Republic of Iraq
The Ministry of Higher Education
& Scientific Research
Al-Ma'arif University College
Department of Computer Science

Lecturer Name: Shihab Hamad Khaleefah
Academic Status: BhD. In Computer Science
Qualification: - Lecturer
Course Material: Database Management System
Grade Level: Second Stage, Lec. 6-8

THE Data Manipulation Language (DML) & Data Query Language (DQL) OPERATORS IN STRUCTURED QUERY LANGUAGE (SQL)

DML & DQL OPERATORS IN STRUCTURED QUERY LANGUAGE

1. INSERT INTO

The **INSERT INTO** statement is used to insert a new row in a table. It is possible to write the INSERT INTO statement in two forms.

The first form doesn't specify the column names where the data will be inserted, only their **Values.**

The first syntax is as follows:

```
INSERT INTO table_name
VALUES (value1, value2, value3,...)
```

Example:

```
INSERT INTO CUSTOMER VALUES ('1000', 'Smith', 'John', 12,
'California', '11111111')
```

The second form specifies both the column names and the values to be inserted:

The first syntax is as follows:

```
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)
```

This form is recommended!

Example:

```
INSERT INTO CUSTOMER (CustomerNumber, LastName, FirstName, AreaCode,
Address, Phone)
VALUES ('1000', 'Smith', 'John', 12, 'California', '11111111')
```

*** Insert Data Only in Specified Columns:**

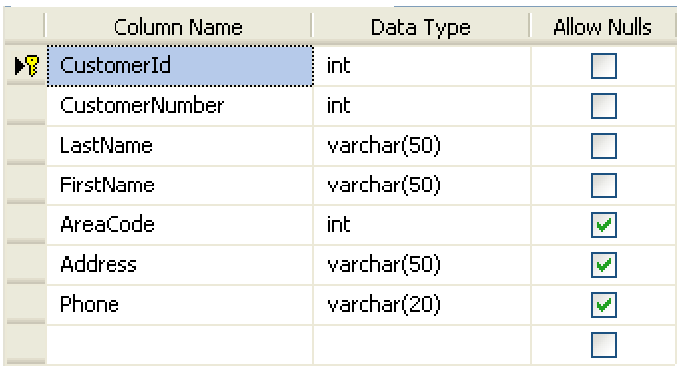
It is also possible to only add data in specific columns.

Example:

```
INSERT INTO CUSTOMER (CustomerNumber, LastName, FirstName)
VALUES ('1000', 'Smith', 'John')
```

Note! You need at least to include all columns that cannot be NULL.

We remember the table definition for the CUSTOMER table:

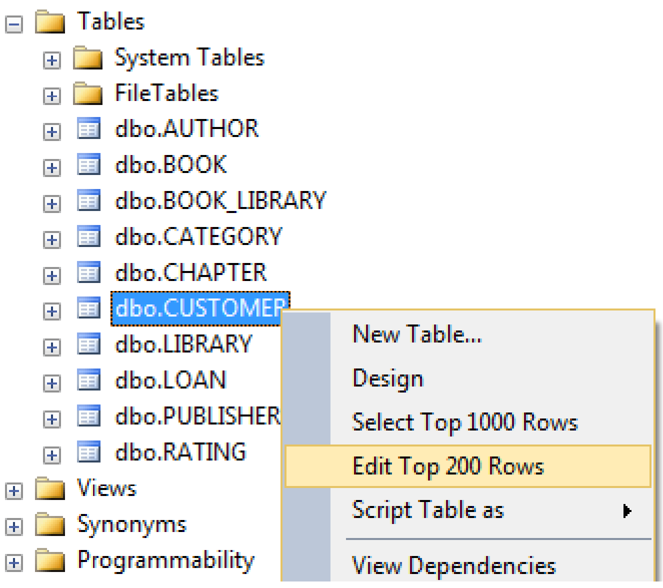


Column Name	Data Type	Allow Nulls
CustomerId	int	☐
CustomerNumber	int	☐
LastName	varchar(50)	☐
FirstName	varchar(50)	☐
AreaCode	int	✓
Address	varchar(50)	✓
Phone	varchar(20)	✓
		☐

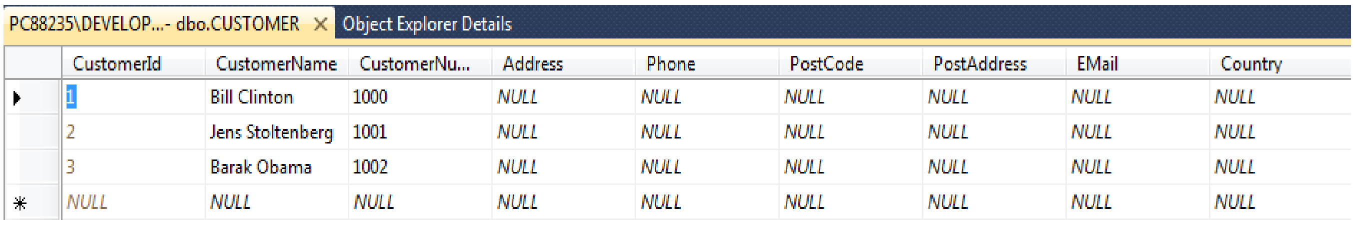
i.e., we need to include at least "CustomerNumber", "LastName" and "FirstName". "CustomerId" is set to "identity(1,1)" and therefore values for this column are generated by the system.

