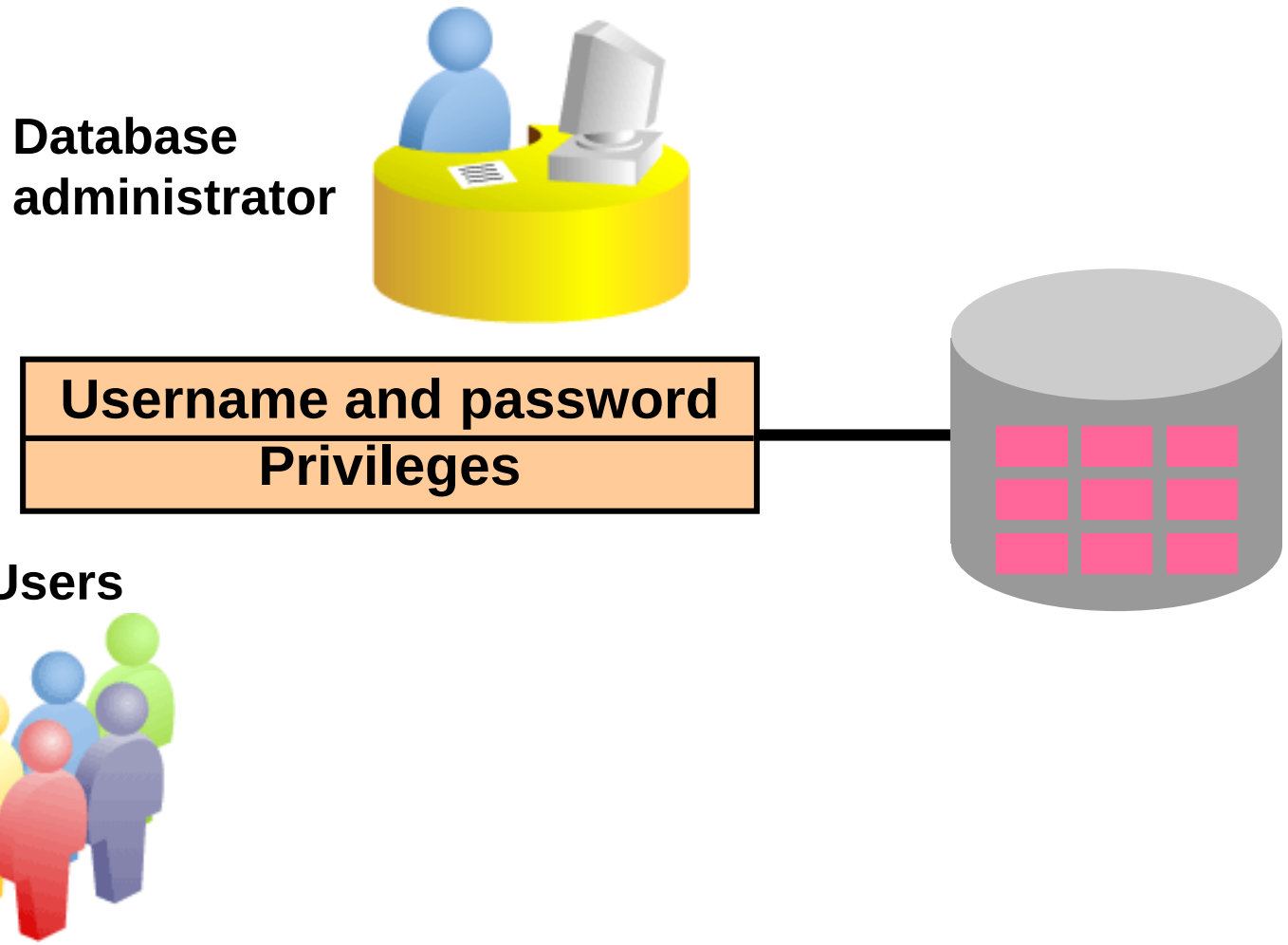
Controlling User Access

Objectives

After completing this lesson, you should be able to do the following:

- Differentiate system privileges from object privileges
- Grant privileges on tables
- View privileges in the data dictionary
- Grant roles
- Distinguish between privileges and roles

