|  |  |  |
| --- | --- | --- |
| **Ex.No. 5** | **Data Control Language and Transaction Control Language** | **Date :** |

**DATA CONTROL LANGUAGE**

It is used to grant or revoke access permissions from any database user.

**1. GRANT COMMAND**

GRANT command gives user's access privileges to the database.

This command allows specified users to perform specific tasks.

Syntax:  
GRANT <privilege list>  
ON <relation name or view name>  
TO <user/role list>;

#### Example : GRANT Command

**GRANT ALL ON employee  
 TO ABC;  
 [WITH GRANT OPTION]**

**In the above example, user 'ABC' has been given permission to view and modify the records in the 'employee' table.**

**To access other user’s table (It has to be granted before)**

**Select \* from user\_name.table\_name;**

**2. REVOKE COMMAND**

REVOKE command is used to cancel previously granted or denied permissions.

This command withdraw access privileges given with the GRANT command.

It takes back permissions from user.

Syntax:  
REVOKE <privilege list>  
ON <relation name or view name>  
FROM <user name>;

**Questions**

**Q1) Give grant permission to your neighbor for any one of your tables. Tell him/her to access (modify the data) your table from their login.**

**Q2) Check the table again from your login. Observe the inference.**

**Q3) Revoke the permission and tell them to try for accessing your table.**

**DATABASE TRANSACTIONS**

* Begin when the first executable SQL statement is executed
* End with one of the following events:
  + COMMIT or ROLLBACK
  + DDL or DCL statement executes (automatic commit)
  + User exits
  + System crashes

**TRANSACTION CONTROL LANGUAGE**

* TCL Statements are COMMIT, ROLLBACK & SAVE POINT.
* Commit

 Commit command is used to save all the transactions to the database.

**Syntax**

**COMMIT;**

* State of Data Before COMMIT or ROLLBACK
* The previous state of the data can be recovered.
* The current user can review the results of the DML operations by using the SELECT statement.
* Other users *cannot* view the results of the DML statements by the current user.
* The affected rows are *locked*; other users cannot change the data within the affected rows.
* State of Data After COMMIT
* Data changes are made permanent in the database.
* The previous state of the data is permanently lost.
* All users can view the results.
* Locks on the affected rows are released; those rows are available for other users to manipulate.
* All savepoints are erased.
* State of Data After ROLLBACK
* Discard all pending changes by using the ROLLBACK statement.
* Data changes are undone.
* Previous state of the data is restored.
* Locks on the affected rows are released.

**Syntax**

**Rollback;**

**Q1) Create the department table add the details and commit the data.**

**Data for DEPT table**

|  |  |  |
| --- | --- | --- |
| DEPTNO | DNAME | LOC |
| 10 | ACCOUNTING | NEW YORK |
| 20 | RESEARCH | DALLAS |
| 30 | SALES | CHICAGO |
| 40 | OPERATIONS | BOSTON |
| 50 | MANUFACTURING | BOSTON |

**Q2) Update the location of dept number ‘40’ as ‘**San Francisco’ and don’t commit the table.

# SQL> SQL>

**Q3) Rollback to the previous state of data in the deparment table and specify the output of the table with select query.**

**SQL>**

Q4) Delete all the ENTRIES from department table from the location CHICAGO.

# SQL>

* **Rolling Back to a Marker**
* **Create a marker within a current transaction by using**

**SAVEPOINT savepoint-name**

* **Roll back to that marker by using <ROLLBACK TO savepoint-name>**

**ROLLBACK TO [SAVEPOINT] savepoint\_name**

## Parameters

|  |  |
| --- | --- |
| **savepoint\_name** | **Specifies the name of the savepoint to roll back to.** |

**Q5)** Do the following operations one after another

Change LOC=’BOSTON’ for deptno=40 in DEPT table

# SQL>

# Q6) Create SAVEPOINT in the name ‘update\_over’

# SQL>

# Q7) Insert another row in DEPT table with your own values

# SQL>

# Q8) Display the updated data as of now in the department table.

# Q9) Create SAVEPOINT in the name ‘update\_another’

# SQL>

# Q10) Display the data. Then Rollback the transaction upto the point ‘update\_over’ and display the details of table and write the inference.

# SQL>