



Cross Platform Migration

Transportable Tablespaces to the Extreme



ROY SWONGER

Vice President

Database Upgrade, Utilities & Patching



royfswonger



@royfswonger





MIKE DIETRICH

Senior Director Product Management
Database Upgrade, Migrations & Patching



mikedietrich



@mikedietrichde



<https://mikedietrichde.com>



DANIEL OVERBY HANSEN

Senior Principal Product Manager
Database Upgrade, Migrations & Patching



dohdatabase



@dohdatabase



<https://dohdatabase.com>



RODRIGO JORGE

Senior Principal Product Manager
Database Upgrade, Migrations & Patching



rodrigoaraujorge



@rodrigojorgedba



<https://dbarj.com.br/en>



ALEX ZABALLA

Distinguished Product Manager
Database Upgrade, Migrations & Patching



alexzaballa



@alexzaballa



<https://alexzaballa.com>

Find Slides and Much More on Our Blogs



MikeDietrichDE.com

Mike.Dietrich@oracle.com



dohdatabase.com

Daniel.Overby.Hansen@oracle.com



DBArj.com.br

Rodrigo.R.Jorge@oracle.com



AlexZaballa.com

Alex.Zaballa@oracle.com

Webinar | Get The Slides

<https://MikeDietrichDE.com/slides>



Webinar | Multitenant



Scan me to sign up

Move to Oracle Database 23c

– Everything you need to know about Oracle Multitenant

May 16, 14:00 CET

[Sign up](#)

Episode 1

Release and Patching Strategy

105 minutes – Feb 4, 2021



Episode 2

AutoUpgrade to Oracle Database 19c

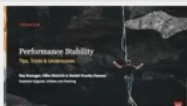
115 minutes – Feb 20, 2021



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:

- All tech, no marketing
- On-demand
- Anytime
- Anywhere





Introduction



Transportable Tablespaces



Abbreviation

XTTS

**Cross-platform
transportable tablespaces**



FTEX

Full Transportable Export/Import



Typically, you use XTTS for
cross-endian migrations

Endianness | The Basics

Big-endian

<i>increasing addresses</i> →					
...	4A _h	6F _h	68 _h	6E _h	...
...	'J'	'o'	'h'	'n'	...

Little-endian

<i>increasing addresses</i> →					
...	6E _h	68 _h	6F _h	4A _h	...
...	'n'	'h'	'o'	'J'	...

Source: <https://en.wikipedia.org/wiki/Endianness>



Big-endian

users01.dbf
users02.dbf
data01.dbf
data02.dbf
...



Little-endian

users01.dbf
users02.dbf
data01.dbf
data02.dbf
...



SuperCluster M8 systems run out of hardware maintenance in June 2024

- List of Oracle Supported Hardware with Last Ship Dates Announced (Doc ID 1450710.1)


```
SQL> select platform_name, endian_format from v$transportable_platform;
```

AIX-Based Systems (64-bit)	Big
Apple Mac OS	Big
HP-UX (64-bit)	Big
HP-UX IA (64-bit)	Big
IBM Power Based Linux	Big
IBM zSeries Based Linux	Big
Linux OS (S64)	Big
Solaris[tm] OE (32-bit)	Big
Solaris[tm] OE (64-bit)	Big
Apple Mac OS (x86-64)	Little
HP IA Open VMS	Little
HP Open VMS	Little
HP Tru64 UNIX	Little
Linux IA (32-bit)	Little
Linux IA (64-bit)	Little
Linux OS (AARCH64)	Little
Linux x86 64-bit	Little
Microsoft Windows IA (32-bit)	Little
Microsoft Windows IA (64-bit)	Little
Microsoft Windows x86 64-bit	Little
Solaris Operating System (x86)	Little
Solaris Operating System (x86-64)	Little


```
SQL> select platform_name, endian_format from v$transportable_platform;
```

AIX-Based Systems (64-bit)	Big
Apple Mac OS	Big
HP-UX (64-bit)	Big
HP-UX IA (64-bit)	Big
IBM Power Based Linux	Big
IBM zSeries Based Linux	Big
Linux OS (S64)	Big
Solaris[tm] OE (32-bit)	Big
Solaris[tm] OE (64-bit)	Big
Apple Mac OS (x86-64)	Little
HP IA Open VMS	Little
HP Open VMS	Little
HP Tru64 UNIX	Little
Linux IA (32-bit)	Little
Linux IA (64-bit)	Little
Linux OS (AARCH64)	Little
Linux x86 64-bit	Little
Microsoft Windows IA (32-bit)	Little
Microsoft Windows IA (64-bit)	Little
Microsoft Windows x86 64-bit	Little
Solaris Operating System (x86)	Little
Solaris Operating System (x86-64)	Little





Concept



Transportable Tablespace

Transportable Tablespaces | Concept

Your data



Rows

`insert into ...`
`update ...`
`delete from ...`



Metadata

`grant select on ...`
`create package ...`
`create view ...`

Transportable Tablespaces | Concept



Rows

Stored in a user tablespace



Copy the data files



Metadata

Stored in *SYSTEM* tablespace



Recreate using Data Pump



All data files must be consistent
when you copy them

- Tablespaces must be *read only*

Transportable Tablespaces | Concept

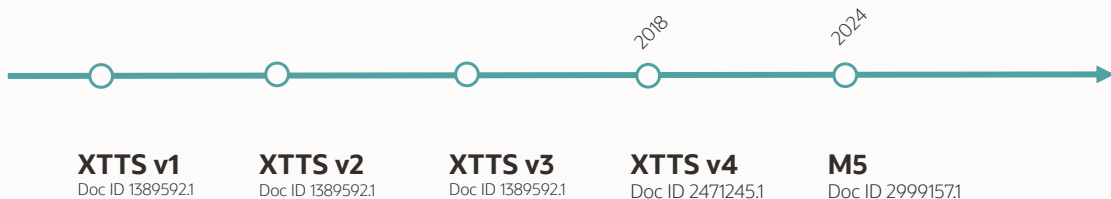
Copy the data files



- 1 File system copy
- 2 Backup / restore / recover

Transportable Tablespaces | Concept

2 Backup / restore / recover



- No multisection backups
- No encrypted tablespaces
- Inefficient parallelism
- Incomplete multitenant support

Transportable Tablespaces | Concept



Rows

Stored in a user tablespace



Copy the data files



Metadata

Stored in *SYSTEM* tablespace



Recreate using Data Pump

Transportable Tablespaces | Concept



Recreate using Data Pump

1 Traditional transportable tablespace

- Selected tablespaces
- Manual metadata recreation
- Complex and error-prone
- Requires 10g (8i in some cases)

2 Full Transportable Export/Import (FTEX)

- Entire database
- Fully automated metadata recreation
- Easy and standardized
- Source must be 11.2.0.3 or higher
- Target must be 12.1 or higher
- **Strongly recommended**

Full Transportable Export/Import

Designed to move an entire database, including:

- Profiles
- Public database links and synonyms
- Directories
- Database triggers except those owned by SYS
- SQL Management Base
 - SQL plan baselines, plan directives, profiles, patches
- And [much more](#)



M5 is the **next-generation** cross-platform transportable tablespace procedure

- New RMAN functionality combined with Full Transportable Export/Import
- Doc ID [2999157.1](#)

Further Reading | XTTS v4

[Blog posts](#)

[YouTube playlist](#)

[Webinars](#) (video and slide deck)

- Migration Strategies – Insights, Tips and Secrets
- Migrating Very Large Databases

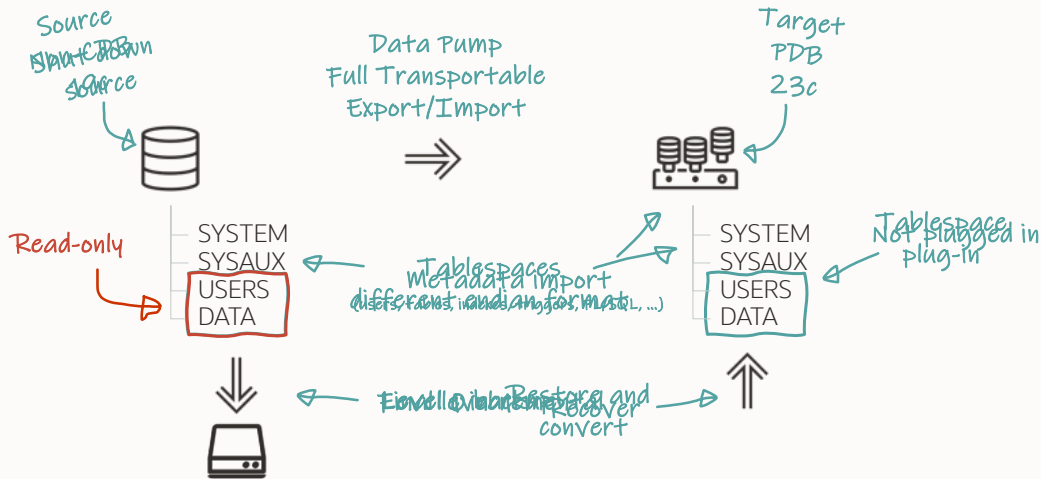
Reference cases

- Top healthcare provider: [230 TB in less than 24 hours](#)
- Pushing the limits: [ExaCC migration using a ZDLRA](#)

