ORACLE

Data Pump Extreme – Deep Dive

Virtual Classroom Series

on Unsplash

Data Pump
Data Pump Extreme – Deep Dive
Best Practices and Real World Scenarios

Virtual Classroom Series – Episode 15

Photo by Benjamin Davies on Unsplash

ROY SWONGER
Vice President
Database Upgrade, Utilities & Patching

royfswonger

@royfswonger

MIKE DIETRICH
Distinguished Product Manager
Database Upgrade and Migrations

mikedietrich

@mikedietrichde

https://mikedietrichde.com

DANIEL OVERBY HANSEN
Senior Principal Product Manager
Cloud Migrations

dohdatabase

@dohdatabase

https://dohdatabase.com

RODRIGO JORGE
Senior Principal Product Manager
Database Patching and Upgrade

rodrigoaraujorge

@rodrigojorgedba

https://dbarj.com.br/en

BILL BEAUREGARD
Senior Principal Product Manager
Data Pump and SQL Loader

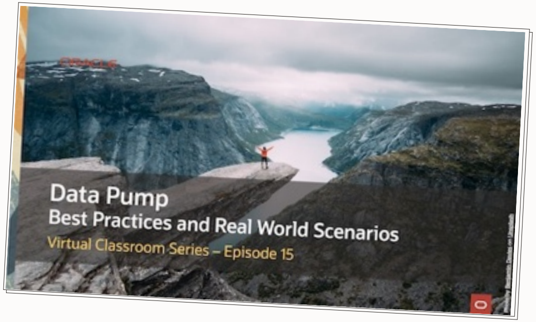
in william-beauregard-3053791

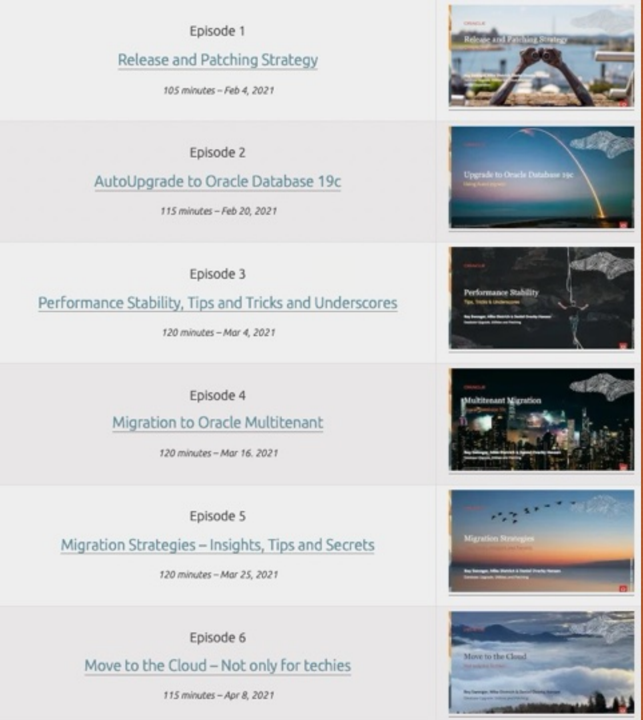
KLAUS GRONAU
Consulting Member
of Technical Staff

klaus-gronau-39a43aa9

Webinar | Get The Slides

https://MikeDietrichDE.com/slides


Data Pump
Best Practices and Real World Scenarios
Virtual Classroom Series – Episode 15



Episode 1

Release and Patching Strategy

105 minutes – Feb 4, 2021

Episode 2

AutoUpgrade to Oracle Database 19c

115 minutes – Feb 20, 2021

Episode 3

Performance Stability, Tips and Tricks and Underscores

120 minutes – Mar 4, 2021

Episode 4

Migration to Oracle Multitenant

120 minutes – Mar 16. 2021

Episode 5

Migration Strategies – Insights, Tips and Secrets

120 minutes – Mar 25, 2021

Episode 6

Move to the Cloud – Not only for techies

115 minutes – Apr 8, 2021

Recorded Web Seminars

https://MikeDietrichDE.com/videos

More than 30 hours of technical content, on-demand, anytime, anywhere

deep dive

DATA PUMP

with development

Best Practices

Essentials

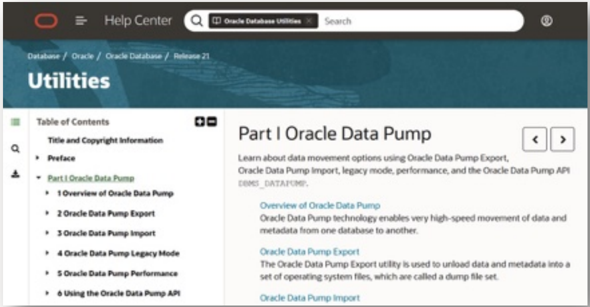
Extreme

Advanced

"Oracle Data Pump technology enables very high-speed movement of data and metadata from one database to another."

Oracle Database Utilities 19c

Data Pump | Documentation



Help Center

Oracle Database Utilities

Search

Database / Oracle / Oracle Database / Release 21

Utilities

Part I Oracle Data Pump

Learn about data movement options using Oracle Data Pump Export, Oracle Data Pump Import, legacy mode, performance, and the Oracle Data Pump API DBMS_DATAPUMP.

Overview of Oracle Data Pump
Oracle Data Pump technology enables very high-speed movement of data and metadata from one database to another.

Oracle Data Pump Export
The Oracle Data Pump Export utility is used to unload data and metadata into a set of operating system files, which are called a dump file set.

Oracle Data Pump Import

Oracle Database 19c – Utilities Guide

Oracle Database 21c – Utilities Guide

Data Pump | Documentation

Oracle Database 19c – Utilities Guide

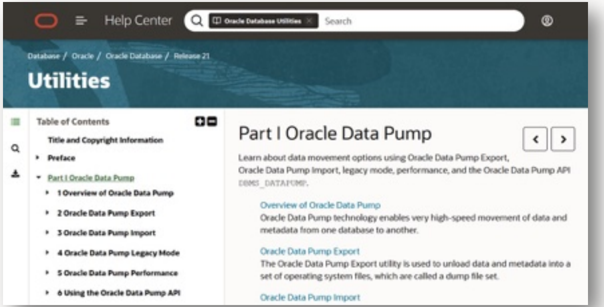
Oracle Database 21c – Utilities Guide

ADB Data Pump Export to Object Store

ADB Import Data Using Oracle Data Pump

ADB DB Links for Network Import

Data Pump Best Practices



Database / Oracle / Oracle Database / Release 21

Utilities

Table of Contents

Title and Copyright Information
- Preface
- Part I Oracle Data Pump
  - 1 Overview of Oracle Data Pump
  - 2 Oracle Data Pump Export
  - 3 Oracle Data Pump Import
  - 4 Oracle Data Pump Legacy Mode
  - 5 Oracle Data Pump Performance
  - 6 Using the Oracle Data Pump API

Part I Oracle Data Pump

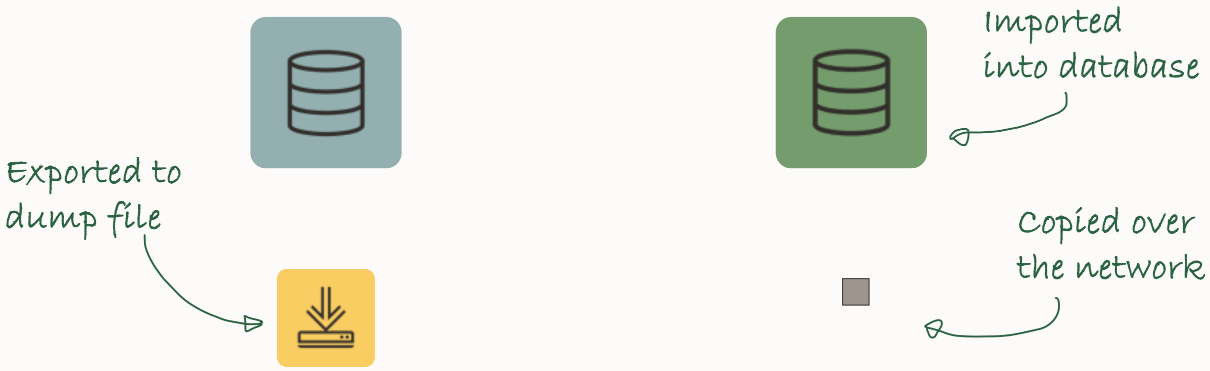
Learn about data movement options using Oracle Data Pump Export, Oracle Data Pump Import, legacy mode, performance, and the Oracle Data Pump API `DBMS_DATAPUMP`.

Overview of Oracle Data Pump
Oracle Data Pump technology enables very high-speed movement of data and metadata from one database to another.

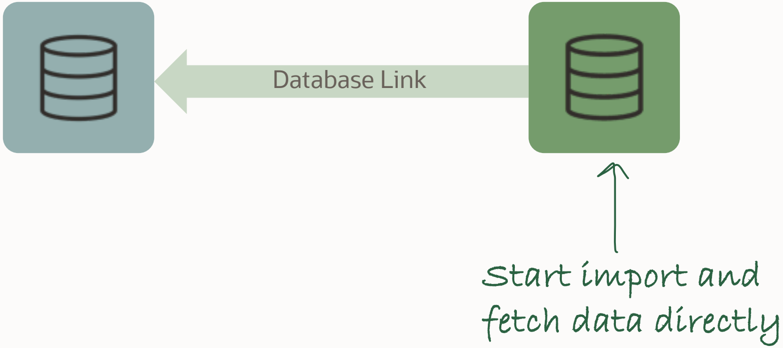
Oracle Data Pump Export
The Oracle Data Pump Export utility is used to unload data and metadata into a set of operating system files, which are called a dump file set.

Oracle Data Pump Import

Data Pump | Dump File



Data Pump | Dump File



Data Pump | Mode Comparison

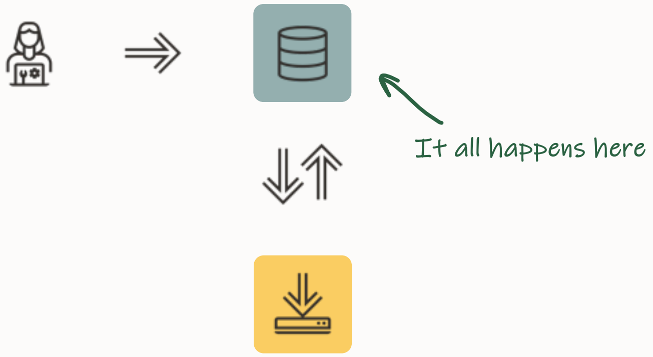
DUMP FILE	NETWORK
Requires access to file system	SQL*Net connectivity
Requires disk space for dump files	No extra disk space needed
Full functionality	Restricted functionality

Pro tip: Read more about how Data Pump moves data

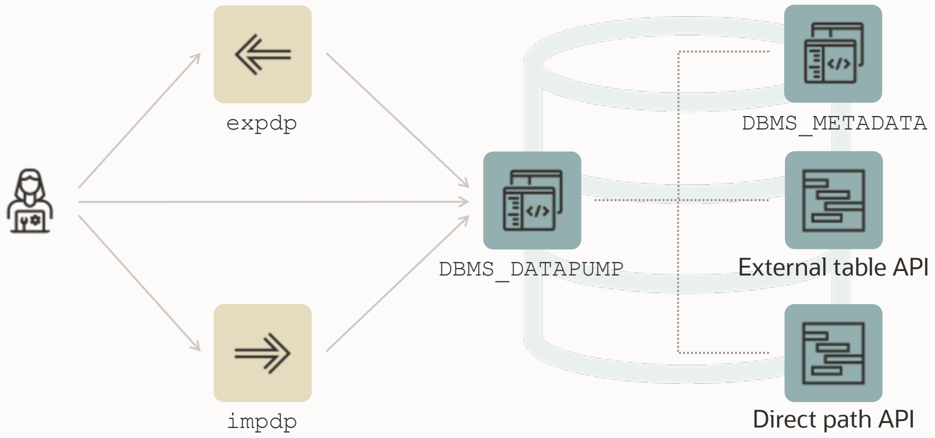
Data Pump | Architecture

Data Pump is server-based,

not client-based



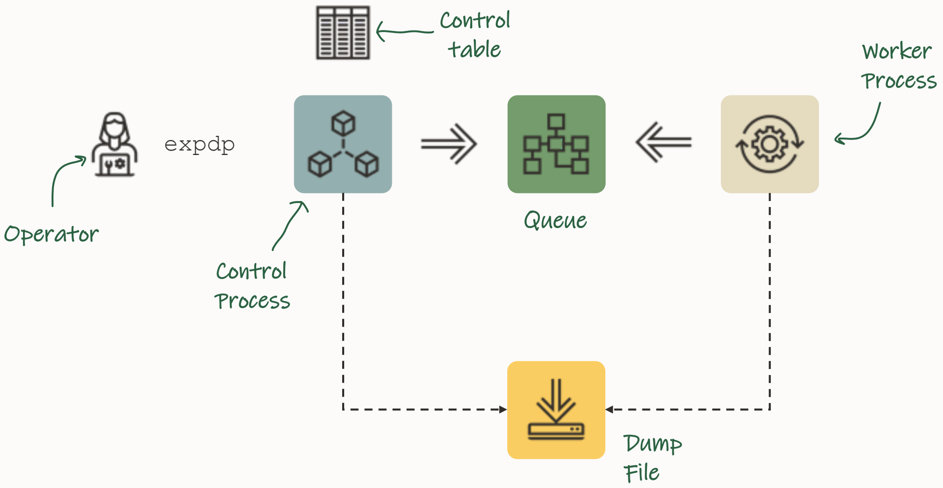
Data Pump | Architecture



Data Pump | Architecture Block Diagram

Clients	`Expdp/impdp` & Interactive commands	Apps & services: ZDM, ODM, SQLcl, OEM…	
APIs	`DBMS_DATAPUMP`	`DBMS_METADATA`	`DBMS_TTS`
Engine	Data Pump Engine		
Access Methods & Drivers	External Tables via `ORACLE_DATAPUMP` Driver	Direct Path `DPAPI` API	Conventional Path (SQL)
Transport media	Dumpfile set	Network Mode Import	

Data Pump | Architecture



Data Pump | Architecture

Control Table

A regular heap table containing:

- Job info and parameters
- Current status
- Object information
- Index into the dump files
- Enables restarts

Data Pump | Architecture

Control Table

- Control table is dropped upon successful completion of a job
- Optionally, kept using `KEEP_MASTER=Y`
- Can be queried like any other table
- Last object exported to dump file & first imported
- Can be queried for troubleshooting

Data Pump | Architecture

```
$ expdp dpuser/oracle schemas=app keep_master=y
```

Data Pump | Architecture

```
SQL> select name, value_t from dpuser.sys_export_schema_01;

NAME                   VALUE_T
---------------------- ---------------------------------------------
SYS_EXPORT_SCHEMA_01   DB19.LOCALDOMAIN
LOG_FILE_DIRECTORY     DATA_PUMP_DIR
LOG_FILE_NAME          export.log
CLIENT_COMMAND         dpuser/******** schemas=app keep_master=y
SCHEMA_LIST            'APP'
SCHEMA_EXPR            IN ('APP')
COMPRESSION            METADATA_ONLY
COMPRESSION_ALGORITHM  BASIC
DATA_ACCESS_METHOD     AUTOMATIC
.
.
.
```

Data Pump | Unloading and loading

Data Files

Direct Path

External Tables

Insert as Select

Conventional Path

Used for transportable tablespace

Only metadata is unloaded into/loaded from dumpfile

Data remains in data files

Pro tip: Cross-endian data migration requires data files are converted

Data Pump | Unloading and loading

Data Files

Direct Path

External Tables

Insert as Select

Conventional Path

Unloads from / load into data files directly

Circumvents SQL layer

Fast

Not usable in all situations

Pro tip: Data Pump automatically selects the best unload/load method

Data Pump | Unloading and loading

Data Files

Direct Path

External Tables

- Use SQL layer to unload to / load from external table
- Can use `APPEND` hint for faster load
- Very good parallel capabilities
- Dump file format similar to direct path

Insert as Select

Conventional Path

Pro tip: Data unloaded with Data Pump is not compatible with a regular external table (`CREATE TABLE ... ORGANIZATION EXTERNAL ...`)

Data Pump | Unloading and loading

Data Files

Direct Path

External Tables

Insert as Select

- Used by network link imports only
- Will disable use of direct path
- Not very common

Conventional Path

Data Pump | Unloading and loading

Data Files

Direct Path

External Tables

Insert as Select

Conventional Path

- Used as last resort
- Slower
- Import only

Data Pump | Unloading and loading

```
ACCESS_METHOD=[AUTOMATIC | DIRECT_PATH | EXTERNAL_TABLE | CONVENTIONAL_PATH | INSERT_AS_SELECT]
```

Pro Tip: Current table stats will help Data Pump make the right choice

Data Pump | Unloading and loading

```
$ expdp dpuser/oracle schemas=app metrics=y
```

Data Pump | Metadata

A category of metadata is described by an object path

Examples:

```
TABLE
TABLE/INDEX
TABLE/STATISTICS/TABLE_STATISTICS
TABLE/TRIGGER
```

You can get a full list of object paths from these views:

```
DATABASE_EXPORT_OBJECTS
SCHEMA_EXPORT_OBJECTS
TABLE_EXPORT_OBJECTS
```

Data Pump | Metadata

```
SQL> select object_path, comments from schema_export_objects;

OBJECT_PATH                   COMMENTS
---------------------------- ---------------------------------------------------
ALTER_FUNCTION                Recompile functions
ALTER_PACKAGE_SPEC            Recompile package specifications
ALTER_PROCEDURE               Recompile procedures
ANALYTIC_VIEW                 Analytic Views
AQ                            Advanced Queuing
ASSOCIATION                   Statistics type associations
ATTRIBUTE_DIMENSION           Attribute Dimensions
AUDIT_OBJ                     Object audits on the selected tables
CLUSTER                       Clusters in the selected schemas and their indexes
CLUSTERING                    Table clustering
.
.
.
```

Use `INCLUDE` or `EXCLUDE` to add or remove a specific category of metadata

`INCLUDE` and `EXCLUDE` are mutually exclusive. Now, from Oracle Database 21c they can be combined

Data Pump | Metadata

Some metadata has dependencies.

Example: Excluding a table will also exclude

- Indexes
- Constraints
- Grants
- Triggers
- And the like upon that table

Example: Excluding an index will also exclude

- Statistics on that index

Getting Started

Data Pump

Data Pump | Getting Started

Privilege
Directory
Streams Pool
Mode
Parameter File
Consistency

You can export your own schema.

In addition, two predefined roles:

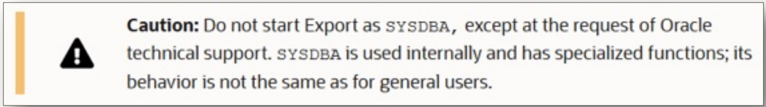
- `DATAPUMP_EXP_FULL_DATABASE`
- `DATAPUMP_IMP_FULL_DATABASE`

Pro tip: These roles are powerful - use caution when granting them

Data Pump | Getting Started

Privilege
Directory
Streams Pool
Mode
Parameter File
Consistency

Don't use `SYS AS SYSDBA`



Caution: Do not start Export as `SYSDBA`, except at the request of Oracle technical support. `SYSDBA` is used internally and has specialized functions; its behavior is not the same as for general users.

Database 19c, Utilities Guide

Do use `SYSTEM` with the Data Pump roles

Data Pump | Getting Started

Privilege

Directory

Streams Pool

Mode

Parameter File

Consistency

Tablespace quota

- For create control table

Example of a Data Pump user

```
SQL> create user dpuser identified by oracle;

SQL> grant datapump_exp_full_database to dpuser;
SQL> grant datapump_imp_full_database to dpuser;

SQL> alter user dpuser quota unlimited on users;
```

Data Pump | Getting Started

- Privilege
- **Directory**
- Streams Pool
- Mode
- Parameter File
- Consistency

Needed to store dump files and log files

```
$ mkdir /home/oracle/dp

SQL> create directory dpdir as '/home/oracle/dp';
SQL> grant read, write on directory dpdir to dpuser;
```

If you don't specify a directory, the default is `DATA_PUMP_DIR`

```
SQL> select directory_path
     from   dba_directories
     where  directory_name='DATA_PUMP_DIR';
```

Tech Tip: To Export to ASM, the directory must point to a disk group

Data Pump | Getting Started

- Privilege
- Directory
- **Streams Pool**
- Mode
- Parameter File
- Consistency

Ensure `STREAMS_POOL_SIZE` is at a reasonable value

```
SQL> select current_size/1024/1024 as current_size_mb
     from   v$sga_dynamic_components
     where  component='streams pool';
```

Typically, in the range of 64M to 256M is adequate

```
SQL> alter system set streams_pool_size=256m scope=both;
```

Pro tip: Read about how other parameters affect Data Pump

Data Pump | Getting Started

Privilege
Directory
Streams Pool
Mode
Parameter File
Consistency

What do you want to export or import?

- Full
- <u>Schema</u>
- Table
- Tablespace
- Transportable Tablespace

Pro tip: Ensure user's default tablespace exists on the target

Export objects not in a schema mode export by performing `FULL=Y` and `INCLUDE=SYNONYM,ROLE,PROFILE` …

Data Pump | Getting Started

Privilege
Directory
Streams Pool
Mode
Parameter File
Consistency

You can specify Data Pump options in two ways:

On command line

```
$ expdp dpuser schemas=app directory=dp_dir
```

In parameter file - recommended

```
$ cat export.par
schemas=app
directory=dp_dir

$ expdp dpuser parfile=export.par
```

Data Pump | Getting Started

Privilege
Directory
Streams Pool
Mode
Parameter File
Consistency

Recommended to use parameter files

- Certain characters must be escaped
- These characters vary by operating system
- Escape characters vary by operating system
- Avoid typing long commands

Data Pump | Getting Started

Privilege
Directory
Streams Pool
Mode
Parameter File
Consistency

Default, consistent on a "per-table-basis"

Export starts at	SCN 100
Table *A* exported at	SCN 110
Table *B* exported at	SCN 125
Table *C* exported at	SCN 137
Table *D* exported at	SCN 142

Data Pump | Getting Started

Privilege
Directory
Streams Pool
Mode
Parameter File
Consistency

Optionally, make export completely consistent

Export starts at SCN 100

Table *A* exported at SCN 100

Table *B* exported at SCN 100

Table *C* exported at SCN 100

Table *D* exported at SCN 100

Data Pump | Getting Started

Privilege
Directory
Streams Pool
Mode
Parameter File
Consistency

Consistent as of timestamp

```
$ expdp dpuser ... flashback_time=systimestamp
```

Consistent as of SCN

```
$ expdp dpuser ... flashback_scn=nnn
```

Legacy mode

```
$ expdp dpuser ... consistent=y
```

Requires UNDO

Pro tip: Export from your standby database

Flashback is not required to use the parameters `flashback_scn` or `flashback_systimestamp`. Oracle Database will get data from UNDO to ensure database wide consistency.

Data Pump | Getting Started

Watch on YouTube

Faster Patching

DPLoad Patch and Bundle Patch

Data Pump Bundle Patch - MOS Note: 2819284.1

100+

fixes

Data Pump Bundle Patch

Data Pump bundle Patch

Fewer Bugs

Important patches are included.
Monitor for bugs that affects many customers.

Faster Patching

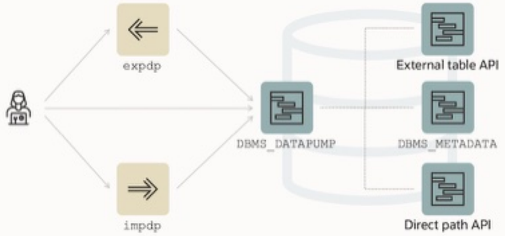
The bundle patch changes the way Data Pump is patched. Subsequent patches apply faster.

Patching | DPload vs DP Bundle

DPload Patch

- Speeds up Data Pump patching
- No downtime
- Rolling

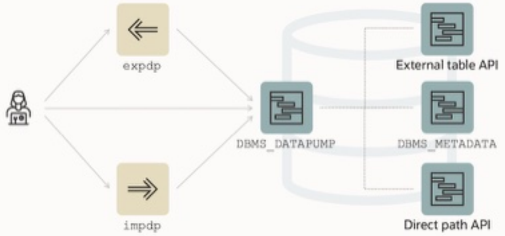
dpload.sql



Data Pump Bundle Patch

- Speeds up Data Pump patching
- Adds lots of important fixes
- Non-rolling

dpload.sql



The DPload Patch speeds up future Data Pump patching by 50-80%

Patching | Symptoms

`dpload.sql` is used when Data Pump gets patched

- Patching can be time consuming

```
Rem    NAME
Rem      dpload.sql - load entire Data Pump utility.
Rem
Rem    DESCRIPTION
Rem      load entire Data Pump utility (including the metadata layer).
Rem
Rem    NOTES
Rem      When there are changes to any of the following Data Pump files it
Rem      is necessary to unload and then reload the entire Data Pump utility.
Rem         src/server/datapump/services/prvtkupc.sql
Rem         src/server/datapump/ddl/prvtmetd.sql
Rem         admin/dbmsdp.sql
Rem         admin/catmeta.sql
Rem         admin/catmettypes.sql
Rem         admin/catmetviews.sql
Rem         admin/catmetinsert.sql
```

Update to the latest Release Update and then apply the Data Pump bundle patch

Data Pump Recommended Proactive Patches
For 19.10 and Above (Doc ID 2819284.1)

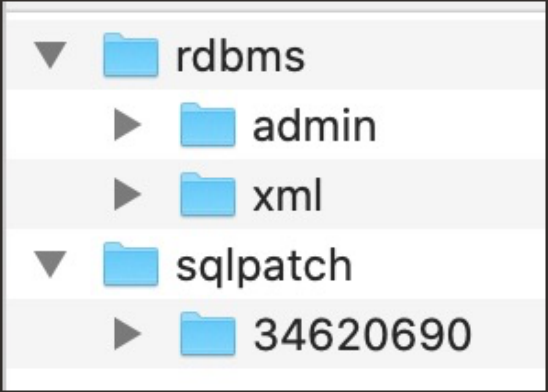
The Data Pump bundle patch is not in the Oracle Database Release Update

It is not RAC Rolling and Standby-first Installable

The Data Pump bundle patch is **not** RAC Rolling and Standby-first Installable

But … it's much easier than it looks like

Data Pump Bundle Patch Contents



Bundle Patch contains only:

- sql
- plsql
- xml

But it does not contain any files which require a compilation/make of `rdbms`

➔ It can be applied ONLINE

```
OPatch continues with these patches:   34620690

Do you want to proceed? [y|n]
y
User Responded with: Y
All checks passed.
Backing up files...
Applying interim patch '34620690' to OH
'/u01/app/oracle/product/19'


Patching component oracle.rdbms, 19.0.0.0.0...


Patching component oracle.rdbms.dbscripts, 19.0.0.0.0...
Patch 34620690 successfully applied.
```

When you run datapatch, ensure that there are no active Data Pump jobs

The OPatch Lsinventory for the Data Pump Bundle includes patch descriptions, for instance:

BUG 34972375 - DATAPUMP BUNDLE PATCH 19.18.0.0.0

Non-Binary Online Patching Safeguards

Installing the Data Pump Bundle Patch when Data Pump is in use:
Built-in 3-minute timeout before signaling an error

```
BEGIN ku$_dpload.initial_phase; END;
*
ERROR at line 1:
ORA-20000: Retry dpload.sql script later when
Data Pump and Metadata API are not in use; current users are:
pid:11720, user:SYS, machine:<Machine>, sid:263,
module:sqlplus@<ConnectString> (TNS V1-
ORA-06512: at "SYS.KU$_DPLOAD", line 1042
ORA-06512: at line 1
```

Non-Binary Online Patching Safeguards

Attempting to run Data Pump while patching is in progress:

```
Connected to: Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
ORA-31626: job does not exist
ORA-31637: cannot create job SYS_EXPORT_FULL_01 for user SYSTEM
ORA-06512: at "SYS.KUPV$FT", line 1142
ORA-06512: at "SYS.DBMS_SYS_ERROR", line 95
ORA-06512: at "SYS.KUPV$FT", line 1751
ORA-39062: error creating master process DM00
ORA-39107: Master process DM00 violated startup protocol. Master error:
…
```

Note: With the 19.14 (or later) Data Pump Bundle Patch installed you will see a much better error message:

```
ORA-39442: Data Pump software update in progress
```

Importing a complete application with data goes from almost 2.5 hours to 48 minutes
– by just applying the Data Pump bundle patch

A global provider of financial services

deep dive

DATA PUMP

with development

Use a Data Pump parameter (.par) file

- Avoid errors typing long commands

```
$ cat export.par
schemas=app
directory=dp_dir
```

```
$ expdp dpuser parfile=export.par
```

