Database Security – master, 2nd year
Laboratory 3

Management of database users and the computational resources to which they have access

Keywords: • Identity management • Authentication (local, external) • Authorization	• User's schema • Resource profile • Resource consumption plan • The package DBMS_RESOURCE_MANAGER

1. Introduction

- ***Identity management***, in brief terms, is the answer to two questions: who is the user and what resources can he access.
- The constituent elements of an identity management architecture in an information system, in general, are:
  - the central authority granting identities;
  - the methods of identity verification (***authentication***);
  - the mechanisms of identity propagation, in case of an indirect / mediated connection;
  - the plans for access to information and computing resources (***authorization***).
- In what follows, the document is structured in a first part of design and a second part of implementation of an identity management configuration in a database.

2. Part I: Designing the identity management configuration in the database

- In order to design an identity management configuration, it is recommended to go through the following design steps:
  1) identification of the main users of the application (individuals, groups, organizations etc.);
  2) identification of the processes within the application, with the decomposition into functions in the application required to perform the processes (workflows);
  3) building the process-user matrix to highlight the access needs of users' resources;
  4) identification of the entities from the application data modeling;
  5) building the entity - process matrix;
  6) building the entity-user matrix resulting from the process-user and entity-process matrices.

2.1 Going through the design stages in the context of an e-learning application

2.1.1 Who are the users of the application? What identification attributes do they have?

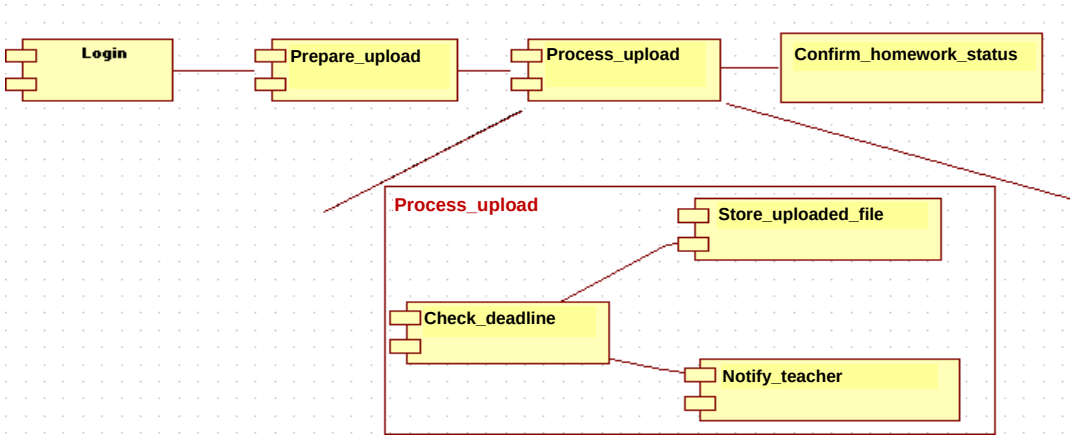
- The users of the application are:
  - Students (full-time / distance) – registration number, name-surname-group, PNC (CNP)
  - Teachers (teaching staff: professors, assistants) – name-surname-department, department - position in department, PNC
  - Secretaries – PNC
  - Application and database administrator – PNC
  - Alumni (graduates) – expired registration number, issued diploma code, PNC
  - The general public – PNC
- At the level of all these users, the PNC is an identification attribute. Another identifying attribute is the confirmed email address. As the field evolves, biometric elements can be considered as identifying attributes.
- However, most of this information can be found and therefore cannot be the sole basis for identity verification (ie for authentication). Therefore, the tandem *identification attribute – password* is usually constructed.

2.1.2 What are the processes within the application?

- Within the application there are the following processes:
  P1: Course configuration
  P2: View the courses' curricula for the academic year
  P3: Course participant enrollment
  P4: Add course materials
  P5: Establish course evaluation session
  P6: Add homework requirements
  P7: Submit homework solution
  P8: Homework assessment and grading
  P9: Record the note for the evaluation session
  P10: View the grade recorded in an assessment session per course participant
  P11: Sending feedback from course participants to a course teacher
  P12: View received feedback
  P13: User administration
  P14: Consult course materials
  P15: View the courses that a student attends

- What is the 2-tier functional decomposition of these processes?

We present a two-tiered functional decomposition of the "Send homework" process (P7)



- For the moment, we note that there may be a boundary in this stream up to which the user's identity propagates. After that, in the following steps, this is replaced by another identity.

