Optimizing ETL for Oracle

Jean-Pierre Dijcks
Data Warehouse Product Management
Agenda

• The Three Layers of ETL
• Think about Hardware
• Think about your Access Methods
• Hardware Trends and ETL
• Parallel Loading and Partitioning
• Q&A
The Three Layers of ETL

The ETL Ecosystem

1. Your ETL Tool of Choice
2. The Data Warehouse Database
3. Hardware and Storage
The Three Layers of ETL

The ETL Ecosystem

- OWB or ODI
- Oracle 11g
- Oracle Exadata
Balanced Configuration

“The weakest link” defines the throughput

CPU Quantity and Speed dictate
number of HBAs
capacity of interconnect

HBA Quantity and Speed dictate
number of Disk Controllers
Speed and quantity of switches

Controllers Quantity and Speed dictate
number of Disks
Speed and quantity of switches

Disk Quantity and Speed
Data Warehouse hardware configuration best practices

• Build a balance hardware configuration
  • Total throughput = # cores X 100-200 MB (depends on chip set)
  • Total HBA throughput = Total core throughput
    • If total core throughput = 1.6GB will need 4 4Gb HBAs
  • Use 1 disk controller per HBA Port (throughput capacity must be equal)
  • Switch must be same capacity as HBA and disk controllers
  • Max of 10 physical disks per controller (Use smaller faster drives)

• Minimum of 4GB of Memory per core (8GB if using compression)
• Interconnect bandwidth should equal IO bandwidth (Infiniband)
• Use ASM with RAID 1 mirroring for redundancy
  • Create two ASM diskgroups (1 for data, 1 for flash recovery area)
  • Use ATTRIBUTE 'au_size' to increase allocation unit
Most Bang for the Buck

- Transporting data around can be expensive
- Most performance can be gained when loading into the database
- Focus should therefore be on:
  - Where is data now
  - How to get into a format that allows the fastest load methods
  - Ensuring the database and data sets are set up for this load
- Spend your time wisely and try to achieve the biggest improvements that can be made
- Minimize staging data (writes are expensive)
Bulk Access Strategies

- Own your access strategy
  - Dictate how you want data
  - Set SLAs for your data access provider
- Do not join over database links
  - No parallel access
  - Limited functionality when using ODBC
- Do not accept XML files
  - Or more generically => Keep it Simple
SQL Loader or External Tables

• And the winner is => External Tables

• Why:
  • Full usage of SQL capabilities directly on the data
  • Automatic use of parallel capabilities (just like a table)
  • No need to stage the data once more
  • Better allocation of space when storing data
  • Interesting capabilities like
    • The usage of data pump
    • The usage of pre-processing
Tips for External Tables

• File locations and size
  • Never locate the data files on the same disks as the RDBMS
  • When using multiple files the file size should be similar
  • Use largest to smallest in LOCATION clause if not similar in size

• File Formats
  • Use a format allowing position-able and seek-able scans
  • Delimitate clearly and use well known record termination to allow for automatic Granulation

• Consider compressing data files and uncompressing during loading
Parallel with Compressed Files

- When using compression Oracle will not parallelize the load (not even when you use External Table)
- Use Multiple Compressed Files per ET to get parallel loading
- To Create these multiple files:
  - Either create the archives in equal chunks when the data is generated and compressed
  - Use the pre-processing steps to “dynamically” break down the master archive into smaller files
Tips for External Tables (2)

- New functionality in 11.1.0.7 and 10.2.0.5
- For Example
  - Allows compression of large files
  - Speeding up transport of files to the load location
  - Simplify process by encapsulating “unzip”

```
CREATE TABLE sales_external
 (...)
 ORGANIZATION EXTERNAL
 ( TYPE ORACLE_LOADER
   DEFAULT DIRECTORY data_dir1
   ACCESS PARAMETERS
   (RECORDS DELIMITED BY NEWLINE
     PREPROCESSOR exec_dir: 'gunzip'
     FIELDS TERMINATED BY '|' )
 )
 LOCATION (...)
 )
```
Tips for External Tables (3)

• Has been there since 10.1.x
• For Example
  • Allows utilization of Data Pump as an ETL extraction method
  • Great for unloading from Oracle to Oracle

CREATE TABLE inventories_xt
  ORGANIZATION EXTERNAL
  (
    TYPE ORACLE_DATAPUMP
    DEFAULT DIRECTORY def_dir1
    LOCATION ('inv_xt.dmp')
  )
AS SELECT * FROM inventories where <anything goes>;}
Bulk Data Loading – Example

SQL Server

Oracle Source

Oracle Target (10.2 / 11.1)

BCP Unload

Data Pump Unload

FTP

External Tables

FTP
Bulk Data Loading – Embedded Tools

- BCP Unload
- Data Pump Unload
- FTP

Oracle Target (10.2 / 11.1)
- External Tables

SQL Server
Oracle Source

Load Code Templates
Integration Code Templates

Oracle Warehouse Builder 11g R2
Hardware Trends

• Commodity hardware platforms
  • Intel Chips
  • 64 bit Linux OS

• Clustered environments
  • Increasing CPU counts
  • Increasing memory sizes available

• Larger systems
  • A lot more data
  • Compute power you didn’t think you could have
Hardware Trends and ETL

• Fact: External table is the best way to load the DW
• Fact: If you run Linux, you probably run Oracle RAC

• Oops: You cannot run a cluster-parallelized SQL statement on an external table unless the files are on shared storage
• Oops: A simple single NFS filer will bottleneck your solution in ETL (not enough throughput)
Introducing DBFS (Database Machine)

- Database File System

- DBFS is a file system interface for storing “unstructured data” in Oracle Database
  - Built on SecureFiles
  - It is cluster-coherent and scalable with Real Application Clusters

- FUSE
  - Filesystem in Userspace (http://fuse.sourceforge.net/)

- Combining DBFS with FUSE offers mountable filesystems for Linux x64 (e.g. Database Machine!)
Access Strategies – Example

SQL Server
BCP Unload

Oracle Source
Data Pump Unload

Oracle Target (11.2)

External Tables

FTP
SCP

DBFS in its own instance holds files in SecureFiles
Direct Path Load

• Data is written directly to the database storage using multiple blocks per I/O request using asynchronous writes.

• A CTAS command always uses direct path but an IAS needs an APPEND hint:

```
Insert /*+ APPEND */ into Sales partition(p2)
Select * From ext_tab_for_sales_data;
```

• Ensure you do direct path loads in parallel:
  • Specify parallel degree either with hint or on both tables.
  • Enable parallel DML by issuing alter session command:
```
ALTER SESSION ENABLE PARALLEL DML;
```
1. Create external table for flat files

2. Use CTAS command to create non-partitioned table TMP_SALES

3. Create indexes

4. Gather Statistics

5. Alter table Sales exchange partition May_24_2008 with table tmp_sales

6. Sales table now has all the data
Database Settings for Data Loading

- Use Parallel loading
  - Create the table with PARALLEL on it
  - Hint with PARALLEL
  - Don’t forget “Alter session enable parallel DML”

- Use these settings:
  - PARALLEL_EXECUTION_MESSAGE_SIZE=16KB
  - PARALLEL_MIN_SERVERS= 2*Default_DOP
  - PARALLEL_MAX_SERVERS= Max_conc_queries * Default_DOP

- DB_BLOCK_SIZE = 8KB / 16KB