

Lesson 2: SQL Basics

December 27, 2016 © We Can Code IT. All Rights Reserved.

Audience

- Those who are new to relational DBs and SQL.

Objective

To understand basic **SQL statements**.

To create simple **Queries: SELECT, DISTINCT, Sorting & Ordering**.

To go beyond simple columns: **Concatenation, Mathematical, Dates, Null**.

To understand how to **filter data using the WHERE clause**.

Software & Resources Needed

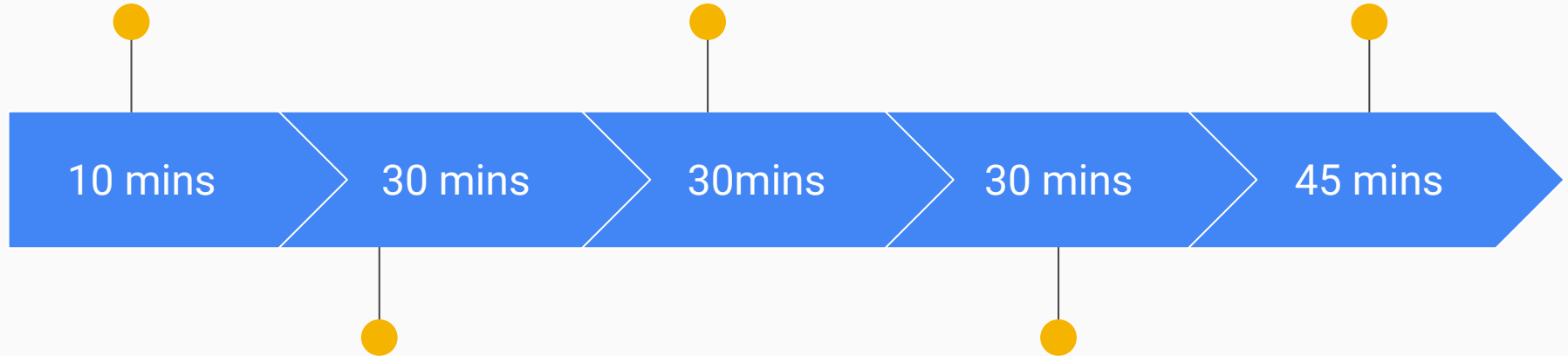
- Database Server with Interface
 - [MS SQL Server with SQL Server Management Studio](#)
- OR
- [MySQL](#) with MySQL Workbench (part of MySQL install)
- OR
- Online (sqlfiddle.com)
- Data
 - [Test data](#) on Github repo.
- [Github repo](#) for information

Overview of Day 2

Basic SQL Queries
(SELECT)