Further Reading | Transportable

- [Master Note for Transportable Tablespaces \(TTS\) -- Common Questions and Issues \(Doc ID 1166564.1\)](#)
- [Transportable Tablespace \(TTS\) Restrictions and Limitations: Details, Reference, and Version Where Applicable \(Doc ID 1454872.1\)](#)
- [V4 PERL Scripts to reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup \(Doc ID 2471245.1\)](#)
- [Known Issues for Cross Platform Transportable Tablespaces XTTS \(Doc ID 2311677.1\)](#)
- [Cross Platform Database Migration using ZDLRA \(Doc ID 2460552.1\)](#)
- [11G – Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup \(Doc ID 1389592.1\)](#)
- [12C – Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backup \(Doc ID 2005729.1\)](#)
- Blog post: [What Is a Self-contained Transportable Tablespace Set](#)

Concept



Concept

Transportable Tablespaces enables

- Migration to a higher database release
- Migration between non-CDB and PDB
- Cross-endian and same-endian migration
- Cross-platform and same-platform migration



Requirements

Transportable Tablespace



The target database must have the same or higher **COMPATIBLE** setting



Target database must use the same character set

- A few [exceptions](#) exist
- Applies to national character set as well


```
--To check database character set.  
--Optionally, convert source database to Unicode  
--using Oracle Database Migration Assistant for Unicode (DMU)  
  
select * from nls_database_parameters;
```


Character Set

PDBs may use a different character set than the CDB, if the CDB character set is AL32UTF8.

1. Keep *production* CDB on AL32UTF8
2. Provision *temporary* CDB with the desired character set
3. Create a new empty PDB in the *temporary* CDB
4. Clone custom PDB to *production* CDB



We recommend using the same time zone file version in the target database

--To determine the active time zone file version

```
select * from v$timezone_file;
```




Target time zone file version must be equal to or higher than source

Timestamp with Time Zone

If you have `TIMESTAMP WITH TIME ZONE (TSTZ)` and time zone file version does **not** match:

- Full transportable export/import
 - Data Pump updates `TSTZ` columns during the import
 - Time-consuming
- Traditional transportable jobs:
 - Data Pump skips the entire table on import
 - Perform regular import after migration


```
-- By default, DBCA picks the newest available time zone file  
-- To choose any time zone file during database creation  
-- https://mikedietrichde.com/2016/12/08/create-a-database-with-non-default-time-zone/
```

```
export ORA_TZFILE=$ORACLE_HOME/oracore/zoneinfo/timetzlrg_40.dat
```

```
./dbca
```




If you have **TIMESTAMP WITH LOCAL TIME ZONE**,
you must use the same **DBTIMEZONE**

--To determine the database time zone

```
select dbtimezone from v$instance;
```


Timestamp with Local Time Zone

The database time zone (DBTIMEZONE) affects only:

- `TIMESTAMP WITH LOCAL TIME ZONE (TSLTZ)`
- `CURRENT_DATE`
- `CURRENT_TIMESTAMP`
- `LOCALTIMESTAMP`



SYSDATE and **SYSTIMESTAMP** do **not** rely on database time zone (**DBTIMEZONE**)

- Determined by operating system time zone and clock

--You can change the database time zone.
--A PDB can have a different time zone.
--You can only change the database time zone if you have no TSLTZ columns.

```
alter database cdb1 set time_zone = '+02:00';
```

```
alter pluggable database pdb1 set time_zone = '+04:00';
```


Timestamp with Local Time Zone

If you have `TIMESTAMP WITH LOCAL TIME ZONE (TSLTZ)` and `DBTIMEZONE` does not match:

- Data Pump [skips the entire table](#) on import
- Perform regular import after migration



M5 Migration Script

Interview



Watch on [YouTube](#)


```
# source database
```

```
RUN
```

```
{
```

```
ALLOCATE CHANNEL d1 DEVICE TYPE DISK FORMAT '...';
```

```
ALLOCATE CHANNEL d2 DEVICE TYPE DISK FORMAT '...';
```

```
BACKUP
```

```
    FILESPERSET 1
```

```
    SECTION SIZE 64G
```

```
    TAG UP19_L0_240206101548
```

```
    TABLESPACE <list-of-tablespace>;
```

```
}
```



```
# source database
```

```
RUN
```

```
{
```

```
  ALLOCATE CHANNEL d1 DEVICE TYPE DISK FORMAT '...';
```

```
  ALLOCATE CHANNEL d2 DEVICE TYPE DISK FORMAT '...';
```

```
  BACKUP
```

```
    FILESPERSET 1
```

```
    SECTION SIZE 64G
```

```
    TAG UP19_L0_240206101548
```

```
    TABLESPACE <list-of-tablespace>;
```

```
}
```

```
# target database
```

```
RUN
```

```
{
```

```
  ALLOCATE CHANNEL DISK1 DEVICE TYPE DISK FORMAT '...';
```

```
  ALLOCATE CHANNEL DISK2 DEVICE TYPE DISK FORMAT '...';
```

```
  RESTORE ALL FOREIGN DATAFILES TO NEW FROM BACKUPSET
```

```
    '<backup-set-1>',
```

```
    '<backup-set-2>',
```

```
    ...
```

```
    '<backup-set-n>'
```

```
};
```


Benefits

M5 procedure supports:

- Encrypted tablespaces
- Multisection backups
- Migrating multiple databases into the same CDB simultaneously
- Compressed backup sets
- Better parallelism

Requirements

- Source and target database **must**
 - be 19.18.0 or higher
 - use Data Pump Bundle Patch
- Target must use Oracle Managed Files (OMF)
 - Parameter `DB_CREATE_FILE_DEST` PARAMETER

Demo



Source

- Oracle Database 19c
- Non-CDB, called *UP19*

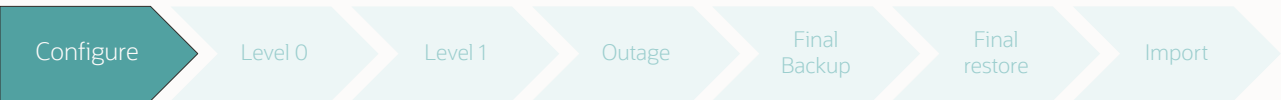
Target

- Oracle Database 19c
- CDB, called *CDB2*
- PDB, called *PDB1*

- M5 script on shared NFS mount point

Watch on [YouTube](#)

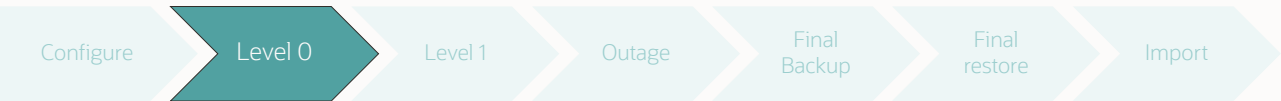
M5 Workflow



- Download M5 script from Doc ID [2999157.1](#)
- Configure shared NFS
- Edit `dbmig_ts_list.txt`
- Edit `dbmig_driver.properties`
- Create new, empty target database

