

Lab 7 Submittal

Project one:

1)

The SQL statement given, known as the DCL (Data Control Language) statement, allows Kay, Luke, and Jim to each have the SELECT permission on the "employees" table.

What this statement does is as follows:

1. Giving SELECT Privilege: To provide particular permissions on a database object, in this example the "employees" table, use the phrase "GRANT". The right to run SELECT queries on the designated table is being provided here.

2. Specified Users: Following the "TO" keyword, Kay, Luke, and Jim are stated specifically. The people to whom the permission has been granted are these users. They will now be able to access the "employees" table and obtain data (using SELECT queries).

This statement basically says that Kay, Luke, and Jim can read (retrieve) data from the "employees" database, but they can't necessarily edit or remove records unless they are specifically given permission to do so.

2)

A statement in Data Control Language is "GRANT INSERT, UPDATE ON locations TO jack" in SQL. It expressly gives user Jack the ability to update, which modifies already-existing entries, and insert new records into the "locations" table in the database. With this authority, Jack may use INSERT statements to add new records to the "locations" database and UPDATE statements to update the records that are already there. In other words, Jack is granted explicit access to add new location data and edit existing entries in the "locations" table, but he is not authorized to carry out additional actions such as removing records or changing the structure of the database.

3)

A Data Control Language (DCL) command called "REVOKE SELECT, INSERT, DELETE ON departments TO Winifred" removes Winifred's access rights to the "departments" table's SELECT, INSERT, and DELETE functions. With this command, Winifred's prior authorization to execute INSERT (add records), DELETE (delete records), and SELECT queries (retrieve data) on the database's "departments" table is expressly removed. Winifred's access and skills to manipulate data in the "departments" table will be essentially limited following this revocation, as she will no longer possess the authorization to carry out these activities on the designated table.

4)

The following SQL instructions would be used to establish a position called "Director of IT" and assign it to the user Jim Spear:

CREATE ROLE "Director of IT";

GRANT "Director of IT" TO "Jim Spear";

First, this SQL script uses the 'CREATE ROLE' command to create a role called "Director of IT". Next, it uses the 'GRANT' statement to assign this newly generated position to the user "Jim Spear," giving Jim the rights related to the "Director of IT" role.

5)

```
CREATE ROLE order_entry_op LOGIN PASSWORD 'whe654';  
GRANT SELECT, INSERT, UPDATE, DELETE ON orders TO order_entry_op;  
GRANT SELECT, INSERT, UPDATE, DELETE ON order_items TO order_entry_op;
```

In this above example:

Create A ROLE order_entry_op LOGIN PASSWORD 'whe654'; establishes the role with the given name and uses the LOGIN PASSWORD clause to provide a password for it.

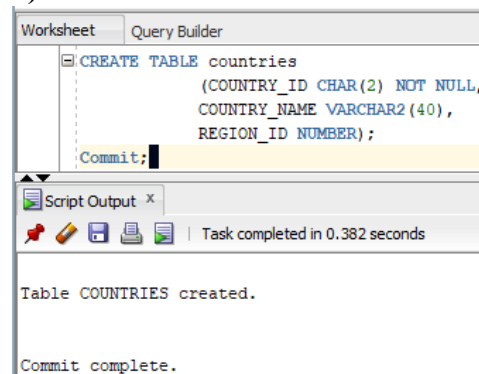
GRANT SELECT, INSERT, UPDATE, DELETE ON orders TO order_entry_op; gives the "order_entry_op" role access to the "orders" table's SELECT, INSERT, UPDATE, and DELETE functions.

GRANT SELECT, INSERT, UPDATE, DELETE ON order_items TO order_entry_op; gives the "order_entry_op" role access to the same rights (SELECT, INSERT, UPDATE, DELETE) on the "order_items" table.

Project Two:

Step 1: Using Rollbacks, Save Points and Commit

1)



2)

```
----- 2 -----
INSERT INTO countries(COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('CA','Canada', 2);

INSERT INTO countries(COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('DE','Germany', 1);

INSERT INTO countries(COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('UK','United Kingdom', 1);

INSERT INTO countries(COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('US','United States of America', 2);

INSERT INTO countries(COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('IN','India', 3);
Commit;
```

Script Output x | Task completed in 0.236 seconds

1 row inserted.

1 row inserted.

1 row inserted.

Commit complete.

