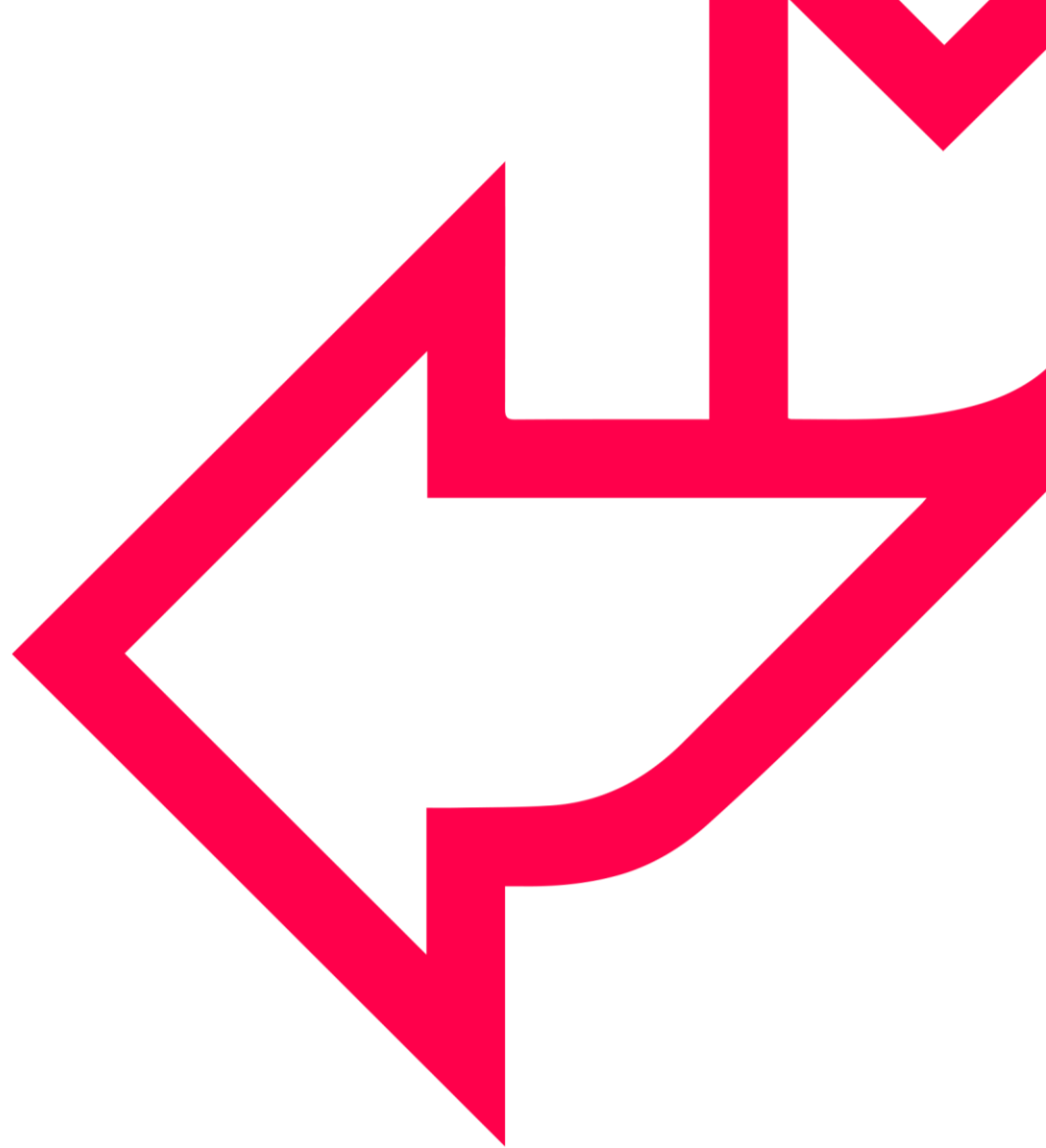




Filtering Rows





OVERVIEW

- WHERE clause
- WHERE expression
- Operators
- Unknown values
- Best practices
- Hands-on lab