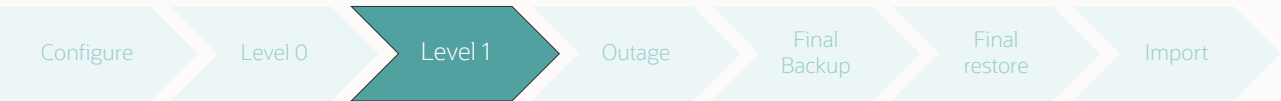


M5 Workflow



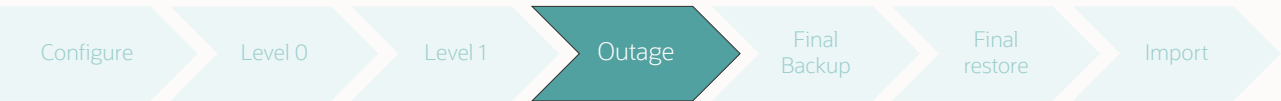
- Start initial level 0 backup
- Use driver script `dbmig_driver_m5.sh L0`
- Driver script creates a restore script
- Restore using `restore_L0_<source_sid>_<timestamp>.cmd`
- Check logs

M5 Workflow



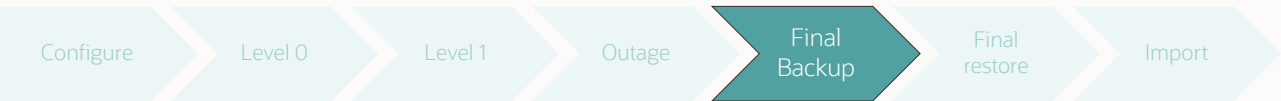
- Start level 1 incremental backup
- Use driver script `dbmig_driver_m5.sh` **L1**
- Driver script creates a restore script
- Restore using `restore_L1_<source_sid>_<timestamp>.cmd`
- Check logs
- Repeat as often as desired

M5 Workflow



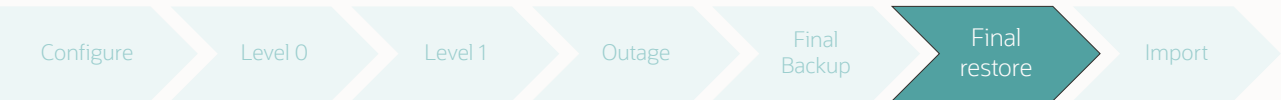
- Maintenance window begins
- Read-only sessions can still use the database

M5 Workflow



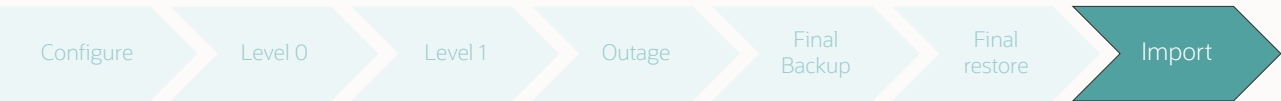
- Start final level 1 incremental backup
- Use driver script `dbmig_driver_m5.sh` **L1F**
- Sets tablespaces read-only
- Performs level 1 incremental backup
- Start Data Pump full transportable export
- Optionally, shut down source database

M5 Workflow



- Driver script created a restore script
- Restore using `restore_L1F_<source_sid>_<timestamp>.cmd`
- Check logs

M5 Workflow



- Copy Data Pump dump file to *DATA_PUMP_DIR*
- Use import driver script in test mode
- Start `impdp.sh <dump_file> <restore_log> test`
- Check generated parameter file
- Use `impdp.sh <dump_file> <restore_log> run`
- Check Data Pump log file

Recommendations

- Use a shared NFS mount point
 - Attach to source and target
 - Use for script, backups, logs, etc.
- If NFS is not possible
 - Manually copy files from source to target after each run
 - M5 can copy scripts using **DEST_SERVER** (for ZDLRA only)



Manually remove **restore point**
in target database after migration



You must manually add
new tablespaces to `dbmig_ts_list.txt`

- Failure to do so results in errors during Data Pump import



Keep backups on the source database.
RMAN might need them for recovery

- Remove the backups using proper RMAN commands



Always use the latest version of M5 script

- Download from Doc ID [29991571](#)



Best Practices



Before Migration



We **strongly** recommend
using a recent Release Update

- Never migrate without the Data Pump Bundle Patch
- Applies to source and target database

Stay Current

1. Get the latest Release Update

- Assistant: Download Reference for Oracle Database/GI Update, Revision, PSU, SPU(CPU), Bundle Patches, Patchsets and Base Releases (Doc ID [2118136.2](#))

2. Get important one-off patches

- Oracle Database 19c Important Recommended One-off Patches (Doc ID [555.1](#))
- Oracle Database Patches to Consider for 19c (Doc ID [2781612.2](#))

3. Get the Data Pump Bundle Patch

- Data Pump Recommended Proactive Patches For 19.10 and Above (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



Accurate dictionary and fixed objects stats
are vital for Data Pump performance

Statistics | When to Gather

Gather dictionary statistics:

- Reasonable time before export
- Before import
- Immediately after import

Gather fixed objects statistics:

- Reasonable time before export
- After import when the system is warmed up
 - The database must be in *production* state


```
begin
```

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

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

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

```
end;
```

```
/
```



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

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



Clean up your database

- Less data, faster migration

Clean Up

Remove unneeded data and metadata

Everything matters - but **metadata matters most**

- Metadata affects Data Pump
- Data affects the level 0 backup

Our experience shows there is **always** data and metadata to remove



Execute a dictionary check

- Use DBMS DICTIONARY CHECK
- Formerly known as *health check*


```
SQL> set serveroutput on size unlimited
SQL> execute dbms_dictionary_check.full
```

```
dbms_dictionary_check on 07-MAR-2023 03:17:48
```

```
-----
Catalog Version 21.0.0.0.0 (2300000000)
```

```
db_name: ORCL
```

```
Is CDB?: NO
```

```
Trace File: /oracle/log/diag/rdbms/orcl/orcl/trace/orcl_ora_2574906_DICTCHECK.trc
```

Procedure Name	Catalog Version	Fixed Vs Release	Timestamp	Result
...
.- OIDOnObjCol	... 2300000000	<= *All Rel*	03/07 03:17:48	PASS
.- LobNotInObj	... 2300000000	<= *All Rel*	03/07 03:17:48	PASS
.- SourceNotInObj	... 2300000000	<= *All Rel*	03/07 03:17:48	PASS
...				

Dictionary Check

Dictionary check produces a report:

- Review findings
- Optionally, use the [repair](#) option

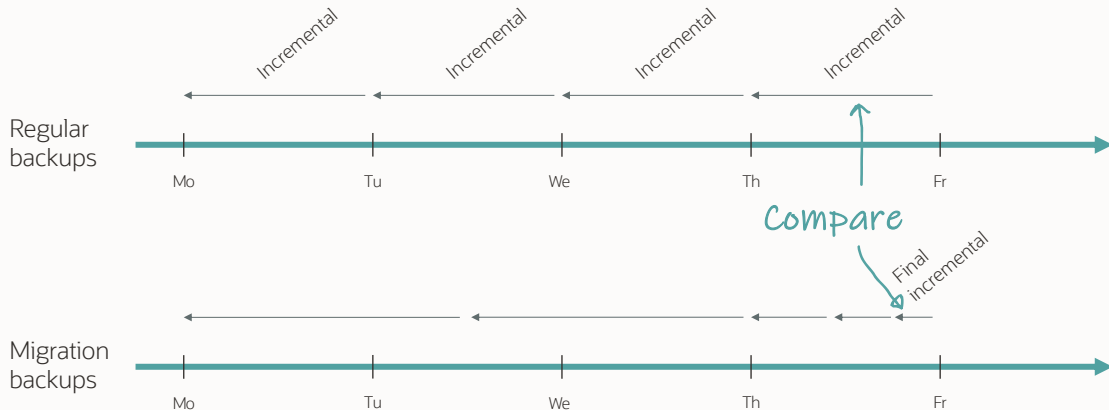
Also available through:

- [Autonomous Health Framework](#)
- [EXAchk](#)
- [ORAchk](#)
- MOS Doc ID [136697.1](#) (older releases only)



How long will the final level 1
incremental backup take?

Estimate Final Incremental



Estimate Final Incremental

Compare V\$RMAN_BACKUP_SUBJOB_DETAILS

- On the final day, you run several incrementals
- Final incremental may use more channels

Real-life example:

- Daily incremental 2 TB
- Final incremental 951 GB

Estimate Final Incremental

Don't use redo generation to estimate final level 1 backup

Real-life example:

- Redo per day 15 TB
- Daily incremental 2 TB
- Final incremental 951 GB



Best Practices

During Migration



Use Block Change Tracking for faster incremental backups

- Check the License Guide for details


```
[oracle@hol m5]$ ./dbmig_driver_m5.sh L1
```

Properties file found, sourcing.

