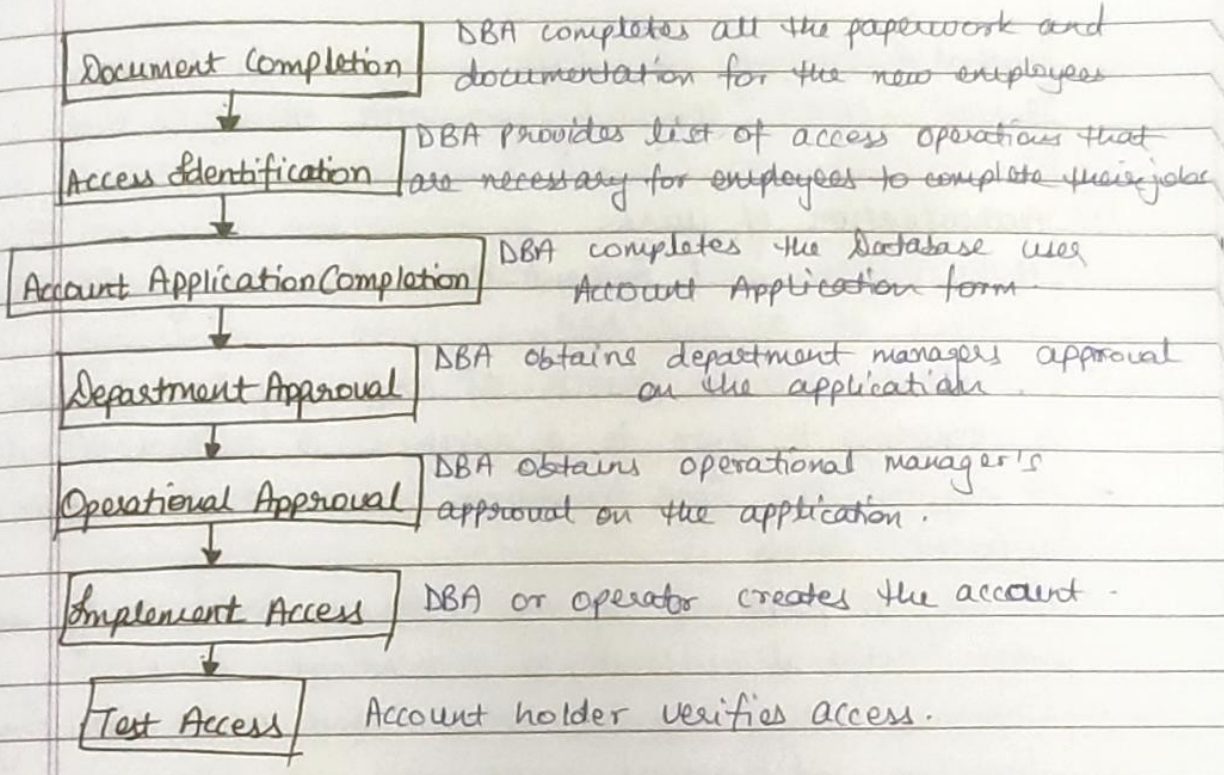
DSP Unit 2

(*) Administration of Users:
→ Authentication and Authorization are essential services for every OS.
→ Another service is Administration of Users.
→ Administrators use the following functionalities:
(a). Creating Users
(b). Setting Password Policies
(c). Grant Privileges

Documentation of User Administration:
→ Documentation is the main part of Administration Process.
→ Many security violations are caused by negligence and ignorance and in particular by failing to consider documentation.
→ Everything is documented for 2 reasons:
(a) To retrace exactly what happened when security breach occurs
(b). To ensure administration consistency.
→ Documentation includes the following:
(a). All policies for handling new and terminated employees, managers, system and database managers, human resources, etc.
(b). A detailed document should describe guidelines for every task that is required for all common administration situations.
(c). Providing user manuals and operational manuals.
→ Documentation provides a full description of all the predefined roles and the tasks for which the role is responsible.
→ It includes a detailed description of administration staff.
→ Administrators have the following responsibilities:
(a). Add New Users
(b). Delete New Users
(c). Manage User Access
(d) Set User connection Privilege
(e). Edit User Permissions
(f). View existing user Permissions
(g). Set Password Policies.



