Структура СУБД

Використання СУБД

На малюнку 1 показані 2 різних джерела управляючих інструкцій:

- пересічні користувачі та прикладні програми, які запитують або змінюють дані;
- адміністратор бази даних (database administrator – DBA), особа або група осіб, яка відповідає за підтримку та розвиток структури, або *схеми* даних.

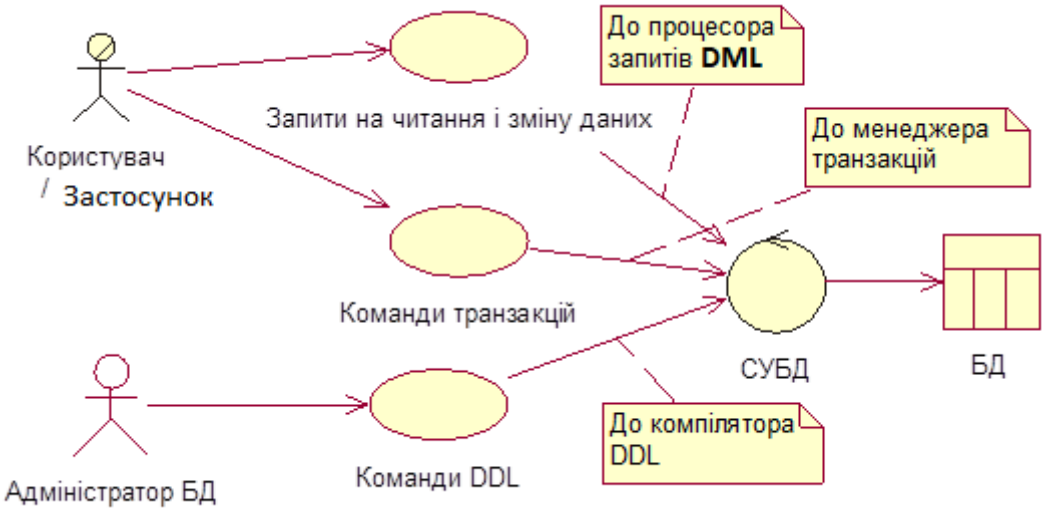

Рисунок 1 - Діаграма варіантів використання СУБД

Середовище SQL

Середовище SQL – це робоче оточення, в якому здатні існувати дані, а також здатні виконуватись SQL-операції їхньої обробки. Іншими словами, це діючий екземпляр СУБД.

Згідно стандарту середовище SQL складається з наступних структур:

- *Схеми (БД)* – колекції персистентних (збережених) таблиць, віртуальних таблиць, обмежень, тригерів, збережених процедур.
- *Каталоги* – колекції схем. Кожний каталог містить одну або кілька „звичайних” схем (чиї імена в межах каталогу мають бути унікальними) та одну спеціальну схему, що має назву INFORMATION_SCHEMA, яка описує всі інші схеми каталогу.
- *Кластери* – колекції каталогів. Кожному користувачу відповідає певний кластер – множина доступних каталогів. Це найширший контекст, в якому можна виконати команду SQL.

