**ЛАБОРАТОРНА РОБОТА №4**

ОБМЕЖЕННЯ ЦІЛІСНОСТІ ДАНИХ У T-SQL

Мета: надання студентам практичних навичок у використанні обмежень цілісності даних в T-SQL. Студенти ознайомляться з різними типами обмежень, такими як PRIMARY KEY, FOREIGN KEY, UNIQUE, CHECK, DEFAULT та NOT NULL, та навчаться застосовувати їх для забезпечення точності, повноти та надійності даних у реляційній базі даних. Вивчення цих обмежень допоможе студентам розуміти, як вони можуть контролювати та керувати даними, що зберігаються, а також забезпечити валідність та відповідність даних бізнес-правилам і вимогам. Це також включає практичне використання цих обмежень у різних сценаріях, що дозволить студентам краще зрозуміти можливості та обмеження SQL в контексті реального проектування баз даних.

**ТЕОРЕТИЧНІ ОСНОВИ**

**1. PRIMARY KEY**

PRIMARY KEY визначає унікальний ідентифікатор для кожного рядка в таблиці. Він не може мати значення NULL і повинен бути унікальним у всій таблиці. Кожна таблиця може мати тільки один PRIMARY KEY, який може складатися з одного або декількох полів (так званий складений ключ).

*Приклад встановлення обмеження*

CREATE TABLE Students (

StudentID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50)

);

**2. FOREIGN KEY**

FOREIGN KEY встановлює відношення між двома таблицями. Це зв'язок, який визначається між PRIMARY KEY однієї таблиці та колонкою іншої таблиці. Обмеження FOREIGN KEY забезпечує цілісність даних шляхом запобігання дій, які можуть розірвати цей зв'язок.

*Приклад встановлення обмеження*

CREATE TABLE Enrollments (

StudentID INT,

CourseID INT,

FOREIGN KEY (StudentID) REFERENCES Students(StudentID)

);

**3. UNIQUE**

UNIQUE обмеження забезпечує унікальність значень у колонці або наборі колонок. Відрізняючись від PRIMARY KEY, UNIQUE дозволяє мати NULL значення, але всі не-NULL значення мають бути унікальними.

*Приклад встановлення обмеження*

CREATE TABLE Users (

UserID INT PRIMARY KEY,

Username VARCHAR(50) UNIQUE

);

**4. CHECK**

CHECK обмеження на колонку або таблицю вимагає, щоб певні умови були істинними для значення в колонці. Це може бути діапазон значень, певні строки або будь-які інші умови, які можна виразити через логічний вираз.

*Приклад встановлення обмеження*

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

Price MONEY CHECK (Price > 0)

);

**5. DEFAULT**

DEFAULT обмеження надає стандартне значення для колонки, якщо при вставці рядка значення не вказане. Це корисно для визначення "попередніх налаштувань" для полів.

*Приклад встановлення обмеження*

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

OrderDate DATE DEFAULT GETDATE() -- Встановлює поточну дату за замовчуванням

);

**6. NOT NULL**

NOT NULL обмеження вказує, що колонка не може мати значення NULL. Це гарантує, що кожен рядок в таблиці матиме дійсне значення в цій колонці.

*Приклад встановлення обмеження*

CREATE TABLE Members (

MemberID INT PRIMARY KEY,

LastName VARCHAR(50) NOT NULL -- Прізвище повинно бути завжди вказане

);

Ці обмеження (PRIMARY KEY; FOREIGN KEY; UNIQUE; CHECK; DEFAULT; NOT NULL) використовуються для забезпечення точності та надійності даних, що зберігаються в базі даних. Вони є фундаментальними компонентами реляційної моделі баз даних і важливими інструментами для будь-якого розробника баз даних.

**ЗАВДАННЯ**

**Блок 1: Додавання обмежень після створення таблиць**

***Завдання 1.1: Створення бази даних та таблиць***

- Створіть базу даних `UniversityDB`.

CREATE DATABASE UniversityDB;

USE UniversityDB;

- Створіть таблицю `Departments` без обмежень.

CREATE TABLE Departments (

DepartmentID INT NOT NULL,

DepartmentName VARCHAR(100)

);

- Створіть таблицю `Professors` без обмежень.

CREATE TABLE Professors (

ProfessorID INT NOT NULL,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Age TINYINT,

DepartmentID INT,

Email VARCHAR(100)

);

***Завдання 1.2: Додавання обмежень до існуючих таблиць***

*- Додайте PRIMARY KEY до `Departments` та `Professors`.*

ALTER TABLE Departments ADD CONSTRAINT PK\_Department PRIMARY KEY (DepartmentID);

ALTER TABLE Professors ADD CONSTRAINT PK\_Professor PRIMARY KEY (ProfessorID);

*- Додайте UNIQUE обмеження на `Email` в `Professors`.*

ALTER TABLE Professors ADD CONSTRAINT UQ\_Professor\_Email UNIQUE (Email);

*- Додайте FOREIGN KEY обмеження до `Professors`, щоб пов'язати `DepartmentID` з `Departments`.*

ALTER TABLE Professors ADD CONSTRAINT FK\_Professor\_Department FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID);

*- Додайте CHECK обмеження на `*Age*` в `* Professors*`, щоб значення віку не було менше 20.*

ALTER TABLE Professors ADD CONSTRAINT CK\_Age CHECK (Age > = 20);

*- Додайте NOT NULL обмеження на `FirstName` і `LastName` в `Professors`.*

ALTER TABLE Professors ALTER COLUMN FirstName VARCHAR(50) NOT NULL;

ALTER TABLE Professors ALTER COLUMN LastName VARCHAR(50) NOT NULL;