Controlling User Access



Privileges

- Database security:
  - System security
  - Data security
- System privileges: Gaining access to the database
- Object privileges: Manipulating the content of the database objects
- Schemas: Collection of objects such as tables, views, and sequences

System Privileges

- More than 100 privileges are available.
- The database administrator has high-level system privileges for tasks such as:
  - Creating new users
  - Removing users
  - Removing tables
  - Backing up tables

Creating Users

The DBA creates users with the `CREATE USER` statement.

```
CREATE USER user
IDENTIFIED BY   password;
```

```
CREATE USER  HR
IDENTIFIED BY   HR;
User created.
```

User System Privileges

- After a user is created, the DBA can grant specific system privileges to that user.

```
GRANT privilege [, privilege...]
TO user [, user| role, PUBLIC...];
```

- An application developer, for example, may have the following system privileges:
  - CREATE SESSION
  - CREATE TABLE
  - CREATE SEQUENCE
  - CREATE VIEW
  - CREATE PROCEDURE

Granting System Privileges

The DBA can grant specific system privileges to a user.

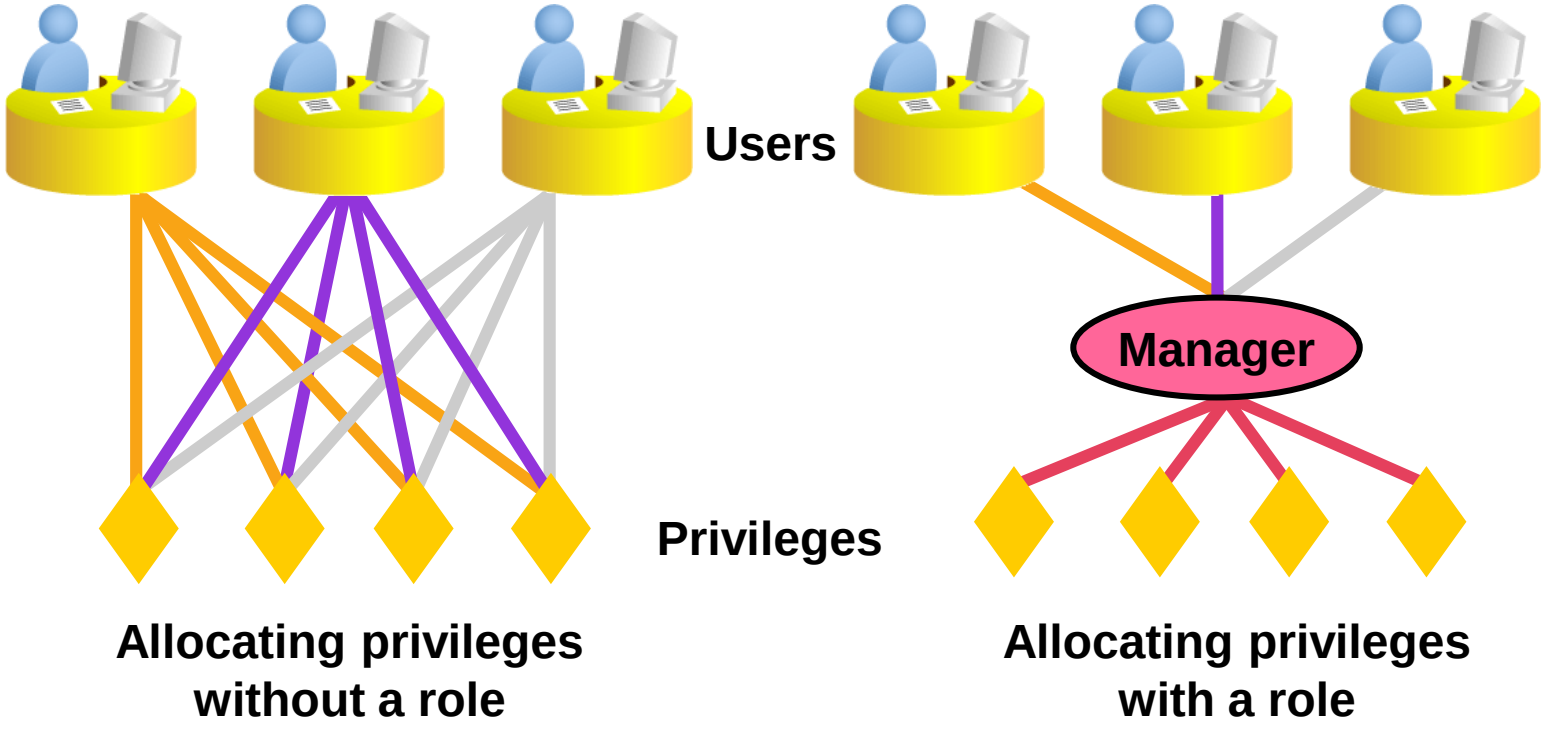
```
GRANT  create session, create table,
       create sequence, create view
TO     hr;
Grant succeeded.
```