Pro tip: there is no practical limitation on the Data Pump parameter file size

Data Pump | Best Practices

Statistics
Diagnostics
Parallel
LOBs
Dump files
Compression
Checksum
Multitenant

Ensure dictionary statistics are current

- Before export
- After import

```
SQL> begin
        dbms_stats.gather_schema_stats('SYS');
        dbms_stats.gather_schema_stats('SYSTEM');
     end;
```

Pro tip: You can also use `GATHER_DICTIONARY_STATS`

Data Pump | Best Practices

Statistics
Diagnostics
Parallel
LOBs
Dump files
Compression
Checksum
Multitenant

Exclude optimizer statistics from export

```
#Regular export
expdp ... exclude=statistics

#Transportable tablespaces
expdp ... exclude=table_statistics,index_statistics
```

After import:

- Gather statistics in target database
- Or transport statistics using `DBMS_STATS`

Data Pump | Best Practices

Statistics
Diagnostics
Parallel
LOBs
Dump files
Compression
Checksum
Multitenant

Always include diagnostics in logfile

```
expdp ... logtime=all metrics=yes

impdp ... logtime=all metrics=yes
```

If you need help, we will always ask
for a logfile with these parameters set

Include diagnostics in the logfile

Always use `METRICS=YES` and `LOGTIME=ALL`

Data Pump | Best Practices - Log Files

- No diagnostics

```
Processing object type DATABASE_EXPORT/FINAL_POST_INSTANCE_IMPCALLOUT/MARKER
Processing object type DATABASE_EXPORT/AUDIT_UNIFIED/AUDIT_POLICY_ENABLE
Processing object type DATABASE_EXPORT/POST_SYSTEM_IMPCALLOUT/MARKER
. . exported "SYS"."KU$_USER_MAPPING_VIEW"                5.890 KB      25 rows
. . exported "SYSTEM"."REDO_DB"                           25.59 KB       1 rows
```

- Full diagnostics

```
02-NOV-21 19:43:56.061: W-1 Processing object type DATABASE_EXPORT/FINAL_POST_INSTANCE_IMPCALLOUT/MARKER
02-NOV-21 19:43:56.064: W-1      Completed 1 MARKER objects in 0 seconds
02-NOV-21 19:43:59.171: W-1 Processing object type DATABASE_EXPORT/AUDIT_UNIFIED/AUDIT_POLICY_ENABLE
02-NOV-21 19:43:59.195: W-1      Completed 2 AUDIT_POLICY_ENABLE objects in 0 seconds
02-NOV-21 19:43:59.380: W-1 Processing object type DATABASE_EXPORT/POST_SYSTEM_IMPCALLOUT/MARKER
02-NOV-21 19:43:59.387: W-1      Completed 1 MARKER objects in 0 seconds
02-NOV-21 19:43:59.830: W-1 . . exported "SYS"."KU$_USER_MAPPING_VIEW"    5.890 KB  25 rows in 0 seconds using external_table
02-NOV-21 19:43:59.923: W-1 . . exported "SYSTEM"."REDO_DB"               25.59 KB   1 rows in 0 seconds using direct_path
```

Data Pump | Best Practices

- Statistics
- Diagnostics
- **Parallel**
- LOBs
- Dump files
- Compression
- Checksum
- Multitenant

Use parallel to speed up the exports and imports

```
expdp ... parallel=n

impdp ... parallel=n
```

Degree of parallelism

• Cloud	Number of OCPUs
• On-prem (x86-64)	CPU cores x 2
• On-prem (other)	Depends

Pro tip: Hyperthreaded cores does not count

Data Pump | Best Practices

Statistics
Diagnostics
Parallel
LOBs
Dump files
Compression
Checksum
Multitenant

For exports

- For optimal performance, use multiple dump files
- Enables concurrent write to dump files

For imports

- Number of dump files does not affect parallel import
- Each worker process needs shared access to read from dump files
- Level of parallelism can be different than export

Pro tip: Parallelsim only supported in Enterprise Editiont

Parallelism and RAC

- Control process is created on the node where the Data Pump job is initiated
- Worker threads, including Parallel Query (PQ) processes are distributed across the RAC nodes
- `CLUSTER=N` parameter prevents distributing workers across nodes
- Parameters, such as job_queue_processes govern parallelism on each node

Parallelism and Indexes

Prior to 12.2:

- One index at a time is created using the parallelism of the job.
  - PARALLEL=32 will generate CREATE INDEX…PARALLEL 32 for each index.

12.2 and newer:

- multiple indexes are created at the same time up to the limit of parallelism, each with CREATE INDEX…PARALLEL 1

Data Pump | Best Practices

- Statistics
- Diagnostics
- Parallel
- **LOBs**
- Dump files
- Compression
- Checksum
- Multitenant

> *SecureFiles is the default storage mechanism for LOBs starting with Oracle Database 12c, and Oracle strongly recommends SecureFiles for storing and managing LOBs, rather then BasicFiles. BasicFiles will be deprecated in a future release.*

Database SecureFiles and Large Objects Developer's Guide

Data Pump | Best Practices

- Statistics
- Diagnostics
- Parallel
- **LOBs**
- Dump files
- Compression
- Checksum
- Multitenant

Always transform LOBs to SecureFile LOBs during import

```
impdp ... transform=lob_storage:securefile
```

It is faster to import LOBs,
when they are transformed to SecureFile LOBs

BasicFile LOBs do not allow parallel DML

Pro tip: You can also set database parameter `db_securefile=ALWAYS`

Data Pump | Best Practices

Importing as BasicFiles

```
... imported "SCHEMA"."TABLE"      31.83 GB   681025 rows in 804 seconds using direct_path
```

Importing as SecureFiles

```
... imported "SCHEMA"."TABLE"     31.83 GB   681025 rows in 261 seconds using external_table
```

If you set the database parameter `db_securefile=ALWAYS` there is no need to use the Data Pump parameter `TRANSFORM=LOB_STORAGE:SECUREFILE`

Can the space required for compressed dump file set be estimated?

Data Pump | Best Practices

Statistics
Diagnostics
Parallel
LOBs
Dump files
Compression
Checksum
Multitenant

Always export to multiple files

```
expdp ... dumpfile=mydump%L.dmp
```

Optionally, limit individual file size

```
expdp ... dumpfile=mydump%L.dmp filesize=5G
```

Import using a wild card in the dumpfile name or list all of the file names

Pro tip: You can also use the old `%U` format

Can the space required for compressed dump file set be estimated?

Space requirement can’t be predicted

Data types have variable levels of compression:
- Scalar data compresses a lot
- Binary data (BLOBs, images, etc) doesn’t

Depends too on whether data is compressed in the database and whether it’s encrypted

Pro tip: Dumpfile sizes depend on how workers are assigned

Can dumpfiles be concatenated? No.

Data Pump | Best Practices

Use compression to speed up your export

```
compression=all
compression_algorithm=medium
```

Requires Advanced Compression Option

Best Practice: Apply TDE at the tablespace level

set `TRANSFORM=TABLE_COMPRESSION_CLAUSE:NONE`

- Prevents the compression clause in the dumpfile.
- Enables the tablespace compression setting

Data Pump | Best Practices

Statistics
Diagnostics
Parallel
LOBs
Dump files
Compression
Checksum
Multitenant

Compression algorithms

BASIC:	The same algorithm used in previous versions. Good compression, without severely impacting on performance
LOW:	For use when reduced CPU utilization is a priority over compression ratio
MEDIUM:	Recommended option. Similar characteristics to BASIC, but uses a different algorithm
HIGH:	Maximum available compression, but more CPU intensive

Data Pump | Best Practices

Statistics
Diagnostics
Parallel
LOBs
Dump files
Compression
Checksum
Multitenant

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

Data Pump | Best Practices

- Statistics
- Diagnostics
- Parallel
- LOBs
- Dump files
- **Compression**
- Checksum
- Multitenant

Default compression type

```
compression=metadata_only
```

Only metadata information is compressed.

Does not require Advanced Compression Option

Pro tip: Importing a compressed dump file does not require a license for ACO

Data Pump | Best Practices

What can happen to a dump file when it is transferred?

- Tampering
- Corruption

NEW IN 21c

Data Pump | Best Practices

Statistics
Diagnostics
Parallel
LOBs
Dump files
Compression
Checksum
Multitenant

Data Pump can calculate checksum on export

```
expdp ... checksum_algorithm=sha384
```

Verify dump file integrity on import

```
impdp... verify_only=yes


impdp ... verify_checksum=yes
```

NEW IN 21c

Data Pump | Best Practices

Alternatively, calculate checksum yourself

```
[oracle@hol]$ md5sum metal*.dmp




5edf66ed92086b4f69580fc27b75f662  metal_01.dmp
59eb25ff2a0f648c051a9212e0861979  metal_02.dmp
29951a56abe074d9151c27728d88e9eb  metal_03.dmp
c8860e7a71e74f8013068240b598c116  metal_04.dmp
0d05d258e4b501c657cd9490b7e48715  metal_05.dmp
1e367394a31e2ce45d2aeb6a3d4f9507  metal_06.dmp
9c276aa580c0e57c0829f274d04d15de  metal_07.dmp
0d560d0ce57c47425424e17604d8ec49  metal_08.dmp
```

```
SQL> SELECT object_name, checksum
     FROM DBMS_CLOUD.LIST_OBJECTS(
          '<credential_name>',
          '<location_uri>');

metal_01.dmp 5edf66ed92086b4f69580fc27b75f662
metal_02.dmp 59eb25ff2a0f648c051a9212e0861979
metal_03.dmp 29951a56abe074d9151c27728d88e9eb
metal_04.dmp c8860e7a71e74f8013068240b598c116
metal_05.dmp 0d05d258e4b501c657cd9490b7e48715
metal_06.dmp 1e367394a31e2ce45d2aeb6a3d4f9507
metal_07.dmp 9c276aa580c0e57c0829f274d04d15de
metal_08.dmp 0d560d0ce57c47425424e17604d8ec49
```

- Windows: `Get-FileHash *.dmp -Algorithm MD5`
- Errors manifests as `ORA-31693 ORA-29913 ORA-29104`

Data Pump | Best Practices

Statistics
Diagnostics
Parallel
LOBs
Dump files
Compression
Checksum
Multitenant

Avoid *noisy-neighbour* if resources are insufficient

Restrict number of concurrent Data Pump jobs in a PDB

```
SQL> alter system set max_datapump_jobs_per_pdb=2
     container=all;
```

Restrict the parallel degree in a Data Pump job

```
SQL> alter system set max_datapump_parallel_per_job=2
     container=all;
```

Doing a timezone upgrade during import is a best practice

```
$ expdp ... schemas=hr,oe include=table exclude=statistics
```

Include and Exclude in the same command

```
$ expdp ... schemas=hr,oe include=table exclude=statistics
```

Use the Universal Data Pump Client

Data Pump and SQL*Loader clients are in the “Tools” package of the Instant Client

deep dive

DATA PUMP

with development

Best Practices

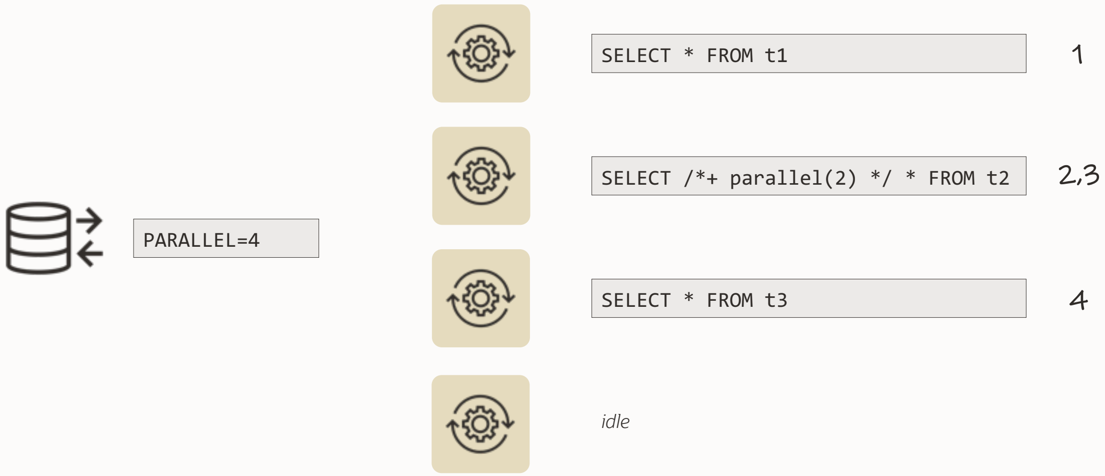
Essentials

Extreme

Advanced

Use `PARALLEL` to speed up your Data Pump job

Parallel



```
PARALLEL=4
```

```
SELECT * FROM t1
```
1

```
SELECT /*+ parallel(2) */ * FROM t2
```
2,3

```
SELECT * FROM t3
```
4

idle

Why isn’t my job using all the PARALLEL that I gave it?

Why Might Data Pump Workers Be Idle?

Some possibilities…

1. Data Pump might be using Parallel Query
   - PX processes count against the total parallelism
2. BasicFile LOBs do not allow parallel DML
3. Export parallelism requires multiple dumpfiles
4. `NETWORK_LINK` jobs
   - Export and import metadata serially
   - Cannot use Parallel Query (one worker per partition/subpartition, but no PQ within a partition)

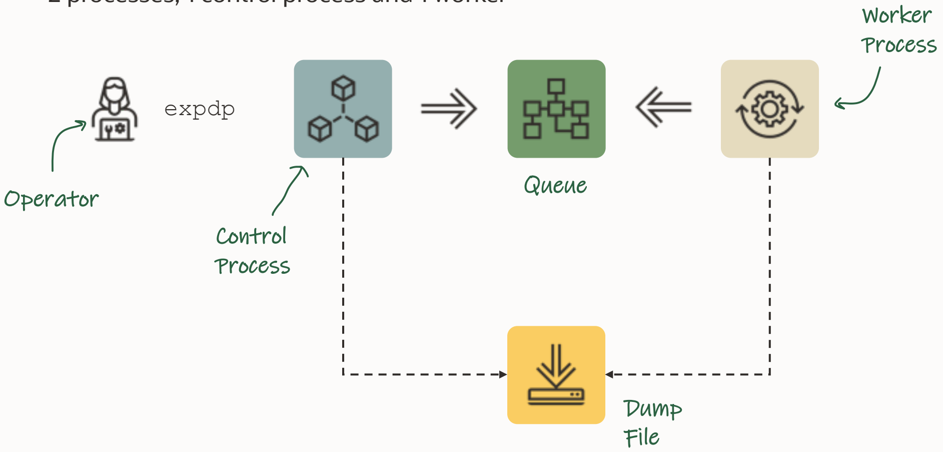
Everything Parallel

Export

Parallel | Control and Worker process

If you the default or `PARALLEL=1`

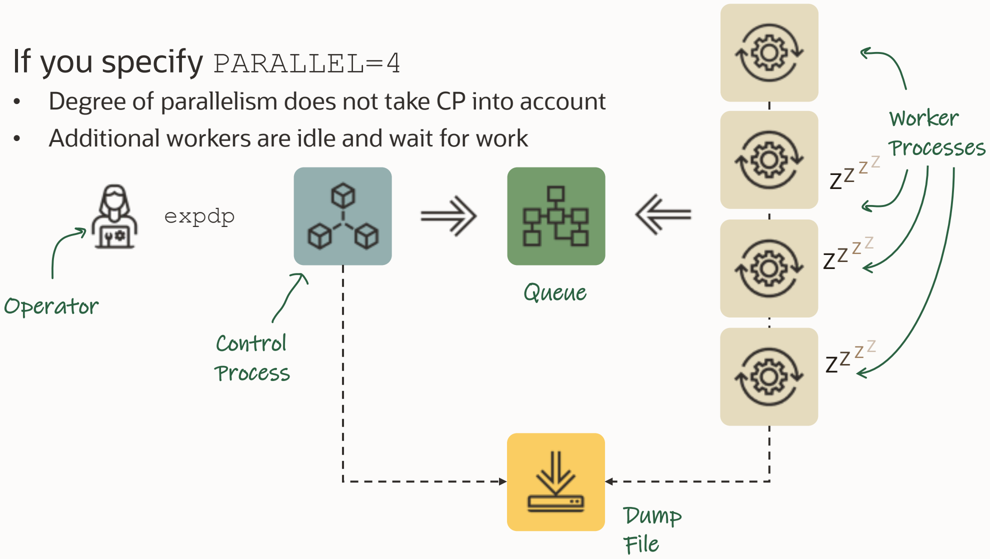
- 2 processes, 1 control process and 1 worker



Parallel | Degree of Parallelism

If you specify `PARALLEL=4`

- Degree of parallelism does not take CP into account
- Additional workers are idle and wait for work



Data Pump: Parallel Operations

Overview

- A job starts w/ a minimum of 2 processes: a Control Process (CP) + 1 Worker

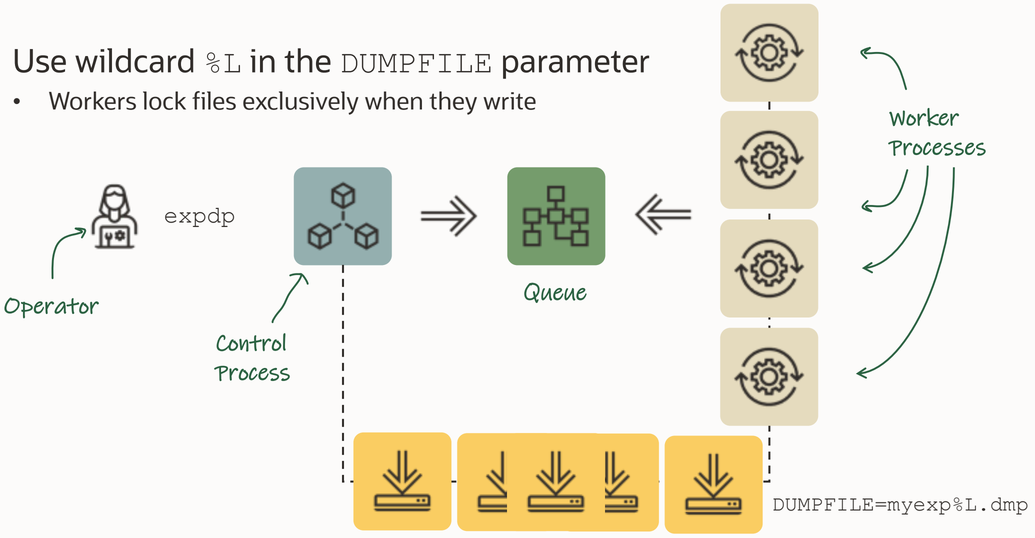
- The user can specify a Degree of Parallelism (DOP) for the Data Pump job
  - DOP = maximum number of ACTIVE parallel processes (Active workers + PQ processes)
    - DOP does not include the CP or shadow process
    - Any additional workers are idle and wait for work
    - Data Pump will only start the number of threads needed to complete the task

- Control Process: Verifies parameters & job description, controls & assigns work items to workers

Parallel | expdp - Multiple Dump Files

Use wildcard `%L` in the `DUMPFILE` parameter

- Workers lock files exclusively when they write

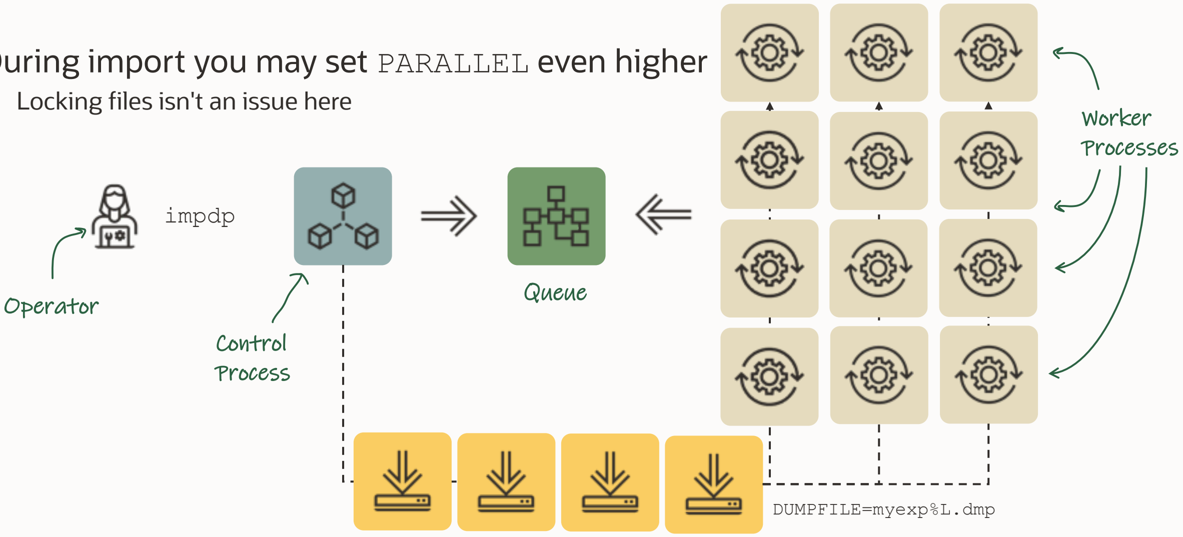


Use `PARALLEL` together with wildcard `%L` in the `DUMPFILE` parameter to create multiple dump files

Parallel | impdp - Multiple Dump Files

During import you may set `PARALLEL` even higher

- Locking files isn't an issue here

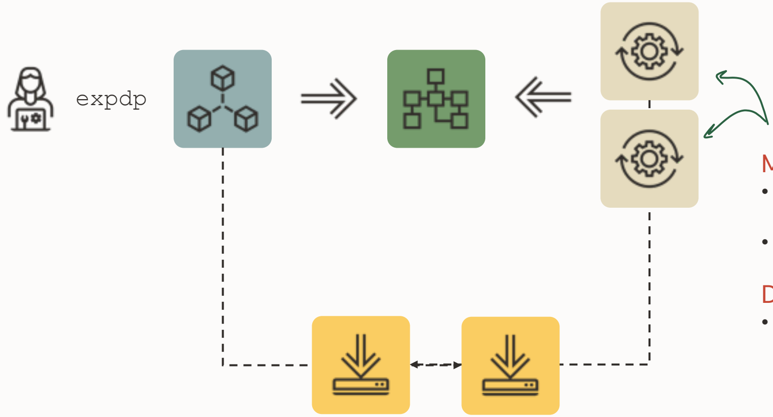


You forgot to set `PARALLEL`?
You can change it at runtime

What are Data Pump workers doing?
And how gets work assigned to each worker?

Parallel | Data Pump Worker: expdp

What does a worker process do during `expdp`?



Metadata

- Collect object metadata with `DBMS_METADATA`
- Write metadata to dumpfile as XML

Data

- Unload data via the Data Layer

Parallel | How export works

- Export (and network mode import) work in multiple phases

- Metadata
  - Each object path is a metadata work unit
  - Workers begin collecting metadata for export
  - Metadata unloaded in any order with parallel worker processes

- Data
  - Metadata for `TABLE_DATA` items contain a size estimate that drives order of export
  - Data work items are scheduled as worker processes become available.

Pro Tip: current dictionary and object statistics are crucial to data pump export performance!

Parallel | How is work assigned to parallel Worker processes?

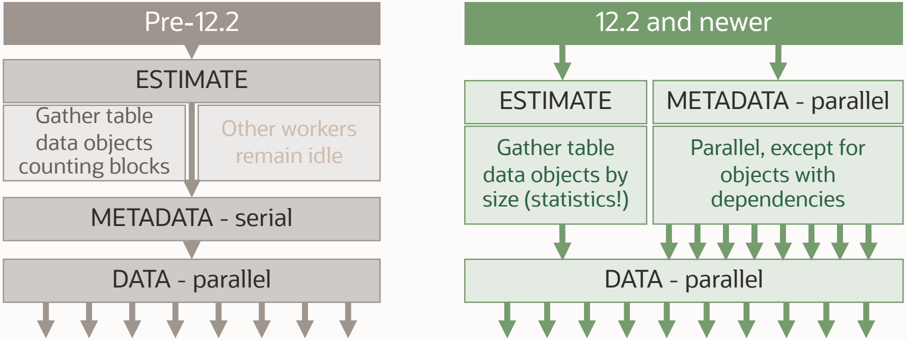
- `TABLE_DATA` work unit is:
  - Subpartition for subpartitioned tables
  - Partition for partitioned tables
  - Table for non-partitioned tables
  - Specify all data for the table as one work unit, regardless of partitioning w/

    ```
    DATA_OPTIONS=GROUP_PARTITION_TABLE_DATA
    ```

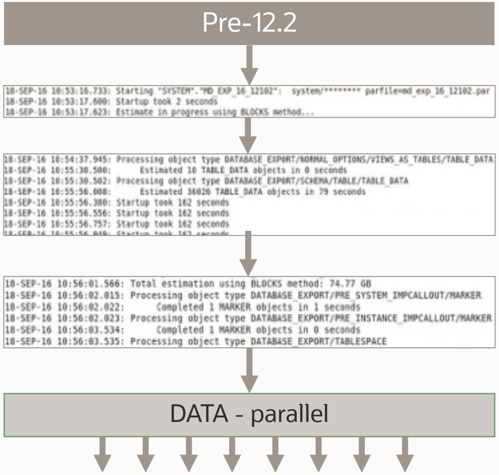
- The metadata work unit is a single object path

Parallel | Metadata Export - Comparison

- Since 12.2: Metadata export happens concurrently with estimate phase for table data
- Most metadata & data objects are exported in parallel when `PARALLEL=2` or greater


Pre-12.2
ESTIMATE
Gather table data objects counting blocks
Other workers remain idle
METADATA - serial
DATA - parallel
12.2 and newer
ESTIMATE
METADATA - parallel
Gather table data objects by size (statistics!)
Parallel, except for objects with dependencies
DATA - parallel

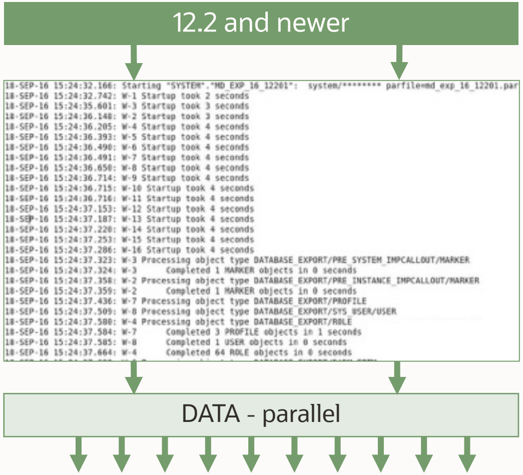
Parallel | Metadata Export - Logfiles



```
18-SEP-16 10:53:16.733: Starting "SYSTEM"."MD_EXP_16_12102":  system/******** parfile=md_exp_16_12102.par
18-SEP-16 10:53:17.600: Startup took 2 seconds
18-SEP-16 10:53:17.623: Estimate in progress using BLOCKS method...
```

```
18-SEP-16 10:54:37.945: Processing object type DATABASE_EXPORT/NORMAL_OPTIONS/VIEWS_AS_TABLES/TABLE_DATA
18-SEP-16 10:55:30.500:      Estimated 10 TABLE_DATA objects in 0 seconds
18-SEP-16 10:55:30.502: Processing object type DATABASE_EXPORT/SCHEMA/TABLE/TABLE_DATA
18-SEP-16 10:55:56.008:      Estimated 36026 TABLE_DATA objects in 79 seconds
18-SEP-16 10:55:56.380: Startup took 162 seconds
18-SEP-16 10:55:56.556: Startup took 162 seconds
18-SEP-16 10:55:56.757: Startup took 162 seconds
```

```
18-SEP-16 10:56:01.566: Total estimation using BLOCKS method: 74.77 GB
18-SEP-16 10:56:02.015: Processing object type DATABASE_EXPORT/PRE_SYSTEM_IMPCALLOUT/MARKER
18-SEP-16 10:56:02.022:      Completed 1 MARKER objects in 1 seconds
18-SEP-16 10:56:02.023: Processing object type DATABASE_EXPORT/PRE_INSTANCE_IMPCALLOUT/MARKER
18-SEP-16 10:56:03.534:      Completed 1 MARKER objects in 0 seconds
18-SEP-16 10:56:03.535: Processing object type DATABASE_EXPORT/TABLESPACE
```



```
18-SEP-16 15:24:32.166: Starting "SYSTEM"."MD_EXP_16_12201":  system/******** parfile=md_exp_16_12201.par
18-SEP-16 15:24:32.742: W-1 Startup took 2 seconds
18-SEP-16 15:24:35.601: W-3 Startup took 3 seconds
18-SEP-16 15:24:36.148: W-2 Startup took 3 seconds
18-SEP-16 15:24:36.205: W-4 Startup took 4 seconds
18-SEP-16 15:24:36.393: W-5 Startup took 4 seconds
18-SEP-16 15:24:36.490: W-6 Startup took 4 seconds
18-SEP-16 15:24:36.491: W-7 Startup took 4 seconds
18-SEP-16 15:24:36.650: W-8 Startup took 4 seconds
18-SEP-16 15:24:36.714: W-9 Startup took 4 seconds
18-SEP-16 15:24:36.715: W-10 Startup took 4 seconds
18-SEP-16 15:24:36.716: W-11 Startup took 4 seconds
18-SEP-16 15:24:37.153: W-12 Startup took 4 seconds
18-SEP-16 15:24:37.187: W-13 Startup took 4 seconds
18-SEP-16 15:24:37.220: W-14 Startup took 4 seconds
18-SEP-16 15:24:37.253: W-15 Startup took 4 seconds
18-SEP-16 15:24:37.286: W-16 Startup took 4 seconds
18-SEP-16 15:24:37.323: W-3 Processing object type DATABASE_EXPORT/PRE_SYSTEM_IMPCALLOUT/MARKER
18-SEP-16 15:24:37.324: W-3      Completed 1 MARKER objects in 0 seconds
18-SEP-16 15:24:37.358: W-2 Processing object type DATABASE_EXPORT/PRE_INSTANCE_IMPCALLOUT/MARKER
18-SEP-16 15:24:37.359: W-2      Completed 1 MARKER objects in 0 seconds
18-SEP-16 15:24:37.436: W-7 Processing object type DATABASE_EXPORT/PROFILE
18-SEP-16 15:24:37.509: W-8 Processing object type DATABASE_EXPORT/SYS_USER/USER
18-SEP-16 15:24:37.580: W-4 Processing object type DATABASE_EXPORT/ROLE
18-SEP-16 15:24:37.584: W-7      Completed 3 PROFILE objects in 1 seconds
18-SEP-16 15:24:37.585: W-8      Completed 1 USER objects in 0 seconds
18-SEP-16 15:24:37.664: W-4      Completed 64 ROLE objects in 0 seconds
```