Insert Data in the Designer Tools:

When you have created the tables you can easily insert data into them using the designer tools. Right-click on the specific table and select "Edit Top 200 Rows":



Then you can enter data in a table format, similar to, e.g., MS Excel:



PC88235\DEVELOP...- dbo.CUSTOMER | Object Explorer Details

	CustomerId	CustomerName	CustomerNu...	Address	Phone	PostCode	PostAddress	EMail	Country
▶	1	Bill Clinton	1000	NULL	NULL	NULL	NULL	NULL	NULL
	2	Jens Stoltenberg	1001	NULL	NULL	NULL	NULL	NULL	NULL
	3	Barak Obama	1002	NULL	NULL	NULL	NULL	NULL	NULL
✱	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2. UPDATE

The UPDATE statement is used to update existing records in a table.

The syntax is as follows:

UPDATE table_name

SET column1=value, column2=value2,...

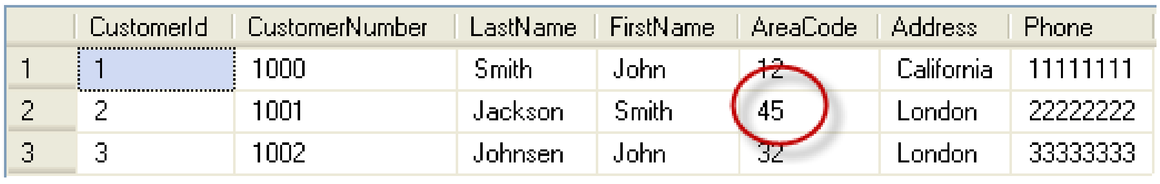
WHERE some_column=some_value

Note! Notice the WHERE clause in the UPDATE syntax. The WHERE clause specifies which record or records that should be updated. If you omit the WHERE clause, all records will be updated!

Example:

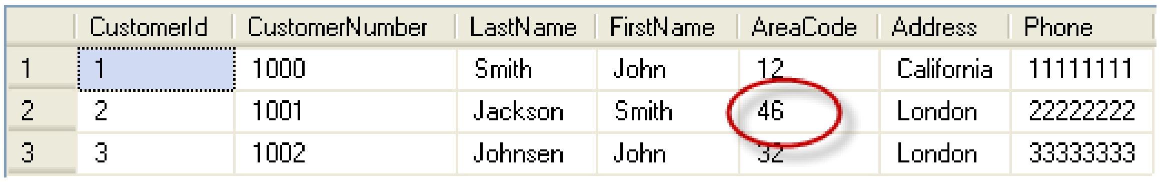
```
update CUSTOMER set AreaCode=46 where CustomerId=2
```

Before update:



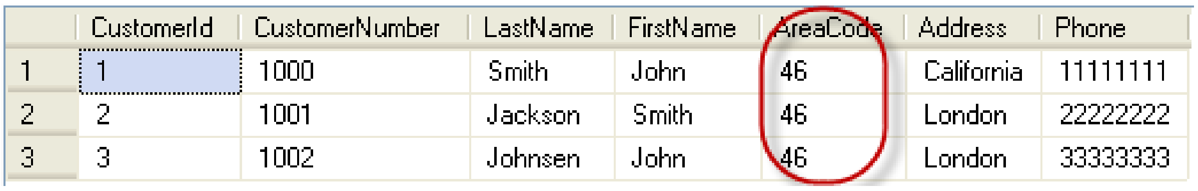
	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	2	1001	Jackson	Smith	45	London	22222222
3	3	1002	Johnsen	John	32	London	33333333

After update:



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	2	1001	Jackson	Smith	46	London	22222222
3	3	1002	Johnsen	John	32	London	33333333

If you don't include the WHERE clause the result becomes:

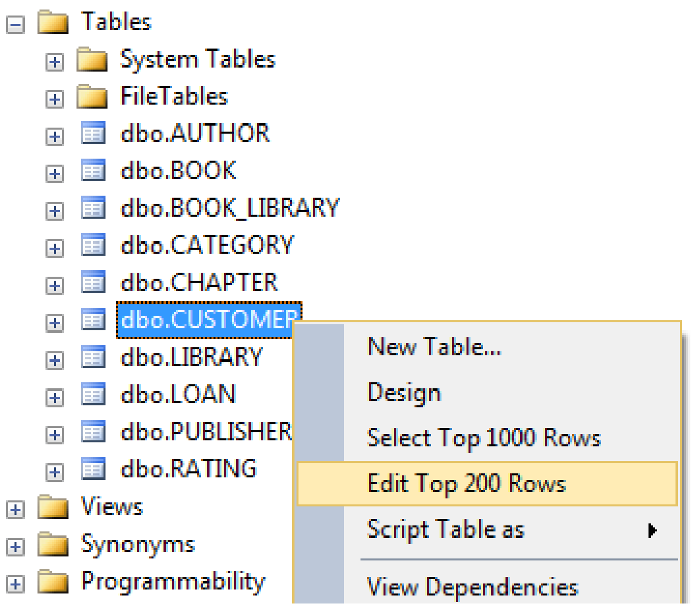


	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	46	California	11111111
2	2	1001	Jackson	Smith	46	London	22222222
3	3	1002	Johnsen	John	46	London	33333333

