**Что такое MERGE в T-SQL?**

**MERGE** – операция в языке T-SQL, при которой происходит обновление, вставка или удаление данных в таблице на основе результатов соединения с данными другой таблицы или SQL запроса. Другими словами, с помощью MERGE можно осуществить слияние двух таблиц, т.е. синхронизировать их.

В операции MERGE происходит объединение по ключевому полю или полям основной таблицы **(в которой и будут происходить все изменения**) с соответствующими полями другой таблицы или результата запроса. В итоге если условие, по которому происходит объединение, истина

(**WHEN MATCHED**),

то мы можем выполнить операции обновления или удаления, если условие не истина, т.е. отсутствуют данные

(**WHEN NOT MATCHED**),

то мы можем выполнить операцию вставки

(**INSERT добавление данных**),

также если в основной таблице присутствуют данные, которое отсутствуют в таблице (*или результате запрос*а) источника

(**WHEN NOT MATCHED BY SOURCE**), то мы можем выполнить обновление или удаление таких данных.

В дополнение к основным перечисленным выше условиям можно указывать «*Дополнительные условия поиска*», они указываются через ключевое слово **AND**.

**Упрощённый синтаксис MERGE**

MERGE <Основная таблица>

USING <Таблица или запрос источника>

ON <Условия объединения>

[ WHEN MATCHED [ AND <Доп. условие> ]

THEN <UPDATE или DELETE>

[ WHEN NOT MATCHED [ AND Доп. условие> ]

THEN <INSERT> ]

[ WHEN NOT MATCHED BY SOURCE [ AND <Доп. условие> ]

THEN <UPDATE или DELETE> ] [ ...n ]

[ OUTPUT ]

;

**Важные моменты при использовании MERGE:**

* В конце инструкции MERGE обязательно должна идти точка с запятой (;) иначе возникнет ошибка;
* Должно быть, по крайней мере, одно условие MATCHED;
* Операцию MERGE можно использовать совместно с [CTE (обобщенным табличным выражением)](https://info-comp.ru/obucheniest/495-the-with-in-t-sql-or-common-table-expression.html);
* В инструкции MERGE можно использовать ключевое слово OUTPUT, для того чтобы посмотреть какие изменения были внесены. Для идентификации операции здесь в OUTPUT можно использовать переменную $action;
* На все операции к основной таблице, которые предусмотрены в MERGE (удаления, вставки или обновления), действуют все ограничения, определенные для этой таблицы;
* Функция @@ROWCOUNT, если ее использовать после инструкции MERGE, будет возвращать общее количество вставленных, обновленных и удаленных строк;
* Для того чтобы использовать MERGE необходимо разрешение на INSERT, UPDATE или DELETE в основной таблице, и разрешение SELECT для таблицы источника;
* При использовании MERGE необходимо учитывать, что все триггеры AFTER на INSERT, UPDATE или DELETE, определенные для целевой таблицы, будут запускаться.

А теперь переходим к практике. И для начала давайте определимся с исходными данными.

## Исходные данные для примеров операции MERGE

У меня в качестве SQL сервера будет выступать [Microsoft SQL Server 2016 Express](https://info-comp.ru/sisadminst/535-install-microsoft-sql-server-2016-express.html" \o "Описание установки Microsoft SQL Server 2016 Express и Management Studio на Windows 8.1" \t "_blank). На нем есть тестовая база данных, в которой я создаю тестовые таблицы, например, с товарами: **TestTable** – это у нас будет целевая таблица, т.е. та над которой мы будем производить все изменения, и **TestTableDop** – это таблица источник, т.е. данные в соответствии с чем, мы будем производить изменения.