3)

```
Commit;
SELECT * FROM countries;
```

Script Output x | Query Result x | All Rows Fetched: 5 in 0.149 seconds

COUNTRY_ID	COUNTRY_NAME	REGION_ID
1 CA	Canada	2
2 DE	Germany	1
3 UK	United Kingdom	1
4 US	United States of America	2
5 IN	India	3

3a)

```
INSERT INTO countries(COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('GR', 'Greece',1);
Commit;
```

Script Output x | Task completed in 0.203 seconds

1 row inserted.

Commit complete.

3b)

```
SAVEPOINT HS_countries_1;
```

Script Output x

Task completed in 0.117 seconds

Savepoint created.

3c & 3d)

```
INSERT INTO countries(COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('MX', 'Mexico', 2);
```

Script Output x

Task completed in 0.078 seconds

1 row inserted.

By adding 'Mexico' with the country code 'MX' and Region ID 2 to the "countries" table, this SQL INSERT statement fills up the corresponding columns.

```
SELECT * FROM countries;
```

Script Output x Query Result x

All Rows Fetched: 7 in 0.059 seconds

	COUNTRY_ID	COUNTRY_NAME	REGION_ID
1	CA	Canada	2
2	DE	Germany	1
3	UK	United Kingdom	1
4	US	United States of America	2
5	IN	India	3
6	GR	Greece	1
7	MX	Mexico	2

Here all the records in the countries table are shown

```
ROLLBACK to HS_countries_1;
```

Script Output x Query Result x

Task completed in 0.088 seconds

Rollback complete.

A database rollback is indicated by the command "ROLLBACK to HS_countries_1;". By rolling back modifications made since the last savepoint, 'HS_countries_1,' it restores the database to that particular savepoint's state. This operation restores data consistency up to the establishment of 'HS_countries_1,' undoing any changes or transactions made after that.

```
SELECT * FROM countries;
```

COUNTRY_ID	COUNTRY_NAME	REGION_ID
1 CA	Canada	2
2 DE	Germany	1
3 UK	United Kingdom	1
4 US	United States of America	2
5 IN	India	3
6 GR	Greece	1

Here all the records are listed again

```
INSERT INTO countries(COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('MX', 'Mexico', 2);
```

1 row inserted.

Here again the record with COUNTRY_ID 'MX' which is Mexico is inserted.

```
Commit;
```

Commit complete.

Commit to save the changes

4)

```
INSERT INTO countries(COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('SP', 'Spain', 1);
```

1 row inserted.

Record inserted

```
SAVEPOINT HS_countries_2;
```

Savepoint created.

Savepoint has been created

```
SELECT * FROM countries;
```

COUNTRY_ID	COUNTRY_NAME	REGION_ID
1 CA	Canada	2
2 DE	Germany	1
3 UK	United Kingdom	1
4 US	United States of America	2
5 IN	India	3
6 GR	Greece	1
7 MX	Mexico	2
8 SP	Spain	1

Listed all the records in countries table

5)

```
INSERT INTO countries (COUNTRY_ID, COUNTRY_NAME, REGION_ID)
VALUES ('NT', 'Netherlands', 1);
```

COUNTRY_ID	COUNTRY_NAME	REGION_ID
1 CA	Canada	2
2 DE	Germany	1
3 UK	United Kingdom	1
4 US	United States of America	2
5 IN	India	3
6 GR	Greece	1
7 MX	Mexico	2
8 SP	Spain	1
9 NT	Netherlands	1

Inserted the Netherlands record

5a)

```
Rollback to HS_countries_2;
```

Script Output x Query Result x

Task completed in 0.081 seconds

Rollback complete.

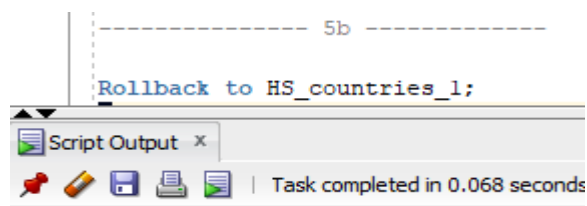
Rollback performed to second savepoint

```
SELECT * FROM countries;
```

COUNTRY_ID	COUNTRY_NAME	REGION_ID
1 CA	Canada	2
2 DE	Germany	1
3 UK	United Kingdom	1
4 US	United States of America	2
5 IN	India	3
6 GR	Greece	1
7 MX	Mexico	2
8 SP	Spain	1

