ORACLE

Oracle Data Pump Best Practices

Oracle Database Utilities

August 2021 | Version 2.00

PURPOSE STATEMENT

This document provides recommended best practices for Oracle Data Pump for migrating Oracle Database from on-premises to on-premises and to Oracle Cloud. It is intended solely to help you assess the benefits of upgrading to Oracle Database and plan your I.T. projects.

DISCLAIMER

Table of Contents

INTRODUCTION

There are multiple approaches for migrating on-premises Oracle databases. A common method is Oracle Data Pump, a feature of Oracle Database since release 10g and successor to the Oracle Export and Import (exp/imp) utilities in release 9i and earlier. Oracle Data Pump is useful for migrating data among schemas, databases of different versions and on different operating systems, and from on-premises to on-premises and to Oracle Cloud. It should be considered for migrating an on-premises Oracle database to Oracle cloud in the following circumstances:

- the source Oracle Database is release 10g or higher; (Older databases must use the original Export and Import utilities that came with the database.)
- migrating data cross-endian;
- migrating from a non-CDB Oracle database to an Oracle multitenant database;
- migrating with changes to database structure; or,
- combining a migration and a version upgrade.

Migrating data using Oracle Data Pump is a three-step process. Assuming your cloud instance and PDB are created:

- export the on-premise database using expdp,
- copy the dump files to the target system or to the Oracle Object Store, if required
- import into the cloud PDB using impdb.

This white paper describes some best practices for using Oracle Data Pump to help make the process go smoothly and successfully.

DO NOT INVOKE EXPORT USING SYS as SYSDBA

SYSDBA is used internally and has specialized functions; it doesn't behave like generalized users.

You should not typically need to start Export as SYSDBA except at the request of Oracle technical support or when importing a transportable tablespace set.

USE A PARAMETER FILE

A parameter file, also referred to as a "parfile", enables you to specify command-line parameters in a file for ease of reuse. It also helps avoid typographical errors from typing long ad hoc Data Pump commands on the command line, especially if you use parameters whose values require quotation marks.

Here is an example of a parfile:

```
DIRECTORY=my_data_pump_dir
DUMPFILE=dumpfile.dmp
LOGFILE=logfile.log
SCHEMAS=HR
EXCLUDE=STATISTICS
LOGTIME=ALL
METRIC=YES
FLASHBACK_TIME=SYSTIMESTAMP
```

The command to execute the par file looks like this:

```
expdp parfile=my_data_pump_parfile.par
```

MAKE A CONSISTENT DATA PUMP EXPORT

By default, Oracle Data Pump preserves consistency within a single database table. For example, if you export a table that has 1000 partitions, the exported table will be consistent as of the specific System Change Number (SCN) at which you started the export. When exporting multiple tables, the next table exported would then be consistent as of a different SCN. For any export of more than one table, you will probably want your export dump file to represent all of the objects as of the same SCN.

This can be accomplished by using either `FLASHBACK_SCN=<scn>` or `FLASHBACK_TIME=<timestamp>` to enable the Flashback Query utility. A particularly convenient approach is to specify FLASHBACK_TIME=SYSTIMESTAMP.

Using `FLASHBACK_SCN`, the export operation is performed with data that is consistent up to the specified SCN. For example, this command assumes that an existing SCN value of 384632 exists. It exports the hr schema up to SCN 384632:

```
expdp hr DIRECTORY=dpump_dir1 DUMPFILE=hr_scn.dmp FLASHBACK_SCN=384632
```

Using FLASHBACK_TIME=<timestamp>, the export operation is performed with data that is consistent up to the SCN that most closely matches the specified time. For example, this export operation is performed with data that is consistent up to the SCN closest to the specified time.

```
FLASHBACK_TIME="TO_TIMESTAMP('27-10-2012 13:16:00', 'DD-MM-YYYY HH24:MI:SS')"
```

When specifying `FLASHBACK_TIME=SYSTIMESTAMP`, the timestamp will be that of the current system time. Finally, you can still use the release 11.2 legacy interface, `CONSISTENT=Y` which is translated directly to `FLASHBACK_TIME=SYSTIMESTAMP`.

NOTE: If you use FLASHBACK_TIME or FLASHBACK_SCN, Data Pump Export must retain UNDO records for the duration of the export. If the database has insufficient UNDO retention, the result will be a "snapshot segment too old" error as Data Pump attempts to access UNDO records that are no longer available to the database.