***Запрос для создания таблиц.***

--Целевая таблица

CREATE TABLE dbo.TestTable(

ProductId INT NOT NULL,

ProductName VARCHAR(50) NULL,

Summa MONEY NULL,

CONSTRAINT PK\_TestTable PRIMARY KEY CLUSTERED (ProductId ASC)

)

--Таблица источник

CREATE TABLE dbo.TestTableDop(

ProductId INT NOT NULL,

ProductName VARCHAR(50) NULL,

Summa MONEY NULL,

CONSTRAINT PK\_TestTableDop PRIMARY KEY CLUSTERED (ProductId ASC)

)

***Далее я их наполняю тестовыми данными.***

--Добавляем данные в основную таблицу

INSERT INTO dbo.TestTable

(ProductId,ProductName,Summa)

VALUES

(1, 'Компьютер', 0)

GO

INSERT INTO dbo.TestTable

(ProductId,ProductName,Summa)

VALUES

(2, 'Принтер', 0)

GO

INSERT INTO dbo.TestTable

(ProductId,ProductName,Summa)

VALUES

(3, 'Монитор', 0)

GO

--Добавляем данные в таблицу источника

INSERT INTO dbo.TestTableDop

(ProductId,ProductName,Summa)

VALUES

(1, 'Компьютер', 500)

GO

INSERT INTO dbo.TestTableDop

(ProductId,ProductName,Summa)

VALUES

(2, 'Принтер', 300)

GO

INSERT INTO dbo.TestTableDop

(ProductId,ProductName,Summa)

VALUES

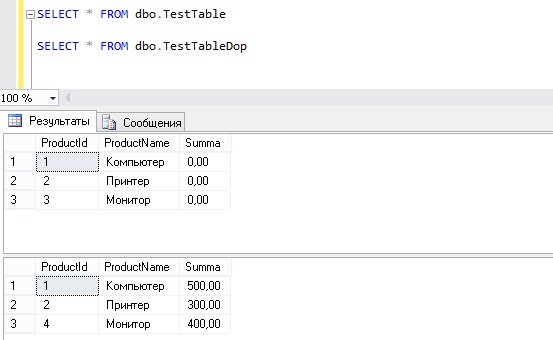
(4, 'Монитор', 400)

GO

***Посмотрим на эти данные.***

SELECT \* FROM dbo.TestTable

SELECT \* FROM dbo.TestTableDop



Видно, что в целевой таблице значение поля Summa = 0, а также есть несоответствие некоторых идентификаторов, т.е. у нас есть товары, которые есть в одной таблице, при этом они отсутствуют в другой.

## Пример 1 – обновление и добавление данных с помощью MERGE

Это, наверное, классический вариант использования MERGE, когда мы по условию объединения обновляем данные, а если таких данных нет, то добавляем их. Для наглядности в конце инструкции MERGE я укажу ключевое слово OUTPUT, для того чтобы посмотреть какие именно изменения мы произвели, а также сделаю выборку итоговых данных.

MERGE dbo.TestTable AS T\_Base --Целевая таблица

USING dbo.TestTableDop AS T\_Source --Таблица источник

ON (T\_Base.ProductId = T\_Source.ProductId) --Условие объединения

WHEN MATCHED THEN --Если истина (UPDATE)

UPDATE SET ProductName = T\_Source.ProductName, Summa = T\_Source.Summa

WHEN NOT MATCHED THEN --Если НЕ истина (INSERT)

INSERT (ProductId, ProductName, Summa)

VALUES (T\_Source.ProductId, T\_Source.ProductName, T\_Source.Summa)