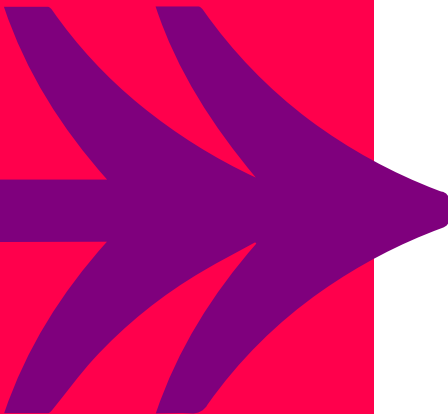




OBJECTIVES

At the end of this module, you will be able to:

- write a query that uses an equality filter.
- write a query that use a comparison filter.
- write a query that uses the IN and BETWEEN operators.
- write a query that looks for text within a column.
- write a query that correctly identifies NULL values.





WHERE CLAUSE

```
SELECT <<field(s)>>  
FROM <<table(s)>>  
WHERE <<condition(s)>>  
GROUP BY <<field(s)>>  
HAVING <<condition(s)>>  
ORDER BY <<field(s)>>
```



QA WHERE expression

- **Is a predicate**
 - An expression that returns either true or false
- **SQL retrieves all rows that are true**
 - Or possibly none at all
- **Example:**

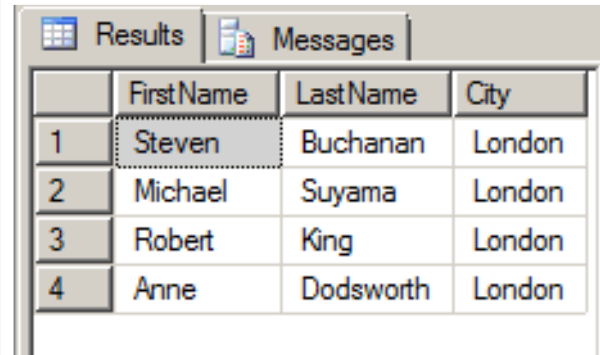
```
SELECT
    ProductID,
    ProductName,
    UnitsInStock,
    UnitsOnOrder
FROM
    dbo.Products
WHERE
    Discontinued = 0
```

QA Operators

Type	Operators
• Comparison operators	• =, <, >, <>, !=, >=, <= <i>UnitsInStock > 0</i>
• Logical operators	• AND, OR, NOT <i>UnitsInStock = 0 AND UnitsOnOrder = 0</i>
• “Special” operators	• BETWEEN, IN, LIKE <i>OrderDate BETWEEN '2012/01/01' AND '2012/01/31'</i>

QA Comparison operators

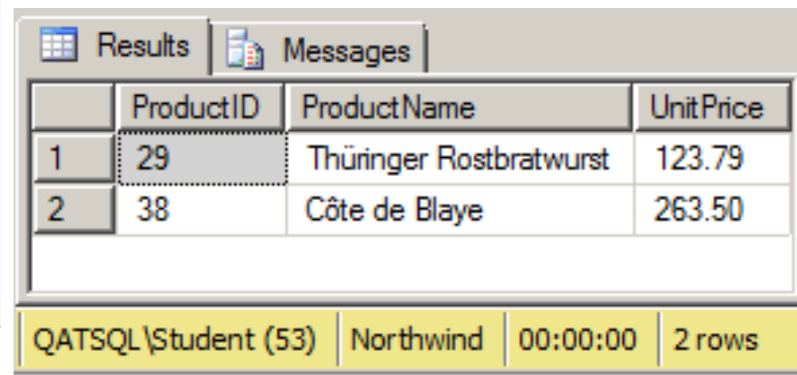
```
SELECT
    FirstName, LastName,
    City
FROM
    dbo.Employees
WHERE
    Country = 'UK'
```



A screenshot of a SQL Server Results window. The 'Results' tab is active, displaying a table with four columns: 'First Name', 'Last Name', and 'City'. There are four rows of data, numbered 1 through 4 in the first column. The first row shows 'Steven Buchanan' from 'London'. The second row shows 'Michael Suyama' from 'London'. The third row shows 'Robert King' from 'London'. The fourth row shows 'Anne Dodsworth' from 'London'.

