Class: BCA Semester -VI
Subject: Data Warehousing with SQL Server 2012
Academic Year – 2018 – 2019

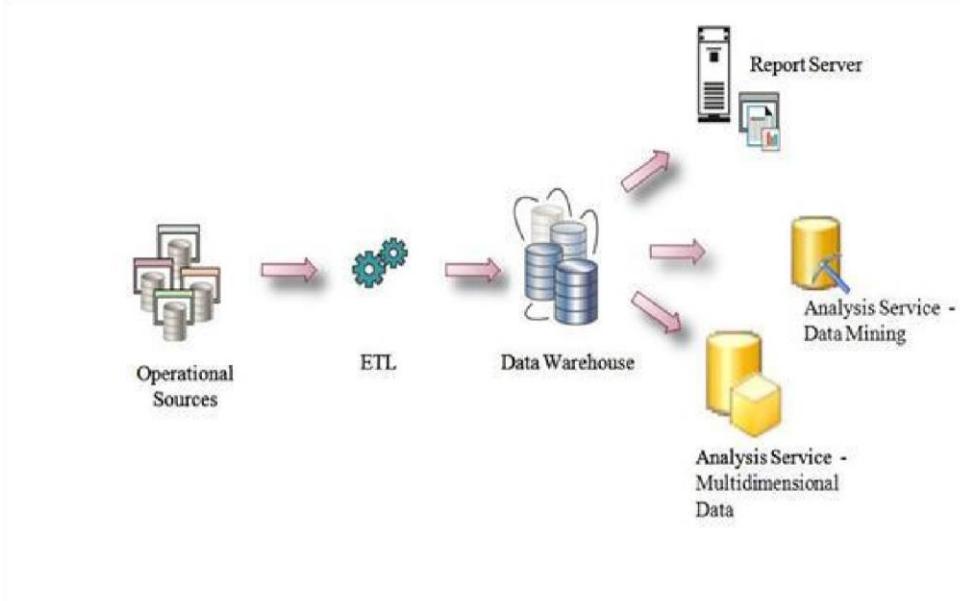
UNIT-5

Deploying and Configuring SSIS Packages, Consuming Data in Data Warehouse

Overview of SSIS Development

Overview of SSIS

SQL Server Integration Services (SSIS) is a platform for building high performance data integration and workflow solutions. It allows creation of packages or SSIS packages which are made up of tasks that can move data from source to destination and alter it if required. SSIS is basically an ETL (Extraction, Transformation, and Load) tool whose main purpose is to do extraction, transformation and loading of data but it can be used for several other purposes, for example, to automate maintenance of SQL Server databases, update multidimensional cube data or send e-mails detailing the status of the operation as defined by the user. SSIS is a component of SQL Server 2005/2008 and is the successor of DTS (Data Transformation Services) which had been in SQL Server 7.0/2000.



Typical Use of Integration Services:

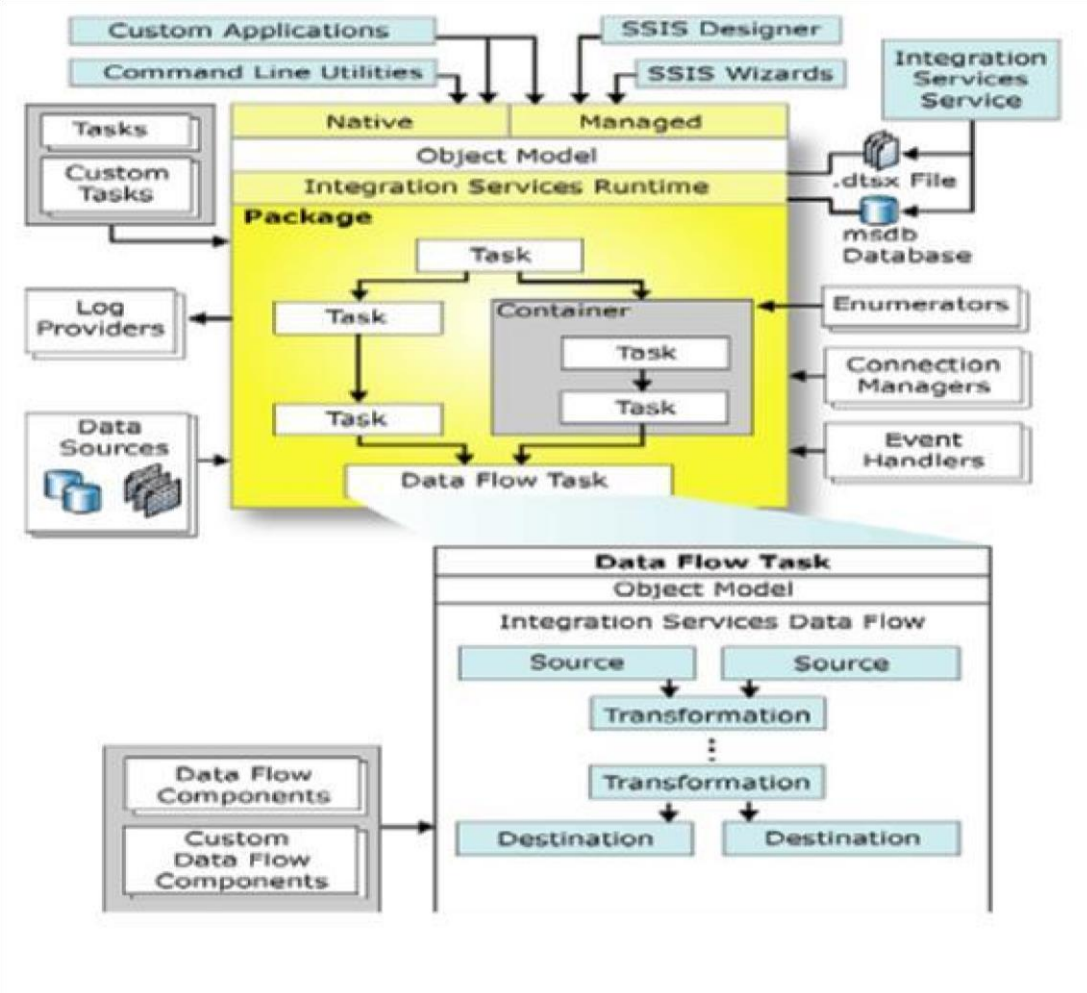
- Merging Data from Heterogeneous Data Stores

- Populating Data Warehouses and Data Marts
- Cleaning and Standardizing Data
- Building Business Intelligence into a Data Transformation Process Automating Administrative Functions and Data Loading

SSIS Architecture

Microsoft SQL Server Integration Services (SSIS) consist of four key parts:

- SSIS Service
- SSIS Object Model
- SSIS runtime engine and the runtime executables
- SSIS dataflow engine and the dataflow components



Integration Services Service

- Monitors running Integration Services packages and manages the storage of packages
- Integration Services object model
- Includes native and managed application programming interfaces (API) for accessing
- Integration Services tools, command-line utilities, and custom applications

SSIS Run-time Engine & executables

1. Runs packages
2. Supports logging, debugging, config, connections, & transactions

3. SSIS Run-time executables
4. Package, Containers, Tasks and Event Handlers

SSIS Data-flow Engine & components

- Provides In-Memory buffers to move data
- Calls Source Adaptors to files & DBs
- Provides Transformations to modify data
- Destination Adaptors to load data into data stores
- Components
- Source, Destination Adaptors & transformations

Overview of SSIS Development

Integration Services includes two studios for working with packages:

- SQL Server Data Tools (SSDT) for developing the Integration Services packages that a business solution requires. SQL Server Data Tools (SSDT) provides the Integration Services project in which you create packages.
- SQL Server Management Studio for managing packages in a production environment.

SQL Server Data Tools

Working in SQL Server Data Tools (SSDT), you can perform the following tasks:

- Run the SQL Server Import and Export Wizard to create basic packages that copy data from a source to a destination.
- Create packages that include complex control flow, data flow, event-driven logic, and logging.
- Test and debug packages by using the troubleshooting and monitoring features in SSIS Designer, and the debugging features in SQL Server Data Tools (SSDT).
- Create configurations that update the properties of packages and package objects at run time.
- Create a deployment utility that can install packages and their dependencies on other computers.
- Save copies of packages to the SQL Server msdb database, the SSIS Package Store, and the file system.

SQL Server Management Studio

SQL Server Management Studio provides the Integration Services service that you use to manage packages, monitor running packages, and determine impact and data lineage for Integration Services and SQL Server objects.

Working in SQL Server Management Studio, you can perform the following tasks:

- Create folders to organize packages in a way that is meaningful to your organization.
- Run packages that are stored on the local computer by using the Execute Package utility.
- Run the Execute Package utility to generate a command line to use when you run the **dtexec**command prompt utility (dtexec.exe).
- Import and export packages to and from the SQL Server msdb database, the SSIS Package Store, and the file system.

Deploying SSIS Projects

Deploying is nothing but moving something from one place to another place. SSIS project deployment is nothing but uploading / moving project to production server or main server.

SSISDB Catalog

SSISDB catalog serves as the SSIS project and package repository and is the recommended deployment target for SQL Server 2012 SSIS solutions.

The SSISDB catalog stores SSIS metadata and definitions: projects, packages, parameters, and environment properties. When SSIS packages are executed, they are loaded from the SSISDB catalog, together with the corresponding configurations (parameters and connection managers).

The SSISDB catalog also stores information gathered from SSIS executions, to provide an overview of the operational history.

Quick Check

1. What is SSISDB?
2. What is SSIS server?
3. How can SSIS projects be deployed?

Quick Check Answers

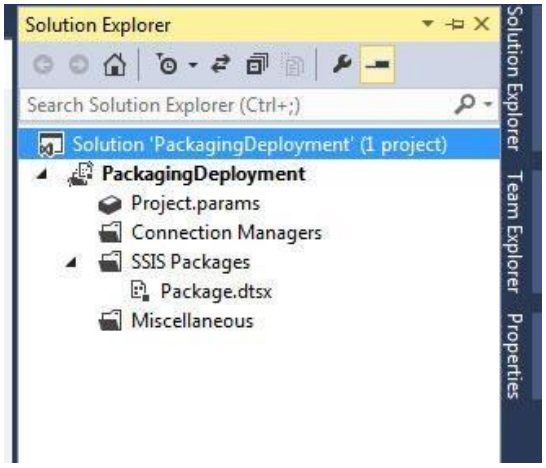
1. SSISDB is a special database provided by SQL Server 2012 to be used as the principal SSIS solution repository.
2. SSIS server is a name used to refer to an instance of SQL Server hosting the SSISDB catalog. Any instance of SQL Server 2012 can be used as the SSIS server, except SQL Server Express.
3. Under the project deployment model, SSIS project deployment is integrated into SQL Server Data Tools (SSDT), as well as SQL Server Management Studio (SSMS).

SSIS Project Deployment Step by Step

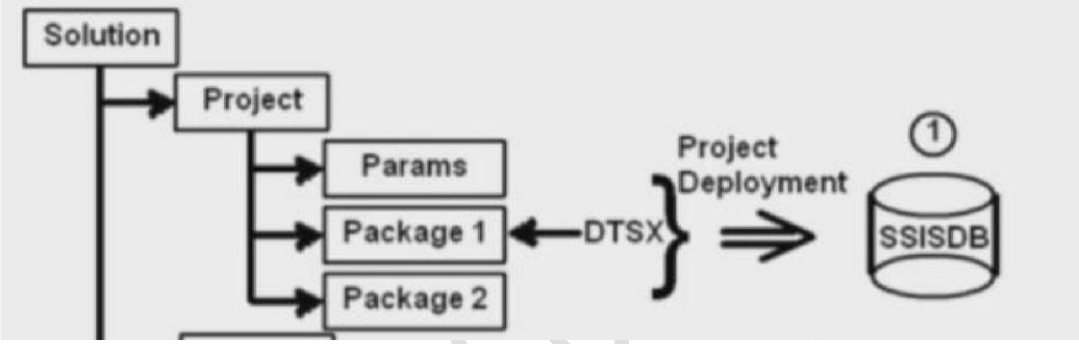
Step 1

Create a new fresh SSIS project. Take appropriate control flow and data flow task. Apply transformation.

In order to understand deployment completely first we need to see what will get deployed to production server.



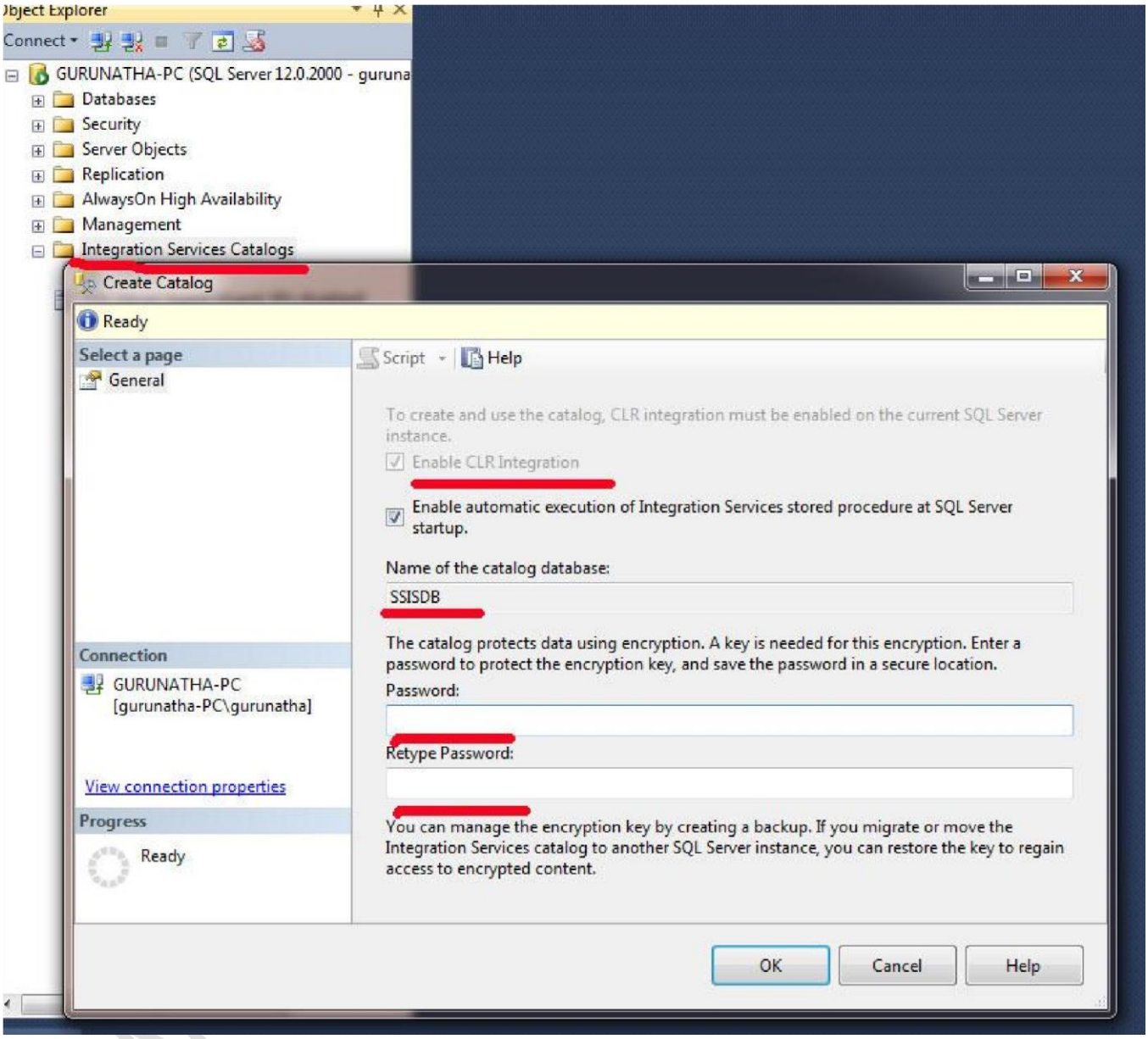
It shows how solution of a SSIS project is structured if you see above image there is a solution -> project -> parameters & package -> Package.dtsx as shown in below image.



So when we do package depolyment all this files gets uploaded to production server (SQL SERVER SSISDB).

Step 2

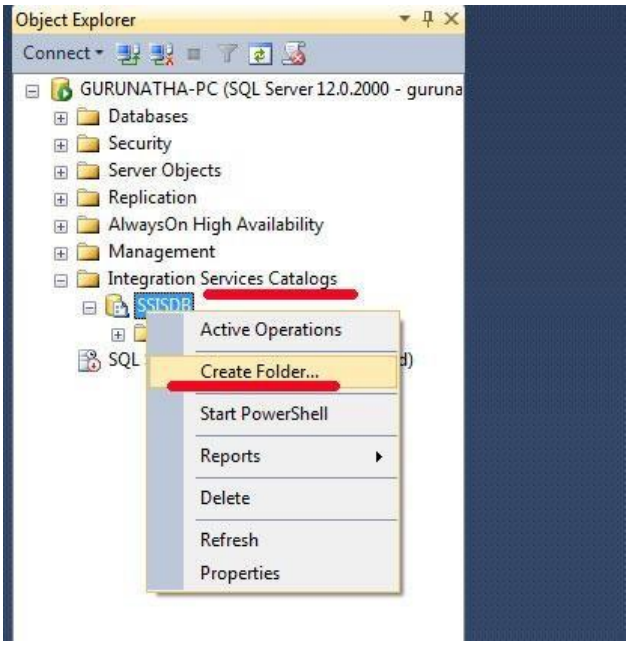
This is optional step but still if any one doesn't have SSISDB in their SQL Server for them it is useful. Open your SQL Management Studio and Find Integration Services Catalogs -> right click and create catalog -> Check Enable CLR Integration to ensure smooth running of catalog -> Type your password and save it.

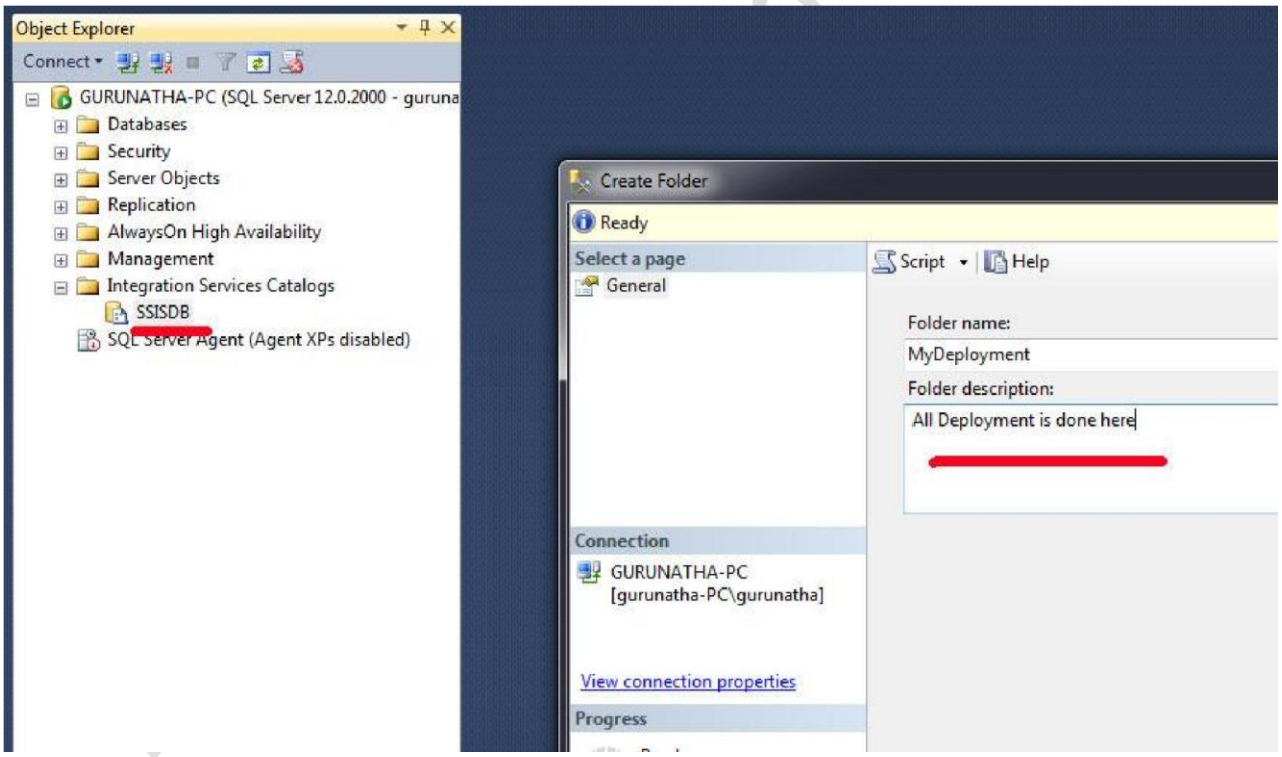


This will create your SSISDB catalog under Integration Services Catalogs.

Step 3

Once your SSISDB catalog is ready just right click on that and create folder why because to save our deployment files we need a folder under SSISDB catlog.

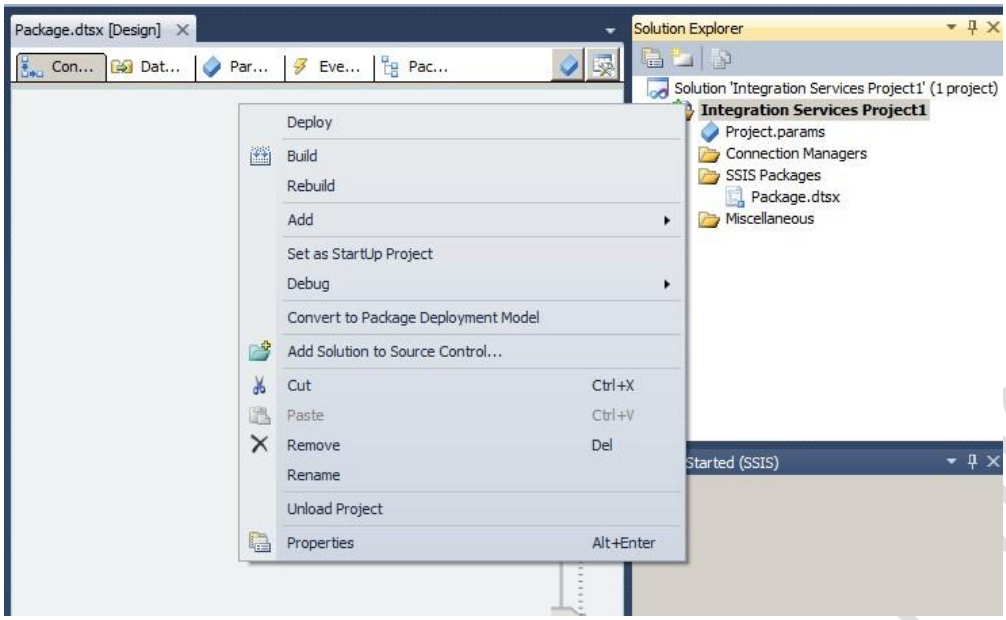




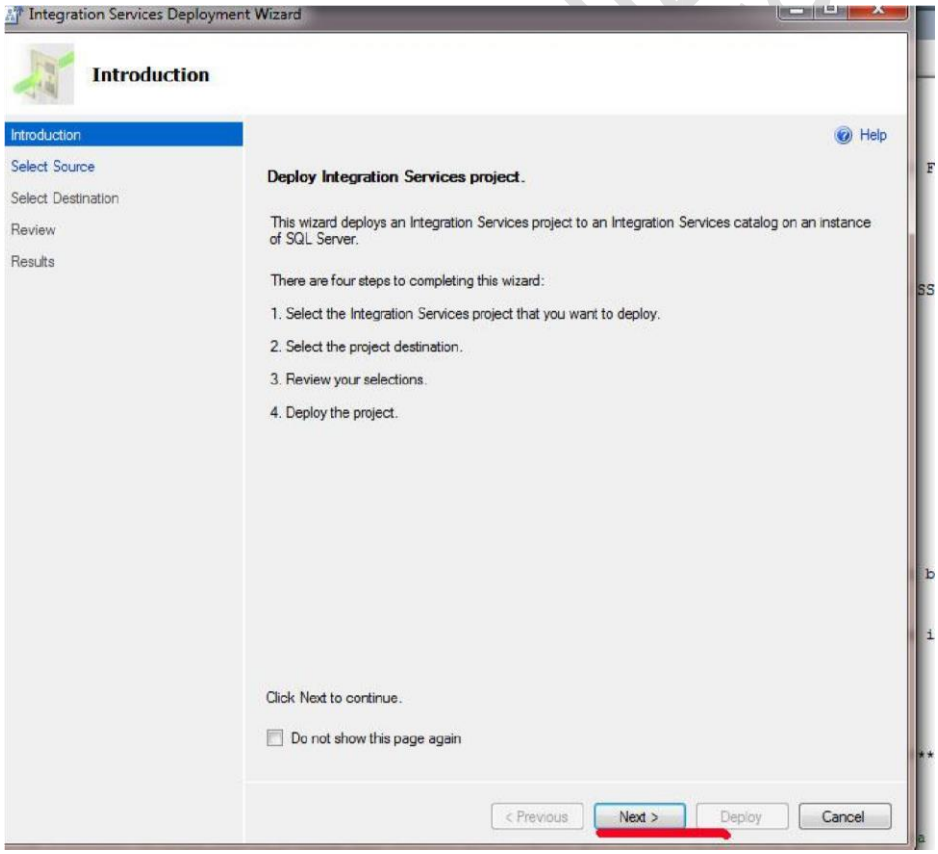
As you see we have successfully created a folder under SSISDB. Now this will our place to store deployment project.