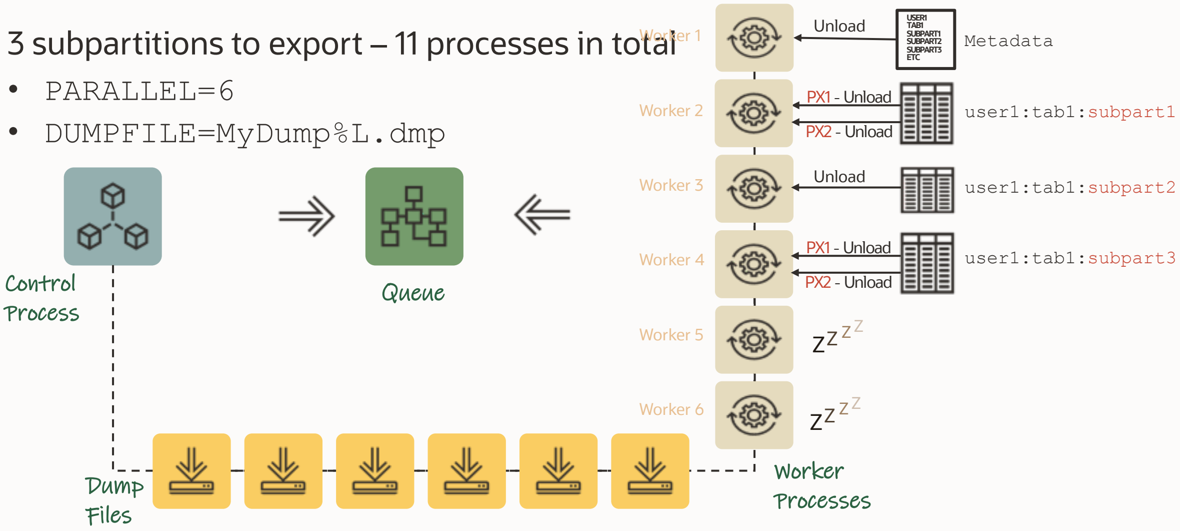
Pro Tip

Ensure statistics on objects and SYS/SYSTEM are current

Parallel | Export Example

3 subpartitions to export – 11 processes in total

- PARALLEL=6
- DUMPFILE=MyDump%L.dmp



Data Pump: 2 kinds of Parallelism for Dumpfiles

- Inter-partition parallelism
  - For tables with partitions below a particular threshold in size
  - One connection and worker per partition or sub partition

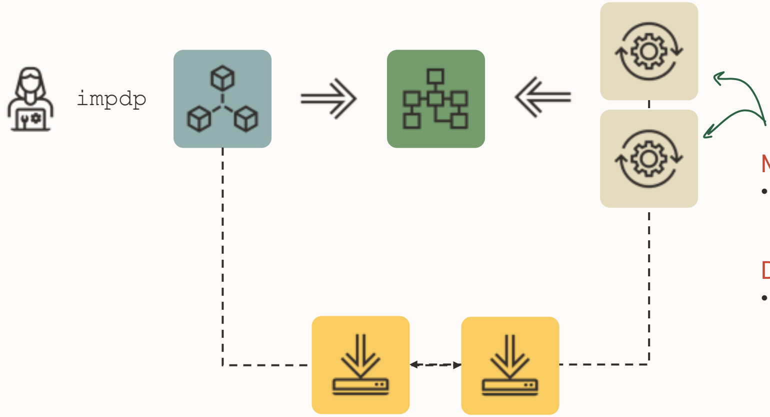
- Parallel query (PQ) intra-partition parallelism
  - Invoked for larger tables or partitions
  - Worker process is the query coordinator, and not included in the DOP limit

Everything Parallel

Import

Parallel | Data Pump Worker: impdp

What does a worker process do during `impdp`?



Metadata
- Convert metadata from XML into DDL and PL/SQL using `DBMS_METADATA`

Data
- Load data via the Data Layer

Parallel | How import works

- No estimate phase
- Reading from Control Table in the dump file
  - Ordering happens strictly by object path
- Metadata (except indexes and package bodies)
  - Mostly in parallel
- Data
  - Loading in parallel with multiple workers and maybe PQ processes
- Remaining metadata
  - Indexes and package bodies

Pro Tip: If you have very large indexes, a manual index built-up via script may be better

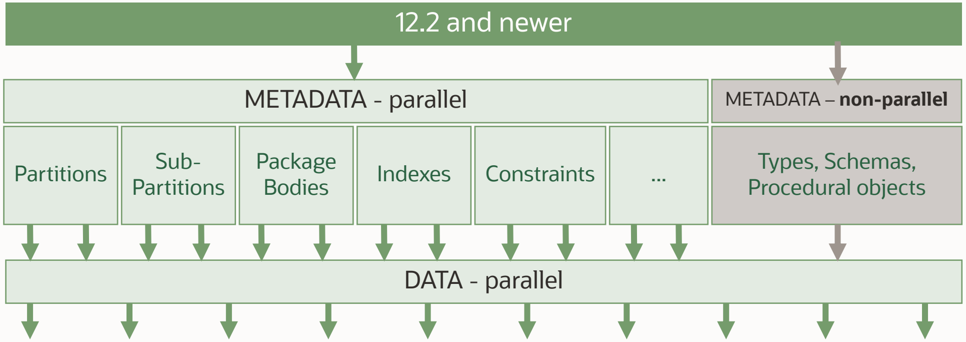
Pro Tip: MOS Note: 1288037.1 - How To Relate DataPump PARALLEL Parameter To Parallel Query Slaves

Parallel metadata import even works when export wasn't done in parallel e.g., in a release before Oracle 12.2

Parallel | Metadata Import

Since 12.2: Metadata import happens concurrently

- Most metadata & data objects are imported in parallel when `PARALLEL=2` or greater

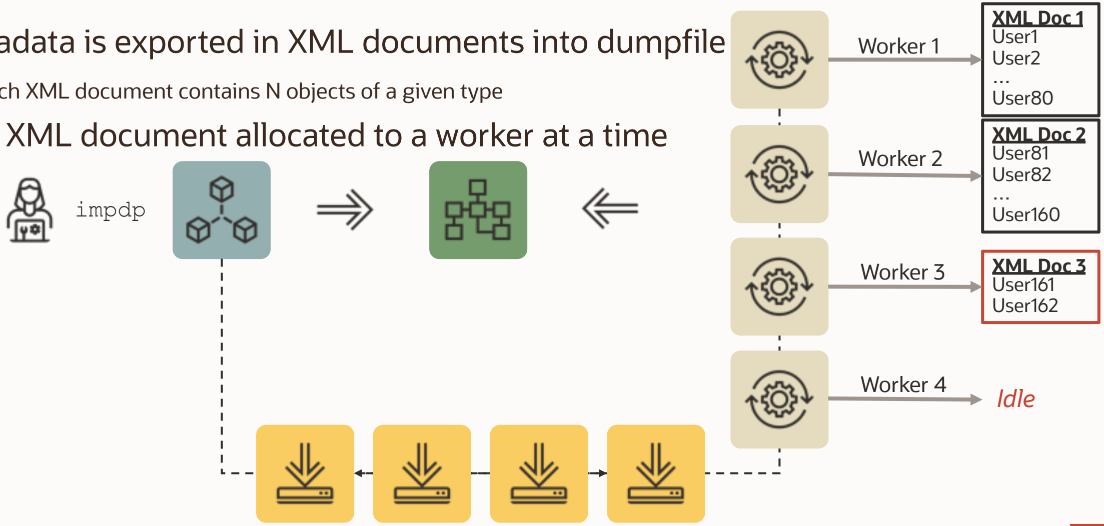


Parallel | Metadata Import - Internals

Metadata is exported in XML documents into dumpfile

- Each XML document contains N objects of a given type

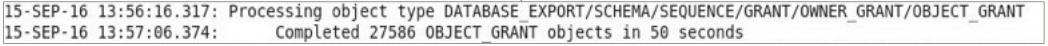
One XML document allocated to a worker at a time



Parallel | Metadata Import - Logfiles

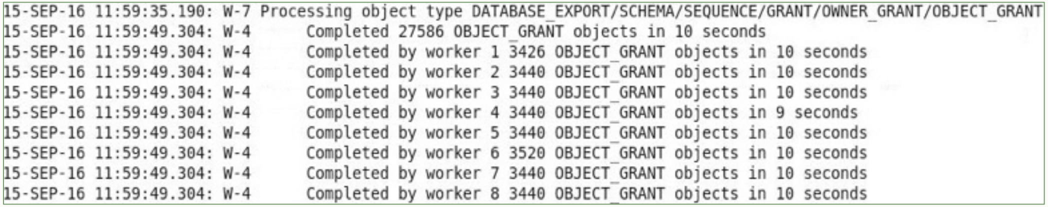
Comparison with `PARALLEL=8` for 27586 object grants - `METRICS=Y`

50 sec Pre-12.2



```
15-SEP-16 13:56:16.317: Processing object type DATABASE_EXPORT/SCHEMA/SEQUENCE/GRANT/OWNER_GRANT/OBJECT_GRANT
15-SEP-16 13:57:06.374:      Completed 27586 OBJECT_GRANT objects in 50 seconds
```

14 sec 12.2 and newer



```
15-SEP-16 11:59:35.190: W-7 Processing object type DATABASE_EXPORT/SCHEMA/SEQUENCE/GRANT/OWNER_GRANT/OBJECT_GRANT
15-SEP-16 11:59:49.304: W-4      Completed 27586 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 1 3426 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 2 3440 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 3 3440 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 4 3440 OBJECT_GRANT objects in 9 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 5 3440 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 6 3520 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 7 3440 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 8 3440 OBJECT_GRANT objects in 10 seconds
```

Parallel | Performance Comparison Example

EBS test database

- Import time in seconds
- `PARALLEL=32`

Object Type	*Count*	*Elapsed (sec) 11.2.0.4*	*Elapsed (sec) 12.2.0.1*
OBJECT_GRANT (owner)	27,586	49	22
SYNONYM	43,254	105	44
TYPE	4,364	108	110
PROCACT_SCHEMA	606	198	175
TABLE	33,164	923	248
OBJECT_GRANT (table)	358,649	541	157
INDEX	53,190	**6721**	272
PACKAGE	53,217	424	54
VIEW	34,690	538	184
PACKAGE BODY	52,092	1363	959

Photo by Jason Smith on Unsplash

Everything Parallel?

No parallelism

Network import does not support loading metadata in parallel

No parallelism on BasicFile LOBs

👍 Convert *old* BasicLOBs to SecureFile LOBs

No parallelism for metadata export and import with Transportable Tablespaces

Parallel metadata export and import with Transportable Tablespace is added in Oracle Database 21c

deep dive
DATA PUMP
with development

Best Practices

Essentials

Extreme

Advanced

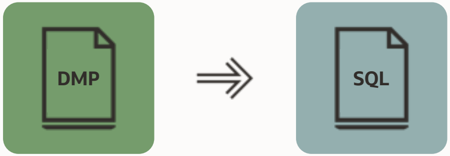
SQLFILES

You can extract metadata information from a dump file using parameter `SQLFILE`

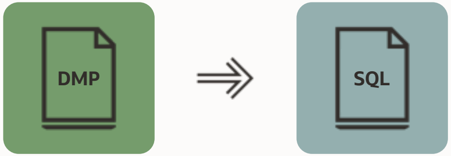
SQLFILE | Generate SQL Statements

Generate DDLs that `impdp` will execute

```
$ more import.par
...
sqlfile=all_statements.sql
...

$ impdp system parfile=import.par
```



SQLFILE | Generate SQL Statements



```
CREATE USER "TPCC" IDENTIFIED BY VALUES '...'
      DEFAULT TABLESPACE "TPCCTAB"
      TEMPORARY TABLESPACE "TEMP";
GRANT UNLIMITED TABLESPACE TO "TPCC";
GRANT "CONNECT" TO "TPCC";
GRANT "RESOURCE" TO "TPCC";
DECLARE
  TEMP_COUNT NUMBER;
  SQLSTR VARCHAR2(200);
BEGIN
  SQLSTR := 'ALTER USER "TPCC" QUOTA UNLIMITED ON "TPCCTAB"';
  EXECUTE IMMEDIATE SQLSTR;
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE = -30041 THEN
      SQLSTR := 'SELECT COUNT(*) FROM USER_TABLESPACES
              WHERE TABLESPACE_NAME = ''TPCCTAB'' AND CONTENTS =
''TEMPORARY''';
      EXECUTE IMMEDIATE SQLSTR INTO TEMP_COUNT;
      IF TEMP_COUNT = 1 THEN RETURN;
      ELSE RAISE;
      END IF;
    ELSE
      RAISE;
    END IF;
END;
/
```

Example

Creating big indexes

SQLFILE | Example - Creating Indexes

Imagine a schema

Tables	
Small tables	10.000
Big tables	1

Indexes	
Small indexes	10.000
Big indexes	1

SQLFILE | Example - Creating Indexes

Data Pump creates indexes with parallel degree 1

Many indexes are created simultaneously

Very efficient for many small indexes

Very <u>in</u>efficient for large indexes

SQLFILE | Example

Data Pump creates
small indexes

You create big indexes
with desired parallel degree

SQLFILE | Example

Find indexes of interest

```
SQL> select   segment_name, round(bytes/1024/1024/1024) as GB
     from     user_segments
     where    segment_type='INDEX'
     order by GB desc;
```

```
$ cat import.par
...
exclude=INDEX:"='BIG1','BIG2','BIG3'"
...

impdp ... parfile=import.par
```

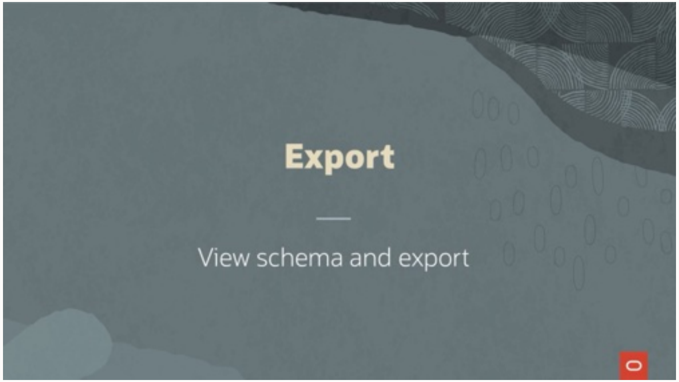
SQLFILE | Example

Generate metadata for big indexes

```
$ cat import-sqlfile.par
...
include=INDEX:"='BIG1','BIG2','BIG3'"
sqlfile=index.sql
...

impdp ... parfile=import-sqlfile.par
```

```
SQL> CREATE INDEX BIG1 .... PARALLEL n;
SQL> ALTER INDEX INDEX BIG1 .... PARALLEL 1;
...
```

SQLFILE | Example



Watch on YouTube

You can also get index definition from DBMS_METADATA.GET_DDL

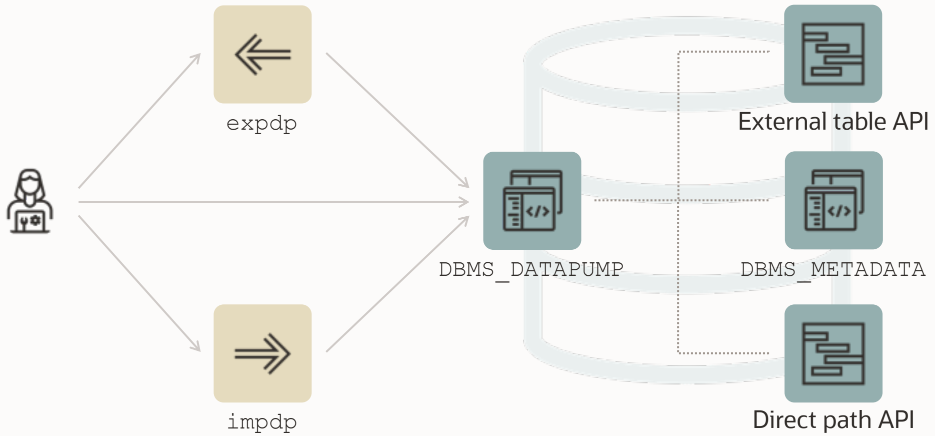
Photo by Louis Hansel on Unsplash

DBMS_DATAPUMP

Usage

`DBMS_DATAPUMP` is
a supported and documented API

DBMS_DATAPUMP | Overview



DBMS_DATAPUMP | API

The Data Pump API (`DBMS_DATAPUMP`) is used many places:

- Zero Downtime Migration
- Enterprise Manager
- SQL Developer
- SQLcl

  ...

You can use it as well,
it is documented and supported

DBMS_DATAPUMP | API

Ideas:

- Use Data Pump functionality without installing a client
- Schedule export or imports using `DBMS_SCHEDULER`
- Dynamically build Data Pump jobs
- Integrate into automation tools (Ansible, Puppet)
- Accessible via ORDS / REST API as well
- Rename schema using a loopback database link
- Take a snapshot of a schema during application development

DBMS_DATAPUMP | Comparison

Client

```
expdp directory=mydir \
   logfile=exp.log \
   dumpfile=exp%u.dmp \
   schemas=app \
   parallel=4 \
   metrics=y \
   logtime=all
```

API

```
h1 := DBMS_DATAPUMP.OPEN(
        operation => 'EXPORT',
       job_mode => 'SCHEMA',
       remote_link => null,
       job_name => 'MY_JOB',
       version => null);

-- Create a Data Pump job to do a schema
-- export. Give it a meaningful name
```

DBMS_DATAPUMP | Comparison

Client

```
expdp directory=mydir \
   logfile=exp.log \
   dumpfile=exp%u.dmp \
   schemas=app \
   parallel=4 \
   metrics=y \
   logtime=all
```

API

```
DBMS_DATAPUMP.METADATA_FILTER(
   handle => h1,
   name => 'SCHEMA_EXPR',
   value => 'IN (''APP'')');

-- Specify the schema to be exported. We let
-- the object_path parameter default in this
-- call, so this applies to all objects in
-- the job
```

DBMS_DATAPUMP | Comparison

Client

```
expdp directory=mydir \
   logfile=exp.log \
   dumpfile=exp%u.dmp \
   schemas=app \
   parallel=4 \
   metrics=y \
   logtime=all
```

API

```
DBMS_DATAPUMP.ADD_FILE(
   handle => h1,
   filename => 'exp%u.dmp',
   directory => 'MYDIR',
   filetype=>DBMS_DATAPUMP.KU$_FILE_TYPE_DUMP_FILE);

-- Specify the dumpfile for the job using a
-- wildcard. The directory object must be
-- supplied for each file added to the job
-- FILETYPE defaults to dumpfile but we
-- specify it anyway to be clear
```

DBMS_DATAPUMP | Comparison

Client

```
expdp directory=mydir \
   logfile=exp.log \
   dumpfile=exp%u.dmp \
   schemas=app \
   parallel=4 \
   metrics=y \
   logtime=all
```

API

```
DBMS_DATAPUMP.ADD_FILE(
   handle => h1,
   filename => 'exp.log',
   directory => 'MYDIR',
   filetype=>DBMS_DATAPUMP.KU$_FILE_TYPE_LOG_FILE);

-- Specify the log file for the job. The directory
-- object must be supplied for each file added to
-- the job.
```

DBMS_DATAPUMP | Comparison

Client

```
expdp directory=mydir \
   logfile=exp.log \
   dumpfile=exp%u.dmp \
   schemas=app \
   parallel=4 \
   metrics=y \
   logtime=all
```

API

```
DBMS_DATAPUMP.SET_PARALLEL(
   handle => h1,
   degree => 4 );

-- Set the parallelism for the job
-- Or get a little creative

select value into parallel_degree
from   v$parameter
where  name='cpu_count';
DBMS_DATAPUMP.SET_PARALLEL(
   handle => h1,
   degree => parallel_degree);
```

DBMS_DATAPUMP | Comparison

Client

```
expdp directory=mydir \
   logfile=exp.log \
   dumpfile=exp%u.dmp \
   schemas=app \
   parallel=4 \
   metrics=y \
   logtime=all
```

API

```
DBMS_DATAPUMP.SET_PARAMETER(
   handle => h1,
   name => 'METRICS',
   value => 1);

DBMS_DATAPUMP.SET_PARAMETER(
   handle => h1,
   name => 'LOGTIME',
   value => 'ALL');

-- set other job parameters
```

DBMS_DATAPUMP | Comparison

Client	API
	see code below

```
DBMS_DATAPUMP.START_JOB (
   handle => h1);

-- now start the job
-- wait for it to complete

DBMS_DATAPUMP.WAIT_FOR_JOB (
   handle => h1,
   job_state);
```

👍 Use 10046 trace to generate DBMS_DATAPUMP calls

Data Pump | Generate PL/SQL

1. Enable SQL trace on a test database

```
SQL> alter system
     set event='10046 trace name context forever, level 4';
```

2. Execute your Data Pump command

```
$ impdp system ... parfile=import.par
```

3. Examine the trace file

```
$ vi ORCL_ora_12345.trc
```

Pro tip: Grep for *DBMS_DATAPUMP* to find the right trace file

The documentation has many good examples on using `DBMS_DATAPUMP`

Restartability

Export and Import

Data Pump export and import jobs can be stopped and restarted again

Restart | Export

Export can be restarted after the ESTIMATE phase has been completed

- Tracked in the Control Table
- Workers create/update records with COMPLETION_TIME
- Restart: Workers check for records with missing COMPLETION_TIME

OBJECT_TYPE	START_TIME	COMPLETION_TIME
TABLESPACE	12-SEP-2021:9:04.01	12-SEP-2021:9:05.23
USER	12-SEP-2021:9:05.27	

- Example
  - USER object is incomplete
  - Will be removed and restarted

Restart | Import

Import can be restarted using the Control Table

- Workers track import status via `STATE` and `STATUS`

OBJECT	OBJECT_SCHEMA	OBJECT_NAME	PROCESSING_STATE	PROCESSING_STATUS
TABLE	SCOTT	EMP	W	C
TABLE	SCOTT	DEPT	U	C
INDEX	SCOTT	IDX1_EMP	R	C
INDEX	SCOTT	IDX1_DEPT	R	C

- R = objects were Retrieved (exported)
- C = objects are Current (successfully imported
- W = objects are Written (imported)
- U = objects are Unknown (import started but did not finish)

Interactive Command Mode

Usage

Use Interactive Command Mode

Use Interactive Command Mode

- ✓ Monitor & trace performance
- ✓ Change parameters
- ✓ Change attributes
- ✓ Start/stop jobs and workers

1. Press `CTRL+C` in Data Pump session

2. Attach from different Data Pump session

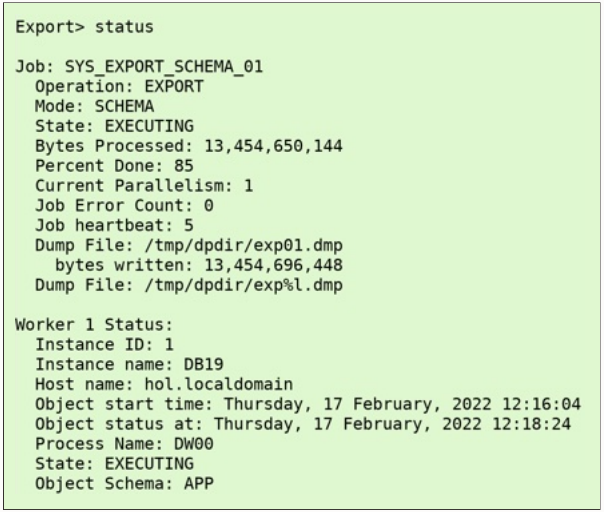
```
$ expdp .... attach=<job name>
$ impdp .... attach=<job name>
```

Pro tip: get the logfile updates you can use tail -f or its equivalent

Interactive Command Mode | Overview

Interact with a running job

- Changing parameters
- Changing attributes
- Monitoring
- Starting/stopping jobs and workers



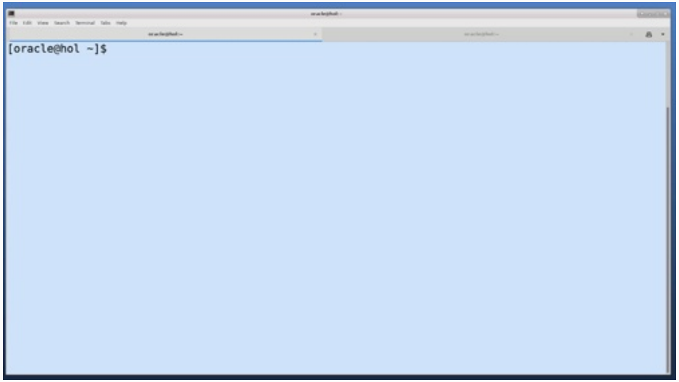
```
Export> status

Job: SYS_EXPORT_SCHEMA_01
  Operation: EXPORT
  Mode: SCHEMA
  State: EXECUTING
  Bytes Processed: 13,454,650,144
  Percent Done: 85
  Current Parallelism: 1
  Job Error Count: 0
  Job heartbeat: 5
  Dump File: /tmp/dpdir/exp01.dmp
    bytes written: 13,454,696,448
  Dump File: /tmp/dpdir/exp%l.dmp

Worker 1 Status:
  Instance ID: 1
  Instance name: DB19
  Host name: hol.localdomain
  Object start time: Thursday, 17 February, 2022 12:16:04
  Object status at: Thursday, 17 February, 2022 12:18:24
  Process Name: DW00
  State: EXECUTING
  Object Schema: APP
```

Different commands are available for exports and imports.

Command	Mode	Description
`PARALLEL=n`	Both	Change the parallelism for current job. Increases almost immediately.
`STATUS`	Both	Get the job and worker status. Includes Operation, Mode, State, Percent Done, and Current Parallelism .
`STATUS=120`	Both	As above but refreshes every 120 second
`FILESIZE=n`	Export	Changes the file size (in bytes) of the dump files. Optionally specify denominator, e.g., FILESIZE=5G
`ADD_FILE=name`	Export	Adds an additional dump file. Or a dump file pattern, e.g., ADD_FILE=more_files%L.dmp
`TRACE=nnn`	Both	Adds tracing, see MOS ID 286496.1 for details

See the documentation for More commands

Interactive Command Mode | Demo



```
[oracle@hol ~]$
```

Watch on YouTube

Photo by Mitchell Luo on Unsplash

Troubleshooting

In case of an issue ...

Data Pump is not doing what you'd expect?
What should you check and collect?