2.1.3 Building the process-user matrix

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Full-time students		X					X			X	X			X	X
Distant students		X					X			X	X			X	X
Teachers	X	X	X	X	X	X		X	X			X		X	
Assistants		X	X	X		X		X				X		X	
Secretaries		X			X					X					X
Alumni (graduates)		X													
Aplication's administrator	X	X	X										X		X
General public		X													

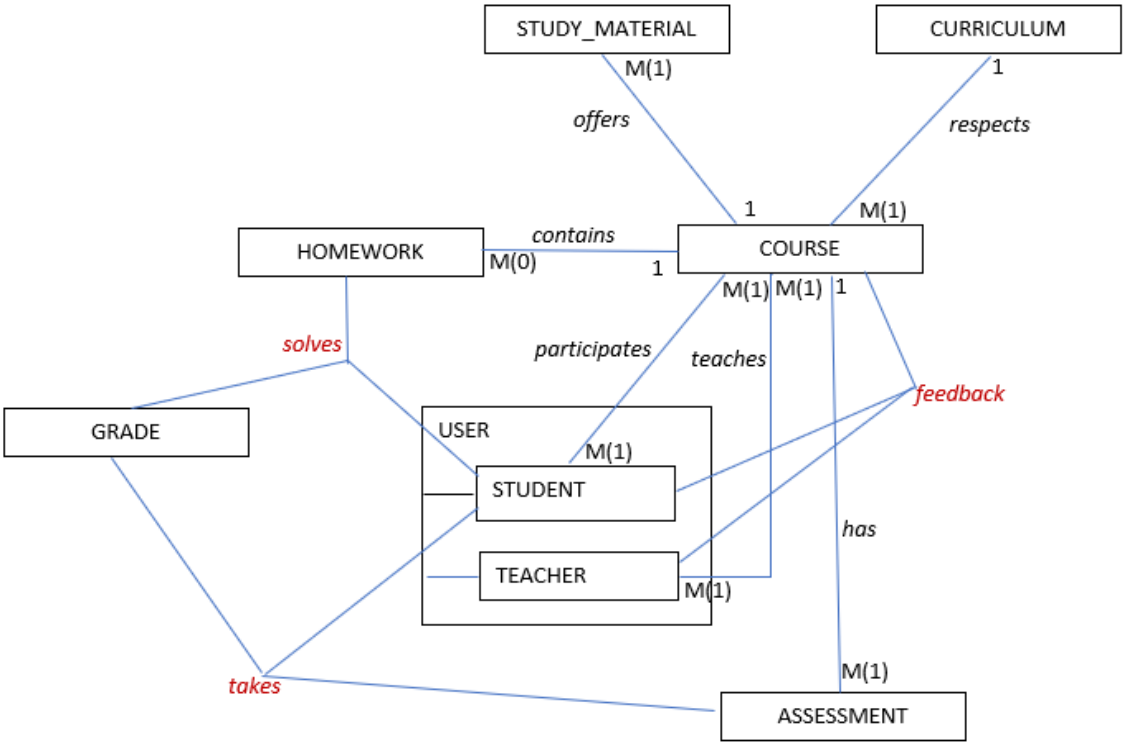
2.1.4 Identifying entities by modeling application data

- The e-learning system is a platform for modern education.
- The courses of an academic year can be configured in the system. These courses should respect the subject's published curriculum. The same subject curriculum can represent the basis of several courses taught in successive years.
- A course (from an academic year) is taught by one or more teachers, as professors or assistants. The course is intended for students (undergraduates, master students) in the current generation and those in arrears, who will be enrolled in the ongoing course until they manage to pass the discipline's exam.
- The teachers of a course can provide study materials to students in a centralized manner. At the same time, the system allows publishing and collecting of homework in a unitary and rigorous manner regarding deadlines. For each completed homework the student

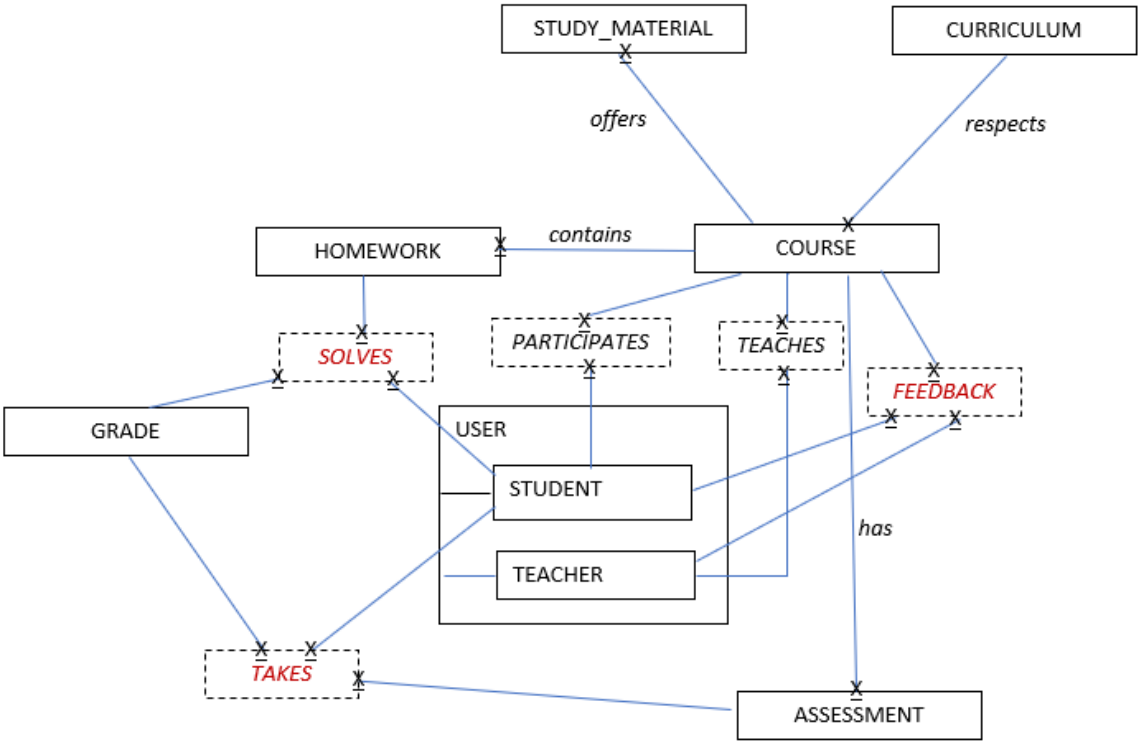
receives a grade. The virtual catalog allows the recording of grades for homework and those resulting from the assessment of students, as well as integration with other administrative computer systems of the faculty.

