Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №4**

По дисциплине «Проектирование баз знаний»

Тема: «Хранимые процедуры, функции и триггеры в MS SQL Server 2005/2008.»

**Выполнил:**

Студент 3 курса

Группы ИИ-23

Романюк А. П.

**Проверил:**

Савонюк В.А.

Брест 2024

**Цель работы:** Изучение и решение заданий по реализации хранимых процедур, функций и триггеровразных видов и фильтров в базах данных.

**Ход работы:**

**Вариант 9**

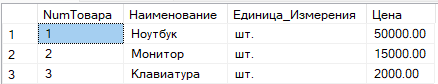
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вариант 9 | **-** | Отгрузка товаров | |  |
| 1 | **№товара** | наименование | ед.измерения | цена |
| 2 | **№накладной** | **№товара** | количество | стоимость |
| 3 | **№накладно1** | отв.лицо | дата отгрузки | средний балл |

Задание.

1. Изучить материал, приведенный в “К лабораторной работе 4.doc”.

2. На основании логической модели (в соответствии с вариантом Лабораторной работы №1) создать и проверить на работоспособность хранимые процедуры, функции и триггеры для каждой из таблиц БД: реализовать хранимые процедуры, функции и триггеры разных видов для каждой таблицы .

**Таблица Товары**

-- Создание таблицы Товары

CREATE TABLE Товары (

[NumТовара] INT PRIMARY KEY,

Наименование VARCHAR(255),

Единица\_Измерения VARCHAR(50),

Цена DECIMAL(10, 2)

);

GO

-- Вставка данных в таблицу Товары

INSERT INTO Товары ([NumТовара], Наименование, Единица\_Измерения, Цена)

VALUES

(1, 'Ноутбук', 'шт.', 50000.00),

(2, 'Монитор', 'шт.', 15000.00),

(3, 'Клавиатура', 'шт.', 2000.00);

**Процедура:**

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание-- Процедура для добавления нового товара

CREATE PROCEDURE InsertNewProduct

@NumТовара INT,

@Наименование VARCHAR(255),

@Единица\_Измерения VARCHAR(50),

@Цена DECIMAL(10, 2)

AS

BEGIN

INSERT INTO Товары ([NumТовара], Наименование, Единица\_Измерения, Цена)

VALUES (@NumТовара, @Наименование, @Единица\_Измерения, @Цена);

END;

**Функция:**

-- Функция для получения цены товара по его номеру

CREATE FUNCTION GetProductPrice(@NumТовара INT)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @Цена DECIMAL(10, 2);

SELECT @Цена = Цена FROM Товары WHERE [NumТовара] = @NumТовара;

RETURN @Цена;

END;

**Триггер:**

-- Триггер для проверки цены товара

CREATE TRIGGER PriceCheck

ON Товары

FOR INSERT

AS

BEGIN

IF EXISTS (SELECT \* FROM inserted WHERE Цена > 100000)

BEGIN

ROLLBACK TRANSACTION;

RAISERROR('Цена не может превышать 100,000!', 16, 1);

END

END;

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание**Таблица Накладные**

-- Создание таблицы Накладные

CREATE TABLE Накладные (

[NumНакладной] INT PRIMARY KEY,

[NumТовара] INT,

Количество INT,

Стоимость DECIMAL(10, 2),

FOREIGN KEY ([NumТовара]) REFERENCES Товары([NumТовара])

);

GO

-- Вставка данных в таблицу Накладные

INSERT INTO Накладные ([NumНакладной], [NumТовара], Количество, Стоимость)

VALUES

(101, 1, 10, 500000.00),

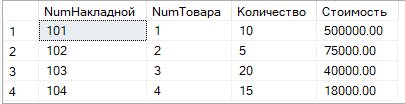
(102, 2, 5, 75000.00),

(103, 3, 20, 40000.00);

GO

**Процедура:**

-- Процедура для вставки новой накладной

CREATE PROCEDURE InsertInvoice

@NumНакладной INT,

@NumТовара INT,

@Количество INT,

@Стоимость DECIMAL(10, 2)

