

The Oracle logo is displayed in a red, sans-serif font. It is positioned in the upper left corner of the slide, partially overlaid by a decorative graphic consisting of a red and orange patterned shape and a dark blue wavy line.

ORACLE

Oracle Data Pump

New Features, Internals and Secrets



Roy Swonger

Vice President



royfswonger



Daniel Overby Hansen

Senior Principal Product Manager



dohdatabase



@dohdatabase



<https://dohdatabase.com>

Architecture

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

Data Pump New Features in Release 23c



New dump file format

- Needed for Native OCI API for the Object Store
- Replaces previous Swift API


```
# New format - default
# Trailer block format = 6.1 dump file version
expdp ... version=23
```

```
# Header block format (legacy) = 5.1 dump file version
expdp ... version=19
```

Pro Tip: [Doc ID 864582.1](#)
discusses header blocks



Data Pump is fully backwards compatible

- Import from previous releases
- Export to previous release using **version** parameter

Data Pump | Dump File Format

- When exporting to the object store, Data Pump chooses the dump file format based on your credential format
- OCI Native API supports only `version=23`
- SWIFT API supports all versions
- Export using `version=19` and OCI Native credentials aborts:
ORA-39463 "header block format is not supported for object-store URI dump file"



Support for SQL domains and boolean and vector data types



New transformation allows
removal of sharding on import

- `TRANSFORM=INCLUDE_SHARDING_CLAUSES:[Y|N]`



New transformation allows
removal of ACDR metadata on import

- `TRANSFORM=OMIT_ACDR_METADATA:[Y|N]`

Data Pump New Features in Release 21c



Universal Data Pump client

- Client and database release no longer have to match

-- Include and exclude keywords are no longer mutually exclusive
-- Works for all object paths and on import as well.

expdp ... include=tables exclude=statistics



Transportable jobs are restartable

```
-- Any transportable jobs can now run in parallel  
-- Parallel unload/load of metadata provide a significant performance boost
```

```
expdp ... full=y transportable=always parallel=16
```

```
expdp ... tablespace=<list> parallel=16
```

```
impdp ... parallel=16
```


Parallel Transportable | Benchmark

Oracle E-Business Suite database

600.000+ objects

Export parallel 1 2h 2m

Import parallel 1 6h 44m

Total 8h 46m

Export parallel 16 1h 8m

Import parallel 16 1h 23m

Total 2h 31m

Parallel Transportable | Architecture

Parallel export:

- Each worker processes an object path serially
- Parallel happens by multiple workers working on multiple object paths

Parallel import:

- Control process orders the object paths
- All workers work on one object path in parallel
- Parallel happens by all workers working on the same object path

- After export, store a checksum in the dump file.
- Detects in-flight corruption or alteration.
- Specify other algorithms using checksum_algorithm parameter.

```
expdp ... checksum=yes
```

```
impdp ... verify_checksum=yes  
         verify_only=yes
```



remap_tablespace allows % wildard

- e.g., for ADB-S, REMAP_TABLESPACE=% : DATA

The Data Pump LOB Mystery

A short history of binary data types

v4

LONG and LONG RAW

8.0

CLOB and BLOB

11g

SecureFile LOBs

v4

LONG and LONG RAW

8.0

BasicFile LOBs

11g

SecureFile LOBs

v4

LONG and LONG RAW

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

8.0

BasicFile LOBs

- Performance constraints
- No Parallel DML allowed
- Max size: (4GB - 1) * DB_BLOCK_SIZE

11g

SecureFile LOBs

- Improved performance
- Data Pump can use multiple workers or Parallel Query
- 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

--Converting a BasicFile LOB to SecureFile during import,
--is faster than not converting it.
--Overview of Oracle LOBs (Doc ID: 1490228.1)

`impdp ... transform=lob_storage:securefile`

Different LOB types

Internal LOBs stored **inside** the database

- CLOB
- NCLOB
- BLOB

External LOBs stored **outside** the database

- BFILE

Initialization Parameter

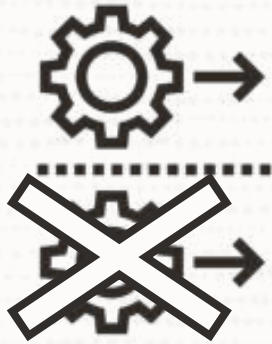
DB_SECUREFILE

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

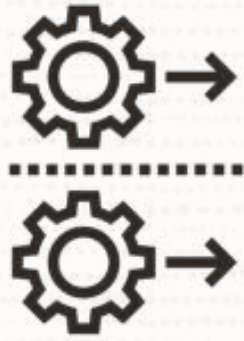
Tablespace must use Automatic Segment Space Management (ASSM)

