CHAPTER 13

Data Pump

Data Pump was introduced in Oracle Database 10g. It replaces the older exp/imp utilities. Data Pump enables you to quickly move data and/or metadata from one environment to another. You can use Data Pump in a variety of ways:

- Point-in-time logical backups of the entire database or subsets of data
- Replicating entire databases or subsets of data for testing or development
- Quickly generating Data Definition Language (DDL) required to re-create objects

Sometimes DBAs hold on to the old exp/imp utilities because they’re familiar with the syntax and these utilities get the job done quickly. Even if those legacy utilities are easy to use, you should consider using Data Pump going forward. Data Pump contains substantial functionality over the old exp/imp utilities:

- Performance with large data sets, allowing you to efficiently export and import gigabytes of data
- Interactive command-line utility, which gives you the ability to disconnect and then later re-attach to active Data Pump jobs
- Ability to export and import large amounts of data from a remote database directly into a local database without creating a dump file
- Ability to make on-the-fly changes to schemas, tablespaces, datafiles, and storage settings from export to import
- Sophisticated filtering of objects and data
- Security controlled via database-directory objects
- Advanced features such as compression and encryption

This chapter begins with a discussion on the Data Pump architecture. Subsequent topics include basic export and import tasks, moving data across networks, filtering data, and running Data Pump in legacy mode.
Data Pump Architecture

Data Pump consists of the following components:

- `expdp` (Data Pump export utility)
- `impdp` (Data Pump import utility)
- `DBMS_DATAPUMP` PL/SQL package (Data Pump API)
- `DBMS_METADATA` PL/SQL package (Data Pump Metadata API)

The `expdp` and `impdp` utilities use the `DBMS_DATAPUMP` and `DBMS_METADATA` built-in PL/SQL packages when exporting and importing data and metadata. The `DBMS_DATAPUMP` package moves entire databases or subsets of data between database environments. The `DBMS_METADATA` package exports and imports information about database objects.

**Note** You can call the `DBMS_DATAPUMP` and `DBMS_METADATA` packages independently (outside of `expdp` and `impdp`) from SQL*Plus. I rarely call these packages directly from SQL*Plus; but you may have a specific scenario where it’s desirable to interact directly with them. See the *Oracle Database PL/SQL Packages and Types Reference* guide (available on OTN) for more details.

When you start a Data Pump export or import job, a master operating-system process is initiated on the database server. This master process name has the format `ora_dmNN_<SID>`. On Linux/Unix systems, you can view this process from the operating-system prompt using the `ps` command:

```
$ ps -ef | grep ora_dm
oracle 14950   717   0 10:59:06 ?       0:10 ora_dm00_STAGE
```

Depending on the degree of parallelism and the work specified, a number of worker processes are also started. The master process coordinates the work between master and worker processes. The worker process names have the format `ora_dwNN_<SID>`.

Also, when a user starts an export or import job, a database status table is created (owned by the user who starts the job). This table exists for the duration of the Data Pump job. The name of the status table is dependent on what type of job you’re running. The table is named with the format `SYS_<OPERATION>_<JOB_MODE>_NN`, where `OPERATION` is either `EXPORT` or `IMPORT`. `JOB_MODE` can be `FULL`, `SCHEMA`, `TABLE`, `TABLESPACE`, and so on.

For example, if you’re exporting a schema, a table is created in your account with the name `SYS_EXPORT_SCHEMA_NN`, where `NN` is a number that makes the table name unique in the user’s schema. This status table contains information such as the objects exported/imported, start time, elapsed time, rows, error count, and so on. The status table has over 80 columns.

The status table is dropped by Data Pump upon successful completion of an export or import job. If you use the `KILL_JOB` interactive command, the master table is also dropped. If you stop a job with the `STOP_JOB` interactive command, the table isn’t removed and is used in the event you restart the job.

If your job terminates abnormally, the master table is retained. You can delete the status table if you don’t plan to restart the job.