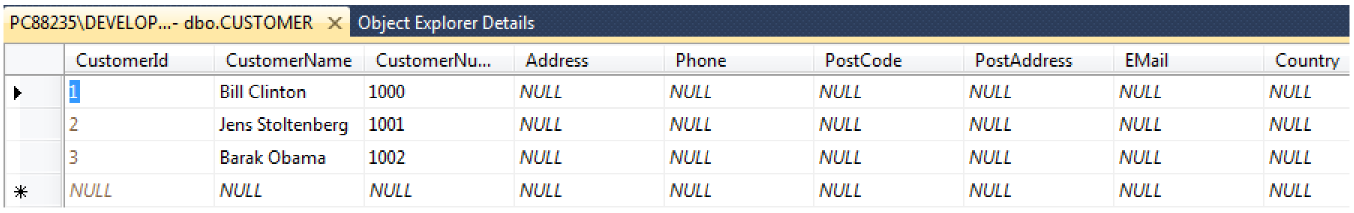
→ So make sure to include the WHERE clause when using the UPDATE command!

Update Data in the Designer Tools:

The same way you insert data you can also update the data. Right-click on the specific table and select "Edit Top 200 Rows":



Then you can change your data:



	CustomerId	CustomerName	CustomerNu...	Address	Phone	PostCode	PostAddress	EMail	Country
▸	1	Bill Clinton	1000	NULL	NULL	NULL	NULL	NULL	NULL
	2	Jens Stoltenberg	1001	NULL	NULL	NULL	NULL	NULL	NULL
	3	Barak Obama	1002	NULL	NULL	NULL	NULL	NULL	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3. DELETE

The DELETE statement is used to delete rows in a table.

The syntax is as follows:

DELETE FROM table_name

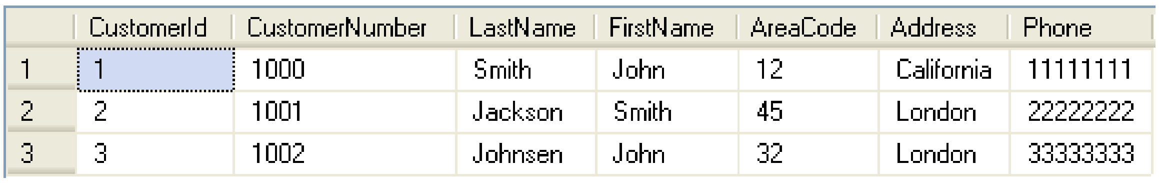
WHERE some_column=some_value

Note! Notice the WHERE clause in the DELETE syntax. The WHERE clause specifies which record or records that should be deleted. If you omit the WHERE clause, all records will be deleted!

Example:

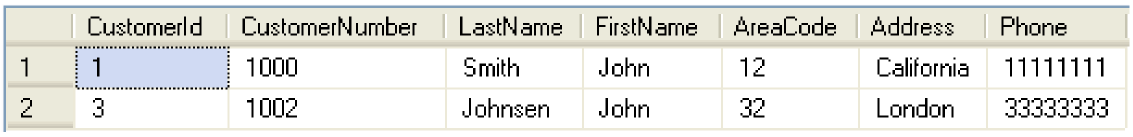
```
delete from CUSTOMER where CustomerId=2
```

Before delete:



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	2	1001	Jackson	Smith	45	London	22222222
3	3	1002	Johnsen	John	32	London	33333333

After delete:



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	3	1002	Johnsen	John	32	London	33333333

Delete All Rows:

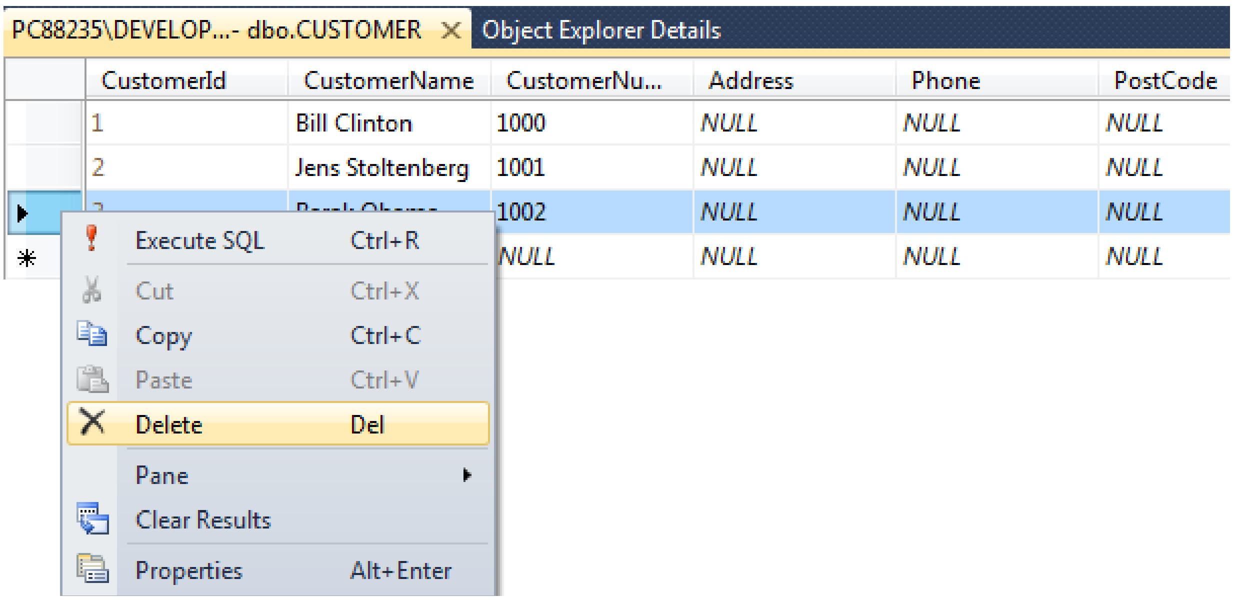
It is possible to delete all rows in a table without deleting the table. This means that the table structure, attributes, and indexes will be intact:

```
DELETE FROM table_name
```

