

What's New in 12c for Backup & Recovery?



By:

Francisco Munoz Alvarez 

About the Speaker



Francisco Munoz Alvarez



Oracle ACE Director

CLOUG (Chilean Oracle Users Group) President

LAOUC (Latin American Oracle Users Group Council) President

NZOUG (New Zealand Oracle Users Group) President

8/9/10g/11g OCP, RAC OCE, AS OCA, E-Business OCP, SQL/PLSQL OCA, Oracle 7 OCM

Oracle 7, 11GR2, 12cR1 Beta Tester

OVM 3 Beta Tester

ITIL Certified

2010 ACE Director of the year by Oracle Magazine

Blog: www.oraclenz.com - Email: mbatec@Hotmail.com - Twitter : [fcomunoz](#)

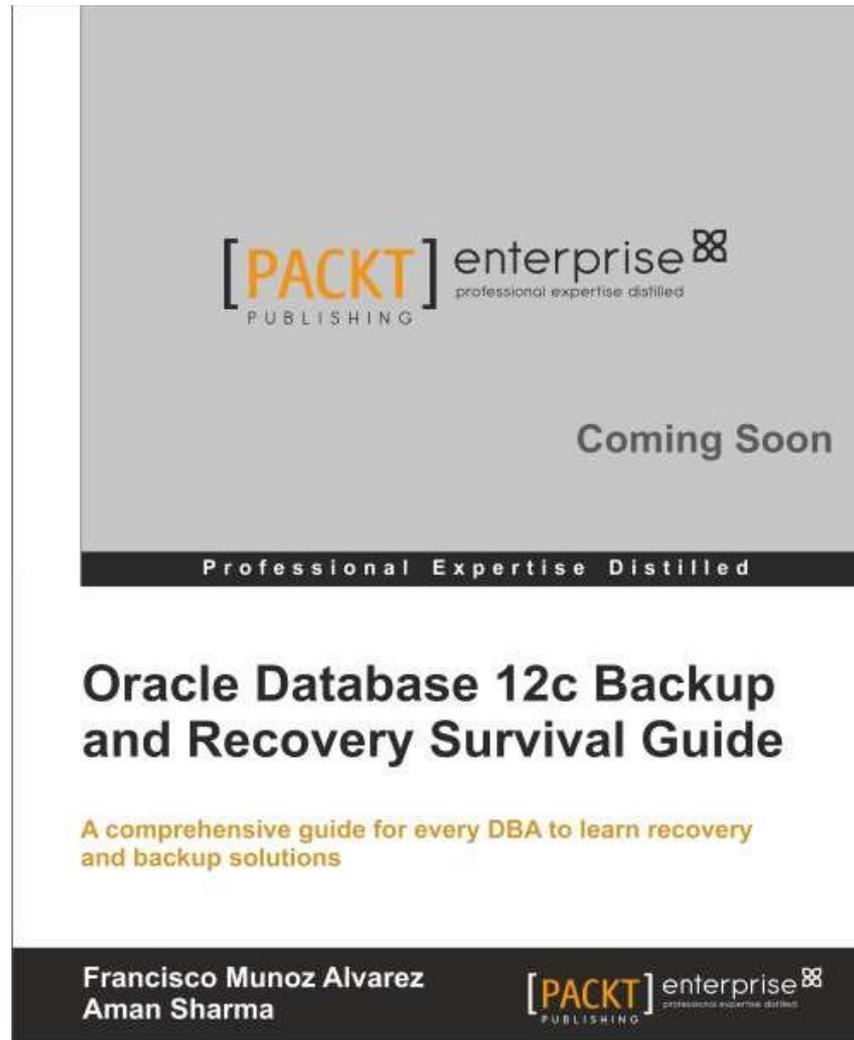
Revera Limited

Oracle Professional Services Manager

www.revera.co.nz

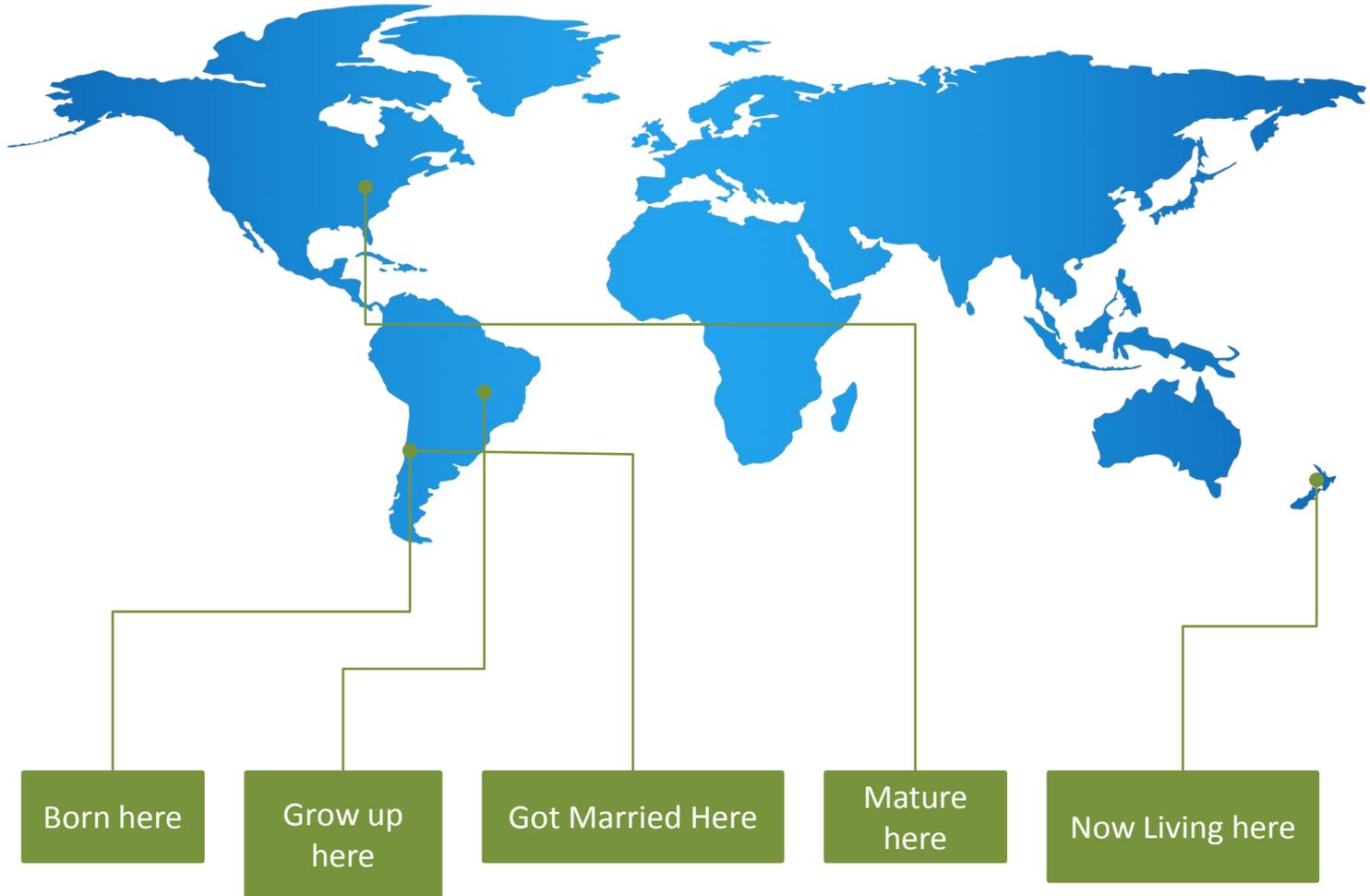
Some Conferences





Coming Soon: September 2013

OK, are you from?



AGENDA

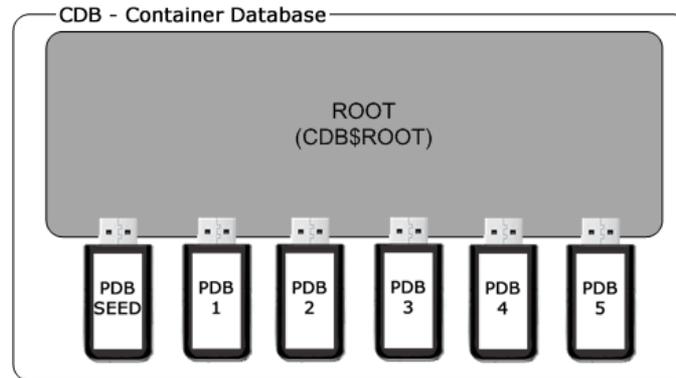
- **Multitenant Container**
- **RMAN New Features and Enhancements**
- **Data Pump New Features and Enhancements**
- **Enterprise Manager Express**
 - **Oracle Flashback**