NOTE: You can also turn an Oracle Data Guard standby database into a temporary snapshot to do an export. Data Pump requires creation of tables in order to coordinate the export. That requires a read-write instance. Currently, it is not possible with Active Data Guard and DML Redirect because it allows for DML only - not DDL for the CREATE TABLE command.

EXCLUDE STATISTICS FROM EXPORT AND IMPORT

Oracle recommends that you do not export statistics during export. This will provide better performance on both export and import, even accounting for the need to gather statistics after the import. You can exclude statistics from an export operation using the `EXCLUDE=STATISTICS` parameter or `EXCLUDE=TABLE_STATISTICS,INDEX_STATISTICS` for Transportable Tablespaces. Instead, follow the best practice of either creating fresh statistics on the target database or using a `DBMS_STATS` staging table for transporting statistics.

INCLUDE DIAGNOSTIC PARAMETERS DURING EXPORT AND IMPORT

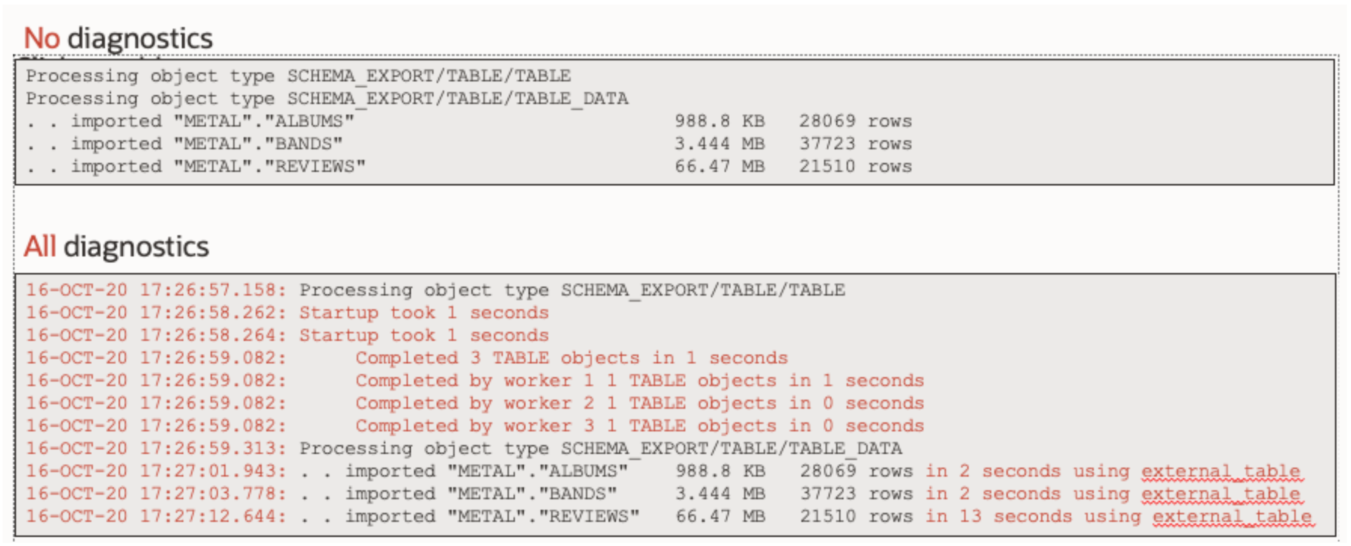
Timestamp the messages that are displayed during an export operation using `LOGTIME=ALL`. This parameter is available beginning with Oracle Database release 12.1. Having timestamps on every line in the logfile helps when assessing export and import performance.

Record the number of objects and the elapsed time about the job in the Oracle Data Pump log file. Accomplish this using the parameter `METRICS=YES`. This gives an extra level of detail, such as the work performed by each process in a `PARALLEL` export or import.

Timestamp the messages that are displayed during an import operation using `LOGTIME=ALL`. This parameter is available beginning with Oracle Database release 12.1.

Record the number of objects and the elapsed time about the job in the Oracle Data Pump log file using the parameter `METRICS=YES`.