Note! Make sure to do this only when you really mean it! You cannot UNDO this statement!

Delete Data in the Designer Tools:

You delete data in the designer by right-click on the row and select “Delete”:

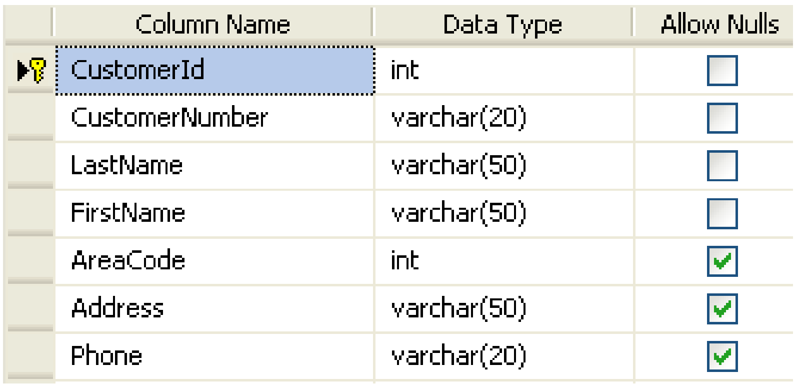


CustomerId	CustomerName	CustomerNu...	Address	Phone	PostCode
1	Bill Clinton	1000	NULL	NULL	NULL
2	Jens Stoltenberg	1001	NULL	NULL	NULL
3	Barak Obama	1002	NULL	NULL	NULL
		NULL	NULL	NULL	NULL

4. SELECT

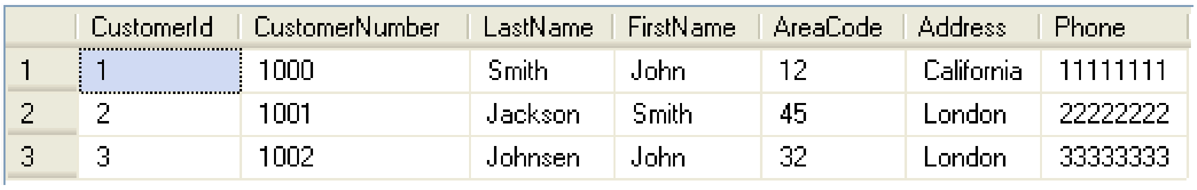
The **SELECT** statement is probably the most used SQL command. The SELECT statement is used for retrieving rows from the database and enables the selection of one or many rows or columns from one or many tables in the database. We will use the CUSTOMER table as an example.

The CUSTOMER table has the following columns:



Column Name	Data Type	Allow Nulls
CustomerId	int	☐
CustomerNumber	varchar(20)	☐
LastName	varchar(50)	☐
FirstName	varchar(50)	☐
AreaCode	int	☑
Address	varchar(50)	☑
Phone	varchar(20)	☑

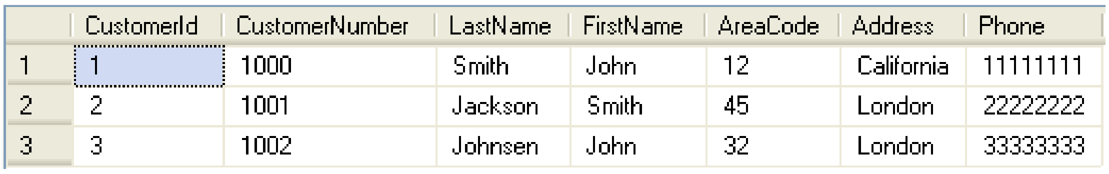
The CUSTOMER table contains the following data:



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	2	1001	Jackson	Smith	45	London	22222222
3	3	1002	Johnsen	John	32	London	33333333

Example:

```
select * from CUSTOMER
```

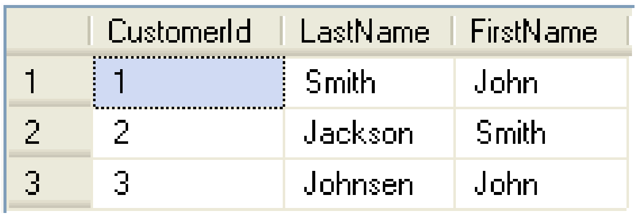


	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	2	1001	Jackson	Smith	45	London	22222222
3	3	1002	Johnsen	John	32	London	33333333

Note! This simple example gets all the data in the table CUSTOMER. The symbol “*” is used when you want to get all the columns in the table.