	First Name	Last Name	City
1	Steven	Buchanan	London
2	Michael	Suyama	London
3	Robert	King	London
4	Anne	Dodsworth	London

```
SELECT
    ProductID, ProductName,
    UnitPrice
FROM
    dbo.Products
WHERE
    UnitPrice > 100
```



A screenshot of a SQL Server Results window. The 'Results' tab is active, displaying a table with four columns: 'ProductID', 'ProductName', and 'UnitPrice'. There are two rows of data, numbered 1 through 2 in the first column. The first row shows '29' for 'ProductID', 'Thüringer Rostbratwurst' for 'ProductName', and '123.79' for 'UnitPrice'. The second row shows '38' for 'ProductID', 'Côte de Blaye' for 'ProductName', and '263.50' for 'UnitPrice'. At the bottom of the window, a status bar shows 'QATSQL\Student (53) | Northwind | 00:00:00 | 2 rows'.

	ProductID	ProductName	UnitPrice
1	29	Thüringer Rostbratwurst	123.79
2	38	Côte de Blaye	263.50

QATSQL\Student (53) | Northwind | 00:00:00 | 2 rows

QA Logical operators

```
SELECT
    ProductID, ProductName,
    CategoryID, UnitPrice
FROM
    dbo.Products
WHERE
    CategoryID = 7
    OR CategoryID = 8
    AND UnitPrice > 30
```

Results		Messages		
	ProductID	ProductName	CategoryID	UnitPrice
1	7	Uncle Bob's Organic Dried Pears	7	30.00
2	10	Ikura	8	31.00
3	14	Tofu	7	23.25
4	18	Camaron Tigers	8	62.50
5	28	Rössle Sauerkraut	7	45.60
6	51	Manjimup Dried Apples	7	53.00
7	74	Longlife Tofu	7	10.00

QL\SQL2012 (11.0 CTP) | QATSQL\Student (54) | Northwind | 00:00:00 | 7 rows

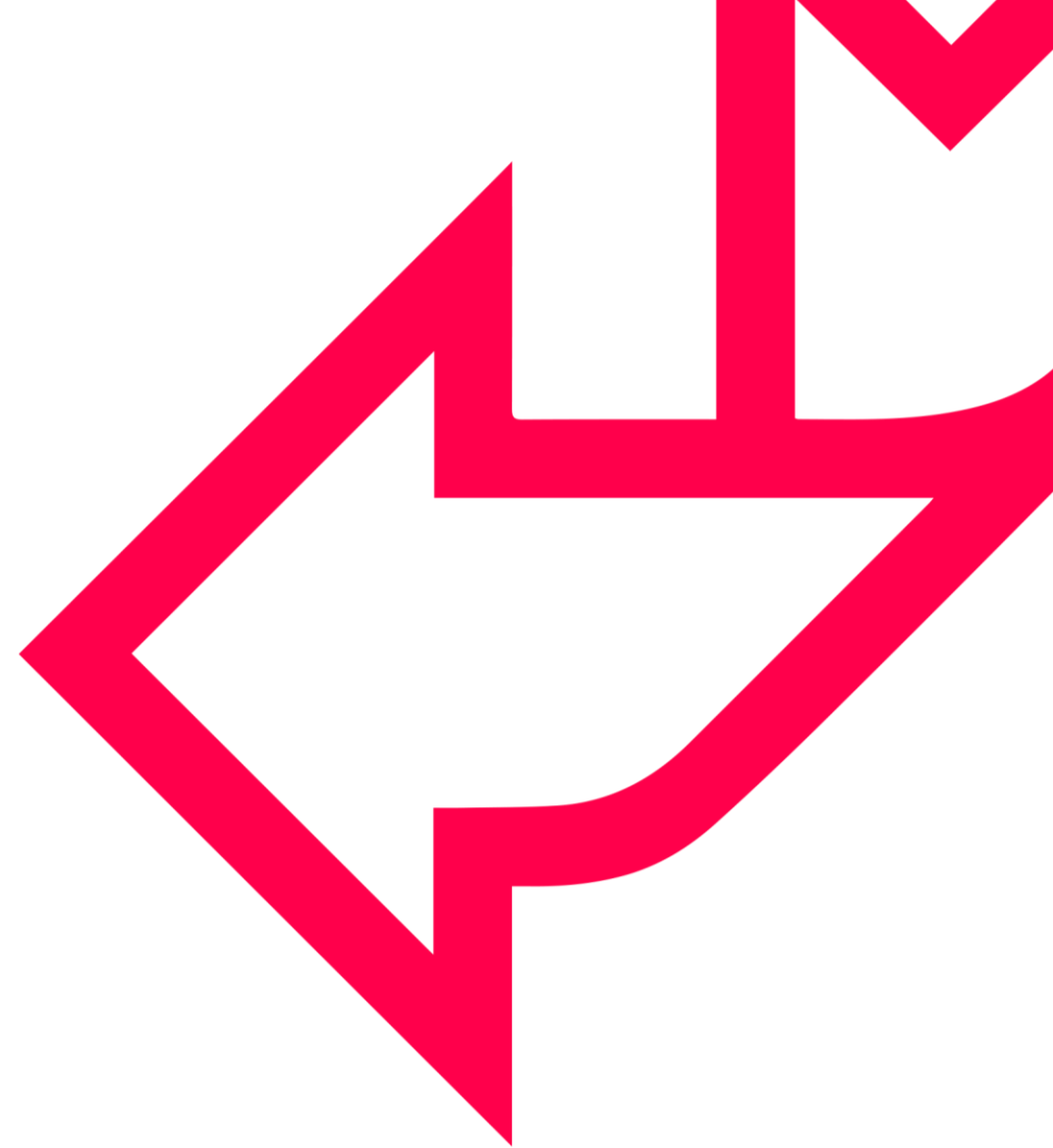
```
...
WHERE
    (CategoryID = 7
    OR CategoryID = 8)
    AND UnitPrice > 30
```

