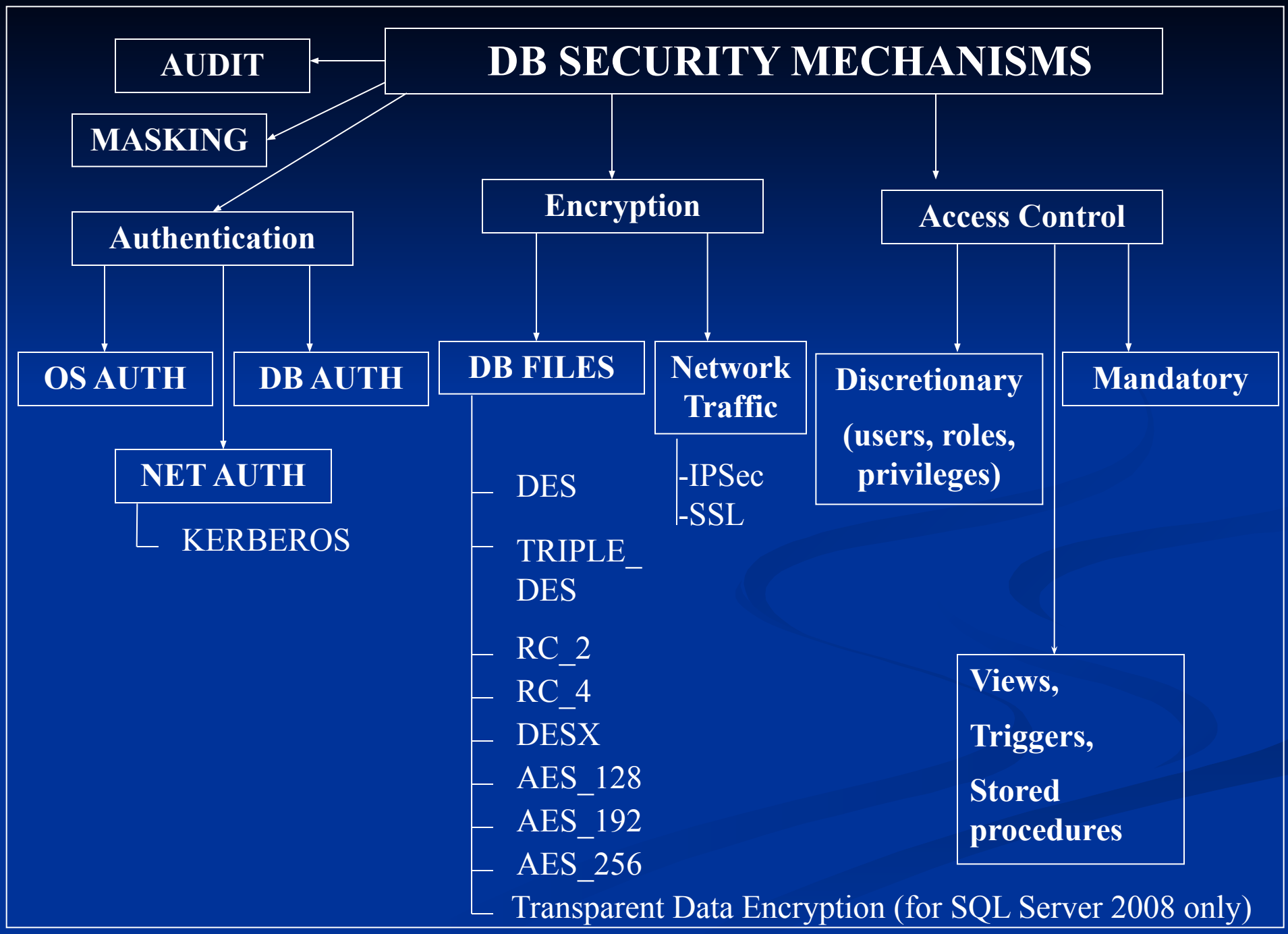