The records after performing rollback

5b)



```
SELECT * FROM countries;
```

Script Output x Query Result x

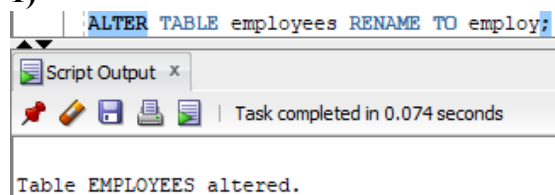
All Rows Fetched: 6 in 0.045 seconds

COUNTRY_ID	COUNTRY_NAME	REGION_ID
1 CA	Canada	2
2 DE	Germany	1
3 UK	United Kingdom	1
4 US	United States of America	2
5 IN	India	3
6 GR	Greece	1

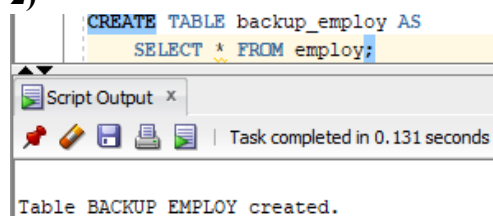
Output after performing the rollback to the first savepoint

Project 3:

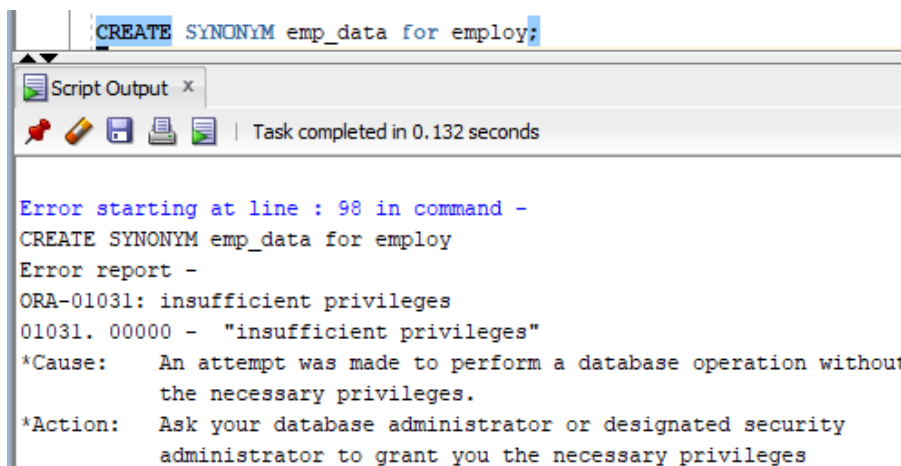
1)



2)



3)



The screenshot shows a SQL script execution window. At the top, the command `CREATE SYNONYM emp_data for employ;` is entered. Below the command bar, a status bar indicates "Task completed in 0.132 seconds". The main area displays an error report starting at line 98. The error is ORA-01031: insufficient privileges. The report details the cause as an attempt to perform a database operation without necessary privileges and suggests asking a database administrator for the required privileges.

```
CREATE SYNONYM emp_data for employ;

Error starting at line : 98 in command -
CREATE SYNONYM emp_data for employ
Error report -
ORA-01031: insufficient privileges
01031. 00000 - "insufficient privileges"
*Cause:      An attempt was made to perform a database operation without
              the necessary privileges.
*Action:     Ask your database administrator or designated security
              administrator to grant you the necessary privileges
```

We cannot create synonym as we don't have the necessary privileges.

4)

Yes, a Virtual Private Network (VPN) may make a big difference in network security. It accomplishes this by establishing an encrypted, safe link across a public network, like the internet. Information is protected from possible interception or eavesdropping by means of encryption during transmission between the user's device and the VPN server.

VPNs offer a number of security advantages:

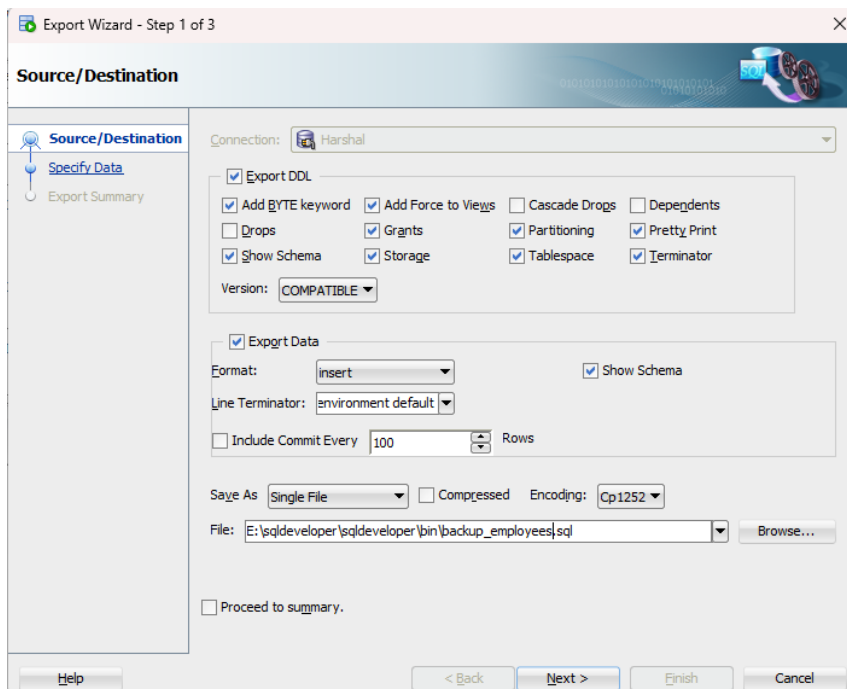
Data Encryption: VPNs protect sensitive data from hackers by encrypting it, rendering it unreadable to unauthorized users.

Anonymity and Privacy: VPNs provide an extra degree of anonymity by hiding IP addresses, which helps to avoid tracking and monitoring online activity.

Secure distant Access: Virtual Private Networks (VPNs) provide safe access to private networks over public connections for enterprises or distant workers, guaranteeing data security and integrity.

VPNs greatly improve security, but they are only one part of an all-encompassing security plan. Ensuring overall network security also requires implementing robust authentication mechanisms and adhering to best practices like software updates.

5)



Export Wizard - Step 1 of 3

Source/Destination

Connection: Harshal

☒ Export DDL

☒ Add BYTE keyword ☒ Add Force to Views ☐ Cascade Drops ☐ Dependents

☐ Drops ☒ Grants ☒ Partitioning ☒ Pretty Print

☒ Show Schema ☒ Storage ☒ Tablespace ☒ Terminator

Version: COMPATIBLE

☒ Export Data

Format: insert ☒ Show Schema

Line Terminator: environment default

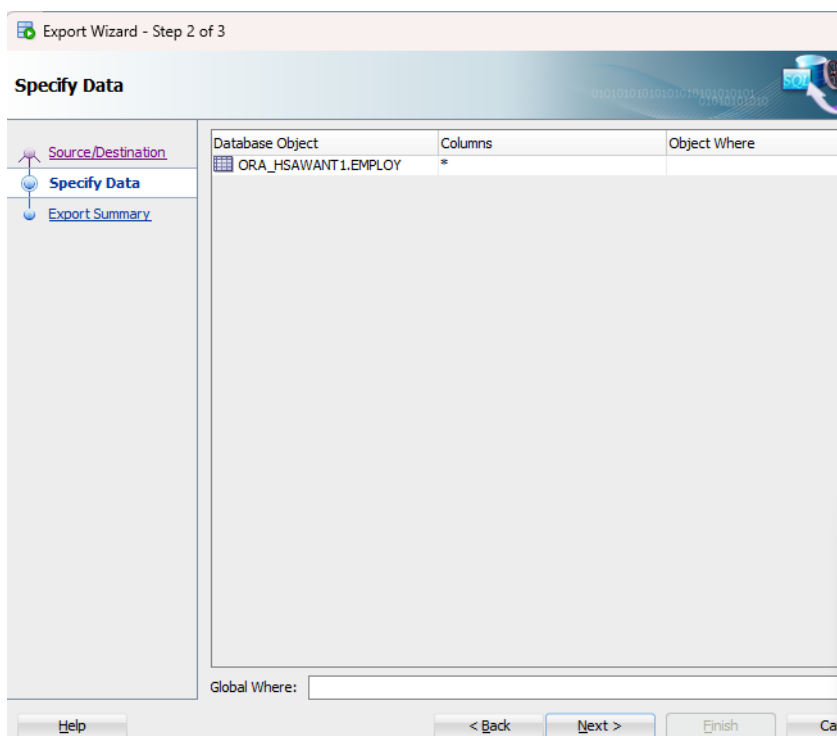
☐ Include Commit Every 100 Rows

Save As: Single File ☐ Compressed Encoding: Cp1252

File: E:\sqldeveloper\sqldeveloper\bin\backup_employees.sql

☐ Proceed to summary.

