Data Pump: Not Just for Data Moves

Arup Nanda

Starwood Hotels

Data Pump

- “Export/Import on Steroids”
- On the server
- Uses the directory object to create and read dump files
- Not compatible with Original Export/Import

But it’s not *just* that!

- Data Pump has other powerful functionality to help your efforts – both long term and every day
- Your friend in strategic and tactical planning
- These benefits are often not highlighted in normal sources of information – presentations, articles, books, training classes, word of mouth, etc.
- This session helps you to uncover those gems in the dark

Regulatory Compliance

- Most regulations require
  - Repository of all source code – including the stored code - baseline
  - Repository of all metadata - baseline
  - Tracking of changes to source code
  - Version control
- There are specialized tools
  - And you can build them in-house

Metadata Management

- The CONTENT parameter controls what is exported:

  `ALL | DATA_ONLY | METADATA_ONLY`

- The INCLUDE parameter controls what objects are included:

  INCLUDE = *object_type*[:*name*] [, ...]

  or

  INCLUDE = *object_type*[:*name*]

  INCLUDE = *object_type*[:*name*] [, ...]

Example

- First, take a baseline of all procedures:

```
expdp directory=dump_dir dumpfile=md.dmp
  include=PROCEDURE SCHEMAS=ARUP
```

- Only a specific procedure:

```
expdp directory=dump_dir dumpfile=md1.dmp
  include=PROCEDURE:\"=\'PROC1\'\" SCHEMAS=ARUP
```

- Multiple object types:

```
expdp directory=dump_dir dumpfile=md3.dmp
  schemas=ARUP
  include=PROCEDURE:\"=\'PROC1\'\",FUNCTION:\"=\'
  PROC1\'\"
```

Show Metadata

- Create a SQL File:

```
impdp directory=dump_dir
  dumpfile=md3.dmp sqlfile=a.sql
```

- A file called a.sql is created with all the object creation DDL statements.
- You can filter too:
  - `INCLUDE=PROCEDURE, PACKAGE`
  - `EXCLUDE=PROCEDURE:"='PROC1'"`

Metadata Information

```
-- CONNECT SYS
-- new object type path is:
   SCHEMA_EXPORT/PROCEDURE/PROCEDURE
-- CONNECT ARUP
CREATE PROCEDURE "ARUP"."PROC1"
as
begin
  dbms_output.put_line ('Some text');
end;
/
-- new object type path is:
   SCHEMA_EXPORT/PROCEDURE/ALTER_PROCEDURE
ALTER PROCEDURE "ARUP"."PROC1"
  COMPILE
    PLSQL_OPTIMIZE_LEVEL=  2
    PLSQL_CODE_TYPE=  INTERPRETED
    PLSQL_DEBUG=  FALSE
 REUSE SETTINGS TIMESTAMP '2006-08-11 13:27:55'
```

Building a Repository

- Required for regulatory compliance
- Options:
  - Dump: `expdp directory=dump_dir dumpfile=md`*`mmddyy`*`.dmp`
  - SQL File: `impdp directory=dump_dir dumpfile=md`*`mmddyy`*`.dmp sqlfile=md`*`mmddyy`*`.sql`
- Move periodically

```
find . -name "*.dmp" -ctime +30 -exec mv
  {} {}.old\;
```

Create a User Like …

- Problem:
  - Quickly create a user like another, with all its grants, system privs, ts quotas, etc.
- Old Solution:
  - Painstakingly get the information from the data dictionary and construct a SQL file
- Data Pump Solution:
  - `expdp schemas=arup content=metadata_only`
  - `impdp remap_schema=ARUP:NEWUSER`
  - It creates the new user

Create Tablespaces

- Problem:
  - You want to create the same tablespaces in test database as in production
  - “backup controlfile to trace” will not work
  - Only option: RMAN Cloning
- Data Pump Solution:
  - From the full dump, extract the tablespaces:

    ```
    include=TABLESPACE
    ```

Smaller Datafiles

- Problem:
  - The test database is smaller
- Data Pump Solution
  - Simply use the parameter `transform=pctspace:10`
- Before:

```
CREATE UNDO TABLESPACE "UNDOTBS1" DATAFILE
  '/u01/undotbs101.dbf' SIZE 17179869184,
```

- After:

```
CREATE UNDO TABLESPACE "UNDOTBS1" DATAFILE
  '/u01/undotbs101.dbf' SIZE 1717986918,
```

Data File Name Change

- Problem:
  - You are moving some tables from a database to another
  - The file structures are different
- Old Solution:
  - Examine the old file structures
  - Create the tablespace with the new files in the target database
  - Grant quota on the new tablespace to the user
  - Pre-create the tables on the target database
  - Import data

Data File Name Change

- Data Pump Solution: One Step

```
DIRECTORY=tmp_dir FULL=Y
  DUMPFILE=db_full.dmp
  REMAP_DATAFILE=‘/u01/data1.dbf':'/u02/d
  ata1.dbf'
```

- Very useful in creating data on a different system

```
REMAP_DATAFILE='/u01/data1.dbf':'C:\orada
  ta\data1.dbf'
```

Create Prod Objects

- Problem:
  - You want to replicate all the production objects in the test database
  - The only option: RMAN Clone
- Data Pump Solution:
  - `EXCLUDE=TABLE,VIEW`
  - Includes all objects other than tables and view
  - Has tablespace, sequences, roles, profiles, [public] synonyms, MVs, Streams …

Segment Transforms

- Problem:
  - Initial extent too large
  - Tablespace does not exist
- Old Solution:
  - Drop table; index file option to create SQL file; modify SQL; create table; import data
- Data Pump Solution:

```
impdp tables=test dumpfile=a
  directory=tmp_dir
  transform=segment_attributes:n:table
```