Results		Messages		
	ProductID	ProductName	CategoryID	UnitPrice
1	10	Ikura	8	31.00
2	18	Camaron Tigers	8	62.50
3	28	Rössle Sauerkraut	7	45.60
4	51	Manjimup Dried Apples	7	53.00

2 (11.0 CTP) | QATSQL\Student (54) | Northwind | 00:00:00 | 4 rows



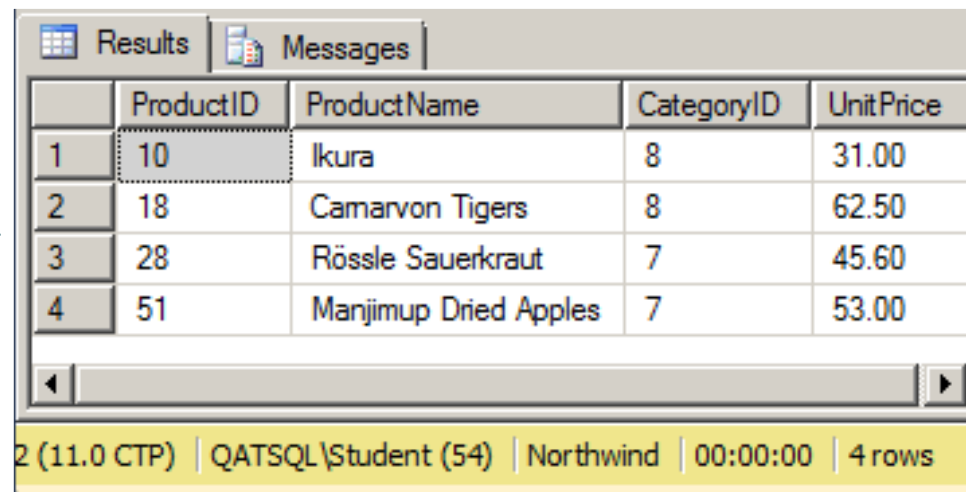
Unknown values



QA Specifying a list of values - IN

```
SELECT
    ProductID, ProductName,
    CategoryID, UnitPrice
FROM
    dbo.Products
WHERE
    (CategoryID = 7
    OR CategoryID = 8)
    AND UnitPrice > 30
```

```
SELECT
    ProductID, ProductName,
    CategoryID, UnitPrice
FROM
    dbo.Products
WHERE
    CategoryID IN (7, 8)
    AND UnitPrice > 30
```