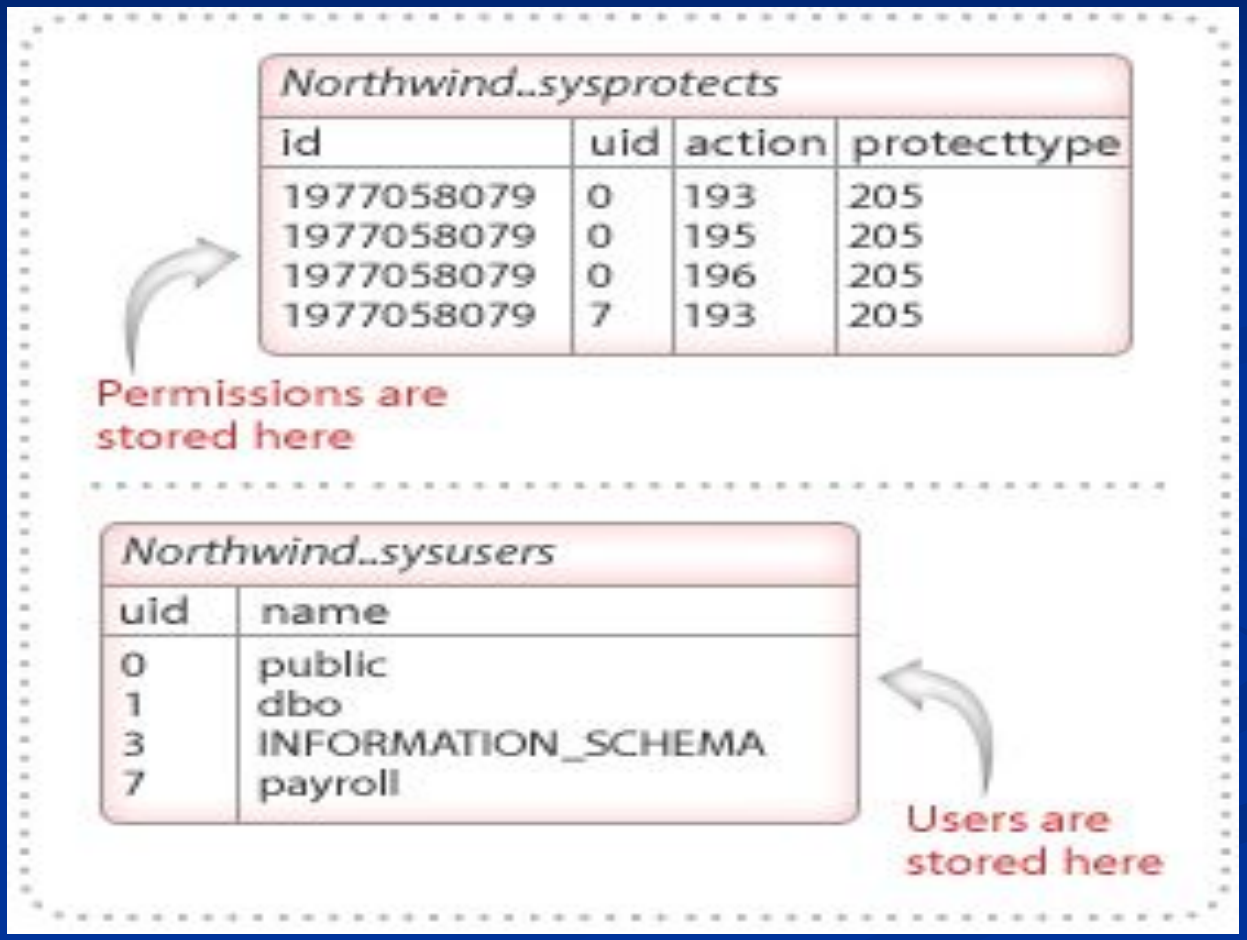