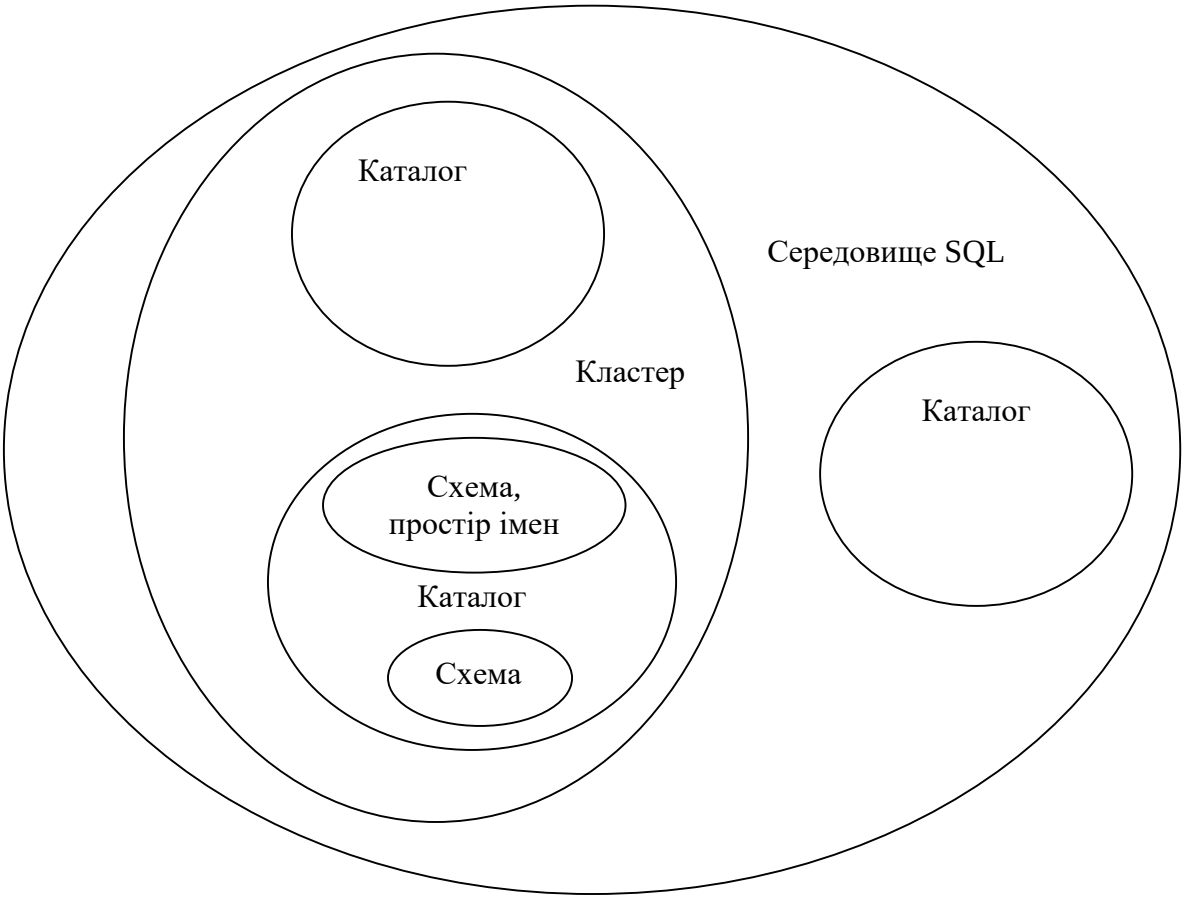

Рисунок 2 - Ієрархія структурних елементів середовища SQL

Приклад 1. Нарис оголошення схеми кінематографічної БД:
CREATE SCHEMA MovieSchema
CREATE TABLE MovieStar <оголошення атрибутів відношення MovieStar>
<Оголошення інших базових таблиць>
CREATE VIEW MovieProd < оголошення віртуальної таблиці MovieProd >
< Оголошення інших віртуальних таблиць >
CREATE TRIGGER NetWorthTrigger <оголошення тригеру NetWorthTrigger >
<Оголошення інших тригерів>
<Оголошення інших елементів>
Встановлення поточної схеми: SET SCHEMA <ім’я схеми>;
Для встановлення поточного каталогу стандарт рекомендує команду
SET CATALOG <ім’я каталогу>;

Для модулів (збережених процедур, тригерів, функцій або пакетів) і таблиць (базових і віртуальних) застосовується один і той же простір імен. Це означає, що в межах однієї *схеми* бази даних всі об'єкти, що використовують один і той же простір імен, повинні мати унікальні імена. Наприклад, не можна надати однакові імена процедурі і тригеру або розрізу.

В Oracle поняття *схема* використовується замість поняття *база даних.* Хоча в MsSQL *база даних* грає центральну роль, можна виконати запит до таблиць кількох різних баз даних.

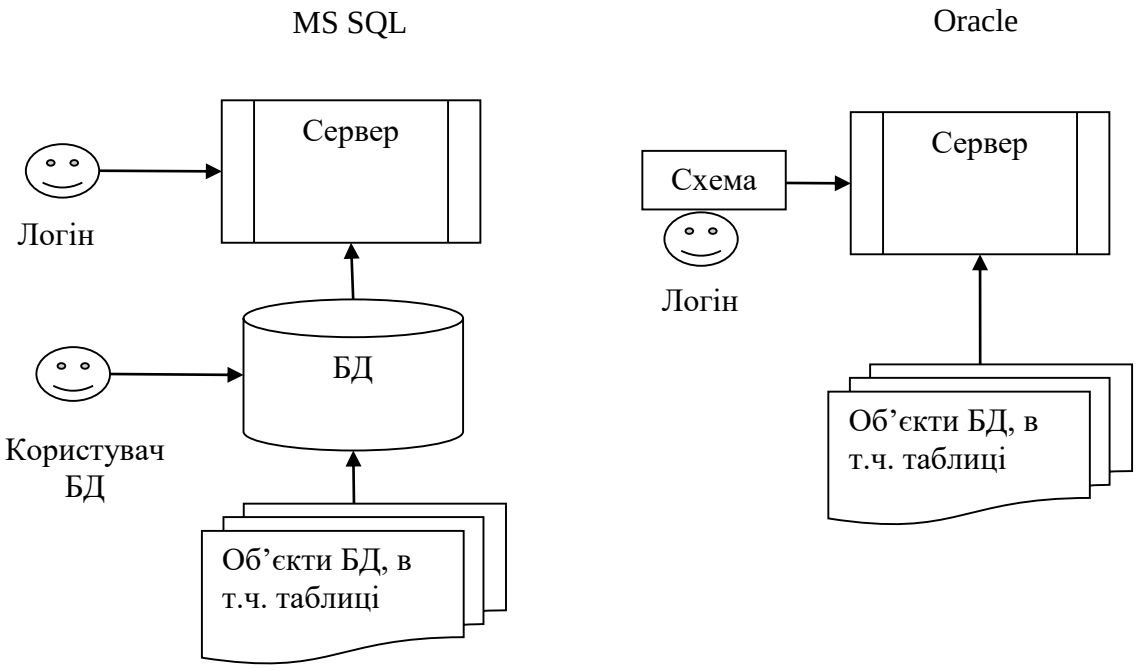

Рисунок 3 – БД в Ms SQL і схема в Oracle.

Клієнти, сервери і з'єднання

В контексті середовища SQL існують *процеси* двох спеціальних категорій:

- *SQL-сервер* – забезпечує виконання операцій над елементами баз даних. Як правило, виконується на потужному комп'ютері, на якому також зберігаються БД.
- *SQL-клієнт* – дозволяє користувачу під'єднатись до сервера та активізувати необхідні йому прикладні операції. Як правило, виконується на віддаленій робочій станції, хоча може виконуватись і на комп'ютері сервера.

Під час запуску програми, яка звертається до сервера, на клієнтському комп'ютері треба встановити *з'єднання*. Команда стандарту SQL (нема в MsSQL):

CONNECT TO <ім'я серверу> AS < ім'я з'єднання>
AUTHORIZATION < ім'я і пароль користувача>

В MsSQL:

CONNECT TO {[*server_name***.**]*database_name*} [AS *connection_name*] USER [*login*[**.***password*] | $integrated]

В Oracle синтаксис оператора CONNECT залежить від оболонки на кшталт SQL*Plus.