Step1



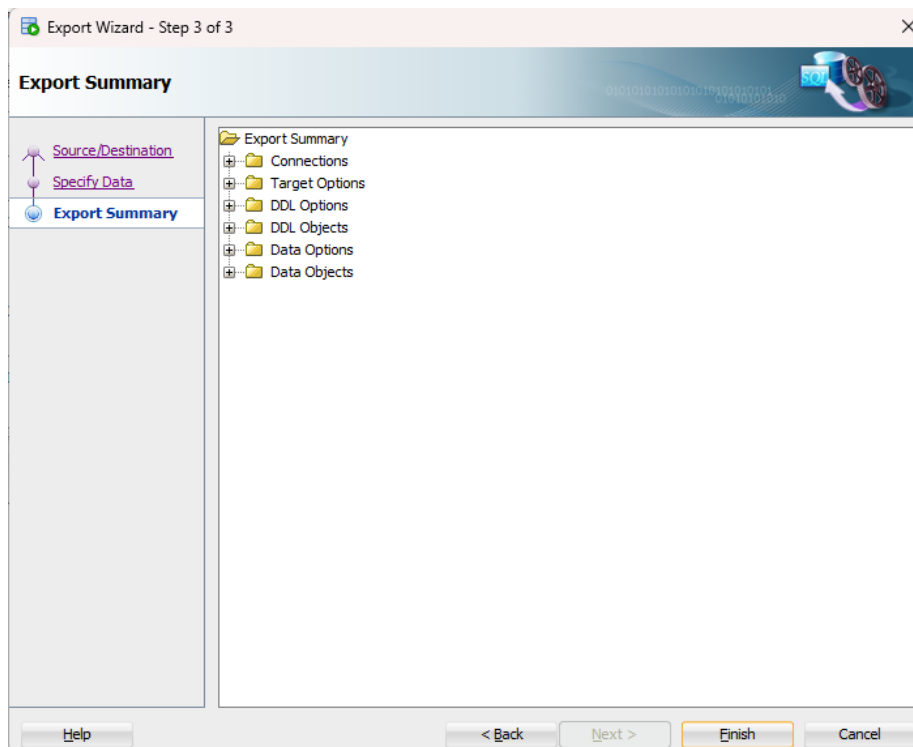
Export Wizard - Step 2 of 3

Specify Data

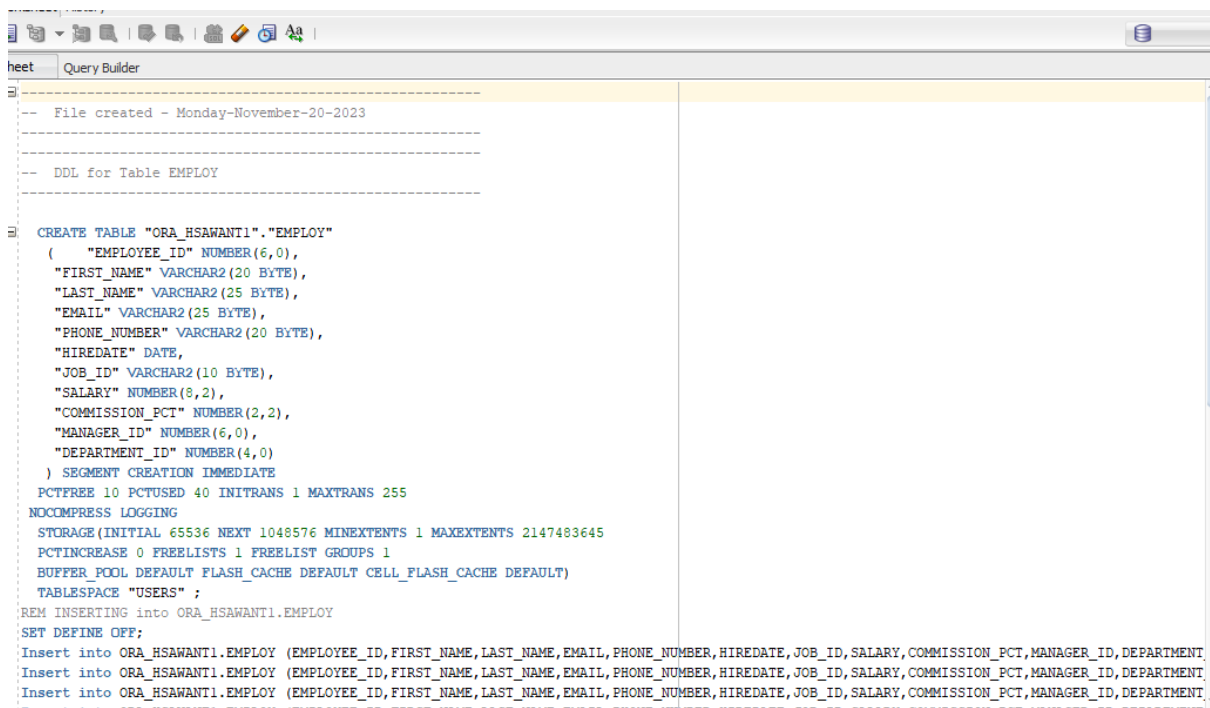
Database Object	Columns	Object Where
ORA_HSAWANT1.EMPLOY	*	