If you only want a few columns, you may specify the names of the columns you want to retrieve, example:

```
select CustomerId, LastName, FirstName from CUSTOMER
```



	CustomerId	LastName	FirstName
1	1	Smith	John
2	2	Jackson	Smith
3	3	Johnsen	John

So in the simplest form we can use the SELECT statement as follows:

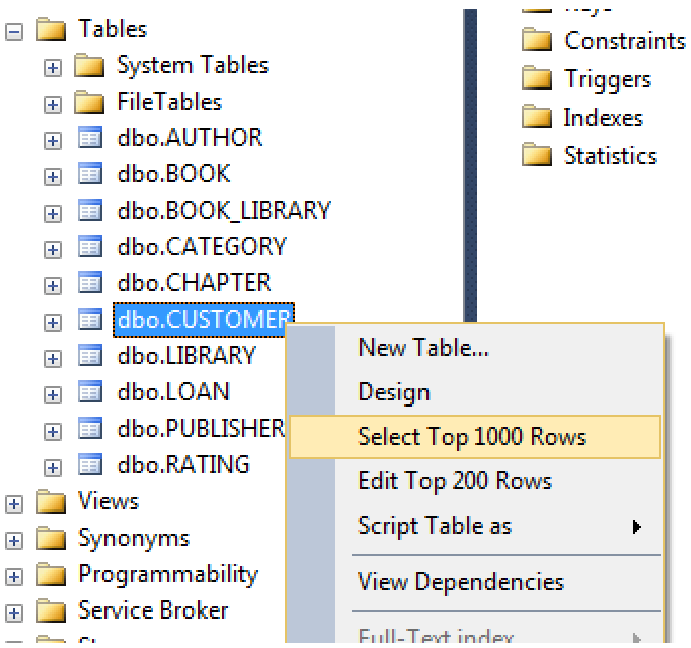
```
select <column_names> from <table_names>
```

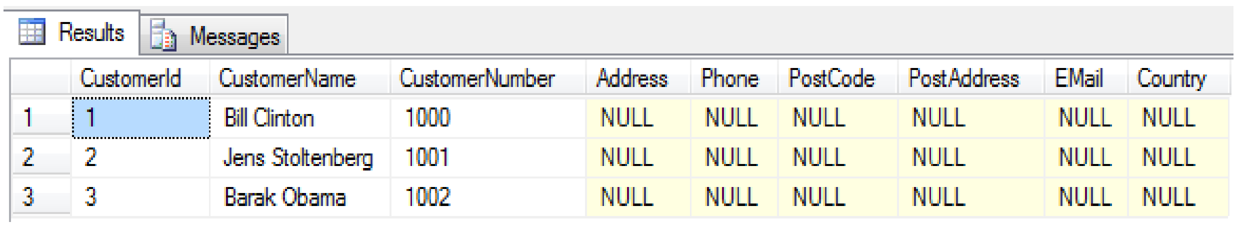
If we want all columns, we use the symbol "*"

Note! SQL is not case sensitive. SELECT is the same as select.

Select Data in the Designer Tools:

Right-click on a table and select "Select Top 1000 Rows":



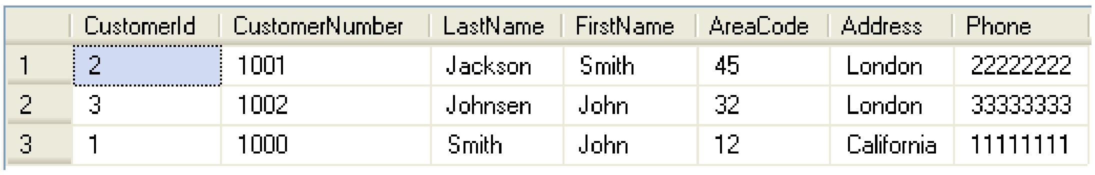

Results | Messages

	CustomerId	CustomerName	CustomerNumber	Address	Phone	PostCode	PostAddress	EMail	Country
1	1	Bill Clinton	1000	NULL	NULL	NULL	NULL	NULL	NULL
2	2	Jens Stoltenberg	1001	NULL	NULL	NULL	NULL	NULL	NULL
3	3	Barak Obama	1002	NULL	NULL	NULL	NULL	NULL	NULL

4.1 The ORDER BY Keyword

If you want the data to appear in a specific order you need to use the "order by" keyword. Example:

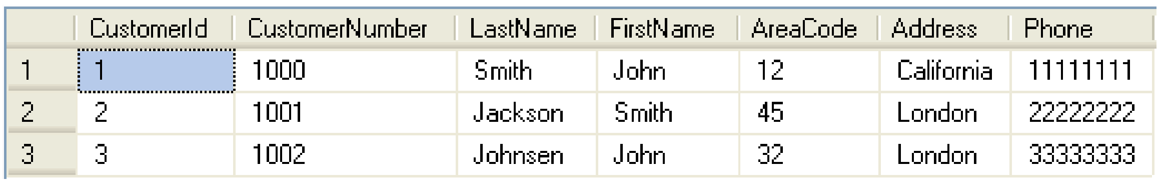
select * from CUSTOMER order by LastName



