

# SQL Basics – The WHERE clause

What if I don't want ALL the rows from the input table to appear in the answer set?

We use the **WHERE** clause of the **SELECT** statement to define a **CONDITION** that determines whether a row from the input table will appear in the answer set.

If the condition is **TRUE**, the row appears in the answer set

# SQL Basics – The WHERE clause

SELECT statement with WHERE clause:

```
SELECT <column1>, <column2>, <column3>,  
      <literal>, <expression> AS <alias>  
FROM   <table>  
WHERE  <condition> ;
```

# SQL Basics – WHERE Conditions

The condition in the WHERE clause takes this format:

< operand > < operator > < operand >

Operands may be **columns**, **literals** or **expressions**

Operator may be

= Equals	!= Not equals	<> Not equals
Like	Between	In
> Greater than	>= Greater than or equals	
< Less than	<= Less than or equals	

# SQL Basics - Examples

```
select customerid, contactname, country  
  from "alanparadise/nw"."customers";
```

```
select customerid, contactname, country  
  from "alanparadise/nw"."customers"  
 where country = 'Brazil';
```

```
select customerid, contactname, country  
  from "alanparadise/nw"."customers"  
 where Country <> 'Brazil';
```

```
select productid, productname, unitprice  
  from "alanparadise/nw"."products"  
 where unitprice > 60;
```

# SQL Basics – LIKE operator

LIKE is used for text/strings only (not numeric columns)

LIKE requires use of a WILDCARD character

 means "zero or more of any character"

 (underscore) means "exactly one of any character"

# SQL Basics - Examples

```
select customerid, contactname, country
  from "alanparadise/nw"."customers"
 where contactname like 'M%';
```

```
select customerid, contactname, country
  from "alanparadise/nw"."customers"
 where contactname like '%m%';
```

```
select customerid, contactname, country
  from "alanparadise/nw"."customers"
 where contactname like '%M%';
```

```
select customerid, contactname, country
  from "alanparadise/nw"."customers"
 where contactname like '_a%';
```

# SQL Basics – IN operator

The IN operator compares an operand to a list of values

```
WHERE <operand> in (<value>, <value>, <value>)
```

The <value> may be a literal or a column

The <value> must match the data type of the operand

# SQL Basics - Examples

```
select productid, productname, unitprice, supplierid  
    from "alanparadise/nw"."products"  
    where supplierid in (2, 4, 6, 8);
```

```
select supplierid, companyname, region  
    from "alanparadise/nw"."suppliers"  
    where region in ('LA', 'MI', 'OR');
```



# SQL Basics – IN operator

```
WHERE <operand> in (<value1>, <value2>, <value3>)
```

This is the equivalent of

```
WHERE <operand> = <value1> OR
```

```
    <operand> = <value2> OR
```

```
    <operand> = <value3>
```

# SQL Basics – BETWEEN operator

WHERE <operand> between <value1> and <value2>

The between is inclusive

The <value> must match the data type of the operand

# SQL Basics – BETWEEN operator

```
WHERE <operand> between <value1> and <value2>
```

This is the equivalent of

```
WHERE <operand> >= <value1> AND  
      <operand> <= <value2>
```

# SQL Basics

## Examples

```
select ProductID, ProductName, UnitPrice
    from "alanparadise/nw"."products"
    where UnitPrice between 20 and 30;
```

```
select Lastname, Firstname
    from "alanparadise/nw"."employees"
    where Lastname between 'A' and 'M';
```

# SQL Basics – Boolean

Multiple conditions in a WHERE clause can be combined with

"AND", "OR"

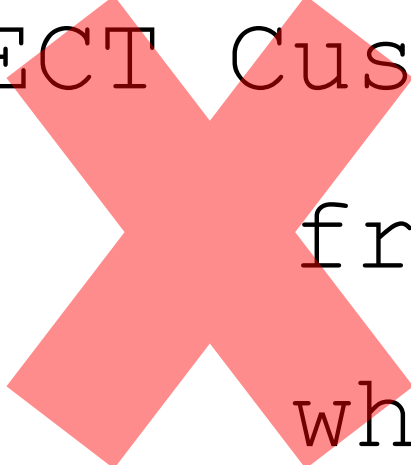
Any condition in a WHERE clause can be negated with

"NOT"

# SQL Basics: Boolean

Boolean expressions are not English ! **NOT** negates the whole condition

```
SELECT customerid, contactname, region, country
    from "alanparadise/nw"."customers"
    where country = 'Brazil';
```



```
SELECT Customerid, contactname, region, country
    from "alanparadise/nw"."customers"
    where country NOT = 'Brazil';
```

```
SELECT customerid, contactname, region, country
    from "alanparadise/nw"."customers"
    where NOT country = 'Brazil';
```

# SQL Basics

When combining WHERE conditions using Boolean operators, please make a habit of **using parentheses**

```
SELECT Productname, SupplierID, CategoryID,  
       UnitPrice, UnitsInStock  
from "alanparadise/nw"."products"  
WHERE SupplierID = 1 AND CategoryID = 2 OR  
       CategoryID = 3 AND UnitPrice > 20 OR  
       UnitsInStock < 12;
```

```
SELECT Productname, SupplierID, CategoryID,  
       UnitPrice, UnitsInStock  
from "alanparadise/nw"."products"  
WHERE SupplierID = 1 AND (CategoryID = 2 OR  
       CategoryID = 3 AND UnitPrice > 20) OR  
       UnitsInStock < 12;
```

# SQL Basics – Lab # 2

This concludes Module 2, Lesson 2, "The Where clause"

Next step: Follow the instructions for Lab # 2