Example

- Example
  - `impdp tables=test dumpfile=a directory=tmp_dir sqlfile=a.sql`
  - `impdp tables=test dumpfile=a directory=tmp_dir sqlfile=a.sql transform=segment_attributes:n:table`
- Apply this to all objects, not just tables:

  `transform=segment_attributes:n`
- This parameter removes
  - physical attributes
  - storage attributes
  - tablespaces
  - logging

Reducing Size

- Reduces the original initial extent of tables
  - PCTSPACE:n – reduces the initial extent by n%
  - SAMPLE=s – samples the data by s%
- Example:

  -
    ```
    expdp DIRECTORY=tmp_dir
    DUMPFILE=a.dmp SAMPLE=10
    TRANSFORM=PCTSPACE:30
    ```

Sub-setting a Table

- Create a table of n% of the production data for testing purpose in QA
- Options
  - Export and then Import
  - Import from previous Export

Export/Import

- Randomly 10% of table ARUP.TEST
  - `$ expdp SAMPLE=ARUP.TEST:10`
  - `$ expdp SAMPLE=10`
- Specific rows
  - `$ expdp QUERY=“WHERE COL1>100"`
  - Can also use ORDER BY

  -
    ```
    expdp arup/arup directory=demo_dir
    dumpfile=employees.dmp
    query=employees:\"where salary\>10000\
    order by salary" tables=employees
    ```

Import from Full Dump

- `$ impdp QUERY=CUSTOMERS:“WHERE TOTAL_SPENT > 10"`
- Can also use ORDER BY
- Can be used to quickly populate QA databases
- Does not take care of referential integrity constraints
- So, use when you can select as a part of a set, i.e. specific values

Refresh a Table Definition

- Problem:
  - A table in QA has gone out of sync with PROD. Need to refresh the table definition very quickly.
- Old Solution:
  - Painstakingly build the SQL from data dictionary
  - Make sure captured all the grants, triggers, constraints, etc.
- Data Pump Solution
  - `$ expdp tables=TAB1 content=metadata_only`
  - `$ impdp full=y table_exists_action=replace`
  - Can also be used for refreshing from a repository

Changing Table’s Owner

- Problem:
  - You have created a table on a wrong schema
- Old Solution:
  - You can’t change the owner of a table
  - Create the SQL to create table in the new schema, including all grants, triggers, constraints and so on …
  - Export the table
  - Drop the table in old schema
  - Import the table into the new schema
- Data Pump Solution: one line:
  - `$ impdp remap_schema=“OLDU:NEWU” network_link=maindb directory=…`

External Tables

- External tables are text files outside the database, but are visible to the database as tables
- Can be queried, but not changed
- Data Pump can create external tables
  - Not ASCII text, binary
  - Portable across operating systems

Example

```
create table trans_ext (
   trans_id,
   trans_dt,
   product_code,
   store_id,
   trans_amount
)
organization external
(
   type oracle_datapump
   default directory tmp_dir
   location ('trans_ext.dmp')
)
as
select * from trans
order by trans_id;
```

External file created

```
create table trans_external (
   trans_id     number,
   trans_dt     date,
   product_code number,
   store_id     number,
   trans_amount number(12,2)
)
organization external
(
   type oracle_datapump
   default directory tmp_dir
   location ('trans_ext.dmp')
)
```

```
select *
from trans_external
```

Uses of External Tables

- Portable -> any platform
- Offline -> Receiver need not be online, good for publishing data
- Creation -> No coding needed
- Order and Group -> IOTs, ETL
- Faster Loading ->

```
INSERT /*+ APPEND */ INTO TRANS
SELECT * FROM TRANS_EXTERNAL;
```

Creating IOTs

- Index Organized Table is a table built on a primary key index, so that a query by PK will get all the table data from the index itself without a trip to the table.
- Sorts in IOTs take longer if they are in a random order.
- You can create the IOT in a pre-sorted manner to reduce the sort time.

General Tips 'n Tricks

- Parallelizing
  - When you run in parallel, make sure you have that many files as parallel degree

  -
    ```
    expdp ananda/abc123 tables=CASES
    directory=DPDATA1
    dumpfile=expCASES_%U.dmp
    parallel=4 job_name=Cases_Export
    ```
  - Files created as `expCASES_01,02`,etc.
  - http://www.oracle.com/technology/products/database/utilities/pdf/parallel_cap_datapump.pdf

Monitoring

- Sessions:

```
select sid, serial#
from v$session s, dba_datapump_sessions d
where s.saddr = d.saddr;
```

- PQ Sessions:

```
select sid from v$px_session
where qcsid = 23;
```

- Long Sessions:

```
select sid, serial#, sofar, totalwork
from v$session_longops
where opname = 'CASES_EXPORT'
and sofar != totalwork;
```

Troubleshooting - TRACE

- Command line parameter

  `TRACE=<CompID>0300`

- CompID is the component to trace
  - `1FF` – Full Tracing
  - `048` – Standard Tracing

```
$ impdp u/w trace=0480300 schema=…
```

SQL Trace

- Get the SID, Serial#

```
select sid,serial#,username,program
from v$session
where upper(program) like '%(DW%)'
or upper(program) like '%(DM%)';
```

- Then Trace the session

```
dbms_system.set_ev(<SID>,
  <Serial#>,10046,12,'')
```

In Conclusion

- Data Pump is *not* just a tool to move data; it has powerful functionalities beyond data movement, such as
  - Metadata repository – regulatory compliance
  - Version Control – regulatory compliance and convenience
  - Building a smaller sized object – quick refreshes
  - Cloning users
  - Changing table owners
  - Changing data files on the fly
  - Building IOTs
  - Publishing offline information to heterogeneous sources
  - And much more … limited by imagination!

Thank you!

Questions?