donde los parámetros y los valores se introducen uno después de otro separándose por comas. Se puede obtener una ayuda en línea introduciendo el siguiente comando y recibiendo la pantalla de ayuda que enumera los parámetros de export:

```
/oracle> exp help=y
```

2.2.3 Parámetros usados en Export.

La Tabla 2.4. muestra los parámetros de uso más habitual en la extracción de datos con destino al data warehouse. La única palabra clave obligatoria es **userid** a la que debe seguir un valor que sea una cuenta válida en Oracle.