- The assessment of knowledge for a course (in an academic year) can take place in the pre-established exam sessions of that year. A student's failure to pass the discipline in one year entails the need to complete the whole course, including the homework for the course of the same name in the following academic year.

  For communication, the system allows the transmission of feedback from course participants to course teachers.



---------- ***E/R Diagram***----------



----------***Conceptual diagram*** ----------

Relation schemas:

CURRICULUM (id #, subject_name, year_of_study, nb_course_hours,
nb_seminar_hours, content_description, nb_credits)
COURSE (id#, id_curriculum, academic_year)
STUDY_MATERIAL(id_material#, id_course#, title, link, mandatory_degree)
HOMEWORK (id_homework#, id_course#, title, requirement, max_points, deadline)
GRADE (id#, value)
USER (id#, last_name, first_name, user_type)
STUDENT (id_user#, registration_year, current_year, current_group)
TEACHER (id_user#, department, scientific_degree)
ASSESSMENT (id_assessment#, id_course#, assessment_date, assessment_room)
PARTICIPATES (id_user#, id_course#, flag_arrears)
TEACHES (id_user#, id_course#, flag_course, flag_lab)
SOLVES (id #, id_user, id_homework, id_grade, link_homework_file, flag_copied*,
id_copied_solution*)
TAKES (id#, id_user, id_assessment, id_grade, flag_copied*, id_copied_assessment*)
FEEDBACK (id #, id_course, id_student, id_teacher, message, satisfaction_degree,
feedback_date)

Remark: Fields marked with a "*" in the tables *SOLVES* and *TAKES* are subject to some

special comments that will be made during the lab. (Hint: are the diagrams correct if these fields are present in the 2 relation schemas?)

2.1.5 Building the entity - process matrix

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
CURRICULUM	S	S													
COURSE	I,U	S	S												S
STUDY_MATERIAL				I,U,D										S	
HOMEWORK						I,U	S	S							
GRADE								S	S	S					
USER													I,U		
STUDENT			S										I,U		S
TEACHER											S		I,U		
ASSESSMENT					I,U				S	S					
PARTICIPATES			I,U												S
TEACHES	I,U										S				
SOLVES							I,U	U							
TAKES									I,U	S					
FEEDBACK											I,U,D	S			

Legend: I= Insert , U= update , D= delete, S= select

2.1.6 Construction of the entity-user matrix, resulting from the process-user and entity-process matrices

	Full-time students	Part-time students	Professors	Assistants	Secretaries	Alumnii	App & DB admin	Gen. public
CURRICULUM	S	S	S	S	S	S	S	S
COURSE	S	S	I,U,S	S	S	S	I,U,S	S
STUDY_MATERIAL	S	S	I,U,D,S	I,U,D,S				
HOMEWORK	S	S	I,U,S	I,U,S				
GRADE	S	S	S	S	S			
USER							I,U	
STUDENT	S	S	S	S	S		S,I,U	
TEACHER	S	S					I,U	
ASSESSMENT	S	S	I,U,S		I,U,S			
PARTICIPATES	S	S	I,U	I,U	S		I,U,S	
TEACHES	S	S	I,U				I,U	
SOLVES	I,U	I,U	U	U				
TAKES	S	S	I,U		S			
FEEDBACK	I,U,D	I,U,D	S	S				

Legend: I= Insert , U= update , D= delete, S= select

2.2 Analysis of the entity-user matrix

- Following this design part of the identity management configuration in the database, the information from the entity-user matrix will be analyzed and interpreted.

- We will look for answers to the following questions:

2.2.1 Who are the database users, what user accounts will we create?

- In stage 1) we discovered user classes. However, for audit and non-repudiation reasons, the accounts will be defined individually for users identified within the organization. Another reason is that if a single student account is created, for example, there are restrictions on the number of parallel connections allowed (*sessions_per_user*) and thus some students will be denied the connection, being asked to wait until connection slots are released by other students, which would be a weak point for the quality of the application.
- Thus, for example, we will define 1000 student accounts (one for each student of the faculty), 50 accounts of teachers (one for each teacher), 5 secretaries accounts.
- Two more special accounts will be created: a guest account (*ELEARN_GUEST*) for the general public and an application administrator account (*ELEARN_APP_ADMIN*).

