

Slide 1: Filtering, Sorting and Calculating Data with SQL

Filtering, Sorting and Calculating Data with SQL

Sadie St. Lawrence, MSA
Data Scientist, VSP

UC DAVIS
EXTENSION

Slide 2: New Clauses and Operators in SQL

New Clauses and Operators in SQL

WHERE

NOT

BETWEEN

LIKE

IN

ORDER BY

OR

GROUP BY

Slide 3: Wildcards

Wildcards

Wildcards allow more precise search capabilities

Discuss advantages and disadvantages of using wildcards

List best practices for working with wildcards

Slide 4: Math Operators

Math Operators

Using computational math operators

Using aggregate math functions:

AVERAGE

COUNT

MAX

MIN

Slide 1: Basics of Filtering with SQL

Basics of Filtering with SQL

Sadie St. Lawrence, MSA
Data Scientist, VSP

UCDAVIS
EXTENSION

Slide 2: Learning Objectives

Learning Objectives

Describe the basics of filtering your data

Use the WHERE clause with common operators

Use the BETWEEN clause

Explain the concept of a NULL value

Slide 3: Why Filter?

Why Filter?

Be specific about the data you want to retrieve

Reduce the number of records you retrieve

Increase query performance

Reduce the strain on the client application

Governance limitations

Slide 4: WHERE Clause Operators

WHERE Clause Operators

```
SELECT column_name, column_name  
FROM table_name  
WHERE column_name operator value;
```


Slide 5: WHERE Clause Operators

WHERE Clause Operators

```
SELECT column_name, column_name
FROM table_name
WHERE column_name operator value;
```

Operator	Description
=	Equal
<>	Not equal. Note: In some versions of SQL this operator may be written as !=
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
IS NULL	Is a null value

Slide 6: Filtering on a Single Condition

Filtering on a Single Condition

	ProductName	UnitPrice	SupplierID
1	Tofu	23.25	6

```
SELECT ProductName
,UnitPrice
,SupplierID
FROM Products
WHERE ProductName = 'Tofu';
```

Slide 7: Filtering on a Single Value

Filtering on a Single Value

	ProductName	UnitPrice	SupplierID
1	Mishi Kobe Niku	97	4
2	Sir Rodney's Marmalade	81	8
3	Theringer Rostbratwurst	123.79	12
4	Cote de Blaye	263.5	18

```
SELECT ProductName
,UnitPrice
,SupplierID
FROM Products
WHERE UnitPrice >= 75;
```

Slide 8: Checking for Non-Matches

Checking for Non-Matches

```
SELECT ProductName
,UnitPrice
,SupplierID
FROM Products
WHERE ProductName <>'Alice
Mutton';
```

	ProductName	UnitPrice	SupplierID
1	Chai	18	1
2	Chang	19	1
3	Aniseed Syrup	10	1
4	Chef Anton's Cajun Seasoning	22	2
5	Chef Anton's Gumbo Mix	21.35	2
6	Grandma's Boysenberry Spread	25	3
7	Uncle Bob's Organic Dried Pears	30	3
8	Northwoods Cranberry Sauce	40	3
9	Mishi Kobe Niku	97	4
10	Ikura	31	4
11	Queso Cabrales	21	5
12	Queso Manchego La Pastora	38	5
13	Konbu	6	6

Slide 9: Filtering with a Range of Values

Filtering with a Range of Values

```
SELECT ProductName
,UnitPrice
,SupplierID
,UnitsInStock
FROM Products
WHERE UnitsInStock BETWEEN 15
AND 80;
```

	ProductName	UnitPrice	SupplierID	UnitsInStock
1	Chai	18	1	39
2	Chang	19	1	17
3	Chef Anton's Cajun Seasoning	22	2	53
4	Uncle Bob's Organic Dried Pears	30	3	15
5	Mishi Kobe Niku	97	4	29
6	Ikura	31	4	31
7	Queso Cabrales	21	5	22
8	Konbu	6	6	24
9	Tofu	23.25	6	35
10	Genen Shouyu	15.5	6	39
11	Pavlova	17.45	7	29
12	Carnarvon Tigers	62.5	7	42
13	Teatime Chocolate Biscuits	9.2	8	25

Slide 10: Filtering No Value

Filtering No Value

No value returned because there are no null values in the product name

```
SELECT ProductName
,UnitPrice
,SupplierID
,UnitsInStock
FROM Products
WHERE ProductName IS NULL;
```

Slide 11: Summary

Summary

Operator	Description
=	Equal
<>	Not equal. Note: In some versions of SQL this operator may be written as !=
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
IS NULL	Is a null value