LOG and CMD directories found

2024-02-20 06:59:56 - 1708408796651: Requested L1 backup for pid 21485. Using DISK destination, 4 channels and 64G section size.

2024-02-20 06:59:56 - 1708408796659: Performing L1 backup for pid 21485

RMAN> 2> 3> 4> RMAN> RMAN> 2> 3> 4> 5> 6> 7> 8> 9> 10> 11> 12> RMAN>

2024-02-20 07:00:55 - 1708408855044: No errors or warnings found in backup log file for ...

2024-02-20 07:00:55 - 1708408855059: Manually copy restore script to destination

2024-02-20 07:00:55 - 1708408855062: => /u01/app/oracle/m5/cmd/restore_L1_UP19_240220065956.cmd

2024-02-20 07:00:55 - 1708408855076: Saving SCN for next backup for pid 21485

BACKUP_TYPE	INPUT_BYTES(MB)	OUTPUT_BYTES(MB)	STATUS
DATAFILE FULL	4458	1.32	COMPLETED

↑ If BCT is in use,
this number decreases

Block Change Tracking

If INPUT_BYTES matches database size:

Option 1: Increase BCT bitmap size

- Increase parameter `_bct_bitmaps_per_file` (Doc ID [452455.1](#))

Option 2: Prevent standby database from updating primary bitmap

- Prevent with parameter `_disable_primary_bitmap_switch` (Doc ID [2144267.1](#))
- If changed, after switchover first incremental needs to fully scan



Converting to snapshot standby resets the change tracking bitmap

- The next incremental must fully scan the database



Block Change Tracking on standby
database requires Active Data Guard

Further Reading

Block Change Tracking Inside Out (Doc ID [1528510.1](#))



Exclude statistics from Data Pump metadata export

- Use `exclude=statistics`
- Excluded by default in M5 script

exclude=statistics



- 1 Table statistics
- 2 Index statistics
- 3 Statistics preferences
- 4 Column usage information

exclude=statistics



- 1 Table statistics
- 2 Index statistics
- 3 Statistics preferences
- 4 Column usage information



Statistics

1 Regather

Gather on database or schema using **DBMS_STATS**

Time-consuming

Resource-intensive

2 Transfer from source

Go [warp speed](#) with **CONCURRENT** and **AUTO_DEGREE**

Mostly suitable when migrating from older release



Statistics

1 Regather

Export and import using **DBMS_STATS**

Fast

2 Transfer from source

Not resource-intensive

Run [multiple imports simultaneously](#)

Check [YouTube playlist](#) and [blog post](#)

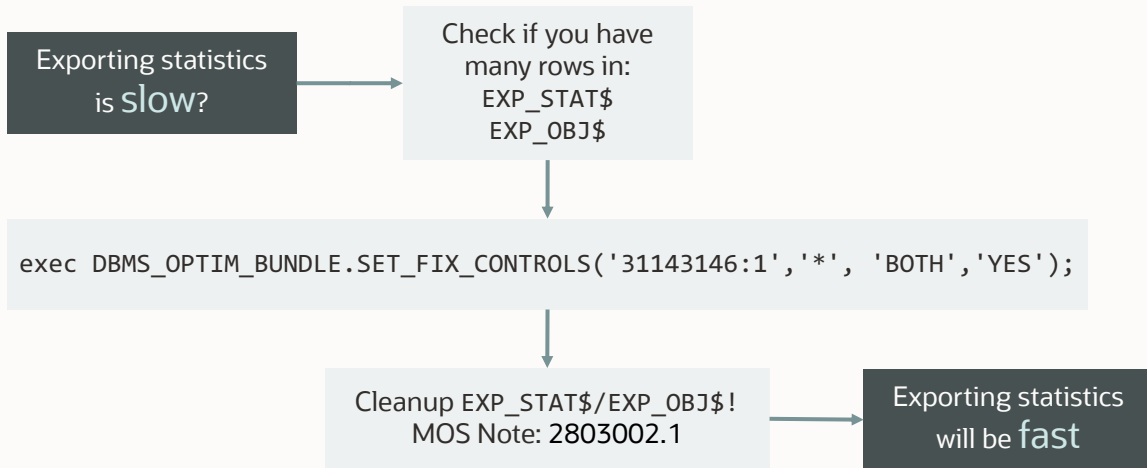
Our preferred option



Save downtime by exporting stats before the maintenance window

- Reduce tasks in the maintenance window
- Store stats staging table in a tablespace that you transport

Caution: Expression Statistics



exclude=statistics



- 1 Table statistics
- 2 Index statistics
- 3 Statistics preferences
- 4 Column usage information



Statistics Preferences

```
BEGIN
```

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

```
END;
```


Statistics Preferences

Table 171-131 SET_TABLE_PREFS Procedure Parameters

Parameter	Description
ownname	Owner name
tabname	Table name
pname	<p>Preference name. You can set the default value for following preferences:</p> <ul style="list-style-type: none">• 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.

Statistics Preferences

- If you use global statistics preferences, you must redefine those in the target database
 - Use [DBMS_STATS.SET_GLOBAL_PREFS](#)
- If you use table-level statistics preferences, you can transport those from source database
 - Use [DBMS_STATS.EXPORT_TABLE_PREFS](#)



You often use statistics preferences to solve a particular problem

- Evaluate whether that problem exists in the target environment

exclude=statistics



- 1 Table statistics
- 2 Index statistics
- 3 Statistics preferences
- 4 Column usage information

Column Usage Information

- Information on how you join tables
- Used during statistics gathering 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$`



Take care when you gather statistics
without column usage information

- Relevant only for workload that depends on histograms



EXCLUDE

EXCLUDE=STATISTICS

COL_USAGE\$ empty



REGATHER

First time only

METHOD_OPT =>
SIZE SKEWONLY



GO LIVE

Database updates
column usage
information



REGATHER

Use default

METHOD_OPT =>
SIZE AUTO



If you **include** statistics Data Pump includes everything, except global stats preferences

- Not recommended: **include=statistics**



Always add Data Pump diagnostic information to the log file

- M5 script adds `LOGTIME=ALL` and `METRICS=YES`



Use Data Pump parallel metadata export and import

- Available in Oracle Database 21c


```
-- 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

Parallel degree in export and import is completely independent

Source

Oracle Database 19c

expdp ... `parallel=1`



Target

Oracle Database 23c

impdp ... `parallel=32`

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:

- One worker plugs in tablespaces
- 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
- One worker completes the import



Ensure adequate `streams_pool_size` during Data Pump export and import

- Avoid waits caused by SGA resize operations
- 512M should be sufficient



Disable the resource manager
during the migration


```
alter system set resource_manager_plan='';
```




Disable all maintenance windows in the maintenance window group

- Disabling the group is **not** sufficient


```
--Find all the maintenance windows and disable them individually.  
--Your database might have different windows enabled, so select from the database.  
--You can also select from dba_scheduler_wingroup_members.
```