2.2.2 Who is the owner of each object of the database (table, index, etc.)?

- **The owner of a database object** is that user who has unlimited rights of use and administration over that object, rights that cannot be revoked by anyone (not even by the database administrator!).
- A user who owns at least one object in the database cannot be deleted from the system.
- **A user's schema** represents the entire set of objects (tables, views, indexes, database-level triggers, sequences, synonyms, PL/SQL functions, PL/SQL procedures, PL/SQL packages) that have a certain owner database user.

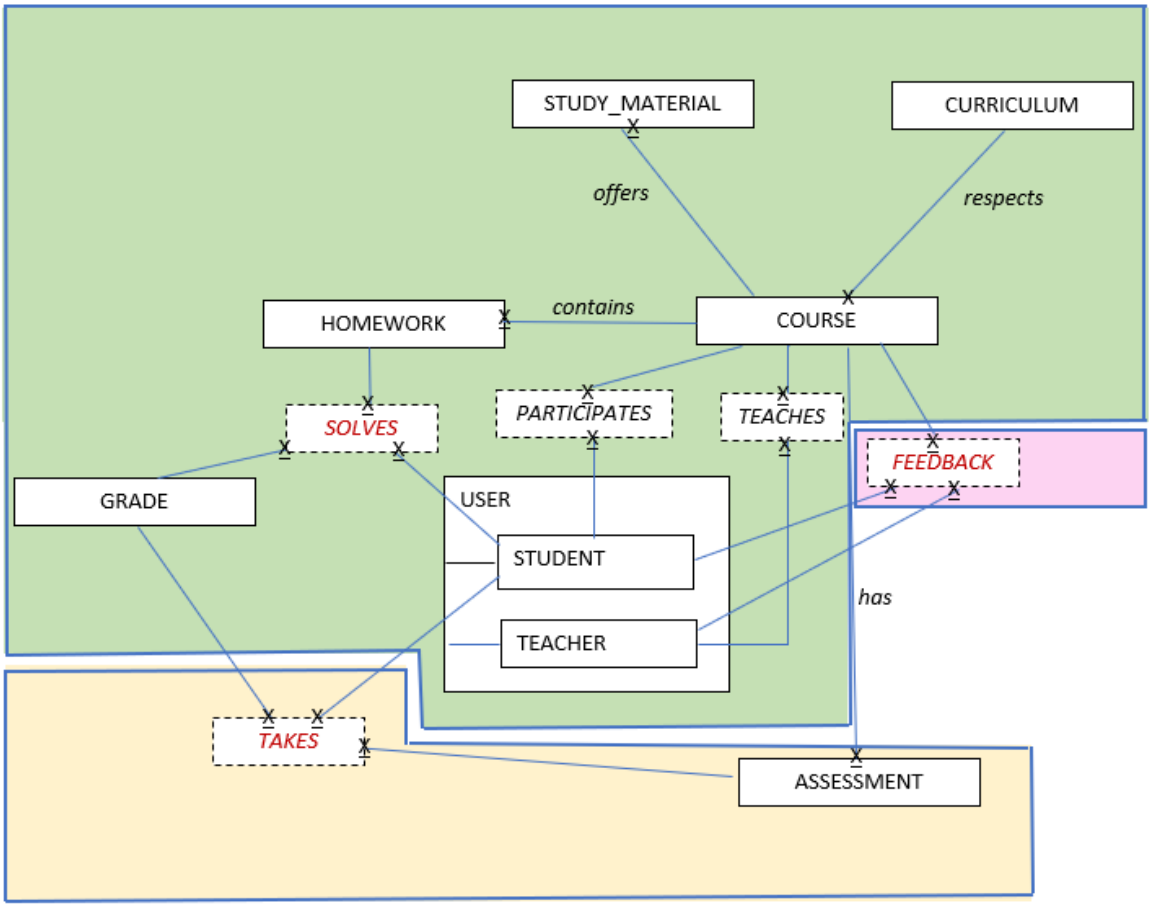
Remark: Keep in mind that there is always a 1: 1 relationship between a user and a user's schema.

- A good practice when developing a database-driven application is to create an account for an application administrator. This user will own all objects in that application; in other words, all the objects of the respective application will be found in the schema of the application administrator user.

  Exceptions are allowed to the extent that overhead of differentiated administration is justified.

- For our e-learning application, we assume that there are the following requirements in the system specifications:
  - The *EVALUATION* and *TAKES* tables to be owned by a separate owner *ELEARN_CAT*;
  - The *FEEDBACK* table to be held by each individual teacher in the system; in other words, if we have 50 teacher users, in each of their schemas we will have a *FEEDBACK* table;

- the rest of the tables will be owned by the application administrator *ELEARN_APP_ADMIN*.