Source: http://www.w3schools.com/sql/sql_where.asp

Slide 1: Advanced Filtering: IN, OR, and NOT

Advanced Filtering: IN, OR, and NOT

Sadie St. Lawrence, MSA
Data Scientist, VSP

UC DAVIS
EXTENSION

Slide 2: Learning Objectives

Learning Objectives

Use the IN and OR operators to filter your data and get results you want

Differentiate between use of the IN and BETWEEN operators

Discuss importance of order of operations

Explain how and when to use the NOT operator

Slide 3: IN Operator

IN Operator

Specifies a range of conditions

Comma delimited list of values

Enclosed in ()

Slide 4: IN Operator Example

IN Operator Example

```
SELECT
ProductID
,UnitPrice
,SupplierID
From Products
WHERE SupplierID IN (9, 10,
11);
```

	ProductID	UnitPrice	SupplierID
1	22	21	9
2	23	9	9
3	24	4.5	10
4	25	14	11
5	26	31.23	11
6	27	43.9	11

Slide 5: OR Operator

OR Operator

DBMS will not evaluate the second conditions in a WHERE clause if the first condition is met

Use for any rows matching the specific conditions

	ProductID	UnitPrice	SupplierID	ProductName
1	14	23.25	6	Tofu

```
ProductName  
,ProductID  
,UnitPrice  
,SupplierID  
,ProductName  
From Products  
Where ProductName = 'Tofu' OR  
'Konbu';
```

Slide 6: IN vs. OR

IN vs. OR

IN works the same as OR

Benefits of IN

- Long list of options

- IN executes faster than OR

- Don't have to think about the order with IN

- Can contain another SELECT

Slide 7: OR with AND

OR with AND

	ProductID	UnitPrice	SupplierID
1	22	21	9
2	23	9	9
3	26	31.23	11
4	27	43.9	11

SELECT

ProductID

,UnitPrice

,SupplierID

FROM Products**WHERE** SupplierID = 9 **OR**
SupplierID = 11**AND** UnitPrice > 15;

	ProductID	UnitPrice	SupplierID
1	22	21	9
2	26	31.23	11
3	27	43.9	11

SELECT

ProductID

,UnitPrice

,SupplierID

FROM Products**WHERE** (SupplierID = 9 **OR**
SupplierID = 11)**AND** UnitPrice > 15;

Slide 8: Order of Operations

Order of Operations

Can contain AND and OR Operators

SQL processes AND **before** OR

Use ()

Don't rely on the default order of operations. You are better being specific and getting in the habit of using the ()

Slide 9: NOT Operator

NOT Operator

EmployeeID	LastName	FirstName	Title	TitleOfCourtesy	BirthDate	HireDate	Address	City	Region	
1	2	Fuller	Andrew	Vice President, Sales	Dr.	2/19/1952 12:00:00 AM	8/14/1992 12:00:00 AM	908 W. Capital Way	Tacoma	WA
2	3	Leverling	Janet	Sales Representative	Ms.	8/30/1963 12:00:00 AM	4/1/1992 12:00:00 AM	722 Moss Bay Blvd.	Kirkland	WA
3	4	Peacock	Margaret	Sales Representative	Mrs.	9/19/1937 12:00:00 AM	5/3/1993 12:00:00 AM	4110 Old Redmond Rd.	Redmond	WA

```
SELECT *  
FROM Employees  
WHERE NOT City='London' AND  
NOT City='Seattle';
```


Slide 1: Using Wildcards in SQL

Using Wildcards in SQL

Sadie St. Lawrence, MSA
Data Scientist, VSP

UC DAVIS
EXTENSION

Slide 2: Learning Objectives

Learning Objectives

Explain the concept of wildcards
Advantages and disadvantages
Usefulness

Describe how to use the LIKE operator with
wildcards

Write appropriate syntax when using
wildcards

Slide 3: What Are Wildcards?

What Are Wildcards?

Special character used to match parts of a value

Search pattern made from literal text, wildcard character, or a combination

Uses LIKE as an operator (though technically a predicate)

Slide 4: What Are Wildcards?

What Are Wildcards?