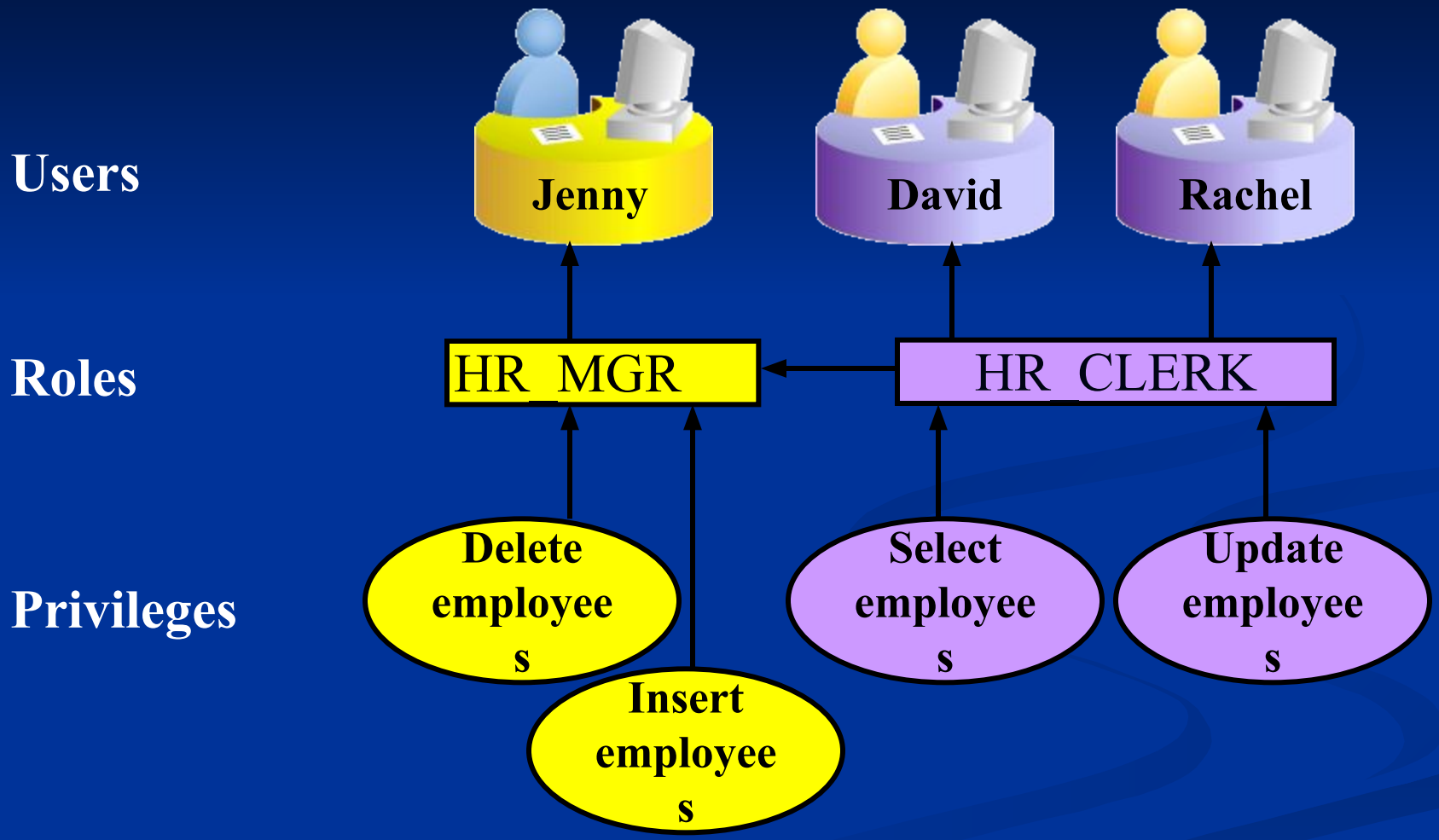