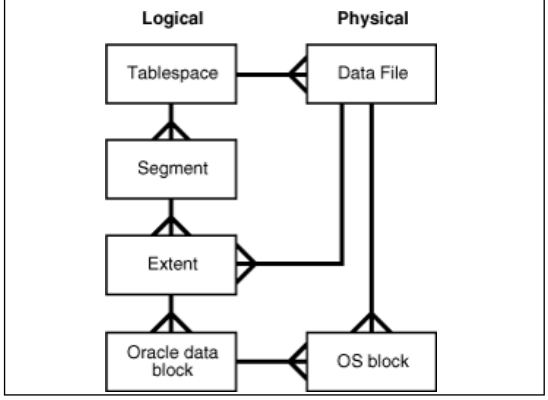


Highlighting the user schemas on the conceptual diagram

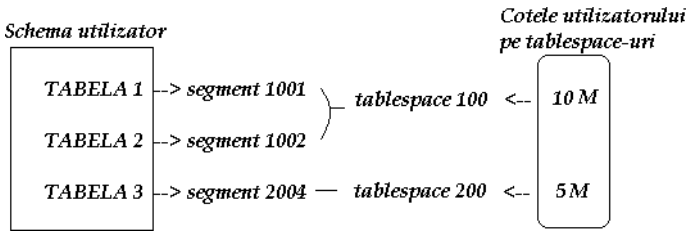
2.2.3 Setting the storage space for the created users

- As stated in the previous point, some users will own tables, others will not. It is therefore natural to consider a differentiated distribution of storage space between users.
- A database user has two storage spaces:
  - storage space for the objects in his schema, over which he is the owner;
  - temporary storage space, used by Oracle in different situations (for example, in sorting records when processing a query with the *ORDER BY* clause or in building the hash table in the build stage of processing the join operation by the *HASH JOIN* method).

Sursa: http://docs.oracle.com/cd/E11882_01/server.112/e25789/logical.htm



- Without going into details about tablespaces in Oracle, we note the following:
  - The tablespace is a *logical* storage unit, consisting of segments; a segment stores the data of a single object of a user (1 unpartitioned table without CLOB = 1 segment, 1 unpartitioned index = 1 segment etc.)
  - The tablespace binds the database to the file system, as a tablespace has at least one physical datafile associated to it.
- A user may have different quotas on different tablespaces for storing objects in his schema. An example would be the following:



- When a user runs out of allocated storage space, the database administrator can choose to increase an existing user quota or decide to allocate a quota from a new tablespace.

- From the point of view of the e-learning application, only one tablespace will be used. Its size will be divided into quotas as follows:
  - The user *ELEARN_APP_ADMIN*: unlimited
  - The user *ELEARN_CAT*: 10MB
  - The teacher users *ELEARN_professor1*, *ELEARN_professor2*, *ELEARN_assistant3*: 2MB each
  - The rest of the users: 0MB → they will not be able to create objects.

2.2.4 Establishing access to computing resources and other configurations

- The next step, after creating user accounts and establishing storage spaces, is to impose restrictions on access to resources for users.
- The purpose is to ensure the efficient functioning of the database, the avoidance of the

monopoly by a user over resources etc.

- The performance parameters that are often found in these configurations refer to:
  - estimated execution time of the statements;
  - CPU consumption;
  - the degree of parallelism accepted in multi-processor systems;
  - the number of open sessions per user;
  - the idle time.
- It is good practice to reuse resource access settings by creating profiles or consumption plans.
- A **resource profil**e is the grouping under a title of a multitude of resource limitations that restrict the use of the database and Oracle instance's resources for a user. At one time, a user may have only one resource profile. The same resource profile can be set for multiple users.
- Regarding the e-learning application, it is requested:
  a) for the application's guests (*ELEARN_GUEST*) the maximum number of allowed connections to be 5, the maximum allowed idle time to be 3 minutes, and the total connection time not to exceed 15 minutes even during the activity periods;
  b) for teacher users and student users, the CPU consumption per call should not exceed the threshold of 60 seconds; they should be entitled to only one session at a time, the lifespan of the password should be 7 days and the password may be wrong up to 10 times.
- **A resource consumption plan** is a tool by which the DBMS takes control of the allocation of computational resources to user-session groups, called consumption groups. A complex resource plan can be represented as a top-down decomposition of a resource consumption in the form of a tree in which the root is the main plan, in the leaves are the consumption groups, and on the intermediate levels are the consumption sub-plans.

2.2.5 Temporarily blocking users' access to the database

- The decision to temporarily block a user's access to the database can have various reasons: non-compliance by the user with some rules / regulations, going on vacation during the summer (with the sealing of the offices) etc.
- These temporary locks are reversible.

3. Part II: Implementing the identity management configuration in the database

To implement the identity management configuration, several steps are required. These will be performed by the database administrator.

3.1 Creating user accounts

- We recall that authentication means verifying a user's identity. We present two authentication methods among those supported by Oracle.
- **The first method is the local authentication, at the database's level**. This is done by username and password.
  - A clause can be specified to force the user to change his password the first time he logs in to the account.
  - Necessarily, in order to allow the newly created user to connect to the database, we must give him the privilege (right) to create a session.
  - **Syntax:**

```
CREATE USER student1 IDENTIFIED BY pwdstudent PASSWORD EXPIRE;
GRANT CREATE SESSION TO student1;
```

Remarks:

- If the user is created with the expired password clause, but has not yet been granted the session creation privilege, then the user will be able to change his password, but not log in.
- It is recommended that the password expires after the account has been created so that there is no suspicion that the account password is also known to the DBA – in terms of non-repudiation of the database activity.
- For a guest account it will not be necessary to expire the password; this will remain as set by the admin.
- Regarding the passwords, they are stored encrypted in the database.
- The rule that the username cannot exceed 30 characters applies.

