# MERGE

## 什麼是 MERGE？

**簡單來說：**
`MERGE` 是 SQL 中的一種進階語法，用於在單一語句中執行插入（INSERT）、更新（UPDATE）或刪除（DELETE）操作，根據來源數據與目標數據的比對結果來決定具體動作。它就像一個「數據同步工具」，能高效地將來源表的數據合併到目標表，特別適合資料庫同步或 ETL（提取、轉換、載入）場景。

---

## 詳細解釋

### 定義

`MERGE`（也稱為 UPSERT，即 Update + Insert）是一種 SQL 語句，允許根據條件比較來源表（Source Table）和目標表（Target Table）的數據，然後執行相應的操作：

- 如果來源數據與目標數據**匹配**，則更新目標數據。
- 如果來源數據**不存在於目標表**，則插入新數據。
- 如果目標數據**不在來源表中**，則可選擇刪除。

`MERGE` 提高了效率，因為它將原本需要多個語句的操作整合成一個，避免多次掃描表格。

### 語法

```
MERGE INTO target_table AS target
USING source_table AS source
ON target.key_column = source.key_column
WHEN MATCHED THEN
    UPDATE SET target.column1 = source.column1, ...
WHEN NOT MATCHED [BY TARGET] THEN
    INSERT (column1, column2, ...)
    VALUES (source.column1, source.column2, ...)
WHEN NOT MATCHED BY SOURCE THEN
    DELETE;
```

- **MERGE INTO**：指定目標表。
- **USING**：指定來源表或查詢。
- **ON**：定義匹配條件（通常是主鍵或唯一鍵）。
- **WHEN MATCHED**：當數據匹配時執行的動作（更新或刪除）。
- **WHEN NOT MATCHED [BY TARGET]**：當來源數據不在目標表時執行的動作（插入）。
- **WHEN NOT MATCHED BY SOURCE**：當目標數據不在來源表時執行的動作（刪除）。

### 支援資料庫

- SQL Server：完整支援（2008 及以上）。
- Oracle：完整支援。
- PostgreSQL：從 15 版起支援 `MERGE`；早期版本可用 `ON CONFLICT` 實現類似功能。
- MySQL：不原生支援 `MERGE`，但可用 `INSERT ... ON DUPLICATE KEY UPDATE` 替代。

---

## 簡單範例

### 場景

假設你有兩個表格：

- **目標表（Target Table）**：`Products`（儲存當前產品庫存）。
- **來源表（Source Table）**：`ProductUpdates`（來自供應商的最新產品數據）。

你需要：

1. 如果產品 ID 已存在於 `Products`，更新其價格和庫存。
2. 如果產品 ID 不在 `Products`，插入新產品。
3. 如果 `Products` 中的產品不在 `ProductUpdates`，刪除它（例如停產）。

### 表格結構

```
-- 目標表：當前產品庫存
CREATE TABLE Products (
    product_id INT PRIMARY KEY,
    product_name VARCHAR(50),
    price DECIMAL(10, 2),
    stock INT
);

-- 來源表：最新產品數據
CREATE TABLE ProductUpdates (
    product_id INT PRIMARY KEY IDENTITY(1,1),
    product_name NVARCHAR(200) NOT NULL,
    category_id INT,
    price DECIMAL(10, 2) NOT NULL,
    stock INT NOT NULL DEFAULT 0,
    description NVARCHAR(MAX),
    created_at DATETIME DEFAULT GETDATE(),
    CONSTRAINT FK_Products_Categories_2 FOREIGN KEY (category_id) REFERENCES
Categories(category_id)
);

INSERT INTO Products (product_name, category_id, price, stock, description)
VALUES
-- 筆記型電腦
(N'ASUS VivoBook 16', 1, 25900.00, 15, N'16吋FHD/i5-1135G7/8G/512G PCIe SSD'),
(N'Acer Swift 8', 1, 26900.00, 10, N'18吋FHD/Ryzen 5 5500U/8G/512G PCIe SSD'),
```

```
(N'MSI Modern 15', 1, 22900.00, 8, N'18吋FHD/i5-1135G7/8G/512G PCIe SSD'),
(N'Apple MacBook Air M3', 1, 29900.00, 12, N'13.6吋/M2晶片/8G/256G SSD')
```