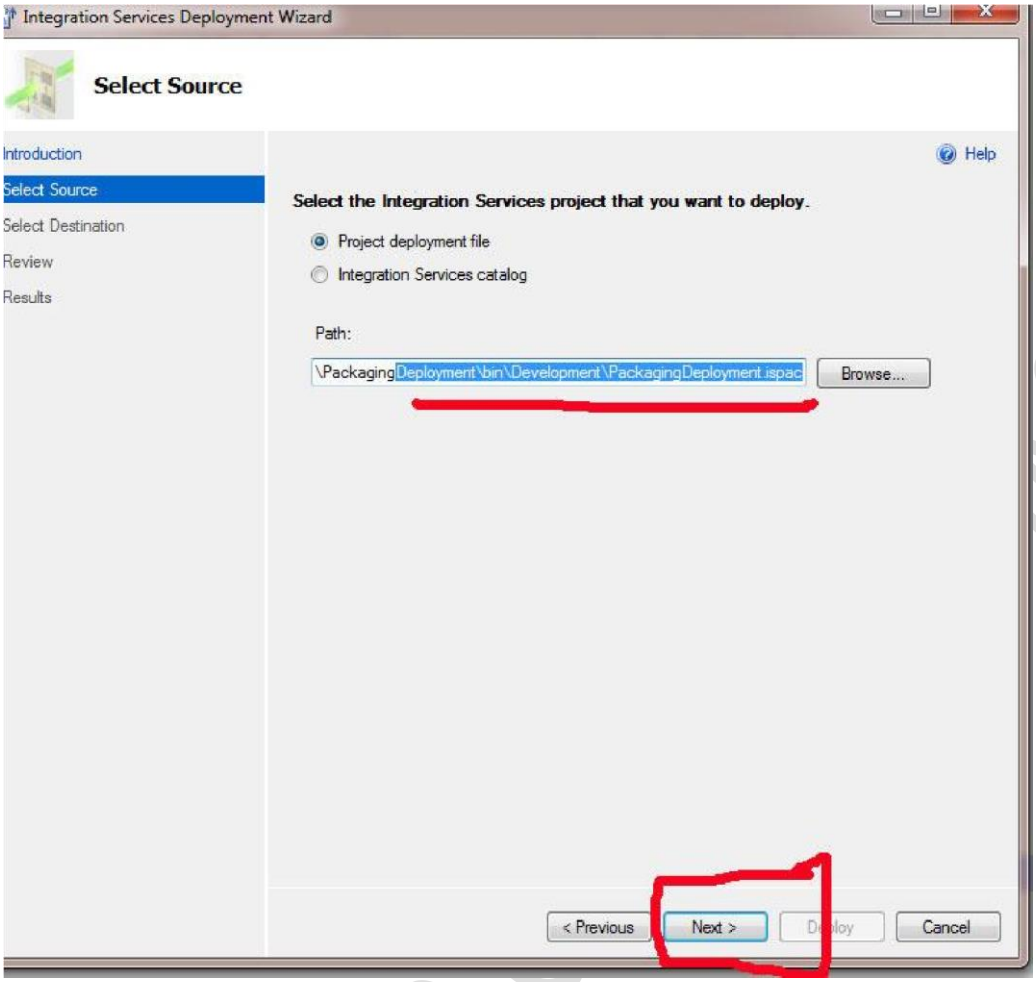
Step 4

Now go to SSIS project and go to solution explorer do the steps as shown in below image.

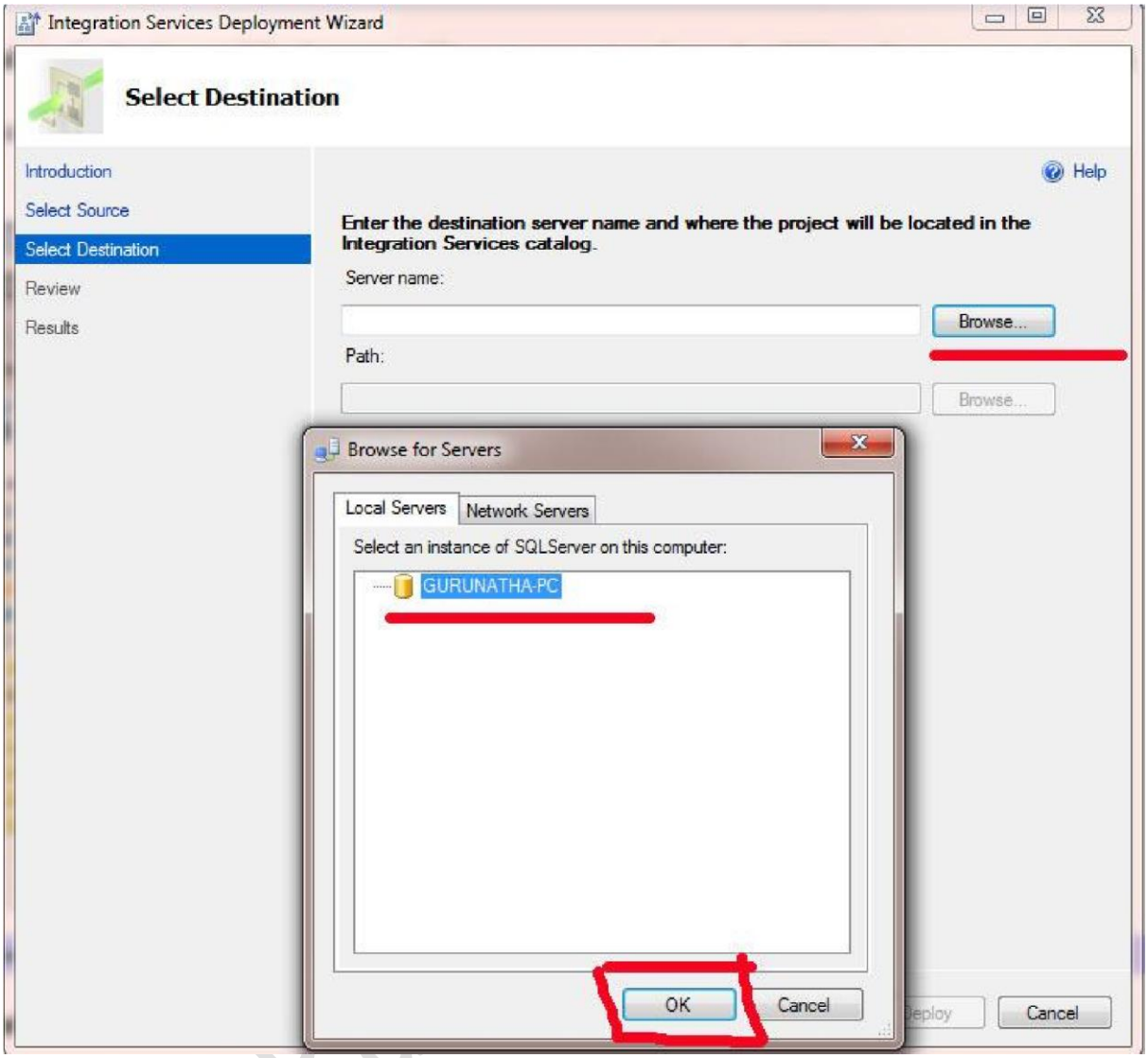


Right click on integration service project. It will open a pop-up from there click on deploy. It will open following wizard from there click on next.

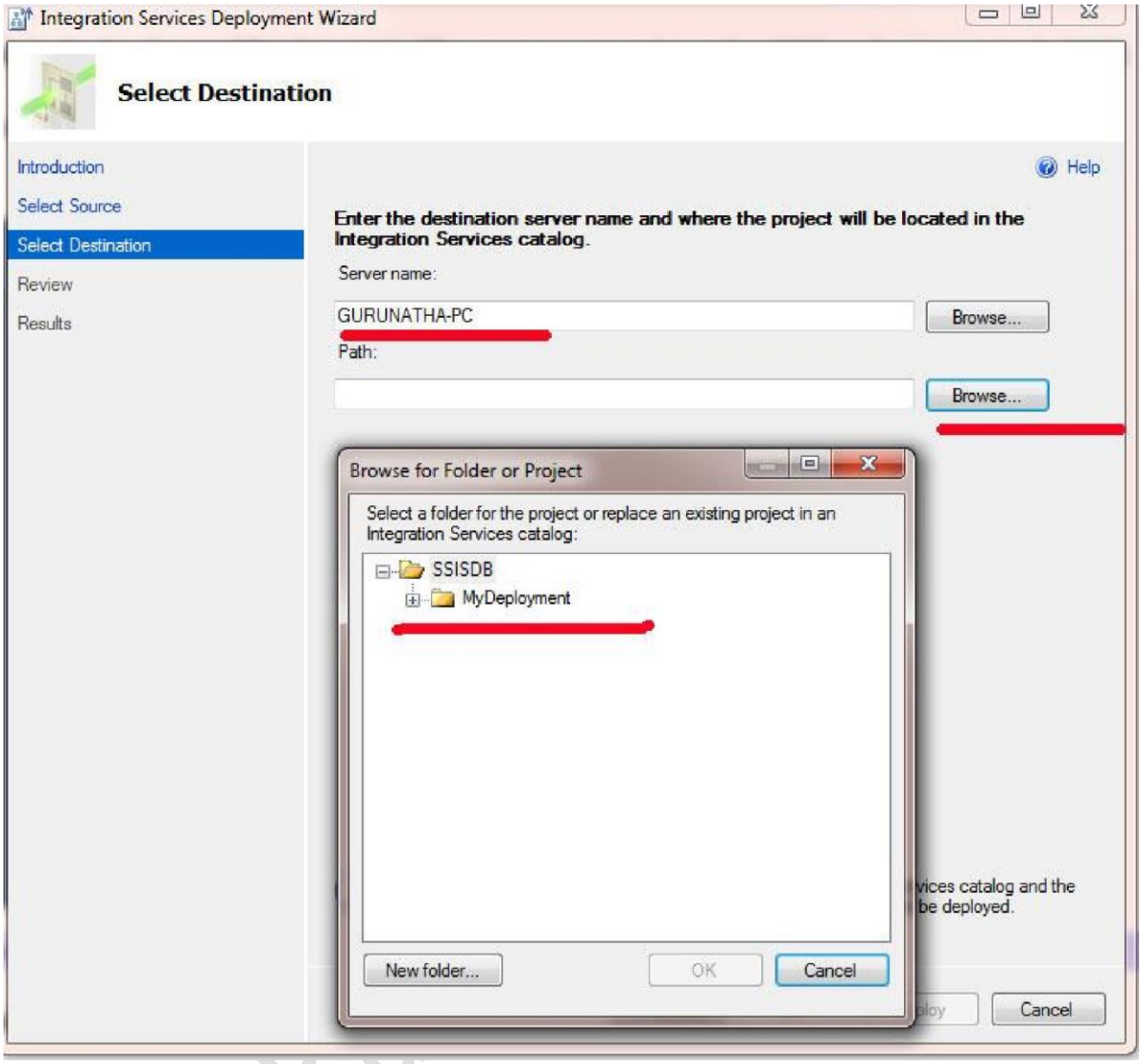




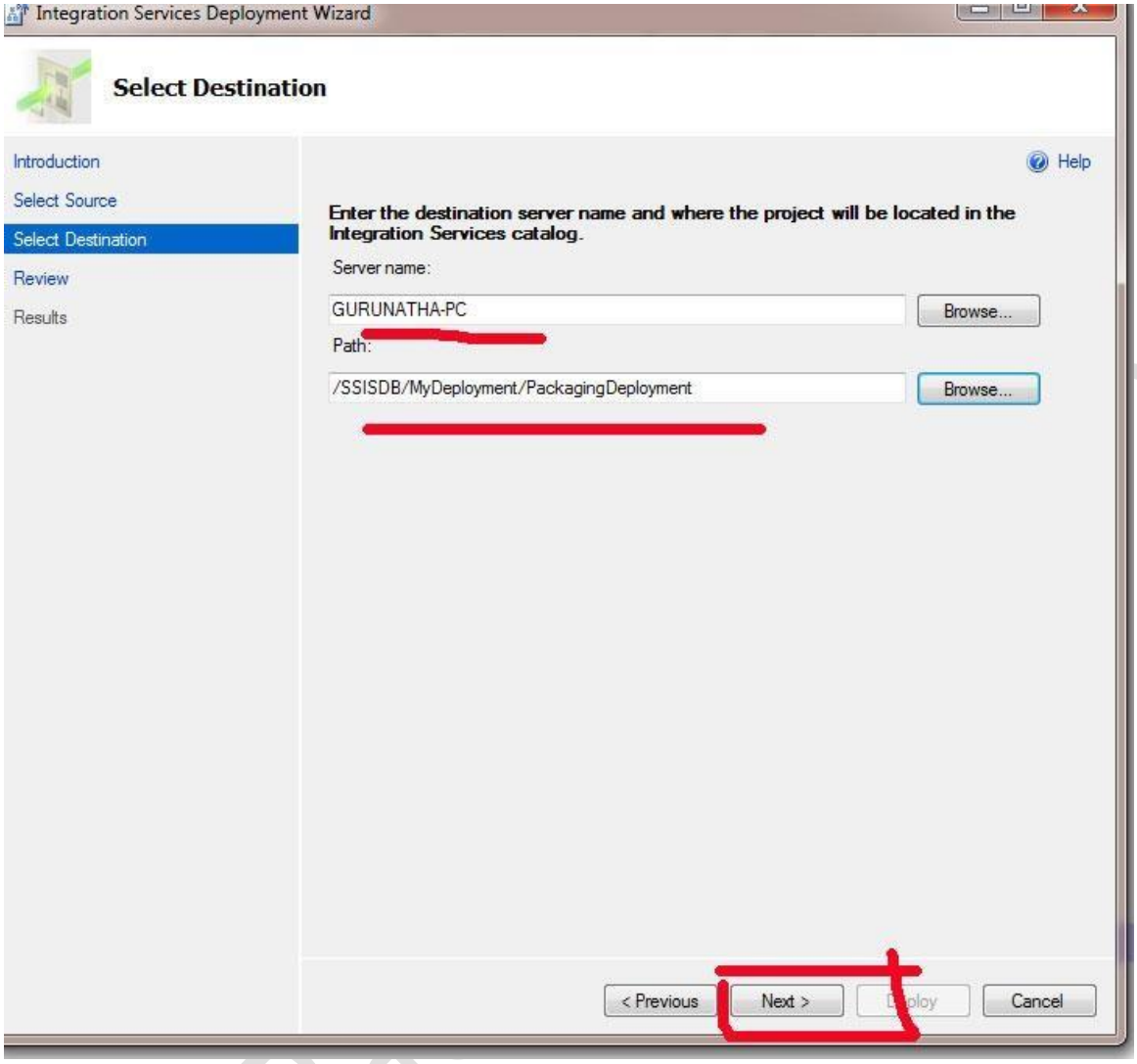
Since we are doing project deployment we need to choose that and click on next.



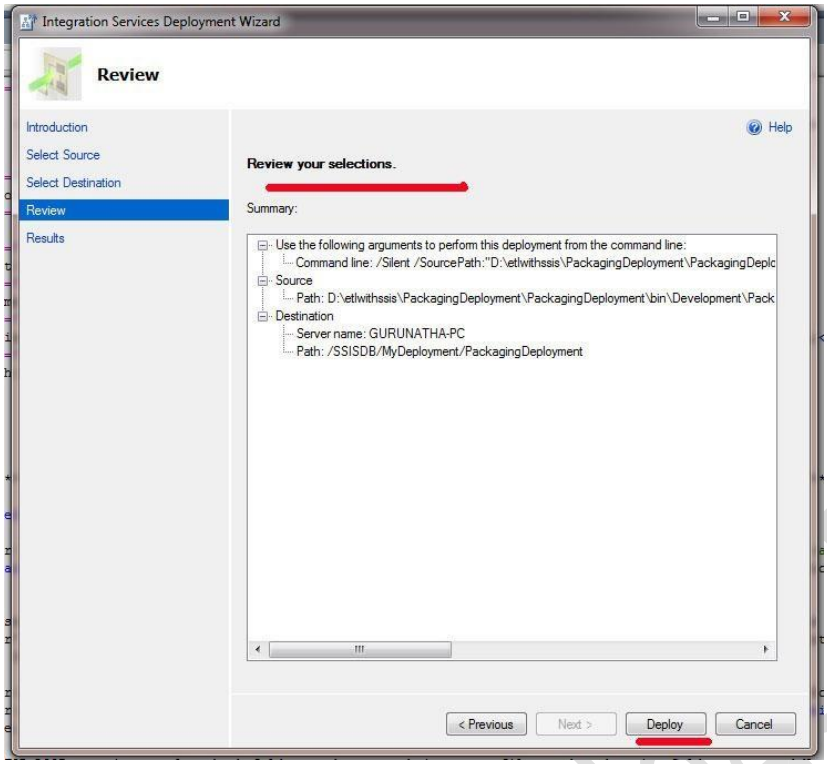
Here we need to select our Server Name because we are deploying this project to SQL Server. So choose server



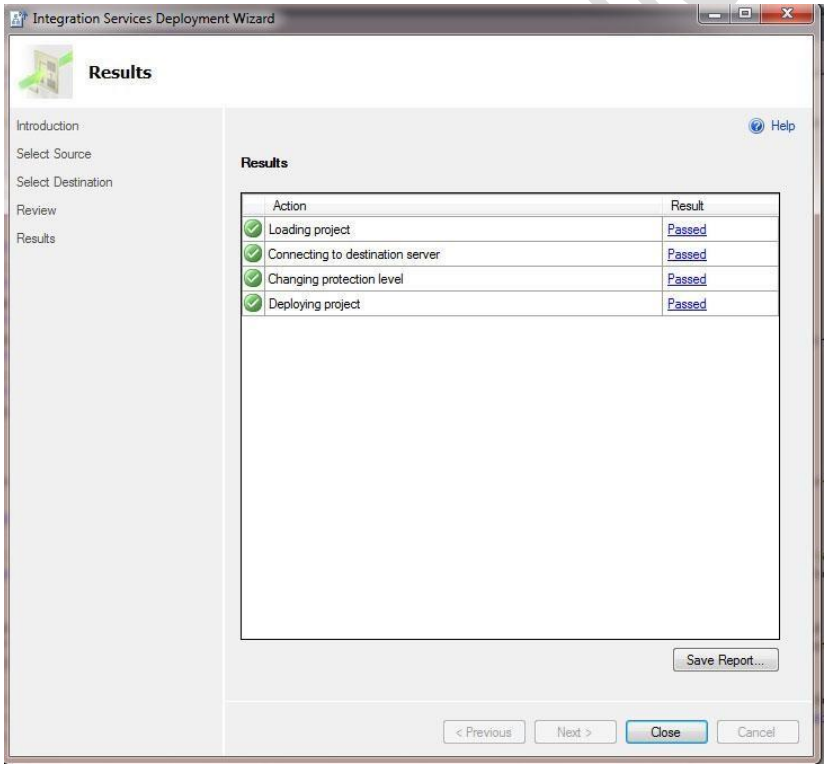
Once your server is selected next thing we need to select path the path is nothing but the folder which we have created under SSISDB catalog.



Select the SSISDB catalog folder path and click on next step to review deployment.



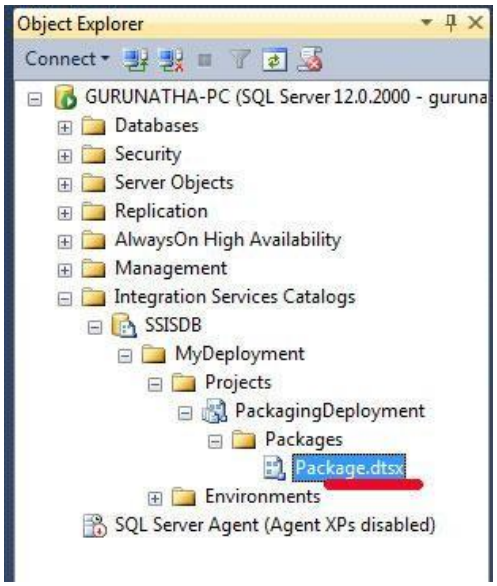
Finally click on Deploy button to deploy the package.



As you see our project is deployed successfully.

Step 5

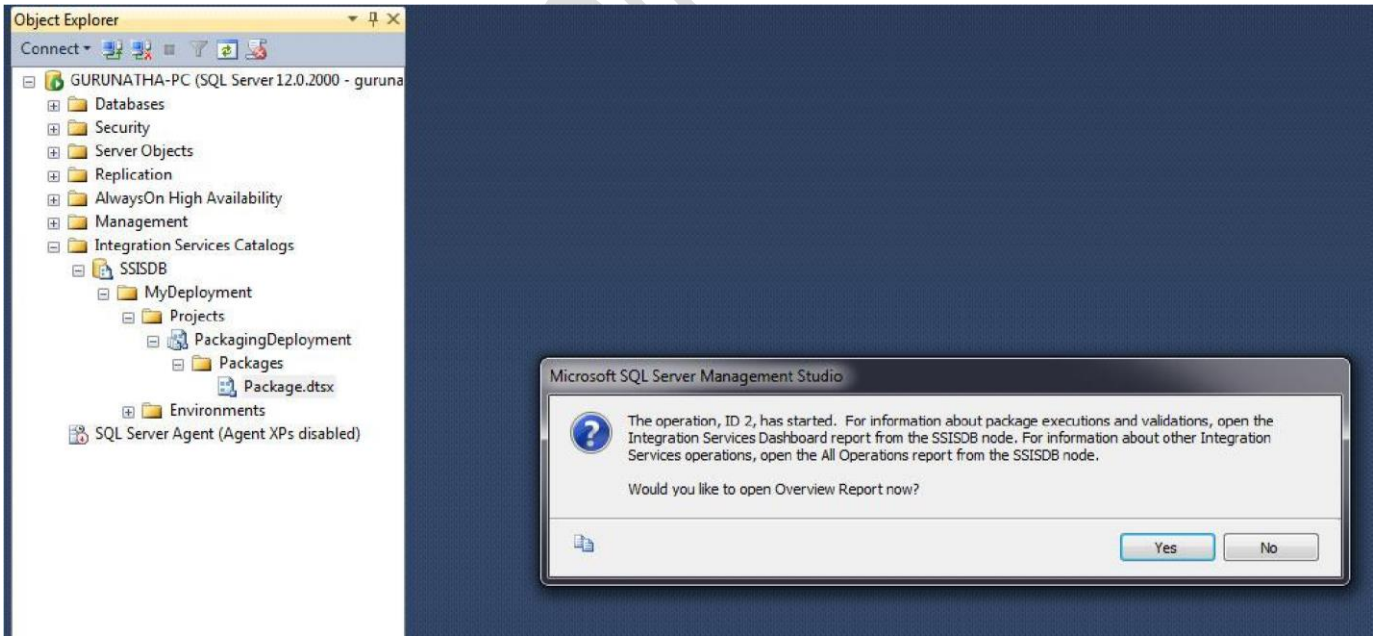
Go to SQL Server Management Studio -> SSISDB Catalog -> MyDeployment. To check deployment is processed successfully or not.

