What’s New with Oracle Data Pump in Oracle Database 12c

Dean Gagne
Consulting Member of Technical Staff
Program Agenda

- Full Transportable Export/Import
- Exporting Views as Tables
- Detailed Timestamps for Data Pump Operations
- Improved Security
- Enhanced Compression Options
- Additional TRANSFORM Options
Full Transportable Export/Import
Transportable Tablespaces

Background

- Available starting with Oracle Database 8i
  - Can go cross-platform starting with 10g
- Simple, elegant concept
  - Create “empty” database on destination
  - Copy tablespace files from source, plug into target
  - Can be very fast if not going cross-endian
- Can quickly get very complex
  - Movement of metadata is often the biggest issue
Transportable Tablespaces
Steps for a Transportable Tablespace Migration

1. Network mode import to get user metadata for TTS import (e.g. user definitions for table owners)
2. Full non-TTS metadata-only export on source to capture definitions of all non-TTS objects (packages, views, triggers, user-defined objects, etc.)
3. TTS export on source to capture tablespace object definitions
4. Copy tablespaces to target
5. TTS import on target to create user tablespaces and TTS objects
6. Non-TTS import from metadata-only export to create non-TTS objects
7. Other actions to get user data stored in SYSTEM/SYSAUX, etc.
Full Transportable Export/Import

What it is

- Combines usability of Data Pump with speed of Transportable Tablespaces
- Full Transportable Export already exists in 11.2.0.3!
  - Initiated by `expdp FULL=Y TRANSPORTABLE=ALWAYS VERSION=12`
  - No need for `VERSION=12` when exporting from 12.1 or later releases
- Full Transportable Import starting with 12.1.0.1
- Cross-platform, cross-endian (with `RMAN CONVERT` of data files)
- Works with dump files or over a `NETWORK_LINK`
Full Transportable Export/Import

How it works

- Distinction between *administrative* and *user* tablespaces
  - Administrative tablespaces are SYSTEM, SYSAUX, etc.
  - Objects with storage in user tablespaces move transportably
  - Objects with storage in administrative tablespaces move conventionally

- Supports transport of
  - Encrypted tablespaces and tables with encrypted columns
  - Timestamp-with-timezone data (TSTZ) across different timezone versions

- Oracle-supplied components such as MultiMedia, Auditing and Spatial register their metadata for export using internal APIs and callbacks
Full Transportable Export/Import

SOURCE Database 11.2.0.3
- INC BCK1
- INC BCK0
- USERS
- SYSTEM
- SYSAUX
- UNDO
- TEMP

DESTINATION Database 12.1.0.1
- SYSTEM
- SYSAUX
- UNDO
- TEMP

Data Pump

Convert and apply backups
Example: Full Transportable Network Import

11.2.0.3 ➔ 12c PDB
Example: Migrate 11.2.0.3 ➔ 12c PDB

Scenario

- Source: 11.2.0.3 database instance (SID=V112) on Windows x86
- Target: 12c PDB (SID=PDB12) on Oracle Exadata (OEL 6)

<table>
<thead>
<tr>
<th>Tablespace Name</th>
<th>Encrypted?</th>
<th>Datafile Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>NO</td>
<td>/data1/oracle/dbs/t_db1.f</td>
</tr>
<tr>
<td>SYSAUX</td>
<td>NO</td>
<td>/data1/oracle/dbs/t_ax1.f</td>
</tr>
<tr>
<td>UNDO</td>
<td>NO</td>
<td>/data2/oracle/dbs/t_undo1.f</td>
</tr>
<tr>
<td>HR</td>
<td>YES</td>
<td>/data3/oracle/dbs/hr_1.f</td>
</tr>
<tr>
<td>ENGTABLES</td>
<td>NO</td>
<td>/data4/oracle/dbs/eng_1.f</td>
</tr>
<tr>
<td>ENGINDEXS</td>
<td>NO</td>
<td>/data5/oracle/dbs/eng_2.f</td>
</tr>
</tbody>
</table>
Example: Migrate 11.2.0.3 ➔ 12c PDB

Three Phases of Minimal Downtime Migration

1. Pre-transport datafile migration using incrementally-updated backups
   - Source database remains in production
2. Transport using Data Pump import over NETWORK LINK: Source unavailable
   - Downtime occurs here
3. Post-transport cleanup
   - Target database now in production
Example: Migrate 11.2.0.3 ➔ 12c PDB

Phase 1: Incrementally Updated Backup

1. Enable V112 for archive logging:
   ALTER DATABASE ARCHIVELOG;

2. Enable block change tracking:
   ALTER DATABASE ENABLE BLOCK CHANGE TRACKING USING FILE '<name>' REUSE;

3. Recovery area should be accessible by target instance
Example: Migrate 11.2.0.3 ➔ 12c PDB

Phase 1: Incrementally Updated Backup

4. Simple RMAN script:
   ```sql
   BACKUP INCREMENTAL LEVEL 1 FOR RECOVER OF COPY WITH TAG 'oow' DATABASE;
   RECOVER COPY OF DATABASE WITH TAG 'oow';
   ```