Операції SQL, що виконуються за допомогою з'єднання, в сукупності утворюють *сеанс* (session). Сеанс асоціюється з поточним каталогом, поточною схемою та авторизованим користувачем.

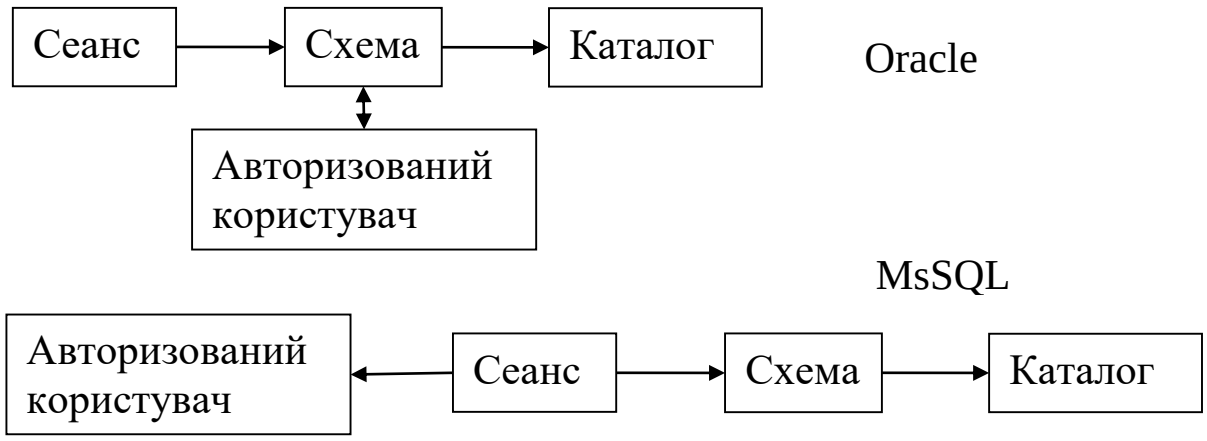

Рисунок 4 – ER-діаграми структур каталогів Oracle та MsSQL

Модулі і агенти

У SQL терміном *модуль* позначається прикладна програма. 3 можливі категорії модулів:

- Базовий інтерфейс SQL (generic SQL interface) – "консоль" для безпосереднього введення команд SQL.
- „Впроваджені" команди SQL (embedded SQL) – тобто вбудовані в текст прикладної програми. Модулем є відкомпільована версія цього застосунку.
- Модулі загального вигляду (generalized modules) – зкомпільовані з базової мови (в тому числі з процедурного SQL) фрагменти прикладної програми, що містять вбудовані оператори SQL і зберігаються на сервері як збережені процедури.

Процес виконання модуля SQL називають *SQL-агентом* (SQL agent). Тобто на відміну від модуля це набір динамічних характеристик коду в конкретний момент часу.
Агент – модуль, що виконується. SQL-агент – модуль, що виконується на сервері для SQL-клієнта.

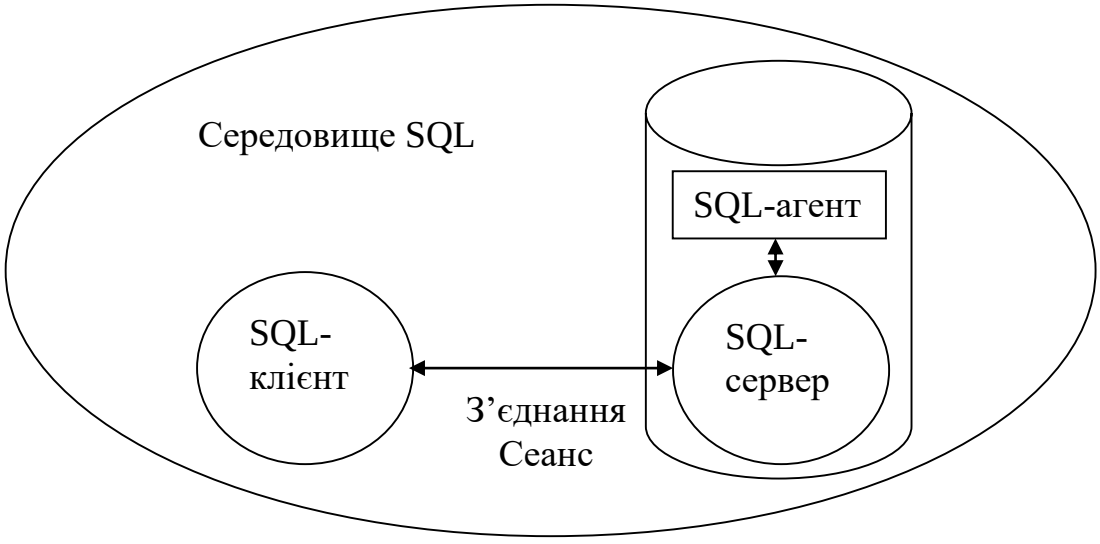

Рисунок 5 – Взаємодія між процесами клієнта і сервера в середовищі SQL.

