Great Lakes Oracle Conference

Oracle Data Pump

Creative Solutions (*for Developer & DBA*)

Biju Thomas
OneNeck IT Solutions
@biju_thomas

Oracle ACE Director

BIJU THOMAS

BijoosCom

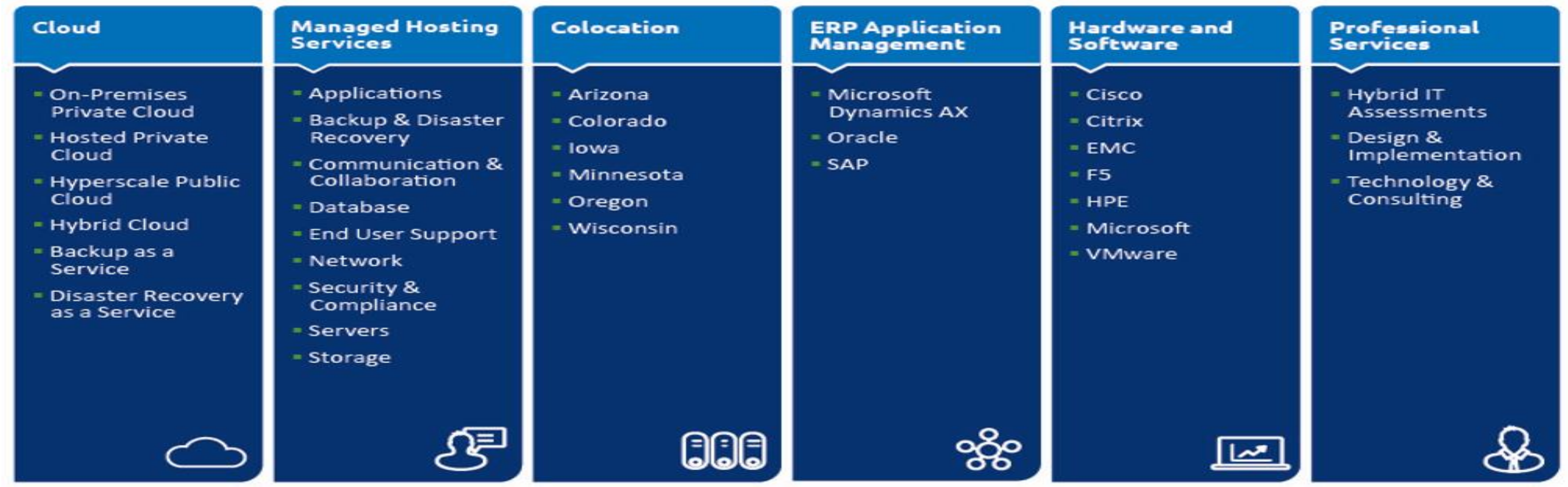
- ❖ Principal Solutions Architect with OneNeck IT Solutions
- ❖ Over 20 years of Oracle Database development and administration expertise
- ❖ Over 10 years of Oracle E-Business Suite Architecture & Tuning expertise
- ❖ First book published in September 2000, seventh in 2015
- ❖ DBA blog since 1997 – www.bijoos.com
- ❖ Daily Oracle Tidbits #oratidbit
- ❖ Oracle ACE Director

OCP Oracle8i DBA SQL and PL/SQL Study Guide
OCP Oracle8i DBA Architecture & Administration and Backup & Recovery Study Guide
OCA/OCP Introduction to Oracle9i SQL Study Guide
OCA/OCP Oracle9i DBA Fundamentals I Study Guide
OCP Oracle 10g New Features for Administrators
OCA Oracle Database 11g Administrator Certified Associate Study Guide
OCA Oracle Database 12c Administrator Certified Associate Study Guide
Biju Thomas
Oracle Database Problem Solving and Troubleshooting Handbook

OneNeck IT SOLUTIONS a TDS® Company

OneNeck IT Solutions at a Glance

- Hybrid & Multi-Cloud Solutions
- Managed Services, Advisory Services
- Coast to Coast Data Centers



Cloud

- On-Premises Private Cloud
- Hosted Private Cloud
- Hyperscale Public Cloud
- Hybrid Cloud
- Backup as a Service
- Disaster Recovery as a Service