Example:



No diagnostics

```
Processing object type SCHEMA_EXPORT/TABLE/TABLE
Processing object type SCHEMA_EXPORT/TABLE/TABLE_DATA
. . imported "METAL"."ALBUMS"                           988.8 KB   28069 rows
. . imported "METAL"."BANDS"                            3.444 MB   37723 rows
. . imported "METAL"."REVIEWS"                          66.47 MB   21510 rows
```

All diagnostics

```
16-OCT-20 17:26:57.158: Processing object type SCHEMA_EXPORT/TABLE/TABLE
16-OCT-20 17:26:58.262: Startup took 1 seconds
16-OCT-20 17:26:58.264: Startup took 1 seconds
16-OCT-20 17:26:59.082:      Completed 3 TABLE objects in 1 seconds
16-OCT-20 17:26:59.082:      Completed by worker 1 1 TABLE objects in 1 seconds
16-OCT-20 17:26:59.082:      Completed by worker 2 1 TABLE objects in 0 seconds
16-OCT-20 17:26:59.082:      Completed by worker 3 1 TABLE objects in 0 seconds
16-OCT-20 17:26:59.313: Processing object type SCHEMA_EXPORT/TABLE/TABLE_DATA
16-OCT-20 17:27:01.943: . . imported "METAL"."ALBUMS"    988.8 KB   28069 rows in 2 seconds using external_table
16-OCT-20 17:27:03.778: . . imported "METAL"."BANDS"     3.444 MB   37723 rows in 2 seconds using external_table
16-OCT-20 17:27:12.644: . . imported "METAL"."REVIEWS"   66.47 MB   21510 rows in 13 seconds using external_table
```

IMPROVE PERFORMANCE WITH PARALLELISM AND CURRENT STATISTICS

USE PARALLELISM

Accomplish more work in less time using parallelism. A new Data Pump Job consists of at least two background processes: a Control process and a Worker process, and 2 sessions. The Data Pump `PARALLEL` parameter creates additional background processes and sessions during an export or import.

The `PARALLEL=n` parameter specifies the maximum number of processes of active execution operating on behalf of the export or import job. Typically, the value for n should be twice the number of CPU cores but be prepared to adjust it if need be.

Oracle Data Pump has imported package bodies in parallel by the `PARALLEL` parameter for several releases. With release 12.2 and higher, for export and import by dumpfile only, Oracle Data Pump imports most metadata and most database objects in parallel. The `PARALLEL` parameter also determines how many indexes get created in parallel. Database metadata objects that have dependencies are still imported in a serial fashion, such as types (due to inheritance), schemas and procedural actions.

If you are using Oracle Database release 11.2.0.4 or 12.1.0.2, you can apply the patch for bug 22273229 to enable parallel import of constraints and indexes.

Note: Data Pump uses two kinds of parallelism. One worker per partition or table is used for sub-partitions and small tables (Inter-table parallelism.) One parallel query PX process (formerly called a PQ slave) is used for each large partition or a large unpartitioned table (Intra-table parallelism.)

SPECIFY WILDCARDS TO CREATE MULTIPLE DUMPFILES

When using parallelism, use the `%U` or `%L` substitution variable when specifying the dumpfile name. This enables simultaneous parallel writes to multiple logfiles. Also, multiple dump files of a smaller size can be easier to managc and copy.

For example:

For 1 to 99 files use `%U` (expands the file names into a 2-digit, fixed-width, incrementing integer)

```
        dumpfile=dumpfile%U.dmp
        filesize=n
```

For 1 to greater than 99 files use `%L` (expands the file names into a 3-digit to 10-digit, variable-width integer)

```
        dumpfile=dumpfile%L.dmp
        filesize=n
```

Otherwise you end up with parallel writes to a single logfile, which can impact performance.

GATHER ACCURATE STATISTICS BEFORE AND AFTER DATA PUMP OPERATIONS

It can be helpful having accurate statistics prior to an export operation because this helps ensure the best possible export performance. It is also helpful to gather statistics after import. The frequency of statistics updates that is needed to keep statistics current depends on how volatile the database is. Statistics gathering includes both dictionary statistics and object statistics. Dictionary statistics are used when data pump filters, orders and collects metadata objects at various stages of the export. Object statistics are used to estimate table and index sizes, which helps produce optimal ordering and parallelism.

Concurrent with metadata export, table sizes are estimated and ranked from largest to smallest for parallel export. The table sizes are estimated using statistics. You can collect statistics using the `dbms_stats` package, with the `gather_table_stats`, `gather_schema_stats`, or `gather_database_stats` procedure.