Логічні компоненти бази даних

Об’єкти БД

Відштовхуючись від MsSQL, перелічимо дані, доступні користувачу в базі в вигляді кількох множин сутностей (класів об’єктів):

Таблиця	= відношення. Двовимірний об’єкт, що складається з рядків і стовпчиків. У кожній таблиці зберігається інформація про один з класів об'єктів, що моделюються базою даних.
Тип даних	Атрибут, що задає тип інформації, яка може зберігатися в стовпці, параметрі або змінній. SQL Server підтримує декілька системних типів даних; також можна визначати користувацькі типи даних.
View, розріз	Об'єкт БД, на який в SQL-операторах можна посилатися як на таблицю. Визначаються за допомогою sql-операторів і є аналогами результуючих наборів, що отримуються при виконанні цих операторів.
Збережена процедура	Відкомпільований набір SQL-операторів, що зберігається під певним ім'ям і обробляється як єдине ціле. SQL Server надає системні збережені процедури, програміст може додати користувацькі.
Функція	Фрагмент коду, що діє як єдина логічна сутність. Функцію можна викликати по імені, при цьому дозволяється задати ряд необов'язкових вхідних параметрів. Вона повертає відомості про стан і необов’язкові вихідні значення. Як і процедури, передбачені системні і користувацькі функції.
Індекс	Об'єкт реляційної БД, що забезпечує швидкий доступ до рядків таблиці на основі впорядкованих значень ключа, а також унікальність рядків у таблиці. Первинний ключ таблиці індексується автоматично. При повнотекстовому пошуку інформація про ключові слова і їх розташування в стовпці зберігаються в повнотекстовому індексі.
Обмеження	Властивість, що призначається стовпцю таблиці, для запобігання занесенню неприпустимих даних в стовпець (UNIQUE, Primary_key, CHECK, NOT NULL).
Замовчання	Значення, що автоматично привласнюється системою даним, параметру, режиму зіставлення або імені, якщо воно не задане користувачем. Також визначає дію, що автоматично виконується при конкретних подіях у відсутність дій, заданих користувачем.
Тригер	Збережена процедура, яка виконується при модифікації даних в певній таблиці. Тригери часто створюють для узгодженості логічно зв'язаних даних в різних таблицях.

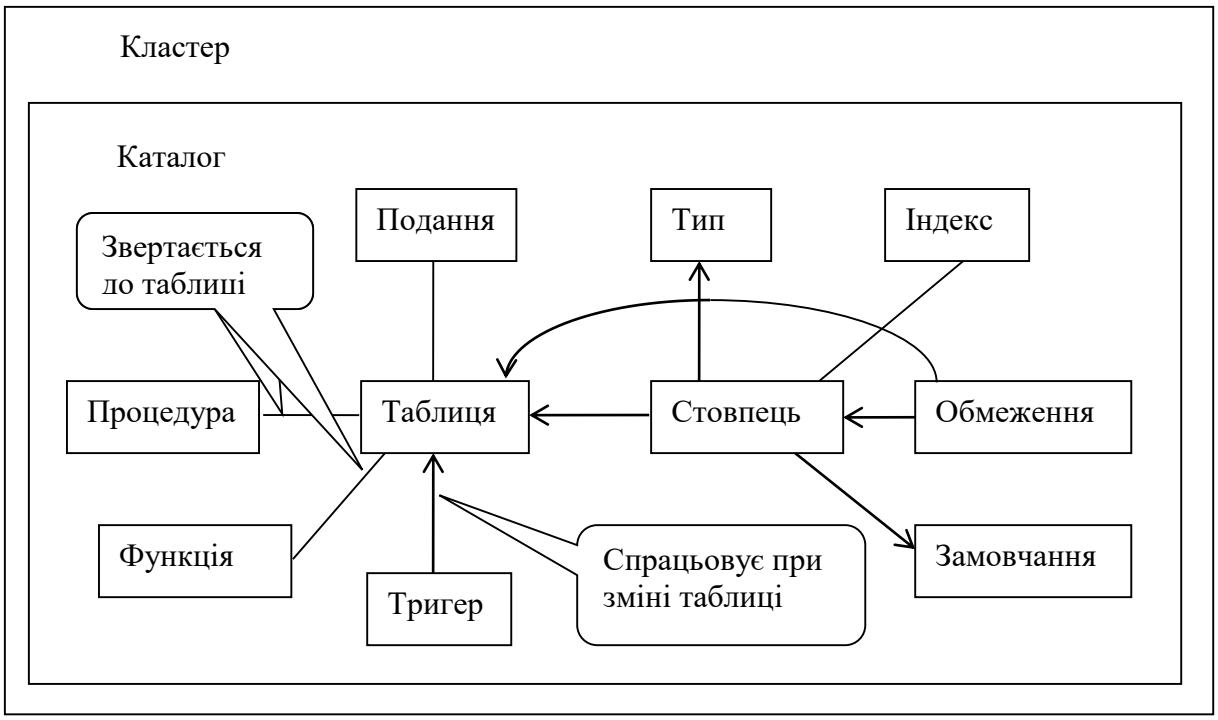

Рисунок 6 – Співвідношення логічних компонентів бази даних у вигляді ER-діаграми

Режими співставлення

На відміну від перелічених вище об'єктів, режими співставлення не є візуальними.
Режими співставлення задають бітові комбінації, що представляють кожний символ, а також правила сортування і порівняння символів.

Для MsSQL 7.0 (1999р.) і попередників режим співставлення був один для всіх баз і об'єктів. Починаючи з MsSQL2000 можна задавати різні режими співставлення для різних стовпчиків таблиці, змінних і параметрів процедур.

SQL Server 2000 (і пізніші версії) підтримує декілька режимів співставлення, які визначають правила використання символів для мови (наприклад, македонської або польської) або для алфавіту (наприклад, Latin I_general —для латинського алфавіту). Кожен режим співставлення SQL Server визначає три властивості:
• порядок сортування даних unicode-тіпів (nchar, nvarchar і ntext);
• порядок сортування даних не-unicode (char, varchar і text);
• кодову сторінку для зберігання символьних даних у форматі, відмінному від Unicode.