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	2	1001	Jackson	Smith	45	London	22222222
2	3	1002	Johnsen	John	32	London	33333333
3	1	1000	Smith	John	12	California	11111111

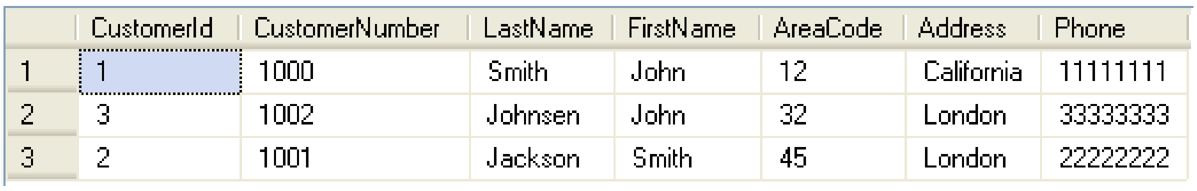
You may also sort by several columns, e.g. like this:

select * from CUSTOMER order by Address, LastName



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	2	1001	Jackson	Smith	45	London	22222222
3	3	1002	Johnsen	John	32	London	33333333

If you use the "order by" keyword, the default order is ascending ("**asc**"). If you want the order to be opposite, i.e., descending, then you need to use the "**desc**" keyword.



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	3	1002	Johnsen	John	32	London	33333333
3	2	1001	Jackson	Smith	45	London	22222222

4.2 SELECT DISTINCT

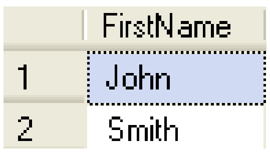
In a table, some of the columns may contain duplicate values. This is not a problem, however, sometimes you will want to list only the different (distinct) values in a table. The DISTINCT keyword can be used to return only distinct (different) values.

The syntax is as follows:

select distinct <column_names> from <table_names>

Example:

```
select distinct FirstName from CUSTOMER
```



	FirstName
1	John
2	Smith

4.3 WHERE Clause

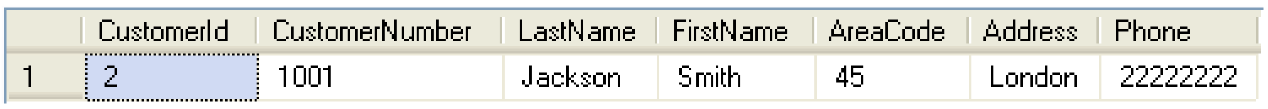
The WHERE clause is used to extract only those records that fulfill a specified criterion.

The syntax is as follows:

select <column_names>
from <table_name>
where <column_name> operator value

Example:

```
select * from CUSTOMER where CustomerNumber='1001'
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	2	1001	Jackson	Smith	45	London	22222222

Note! SQL uses single quotes around text values, as shown in the example above.

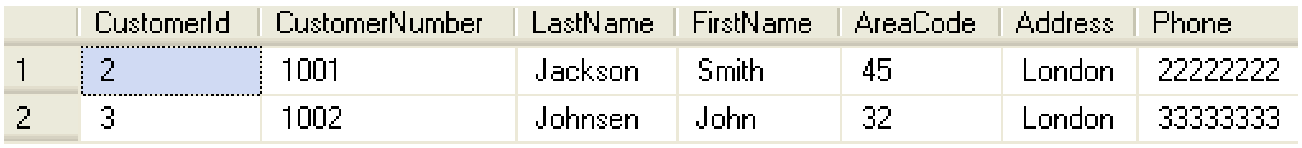
4.3.1 Operators

With the ***WHERE*** clause, the following operators can be used:

Operator	Description
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern
IN	If you know the exact value you want to return for at least one of the columns

Examples:

```
select * from CUSTOMER where AreaCode>30
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	2	1001	Jackson	Smith	45	London	22222222
2	3	1002	Johnsen	John	32	London	33333333

4.3.2 LIKE Operator

SQL includes a string-matching operator for comparisons on character strings. The LIKE operator is used to search for a specified pattern in a column. The operator like uses patterns that are described using two special characters:

- percent (%). The % character matches any substring.
- underscore (_). The _ character matches any character.

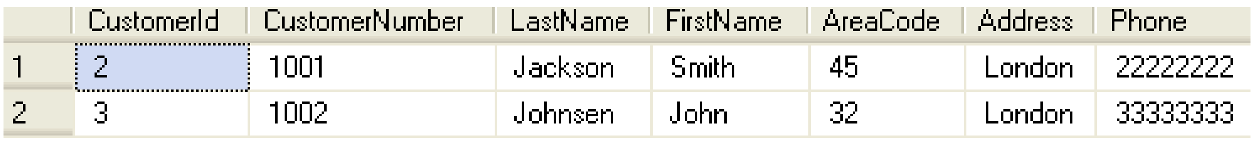
- **Patterns are case sensitive.**
- **Pattern matching examples:**
    - **'Intro%' matches any string beginning with "Intro".**
    - **'%Comp%' matches any string containing "Comp" as a substring.**
    - **'_ _ _' matches any string of exactly three characters.**
    - **'_ _ _ %' matches any string of at least three characters.**