Troubleshooting | Step-by-step

Log files

Always use `METRICS=YES` and `LOGTIME=ALL`

Enabling diagnostic information does not generate overhead

Troubleshooting | Log files

Log files are essential

- Use `METRICS=YES` and `LOGTIME=ALL`
  - Without log parameters:

```
Processing object type DATABASE_EXPORT/FINAL_POST_INSTANCE_IMPCALLOUT/MARKER
Processing object type DATABASE_EXPORT/AUDIT_UNIFIED/AUDIT_POLICY_ENABLE
Processing object type DATABASE_EXPORT/POST_SYSTEM_IMPCALLOUT/MARKER
. . exported "SYS"."KU$_USER_MAPPING_VIEW"                5.890 KB      25 rows
. . exported "SYSTEM"."REDO_DB"                           25.59 KB       1 rows
```

```
DATABASE_EXPORT/FINAL_POST_INSTANCE_IMPCALLOUT/MARKER
02-NOV-21 19:43:56.064: W-1      Completed 1 MARKER objects in 0 seconds
02-NOV-21 19:43:59.171: W-1 Processing object type DATABASE_EXPORT/AUDIT_UNIFIED/AUDIT_POLICY_ENABLE
02-NOV-21 19:43:59.195: W-1      Completed 2 AUDIT_POLICY_ENABLE objects in 0 seconds
02-NOV-21 19:43:59.380: W-1 Processing object type DATABASE_EXPORT/POST_SYSTEM_IMPCALLOUT/MARKER
02-NOV-21 19:43:59.387: W-1      Completed 1 MARKER objects in 0 seconds
02-NOV-21 19:43:59.830: W-1 . . exported "SYS"."KU$_USER_MAPPING_VIEW"    5.890 KB  25 rows in 0 seconds using external_table
02-NOV-21 19:43:59.923: W-1 . . exported "SYSTEM"."REDO_DB"               25.59 KB   1 rows in 0 seconds using direct_path
```

Troubleshooting | Log files

Check `alert.log` and upload it with an SR

```
2022-02-21T11:31:23.315021+01:00
db_recovery_file_dest_size of 18432 MB is 1.23% used. This is a
user-specified limit on the amount of space that will be used by this
database for recovery-related files, and does not reflect the amount of
space available in the underlying filesystem or ASM diskgroup.
2022-02-21T11:31:25.810983+01:00
DM00 started with pid=80, OS id=17226, job DPUSER.SYS_EXPORT_SCHEMA_01
2022-02-21T11:31:56.980017+01:00
Thread 1 advanced to log sequence 20 (LGWR switch),  current SCN: 6660216
  Current log# 2 seq# 20 mem# 0: /u02/oradata/DB19/redo02.log
2022-02-21T11:31:57.197532+01:00
ARC1 (PID:16810): Archived Log entry 3 added for T-1.S-19 ID 0x31223092 LAD:1
2022-02-21T11:32:01.650969+01:00
TABLE SYS.WRP$_REPORTS: ADDED INTERVAL PARTITION SYS_P865 (4435) VALUES LESS THAN (TO_DATE(' 2022-02-22 01:00:00',
'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLE SYS.WRP$_REPORTS_DETAILS: ADDED INTERVAL PARTITION SYS_P866 (4435) VALUES LESS THAN (TO_DATE(' 2022-02-22
01:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
TABLE SYS.WRP$_REPORTS_TIME_BANDS: ADDED INTERVAL PARTITION SYS_P869 (4434) VALUES LESS THAN (TO_DATE(' 2022-02-21
01:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIAN'))
2022-02-21T11:32:12.822559+01:00
ALTER SYSTEM SET streams_pool_size=256M SCOPE=BOTH;
```

Troubleshooting | Log files

Check for Data Pump trace files in

```
Trace file /u01/app/oracle/diag/rdbms/db19/DB19/trace/DB19_dm00_17468.trc
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.14.0.0.0
Build label:    RDBMS_19.14.0.0.0DBRU_LINUX.X64_211224.3
ORACLE_HOME:    /u01/app/oracle/product/19
System name:    Linux
Node name:      hol.localdomain
Release:        5.4.17-2136.302.7.2.1.el7u
Version:        #2 SMP Tue Jan 18 13:44:44
Machine:        x86_64
Instance name: DB19
Redo thread mounted by this instance: 1
Oracle process number: 58
Unix process pid: 17468, image: oracle@hol


*** 2022-02-21T11:33:25.374300+01:00
*** SESSION ID:(253.19643) 2022-02-21T11:3
*** CLIENT ID:() 2022-02-21T11:33:25.37431
*** SERVICE NAME:(SYS$USERS) 2022-02-21T11
*** MODULE NAME:(Data Pump Master) 2022-02
*** ACTION NAME:(SYS_EXPORT_SCHEMA_01) 202
*** CLIENT DRIVER:() 2022-02-21T11:33:25.374327+01:00
```

```
=================== skgfqio Request Dump =======================
OSD Context: aiopend=0, aiodone=0, limfsiz=4294967295l, sigwinchslot=0
Request flags: READ
- - - - skgfrrq request element 1 - - - -
BLOCKNO = 1
IOV: addr=0x0x6ef687d8, fib=0x0x6d0d2478, maxaio=0, seal=0x45726963,
fd=260
     fsync required?=TRUE, offset=18446744073709551615, aiopend=0
FIB: addr=0x0x6d0d2478, lblksiz=0, ora ftype=18, pblksiz=512, filsiz=1
     maxvec=16, fname=/home/oracle/dp/export.log, serr=0, seal=0x45726963
     fstype=0x58465342, unix ftype=0x81a4, last
block=18446744073709551615
IOSB: addr=0x0x7f0da829dc38, status=3, time=0, qstatus=8,AIO start
time=139696632618072
err=27072 errno=25 ose[0]=4 ose[1]=1 ose[2]=333
BUFFER: addr=0x0x7f0da76b2000, len=4096
```

Background Process

CONTROL PROCESS

Typically one: `dm00`

```
DB19_dm00_17468.trc
```

WORKERS

Typically many: `dw`*`nn`*

```
DB19_dw00_17469.trc
DB19_dw01_17470.trc
DB19_dw02_17471.trc
DB19_dw03_17472.trc
```

Troubleshooting | Step-by-step

Log files

Database Views

Troubleshooting | Database Views

Monitor a Data Pump process in `DBA_DATAPUMP_JOBS`

- MOS Note: 1471766.1 - How To Monitor The Progress Of Data Pump Jobs
- Use parameter `JOB_NAME` with expdp and impdp

```
SQL> select * from DBA_DATAPUMP_JOBS;

OWNER_NAME   JOB_NAME   OPERATION   JOB_MODE   STATE  DEGREE  ATTACHED  DATAPUMP_SESSIONS
-----------  ---------- ----------- ---------- ------ ------- --------- -----------------
SYS          MYEXPDP1   EXPORT      FULL       EXECUT       1         1                 3
```

Troubleshooting | Database Views

Monitor a Data Pump process in `DBA_DATAPUMP_SESSIONS`

- MOS Note: 1528301.1 - Finding Out The Current SQL Statement A Data Pump Process Is Executing
  - Use the script from MOS Note: 1528301.1 to:
    - Diagnose possible hangs
    - Slow Data Pump processes

Troubleshooting | Database Views

Monitor a Data Pump process in `V$SESSION_LONGOPS`

- MOS Note: 455720.1 - How can we monitor a DataPump Job's Progress?
- Use parameter `JOB_NAME` with expdp and impdp

```
select sid, serial#, sofar, totalwork
from   V$SESSION_LONGOPS
where  opname = '<your export job name>' and
       sofar != totalwork;
```

  - `sofar`:
    Shows how much work in MB has been done so far in relation to `totalwork`
  - `totalwork`:
    Shows the total amount of work in MB

Monitoring

Important MOS notes:

How To Monitor The Progress Of Datapump Jobs (Doc ID 1471766.1)

Finding Out The Current SQL Statement A Data Pump Process Is Executing (Doc ID 1528301.1)

How can we monitor a DataPump Job's Progress? (Doc ID 455720.1)

Troubleshooting | Step-by-step

Log files

Database Views

Tracing

TRACING

Command Line
Using parameter TRACE

Interactive Console
Using command **TRACE**

Trace Event
Adding specific event to SPFile

Tracing | Best Practice

- Requires privileged user or role
  - `DBA`
  - `EXP_FULL_DATABASE`
  - `IMP_FULL_DATABASE`

- Ensure `MAX_DUMP_FILE_SIZE` is large enough to capture the trace (default=unlimited)

- Most important `TRACE` bitmaps:
  - `1FF0300` Recommended Tracing
  - `1FFF0300` Full Tracing
  - For a comprehensive list and further explanation, see MOS Note: 286496.1

Data Pump trace is written to `dm`*nn* and `dw`*nn* trace files

- Located in trace directory in `diagnostic_dest`

Tracing

```
SQL> # Data Pump specific trace
SQL> alter system set events = '39089 trace name context forever, level 0x300' ;

SQL> # SQL trace on Data Pump processes and parallel processes
SQL> alter system set events
     'sql_trace {process:pname=dw | process:pname=dm | process:pname=p0} level=8';
```

Tracing

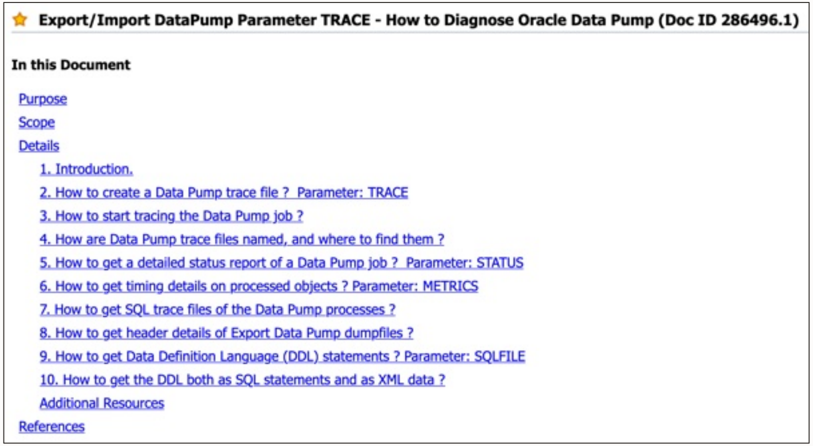
Important MOS notes:

Export/Import DataPump Parameter TRACE - How to Diagnose Oracle Data Pump
(Doc ID 286496.1)

How To Collect 10046 Trace (SQL_TRACE) Diagnostics for Performance Issues
(Doc ID 376442.1)

Troubleshooting | TRACE

MOS Note: 286496.1 - DataPump Parameter TRACE - How to Diagnose Oracle Data Pump



Export/Import DataPump Parameter TRACE - How to Diagnose Oracle Data Pump (Doc ID 286496.1)

In this Document

Troubleshooting | TRACE

Best practices

- Requires privileged user - DBA role or EXP[/IMP]_FULL_DATABASE Role
- Ensure `MAX_DUMP_FILE_SIZE` is large enough to capture the trace (default=unlimited)

Three options

- TRACE parameter
- TRACE in interactive mode
- TRACE event in SPFILE

Troubleshooting | TRACE

TRACE parameter

- Allows tracing of each individual component of Data Pump
- Bitmap format

```
trace= 1FF0B00
```

- Most important TRACE bitmaps:
  - `1FF0300  – Recommended Tracing`
  - `1FFF0300 – Full Tracing`
  - For a comprehensive list and further explanation, see MOS Note: 286496.1

- Generates two trace files:
  - `<SID>_dm<number>_<process_id>.trc` - Control process trace
  - `<SID>_dw<number>_<process_id>.trc` - Worker trace file (one for each worker)

Troubleshooting | TRACE

TRACE option in interactive mode

- Type `^c` while the job is running
- Add tracing
- Resume job

```
impdp user/password attach=user.imp_job_1 trace=400300


Import> start_job [=SKIP_CURRENT=YES]
```

Troubleshooting | TRACE

TRACE event 39089

- Add event to SPFILE/PFILE or set it via ALTER SYSTEM

```
EVENT="39089 trace name context forever, level 0x300"
```

```
ALTER SYSTEM SET EVENTS = '39089 trace name context forever, level 0x300' ;
ALTER SYSTEM SET EVENTS = '39089 trace name context off' ;
```

- For further explanation, see MOS Note: 286496.1

Use 10046 trace to generate `DBMS_DATAPUMP` calls

Troubleshooting | TRACE

TRACE event 10046

- Multi-purpose SQL trace event
- Usually set for the worker process(es)
- For further explanation, see:
  MOS Note: 376442.1 - How To Collect 10046 Trace (SQL_TRACE) Diagnostics for Performance Issues

Generate PL/SQL

1. Enable SQL trace on a test database

```
SQL> alter system
     set event='10046 trace name context forever, level 4';
```

2. Execute your Data Pump command

```
$ impdp system ... parfile=import.par
```

3. Examine the trace file

```
$ vi ORCL_ora_12345.trc
```

Pro tip: Grep for `DBMS_DATAPUMP` to find the right trace file

Troubleshooting Jobs Using the CONTROL TABLE

Data Pump | Troubleshooting Using the CONTROL TABLE

Control table contents

Some Useful columns in the Control Table

- `Process_order` (consists of positive & negative numbers)
- `Object_type`
- `Object_schema`
- `Object_name`
- `Processing_state`
- `Processing_status`

Data Pump | Troubleshooting Using the CONTROL TABLE

Interesting `Process_order` numbers

Positive process order numbers describe objects that have been exported.

Negative process order numbers describe the job

- `-1/-2 Job state row – contains job status`
- `-5/-6 completion rows – status for each object type`
- `-41/-42 – worker status rows`
- `-51/-52 – data filter rows`
- `-53/-54 – metadata filter rows`
- `-57/-58 – metadata transform rows`
- `-59/-60 – job parameter rows`

Data Pump | Troubleshooting Using the CONTROL TABLE

Examine the control table during the job

- Use parameter `ABORT_STEP` to stop job at a particular point & examine the Control Table

```
ABORT_STEP=-1
```

  - Abort the job after setup & before exporting or importing any objects

```
ABORT_STEP=n
```

  - If `n > 0` Abort the job at the object that is stored in the Control Table with its corresponding process order number

Documentation: Database Utilities Guide

Data Pump | Troubleshooting Using the CONTROL TABLE

Example: What is in my `impdp` job?

```
impdp system/manager DIRECTORY=dpump_dir1 DUMPFILE=expdat.dmp SCHEMAS=hr
ABORT_STEP=-1

     SQL> select object_type, object_schema, object_name from
     SYSTEM.IMP_SCHEMA where process_order > 0 and duplicate = 0 and
     processing_status='C' and processing_state = 'R';

     OBJECT_TYPE       OBJ        OBJECT_NAME

     ----------------- ---        -------------------------

     PROCEDURE          HR        ADD_JOB_HISTORY

     ALTER_PROCEDURE    HR        ADD_JOB_HISTORY

     INDEX              HR        REG_ID_PK

     INDEX              HR        LOC_ID_PK

     INDEX              HR        DEPT_ID_PK
```

Documentation: Database Utilities Guide

Data Pump | Troubleshooting Using the CONTROL TABLE

Keep the control table for a retrospective review after the job ends

The parameter `keep_master=y` retains the `CONTROL TABLE` after the job ends

```
expdp system/manager parallel=5 directory=dpump_dir dumpfile=scott.dmp
keep_master=y
```

Data Pump | Troubleshooting Using the CONTROL TABLE

Retrieve the Control Table from the dumpfile

```
impdp system/manager directory=dpump_dir dumpfile=mydmp.dmp master_only=y
```

Documentation: [Database Utilities Guide]

Data Pump | Troubleshooting Using the CONTROL TABLE

use SQL to get details from the Control Table

What was my expdp command? (example of negative process order number)

```
SQL> Select name, value_t
from SYSTEM.EXPORT_JOB_1
where process_order = -59 and name = ‘CLIENT_COMMAND’;
--------------
CLIENT_COMMAND
system/******** tables=scott.emp directory=dpump_dir dumpfile=ss.dmp
reuse_dumpfiles=y
```

Documentation: Database Utilities Guide

Data Pump | Troubleshooting Using the CONTROL TABLE

How many workers started? (example of negative process order number)

```
select count(*) from hr.sys_import_table_01 where process_order = -42;


COUNT(*)
--------
1
```

Data Pump | Troubleshooting Using the CONTROL TABLE

Did the data unload in parallel? (example of positive process order number)

```
SQL> select m.object_schema, m.object_name, (select
count(*) from system.export_table t

where t.process_order = m.process_order and
t.duplicate!=0) pq_count

from system.sys_export_table_01 m

where m.process_order > 0 and m.object_type='TABLE_DATA'


SCHEMA NAME PQ_COUNT

------ ---- --------

SCOTT  EMP  2
```

Data Pump | Troubleshooting Using the CONTROL TABLE

What object types are left? (example of positive process order number)

```
SQL> select unique object_type_seqno, object_type from
system.sys_import_full_01
where process_order > 0 AND processing_state = ‘R’
and processing_status = ‘C’;


OBJECT_PATH_SEQNO OBJECT_TYPE
----------------- ------------
103                         PROCEDURE
119                         ALTER_PROCEDURE
137                         VIEW
```

Data Pump | Troubleshooting Using the CONTROL TABLE

What's left for the current object? (example of positive process order number)

```
SQL> select object_schema, object_name
 from system.sys_import_full_01
 where process_order > 0 and processing_state = 'R' and
processing_status =  'C' and object_path_seqno = 103;


OBJECT_SCHEMA  OBJECT_NAME
-------------   -----------
HR             ADD_JOB_HISTORY
HR             SECURE_DML
```

Data Pump | Troubleshooting Using the CONTROL TABLE

Get metrics on exported data (example of positive process order number)

```
SQL> select sum(dump_orig_length), processing_state from
"SYSTEM"."SYS_IMPORT_FULL_01" where process_order > 0 and duplicate = 0
and object_type = 'TABLE_DATA' group by processing_state;


SUM(DUMP_ORIG_LENGTH) P

--------------------- -

2525400              R -- exported

2324944              E -- estimated
```

Data Pump | Troubleshooting Using the CONTROL TABLE

Get metrics on imported data (example of positive process order number)

```
SQL> select sum(dump_orig_length), processing_state
 from "SYSTEM"."SYS_IMPORT_FULL_01" where process_order > 0 and
duplicate = 0  and object_type = 'TABLE_DATA' group by processing_state;

SUM(DUMP_ORIG_LENGTH) P
--------------------- -
13408128             W -- already imported
 2525400             R -- to be imported
   24944             X -- excluded
```

Data Pump Use Cases

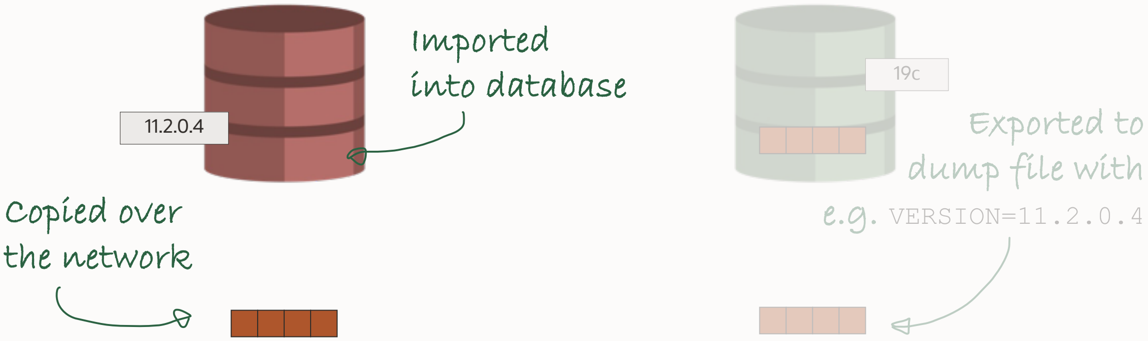
Data Pump | With Data Guard

Import into Data Guard environment - works seamlessly

1. Ensure `STANDBY_FILE_MANAGEMENT=AUTO`
   - Optionally, configure `DB_FILE_NAME_CONVERT` as well
2. Create new PDB from PDB$SEED
   - Propagates automatically to standby database
3. Import with Data Pump
   - Import happens implicitly on standby by redo apply
   - Tablespaces are automatically created

Pro tip: An export job creates a control table in the database so it is not possible to export from an active standby with Active Data Guard

Data Pump | As Fallback



Data Pump | As Fallback

To create a dump file compatible with a lower release

```
version=11.2.0.4
```

Other options are

- `COMPATIBLE`
- `LATEST`

Export/Import DataPump Parameter VERSION -
Compatibility of Data Pump Between Different Oracle Versions (Doc ID 553337.1)

Pro tip: Read more about VERSION in the documentation

Data Pump | TDE

TDE Tablespace Encryption is supported by Data Pump

Dump files can be encrypted using:

- Passphrase (`ENCRYPTION_MODE=PASSWORD`)
- Keystore (`ENCRYPTION_MODE=TRANSPARENT`)

If exporting encrypted database to unencrypted dump file:

- ORA-39173: Encrypted data has been stored unencrypted in dump file set
- Recorded in unified audit trail

Pro tip: Encrypting a Data Pump export requires Advanced Security Option

Data Pump | RAC

Worker processes are spawned on all instances

- Directory must point to shared storage

To keep all workers on same instance

- `CLUSTER=N`
- Directory can exist on local storage

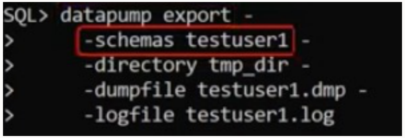
Running into Library Cache Locks on RAC?

'Library Cache Lock' (Cycle) Seen During DataPump Import in 12.2 RAC Environment (Doc ID 2407491.1)

Data Pump | SQLcl

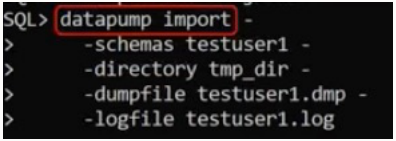
Data Pump support in SQLcl version 21.4

- Export:



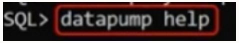
```
SQL> datapump export -
>        -schemas testuser1 -
>        -directory tmp_dir -
>        -dumpfile testuser1.dmp -
>        -logfile testuser1.log
```

- Import:



```
SQL> datapump import -
>        -schemas testuser1 -
>        -directory tmp_dir -
>        -dumpfile testuser1.dmp -
>        -logfile testuser1.log
```

Resources:

- Help in SQLcl: 

```
SQL> datapump help
```

- SQLcl Data Pump part 1
  https://www.youtube.com/watch?v=fvBrfl8cp4s
- SQLcl Data Pump part 2:
  https://www.youtube.com/watch?v=Oqnw-8pZUjE
- Blog:
  https://www.thatjeffsmith.com/archive/2021/12/oracle-sqlcl-datapump/

Data Pump | Goldengate

- Instantiation filtering tells GG the SCN at which each table was exported

- GoldenGate starts capturing at the beginning of the export, but applies changes after the exported SCN
  - If Table A was exported at SCN Y, GG applies changes starting at SCN Y+1.
  - If table B was exported at SCN Z GG applies changes starting at at SCN Z+1.

A PDB can be imported into non-cdb database

Migrate from ADB to on-premises using a preauthenticated URI and perhaps `remap_schema` from the PDB_ADMIN to your on-premises user

Data Pump Marker Objects and Excluding Statistics

Data Pump | Marker Objects

Purpose

- Used by Export & Import to order (sequence) actions for other object types
- Indicate a point during the job where some action is required
- Cannot be an excluded object type b/c they are used for several internal purposes
- Examples:
  - Mark the dependency that package bodies have on package specifications
  - If statistics are included in the export a marker is set to trigger a call to the `DBMS_STATS` package to import statistics on objects imported.
  - If `EXCLUDE=STATISTICS` is specified on export all objects in an object path for statistics are excluded, including the STATISTICS marker

21c New Features

Data Pump, TTS & SQL*Loader

What’s New for Oracle Cloud in Data Pump 21c?

Important for cloud migrations – but also helpful on premises

- Export dumpfiles directly to object store for ADB
- Validate a dumpfile by checksum before using it
- Include and exclude objects in the same export or import operation
- Control index compression during import

Documentation: Database Utilities Guide

Data Pump 21c | Export Cloud Dumpfiles to Object Store

Export into object store from Autonomous Database & on-premises databases

- Export `CREDENTIAL` parameter specifies your object store authentication
- Export `DUMPFILE` parameter specifies the URI for dumpfiles in the object store
- Example:

```
expdp hr DEFAULT_DIRECTORY=dir1 DUMPFILE=https://objectstorage.us-ashburn-
1.oraclecloud.com/n/namespace-string/b/bucketname/o/exp%u.dmp
CREDENTIAL=user-credential schemas=schema_name, exclude=cluster, db_link
parallel=# encryption_pwd_prompt=yes
```

- Details:
  - `CREDENTIAL` expects `DUMPFILE` list of URIs as a comma-delimited string
  - `DEFAULT_DIRECTORY` parameter specifies the location of the local log files
  - `LOGFILE` allows directory object names as part of the file names

Documentation: Database Utilities Guide: `CREDENTIAL`, `DUMPFILE`

Data Pump | Import from Oracle Object Store Dumpfiles to ADB

The differences in recommended parameters for ADB Services

Red text = ADW, ATP-Shared only parameters,   Blue text = ATP-Dedicated only

-
  ```
  impdp admin/password@ADWC1_high
     directory=data_pump_dir
     credential=def_cred_name
     dumpfile=https://objectstorage.us-ashburn-1.oraclecloud.com
     /n/namespace-string/b/bucketname/o/export%u.dmp
     parallel=16
     encryption_pwd_prompt=yes
     transform=segment_attributes:n
     transform=dwcs_cvt_iots:y
      transform=constraint_use_default_index:y
     exclude=cluster,indextype,db_link
     nologfile=yes
  ```

Pro tip: Cloud services evolve quickly, so check the online documentation for the latest syntax and options!