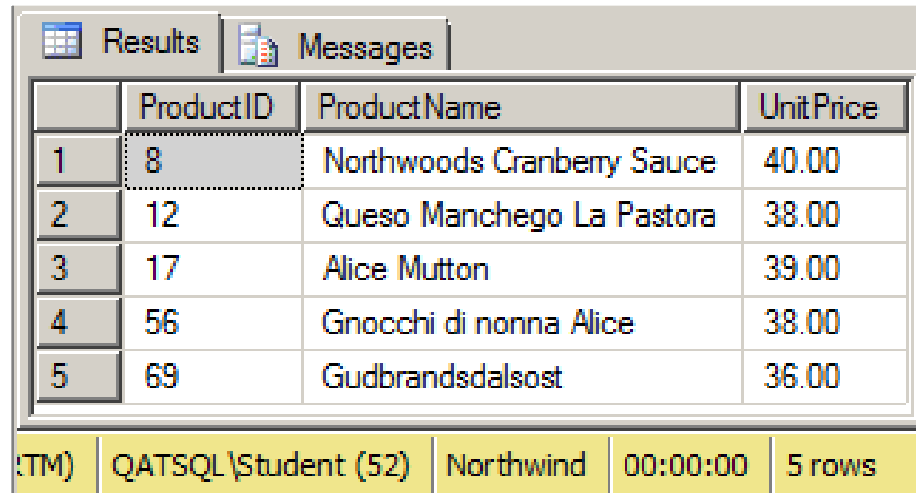
	ProductID	ProductName	CategoryID	UnitPrice
1	10	Ikura	8	31.00
2	18	Camaron Tigers	8	62.50
3	28	Rössle Sauerkraut	7	45.60
4	51	Manjimup Dried Apples	7	53.00

2 (11.0 CTP) | QATSQL\Student (54) | Northwind | 00:00:00 | 4 rows

QA Specifying a range of values - BETWEEN

```
SELECT
    ProductID, ProductName,
    UnitPrice
FROM
    dbo.Products
WHERE
    UnitPrice >= 35
    AND UnitPrice <= 40
```

```
SELECT
    ProductID, ProductName,
    UnitPrice
FROM
    dbo.Products
WHERE
    UnitPrice
        BETWEEN 35 AND 40
```



	ProductID	ProductName	UnitPrice
1	8	Northwoods Cranberry Sauce	40.00
2	12	Queso Manchego La Pastora	38.00
3	17	Alice Mutton	39.00
4	56	Gnocchi di nonna Alice	38.00
5	69	Gudbrandsdalsost	36.00

(TM) | QATSQL\Student (52) | Northwind | 00:00:00 | 5 rows

QA String comparisons - LIKE

```
SELECT
    FirstName, LastName,
    Title
FROM
    dbo.Employees
WHERE
    Title LIKE 'Sales%'
```

	FirstName	LastName	Title
1	Nancy	Davolio	Sales Representative
2	Janet	Leverling	Sales Representative
3	Margaret	Peacock	Sales Representative
4	Steven	Buchanan	Sales Manager
5	Michael	Suyama	Sales Representative
6	Robert	King	Sales Representative
7	Anne	Dodsworth	Sales Representative

QATSQL\Student (54) | Northwind | 00:00:00 | 7 rows

```
SELECT
    FirstName, LastName,
    Title
FROM
    dbo.Employees
WHERE
    Title LIKE '%sales%'
```

	FirstName	LastName	Title
1	Nancy	Davolio	Sales Representative
2	Andrew	Fuller	Vice President, Sales
3	Janet	Leverling	Sales Representative
4	Margaret	Peacock	Sales Representative
5	Steven	Buchanan	Sales Manager
6	Michael	Suyama	Sales Representative
7	Robert	King	Sales Representative
8	Laura	Callahan	Inside Sales Coordinator
9	Anne	Dodsw...	Sales Representative