Multitenant Container

What is New (Concepts)

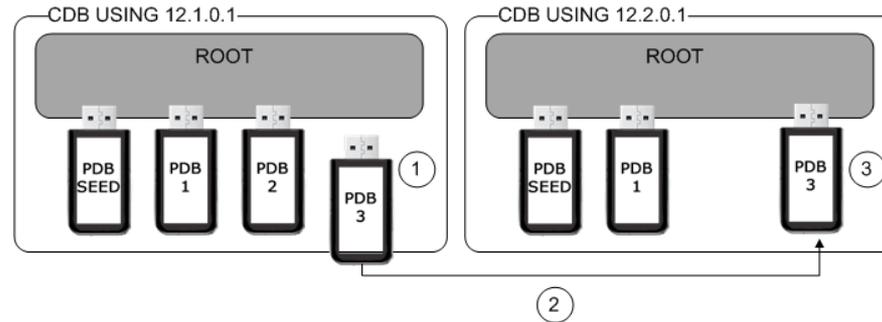
- **Single Tenant Container:** 12c Database with only one Pluggable Database.
- **Multitenant Container:** 12c Database with more than one Pluggable Database.
- **Non-CDB:** Database without the use of a container. A database using the same concept as previous 12c release (without use of Pluggable Databases).

Multitenant Container



We are now able to have multiple databases sharing a single instance and Oracle binaries, and each of the databases will have their own configuration and parameters and what is better, each database will be completely isolated of each other without either know that each other exists.

Multitenant Container



Many years ago, the introduction of external storages gave us the possibility to store data on external devices and the flexibility to plug and unplug them to any system independent of their OS (Operating System). As per example, you can connect an external device to a system using Windows XP and read your data without any problems, later you can unplug it and connect it to a laptop running Windows 7 and you will still be able to read your data. Now with the introduction of Oracle Pluggable Databases, we will be able to do something similar with Oracle when upgrading a PDB, making this process simple and easy. All you will need to do to upgrade a PDB as per example is to unplug your PDB (step 1) that is using a CDB running 12.1.0.1, copy the PDB to the destination location with a CDB that is using a later version like 12.2.0.1 (step 2), and plug the PDB to the CDB (step 3), and your PDB is now upgraded to 12.2.0.1.

Multitenant Container

Many years ago, the introduction of external storages gave us the possibility to store data on external devices and the flexibility to plug and unplug them to any system independent of their OS (Operating System). As per example, you can connect an external device to a system using Windows XP and read your data without any problems, later you can unplug it and connect it to a laptop running Windows 7 and you will still be able to read your data. Now with the introduction of Oracle Pluggable Databases, we will be able to do something similar with Oracle when upgrading a PDB, making this process simple and easy. All you will need to do to upgrade a PDB as per example is to unplug your PDB (step 1) that is using a CDB running 12.1.0.1, copy the PDB to the destination location with a CDB that is using a later version like 12.2.0.1 (step 2), and plug the PDB to the CDB (step 3), and your PDB is now upgraded to 12.2.0.1.

Multitenant Container

Some key points about Pluggable Databases are:

- You can have many PDBs you want inside a single container (A CDB can contain a maximum of 253 PDBs);
- A PDB is fully backwards compatible with an ordinary pre 12.1 database;
- A system administrator can connect to a CDB as a whole and see a single system image;
- If you are not ready to make use of this new concept, you can still be able to create a database on 12c as before. It is called, non-CDB (non-container database);
- Each instance in RAC opens the CDB as a whole. A foreground session see only the single PDB it is connected to and see it just like a non-CDB database;
- The Resource Manager is extended with some new between PDB capabilities;
- Fully integrated with Oracle Enterprise Manager 12c and SQL Developer;
- Fast provisioning of new databases (empty or as a copy/clone of an existing PDB);
- On Clone triggers can be used to scrub or mask data during a clone process;
- Fast unplug and plug between CDBs;
- Fast path or upgrade by unplugging a PDB and plugging it into a different CDB already patched or with a later database version;
- Separation of duties between DBA and application administrators;
- Communication between PDBs is allowed via intra-CDB dblinks;
- Every PDB has a default service with its name (Listener);
- An unplugged PDB carries its lineage, opatch, encryption key info, and much more;
- All PDBs in a CDB should use the same character set;
- All PDBs shares the same control files, spfile, redo log files, flashback log files and UNDO;
- Flashback PDB is not available on 12.1, it expected to be available with 12.2;
- Allows multitenancy of Oracle Databases, very useful for centralization, especially if using Exadata.