Passing On Your Privileges

- **Give a user authority to pass along privileges.**

```
GRANT   create session, create table,
        create sequence, create view
TO      hr
WITH    ADMIN OPTION;
Grant succeeded.
```

What Is a Role?



Creating and Granting Privileges to a Role

- Create a role

```
CREATE ROLE manager;
Role created.
```

- Grant privileges to a role

```
GRANT create table, create view
TO manager;
Grant succeeded.
```

- Grant a role to users

```
GRANT manager TO DE_HAAN, KOCHHAR;
Grant succeeded.
```

Predefined Roles

CONNECT	CREATE SESSION, SET CONTAINER
RESOURCE	CREATE CLUSTER, CREATE INDEXTYPE, CREATE OPERATOR, CREATE PROCEDURE, CREATE SEQUENCE, CREATE TABLE, CREATE TRIGGER, CREATE TYPE
SCHEDULER_ ADMIN	CREATE ANY JOB, CREATE EXTERNAL JOB, CREATE JOB, EXECUTE ANY CLASS, EXECUTE ANY PROGRAM, MANAGE SCHEDULER
DBA	Most system privileges, several other roles. Do not grant to nonadministrators.
SELECT_ CATALOG_ ROLE	No system privileges, but HS_ADMIN_ROLE and over 1,700 object privileges on the data dictionary

Changing Your Password

- The DBA creates your user account and initializes your password.
- You can change your password by using the `ALTER USER` statement.

```
ALTER USER HR
IDENTIFIED BY employ;
User altered.
```

Object Privileges

Object Privilege	Table	View	Sequence	Procedure
ALTER	√		√	
DELETE	√	√		
EXECUTE				√
INDEX	√			
INSERT	√	√		
REFERENCES	√			
SELECT	√	√	√	
UPDATE	√	√		

Object Privileges

- Object privileges vary from object to object.
- An owner has all the privileges on the object.
- An owner can give specific privileges on that owner's object.

```
GRANT       object_priv [(columns)]
ON          object
TO          {user|role|PUBLIC}
[WITH GRANT OPTION];
```

Granting Object Privileges

- Grant query privileges on the `EMPLOYEES` table.

```
GRANT  select
ON     employees
TO     sue, rich;
Grant succeeded.
```

- Grant privileges to update specific columns to users and roles.

```
GRANT  update (department_name, location_id)
ON     departments
TO     scott, manager;
Grant succeeded.
```

Passing On Your Privileges

- Give a user authority to pass along privileges.

```
GRANT  select, insert
ON     departments
TO     scott
WITH   GRANT OPTION;
Grant succeeded.
```

- Allow all users on the system to query data from Alice's `DEPARTMENTS` table.

```
GRANT  select
ON     alice.departments
TO     PUBLIC;
Grant succeeded.
```

Confirming Privileges Granted

Data Dictionary View	Description
ROLE_SYS_PRIVS	System privileges granted to roles
ROLE_TAB_PRIVS	Table privileges granted to roles
USER_ROLE_PRIVS	Roles accessible by the user
USER_TAB_PRIVS_MADE	Object privileges granted on the user's objects
USER_TAB_PRIVS_RECD	Object privileges granted to the user
USER_COL_PRIVS_MADE	Object privileges granted on the columns of the user's objects
USER_COL_PRIVS_RECD	Object privileges granted to the user on specific columns
USER_SYS_PRIVS	System privileges granted to the user

Revoking Object Privileges

- You use the `REVOKE` statement to revoke privileges granted to other users.
- Privileges granted to others through the `WITH GRANT OPTION` clause are also revoked.