Expressions, Casting,
NULL

Practice



SELECT, DISTINCT
ORDER BY

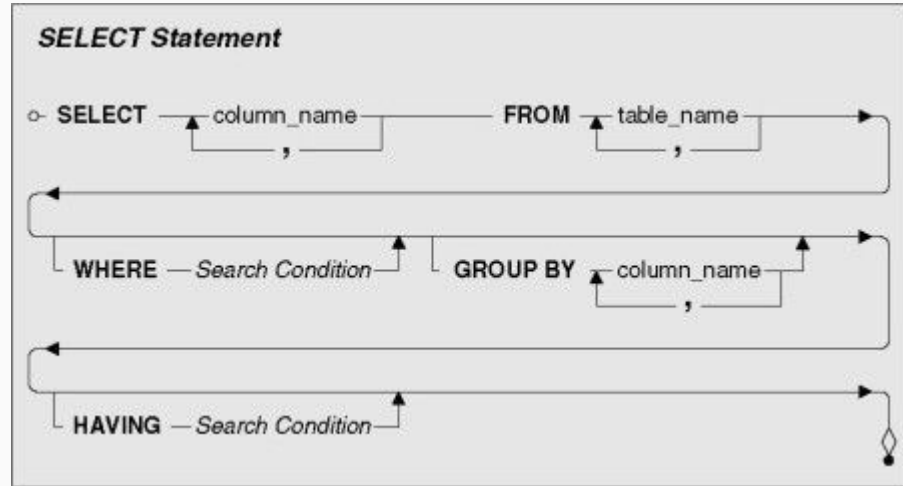
Filtering using WHERE
Clauses

Day 2, Part A: Creating a simple query

Basic SQL Queries

Basics of SELECT Statements

- At the heart of SQL is the SELECT statement.
- You select one or more columns from one or more tables.
- You can choose to filter the results using WHERE or HAVING clauses.
- You can choose to GROUP BY, and ORDER BY (not shown) different criteria.
- **This is how you change mere data into potentially useful information.**



Example

Open up SSMS or MySQL Workbench and using the SalesOrdersExample database, select all columns of the Customers table.

You can select all columns of a table by using an asterisk (*) instead of column names.

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The left pane shows the 'Object Explorer' with the 'SalesOrdersExample' database selected. The 'Tables' folder is expanded, and the 'Customers' table is highlighted. The right pane shows the 'Properties' window for the 'Customers' table, displaying various connection parameters and details.

The central pane shows the 'Query Editor' with the following SQL query:

```
SELECT * from Customers;
```

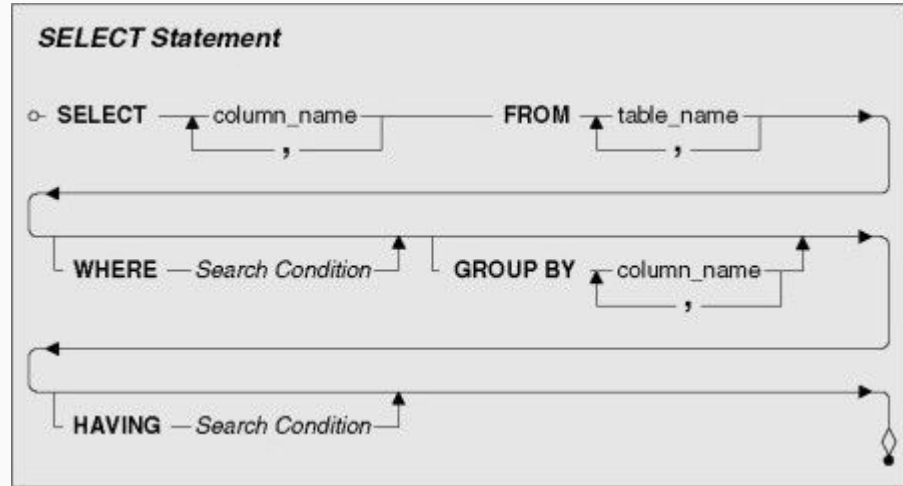
The 'Results' pane displays the query output, showing 10 rows of customer data. The status bar at the bottom indicates that the query was executed successfully, returning 28 rows.

CustomerID	CustFirst Name	CustLast Name	CustStreet Address	CustCity	CustState	CustZip Code	CustArea Code	Cu
1001	Suzanne	Vescas	15127 NE 24th, #383	Redmond	WA	98052	425	55
1002	William	Thompson	122 Spring River Drive	Duvall	WA	98019	425	55
1003	Gary	Hallmark	Route 2, Box 2038	Auburn	WA	98002	253	55
1004	Robert	Brown	672 Lamont Ave	Houston	TX	77201	713	55
1005	Dean	McCrae	4110 Old Redmond Rd.	Redmond	WA	98052	425	55
1006	John	Vescas	15127 NE 24th, #383	Redmond	WA	98052	425	55
1007	Mariya	Sergienko	901 Pine Avenue	Portland	OR	97208	503	55
1008	Neil	Patterson	233 West Valley Hwy	San Diego	CA	92199	619	55
1009	Andrew	Cencini	507 - 20th Ave. E. Apt. 2A	Seattle	WA	98105	206	55
1010	Angel	Kennedy	667 Red River Road	Austin	TX	78710	512	55