Global Where:

Step2



Step3



The output file created after exporting and finishing the wizard.

6)

Recovery: Following a loss or failure, recovery refers to the procedure of returning data, files, or a system to its prior condition. To restore lost or damaged data, this may entail employing backups, redundant systems, or specialist recovery tools.

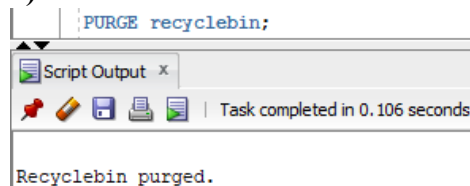
Restore: Restoring data or a system to its initial or a known good condition is the act of doing restoration. In order to restore files, databases, or whole systems to their pre-data loss condition, this frequently entails employing backups or recovery points.

Recycle Bin: Deleted files are stored in the Recycle Bin, a temporary storage space in Windows and other operating systems, until they are permanently removed. It serves as a safety net by enabling users to recover irreversibly lost files if necessary.

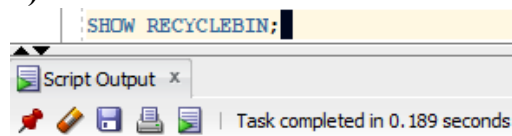
Flashback: A feature that permits point-in-time searches or data recovery is known as such in database management systems such as Oracle. Without really restoring the complete database, it lets users see data as it was at a certain point in the past.

Project 4:

a)

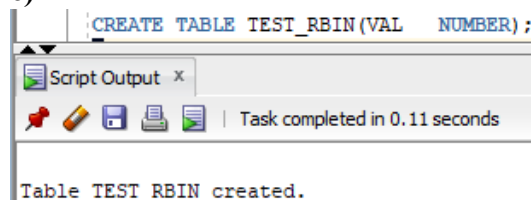


b)



It is empty, as we haven't deleted anything yet.

c)



Create database

```
INSERT INTO TEST_RBIN (VAL) VALUES (10);
```

Script Output x
Task completed in 0.084 seconds

1 row inserted.

Insert record

```
COMMIT;
```

Script Output x
Task completed in 0.074 seconds

Commit complete.

Commit

```
SELECT * FROM TEST_RBIN;
```

Script Output x Query Result x
All Rows Fetched: 1 in 0.055 seconds

	VAL
1	10

Listing the record in that table

```
DROP TABLE TEST_RBIN;
```

Script Output x Query Result x
Task completed in 0.104 seconds

Table TEST RBIN dropped.

Deleting the table

```
SELECT * FROM recyclebin;
```

Script Output x Query Result x
All Rows Fetched: 1 in 0.058 seconds

	OBJECT_NAME	ORIGINAL_NAME	OPERATION	TYPE	TS_NAME	CREATETIME	DROPTIME	DROPSCN	PARTITION_NAME	CAN_UN
1	BIN\$KX0XucJzvRhiTpa2dpf0kMg==50	TEST_RBIN	DROP	TABLE	USERS	2023-11-20:20:46:07	2023-11-20:20:47:37	154575335	(null)	YES

Showing the recycle bin content

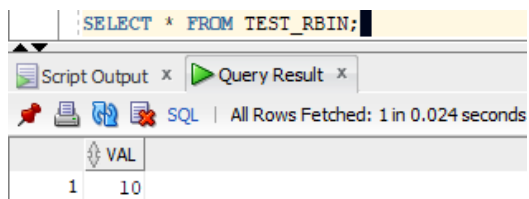
D)

```
FLASHBACK TABLE TEST_RBIN TO BEFORE DROP;
```

Script Output x
Task completed in 0.076 seconds

Flashback succeeded.