DB SECURITY MECHANISMS
AUDIT
MASKING
Authentication
Encryption
Access Control
OS AUTH
DB AUTH
NET AUTH
KERBEROS
DB FILES
Network Traffic
-IPSec
-SSL
DES
TRIPLE_ DES
RC_2
RC_4
DESX
AES_128
AES_192
AES_256
Transparent Data Encryption (for SQL Server 2008 only)
Discretionary (users, roles, privileges)
Mandatory
Views,
Triggers,
Stored procedures

Discretionary Access Control

- **USERS**
- **ROLES**
- **PRIVILEGES**
- **PROFILES** (In Oracle)



USERS AND ROLES



Privileges

- Database security:
  - System security
  - Data security
- System privileges: Gaining access to the database
- Object privileges: Manipulating the content of the database objects

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

Oracle Access Control

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     scott;
Grant succeeded.
```

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.
```

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 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.
```

PROFILE CREATION

```
CREATE PROFILE profile LIMIT
    [SESSIONS_PER_USER          max_value]
    [CPU_PER_SESSION            max_value]
    [CPU_PER_CALL               max_value]
    [CONNECT_TIME               max_value]
    [IDLE_TIME                  max_value]
    [LOGICAL_READS_PER_SESSION max_value]
    [LOGICAL_READS_PER_CALL     max_value]
    [COMPOSITE_LIMIT            max_value]
    [PRIVATE_SGA                max_bytes]

   max_value :== {integer|UNLIMITED|DEFAULT}
   max_bytes :== {integer[K|M]|UNLIMITED|DEFAULT}
```

PROFILE CREATION

```
CREATE PROFILE profile LIMIT
    [FAILED_LOGIN_ATTEMPTS     max_value]
    [PASSWORD_LIFE_TIME        max_value]
    [ {PASSWORD_REUSE_TIME
      |PASSWORD_REUSE_MAX}     max_value]
    [ACCOUNT_LOCK_TIME         max_value]
    [PASSWORD_GRACE_TIME       max_value]
    [PASSWORD_VERIFY_FUNCTION
     {function|NULL|DEFAULT} ]
```

SQL Server Access Control

LOGIN CREATION

CREATE LOGIN *login_name* { WITH <option_list1> | FROM <sources> }

<sources> ::= WINDOWS [WITH <windows_options> [**,**...]]
| CERTIFICATE *certname*
| ASYMMETRIC KEY *asym_key_name*

<option_list1> ::=
PASSWORD = '*password*' [HASHED] [MUST_CHANGE]
[**,** <option_list2> [**,**...]]

<option_list2> ::= SID = sid | DEFAULT_DATABASE = *database* |
DEFAULT_LANGUAGE = *language* |
CHECK_EXPIRATION = { ON | OFF} |
CHECK_POLICY = { ON | OFF} [CREDENTIAL = *credential_name*]

<windows_options> ::=
DEFAULT_DATABASE = *database* |
DEFAULT_LANGUAGE = *language*

NOT RECOMMENDED (For SQL Server authentication only)
sp_addlogin

USER AND ROLE CREATION

CREATE USER *user_name*
[{ { FOR | FROM }
{ LOGIN *login_name*
| CERTIFICATE *cert_name*
| ASYMMETRIC KEY *asym_key_name*
}
| WITHOUT LOGIN]
[WITH DEFAULT_SCHEMA = *schema_name*]

CREATE ROLE *role_name* [AUTHORIZATION *owner_name*]

NOT RECOMMENDED: **sp_adduser, sp_addgroup**

EXAMPLE

