**ПРАКТИЧНА РОБОТА №3**

**ОСНОВИ DDL ТА ТИПИ ДАНИХ У T-SQL**

Мета: Ознайомити студентів з основними командами DDL у T-SQL та типами даних у SQL Server.

**1. Введення в DDL:**

DDL, або Data Definition Language (мова визначення даних), – це підмножина SQL, що використовується для визначення та керування структурами даних в реляційних базах даних. Основне призначення DDL – створення, зміна та видалення об'єктів бази даних, таких як таблиці, індекси, об'єднання, тригери тощо.

Основні команди DDL включають:

**1. CREATE: Використовується для створення нових об'єктів у базі даних.**

Наприклад:

**`CREATE DATABASE` – створює нову базу даних.**

CREATE DATABASE MyNewDatabase;

**`CREATE TABLE` – створює нову таблицю.**

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DateOfBirth DATE

);

**`CREATE INDEX` – створює новий індекс.**

Індекс створюється для покращення швидкості пошуку. У наступному прикладі створюється індекс для стовпця LastName у таблиці Employees.

CREATE INDEX idx\_Employees\_LastName ON Employees (LastName);

**2. ALTER: Дозволяє модифікувати існуючі об'єкти бази даних.** За допомогою цієї команди можна додавати, змінювати або видаляти стовпці в таблицях, змінювати типи даних стовпців та інше. Наприклад:

`ALTER TABLE` – змінює структуру існуючої таблиці.

Додавання нового стовпця `Email` до таблиці `Employees`:

ALTER TABLE Employees ADD Email VARCHAR(100);

**3. DROP: Використовується для видалення об'єктів з бази даних.** Ця команда повністю видаляє об'єкт, тому її слід використовувати обережно. Наприклад:

`DROP DATABASE` – видаляє базу даних.

DROP DATABASE MyNewDatabase;

`DROP TABLE` – видаляє таблицю.

DROP TABLE Employees;

`DROP INDEX` – видаляє індекс.

DROP INDEX idx\_Employees\_LastName ON Employees;

**4. TRUNCATE: Ця команда видаляє всі записи з таблиці, але залишає саму таблицю недоторканою.** Відрізняється від `DELETE` тим, що не записує кожне видалення в журнал транзакцій, тому виконується швидше.

TRUNCATE TABLE Employees;

Хоча DDL фокусується на структурі даних, не слід плутати його з DML (Data Manipulation Language), який зосереджений на вставці, оновленні та видаленні конкретних даних у таблицях (наприклад, команди `SELECT`, `INSERT`, `UPDATE` та `DELETE`).

Важливо зауважити, що в залежності від системи управління базами даних (СУБД) можливі додаткові DDL-команди або варіації використання стандартних команд.

**2. Типи даних у SQL Server:**

- Числові типи: `INT`, `DECIMAL`, `FLOAT` тощо.

- Символьні типи: `CHAR`, `VARCHAR`, `TEXT` тощо.

- Типи даних дати та часу: `DATETIME`, `DATE`, `TIME` тощо.

- Спеціальні типи: `XML`, `GUID` та ін.

Звісно, давайте розглянемо детальніше основні типи даних у T-SQL:

2.1. Числові типи:

- INT: Це цілочисельний тип даних. Має розмір 4 байти і може містити значення від -2^31 (-2,147,483,648) до 2^31-1 (2,147,483,647).

- DECIMAL (NUMERIC): Дійсні числа з фіксованою крапкою. Вони приймають два аргументи: загальну кількість цифр і кількість цифр після десяткової крапки. Наприклад, `DECIMAL(10,2)` може представляти число з 8 цифрами перед крапкою та 2 цифрами після крапки.

- FLOAT: Дійсні числа з рухомою крапкою. FLOAT(24) називається REAL. Числа FLOAT можуть мати великий діапазон, але їх точність є обмеженою.

2.2. Символьні типи:

- CHAR: Символьний рядок фіксованої довжини. Наприклад, `CHAR(10)` має довжину 10 символів.

- VARCHAR: Символьний рядок змінної довжини. Наприклад, `VARCHAR(100)` може містити до 100 символів, але фактичний розмір даних буде залежати від довжини збереженого рядка.

- NVARCHAR - це тип даних у T-SQL, призначений для зберігання рядкових значень, які могут включати символи з різних мовних наборів.

Основні характеристики NVARCHAR:

1. Юнікод: Основна відмінність між NVARCHAR та VARCHAR полягає в тому, що NVARCHAR підтримує Юнікод, дозволяючи представляти символи з майже всіх мовних систем світу.

2. Довжина: За замовчуванням, NVARCHAR може зберігати до 4000 символів. Однак існує також спеціальна форма - NVARCHAR(MAX), яка дозволяє зберігати до приблизно 2^30-1 символів.