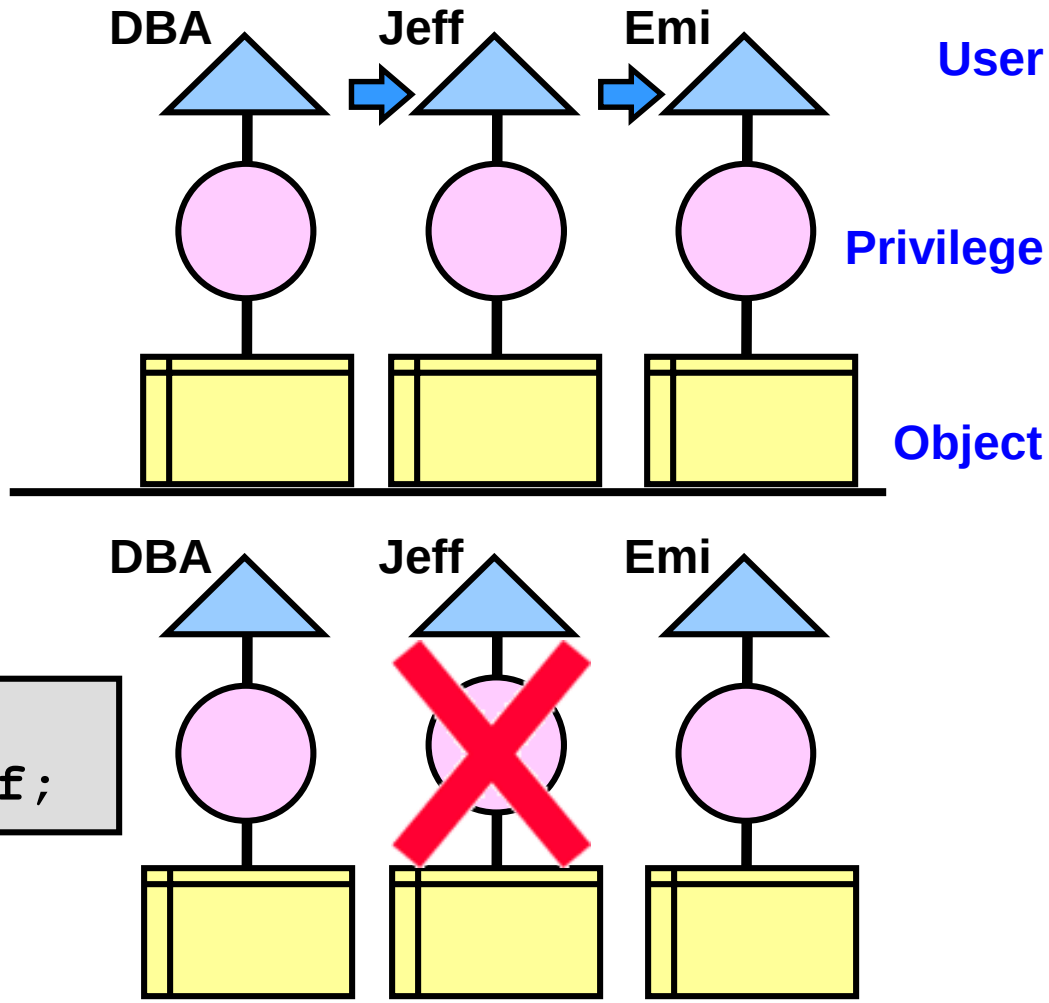
```
REVOKE {privilege [, privilege...]|ALL}
ON     object
FROM   {user[, user...]|role|PUBLIC}
[CASCADE CONSTRAINTS];
```

