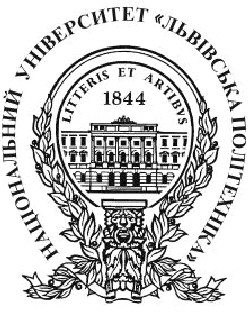
Міністерство освіти і науки України НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»



Кафедра ЕОМ

Звіт

З лабораторної роботи №5, варіант 7 З дисципліни «Організація баз даних»

На тему: «Створення та використання представлень, функцій та тригерів»

Виконав: ст. гр. КІ-204

Говдиш Денис

Прийняв: викладач

Трач І.Б

Львів – 2023

**Мета:** отримати навики використання функцій, представлень та тригерів

**Завдання:**

**Изображение выглядит как текст

Автоматически созданное описание**

**Код**

use tour\_agency;

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name VARCHAR(50),

Email VARCHAR(100),

Phone VARCHAR(20)

);

CREATE TABLE Bookings (

BookingID INT PRIMARY KEY,

Customer\_booking\_ID INT,

TourID INT,

BookingDate DATE,

CustomerID int,

Foreign key (CustomerID) references Customers(CustomerID)

);

CREATE TABLE TourDates (

TourDateID INT PRIMARY KEY,

Tour\_dates\_ID INT,

Date DATE,

AvailableSlots INT

-- Foreign key (Tour\_dates\_ID) references TourGuides (Tour\_guides\_ID)

);

CREATE TABLE TourGuides (

GuideID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100),

Tour\_guides\_ID INT

-- Foreign key (Tour\_guides\_ID) references TourDates (Tour\_dates\_ID)

);

CREATE TABLE BookingTourDates (

BookingID INT,

TourDateID INT,

PRIMARY KEY (BookingID, TourDateID),

FOREIGN KEY (BookingID) REFERENCES Bookings(BookingID),

FOREIGN KEY (TourDateID) REFERENCES TourDates(TourDateID)

);

INSERT INTO Customers (CustomerID, Name, Email, Phone)

VALUES

(1, 'John Smith', 'john.smith@example.com', '0961234567'),

(2, 'Jane Doe', 'jane.doe@example.com', '0962345678'),

(3, 'Bob Johnson', 'bob.johnson@example.com', '0963456789'),

(4, 'Alice Lee', 'alice.lee@example.com', '0964567890'),

(5, 'Tom Wilson', 'tom.wilson@example.com', '0965678901'),

(6, 'Sarah Brown', 'sarah.brown@example.com', '0966789012'),

(7, 'David Davis', 'david.davis@example.com', '0967890123'),

(8, 'Emily Smith', 'emily.smith@example.com', '0968901234'),

(9, 'Oliver Brown', 'oliver.brown@example.com', '0969012345'),

(10, 'Grace Taylor', 'grace.taylor@example.com', '0960123456');

INSERT INTO Bookings (BookingID, BookingDate, TourID)

VALUES

(1, '2022-05-15', 123),

(2, '2022-06-20', 456),

(3, '2022-07-12', 789),

(4, '2022-08-05', 123),

(5, '2022-09-01', 456),

(6, '2022-10-10', 789);

INSERT INTO TourGuides (GuideID, FirstName, LastName, Email)

VALUES

(1, 'John', 'Smith', 'john.smith@example.com'),

(2, 'Jane', 'Doe', 'jane.doe@example.com'),

(3, 'Bob', 'Johnson', 'bob.johnson@example.com'),

(4, 'Alice', 'Lee', 'alice.lee@example.com'),

(5, 'Tom', 'Wilson', 'tom.wilson@example.com'),

(6, 'Sarah', 'Brown', 'sarah.brown@example.com'),

(7, 'David', 'Davis', 'david.davis@example.com');

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(50),

Price int,

Category VARCHAR(50)

)

INSERT INTO Products (ProductID, ProductName, Price, Category)

VALUES (1, 'Chocolate Bar', 20, 'Snacks'),

(2, 'Laptop', 14000, 'Electronics'),

(3, 'T-Shirt', 100, 'Clothing'),

(4, 'Headphones', 6000, 'Electronics'),

(5, 'Sneakers', 50, 'Clothing');

Створення представлень

CREATE VIEW Bookings\_info AS

SELECT

Bookings.BookingID,

Bookings.BookingDate,

Bookings.Customer\_booking\_ID,

Bookings.CustomerID,

Bookings.TourID

FROM Bookings

LEFT JOIN Customers

ON Bookings.BookingID = Customers.CustomerID;

GO

CREATE VIEW customer\_info As

SELECT Customers.CustomerID,Customers.Email as customer\_MAIL,Customers.Phone as phone\_ukr