AS

BEGIN

INSERT INTO Накладные ([NumНакладной], [NumТовара], Количество, Стоимость)

VALUES (@NumНакладной, @NumТовара, @Количество, @Стоимость);

END;

GO

**Функция:**

-- Функция для получения общей стоимости накладной

CREATE FUNCTION GetInvoiceTotal(@NumНакладной INT)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @Total DECIMAL(10, 2);

SELECT @Total = SUM(Стоимость) FROM Накладные WHERE [NumНакладной] = @NumНакладной;

RETURN @Total;

END;

GO

**Триггер:**

-- Триггер для проверки количества в накладной

CREATE TRIGGER QuantityCheck

ON Накладные

FOR INSERT, UPDATE

AS

BEGIN

IF EXISTS (SELECT \* FROM inserted WHERE Количество < 1)

BEGIN

ROLLBACK TRANSACTION;

RAISERROR('Количество не может быть меньше 1!', 16, 1);

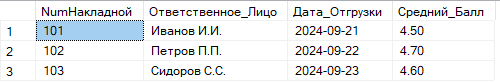
END

END;

GO

**Таблица Информация о накладных:**

-- Создание таблицы Информация\_О\_Накладных

CREATE TABLE Информация\_О\_Накладных (

[NumНакладной] INT,

Ответственное\_Лицо VARCHAR(255),

Дата\_Отгрузки DATE,

Средний\_Балл DECIMAL(3, 2),

PRIMARY KEY ([NumНакладной]),

FOREIGN KEY ([NumНакладной]) REFERENCES Накладные([NumНакладной])

);

GO

-- Вставка данных в таблицу Информация\_О\_Накладных

INSERT INTO Информация\_О\_Накладных ([NumНакладной], Ответственное\_Лицо, Дата\_Отгрузки, Средний\_Балл)

VALUES

(101, 'Иванов И.И.', '2024-09-21', 4.5),

(102, 'Петров П.П.', '2024-09-22', 4.7),

(103, 'Сидоров С.С.', '2024-09-23', 4.6);

GO

**Процедура:**

-- Процедура для обновления среднего балла

CREATE PROCEDURE UpdateAvgScore

@NumНакладной INT,

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание @Средний\_Балл DECIMAL(3, 2)

AS

BEGIN

UPDATE Информация\_О\_Накладных

SET Средний\_Балл = @Средний\_Балл

WHERE [NumНакладной] = @NumНакладной;

END;

GO

**Функция:**

-- Функция для получения ответственного лица по номеру накладной

CREATE FUNCTION GetResponsiblePerson(@NumНакладной INT)

RETURNS VARCHAR(255)

AS

BEGIN

DECLARE @Ответственное\_Лицо VARCHAR(255);

SELECT @Ответственное\_Лицо = Ответственное\_Лицо FROM Информация\_О\_Накладных WHERE [NumНакладной] = @NumНакладной;

RETURN @Ответственное\_Лицо;

END;

GO

**Триггер:**

-- Триггер для логирования изменения среднего балла

CREATE TRIGGER LogAvgScoreChange

ON Информация\_О\_Накладных

AFTER UPDATE

AS

BEGIN

IF UPDATE(Средний\_Балл)

BEGIN

DECLARE @NumНакладной INT, @OldScore DECIMAL(3, 2), @NewScore DECIMAL(3, 2);

SELECT @NumНакладной = inserted.[NumНакладной], @NewScore = inserted.Средний\_Балл, @OldScore = deleted.Средний\_Балл

FROM inserted

JOIN deleted ON inserted.[NumНакладной] = deleted.[NumНакладной];

PRINT 'Средний балл для накладной ' + CAST(@NumНакладной AS VARCHAR) + ' изменен с ' + CAST(@OldScore AS VARCHAR) + ' на ' + CAST(@NewScore AS VARCHAR);

END

END;

GO

**Вывод:** Изучил и решил задания по реализации хранимых процедур, функций и триггеровразных видов и фильтров в базах данных.