```
select window_name from dba_autotask_window_clients;
```

```
exec dbms_scheduler.disable(name => 'SYS.MONDAY_WINDOW');  
exec dbms_scheduler.disable(name => 'SYS.TUESDAY_WINDOW');  
...  
exec dbms_scheduler.disable(name => 'SYS.SUNDAY_WINDOW');
```




Do not use `SYS AS SYSDBA`
for your Data Pump jobs

Use `SYSTEM`

Check `DATAPUMP_EXP_FULL_DATABASE` / `DATAPUMP_IMP_FULL_DATABASE`

Credentials

The control table goes into the default tablespace of the user running the Data Pump job

Expect the control table to grow to several GB. At one customer it was 4 GB in size.

For import, create a dedicated user and set the default tablespace to one that you can scratch after the migration. This ensures you don't use unnecessary space in SYSTEM tablespace.



Miscellaneous parameters

Miscellaneous Parameters

- Exclude SYS when importing into a PDB:
`impdp ... exclude=sys_users`



You can restart Data Pump
transportable jobs

- Available in Oracle Database 21c

Best Practices

Practice, practice, practice

- Start on a small database
- Prove it works on a production-size database

To ensure consistency and avoid human error

- Automate
- Save all logs and output
- Data Pump, RMAN

Clean-up procedure

- In case of failure and rollback
- To repeat tests



Best Practices

After Migration

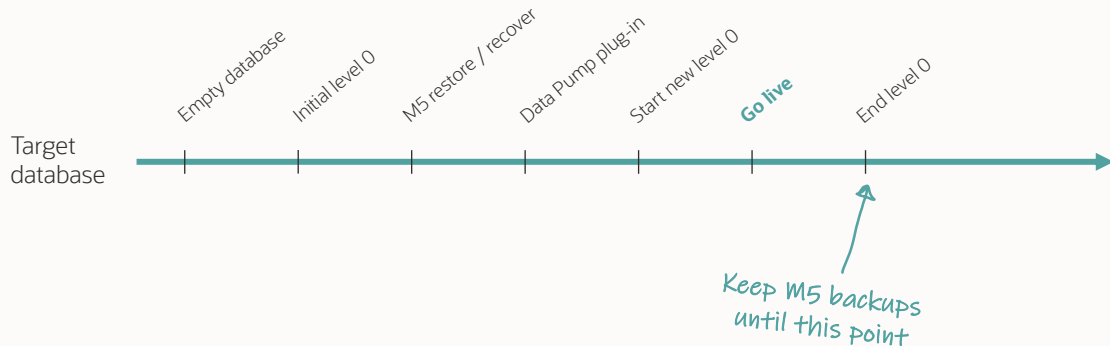


Be sure to **shut down** the source database



Keep backups on the target database
until a new level 0 backup completes

Target Backup



Target Backup

To remove M5 backups from target database use RMAN:

- DELETE
- CHANGE ... UNCATALOG



Advanced Scenarios



Using ZDLRA as a staging platform

- Instead of local storage or NFS



Use the latest **libra.so**

- On source, target and ZDLRA
- Doc ID [2219812.1](#)



Consider to stall migration backups to prevent them from being virtualized

- Doc ID [2999157.1](#), appendix 4



ZDLRA throttles the number of channels based on appliance size

- Even if you specify more channels, ZDLRA might use fewer channels
- Normally, 64 channels

ZDLRA | Additional Information

Known issue: VPC user password may not contain dollar-sign (\$)

Workaround: use a secure external password store

XTTS v4 note for ZDLRA:

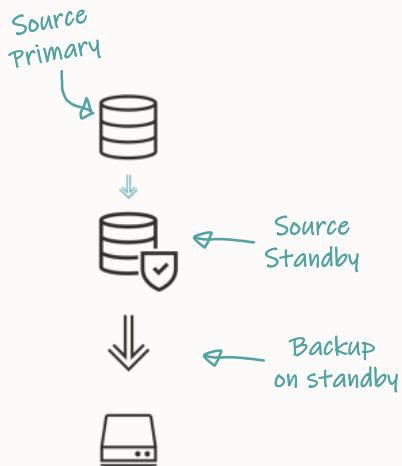
Cross Platform Database Migration using ZDLRA (Doc ID [2460552.1](#))



Taking backups on standby database

- Offload primary database

Backup on Standby



Backup on Standby

Backup for test

- Start M5 driver script on standby host
- Connect to standby database
- Set **BKP_FROM_STDBY=1** in M5 properties file

Backup for migration

- For level 0 (**L0**) and level 1 (**L1**):
 - Start M5 driver script on standby host
 - Connect to standby database
- For final level 1 (**L1F**):
 - Start M5 driver script on primary host
 - Connect to primary database

Backup from Standby

Effect of setting BKP_FROM_STDBY=1 in M5 properties file

Affects only final level 1 (L1F):

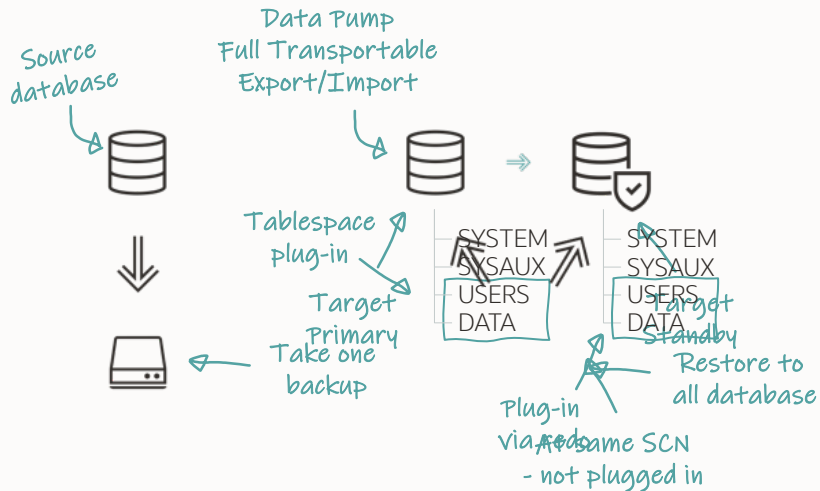
- Convert to snapshot standby
- Snapshot standby resets BCT bitmap requiring final incremental to fully scan the database
- Recommended for testing only



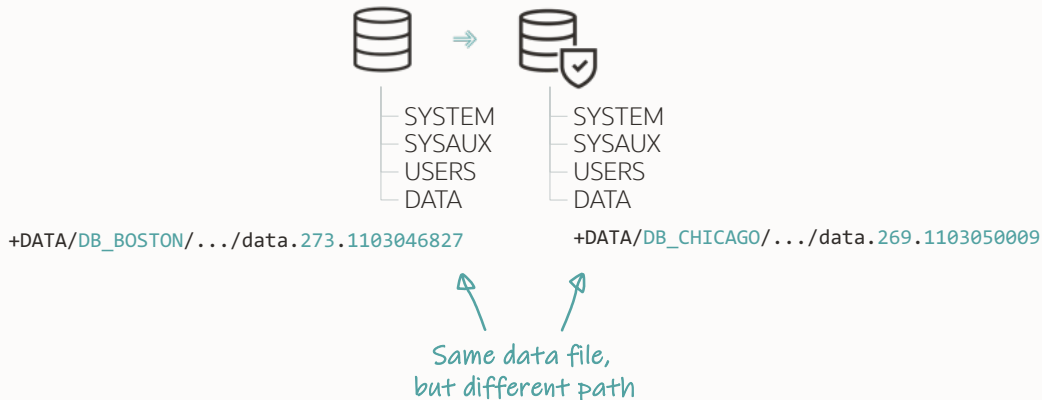
Build target standby database as part of the migration

- Protect the target database immediately

Target Standby Database



Target Standby Database





Tablespace plug-in propagates to standby database via redo

- Expects data files with identical path / name on standby

Target Standby Database

Redo contains data file path from primary,
but it's different on standby

- 1 Use ASM aliases
- 2 Rename standby data files after plug-in failure



Using Advanced Queueing

- AQ creates some queue structures on demand only
- [Blog post](#)

Advanced Queueing

Source database

`<queue_table_name>`
`AQ$_<queue_table_name>_E`
`AQ$_<queue_table_name>_I`
`AQ$_<queue_table_name>_T`
`AQ$_<queue_table_name>_F`
`AQ$_<queue_table_name>_C`
`AQ$_<queue_table_name>_D`
`AQ$_<queue_table_name>_G`
`AQ$_<queue_table_name>_H`
`AQ$_<queue_table_name>_L`
`AQ$_<queue_table_name>_P`
`AQ$_<queue_table_name>_S`
`AQ$_<queue_table_name>_V`

Queue table

Queue
infrastructure

Target database

`<queue_table_name>`
`AQ$_<queue_table_name>_E`
`AQ$_<queue_table_name>_I`
`AQ$_<queue_table_name>_T`
`AQ$_<queue_table_name>_F`



Take into account in comparing source and target database object count

- Understanding How Advanced Queueing (AQ) Objects Are Exported And Imported. (Doc ID [2291530.1](#))



Data Pump does not start queues

- Manually start queues after migration
- Use `DBMS_AQADM.START_QUEUE`

Further Reading

Understanding How Advanced Queueing (AQ) Objects Are Exported And Imported. (Doc ID [2291530.1](#))

What Objects Are Created When Creating a Queue Table ? (Doc ID [224027.1](#))

[Things to Consider When Importing Advanced Queues using Oracle Data Pump](#)



Data Pump moves audit policies and records

- During Full Transportable Export/Import
- Traditional and unified auditing



Archive audit records in advance
using DBMS_AUDIT_MGMT

- Optionally, export audit records to target database



Configure a separate audit tablespace in target database

- Migrate audit records directly into dedicated tablespace
- Don't use *SYSAUX* for audit records

Audit Trail

Target database already has auditing infrastructure

- Schema, table, potentially also dedicated tablespace
- Not possible to move audit tablespace via transportable tablespace

Data Pump always unloads the audit records into the dump file

If you have a huge audit trail, **empty it!** Or ...

- You get a huge dump file
- You extend the maintenance window



Data Pump does not move AWR data

AWR

Consider whether it makes sense to move AWR data

- New and more powerful hardware
- New platform and architecture
- Potentially also newer version and into multitenant

To export AWR data

- How to Export and Import the AWR Repository From One Database to Another (Doc ID [785730.1](#))
- Do it outside of the maintenance window



But we are on Windows

Windows



M5 script is written for Bash shell

- Use Windows Terminal
- Or another Bash shell for Windows

↑
Untested



Run commands manually

- Data Pump
- RMAN
 BACKUP TABLESPACE
 RESTORE ALL FOREIGN DATAFILES



In the unlikely event of ...

- Rollback and fallback options

Rollback and Fallback

- To roll back (before go-live):
Source environment is preserved
- To fall back (after go-live):
Reverse process back to source
- If you also migrated to a PDB:
Reverse process back to source
- If you also upgraded:
Data Pump and GoldenGate



Edge Cases



Using binary XML