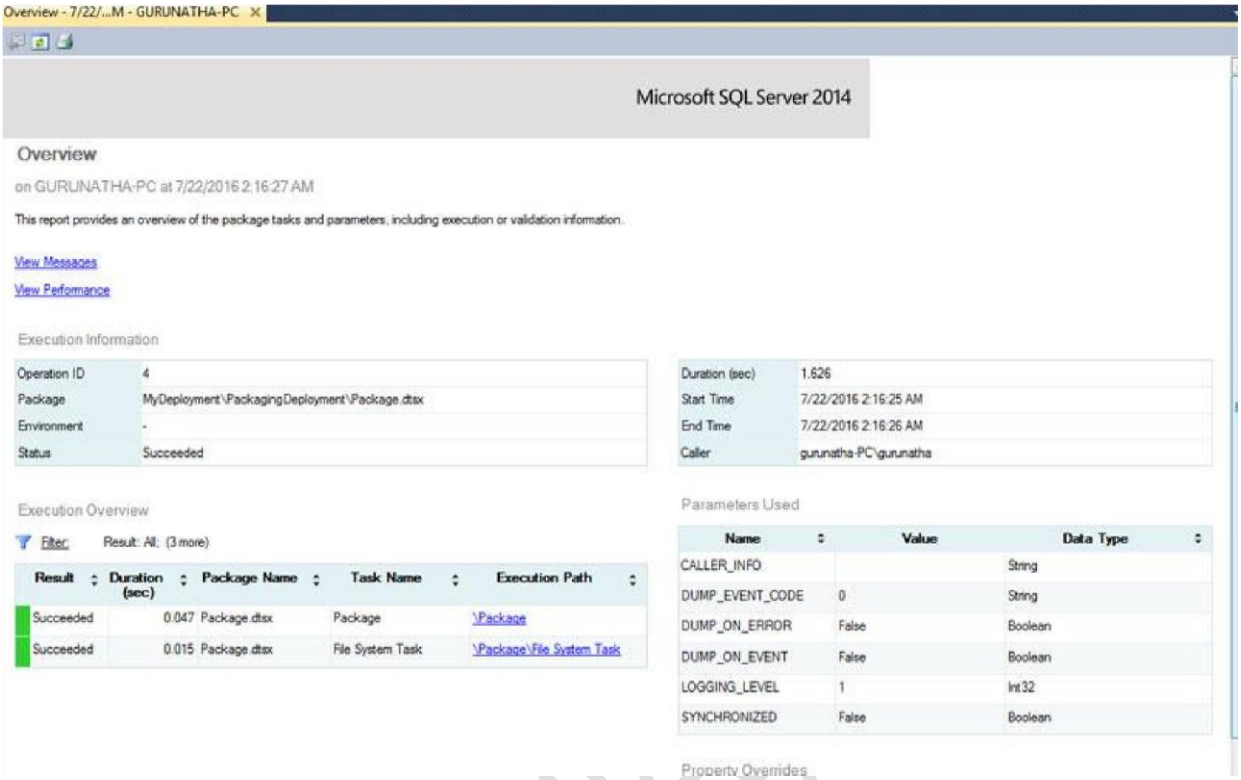


As you see we have deployed our project successfully.

Step 6

Now just right click on .dtsx and Execute the project .





Planning SSIS Package Execution

Several methods and techniques are available for the execution of SSIS solutions. They can be divided into two groups:

■■ On-demand execution allows SSIS processes to be initiated from various existing systems programmatically, in which they are invoked from user applications, as well as through user intervention, in which they are executed by human operators directly and interactively.

■■ Automated execution allows SSIS processes to be initiated automatically, without user interaction and even without the need for proprietary applications.

On-Demand SSIS Execution

After deployement, SSIS processes can be executed through user interaction (by using SSMS) or programmatically (by using the DTExec utility, Transact-SQL, Windows PowerShell, or the SSIS managed API).

Typically, on-demand execution is used when SSIS processes cannot be started automatically. On-demand execution is also used if an SSIS process cannot be started automatically simply because it is not used on a regular basis. An example is a process used in migrating production data into a development and testing data store. Though it might be beneficial to have up-to-date production data available in the testing environment, sometimes the testing data store contains special data samples that were created artificially and should not be overwritten. However, most SSIS processes, especially in data warehousing systems, are automated.

SQL Server Management Studio

SSMS supports the execution of packages stored on the SSIS server as well as packages that are accessible through the older SSIS service Packages stored in the SSISDB catalog can be started by using the Execute Package dialog box. This dialog box also provides access to parameters, connection managers, and package properties, so that these can be set before each execution.

The dialog box can also be used to script the execution; instead of executing the package immediately, you can configure, script, and execute the package later. The script can be reused and modified to use different parameter values, connection managers, or other properties.

Exercise 1 Validate SSIS Projects and Packages by Using SSMS

1. Start SSMS and connect to the instance of SQL Server 2012 that you set up as the SSIS server
2. In the Object Explorer, under Integration Services Catalogs, expand the SSISDB catalog node to display all deployed projects.
3. Right-click the TK 463 Chapter 10 project, and on the shortcut menu, select Validate to start project validation.
4. In the Validate Project dialog box, shown in Figure 12-1, you can configure the validation criteria before initiating validation.

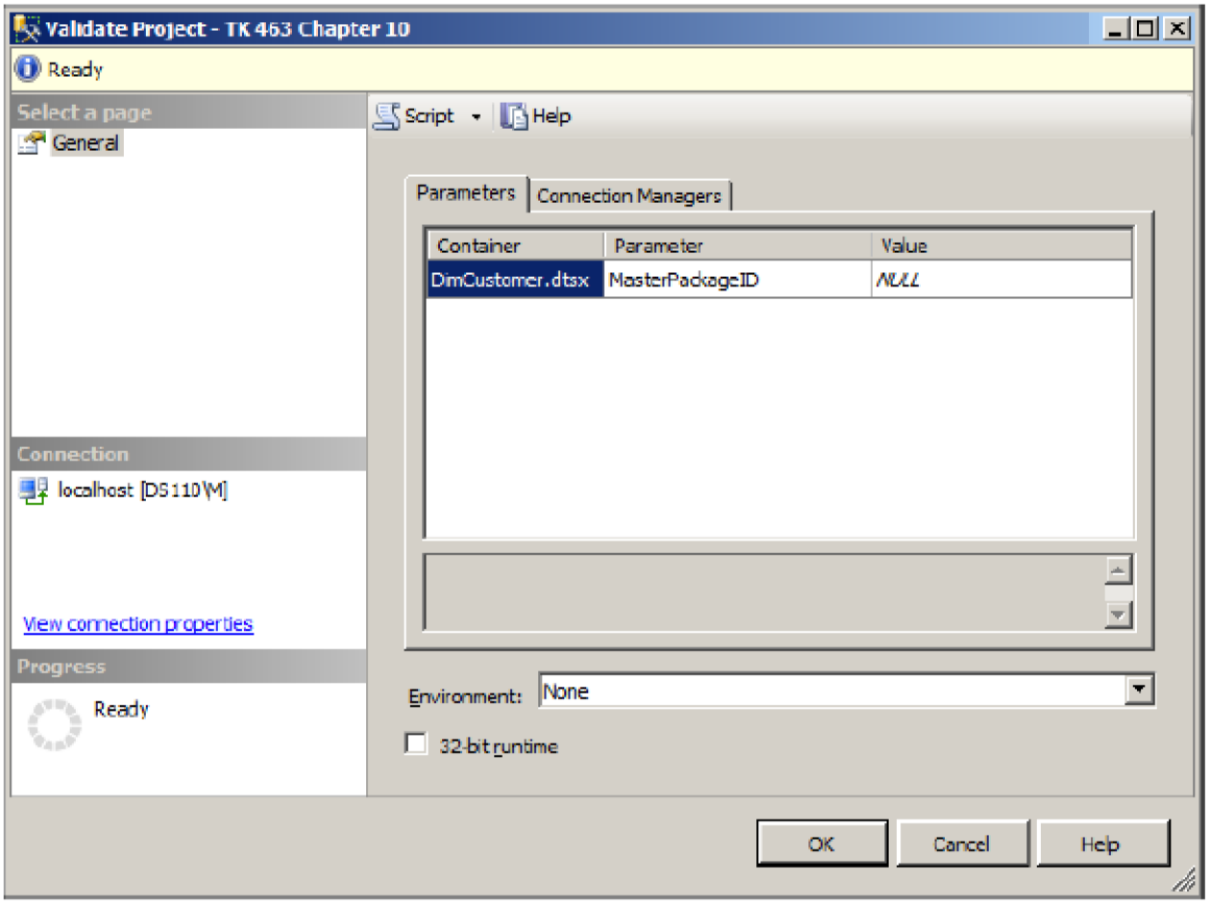


Figure 12-1 The Validate Project dialog box.

On the right side of the window, on the Parameters tab, you can set parameter values, and on the Connection Managers tab, you can configure the connection managers to be used in the validation. This way, you can

verify whether certain parameter or connection manager settings can be used in the execution of affected packages.

Below the tabs, in the Environment list box, you can set the criteria for environmentbased validation. You will learn more about this option in the practice in Lesson 2. The check box at the bottom of the Validate Project dialog box can be used to specify that the 32-bit runtime be used for the validation. Do not make any changes, because the default settings should be used for this validation.

5. When ready, click OK to start the validation.

After the validation is started, you will be prompted by SSMS, as shown in Figure 12-2, to indicate whether to open the built-in Validation Overview report to observe the validation process and the results.

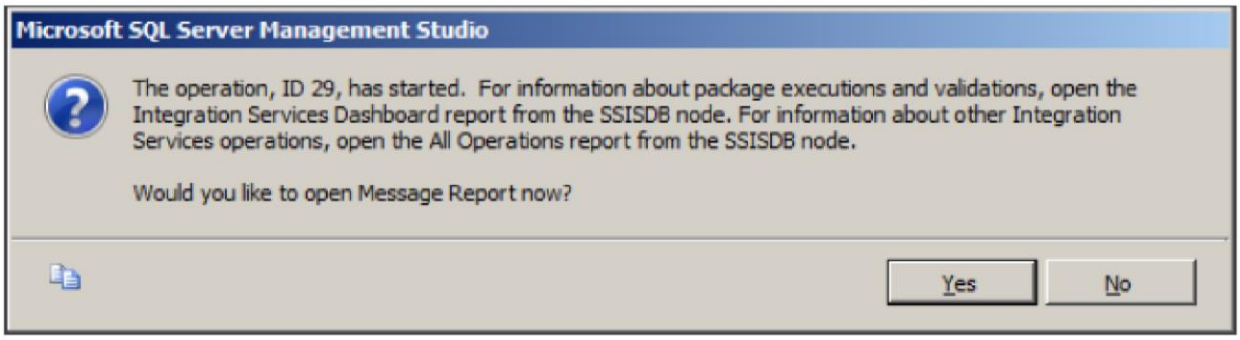


Figure 12-2 The Validation Report prompt.

6. Click Yes. A project validation report will be generated, listing the validation activities and their results. While the validation is running, the Status property in the header of the report will display Running, and after the validation has completed, it will display Succeeded if the validation was successful or Failed if it was not.

The validation should complete in just a few seconds. Refresh the report, if needed, to see the final validation result. Browse through the report to see what was checked and to see the validation results.

7. In the Object Explorer, expand the TK 463 Chapter 10 project node to display all the packages it contains.
8. Right-click the Master.dtsx package, and on the shortcut menu, select Validate to start package validation.

As you can see, the Validate Package dialog box has the same structure and functionalities as the Validate Project dialog box.

9. Click OK to validate the package, and in the prompt that appears after validation has started, click Yes to generate the package validation report.
10. Browse through the report and compare it to the one generated earlier.
11. When done, leave SSMS open, because you will need it in the following exercise.

Exercise 2 Execute and Monitor an SSIS Package by Using SSMS

1. In SSMS, in the Object Explorer, right-click the Master.dtsx package again, and this time select Execute from the shortcut menu to initiate the execution.
The Execute Package dialog box, shown in Figure 12-3, is similar to the Validate dialog boxes but has an Advanced tab.

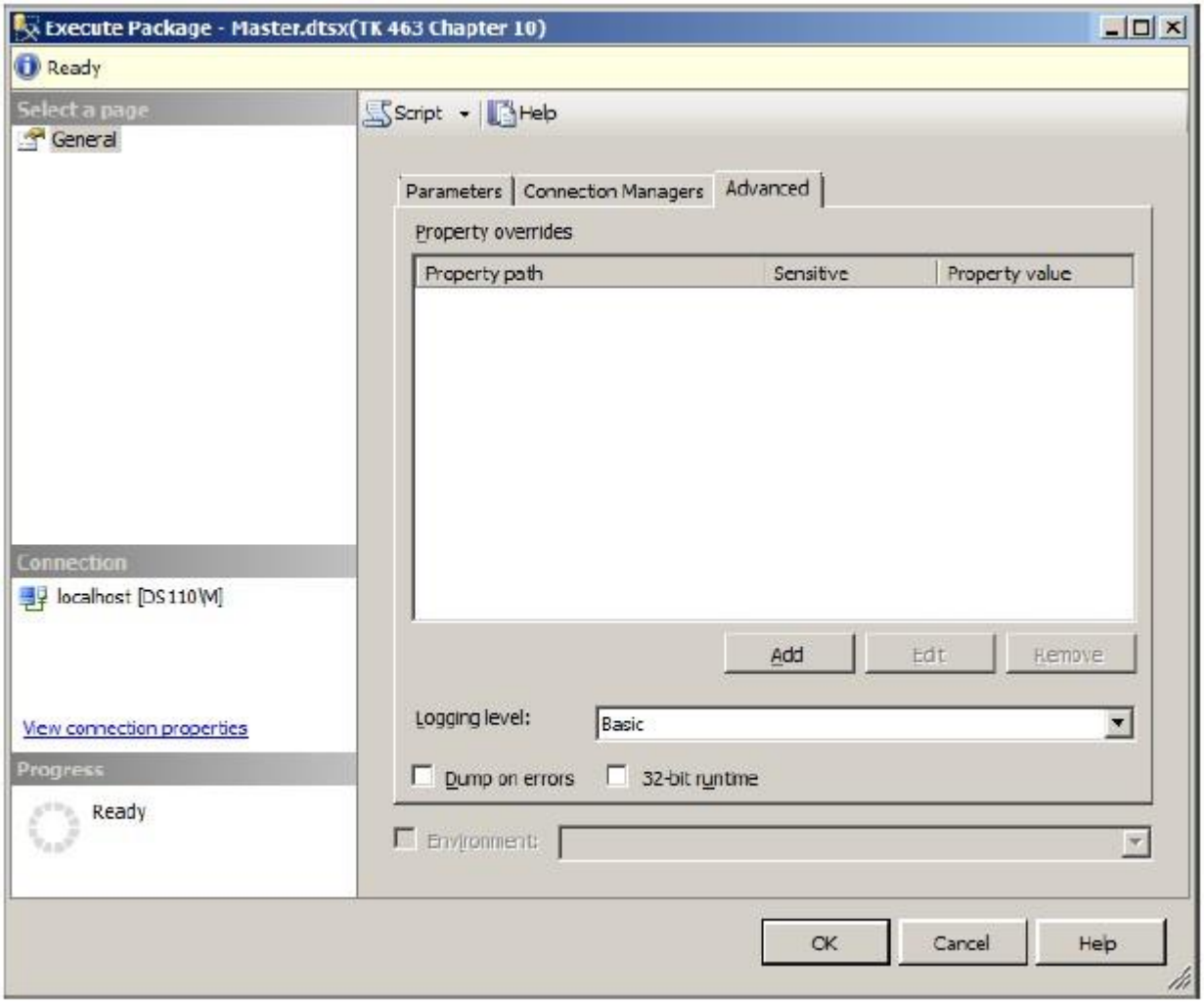


Figure 12-3 The Execute Package dialog box.

The Parameters and Connection Managers tabs can be used to set parameter values and configure the connection managers for the execution, as in the Validation dialog boxes.

The Advanced tab provides a few additional options: the Property Overrides grid can be used to set any additional package properties; you do so by providing the full path to a property and the value for it.

Below the grid, in the Logging Level list box, you can select the logging mode for the execution. For this execution, leave the Basic logging level selected.

The Dump On Errors check box, when selected, instructs the SSIS runtime to create a memory dump in case errors are encountered during execution.

2. When ready, click OK to start the execution.
3. When prompted whether to generate the execution report, click Yes. An Overview report will be generated.

While the package is being executed, the Status property in the header of the report will display Running, and after the execution has completed, it will display Succeeded if the validation was successful or Failed if it was not.

4. Three distinct reports are created for each execution:

■■ The Overview report, displaying the general information about the execution
■■ The Messages report, providing a list of messages that were generated during execution
■■ The Performance report, displaying performance information about the execution

Browse all three reports to see what kind of information about the execution each of them provides.

5. Repeat steps 1 through 3 to execute the same package once more, but this time, on the Advanced tab of the Execute Package dialog box, select the Verbose logging mode.
6. Allow SSMS time to generate the report. After the execution has completed (which you will know if you refresh the report after a couple of minutes), compare the newly generated execution report with the one created earlier. Again, compare the information available in all three report types.

DTExecUI

The DTExecUI utility can be used to execute SSIS packages that are accessible through the older SSIS service.

This utility is only provided for compatibility with older versions—to execute SSIS packages created in previous versions of SQL Server. The DTExecUI executable is located at %ProgramFiles(x86)%\Microsoft SQL Server\110\Tools\Binn\ManagementStudio; only a 32-bit edition of this tool exists. To execute an SSIS package by using the 64-bit SSIS runtime, you must use another tool.

Exercise 3 Execute an SSIS Package by Using DTExec

1. Using Windows Explorer, navigate to the C:\TK463\Chapter12\Code folder and locate the DTExec_Example.cmd command file.
2. Right-click the file and select Edit on the shortcut menu to open the file in Notepad or another appropriate text editing tool.

Inspect the DTExec command line and the arguments.

3. When you are done reviewing the command line, close the editing tool and return to Windows Explorer.
4. Double-click the DTExec_Example.cmd command file to execute it, and observe the execution progress in the Overview report that you used in Exercise 2.

Transact-SQL, Windows PowerShell, the SSIS Managed AP I, and DTExec

In addition to being executed by user interaction, SSIS packages can be executed programmatically, by using Transact-SQL, Windows PowerShell, the SSIS managed application programming interface (API), or the DTExec command-line utility.

Programmatic execution using Transact-SQL, Windows PowerShell, or the SSIS managed API is performed in two or more steps:

1. First, a new execution operation is created. The creation references a particular package and can include a reference to an environment that provides package configuration information.
2. Next, execution properties can be set. For instance, this might be done if no configuration was provided at creation, if certain parameters are not set by the configuration, or if certain configuration properties need to be overridden.
3. Finally, the execution is started. The invocation can be asynchronous (which it is by default), meaning that the package is started and then the control is returned to the caller immediately; or it can be invoked synchronously, meaning that the invocation procedure will keep running until the execution has completed.

Listing 12-1 shows an example of SSIS package execution using Transact-SQL. The execution is performed by using SSISDB system stored procedures. **LISTING 12-1** Executing a Package by Using Transact-SQL

```
DECLARE @execution_id BIGINT;
DECLARE @use32bitruntime BIT = CAST(0 AS BIT);
DECLARE @logging_level INT = 1;
EXEC catalog.create_execution
@folder_name = N'TK 463 Chapter 11',
@project_name = N'TK 463 Chapter 10',
@package_name = N'Master.dtsx',
@use32bitruntime = @use32bitruntime,
@reference_id = NULL,
@execution_id = @execution_id OUTPUT;
EXEC catalog.set_execution_parameter_value
@execution_id,
@object_type = 50,
@parameter_name = N'LOGGING_LEVEL',
@parameter_value = @logging_level;
EXEC catalog.start_execution
@execution_id; GO
```

In Listing 12-1, you can observe all three execution steps:

1. A package named Master.dtsx, which is part of the TK 463 Chapter 10 project and is located in the TK 463 Chapter 11 folder of the SSISDB catalog, is selected for execution, and the execution is created by using the 64-bit runtime without an environment reference.
2. The LOGGING_LEVEL execution property is set.
3. The execution is started asynchronously.

Windows PowerShell can also be used to access objects on the SSIS server and can therefore also be used to validate and execute SSIS projects and/or packages.
Before SSIS packages can be executed by using Windows PowerShell, the SSIS management assembly must be loaded. Then the operation is prepared and executed, as described earlier in this section. An example is shown in Listing 12-2.

Listing 12-2 Executing a Package by Using Windows PowerShell

```
# Assign SSIS namespace to variable
```

```
$ssisNamespace = "Microsoft.SqlServer.Management.IntegrationServices"
# Load the SSIS Management Assembly
$assemblyLoad = [Reflection.Assembly]::Load($ssisNamespace + ", Version=11.0.0.0,
Culture=neutral, PublicKeyToken=89845dcd8080cc91")
# Create a connection to a SQL Server instance
$connectionString = "Data Source=localhost;Initial Catalog=master;Integrated
Security=SSPI; "
$connection = New-Object System.Data.SqlClient.SqlConnection $connectionString
# Instantiate the SSIS object
$ssis = New-Object $ssisNamespace".IntegrationServices" $connection
# Instantiate the SSIS package
$catalog = $ssis.Catalogs["SSISDB"]
$folder = $catalog.Folders["TK 463 Chapter 11"]
$project = $folder.Projects["TK 463 Chapter 10"]
$package = $project.Packages["Master.dtsx"]
# Set package parameter(s)
$package.Parameters["somePackageParameter"].Set
( [Microsoft.SqlServer.Management.IntegrationServices.ParameterInfo+ParameterValueType]: :Literal,
"parameter value")
$package.Alter()
# Execute SSIS package ($environment is not assigned)
$executionId = $package.Execute("false", $environment)
```

Currently, no SSIS Windows PowerShell cmdlets are available; however, the entire SSIS object model is accessible via the SSIS management assembly (―Microsoft.SqlServer.Management. IntegrationServices, Version=11.0.0.0, Culture=neutral, PublicKeyToken=89845dcd8080cc91‖).

SSIS objects can be accessed through the SSIS managed API. They are exposed through the Microsoft.SqlServer.Management.IntegrationServices namespace in the Microsoft .NET Framework. In Listing 12-3, you can see an example of SSIS package execution via the SSIS managed API, written in Microsoft Visual C#.

Listing 12-3 Executing a Package by Using SQL Server Management Objects using

```
System;
using Microsoft.SqlServer.Management.IntegrationServices; using
SMO = Microsoft.SqlServer.Management.Smo;
…
// Create a connection to a SQL Server instance
SMO.Server ssisServer = new SMO.Server("localhost");
Console.WriteLine("Connected to:\t\t" + ssisServer.Name); //
Instantiate the SSIS object
IntegrationServices ssis = new IntegrationServices(ssisServer);
// Instantiate the SSIS package
Catalog catalog = ssis.Catalogs["SSISDB"];
CatalogFolder folder = catalog.Folders["TK 463 Chapter 11"];
ProjectInfo project = folder.Projects["TK 463 Chapter 10"];
```

```
PackageInfo package = project.Packages["Master.dtsx"];
Console.WriteLine("Selected package:\t" + System.IO.Path.Combine(catalog.Name, folder.
Name, project.Name, package.Name));
// Set package parameter(s)
package.Parameters["someParameter"].Set
(
ParameterInfo.ParameterValueType.Literal, "parameter value"
);
package.Alter();
catalog.ServerLoggingLevel = Catalog.LoggingLevelType.None;
// Execute SSIS package package.Execute(false,
null);
```

To access a SQL Server instance for administrative purposes, consider using Server Management Objects (SMO), as shown in Listing 12-3 (Microsoft.SqlServer.Management.Smo).

The DTExec command-line utility can be used to execute packages stored in the SSISDB catalog, as well as those accessible through the older SSIS service (that is, those that are stored in the msdb system database or in the managed file system). DTExecUI cannot be used to execute packages stored in the SSISDB catalog.

Listing 12-4 shows an example of the use of the DTExec command-line utility to invoke the execution of an SSIS package stored in the SSISDB catalog.

Listing 12-4 Executing a Package by Using DTExec

```
DTExec /Server localhost /ISServer "\SSISDB\TK 463 Chapter 11\TK 463 Chapter 10\Master. dtsx"
/Par $ServerOption::LOGGING_LEVEL(Int32);1
```

Exercise 4 Execute an SSIS Package by Using Windows PowerShell

1. Using Windows Explorer, navigate to the C:\TK463\Chapter12\Code folder and locate the PowerShell_Example.ps1 Windows PowerShell script file.
2. Right-click the file and select Edit on the shortcut menu to open the file either in Notepad or in the Windows PowerShell Integrated Scripting Environment (ISE). Inspect the script, following the comments inside it.
3. When you are done reviewing the script, close the editing tool and return to Windows Explorer.
4. Right-click the PowerShell_Example.ps2 Windows PowerShell script again, and select Run with PowerShell on the shortcut menu to execute it.

Observe the execution progress in the Overview report that you used in the preceding exercises.

Automated SSIS Execution

In the majority of cases, SSIS process execution is automated. This is especially true for data warehousing solutions: after a data warehousing process has been developed and deployed, it will be scheduled for automatic execution during maintenance windows.

Typically, data warehouse maintenance operations are performed at times when systems are not being used by the end users (for instance, at night).

Alternatively, data warehouse maintenance could also be performed on standby servers, and after it has completed, the live servers can be taken offline and the standby servers brought online to replace them.