--Посмотрим, что мы сделали

OUTPUT $action AS [Операция], Inserted.ProductId,

Inserted.ProductName AS ProductNameNEW,

Inserted.Summa AS SummaNEW,

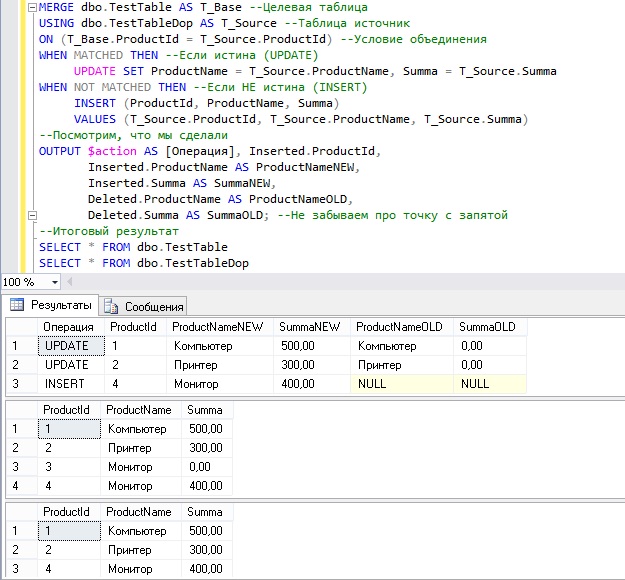
Deleted.ProductName AS ProductNameOLD,

Deleted.Summa AS SummaOLD; --Не забываем про точку с запятой

--Итоговый результат

SELECT \* FROM dbo.TestTable

SELECT \* FROM dbo.TestTableDop



Мы видим, что у нас было две операции UPDATE и одна INSERT. Так оно и есть, две строки из таблицы TestTable соответствуют двум строкам в таблице TestTableDop, т.е. у них один и тот же ProductId, у данных строк в таблице TestTable мы обновили поля ProductName и Summa. При этом в таблице TestTableDop есть строка, которая отсутствует в TestTable, поэтому мы ее и добавили через INSERT.

## Пример 2 – синхронизация таблиц с помощью MERGE

Теперь, допустим, нам нужно синхронизировать таблицу TestTable с таблицей TestTableDop, для этого мы добавим еще одно условие WHEN NOT MATCHED BY SOURCE, суть его в том, что мы удалим строки, которые есть в TestTable, но нет в TestTableDOP. Но для начала, для того чтобы у нас все три условия отработали (в частности WHEN NOT MATCHED) давайте в таблице TestTable удалим строку, которую мы добавили в предыдущем примере. Также здесь я в качестве источника укажу запрос, чтобы Вы видели, как можно использовать запросы в качестве источника.

--Удаление строки с ProductId = 4

--для того чтобы отработало условие WHEN NOT MATCHED

DELETE dbo.TestTable WHERE ProductId = 4

--Запрос MERGE для синхронизации таблиц

MERGE dbo.TestTable AS T\_Base --Целевая таблица

--Запрос в качестве источника

USING (SELECT ProductId, ProductName, Summa

FROM dbo.TestTableDop) AS T\_Source (ProductId, ProductName, Summa)

ON (T\_Base.ProductId = T\_Source.ProductId) --Условие объединения

WHEN MATCHED THEN --Если истина (UPDATE)

UPDATE SET ProductName = T\_Source.ProductName, Summa = T\_Source.Summa

WHEN NOT MATCHED THEN --Если НЕ истина (INSERT)

INSERT (ProductId, ProductName, Summa)

VALUES (T\_Source.ProductId, T\_Source.ProductName, T\_Source.Summa)

--Удаляем строки, если их нет в TestTableDOP

WHEN NOT MATCHED BY SOURCE THEN

DELETE

--Посмотрим, что мы сделали

OUTPUT $action AS [Операция], Inserted.ProductId, Inserted.ProductName AS ProductNameNEW,

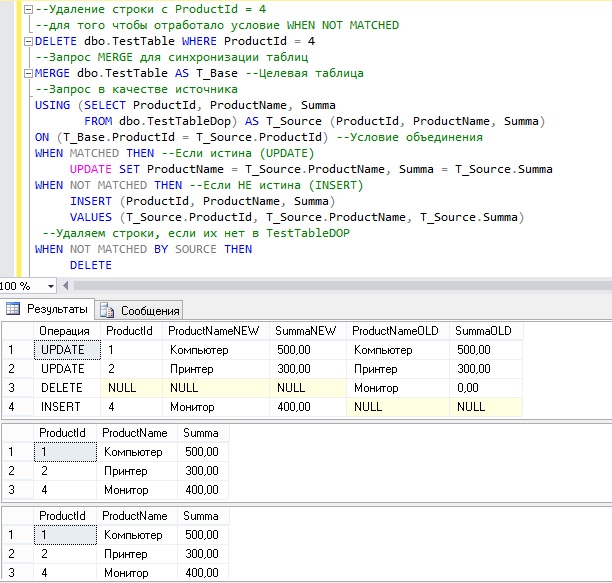
Inserted.Summa AS SummaNEW,Deleted.ProductName AS ProductNameOLD,