Data Pump & LOBs

Things to know and consider



No parallelism with BasicFile LOBs

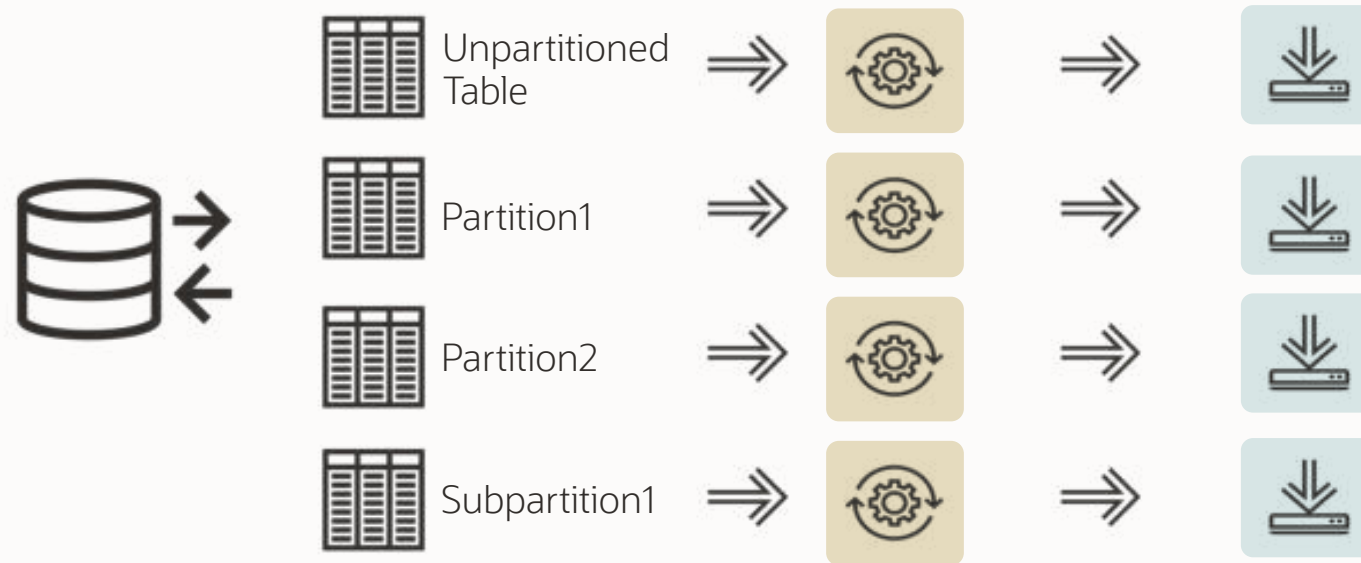


Always use SecureFile LOBs

But why is there only one worker?