*- Додайте DEFAULT значення для `DepartmentName` в `Departments`.*

ALTER TABLE Departments ADD CONSTRAINT DF\_DepartmentName DEFAULT 'General' FOR DepartmentName;

***Завдання 1.3 Введення та тестування даних***

*- Введіть дані до `Departments`. Якщо `DepartmentName` не вказано, використовуйте значення за замовчуванням.*

INSERT INTO Departments (DepartmentID) VALUES (1); -- Застосується DEFAULT значення 'General'

*- Введіть дані до `Professors`, включаючи їх прив'язку до відділів через `DepartmentID`.*

INSERT INTO Professors (ProfessorID, FirstName, LastName, Age, DepartmentID, Email)

VALUES (1, 'John', 'Smith', 57, 1, 'john.smith@university.com');

*- Спробуйте вставити запис до `Professors` з існуючим `Email`, щоб перевірити UNIQUE обмеження.*

INSERT INTO Professors (ProfessorID, FirstName, LastName, DepartmentID, Email)

VALUES (2, 'Jane', 'Doe', 1, 35, 'john.smith@university.com'); -- Це викличе помилку через UNIQUE обмеження

*- Спробуйте вставити запис до `Professors` з існуючим віком менше 20, щоб перевірити CHECK обмеження.*

INSERT INTO Professors (ProfessorID, FirstName, LastName, DepartmentID, Email)

VALUES (3, 'Bob', 'Doe', 1, 17, 'bob@university.com'); -- Це викличе помилку через CHECKобмеження

***Завдання 1.4. Очищення***

- Видаліть таблиці `Professors` та `Departments`.

DROP TABLE Professors;

DROP TABLE Departments;

- Видаліть базу даних `UniversityDB`.

DROP DATABASE UniversityDB;

**Блок 2: Створення таблиць з обмеженнями**

***Завдання 2.1: Створення бази даних і таблиць***

*- Створіть базу даних `CompanyDB`.*

CREATE DATABASE CompanyDB;

USE CompanyDB;

*- Створіть таблицю `Employees` з PRIMARY KEY, NOT NULL, UNIQUE, і DEFAULT обмеженнями.*

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

Email VARCHAR(100) UNIQUE,

Position VARCHAR(50) DEFAULT 'Junior Developer'

);

*- Створіть таблицю `Departments` з PRIMARY KEY та CHECK обмеженнями.*

CREATE TABLE Departments (

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(100) NOT NULL CHECK (DepartmentName <> ''),

Budget DECIMAL(10, 2) CHECK (Budget > 0),

DefaultRoomNumber INT DEFAULT 101

);

*- Створіть таблицю `Assignments` з FOREIGN KEY, що вказує на `Employees` та `Departments`.*

CREATE TABLE Assignments (

AssignmentID INT PRIMARY KEY,

EmployeeID INT NOT NULL,

DepartmentID INT NOT NULL,

AssignmentDate DATE DEFAULT GETDATE(),

FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID),

FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)

);

***Завдання 2.2. Введення та тестування даних***

*- Введіть дані до `Departments` і `Employees`, потім створіть призначення у `Assignments`.*

INSERT INTO Departments (DepartmentID, DepartmentName, Budget)

VALUES (1, 'Human Resources', 50000);

INSERT INTO Employees (EmployeeID, FirstName, LastName, Email)

VALUES (1, 'John', 'Smith', 'john.smith@company.com');

INSERT INTO Assignments (AssignmentID, EmployeeID, DepartmentID)

VALUES (1, 1, 1);

*- Спробуйте вставити запис до `Assignments` з неіснуючим `EmployeeID` або `DepartmentID`, щоб перевірити FOREIGN KEY обмеження.*

INSERT INTO Assignments (AssignmentID, EmployeeID, DepartmentID)

VALUES (2, 999, 1); -- Це викличе помилку через FOREIGN KEY обмеження

***Завдання 2.3: Очищення***

*- Видаліть таблиці `Assignments`, `Employees` та `Departments`.*

DROP TABLE Assignments;

DROP TABLE Employees;

DROP TABLE Departments;

*- Видаліть базу даних `CompanyDB`.*

DROP DATABASE CompanyDB;

**ДОДАТКОВІ МАТЕРІАЛИ:**

<https://www.w3schools.com/sql/sql_constraints.asp>

**КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Що таке обмеження PRIMARY KEY і які дві основні умови воно забезпечує?

2. Як створити складений PRIMARY KEY? Надайте приклад коду T-SQL.

3. Поясніть, що таке FOREIGN KEY та як він забезпечує цілісність даних між таблицями.

4. Що станеться, якщо ви спробуєте вставити дані у таблицю, які порушують FOREIGN KEY обмеження?

5. Опишіть, для чого використовується UNIQUE обмеження і чим воно відрізняється від PRIMARY KEY.

6. Наведіть приклад коду T-SQL, який додає обмеження CHECK до існуючої таблиці.

7. Як можна використовувати DEFAULT обмеження при створенні нової таблиці? Дайте приклад.

8. Яке обмеження ви б використали, щоб гарантувати, що колонка `Email` в таблиці `Customers` завжди містить унікальне значення?

9. Які дії ви можете виконати, якщо потрібно змінити або видалити обмеження, яке вже було застосоване до таблиці?

10. Чому важливо встановлювати обмеження цілісності даних в базі даних?

11. Чим відрізняється обмеження NOT NULL від обмеження CHECK?

12. Як можна модифікувати обмеження DEFAULT для існуючої колонки? Надайте приклад T-SQL команди.

13. Що таке каскадні операції в контексті FOREIGN KEY обмежень і які проблеми можуть вони вирішити?