- **The second method is the external authentication, at the level of the operating system.** This means that, once the user has logged in to the operating system with the corresponding credentials (username / password), he can immediately access the database without further login.
  - Among specialists, there are some reservations about this dependence on operating system security. Another aspect of these reservations relates to remote connection using external authentication which implies the risk that, once the username is disclosed, anyone can bring a laptop in the institution's network, create an account in the operating system on the laptop with that name and, thus, get database access.
  - The following database username scheme is used based on the operating system

username (for example, Windows), *if the user has an account on the server's operating system*:

os-user	os_authent_prefix	database user
MM-S101\TOM	"OPS$"	"OPS$MM-S101\TOM"

where:

→ *os-user* = USERDOMAIN\username – is determined on the command line (in the terminal) at the level of the (Windows) operating system, under a working session of the new user:

```
echo %username%
echo %USERDOMAIN%
```

→ *os_authent_prefix* – is determined using the query (the default value is OPS$):

```
SELECT value FROM v$parameter
WHERE name = 'os_authent_prefix';
```

- **Syntax:**
  - For the users created on the server (with Windows operating system):

```
CREATE USER "OPS$domain\name" IDENTIFIED EXTERNALLY;
GRANT CREATE SESSION TO "OPS$domain\name";
GRANT CONNECT TO "OPS$domain\name";
```

  - For the users created on the workstations:

```
CREATE USER "host\username" IDENTIFIED EXTERNALLY;
GRANT CREATE SESSION TO "host\username";
GRANT CONNECT TO "host\username";

ALTER SYSTEM SET REMOTE_OS_AUTHENT=TRUE SCOPE=SPFILE;
[SHUTDOWN IMMEDIATE & STARTUP]
```

Remark: For external users, the password expiration rule does not work and no password is stored in the data dictionary. However, the rule that the length of the username, OPS$domain/name, cannot exceed 30 characters applies.

- Consulting the ***DBA_USERS*** view provides information about the created users.

```
SELECT USERNAME, AUTHENTICATION_TYPE, ACCOUNT_STATUS, CREATED,
       EXPIRY_DATE
FROM DBA_USERS
WHERE USERNAME LIKE '%ELEARN_%' ORDER BY CREATED;
```

3.2 Setting the storage space for the created users

- Immediately after creation, without additional configurations, you can find the default value of the tablespaces assigned to the created users, using the query:

```
SELECT USERNAME,DEFAULT_TABLESPACE,TEMPORARY_TABLESPACE
FROM DBA_USERS
```

```
WHERE USERNAME LIKE '%ELEARN_%'
ORDER BY CREATED;
```

USERNAME	DEFAULT_TABLESPACE	TEMPORARY_TABLESPACE
ELEARN_APP_ADMIN	USERS	TEMP
ELEARN_STUDENT1	USERS	TEMP
ELEARN_STUDENT2	USERS	TEMP
ELEARN_STUDENT3	USERS	TEMP
ELEARN_STUDENT4	USERS	TEMP
ELEARN_STUDENT5	USERS	TEMP
ELEARN_STUDENT6	USERS	TEMP
ELEARN_STUDENT7	USERS	TEMP
ELEARN_STUDENT8	USERS	TEMP
ELEARN_STUDENT9	USERS	TEMP
ELEARN_STUDENT10	USERS	TEMP

USERNAME	DEFAULT_TABLESPACE	TEMPORARY_TABLESPACE
ELEARN_PROFESOR1	USERS	TEMP
ELEARN_PROFESOR2	USERS	TEMP
ELEARN_ASISTENT3	USERS	TEMP
ELEARN_GUEST	USERS	TEMP
OPS$MM-33C58500149B\ELEARN_CAT	USERS	TEMP

16 rows selected.

- **Syntax:**

```
ALTER USER username QUOTA quota_value ON tablespace_name;
```

- In this way, a user can be set the quota on the default tablespace (*USERS*) or can be allocated spaces in new nominated tablespaces.

- Possible quota values can be:
  - unlimited;
  - 0 – case in which the user cannot create any object in the future in the respective tablespace;
  - a value expressed in MB, for example 10M.

Remark: If we want the storage space for users U_1, U_2, .., U_n in a tablespace to be x_1, x_2, .., x_n MB, we go through the steps:

1) we find the total size of the respective tablespace, in MB:

```
SELECT tablespace_name, ROUND(SUM(bytes) / 1048576) TotalSpace
FROM dba_data_files
GROUP BY tablespace_name;
```

TABLESPACE_NAME	TOTALSPACE
UNDOTBS1	55
SYSAUX	580
USERS	5
SYSTEM	680
EXAMPLE	100

2) we make sure that the proposed configuration is feasible (does not exceed the size of the tablespace)

3) we set the quota x_i for each user U_i , $i = 1, ..., n$.

```
ALTER USER Ui QUOTA xi ON tablespace_name;
```

3.3 Establishing access to computing resources and other configurations

We will present two ways to limit access to computing resources in Oracle: resource profiles and consumption plans.

