**Database Security and Reliability**

**TASK:**

**Specifications:**

The final project is meant to be comprehensive. It requires you to pull all your knowledge together to implement database security. **Please note that the specifications for the final project may be different from your previous projects.**

The following steps list the specifications for the final project:

1. Create the following database users with the parameters specified in the table.

|  |  |  |
| --- | --- | --- |
| User | Password | Other Parameters |
| DBSEC\_ADMIN | tec5363admin | Tablespace: USERS  Temporary tablespace: TEMP |
| DBSEC\_CLERK | tec5363clerk | same as above |
| DBSEC\_DEV | tec5363dev | same as above |
| DBSEC | tec5363#1 | same as above |
| VPD\_CLERK1 | jessie$22 | same as above |
| VPD\_CLERK2 | lassie$46 | same as above |

1. Drop CUSTOMERS table. In Week 05 -> Learning Materials, you have downloaded TEC5363\_Student\_Resource\_Files.zip. Utilize Chapter 04 code to create SUPPLIERS table (You need to add three new columns: UPD\_TIME, UPD\_USER, and UPD\_STAT). The data file is attached to this project. Your SUPPLIERS table should have about 214 records. The DBSEC user is the owner of the SUPPLIERS table, which has the following columns.

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| SUPPLIER\_ID | NUMBER(4) NOT NULL |
| SUPPLIER\_NAME | VARCHAR2(80) |
| CONTACT\_NAME | VARCHAR2(30) |
| URL | VARCHAR2(255) |
| PHONE | VARCHAR2(15) |
| ADDR\_LINE | VARCHAR2(80) |
| CITY | VARCHAR2(30) |
| STATE | VARCHAR2(30) |
| ZIP\_CODE | VARCHAR2(9) |
| STATUS | CHAR(1) |
| PERFERRED | CHAR(1) |
| COMMENTS | VARCHAR2(1024) |
| UPD\_TIME | DATE |
| UPD\_USER | VARCHAR2(30) |
| UPD\_STAT | CHAR(1) |

Note that your previous projects are based on CUSTOMERS table. In the final project, you will be working on SUPPLIERS table. You need to change your script accordingly.

1. Create password complexity policy function (named "password\_complexity"). The password policy will enforce the following complexity so that the password:

* + Is at least eight (8) characters long
  + Differs from the user name

Note that to simplify the final project, you only need the above two rules. You will lose points if your script includes not required rules.

1. Create profiles and manage all database users using the profiles including enforcing the password complexity.

You will create the following profiles:

|  |  |  |
| --- | --- | --- |
| **Profile** | **Resources** | **Password** |
| DBSEC\_ADMIN\_PR  OF | SESSIONS\_PER\_USER=5  CONNECT\_TIME=8 hours  IDLE\_TIME=1 hour | PASSWORD\_LIFE\_TIME= 1 month  PASSWORD\_GRACE\_TIME=7 days PASSWORD\_VERIFY\_FUNCTION=  password\_complexity |
| DBSEC\_DEV\_PROF | CONNECT\_TIME=12 hours  IDLE\_TIME=2 hours  CPU\_PER\_CALL=1 minute | PASSWORD\_LIFE\_TIME= 1 month  PASSWORD\_GRACE\_TIME=14 days PASSWORD\_VERIFY\_FUNCTION=  password\_complexity |
| DBSEC\_CLERK\_PR  OF | SESSIONS\_PER\_USER=1  CPU\_PER\_CALL=5 seconds  CONNECT\_TIME=8 hours  IDLE\_TIME=30 minutes  LOGICAL\_READS\_PER\_CA  LL=10 KB | FAILED\_LOGIN\_ATTEMPTS=3  PASSWORD\_LIFE\_TIME= 1 month  PASSWORD\_LOCK\_TIME=3 days  PASSWORD\_GRACE\_TIME=14 days PASSWORD\_VERIFY\_FUNCTION=  password\_complexity |