RMAN New Features and Enhancements

Container and Pluggable Database Backup & Restore

As we saw before, the introduction of 12c and the new Pluggable Database concept made possible to easily centralize multiple databases maintaining the individuality of each one when using a single instance. The introduction of this new concept also forced Oracle to introduce some new enhancements to the already existent BACKUP, RESTORE and RECOVERY commands, to enable us to be able to make an efficient backup or restore of the complete CDB including all PDBs, or just one of more PDBs or if you want to be more specific, you can also just backup or restore one or more tablespace from a PDB.

Container and Pluggable Database Backup & Restore

Some examples of how to use RMAN commands when performing a backup on 12c are:

```
RMAN> BACKUP DATABASE; (To backup the CBD + all PDBs)
RMAN> BACKUP PLUGGABLE DATABASE pdb1,pdb2; (To backup all specified PDBs)
RMAN> BACKUP TABLESPACE pdb1:example; (To backup a specific tablespace in a PDB)
```

Some examples when performing RESTORE operations are:

```
RMAN> RESTORE DATABASE; (To restore an entire CDB, including all PDBs)
RMAN> RESTORE DATABASE root; (To restore only the root container)
RMAN> RESTORE PLUGGABLE DATABASE pdb1; (To restore a specific PDB)
RMAN> RESTORE TABLESPACE pdb1:example; (To restore a tablespace in a PDB)
```

Finally some example of RECOVERY operations:

```
RMAN> RECOVER DATABASE; (Root plus all PDBs)
RMAN> RUN {
    SET UNTIL SCN 1428;
    RESTORE DATABASE;
    RECOVER DATABASE;
    ALTER DATABASE OPEN RESETLOGS; }
RMAN> RUN {
    RESTORE PLUGGABLE DATABASE pdb1 TO RESTORE POINT one;
    RECOVER PLUGGABLE DATABASE pdb1 TO RESTORE POINT one;
    ALTER PLUGGABLE DATABASE pdb1 OPEN RESETLOGS; }
```

Backup Privileges

Oracle database 12c provides separation support for the separation of DBA duties for the Oracle database, by introducing task specific and least privileged administrative privilege for backup that do not requires the SYSDBA privilege. The new system privilege introduced with this new release is SYSBACKUP.

Note: Avoid the use of the SYSDBA privilege for backups, unless it is strictly

When connecting to the database using the AS SYSDBA system privilege, you are able to see any object structure and all data within the object, but if connecting using the new system privilege AS SYSBACKUP, you will still be able to see the structure of an object but not the object data. If you try to see any data using the SYSBACKUP privilege, you will be raised the ORA-01031: insufficient privileges message.

Tighter security policies requires a separation of duty, and the new SYSBACKUP privilege facilitate the implementation of the separation of duties, allowing backup and recovery operations to be perform without implicit access to data, so if the access to data is required to one specific user, it will need to be granted explicitly to this user.

Backup Privileges

RMAN had introduced some changes when connecting to a database as:

- **TARGET:** It will require the user to have the `SYSBACKUP` administrative privilege to be able to connect to the `TARGET` database;
- **CATALOG:** As in earlier versions a user was required to have the `RECOVERY_CATALOG_OWNER` role assigned to be able to connect to the RMAN catalog, now it will need to have assigned the `SYSBACKUP` privilege to be able to connect to the catalog.;
- **AUXILIARY:** It will require the `SYSBACKUP` administrative privilege to connect to the `AUXILIARY` database.

Some important point about the `SYSBACKUP` administrative privilege are:

- Includes permissions for backup and recovery operations;
- Do not include data access privileges such as `SELECT ANY TABLE` that the `SYSDBA` privilege has;
- Can be granted to the `SYSBACKUP` user that is created during the database installation process;

It's the `DEFAULT` when a RMAN connection string is issued and do not contain the "`AS SYSBACKUP`" clause);

```
$ RMAN TARGET /
```

Backup Privileges

Note: Before connect as the SYSBACKUP user created during the database creation process, you will need to unlock the account and grant the SYSBACKUP privilege to it. When you use the GRANT command to give the SYSBACKUP privilege to a user, the user name and privilege information will be automatically added to the database password file.