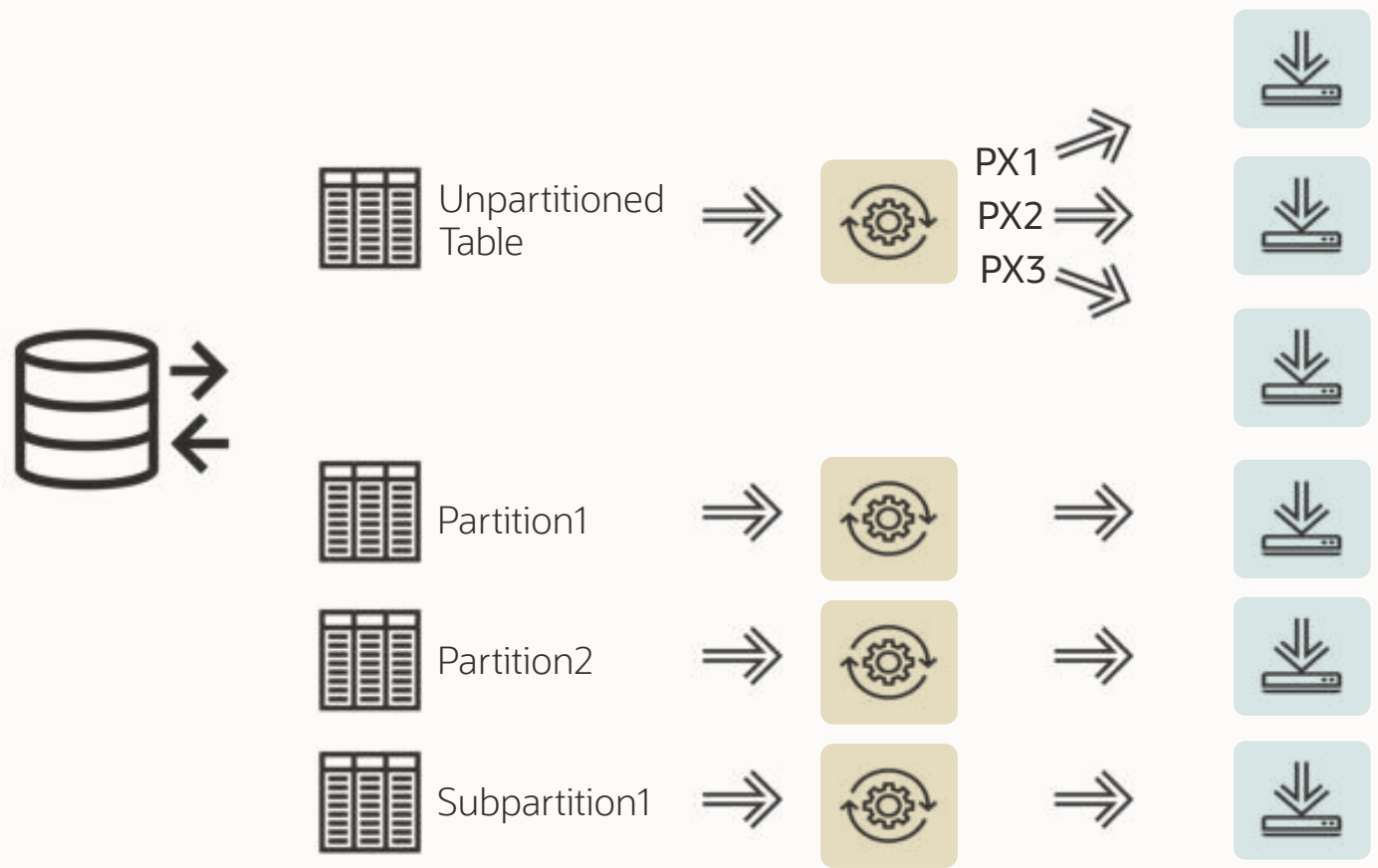
Data Pump | Parallel Worker Activity

Data Pump employs one worker per table data object



Data Pump | Worker as PQ Coordinator

If a table data object is >250MB, Data Pump can invoke parallel query



LOB Export | Example Table



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



```
CREATE TABLE tab1 ( id NUMBER, blob_data BLOB )  
LOB (blob_data) store as securefile;
```



```
BEGIN ... DBMS_LOB.LOADBLOBFROMFILE ...
```



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

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



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

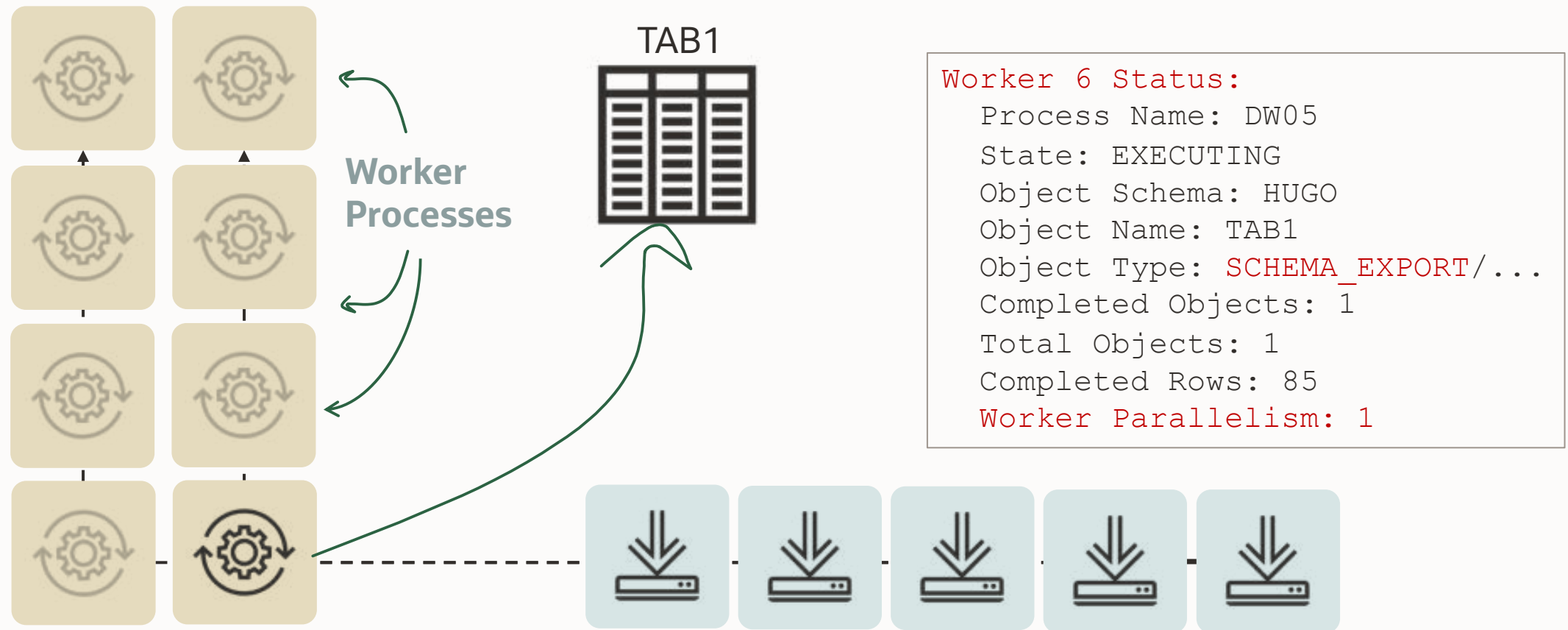
- Smaller LOBs less than 4000 byte store in-line
- Up to 8000 bytes in Oracle Database 23c