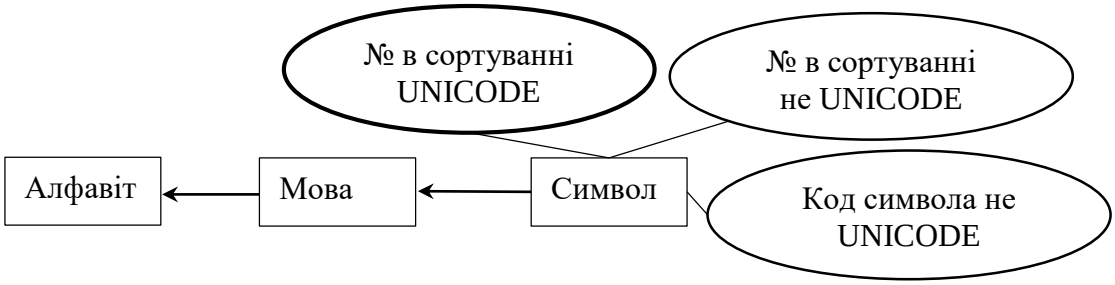

Рисунок 7 – Дані алфавітів у вигляді ER-діаграми. Для повноти треба ще атрибути символа: піктограма і транскріпція.

Безпека бази даних

Доступ до MsSQL

Доступ до сервера MsSQL 2008 грунтується на чотирьох поняттях:

Таблиця 1

Логіни (ідентифікатори користувачів)	Логіни асоціюються з користувачами, що підключаються до SQL Server. Вони задаються членами фіксованої серверної ролі sysadmin.
Ролі на сервері	У Ms SQL Server 2008 маємо 9 наперед визначених ролей (рис. 8)
Користувачі (БД)	Дозволяють ідентифікувати користувачів у *базі даних* (схемі). Всі права доступу і власності на об'єкти БД контролюються на основі імені користувача. Призначені для користувача облікові імена унікальні для бази даних, наприклад облікове ім'я xyz в бухгалтерській БД відрізняється від облікового імені xyz в конструкторській БД (це різні користувачі). Облікові імена визначаються членами фіксованої ролі бази даних db_owner
Ролі в БД	Роль у БД нагадує групу користувачів домена Windows. Вона дозволяє об'єднувати користувачів у групу, а отже, застосовувати права доступу до цих користувачів як до єдиного цілого. Надання прав доступу, відмова в їх наданні, відкликання прав доступу також здійснюється відносно всіх членів ролі. Можна встановити роль, що описує завдання, які повинні виконувати співробітники вашої організації, що обіймають певну посаду, і надати цій ролі відповідні права.

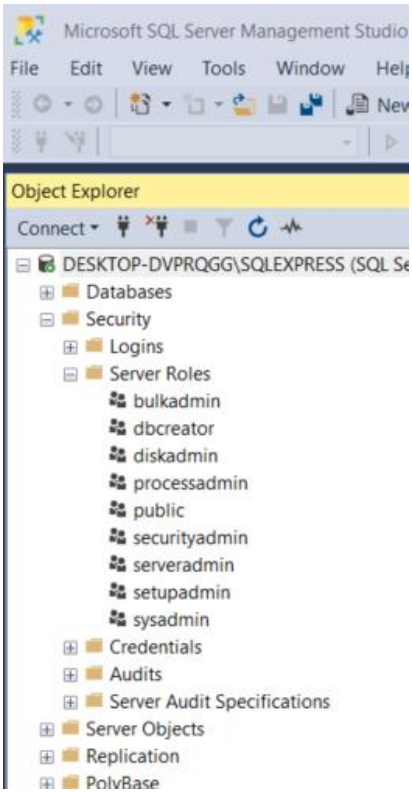

Рисунок 8 – Вбудовані для сервера ролі

Для кожного логіну можливо задати 2 режими перевірки:

- Режим перевірки достовірності Windows. У цьому режимі SQL Server при перевірці достовірності логіну, що запитує доступ до екземпляра SQL Server, покладається на операційну систему. Користувачу не потрібно надавати які-небудь облікові дані в рядку підключення.
- Комбінований режим перевірки достовірності (SQL Server і Windows). У цьому режимі користувач може підключитися до SQL Server з використанням режиму перевірки достовірності або Windows, або SQL Server. У останньому випадку SQL Server перевіряє облікові дані користувача на відповідність діючим іменам входу SQL Server, в цьому разі користувач повинен вказати в рядку з'єднання, ім'я користувача і пароль.

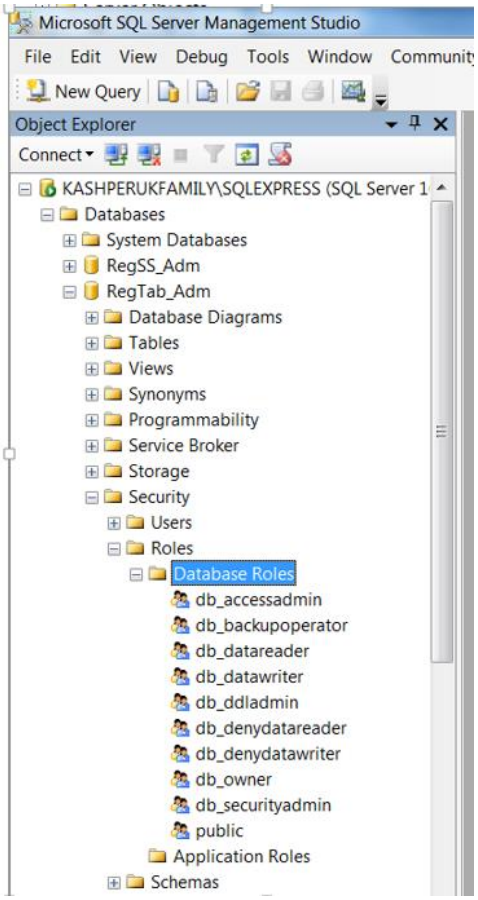

Рисунок 9 – Вбудовані ролі для БД

Як бачимо, є можливість задати ролі для застосунку (Application roles).