Revoking Object Privileges

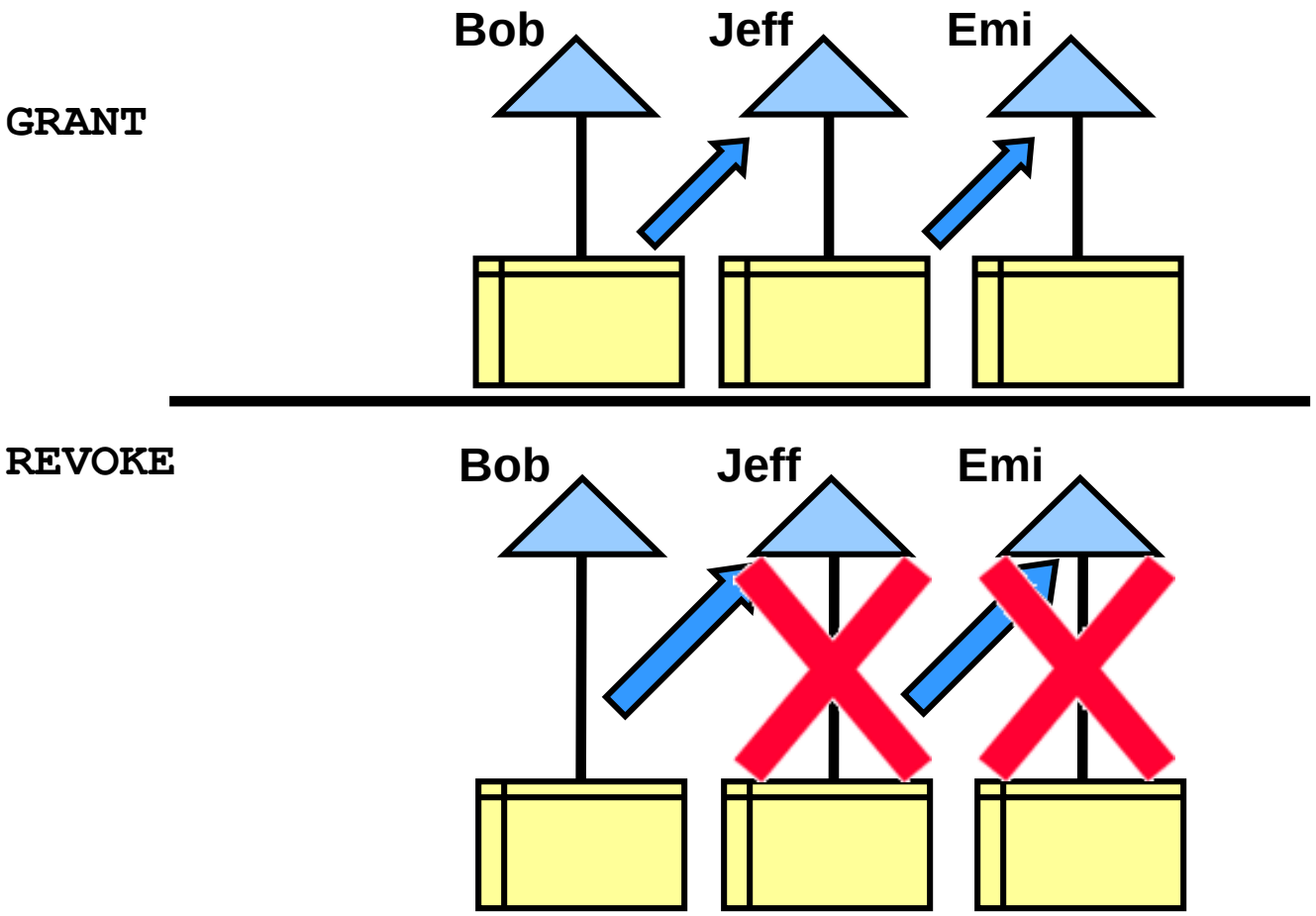
As user Alice, revoke the `SELECT` and `INSERT` privileges given to user Scott on the `DEPARTMENTS` table.

```
REVOKE  select, insert
ON      departments
FROM    scott;
Revoke succeeded.
```

Revoking System Privileges with ADMIN OPTION



Revoking Object Privileges with GRANT OPTION



Secure Roles

- Roles may be protected through authentication.

```
CREATE ROLE secure_application_role
IDENTIFIED BY <password>;
```