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





Maybe the table's PARALLEL DEGREE
is too low?

LOB Export | Parallel Degree



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

DEGREE

1

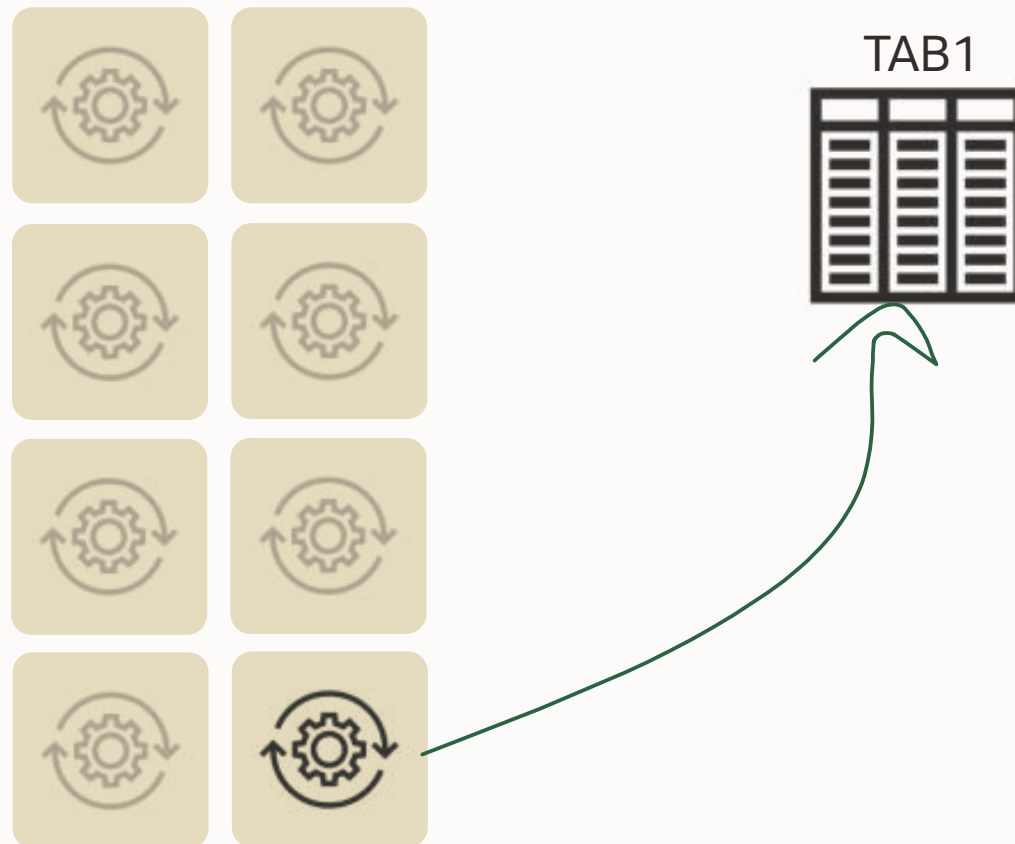
```
select degree
from DBA_TABLES
where table_name='TAB1';
```

