Combining and Filtering Data with T-SQL

Filtering and Controlling Flow

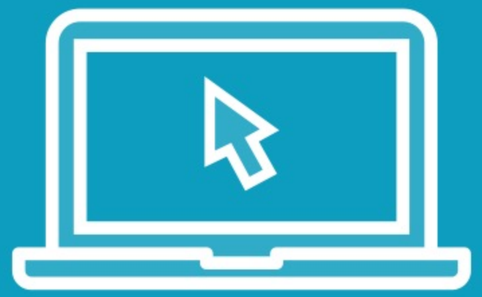
Tamara Pattinson
Consultant | Software Engineer | Instructor

@PattinsonTamara www.tamarapattinson.com

Overview

- WHERE syntax
- >, >=, <, <=, %, =, !=
- AND / OR
- Filter previous queries using WHERE
- Nested SELECT queries and optimization concerns
- Using filtering to anaylze for data integrity and query validation

Demo



WHERE syntax

Quantifiers

- Greater than
- Less than
- Equals
- Not Equal

Adding nested selects in WHERE clauses

Filtering on NULL

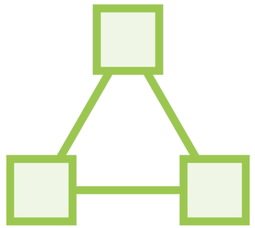
Applying T-SQL Logical Operators

```
/* Logical Operators */


/* both expressions must be true*/
 (10 > 5) AND (10 > 8)

/* either expression must be true*/
 (10 < 5) OR (10 > 8)

/* pattern filtering using LIKE */
 LIKE '%oma%' LIKE 'D%', LIKE '%n'
 NOT LIKE '%oma%' LIKE 'D%', LIKE '%n'

/* range of values */
BETWEEN 1 AND 5
NOT BETWEEN 1 AND 5
```

Filtering the Results in T-SQL Queries

Expression operators (=, !=, >, >=, <, <=)

Logical operators (AND, OR, LIKE, BETWEEN, NOT)



Logic (IF, IF/ELSE, WHILE)

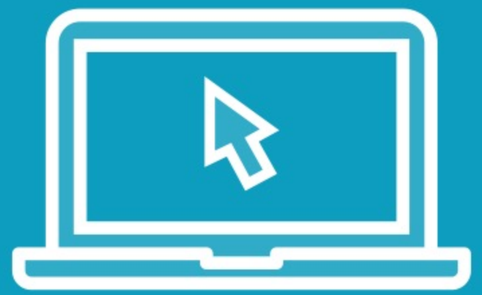
IF it’s cold OR windy
  wear a coat

IF a good movie is playing
  GOTO: Movie

ELSE
  read a book

WHILE the dishes are dirty
  keep washing!

Demo



Controlling Flow

- IF
  - IF/ELSE
  - IF/ELSE IF/ELSE
  - IIF vs IF
- WHILE
  - BREAK
  - CONTINUE
- GOTO

```
/* IF */
DECLARE @num1 AS INT, @num2 AS INT
SET @num1 = 5
SET @num2 = 10

/* Evaluate for variables */
IF @num1 < @num2
  BEGIN
     PRINT  '@num1 is less than  @num2'
  END

/* Executes everything from here */
SELECT * FROM table
```

```
/* IF/ELSE */
DECLARE @num1 AS INT, @num2 AS INT
SET @num1 = 5
SET @num2 = 10

/* Evaluate for variables */
IF @num1 < @num2
  BEGIN
     PRINT  ‘ less than ‘
  END
ELSE
     PRINT   ‘not less than’
```

```
/* IF/ELSE IF/ELSE */
DECLARE @num1 AS INT, @num2 AS INT
SET @num1 = 5
SET @num2 = 10

/* Evaluate for variables */
IF @num1 < @num2
  BEGIN
    PRINT  ‘ less than ‘
  END
ELSE IF @num1 > @num2
  BEGIN
    PRINT   ‘greater than’
  END
ELSE
    PRINT   ‘equal’
```

```
/* WHILE */

DECLARE @firstNum AS INT
SET @firstNum = 0

WHILE @firstNum < 3
   BEGIN
      SET @firstNum = @firstNum + 1
      PRINT @firstNum
   END
PRINT 'Finished WHILE’
```

```
/* IF/ELSE/GOTO */
DECLARE @num1 AS VARCHAR,
DECLARE @num2 AS VARCHAR
SET @num1 = 5
SET @num2 = 10

/* Evaluate for variables */
IF ISNUMERIC( @num1 ) OR ISNUMERIC ( @num2 )
          AND @num2 > 0
 BEGIN
   GOTO  CalcValue
 END
ELSE
   BEGIN
      GOTO    PrintMessage
   END

CalcValue:
           PRINT   @num1/@num2
PrintMessage:
          PRINT 'these values are not numeric or @num2 is 0'
```

Filtering and Controlling Flow Summary

Expressions and Comparison Operators

- WHERE
- >, >=, <, <=, =, !=
- LIKE / NOT LIKE

- **Nested SELECT queries and optimization concerns**
- **Using filtering to anaylze for data integrity and query validation**

Applying T-SQL Logical Operators

- AND, OR
- BETWEEN
- Compounded logical operators

Controlling Flow with T-SQL Logic

- IF/IIF
- WHILE
- BREAK CONTINUE GOTO

Exercise Files

M5 - Comparison operators and NULL
M5 – Controlling Flow ISNUMERIC and GOTO
M5 - Controlling Flow with IF
M5 – Controlling Flow with WHILE-BREAK-CONTINUE
M5 – Filtering for NULL 1
M5 – Filtering for NULL 2
M5 – Filtering Query Results
M5 – Filtering the table example
M5 – Filtering using IN and NOT IN
M5 – LIKE and NOT LIKE filtering
M5 – Logical and Logical Operators
M5 – Logical Operators
M5 – Using quantifiers to filters

Up Next:
Limiting the Results with T-SQL Functions