The V\$PWFILERS_USERS view contains all information regarding users within the database password file and indicates whether a user has been granted any privileged system privilege. Let's take a closer look to this view:

```
SQL> DESC v$pwfile_users
```

| Name | Null? | Type |
|-----------|-------|---------------|
| ----- | ----- | ----- |
| USERNAME | | VARCHAR2 (30) |
| SYSDBA | | VARCHAR2 (5) |
| SYSOPER | | VARCHAR2 (5) |
| SYSASM | | VARCHAR2 (5) |
| SYSBACKUP | | VARCHAR2 (5) |
| SYSDG | | VARCHAR2 (5) |
| SYSKM | | VARCHAR2 (5) |
| CON_ID | | NUMBER |

Backup Privileges

As you can see, this view now contains some new columns, such as:

- **SYSBACKUP**: Indicates if the user is able to connect using the SYSBACKUP privileges;
- **SYSDG**: Indicates if the user is able to connect using the SYSDG (new for Data Guard) privileges;
- **SYSKM**: Indicates if the user is able to connect using the SYSKM (new for Advance Security) privileges;
- **CON_ID**: ID of the current container. If 0 it will indicate that it related to the entire container database (CDB) or to an entire traditional database (non-CDB), if the value is 1, then this user have the access only to root, if other value will identify a specific container ID.

SQL and DESCRIBE

As you well know, you are able to execute SQL commands and PL/SQL procedures from the RMAN command line, starting with 12.1 Oracle do not require the use the SQL prefix or quotes for most SQL commands in RMAN.

You can now run some simple SQL commands at RMAN such as:

```
RMAN> SELECT TO_CHAR(sysdate, 'dd/mm/yy - hh24:mi:ss')
```

```
2> FROM dual;
```

```
TO_CHAR(SYSDATE, 'DD
```

```
-----
```

```
17/09/12 - 02:58:40
```

SQL and DESCRIBE

```
RMAN> DESC v$datafile
```

| Name | Null? | Type |
|-----------------------|-------|---------------|
| FILE# | | NUMBER |
| CREATION_CHANGE# | | NUMBER |
| CREATION_TIME | | DATE |
| TS# | | NUMBER |
| RFILE# | | NUMBER |
| STATUS | | VARCHAR2 (7) |
| ENABLED | | VARCHAR2 (10) |
| CHECKPOINT_CHANGE# | | NUMBER |
| CHECKPOINT_TIME | | DATE |
| UNRECOVERABLE_CHANGE# | | NUMBER |
| UNRECOVERABLE_TIME | | DATE |
| LAST_CHANGE# | | NUMBER |
| LAST_TIME | | DATE |
| OFFLINE_CHANGE# | | NUMBER |
| ONLINE_CHANGE# | | NUMBER |
| ONLINE_TIME | | DATE |
| BYTES | | NUMBER |
| BLOCKS | | NUMBER |
| CON_ID | | NUMBER |

```
RMAN> ALTER TABLESPACE users
```

```
2> ADD DATAFILE '/u01/app/oracle/oradata/cdb1/pdb1/user02.dbf' size 50M;
```

Note: Remember that the SYSBACKUP privilege does not grant access to user tables or views, but the SYSDBA privilege does.

Multisection Backups for Incremental

Oracle Database 11g introduced multisection backups to allow us to backup and restore very large files using backup sets (Remember that Oracle datafiles can have up to 128 TB in size), now with Oracle database 12c we are able to make use of image copies when creating multisection backups as a complement of the previous backup set functionality.

This help us to reduce image copy creation time (for backups, transporting tablespaces, cloning and doing a TSPITR (Tablespace Point in Time Recovery), it also improve backups when using Exadata.

Multisection Backups for Incremental

The main restrictions to make use of this enhancement are:

- The COMPATIBLE initialization parameter needs to be set to 12.0 or higher to make use of the new image copy multisection backup feature;
- This is only available for datafiles, and cannot be used to backup control or password files;
- Not to be used with a large number of parallelism, when a file resides on a small number of disks, to avoid each process to compete with each other when accessing the same device.

Another new feature introduced with regarding multisection backups, is the ability to create multisection backups for incremental backups. This will allow the RMAN to only backup the data that has changed since the last backup, consequently enhancing the performance of multisection backups due that they are processed independently, either serially or in parallel.

