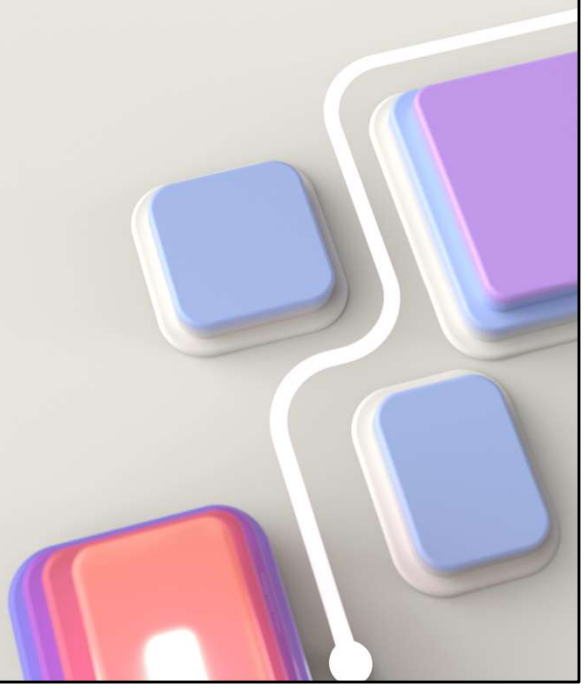
Microsoft

5: Modifying data

Agenda

- Inserting data into tables
- Modifying and deleting data

1: Inserting data into tables



Options for inserting data into tables

INSERT…VALUES

- Inserts explicit values
- You can omit identity columns, columns that allow NULL, and columns with default constraints
- You can also explicitly specify NULL and DEFAULT

INSERT…SELECT

Inserts the results returned by a query into an existing table

SELECT…INTO

Creates a new table from the results of a query

Identity columns

IDENTITY property of a column generates sequential numbers automatically for insertion into a table

- Optional seed and increment values can be specified when creating the table
- Use system variables and functions to return last inserted identity:

  @@IDENTITY: The last identity generated in the session

  SCOPE_IDENTITY(): The last identity generated in the current scope

  IDENT_CURRENT('<*table_name*>'): The last identity inserted into a table

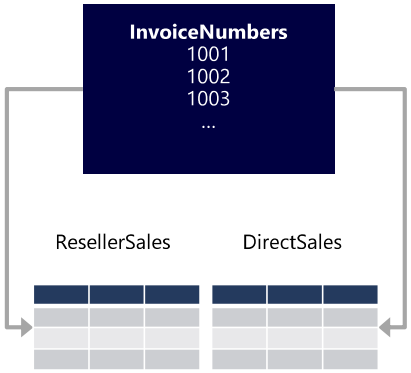
```
INSERT INTO Sales.Promotion (PromotionName,StartDate,ProductModelID,Discount,Notes)
VALUES
('Clearance Sale', '01/01/2021', 23, 0.10, '10% discount')
…
SELECT SCOPE_IDENTITY() AS PromotionID;
```

Sequences

Sequences are objects that generate sequential numbers

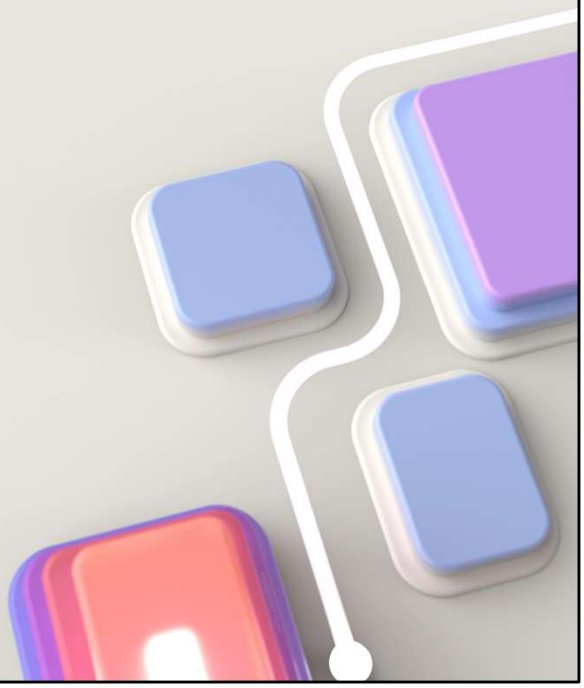
- Exist independently of tables, so offer greater flexibility than Identity
- Use SELECT NEXT VALUE FOR to retrieve the next sequential number

Can be set as the default value for a column




```
CREATE SEQUENCE Sales.InvoiceNumber AS INT
START WITH 1000 INCREMENT BY 1;
…
SELECT NEXT VALUE FOR Sales.InvoiceNumber;
```

2: Modifying and deleting data



Updating data in a table

Updates all rows in a table or view

- Set can be filtered with a WHERE clause
- Set can be defined with a FROM clause

Only columns specified in the SET clause are modified

```
UPDATE Sales.Promotion
SET Notes = '25% off socks'
WHERE PromotionID = 2;
```

Deleting data from a table

DELETE removes rows that match the WHERE predicate

- Caution: DELETE without a WHERE clause deletes all rows!

```
DELETE FROM Production.Product
WHERE discontinued = 1;
```

TRUNCATE TABLE clears the entire table

- Storage physically deallocated, rows not individually removed
- The operation is minimally logged to optimize performance
- TRUNCATE TABLE will fail if the table is referenced by a foreign key constraint in another table

```
TRUNCATE TABLE Sales.Promotion;
```

Merging data in a table

MERGE modifies data based on a condition

- When the source matches the target
- When the source has no match in the target
- When the target has no match in the source

```
MERGE INTO Sales.Invoice as i
USING Sales.InvoiceStaging as s
ON i.SalesOrderID = s.SalesOrderID
WHEN MATCHED THEN
    UPDATE SET i.CustomerID = s.CustomerID,
                i.OrderDate = GETDATE(),
                i.PONumber = s.PONumber,
                i.TotalDue = s.TotalDue
WHEN NOT MATCHED THEN
    INSERT (SalesOrderID, CustomerID, OrderDate, PONumber, TotalDue)
    VALUES (s.SalesOrderID, s.CustomerID, s.OrderDate, s.PONumber, s.TotalDue);
```

Lab: Modifying data

- Insert data
- Update data
- Delete data