- Roles may also be secured programmatically.

```
CREATE ROLE secure_application_role
IDENTIFIED USING <security_procedure_name>;
```

- Creating a role.

```
CREATE ROLE test_role;
Role created.
```

- Creating a role protected by authentication.

```
CREATE ROLE pwd_role
IDENTIFIED BY pwd123;
Role created.
```

Setting default role

- A default role means that the role is always enabled for the current session at logon.
- Setting new roles to user and review

```
GRANT test_role, pwd_role TO myuser;
Grant succeeded.
SELECT * FROM   dba_role_privs;
…
n row(s) selected.
```

- Setting a default role for the none

```
ALTER USER myuser DEFAULT ROLE test_role;
User altered.
SELECT * FROM   dba_role_privs;
n row(s) selected.
```

Setting default role

- A default role means that the role is always enabled for the current session at logon.
- Setting new roles to user

```
GRANT test_role, pwd_role
TO myuser;
Grant succeeded.
```

- Review in metadata

```
SELECT * FROM   dba_role_privs;
…
n row(s) selected.
```

NOTE: user's every role are default

Setting default role

- **Setting no default role to user and review**

```
ALTER USER myuser DEFAULT ROLE none;
User altered.
SELECT * FROM   dba_role_privs;
…
n row(s) selected.
```

- **Setting default role to user and review**

```
ALTER USER myuser DEFAULT ROLE test_role;
User altered.
SELECT * FROM   dba_role_privs;
…
n row(s) selected.
```

Setting default role ALL … EXCEPT

- Setting no default role to user

```
ALTER USER myuser
DEFAULT ROLE ALL EXCEPT pwd_role;
User altered.
```

- Reviewing

```
SELECT *
FROM   dba_role_privs
ORDER BY 1;
…
n row(s) selected.
```

Secure Roles

- Roles may be nondefault.

```
SET ROLE role;
```

- Roles may be nondefault.

```
SET ROLE role_name IDENTIFIED BY password;
```

- Setting role in a session.

```
CONNECT myuser/mypassword
Connected.
SELECT * FROM session_roles;
1 row selected.
```

```
SET ROLE pwd_role IDENTIFIED BY pwd123;
Role set.
```

```
SELECT * FROM session_roles;
1 row selected.
```

Dropping users

- Use the DROP USER statement to remove a database user and optionally remove the user's objects

- Syntax

```
DROP  USER  user [ CASCADE ];
```

- Drop the user and its objects

```
DROP  USER  myuser CASCADE;
User dropped.
```