Simple Queries: SELECT, DISTINCT, Sorting and Ordering

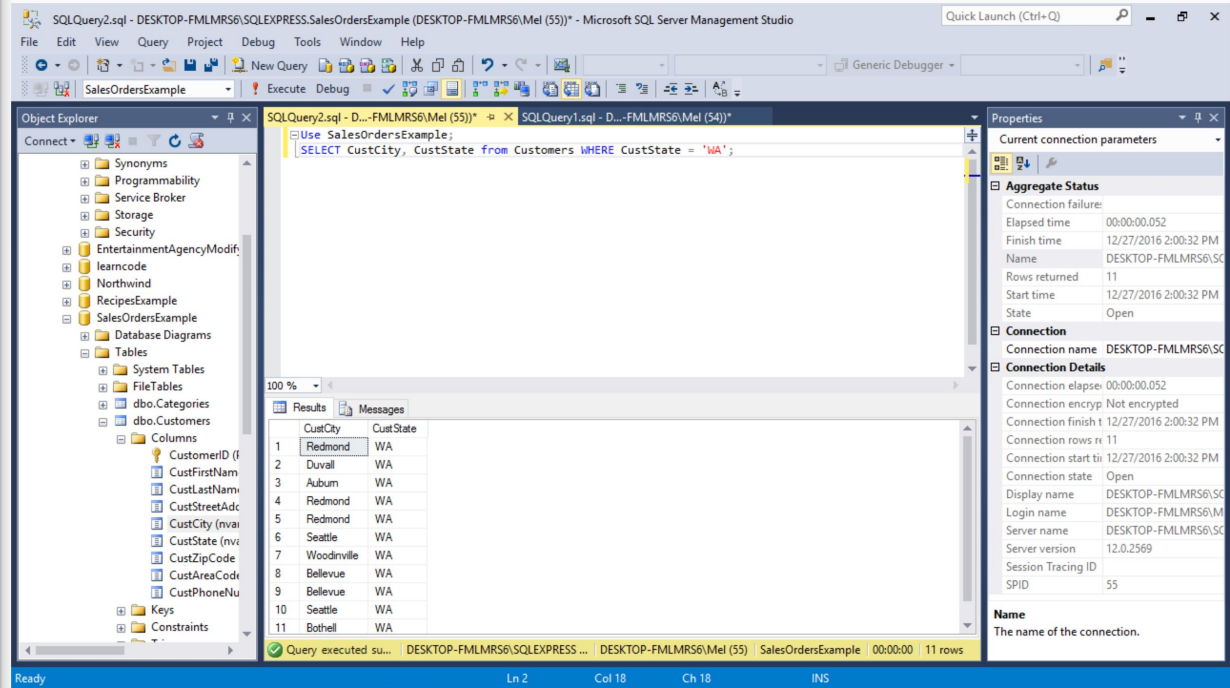
SELECT Statement

- When you query a database, you are using a SELECT statement.
- Formed using clauses (distinct keywords).



Clauses

- **SELECT**
 - Required. The biggie. You must include this. Tells DB what information you want to retrieve.
- **FROM**
 - Also required. Tells db from where to gather the data. You list tables and views here.
- **WHERE**
 - Optional. Filters the rows returned by the FROM clause.



The screenshot displays the Microsoft SQL Server Management Studio interface. The main window shows a query in the SQLQuery2.sql file: `USE SalesOrdersExample; SELECT CustCity, CustState from Customers WHERE CustState = 'WA';`. The Object Explorer on the left shows the database structure, including the SalesOrdersExample database and its tables. The Results pane at the bottom shows the output of the query, which is a list of 11 rows with columns CustCity and CustState. The Properties pane on the right shows the connection details for the current connection.