5. First execution creates level 0 copy of all datafiles
   - Source remains available for production

6. Subsequent invocations backup and apply to the datafiles only changed blocks since previous invocation
   - Block change tracking makes it fast
Example: Migrate 11.2.0.3 ➔ 12c PDB

Phase 1: Incrementally Updated Backup

7. Create the target CDB with target PDB
   Use DBCA to create the CDB
   CREATE PLUGGABLE DATABASE pdb12 ADMIN USER oow IDENTIFIED BY <password>;

8. Login to PDB then create database link back to 11.2.0.3 instance
   CREATE PUBLIC DATABASE LINK v112 USING 'v112';
   User who runs impdp on target must exist on source with DATAPUMP_EXP_FULL_DATABASE role.

9. In PDB12, create a directory object for the log file:
   CREATE DIRECTORY oow_dir AS '/recovery1/data';
   GRANT READ, WRITE ON DIRECTORY oow_dir TO oow;
Example: Migrate 11.2.0.3 → 12c PDB
Phase 2: Prepare and Transport the Tablespace Data Files

1. Set the transportable tablespaces read-only on V112:
   ALTER TABLESPACE hr READ ONLY;
   ALTER TABLESPACE engtables READ ONLY;
   ALTER TABLESPACE engindexes READ ONLY;

2. Execute the RMAN incremental backup and apply script one last time
   - The backup copies are now consistent
   - Assume backup copies are located in /recovery1/data
3. Run `impdp` on target as a user with `DATAPUMP_IMP_FULL_DATABASE` role

```bash
impdp oow/<passwd>@pdb12   NETWORK_LINK=v112
VERSION=12   FULL=Y   TRANSPORTABLE=ALWAYS
ENCRIPTION_PASSWORD=<enc_passwd>
METRICS=Y   LOGFILE=oow_dir:v112fullimp.log
TRANSPORT_DATAFILES='/recovery1/data/hr_1.f'
TRANSPORT_DATAFILES='/recovery1/data/eng_1.f'
TRANSPORT_DATAFILES='/recovery1/data/eng_2.f'
```
Example: Migrate 11.2.0.3 ➔ 12c PDB

Phase 3: Validate and Clean Up

- Carefully review import log and resolve any errors
- Transported tablespaces on target are read / write at end of import
- Validate PDB contents against source
- Set source tablespaces back to read / write … or not.
- Done!
Exporting and Importing Views as Tables
Exporting Views as Tables

VIEWS_AS_TABLES=[schema_name.]view_name[:table_name], ...

- Specifies that one or more *views* are to be exported *as tables*.
  - Exports a table with the same columns as view
  - Row data is fetched from the view
  - Dependent objects also exported, such as grants and constraints
- Improves performance of `QUERY` over `dblink`
- `TABLE_NAME`: Template table serves as source of metadata
  - Required if the database is read-only
Exporting Views as Tables

Example: export

```sql
SQL> create view hr.my_view (first, last, dept) as
    select e.first_name, e.last_name,
            d.department_name
    from employees e, departments d
    where e.department_id = d.department_id;

SQL> exit

>$ expdp system/manager views_as_tables=hr.my_view ...
Processing object type TABLE_EXPORT/VIEWS_AS_TABLES/TABLE
  . . exported "HR"."MY_VIEW" 8.570 KB 106 rows
```
Exporting Views as Tables

Example: import

```sql
impdp system/manager remap_table=my_view:my_table ...
Processing object type TABLE_EXPORT/VIEWS_AS_TABLES/TABLE_DATA
. . imported "HR"."MY_TABLE" 8.570 KB 106 rows
```

- Without the `remap_table` parameter, Data Pump creates a table called `my_view`
- `remap_table:source_name:target_name` remaps the name and creates a table with the target name
Detailed Timestamps for Data Pump Operations
Detailed Timestamps with LOGTIME

LOGTIME=[ NONE | STATUS | LOGFILE | ALL ]

- Messages displayed during Data Pump operations can include timestamps
- Information can be helpful in diagnosing performance problems and estimating the timing of future similar operations.
- Available options are defined as follows:
  - **NONE**: (default) No timestamps on status or log file messages
  - **STATUS**: Timestamps on status messages only
  - **LOGFILE**: Timestamps on log file messages only
  - **ALL**: Timestamps on both status and log file messages
Detailed Timestamps with LOGTIME

Example

> expdp hr DIRECTORY=dpump_dir1 DUMPFILE=expdat.dmp SCHEMAS=hr
   LOGTIME=ALL

10-JUL-12 10:12:22.300: Starting "HR"."SYS_EXPORT_SCHEMA..."

10-JUL-12 10:12:22.915: Estimate in progress using BLOCKS method...