CTP) | QATSQL\Student (54) | Northwind | 00:00:00 | 9 rows

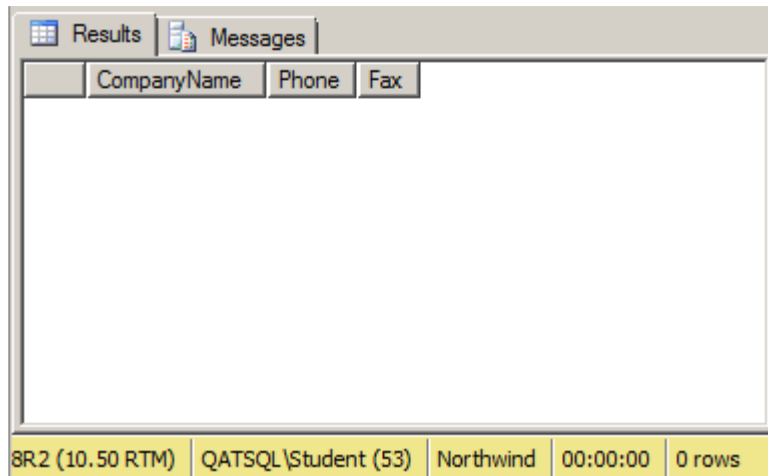
Filtering on calculated values

- Reuse the expression in the WHERE clause
- Has to be evaluated for every row

```
SELECT
    ProductID,
    ProductName,
    UnitsInStock + UnitsOnOrder AS FutureStock
FROM
    dbo.Products
WHERE
    --FutureStock < 100 --won't work!
    --use:
    (UnitsInStock + UnitsOnOrder) < 100
```

QA Finding unknown values (NULLs)

```
SELECT
    CompanyName, Phone,
    Fax
FROM
    dbo.Suppliers
WHERE
    Fax = NULL
```

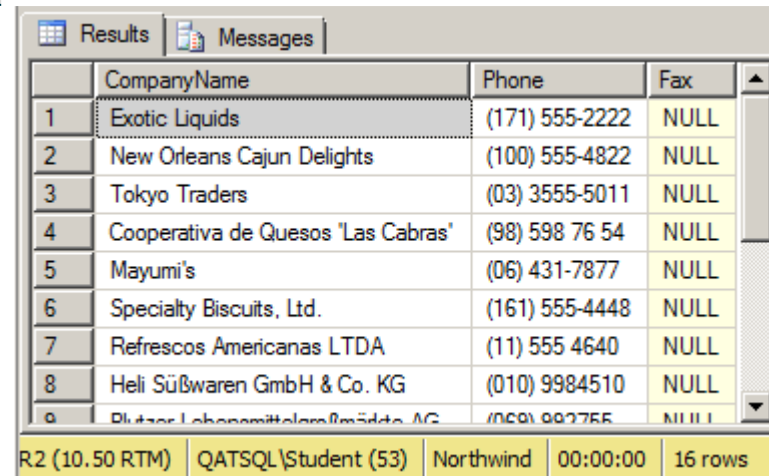


The screenshot shows a SQL Server Enterprise Manager window with the 'Results' tab selected. The query result is empty, showing only the column headers: CompanyName, Phone, and Fax. The status bar at the bottom indicates '0 rows'.

	CompanyName	Phone	Fax
--	-------------	-------	-----

8R2 (10.50 RTM) | QATSQL\Student (53) | Northwind | 00:00:00 | 0 rows

```
SELECT
    CompanyName, Phone,
    Fax
FROM
    dbo.Suppliers
WHERE
    Fax IS NULL
```



The screenshot shows a SQL Server Enterprise Manager window with the 'Results' tab selected. The query result displays 16 rows of data from the Suppliers table, where the Fax column is NULL. The status bar at the bottom indicates '16 rows'.