Managed Hosting Services

- Applications
- Backup & Disaster Recovery
- Communication & Collaboration
- Database
- End User Support
- Network
- Security & Compliance
- Servers
- Storage

Colocation

- Arizona
- Colorado
- Iowa
- Minnesota
- Oregon
- Wisconsin

ERP Application Management

- Microsoft Dynamics AX
- Oracle
- SAP

Hardware and Software

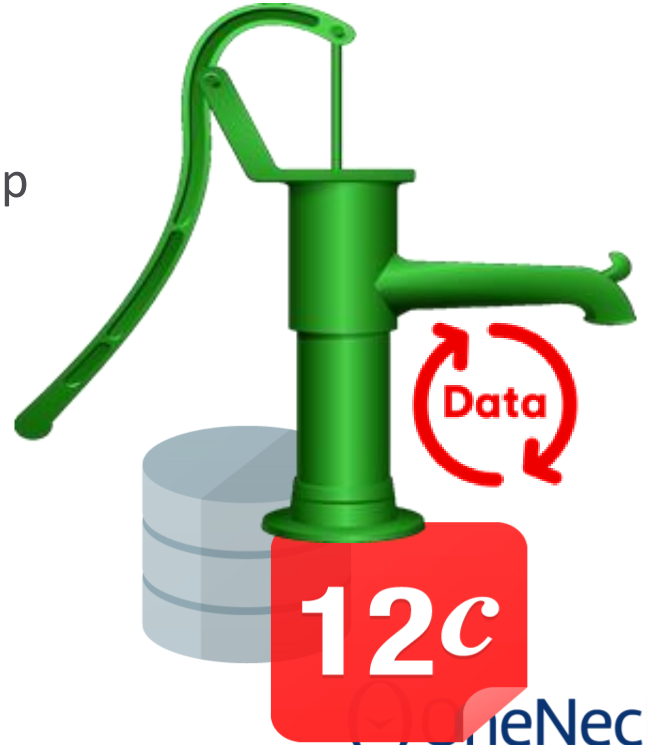
- Cisco
- Citrix
- EMC
- F5
- HPE
- Microsoft
- VMware

Professional Services

- Hybrid IT Assessments
- Design & Implementation
- Technology & Consulting

Data Pump Capabilities
Session Objectives & Agenda

- Data Pump Basics
  - Recap of what most us do with Data Pump
- Data Pump Tips
  - To help usability and performance
- Data Pump Scenarios
  - Problems and solutions



OneNeck IT SOLUTIONS a TDS® Company

Data Pump Basics

A quick recap of what most us do with DP

Export Import Tools - Introduction

- Legacy tool – exp / imp
  - Available since Oracle 5 days
  - 12c/18c supports exp / imp
  - Less flexibility, newer object types not supported
  - New features in Oracle Database 10gR2 and later releases are not supported
  - Client side tool – dump file created on the client machine
- Data Pump tool – expdp / impdp
  - Introduced in 10g
  - Each major version of database comes with multiple enhancements
  - Granular selection of objects or data to export / import
  - Jobs can be monitored, paused, stopped, restarted
  - Server side tool – dump file created on the DB server
  - PL/SQL API – DBMS_DATAPUMP, DBMS_METADATA

Data Pump – Common Parameters

expdp / impdp / DBMS_DATAPUMP

- DIRECTORY (*default DATA_PUMP_DIR*)
- DUMPFILE (*default expdat.dmp*)
- LOGFILE (*default export.log*)
- CONTENT =[ALL | DATA_ONLY | METADATA_ONLY]
- EXCLUDE
- INCLUDE

- Modes
  - TABLES
  - SCHEMAS
  - FULL
- Additional Modes
  - TABLESPACES
  - TRANSPORT_TABLESPACES

Data Pump Interactive Mode

expdp / impdp

- Press ^C to exit out of the datapump log output mode to interactive mode.
- ATTACH=<job_name>
- Query DBA_DATAPUMP_JOBS → JOB_NAME, STATE
  - Tip: Use JOB_NAME parameter in the export or import, no need to look up the job name.
- STOP_JOB[=IMMEDIATE]
- START_JOB[=SKIP_CURRENT]
- KILL_JOB