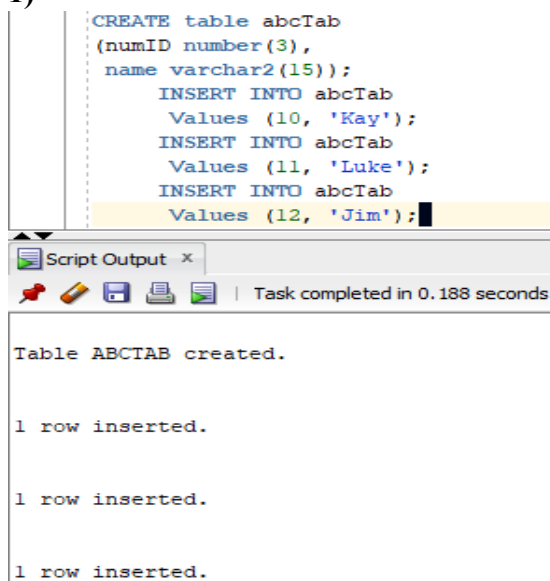
Performing flashback



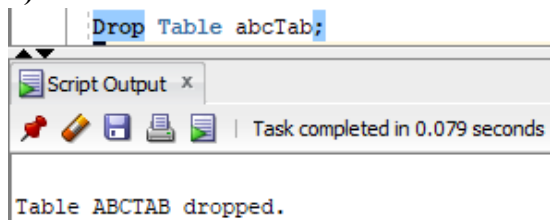
Listing records after performing flashback

E)

1)



2)



Deleting the table again

3)

```
CREATE table abcTab
(numID number(3),
 name varchar2(15));
INSERT INTO abcTab
Values (10, 'Kay');
INSERT INTO abcTab
Values (11, 'Luke');
INSERT INTO abcTab
Values (12, 'Jim');
```

Script Output x | Task completed in 0.252 seconds

Table ABCTAB created.

1 row inserted.

1 row inserted.

1 row inserted.

Populating the table

4)

```
Drop Table abcTab;
```

Script Output x | Task completed in 0.104 seconds

Table ABCTAB dropped.

Deleting the table again

5)

```
SELECT * FROM recyclebin;
```

Script Output x | Query Result x | All Rows Fetched: 2 in 0.048 seconds

OBJECT_NAME	ORIGINAL_NAME	OPERATION	TYPE	TS_NAME	CREATETIME	DROPTIME	DROPSCN	PARTITION_NAME	CAN_UND
1 BINcZxh4JLStRfavr0S15AojiQ==f0	ABCTAB	DROP	TABLE	USERS	2023-11-20:20:53:16	2023-11-20:20:55:26	154575924 (null)		YES
2 BINcP5V6JP2XTh6Q4hjEfShTeg==f0	ABCTAB	DROP	TABLE	USERS	2023-11-20:20:56:34	2023-11-20:20:57:38	154576062 (null)		YES

Recycle bin after performing the above actions

Project five:

Step 1: Using Grants, Privileges, Rollbacks, Save Points and Commits

1)

Here are the suggested steps the DBA or IT director should take:

(a) Tammy Smyth's Retiring:

- ➔ Following her announcement of her retirement from DDD Incorporated after 10 years of service, the DBA/IT Director has to quickly verify all of her access credentials in all systems and databases. Removing her access to sensitive information and vital

systems in accordance with corporate policy is the first thing that has to happen. To guarantee a smooth transition, duties may be moved or reallocated concurrently. It is advisable to take into account the process of backing up or archiving crucial papers or data associated with her job. Additionally, a comprehensive record of the access revocation procedure should be kept for future audits or references.

(b) Immediate Firing of Jennie Sanders:

- ➔ The DBA/IT Director is required to promptly stop all system and database access that was provided to Jennie Sanders upon her immediate termination for suspected employee theft while supervising the retail department at Big Blue Mart. To make sure that no illegal data was retrieved or altered before her firing, a thorough data audit is essential. In addition, if theft is suspected, the DBA could carry out a forensic examination to collect proof of any possible data alteration or theft in order to support more research and security protocols.

(c) Marina Poulos's New role:

- ➔ The DBA/IT Director has to swiftly modify Marina Poulos' access rights as she moves from her present job as a chemist in the research and development facility to a new one creating the operations research program. This entails changing her permissions to access databases, apps, and systems that are relevant to her new roles. Simultaneously, auditing paperwork describing the modifications to her duties and access is necessary. Furthermore, ensuring that Marina has the necessary training and resources for her new position guarantees a seamless transition and top performance in her updated role within the organization.

2)

Several best practices are involved in database security while using an IDE such as SQL Developer in order to protect sensitive data and preserve data integrity. These are important behaviors to think about:

Access Control: Use RBAC to precisely regulate access and set stringent user rights.

