Abylkas Saginov Karaganda Technical University

Information Technologies and Security Department

**Laboratory work №8**

**Discipline**: Database management systems

**Topic**: Creating Database Triggers

**Prepared by:**

Student of SIB-22-5 group

Batyrkhanov Arman

**Approved by:**

Senior Lecturer

Klyuyeva Ye.G.

**Karaganda 2024**

1. **RecordingOrderAcceptance**

CREATE TRIGGER ЗаписьПриказаПриема

ON Работники

AFTER INSERT

AS

INSERT INTO Приказы(табельный\_номер, дата, код\_типа\_приказа, текст\_приказа)

SELECT i.табельный\_номер, GETDATE(), 1, 'Прием на работу'

FROM inserted i;

1. **ProtectionFromDeletionEducation**

CREATE TRIGGER ЗащитаОтУдаленияОбразования

ON Образование

INSTEAD OF DELETE

AS

IF EXISTS (

SELECT 1

FROM deleted d

JOIN Работники r ON d.код\_образования = r.код\_образования

)

RAISERROR ('Нельзя удалить запись об образовании, так как она связана с работниками', 16, 1);

ELSE

DELETE FROM Образование WHERE код\_образования IN (SELECT код\_образования FROM deleted);

1. **StateOverflowCheck**

CREATE TRIGGER ПроверкаПереполненияШтата

ON Штатное\_расписание

AFTER UPDATE

AS

IF EXISTS (

SELECT 1

FROM Штатное\_расписание sr

JOIN Подразделения p ON sr.код\_подразделения = p.код\_подразделения

WHERE sr.общее\_количество < (

SELECT COUNT(\*)

FROM Работники r

WHERE r.код\_подразделения = sr.код\_подразделения

)

)

RAISERROR ('Превышено общее количество сотрудников для подразделения', 16, 1);

ROLLBACK;

1. **FullYears**

CREATE FUNCTION dbo.fn\_ПолныхЛет (@первая\_дата DATE)

RETURNS INT

AS

BEGIN

DECLARE @лет INT;

SET @лет = DATEDIFF(YEAR, @первая\_дата, GETDATE())

- CASE

WHEN DATEADD(YEAR, DATEDIFF(YEAR, @первая\_дата, GETDATE()), @первая\_дата) > GETDATE()

THEN 1

ELSE 0

END;

RETURN @лет;

END;

1. **Calculate Dependents' Allowance**

CREATE FUNCTION dbo.fn\_Рассчитать\_начисления\_на\_иждивенцев (@табельный\_номер ТабельныйНомер)

RETURNS INT

AS

BEGIN

DECLARE @начисления INT;

DECLARE @оклад DECIMAL(18, 2);

DECLARE @кол\_во\_иждивенцев INT;

SELECT @оклад = оклад, @кол\_во\_иждивенцев = кол\_во\_иждивенцев

FROM Работники

WHERE табельный\_номер = @табельный\_номер;

SET @начисления = 0;

DECLARE @I INT = 0;

WHILE @I < @кол\_во\_иждивенцев

BEGIN

SET @начисления = @начисления + (@оклад \* 0.1);

SET @I = @I + 1;

END

RETURN @начисления;

END;

1. **Highly Paid Employees**

CREATE FUNCTION ВысокооплачиваемыеСотрудники(@КодПодразделения INT)

RETURNS TABLE

AS

RETURN

(

SELECT

ФИО AS ИмяСотрудника,

оклад AS Оклад,

дата\_приема AS ДатаПриема

FROM Работники

WHERE код\_подразделения = @КодПодразделения

AND оклад > (

SELECT

AVG(оклад)

FROM Работники

WHERE код\_подразделения = @КодПодразделения

)

);

1. **Employees of the Division**

CREATE FUNCTION СотрудникиПодразделения(@КодПодразделения INT)

RETURNS TABLE

AS

RETURN

(

SELECT

табельный\_номер, ФИО, оклад, дата\_приема

FROM Работники

WHERE код\_подразделения = @КодПодразделения

);

**Control questions**

1. **What is the purpose of user-defined functions?**

User-defined functions in MS SQL Server allow users to encapsulate reusable logic within a single object that can return a value or a table. Their purpose is to simplify complex calculations, enhance code reusability, standardize logic across queries, and improve readability by abstracting repetitive tasks into a callable function.

1. **What are the methods for creating user-defined functions?**

User-defined functions can be created using the CREATE FUNCTION statement. Scalar functions return a single value and are defined with a specific return type. Table-valued functions return a table and can be inline (defined with a single RETURN statement) or multi-statement (containing multiple steps to build the result table).

1. **What is the purpose of triggers?**

Triggers in MS SQL Server are special types of stored procedures that automatically execute in response to specific events on a table or view, such as INSERT, UPDATE, or DELETE. Their purpose is to enforce business rules, maintain referential integrity, automatically audit changes, or perform cascading operations when certain database events occur.

1. **What are the methods for creating triggers?**

Triggers are created using the CREATE TRIGGER statement, specifying the triggering event (AFTER or INSTEAD OF) and the associated SQL logic. For example, a trigger can be defined to run after an INSERT operation to validate or log the inserted data.

1. **What are the classifications of triggers?**

Triggers are classified into AFTER triggers and INSTEAD OF triggers. AFTER triggers execute after the specified event has occurred, while INSTEAD OF triggers override the specified event, allowing for custom logic instead of the standard operation.

1. **What does managing triggers involve?**

Managing triggers includes creating, modifying, or deleting triggers using the CREATE, ALTER, or DROP statements. It also involves enabling or disabling triggers using ENABLE TRIGGER or DISABLE TRIGGER commands, ensuring correct performance, debugging their logic, and auditing their effects to prevent unintended database changes.