You will create the following roles:

|  |  |
| --- | --- |
| **Role Name** | **Privileges** |
| DBSEC\_ADMIN\_ROLE | SELECT and ALTER on all DBSEC tables |
| DBSEC\_CLERK\_ROLE | SELECT, INSERT, and UPDATE on all DBSEC tables |
| DBSEC\_SUPERVISOR\_ROLE | SELECT, INSERT, UPDATE and DELETE on all DBSEC tables |
| DBSEC\_QUERY\_ROLE | SELECT only on SUPPLIERS table owned by DBSEC |

You will assign roles and profiles to database users as follows:

|  |  |  |
| --- | --- | --- |
| **User Name** | **Role** | **Profile** |
| DBSEC\_ADMIN | DBSEC\_ADMIN\_ROLE | DBSEC\_ADMIN\_PROF |
| DBSEC\_CLERK,  VPD\_CLERK1,  VPD\_CLERK2 | DBSEC\_CLERK\_ROLE | DBSEC\_CLERK\_PROF |
| DBSEC\_DEV | DBSEC\_ADMIN\_ROLE plus DBSEC\_SUPERVISOR\_ROLE | DBSEC\_DEV\_PROF |

1. Implement view on SUPPLIERS table, similar to Project 7. You will create a VIEW named as SUPPLIER\_VIEW to display only rows that belong to the logged on user.

Connect to VPD\_CLERK2, insert a new row to SUPPLIER\_VIEW. Use your own name for the CONTACT\_NAME. Make up other values. You will enable VPD\_CLERK2 to access the SUPPLIERS data through the view.

Verify your data insertion by query the VIEW. You (as VPD\_CLERK2) should only see the rows that you have inserted. This proves the success of your VIEW implementation.

1. Implement virtual private database (VPD) on SUPPLIERS table so that only the owner of data can access their own row.

First, you will create a policy function, named "DBSEC\_ROW\_OWNER" so that only the data that belong to the current user will be accessed. Then, you will add the policy using DBMS\_RLS.ADD\_POLICY function.

Connect to VPD\_CLERK2, query the "SUPPLIERS" table to verify your policies.

1. Audit the activities on SUPPLIERS table.

You will design and implement the following auditing functions on the SUPPLIERS table.

* + Track all "SELECT" activities on the SUPPLIERS table, including database user, operating system user, and time when the operation is performed.
  + Track the changes when STATUS is set to 'A'.
  + Track the SUPPLIERS table when a supplier record was deleted.

You need to create SQL script to perform the following tasks: o Add a policy to implement audit task 1 (SELECT). The policy will be named as AUDIT\_POLICY\_1\_SELECT.

* + Add a policy to implement audit task 2 (STATUS). The policy will be named as

AUDIT\_POLICY\_2\_STATUS.

* + Add a policy to implement audit task 3 (DELETE on SUPPLIERS table). The policy will be named as AUDIT\_POLICY\_3\_DELETE.
  + Turn on the audit function. o Connect to user DBSEC, and perform the following DML functions for testing purpose:

# SELECT SUPPLIER\_ID, SUPPLIER\_NAME, CONTACT\_NAME

|  |
| --- |
| FROM SUPPLIERS  WHERE SUPPLIER\_ID = 112; |
| UPDATE SUPPLIERS SET  STATUS = 'A'  WHERE SUPPLIER\_ID = 112; |
| DELETE from SUPPLIERS WHERE SUPPLIER\_ID = 112; |

* Show the audit trail including when and who has used/modified the SUPPLIERS table.
* Roll back the above changes.

8. Audit the user activities of two users, VPD\_CLERK1 and DBSEC\_CLERK, similar to Project 10. You will set up an audit mechanism to monitor all activities by the two database users, including preventing them from changing in SUPPLIERS table, revoking CREATE TABLE privilege, and setting up the audit. Write script to test your setting and show audit result

Directly revoke CREATE TABLE privilege from users VPD\_CLERK1 and DBSEC\_CLERK, not using a trigger.

To be safe, drop "trigger\_before\_change" first.

Use a trigger called "trigger\_before\_change" to prevent VPD\_CLERK1 and

DBSEC\_CLERK from changing in SUPPLIERS table, such as insert, update, and delete.

Test CREATE TABLE privilege for VPD\_CLERK1, and test the trigger for both users. For example, each user can try to insert a new record to SUPPLIERS table. Insert is safer than update and delete, because the table probably doesn't have the record that you want to update or delete.