DEGREE

8

LOB Export | Parallel Degree

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



```
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

Table



TAB1



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 as if Data Pump does not 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*
the LOB segment?



No, **not possible.**
So what's next?

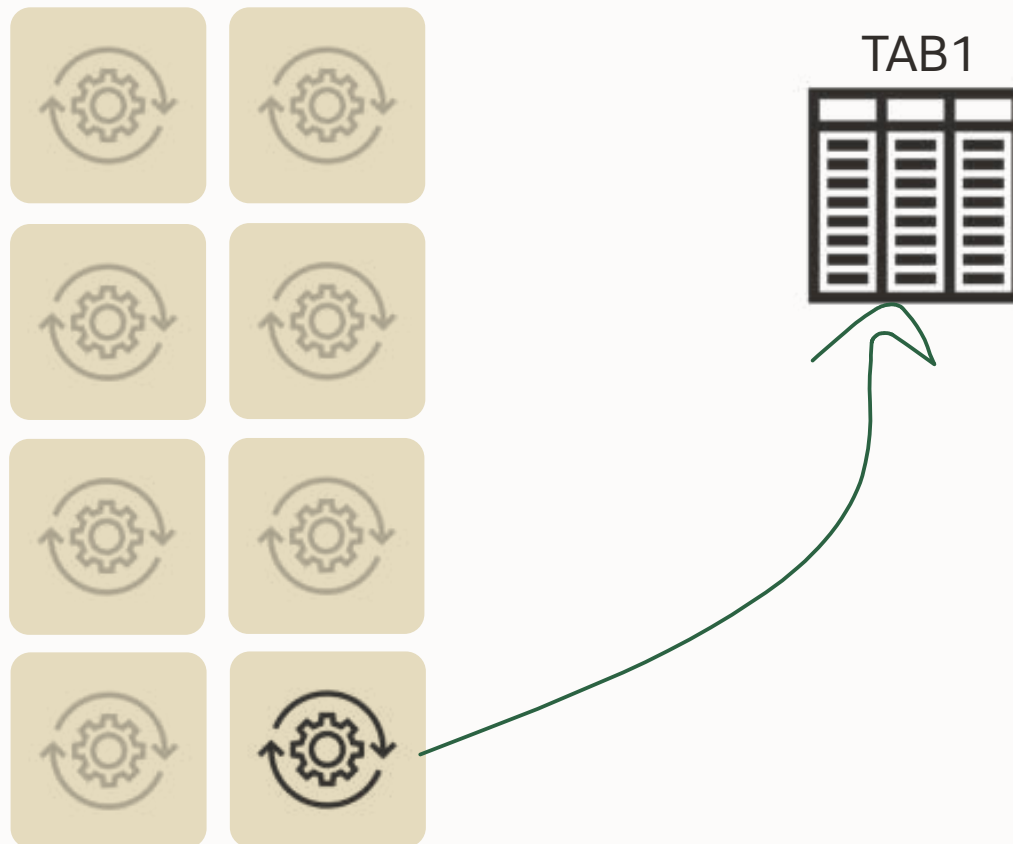
LOB Export | Manipulating Statistics



```
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
Worker Parallelism: 7
```



How about another approach ...

LOB Export | **ESTIMATE=BLOCKS**

expdp hugo/oracle **ESTIMATE=BLOCKS**...

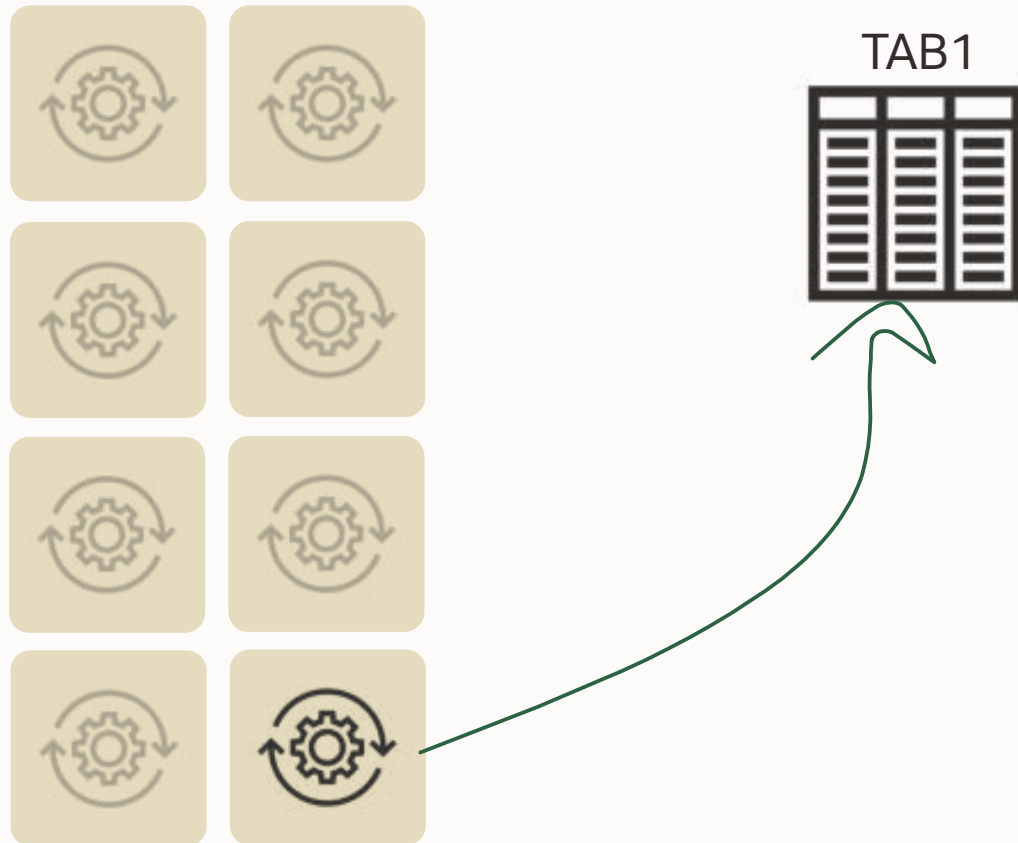