In both cases, the entire process can and should be automated: in the former case, because most likely no administrator would be available during the night to execute the process manually, and in the latter case, because no human operator could ever perform all the steps of such a complex process as quickly and as reliably as a machine can.

SQL Server Agent

SQL Server Agent is a component of the SQL Server platform; its principal purpose is to facilitate automatic execution of processes in the SQL Server environment on a predefined schedule or in response to predefined environment conditions.

In SQL Server 2012, SQL Server Agent supports the execution of SSIS packages stored on an SSIS server, as well as packages accessible through the older SSIS service.

Jobs and Job Steps

The principal unit of work in SQL Server Agent processes is a *job*. A SQL Server Agent job does not actually represent any specific operations; it serves as a container for *job steps*, which represent the actual operations.

A SQL Server job step can invoke executable programs and operating system commands, Transact-SQL statements, Windows PowerShell scripts, ActiveX scripts, SQL Server Replication tasks, SQL Server Analysis Services tasks, and SSIS packages.

A job must contain at least one step in order to perform any operations; multiple steps are executed in the order specified when the job is created. Each step can be configured to log information about its execution to the job step log in the msdb system database (some operation types can even log their execution to a file). Steps can also be configured to respond to failures (such as execution errors) by using notifications, alerts, or operators.

The principal execution mode in SQL Server Agent is to execute a job as a whole; however, execution of individual steps is also supported. When a job is executed as a whole, all steps are executed in the specified order, starting with step number one. However, if a specific step is chosen to start the execution, after the selected step has completed, the execution will advance to any following job step, and the rest of the steps will be performed in the specified order.

Schedules

SQL Server Agent jobs can be scheduled to run whenever SQL Server Agent is started, when the server CPU is idle (that is, when CPU utilization falls below a certain level), or on a specified date and time.

To execute a job when the CPU is idle, the CPU utilization condition must first be set for the server. CPU idle time is measured as a percentage of utilization below a specified level for a specified time. Whenever this occurs, all jobs scheduled to run at CPU idle time will bestarted automatically.

Jobs can be scheduled to run at a specified date and time—once or repeatedly. When jobs are executed repeatedly, the following schedule types can be used:

■■ **Overall recurrence frequency** For jobs executed once per a specified period (once per day, week, month, or year)

■■ **Daily frequency** For jobs executed continuously during the day

■■ **Specific duration** For jobs executed after a specified period of time has passed Every schedule has a start time, denoting when the schedule was created or modified. The end time can also be set, indicating that the schedule will cease to be used after the end time has passed. If the end time is not set, the schedule is used indefinitely.

Exercise 5 Create, Schedule, and Execute an SSIS Package by Using SQL Server Agent

1. In SSMS, in the Object Explorer, under the SQL Server Agent node, expand Jobs. At least one SQL Server job should be listed: SSIS Server Maintenance Job. This job was created when you created the SSISDB catalog in Lesson 2 of Chapter 11.
2. Right-click Jobs, and on the shortcut menu, select New Job to start the creation of a new SQL Server Agent job.
3. Use the New Job editor to configure a new job. On the General page, provide a name for the job (**TK 463 Chapter 10 Master Package**) and a description if you want, as shown in Figure 12-4.

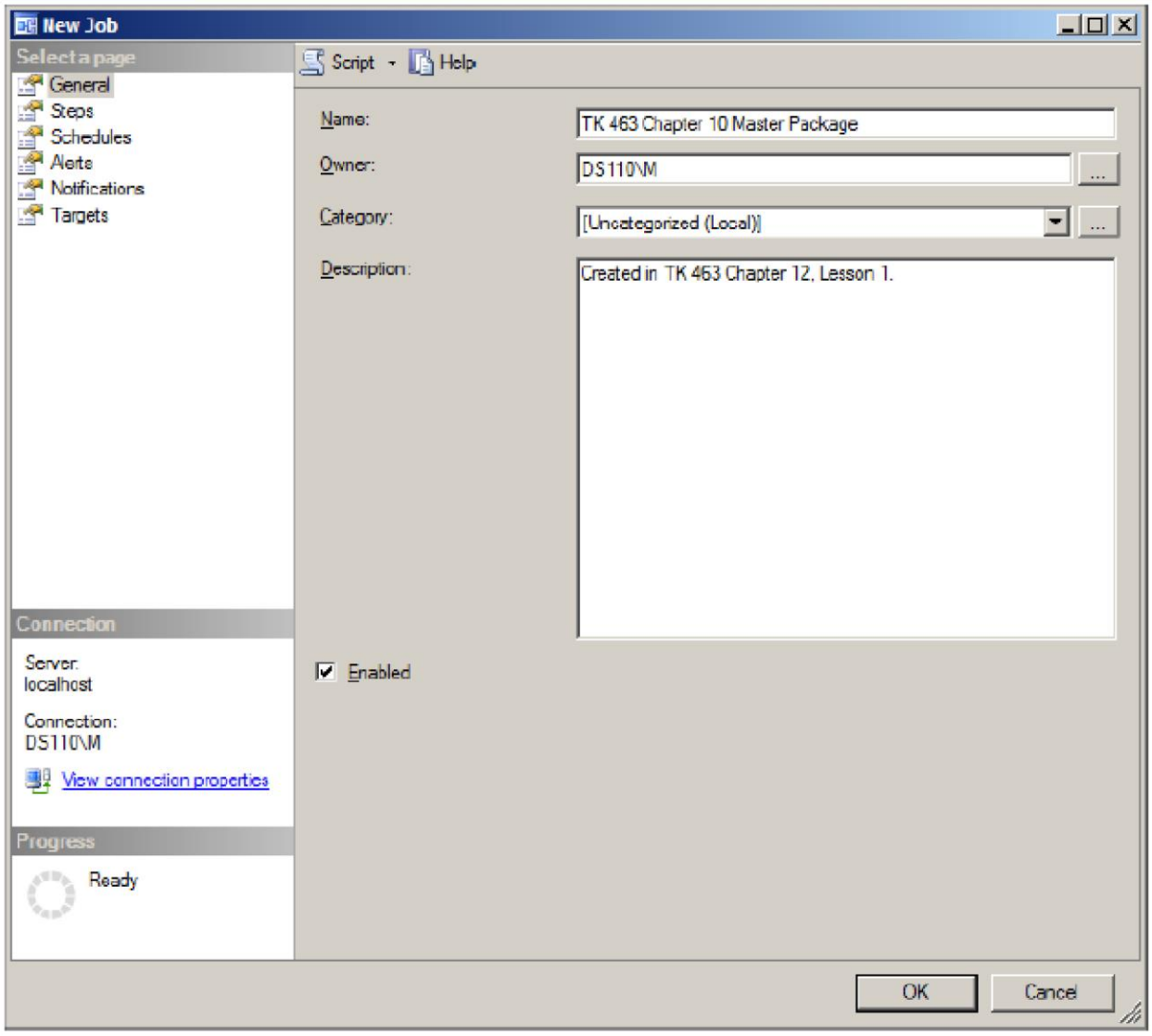


FIGURE 12-4 The General page of the New Job dialog box.

4. On the Steps page, click New to open the New Job Step editor.
5. Create a job step and configure it to execute an SSIS package by using the information provided in Table 12-4.

TABLE 12-4 New Step Configuration

Page	Property	Value
General	Step name	Master Package
General	Type	SQL Server Integration Services Package
General	Run as	SQL Server Agent Service Account
General (Package tab)	Package source	SSIS Catalog
General (Package tab)	Server	localhost

The dialog box should look similar to the one shown in Figure 12-5.

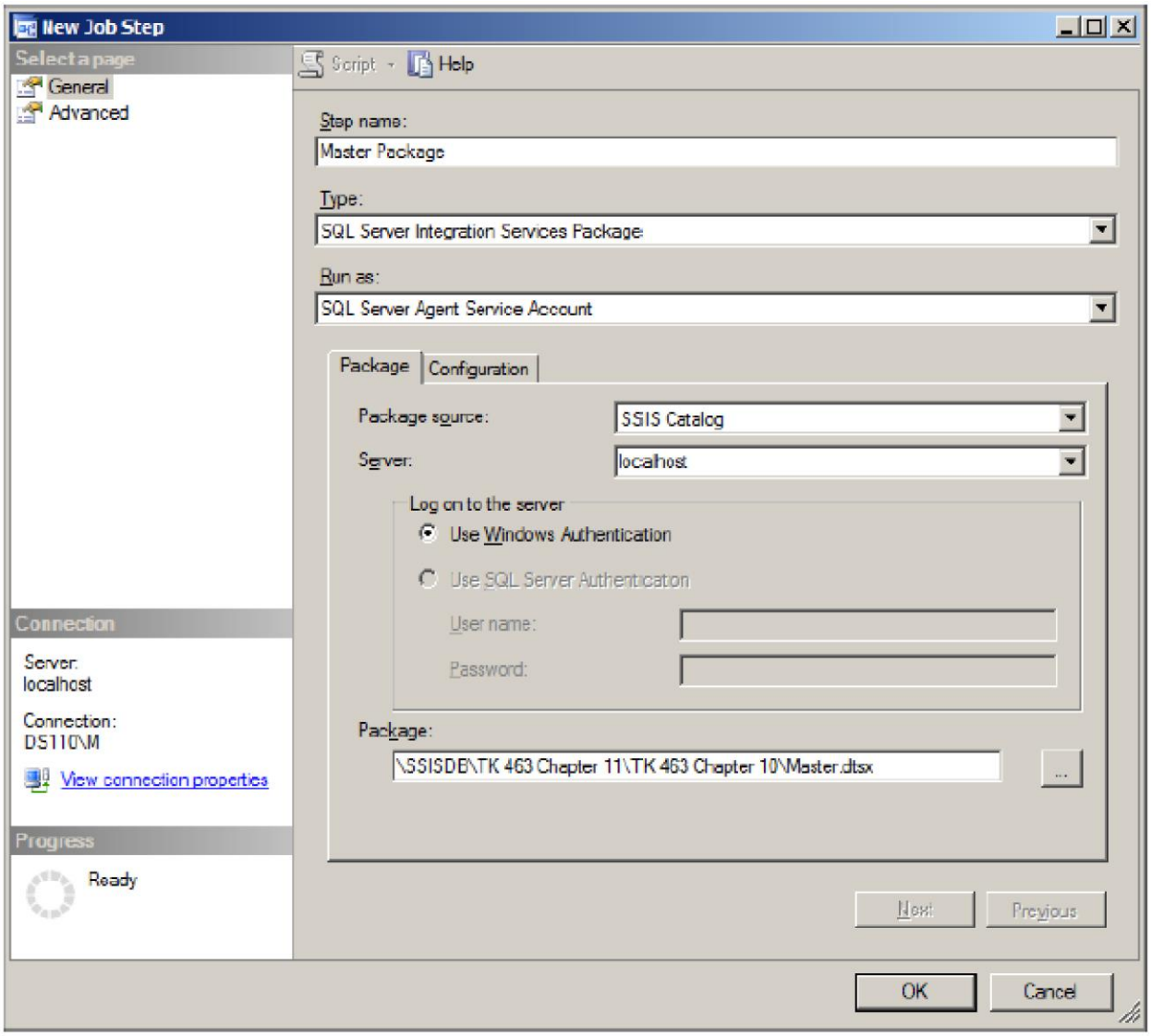


FIGURE 12-5 The New Job Step dialog box.

6. To select the package, on the Package tab, click the ellipsis (…) on the right side of the Package text box and use the Select An SSIS Package dialog box to navigate in the SSISDB catalog to the TK 463 Chapter 11 folder. In the TK 463 Chapter 10 project, select the Master.dtsx package. Click OK to confirm the selection. On the Configuration tab, you can adjust the configuration settings for the package: you can assign parameter values, configure connection managers, and adjust advanced settings, such as assigning values to package properties, enabling memory dumps in case errors are encountered, or setting the package to be executed using the 32-bit runtime. In this exercise, you will not need any additional configuration settings.

7. On the Advanced page of the New Job Step dialog box, you can set additional settings for this job step, such as how to respond to success or failure, and you can adjust SQL Server Agent logging settings. In this exercise, you will not need to adjust any advanced job step configuration settings, so just review them and leave them unchanged.

8. When you are ready, click OK to confirm the creation of a new job step. The job should now contain a single step named Master Package, configured to execute the Master.dtsx package of the TK 463 Chapter 10 project.
9. In the New Job editor, on the Schedules page, click New to open the New Job Schedule editor.
10. Configure the schedule by using the information provided in Table 12-5.

TABLE 12-5 New Job Schedule Configuration

Property	Value
Name	Continuous
Schedule type	Recurring
Enabled	(Selected)
Occurs	Daily
Recurs every	1 day(s)
Occurs every	3 minute(s)

The completed dialog box should look similar to the one shown in Figure 12-6.

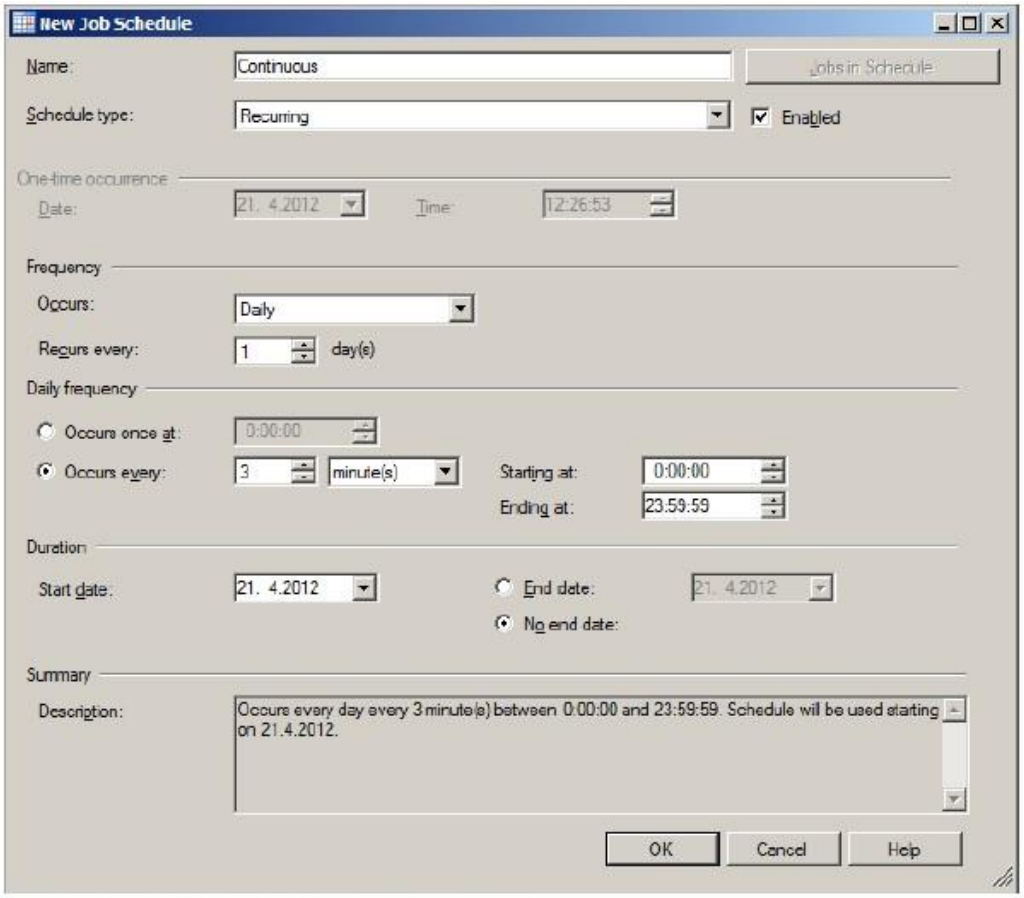


FIGURE 12-6 The New Job Schedule dialog box.

11. When you are done, click OK to complete the creation of a new schedule.
12. In the New Job editor, click OK to complete the creation of the new job.
13. In SSMS, in the Object Browser, double-click the Job Activity Monitor node, or rightclick it and select View Job Activity from the shortcut menu, to open the Job Activity Monitor. The job you just created should be executed automatically every three minutes; refresh the Job Activity Monitor to capture its execution.
14. Right-click the TK 463 Chapter 10 Master Package job in the Job Activity Monitor, and select View History from the shortcut menu to open the Log File Viewer. Observe the SQL

Server Agent log entries for the job you just created. When you're done, close the Log File Viewer and return to the Job Activity Monitor.

15. Right-click the TK 463 Chapter 10 Master Package, and on the shortcut menu, select Disable Job. This will prevent the job from running automatically. When you are done, close the Job Activity Monitor.
16. Close SSMS.

Monitoring SSIS Execution

SSIS execution monitoring provides information about SSIS processes as they are being executed, allowing you to track their progress and status. More importantly, information about each

execution is written to the database and is available for analysis even after the processing has completed.

Typically, SSIS execution monitoring serves two purposes:
■■ It provides an overview of how SSIS solutions are used, how they use available resources, and whether they are completed successfully or not. At this level, the information about SSIS executions is useful for resource planning and problem detection.
■■ It supports SSIS troubleshooting activities when problems with SSIS execution have been detected. Troubleshooting usually requires more detailed information about SSIS execution than that used for a more general overview. Additional information that might not have been captured by SSIS logging or auditing might also be required, especially when you are troubleshooting performance issues or problems that originate outside the SSIS or the SQL Server platform.

Operations
Operations represent actions performed against SSISDB objects and the SSISDB catalog itself.

Validations
Validations are used to check SSISDB projects or packages for common problems, especially in project or package configurations. Executions

The execution of an SSISDB package begins with the creation of a new *execution instance*; the principal purpose of the execution instance is to provide a reference between a package being prepared for execution and its corresponding configuration. Another purpose is to document (log) the execution itself.

Packages stored in the SSISDB catalog are *abstract entities*. When an execution instance is created and a configuration is assigned to it, a *concrete instance* of the package is created, containing all of the information that is required for its execution. The abstract entity of a package *can* be executed in as many different ways as there are possible configurations, but a concrete instance of the package *is* executed by using a specific configuration.

When the execution instance is created, an environment containing configuration information expected by the package can be assigned to it. Only a single environment reference can be assigned to a particular execution instance.

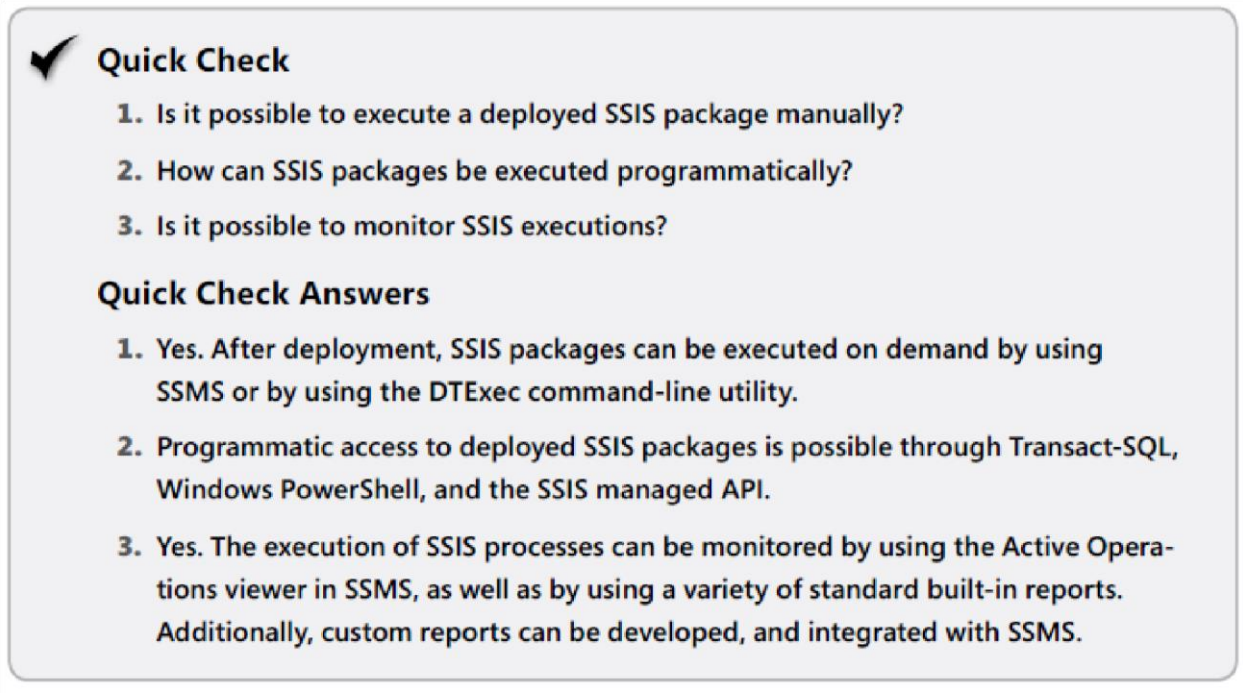
After the execution instance has been created, additional configurations can be assigned to it. These direct assignments can target the properties of the instance, as well as the parameters and/or properties of the package being executed. For instance, direct assignments can be used to supply values to properties and parameters that are not supplied by the environment, or to override the values supplied by the environment.

After the execution instance has been configured, it can be started. The action of starting an execution instance begins with the validation of the package; if the validation is successful, the processing begins.

SSIS Monitoring in SQL Server Management Studio SSMS provides essential SSIS monitoring functionalities, most of which are implemented by using reports and some by using built-in viewers.

Operations can be monitored by using the Active Operations dialog box, which is accessible via the Object Browser in SSMS. When an operation, such as a validation or an execution, is started by using SSMS, you are prompted to open the corresponding report, which provides an overview of the active operation.

All available reports can be accessed through the Object Browser, via the shortcut menu of each node (catalog, project, package, or environment)



Quick Check

1. Is it possible to execute a deployed SSIS package manually?
2. How can SSIS packages be executed programmatically?
3. Is it possible to monitor SSIS executions?

Quick Check Answers

1. Yes. After deployment, SSIS packages can be executed on demand by using SSMS or by using the DTExec command-line utility.
2. Programmatic access to deployed SSIS packages is possible through Transact-SQL, Windows PowerShell, and the SSIS managed API.
3. Yes. The execution of SSIS processes can be monitored by using the Active Operations viewer in SSMS, as well as by using a variety of standard built-in reports. Additionally, custom reports can be developed, and integrated with SSMS.

Introduction to Business Intelligence

―The processes, technologies and tools needed to turn data into information and information into knowledge and knowledge into plans that drive profitable business action. BI encompasses data warehousing, business analytics and knowledge management.‖

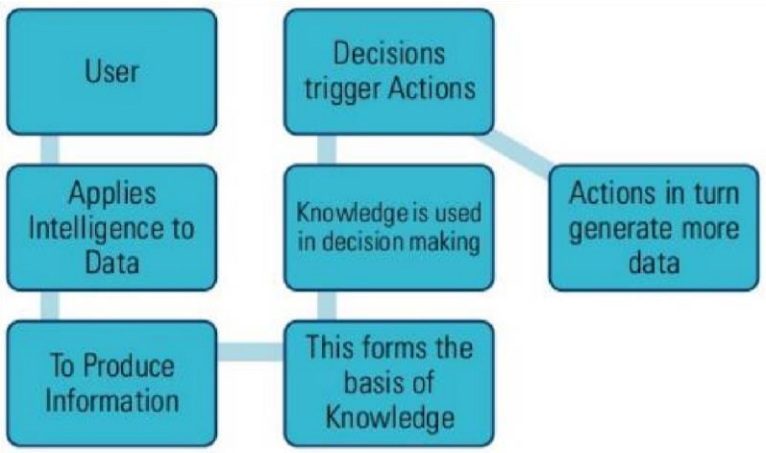
―Business intelligence (BI) is the use of computing technologies for the identification, discovery and analysis of business data – like sales revenue, products, costs and incomes. BI technologies provide current, historical and predictive views of internally structured data for products and departments by establishing more effective decision-making and strategic operational insight.‖

The Decision making process

Information and knowledge form the backbone of the decision making process.