16. Creating Users:

→ One of the main tasks you will perform as a database operator or DBA.

→ In most organizations, this process is standardized & well documented

→ The DBA writes a script to create a user for every developer working on the project.

→ This script grants privileges to read and write data.

→ Creating user is generally an easy task once a policy is documented and followed.

→ Steps to create login in SQL server.

Step 1). In object Explorer, expand the databases folder.
Expand the database in which to create new database user.
Navigate to security → logins → New Login → Enter.

Step 2). In the next screen, enter login Name, Select SQL server Authentication, Enter Password, click Ok.

Step 3). Login is created and can be seen by expanding.

Method 2: Creating a login using T-SQL command:
Syntax: CREATE LOGIN mylogin WITH PASSWORD = '123';

(*). Authentication of users:
→ Authorization and Authentication are one of the primary services of OS and DBA.
→ Authentication is the process of confirming that a user who is attempting to login to a database is authorized to do so.
→ It requires one more dimension because it may happen at different levels.
→ It may be performed by the database itself, or by some other external method, to authenticate users.
→ Databases employ foolproof authentication modes like fingerprint recognition and retinal scans.
→ It can be managed either locally within the database or centrally in a directory service.
→ Primary types of data authentication include:
(a). Local Authentication and local authorization: This method provides simple password method.
* Allows you to quickly create a database and access it.
* It is useful for environments like development.
* Every action has to be managed by the local DBA.
(b). Central authentication and local Authorization:
Used via various 3rd Party Authentication services such as Kerberos, Remote Authentication Dial In User Service (RADIUS), SSL Authorization, etc.
* Kerberos is 3rd party Authentication system that relies on shared sec
* RADIUS is client/server security protocol known for enabling remote authentication and access.
* SSL uses digital certificates.

→ While authentication is centralized, authorization for most part remains locally managed.

(*) Central Authentication and Central Authorization:

(*) Centralized management of users.

(*) Onboarding new users, assigning resources, changing privileges and removing access when they leave.

(*) Centrally Managed Users (CMU): Provides a simpler integration.

(*) SQL Server User:

Creating a user: A user is an account that you can use to access SQL Server.

Method 1: Using SQL Server Management Studio.

Step 1) (*) Expand the database folder from Object Explorer after connecting to SQL Server.

(*) Identify the database for which you want to create User.

(*) Expand the database then Expand the Security folder.

(*) Right Click the Users folder then Choose "New User"

Step 2) Enter User details. (User Name, Login Name, etc).
Click OK.

Step 3) User is created.

Method 2: Creating user using T-SQL.

Syntax: Create user <username> for login <login-name>

Eg: create user Guru99 for login Mylogin.

Note: SQL server will throw an error if you create a user for the same login.

(*) Removing, Modifying users:

Method 1: SQL Server: DROP USER statement:

→ The DROP USER statement is used to remove a user from the SQL server database.

→ Syntax: DROP USER <username>;
where username is the name of the user to remove from SQL server database.

→ Before you can drop a user, you must delete the objects owned by the user or transfer the ownership of those objects.

Method 2: Using SQL Server Management Studio:
Step 1) Open SSMS.
Step 2). Connect to SQL server.
Step 3). In object Explorer, go to «Security» node then logins.
Step 4). Right click on the SQL server login that you want to drop and then click on "Delete".
Step 5). SSMS will show a warning message. Click OK.

→ Modify users:

Modifying an ORACLE user:

SQL> ALTER USER SCOTT IDENTIFIED BY LION;
User Altered.

(v). Default Users.
→ The users that are created at the time of software installation.
→ They cannot be removed or deleted.
→ They are used to manage access to the device's resource

(*) Remote Users:

→ Individuals or applications that access a database from a location different from the physical location where the database is hosted.

→ The users connect to the database over a network, typically using client-server architecture

→ Can interact with the DB by sending queries, retrieving data, modifying records, or performing other operations, depending upon access privileges and other permissions granted by DBA.