Користувач БД необов'язково створюється на основі логіну.

Схема є колекцією об'єктів БД, якими володіє один суб'єкт (*принципал*), при цьому імена об'єктів БД у схемі мають бути унікальні.

- Індивідуальний принципал – це логін або користувач Windows.
- Груповий принципал – це роль або група Windows.

Принципали є власниками схем. У MsSql 2005 і вище схеми не відповідають користувачам. Переваги цього розділення:

- Один принципал може володіти кількома схемами.
- Кілька індивідуальних принципалів, що становлять роль або групу Windows, можуть володіти одною схемою.

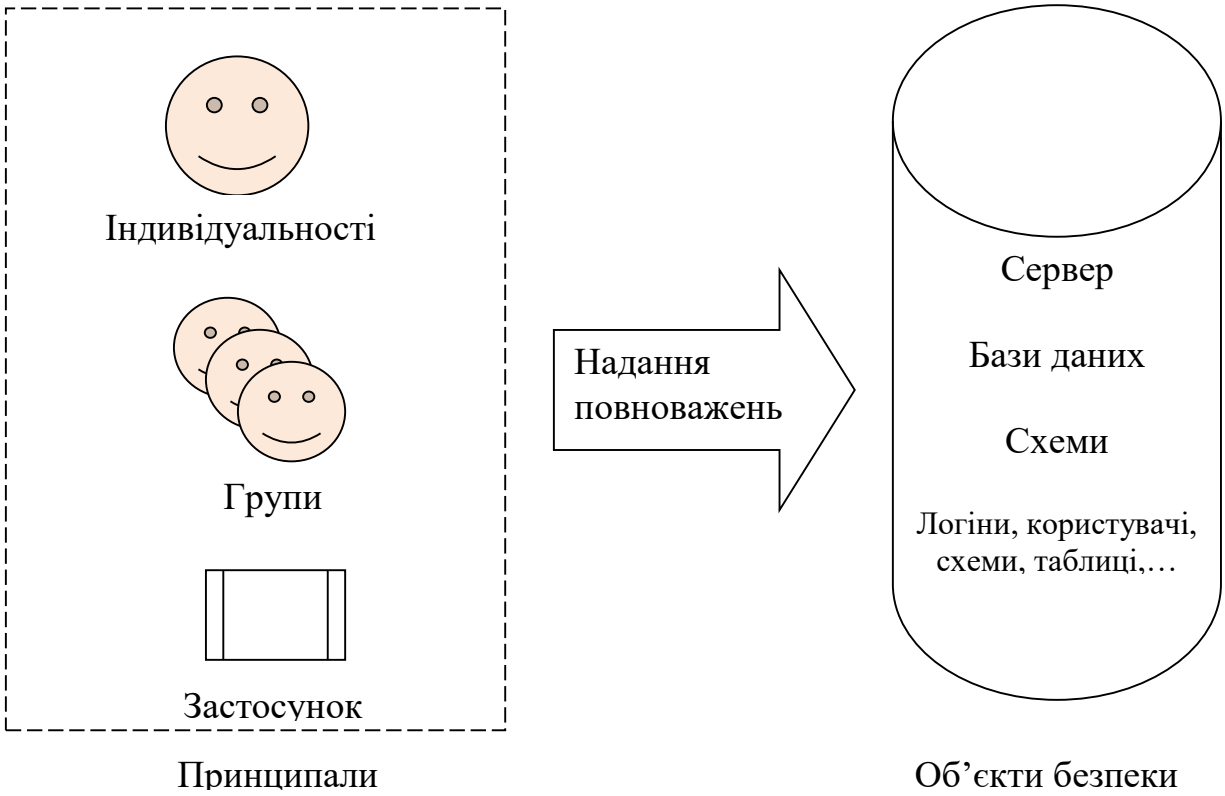

Рисунок 10 – Модель авторизації Database Engine Ms SQL Server

Модель авторизації розділяє світ на принципалів та об’єкти безпеки. Кожний об’єкт безпеки має пов’язані з ним повноваження, які можуть бути надані принципалу.

Взаємозв’язок понять контроля доступу до сервера Ms SQL Server можна відобразити наступною ER-діаграмою:

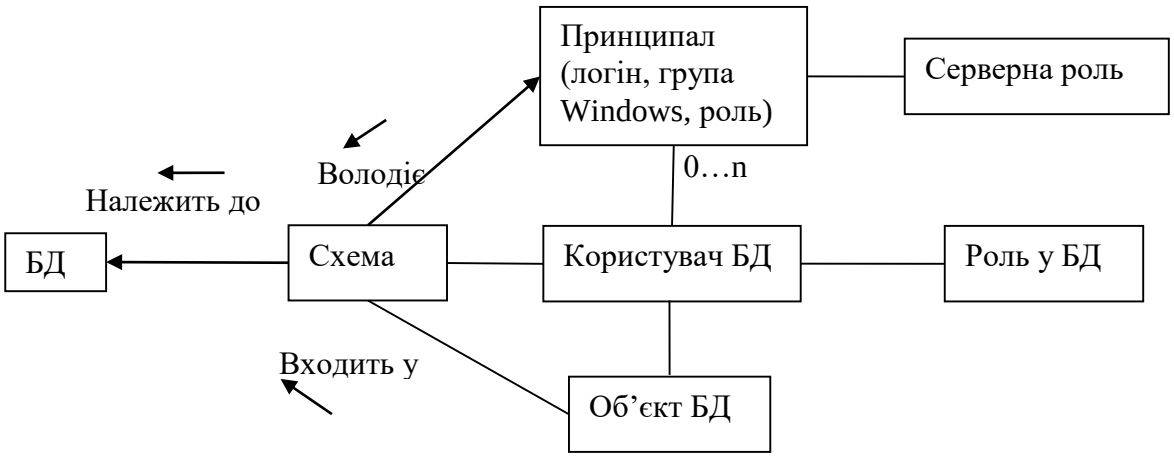

Рисунок 11 – ER-діаграма взаємозв’язку понять контроля доступу до сервера Ms SQL Server 2008

У схеми є лише один принципал-власник, що відображено на рис. 11. Але цей принципал може надати повноваження кільком іншим принципалам на об’єкти схеми. Принципал на рівні БД може бути користувачем БД, роллю або роллю застосунку. Власність на схему задається під час створення схеми оператором:
CREATE SCHEMA <ім’я схеми> AUTHORIZATION <логін принципала>

Оператори авторизації Transact-SQL

Оператор	*Призначення*	*Синтаксис*

GRANT	Надає повноваження доступу до об’єктів безпеки	`GRANT { ALL [ PRIVILEGES ] }`
		`  | permission [ ( column [ ,...n ] ) ] [ ,...n ]`
		`  [ ON [ class :: ] securable ] TO principal [ ,...n ]`
		`  [ WITH GRANT OPTION ] [ AS principal ]`

DENY	Забороняє доступ до об’єктів безпеки	`DENY { ALL [ PRIVILEGES ] }`
		`  | permission [ ( column [ ,...n ] ) ] [ ,...n ]`
		`  [ ON [ class :: ] securable ] TO principal [ ,...n ]`
		`  [ CASCADE] [ AS principal ]`

REVOKE	Відміняє раніше надані дозволи або заборони	`REVOKE [ GRANT OPTION FOR ]`
		`  {`
		`  [ ALL [ PRIVILEGES ] ]`
		`  |`
		`  permission [ ( column [ ,...n ] ) ] [ ,...n ]`
		`  }`
		`  [ ON [ class :: ] securable ]`
		`  { TO | FROM } principal [ ,...n ]`
		`  [ CASCADE] [ AS principal ]`

permission	Ім’я повноваження, як-от: SELECT, INSERT, EXECUTE, ALTER, VIEW DEFINITION, … (усього 12)
Column	Визначає ім’я колонки, на яку надаються дозволи
Class	Конкретизує клас securable, на який надається дозвіл. Визначник можливості "::" потрібний. Клас може бути SCHEMA, DATABASE або OBJECT, в останньому випадку об’єктом

	безпеки Securable може бути таблиця, розріз, збережена процедура
Securable	Визначає об’єкт безпеки, на який надається дозвіл
TO principal	Ім’я принципала
GRANT OPTION	Вказує, що отримувач повноваження також отримає здатність надати вказане повноваження іншим принципалам.
AS principal	Вказує принципала, від якого виконавець цього запиту набув право надати повноваження.
GRANT OPTION FOR	Видаляє ефект `WITH GRANT OPTION` у відповідному операторі GRANT. Принципал не втрачає свої повноваження, але більше не може надавати їх іншим.

Наприклад, оператори
```
USE sample;
GRANT VIEW DEFINITION ON OBJECT::employee TO peter;
```
означають надання дозволу користувачу `peter` продивлятись метадані таблиці `employee` БД `sample`.

GRANT можна розглядати як «позитивну», а DENY як «негативну» авторизацію. Зазвичай DENY використовується для відміни вже існуючих для групи або ролі повноважень для невеликої кількості членів.

REVOKE відміняє як «позитивні», так і «негативні» повноваження.

Таблиця 1 - Приклади використання операторів авторизації

GRANT CREATE TABLE, CREATE PROCEDURE TO Peter, Mary	Користувачі Peter, Mary можуть створювати таблиці і процедури
GRANT SELECT ON employee TO Peter, Mary	Користувачі Peter, Mary можуть читати рядки таблиці employee
GRANT UPDATE ON works_on (emp_no, enter_date) TO Peter	Користувач Peter може змінювати 2 стовпця таблиці works_on
GRANT VIEW DEFINITION ON SCHEMA::dbo TO Peter	Peter отримує доступ до метаданих усіх об’єктів схеми dbo поточної БД
GRANT SELECT ON project TO public; DENY SELECT ON project TO Peter, Mary	Усім користувачам БД, крім двох, надаються повноваження на читання рядків таблиці project
REVOKE SELECT ON project TO public;	Відміняється повноваження ролі public на читання таблиці project. Існуючі «негативні» повноваження для користувачів Peter, Mary (попередній рядок таблиці) не відміняються!

Доступ до Oracle

Доступ до сервера Oracle ґрунтується на наступних поняттях.

Таблиця 2 – Базові поняття доступу до сервера Oracle

Привілей	Існують привілеї двох різних видів: об'єктні і системні. Об'єктний привілей (object privilege) дозволяє виконання визначених операції над конкретним об'єктом (наприклад, над таблицею) – як-от SELECT,

	ALTER, EXECUTE (всього 12). Системний привілей (system privilege) дозволяє виконання операцій над цілим класом об'єктів.
Схема = Користувач	Той, кому надаються привілеї
Роль	Сукупність привілеїв, як об'єктних, так і системних.