- HELP
- ADD_FILE
- FILE_SIZE
- PARALLEL
- STATUS

- CONTINUE_CLIENT
- EXIT_CLIENT

INCLUDE & EXCLUDE

Two mutually exclusive powerful parameters

```
EXCLUDE=object_type[:name_clause] [, ...]
INCLUDE=object_type[:name_clause] [, ...]
```

- Name_Clause
  - Optional
  - SQL expression allowed

- Examples:
  - INCLUDE=TABLE:"IN ('EMPLOYEES', 'DEPARTMENTS')"
  - INCLUDE=PROCEDURE
  - INCLUDE=INDEX:"LIKE 'EMP%'“
  - EXCLUDE=SCHEMA:"='HR'"

OneNeck IT SOLUTIONS a TDS® Company

Data Pump Tips

Improve Productivity

Valid INCLUDE / EXCLUDE Values

Granular Object Filtering Options

- Full database export/import
  - *DATABASE_EXPORT_OBJECTS*.
- Schema level export/import
  - *SCHEMA_EXPORT_OBJECTS*
- Table level or tablespace level export/import
  - *TABLE_EXPORT_OBJECTS*

Looking for valid options to exclude constraints.

```
SQL> SELECT object_path, comments
     FROM   database_export_objects
     WHERE  object_path like '%CONSTRAINT%'
     AND    object_path not like '%/%';

OBJECT_PATH
COMMENTS
--------------------------------------------
-------
CONSTRAINT
Constraints (including referential
constraints)

REF_CONSTRAINT
Referential constraints
```

Estimating Space Requirement

- ESTIMATE_ONLY=Y
- Gives the expected export size of the tables in the dump file.

- ESTIMATE=STATISTICS
- Default is BLOCKS, which is more accurate
- If estimated size of the file is not a concern, always use STATISTICS

Tools / API

- PL/SQL
- SQL Developer
- OEM Cloud Control
- Database Express

Data Pump in PL/SQL

DBMS_DATAPUMP

- *OPEN*: To define an export or import job
- *ADD_FILE*: To define the dump file name and log file name
- ***METADATA_FILTER***: provide the exclude or include clauses here
- ***SET_PARAMETER***: parameter and values
- *START_JOB*: kick off the export or import job
- *WAIT_FOR_JOB*: wait until the job completes
- *DETACH*: close the job.

OneNeck IT SOLUTIONS a TDS® Company

Bulk Load Statistics Collection

- **_OPTIMIZER_GATHER_STATS_ON_LOAD=FALSE**
- 12c adds this new parameter that can slow down import operations when left at it's default setting of TRUE.
- With 12c, if you intend on using EXCLUDE=STATISTICS for an import operation, then also set _OPTIMIZER_GATHER_STATS_ON_LOAD=FALSE at the database level.
- Manually gather database statistics after that import operation has completed successfully.
  - Init Param: _OPTIMIZER_GATHER_STATS_ON_LOAD=FALSE
  - DP Param: EXCLUDE=STATISTICS
  - SQL: EXEC DBMS_STATS.GATHER_DATABASE_STATS;

Dictionary Statistics

- Collect *DBMS_STATS.GATHER_DICTIONARY_STATS* before a FULL Data Pump export job for better performance, if dictionary stats is not collected recently.

Collect Statistics After Import

- Exclude statistics and Collect statistics after import
  - EXCLUDE=STATISTICS

RAC Considerations

- On RAC database use CLUSTER=N
  - Also, set PARALLEL_FORCE_LOCAL=TRUE

Disable Archive Logging During Import

- No Logging Operation during import
  - TRANSFORM=DISABLE_ARCHIVE_LOGGING:Y
  - The objects are created by the Data Pump in NOLOGGING mode, and are switched to its appropriate LOGGING or NOLOGGING once import completes. If your database is running in FORCE LOGGING mode, this Data Pump no logging will have no impact.