CustCity	CustState
Redmond	WA
Duvall	WA
Auburn	WA
Redmond	WA
Redmond	WA
Seattle	WA
Woodinville	WA
Bellevue	WA
Bellevue	WA
Seattle	WA
Bothell	WA

Properties pane details:

- Current connection parameters
- Aggregate Status
 - Connection failure:
 - Elapsed time: 00:00:00.052
 - Finish time: 12/27/2016 2:00:32 PM
 - Name: DESKTOP-FMLMRS6\SQL
 - Rows returned: 11
 - Start time: 12/27/2016 2:00:32 PM
 - State: Open
- Connection
 - Connection name: DESKTOP-FMLMRS6\SQL
- Connection Details
 - Connection elapse: 00:00:00.052
 - Connection encrypt: Not encrypted
 - Connection finish: 12/27/2016 2:00:32 PM
 - Connection rows: 11
 - Connection start time: 12/27/2016 2:00:32 PM
 - Connection state: Open
 - Display name: DESKTOP-FMLMRS6\SQL
 - Login name: DESKTOP-FMLMRS6\SQL
 - Server name: DESKTOP-FMLMRS6\SQL
 - Server version: 12.0.2569
 - Session Tracing ID: SPID 55
- Name: The name of the connection.

Clauses (cont)

- GROUP BY
 - If you use an aggregate function in the SELECT clause, you can use this clause to section the information into groups.
 - Aggregate functions are built in functions like SUM, COUNT, etc.
- HAVING
 - Similar to a WHERE clause, but used to filter aggregate functions.

The screenshot shows the Microsoft SQL Server Management Studio interface. The central pane displays a SQL query in the 'SQLQuery2.sql' editor:

```
USE SalesOrdersExample;  
SELECT count(*) as 'Number of Customers Here', CustCity, CustState  
FROM Customers  
WHERE CustState = 'WA'  
GROUP BY CustCity, CustState  
HAVING CustCity in ('Redmond', 'Seattle');
```

The 'Results' pane at the bottom shows the output of the query as a table with three columns: 'Number of Customers Here', 'CustCity', and 'CustState'. The results are as follows:

	Number of Customers Here	CustCity	CustState
1	3	Redmond	WA
2	2	Seattle	WA

The 'Properties' pane on the right shows connection details for the 'DESKTOP-FMLMRS6\SC' connection. The 'Connection Details' section includes:

- Connection name: DESKTOP-FMLMRS6\SC
- Connection elapsed: 00:00:00.098
- Connection encrypt: Not encrypted
- Connection finish time: 12/27/2016 2:03:32 PM
- Connection rows: 2
- Connection start time: 12/27/2016 2:03:31 PM
- Connection state: Open
- Display name: DESKTOP-FMLMRS6\SC
- Login name: DESKTOP-FMLMRS6\M
- Server name: DESKTOP-FMLMRS6\SC
- Server version: 12.0.2569
- Session Tracing ID: SPID
- SPID: 55

The status bar at the bottom indicates 'Query executed successfully' and shows the current position in the query: 'Ln 6 Col 43 Ch 43 INS'.

Example: SELECT all FROM Table

Open up SSMS or MySQL Workbench and using the SalesOrdersExample database, select all columns of the Customers table.

You can select all columns of a table by using an asterisk (*) instead of column names.

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The query editor displays the following SQL query:

```
USE SalesOrdersExample;
SELECT * FROM Customers;
```

The query has been executed, and the results are shown in the Results pane. The table contains 10 rows of customer data:

CustomerID	CustFirst Name	CustLast Name	CustStreet Address	CustCity	CustState	CustZip Code	CustArea Code	CustPhone Number
1001	Suzanne	Viescas	15127 NE 24th, #383	Redmond	WA	98052	425	55
1002	William	Thompson	122 Spring River Drive	Duvall	WA	98019	425	55
1003	Gary	Hallmark	Route 2, Box 2038	Auburn	WA	98002	253	55
1004	Robert	Brown	672 Lamont Ave	Houston	TX	77201	713	55
1005	Dean	McCrae	4110 Old Redmond Rd.	Redmond	WA	98052	425	55
1006	John	Viescas	15127 NE 24th, #383	Redmond	WA	98052	425	55
1007	Mariya	Sergienko	901 Pine Avenue	Portland	OR	97208	503	55
1008	Neil	Patterson	233 West Valley Hwy	San Diego	CA	92199	619	55
1009	Andrew	Cencini	507 - 20th Ave, E. Apt. 2A	Seattle	WA	98105	206	55
1010	Angel	Kennedy	667 Red Red Road	Austin	TX	78710	512	55