- **The first way is to group the resource access restrictions into profiles and attach profiles to users.**
  - The following steps are used to set access limits to computational resources for a user:

    1) Create a computational resource limitation profile, which includes values for the desired parameters. Optionally, the profile can also include limitations on a user's password (which only take effect in the case of local authentication).

    2) The limitation profile is attached to the user's account. From this moment, the user's activity is carried out entirely under the auspices of the new profile.

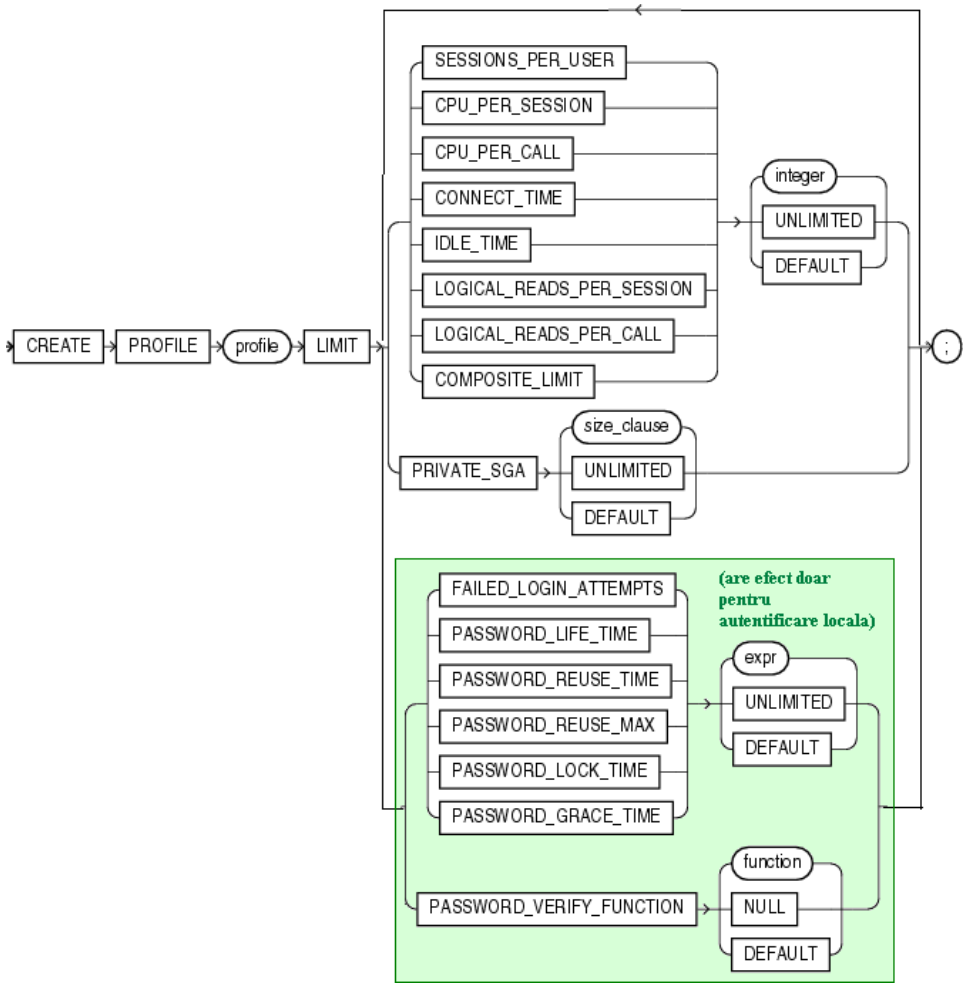
Note that there is an m:1 relationship between the user and the profile limiting access to computing resources, i.e. a user has only one profile (the last one that was configured), but the same profile can be attached to multiple user accounts.

  - **Syntax:**

    1) Creating a profile of computational resource limitations

    The diagram on the next page shows that a verification function (*PASSWORD_VERIFY_FUNCTION*) can be set at the password level to ensure that passwords follow certain rules. The signature of this function is:

```
CREATE OR REPLACE FUNCTION my_verification_function (username
 VARCHAR2, new_password VARCHAR2, old_password VARCHAR2) RETURN
 BOOLEAN AS …
```



Remark: No commas are placed between the parameters. The units of measurement can be found in the documentation (https://docs.oracle.com/database/121/SQLRF/statements_6012.htm#SQLRF01310).