Documentation ADW, ATP-Shared, ATP-Dedicated

DEMO: 21c Export to the Oracle Object Store

```
[work][mike]:
```

Data Pump | Import into ATP-Dedicated using Network Mode import

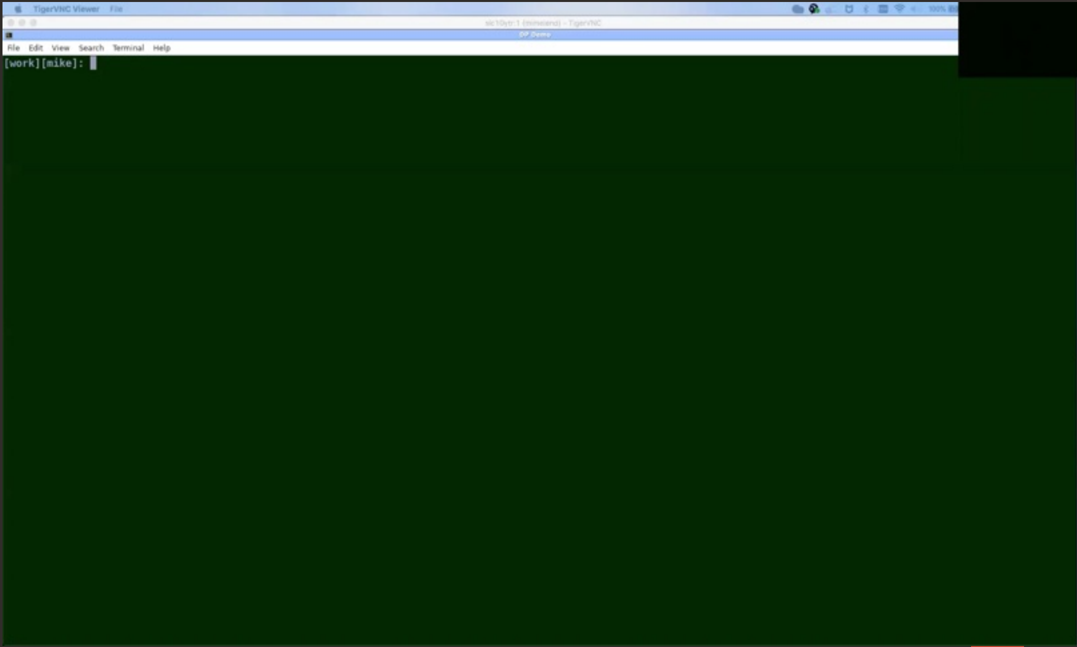
An Oracle Database 19c capability

- Export and import in one step w/o referencing the object store and dumpfiles
- Format:

```
 impdp user/pwd@host_name:port_#/service_name
     schema=schema_name
     network_link=link_name
     parallel=#
     transform=segment_attributes:n
     exclude=cluster
     nologfile=yes
```

Documentation ATP-Dedicated

DEMO: 19c Network Mode import into Oracle ATP-D



```
[work][mike]:
```

Data Pump 21c | Validate a Dumpfile with a Checksum

Confirm dumpfile is valid after a network transfer and has no malicious changes

- Export `CHECKSUM` & `CHECKSUM_ALGORITHM` parameters generate SHA or CRC
- Import `VERIFY_CHECKSUM` parameter uses the checksum to validate dumpfile(s)
- Example:

```
expdp hr DIRECTORY=dpump_dir1 DUMPFILE=hr.dmp CHECKSUM_ALGORITHM=SHA384
impdp hr DIRECTORY=dpump_dir1 DUMPFILE=hr.dmp VERIFY_CHECKSUM=YES
```

Documentation: Database Utilities Guide:

`CHECKSUM`, `CHECKSUM_ALGORITHM`, `VERIFY_CHECKSUM`, `VERIFY_ONLY`

Data Pump 21c | Validate a Dumpfile with a Checksum

Confirm dumpfile is valid after a network transfer and has no malicious changes

- Details:
  - `CHECKSUM_ALGORITHM =[CRC32|SHA256|SHA384|SHA512]`
  - If only `CHECKSUM=YES` is specified then `CHECKSUM_ALGORITHM = SHA256`
  - Must have `COMPATIBLE = 21.0` or higher
  - `VERIFY_ONLY` parameter can be used to verify a dumpfile set without importing
    - VERIFY_ONLY and VERIFY_CHECKSUM are mutually exclusive

Documentation: Database Utilities Guide: `VERIFY_ONLY`

Checksum

```
$ # Calculate a checksum using the designated algorithm
$ # Stored encrypted in dump file header
$ expdp ... checksum_algorithm=sha384

$ # Verify the sum, no import
$ impdp ... verify_only=yes

$ # Verify the checksum and import
$ # Default, if dump file has a checksum
$ impdp ... verify_checksum=yes
```

Data Pump 21c | Include & Exclude objects in Same Job

It’s easier to migrate to Oracle Cloud / on-premises by being more specific

Include and exclude objects within the same export or import job

- Example:

```
expdp hr DIRECTORY=dpump_dir1 DUMPFILE=exp%u.dmp SCHEMAS=hr,oe
include=table exclude=statistics
```

- Details:
  - `INCLUDE` parameter processed first, include all objects identified by the parameter
  - `EXCLUDE` parameter(s) processed next. It removes any objects in the list of include objects

Documentation: Database Utilities Guide: Export: include, exclude Import: include, exclude

Data Pump 21c | Include & Exclude objects in Same Job

Example

```
include=table

exclude=statistics

…
Processing object type SCHEMA_EXPORT/TABLE/TABLE_DATA
Total estimation using BLOCKS method: 448 KB
Processing object type SCHEMA_EXPORT/TABLE/TABLE
Processing object type SCHEMA_EXPORT/TABLE/PRE_TABLE_ACTION
Processing object type SCHEMA_EXPORT/TABLE/GRANT/OWNER_GRANT/OBJECT_GRANT
Processing object type SCHEMA_EXPORT/TABLE/COMMENT
Processing object type SCHEMA_EXPORT/TABLE/INDEX/INDEX
Processing object type SCHEMA_EXPORT/TABLE/CONSTRAINT/CONSTRAINT
Processing object type SCHEMA_EXPORT/TABLE/INDEX/STATISTICS/INDEX_STATISTICS
Processing object type SCHEMA_EXPORT/TABLE/CONSTRAINT/REF_CONSTRAINT
Processing object type SCHEMA_EXPORT/TABLE/TRIGGER
Processing object type SCHEMA_EXPORT/TABLE/STATISTICS/TABLE_STATISTICS
Processing object type SCHEMA_EXPORT/TABLE/POST_TABLE_ACTION
```

Data Pump 21c | Control Import Index Compression

Take control of index compression on import and specify index compression for ADB

- Compress indexes during import with `INDEX_COMPRESSION_CLAUSE` transform
- Example:

```
impdp hr DIRECTORY=dpump_dir1 DUMPFILE=hr.dmp SCHEMAS=hr
TRANSFORM=INDEX_COMPRESSION_CLAUSE:COMPRESS ADVANCED LOW;
```

- Syntax:
  - `INDEX_COMPRESSION_CLAUSE [NONE | compression_clause]`

Documentation: Database Utilities Guide: `TRANSFORM=INDEX_COMPRESSION_CLAUSE`

Data Pump 21c | Remap_Tablespace allows % wildcard

Remap all source tablespaces to one destination tablespace

- `remap_tablespace` accepts a wildcard character for the source tablespace(s).

- Syntax

```
REMAP_TABLESPACE = %:<Permanent_tablespace_name>
```

- Details
  - Maps all permanent source tablespaces to one named destination tablespace.
  - Import job aborts if you attempt to remap to a destination tablespace that is temporary or non-existent or multiple tablespaces

Documentation: Database Utilities Guide: `REMAP_TABLESPACE`

Data Pump 21c | TABLESPACE transform parameter

Map multiple source tablespaces to the default destination tablespace

- Set to N so TABLESPACE clause is not emitted in DDLs during import
- The user default tablespace is used for all object creation

- Syntax

```
TRANSFORM=TABLESPACE:[Y | N]
```

- Details
  - Maps all permanent source tablespaces to the default destination tablespace.
  - `TRANSFORM=TABLESPACE:N` is mutual exclusive with `REMAP_TABLESPACE`. If both are specified, then the import reports an invalid parameter error

Documentation: Database Utilities Guide: `REMAP_TABLESPACE`

Universal Data Pump Client

Before: client and server version had to match

12.1.0.2 client to expdp from 12.1.0.2 server, 19c client to impdp to 19c server

Now (since 21c): Data Pump client is backward compatible

Always use the latest client, attach to any supported database version

Data Pump and SQL*Loader clients are in the “Tools” package of the Instant Client

Data Pump

21c Transportable Tablespaces

Data Pump 21c | TTS Can Export Metadata in Parallel

Parallelize transportable tablespace metadata operations for VLDB

- Transportable Tablespace & Full Transportable exports can have parallelism > 1
- Enhances performance for full transportable and for databases, such as Oracle E-Business suite that have a lot of metadata
- Example:

  -
    ```
    expdp system hr DIRECTORY=dpump_dir1 DUMPFILE=tts.dmp
    TRANSPORT_TABLESPACES=tbs_1 TRANSPORT_FULL_CHECK=YES
    LOGFILE=tts.log PARALLEL=4
    ```

Documentation: Database Utilities Guide

- Export: `TRANSPORT_TABLESPACES`, `TRANSPORT_FULL_CHECK`
- Import: `TRANSPORT_TABLESPACES`, `TRANSPORT_FULL_CHECK`

Data Pump 21c | Transportable Jobs are Restartable

Resume stopped transportable tablespace jobs

- Transportable tablespace TRANSPORTABLE jobs are restartable
- Like other Data Pump export/import modes that are always restartable
- Not having to start over enhances system availability and saves time because the source database is generally READ ONLY during the TTS export.
- Example

```
expdp system hr ATTACH=hr.export_job
Export> START_JOB
```

Documentation: Database Utilities Guide: START_JOB

SQL*Loader

21c New Features

SQL*Loader 21c | Native JSON Datatype

Load first-class native JSON data type

- Use Direct and Conventional Path load methods
- Load data files and LOBFILEs (unstructured ASCII & binary)

Documentation: JSON

SQL*Loader 21c | Object Store Load w/ User-Defined Credentials

Oracle SQL*Loader can read data from files in an object store

- Step 1: Create the sqlldr credential in the client wallet:

```
mkstore -wrl /usr/wallet -createEntry
oracle.sqlldr.credential.obm_scott.username <username>

mkstore -wrl /usr/wallet -createEntry
oracle.sqlldr.credential.obm_scott.password <password>
```

Documentation: Database Utilities Guide: Loading Tables Using Data Stored into Object Storage

SQL*Loader 21c | Object Store Load w/ User-Defined Credentials

Oracle SQL*Loader can read data from files in an object store

- Step 2: Create SQL*Loader control file - dept.ctl:

```
LOAD DATA
INFILE  https://objectstorage.us-phoenix-
1.oraclecloud.com/exp%u.dmp/myfiles/dept1.csv'
truncate
INTO TABLE dept
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
(DEPTNO, DNAME, LOC)
```

- Step 3: Run SQL*Loader

```
sqlldr scott dept.ctl credential=obm_scott log=dept.log
```

Data Pump

19c New Features

Data Pump 19c | What is New in Data Pump 19c?

- Suppress Encrypted Columns Clause
- Set Max Data Pump Jobs & Parallelism
- Explicitly Enable Authenticated Roles
- Use Any Object Store Credentials
- Wildcards in Object Store Dumpfile Name
- Transportable Tablespaces Test Mode
- Transportable Tablespaces Import Read-Only Tablespaces

Documentation: Database Utilities Guide

Data Pump 19c | Suppress Encrypted Columns Clause

Migrate to a database having TDE encrypted tablespaces

- TDE does not support encrypted columns (e.g., Oracle Cloud)

Example:

```
Impdp system hr DIRECTORY=dpump_dir1 DUMPFILE=hr.dmp SCHEMAS=hr
TRANSFORM=OMIT_ENCRYPTION_CLAUSE:Y
```

- Details:
  - Default: N - column encryption clauses in CREATE TABLE are enabled
  - Valid for TABLE object types

Documentation: Database Utilities Guide: TRANSFORM=OMIT_ENCRYPTION_CLAUSE

Data Pump 19c | Set Max Data Pump Jobs & Parallelism

DBA can more easily govern Oracle Data Pump resource utilization

- `MAX_DATAPUMP_JOBS_PER_PDB` database parameter (Changed)
  - Default: 100, Range: 0 to 250 or Auto - 50% of SESSIONS
  - Must be same for each RAC instance, dynamic, and modifiable per-PDB

- `MAX_DATAPUMP_PARALLEL_PER_JOB` database parameter (New)
  - Default: 50, Range: 1 to 250 or Auto - 50% of SESSIONS
  - Can be different for each RAC instance, dynamic, and modifiable per-PDB

Documentation: Database Utilities Guide:

`MAX_DATAPUMP_JOBS_PER_PDB` `MAX_DATAPUMP_PARALLEL_PER_JOB`

Data Pump 19c | Explicitly Enable Authenticated Roles

Specify whether to use authenticated roles for export and import

- Available for `expdp` and `impdp` clients, and for the Oracle Data Pump PL/SQL API
- Example:

```
ENABLE_SECURE_ROLES=YES | NO
```

Details:

- Default: `NO` – does not enable authenticated protected roles
- Beginning with release 19c you must explicitly enable authenticated roles for an export or import job

Documentation: Database Utilities Guide:

Export: `ENABLE_SECURE_ROLES` Import: `ENABLE_SECURE_ROLES`

Data Pump 19c | Use Any Object Store Credentials

Import from Oracle Cloud, AWS or Azure object store into ADB

- `CREDENTIAL` Data Pump impdp client CLI parameter
- No longer constrained to using the ADB default credential
- Object store credentials added to database DBMS_CLOUD.CREATE_CREDENTIAL()
- Data Pump validates the credential exists and the user has read access

Documentation: Database Utilities Guide: `CREDENTIAL`

Data Pump 19c | Use Any Object Store Credentials

Import from Oracle Cloud, AWS or Azure object store into ADB

- Example

-
  ```
  BEGIN
     DBMS_CLOUD.CREATE_CREDENTIAL(
     credential_name => ‘MY_CRED_NAME’,
     username => 'adwc_user@oracle.com’,
     password => ‘Auth token' );
  ```

-
  ```
  END;   (or password for OCI/C)
  ```

-
  ```
  impdp admin/password@ADWC1_high directory=data_pump_dir
     credential=MY_cred_name  …
  ```

Documentation ADW, ATP-Shared, ATP-Dedicated

Data Pump 19c | Wildcards in Dumpfile Name

Import from Oracle Cloud, AWS or Azure object store into ADB

- Oracle Data Pump allows %U & %L wildcards for dumpfile in object store
- Dumpfile Specification
  - A wildcard character can be specified in the file-name of a URL
  - Can't be used in bucket-name
- Example:

  -
    ```
    impdp admin/password@ATPC1_high
     directory=data_pump_dir credential=my_cred_name
     dumpfile= https://objectstorage.us-phoenix-
    1.oraclecloud.com/atpc/atpc_user/exp%u.dmp…
    ```

Documentation: Database Utilities Guide: `Dumpfile`

Data Pump

19c Transportable Tablespaces

Data Pump 19c | Transportable Tablespaces Test Mode

Preview an export for time to completion and check for unforeseen closure issues

- Test a TTS or Full Transportable `expdp` without setting the source `READ_ONLY`
- Resulting dumpfile cannot be imported, If attempted, an error will be issued
- Syntax:
  - `TTS_CLOSURE_CHECK: ON | OFF | FULL | TEST_MODE`
- Details: Testing for degree of constraint closure checking to be performed can be
  - `ON` - self-containment closure check
  - `OFF` – none
  - `FULL` - bi-directional
  - `TEST_MODE` - tablespaces are not required to be read-only

Documentation: Database Utilities Guide: `TTS_CLOSURE_CHECK`

Data Pump 19c | TTS Import w/ Read-Only Tablespaces

Allows Read-Only Tablespaces during Transportable Tablespaces import

- Restores pre-12.2 ability for mounting tablespace files on 1 or more databases at once
- Syntax:
  - `TRANSPORTABLE=NEVER|ALWAYS|KEEP_READ_ONLY|NO_BITMAP_REBUILD`
- Details:
  - `NEVER` – import job uses direct path or external table method to load data
  - `ALWAYS` - import job uses the transportable option
  - `KEEP_READ_ONLY` - Prevents fix-up of timezone data and rebuilding of bitmaps
  - `NO_BITMAP_REBUILD` - only update timestamp w/ timezone for faster migration

Documentation: Database Utilities Guide: `TRANSPORTABLE`

Data Pump 19c | TTS Import Read-Only Tablespaces

Example:

```
impdp system DIRECTORY=dpump_dir DUMPFILE=dumpfile_name
TRANSPORT_DATAFILES=datafile_name TRANSPORTABLE=KEEP_READ_ONLY
```

Documentation: Database Utilities Guide: `TRANSPORTABLE`

Data Pump

18c New Features

Data Pump 18c | What is New in Data Pump 18c?

- Import `DATA_OPTIONS` parameter `CONTINUE_LOAD_ON_FORMAT_ERROR` option
- Warning on import that encrypted fixed user database link passwords must be reset
- Full and partial export and import jobs can include a unified audit trail

Documentation: Database Utilities Guide:

`DATA_OPTIONS=CONTINUE_LOAD_ON_FORMAT_ERROR`

Data Pump 18c | Continue Load on Format Error

Partial import of a corrupt dump file

- `CONTINUE_LOAD_ON_FORMAT_ERROR is` an import `DATA_OPTIONS` parameter
  - Condition: if a stream format error is encountered while loading table data
  - Action: Import skips forward to the start of next granule

Documentation: Database Utilities Guide:

`DATA_OPTIONS=CONTINUE_LOAD_ON_FORMAT_ERROR`

Data Pump 18c | TDE Encrypted Password Warning

Import warns about the need for password reset after import

- Import (like export) generates an ORA-39395:

```
Warning: object <database link name> requires password reset after import
```

- If fixed-user database passwords are encrypted, database link passwords are not exported
  Export stores a known invalid password
- Reset the database link password with the command:

```
ALTER DATABASE LINK database_link_name CONNECT TO schema_name IDENTIFIED
BY password;
```

Documentation: `ALTER DATABASE LINK`

Data Pump 18c | Employ a Unified Audit Trail

Monitor and record specific user database actions and centralize the audit records

- A unified audit trail can be included for full or partial export and import
- Centralizes all audit records in one place
- How to use a unified audit trail:
  - Create or alter a policy: `SQL CREATE AUDIT POLICY`
  - Enable and disable the policy with SQL statements: `AUDIT` and `NOAUDIT`

Documentation: `SQL CREATE AUDIT POLICY`

Data Pump

12.2 New Features

Data Pump 12.2c | What is New in Data Pump 12.2?

- Parallel Export/Import of Metadata
- Substitution Variables & Wildcards
- REMAP_DIRECTORY
- Long Identifier support
- TRUST_EXISTING_TABLE_PARTITIONS
- Validation & Verification options
- …and more 12.2 Features

Documentation: Database Utilities Guide

Data Pump 12.2 | Parallel Metadata Export

Metadata export happens concurrent w/ estimate phase for table data

Pre-12.2	New Feature in 12.2
• Start with ESTIMATE phase • Gather table data objects • Other workers remain idle until data objects are gathered • Metadata exported serially • Data exported in parallel	• Start with Analysis step • Metadata objects passed to workers as they are found • e.g. Worker 1 finds a set of TABLE definitions, they are handed off to worker 2 • ESTIMATE phase still happens, but metadata no longer held up by estimate

Data Pump 12.2 | Parallel Metadata Export

Details

- Source database must be 12.2 or higher
- ESTIMATE phase uses STATISTICS only (pre-12.2 it counted blocks)
- Restart works as always
- Database metadata objects that have dependencies are imported serially
  - types (due to inheritance), schemas and procedural actions.

Documentation: Database Utilities Guide: `Parallel`

Data Pump 12.2 | Parallel Metadata Export

Details

- 2 types of DP parallelism used:
  - One worker per sub-partition or per small table (Inter-table parallelism)
  - Parallel query: (Intra-table parallelism) - 1 PX process per large partition or large unpartitioned table
- Collect statistics using the `dbms_stats` package: `gather_table_stats`, `gather_schema_stats`, or `gather_database_stats` procedure.

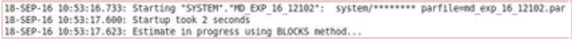
Documentation: Database Utilities Guide: `Parallel`

Data Pump 12.2 | Parallel Metadata Export

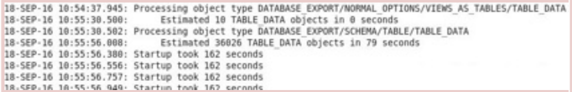
Logfile Differences

Documentation: Database Utilities Guide

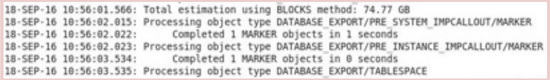
12.1.0.2



```
18-SEP-16 10:53:16.733: Starting "SYSTEM"."MD_EXP_16_12102":  system/******** parfile=md_exp_16_12102.par
18-SEP-16 10:53:17.600: Startup took 2 seconds
18-SEP-16 10:53:17.623: Estimate in progress using BLOCKS method...
```

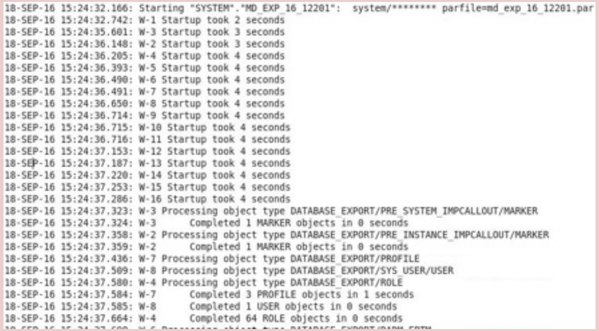


```
18-SEP-16 10:54:37.945: Processing object type DATABASE_EXPORT/NORMAL_OPTIONS/VIEWS_AS_TABLES/TABLE_DATA
18-SEP-16 10:55:30.500:      Estimated 10 TABLE_DATA objects in 0 seconds
18-SEP-16 10:55:30.502: Processing object type DATABASE_EXPORT/SCHEMA/TABLE/TABLE_DATA
18-SEP-16 10:55:56.008:      Estimated 36026 TABLE_DATA objects in 79 seconds
18-SEP-16 10:55:56.380: Startup took 162 seconds
18-SEP-16 10:55:56.556: Startup took 162 seconds
18-SEP-16 10:55:56.757: Startup took 162 seconds
18-SEP-16 10:55:56.949: Startup took 162 seconds
```



```
18-SEP-16 10:56:01.566: Total estimation using BLOCKS method: 74.77 GB
18-SEP-16 10:56:02.015: Processing object type DATABASE_EXPORT/PRE_SYSTEM_IMPCALLOUT/MARKER
18-SEP-16 10:56:02.022:      Completed 1 MARKER objects in 1 seconds
18-SEP-16 10:56:02.023: Processing object type DATABASE_EXPORT/PRE_INSTANCE_IMPCALLOUT/MARKER
18-SEP-16 10:56:03.534:      Completed 1 MARKER objects in 0 seconds
18-SEP-16 10:56:03.535: Processing object type DATABASE_EXPORT/TABLESPACE
```

12.2.0.1



```
18-SEP-16 15:24:32.166: Starting "SYSTEM"."MD_EXP_16_12201":  system/******** parfile=md_exp_16_12201.par
18-SEP-16 15:24:32.742: W-1 Startup took 2 seconds
18-SEP-16 15:24:35.601: W-3 Startup took 3 seconds
18-SEP-16 15:24:36.148: W-2 Startup took 3 seconds
18-SEP-16 15:24:36.205: W-4 Startup took 4 seconds
18-SEP-16 15:24:36.393: W-5 Startup took 4 seconds
18-SEP-16 15:24:36.490: W-6 Startup took 4 seconds
18-SEP-16 15:24:36.491: W-7 Startup took 4 seconds
18-SEP-16 15:24:36.650: W-8 Startup took 4 seconds
18-SEP-16 15:24:36.714: W-9 Startup took 4 seconds
18-SEP-16 15:24:36.715: W-10 Startup took 4 seconds
18-SEP-16 15:24:36.716: W-11 Startup took 4 seconds
18-SEP-16 15:24:37.153: W-12 Startup took 4 seconds
18-SEP-16 15:24:37.187: W-13 Startup took 4 seconds
18-SEP-16 15:24:37.220: W-14 Startup took 4 seconds
18-SEP-16 15:24:37.253: W-15 Startup took 4 seconds
18-SEP-16 15:24:37.286: W-16 Startup took 4 seconds
18-SEP-16 15:24:37.323: W-3 Processing object type DATABASE_EXPORT/PRE_SYSTEM_IMPCALLOUT/MARKER
18-SEP-16 15:24:37.324: W-3      Completed 1 MARKER objects in 0 seconds
18-SEP-16 15:24:37.358: W-2 Processing object type DATABASE_EXPORT/PRE_INSTANCE_IMPCALLOUT/MARKER
18-SEP-16 15:24:37.359: W-2      Completed 1 MARKER objects in 0 seconds
18-SEP-16 15:24:37.436: W-7 Processing object type DATABASE_EXPORT/PROFILE
18-SEP-16 15:24:37.509: W-8 Processing object type DATABASE_EXPORT/SYS_USER/USER
18-SEP-16 15:24:37.580: W-4 Processing object type DATABASE_EXPORT/ROLE
18-SEP-16 15:24:37.584: W-7      Completed 3 PROFILE objects in 1 seconds
18-SEP-16 15:24:37.585: W-8      Completed 1 USER objects in 0 seconds
18-SEP-16 15:24:37.664: W-4      Completed 64 ROLE objects in 0 seconds
```

Data Pump 12.2 | Parallel Metadata Import

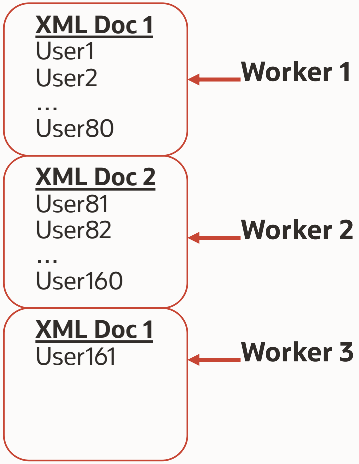
Metadata export happens concurrent w/ estimate phase for table data

Pre-12.2	New Feature in 12.2
• One worker per partition/subpartition • PQ used if partitions are large enough • Package bodies loaded in parallel With patch for bug 22273229 • Indexes built in parallel • Constraints created in parallel • Backport to 12.1.0.2, 11.2.0.4	• Added parallel import of most other metadata objects • Some exceptions • Types (due to inheritance) • Schemas • Procedural actions

Data Pump 12.2 | Parallel Metadata Import

Internals

- Metadata is exported in XML documents
  - Each XML document in dumpfile contains n objects of a given type
- XML documents are allocated to workers 1 document at a time
- Example: 161 users to import
  - Users are exported with up to 80 users per XML document
  - What happens with PARALLEL=8?
- Notes:
  - Works for conventional (dumpfile) jobs
  - Not (yet) for transportable jobs or network mode
  - Restart works same as always
  - Status command will show multiple workers on metadata

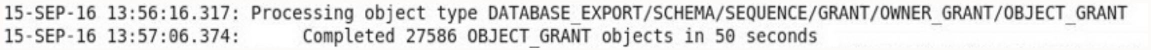


(Workers 8-n would be idle)

Data Pump 12.2 | Parallel Metadata Import

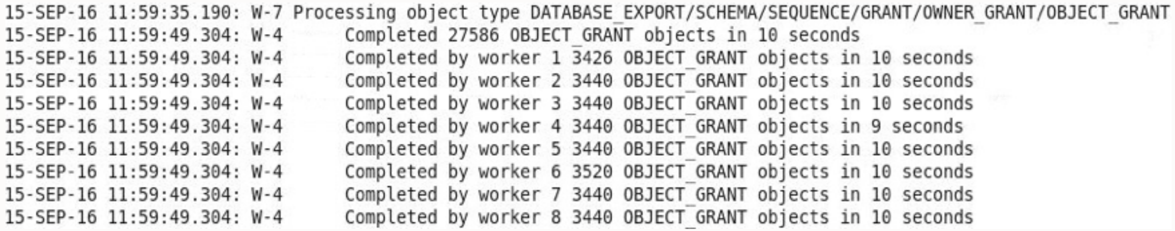
Logfile

- Comparison with `PARALLEL=8` for 27586 object grants and `METRICS=Y`
- 12.1.0.2



```
15-SEP-16 13:56:16.317: Processing object type DATABASE_EXPORT/SCHEMA/SEQUENCE/GRANT/OWNER_GRANT/OBJECT_GRANT
15-SEP-16 13:57:06.374:      Completed 27586 OBJECT_GRANT objects in 50 seconds
```

- 12.2.0.1



```
15-SEP-16 11:59:35.190: W-7 Processing object type DATABASE_EXPORT/SCHEMA/SEQUENCE/GRANT/OWNER_GRANT/OBJECT_GRANT
15-SEP-16 11:59:49.304: W-4      Completed 27586 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 1 3426 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 2 3440 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 3 3440 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 4 3440 OBJECT_GRANT objects in 9 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 5 3440 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 6 3520 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 7 3440 OBJECT_GRANT objects in 10 seconds
15-SEP-16 11:59:49.304: W-4      Completed by worker 8 3440 OBJECT_GRANT objects in 10 seconds
```

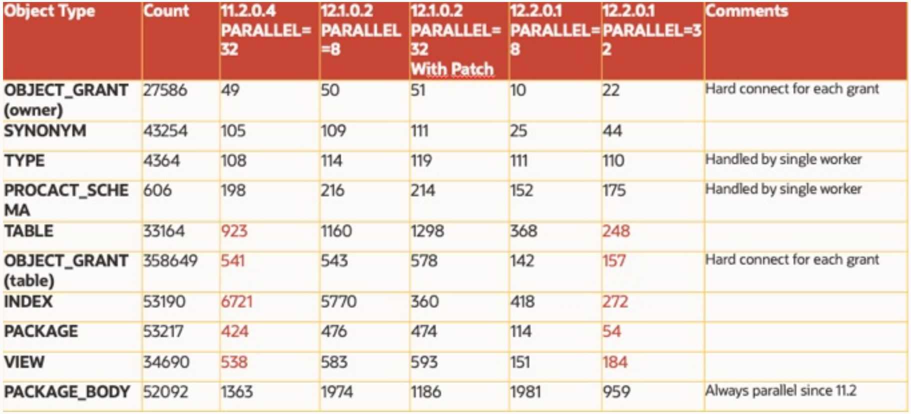
Documentation: Database Utilities Guide: `Parallel`

Data Pump 12.2 | Parallel Metadata Import

Performance

- Examples from E-Business Suite test database

*import time in seconds



Object Type	Count	11.2.0.4 PARALLEL=32	12.1.0.2 PARALLEL=8	12.1.0.2 PARALLEL=32 With Patch	12.2.0.1 PARALLEL=8	12.2.0.1 PARALLEL=32	Comments
OBJECT_GRANT (owner)	27586	49	50	51	10	22	Hard connect for each grant
SYNONYM	43254	105	109	111	25	44	
TYPE	4364	108	114	119	111	110	Handled by single worker
PROCACT_SCHEMA	606	198	216	214	152	175	Handled by single worker
TABLE	33164	923	1160	1298	368	248	
OBJECT_GRANT (table)	358649	541	543	578	142	157	Hard connect for each grant
INDEX	53190	6721	5770	360	418	272	
PACKAGE	53217	424	476	474	114	54	
VIEW	34690	538	583	593	151	184	
PACKAGE_BODY	52092	1363	1974	1186	1981	959	Always parallel since 11.2