Dropping roles

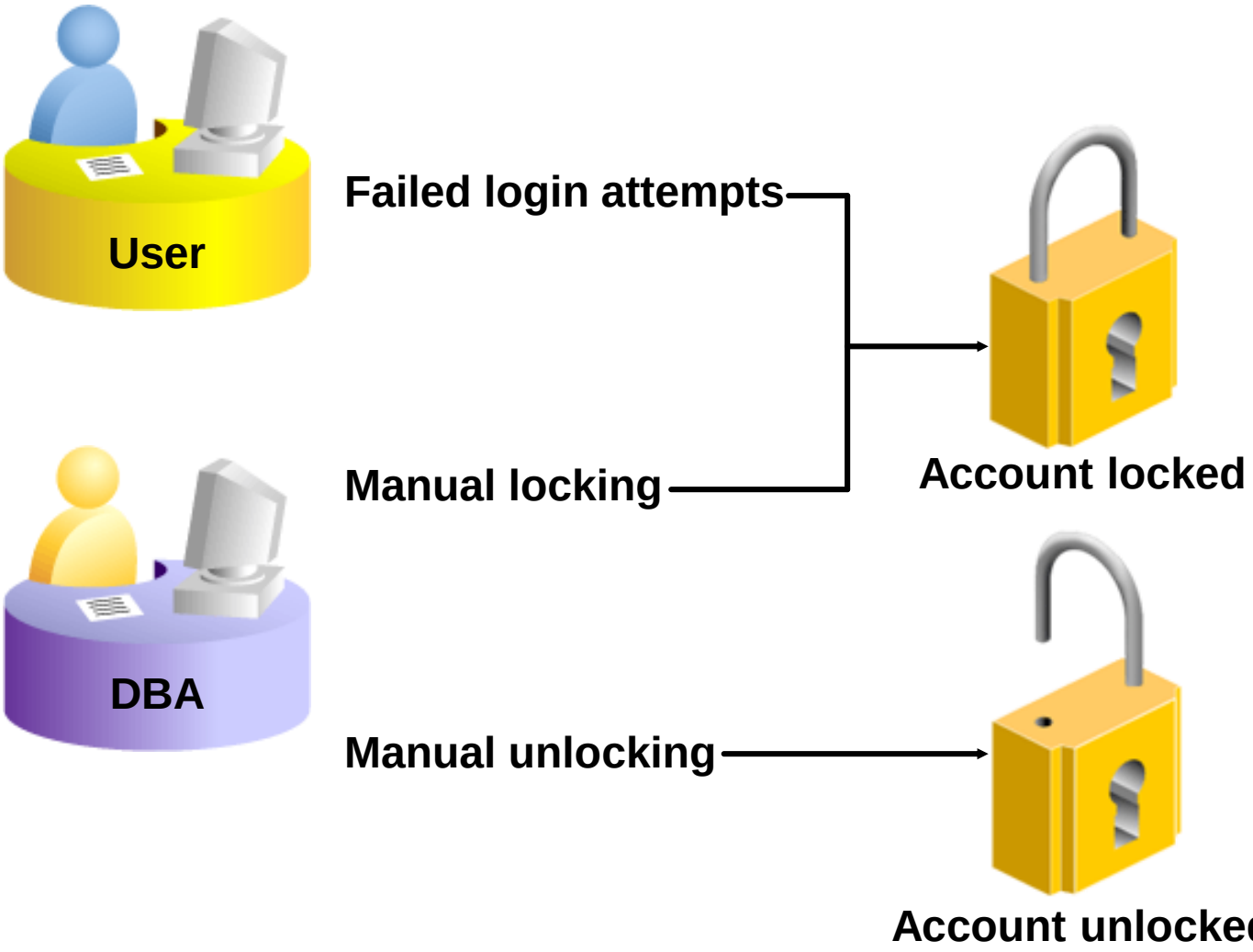
- Once a role has been created in Oracle, you might at some point need to drop the role.
- Syntax

```
DROP ROLE role_name;
```

- Drop the user and its objects

```
DROP ROLE test_role;
Role dropped.
DROP ROLE pwd_role;
Role dropped.
```

Locking and Unlocking Accounts



Locking and Unlocking Accounts

- To temporarily deny access to the database for a particular user, you can lock the user account. You can unlock the user account when you want to allow database access again for that user.
- Syntax

```
ALTER USER username ACCOUNT LOCK;
ALTER USER username ACCOUNT UNLOCK;
```

- Drop the user and its objects

```
ALTER USER hr ACCOUNT UNLOCK;
User altered.
```

How to set tablespace quota to user

- **Check quota of use. Example for HR:**

```
SELECT TABLESPACE_NAME,
BYTES / 1024 / 1024 "UTILIZIED_SPACE",
MAX_BYTES / 1024 / 1024 "QUOTA_ALLOCATED"
FROM dba_ts_quotas
WHERE username = 'HR';
```

- **Set tablespace quota to 10G for user:**

```
ALTER USER HR QUOTA 10G ON USERS;
```

- **Grant unlimited tablespace quota:**

```
ALTER USER HR QUOTA UNLIMITED ON USERS;
```

Summary

In this lesson, you should have learned about statements that control access to the database and database objects.

Statement	Action
`CREATE USER`	Creates a user (usually performed by a DBA)
`GRANT`	Gives other users privileges to access the objects
`CREATE ROLE`	Creates a collection of privileges (usually performed by a DBA)
`ALTER USER`	Changes a user's password and others
`REVOKE`	Removes privileges on an object from users
`SET ROLE`	Set roles in a session
`DROP ROLE`	Drop a role from the database
`DROP USER`	Drop a user from the database

Practice 1: Overview

This practice covers the following topics:

- Granting other users privileges to your table
- Modifying another user's table through the privileges granted to you
- Creating a synonym
- Querying the data dictionary views related to privileges