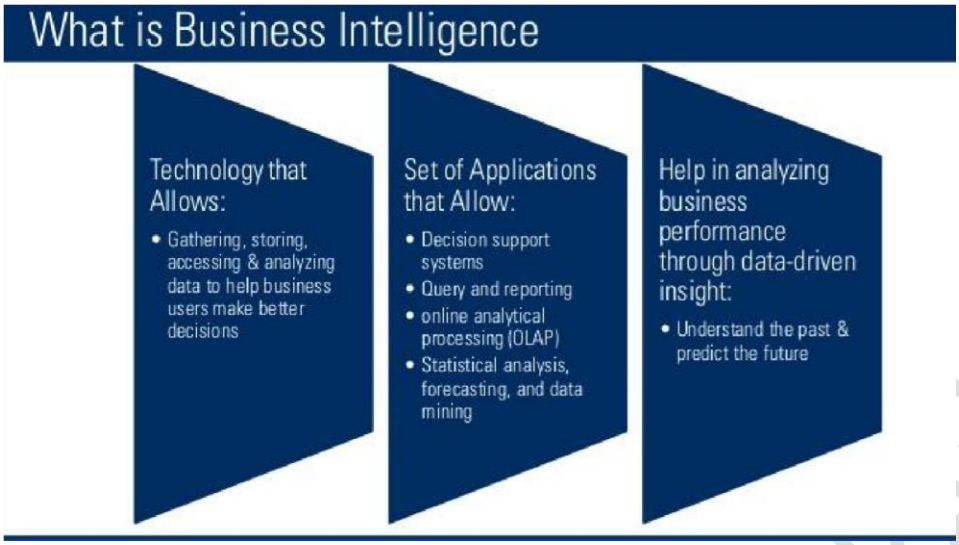
Decision Action Cycle

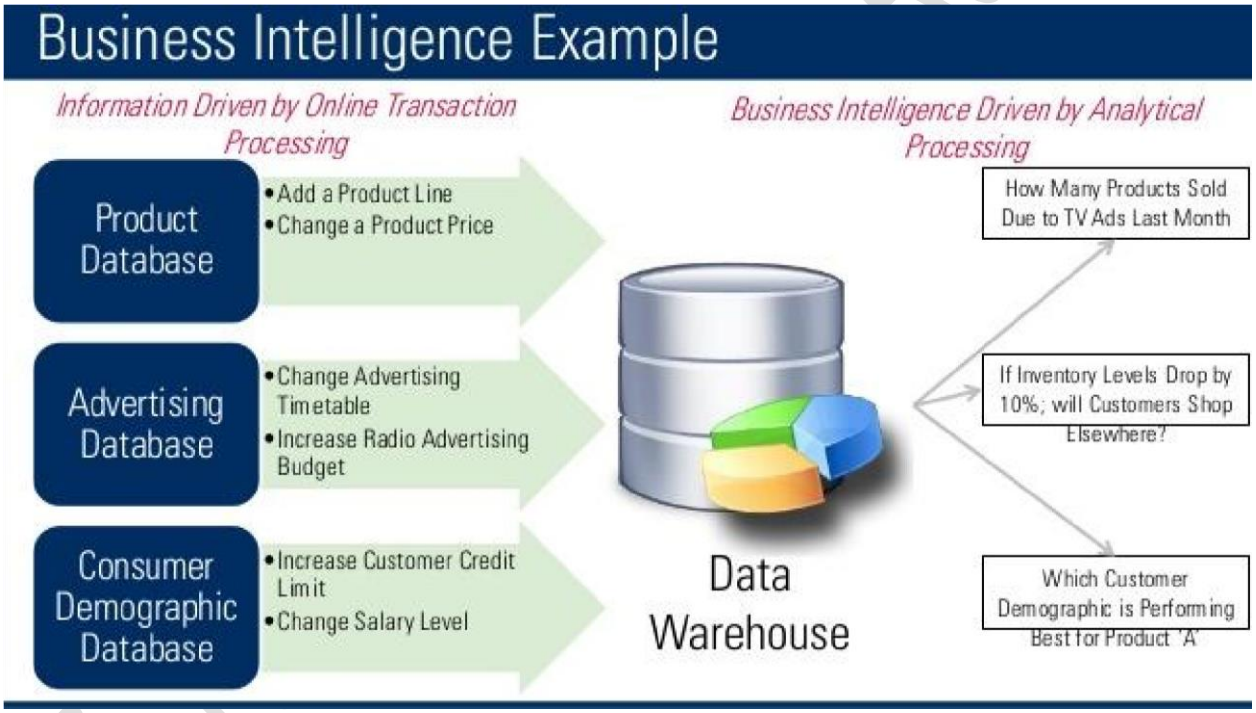
Decision Action Cycle User Applies Intelligence to Data To Produce Information This forms the basis of Knowledge, Knowledge is used in decision making Decisions trigger Actions, Actions in turn generate more data.

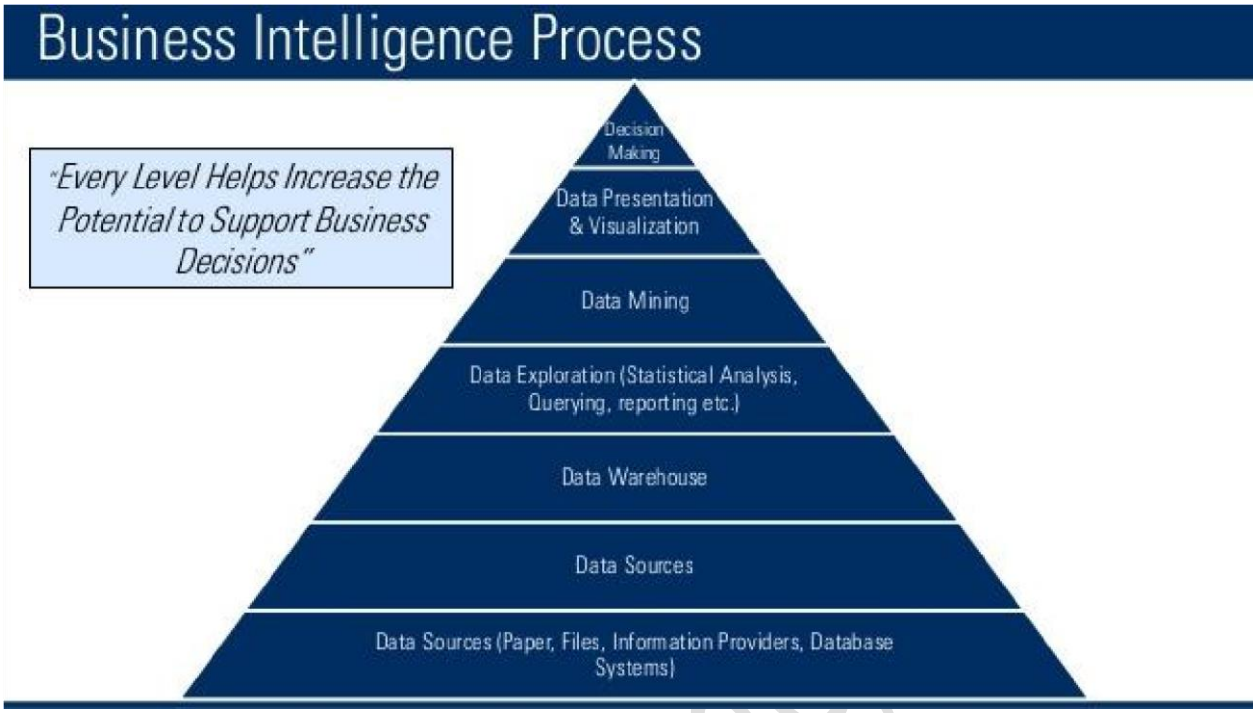


What is Business Intelligence? Technology that Allows:

- Gathering, storing, accessing & analyzing data to help business users make better decisions
- Set of Applications that Allow:
    - Decision support systems
    - Query and reporting
    - online analytical processing (OLAP)
    - Statistical analysis, forecasting, and data mining
- Help in analyzing business performance through data-driven insight: • Understand the past & predict the future


What is Business Intelligence
Technology that Allows:
• Gathering, storing, accessing & analyzing data to help business users make better decisions
Set of Applications that Allow:
• Decision support systems
• Query and reporting
• online analytical processing (OLAP)
• Statistical analysis, forecasting, and data mining
Help in analyzing business performance through data-driven insight:
• Understand the past & predict the future


Business Intelligence Example
Information Driven by Online Transaction Processing
Business Intelligence Driven by Analytical Processing
Product Database
• Add a Product Line
• Change a Product Price
Advertising Database
• Change Advertising Timetable
• Increase Radio Advertising Budget
Consumer Demographic Database
• Increase Customer Credit Limit
• Change Salary Level
Data Warehouse
How Many Products Sold Due to TV Ads Last Month
If Inventory Levels Drop by 10%; will Customers Shop Elsewhere?
Which Customer Demographic is Performing Best for Product 'A'



Business Intelligence Component

OLTP (Online Transaction Processing)
Online transaction processing, or OLTP, is a class of information systems that facilitate and manage transaction-oriented applications, typically for data entry and retrieval transaction processing

• Online transaction processing increasingly requires support for transactions that span a network and may include more than one company. For this reason, new online transaction processing software uses client or server processing and brokering software that allows transactions to run on different computer platforms in a network.

Advantages:

- Reduces paper trails
- Simple & effective
- Highly accurate Advantages

Disadvantages:

- Needs technical resources
- Needs maintenance
- Costly

OLAP(Online Analytical Processing)
Online analytical processing, or OLAP is an approach to answering multi-dimensional analytical (MDA) queries swiftly OLAP tools enable users to analyse multidimensional data interactively from multiple perspectives.

OLAP consists of three basic analytical operations: consolidation (roll-up), drill-down, and slicing and dicing. Consolidation involves the aggregation of data that can be accumulated and computed in one or more dimensions.

Types of OLAP

- Multidimensional
- Relational
- Hybrid

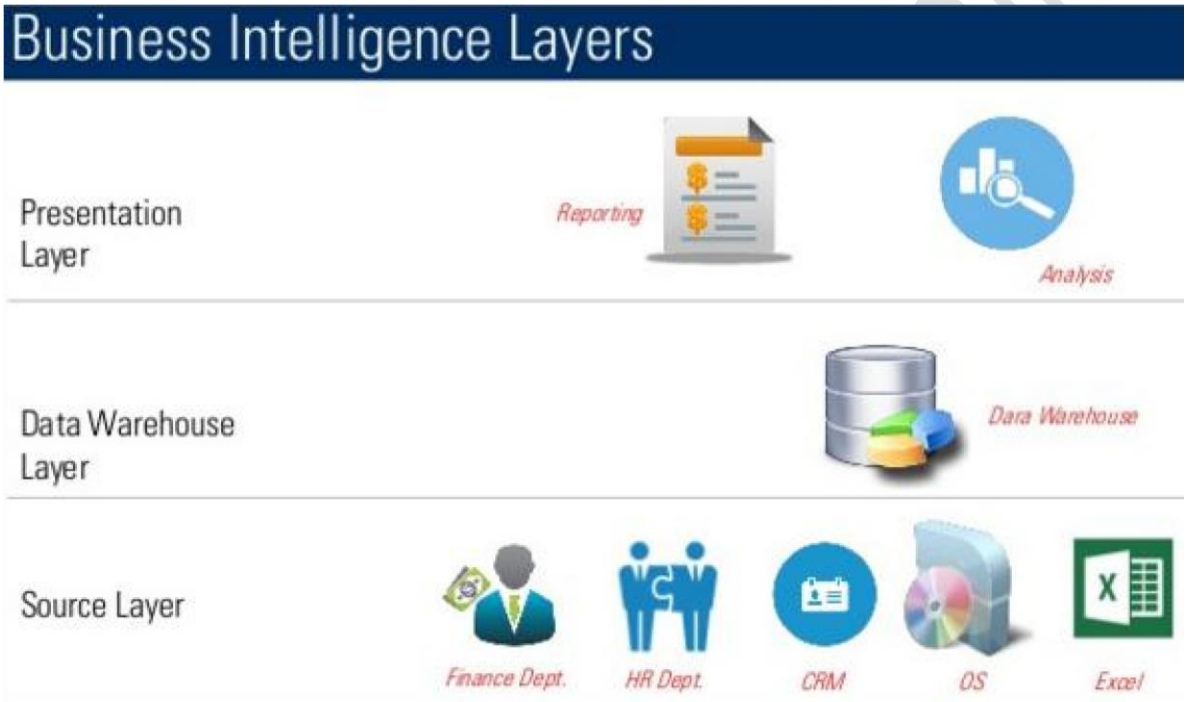
	OLTP	OLAP
User	Clerk/IT professional	Knowledge Worker
Function	Day to Day Operations	Decision Support
Database Design	Application Oriented	Subject Oriented
Data	Current, Isolated	Historical, Consolidated
View	Detailed, Flat Relational	Summarized, Multidimensional
Usage	Structured, Repetitive	Ad Hoc
Access	Read/Write	Read

Data Warehouse

A decision support database that is maintained separately from the organization's operational database. A consistent database source that bring together information from multiple sources for decision support queries Support information processing by providing a solid platform of consolidated, historical data for analysis. ―Data warehousing is the process of constructing and using data warehouses‖

Data Mining & it's Process

Data Mining is the computational process of discovering patterns in large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. • Before data mining algorithms can be used, a target data set must be assembled Pre Processing •Involves Anomaly Detection •Clustering •Classification •Regression •Summarization Data Mining •The final step of knowledge discovery from data is to verify that the patterns produced by the data mining algorithms occur in the wider data set Result Validation



Who makes decision?

Decision making at different levels:

– Operational

• Related to daily activities with short-term effect

• Structured decisions taken by lower management

– Tactical

• Semi-structured decisions taken by middle management

– Strategic

• Long-termeffect

• Unstructured decisions taken by top management Strategic Tactical Operational

BI is a collection of applications and software that analyzes various aspects of data and presents it in forms that enhance decision making. It has evolved from generating rudimentary reports and tools used for historical query, to include a host of components such as forecasting, online analytical processing, predictive modeling, data management, data mining, and optimization. Armed with these essential tools, companies can accurately assess what is or is not working at present, discern what historical factors occurred to make it so, and readily identify future trends to maximize their potential.

Trending

One of the most tangible benefits of BI is its ability to offer predictive analysis of data that relates to future occurrences. Analytics such as data mining, forecasting, and online analytical processing (OLAP) discern relationships between data that make it possible to determine future trends. Data mining tools analyze specific components of data through parameters such as path analysis and association, which in turn determine useful relationships between data events. Clustering parameters deliver a visual representation of information not previously discerned, while forecasting provides a solid basis for the anticipation of trend fluctuation.

Data integration is a vital feature of any BI solution and can account for factors that actually affect specific market concerns, such as customer demographics, economic conditions, and marketplace environments. These predictive analytics allow enterprises to refine business processes in preparation for the future. One can argue that the individual tailoring of business processes is an essential distinction between enterprises in the same industry. They often utilize similar technologies and products, but customize them in vastly different ways.

Predictive Modeling

Another feature of most competitive BI products, particularly those involved in various aspects of design, is predictive modeling. This facet of BI works in conjunction with forecasting and analytics to help provide visual representations of hypothetical scenarios, permitting for companies to literally ―see‖ which option works best. Furthermore, its integration with other BI tools allows it to access the same data that is relevant to a company, therefore influencing the effectiveness of alternative business scenario models.

BI offers predictive modeling for both quantitative and qualitative factors. Users simply input their company's data into a variety of channels that analyze and produce different outputs, all of which utilize the same predefined rules that address the particular concern of a hypothetical situation. This presentation of multiple scenarios enables users to efficiently and cheaply examine their options and predicted results before selecting the most desirable. Additionally, this aspect of BI encourages a culture of innovation and creativity, since employees can readily attempt different situations with minimal cost and waste of company expenses.

A further boon of BI is the degree of flexibility in the types of data it can analyze and incorporate into the decision making process. It takes into account a host of logistical and marketing concerns that could affect a company's processes and, with predictive modeling, can provide an impact analysis that offers end-to-end results of varying scenarios. This particular feature is ideal for

potential actions that have high implementation costs, since the design aspect is readily calculable via BI. Predictive modeling can also yield representations that are accessible to nontechnical users who can easily manipulate them via assorted input factors. The total integration of all of the individual components of BI – forecasting, reporting, OLAP, data management and predictive modeling – allows for real-time analysis of the most viable courses of actions for businesses based on their own data.

Centralization

BI facilitates the centralization of data so that it is accessible to a variety of departments and end users. This is particularly useful for enterprises consisting of multiple silos, but still requiring a comprehensive overview of processes between departments and the entire enterprise. BI products can be specific to a particular division of a company, provide data that pertains to a specific project, and to the company's objectives as a whole. They include data integration technology that allows for the storing of all data relevant to a particular function of a company such as sales, orders, shipping, and pricing. Competitive BI solutions can also account for outside factors relating to a specific industry, including nationwide factors such as GDP, interest rates, or competitor data.

BI permits users to access the information they need in a single view that can be stratified as necessary. OLAP provides extremely specific representations of data – frequently stored in a multidimensional database – so that users can examine it from a variety of viewpoints with attributes that may include time, pricing, and other quantifiable information. Users can choose to extract whatever data is most useful at the time and analyze it in relation to others. Mobile applications allow for access anywhere there is an Internet connection.

Customer Benefits

BI presents a number of distinct advantages for businesses that deal directly with consumers. It offers valuable data that can assist with opportunities for up-selling and cross-selling products, giving users the ability to readily identify new markets. Several BI products utilize decision support systems that present data relevant for comparisons and a particular demographic. Users can access previous customer habits as well as determine products and services which directly correlate with such habits; there are abundant cross promotional opportunities. Sales and marketing efforts become increasingly streamlined as a result, increasing the likelihood for customer satisfaction. The usage of analytics and predictive models enables enterprises to identify what services will affect their customers the most, as well as to record the information gained from customer contact. By keeping this information in a single database that representatives from various company divisions can access, BI products enable a more fulfilling customer experience, increase customer retention, and allow sales and marketing personnel to make the most of their resources; overall company efficiency is increased, reducing costs and increasing revenues.

BI facilitates proper enterprise structure by providing an assortment of data that details its needs. It can identify areas that have a dearth or surplus of attention, expediting the streamlining of company resources that can assist in productivity. Competitive BI solutions allow for immediate updating, granting users the most current information about company processes and their effects. These effects are critical for inventory applications, utilizing real-time and predictive analytics to eliminate overstocking and to allow customers to receive their products and services in a timely

fashion. Prudent BI users can also accumulate data regarding the company processes of competitors in an effort to analyze which approaches are working or not, and to elucidate points of distinction between them.

Introduction to Reporting

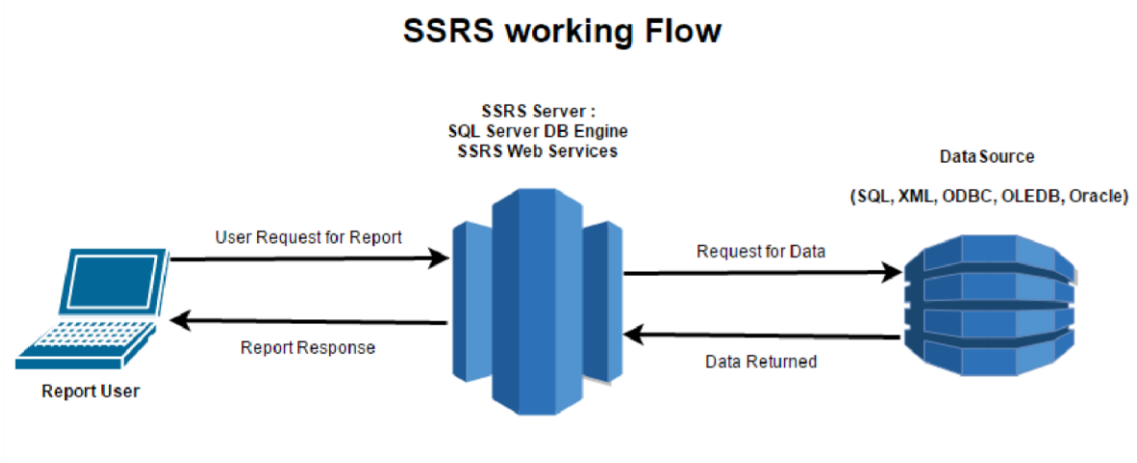
Reports display a collection of information from the records in your Workbooks system. You can create simple reports to list a set of fields, being returned in the form of columns and rows, much like a spreadsheet. Or you can build a summary report which summarises information, and groups information together. You can use formulae such as totals, counts or averages.

SSRS is a component in the Microsoft Business Intelligence (BI), which helps in enterprise data analysis. SSRS is a comprehensive, extensible reporting tool used to design, test and deploy reports. Using SSRS, you can create a wide variety of reports and deploy them to the server. These deployed reports can be viewed in browsers. These reports can be exported in the respective file formats as the user needs.

SSRS comes with a lot of advantages than the previous reporting tools such as Crystal Reporting Services. As SSRS is a part of SQL Engine, thus it's good at execution measurements. Moreover, it provides a wide range of options for reports creation and Chart and Graphs controls for KPI reports. In the earlier days, we needed to depend on third party tools like fusion charts etc.

Workflow of SSRS

The SSRS Server gets a user request for a report, following which it creates a request for data to Data Source. After getting the Data, it generates the Report in a specified format and returns the report as response to the user. The presented flow charts below show the described process.



Benefits of SSRS over Crystal Reports 1.

SSRS may use Non-SQL Server Data.

2. SSRS is —Free‖ with MS SQL Server.
3. SSRS is —Free‖ with SQL Express (with limitations).

4. Crystal reports need a Crystal Report Server or a third Party tool for development, while SSRS is built-in with MS-SQL.
5. SSRS is a preferred choice for complex reports, where performance is needed.
6. SSRS has a Query Builder (Designer).
7. Sub-Reports help in reusing your existing reports to create newer ones.

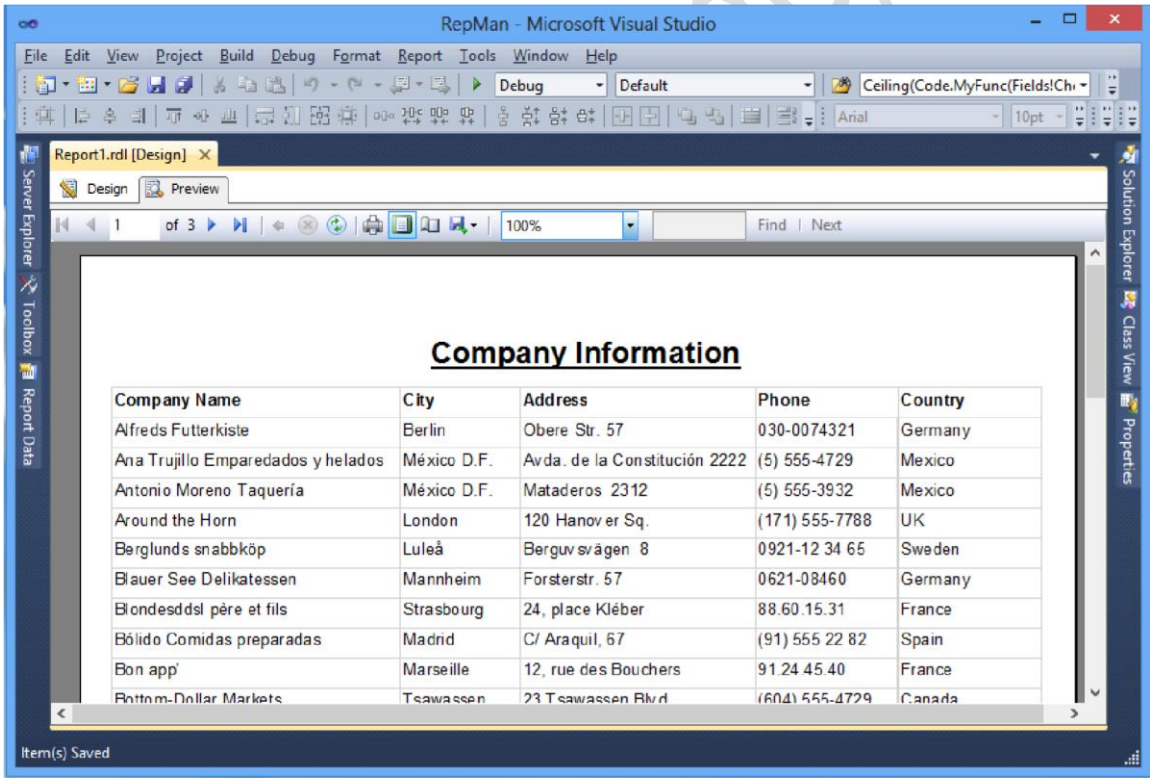
Features of SSRS

1. SSRS is a report creator and rendering engine for exporting reports into various file formats.
2. Data from variety of data sources like SQL server, OLEDB, XML, Oracle etc.
3. Embed graphics, images and external content
4. SharePoint Integration
5. Report Access based on URL
6. Graphs and Charts controls for creating KPI based reports (2008 onward)