Active Duplicate

The Active Duplicate feature generates an online backup on the TARGET database and directly transmit it via an inter-instance network connection to the AUXILIARY database for duplication (not written to disk in the source server), consequently reducing impact on the TARGET database by off-loading the data transfer operation to the AUXILIARY database, also reducing duplication time.

This very useful feature has now received some important enhancements, on 11g when this feature was initially introduced, it only allowed us to use a push process based on image copies, now it allow us to make use of the already known push process, or to make use of the newly introduced pull process from the AUXILIARY database that is based on backup sets (that is now the new DEFAULT and automatically copy across all datafiles, control files, SPFILE and archive log files), and then performs the restore of all files and uses a memory script to complete the recovery operation and open the AUXILIARY database. RMAN will dynamically determine based on your DUPLICATE clauses which process will be used (push or pull).

Active Duplicate

You can now choose your choice of compression, section size, and encryption to be used during the Active Duplication process. As per example, if you specify the SET ENCRYPTION option before the DUPLICATE command, all backups sent from the target to the auxiliary database will be encrypted.

Tip: For an effective use of parallelism, allocate more AUXILIARY channels instead of TARGET channels as in earlier releases.

Finally, another important new enhancement is the possibility to finish the duplication process with the AUXILIARY database in not open state (The DEFAULT is to open the AUXILIARY database after the duplication is completed). This option is very useful when you are required to:

- Modify the block change tracking;
- Configure fast incremental backups or flashback database settings;
- Move the location of the database, as per example to ASM;
- Upgrade the AUXILIARY database (due that the database must not be open with reset logs, prior to apply the upgrade scripts);
- Or when you know that the attempt to open the database would produce errors.

Active Duplicate

To make it clearer, let's take a closer look of what operations RMAN will perform when a DUPLICATE command is used:

1. Create a SPFILE for the AUXILIARY instance;
2. Mounts the backup control file;
3. Restores the TARGET datafiles on the AUXILIARY database;
4. Performs incomplete recovery using all available incremental backups and archived redo log files;
5. Shutdown and restart the AUXILIARY instance in NOMOUNT mode;
6. Creates a new control file, create and stores the new DBID in the datafiles (It will not happen if the FOR STANDBY clause is in use);
7. MOUNT and OPEN the duplicate database using the RESETLOGS option, and creates the online redo log files by DEFAULT. If the NOOPEN option is used, the duplicated database will not be opened with RESETLOGS and will remain in MOUNT state

Here are some examples of how to use the DUPLICATE command with PDBs:

```
RMAN> DUPLICATE TARGET DATABASE TO <CDB1>;  
RMAN> DUPLICATE TARGET DATABASE TO <CDB1>  
      PLUGGABLE DATABASE <PDB1>, <PDB2>, <PDB3>;
```

Table Recovery

On previous versions of Oracle database, the process to recover a table to an specific point-in-time was never easy, but now Oracle solved this major issue introducing the possibility to do a point-in-time recovery of a table, group of tables or even table partitions without affect the remaining database objects using RMAN, making this process easy and faster than ever before. Remember that previously Oracle have introduced features like database point-in-time recovery (DBPITR), tablespace point-in-time recovery (TSPITR) and Flashback database, this is an evolution of the same technology and principles.

Note: You cannot recover tables or table partitions from the SYS, SYSTEM, and SYSAUX schemas, or even from a STANDBY database

The recovery of tables and table partitions is useful when in the following situations:

- To recover a very small set of tables to a particular point-in-time;
- To recover a tablespace that is not self-contained to a particular point-in-time, remember that TSPITR can only be used if the tablespace is self-contained;
- To recover tables that are corrupted or deleted with the PURGE option, so the FLASHBACK DROP functionality is not possible to be used;
- When logging for a Flashback table is enabled, but the flashback target time or SCN is beyond the available UNDO;
- To recover data that was lost after a data definition language (DDL) operation that changed the structure of a table.

