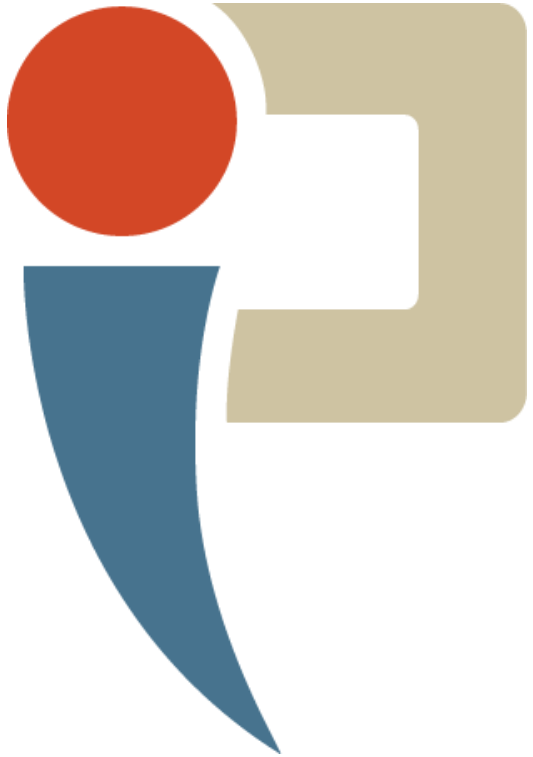


POLUDO INSTITUTE OF TECHNOLOGY & MEDIA

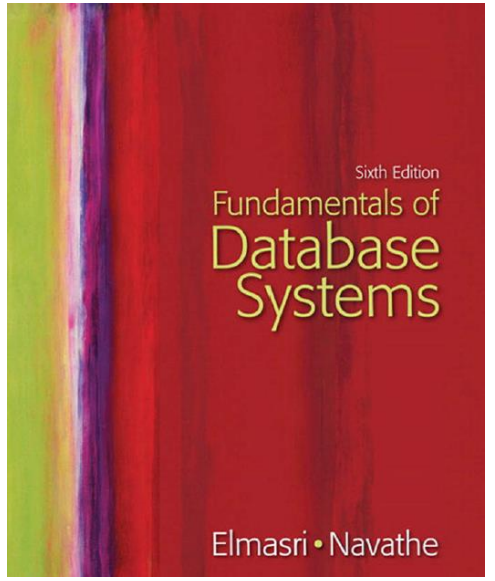
SQL training

Module 2

Instructor: Jesse Silva

Where can we find help?

- Books
  - **Fundamentals of database systems – 6th edition**
  - **Head First SQL - Lynn Beighley**



- Internet
  - W3Schools

    http://www.w3schools.com/sql
  - Structured Query Language @ hit: http://home.hit.no/~hansha/documents/database/documents/Structured%20Query%20Language.pdf
  - SQL Server tutorial @ Lynda.com http://www.lynda.com/SQL-Server-training-tutorial
  - SQL Server tutorials @ microsoft
    - https://msdn.microsoft.com/en-us/library/ms167593(v=sql.105).aspx

What will we see here?

- SQL Syntax
- Use the Select Statement to retrieve data
- Filtering your results : The Where clause
- SQL Wildcards characters
- Working with null values
- SQL Aggregate Functions
- SQL Scalar functions
- Practicing

Syntax

The SQL Syntax

- SQL Syntax is a set of rules and guidelines used to manipulate the database components
- The SQL statements must start with one of these Keywords:
  - SELECT
  - INSERT
  - UPDATE
  - DELETE
  - ALTER
  - DROP
  - CREATE
  - USE
  - SHOW

- All the statements must end with a semicolon
- The statements are case sensitive

SELECT

- The SELECT statement is used to select data from a database
- The result is stored in a result table, called the result-set.

SELECT – Syntax

SPECIFIC COLUMNS
(NO LIMITS, SINCE IT EXISTS)

```
SELECT column_name,column_name,…,column_name
FROM table_name;
```

ALL THE COLUMNS!

```
SELECT *
FROM table_name;
```

SELECT - Example

```
select ContactName,Address,City from Customers
```

```
select  * from Customers
```

SELECT DISTINCT

- We use it to bring only different values
  - We can use the keyword distinct without identify the columns names

Syntax

```
SELECT DISTINCT column_name as alias_name
FROM table_name as alias_name
```

SELECT DISTINCT

Example

- Compare the two results:

```
select city from customers
```

city
Boise
Bräcke
Brandenburg
Bruxelles
Buenos Aires
Buenos Aires
Buenos Aires
Butte
Campinas
Caracas

```
select distinct city from customers
```

city
Bruxelles
Buenos Aires
Butte
Campinas
Caracas
Charleroi
Cork
Cowes
Cunewalde
Elgin
Eugene

WHERE

```
SELECT column_name,column_name,…,column_name
FROM table_name
where column_name operator value;
```

Operators

=	>	>=	between	in	
<> or !=	<	<=	like	AND	OR

SELECT TOP

```
SELECT top percent|number column_name,...,column_name
FROM table_name
```

Example

```
select top 5 * from orders order by orderDate desc
```

ALIASES

```
SELECT column_name as alias_name
FROM table_name as alias_name
```

Example

```
select  CustomerName as ‘Customer name’ from Customers
```

NULL VALUES

Null values represent missing unknown data in SQL databases.

