Islamic University of Technology

CSE-4410
Database Management Systems - II Lab

Lab Report - 2

Name: Mukit Mahdin
ID: 200042170
Department: CSE
Program: SWE

Introduction :

A tablespace in a relational database is a logical container for one or more datafiles, which store the actual data for the tables and other database objects. A tablespace can be thought of as a container for storing data on disk. Each tablespace in a database corresponds to one or more physical files on disk, known as datafiles. Datafiles are the actual place where the data is stored. Datafiles are organized into one or more tablespaces, which provide a way to group related logical structures togeth

Task - 1 : Create two tablespaces tbs1, tbs2.

```
CREATE TABLESPACE TBS1
 DATAFILE 'tbs1_data.dbf' SIZE 1m;

CREATE TABLESPACE TBS2
 DATAFILE 'tbs2_data.dbf' SIZE 1m;
```

Explanation : These is the set of commands in SQL to create two tablespaces named TBS1 and TBS2, each with a single datafile named 'tbs1_data.dbf' and 'tbs2_data.dbf' respectively, and each datafile has a size of 1 megabyte. These tablespaces are logical containers for datafiles, which store the actual data for the tables and other database objects.

Task - 2 : Set quota for a single user on both tablespaces.

```
CREATE USER TESTUSER IDENTIFIED BY 123;
ALTER USER TESTUSER
QUOTA 1M ON TBS1;
ALTER USER TESTUSER
QUOTA 1M ON TBS2;

GRANT QUOTA ON TBS1 TO TESTUSER;
GRANT QUOTA ON TBS2 TO TESTUSER;
```

The set of commands in SQL to create a user named TESTUSER with a password of 123, then assign a quota of 1 megabyte on tablespaces TBS1 and TBS2 for that user, and then grant the user the ability to use the assigned quota on both tablespaces TBS1 and TBS2. The first command creates a user named TESTUSER with a password of 123. The next two commands assign a quota of 1 megabyte on tablespaces TBS1 and TBS2 for the user TESTUSER. The last two commands grant the user TESTUSER the ability to use the assigned quota on tablespaces TBS1 and TBS2.

Task - 3 : Create two tables student (name, id, fk[dept]) and department (id, name) in tbs1.

```
CREATE TABLE DEPARTMENT(
    ID NUMBER PRIMARY KEY,
    DEPT_NAME VARCHAR(32)
) TABLESPACE TBS1;

CREATE TABLE STUDENT(
    ID NUMBER PRIMARY KEY,
    STUDENT_NAME VARCHAR2(32),
    DEPT NUMBER,
    CONSTRAINT FK_DEPARTMENT FOREIGN KEY(DEPT) REFERENCES
DEPARTMENT(ID)
) TABLESPACE TBS1;
```

Task - 4 : Create another table course (code, name, credit, fk[offer_by]) in tbs2.

```
CREATE TABLE COURSE(
    CODE INT PRIMARY KEY,
    NAME VARCHAR(50),
    CREDIT INT,
    OFFERED_BY INT,
    CONSTRAINT FK_DEPT FOREIGN KEY(OFFERED_BY) REFERENCES
DEPARTMENT(ID)
) TABLESPACE TBS2;
```

Task - 5 : Insert a large amount of data in the student table and course table.

```
BEGIN
  INSERT INTO department (id, dept_name) VALUES (0, 'CIVIL');
  INSERT INTO department (id, dept_name) VALUES (1, 'Computer
Science');
  INSERT INTO department (id, dept_name) VALUES (2, 'Mechanical
Engineering');
  INSERT INTO department (id, dept_name) VALUES (3, 'Electrical
Engineering');
  INSERT INTO department (id, dept_name) VALUES (4, 'Biology');
  INSERT INTO department (id, dept_name) VALUES (5, 'Chemistry');
END;
/
```

```
DECLARE
  i NUMBER := 0;
BEGIN
  FOR i IN 1..2000 LOOP
    INSERT INTO student (id, student_name, dept)
    VALUES (i, 'student' || i, MOD(i,6));
  END LOOP;
END;
/

DECLARE
  i NUMBER := 1;
BEGIN
  FOR i IN 1..2000 LOOP
    INSERT INTO course (code, name, credit, offered_by)
    VALUES (i, 'course' || i, 4, MOD(i,6));
  END LOOP;
END;
/
```

Explanation : The set of SQL commands that demonstrate the use of a PL/SQL loop to insert data into two different tables, "student" and "course", in a database. The first block of code uses a PL/SQL loop to insert 2000 records into the "student" table, with the values for the columns "id", "student_name", and "dept" being determined by the loop variable "i". The value for the "id" column is simply the value of the loop variable, the value for the "student_name" column is the string "student" concatenated with the value of the loop variable, and the value for the "dept" column is the result of taking the modulus of the loop variable and 6.

The second block of code uses a similar PL/SQL loop to insert 2000 records into the "course" table, with the values for the columns "code", "name", "credit", and "offered_by" being determined in the same way as in the first block of code.

Task - 6 : Check the free space of the tablespaces.

```
SELECT tablespace_name , bytes /1024/1024 MB
FROM dba_free_space
WHERE tablespace_name ='TBS1';

SELECT tablespace_name , bytes /1024/1024 MB
FROM dba_free_space
WHERE tablespace_name ='TBS2';
```

Explanation : The "dba_free_space" view in a database to retrieve information about the available free space in tablespaces TBS1 and TBS2.

The first statement retrieves the name of the tablespace and the available free space in megabytes for tablespace TBS1. It does so by querying the "dba_free_space" view, which is a system-defined view that contains information about the space usage of all tablespaces in a database. The statement selects the "tablespace_name" and "bytes" columns from the view, and then divides the value of the "bytes" column by 1024 twice to convert it to megabytes.

The second statement is similar to the first one and retrieves the name of the tablespace and the available free space in megabytes for tablespace TBS2.

Task - 7 : Extend tbs1 by adding extra datafiles.

```
ALTER TABLESPACE TBS1
ADD DATAFILE 'tbs1_extra1.dbf' SIZE 5M;
```

Task - 8 : Extend tbs2 by resizing datafiles.

```
ALTER DATABASE
DATAFILE 'tbs2_data.dbf' RESIZE 10m;
```

Task - 9 : Check the size of the tablespaces.

```
SELECT tablespace_name , bytes /1024/1024 MB
FROM dba_data_files
WHERE tablespace_name ='TBS1';

SELECT tablespace_name , bytes /1024/1024 MB
FROM dba_data_files
WHERE tablespace_name ='TBS2';
```

Task - 10 : Delete table space tbs1 including the datafiles.

```
DROP TABLESPACE tbs1 INCLUDING CONTENTS AND DATAFILES;
```

Task - 11 : Delete table space tbs2 excluding the datafiles.

```
DROP TABLESPACE tbs2 INCLUDING CONTENTS;
```