10-JUL-12 10:12:24.422: Processing object type SCHEMA_EXPORT...

10-JUL-12 10:12:24.498: Total estimation using BLOCKS method: ...

10-JUL-12 10:13:06.824: .. exported "HR"."REGIONS" 5.515 KB 4 rows...

10-JUL-12 10:13:07.500: Master table "HR"."SYS_EXPORT ...
Improved Security
No Echo of Encryption Password
ENCRIPTION_PWD_PROMPT = [Y | N]

- Option to indicate whether the Oracle Data Pump client should prompt for passwords
  - Y: prompt for password
  - N: (default) retrieve the value from the command line.

- Improves security
  - Password no longer exposed to OS commands (e.g. ps)
  - Remove plan text passwords from scripts
No Echo of Encryption Password

Example

```bash
>$ impdp hr ENCRYPTION_PWD_PROMPT=YES dumpfile=hr.dmp ...

Password: <user password entered here>

Connected to: Oracle Database 12c Enterprise Edition Release 12.1.0.1.0 - 64bit Production

With the Partitioning, Advanced Analytics and Real Application Testing options

Encryption Password: <encryption password here>
```
Auditing for Data Pump Commands

- Oracle Data Pump commands can now be audited
- Syntax
  
  ```sql
  CREATE AUDIT POLICY policy_name
  ACTIONS COMPONENT=DATAPUMP { EXPORT | IMPORT | ALL };
  ```

- See [Oracle Database Security Guide](#) for details
Enhanced Compression Options
Enhanced Compression Options

**COMPRESSION_ALGORITHM=[ BASIC | LOW | MEDIUM | HIGH ]**

- Trade off performance against the dump file size
- **BASIC**: (default) - good combination of compression ratio and speed
  - same algorithm as in previous versions of Oracle Data Pump.
- **LOW**: Best for CPU-limited environments
- **MEDIUM**: Similar to BASIC, good compression ratios and speed
  - Different algorithm than BASIC
  - Recommended for most environments
- **HIGH**: Best for disk- or network-limited environments
Enhanced Compression Options

Example

```bash
$ expdp hr DIRECTORY=dpump_dir1 DUMPFILE=hr.dmp
COMPRESSION=DATA_ONLY COMPRESSION_ALGORITHM=LOW

- Compresses only the data, leaving metadata uncompressed
- Fewer CPU resources used, at the expense of a less than optimal compression ratio

$ expdp hr DIRECTORY=dpump_dir1 DUMPFILE=hr.dmp COMPRESSION=ALL
COMPRESSION_ALGORITHM=BASIC

$ expdp hr DIRECTORY=dpump_dir1 DUMPFILE=hr.dmp COMPRESSION=ALL

- Compresses both metadata and table data using the basic level of compression
- Omitting the COMPRESSION_ALGORITHM parameter altogether is equivalent to specifying BASIC as the value
```
Additional TRANSFORM Options
Background: TRANSFORM Options for 11.2
Alter object creation DDL during import

- **SEGMENT_ATTRIBUTES**: change physical storage, tablespace, logging
- **STORAGE**: keep or ignore storage clauses
- **OID**: create new OIDs on import
- **PCTSPACE**: alter extent allocations
- **SEGMENT_CREATION**: deferred or immediate

Oracle Database 12c includes even more options for additional control!
Change Table Compression at Import Time

TRANSFORM = TABLE_COMPRESSION_CLAUSE: [NONE | <compression_clause>]

- Specify table compression on import, regardless of options used during export
  - Affects all tables created by the import job
- Syntax:
  - (default): no transform specified, compression clause from dumpfile
  - NONE: Table compression clause from dumpfile is omitted, table is created using default for tablespace
  - <compression_clause>: any valid table compression clause
    - NOCOMPRESS, “COMPRESS BASIC”, and so on
No Logging Options for Import

TRANSFORM = DISABLE_ARCHIVE_LOGGING:[Y|N]:[object_type]

- Disable redo logging when loading tables and/or creating indexes
- Benefits:
  - Reduced disk space for redo logs during import
  - Reduced maintenance of redo logs by DBA
- Applies to both TABLES and INDEXES if not object type specified
- Logging attributes restored to original settings after data is loaded
No Logging Options for Import

Notes

- To ensure recovery from media failure, DBA should perform an RMAN backup after the import completes.
- Redo logging for other operations of Oracle Data Pump still happens.
  - CREATE and ALTER statements, except CREATE INDEX
  - All operations against the master table during the import.
- Valid for both file mode imports and network mode imports.
  - Not valid for transportable tablespace imports.
- Note: If the database is in FORCE LOGGING mode, then DISABLE_ARCHIVE_LOGGING option will not disable any logging
Conclusion

- Information and Resources
  - OTN Page: otn.oracle.com/goto/datapump
  - White paper: Oracle Database 12c: Full Transportable Export/Import
  - Oracle Database Utilities Guide

THANK YOU!!!