To filter using null values:

```
select * from Customers where Country = null
```

```
select * from Customers where Country is null
```

WILDCARDS

- %

  Replaces zero or more characters

  - `select * from Customers where contactname like’%Moreno'`

- _

  Replaces exactly one character

  - `select * from Customers where City like ’Be_l_n'`

- [*charlist*]

  A range of characters to be matched

  - `select * from Customers where city like '[bsp]%'`

- [^*charlist*] or [!*charlist*]

  A Range of characters that must not be matched

  - `select * from Customers where city like ‘[!bsp]%'`

THE SQL AGGREGATE FUNCTIONS

Perform a calculation on a set of values and return a single value and are frequently used with the GROUP BY clause of the SELECT statement.

AVG

Returns the average value of a field (must be a numeric column).

Ex: `select avg(price) as averagePrice from products`

COUNT

Returns the number of records that matches a specified criteria. As a standard it will not count NULL values.

Ex:

```
select count(price) from products
```

```
select count (*) from products
```

FIRST

Originally the first() function is supported only by the MS Access, but we can use a workaround to have the same result:

```
select TOP 1 City from Customers
```

LAST

As we did with the ***first*** function, we can use a workaround to have the same result, this time using our sorter function:

```
select TOP 1 City from Customers order by City desc
```

MAX

Returns the largest value of the selected column.

Ex:

```
select MAX(price) from Products
```

MIN

Returns the minimum value of the selected column.

Ex:

```
select MIN(price) from Products
```

SUM

Returns the sum of a numeric column. It is not possible to use this function on non-numeric columns

Ex:

```
select sum(price) from products
```

GROUP BY

```
SELECT column_name, aggregate_function(column_name)
FROM table_name
WHERE column_name operator value
GROUP BY column_name;
```

Ex:

```
select productID, sum(quantity)
from OrderDetails
group by productID
```

HAVING

```
SELECT productID, sum(quantity) as #
FROM OrderDetails
GROUP BY productIDID
WHERE sum(quantity) >400
```

```
SELECT productID, sum(quantity) as #
FROM OrderDetails
GROUP BY productID
HAVING sum(quantity) >400
```

THE SQL SCALAR FUNCTIONS

SQL scalar functions return a single value, based on its input values. It doesn't matter what type it is, as long as it's only a single value rather than a table value.

UPPER and LOWER

The UPPER function converts the value of a field to uppercase while the LOWER function convert it to a lowercase.

Ex:

```
select upper(City) from Customers
select ucase(City) from Customers

select lower(City) from Customers
select lcase(City) from Customers
```

SUBSTRING

```
SELECT SUBSTRING(column_name,start_index,length) as
some_name
FROM table_name ...
```

Ex:

```
select substring(CustomerName,1,5)
from Customers
```

LEN

This function only returns the length of the value in a text field. If you use it on number fields, they will be considered as plain text by the function.

Ex:

```
select contactname, customername,
len(customername) as lcn
from Customers
```

ROUND

This function will round a given numeric field to the number of decimals you want.

```
SELECT ROUND(column_name,decimals) as some_name
FROM table_name …
```

Ex:

```
select productname, round(price,1) as
rounded_price from products
```

GETDATE

Returns the current System date and time

Ex:

```
SELECT GETDATE() AS CurrentDateTime
```

FORMAT

This function is used to format the way a field will be displayed.

Ex:

```
SELECT
FORMAT (OrderDate, 'd', 'en-US' ) AS 'US English ',
FORMAT (OrderDate, 'd', 'en-gb' ) AS 'Great Britain English ',
FORMAT (OrderDate, 'd', 'de-de' ) AS 'German ',
FORMAT (OrderDate, 'd', 'zh-cn' ) AS 'Simplified Chinese (PRC)'
from orders;
```

Practicing

Create a Select statement that can give us the average price of the table Products with only two decimal cases.

Practicing

Our Logistic manager asked us to give him the list of all the orders that were placed between the years of 1996 and 1999. He wants this list to contains the id of the employee that worked on the order, the date of the order and also the shipper responsible for the delivery.

Practicing

Now he decided that it is too much information! So he wants us to summarize the information, giving him the total number of order placed by each employee for every shipper.

Practicing

Our Sale department is starting a new Event that will give prizes from our customers as a fidelity program proze. In this phase of the event, they will include only the Countries which the name starts with A, B, C, D, E and F.

To send the prize, they asked us to make a list with the customers name, the contact name and the complete address. We can get this information from the Customers table.

Let’s give them that information based on what we believe is useful for them, but let’s be careful to not give unreadable or useless information.

HOMEWORK

- Visit the https://www.w3schools.com website and explore what we have learned today

- Go ahead and try to know even more about:
  - SQL Wildcards characters
  - SQL Aggregate Functions
  - SQL Scalar functions

- Try some of the exercises and test yourself, as they are only the foundation to what comes next!