→ The requests are transmitted over the network to the DBMS server that processes the request and sends back the results.

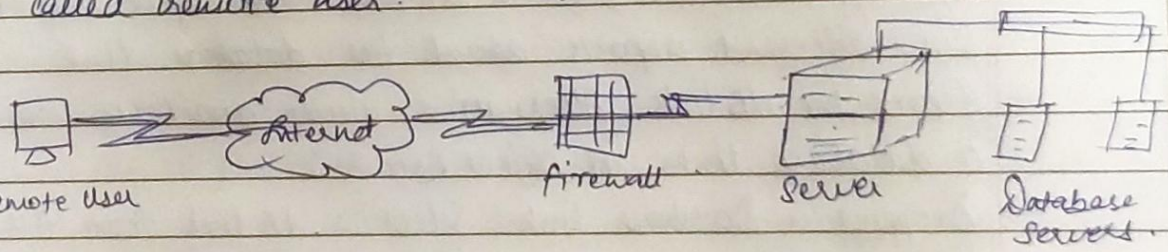
→ Access is commonly facilitated through protocols such as ODBC (Open Database Connectivity), JDBC (Java DB Connectivity, etc.

→ Security methods such as authentication and encryption are usually employed to protect the confidentiality and integrity

→ It allows for distributed and collaborative data management.

→ Promotes flexibility and scalability.

→ When a user or administrator logs on to the database through the local area Network or through internet or VPN, it will be called remote user.



(*) Database Links:

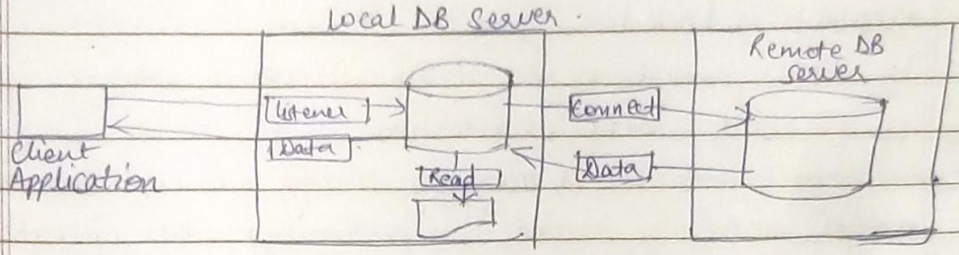
→ Connection from one database to another remote database.

→ Remote database can be an Oracle Database or any ODBC compliant database such as SQL server or MySQL.

→ It allows the user to access database objects such as tables &

views from another database.

→ It enables a user to perform DML or any other valid SQL statements on a database.



→ You can perform DQL and DML via a database link but you cannot perform DDL. (You cannot do CREATE, ALTER TABLE, CREATE/ALTER INDEX) through a DB link.

→ Two types of Database links: ① Public ② Private.

→ Private DB links are visible to owners while public DB links are visible to all users in the DB.

→ Public DB links may pose some potential security risks.

→ Managing DB links: Once you create a DB link, you can access the remote objects by appending @dblink to the table or view name, where dblink is the name of database link.

→ Creating a DB link: Create using Object Browser.

→ Browsing a DB link: Select a DB link from Object selection pane and view different reports about the database link.

→ Reports for DB links: Alternative views available while viewing a database links in Object Browser.

→ Dropping a Database link: Select a DB link from the Object selection pane and click on drop.

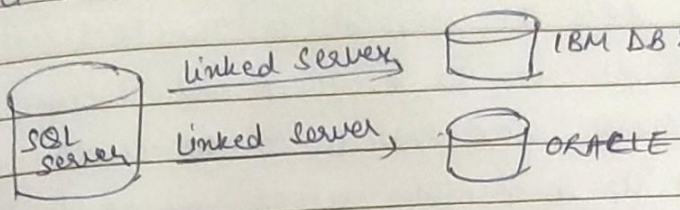
→ These are one-way connections and each link connects a database to only one other database.

→ Across this link one DB sends queries or updates to another database.