### MERGE 語句

```
MERGE INTO NewProducts AS target
USING Products AS source
ON target.product_id = source.product_id
WHEN MATCHED THEN
    UPDATE SET
        target.product_name = source.product_name,
        target.category_id = source.category_id,
        target.description = source.description,
        target.price = source.price,
        target.stock = source.stock
WHEN NOT MATCHED THEN
    INSERT ( product_name, category_id, description, price, stock)
    VALUES ( source.product_name, source.category_id, source.description, source.price,
source.stock)
WHEN NOT MATCHED BY SOURCE THEN
    DELETE;
```

### 執行結果

執行後，`Products` 表變為：

| product_id | product_name | price | stock |
|---|---|---|---|
| 1 | Laptop | 950.00 | 60 |
| 2 | Phone | 550.00 | 90 |
| 4 | Mouse | 20.00 | 200 |

**解釋**：

- **匹配**：`product_id 1 和 2` 存在於兩表，更新其 `price` 和 `stock`。
- **不匹配（來源）**：`product_id 4` 只在 `ProductUpdates`，插入新記錄。
- **不匹配（目標）**：`product_id 3` 只在 `Products`，被刪除。

### 查詢結果

```
SELECT * FROM Products;
```

---

## 進階功能

1. **條件更新**：
   - 可以在 `WHEN MATCHED` 中加入條件。
   - 例子：

     ```
     WHEN MATCHED AND source.stock > 0 THEN
         UPDATE SET target.stock = source.stock
     ```

2. **多個匹配動作**：
   - SQL Server 支援多個 `WHEN MATCHED` 子句（需搭配條件）。
3. **效能優化**：
   - 使用索引於 `ON` 條件的欄位（如 `product_id`）以加速匹配。
4. **日誌記錄**：
   - 使用 `OUTPUT` 子句記錄 `MERGE` 的操作結果。
   - 例子：

     ```
     MERGE INTO Products AS target
     USING ProductUpdates AS source
     ON target.product_id = source.product_id
     WHEN MATCHED THEN
         UPDATE SET target.price = source.price
     WHEN NOT MATCHED THEN
         INSERT (product_id, price)
         VALUES (source.product_id, source.price)
     OUTPUT $action, inserted.*, deleted.*;
     ```

---

## 應用場景

1. **資料倉儲**
   - 將每日銷售數據合併到歷史表，更新現有記錄並插入新記錄。
2. **ETL 流程**
   - 使用 SQL Server Integration Services (SSIS) 同步來源與目標數據。
3. **庫存管理**
   - 根據供應商更新同步產品價格和庫存。
4. **資料同步**
   - 在分散式系統中保持多個資料庫一致。

---

## 優點

- **高效性**：單一語句取代多個 `INSERT`、`UPDATE`、`DELETE`，減少掃描次數。
- **簡潔性**：語法清晰，易於維護。
- **靈活性**：支援條件邏輯和多種操作。

## 挑戰

- **學習曲線**：對新手來說，語法較複雜。
- **資料庫支援**：部分資料庫（如 MySQL）不支援，需用替代語法。
- **效能風險**：若表格龐大且無適當索引，可能變慢。

---

## 結論

`MERGE` 是 SQL 中強大的工具，適合需要同步或更新數據的場景。它就像一個「數據合併機」，根據匹配條件自動決定插入、更新或刪除，簡化複雜操作。以上範例展示了 SQL Server 中的基本應用，若想深入其他資料庫的替代方案（如 MySQL 的 `ON DUPLICATE KEY UPDATE`）或進階用法，請告訴我，我可以再細講！