Table Recovery (How to)

1. First check if all the prerequisites to do a table recovery are met;
2. Start a RMAN session with the CONNECT TARGET command;
3. Use the RECOVER TABLE command with all required clauses;
4. RMAN will determine what backup contains the data that needs to be recovered based on the point-in-time specified;
5. RMAN creates an AUXILIARY instance, you can also specify the location of the AUXILIARY instance files using the AUXILIARY DESTINATION or SET NEWNAME clauses;
6. RMAN recovers the specified objects into the AUXILIARY instance;
7. RMAN creates a Data Pump export dump file that contains the objects;
8. RMAN imports the recovered objects from the dump file previously created into the TARGET database, if you want to manually import the objects to the TARGET database, you can make use of the clause NOTABLEIMPORT in the RECOVER command to achieve this goal;
9. RMAN optionally offers the possibility to rename the recovered objects in the TARGET database using the REMAP TABLE clause, or also import the recovered objects to a different tablespace using the REMAP TABLESPACE clause;

An example of how to use the new RECOVER TABLE command will be:

```
RMAN> RECOVER TABLE SCOTT.test  
      UNTIL SEQUENCE 5481 THREAD 2  
      AUXILIARY DESTINATION '/tmp/recover'  
      REMAP TABLE SCOTT.test:my_test;
```

Data Pump New Features and Enhancements

Disabling Logging on Data Pump Import

A new feature on Data Pump introduced with 12c is the possibility to disable logging generation during an import operation, allowing us to have faster imports due that the redo log information is not written to disk or even archived. This is particularly useful for large data loads like database migrations.

Tip: When making use of a NOLOGGING option, always perform a full RMAN backup after the NOLOGGING operation is completed.

This new feature is now possible due to the introduction of a new metadata TRANSFORM parameter called `DISABLE_ARCHIVE_LOGGING`, it can be used on the `impdp` command line or even when using the `DBMS_DATAPUMP.METADATA_TRANSFORM` PL/SQL procedure.

Disabling Logging on Data Pump Import

Not all operations are not logged when using `DISABLE_ARCHIVE_LOGGING`, the following operations during an import process will still be logged:

CREATE and ALTER statements (the only exception is the CREATE INDEX statement);

All operations against the master table that is being used by Data Pump to coordinate its activities.

Note: If the database is in `FORCE LOGGING` mode, logging will not be disabled during an import operation that is making use of the `DISABLE_ARCHIVE_LOGGING` metadata `TRANSFORM` parameter.

Here is one example of how this new feature can be used:

```
$ impdp test/test TABLES=test.test1 DIRECTORY=datapump  
DUMPFILE=test_test1.dmp TRANSFORM=DISABLE_ARCHIVE_LOGGING:Y
```

Exporting Views as Tables

Another new feature introduced to Data Pump, is the option to export a view as a table. In this case, Data Pump will include in the dump file the corresponding table definition and all data that was visible in the view, instead to only write the view definition. This allows the Data Pump import to create it as a table with the same columns and data as the original view during the import process. All objects depending on the view will be also exported as if they were defined on the table, grants, and constraints that are associated with the view will be now recorded as grants and constraints on the corresponding table in the dump file.

Exporting Views as Tables

This new ability to export views as tables can be used to:

- Export data subsets for development and testing;
- Data Replication;
- Export a denormalized data set spanning multiple related tables (When moving data from an OLTP system to a Data Warehouse system);
- Offline archival purposes.

```
$ expdp test/test DIRECTORY=datapump  
DUMPFILE=testvwasatable.dmp VIEWS_AS_TABLES=employees_v
```

Extended Character Data Types

This new version of the Oracle database extends the maximum size of the VARCHAR2, NVARCHAR2 from 4000 bytes to 32767 bytes and the RAW data type from 2000 bytes to 32767 bytes. Columns with a declared length of 4000 bytes or less will be stored inline and for columns with a length greater than 4000 bytes are called extended character data type columns and will be stored out-of-line, leveraging the Oracle Large Object (LOB) technology. Data Pump utilities and the related packages DBMS_DATAPUMP and DBMS_METADATA (PL/SQL) are now modified to support the extended data types.

Note: Your COMPATIBLE parameter in the SPFILE should be 12.0 or greater and the MAX_SQL_STRING_SIZE parameter should be set to EXTENDED to allow Data Pump to support the extended character data types.

Encryption Password

Previously Data Pump allowed us to generate dump files with a password-based encryption key in the command line, making this easily to be hacked. Oracle now has enhanced this option adding to Data Pump the ability to prompt the user to add the encryption password without the value being echoed as it is entered. The new command line parameter `ENCRYPTION_PWD_PROMPT=Y` can be now used silently at runtime, making it not visible by commands like `ps` and also the password will not be stored in scripts.