Parallelizing Import

- Parallelizing Data Pump jobs using PARALLEL parameter
- Impact of PARALLEL in impdp
  - First worker – metadata tablespaces, schema & tables
  - Parallel workers load table data
  - One worker again loads metadata, till package spec
  - Parallel workers load package bodies
  - One worker loads remaining metadata

Bigger DP Jobs

- Bigger STREAMS_POOL_SIZE

- For bigger jobs, try to exclude INDEX in full export/import, and create indexes immediately after the table import completes using script, may be in multiple sessions.

DISABLE_APPEND_HINT

- One of the DATA_OPTIONS parameters.
- Use if the imported table is being used by the application.

- Use if splitting the import into multiple smaller imports that kick off in parallel.

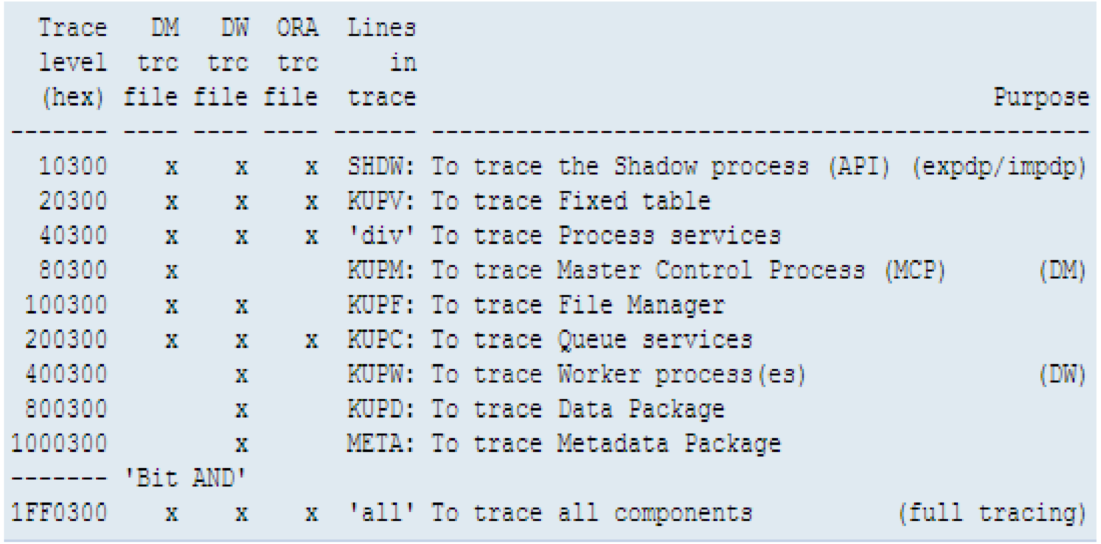
```
$ impdp TABLES=sales_data CONTENT=DATA_ONLY
        DUMPFILE=dpump_dir:salesdata08.dmp
           DATA_OPTIONS=DISABLE_APPEND_HINT
```

Dictionary Views

- DBA_DATAPUMP_JOBS
  - Shows a summary of all active Data Pump jobs on the system.
- DBA_DATAPUMP_SESSIONS
  - Shows all sessions currently attached to Data Pump jobs.
- V$SESSION_LONGOPS
  - Shows progress on each active Data Pump job.

Tracing Datapump

- Export/Import DataPump Parameter TRACE - How to Diagnose Oracle Data Pump (Doc ID 286496.1)
- Parameter TRACE=<7 digit hex number> (480300 is common)



```
  Trace   DM   DW  ORA  Lines
  level  trc  trc  trc     in
  (hex) file file file  trace                                         Purpose
------- ---- ---- ---- ------ -----------------------------------------------
  10300    x    x    x  SHDW: To trace the Shadow process (API) (expdp/impdp)
  20300    x    x    x  KUPV: To trace Fixed table
  40300    x    x    x  'div' To trace Process services
  80300    x            KUPM: To trace Master Control Process (MCP)      (DM)
 100300    x    x       KUPF: To trace File Manager
 200300    x    x    x  KUPC: To trace Queue services
 400300         x       KUPW: To trace Worker process(es)                (DW)
 800300         x       KUPD: To trace Data Package
1000300         x       META: To trace Metadata Package
------- 'Bit AND'
1FF0300    x    x    x  'all' To trace all components          (full tracing)
```

How to Gather the Header Information and the Content of an Export Dumpfile ? (Doc ID 462488.1)

Tracking Time

- METRICS=YES
- The number of objects and the elapsed time are recorded in the Data Pump log file.