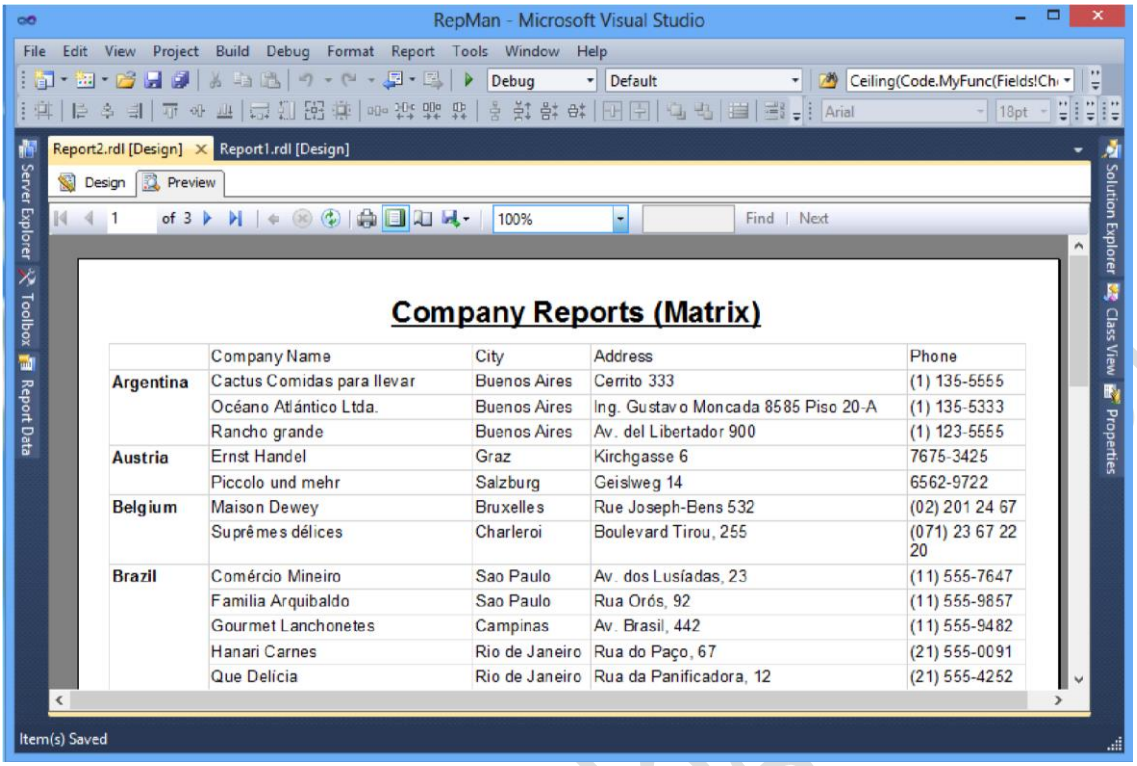
Reports Types Supported by SSRS

SSRS supports a wide variety of report types including:

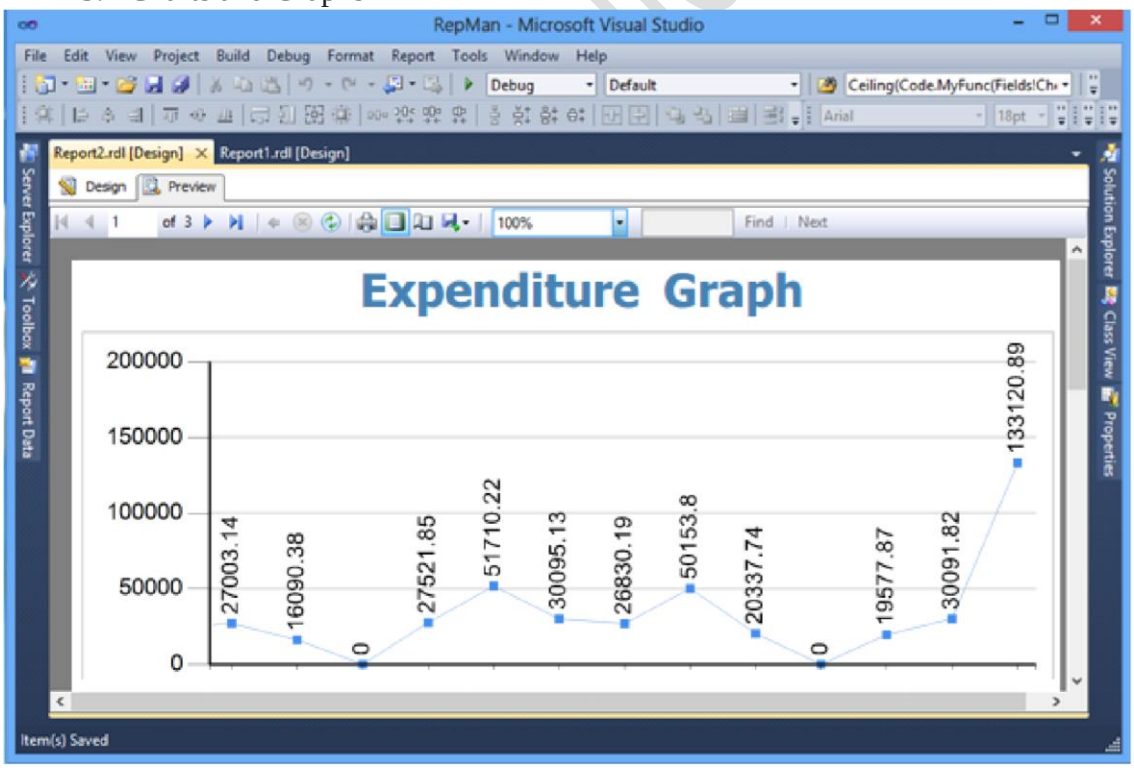
1. Tabular Reports (Tablix)



2. Tabular Reports (Tablix)



3. Charts and Graphs



Additional Report Types

1. Reports that have Totals and Subtotals
2. Reports with Parameters and Groups
3. Reports containing Aggregate Data Functions (Average, Division, Percentage etc.)
4. Drill-Down Reports

Output Formats Provided by SSRS

1. HTML
2. PDF
3. Excel
4. Word
5. TIFF
6. CSV
7. XML

Data Sources

SSRS interacts with the following data sources as needed:

1. Microsoft SQL Server
2. Microsoft Access
3. OLEDB Data Source
4. ODBC Data Source
5. XML Data Source
6. Oracle

SSRS Architecture

An SSRS deployment must be associated with a SQL Server instance. On the instance will be two databases, by default:

1. ReportServer – contains the report definitions, configuration, history, security of deployed reports and more
2. ReportServerTempdb – much like tempdb, it is used as a workspace for building reports and doesn't maintain any objects permanently.

We will also need a location for the Report Server Web Service, which can be on the same server as the databases, as in the simple deployment architecture shown in Figure 1, or on a different server. On whichever server we choose, we will have access to a Report Manager website that allows us to deploy and manage the reports. End users can run reports from Report Manager, create subscriptions, and publish their own reports if they have permission.

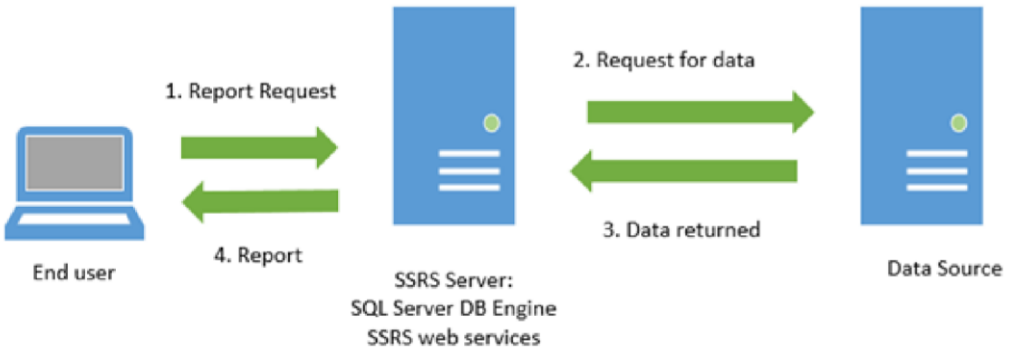


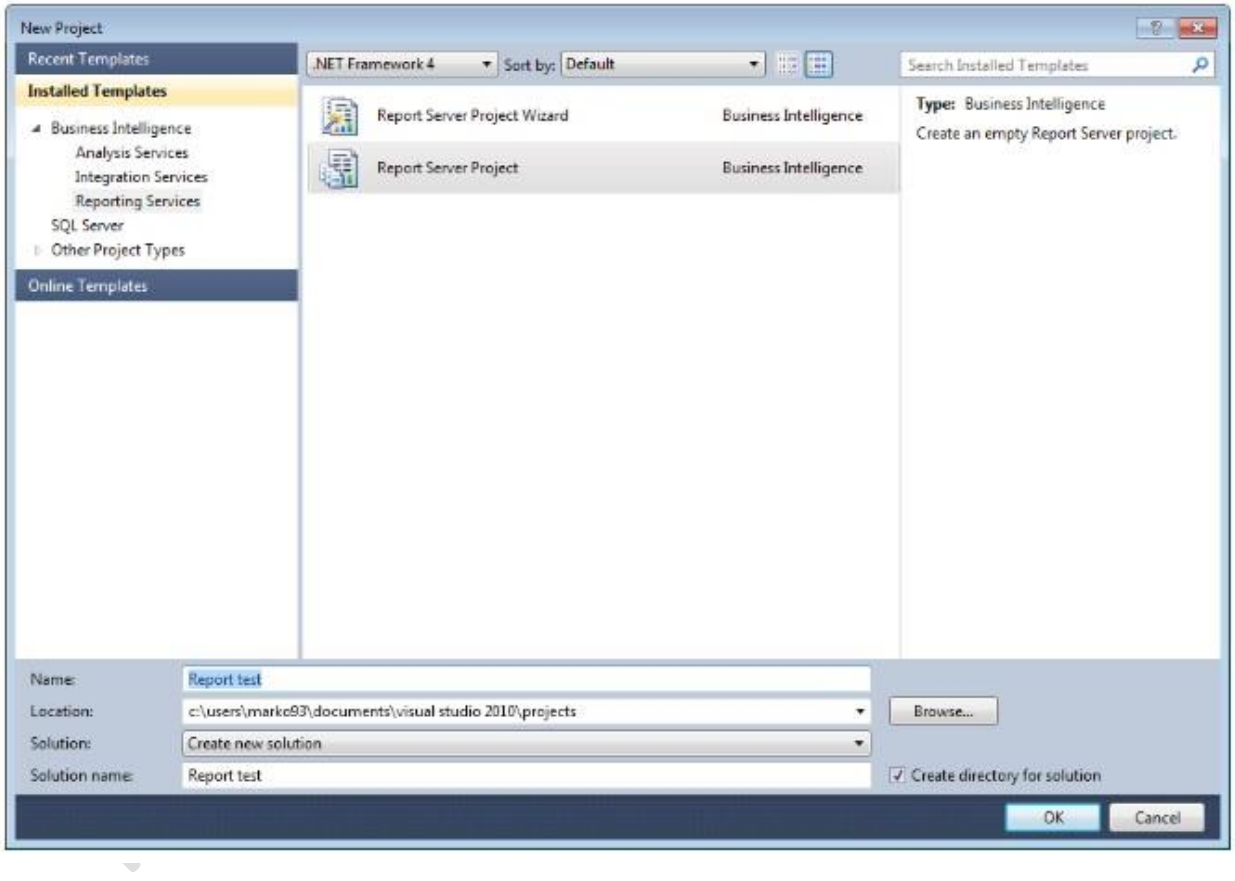
Figure 1

The end user sends an HTTP request for a report, providing any required parameters. The SSRS server finds the metadata of the report and sends a request for data to the data sources. The data returned by the data sources is merged with the report definition into a report. As the report is generated, it is returned to the client.

Create a report server project

The first thing to do is to create a report server project because we will need this later for saving report definition files and any other files that you need for creating SSRS reports. The first step is to open SQL Server Data Tools. Click on the File menu, findNew and then Project.

After this, you need to click on the Business Intelligence. Click on the Reporting Services and then Report Server Project. If you want to display Report test project to get you started, you can typeReport test in NameThe last step is to click OK to finish.



Create a new report definition file

In the View menu find the Solution Explorer, and then right-click on theReports folder. In the Add section click New Item.

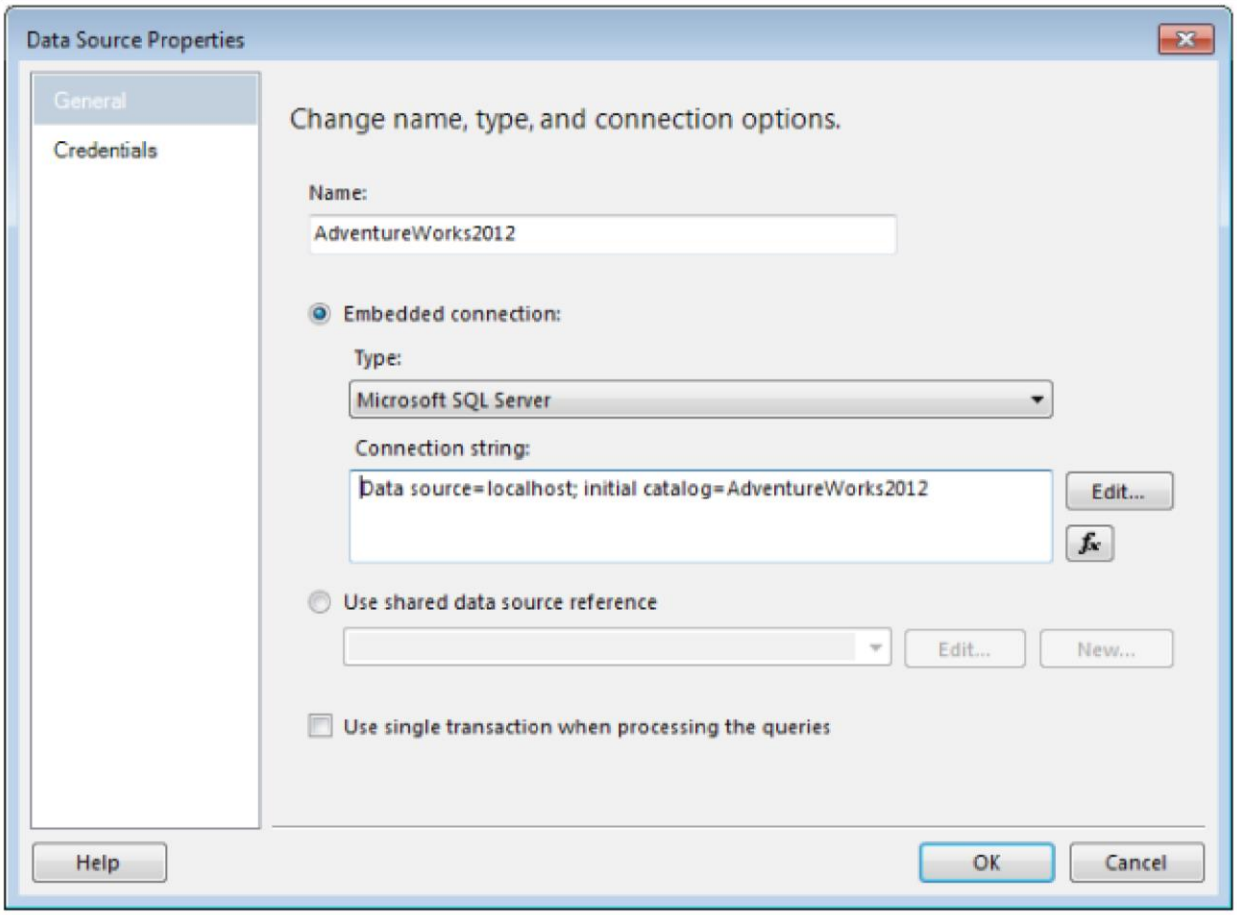
In the window Add New Item, click Report.

As you can see Report Designer has two available views. In Design view you define your report layout, and in the Preview view you can run your report.

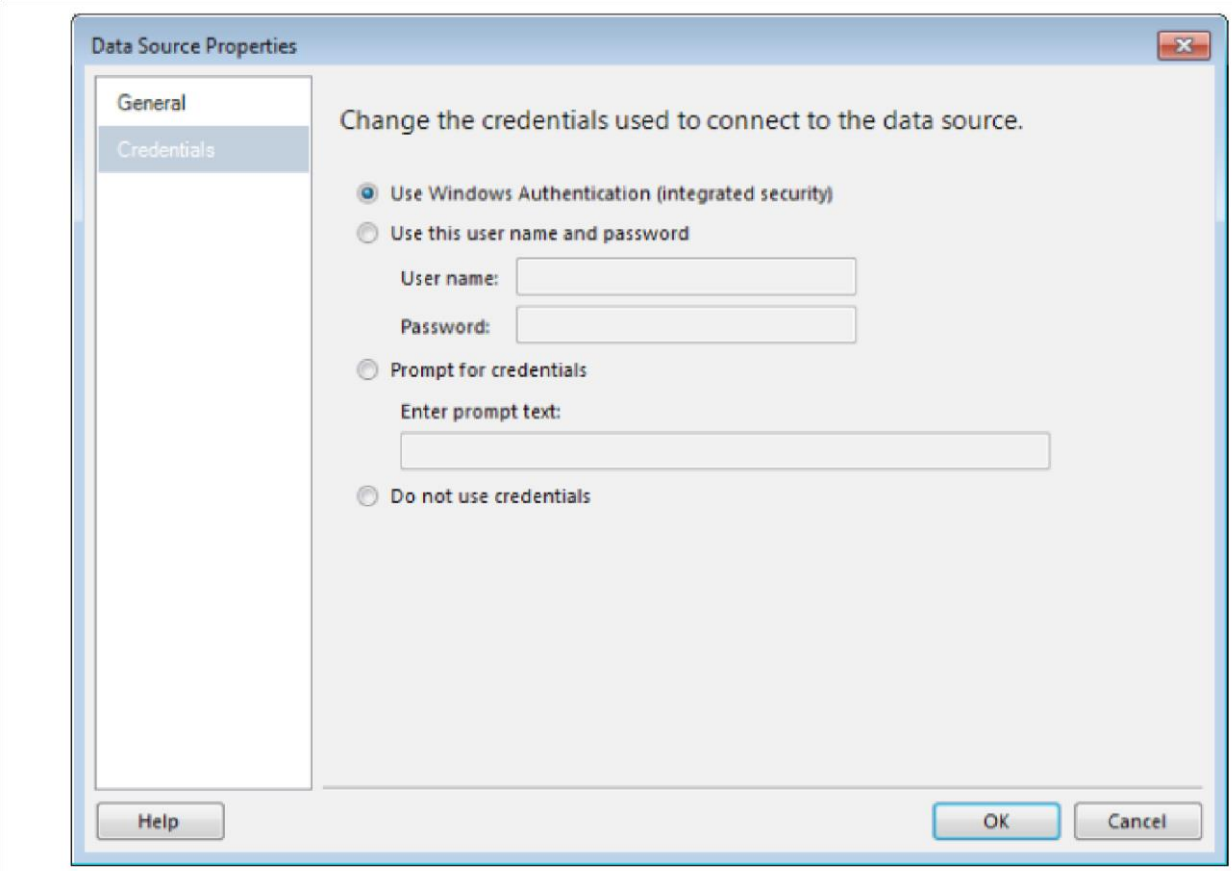
Defining data source and dataset
Setting up a connection

In order to retrieve data from a database or from some other resource, you will need to define the data source. In the following section, you will see how you can define the data source. We will use AdventureWorks2012 database as an example.

In the View menu find and click Report Data, then New and after that Data Source
Type AdventureWorks2012 in the field Name
Select Embedded connection As Type
select Microsoft SQL Server In
Connection string field type:
Data source=localhost; initial catalog=AdventureWorks2012



Note: If the database is not on the local computer, replace localhost with name of your database server instance.
After this, click on the Credentials and then Use Windows Authentication. Click OK, and you're done.



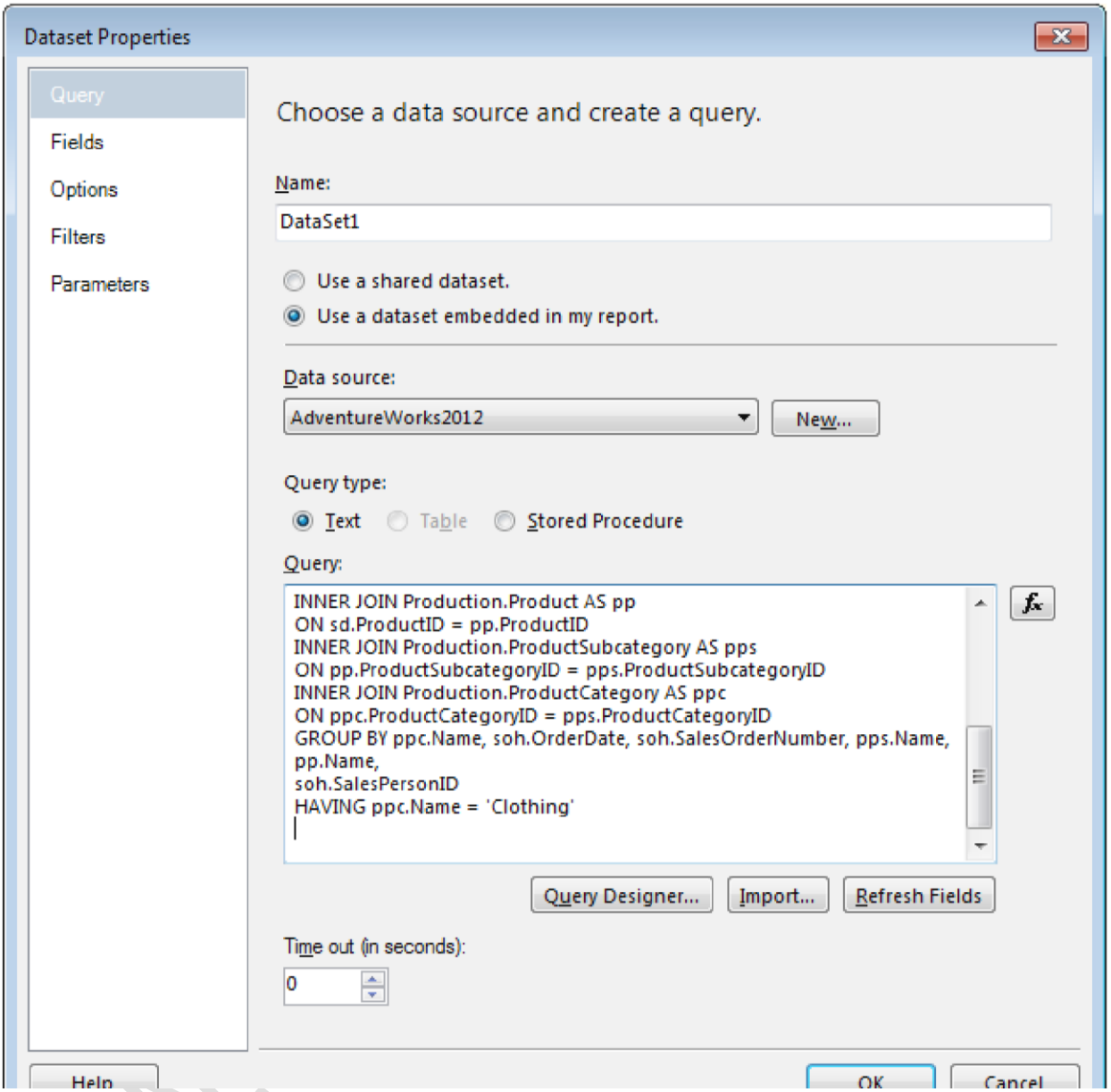
Define a T-SQL query for report data

Define a T-SQL query for report data

In this section you will learn how to specify a dataset. You will need some basic SQL knowledge in order to create a query and define what information you will need from your database. The query you will see is just an example, and for your own purposes and database, you will have to change that step.

- In the Report Data pane find and click New, and then Dataset. In the Dataset Properties dialog box as Name type DataSet1 Make sure to check Use a dataset embedded in my report. Then select a AdventureWorks2012 as your data source,
- Check Text as a Type, and type query into the Query input:

Source: Defining a Dataset for the Table Report (Reporting Services)



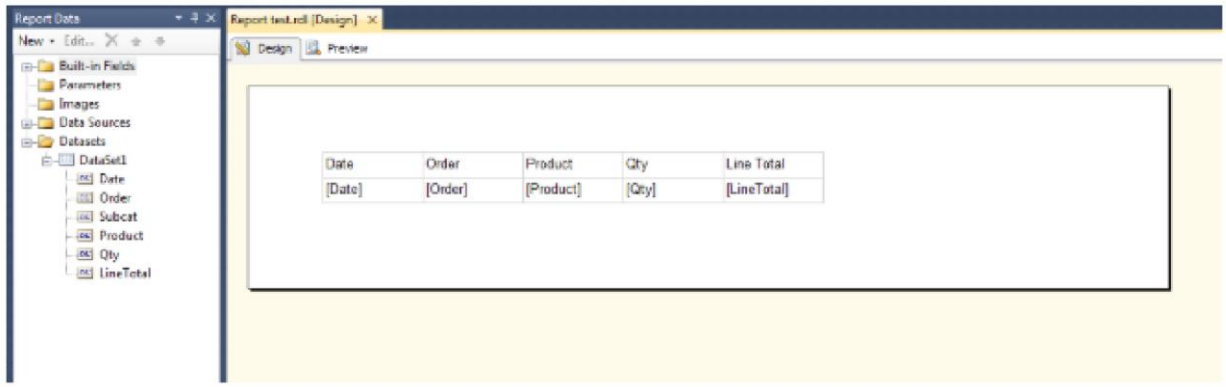
Note: soh, pps, sd, pp, ppc are just shorter names for tables that we have in AdventureWorks2012 database.