The Properties window on the right shows the current connection parameters:

- Aggregate Status:** Connection failure: Elapsed time: 00:00:00.072, Finish time: 12/27/2016 1:48:31 PM, Name: DESKTOP-FMLMR56\SC, Rows returned: 28, Start time: 12/27/2016 1:48:31 PM, State: Open.
- Connection:** Connection name: DESKTOP-FMLMR56\SC.
- Connection Details:** Connection elapsed: 00:00:00.072, Connection encrypt: Not encrypted, Connection finish time: 12/27/2016 1:48:31 PM, Connection rows returned: 28, Connection state: Open, Display name: DESKTOP-FMLMR56\SC, Login name: DESKTOP-FMLMR56\M, Server name: DESKTOP-FMLMR56\SC, Server version: 12.0.2569, Session Tracing ID: SPID 55.

The status bar at the bottom indicates: Ready, Ln 2, Col 25, Ch 25, INS. The query execution summary shows: Query executed successfully, DESKTOP-FMLMR56\SQL EXPRESS..., DESKTOP-FMLMR56\Mel (55), SalesOrdersExample, 00:00:00, 28 rows.

Example: SELECT 2 columns FROM table

Continuing with the SalesOrdersExample database, select the city (CustCity) and state (CustState) of all the customers in the Customers table.

You can select multiple columns from a table by listing the column names after SELECT, separated by commas.

The screenshot displays the Microsoft SQL Server Management Studio interface. The central query editor shows the following SQL code:

```
USE SalesOrdersExample;
SELECT CustCity, CustState FROM Customers;
```

The left-hand Object Explorer pane shows the database structure, with the 'Customers' table under the 'SalesOrdersExample' database selected.

The bottom Results pane displays the query output as a table with 11 rows and 2 columns:

	CustCity	CustState
1	Redmond	WA
2	Duvall	WA
3	Auburn	WA
4	Houston	TX
5	Redmond	WA
6	Redmond	WA
7	Portland	OR
8	San Diego	CA
9	Seattle	WA
10	Austin	TX
11	Woodinville	WA

The right-hand Properties pane shows connection details for the 'DESKTOP-FMLMR56\SQLEXPRESS' connection.

At the bottom, a status bar indicates: 'Query executed successfully' | 'DESKTOP-FMLMR56\SQLEXPRESS...' | 'DESKTOP-FMLMR56\Mel (55)' | 'SalesOrdersExample' | '00:00:00' | '28 rows'.

Example: SELECT one column FROM table

Continuing with the SalesOrdersExample database, select just the city (CustCity) of all the customers in the Customers table.

You can select a single column from a table by listing that column name after the SELECT keyword.

Note the number of rows returned (circled in image to the right).

The screenshot displays the Microsoft SQL Server Management Studio interface. The central pane shows a SQL query: `USE SalesOrdersExample; SELECT CustCity from Customers;`. The left pane shows the Object Explorer with the 'Columns' folder under the 'Customers' table selected. The bottom pane shows the query results in a grid format, listing 11 rows of customer cities. A red circle highlights the status bar at the bottom right, which indicates '28 rows' returned. The right pane shows the Properties window with connection details.

CustCity
Redmond
Duvall
Auburn
Houston
Redmond
Redmond
Portland
San Diego
Seattle
Austin
Woodinville

Query executed successfully. 28 rows returned.

Example: SELECT DISTINCT records FROM table

Continuing with the SalesOrdersExample database, let's learn how to select non-duplicated, aka DISTINCT, cities (CustCity) from the customers in the Customers table.

Note: by using the DISTINCT keyword, we don't get repeated results. Now only 21 (see red circle to the right) records are returned instead of 28. This means we have several customers from the same city.