- LOGTIME=[NONE | STATUS | LOGFILE | ALL]
  - STATUS → Console only
  - LOGFILE → Log file only
  - ALL → Console and Logfile
- Specifies that messages displayed during **export/import** operations be time stamped.
- Use the timestamps to figure out the elapsed time between different phases of a Data Pump operation.

Export from Read-Only Database

- Read-only database: 11.2.0.4
- Running database: 12.1.0.1
- Create a database link (e.g.: my_dblink) from the local 12.1.0.1.0 database to the 11.2.0.4.0 read-only source database.
- Start the export job using the 12.1.0.1.0 Export Data Pump client that connects to the 12.1.0.1.0 local database. Use the parameters NETWORK_LINK=my_dblink VERSION=11.2 to create a dumpfile set with a lower compatibility level.
- Transfer the dumpfile set to the server where the target database is located.
- Start the import of the data using the 11.2.0.4.0 Import Data Pump client that connects to the 11.2.0.4.0 target database.

Scenarios

Problem / Solution

Building a Metadata Repository

Use for documentation or change control purposes

- Source Code Generation
- Baseline Code – populate version control tool

- Export using data pump
  - CONTENT=METADATA_ONLY
  - Use FULL, SCHEMAS, INCLUDE, EXCLUDE as appropriate.
- Import using data pump with SQLFILE parameter
  - You can extract DDL for specific objects by using INCLUDE or EXCLUDE parameters.

```
$ expdp
▪   CONTENT=METADATA_ONLY
▪   SCHEMAS=MYAPP
$ impdp
▪   SQLFILE=myapp_procs.txt
▪   INCLUDE=PROCEDURE
$impdp
▪   SQLFILE=myapp_mytab.txt
▪   INCLUDE=TABLE:"LIKE 'MYTAB%'”
$ impdp
▪   SQLFILE=myapp_tab.txt
▪   EXCLUDE=PROCEDURE,FUNCTION,PA
    CKAGE,TRIGGER,REF_CONSTRAINT
```

Create User / Schema as another

Duplicate user accounts easily

- Create a user like another user with objects, but no data.

- Create a user like another user with tablespace quotas, roles assigned, system privileges, object privileges.

```
expdp SCHEMAS=XX
CONTENT=METADATA_ONLY
impdp
REMAP_SCHEMA=XX:XXNEW

expdp SCHEMAS=XX
CONTENT=METADATA_ONLY
impdp
REMAP_SCHEMA=XX:XXNEW
INCLUDE=
```

Exporting PUBLIC Objects

- The EXCLUDE and INCLUDE parameters are very flexible and powerful, and they accept a subquery instead of hard coded values.
- Export only the public database links.
  - *full=y*
  - *include=db_link:"IN (SELECT db_link from dba_db_links Where Owner = 'PUBLIC')"*
- Export public synonyms defined on the schema HR
  - *full=y*
  - *include=PUBLIC_SYNONYM:"IN (SELECT synonym_name from dba_synonyms Where Owner = 'PUBLIC' and table_owner = 'HR')"*

Exporting Database Links

- Problem: cannot qualify the database link with a schema name to create the database link.
- Use INCLUDE parameter to export only database links
- The database links for the whole database can be saved to a dump file using

  -
    ```
    $ expdp dumpfile=uatdblinks.dmp directory=clone_save
    full=y include=db_link userid =\"/ as sysdba\"
    ```
- If you are interested only in the private database links belonging few schema names

  -
    ```
    $ expdp dumpfile=uatdblinks.dmp directory=clone_save
    schemas=APPS,XX,XXL include=db_link userid =\"/ as
    sysdba\"
    ```

Verifying Content of Dump File

- Use the impdp import with SQLFILE parameter, instead of doing the actual import, the metadata from the dump file is written to the file name specified in SQLFILE parameter.
  - *`$ impdp dumpfile=pubdblinks.dmp directory=clone_save full=y`* **`sqlfile=testfile.txt`**
- If you are not interested in the rows, make sure you specify CONTENT=METADATA_ONLY
- Validate the full export did exclude HR and SH, use the impdp with SQLFILE. We just need to validate if there is a CREATE USER statement for HR and SH users. INCLUDE=USER only does the user creation, no schema objects are imported.
  - *`sqlfile=validate.txt`*
  - *`include=USER`*