```
CREATE LOGIN testUser
    WITH PASSWORD = '8fdKJl3$nlNv3049jsKK';
USE myDB;
CREATE USER testUSR
    FOR LOGIN testUser
    WITH DEFAULT_SCHEMA = myDB;
GO
```

ADDING USERS TO FIXED SERVER AND DB ROLES

sp_addsrvrolemember [@loginame=] '*login*'
, [@rolename =] '*role*‘

sp_dropsrvrolemember [@loginame =] '*login*'
, [@rolename =] '*role*'

sp_addrolemember [@rolename =] '*role*',
[@membername =] '*security_account*‘

sp_droprolemember [@rolename =] '*role*' ,
[@membername =] '*security_account*’

PRIVELEGES GRANT

SYMPLIFIED SYNTAX

GRANT { ALL [PRIVILEGES] }
permission [**(** *column* [,...n] **)**] [,...n]
[ON [*class* **::**] *securable*]
TO *principal* [,...n] [WITH GRANT OPTION] [AS *principal*]

GRANT OBJECT PRIVELEGES

GRANT <permission> [**,**...n] ON
[OBJECT **::**][*schema_name*]**.** *object_name* [**(** *column* [**,**...n] **)**]
TO <database_principal> [**,**...n]
[WITH GRANT OPTION]
[AS <database_principal>]

<permission> ::= ALL [PRIVILEGES]
| *permission* [**(** *column* [**,**...n] **)**]
<database_principal> ::= *Database_user* | *Database_role* | *Application_role* | *Database_user_mapped_to_Windows_User* | *Database_user_mapped_to_Windows_Group* | *Database_user_mapped_to_certificate* | *Database_user_mapped_to_asymmetric_key* | *Database_user_with_no_login*

PRIVELEGES REVOKE

SYMPLIFIED SYNTAX

REVOKE [GRANT OPTION FOR]
{ [ALL [PRIVILEGES]] |
permission [**(** *column* [**,**...n] **)**] [**,**...n]
}
[ON [*class* **::**] *securable*]
{ TO | FROM } *principal* [**,**...n] [CASCADE] [AS *principal*]