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor contains the following SQL code:

```
USE SalesOrdersExample;  
SELECT DISTINCT CustCity from Customers;
```

The query results are displayed in the 'Results' pane, showing a list of 21 distinct cities. The status bar at the bottom indicates 'Query executed successfully' and '21 rows'.

CustCity
Auburn
Austin
Bellevue
Bothell
Chico
Dallas
Duvall
El Paso
Fremont
Glendale
Houston

The 'Properties' pane on the right shows the 'Aggregate Status' and 'Connection Details' for the current connection. The 'Rows returned' is 21, which is circled in red.

Example: SELECT DISTINCT records FROM table and ORDER BY state

Continuing with the SalesOrdersExample database, let's learn how to select non-duplicated, aka DISTINCT, cities (CustCity) with their states (CustState) from the customers in the Customers table, Ordering the results by state.

NOTE: The result is often ascending by default. Learn how to change to order of the sort in the next slide.

The screenshot displays the Microsoft SQL Server Management Studio interface. The central query editor shows the following SQL query:

```
USE SalesOrdersExample;  
SELECT DISTINCT CustCity, CustState  
FROM Customers  
ORDER BY CustState;
```

The Object Explorer on the left shows the database structure, including the 'Customers' table. The Results pane at the bottom displays the query output as a table with two columns: 'CustCity' and 'CustState'. The results are ordered by state (CustState) in ascending order.

	CustCity	CustState
1	Chico	CA
2	Fremont	CA
3	Glendale	CA
4	Long Beach	CA
5	Palm Springs	CA
6	San Diego	CA
7	Medford	OR
8	Portland	OR
9	Salem	OR
10	Austin	TX
11	Dallas	TX

The Properties pane on the right shows connection details for the 'DESKTOP-FMLMRS6\SC' connection. The status bar at the bottom indicates the query was executed successfully, returning 21 rows.



Example: SELECT DISTINCT records FROM table and ORDER BY state DESC

Descending order

SQLQuery2.sql - DESKTOP-FMLMRS6\SQLEXPRESS\SalesOrdersExample (DESKTOP-FMLMRS6\Mel (55))* - Microsoft SQL Server Management Studio

File Edit View Query Project Debug Tools Window Help

Object Explorer

- Synonyms
- Programmability
- Service Broker
- Storage
- Security
- EntertainmentAgencyModif
- learncode
- Northwind
- RecipesExample
- SalesOrdersExample
- Database Diagrams
- Tables
 - System Tables
 - FileTables
 - dbo.Categories
 - dbo.Customers
 - Columns
 - CustomerID (PK)
 - CustFirstName
 - CustLastName
 - CustStreetAddress
 - CustCity (nvarchar)
 - CustState (nvarchar)
 - CustZipCode
 - CustAreaCode
 - CustPhoneNumber
 - Keys
 - Constraints

SQLQuery2.sql - D:\...FMLMRS6\Mel (55)*

```
Use SalesOrdersExample;  
SELECT DISTINCT CustCity, CustState  
FROM Customers  
ORDER BY CustState DESC;
```

Results

	CustCity	CustState
1	Auburn	WA
2	Bellevue	WA
3	Bothell	WA
4	Duvall	WA
5	Redmond	WA
6	Seattle	WA
7	Woodinville	WA
8	Austin	TX
9	Dallas	TX
10	El Paso	TX
11	Houston	TX

Properties

Current connection parameters

Aggregate Status

Connection failure:

Elapsed time 00:00:00.109

Finish time 12/27/2016 2:34:03 PM

Name DESKTOP-FMLMRS6\SC

Rows returned 21

Start time 12/27/2016 2:34:03 PM

State Open

Connection

Connection name DESKTOP-FMLMRS6\SC

Connection Details

Connection elapsed: 00:00:00.109

Connection encrypt Not encrypted

Connection finish t 12/27/2016 2:34:03 PM

Connection rows re 21

Connection start ti 12/27/2016 2:34:03 PM

Connection state Open

Display name DESKTOP-FMLMRS6\SC

Login name DESKTOP-FMLMRS6\M

Server name DESKTOP-FMLMRS6\SC

Server version 12.0.2569

Session Tracing ID SPID 55

Name The name of the connection.

