LAB 4: DATA TRANSPORTATION

DUE DATE: Week of November 4, 2024

GOAL: This lab assignment is related to the second course learning outcome (performing the roles of a DBA including storage, user management, and data persistence). In particular, the assignment is aimed at investigating methods for managing and moving data.

In this lab, you will undertake the following activities:

1. Use the SQL*Loader utility with conventional and direct path data loading
2. Loading data into the database using the export (exp) and import (imp) utilities

ACTIVITIES:

You have been provided with the following files that contain SQL commands for creating and populating the department and employee tables along with other files to facilitate data upload:

a) loademp1.sql
b) loademp1.ctl
c) loademp.dat
d) lab4setup.sql
e) empdata.sql
f) emp.sql
g) deptdata.sql
h) dept.sql

The lab is divided into two parts. In the first part, you will use the SQL*Loader utility to load data into the database. In the second part, you will load data using export and import.

DELIVERABLES:

1. Review your work with the lab instructor.
2. Record the output for each step and submit on Brightspace (either in a .txt file or take snapshots of the output in a .pdf file).

PART 0: PREPARATION

1. Connect as SYS and run the lab4setup.sql script

PART 1: SQL*LOADER

SQL*Loader is run directly on the database server from the command prompt. Results follow the outline of steps below. Here is a link that describes the SQL*Loader command line parameters: http://download.oracle.com/docs/cd/B28359_01/server.111/b28319/ldr_concepts.htm

1. Open a SQL*Plus session and connect as happy (password "dwarf") to the CIT487 instance. Run the following script to create the load_emp table.

```
$ sqlplus happy/dwarf@mydatabase
…
SQL> @loademp1
SQL> exit
```

2. Run the SQL*Loader utility from the operating system prompt, making sure you are in the directory where you copied the "load" files.

```
$ sqlldr userid=happy/dwarf control=loademp1.ctl
…
Commit point reached - logical record count 64
…
Commit point reached - logical record count 1000
$
```

3. See how many blocks were allocated to the table.

```
$ sqlplus happy/dwarf@mydatabase
SQL> SELECT bytes, blocks, extents, initial_extent, next_extent FROM user_segments
WHERE segment_name = 'LOAD_EMP';

SQL> SELECT tablespace_name, extent_id, bytes, blocks FROM user_extents WHERE
segment_name = 'LOAD_EMP';

SQL> select count(*) from load_emp;
```

4. Delete all but one row (for employee 5600) and run the SQL*Loader utility again:

```
SQL> delete from load_emp where empno <> 5600;
SQL> exit
$ sqlldr userid=happy/dwarf control=loademp1.ctl
…
Commit point reached - logical record count 1000
```

5. Delete all but one row (for employee 5600) and run SQL*Loader utility again using *DIRECT path* load:

```
$ sqlplus happy/dwarf@mydatabase
SQL> select count(*) from load_emp;
SQL> delete from load_emp where empno <> 5600;
SQL> exit
```

```
$ sqlldr userid=happy/dwarf control=loademp1.ctl direct=true;
…
Load completed - logical record count 1000.
```

6. How many blocks contain data? How many blocks have been allocated to the table?

```
$ sqlplus happy/dwarf@mydatabase

SQL> SELECT bytes, blocks, extents, initial_extent, next_extent FROM user_segments
WHERE segment_name = 'LOAD_EMP';

SQL> SELECT tablespace_name, extent_id, bytes, blocks FROM user_extents WHERE
segment_name = 'LOAD_EMP';

SQL> select count(*) from load_emp;
```

PART 2: EXPORT AND IMPORT

To export, you must have CREATE_SESSION privilege on an Oracle database.

7. As SYS, create a test user EXP_TEST with password EXP_TEST, default tablespace USERS, and temporary tablespace TEMP.

```
SQL> CREATE USER exp_test IDENTIFIED BY exp_test DEFAULT TABLESPACE users TEMPORARY
TABLESPACE temp;

SQL> GRANT CREATE SESSION, CREATE TABLE TO exp_test;

SQL> ALTER USER exp_test QUOTA 2M ON users;
```

8. In SQL*Plus, connect as EXP_TEST and run the following script files:

```
SQL> connect exp_test/exp_test@mydatabase
SQL> @dept
SQL> @deptdata
SQL> @emp
SQL> @empdata
SQL> exit
```

9. Export from the user EXP_TEST using the defaults.

```
$ exp
…
Username: exp_test
Password:
…
Enter array fetch buffer size: 4096 >

Export file: expdat.dmp >

(2)U(sers), or (3)T(ables): (2)U >
…
. about to export EXP_TEST's tables via Conventional Path ...
. . exporting table                DEPT        4 rows exported
. . exporting table                    EMP        14 rows exported
…
. exporting statistics
Export terminated successfully without warnings.
```

10. Start SQL*Plus, connect as user exp_test, and drop tables emp and dept

```
$ sqlplus exp_test/exp_test@mydatabase
…
SQL> drop table emp;
SQL> drop table dept;
```

11. Restore the emp and dept tables by using the Import utility:

```
$ imp exp_test/exp_test@mydatabase
…
Import file: expdat.dmp >

Enter insert buffer size (minimum is 8192) 30720>

Export file created by EXPORT:V10.02.01 via conventional path
import done in US7ASCII character set and AL16UTF16 NCHAR character set
List contents of import file only (yes/no): no >

Ignore create error due to object existence (yes/no): no >

Import grants (yes/no): yes >

Import table data (yes/no): yes >

Import entire export file (yes/no): no >
Username: exp_test

Enter table(T) or partition(T:P) names. Null list means all tables for user
Enter table(T) or partition(T:P) name or . if done:

importing EXP_TEST's objects into EXP_TEST
importing table                 "DEPT"          4 rows imported
importing table                  "EMP"        14 rows imported
Import terminated successfully without warnings.
```

12. Start SQL*Plus, connect as user exp_test, and verify that the emp and dept tables were restored correctly:

```
$ sqlplus exp_test/exp_test@mydatabase
…
SQL> select * from dept;
SQL> select * from emp;
```