Can only be used with strings

Cannot be used for non-text datatypes

Helpful for data scientists as they explore string variables

Slide 5: Using % Wildcards

Using % Wildcards

Wildcard	Action
'%Pizza'	Grabs anything ending with the word pizza
'Pizza%'	Grabs anything after the word pizza
'%Pizza%'	Grabs anything before and after the word pizza

Slide 6: Using % Wildcards

Using % Wildcards

Wildcard	Action
'S%E'	Grabs anything that starts with "S" and ends with "E" (Like Sadie)
't%@gmail.com'	Grabs gmail addresses that start with "t" (hoping to find Tom)

Slide 7: Using % Wildcards

Using % Wildcards

% wildcard will not match NULLs

NULL represents no value in a column

Slide 8: Underscore (_) Wildcard

Underscore (_) Wildcard

Matches a single character

Is not supported by DB2

```
WHERE size LIKE '__ pizza'
```

Output:

```
10 inch pizza
```

```
15 inch pizza
```


Slide 9: Underscore (_) Wildcard

Underscore (_) Wildcard

These produce the same results

```
WHERE size LIKE '__ pizza'
```

Output:

10 inch pizza

15 inch pizza

```
WHERE size LIKE '%pizza'
```

Output:

10 inch pizza

15 inch pizza

8 inch pizza

Slide 10: Bracket [] Wildcard

Bracket [] Wildcard

Used to specify a set of characters in a specific location

Does not work with all DBMS

Does **not** work with **SQLite**

Slide 11: Downsides of Wildcards

Downsides of Wildcards

Takes longer to run

Better to use another operator (if possible):
=, <, =>, and etc.

Statements with wildcards will take longer to run if used at the end of search patterns

Placement of wildcards is important

Slide 1: Sorting with ORDER BY

Sorting with ORDER BY

Sadie St. Lawrence, MSA
Data Scientist, VSP

UC DAVIS
EXTENSION

Slide 2: Learning Objectives

Learning Objectives

Discuss the importance of sorting data for analysis purposes

Explain some of the rules related to using the ORDER BY clause

Use the ORDER BY clause to sort data either in ascending or descending order

Slide 3: Why Sort Data?

Why Sort Data?

Data displayed appears in the order of the underlying tables

Updated and deleted data can change this order

Sequence of retrieved data cannot be assumed if order was not specified

Slide 4: Why Sort Data?

Why Sort Data?

Sorting data logically helps keep information you want on top

ORDER BY clause allows user to sort data by particular columns

```
SELECT something  
FROM database  
ORDER BY characteristic
```

Slide 5: Rules for ORDER BY

Rules for ORDER BY

Takes the name of one or more columns

Add a comma after each additional column name

Can sort by a column not retrieved

Must always be the last clause in a select statement

Slide 6: Sorting by Column Position

Sorting by Column Position

```
ORDER BY 2,3
```

2 means 2nd column

3 means 3rd column, etc.

Slide 7: Sort Direction

Sort Direction

DESC descending order

ASC ascending order

Only applies to the column names it directly precedes

Slide 1: Math Operations

Math Operations

Sadie St. Lawrence, MSA
Data Scientist, VSP

UC DAVIS
EXTENSION

Slide 2: Learning Objectives

Learning Objectives

Perform basic math calculations using your data

Discuss the order of operations

Describe analysis possibilities of using math operators and SQL together

Slide 3: Math Operators

Math Operators

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division

Slide 4: Multiplication Example

Multiplication Example

Total units on order multiplied by the unit price to calculate the total order cost

```
SELECT
ProductID
,UnitsOnOrder
,UnitPrice
,UnitsOnOrder * UnitPrice AS Total_Order_Cost
FROM Products
```

Slide 5: Order of Operations

Order of Operations

Parentheses

Exponents

Multiplication

Division

Addition

Subtraction

“Please excuse my dear Aunt Sally”

Slide 6: Combining Math Operations

Combining Math Operations

Unit price, minus my discount, divided by the quantity

```
SELECT
ProductID
,Quantity
,UnitPrice
,Discount
,(UnitPrice - Discount)/Quantity AS
Total_Cost
FROM OrderDetails;
```


Slide 1: Aggregate Functions

Aggregate Functions

Sadie St. Lawrence, MSA
Data Scientist, VSP

UC DAVIS
EXTENSION

Slide 2: Learning Objectives

Learning Objectives

Describe various aggregate functions

Explain how each of the aggregate functions can help you to analyze data

Use various aggregate functions: AVERAGE, COUNT, MIN, MAX, and SUM to summarize and analyze data and

Describe the DISTINCT function

Slide 3: What are Aggregate Functions?

What are Aggregate Functions?