Query executed successfully. DESKTOP-FMLMRS6\SQLEXPRESS ... DESKTOP-FMLMRS6\Mel (55) SalesOrdersExample 00:00:00 21 rows

Ready Ln 4 Col 24 Ch 24 INS

Example: SELECT DISTINCT records FROM table and ORDER BY state ASC

Explicitly stating
Ascending order

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The top menu bar includes File, Edit, View, Query, Project, Debug, Tools, Window, and Help. The toolbar contains various icons for file operations, query execution, and debugging. The Object Explorer on the left shows the database structure, including Synonyms, Programmability, Service Broker, Storage, Security, and Tables. The central pane shows the SQL query editor with the following query:

```

-- Use SalesOrdersExample;
SELECT DISTINCT CustCity, CustState
FROM Customers
ORDER BY CustState ASC;

```

The Results pane at the bottom shows the query output:

	CustCity	CustState
1	Chico	CA
2	Fremont	CA
3	Glendale	CA
4	Long Beach	CA
5	Palm Springs	CA
6	San Diego	CA
7	Medford	OR
8	Portland	OR
9	Salem	OR
10	Austin	TX
11	Dallas	TX

The Properties pane on the right shows the connection details for 'DESKTOP-FMLMRS6\SQLEXPRESS'.

Current connection parameters

- Connection name: DESKTOP-FMLMRS6\SQLEXPRESS
- Connection type: Local
- Connection timeout: 30
- Connection encryption: Not encrypted
- Connection finish time: 12/27/2016 2:36:22 PM
- Connection rows returned: 21
- Connection start time: 12/27/2016 2:36:22 PM
- Connection state: Open
- Display name: DESKTOP-FMLMRS6\SQLEXPRESS
- Login name: DESKTOP-FMLMRS6\SQLEXPRESS
- Server name: DESKTOP-FMLMRS6\SQLEXPRESS
- Server version: 12.0.2569
- Session Tracing ID: SPID
- SPID: 55

Name

The name of the connection.

Example: SELECT records FROM table and ORDER BY 2 columns descending

Using the EntertainmentAgencyExample database, let's see how to select first and last names of agents, and have the results come to us by last name (descending), then first name (descending).

The screenshot displays the Microsoft SQL Server Management Studio interface. The 'Object Explorer' on the left shows the 'EntertainmentAgencyExample' database structure. The 'Query Editor' in the center contains the following SQL query:

```
USE EntertainmentAgencyExample;  
SELECT AgtFirstName, AgtLastName  
FROM Agents  
ORDER BY AgtLastName DESC, AgtFirstName DESC;
```

The 'Results' pane at the bottom shows the output of the query, which is a table with two columns: 'AgtFirstName' and 'AgtLastName'. The results are ordered by last name in descending order, and then by first name in descending order. The status bar at the bottom indicates 'Query executed successfully' and '9 rows'.

AgtFirstName	AgtLastName
Marianne	Wier
Carol	Vescas
Caleb	Vescas
William	Thompson
Karen	Smith
Maria	Patterson
John	Kennedy
Duffy	Dumbwit
Scott	Bishop

The 'Properties' pane on the right shows the 'Current connection parameters' for the connection named 'DESKTOP-FMLMRS6\SQLEXPRESS'.

Try It

Select all records from Agents table in the EntertainmentAgencyExample database.

Try It

Select first and last names of the Customers in the
EntertainmentAgencyExample database.

Try It

Select first and last names of the Agents, ordering by last name (ascending) in the EntertainmentAgencyExample database.

Try It

Select distinct states in the Agents table, ordering by state (decending) using the EntertainmentAgencyExample database.

Try It

using the EntertainmentAgencyExample database,
select cities and states from the Agents table, ordering first by state
(decending), then by city (descending).

Continued in Day
2, Part B: Getting
more than simple
columns