Example: SQLFILE Text

- Public Database Link Export Dump File Content

```
$ cat testfile.txt
-- CONNECT SYS
ALTER SESSION SET EVENTS '10150 TRACE NAME CONTEXT FOREVER, LEVEL 1';
ALTER SESSION SET EVENTS '10904 TRACE NAME CONTEXT FOREVER, LEVEL 1';
ALTER SESSION SET EVENTS '25475 TRACE NAME CONTEXT FOREVER, LEVEL 1';
ALTER SESSION SET EVENTS '10407 TRACE NAME CONTEXT FOREVER, LEVEL 1';
ALTER SESSION SET EVENTS '10851 TRACE NAME CONTEXT FOREVER, LEVEL 1';
ALTER SESSION SET EVENTS '22830 TRACE NAME CONTEXT FOREVER, LEVEL 192 ';
-- new object type path: DATABASE_EXPORT/SCHEMA/DB_LINK
CREATE PUBLIC DATABASE LINK "CZPRD.BT.NET"
   USING 'czprd';
CREATE PUBLIC DATABASE LINK "ERP_ARCHIVE_ONLY_LINK.BT.NET"
   CONNECT TO "AM_STR_HISTORY_READ" IDENTIFIED BY VALUES
'05EDC7F2F211FF3A79CD5A526EF812A0FCC4527A185146A206C88098BB1FF8F0B1'
   USING 'ILMPRD1';
```

Exporting subset of Data

- QUERY parameter
  - Specify a WHERE clause
- SAMPLE parameter
  - Specify percent of rows
- VIEWS_AS_TABLES parameter
  - Create a view to extract required data
  - Export view as table

Subset Using QUERY Parameter

- QUERY parameter
  - The *WHERE* clause you specify in *QUERY* is applied to all tables included in the export
- Multiple QUERY parameters are permitted, restrict QUERY to a single table allowed.
  - *schemas=GL*
  - *query=GL_TRANSACTIONS:"where account_date > add_months(sysdate, -6)"*
  - *query=GL.GL_LINES:" where transaction_date > add_months(sysdate, -6)"*

Subset Data: QUERY Example

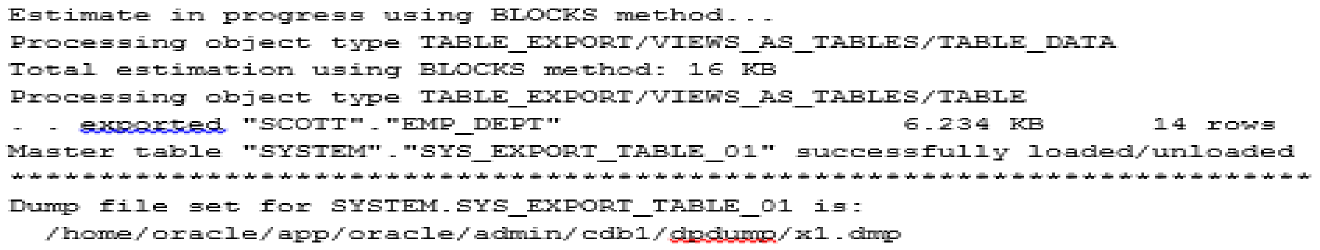
- We want to apply the query condition to all tables that end with TRANSACTION in GL schema. Perform two exports.
- First:
  - *`schemas=GL`*
  - *`include=table:"like '%TRANSACTION'"`*
  - *`query="where account_date > add_months(sysdate, -6)"`*
- Second:
  - *`schemas=GL`*
  - *`exclude=table:"like '%TRANSACTION'"`*

Subset Using SAMPLE Parameter

- Specify a percentage of rows to export.
- Similar to QUERY, you can limit the sample condition to just one table or to the entire export.
- Example: All the tables in GL schema will be exported, for GL_TRANSACTIONS only 20 percent of rows exported and for GL_LINES only 30 percent of rows exported.
  - *`schemas=GL`*
  - *`sample=GL_TRANSACTIONS:20`*
  - *`sample=GL.GL_LINES:30`*