	CompanyName	Phone	Fax
1	Exotic Liquids	(171) 555-2222	NULL
2	New Orleans Cajun Delights	(100) 555-4822	NULL
3	Tokyo Traders	(03) 3555-5011	NULL
4	Cooperativa de Quesos 'Las Cabras'	(98) 598 76 54	NULL
5	Mayumi's	(06) 431-7877	NULL
6	Specialty Biscuits, Ltd.	(161) 555-4448	NULL
7	Refrescos Americanas LTDA	(11) 555 4640	NULL
8	Heli Süßwaren GmbH & Co. KG	(010) 9984510	NULL
9	Plutzer Lebensmittelgroßhandels AG	(069) 907755	NULL
10	Northwind Traders	(170) 555-4444	NULL
11	Que Pasa	(519) 845-6222	NULL
12	Formaggi di Montebello	(051) 755-4444	NULL
13	Formaggi di Montebello	(051) 755-4444	NULL
14	Formaggi di Montebello	(051) 755-4444	NULL
15	Formaggi di Montebello	(051) 755-4444	NULL
16	Formaggi di Montebello	(051) 755-4444	NULL

8R2 (10.50 RTM) | QATSQL\Student (53) | Northwind | 00:00:00 | 16 rows



Best practices



Avoid NOTs

- Including <>

Use brackets around complex expressions

- This also makes them easier to read

Use IN and BETWEEN

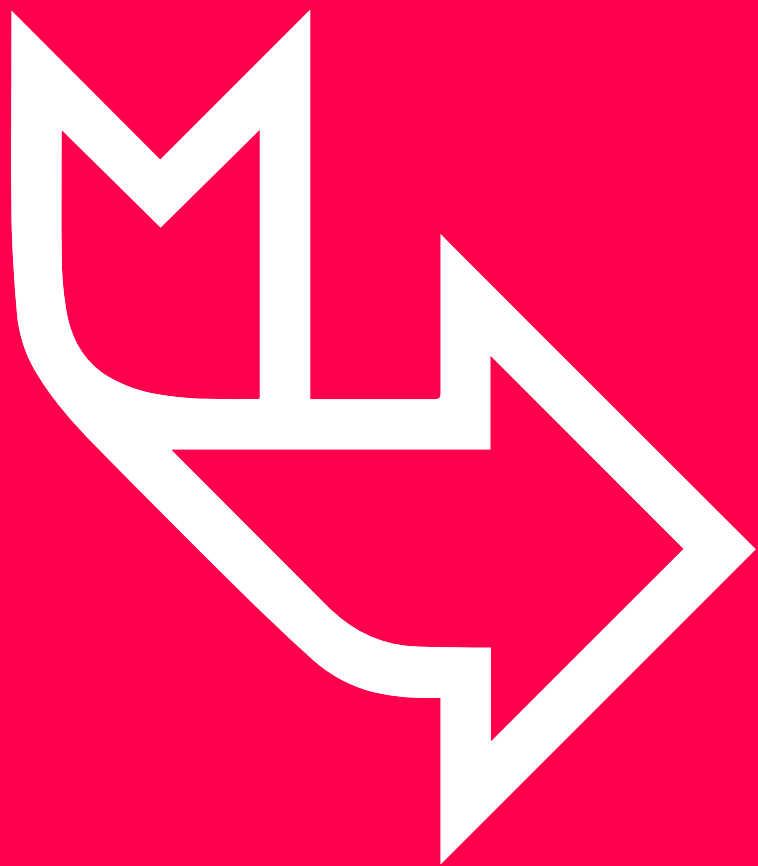
- Rather than lots of ORs and ANDs

Avoid leading wildcards

- Inefficient

Use IS NULL

- Rather than = NULL

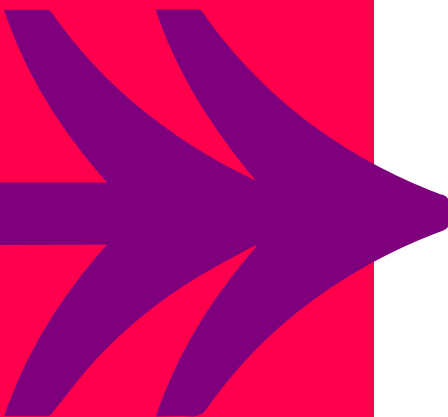


Hands-on lab



HANDS-ON LAB

- **Basic equality filters**
- **Basic comparisons**
- **Logical comparisons**
- **String comparisons**
- **NULL comparisons**





Review

- **WHERE clause**
- **Comparisons**
- **IN, BETWEEN, LIKE**
- **Filtering on calculated expressions**
- **Working with NULLs**