How to add a table and fields to a report layout

After finishing with previous technical details, we can start with more interesting things, like designing your first SSRS report. This part is easy because instead of writing code, you can dragand-drop graphic icons into the report format. This section will show you how to add Table and Fields to your report.

- In the View menu click Toolbox, then find and click Table and drag the mouse to the design area.

- In the left pane, you can expand the dataset Dataset1 in order to see all the fields. Drag one of the field (e.g. field Date) from Report Data to the column in the table
- You can continue adding fields, table will automatically add more columns.



Preview report

If you want to preview your report to see how it all looks, to correct errors, to correct issues or to verify design and data connection, click tab Preview.

Tips and tricks

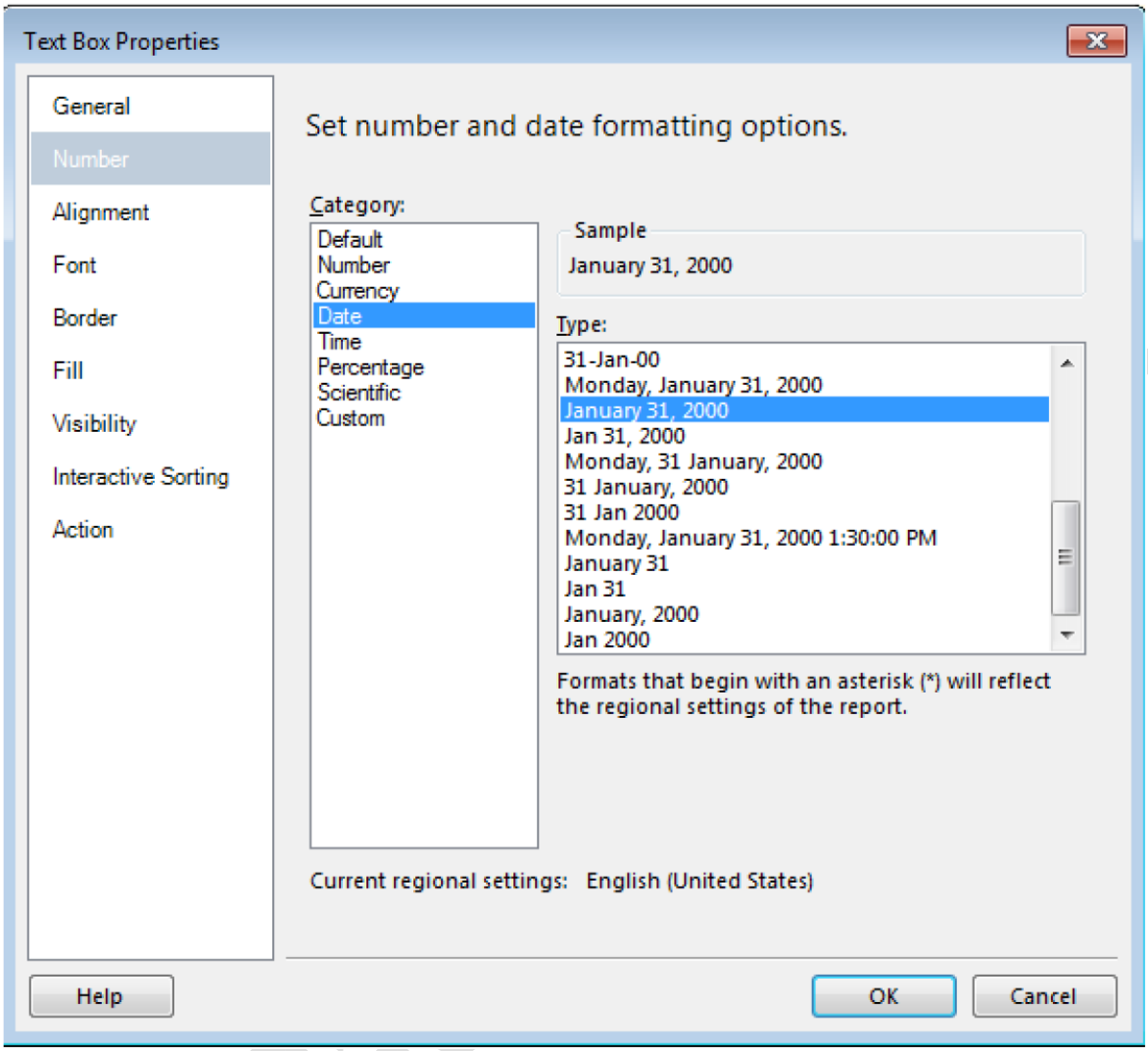
An easier way to add a table is to right-click on the design surface, click Insert and then click Table.

How to format your Report

Format currency and date

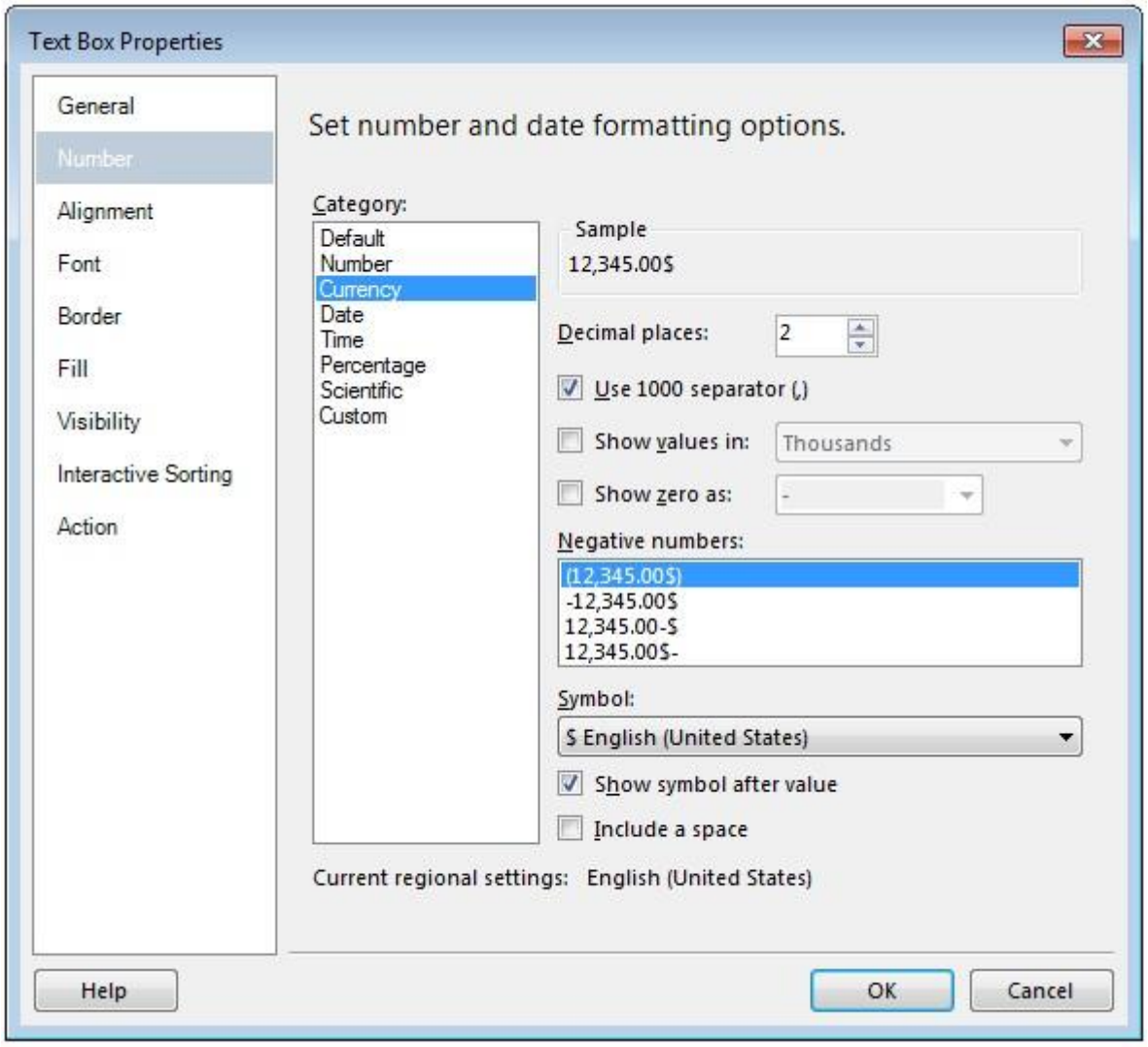
If you want to format Date field to show only the date, follow the steps below:

- In the Design tab, right-click the desired cell, then click Text Box Properties. Find and click Number, then in the field Category click Date
- As Type select date format you want, and click OK to finish.



If in your table you have a field that holds information about currency, but you only see ordinary numbers, you can format that field to display a number as currency:

- In the Design tab, right-click the desired cell, then click Text Box Properties.
- Find and click Number, then in the field Category click Currency In accordance to your need you can change defaults
- Click OK to finish.



Changing column width and text style

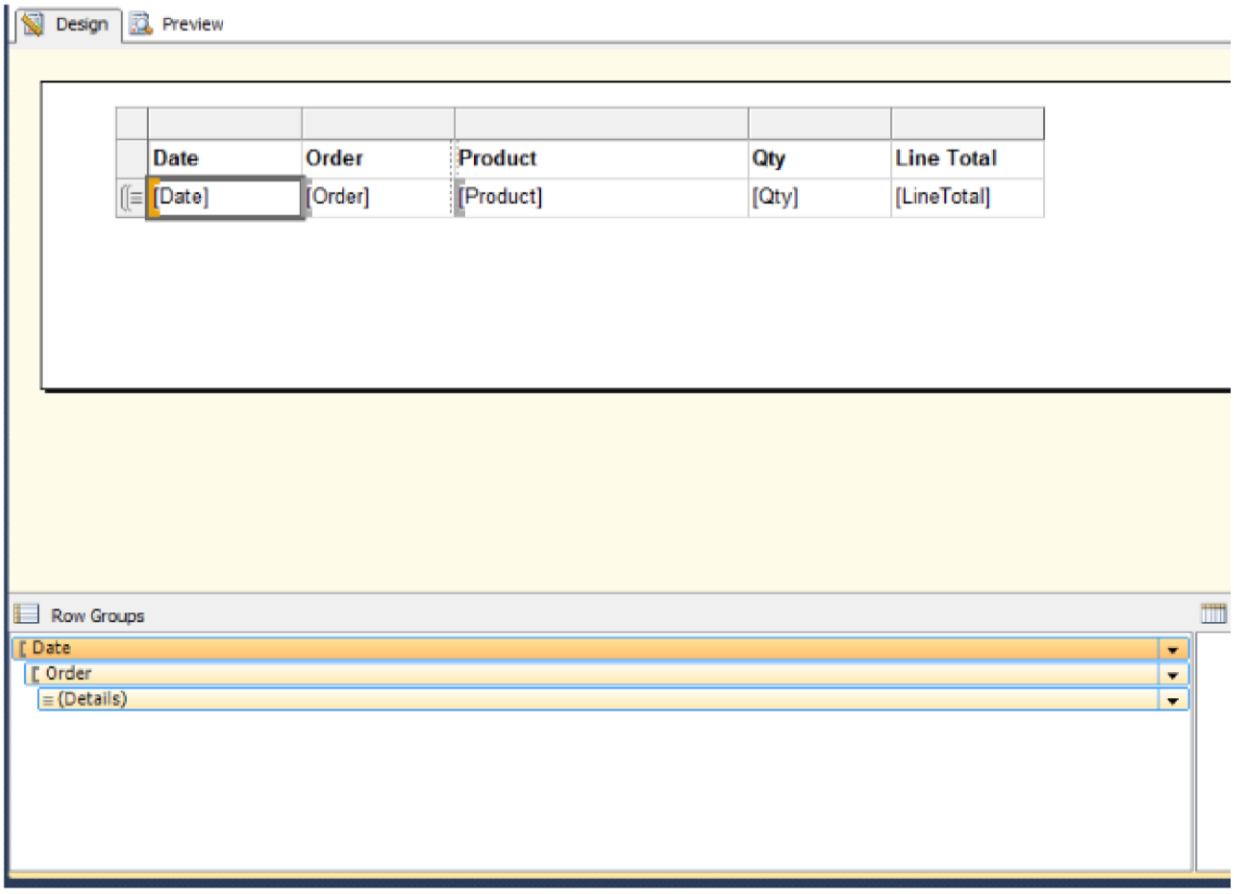
You also have an option to change text style (font, size etc.), and to change column width.
Column width you can change by simply dragging the columns to the desired size.
Text style you can change by clicking the Format menu. In Format menu find Font, then click whatever you need (bold, italic, etc.)

Adding Grouping

If you want to make data set in your SSRS report do the following:

Click Design tab, and then choose pane Row Groups

- Drag the field you want to group to the pane Row Groups From your report pane drag some other field you want to group.
- Delete the old columns to the double line.
- If new columns need to be format, just right-click the cell and then click Text Box Properties, the next steps are the same like in the formatting report section.

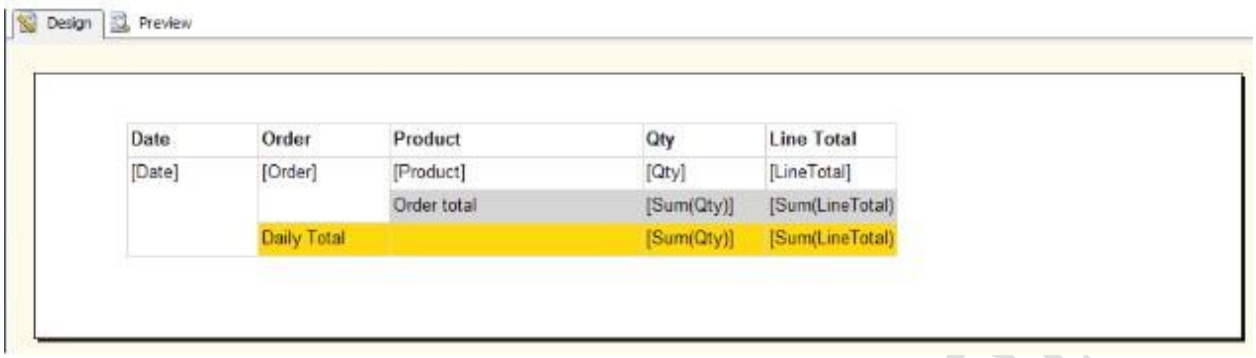


Tips and tricks
You can do the same by right-clicking on the surface and clicking View, and then Grouping.

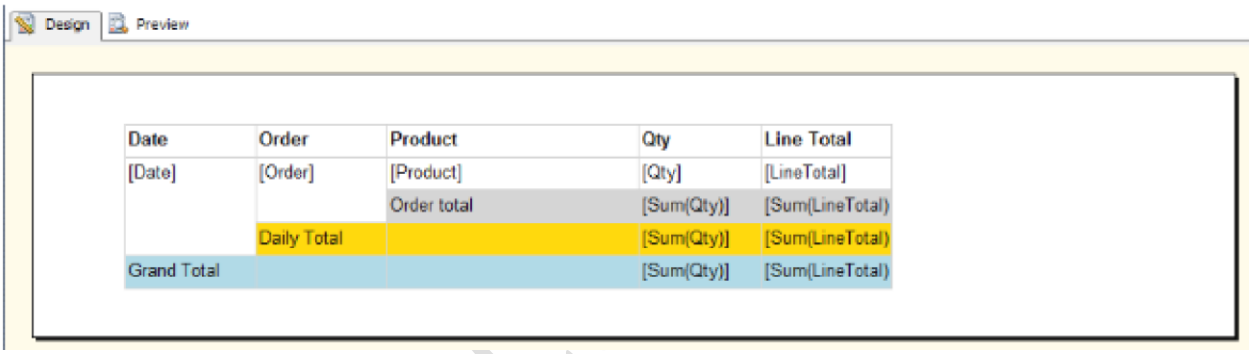
Adding totals
Total is the sum of numeric, non-null data in the data region, and if you want to add totals for a group, you can do that by clicking Add Total for the group in the Grouping pane, and if you want to add totals for an individual cell just click Add Total for the cell.

Add a daily total and grand total

- Right click the cell [Order] and choose Add total, then click After.
- Type Daily to format a new name, Daily Total
- After that, select the new cell [Daily Total], two Sum cells and the empty cell you see between them.
- In the Format menu, choose background color. We chose color orange.



- Right click the cell [Date] and choose Add total, and then After.
- Type Grand to format a new name, Grand Total
- Select the new cell [Grand Total], the two [Sum] cells and the empty cell you see between them.
- In the Format menu, choose background color. We chose color light-blue.



Tips and tricks

After you add total, you can change the default function Sum. There is a list of different function you can use (avg, count, etc.).

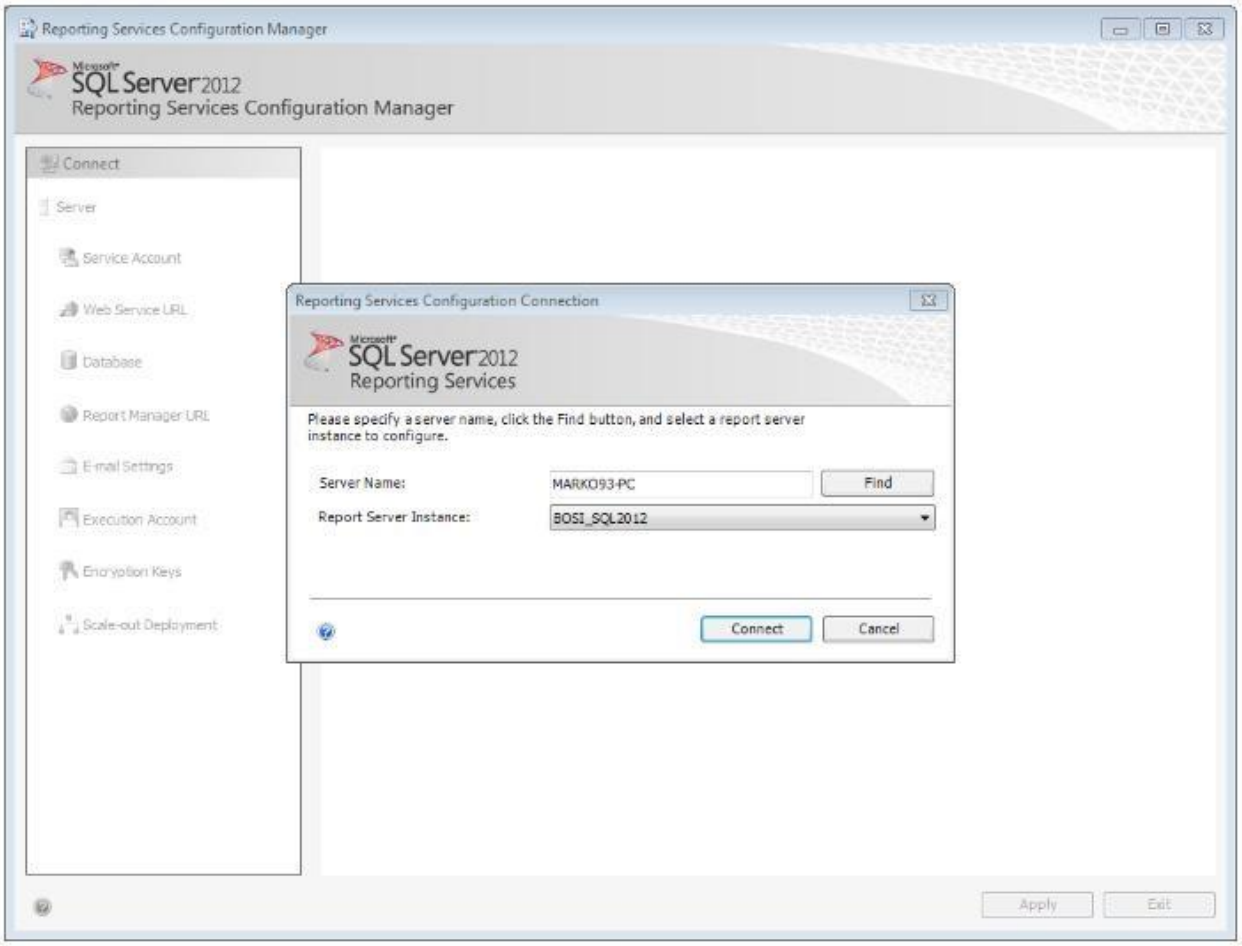
Publish your Report to the report server

Finally, when you finished with creating your first SSRS report, you may want to publish the report:

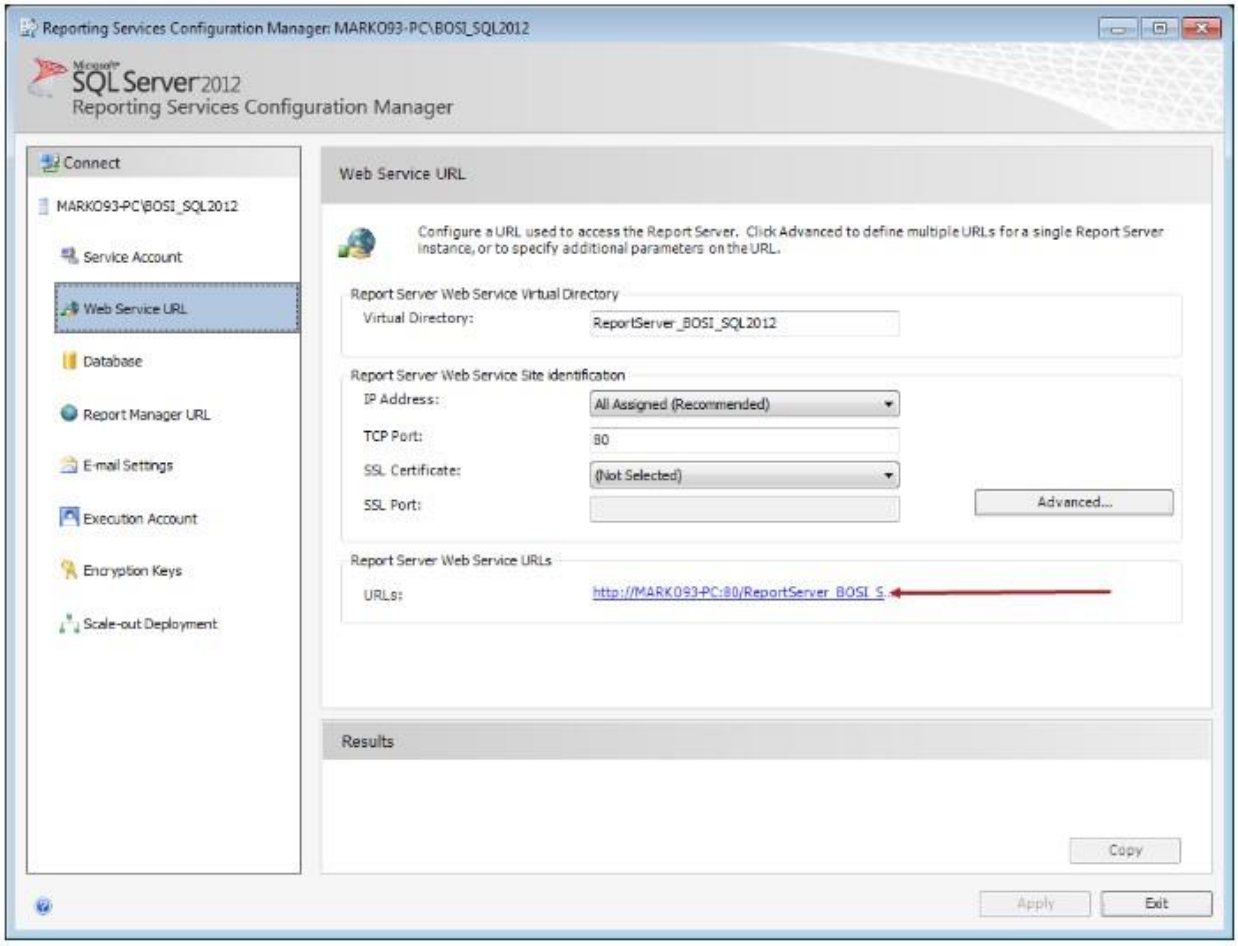
- In the Project menu click Test report Properties
- Put your report server's name in the field TargetServerURL, then click OK.
- In the Build menu click Deploy Test report. You will get a message that indicates whether you have successful or unsuccessful deployment.