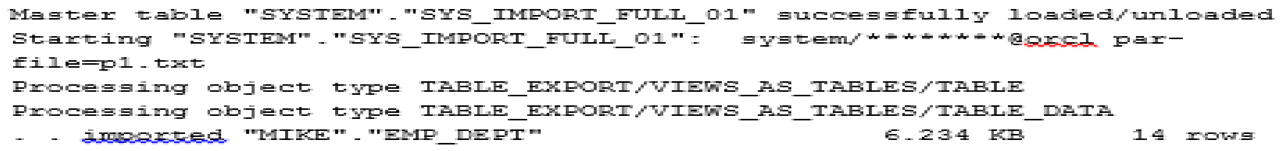
Using VIEWS_AS_TABLES parameter

- Good way to export subset of data, if the data set is based on more than one table or complex joins.
- *views_as_tables=scott.emp_dept*

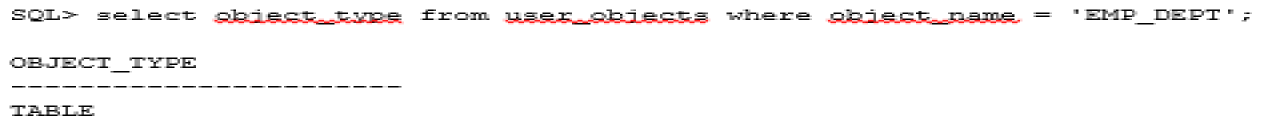


```
Estimate in progress using BLOCKS method...
Processing object type TABLE_EXPORT/VIEWS_AS_TABLES/TABLE_DATA
Total estimation using BLOCKS method: 16 KB
Processing object type TABLE_EXPORT/VIEWS_AS_TABLES/TABLE
. . exported "SCOTT"."EMP_DEPT"                    6.234 KB      14 rows
Master table "SYSTEM"."SYS_EXPORT_TABLE_01" successfully loaded/unloaded
******************************************************************************
Dump file set for SYSTEM.SYS_EXPORT_TABLE_01 is:
  /home/oracle/app/oracle/admin/cdb1/dpdump/x1.dmp
```

  - *Import to another schema in same database* **remap_schema=scott:mike**



```
Master table "SYSTEM"."SYS_IMPORT_FULL_01" successfully loaded/unloaded
Starting "SYSTEM"."SYS_IMPORT_FULL_01":  system/********@orcl par-
file=p1.txt
Processing object type TABLE_EXPORT/VIEWS_AS_TABLES/TABLE
Processing object type TABLE_EXPORT/VIEWS_AS_TABLES/TABLE_DATA
. . imported "MIKE"."EMP_DEPT"                     6.234 KB      14 rows
```



```
SQL> select object_type from user_objects where object_name = 'EMP_DEPT';

OBJECT_TYPE
-----------------------
TABLE
```

How to Mask Data in Non-Prod

- Employee table (HR.ALL_EMPLOYEES) contain the social security number of employee along with other information. When copying this table from production to development database, you want to scramble the social security number, or mask it or whatever a function return value can be.
  - First create a package and a function that returns the scrambled SSN when the production SSN is passed in as a parameter. Let’s call this procedure HR_UTILS.SSN_SCRAMBLE.
  - During import use the REMAP_DATA parameter change the data value.

```
TABLES=HR.ALL_EMPLOYEES
REMAP_DATA=HR.ALL_EMPLOYEES.SSN:HR.HR_UTILS.SSN_SCRAMBLE
```

  - If you want to secure the dump file, it may be appropriate to use this function and parameter in the export itself, thus the dump file will not have SSN, instead the scrambled numbers only.