We recommend using `gather_schema_stats` because it will gather stats on all objects regardless of the staleness information.

Exmple:

```
SQL> BEGIN

        DBMS_STATS.GATHER_SCHEMA_STATS('SYS');

        DBMS_STATS.GATHER_SCHEMA_STATS('SYSTEM');

     END;

$ORACLE_HOME/perl/bin/perl $ORACLE_HOME/rdbms/admin/catcon.pl \

  -l /tmp \

  -b gatherstats -- \

  --x"begin dbms_stats.gather_schema_stats('SYS');
dbms_stats.gather_schema_stats('SYSTEM'); end;"
```

Note: Please be aware, especially for production systems, that gathering statistics can be time consuming and resource intensive. There is the potential for cursor invalidation that leads to hard parses. New statistics can result in new optimizer plans. Automatic maintenance tasks may not gather statistics on indexes if the corresponding table stats are not stale.

SET RESOURCE UTILIZATION APPROPRIATELY

`STREAMS_POOL_SIZE` initialization parameter should be set to a reasonable value in the range of 64MB to 256MB. Oracle Data Pump uses Advanced Queuing (AQ) functionality to communicate between processes. If the `SGA_TARGET` initialization parameter is set, then the `STREAMS_POOL_SIZE` initialization parameter should be set to a reasonable minimum for database usage. See Oracle Database Reference for details on setting the `STREAMS_POOL_SIZE` parameter.

BEGINNING WITH RELEASE 19c, DBAS CAN RESTRICT RESOURCE USAGE FOR DATA PUMP

Set the maximum number of Data Pump jobs and the maximum parallelism for pluggable databases in a multitenant environment.

`MAX_DATAPUMP_JOBS_PER_PDB` parameter restricts the number of jobs created. You can allow the parameter to be set automatically to 50 percent of `SESSIONS`. This value must be same for each RAC instance. It can be set and changed dynamically, and it is modifiable per-PDB.

`MAX_DATAPUMP_PARALLEL_PER_JOB` parameter restricts the amount of parallelism in an individual Data Pump job. You can allow the perimeter to be set automatically to 25 percent of `SESSIONS`. This value can be different for each RAC instance. It can be set and changed dynamically, and it is modifiable per-PDB.

Note: If you encounter the error, "ORA-00018: maximum number of sessions exceeded" or "ORA-00020: maximum number of processes (%s) exceeded" you may be allowing too many jobs or too much parallelism, respectively.

USE A NETWOK LINK FOR SERVERS ON DIFFERENT OS AND STORAGE

You can start an import (`impdp`) from the target database over a database link. No dump file will be generated, which can eliminate the need for temporary storage during a migration. Beginning with Oracle Database release 12.2, there is support for Direct Path Load over dblink, including for `LONG` and `LONG RAW` data, using the parameter `ACCESS_METHOD=DIRECT_PATH`.

USE SECUREFILE LOBS

We recommend using SecureFile LOBs, especially with partitioning. SecureFile LOBs offer superior performance, functionality and scalability over BasicFile LOBs, including:

- parallel IO into and out of tables with LOB columns
- compression
- encryption

You can use the **impdp** parameter `LOB_STORAGE=SECUREFILE` to automatically convert old LOBs to SecureFiles. Tables with SecureFile LOBs storage are automatically created in the target database.

SET DATABASE COMPATIBILITY

The database compatibility level affects Data Pump export and import operations. The source database compatibility level determines compatibility level of the export dumpfile set. Prior to Oracle Database 21c, exports and imports are performed using the `expdp` and `impdp` command line clients that match the version of the source and target databases, respectively. Beginning with Oracle Database 21c, universal command line `expdp` and `impdp` clients are introduced that can be used with any version of the database that supports Data Pump. A network mode import can be performed if the compatibility level of the source database is the same as or differs by one makor version from the target database.

`impdp` can always read older version Data Pump dumpfile sets.

Note: If the target database has a lower compatibility level than the source database, use the expdp parameter `VERSION` to specify the target version.

Note: Beginning with Oracle Database 18c, COMPATIBLE should be set to 18.0.0 or 19.0.0, depending upon your release, with no secondary version number, such as 18.1.0, 19.1.0, 19.2.0, etc.