2) Attaching the new restriction profile to the *username* account:

```
ALTER USER username PROFILE profile_name;
```

- Information about the configuration of each profile is obtained using the query:

```
SELECT * FROM DBA_PROFILES WHERE PROFILE = 'profile_name';
```

- Information on a user's current resource profile is obtained by querying:

```
SELECT USERNAME,PROFILE FROM DBA_USERS;
```

- **The second way is to group users into consumer groups and establish a plan for allocating resources to these groups.**

  The following steps are used to create a simple consumption plan within a PL/SQL block:

  1) Creating a work area to define the plan;

  Syntax:

```
DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA();
```

2) Creating the consumption plan, which is a container for the plan directives;

Syntax:

```
DBMS_RESOURCE_MANAGER.CREATE_PLAN
(PLAN => 'plan_name',
COMMENT => 'This is a plan for ...');
```

3) Creating consumer groups – they represent groups of user sessions that will share the same amount of resources;

Syntax:

```
DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP
(CONSUMER_GROUP => 'group_name',
COMMENT => 'This groups the sessions of users which...');
```

4) Creating static mappings between users and consumer groups (details on dynamic runtime re-mapping, after login, can be found in the documentation: https://docs.oracle.com/database/121/ARPLS/d_resmgr.htm#ARPLS74594)

Syntax:

```
DBMS_RESOURCE_MANAGER.SET_CONSUMER_GROUP_MAPPING
(DBMS_RESOURCE_MANAGER.ORACLE_USER, 'user_name', 'group_name');
```

5) Creation of plan directives – specifying, within the plan, some rules for allocating resources for each consumption group:

Syntax:

```
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE
(PLAN => 'plan_name', GROUP_OR_SUBPLAN => 'group_name',
COMMENT => 'comments...', PARAM1 => VAL1, PARAM2=> VAL2, ...);
```

6) Validation and completion of the implementation of the consumption plan.

Syntax:

```
DBMS_RESOURCE_MANAGER.VALIDATE_PENDING_AREA();
DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA();
```

Remarks:

- The consumption group *OTHER_GROUPS*, which corresponds to the default case, must not be missing from a system with consumption plans. Otherwise the following error will occur:

```
SQL> exec ELEARN_plan_consum
BEGIN ELEARN_plan_consum; END;

*
ERROR at line 1:
ORA-29382: validation of pending area failed
ORA-29377: consumer group OTHER_GROUPS is not part of top-plan ELEARN_PLAN1
ORA-06512: at "SYS.DBMS_RMIN", line 437
ORA-06512: at "SYS.DBMS_RESOURCE_MANAGER", line 798
ORA-06512: at "SYS.ELEARN_PLAN_CONSUM", line 78
ORA-06512: at line 1
```

  If it exists in the system (check in the *DBA_RSRC_CONSUMER_GROUPS* view), then it cannot be recreated or deleted.

- According to Oracle documentation, if the CPU usage is below 100%, the CPU quota rules

in the consumption plan do NOT apply. They only take effect when the system is overloaded.

- Information about consumption plans and plan directives can be found by querying views *DBA_RSRC_PLANS* and *DBA_RSRC_PLANS_DIRECTIVES*.

3.4 Temporarily blocking users' access to the database

Syntax:

1) **Method 1:** revoking the privilege to connect to the database:

```
REVOKE CREATE SESSION FROM username;
```

The privilege can be restored by the command:

```
GRANT CREATE SESSION TO username;
```

Remark: Depending on a user's roles and privileges, there is a possibility that they will still be able to connect to the database after the privilege is revoked. In Lab 4 we will discuss in detail aspects related to privileges and roles.

2) **Method 2:** account lock

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

The account can be unlocked by the command:

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

4. Exercises

1. According to those discussed in point 2.2.1, the following user accounts will be created in the system:

*** with local authentication:**	1 administrator : ELEARN_APP_ADMIN
	10 students: ELEARN_student1,… , ELEARN_student10
	3 teachers: ELEARN_professor1, ELEARN_professor2, ELEARN_assistant3
	1 guest: ELEARN_GUEST
*** with authentication at the operating system level:**	1 catalog manager: ELEARN_CAT

2. Create two triggers for auditing database connections: one trigger will monitor and record logon actions, and the second trigger will record the end of previously opened sessions.
The monitoring results will be stored in the *ELEARN_AUDIT_CONEX* table, which will record: username, session id, identification type, identity, host from which it connected, logon time, logout time.

3. As discussed in point 2.2.3, allocate users storage space quotas on the default tablespace USERS.

Its size will be divided into quotas as follows:

- user *ELEARN_APP_ADMIN*: unlimited
- user *ELEARN_CAT*: 10MB
- users *ELEARN_professor1*, *ELEARN_professor2*, *ELEARN_assistent3*: 2MB each
- other users: 0MB → they will not be able to create objects.

4. Create user profiles (as discussed in point 2.2.4):

a) for guests (*ELEARN_GUEST*) the maximum number of allowed connections should be 5, the maximum allowed inactivity (idle) time should be 3 minutes, and the total connection time should not exceed 15 minutes even during the activity periods;

b) for teacher and student users, the CPU consumption per call should not exceed the threshold of 60 seconds, they should be entitled to only one session at a time, the lifespan of the password should be 7 days and the password might be wrong maximum 10 times.

5. Create a procedure that configures a resource plan with the following rules:

There will be 4 consumption groups: management (admin, *ELEARN_CAT*), tutors (professors, assistants), receivers (students, *GUEST*) and the others – with CPU consumption rates of 20%, 30%, 40%, 10% respectively.