```
12-SEP-23 15:50:30.288: W-7 Startup took 0 seconds
12-SEP-23 15:50:31.409: W-1 Processing object type SCHEMA_EXPORT/TABLE/TABLE_DATA
12-SEP-23 15:50:31.735: W-1 . estimated "HUGO"."TAB1"          10.24 GB
12-SEP-23 15:50:31.735: W-1 . estimated "HUGO"."T1"           11 MB
12-SEP-23 15:50:31.735: W-1 . estimated "HUGO"."T4"            7 MB
12-SEP-23 15:50:31.735: W-1 . estimated "HUGO"."T2"            4 MB
12-SEP-23 15:50:31.735: W-1 . estimated "HUGO"."T5"          256 KB
12-SEP-23 15:50:31.735: W-1 . estimated "HUGO"."T3"            64 KB
.
.
.
```

Tech Tip: Make sure you are on 19.18 or later with the Data Pump Bundle Patch installed!

LOB Export | **ESTIMATE=BLOCKS**

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
Worker Parallelism: 7
```



How do we get more workers to
export data?



Boost parallelism by using
partitioned tables

LOB Export | Which Approach is Better?

Faking Statistics

- Must be done for each table
- Requires testing to get best result
- Could be overwritten by stats gathering

ESTIMATE=BLOCKS

- Just one parameter for the whole export
- Estimate phase adds time to export
- Requires patch applied in 19c

We are working on an approach that combines the best of both. Stay tuned to the upgrade blog!

LOB Export | What if you still have BasicFiles LOBs?

Exporting

- Multiple Data Pump jobs in parallel exporting subsets of rows

```
expdp parallel=1 table=t1 query="where <subset 1>"  
expdp parallel=1 table=t1 query="where <subset 2>"  
expdp parallel=1 table=t1 query="where <subset 3>"  
expdp parallel=1 table=t1 query="where <subset 4>"
```