```
--Create a new table using XMLTYPE  
CREATE TABLE CARS (CARDATA XMLTYPE);
```

```
--XMLTYPE columns consists of two columns
```

```
SELECT COLUMN_NAME, DATA_TYPE, HIDDEN_COLUMN, VIRTUAL_COLUMN FROM USER_TAB_COLS;
```

COLUMN_NAME	DATA_TYPE	HIDDEN_COLUMN	VIRTUAL_COLUMN
CARDATA	XMLTYPE	NO	YES
SYS_NC000002\$	BLOB	YES	NO

Encoded/compressed
XML data

XMLTYPE
is a virtual column

--Detecting Binary XML in your Oracle Database

```
select *  
from   dba_xml_tab_cols  
where  storage_type= 'BINARY'  
       and owner != 'SYS';
```


Binary XML

```
<CARS>
  <CAR>
    <MODEL>
      Volvo V90
    </MODEL>
  </CAR>
</CARS>
```

XML document

INSERT INTO CARS ...



Compress and encode XML to binary format

1. Generate tokens from namespace and tags
2. Insert into central token table
3. Insert encoded XML into CARS


```
INSERT INTO CARS VALUES('<CARS><CAR><MODEL>Volvo V90</MODEL></CAR></CARS>');
```

```
SELECT CARDATA, SYS_NC00002$ FROM CARS;
```

CARDATA	SYS_NC00002\$
<CARS>	9F01639E000000C850B4C81F1FC0085D90566F6C766F20563930D9D9A0
<CAR>	
<MODEL>Volvo V90</MODEL>	
</CAR>	
</CARS>	

SYS_NC000002\$

9F01639E000000C850B4C81F1F00085D90566F6C766F20563930D9D9A0

select * from XDB.X\$QN6LVUTKVD49541E0L8000000001;

Token repository

NMSPCID	LOCALNAME	FLAGS	ID
...
07	CARS	00	50B4
07	CAR	00	1F1F
07	MODEL	00	5D90

```
<CARS>
  <CAR>
    <MODEL>
      Volvo V90
    </MODEL>
  </CAR>
</CARS>
```


Binary XML

- The table contains the encoded binary XML
- The dictionary contains the token repository
- Both are required to read and understand the data



Data Pump moves tokens during
Full Transportable Export/Import

--If a token is already in use in the target database
--Data Pump skips the table to avoid data corruption

impdp ... transport_datafiles=<*list-of-files*>

ORA-39945: Token conflicting with existing tokens



Binary XML

Possible solutions

1 Conventional Data Pump export

- When Data Pump inserts the binary XML, the target database generates a new token

2 Prune the target database tokens

- Re-use all source database tokens in the target database
- Works on brand new, empty target databases only
- Be aware of bug 34425044
- Strongly recommended patching the source and target database using the latest available Release Update and Data Pump Bundle Patch
- [How to Migrate Large Amount of Binary XML Data between Databases \(Doc ID 2309649.1\)](#)



Oracle Database 23c uses a new, transportable format for binary XML

- Used for new tables
- Consider converting old tables

Transportable Binary XML

Modern, scalable and transportable [format](#)

Convert using:

- Create Table As Select
- DBMS_REDEFINITIONS
- Data Pump (TRANSFORM=XMLTYPE_STORAGE_CLAUSE:TRANSPORTABLE BINARY XML)

Binary XML

Binary XML uses tokens to compress/encode XML Data. The token ids and their value (Tag name) are stored in a central token table. During TTS import of data the tokens need to be reused, that means tokens on the exporting side and importing side cannot conflict with each other. In case of a conflict not XL data can be imported and an error message is raised during TTS import.

The Export and Import utilities can be used to move Binary XML storage data between environments. This option works well on smaller datasets, however, since it involves several internal insert commands, it will be very resource intensive on larger data migrations. For customers with a limited timescale planned migration window, this is not a feasible option.

How to move XMLType tables/columns with Binary XML Storage between schemas/databases (Doc ID 1405457.1)

Binary XML

How to find XML token table:

```
select TOKSUF from xdb.xdb$ttset;
```

Add as suffix:

--Tags

```
select * from XDB.X$QN<toksuf>;
```

--Namespaces

```
select * from XDB.X$NM<toksuf>;
```

Staging tables during FTEX:

```
xdb.xdb$tsetmap
```

```
xdb.xdb$ttset
```




Using evolved types in table definitions

--Create a new type. The type is now version 1

--Use the type in a table

```
CREATE TYPE CAR_INFO_TYPE IS OBJECT (model VARCHAR2(40));
```

```
CREATE TABLE CARS (id number, car_info car_info_type);
```

```
INSERT INTO CARS VALUES (1, car_info_type('Volvo V90'));
```

The type is now evolving



--Make a change to the type. The type is now version 2

```
ALTER TYPE CAR_INFO_TYPE ADD ATTRIBUTE horsepower NUMBER CASCADE NOT INCLUDING TABLE DATA;
```

```
INSERT INTO CARS VALUES (2, car_info_type('BMW 530i', 250));
```

Existing data is not updated



--Make another change to the type. The type is now version 3

```
ALTER TYPE CAR_INFO_TYPE ADD ATTRIBUTE color VARCHAR2(20) CASCADE NOT INCLUDING TABLE DATA;
```

```
INSERT INTO CARS VALUES (3, car_info_type('Hyundai Sonata', 160, 'Black'));
```


Evolved Types

```
SELECT * FROM CARS
```



CARS	
1	car_info_type v1: Volvo V90
2	car_info_type v2: BMW 530i, 250
3	car_info_type v3: Hyundai Sonata, 160, Black



DICTIONARY	
car_info_type v1	model
car_info_type v2	model, horsepower
car_info_type v3	model, horsepower, color



Data Pump recreates types during
Full Transportable Export/Import

Evolved Types

- To avoid **data corruption**,
Data Pump must recreate the exact same type evolution in target database
- Due to **implementation restrictions**,
it is not always possible to recreate the exact same type evolution
- In such situations, to avoid corruption,
Data Pump reports ORA-39218 or ORA-39216 on **import**

--Identifying tables with evolved types
--These tables potentially pose a problem during transportable import