Щоб виконати над таблицею Oracle деяку операцію, наприклад INSERT або DELETE, необхідно мати відповідні повноваження, які надаються і відміняються за допомогою SQL-команд GRANT і REVOKE. Тобто, команда REVOKE в Oracle виконує те, що DENY в MsSQL.[1]

Розглянемо наступну групу операторів:

```
CREATE ROLE table_query;
GRANT SELECT ON students TO table_query;
GRANT SELECT ON classes TO table_query;
GRANT SELECT ON rooms TO table_query;
```

Роль table_query має привілеї SELECT на три різні таблиці.
Тепер можна надати цю роль іншим користувачам:

```
GRANT table_query TO userA;
GRANT table_query TO userB;
```

Тепер привілеї SELECT на три таблиці мають користувачі userA і userB. Це спрощує адміністрування: адже без застосування ролі довелось би шість разів надавати привілеї користувачам, а за допомогою ролі – лише п’ять.
У Oracle є зумовлена роль PUBLIC (загальна), яка автоматично надається кожному користувачеві. Тому виконання оператора

```
GRANT привілей TO PUBLIC;
```

одночасно надає вказаний привілей всім користувачам Oracle. У Oracle заздалегідь визначений і ряд інших ролей, в які включені часто вживані системні привілеї. Список цих ролей для Oracle наведений в таблиці 3. Наперед визначений користувач Oracle з ім'ям SYSTEM автоматично отримує всі ці ролі.

[1] https://www.techonthenet.com/oracle/grant_revoke.php

Таблиця 3 - Зумовлені системні ролі в Oracle

Ім'я ролі	Надані привілеї
CONNECT	ALTER SESSION, CREATE CLUSTER, CREATE DATABASE LINK, CREATE SEQUENCE, CREATE SESSION, CREATE SYNONYM, CREATE TABLE, CREATE VIEW
RESOURCE	CREATE CLUSTER, CREATE INDEXTYPE, CREATE OPERATOR, CREATE PROCEDURE, CREATE SEQUENCE, CREATE TABLE, CREATE TRIGGER, CREATE TYPE
DBA	Всі системні привілеї (з ADMIN OPTION; тому ці привілеї можуть бути надані знову), EXP_FULL_DATABASE і IMP_FULL_DATABASE
EXP_FULL_DATABASE	ADMINISTER RESOURCE, BACKUP ANY TABLE, EXECUTE ANY PROCEDURE, EXECUTE ANY TYPE, SELECT ANY TABLE, атакож INSERT, UPDATE, DELETE для деяких системних таблиць

Взаємозв’язок понять контроля доступу до сервера Oracle можна відобразити наступною ER-діаграмою:

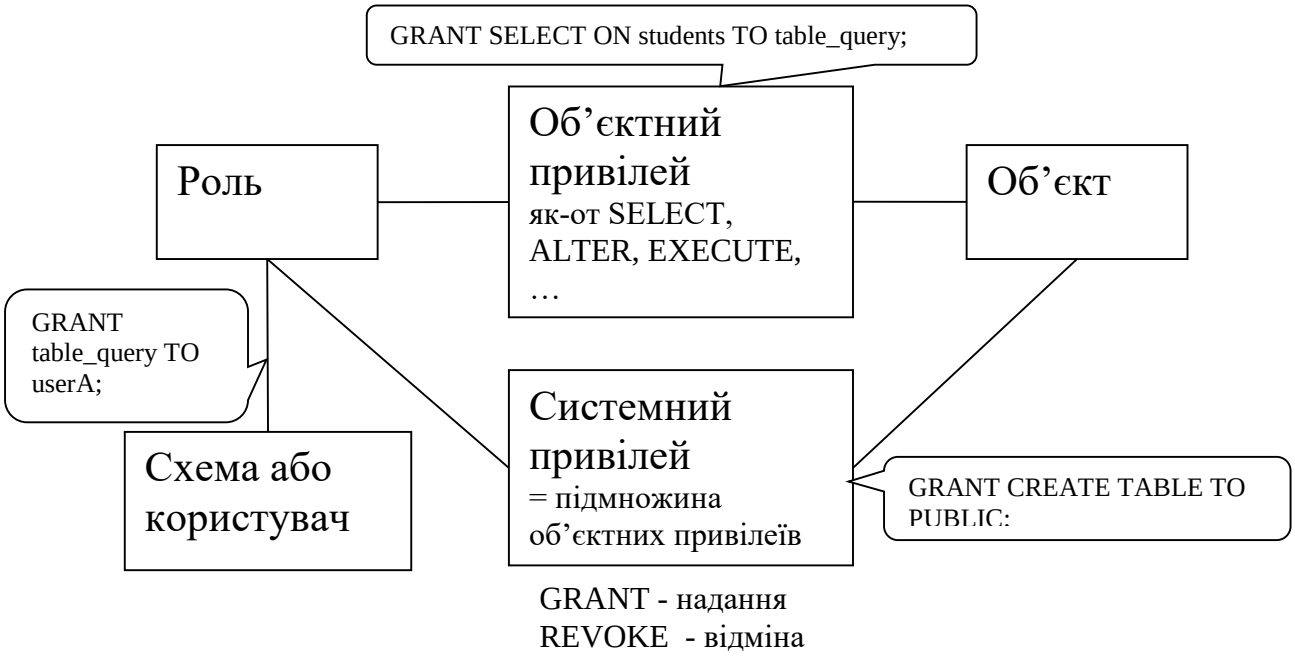

Рисунок 12 – Взаємозв’язок понять контроля доступу до сервера Oracle