Data Pump 12.2 | Substitution Variables for Dumpfile Name

Support parallelism using substitution variables for dumpfile name

- Pre-12.2: `%U` generates a fixed-width 2-digit number
  - e.g. dumpfile=`exp%U.dmp`

New option for 12.2 expdp and impdp:

- `%L` or `%l`: Incrementing number from 01 up to 2,147,483,646

New options in 12.2 expdp only:

- `%d` or `%D`: Day of Month in DD format
- `%m` or `%M`: Number of Month in MM format
- `%y` or `%Y`: Year in YYYY format
- `%t` or `%T`: Full date in YYYYMMDD format

Documentation: Database Utilities Guide: `dumpfile`

```
$ expdp system/oracle directory=mydir       \
  filesize=50K dumpfile=exp%T_%L.dmp full=y
...
...
...
. . exported "WMSYS"."WM$METADATA_MAP"
0 KB       0 rows
Master table "SYSTEM"."SYS_EXPORT_FULL_01" successfully
loaded/unloaded
*******************************************************
***********************
Dump file set for SYSTEM.SYS_EXPORT_FULL_01 is:
/home/oracle/exp20160917_01.dmp
/home/oracle/exp20160917_02.dmp
/home/oracle/exp20160917_03.dmp
...
...
/home/oracle/exp20160917_67.dmp
/home/oracle/exp20160917_68.dmp
/home/oracle/exp20160917_69.dmp
/home/oracle/exp20160917_70.dmp
/home/oracle/exp20160917_71.dmp
/home/oracle/exp20160917_72.dmp
Job "SYSTEM"."SYS_EXPORT_FULL_01" successfully
completed at Sat Sep 17 23:47:31 2016 elapsed 0
00:03:00
```

Data Pump 12.2 | Wildcards for TRANSPORT_DATAFILES

Simplify export/import by using a wildcard in 12.2 instead of listing every file

- Pre-12.2:

```
TRANSPORT_DATAFILES=users01.dbf
TRANSPORT_DATAFILES=users02.dbf
…
TRANSPORT_DATAFILES=data1.dbf
TRANSPORT_DATAFILES=data2.dbf
…
```

- 12.2: wildcards
- * (asterisk) matches multiple characters
- ? (question mark) matches a single character

```
TRANSPORT_DATAFILES=users*.dbf
TRANSPORT_DATAFILES=data?.dbf
…
```

```
$ impdp system/oracle@pdb2 network_link=sourcedb \
version=12 full=y transportable=always metrics=y \
exclude=statistics \
directory=mydir \
logfile=pdb2.log \
transport_datafiles='/u02/oradata/CDB2/pdb2/user*.dbf'
```

Documentation: Database Utilities Guide: `impdb` TRANSPORT_DATAFILES

Data Pump 12.2 | REMAP_DIRECTORY

Map one directory format to another when changing OS or migrating to the cloud

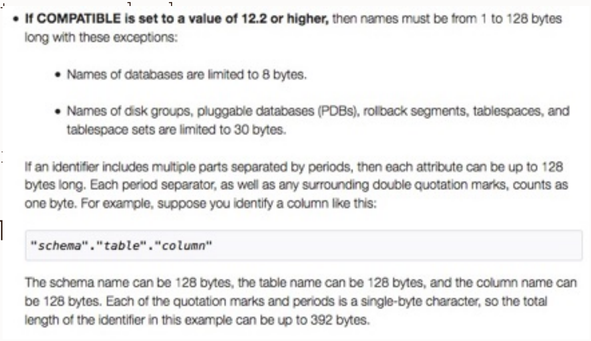
- Applies to DDL where directory specs are used
  - e.g. CREATE TABLESPACE
- Change directory spec without changing filenames
- Useful when moving between OS platforms and to the cloud
  - Example: importing dumpfile created on OpenVMS into database on Linux
  - `REMAP_DIRECTORY="'DB1$:[HRDATA.PAYROLL]':'/db1/hrdata/payroll/'"`

Documentation: Database Utilities Guide: REMAP_DIRECTORY

Data Pump 12.2 | LONG Identifiers

Oracle recommends the superset unicode AL32UTF8 database character set

- Oracle 12.1.0.2: 1-30 bytes
  - CREATE TABLE abcdefghijklmnopqrs
- Oracle 12.2.0.1: 1-128 bytes
  - If COMPATIBLE ≥ 12.2.0
  - CREATE TABLE abcdefghijklmnopqrstuvwxyz_abcde
- Database name: ≤ 8 byte
- Disk Groups, PDBs, rollback segments and tabl
- Target DB must support 128-byte identifiers



- **If COMPATIBLE is set to a value of 12.2 or higher,** then names must be from 1 to 128 bytes long with these exceptions:
  - Names of databases are limited to 8 bytes.
  - Names of disk groups, pluggable databases (PDBs), rollback segments, tablespaces, and tablespace sets are limited to 30 bytes.

If an identifier includes multiple parts separated by periods, then each attribute can be up to 128 bytes long. Each period separator, as well as any surrounding double quotation marks, counts as one byte. For example, suppose you identify a column like this:

```
"schema"."table"."column"
```

The schema name can be 128 bytes, the table name can be 128 bytes, and the column name can be 128 bytes. Each of the quotation marks and periods is a single-byte character, so the total length of the identifier in this example can be up to 392 bytes.

Documentation: AL32UTF8

Data Pump 12.2 | TRUST_EXISTING_TABLE_PARTITIONS

If you are sure partitioning matches table can imported faster in parallel

- Pre-12.2
  - Importing into existing table was done serially
  - Data Pump couldn't be sure that partitioning in DB matched partitioning in dumpfile

- New 12.2 Parameter: `DATA_OPTIONS=TRUST_EXISTING_TABLE_PARTITIONS`
  - Big performance boost
  - If partitions don't match…error:

```
ORA-31693: Table data object "SH"."SALES_BIG_PT":"SALES_2000" failed to load/unload
and is being skipped due to error:
ORA-29913: error in executing ODCIEXTTABLEFETCH callout
ORA-14401: inserted partition key is outside specified partition
```

Documentation: Database Utilities Guide:

`DATA_OPTIONS=TRUST_EXISTING_TABLE_PARTITIONS`

Data Pump 12.2 | Data Validation & Verification

Extra Validation for Things That Should Never Happen

- DATA_OPTIONS=VALIDATE_TABLE_DATA
  - Import only
  - Validates date & number formats of table data
  - Default is no validation
- DATA_OPTIONS=VERIFY_STREAM_FORMAT
  - Export only
  - Default is no verification

```
ORA-02374: conversion error loading table "DPV"."TEST18"
ORA-12899: value too large for column C1 (actual: 500,
maximum: 498)
ORA-02372: data for row: C8 : '
```

```
Starting "SCOTT"."SYS_EXPORT_TABLE_01":  scott/******** tables=t
directory=dmpdir dumpfile=t.dmp reuse_dumpfiles=true
Processing object type TABLE_EXPORT/TABLE/TABLE_DATA
Processing object type TABLE_EXPORT/TABLE/STATISTICS/TABLE_STATISTICS
Processing object type TABLE_EXPORT/TABLE/STATISTICS/MARKER
Processing object type TABLE_EXPORT/TABLE/TABLE
. . exported "SCOTT"."T"                                 5.570
KB       1 rows
ORA-31694: master table "SCOTT"."SYS_EXPORT_TABLE_01" failed to
load/unload
ORA-02354: error in exporting/importing data
ORA-26009: stream verification error: [1], [0], [0], [0]
```

Documentation: Database Utilities Guide:

DATA_OPTIONS=VALIDATE_TABLE_DATA

Data Pump 12.2 | More 12.2 Features

Technical tips

- Use direct path load in network mode!
  - Specify `ACCESS_METHOD=DIRECT_PATH` with `NETWORK_LINK=<dblink>`
  - Allows network import of `LONG` and `LONG RAW`
- Data Pump available in Instant Client
  - Tools package for Instant Client
  - Includes SQL*Loader, expdp, impdp, exp, imp
- Views that describe available transforms
  - `DBMS_METADATA_TRANSFORMS`
  - `DBMS_METADATA_TRANSFORM_PARAMS`
  - `DBMS_METADATA_PARSE_ITEMS`

Data Pump 12.2 | More 12.2 Features

Technical tips

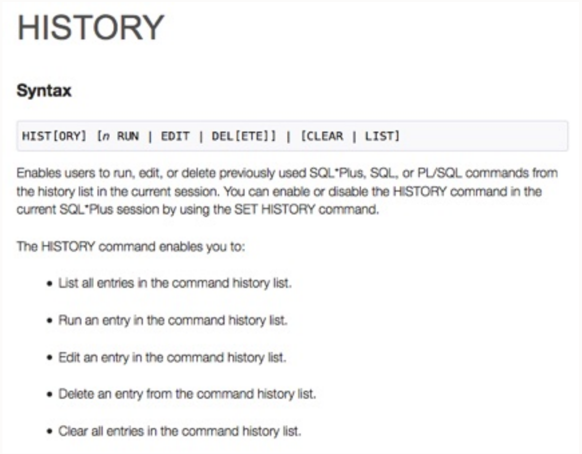
- New interactive commands
  - `TRACE` parameter can be set for a running job
    - No need to stop/restart job for tracing to take effect
  - `STOP_WORKER` command
    - Kill an individual worker you believe to be hung or stuck
  - Both documented in MOS notes
- Enhanced log files
  - When `METRICS =Y`
    - Show worker ID for each item processed
    - Show access method for each table
  - Include contents of parfile in logfile

```
18-SEP-16 15:24:30.950: ;;;
*************************************************************
18-SEP-16 15:24:30.951: ;;; Parfile values:
18-SEP-16 15:24:30.953: ;;;  parfile:
job_name=md_exp_16_12201
18-SEP-16 15:24:30.955: ;;;  parfile:  reuse_dumpfiles=Y
18-SEP-16 15:24:30.957: ;;;  parfile:  logtime=all
18-SEP-16 15:24:30.958: ;;;  parfile:  metrics=Y
18-SEP-16 15:24:30.960: ;;;  parfile:  parallel=16
18-SEP-16 15:24:30.962: ;;;  parfile:  full=Y
18-SEP-16 15:24:30.963: ;;;  parfile:
logfile=md_exp_16_12201.log
18-SEP-16 15:24:30.965: ;;;  parfile:
dumpfile=md16_12201_%U.dmp
18-SEP-16 15:24:30.966: ;;;  parfile:  directory=EBSIMP
18-SEP-16 15:24:30.968: ;;;
*************************************************************
```

Tech Tip | SQL*Plus History

Store and recall SQL*Plus commands

- `SQL> SET HISTORY ON`
- `SQL> SET HISTORY 1000`
- Example:
  - `SQL> hist`
    - Lists all commands from the history
  - `SQL> hist 7 run`
    - Will run the 7th command from the list



HISTORY

Syntax

```
HIST[ORY] [n RUN | EDIT | DEL[ETE]] | [CLEAR | LIST]
```

Enables users to run, edit, or delete previously used SQL*Plus, SQL, or PL/SQL commands from the history list in the current session. You can enable or disable the HISTORY command in the current SQL*Plus session by using the SET HISTORY command.

The HISTORY command enables you to:

- List all entries in the command history list.
- Run an entry in the command history list.
- Edit an entry in the command history list.
- Delete an entry from the command history list.
- Clear all entries in the command history list.

real world scenarios

DATA PUMP

best practices

INTRO

UPGRADE

MOVE

STATISTICS

LOBS

CHARACTERSET

AUTONOMOUS

PARTITIONS

VERIFICATION

You can use Data Pump to move data into a newer release of Oracle Database

- Oracle recommends upgrading the database using AutoUpgrade

Upgrade via Data Pump

Suitable when

- Source is a very old releases
- Small amount of data
- Less complex database
- Going to multitenant
- Re-organization is required

Upgrade via Data Pump

Considerations

- Longer downtime
- AutoUpgrade made upgrades much easier
- A full export might be the best option

real world scenarios

DATA PUMP

best practices

INTRO

UPGRADE

MOVE

STATISTICS

LOBS

CHARACTERSET

AUTONOMOUS

PARTITIONS

VERIFICATION

To migrate your data, you typically use Data Pump in schema or full mode

SCHEMA	Individual schemas and what they own
FULL	All schemas plus more or less everything in the database

Move | Full Export

Objects exported only in full export:

- Audit trail and policies
- Database Vault
- Directories
- Profiles and password verify function
- Public database links
- Public synonyms
- Roles
- SQL Management Objects (plan histories, SQL plan baselines, SQL profiles, etc.)
- Tablespaces
- Users (other than those specified in `SCHEMAS` parameter)
- Workspace manager (for schema export you need to use `DBMS_WM.Export_Schemas`)

  ...

Data Pump never exports grants on SYS objects

- Not even in a full export
- Add them manually following the import

Data Pump never exports AWR

- Not even in a full export
- Use `rdbms/admin/awrextr.sql`

real world scenarios

DATA PUMP

best practices

INTRO

UPGRADE

MOVE

STATISTICS

LOBS

CHARACTERSET

AUTONOMOUS

PARTITIONS

VERIFICATION

1 Include statistics in Data Pump

2 Exclude statistics in Data Pump
Regather statistics after import

3 Exclude statistics in Data Pump
Import statistics using `DBMS_STATS`

On YouTube we have videos on `DBMS_STATS`, including a demo and pro tips

Transporting Statistics | Customer Feedback

> *We have adopted this method for stats. We migrated 60 TB database from AIX to Exadata using cross-platform transportable tablespace without stats.*
>
> *Gathering stats from scratch took more than 36 hours.*
> *We transported the statistics in less then 2 hours.*

Taoqir Hassan, comment on YouTube channel

Generally, we recommend excluding statistics from Data Pump export

- Use `EXCLUDE=STATISTICS`

EXCLUDE=STATISTICS ⇛

Table statistics

Index statistics

Statistics preferences

Column usage information

`EXCLUDE=STATISTICS` ⇛

Table statistics

Index statistics

Statistics preferences

Column usage information

```
BEGIN
   DBMS_STATS.SET_TABLE_PREFS (
      OWNNAME => '...',
      TABNAME => '...',
      PNAME   => 'TABLE_CACHED_BLOCKS',
      PVALUE  => '42'
   );
END;
```

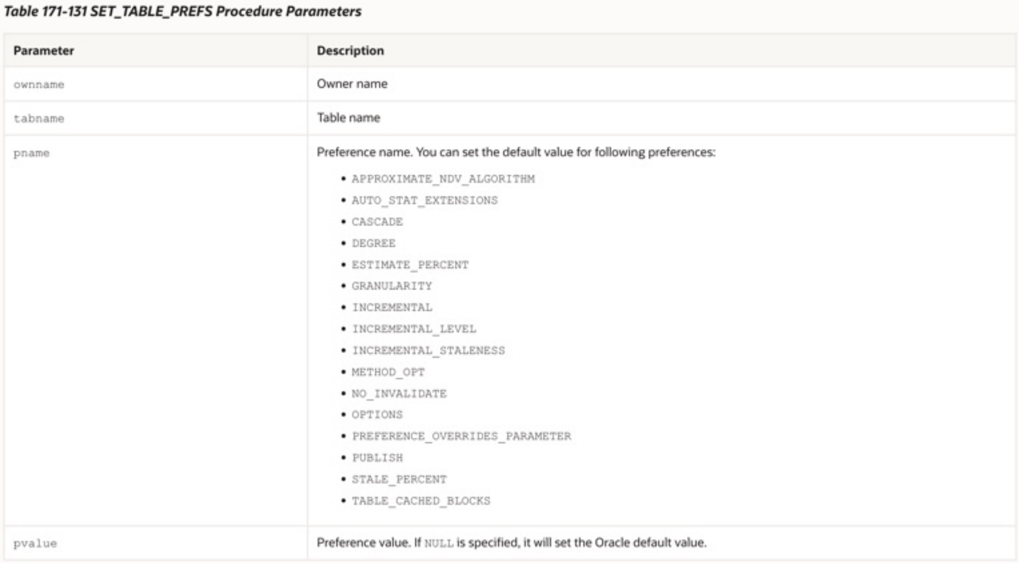


Table 171-131 SET_TABLE_PREFS Procedure Parameters

Parameter	Description
`ownname`	Owner name
`tabname`	Table name
`pname`	Preference name. You can set the default value for following preferences: • `APPROXIMATE_NDV_ALGORITHM` • `AUTO_STAT_EXTENSIONS` • `CASCADE` • `DEGREE` • `ESTIMATE_PERCENT` • `GRANULARITY` • `INCREMENTAL` • `INCREMENTAL_LEVEL` • `INCREMENTAL_STALENESS` • `METHOD_OPT` • `NO_INVALIDATE` • `OPTIONS` • `PREFERENCE_OVERRIDES_PARAMETER` • `PUBLISH` • `STALE_PERCENT` • `TABLE_CACHED_BLOCKS`
`pvalue`	Preference value. If `NULL` is specified, it will set the Oracle default value.

PL/SQL Packages and Types Reference, Oracle Database 19c

Data Pump exports table-level statistics preferences together with table statistics

- In full, schema and table mode
- In transportable, it is controlled by `USER_PREF_STATISTICS`

Data Pump never exports global statistics preferences

- Not even in a full export
- Define manually using `DBMS_STATS.SET_GLOBAL_PREFS`

`DBMS_STATS` package has dedicated procedures for transporting table-level statistics preferences

You often use statistics preferences to solve a particular problem

- Evaluate whether that problem exists in the target environment

`EXCLUDE=STATISTICS` ⇛

Table statistics

Index statistics

Statistics preferences

Column usage information

Statistics | Column Usage Information

- Information on how you join tables
- Used by the optimizer to determine when to create histograms
  `METHOD_OPT => ... SIZE AUTO`
- When missing, statistics gathering creates no or few histograms
- Stored internally in `SYS.COL_USAGE$`

When Data Pump transfers statistics,
it also transfers column usage information

EXCLUDE

`EXCLUDE=STATISTICS`

`COL_USAGE$` empty

REGATHER

First time only

```
METHOD_OPT =>
SIZE SKEWONLY
```

GO LIVE

Column usage information is updated

REGATHER

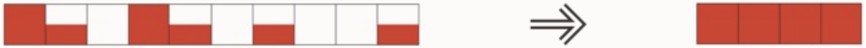
Use default

```
METHOD_OPT =>
SIZE AUTO
```

Importing statistics might be a bad idea

When source and target database do not match

Statistics | When Importing Stats Is Bad



Fragmented table

Blocks	12000
Leaf blocks	11000
B-level	4
Clustering factor	10000

Compacted table

Blocks	12000
Leaf blocks	11000
B-level	4
Clustering factor	10000

```
DBMS_STATS.GATHER_TABLE_STATS(...
```

Blocks	5000
Leaf blocks	4000
B-level	2
Clustering factor	20000

Statistics | When Importing Stats Is Bad

- Potentially a problem
  - Fragmented tables
  - Changing block size
  - Changing character set
  - Compress or decompress

  ...

- Only a problem for table and index base statistics, column statistics remain accurate

Accurate statistics is the starting point for good performance

Comparing STATISTICS options

	Import with Data Pump	Regather	Import with DBMS_STATS
Time	Significant	Significant	Short
Column usage information	Included	Missing	Missing
Accuracy	Potentially inaccurate	Accurate	Potentially inaccurate
Statistics preferences	Included	Missing	Optional

real world scenarios

DATA PUMP

best practices

INTRO

UPGRADE

MOVE

STATISTICS

LOBS

CHARACTERSET

AUTONOMOUS

PARTITIONS

VERIFICATION

A short history of **binary data types**

v4	LONG and LONG RAW
8*i*	CLOB and BLOB
10g	SecureFile LOBs

v4	LONG and LONG RAW
8i	BasicFile LOBs
10g	SecureFile LOBs

v4

LONG and LONG RAW

- Only 1 column per table
- Max size: 2GB - 1

8*i*

BasicFile LOBs

- Performance constraints
- Data Pump can act with one worker only
- Max size: (4GB - 1) * `DB_BLOCK_SIZE`

10g

SecureFile LOBs

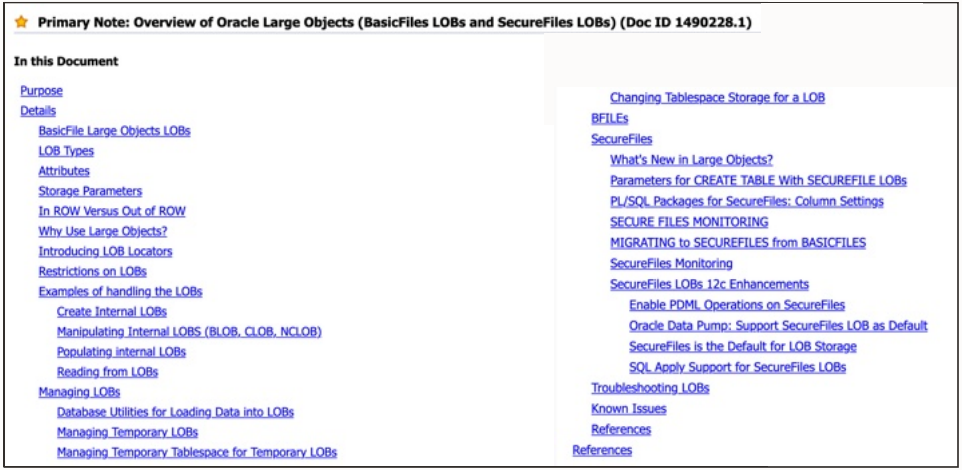
- Improved performance
- Data Pump can act with multiple workers
- Deduplication, encryption and more
- Max size: same as with CLOB/BLOB

As of today, all legacy binary data types should have been migrated to SecureFile LOBs

```
impdp ... transform=lob_storage:securefile
```

Overview on BasicFile and SecureFile LOBs

MOS Note: 1490228.1 - Overview of Oracle LOBs



Primary Note: Overview of Oracle Large Objects (BasicFiles LOBs and SecureFiles LOBs) (Doc ID 1490228.1)

In this Document

Different LOB types

Internal LOBs stored inside the database

- CLOB
- NCLOB
- BLOB

External LOBs stored outside the database

- BFILE

Initialization Parameter

`DB_SECUREFILE`

- `NEVER`
- `PERMITTED`
- `PREFERRED` ← LOBs are created as SecureFile LOBs unless explicitly stated
- `ALWAYS`
- `IGNORE`

Tablespace must use Automatic Segment Space Management (ASSM)

Data Pump & LOBs
Things to know and consider

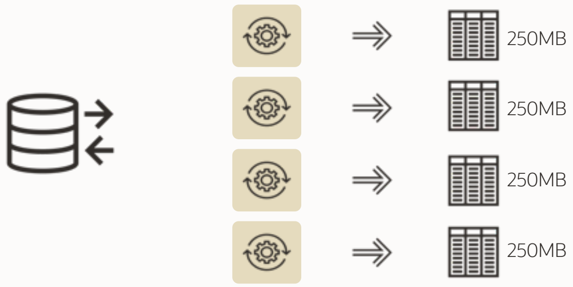
No parallelism with BasicFile LOBs

Always use SecureFile LOBs

"But why is there only one worker?"

Data Pump | Parallel Worker Activity

Normally, Data Pump *employs* one worker per 250MB table segment



LOB Export | Example Table

```
CREATE OR REPLACE DIRECTORY BLOB_DIR AS '/tmp/mydir';

CREATE TABLE tab1 ( id NUMBER, blob_data BLOB );

BEGIN ... DBMS_LOB.LOADBLOBFROMFILE ...

exec DBMS_STATS.GATHER_TABLE_STATS('HUGO','TAB1');
```

10GB

For a complete example, please visit oracle-base.com

LOB data is stored out-of-row in a separate LOB segment

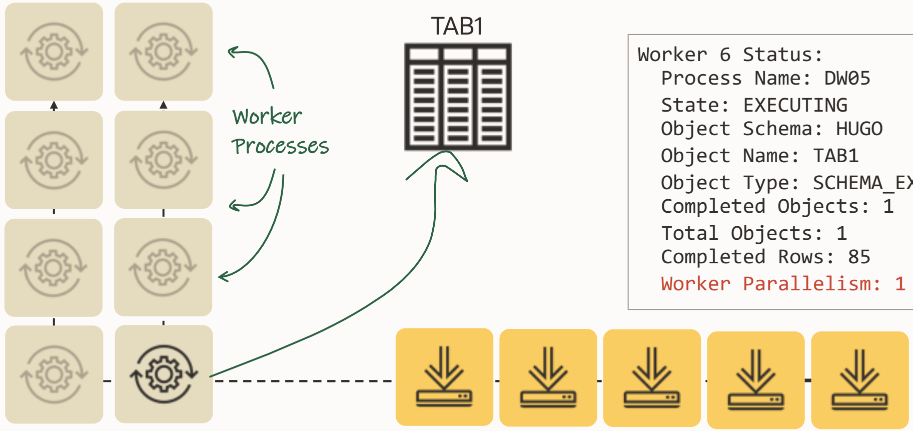
- Smaller LOBs less than 4000 bytes are stored in-row

Starting Data Pump – Test:

```
DIRECTORY=DATA_PUMP_DIR
DUMPFILE=MYDUMP%L.DMP
LOGFILE=MYDUMP01.LOG
SCHEMAS=HUGO
LOGTIME=ALL
METRICS=YES
PARALLEL=8
```

LOB Export | Lazy Workers?

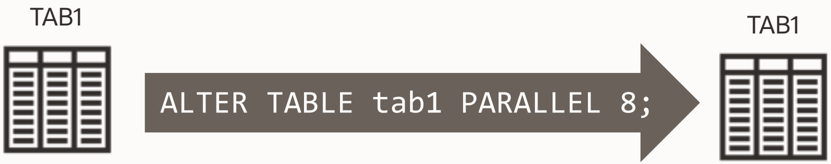
8 workers, 5 dump files – and only 1 worker exports TAB1



```
Worker 6 Status:
  Process Name: DW05
  State: EXECUTING
  Object Schema: HUGO
  Object Name: TAB1
  Object Type: SCHEMA_EXPORT/...
  Completed Objects: 1
  Total Objects: 1
  Completed Rows: 85
  Worker Parallelism: 1
```

Maybe the table's PARALLEL DEGREE is too low?

LOB Export | Parallel Degree



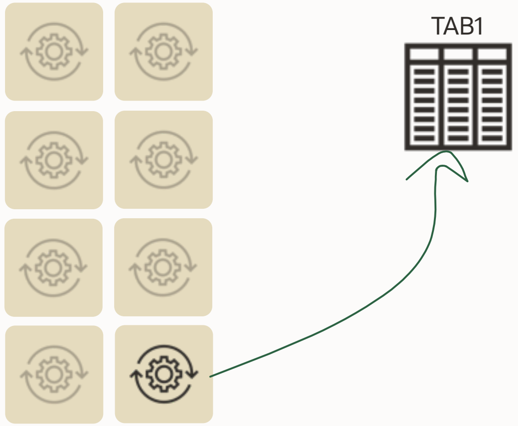
```
ALTER TABLE tab1 PARALLEL 8;
```

```
select degree
from DBA_TABLES
where table_name='TAB1';

 DEGREE
_____________
         1
```

```
select degree
from DBA_TABLES
where table_name='TAB1';

 DEGREE
_____________
         8
```

LOB Export | Parallel Degree

No relief ☹



```
Worker 1 Status:
Process Name: DW08
  State: EXECUTING
  Object Schema: HUGO
  Object Name: TAB1
  Object Type: SCHEMA_EXPORT/...
  Completed Objects: 1
  Total Objects: 1
  Completed Rows: 85
  Worker Parallelism: 1
```

LOB Export | Table Segments and Extents

Segments → TAB1 ← Extents

```
select BYTES, BLOCKS, EXTENTS
from   DBA_SEGMENTS
where  SEGMENT_NAME = 'TAB1'
and    OWNER = 'HUGO';


     BYTES      BLOCKS      EXTENTS
__________  __________  ___________
    131072          16            2
```

```
select ROUND(SUM(BYTES)/1024/1024/1024,2) "GB"
from   DBA_EXTENTS
where  SEGMENT_NAME IN
           (select SEGMENT_NAME
            from   DBA_LOBS
            where  TABLE_NAME = 'TAB1'
            and    OWNER = 'HUGO');

        GB
__________
     10.31
```

LOB Export | Table Statistics

TAB1

Table → ← Columns

```
select NUM_ROWS, BLOCKS, AVG_ROW_LEN
from   DBA_TAB_STATISTICS
where  TABLE_NAME = 'TAB1';


 NUM_ROWS    BLOCKS    AVG_ROW_LEN

_________  ________  _____________
       85        13            720
```

```
select COLUMN_NAME, NUM_DISTINCT,
       SAMPLE_SIZE, AVG_COL_LEN
from   DBA_TAB_COL_STATISTICS
where  TABLE_NAME='TAB1';

COLUMN_N NUM_DIST  SAMPLE_SIZE  AVG_COL_LEN

________ ________ ____________ ____________
ID              1           85            3
BLOB_DATA       0           85          717
```

It looks like Data Pump doesn't know anything about the dimensions of the LOB segment

LOB Export | User Objects

```
select OBJECT_NAME, OBJECT_TYPE from DBA_OBJECTS
where OWNER = 'HUGO';

                 OBJECT_NAME     OBJECT_TYPE
____________________________  ______________
TAB1                          TABLE
SYS_IL0000070285C00002$$      INDEX
SYS_LOB0000070285C00002$$     LOB
```

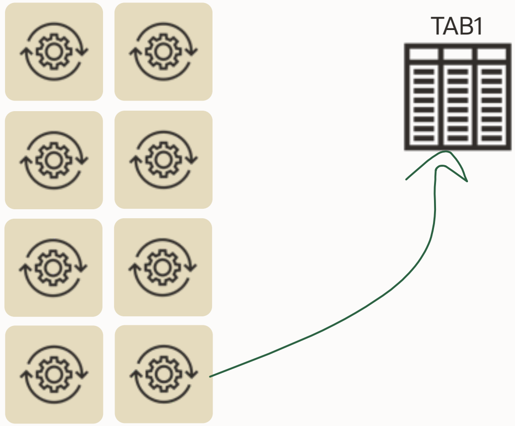
Is it possible to *analyze* a LOB segment?