Used to summarize data

Finding the highest and lowest values

Finding the total number of rows

Finding the average value

Slide 4: Aggregate Functions

Aggregate Functions

Function	Description
AVG ()	Averages a column of values
COUNT ()	Counts the number of values
MIN ()	Finds the minimum value
MAX ()	Finds the maximum value
SUM ()	Sums the column values

Slide 5: AVERAGE Function

AVERAGE Function

Rows containing NULL values are ignored by the AVERAGE function

```
SELECT AVG(UnitPrice) AS avg_price  
FROM products
```

Slide 6: COUNT Function

COUNT Function

COUNT (*) – Counts all the rows in a table containing values or NULL values

```
SELECT COUNT (*) AS  
total_customers  
FROM Customers;
```

Count (column) – Counts all the rows in a specific column ignoring NULL values

```
SELECT COUNT(CustomerID) AS  
total_customers  
FROM Customers
```

Slide 7: MAX and MIN Functions

MAX and MIN Functions

Columns with NULL values are ignored by the MIN and MAX functions

```
SELECT MAX(UnitPrice) AS max_prod_price  
FROM Products
```

```
SELECT MAX(UnitPrice) AS max_prod_price  
,MIN(UnitPrice) AS min_prod_price  
FROM Products
```

Slide 8: SUM Aggregate Function

SUM Aggregate Function

```
SELECT SUM(UnitPrice) AS total_prod_price  
FROM Products
```

```
SELECT SUM(UnitPrice*UnitsInStock) AS total_price  
FROM Products  
WHERE SupplierID = 23;
```


Slide 9: Using DISTINCT on Aggregate Functions

Using DISTINCT on Aggregate Functions

If DISTINCT is not specified, ALL is assumed

Cannot use DISTINCT on COUNT(*)

No value to use with MIN and MAX functions

```
SELECT COUNT(DISTINCT CustomerID)
FROM Customers
```

Slide 1: Grouping Data with SQL

Grouping Data with SQL

Sadie St. Lawrence, MSA
Data Scientist, VSP

UCDAVIS
EXTENSION

Slide 2: Learning Objectives

Learning Objectives

Perform some additional aggregations using the GROUP BY and HAVING clauses

Discuss how NULLs are or aren't affected by the GROUP BY and HAVING clauses

Use the GROUP BY and ORDER BY clauses together to better sort your data

Slide 3: Grouping Data

Grouping Data

Learn how to group data in order to summarize subsets of data

New clauses GROUP BY; HAVING

How to aggregate on a particular value

Slide 4: Grouping Example

Grouping Example

Counts customers after group on region rather than counting the whole table

```
SELECT  
  Region  
  ,COUNT(CustomerID) AS total_customers  
FROM Customers  
GROUP BY Region;
```

Slide 5: Additional GROUP BY Information

Additional GROUP BY Information

GROUP BY clauses can contain multiple columns

Every column in your SELECT statement must be present in a GROUP BY clause, except for aggregated calculations

NULLs will be grouped together if your GROUP BY column contains NULLs

Slide 6: HAVING Clause – Filtering for Groups

HAVING Clause – Filtering for Groups

WHERE does not work for groups

WHERE filters on rows

Instead use HAVING clause to filter for
groups

Slide 7: Grouping Example

Grouping Example

```
SELECT  
CustomerID  
,COUNT (*) AS orders  
FROM Orders  
GROUP BY CustomerID  
HAVING COUNT (*) >=2;
```


Slide 8: WHERE vs. HAVING

WHERE vs. HAVING

WHERE filters before data is grouped

HAVING filters after data is grouped

Rows eliminated by the WHERE clause will not be included in the group

Slide 9: ORDER BY with GROUP BY

ORDER BY with GROUP BY

ORDER BY sorts data

GROUP BY does not sort data

```
SELECT SupplierID
, COUNT(*) AS Num_Prod
FROM Products
WHERE UnitPrice >= 4
GROUP BY SupplierID
HAVING COUNT (*) >=2;
```

Slide 1: Putting It All Together

Putting It All Together

Sadie St. Lawrence, MSA
Data Scientist, VSP

UCDAVIS
EXTENSION

Slide 2: Filtering is Useful

Filtering is Useful

Narrowing down your results

Increasing query & application performance

Understanding your data:

- Finding specific values

- Finding a range of values

- Finding blank values

Slide 3: Key SQL Clauses

Key SQL Clauses

Clause	Description	Required
SELECT	Columns or expressions to be returned	Yes
FROM	Table from which to retrieve data	Only if selecting data from a table
WHERE	Row-level filtering	No
GROUP BY	Group specification	Only if calculating aggregates by group
HAVING	Group-level filter	No
ORDER BY	Output sort order	No