Importing

- Convert to SecureFile LOBs

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

Pro Tip: [Blog post](#) with examples

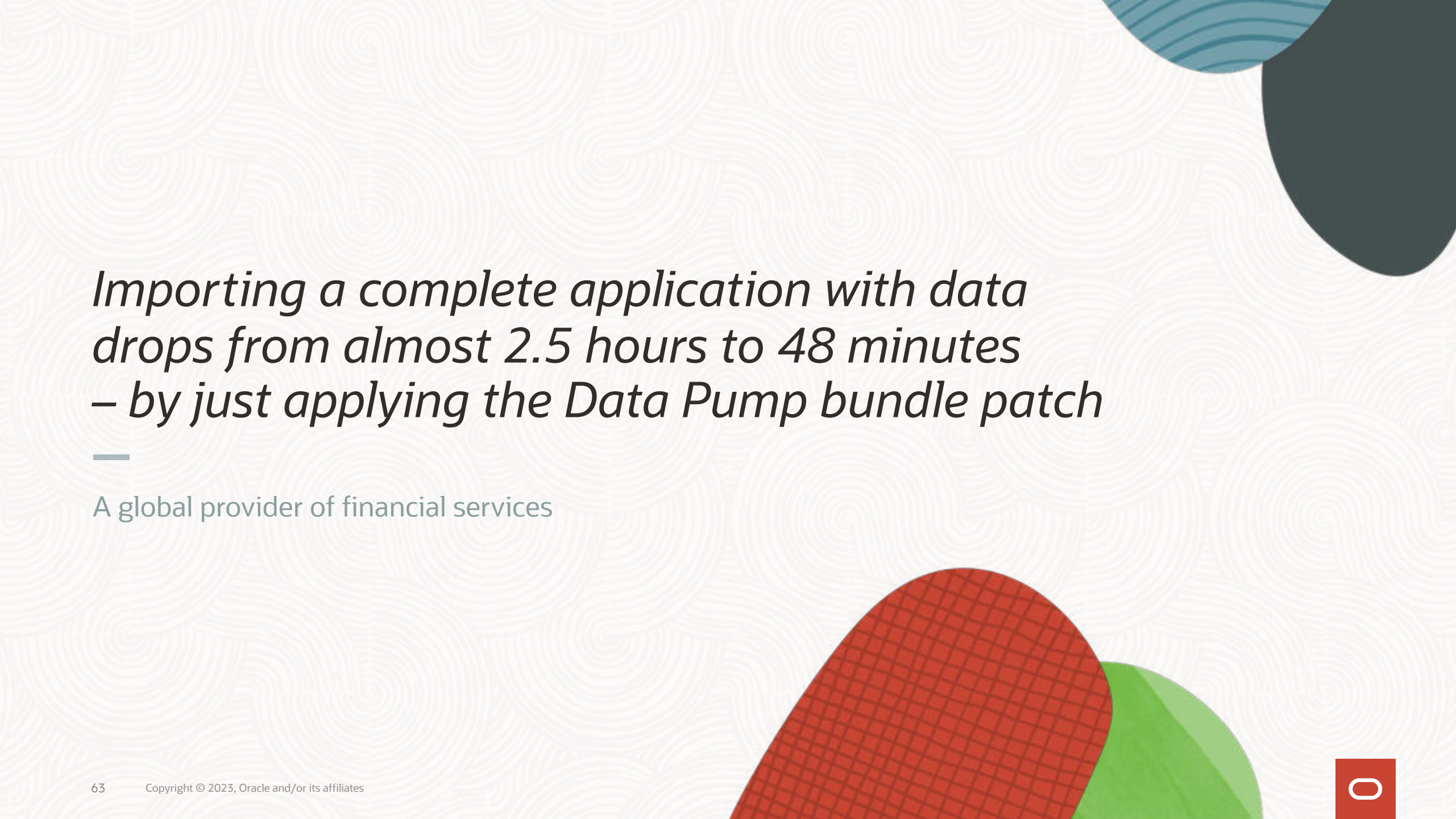
Tips and tricks

if you remember
only **one thing**



Install the Data Pump Bundle Patch

- Contains 166 bug fixes in 19.21.0
- Download from MOS Doc ID [2819284.1](#)



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

A global provider of financial services



Ensure dictionary and fixed objects statistics are accurate

- Before export
- Before import
- Immediately after import


```
begin
```

```
--dbms_stats.gather_dictionary_stats;
```

```
dbms_stats.gather_schema_stats('SYS');
```

```
dbms_stats.gather_schema_stats('SYSTEM');
```

```
dbms_stats.gather_fixed_objects_stats;
```

```
end;
```

```
/
```



```
begin
  --dbms_stats.gather_dictionary_stats;
  dbms_stats.gather_schema_stats('SYS');
  dbms_stats.gather_schema_stats('SYSTEM');
  dbms_stats.gather_fixed_objects_stats;
end;
/
```

"After gathering dictionary stats, our Data Pump export went from 46 to 8 minutes"



Data Pump is hanging - what's going on?

- How to troubleshoot



Attach to a running job and
use the interactive command mode


```
$ expdp user/password@alias ...
```

```
Export: Release 23.0.0.0.0 - Production on Tue Oct 31 14:56:06 2023  
Version 23.3.0.23.09
```

```
Copyright (c) 1982, 2023, Oracle and/or its affiliates. All rights reserved.  
Connected to: Oracle Database 23c EE High Perf Release 23.0.0.0.0 - Production  
31-OCT-23 14:56:13.420: Starting "SYSTEM"."SYS_EXPORT_FULL_01"  
31-OCT-23 14:56:13.799: W-1 Startup on instance 1 took 0 seconds  
31-OCT-23 14:56:30.550: W-2 Startup on instance 1 took 0 seconds  
31-OCT-23 14:56:38.519: W-3 Startup on instance 1 took 0 seconds  
31-OCT-23 14:56:38.529: W-4 Startup on instance 1 took 0 seconds
```



```
$ expdp user/password@alias attach=SYSTEM.SYS_EXPORT_FULL_01
```

```
Export> status
```

```
...
```

```
Worker 1 Status:
```

```
Instance ID: 1
```

```
Instance name: CDB23
```

```
Host name: dbs23
```

```
Object start time: Tuesday, 14 November, 2023 9:22:30
```

```
Object status at: Tuesday, 14 November, 2023 9:30:35
```

```
Process Name: DW00
```

```
State: EXECUTING
```

```
Object Schema: APPS
```

```
Object Name: AP_INVOICE_DISTRIBUTIONS_PKG
```

```
Object Type: DATABASE_EXPORT/SCHEMA/PACKAGE_BODIES/PACKAGE/PACKAGE_BODY
```

```
Completed Objects: 1,938
```

```
Worker Parallelism: 1
```