LOB Export | Manipulating Statistics

```
begin
 DBMS_STATS.SET_TABLE_STATS (
   ownname => 'HUGO',
   tabname => 'TAB1',
   numrows => 10000000,
   numblks => 1000000);
 end;
 /
```

LOB Export | Parallel Degree

Relief ☺ Workers do PQ now!



```
Worker 2 Status:
Process Name: DW01
  State: EXECUTING
  Object Schema: HUGO
  Object Name: TAB1
  Object Type: SCHEMA_EXPORT/...
  Completed Objects: 1
  Total Objects: 1
  Completed Rows: 85
  Completed Bytes: 1,474,081,152
  Worker Parallelism: 7
```

Why only one worker with PQ?
Why not multiple workers?

You can boost parallelism
by using partitioned tables

"And BFILE LOBs?"

BFILE LOBs

External LOBs stored outside the database

Full export:

- Directory definition gets exported/imported
- You must copy the files

Schema export:

- You must create the directory within the database
- You must copy the files

Table export:

- You must create the directory within the database
- You must copy the files

Save downtime by copying the external files in advance

- BFILEs are always read-only

If the directory path changes,
make sure to update the directory object

real world scenarios

DATA PUMP

best practices

INTRO

UPGRADE

MOVE

STATISTICS

LOBS

CHARACTERSET

AUTONOMOUS

PARTITIONS

VERIFICATION

Character Sets | Brief Introduction

- Also known as a code page
- Each character is mapped to a numeric index, called a *codepoint*
- Codepoint stores character data in a computer system
- There are over hundred character sets:
  - International standards, maintained by the International Organization for Standardization (ISO)
  - Country-specific standards
  - Computer system vendor standard

Unicode is not just Unicode.
It has evolved over time.

- Different encodings

Character Sets | Common Character Sets

Character Set	Description and Comments	Type	Oracle DBMS Name	Microsoft Windows Name
ISO 8859-1	Western European Latin-1. Contains characters that are required to represent Western European languages. However, does not include the euro symbol, the trademark (TM) symbol, or the oe ligature.	ISO	WE8ISO8859P1	CP28591
Microsoft Code Page 1252	Microsoft Code Page 1252 - Western European. Very similar to ISO 8859-1, except for the inclusion of additional characters. Includes the euro symbol, trademark (TM) symbol, and oe ligature, but using a different codepoint than ISO 8859-15.	Vendor (Microsoft)	WE8MSWIN1252	CP1252
ISO 8859-2	Central/Eastern European Latin-2. Contains characters that are required for Central European languages, including Czech, Hungarian, and Polish. Does not include the euro symbol.	ISO	EE8ISO8859P2	CP28592
ISO 8859-15	Western European extended Latin-9. Similar to ISO 8859-1, but contains the euro symbol, oe ligature, and several characters that are required for Icelandic.	ISO	WE8ISO8859P15	CP28605
Shift-JIS	Most common Japanese character set. Defines thousands of characters for writing Japanese.	Country (Japan)	JA16SJIS or JA16SJISTILDE	CP932
IBM CCSID 37	IBM Coded Character Set ID 37. Western European Multilingual EBCDIC-based character set.	Vendor (IBM)	WE8EBCDIC37	CP1140
GB18030	Chinese national character set	Country (China)	GB18030	GB18030

Common ENCONDINGS of Unicode

	UTF-8	UTF-16	UTF-32
Bitness	8 bit	16 bit	32 bit
Width	Variable	Variable	Fixed
Minimum bytes per char	1	2	4
Maximum bytes per char	4	4	4
	Maximum US-ASCII compatibility		

Character Sets | Unicode Encoding Example

Unicode Encoding

UTF-32

UTF-16

UTF-8

Character Sets | Unicode Encoding Example

Unicode Encoding	**Latin Small Letter a (U+0061)**
UTF-32	0x0000006
UTF-16	0x0061
UTF-8	0x61

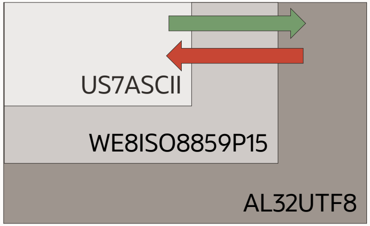
Character Sets | Unicode Encoding Example

Unicode Encoding	**Latin Small Letter a (U+0061)**	**Latin Small Letter ñ (U+00F1)**
UTF-32	0x0000006	0x000000F1
UTF-16	0x0061	0x00F1
UTF-8	0x61	0xC3B1

Character Sets | Unicode Encoding Example

Unicode Encoding	**Latin Small Letter a (U+0061)**	**Latin Small Letter ñ (U+00F1)**	**€ Symbol (U+20AC)**
UTF-32	0x0000006	0x000000F1	0x000020AC
UTF-16	0x0061	0x00F1	0x20AC
UTF-8	0x61	0xC3B1	0xE282AC

Character Sets | Superset and Subset



US7ASCII to AL32UTF8
WE8ISO8859P15 to AL32UTF8
Migrating to superset
No data loss

AL32UTF8 to WE8ISO8859P15
AL32UTF8 to US7ASCII
Migrating to subset
Potential data loss

ORA-39346:
"data loss in character set conversion for object %s"

Cause: Oracle Data Pump import converted a metadata object from the export database character set into the target database character set prior to processing the object. Some characters could not be converted to the target database character set and so the default replacement character was used.

Fix: No specific user action is required. This type of data loss can occur if the target database character set is not a superset of the export database character set.

Character Sets | Recommendations

Use AL32UTF8

- National character set AL32UTF16

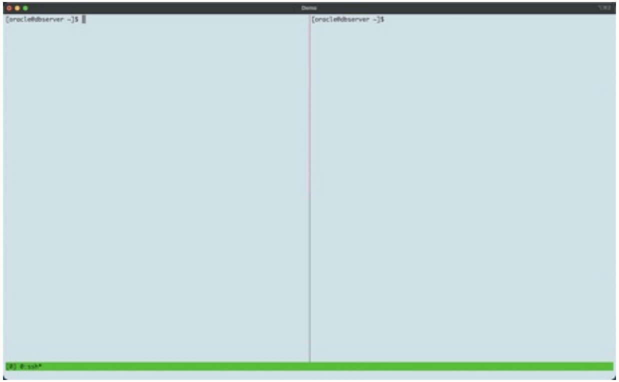
Multitenant can mix different character sets

- Available since Oracle Database 12.2.0.1
- CDB$ROOT must be created with AL32UTF8
- New PDBs will be provisioned with AL32UTF8

So will my data move without issues to AL32UTF8?

It depends …

Character Sets | Demo



```
[oracle@dbserver ~]$
```

```
[oracle@dbserver ~]$
```

```
[0] 0:ssh*
```

Watch on YouTube

Database Migration Assistant for Unicode

1

SCAN

Non-intrusive scan
of the database

2

REPORT

Types of findings:

- Need no conversion
- Needs conversion
- Invalid binary representation
- Exceeds column limit
- Exceeds data type limit

3

FIX

Before migration
or as part of the migration

Use Database Migration Assistant for Unicode before you export data to a different character set

When to use DMU vs Data Pump

- DMU can analyze the database for potential issues.
- DMU can handle many things directly in the database
- DMU is convenient
- Use Data Pump when it’s better to import into a fresh database
- Data Pump provides control and addresses niche requirements

If 4000 Character Limit is a Problem

- convert to CLOB
- Move to Extended Varchar 2 and do proper application testing.

Exercise caution if your partitioning key is a CHAR or VARCHAR2 column

- Comparison using binary collation might change in which partition data is stored

Datatype Comparison Rules | How it works

In binary collation (default), Oracle compares character values like binary values

```
SQL> select testcol,dump(testcol,1016) from test;

   TESTCOL                                   DUMP(TESTCOL,1016)
__________ _________________________________________________
a          Typ=1 Len=1 CharacterSet=AL32UTF8: 61
b          Typ=1 Len=1 CharacterSet=AL32UTF8: 62
```

In AL32UTF, `a < b` because `61 < 62`

https://docs.oracle.com/en/database/oracle/oracle-database/19/sqlrf/Data-Type-Comparison-Rules.html

Datatype Comparison Rules | What if ..

```
SQL> CREATE TABLE MY_TABLE
  2  (
  3    MY_CODE     VARCHAR2(7 CHAR),
  4    MY_DATA     VARCHAR2(30 CHAR)
  5  )
  6  PARTITION BY RANGE (MY_CODE)
  7  (
  8    PARTITION P_PART1 VALUES LESS THAN ('1'),
  9    PARTITION P_PART2 VALUES LESS THAN ('A'),
 10    PARTITION P_PART3 VALUES LESS THAN ('€'),
 11    PARTITION P_MAX VALUES less than (maxvalue)
 12* );

Table MY_TABLE created.
```

https://docs.oracle.com/en/database/oracle/oracle-database/19/sqlrf/Data-Type-Comparison-Rules.html

Datatype Comparison Rules | What if ..

```
SQL> insert into my_table values('¥',1);

1 row inserted.
```

- In AL32UTF8:
  - ¥ = c2,a5
  - € = e2,82,ac
  - ¥ < €
- In WE8ISO8859P15:
  - ¥ = a5
  - € = a4
  - ¥ > €

https://docs.oracle.com/en/database/oracle/oracle-database/19/sqlrf/Data-Type-Comparison-Rules.html

Datatype Comparison Rules | What if ..

AL32UTF8:

```
SQL> SELECT t1.my_code, t1.my_data, t2.subobject_name
  2  FROM   my_table t1, user_objects t2
  3* WHERE  dbms_rowid.rowid_object(t1.rowid) = t2.object_id;

  MY_CODE     MY_DATA     SUBOBJECT_NAME
__________ __________ _________________
¥          1          P_PART3
```

WE8ISO8859P15:

```
SQL> SELECT t1.my_code, t1.my_data, t2.subobject_name
  2  FROM   my_table t1, user_objects t2
  3* WHERE  dbms_rowid.rowid_object(t1.rowid) = t2.object_id;

  MY_CODE     MY_DATA     SUBOBJECT_NAME
__________ __________ _________________
¥          1          P_MAX
```

https://docs.oracle.com/en/database/oracle/oracle-database/19/sqlrf/Data-Type-Comparison-Rules.html

Character Sets | LOBs

CLOB data is stored in a format that is compatible with UCS-2 if the database character set is multibyte

Character Sets | Demo

```
[oracle@dbserver ~]$
```

```
[oracle@dbserver ~]$
```

```
[0] 0:ssh*
```

Watch on YouTube

Character Sets | LOBs

LOB data takes up at least twice as much space

When doing a database sizing estimate, take it into consideration

Example: CLOB uses 128 MB in WE8ISO8859P15

- Estimate 256 MB in AL32UTF8

real world scenarios

DATA PUMP

best practices

INTRO

UPGRADE

MOVE

STATISTICS

LOBS

CHARACTERSET

AUTONOMOUS

PARTITIONS

VERIFICATION

Data Pump is the ideal tool to move to Autonomous Database

Data Pump | Migration to ADB

Data Pump is:

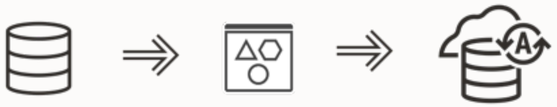
- Universal
- Platform independent
- Release independent

Integrated with:

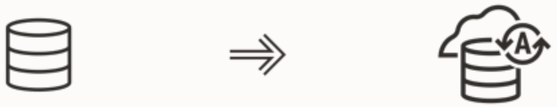
- Zero Downtime Migration
- SQL Developer
- Database Migration Service

Data Pump | Migration to ADB

With Object Storage



Without Object Storage



1. Perform a Data Pump schema mode export
2. Create an Object Storage Credential on target
3. Copy dump files to Object Storage
4. Import to Autonomous Database

1 Perform a Data Pump schema mode export

```
expdp sh/sh@orcl \
schemas=sh \
exclude=cluster,indextype,db_link \
parallel=16 \
dumpfile=export%L.dmp \
encryption_pwd_prompt=yes
```

Import Data Using Oracle Data Pump on Autonomous Database

2 Create Object Storage Credential on target

```
BEGIN
  DBMS_CLOUD.CREATE_CREDENTIAL(
    credential_name => 'obj_store_cred',
    username => 'adb_user@example.com',
    password => 'password'
  );
END;
/
```

Import Data Using Oracle Data Pump on Autonomous Database

3 Copy dump files to Object Storage

Use a tool that supports Swift such as curl

```
curl -v -X PUT -u '<user>:<SWIFT token>' --upload-file <local
file location> https://objectstorage.us-ashburn-
1.oraclecloud.com/n/namespace-string/b/bucketname/o/
export1.dmp
```

Note:
curl does not support wildcards or substitution characters. Use multiple curl commands or a script that supports substitution characters

Copy Files to the Object Storage

4 Import to Autonomous Database

```
impdp admin/password@db2022adb_high \
     directory=data_pump_dir \
     credential=obj_store_cred \
     dumpfile=https://objectstorage..../bucketname/o/export%L.dmp \
     parallel=16 \
     encryption_pwd_prompt=yes
```

Import Data Using Oracle Data Pump on Autonomous Database

As of Oracle Database 21c, you can export directly into OCI Object Storage

- Eliminates the step of copying dump files

```
expdp hr
            DEFAULT_DIRECTORY=dir1
            DUMPFILE=https://objectstorage.us-ashburn-
            1.oraclecloud.com/n/namespace-
            string/b/bucketname/o/exp%u.dmp
            CREDENTIAL=user-credential
            schemas=schema_name,
            exclude=cluster, db_link parallel=#
            encryption_pwd_prompt=yes
```

`CREDENTIAL` is your object store authentication

- It expects `DUMPFILE` to be a comma-delimited string of URIs

`DUMPFILE` is the URI for dump files in the object storage

`DEFAULT_DIRECTORY` is the location of the local log files

`LOGFILE` allows directory object name as part of the file name

Documentation: Database Utilities Guide: `CREDENTIAL`, `DUMPFILE`

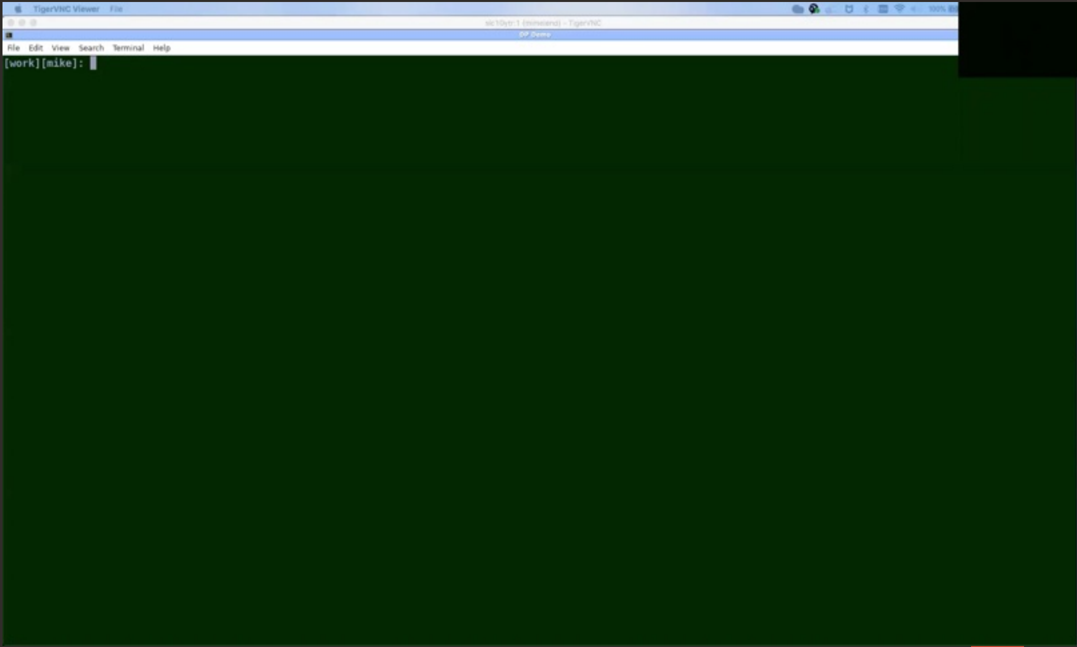
DEMO: 21c Export to the Oracle Object Store & Import to ADB

```
[work][mike]:
```

As of Oracle Database 19c, network mode imports are supported to Autonomous Database

```
impdp admin/password@db2022adb_high \
     schema=schema_name \
     network_link=<link_name> \
     parallel=n \
     transform=segment_attributes:n \
     exclude=cluster \
     nologfile=yes
```

DEMO: 19c Network Mode import into ADB



```
[work][mike]: 
```

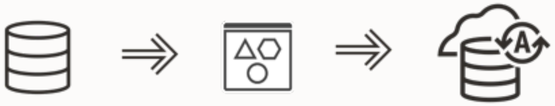
How about `DBMS_CLOUD.EXPORT_DATA (format => json_object('type' value 'datapump');`

- Cannot be imported using Oracle Data Pump `impdp` – not a dumpfile
- On ADB-S import using `DBMS_CLOUD.COPY_DATA` or `DBMS_CLOUD.CREATE_EXTERNAL_TABLE`
- Other Oracle Database, import using `ORACLE_DATAPUMP` access driver

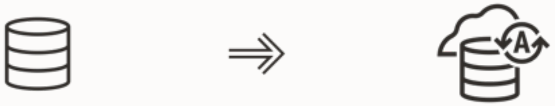
Documentation: EXPORT_DATA Procedure

Data Pump | Migration to ADB

With Object Storage



Without Object Storage



Using this cool hack you can bypass object storage when you import into Autonomous Database

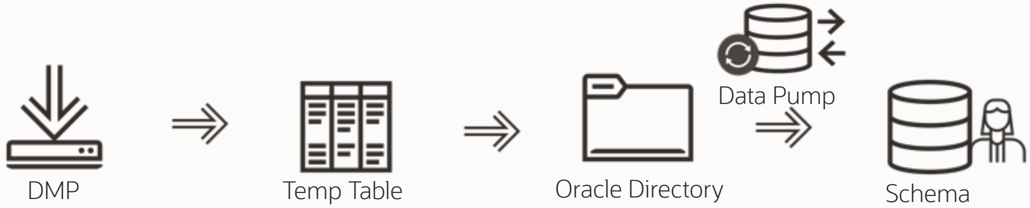
- Applies to export as well

ADB - no Object Storage | Hacks

You will need at least:

- `CREATE SESSION`
- `CREATE TABLE`
- `READ` and `WRITE` in directory (e.g. `DATA_PUMP_DIR`)

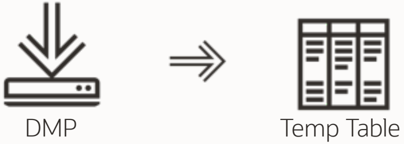
ADB - no Object Storage | Import Strategy



```
DBMS_LOB.READ + UTL_FILE.PUT_RAW

CREATE TABLE TEMP_LOB (
        file_name varchar2(1000) primary key,
        blob_content BLOB
);
```

ADB - no Object Storage | Moving file to BLOB

- Missing step:



ADB - no Object Storage | Moving file to BLOB

Strategies:

1. sqlldr
2. sqlcl
3. Base64 decode / encode

ADB - no Object Storage | Moving file to BLOB

Using sqlldr

- `lob_test.ctl`

```
LOAD DATA
INFILE 'lob_test_data.txt'
append
INTO TABLE lob_tab
FIELDS TERMINATED BY ','
(file_name CHAR(100),
blob_content LOBFILE(file_name) TERMINATED BY EOF)
```

ADB - no Object Storage | Moving file to BLOB

Using sqlldr

```
$ echo 'mydump_meta_backup_20220606_175153.dmp' > lob_test_data.txt
$ sqlldr /@adb_tp control=lob_test.ctl log=lob_test.log bad=lob_test.bad
```

```
SQL> select file_name from lob_tab;
FILE_NAME
----------------------------------------
mydump_meta_backup_20220606_175153.dmp
```

https://www.dbarj.com.br/en/2022/06/how-to-run-impdp-in-adb-when-you-dont-have-access-to-object-storage-or-db-links/

ADB - no Object Storage | Moving file to BLOB

Strategies:

1. sqlldr
2. sqlcl
3. Base64 decode / encode

ADB - no Object Storage | Moving file to BLOB

sqlcl can run JavaScript

- Create javascript code that writes a BLOB into a table
- Example: `upload_file.js`

```
SQL> script
2 ctx.write('My first script\n');
3 /
My first script
SQL>
```

ADB - no Object Storage | Moving file to BLOB

sqlcl:

```
function putFile(filename) {

var blob = conn.createBlob();
var stream = blob.setBinaryStream(0);
var path = java.nio.file.FileSystems.getDefault().getPath(filename);
java.nio.file.Files.copy(path, stream);
stream.flush();

var ret=util.execute(
'insert into lob_tab(file_name, blob_content) values (:file_name , :blob_content)',
{ file_name : filename,
blob_content : blob }
);

if (!ret) {
print('Something unintended happened.');
}

}

putFile('mydump_meta_backup_20220606_175153.dmp');
conn.commit();
```

ADB - no Object Storage | Moving file to BLOB

sqlcl can run JavaScript

Create javascript code that writes a BLOB into a table

Example: `upload_file.js`

Call the created *`js`* from sqlcl

ADB - no Object Storage | Moving file to BLOB

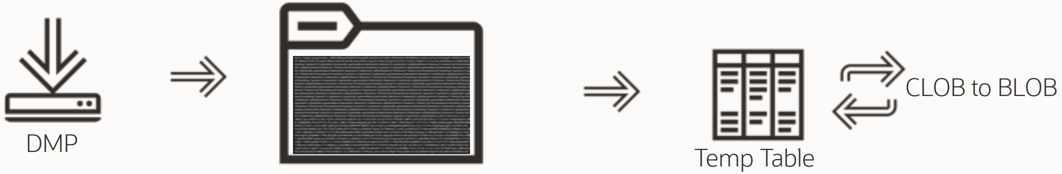
```
SQL> script upload_file.js
SQL>
SQL> select file_name from lob_tab;
FILE_NAME
-----------------------------------------
mydump_meta_backup_20220606_175153.dmp
```

ADB - no Object Storage | Moving file to BLOB

Strategies:

1. sqlldr
2. sqlcl
3. Base64 decode / encode

ADB - no Object Storage | Moving file to BLOB



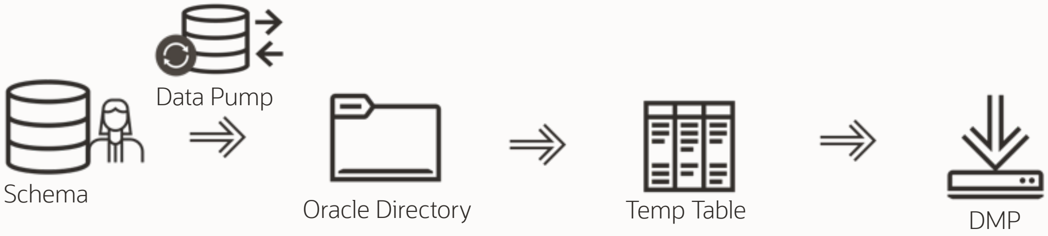
```
$ base64 -i mydump_meta_backup_20220606_175153.dmp
```

```
CREATE TABLE TEMP_LOB (
        file_name varchar2(1000) primary key,
        clob_content CLOB
        blob_content BLOB
);
```

Can you export the same way?