The syntax is as follows:

SELECT column_name(s)
FROM table_name
WHERE column_name LIKE pattern

Example:

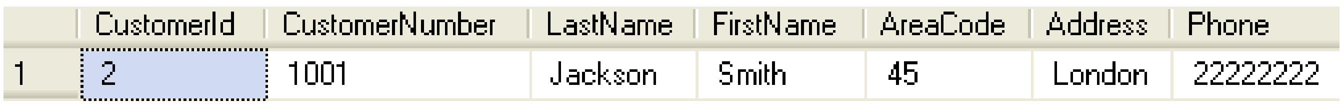
```
select * from CUSTOMER where LastName like 'J%'
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	2	1001	Jackson	Smith	45	London	22222222
2	3	1002	Johnsen	John	32	London	33333333

Note! The "%" sign can be used to define wildcards (missing letters in the pattern) both before and after the pattern.

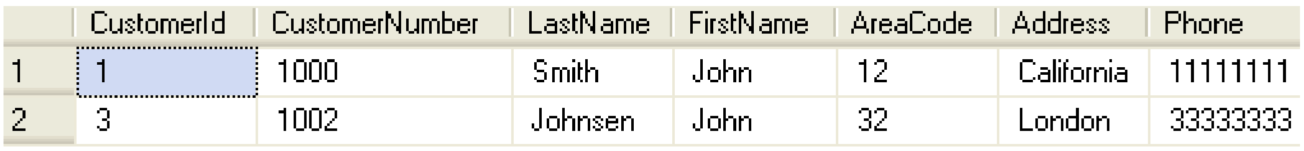
```
select * from CUSTOMER where LastName like '%a%'
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	2	1001	Jackson	Smith	45	London	22222222

You may also combine with the NOT keyword, example:

```
select * from CUSTOMER where LastName not like '%a%'
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	3	1002	Johnsen	John	32	London	33333333

4.3.3 IN Operator

The IN operator allows you to specify multiple values in a WHERE clause.

The syntax is as follows:

```
SELECT column_name(s)
FROM table_name
WHERE column_name  IN (value1,value2,...)
```

// later in the next lecture (Lecture 9) I will discuss - IN Operator- in details with full examples.

4.3.4 BETWEEN Operator

The BETWEEN operator selects a range of data between two values. The values can be numbers, text, or dates.

The syntax is as follows:

```
SELECT column_name(s)
FROM table_name
WHERE column_name
BETWEEN  value1 AND value2
```

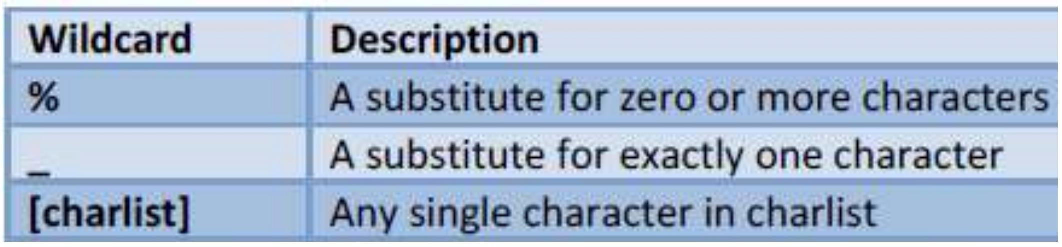
// later in the next lecture (Lecture 9) I will discuss - BETWEEN Operator - in details with full examples.

4.4 Wildcards

SQL wildcards can substitute for one or more characters when searching for data in a database.

Note! SQL wildcards must be used with the SQL LIKE operator.

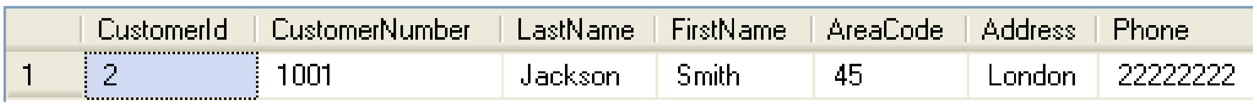
With SQL, the following wildcards can be used:



Wildcard	Description
%	A substitute for zero or more characters
_	A substitute for exactly one character
[charlist]	Any single character in charlist

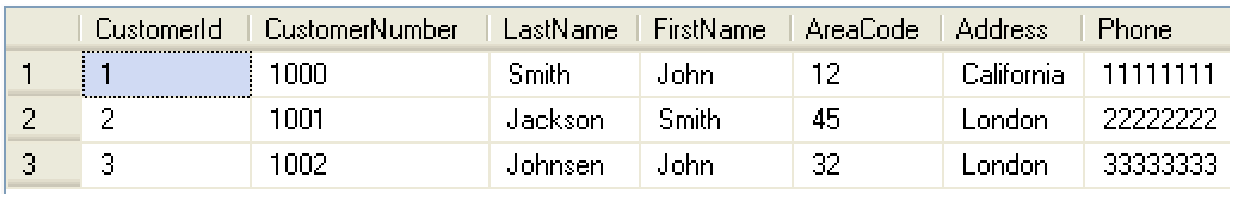
Examples:

```
SELECT * FROM CUSTOMER WHERE LastName LIKE 'J_cks_n'
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	2	1001	Jackson	Smith	45	London	22222222

```
SELECT * FROM CUSTOMER WHERE CustomerNumber LIKE '[10]%'
```



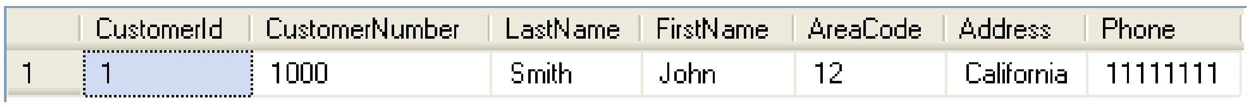
	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	2	1001	Jackson	Smith	45	London	22222222
3	3	1002	Johnsen	John	32	London	33333333

4.5 AND & OR Operators

The AND operator displays a record if both the first condition and the second condition is true. The OR operator displays a record if either the first condition or the second condition is true.

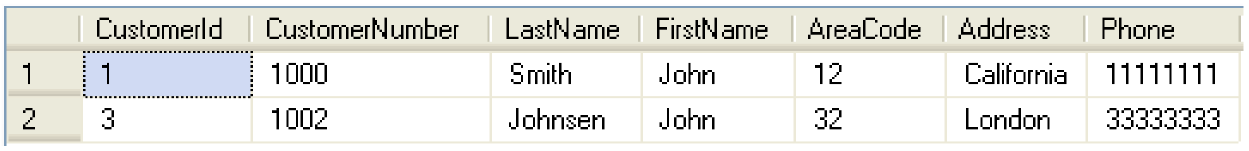
Examples:

```
select * from CUSTOMER where LastName='Smith' and FirstName='John'
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111

```
select * from CUSTOMER where LastName='Smith' or FirstName='John'
```



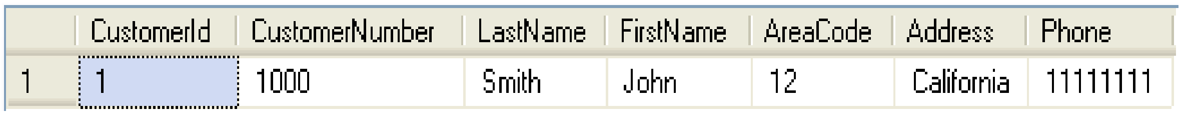
	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	3	1002	Johnsen	John	32	London	33333333

Combining AND & OR:

You can also combine AND and OR (use parenthesis to form complex expressions).

Example:

```
select * from CUSTOMER
where LastName='Smith' and (FirstName='John' or FirstName='Smith')
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111

4.6 SELECT TOP Clause

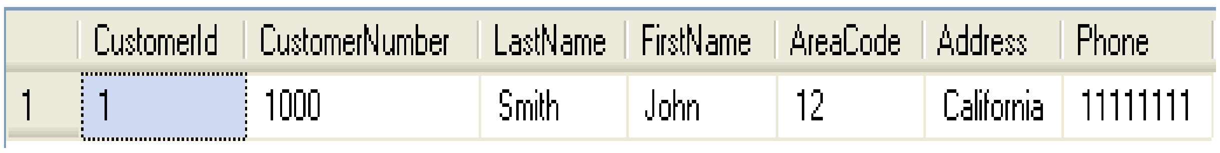
The TOP clause is used to specify the number of records to return.
The TOP clause can be very useful on large tables with thousands of records. Returning a large number of records can impact on performance.

The syntax is as follows:

SELECT TOP number|percent column_name(s)
FROM table_name

Examples:

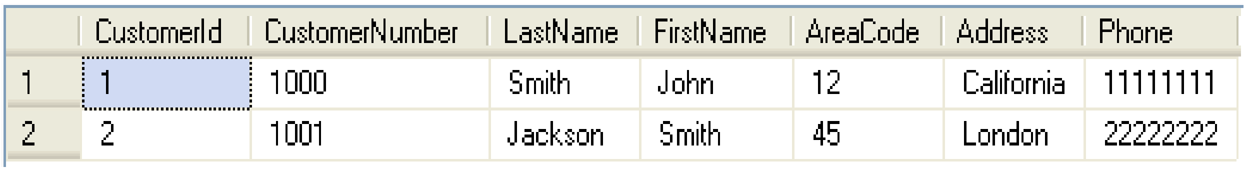
```
select TOP 1 * from CUSTOMER
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111

You can also specify in percent:

```
select TOP 60 percent * from CUSTOMER
```



	CustomerId	CustomerNumber	LastName	FirstName	AreaCode	Address	Phone
1	1	1000	Smith	John	12	California	11111111
2	2	1001	Jackson	Smith	45	London	22222222

// This is very useful for large tables with thousands of records

Summary

In this lecture,

The student's will be able to learn how to Insert Data into all table columns or Only in Specified Columns to the built database and how to deal and used the designer tools to Insert Data. Also will be able to Update and Delete databases by using the SQL query or by using the designer tools, show the cells values (information) or more than column values that stored in the databases by using Select clause. The most important point is to understand the other cooperated operators such as Order by, Distinct, Top Clause and Where Condition Clause and its Operator’s (Like operator, IN operator, AND & OR operators) with entire example illustrates the work of this function.