

Module 4:

MERGE and INSERT OVER DML

Introducing MERGE

- **Four statements in one**
 - SELECT (with JOIN)
 - INSERT
 - UPDATE
 - DELETE
- **And even more...**
 - OUTPUT clause
 - INSERT OVER DML
- **Operates on a join**
 - Between source and target
 - Type of join based on merge clause(s)
- **More maintainable—and efficient—than individual statements**
 - 100% compatible with existing business logic
 - Existing triggers continue to work

MERGE Syntax

```
MERGE target
  USING source
  ON join
    WHEN MATCHED
      UPDATE | DELETE
    WHEN NOT MATCHED [BY TARGET]
      INSERT
    WHEN NOT MATCHED BY SOURCE
      UPDATE | DELETE
;
```

Demo

- Using MERGE to manage a stock portfolio

Demo

- Replicating Tables with MERGE

Demo

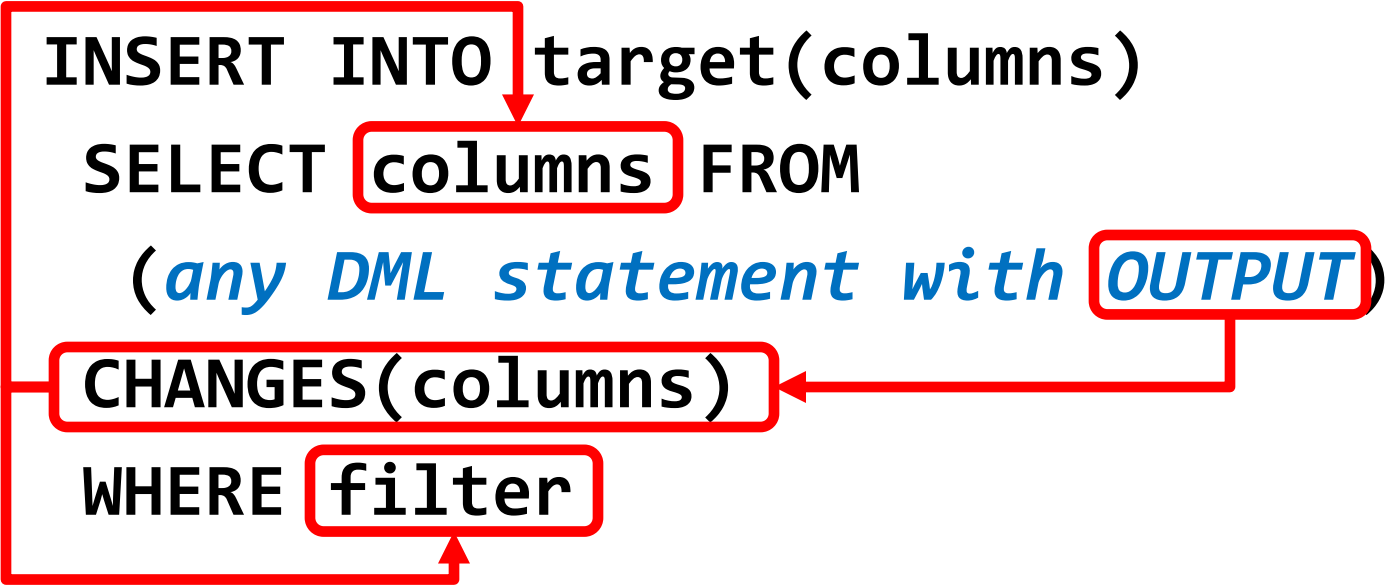
- Examining the Query Execution Plan for MERGE

Introducing INSERT OVER DML

DML Output

- **INSERT, UPDATE, DELETE, and MERGE all support the OUTPUT clause**
 - Captures before-and-after snapshots of modified data via INSERTED and DELETED pseudo-tables (just like triggers)
 - MERGE adds \$action virtual column (returning INSERT, UPDATE or DELETE)
- **OUTPUT INTO can capture the change data to a table or table variable**
 - Suffers from one limitation – no filtering
 - Solution – use INSERT OVER DML
- **INSERT OVER DML Syntax**
 - Wrap an INSERT around any DML that has an OUTPUT (*not* OUTPUT INTO) clause
 - Use CHANGES to map OUTPUT columns from the inner DML statement for use in the outer INSERT statement

INSERT OVER DML Syntax



The diagram illustrates the syntax of the INSERT OVER DML statement. The text is as follows:

```
INSERT INTO target(columns)
  SELECT columns FROM
    (any DML statement with OUTPUT)
  CHANGES(columns)
  WHERE filter
```

Annotations in the diagram include:

- A red box around **INSERT INTO** with an arrow pointing to **target(columns)**.
- A red box around **columns** in the **SELECT** clause.
- A red box around **OUTPUT** in the subquery, with an arrow pointing to **CHANGES(columns)**.
- A red box around **filter** in the **WHERE** clause, with an arrow pointing to the **WHERE** keyword.
- A red box around **CHANGES(columns)**.

Demo

- Capturing Change Data with OUTPUT INTO

Demo

- Capturing Change Data with INSERT OVER DML