Eurasian National University named after L.N. Gumilyov

Faculty of «Information Technologies»

Department of «Information Systems»

**A blue and white logo

Description automatically generated**

**Independent work of the student: 7**

**by discipline** «Development of client server IS»

**Student:** Iskakov Y.K.

**Group:** IS-35

**Checked by:** PhD. Akhmetova. A.Z.

**СРО 7 Модификация данных**

**Упражнение 1: Вставка данных в таблицы**

**Задание 1: Запуск** **SQL Server Management Studio**

1. Откройте среду **SQL Server Management Studio** и подключитесь к серверу, используя проверку подлинности Windows.

**Задание 2: Использование инструкции INSERT для добавления новой записи в таблицу**

1. Создайте инструкцию **INSERT** с предложением **VALUES**, которая добавляет в таблицу **Production.UnitMeasure** новую информацию для существующего продукта.
2. Для добавления записи в таблицу используйте следующие данные:

|  |  |  |
| --- | --- | --- |
| **UnitMeasureCode** | **Name** | **ModifiedDate** |
| F2 | Square Feet | GETDATE() |

1. Выполните запрос и просмотрите результаты.  
   *INSERT INTO Production.UnitMeasure (UnitMeasureCode, Name, ModifiedDate)*

*VALUES ('F2', 'Square Feet', GETDATE());*

*SELECT \* FROM Production.UnitMeasure;*

A screenshot of a computer

Description automatically generated

**Задание 3: Использование инструкции INSERT для добавления нескольких новых записей в таблицу**

1. Создайте инструкцию **INSERT** с предложением **VALUES**, которая добавляет в таблицу **Person.PersonPhone** информацию о новых отделах и менеджерах этих отделов.
2. Используйте одну инструкцию **INSERT** для добавления в таблицу 2-х записей.
3. Для добавления записей в таблицу используйте следующие данные:

|  |  |  |  |
| --- | --- | --- | --- |
| **BusinessEntityID** | **PhoneNumber** | **PhoneNumberTypeID** | **ModifiedDate** |
| 1705 | 864-555-2101 | 3 | GETDATE() |
| 1706 | 712-555-0118 | 1 | GETDATE() |

1. Выполните запрос и просмотрите результаты.

*INSERT INTO Person.PersonPhone (BusinessEntityID, PhoneNumber, PhoneNumberTypeID, ModifiedDate)*

*VALUES (1705, '864-555-2101', 3, GETDATE()),*

*(1706, '712-555-0118', 1, GETDATE());*

*SELECT \* FROM Person.PersonPhone;*

A screenshot of a computer

Description automatically generated

**Упражнение 2: Удаление данных из таблиц**

**Задание 1: Использование инструкции DELETE с предложением WHERE**

* Создайте инструкцию **DELETE**, которая удаляет из таблицы **Production.ProductCostHistory** данные о продуктах, стоимость которых (поле **StandardCost**) превышает $1000.00.
* Выполните запрос и просмотрите результаты.

*DELETE FROM Production.ProductCostHistory*

*WHERE StandardCost > 1000.00;*

*SELECT COUNT(\*) FROM Production.ProductCostHistory;*

*SELECT \* FROM Production.ProductCostHistory;*

A screenshot of a computer

Description automatically generated

**Задание 2: Использование инструкции DELETE с предложением FROM**

* Создайте инструкцию **DELETE**, которая удаляет из таблицы **Sales.SalesPersonQuotaHistory** данные о квоте продаж для продавцов, для которых продажи на текущий момент (поле **SalesYTD** таблицы **Sales.SalesPerson**) превышают $1000000.00.
* Используйте предложение **FROM** в инструкции **DELETE** для объединения таблиц **Sales.SalesPersonQuotaHistory** и **Sales.SalesPerson** по полю **BusinessEntityID**.
* Выполните запрос и просмотрите результаты.

*DELETE pc*

*FROM Sales.SalesPersonQuotaHistory pc*

*JOIN Sales.SalesPerson sp ON pc.BusinessEntityID = sp.BusinessEntityID*

*WHERE sp.SalesYTD > 1000000.00;*

*SELECT \* FROM Sales.SalesPersonQuotaHistory;*

A screenshot of a computer

Description automatically generated

**Задание 3: Использование инструкции DELETE с предложением OUTPUT**

* Создайте инструкцию **DELETE** с предложением **OUTPUT**, которая удаляет все записи из таблицы **Sales.ShoppingCartItem** и возвращает удаленные данные.
* Выполните запрос и просмотрите результаты.

*DELETE FROM Sales.ShoppingCartItem*

*OUTPUT DELETED.\**

A screenshot of a computer

Description automatically generated

**Упражнение 3: Изменение данных в таблицах**

**Задание 1: Использование инструкции UPDATE c предложением SET**

* Создайте инструкцию **UPDATE** с предложением **SET**, которое изменяет данные в таблице **Sales.SalesPerson** и устанавливает значение бонуса (поле **Bonus**) в величину $6000, значение комиссии (поле **CommissionPct**) – в 0.10% и значение квоты продаж (поле **SalesQuota**) – в NULL для всех записей таблицы.
* Выполните запрос и просмотрите результаты.
* *UPDATE Sales.SalesPerson*
* *SET Bonus = 6000,*
* *CommissionPct = 0.10,*
* *SalesQuota = NULL;*
* *SELECT \* FROM Sales.SalesPerson;*

A screenshot of a computer

Description automatically generated

**Задание 2: Использование инструкции UPDATE c предложением WHERE**

* Создайте инструкцию **UPDATE** с предложением **WHERE**, которая изменяет записи в таблице **Production.Product** и устанавливает цвет (поле **Color**) в '**Metallic Red**' для продуктов красного цвета, названия которых (поле **Name**) начинаются с '**Road-250**'.
* Выполните запрос и просмотрите результаты.

*UPDATE Production.Product*

*SET Color = 'Metallic Red'*

*WHERE Color = 'Red' AND Name LIKE 'Road-250%';*

*SELECT \* FROM Production.Product WHERE Name LIKE 'Road-250%';*

A screenshot of a computer

Description automatically generated

**Задание 3: Использование инструкции UPDATE с предложением OUTPUT**

* Создайте инструкцию **UPDATE**, которая увеличивает количество часов отпуска (поле **VacationHours**) на 25% для первых 10 записей таблицы **HumanResources.Employee**.
* Инструкция **UPDATE** должна использовать предложение **OUTPUT** для вставки данных, затронутых инструкцией **UPDATE**, в табличную переменную, созданную с помощью следующего запроса:

|  |
| --- |
| DECLARE @MyTableVar table( EmpID INT NOT NULL,                            Name varchar(50),                            OldVacationHours INT,                            NewVacationHours INT,                            ModifiedDate DATETIME); |

* Используйте предложение **TOP** для выбора первых 10 записей из таблицы **HumanResources.Employee**.
* Включите в запрос инструкцию **SELECT**, которая возвращает данные из табличной переменной **@MyTableVar**.
* Выполните запрос и просмотрите результаты.