How to trace Data Pump jobs

- MOS Doc ID [286496.1](#)


```
-- Change AWR snap interval to 15 minutes and create snapshot
exec dbms_workload_repository.modify_snapshot_settings(null, 15);
exec dbms_workload_repository.create_snapshot;

-- Optionally, enable SQL trace for Data Pump processes or specific SQL ID
alter system set events 'sql_trace {process: pname = dw | process: pname = dm} level=8';
alter system set events 'sql_trace[SQL: 03g1bnw08m4ds ]';

-- Run Data Pump job with trace (Doc ID 286496.1)
expdp ... metrics=yes logfile=all trace=1FF0300
impdp ... metrics=yes logfile=all trace=1FF0300

-- Create AWR snapshot and produce AWR report
exec dbms_workload_repository.modify_snapshot_settings(null, <original-value>);
exec dbms_workload_repository.create_snapshot;
@?/rdbms/admin/awrrpt
```



```
-- Change AWR snap interval to 15 minutes and create snapshot
exec dbms_workload_repository.modify_snapshot_settings(null, 15);
exec dbms_workload_repository.create_snapshot;

-- Optionally, enable SQL trace for Data Pump processes or specific SQL ID
alter system set events 'sql_trace {process: pname = dw | process: pname = dm} level=8';
alter system set events 'sql_trace[SQL: 03g1bnw08m4ds ]';

-- Run Data Pump job with trace (Doc ID 286496.1)
expdp ... metrics=yes logfile=all trace=1FF0300
impdp ... metrics=yes logfile=all trace=1FF0300

-- Create AWR snapshot and produce AWR report
exec dbms_workload_repository.modify_snapshot_settings(null, <original-value>);
exec dbms_workload_repository.create_snapshot;
@?/rdbms/admin/awrrpt
```



```
-- Change AWR snap interval to 15 minutes and create snapshot
exec dbms_workload_repository.modify_snapshot_settings(null, 15);
exec dbms_workload_repository.create_snapshot;

-- Optionally, enable SQL trace for Data Pump processes or specific SQL ID
alter system set events 'sql_trace {process: pname = dw | process: pname = dm} level=8';
alter system set events 'sql_trace[SQL: 03g1bnw08m4ds ]';

-- Run Data Pump job with trace (Doc ID 286496.1)
expdp ... metrics=yes logfile=all trace=1FF0300
impdp ... metrics=yes logfile=all trace=1FF0300

-- Create AWR snapshot and produce AWR report
exec dbms_workload_repository.modify_snapshot_settings(null, <original-value>);
exec dbms_workload_repository.create_snapshot;
@?/rdbms/admin/awrrpt
```



```
-- Change AWR snap interval to 15 minutes and create snapshot
exec dbms_workload_repository.modify_snapshot_settings(null, 15);
exec dbms_workload_repository.create_snapshot;

-- Optionally, enable SQL trace for Data Pump processes or specific SQL ID
alter system set events 'sql_trace {process: pname = dw | process: pname = dm} level=8';
alter system set events 'sql_trace[SQL: 03g1bnw08m4ds ]';

-- Run Data Pump job with trace (Doc ID 286496.1)
expdp ... metrics=yes logfile=all trace=1FF0300
impdp ... metrics=yes logfile=all trace=1FF0300

-- Create AWR snapshot and produce AWR report
exec dbms_workload_repository.modify_snapshot_settings(null, <original-value>);
exec dbms_workload_repository.create_snapshot;
@?/rdbms/admin/awrrpt
```

Data Pump Trace

Collect:

- Data Pump log file
- AWR report
- Data Pump trace files
 - Stored in the database trace directory
 - Control process file name: ***dm***
 - Worker process file names: ***dw***



Other Sessions

How Artificial Intelligence Will Make You a Better Cloud DBA

Roy Swonger

Wednesday 22.11.2023 | 11:00 - 11:45

Blogging, the DSGVO / GDPR, pitfalls and nightmares

Mike Dietrich, Phillipe Fierens

Tuesday 21.11.2023 | 14:00 - 14:45

Out-Of-Place Patching Automation - The answer to almost everything

Mike Dietrich, Phillipe Fierens

Wednesday 22.11.2023 | 14:00 - 14:45

Thank You

