

Lab 11: Last of SQL

Objective

The students should be able to understand the usage of:

- DCL
- Recall With clause.
- Advanced PL/SQL

Submission Requirements

Save your SQL script file and upload it to LMS.

DCL queries

Write and execute DCL queries to strengthen understanding of database administration. The following will be executed on command line and you may save your queries as comments in the solution sql file.

1. Connect to SQLplus as sysdba.
2. `alter session set "_ORACLE_SCRIPT"=true;`
3. Create a new user **lab11user** and set a password as you like.
4. Alter password for this user and set it to a different one using *Alter user* command.
5. Suppose the system has locked the account due to many failed attempts made to connect to lab11user. In a single statement reset the password and unlock the account:
`ALTER USER your_username IDENTIFIED BY new_password ACCOUNT UNLOCK;`
6. View all users from the dba_users view: `SELECT username FROM dba_users;`
7. `conn lab11user;` What happens here?
8. Connect to sysdba again. There are 2 ways to do this:
 - a. `Conn sysdba` (This will prompt for your sys password)
 - b. `Exit` followed by `sqlplus / as sysdba`
9. Grant some privileges to lab11user.
 - a. Create session (This will allow you to connect in a session)
 - b. Create table
10. Create a role
`Create role Manager;`
11. Grant privileges to a role
`Grant Create table to Manager`
12. Grant role to users
`Grant Manager to lab11user;`
13. If your command line is in poor format then set the conditions in SQL before executing (14).


```
SET LINESIZE 200
SET PAGESIZE 100
COLUMN OWNER FORMAT A20
COLUMN TABLE_NAME FORMAT A30
COLUMN GRANTOR FORMAT A20
COLUMN GRANTEE FORMAT A20
```
14. Verify the system privileges:
`SELECT * FROM DBA_ROLE_PRIVS WHERE GRANTEE = 'LAB11USER';`
15. Create a students table with id and name using suitable data types.
16. Insert 5 rows and commit.
17. Grant an object privilege. Allow lab11user to select on students.
18. Verify if the grant has been given.
`SELECT * FROM DBA_TAB_PRIVS WHERE GRANTEE = 'LAB11USER';`
19. Grant lab11user all privileges to this table and commit. Verify as done in (18).
`grant all on students to lab11user;`
20. Revoke delete access on students to lab11user and commit. Verify as done in (18)
21. Connect to lab11user and try to delete. Can you delete a record? Note that here you need to specify sys as prefix since the table is created by SYS user hence, `sys.students`.

You may continue the remainder of the lab on LiveSQL or SQLDeveloper.

Packages

22. Refer to Lab08,

Create a package named emp_manager to group together the following procedures:

- Update_emp_commission (lab08, Q12)
- Delete_employee (lab08, Q13)

Run the update_emp_commission procedure through the package.

Recall With Clause for Subqueries

Use WITH clause to declare your subqueries.

23. Find department names whose total salary is greater than the average organization salary.

Note that the average organization salary is the average of all department totals.

24. Find the employee who has the highest salary amongst all employees.

Railway Reservation System Scenario

Brief: The railway reservation system aims to provide passengers with the ability to inquire about available trains based on source and destination, book or cancel tickets, and check the status of booked tickets.

Description:

Passengers can book their tickets for the train if seats are available. For this, the passenger has to provide the desired train number and the date for which ticket is to be booked. Before booking a ticket for a passenger, the validity of the train number and booking date is checked. Once the train number and booking date are validated, it is checked whether the seat is available. If yes, the ticket is booked with confirmed status and corresponding ticket ID is generated which is stored along with other details of the passenger. After all the available tickets are booked, a certain number of tickets are booked with waiting status. If the waiting lot is also exhausted, then tickets are not booked and a message of non-availability of seats is displayed.

The ticket once booked can be cancelled at any time. For this, the passenger has to provide the ticket ID (the unique key). The ticket ID is searched, and the corresponding record is deleted. With this, the first ticket with waiting status also gets updated to the confirmed status.

This case study is based on a small version of the reservation system focusing on a small schema. Some assumptions to take into account:

- System has 5 trains.
- The booking is open only for the next seven days from the current date.
- Only two categories of tickets can be booked, namely, AC and General.
- The total number of tickets that can be booked in each category is 10.
- The total number of tickets that can be given the status of waiting is 2 for each category.
- The in-between stoppage stations and their bookings are not considered.

Create the following tables and procedure to implement the railway reservation scenario:

1. **TrainList table:** This table consists of details about all the available trains. The information stored in this table includes train number, train name, source, destination, fare for AC ticket, fare for general ticket, and weekdays on which train is available.

Constraints: The train number is the primary key.

2. **Train_Status table:** This table consists of details about the dates on which ticket can be booked for a train and the status of the availability of tickets. The information stored in this table includes train number, train date, total number of AC seats, total number of general seats, number of AC seats booked, and number of general seats booked.

Constraints:

- a. Train number should exist in TrainList table i.e. ensure referential integrity.
- b. Train number and train date forms the composite primary key.

3. **Passenger table:** This table consists of details about the booked tickets. The information stored in this table includes ticket ID, train number, date for which ticket is booked, name, age, sex and address of the passenger, status of reservation (either confirmed or waiting), and category for which ticket is booked (AC or General).

Constraints:

- a. Ticket ID is the primary key
- b. Train number should exist in TrainList table.

4. **Triggers:** Create triggers for trainList and Passenger tables that would automatically increment the primary key.

- a. The train number begins with 500, 510, 520, 530 and so on.
- b. The ticket ID starts with 10001 and is incremented by 1.

5. **Booking Procedure:** In this procedure, the train number, train date, and category is provided by the passenger. Based on the input, the corresponding record is retrieved from the Train_Status table. If the desired category is AC, then total number of AC seats and number of booked AC seats are compared to find whether ticket can be booked or not. Similarly, it can be checked for the general category. If a ticket can be booked, then passenger details are read and stored in the Passenger table and consequently the booked seat counts are modified. If the train is fully booked, 2 tickets can gain a waiting status. Your procedure should check for the waiting seat count for the particular train number, train date and category. If it is less than 2, then the ticket may get a waiting status. If booking or waiting status is not possible, a suitable message is printed.

6. **Cancel Procedure:** In this procedure, ticket ID is read from the passenger and corresponding record is searched in the Passenger table. If the record exists, it is deleted from the table. After successfully deleting the record, the first record with waiting status for the same train and same category are searched from the Passenger table and its status is changed to confirmed.

Hint: you can access the first row by using ROW_NUM=1 in the where clause of your update query, as the table is already ordered by ticket ID which is the primary key of the table.

7. **Testing:** Insert some dummy data and test out your triggers and procedures.