```
select owner, table_name, column_name, data_type_owner, data_type
from   dba_tab_cols
where  (data_type_owner, data_type) IN (
        select distinct u.username, o.name
        from   obj$ o, dba_users u, type$ t
        where  o.owner# = u.user_id
              and oracle_maintained='N'
              and o.oid$ = t.toid
              and t.version# > 1
        group by u.username, o.name);
```

Find tables using these types

Find types with more than one version



Evolved Types | Possible Solutions

- 1 Conventional Data Pump export
- 2 Manually recreate `type` in target database with matching evolution
- 3 Recreate `type` without evolution before export

[Blog post](#) with details



Migrating databases with Oracle E-Business Suite



M5 script has not yet been certified
by E-Business Suite

- XTTS v4 remains the certified approach



Customer Case



Customer

Project

Result

Learnings

- **Advania** - Icelandic tech company
- Operating across Sweden, Norway, UK, Iceland, Finland and Denmark.
- Offering Managed services, Hardware and Software, and Professional Services to clients in both private and public sectors

Customer Case

Customer

Project

Result

Learnings



Oracle SPARC SuperCluster M7

Oracle E-Business Suite 12.2.11

Database size: 13 TB

19.18.0 + Data Pump Bundle Patch



Oracle Exadata Cloud@Customer

19.20.0 + Data Pump Bundle Patch



Customer Case

Customer

Project

Result

Learnings

- Planned 48-hours maintenance window
 - Includes encryption of tablespaces
- Database migration ~16 hours
 - Export: 4h 35m
 - Import: 11h 19m



Customer Case

Customer

Project

Result

Learnings

- 1 Test, test, and test
- 2 Create your own runbook
- 3 Use AWR to identify bottlenecks
- 4 Clone target database before import for easy testing and re-runs

Further Reading | E-Business Suite

MOS notes:

- Using Transportable Tablespaces to Migrate Oracle E-Business Suite Release 12.2 Using Oracle Database 19c Enterprise Edition On a Multitenant Environment (Doc ID [2674405.1](#))
- Oracle E-Business Suite 12.2 Platform Migration from On-Premises to Oracle Cloud Using Transportable Tablespaces with Oracle Database 19c (Doc ID [2725558.1](#))

Blog posts:

- [Collection of EBS upgrade information for Oracle Database 19c](#)



Testing the Migration



FTEX Testing Options

Test the transportable export

- How long will the export take?
- expdp parameter `TTS_CLOSURE_CHECK=TEST_MODE`

Repeat your tests without the need to restore again

- Tablespaces will be taken online by default since Oracle 12c
- impdp parameter option `TRANSPORTABLE=KEEP_READ_ONLY` prevents this



You can **test** the Data Pump export without setting tablespaces in read-only mode

- Performs a metadata export for a full transportable export/import
- Dump file is unusable for import.
- Parameter is available since Oracle Database 19c
- Logfile says: *Dump file set is unusable. TEST_MODE requested.*

expdp ... full=y transportable=always **tts_closure_check=test_mode**

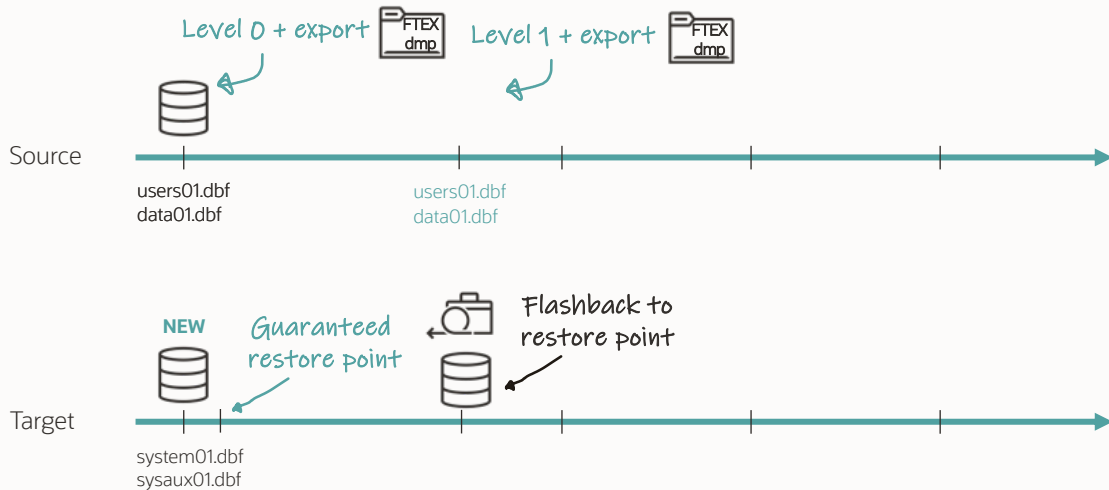


Repeat import tests by
keeping tablespaces in read-only mode

- Performs a real FTEX but keeps the tablespace read-write
- This ensures the data files may be re-used for a different import
- Parameter is available since Oracle Database 19c