Use encryption to prevent unwanted access to data while it's in transit (SSL/TLS) and at rest (TDE).

Regular Updates: To repair security flaws, keep SQL Developer and related tools up to date.

Network security : Network security measures include setting up firewalls, limiting IP access, and using VPNs to provide safe distant connections.

Audit Trails: To trace user activity and identify any security risks, enable and keep an eye on logs.

Parameterized Queries: To stop SQL injection attacks, promote secure coding techniques.

Backups: For data integrity and disaster recovery, regularly create database backups.

Database Hardening: To protect database setups, adhere to platform-specific security requirements.

Training: Inform users about security procedures and make them aware of any dangers.

Notice to Third Parties: When using third-party tools, exercise caution and make sure they come from reliable sources and are updated often to reduce risks.

3)

Role: To make database management easier to administer, roles in database management establish groups of rights or privileges. These are sets of rights that users or other roles may be bestowed with. By enabling administrators to assign or revoke a position, they may simplify access control by granting or rescinding several rights simultaneously. A "Manager" position, for instance, may grant access to particular tables, the ability to carry out particular actions, or the ability to do CRUD operations on data. Roles are allocated to users, making permission management more efficient. (*Ibm Documentation, 2023*)

Profile: Database users' use of resources is governed by their profiles. These are collections of settings that regulate how database resources are used, setting restrictions on things like CPU consumption, session length, and password difficulty for certain users or roles. By limiting excessive consumption and aiding in resource allocation, profiles ensure the equitable and effective use of database resources. (*Sanghi & Haritsa, 2023*)

Privileges: There are certain privileges that are assigned to users or roles in a database system. What users may do with database items is determined by these privileges. SELECT, INSERT, UPDATE, and DELETE, for example, are essential rights that control data manipulation. With respect to permissions, privileges provide you fine-grained control over which items in the database you may see, alter, or work with. (*Asmawi et al., 2008*)

To summarize, privileges offer specific permissions for manipulating database objects, profiles regulate resource utilization, and roles package privileges for simpler management—all of which help to provide strong database security.

4)

There are a lot of hazards involved in granting the PUBLIC role in a database rights. By doing this, all users have access to certain rights, which may result in breaches or illegal access to or alteration of data. This wide access makes security flaws like SQL injection attacks more likely to be exploited by hostile users. Directly granting rights to PUBLIC disregards the least privilege principle, which requires limiting access to the necessary resources. Such acts might reveal private information and breach compliance guidelines, which could have legal ramifications. In order to reduce these risks, it is imperative that you refrain from giving permissions to PUBLIC and instead take a focused approach, allowing permissions only to the roles or users that are particularly needed, and routinely checking access restrictions to make sure they comply with strict security procedures.

5)

Quick Revocation: In order to prevent more illegal access and downloads, the DBA has to quickly remove the PUBLIC role's excessive rights on the "food_club_members" database.

Containment Measures: To lessen the effect of the breach, stop further illegal downloads of the data and stop its distribution.

Investigation and Analysis: To ascertain the scope of the data breach, pinpoint the parties involved, and assess any possible harm from the downloaded data, conduct an exhaustive investigation.

Data Recovery: Make an effort to safeguard or restore the downloaded data in order to guard against abuse and evaluate the compromised information.

Communication and Reporting: Inform the appropriate authorities, interested parties, and impacted parties about the breach and the actions taken to rectify the problem and any possible consequences.

Training and Policy Review: To avoid the over-issuance of privileges, conduct further training on data security measures and review policies, with a strong emphasis on appropriate access control.

Performance Review: Assess Charles's decision-making procedure and implement remedial measures or further training to prevent future occurrences of this kind.

Compliance with Regulations: Ascertain adherence to legal mandates by informing data protection authorities about the breach, if deemed essential, and according to established procedures for handling such situations.

6)

The 3-2-1 rule is a data backup technique that prioritizes data safety and redundancy when it comes to database files. It implies:

1. Three Copies: Keep your database files in at least three copies. This makes sure there are numerous copies of the data for redundancy, including the original data and two more backups.

2. Two Different Media: Keep these copies on two distinct platforms or kinds of storage media. For example, to reduce the possibility of multiple failures of the same media type occurring at the same time, keep one copy on local drives and another on cloud storage.

3. One Offsite Location: Store the database files at a remote or offsite location, with at least one copy. This offers defence against natural calamities that might harm the primary data site, such fires, floods, or other catastrophic occurrences.