FROM Customers

JOIN Bookings on Customers.CustomerID = Bookings.BookingID;

GO

CREATE VIEW TOURDATES\_INFO AS

SELECT TourGuides.FirstName,TourGuides.LastName,TourGuides.GuideID,COUNT(TourGuides.GuideID)AS count\_ID

FROM TourGuides

GROUP BY TourGuides.FirstName,TourGuides.LastName,TourGuides.GuideID;

Додавання стовпця date і тригеру який автоматично заповнює це поле під час INSERT операції:

ALTER TABLE TourGuides ADD date datetime;

CREATE TRIGGER Tour\_guides\_trigers

ON TourGuides

AFTER INSERT

AS

BEGIN

UPDATE TourGuides

SET date = GETDATE()

WHERE GuideID IN (SELECT GuideID FROM inserted);

END;

3 запити із довільними вбудованими функціями:

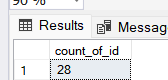
SELECT Customers.Name, Customers.Email, Customers.Phone

FROM Customers

INNER JOIN Bookings ON Customers.CustomerID = Bookings.CustomerID

WHERE Bookings.BookingID < 4;

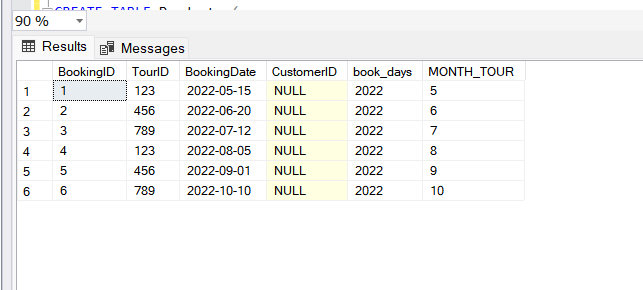
SELECT Sum(TourGuides.GuideID) AS count\_of\_id FROM TourGuides;



use tour\_agency;

SELECT BookingID,TourID,BookingDate,CustomerID, year(BookingDate) as book\_days,MONTH (BookingDate) AS MONTH\_TOUR

FROM Bookings;



\

Своя таблична і скалярна функція

CREATE FUNCTION Get\_Country\_By\_City (@City VARCHAR(50))

RETURNS TABLE

AS

RETURN

(

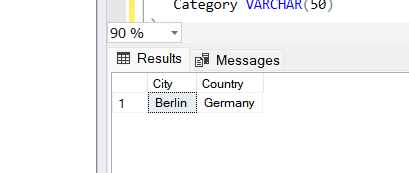
SELECT City, Country

FROM people

WHERE City = @City

);

SELECT \* FROM Get\_Country\_By\_City('Berlin');



Можна викликати так:

CREATE FUNCTION dbo.GetProductPrice (@ProductName VARCHAR(50))

RETURNS int

AS

BEGIN

DECLARE @Price int

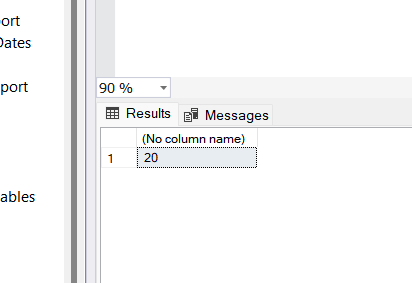
SELECT @Price = Price

FROM Products

WHERE ProductName = @ProductName

RETURN @Price

END



**Контрольні питання:**

1. Функції в SQL Server використовуються для обробки даних та повернення результатів у вигляді значень. Вони можуть використовуватись для створення складних запитів, що об'єднують дані з кількох таблиць та виконують обчислення над цими даними. Функції можуть приймати параметри та повертати різні типи даних, такі як числа, рядки, дати тощо.
2. У SQL Server існує декілька типів табличних функцій:

* Inline-функції: ці функції повертають таблицю та використовуються як підзапит у SELECT, UPDATE або DELETE запитах.
* Multi-Statement Table-Valued-функції: ці функції складаються з кількох SQL запитів та повертають результат у вигляді таблиці.
* Table-Valued Parameters-функції: ці функції приймають вхідні параметри у вигляді таблиць та повертають результат також у вигляді таблиць.

1. Функції користувача (user-defined functions) - це функції, які можна створювати користувачами. Вони дозволяють виконувати власні обчислення та операції над даними, що зберігаються в базі даних. Функції користувача можуть бути скалярними (повертають одне значення) або табличними (повертають таблицю).

**Висновок:** У цій лабораторній роботі я перевірив свої навички з використання функцій, представлень та тригерів.