3. Розмір зберігання: Оскільки NVARCHAR використовує Юнікод, кожен символ зазвичай займає 2 байти. Це вдвічі більше, ніж у відповідного VARCHAR типу для символів латиниці.

4. Використання: NVARCHAR ідеально підходить для даних, які можуть включати символи з різних мов, таких як імена користувачів, адреси та інша інформація в міжнародних додатках.

5. Сумісність: Оскільки NVARCHAR підтримує Юнікод, він може бути сумісний з іншими системами та стандартами, які також використовують Юнікод.

При використанні NVARCHAR, важливо враховувати додаткові вимоги до розміру зберігання, але його здатність представляти широкий спектр символів робить його незамінним для багатомовних застосунків.

- TEXT: Тип для зберігання великих текстових рядків. Відмінний від CHAR і VARCHAR тим, що може містити дуже велику кількість тексту.

2.3. Типи даних дати та часу:

- DATETIME: Зберігає дату та час до секунди. Діапазон від 1753 до 9999 року.

- DATE: Зберігає тільки дату.

- TIME: Зберігає тільки час.

2.4. Спеціальні типи:

- XML: Зберігає XML-документи.

- UNIQUEIDENTIFIER: Зберігає унікальний глобальний ідентифікатор (GUID). Наприклад, "6F9619FF-8B86-D011-B42D-00C04FC964FF".

Існують і інші типи даних у T-SQL, такі як BIT, MONEY, SMALLINT, TINYINT, NVARCHAR, NCHAR тощо. Однак вищенаведені є найпоширенішими та часто використовуваними.

**ЗАВДАННЯ**

Всі завдання виконувати в одному вікні запитів

Кожне завдання супроводжувати коментарем:

/\*

Довільний коментар

\*/

Після виконання всіх завдань зберегти файл з розширення sql

1. Створення бази даних "STUDENTS":

CREATE DATABASE STUDENTS;

Перейдіть до бази даних "STUDENTS":

USE STUDENTS;

2. Створення таблиць:

a. Створіть таблицю "PersonalInfo":

CREATE TABLE PersonalInfo (

StudentID INT PRIMARY KEY,

FirstName NVARCHAR(50),

LastName NVARCHAR(50),

DateOfBirth DATE

);

b. Створіть таблицю "AcademicInfo":

CREATE TABLE AcademicInfo (

RecordID INT PRIMARY KEY,

StudentID INT FOREIGN KEY REFERENCES PersonalInfo(StudentID),

Faculty NVARCHAR(100),

Curator NVARCHAR(100),

EnrollmentYear INT

);

c. Створіть таблицю "ContactInfo":

CREATE TABLE ContactInfo (

ContactID INT PRIMARY KEY,

StudentID INT FOREIGN KEY REFERENCES PersonalInfo(StudentID),

Email NVARCHAR(100),

PhoneNumber NVARCHAR(15)

);

3. Операції зі структурою таблиці "ContactInfo":

a. Додайте стовбець "Address":

ALTER TABLE ContactInfo ADD Address NVARCHAR(200);

b. Видаліть стовбець "PhoneNumber":

ALTER TABLE ContactInfo DROP COLUMN PhoneNumber;

c. Змініть ім'я стовбця "Email" на "EmailAddress":

EXEC sp\_rename 'ContactInfo.Email', 'EmailAddress', 'COLUMN';

4. Змінення типу даних стовбця:

Змініть тип даних для стовбця "Faculty" у таблиці "AcademicInfo" на NVARCHAR(150):

ALTER TABLE AcademicInfo

ALTER COLUMN Faculty NVARCHAR(150);

5. Додайте нову таблицю "Extracurricular":

CREATE TABLE Extracurricular (

ActivityID INT PRIMARY KEY,

StudentID INT FOREIGN KEY REFERENCES PersonalInfo(StudentID),

ActivityName NVARCHAR(100),

JoinDate DATE

);

Після перевірки того, що таблиця була створена правильно, видаліть її:

DROP TABLE Extracurricular;

6. Створення бази даних "TEACHER":

CREATE DATABASE TEACHER;

Переконайтеся, що база даних була створена (можна здійснити перегляд у SQL Server Management Studio або іншому клієнті).

Після перевірки видаліть базу даних:

DROP DATABASE TEACHER;

7. Переключення на потрібну базу даних

USE STUDENTS;

Додати три рядки даних до таблиці "PersonalInfo"

INSERT INTO PersonalInfo (StudentID, FirstName, LastName, DateOfBirth)

VALUES

(1, 'Олександр', 'Петров', '2000-05-15'),

(2, 'Марія', 'Іваненко', '2001-03-22'),

(3, 'Василь', 'Коваленко', '1999-10-10');