(*). Linked Servers:

→ They enable the SQL server Database Engine and Azure SQL Managed Instance to read data from remote data sources and execute commands against remote DB servers outside of the Instance of SQL server.

→ They are configured to enable the DB engine to execute a Transact-SQL Server.



→ They enable you to implement distributed databases that can fetch and update data in other databases.

→ They offer the ability to access data from outside of SQL server

→ Offer the ability to issue distributed queries, updates, commands, and transactions on heterogeneous data sources

→ Offer the ability to address diverse data sources similarly.

→ You can configure a linked server using SQL server Management Studio or by using Transact-SQL statement (sp_addlinkedserver).

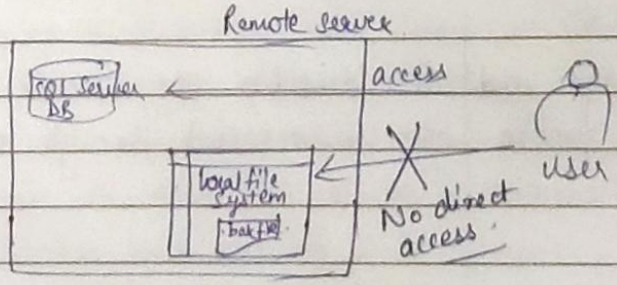
Linked Server components.

OLE DB provider	OLE DB data source.
A DLL that manages and interacts with a specific data source.	It identifies the specific database that can be accessed through OLE DB.

Date:
Page:

(*) Remote Servers:

→ Supported in SQL Server for backward compatibility only.
→ New applications should use linked servers instead.
→ It allows for a client connected to one instance of SQL Server to execute a stored procedure on another instance of SQL server. without establishing a seperate connection.
→ The server to which the client is connected accepts the client request and sends the request to remote server on behalf of client.
→ Remote server processes the request and returns result to original server
→ This server passes those results to the client.
→ When you set up a remote server, you should also consider how to establish security.
→ Both, stored procedures and distributed queries are allowed against linked servers but only stored procedures are allowed against remote servers.
→ These are set up in pairs.
→ We need to configure both the servers to recognize each other as remote servers.
→ The remote access configuration must be set to 1 on both the local and remote computers.
→ From the local server, you can disable the remote server configuration
→ Remote servers provide access to shared data and objects in your organization.



(*). Practices for Administrators and Managers:

① Set business Goals: If you don't spend time deciding what data to collect and how you can use this data effectively, you run the risk of wasting internal resources gathering the wrong data, piling up too much data or missing important data.

→ It gives you lodestar to follow so you don't lose your way.

→ It involves: (a). Creating Profiles and targetting.
(b) Identifying trends and patterns.
(c). Automating and Improving processes.
(d). Informing business decisions.

② Establish policies and procedures, including backup and recovery procedures: → It prepares your team to act effectively if the worst happens.

→ It involves: (a). Collecting and organizing data.
(b). Guard data integrity
(c) Monitor your data.
(d). Set benchmarks.
(e). Map your processes.

③ Make security your priority:

→ Not every disaster is entirely predictable or preventable, so you can improve your data security and manage the risks associated.

→ It involves: (a). Creating a comprehensive maintenance plan.
(b). Develop your backup and recovery procedures.
(c). Build your team's security skills.
(d). Leverage automation to help with security.

④ Focus on the Quality of data: DBA should work to promote a high standard of data quality.

→ Have SMART (Specific, Measurable, Achievable, relevant and

time bound standards.
→ Make sure your DBA have everything necessary to do their jobs successfully.
⑤ Reduce duplicate data. Duplicate data reduces your database performance and can hinder your efforts.
→ They also lead to wasted internal resources and doubled effort by your team.
→ Your entire company must know a few basics about protecting data quality.
→ Teach everyone what good data looks like and how to contribute towards high quality data.
→ Have a plan for duplicate data and test your database.
⑥ Make the data easily accessible:
→ You need to ensure that your users can benefit from the data.
→ Users should know how to use the database.
→ Gather feedback on your database.
→ Build strong file naming and cataloging conventions.
→ Carefully consider meta-data for data sets.
→ Adequate data storage.
→ Use 3-2-1 methodology.
3: Store three copies of your data.
2: Using two types of storage methods.
1: With one of them stored off site.
→ Produce multiple levels of documentation.