ADB - no Object Storage | Export Strategy

Same strategy but in the opposite direction!



real world scenarios

DATA PUMP

best practices

INTRO

UPGRADE

MOVE

STATISTICS

LOBS

CHARACTERSET

AUTONOMOUS

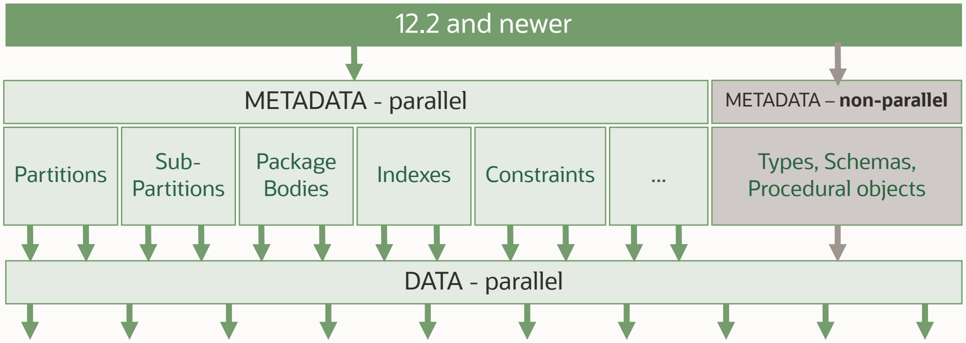
PARTITIONS

VERIFICATION

Parallel | Metadata Import - Recap

Since 12.2: Metadata import happens concurrently

- Most metadata & data objects are imported in parallel when `PARALLEL=2` or greater

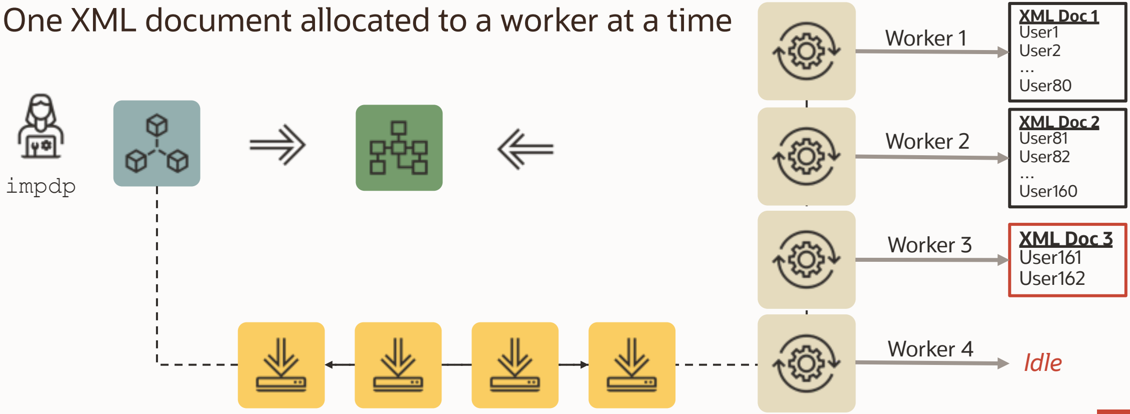


Parallel | Metadata Import - Recap

Metadata is exported in XML documents into dumpfile

- Each XML document contains N objects of a given type

One XML document allocated to a worker at a time



Limitations on parallelism

A recap

Network import does not support loading metadata in parallel

No parallelism on BasicFile LOBs

Convert *old* BasicLOBs to SecureFile LOBs

No parallelism in 19c for metadata export and import with Transportable Tablespaces

As of Oracle Database 21c, full support for parallel metadata export and import in all modes

- Applies to Transportable Tablespaces as well

Parallel Import | Existing Tables

Table
Partition
Subpartition

Known or trusted	Full parallel
Unknown	One worker only

Parallel Import | Existing Tables

Like any process that writes to the database, Data Pump import must acquire a lock on the table, partition or subpartition being loaded

If Data Pump knows that the partitioning scheme on the existing table is identical to that of the table being imported, full parallelism is used

Otherwise, only one worker will access the table at a time

This prevents contention over table locks

Parallel Import | Example

Simple hash-partitioned table:

```
CREATE TABLE dept(department_id NUMBER(4) NOT NULL,
                  department_name VARCHAR2(30))
     PARTITION BY HASH(department_id) PARTITIONS 16;
```

Populate with data:

```
insert into departments_hash (department_id, department_name) values (1,’Logistics’);
…
insert into departments_hash (department_id, department_name) VALUES (16,'Advertising’);
…
insert into departments_hash select * from departments_hash;
insert into departments_hash select * from departments_hash;
…
```

Parallel Import | Existing Tables

Table
Partition
Subpartition

Known or trusted	Full parallel
Unknown `table_exists_action=truncate`	One worker only

Parallel Import | Existing Tables

TRUNCATE

```
20-MAR-23 22:44:42.720: Starting "SYSTEM"."SYS_IMPORT_TABLE_01":  system/******** tables=departments_hash parallel=16
dumpfile=departments_hash%u.dmp directory=mydir metrics=y logtime=all table_exists_action=truncate logfile=imp_truncate.log
...
20-MAR-23 22:44:45.292: W-12 . . imported "SYSTEM".”DEPT":"SYS_P1352"        0 KB        0 rows in 0 seconds using automatic
20-MAR-23 22:44:54.077: W-1 . . imported "SYSTEM"."DEPT":"SYS_P1347"     413.1 MB 28246016 rows in 10 seconds using external_table
20-MAR-23 22:45:01.369: W-9 . . imported "SYSTEM".”DEPT":"SYS_P1343"     304.8 MB 18808832 rows in 7 seconds using external_table
20-MAR-23 22:45:07.833: W-8 . . imported "SYSTEM".”DEPT":"SYS_P1348"     270.0 MB 18874368 rows in 6 seconds using external_table
20-MAR-23 22:45:11.339: W-3 . . imported "SYSTEM".”DEPT":"SYS_P1345"     162.0 MB 9437184 rows in 4 seconds using external_table
20-MAR-23 22:45:15.468: W-6 . . imported "SYSTEM".”DEPT":"SYS_P1351"     153.0 MB 9437184 rows in 4 seconds using external_table
…
20-MAR-23 22:45:33.856: W-5      Completed 16 TABLE_EXPORT/TABLE/TABLE_DATA objects in 49 seconds
20-MAR-23 22:45:34.058: Job "SYSTEM"."SYS_IMPORT_TABLE_01" successfully completed at Mon Mar 20 22:45:34 2023 elapsed 0 00:00:52
```

From the timestamps we can see that each partition is imported serially

Parallel Import | Existing Tables

Table
Partition
Subpartition

Known or trusted
`table_exists_action=replace`

Full parallel

Unknown

One worker only

Parallel Import | Existing Tables

REPLACE

```
20-MAR-23 22:45:57.265: Starting "SYSTEM"."SYS_IMPORT_TABLE_01":  system/******** tables=departments_hash parallel=16
dumpfile=departments_hash%u.dmp directory=mydir metrics=y logtime=all table_exists_action=replace logfile=imp_replace.log
...
20-MAR-23 22:46:00.140: W-14 . . imported "SYSTEM"."DEPT":"SYS_P1352"         0 KB       0 rows in 0 seconds using automatic
20-MAR-23 22:46:06.293: W-3 . . imported "SYSTEM"."DEPT":"SYS_P1344"     135.0 MB 9437184 rows in 7 seconds using direct_path
20-MAR-23 22:46:06.309: W-2 . . imported "SYSTEM"."DEPT":"SYS_P1345"     162.0 MB 9437184 rows in 7 seconds using direct_path
20-MAR-23 22:46:06.311: W-11 . . imported "SYSTEM"."DEPT":"SYS_P1349"     135.0 MB 9437184 rows in 7 seconds using direct_path
20-MAR-23 22:46:06.811: W-5 . . imported "SYSTEM"."DEPT":"SYS_P1355"     117.0 MB 9437184 rows in 7 seconds using direct_path
20-MAR-23 22:46:06.818: W-8 . . imported "SYSTEM"."DEPT":"SYS_P1351"     153.0 MB 9437184 rows in 7 seconds using direct_path
...
20-MAR-23 22:46:11.107: W-6      Completed 16 TABLE_EXPORT/TABLE/TABLE_DATA objects in 12 seconds
20-MAR-23 22:46:11.292: Job "SYSTEM"."SYS_IMPORT_TABLE_01" successfully completed at Mon Mar 20 22:46:11 2023 elapsed 0 00:00:15
```

- This time we see partitions imported in parallel
- But, what if you don't want Data Pump to create the table?

Parallel Import | Existing Tables

Table
Partition
Subpartition

Known or trusted — Full parallel

```
table_exists_action=truncate
data_options=trust_existing_table_partitions
```

Unknown — One worker only

Parallel Import | Existing Tables

TRUST_EXISTING_TABLE_PARTITIONS (1)

```
20-MAR-23 22:46:41.572: Starting "SYSTEM"."SYS_IMPORT_TABLE_01":  system/******** tables=departments_hash parallel=16
dumpfile=departments_hash%u.dmp directory=mydir metrics=y logtime=all table_exists_action=truncate logfile=imp_trust.log
data_options=TRUST_EXISTING_TABLE_PARTITIONS
...
20-MAR-23 22:46:45.354: W-13 . . imported "SYSTEM"."DEPT":"SYS_P1352"         0 KB       0 rows in 0 seconds using automatic
20-MAR-23 22:46:55.085: W-10 . . imported "SYSTEM"."DEPT":"SYS_P1345"     162.0 MB 9437184 rows in 10 seconds using external_table
20-MAR-23 22:46:55.437: W-3 . . imported "SYSTEM"."DEPT":"SYS_P1344"     135.0 MB 9437184 rows in 11 seconds using external_table
20-MAR-23 22:46:55.439: W-11 . . imported "SYSTEM"."DEPT":"SYS_P1349"     135.0 MB 9437184 rows in 11 seconds using external_table
20-MAR-23 22:46:55.676: W-5 . . imported "SYSTEM"."DEPT":"SYS_P1351"     153.0 MB 9437184 rows in 11 seconds using external_table
20-MAR-23 22:46:55.820: W-4 . . imported "SYSTEM"."DEPT":"SYS_P1354"     144.0 MB 9437184 rows in 11 seconds using external_table
...
20-MAR-23 22:46:11.107: W-6      Completed 16 TABLE_EXPORT/TABLE/TABLE_DATA objects in 12 seconds
20-MAR-23 22:46:11.292: Job "SYSTEM"."SYS_IMPORT_TABLE_01" successfully completed at Mon Mar 20 22:46:11 2023 elapsed 0 00:00:15
```

Again we see partitions imported in parallel, but the existing table was re-used

Parallel Import | Example – Change the Partitioning Scheme

Simple hash-partitioned table:

```
CREATE TABLE dept(department_id NUMBER(4) NOT NULL,
            department_name VARCHAR2(30))
     PARTITION BY HASH(department_name) PARTITIONS 16;
```

Now import again with `TRUST_EXISTING_TABLE_PARTITIONS` and see what happens

Parallel Import | Existing Tables

TRUST_EXISTING_TABLE_PARTITIONS (2)

```
20-MAR-23 23:18:34.980: Starting "SYSTEM"."SYS_IMPORT_TABLE_01":  system/******** tables=departments_hash parallel=16
dumpfile=departments_hash%u.dmp directory=mydir metrics=y logtime=all table_exists_action=truncate logfile=imp_trust.log
data_options=TRUST_EXISTING_TABLE_PARTITIONS
...
20-MAR-23 23:18:39.225: ORA-31693: Table data object "SYSTEM"."DEPARTMENTS_HASH":"SYS_P1345" failed to load/unload and is being
skipped due to error:
ORA-02149: Specified partition does not exist
20-MAR-23 23:18:39.236: ORA-31693: Table data object "SYSTEM"."DEPARTMENTS_HASH":"SYS_P1347" failed to load/unload and is being
skipped due to error:
ORA-02149: Specified partition does not exist
...
```

The difference in partitioning scheme is detected and data is not imported!

real world scenarios

DATA PUMP

best practices

Data Pump Error Messages

Are there error messages we can ignore?

Was this job successful?

```
ORA-31634: job already exists
ORA-06512: at "SYS.DBMS_SYS_ERROR", line 79
ORA-06512: at "SYS.KUPV$FT", line 1159
ORA-31637: cannot create job Q1_AGG_TRAF_ROAMER_0 for user KGRONAU
ORA-06512: at "SYS.DBMS_SYS_ERROR", line 95
ORA-06512: at "SYS.KUPV$FT", line 1152
```

What about this one?

```
ORA-31626: job does not exist
ORA-31633: unable to create master table "KGRONAU.Q2_AGG_IND_MO_27"
ORA-06512: at "SYS.DBMS_SYS_ERROR", line 95
ORA-06512: at "SYS.KUPV$FT", line 1163
ORA-00955: name is already used by an existing object
ORA-06512: at "SYS.KUPV$FT", line 1056
ORA-06512: at "SYS.KUPV$FT", line 1044
```

Or this one?

```
ORA-31693: Table data object "APPLSYS"."FND_LOG_MESSAGES" failed
           to load/unload and is being skipped due to error:
ORA-02354: error in exporting/importing data
ORA-01555: snapshot too old: rollback segment number 25
           with name "_SYSSMU25_1608416701$" too small
```

Data Pump finished successfully

But can you trust the expdp/impdp log files?

Log file example | Fatal error

```
ORA-39126: Worker unexpected fatal error in
KUPW$WORKER.LOCATE_DATA_FILTERS
[TABLE_DATA:“KGRONAU"."REACT":"REACT_20220501_1"]
SELECT COUNT(*) FROM DUAL WHERE :1  IN ('REACT_20220605_2758',
'REACT_20220605_2757','REACT_20220605_2756','REACT_20220605_2633',
...
ORA-06512: at "SYS.KUPW$WORKER", line 6415
----- PL/SQL Call Stack -----
  object      line  object
  handle    number  name
5e4ffba30     15370  package body SYS.KUPW$WORKER
…
…
Job "KGRONAU"."Q2_FACT_MOV_SRV_RE_175" stopped due to fatal error at
06:27:31
```

Log file example | More errors

```
…
ORA-31684: Object type USER:"KGRONAU" already exists
…
ORA-39111: Dependent object type
ALTER_FUNCTION:"KGRONAU"."GETDDL_F$" skipped, base object type
FUNCTION:"KGRONAU"."TPGETDDL_F$" already exists
…
ORA-39082: Object type VIEW:"KGRONAU"."MyCaseSensitiveView" created with compilation
warnings
Processing object type SCHEMA_EXPORT/PACKAGE/PACKAGE_BODY
ORA-39346: data loss in character set conversion for object
...
ORA-01653: unable to extend table KGRONAU.MYTABLE by 8192 in tablespace KGRONAU
ORA-39171: Job is experiencing a resumable wait.
...
ORA-12899: value too large for column COD_PAIS_A2 (actual: 3, maximum: 2)
...
ORA-39083: Object type REF_CONSTRAINT:“KGRONAU"."R_CALCEVIK" failed to create with error:
ORA-02298: cannot validate (SCDAT.R_GLTRANS_CALCEVIK) - parent keys not found

Job "KGRONAU"."EDU12_SCHEMA" completed with 42 error(s)
```

And even when it completed successfully ...

Log file example | Successful completion

```
Job "KGRONAU"."Q2_AGG_MYTMN_SENTM_10"
successfully completed at Thu Aug 11 00:23:34 2022 elapsed 0 00:03:59
```

Are you sure? Really??

Log file example | Look closer ...

```
...
W-1      Completed 1 TABLE objects in 17 seconds
W-1      Completed by worker 1 1 TABLE objects in 17 seconds
W-1 Processing object type TABLE_EXPORT/TABLE/TABLE_DATA
…
W-1 . . imported "KGRONAU"."AGG_MYTMN_SENTMSG_D1":"AGG_MYTMN_SENTMSG_D1_20071024"  13.34 KB
231 rows in 2 seconds using external_table
W-1 . . imported "KGRONAU"."AGG_MYTMN_SENTMSG_D1":"AGG_MYTMN_SENTMSG_D1_20071029"  13.34 KB
0 out of 231 rows in 0 seconds using external_table
W-1 . . imported "KGRONAU"."AGG_MYTMN_SENTMSG_D1":"AGG_MYTMN_SENTMSG_D1_20071008"  13.26 KB
0 out of 228 rows in 0 seconds using external_table
...
W-1 Processing object type TABLE_EXPORT/TABLE/GRANT/OWNER_GRANT/OBJECT_GRANT
W-1      Completed 30 OBJECT_GRANT objects in 0 seconds
W-1      Completed by worker 1 30 OBJECT_GRANT objects in 0 seconds
W-1      Completed 98 TABLE_EXPORT/TABLE/TABLE_DATA objects in 2 seconds
…
Job "KGRONAU"."Q2_AGG_MYTMN_SENTM_10" successfully completed at Thu Aug 11 00:23:34 2022
elapsed 0 00:03:59
```

Challenges

How do you validate expdp/impdp results?

Comparison possibilities

$$\begin{tikzcd}\text{Rows} \arrow[r, Rightarrow] & \text{Objects} \arrow[r, Rightarrow] & \text{Content}\end{tikzcd}$$

Comparison possibilities

Rows
Objects
Content

Comparison| Row counts

```
cat MyImpDp.dplog  | grep -w imported | grep -w rows | awk '{print $5,$8}' >myfile


…
"KGRONAU"."AGG_MYTMN_SENTMSG_D1":"AGG_MYTMN_SENTMSG_D1_20071024" 231
"KGRONAU"."AGG_MYTMN_SENTMSG_D1":"AGG_MYTMN_SENTMSG_D1_20071029" 0
…
```

Comparison| Row counts

```
Export Datapump:
cat expdp.dplog |grep -w exported | grep -w rows |awk ...  >>rowcount.txt
cat rowcount.txt |sort -k 1 >>rowcount_src.txt

Import Datapump:
cat impdp.dplog |grep -w imported | grep -w rows |awk ...  >>rowcount.txt
cat rowcount.txt |sort -k 1 >>rowcount_trg.txt

Differences:
diff rowcount_src.txt rowcount_trg.txt
```

How can you validate the amount of rows in a GoldenGate based scenario?

- Important for ZDM Logical Migrations too

Comparison| Row counts

```
SQL> select 'SELECT /*+ PARALLEL(16) */ '|| chr(39)||owner ||'.'||table_name ||'
number of rows: '|| chr(39) || '|| count(1) from 'as rowcount_stmts from
dba_tables where owner='KGRONAU';
||owner ||'."'||table_name||'";'

ROWCOUNT_STMTS
--------------------------------------------------------------------
SELECT /*+ PARALLEL(16) */ 'KGRONAU.DUMMY number of rows: '|| count(1) from
KGRONAU."DUMMY";

SELECT /*+ PARALLEL(16) */ 'KGRONAU.HASH_TEST number of rows: '|| count(1) from
KGRONAU."HASH_TEST";
```

Comparison| Row counts

```
SQL> set heading off
SQL> spool rowcount_src.log
SQL> SELECT /*+ PARALLEL(16) */ 'KGRONAU.DUMMY number of rows: '|| count(1) from
KGRONAU."DUMMY";

KGRONAU.DUMMY number of rows: 6

SQL> SELECT /*+ PARALLEL(16) */ 'KGRONAU.HASH_TEST number of rows: '|| count(1)
from KGRONAU."HASH_TEST";

KGRONAU.HASH_TEST number of rows: 5
```

Comparison possibilities

Rows $\Rightarrow$ **Objects** $\Rightarrow$ Content

How can you validate objects?

Recompile invalid objects

- `@?/rdbms/admin/utlrp`
- `@?/rdbms/admin/utlprp` *n*

Comparison| Object validation with MINUS query

```
select owner c1, object_type c3, object_name c2 from
dba_objects where status != 'VALID’

minus

select owner c1, object_type c3, object_name c2 from
dba_objects@sourcedb where status != 'VALID';
```

Comparison| Object validation with subquery

```
select 'alter VIEW '||co||'.'||cn||' compile; '
                    ||CHR(13)||chr(10)||
       'select line, text from dba_errors where owner=
                   '||''''||co||''''||' and
name='||''''||cn||''''||'
       order by line;'
from (
      select owner co, object_type ct, object_name cn from
      dba_objects where status = 'INVALID' and object_type='VIEW'
      minus
      select owner co, object_type ct, object_name cn from
      dba_objects@sourcedb where status = 'INVALID'  and
      object_type='VIEW' ) ;
```

Comparison| Object validation with a view

```
alter VIEW KGRONAU.MIRROR compile;

select line, text from dba_errors where owner= 'KGRONAU'
and name='MIRROR' order by line;
```

Comparison| Objects - constraints

```
select table_name,count(table_name) from dba_constraints@sourcedb
      where owner='KGRONAU'
      and constraint_name not like 'BIN%'
      group by table_name
minus
select table_name,count(table_name) from dba_constraints
      where owner='KGRONAU'
      and constraint_name not like 'BIN%'
      group by table_name;


TABLE_NAME                COUNT(TABLE_NAME)
------------------------- -----------------
MYTABLE                                   1
```

Comparison possibilities

$$\begin{tikzcd} \text{Rows} \arrow[r, Rrightarrow] & \text{Objects} \arrow[r, Rrightarrow] & \textbf{Content} \end{tikzcd}$$

How can you guarantee data on source matches data on target exactly?

Comparison| Data validation

DBMS_COMPARISON

```
       DBMS_COMPARISON.CREATE_COMPARISON

       DBMS_COMPARISON.COMPARE

      (DBMS_COMPARISON.CONVERGE)
```

Comparison| Create comparison

```
BEGIN
  DBMS_COMPARISON.CREATE_COMPARISON
   ( comparison_name => 'mytable_cmp1',
     schema_name     => 'kgronau',
     object_name     => 'mytable',
     dblink_name     => 'sourcedb'  );
END;
/
```

Comparison| Execute comparison

```
SET SERVEROUTPUT ON
DECLARE
   consistent BOOLEAN;
   scan_info DBMS_COMPARISON.COMPARISON_TYPE;
BEGIN
   consistent := DBMS_COMPARISON.COMPARE
                 (  comparison_name  => 'mytable_cmp1',
                    scan_info        => scan_info,
                    perform_row_dif  => TRUE   );
   DBMS_OUTPUT.PUT_LINE('Scan ID: '||scan_info.scan_id);
   IF consistent=TRUE THEN DBMS_OUTPUT.PUT_LINE('No differences
were found.');
     ELSE DBMS_OUTPUT.PUT_LINE('Differences were found.');
   END IF;
END;
/
```

Comparison| Query differences

```
SELECT
   c.OWNER AS COMPARISON_OWNER, c.COMPARISON_NAME, c.SCHEMA_NAME, c.OBJECT_NAME,
   s.CURRENT_DIF_COUNT AS DIFFERENCES
FROM DBA_COMPARISON c
   INNER JOIN DBA_COMPARISON_SCAN s
      ON c.COMPARISON_NAME = s.COMPARISON_NAME AND c.OWNER = s.OWNER
WHERE s.SCAN_ID = 6;


COMPARISON_OWNER     COMPARISON_NAME           SCHEMA_NAME     OBJECT_NAME     DIFFERENCES
-------------------- ------------------------- --------------- --------------- -----------
KGRONAU              MYTABLE_CMP1              KGRONAU         MYTABLE                   1
```

Comparison| List differences

```
SELECT c.COLUMN_NAME AS IndexColumn, r.INDEX_VALUE AS IndexValue,
   DECODE(r.LOCAL_ROWID,NULL,'-','+') AS LocalRowExists,
   DECODE(r.REMOTE_ROWID,NULL, '-','+') AS RemoteRowExists
FROM DBA_COMPARISON_COLUMNS c
   INNER JOIN DBA_COMPARISON_ROW_DIF r
      ON c.COMPARISON_NAME = r.COMPARISON_NAME AND c.OWNER = r.OWNER
   INNER JOIN DBA_COMPARISON_SCAN s ON r.SCAN_ID = s.SCAN_ID
WHERE
   c.COMPARISON_NAME = 'MYTABLE_CMP1' AND s.PARENT_SCAN_ID = 6 AND
   r.STATUS = 'DIF' AND c.INDEX_COLUMN = 'Y'
ORDER BY r.INDEX_VALUE;

INDEXCOLUMN     INDEXVALUE      LOCALROWEXISTS  REMOTEROWEXISTS
--------------- --------------- --------------- ----------------------------
COL1            10              +               -
```

Comparison| Fix differences

```
DECLARE
   scan_info DBMS_COMPARISON.COMPARISON_TYPE;
BEGIN
   DBMS_COMPARISON.CONVERGE
   (
      comparison_name    => 'MYTABLE_CMP1',
      scan_id            => 6,
      scan_info          => scan_info,
      converge_options   => DBMS_COMPARISON.CMP_CONVERGE_LOCAL_WINS
   );
   DBMS_OUTPUT.PUT_LINE('Local Rows Merged:   '||scan_info.loc_rows_merged);
   DBMS_OUTPUT.PUT_LINE('Remote Rows Merged:  '||scan_info.rmt_rows_merged);
   DBMS_OUTPUT.PUT_LINE('Local Rows Deleted:  '||scan_info.loc_rows_deleted);
   DBMS_OUTPUT.PUT_LINE('Remote Rows Deleted: '||scan_info.rmt_rows_deleted);
END;
/

Local Rows Merged:   0
Remote Rows Merged:  1
Local Rows Deleted:  0
Remote Rows Deleted: 0
```

Comparison| Final validation

```
SQL> select * from mytable@sourcedb
  2  minus
  3  select * from mytable;

      COL1 COL2
---------- ----------------
         1 Hello World!


SQL> select * from mytable
  2  minus
  3  select * from mytable@sourcedb;

no rows selected
```

Data validation | Demo

Watch on YouTube

Data validation | Oracle GoldenGate Veridata

GoldenGate Veridata

https://www.oracle.com/at/a/ocom/docs/middleware/veridata-datasheet.pdf

Comparison| Alternative options

DBMS_CRYPTO package

STANDARD_HASH function

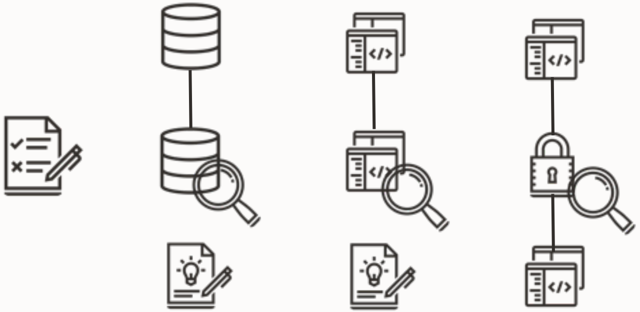
Data Validation | DMBS_CRYPTO Package

```
select col1,col2, rawtohex(DBMS_CRYPTO.Hash (UTL_I18N.STRING_TO_RAW
(col2, 'AL32UTF8'), 2)) as hash2 from HASH_TEST;

      COL1 COL2                 HASH2
---------- -------------------- -----------------------------------
         0 Hello world!!        1D94DD7DFD050410185A535B9575E184
         1 Hello world!         86FB269D190D2C85F6E0468CECA42A20
         2 Hello world!!        1D94DD7DFD050410185A535B9575E184
         3 Hello world!!        1D94DD7DFD050410185A535B9575E184
         4 Hello world!!        1D94DD7DFD050410185A535B9575E184
```

Data Validation | STANDARD_HASH Function

```
select col1, col2 ,STANDARD_HASH (col2 , 'MD5' ) HASH2 FROM
kgronau.hash_test;
      COL1 COL2                 HASH2
---------- -------------------- --------------------------------
         0 Hello world!!        1D94DD7DFD050410185A535B9575E184
         1 Hello world!         86FB269D190D2C85F6E0468CECA42A20
         2 Hello world!!        1D94DD7DFD050410185A535B9575E184
         3 Hello world!!        1D94DD7DFD050410185A535B9575E184
         4 Hello world!!        1D94DD7DFD050410185A535B9575E184
```

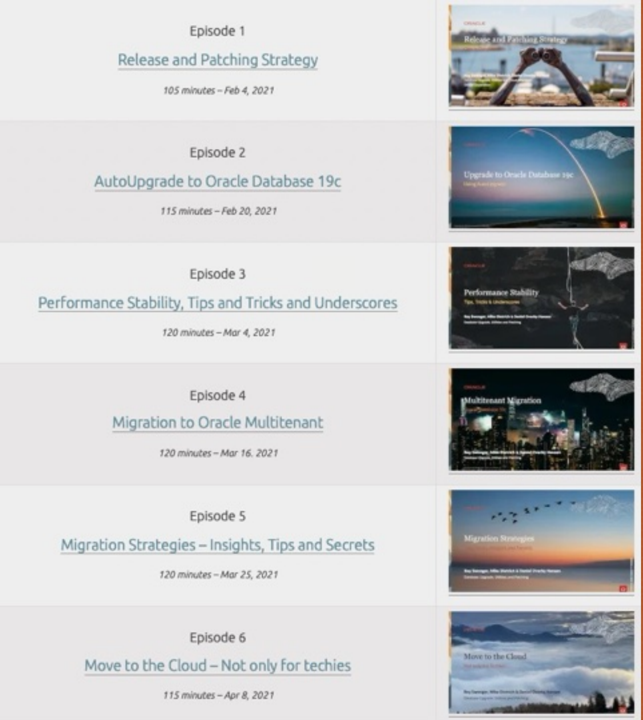
Summary | Comparison and Validation



Validation is
time consuming

Often it can't be done
for all rows in all tables

Photo by Zbynek Burival on Unsplash



Episode 1

Release and Patching Strategy

105 minutes – Feb 4, 2021

Episode 2

AutoUpgrade to Oracle Database 19c

115 minutes – Feb 20, 2021

Episode 3

Performance Stability, Tips and Tricks and Underscores

120 minutes – Mar 4, 2021

Episode 4

Migration to Oracle Multitenant

120 minutes – Mar 16. 2021

Episode 5

Migration Strategies – Insights, Tips and Secrets

120 minutes – Mar 25, 2021

Episode 6

Move to the Cloud – Not only for techies

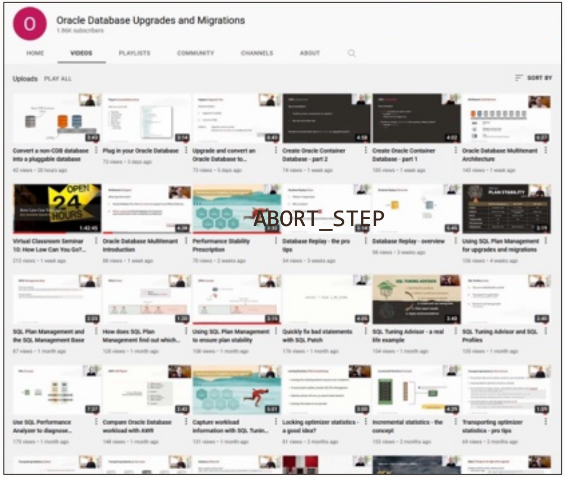
115 minutes – Apr 8, 2021

Recorded Web Seminars

https://MikeDietrichDE.com/videos

More than 30 hours of technical content, on-demand, anytime, anywhere

YouTube | Oracle Database Upgrades and Migrations


Oracle Database Upgrades and Migrations
1.86K subscribers

HOME VIDEOS PLAYLISTS COMMUNITY CHANNELS ABOUT

Uploads PLAY ALL SORT BY

Convert a non-CDB database into a pluggable database
Plug in your Oracle Database
Upgrade and convert an Oracle Database to...
Create Oracle Container Database - part 2
Create Oracle Container Database - part 1
Oracle Database Multitenant Architecture
Virtual Classroom Seminar 10: How Low Can You Go?...
Oracle Database Multitenant Introduction
Performance Stability Prescription
Database Replay - the pro tips
Database Replay - overview
Using SQL Plan Management for upgrades and migrations
SQL Plan Management and the SQL Management Base
How does SQL Plan Management find out which...
Using SQL Plan Management to ensure plan stability
Quickly fix bad statements with SQL Patch
SQL Tuning Advisor - a real life example
SQL Tuning Advisor and SQL Profiles
Use SQL Performance Analyzer to diagnose...
Compare Oracle Database workload with AWR
Capture workload information with SQL Tunin...
Locking optimizer statistics - a good idea?
Incremental statistics - the concept
Transporting optimizer statistics - pro tips

ABORT_STEP

Link

- 200+ videos
- New videos every week
- No marketing
- No buzzword
- All tech

SCAN ME

THANK YOU

Visit our blogs:

https://MikeDietrichDE.com

https://DOHdatabase.com

https://www.dbarj.com.br/en

THANK YOU

Webinars:

https://MikeDietrichDE.com/videos

YouTube channel:

OracleDatabaseUpgradesandMigrations

THANK YOU

Release and Patching Strategy for Oracle Database 23c

May 10, 2023 – 16:00h CET

THANK YOU