REVOKE OBJECT PRIVELEGES

REVOKE [GRANT OPTION FOR] <permission> [**,**...n] ON
[OBJECT **::**][*schema_name*]**.** *object_name*
[**(** *column* [**,**...n] **)**]
{ FROM | TO } <database_principal> [**,**...n]
[CASCADE] [AS <database_principal>]

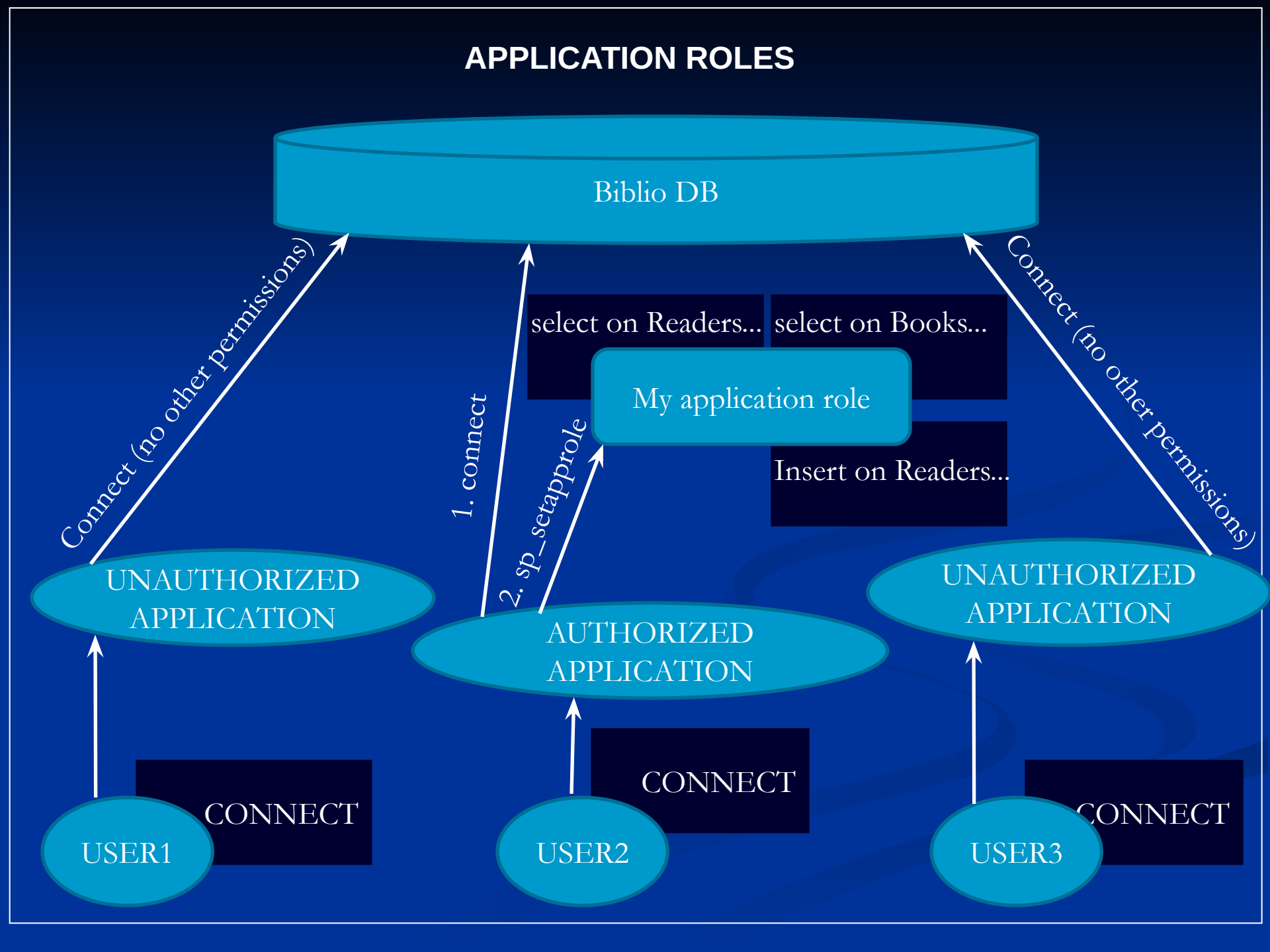
PRIVELEGES DENY

SYMPLIFIED SYNTAX

DENY { ALL [PRIVILEGES] }
| *permission* [**(** *column* [,...n] **)**] [,...n]
[ON [*class* **::**] *securable*]
TO *principal* [,...n] [CASCADE] [AS *principal*]

DENY OBJECT PRIVELEGES

DENY <permission> [**,**...n] ON
[OBJECT **::**][*schema_name*]**.** *object_name* [**(** *column* [**,**...n] **)**]
TO <database_principal> [**,**...n]
[CASCADE] [AS <database_principal>]


APPLICATION ROLES
Biblio DB
Connect (no other permissions)
1. connect
2. sp_setapprole
Connect (no other permissions)
select on Readers...
select on Books...
My application role
Insert on Readers...
UNAUTHORIZED APPLICATION
AUTHORIZED APPLICATION
UNAUTHORIZED APPLICATION
CONNECT
CONNECT
CONNECT
USER1
USER2
USER3

USING APPLICATIONS ROLES

CREATING APPLICATION ROLE

CREATE APPLICATION ROLE *application_role_name*
WITH PASSWORD = '*password*'
[, DEFAULT_SCHEMA = *schema_name*]

SETTING APPLICATION ROLE

sp_setapprole [**@rolename** =] '*role*',
[**@password** =] { **encrypt N**'*password*' }
|
'*password*' [, [**@encrypt** =] { **'none'** | **'odbc'** }]
[, [**@fCreateCookie** =] **true** | **false**] [,
[**@cookie** =] **@cookie OUTPUT**]

ORACLE FINE GRAINED ACCESS CONTROL (VPD)