(*). Profiles, Password Policies, Privileges and roles: Introduction.

(**). Users, Roles and Permissions: They work together to determine who can access what inside your database.

→ Users are the individual accounts for authenticating into project.
→ Each user is assigned a role which defines its access permissions.
→ User profile is a collection of settings and information associated with a user.
→ It contains critical info. that can be used to identify a user.

(*) Roles: Roles are created and the permissions are configured once and then the role is assigned to the users as desired.
SQL database will always need an administrator role and a public role.

(*). Administrator: Provides complete, unrestricted control over the database. This cannot be limited.
You need atleast one user in administrative role.

(*). Public: Public role defines access permissions for unauthenticated requests to the database.
If you enable an access permission for this role, everybody has that permission enabled.
All permissions begin turned off by default.

(*). Custom Roles: In addition to these roles, you may need to create more roles with their own unique set of permissions. The roles you create and the permissions you configure for them are completely open ended and dependent on your project's needs.

(*). Permissions: It wouldn't be safe to give every user full access of data.

→ Users could accidently damage the data or even take malicious actions against the project and its users.

→ Example, a student may be able to see their grade but not able to change it.

→ There are 4 types of permissions: CRUD actions.
Create, Read, Update and Delete.

↳ You can grant any combination of these four permissions.

(*). Business Rules: In many cases, you may need to grant permissions to data based on some conditional logic.
This type of conditional permission is called business rule.

→ Eg: Students must be able to read their own grades and not the grades of other students.

→ Eg: We may need students to read and create answers to an online test, but not update or delete their answers once submitted.

(*). Defining and Using Profiles:

→ The CREATE PROFILE statement allows you to create a new user profile.

Syntax: CREATE PROFILE profile_name
LIMIT {resource-parameters/password_parameters};

→ A user profile is a set of limits on the database resources and the user password.

→ You can use the following clauses to set the limit for resource parameters

(a). SESSIONS_PER_USER
(b). CPU_PER_SESSION
(c). CPU_PER_CALL
(d). CONNECT_TIME
(e). IDLE_TIME
(f). LOGICAL_READS_PER_SESSION.

(g). LOGICAL_READS_PER_CALL (i). COMPOSITE_LIMIT
(h). PRIVATE_SGA

→ You can use the following clauses to set limits for password parameters:

(a). FAILED_LOGIN_ATTEMPTS (d). PASSWORD_REUSE_MAX
(b). PASSWORD_LIFE_TIME (e). PASSWORD_LOCK_TIME
(c). PASSWORD_REUSE_TIME (f). PASSWORD_GRACE_TIME

→ To create a profile, the user needs to have the CREATE PROFILE privilege.

→ To find the current profile of a user,
SELECT username, profile FROM dba_users where username='OT';

→ When you create a user without specifying the profile, oracle will assign default profile to the user.

(*). Using oracle profile to set resource limit example:

→ Create a profile called CRM_USERS and set resource limits.

```
CREATE PROFILE CRM_USERS LIMIT
SESSIONS_PER_USER          UNLIMITED
CPU_PER_SESSION            UNLIMITED
CPU_PER_CALL               3000
CONNECT_TIME               15;
```

→ Create a user called CRM.

```
CREATE USER CRM IDENTIFIED BY abcd1234
PROFILE CRM_USERS;
```

→ Verify the profile of the CRM_user:
SELECT username, profile FROM dba_users WHERE username='CRM';

(**). Using ORACLE Create Profile to set password limit example:

→ Create a new profile called erp_users with password limits:

```
CREATE PROFILE erp_users LIMIT
```

FAILED_LOGIN_ATTEMPTS 5
PASSWORD_LIFE_TIME 90;

→ Create a user named sap and set its profile to erp_users:
CREATE USER sap IDENTIFIED BY abcd1234
PROFILE erp_users;

(x) Designing and Implementing Password Policies:

→ Password is the key to opening the user account.
→ Stronger the password, longer it takes for the hacker to break it.
→ Password policies is a set of guidelines that enhances the robustness of the password and reduces the likelihood of breaking
→ If password is weak, hacker can break it and destroy your data and violate your sense of security.
→ Password complexity: Minimum 8 characters, 1 special character, 1 capital letter, etc.
→ Purpose of password complexity is to decrease the chances of hacker guessing it and breaking it.
→ Password Aging: Indication of how long the password can be used before it expires.
→ Password usage: Indication of how many times the same password can be used.
→ Password storage: Store password in an encrypted manner.
→ Oracle Password security profile parameters include:

(a) failed_login_attempts
(b) password_grace_time
(c) password_life_time
(d) password_lock_time
(e) password_reuse_max
(f) password_reuse_time
(g) password_verify_function.

→ There are two authentication protocols supported by Windows:

(a) NTLM (Network LAN Manager)
(b) Kerberos.

(a) NTLM:

→ NTLM authenticates using a challenge / response methodology.
→ When a user attempts to access a resource, the server hosting the resource "challenges", user to prove his/her identity.
→ User then issues a "response" to the challenge.
→ If the response is correct, then the user is authenticated to the server.
→ The server goes through an authorization process for requested resource.

(b). KERBEROS:

→ It uses a secret key known only to the client and server.
→ Allows server to validate authenticity of client and vice versa.
→ It requires a trusted third resource known as KDC (Key Distribution Center).
→ KDC generates secret key for each session established.
→ Once the key is obtained, the client encrypts its request. The server decrypts the message using same key.
→ This tells the server and the client have the same key for the session which is established.

(*). Granting and Revoking User Privileges in SQL:

→ Involves managing the access rights and permissions of users on database objects.

(*)- Granting Privileges: To grant privileges to a user, you typically use the GRANT statement followed by the privileges & objects on which the privileges should be granted.

→ privilege types: can be one or more of the following privileges:

(a) SELECT
(b). UPDATE
(c). INSERT
(d). DELETE
(e). ALL

→ object_name: Refers to the table(s), view(s) or other database objects on which the privileges should be granted.
→ user_name: Name of the user to whom privileges should be granted.
SYNTAX: GRANT SELECT INSERT ON employees TO john;

(*). Revoking Privileges: We can revoke privileges from the user using the REVOKE statement.
Syntax: REVOKE privilege_type(s) ON object_name FROM user_name;
where user_name is the name of the user from whom the privileges should be revoked.
eg: REVOKE INSERT ON employees FROM john;

(*). Creating, Assigning and Revoking User Roles:
①. Creating User Roles: Using the CREATE ROLE statement.
Eg: CREATE role_name;
Replace role_name with the desired name of the role you want to create.

②. Assigning User Roles: Using the GRANT statement.
Eg: GRANT role_name to username.

③. Revoking User Roles: Using the REVOKE statement.
Eg: REVOKE role_name FROM username.

(*). Best Practices:
①. Least Privilege Principle: Only grant minimum privileges necessary for the users to perform their required tasks. This minimizes the risk of unauthorized access and potential damage caused by accidental or malicious actions.

② Separation of Duties: Create distinct roles for different functional areas. This helps to prevent conflict of interest and prevents unauthorized access.

③ Regular reviewing and auditing: Remove any unnecessary or outdated roles and permissions to maintain a clean and secure database environment.

④ Avoid default Privileges: Avoid granting excessive permissions by default. Instead, create specific roles and assign them on as-needed basis.

⑤ Secure Role Management: Restrict the management of roles by restricting access to privileged accounts.

⑥ Testing and validation: Before assigning and revoking roles, thoroughly test and validate the changes in a controlled environment. Ensure that the new roles or modifications do not disrupt the application functionality.

⑦ Documentation and documentation Review: Maintain clear documentation of user roles, their associated permissions, etc. Document the process of creating, assigning and revoking roles to provide a clear understanding.

⑧ Regular Security Training: Provide regular security training and awareness programs for users, administrators and developers. Educate them about the importance of role based access control, best practices, potential risks, etc.