Note: There is no interoperability between Data Pump and the Original Export and Import utilities. impdp cannot read original emp dumpfiles and imp cannot read Data Pump dumpfiles.

More details about Data Pump compatibility can be found in the MyOracleSupport note: Export/Import DataPump Parameter VERSION - Compatibility of Data Pump Between Different Oracle Versions (Doc ID 553337.1)

USE EXPORT COMPRESSION FOR SPACE UTILIZATION AND PERFORMANCE

Compression of metadata and/or data during an export can reduce dumpfile sizes and the size of the data stream during a Network Mode import. It may also improve performance in some cases, especially for Network Mode. However, additional CPU resources are required to perform transformations on the raw data so testing should be performed.

Compression can be applied to metadata, data, both or neither. The default is `COMPRESSION=METADATA_ONLY`.

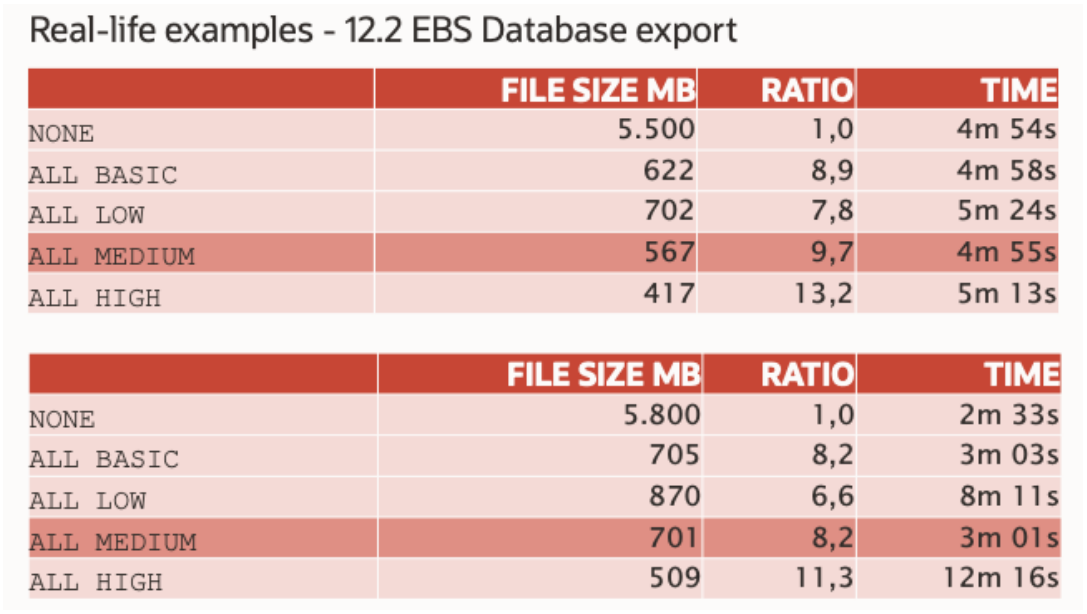
The recommended compression algorithm in most cases is `COMPRESSION=ALL` and `COMPRESSION_ALGORITHM=MEDIUM`. It is faster and has no significant difference in overhead compared to the default of `BASIC`.

COMPRESSION ALGORITHMS	CHARACTERISTICS
BASIC	Good compression without impacting performance
LOW	Use when CPU utilization is more important than the compression ratio
MEDIUM	Recommended. Similar to BASIC with a different algorithm

HIGH	Maximum compression and CPU utilization

Note: Data Pump data compression requires a license for the Advanced Compression Option. No license is required for `COMPRESSION=METADATA_ONLY` or importing a compressed dumpfile.

Note: compression was not supported in Oracle Database 10g release 1 (10.1).


Real-life examples - 12.2 EBS Database export

	FILE SIZE MB	RATIO	TIME
NONE	5.500	1,0	4m 54s
ALL BASIC	622	8,9	4m 58s
ALL LOW	702	7,8	5m 24s
ALL MEDIUM	567	9,7	4m 55s
ALL HIGH	417	13,2	5m 13s

	FILE SIZE MB	RATIO	TIME
NONE	5.800	1,0	2m 33s
ALL BASIC	705	8,2	3m 03s
ALL LOW	870	6,6	8m 11s
ALL MEDIUM	701	8,2	3m 01s
ALL HIGH	509	11,3	12m 16s

DATABASE CHECKS BEFORE EXECUTING A DATA PUMP JOB

AQ_TM_PROCESSES DATABASE PARAMETER

Do not set `AQ_TM_PROCESSES` to zero. A value of zero can reduce the speed of Advanced Queue operations and consequently Data Pump operations that use Advanced Queueing. Either leave this parameter value null or set it to a value greater than 0.