Ignore Storage Attributes

- If you do not wish import to bring any of the attributes associated with a table or index segment, and have the object take the defaults for the user and the destination database/tablespace, you could do so easily with a transform option.
  - `TRANSFORM=SEGMENT_ATTRIBUTES` → `TABLESPACE, STORAGE`
  - `TRANSFORM=STORAGE` → `STORAGE`
- In the example, SALES schema objects are imported, but the tablespace, storage attributes and logging clause all will default to what is defined for the user MIKE and his default tablespace.
  - `SCHEMAS=SALES`
  - `REMAP_SCHEMA=SALES:MIKE`
  - `TRANSFORM=SEGMENT_ATTRIBUTES:N`
- In the above example, if you would like to keep the original tablespace but only want to remove the storage attributes for object type index, you could specify the optional object along with the TRANSFORM parameter.
  - `SCHEMAS=SALES`
  - `REMAP_SCHEMA=SALES:MIKE`
  - `TRANSFORM=SEGMENT_ATTRIBUTES:N:TABLE`
  - `TRNASFORM=STORAGE:Y:INDEX`

Changing Table Partition Properties

- Data Pump import is very flexible to change the properties of objects during import.
- Partitioned table SALES_DATA owned by SLS schema exported using Data Pump exp. During import, need to create the table as non-partitioned under SLS_HIST schema.
  - `REMAP_SCHEMA=SLS:SLS_HIST`
  - `TABLES=SLS.SALES_DATA`
  - `PARTITION_OPTIONS=MERGE`
- Partitioned table SALES_DATA owned by SLS schema exported using Data Pump. During import, need to create separate non-partitioned table for each partition of SALES_DATA.
  - `PARTITION_OPTIONS=DEPARTITION`
  - `TABLES=SLS.SALES_DATA`
  - Since the partition names in SALES_DATA are SDQ1, SDQ2, SDQ3, SDQ4, the new table names created would be SALES_DATA_SDQ1, SALES_DATA_SDQ2, SALES_DATA_SDQ3, SALES_DATA_SDQ4.

OneNeck IT SOLUTIONS a TDS Company

Export Data as of Past Point in Time

- FLASHBACK_TIME=systimestamp
- Default is current timestamp

- Example:

```
flashback_time="to_timestamp('15-11-2016 12:24:00', 'DD-
MM-YYYY HH24:MI:SS')"
```

- Uses SCN close to timestamp to perform consistent export

Export/Import Across Database Link

- NETWORK_LINK=database_link_name
- Export and Import through database link, without writing a dump file.
  - Current user database link is not supported
  - LONG column is not supported (pre 12.2)
  - Database versions cannot differ more than 2 versions
- New in 12.2:
  - ENABLE_NETWORK_COMPRESSION option on the Data Pump DATA_OPTIONS parameter tells Data Pump to compress data before sending it over the network.
  - The DIRECT_PATH option on the Import ACCESS_METHOD parameter

Data Pump Legacy Mode

- If you are familiar with exp/imp, the same parameters may be used with expdp/impdp

CONSISTENT → FLASHBACK_TIME
FEEDBACK → STATUS=30
GRANTS=N → EXCLUDE=GRANT
INDEXES=N → EXCLUDE=INDEX
OWNER → SCHEMAS
ROWS=N → CONTENT=METADATA_ONLY
TRIGGERS=N → EXCLUDE=TRIGGER

IGNORE=Y → TABLE_EXISTS_ACTION=APPEND
INDEXFILE → SQLFILE
TOUSER → REMAP_SCHEMA

Import to Lower Version

- VERSION=[COMPATIBLE | LATEST | version_string]
- The VERSION parameter allows to identify the version of objects.
- Import Data Pump can always read Export Data Pump dumpfile sets created by older versions of the database.
- Export/Import DataPump Parameter VERSION - Compatibility of Data Pump Between Different Oracle Versions [Video] (Doc ID 553337.1)

www.OneNeck.com

Daily #oratidbit on Facebook and Twitter

Tweets: @biju_thomas
Facebook: facebook.com/oraclenotes
Google+: +bijoosoraclenotes
Blog: bijoos.com/oraclenotes

OneNeck®
IT SOLUTIONS
a TDS® Company

ODTUG
Kscope18
ORLANDO, FLORIDA • JUNE 10-14
I'M SPEAKING AT
ORLANDO

Biju.Thomas@GMail.com

Download

Thank you!

OneNeck® IT SOLUTIONS a TDS® Company

@biju_thomas

@oraclenotes