```
impdp ... transportable=keep_read_only
```


impdp ... transportable=keep_read_only



Using a Snapshot Standby

For FTEX, tablespaces must be switched to read-only

- This isn't usually an option on PROD

Use a physical standby database for FTEX testing

- Take L0 backup from the standby
- Take L1 backup from the standby
- Convert it into a snapshot standby
- Switch tablespaces read-only
- Run the Full Transportable Export
- Take the "final" L1 backup
- Convert it back to a physical standby

Repeat!

Testing Without Data

1. Perform a regular Data Pump metadata export
 - Use `content=metadata_only`
2. Import into a test database
 - All metadata is there, but no data
3. Test M5 migration procedure
 - Learn and test the procedure
 - Easier due to very small database



Migrating the Beast

Customer Case

Entain is one of the world's largest sports betting and gaming groups. Leveraging the power of the Entain Platform, they bring moments of excitement into their customers lives through more than 30 iconic brands such as bwin, Coral, Ladbrokes and many more.

Entain operates on over 140 licenses across 40+ territories and employs over 29,000 talented workforce. Entain is listed on the London Stock Exchange and is a constituent of the FTSE 100 Index.

The Entain logo, featuring a stylized Greek letter sigma (Σ) followed by the word "Entain" in a bold, sans-serif font. The background of the entire slide is a blurred image of a soccer match in progress, with a player's leg and a soccer ball visible. A white geometric pattern of overlapping hexagons and diamonds is overlaid on the right side of the image.

Entain



Challenges and Constraints

What is special, what makes it so complex?

Migration Challenges



SPARC SuperCluster



ZDLRA



Exadata X9M Extreme Flash

Migration Challenges

180TB
size



SPARC SuperCluster



ZDLRA



Exadata X9M Extreme Flash

Migration Challenges

15TB
redo/day



SPARC SuperCluster



ZDLRA



Exadata X9M Extreme Flash

Migration Challenges



SPARC SuperCluster



Exadata X9M Extreme Flash



5 Physical Standby DBs

Local, and in different region, 2500km away



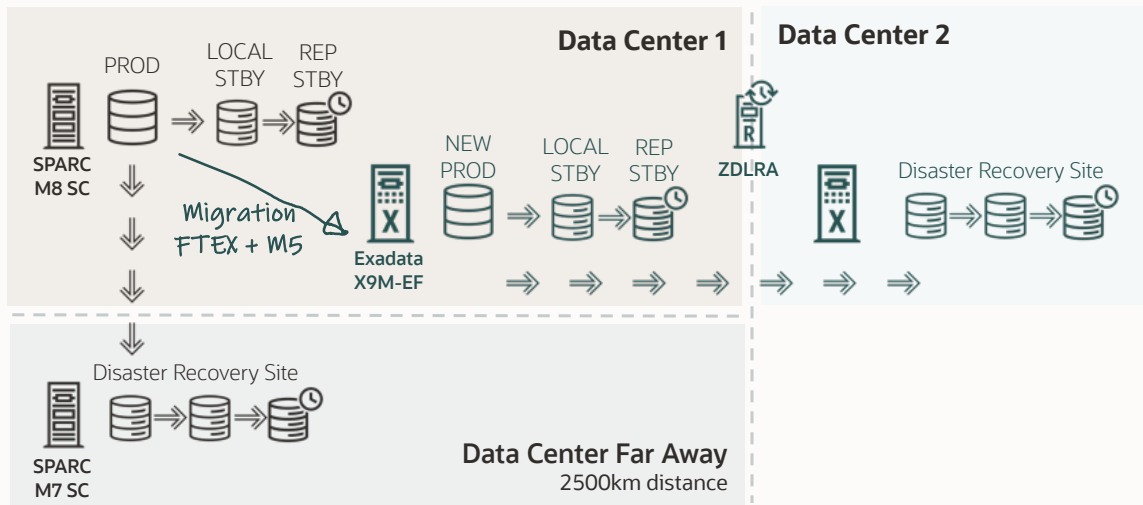
Migration



Full Transportable Export/Import is
the only possible option to migrate

- Oracle GoldenGate unlikely to synch 15TB redo/day
- Data Pump would have taken too long

Migration Plan



Migration Workflow

Configure

Level 0

Level 1

Outage

Final
Backup

Final
restore

Export
Import



SPARC SuperCluster

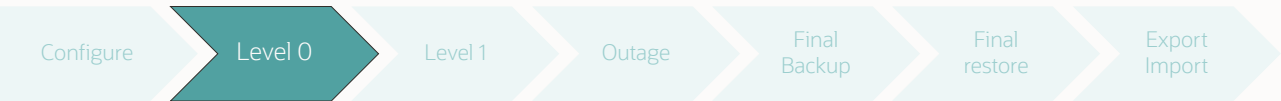


ZDLRA



Exadata X9M Extreme Flash

Migration Workflow



SPARC SuperCluster



Exadata X9M Extreme Flash

Migration Workflow

Configure

Level 0

Level 1

Outage

Final
Backup

Final
restore

Export
Import



SPARC SuperCluster



ZDLRA



Exadata X9M Extreme Flash

Migration Workflow

Configure

Level 0

Level 1

Outage

Final
Backup

Final
restore

Export
Import



SPARC SuperCluster



Exadata X9M Extreme Flash

Migration Workflow

Configure

Level 0

Level 1

Outage

Final
Backup

Final
restore

Export
Import



SPARC SuperCluster



ZDLRA



Exadata X9M Extreme Flash

Migration Workflow

Configure

Level 0

Level 1

Outage

Final
Backup

Final
restore

Export
Import



SPARC SuperCluster



ZDLRA



Exadata X9M Extreme Flash

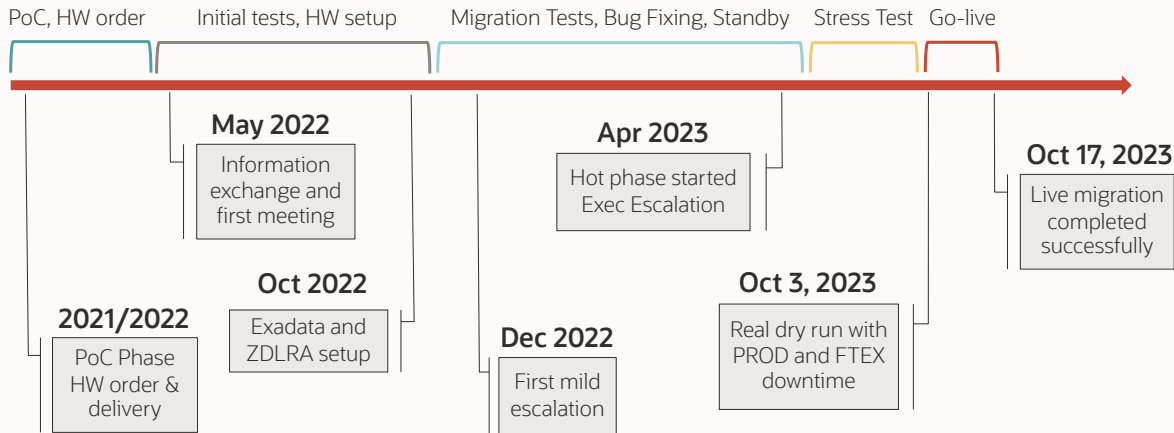




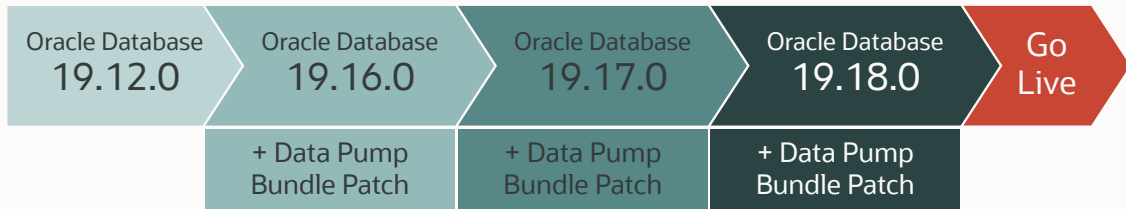
Success Factors

Timelines and the Run Book

Overall Project Timeline



Oracle Database Release Migration Evolution



Key to Success: Runbook

Complex projects absolutely require a **detailed runbook**

ID	Task	Status	Responsible Primary Person	Responsible Secondary Person	Predecessor	Start Time (CEST)	Duration (hh:mm)	End Time (CEST)	Start Time (IST)	End Time (IST)	Actual Start Time (CEST)	Actual Duration	Actual End Time (CEST)	Comments - Blocker
----	------	--------	----------------------------------	------------------------------------	-------------	----------------------	---------------------	--------------------	------------------------	-------------------	--------------------------------	--------------------	------------------------------	-----------------------

- This run book covered over 200 individual tasks

A screenshot of a complex project runbook table. The table has many columns, including task ID, task name, status, responsible primary and secondary persons, predecessor tasks, start and end times in CEST and IST, duration, actual start and end times, and comments. The rows are color-coded in green, blue, and yellow, likely representing different task statuses or categories. The table is densely packed with text and numbers, showing a high level of detail for project management.

Timeline Live Migration

Task	22:00	23:00	0:00	1:00	2:00	3:00	4:00	5:00	6:00	7:00	8:00
Stats and AWR expdp etc											
Application downtime		22:45 h – 8:00 h									
Export / truncate audit		48 min									
Tablespaces read-only											
Final L1 backup			50 min								
Full Transportable Export			52 min								
Recover final L1 backup				30 min							
Full Transportable Import				165 min = 2:45 hrs							
Gather and import stats							30+30 min				
Database migration time			23:48 h – 5:05 h								
Functional/internal tests								175 min = 2:55 hrs			



Monitoring and Troubleshooting



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
```


Monitoring | Other Sources

- Tail alert log
- Use V\$SESSION_LONGOPS
- Use SQL Monitoring
 - Enterprise Manager SQL Monitoring
 - SQL Developer Real Time SQL Monitoring
 - Check license requirements



M5 scripts adds
RMAN trace automatically



To enable Data Pump trace use
DP_TRACE in M5 properties files

- Trace level 3FF0300 suitable for transportable 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)
-- Trace value 3FF0300 is suitable for transportable jobs
-- LOGTIME and METRICS is added by default by M5 script
expdp ... metrics=yes logtime=all trace=3FF0300
impdp ... metrics=yes logtime=all trace=3FF0300

-- 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)
-- Trace value 3FF0300 is suitable for transportable jobs
-- LOGTIME and METRICS is added by default by M5 script
expdp ... metrics=yes logtime=all trace=3FF0300
impdp ... metrics=yes logtime=all trace=3FF0300

-- 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)
-- Trace value 3FF0300 is suitable for transportable jobs
-- LOGTIME and METRICS is added by default by M5 script
expdp ... metrics=yes logtime=all trace=3FF0300
impdp ... metrics=yes logtime=all trace=3FF0300

-- 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)
-- Trace value 3FF0300 is suitable for transportable jobs
-- LOGTIME and METRICS is added by default by M5 script
expdp ... metrics=yes logtime=all trace=3FF0300
impdp ... metrics=yes logtime=all trace=3FF0300

-- 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***

Migration



*A migration is **not** a pet project*

- Plan
- Train
- Focus
- Work together

*A migration is not successful unless
the database **performs acceptably** after going live*

Use our Performance Stability Prescription

Performance Stability Webinar

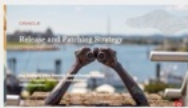
- Watch the [recording](#)
- Get the [slides](#)



Episode 1

Release and Patching Strategy

105 minutes – Feb 4, 2021



Episode 2

AutoUpgrade to Oracle Database 19c

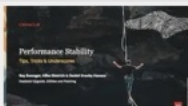
115 minutes – Feb 20, 2021



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

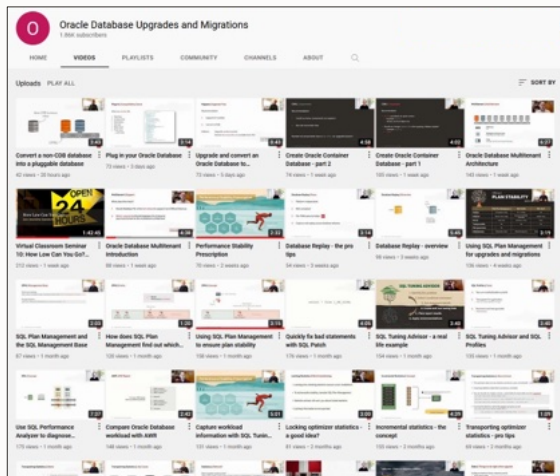
<https://MikeDietrichDE.com/videos>

More than 30 hours of technical content:

- All tech, no marketing
- On-demand
- Anytime
- Anywhere



YouTube | @UpgradeNow



[Link](#)

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



Thank You