Note: The concurrent use of the parameters `ENCRYPTION_PASSWORD` and `ENCRYPTION_PWD_PROMPT` is prohibited and will generate an error.

Compressing Tables on Import

On earlier versions, Data Pump always perform an import operation using the same compression settings that were present during the export process. Data Pump was now enhanced to allow you to specify a compression method at import time regardless of the compression method used when the data was exported, or if you prefer, you can also now uncompress tables during an import.

This new feature is now possible due to the introduction of a new metadata TRANSFORM parameter called TABLE_COMPRESSION_CLAUSE, if set to NONE, the table compression clause is ignored and it will imported using the default compression mode for the tablespace, or you can use any valid table compression clause available such as: NOCOMPRESS, COMPRESS BASIC, COMPRESS FOR OLTP, COMPRESS FOR QUERY, or COMPRESS FOR ARCHIVE.

Note: If the table compression clause has more than one word, than you must use single or double quotation marks.

```
$ impdp test/test DIRECTORY=datapump DUMPFILE=test.dmp  
TRANSFORM=TABLE_COMPRESSION_CLAUSE:'COMPRESS FOR OLTP'
```

Exporting Data from Data Vault

If using Data Pump export without encryption on a database where Database Vault is enabled, Data Pump will now generate a warning message to ensure that users are not exporting sensible data without using encryption by mistake. This is only a warning message and you can choose between continue with the export process or stop it and restart it enabling compression.

Auditing Data Pump Commands

Oracle Database 12c introduce a unified and extensible audit framework that unifies various audit trails from various database components and is extensible to accommodate additional audit types and audit data fields, Data Pump utilities were extended to leverage the new audit framework, providing a comprehensive audit trail of all invocations of Data Pump.

Here we can see some examples of the use of this new audit feature:

```
SQL > CREATE AUDIT POLICY datapump  
2     ACTIONS COMPONENT=DATAPUMP EXPORT;
```

```
SQL> AUDIT POLICY datapump;
```

```
SQL> ALTER AUDIT POLICY datapump  
2     ADD ACTION COMPONENT=DATAPUMP IMPORT;
```

```
SQL> DROP AUDIT POLICY datapump;
```

Enterprise Manager Database Express

Enterprise Manager Database Express

The Oracle Enterprise Manager Database Console or Database Control that many of us used to manage an entire database it's now deprecated and replaced by the new Oracle Enterprise Manager Database Express. This new tool uses flash technology and allow the DBA to easily manage configurations, storage, security and performance of a database.

Note that RMAN, Data Pump and the Oracle Enterprise Manager Cloud Control are now the only tools able to perform backup and recovery operations in a Pluggable Database environment.

Enterprise Manager Database Express

ORACLE Enterprise Manager Database Express 12c

CDB1 (12.1.0.0.2) Configuration Storage Security Performance

Page Refreshed 1:35:34 PM GMT+1200 Auto Refresh 1 Minute

Database Home

Status

- Up Time: 3 minutes, 17 seconds
- Type: Single instance (cdb1)
- Version: 12.1.0.0.2 Enterprise Edition
- Database Name: CDB1
- Platform Name: Linux x86 64-bit
- Host Name: oracle.oracle12c
- Oracle Home: /u01/app/oracle/product/12.1.0/c
- Instance Name: cdb1
- Thread: 1
- Archiver: Started

Performance

Activity Class: Services

Resources

Host CPU

Active Sessions

Memory

Data Storage

Incidents - Last 24 Hours

| Inst... | Time | Inci... | Pro... | Error |
|---------|------|---------|--------|-------|
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Running Jobs

| Inst... | Owner | Name | Ela... | Started |
|---------|-------|------|--------|---------|
| | | | | |

SQL Monitor - Last Hour (20 max)

| Status | Duration | ID | Sessio... | Parallel | Database Time | SQL Text |
|--------|----------|---------------|-----------|----------|---------------|---------------|
| ✓ | 12.0s | f6cz4n8y72xdc | 38 | | 11.1s | SELECT spa... |

Flashback

FLASHBACK

FLASHBACK DATABASE is only available at Container level, not at pluggable database level (12cR1). Possible will be incorporated at release 2 of 12c.

REVIEW

- **Multitenant Container**
- **RMAN New Features and Enhancements**
- **Data Pump New Features and Enhancements**
- **Enterprise Manager Express**
 - **Oracle Flashback**

Questions?

Thank you !