Deleted.Summa AS SummaOLD; --Не забываем про точку с запятой

--Итоговый результат

SELECT \* FROM dbo.TestTable

SELECT \* FROM dbo.TestTableDop



В итоге мы видим, что у нас таблицы содержат одинаковые данные. Для этого мы выполнили две операции UPDATE, одну INSERT и одну DELETE. При этом мы использовали всего одну инструкцию MERGE.

## Пример 3 – операция MERGE с дополнительным условием

Сейчас давайте выполним запрос похожий на запрос, который мы использовали в примере 1, только добавим дополнительное условие на обновление данных, например, мы будем обновлять TestTable только в том случае, если поле Summa, в TestTableDop, содержит какие-нибудь данные (например, мы не хотим использовать некорректные значения для обновления). Для того чтобы было видно, как отработало это условие, давайте предварительно очистим у одной строки в таблице TestTableDop поле Summa (поставим NULL).

--Очищаем поле сумма у одной строки в TestTableDop

UPDATE dbo. TestTableDop SET Summa = NULL

WHERE ProductId = 2

--Запрос MERGE

MERGE dbo.TestTable AS T\_Base --Целевая таблица

USING dbo.TestTableDop AS T\_Source --Таблица источник

ON (T\_Base.ProductId = T\_Source.ProductId) --Условие объединения

--Если истина + доп. условие отработало (UPDATE)

WHEN MATCHED AND T\_Source.Summa IS NOT NULL THEN

UPDATE SET ProductName = T\_Source.ProductName, Summa = T\_Source.Summa

WHEN NOT MATCHED THEN --Если НЕ истина (INSERT)

INSERT (ProductId, ProductName, Summa)

VALUES (T\_Source.ProductId, T\_Source.ProductName, T\_Source.Summa)

--Посмотрим, что мы сделали

OUTPUT $action AS [Операция], Inserted.ProductId,

Inserted.ProductName AS ProductNameNEW,

Inserted.Summa AS SummaNEW,

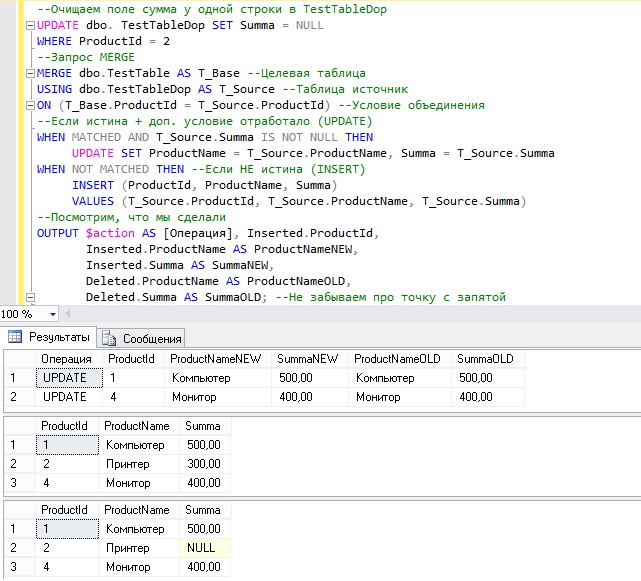
Deleted.ProductName AS ProductNameOLD,

Deleted.Summa AS SummaOLD; --Не забываем про точку с запятой

--Итоговый результат

SELECT \* FROM dbo.TestTable

SELECT \* FROM dbo.TestTableDop



В итоге у меня обновилось всего две строки, притом, что все три строки успешно выполнили условие объединения, но одна строка не обновилась, так как сработало дополнительное условие Summa IS NOT NULL, потому что поле Summa у строки с ProductId = 2, в таблице TestTableDop, не содержит никаких данных, т.е. NULL.

<https://info-comp.ru/obucheniest/561-merge-in-t-sql.html>