_OPTIMIZER_GATHER_STATS_ON_LOAD HIDDEN PARAMETER

Beginning with Oracle Database 12c, Online Statistics Gathering causes automatic statistics generation for data load operations with Create Table as Select (CTAS) or direct path inserts, for example insert with an append hint. However, the default setting, `_OPTIMIZER_GATHER_STATS_ON_LOAD=TRUE`, can slow down import operations.

Remember to reset the parameter to `TRUE` after the Data Pump operation completes or manually gather database statistics using `DBMS_STATS.GATHER_DATABASE_STATS`.

_lm_share_lock_opt hidden parameter for RAC

Starting with Oracle Real Application Clusters (RAC) release 12.2, RAC allows the Library Cache to use the `SHARE` lock (S-lock) optimization.
To avoid 'Library Cache Lock' (Cycle) issues in 12.2 during an impdp operation started with `parallel>1` on RAC, consider setting `"_lm_share_lock_opt"=FALSE` on all RAC instances during the import or execute the metadata import with parallel=1.

More details can be found in the MyOracleSupport note: 'Library Cache Lock' (Cycle) Seen During DataPump Import in 12.2 RAC Environment (Doc ID 2407491.1)

ADDITIONAL PRACTICES FOR ORACLE CLOUD DATABASE

SCHEMA LEVEL EXPORT FOR MIGRATING TO ORACLE CLOUD

Perform a schema level export when migrating Oracle Database to the Oracle Cloud. Use the parameter `schemas=schema_name,`…for example

```
$ expdp system directory=dp_dir schemas=scott logfile=export_scott.log parallel=8 ...
```

This ensures only permissible user schemas are moved to the Oracle Cloud database.

USE A NETWOK LINK FOR AUTONOMOUS DATABASE MIGRATION

You can start an import (`impdp`) from the target autonomous database over a database link. No dump file will be generated.

ACCESS METHOD

The Data Pump Export `ACCESS_METHOD` parameter instructs Export to use a particular method to unload data. By default, it is set to `AUTOMATIC`. Beginning with release 12.2, `LONG` or `LONG RAW` types in the source database can be exported over dblinks using `ACCESS_METHOD=DIRECT_PATH` with `NETWORK_LINK=<dblink>.`

VERIFY DATAFILE WITH CHECKSUM

Data Pump is used for migrating application data from on-premises into the Oracle Cloud. Oracle Database 21c includes a checksum parameter to add a checksum to the dumpfile that helps confirm the file is valid after a transfer to or from the object store and that it has no malicious changes. The checksum helps ensure the integrity of the contents of a dump file beyond the header block with a cryptographic hash and ensures that there are no unintentional errors in a dump file.

Setting the value specifies whether Oracle Data Pump calculates checksums for the export dump file set, and which hash algorithm is used to calculate the checksum.

IMPORTING INTO A NON-PARTITIONED TABLE

If the source database has partitioned tables and you are migrating data into an Autonomous Data Warehouse database that does not use partitioning, use `DATA_OPTIONS=GROUP_PARTITION_TABLE_DATA`. This allows Data Pump to use the parallel query engine to more efficiently load data into the data warehouse.

USE THE AL32UTF8 DATABASE CHARACTER SET

Oracle recommends using `AL32UTF8` as the database character set. It is the most common character set because it is a superset of all other character sets.

CONCLUSION

Oracle Data Pump is a mature, full featured and flexible tool for Oracle Database migration. As discussed in this whitepaper, with each new release it provides more options for optimizing performance and resource utilization. Following the best practices in this white paper will help your Data Pump exports and imports run as smoothly and quickly as possible.

CONNECT WITH US

Call +1.800.ORACLE1 or visit oracle.com.
Outside North America, find your local office at oracle.com/contact.

blogs.oracle.com facebook.com/oracle twitter.com/oracle

Oracle Data Pump Best Practices
August, 2021
Author: William Beauregard
Contributing Authors: Roy Swonger