Configure Report

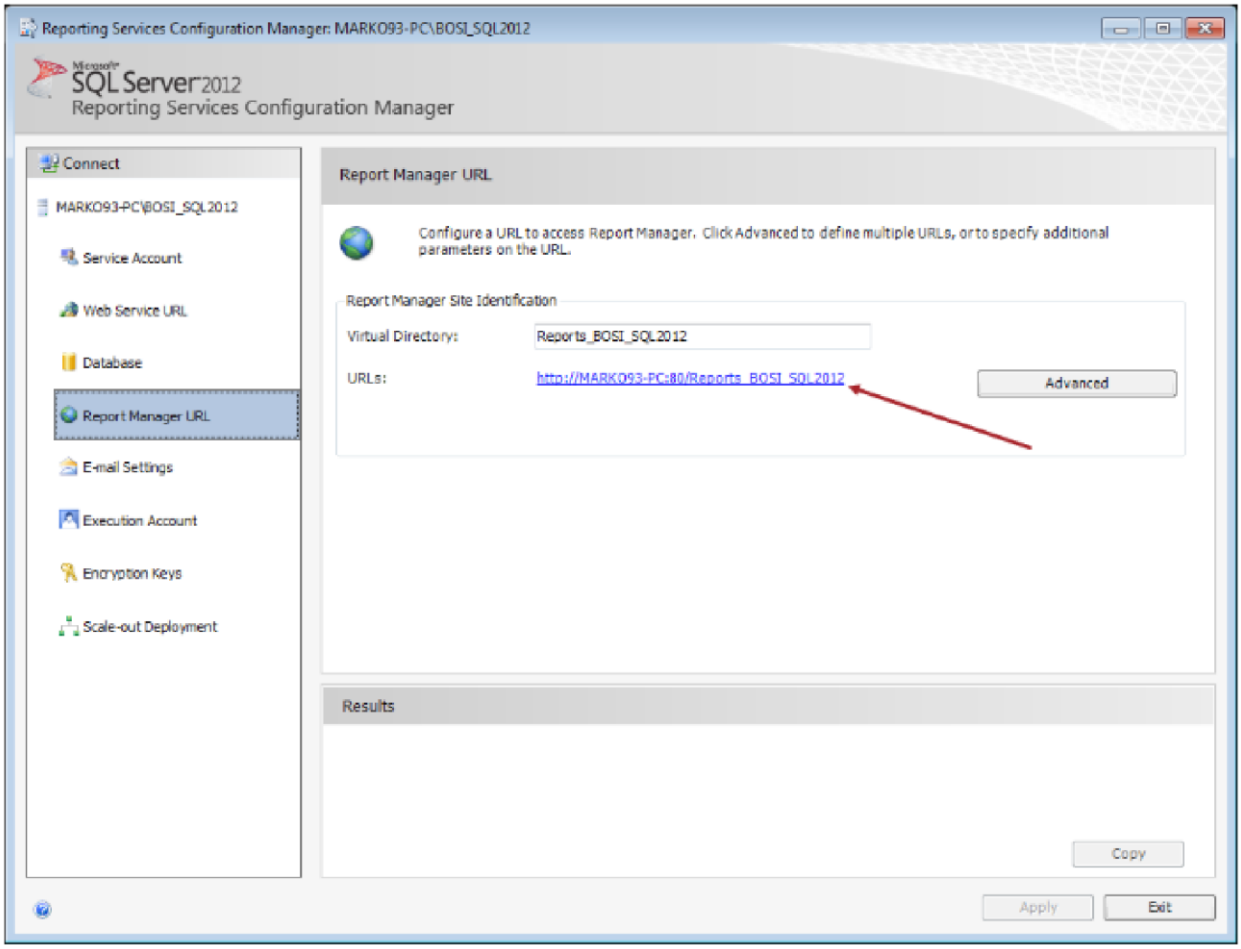
The first step that you need to do in configuration is to launch the SQL Server Reporting Service Configuration Manager and connect to the Reporting Service.



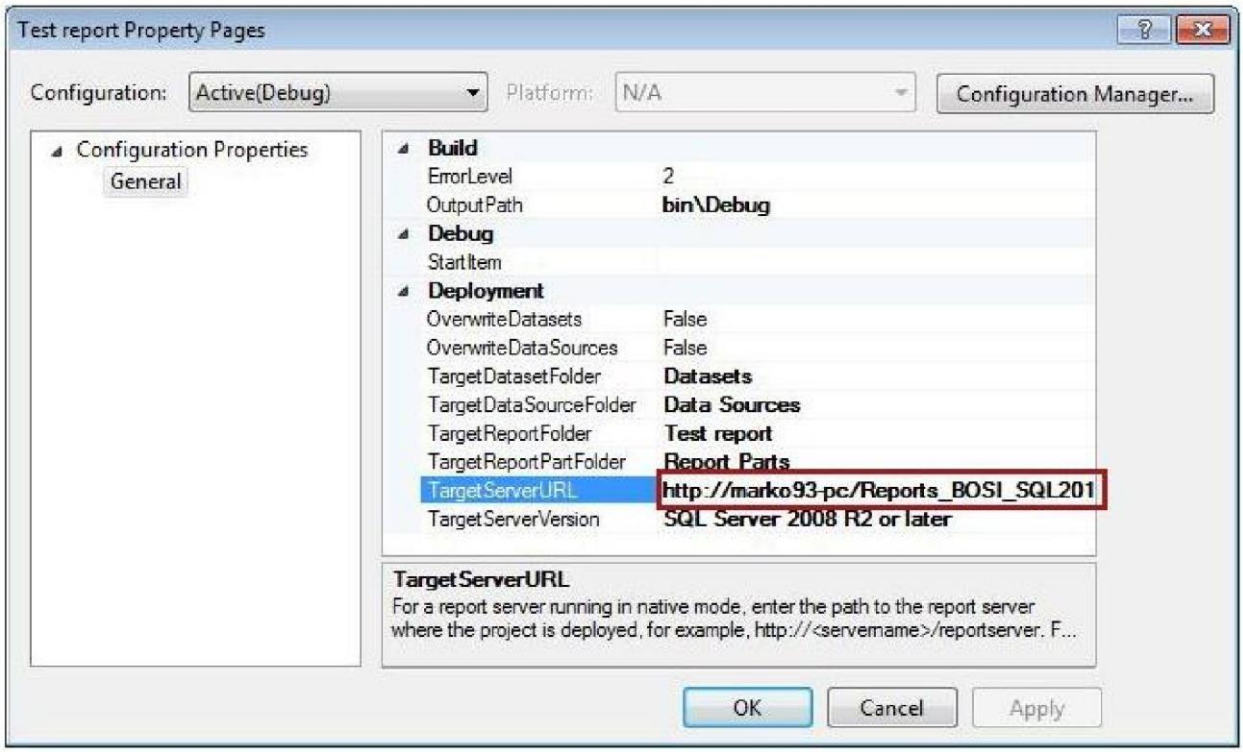
After that in section Web Service URL you can find URL to the report server.



In section Report Manager URL you can see URL for viewing and managing reports.



In the Project menu open the Test report Properties.
Note: Prefix ―Test report‖ is name of your report.
Final step in configure target server URL is to fill the Web Service URL in the TargerServerURL property with correct URL that you find in Report Manager URL.



Introduction to Data Analysis

The volume of data that one has to deal has exploded to unimaginable levels in the past decade, and at the same time, the price of data storage has systematically reduced. Private companies and research institutions capture terabytes of data about their users' interactions, business, social media, and also sensors from devices such as mobile phones and automobiles. The challenge of this era is to make sense of this sea of data. This is where **big data analytics** comes into picture.

Big Data Analytics largely involves collecting data from different sources, manage it in a way that it becomes available to be consumed by analysts and finally deliver data products useful to the organization business.

The process of converting large amounts of unstructured raw data, retrieved from different sources to a data product useful for organizations forms the core of Big Data Analytics.

Microsoft SQL Server Analysis Services (SSAS)

Microsoft SQL Server Analysis Services (SSAS) delivers online analytical processing (OLAP) and data mining functionality for business intelligence applications. Analysis Services supports OLAP by letting you design, create, and manage multidimensional structures that contain data aggregated from other data sources, such as relational databases. For data mining applications, Analysis Services lets you design, create, and visualize data mining models that are constructed from other data sources by using a wide variety of industry-standard data mining algorithms.

The basic idea of OLAP is fairly simple. Let's think about that book ordering data for a moment. Suppose you want to know how many people ordered a particular book during each month of the year. You could write a fairly simple query to get the information you want. The catch is that it might take a long time for SQL Server to churn through that many rows of data.

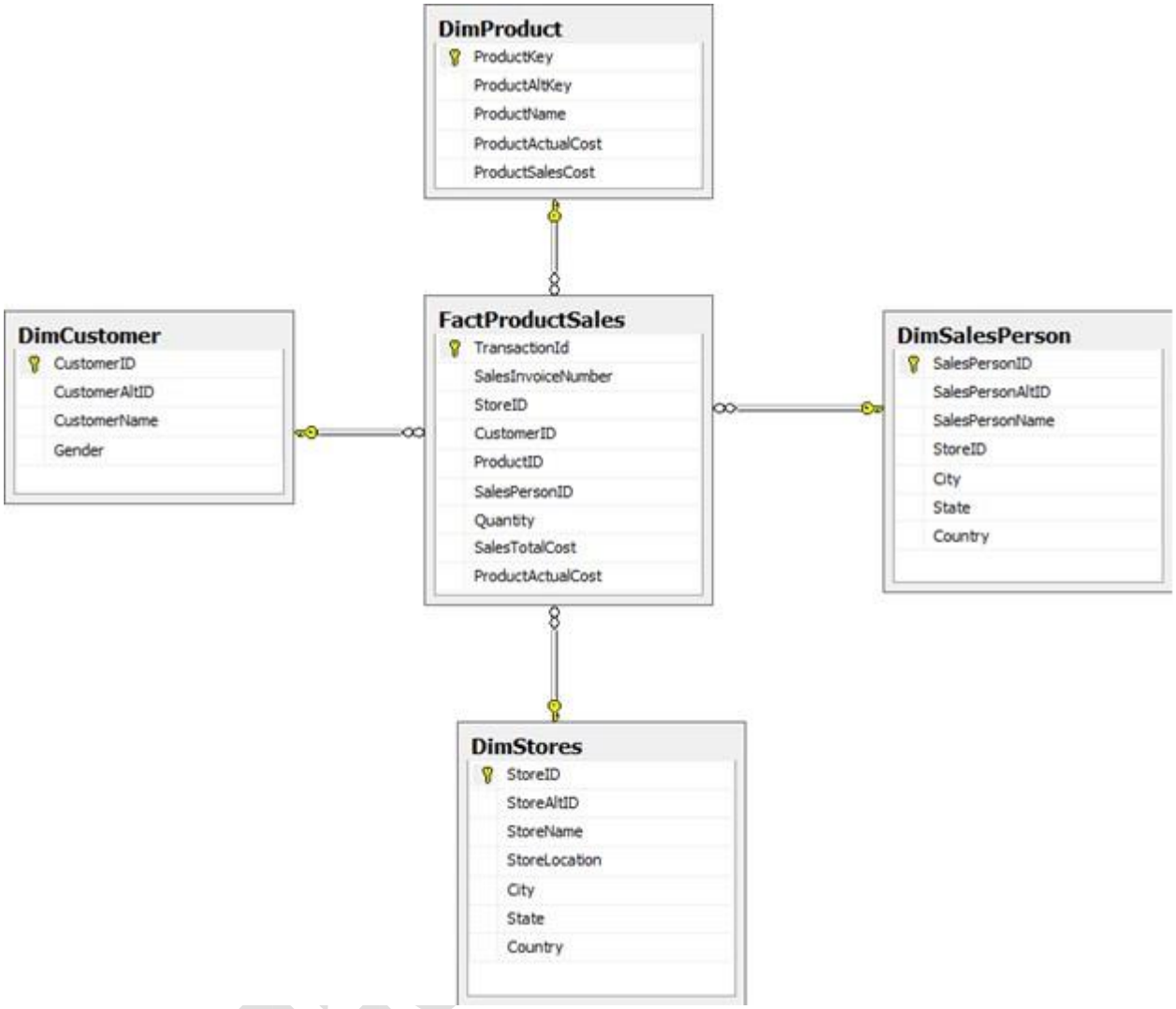
And what if the data was not all in a single SQL Server table, but scattered around in various databases throughout your organization? The customer info, for example, might be in an Oracle database, and supplier information in a legacy xBase database. SQL Server can handle distributed heterogeneous queries, but they are slower.

What if, after seeing the monthly numbers, you wanted to drill down to weekly or daily numbers? That would be even more time -consuming and require writing even more queries.

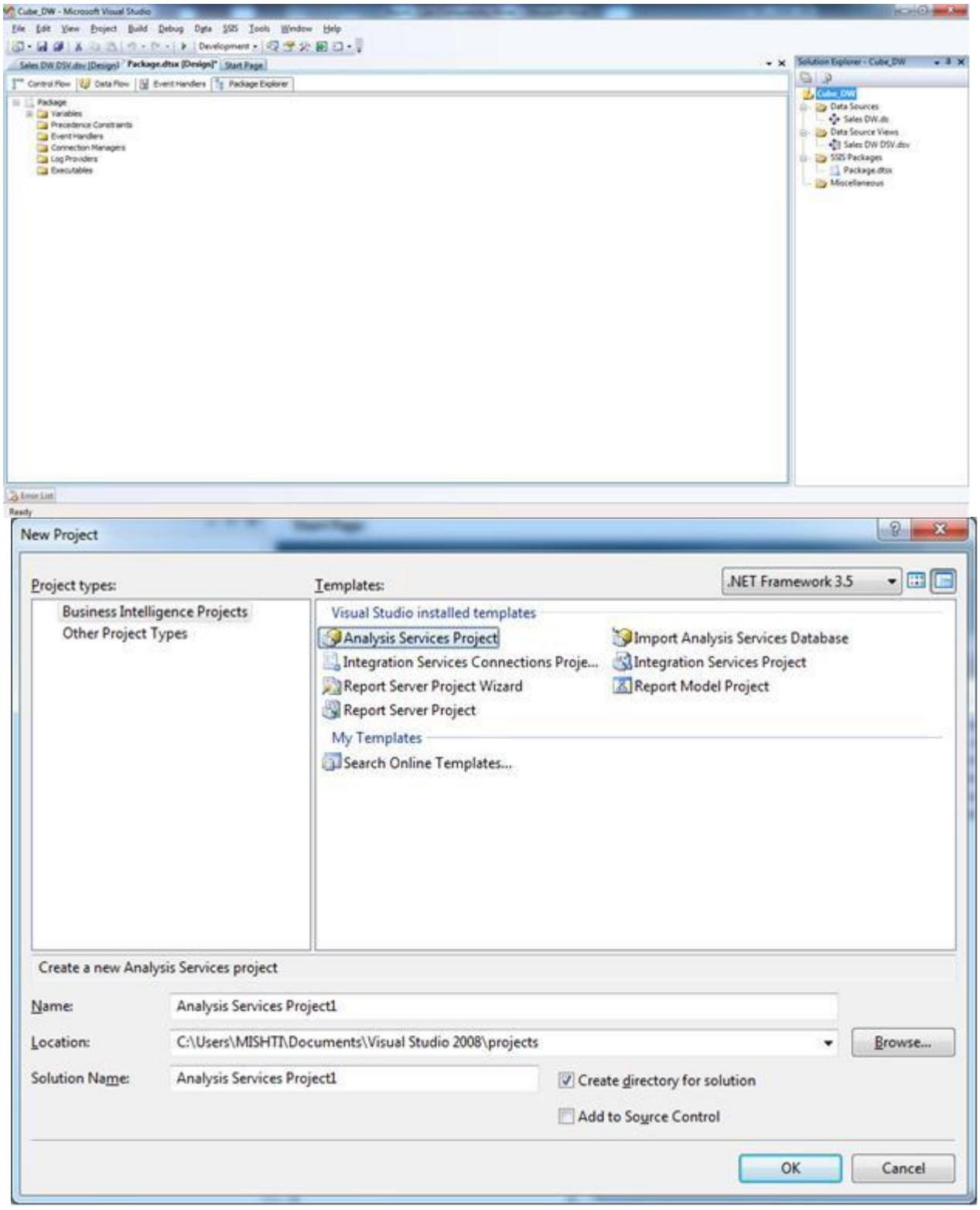
This is where OLAP comes in. The basic idea is to trade off increased storage space now for speed of querying later. OLAP does this by precalculating and storing aggregates. When you identify the data that you want to store in an OLAP database, Analysis Services analyzes it in advance and figures out those daily, weekly, and monthly numbers and stores them away (and stores many other aggregations at the same time). This takes up plenty of disk space, but it means that when you want to explore the data you can do so quickly.

OLAP Cube Creation

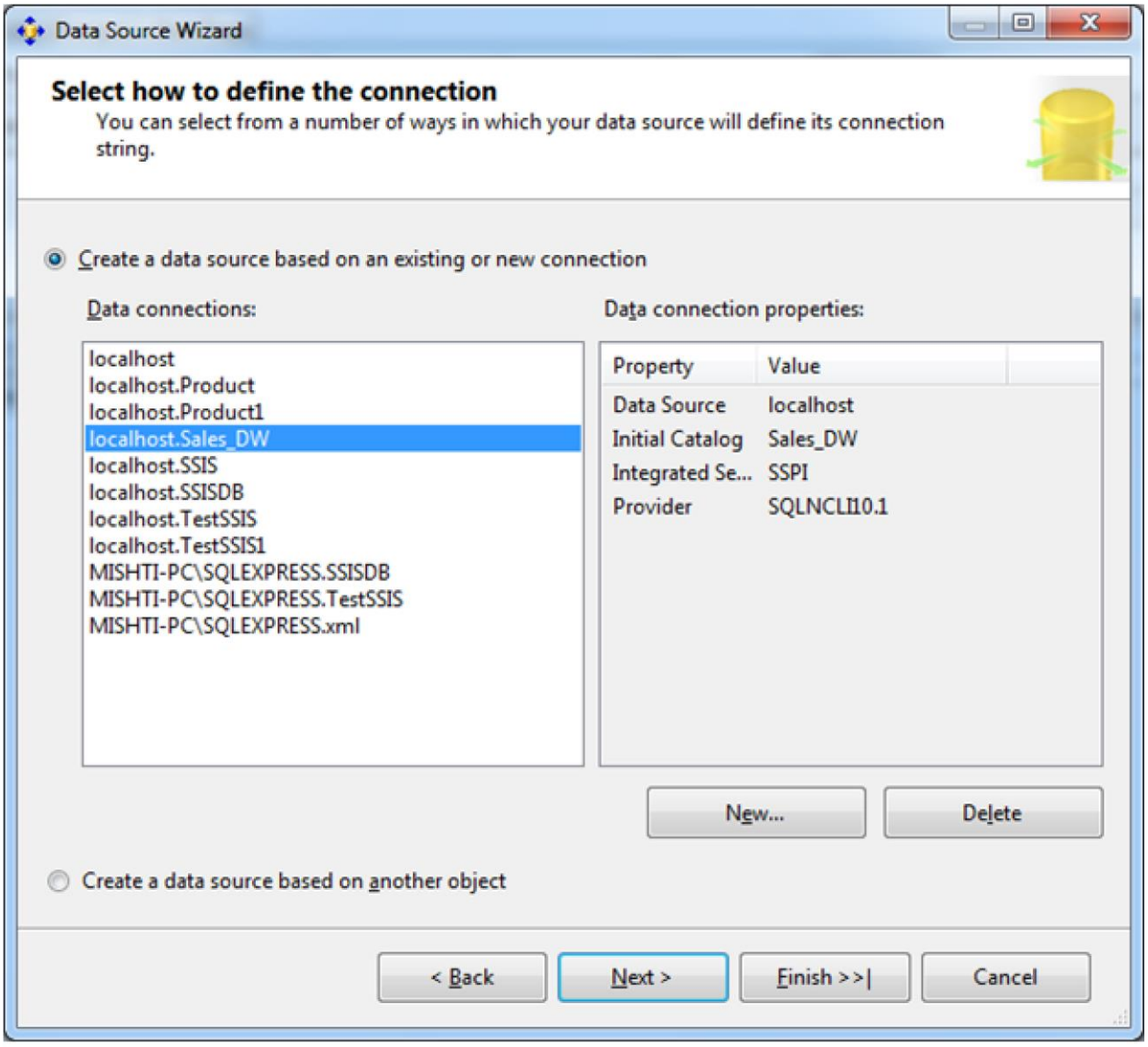
1. First, create a data warehouse in Microsoft SQL Server studio. For instance, following is a sample Sales data warehouse –



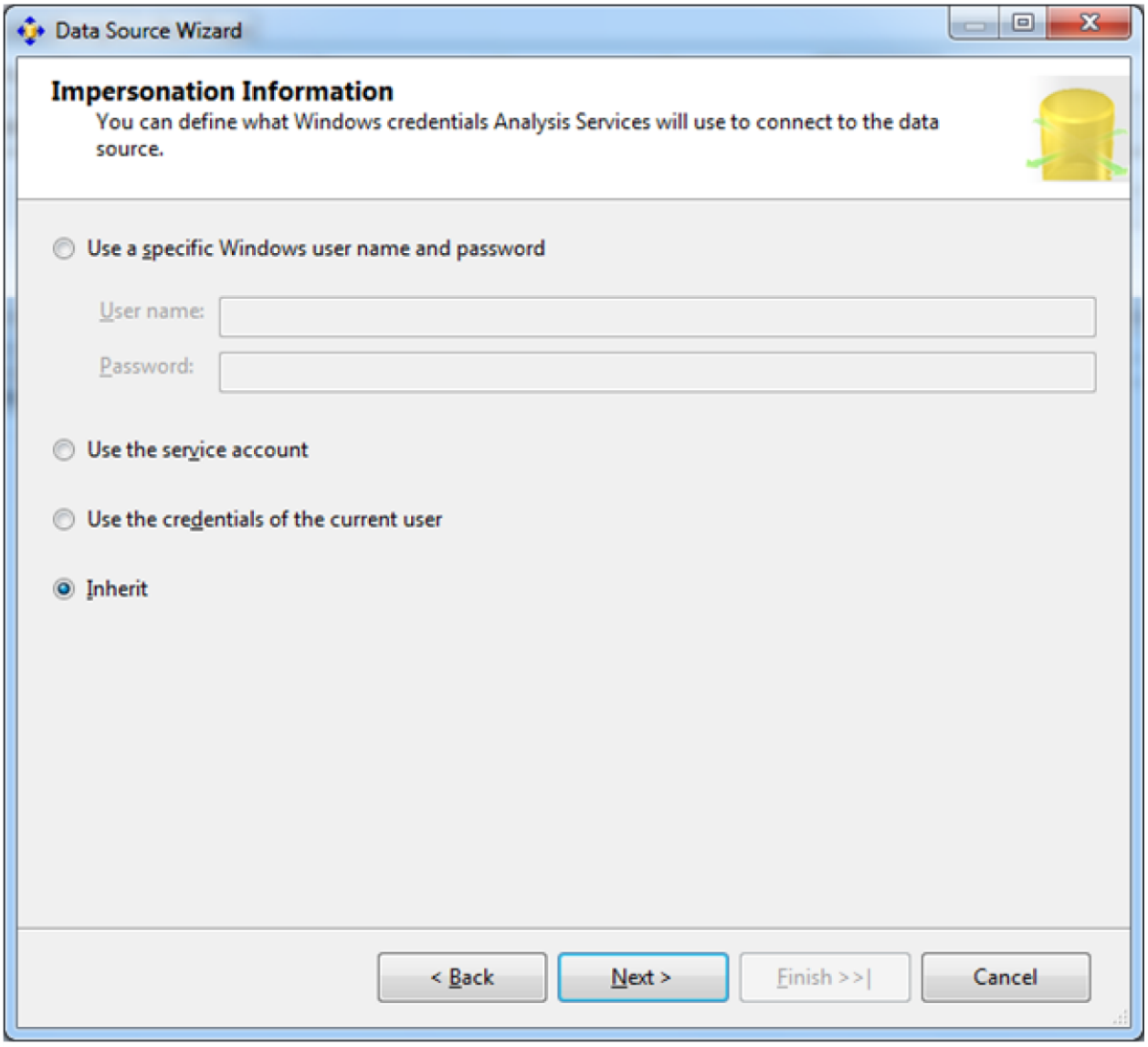
2. Create **new analysis service project** in Microsoft Business Intelligence Development Studio –



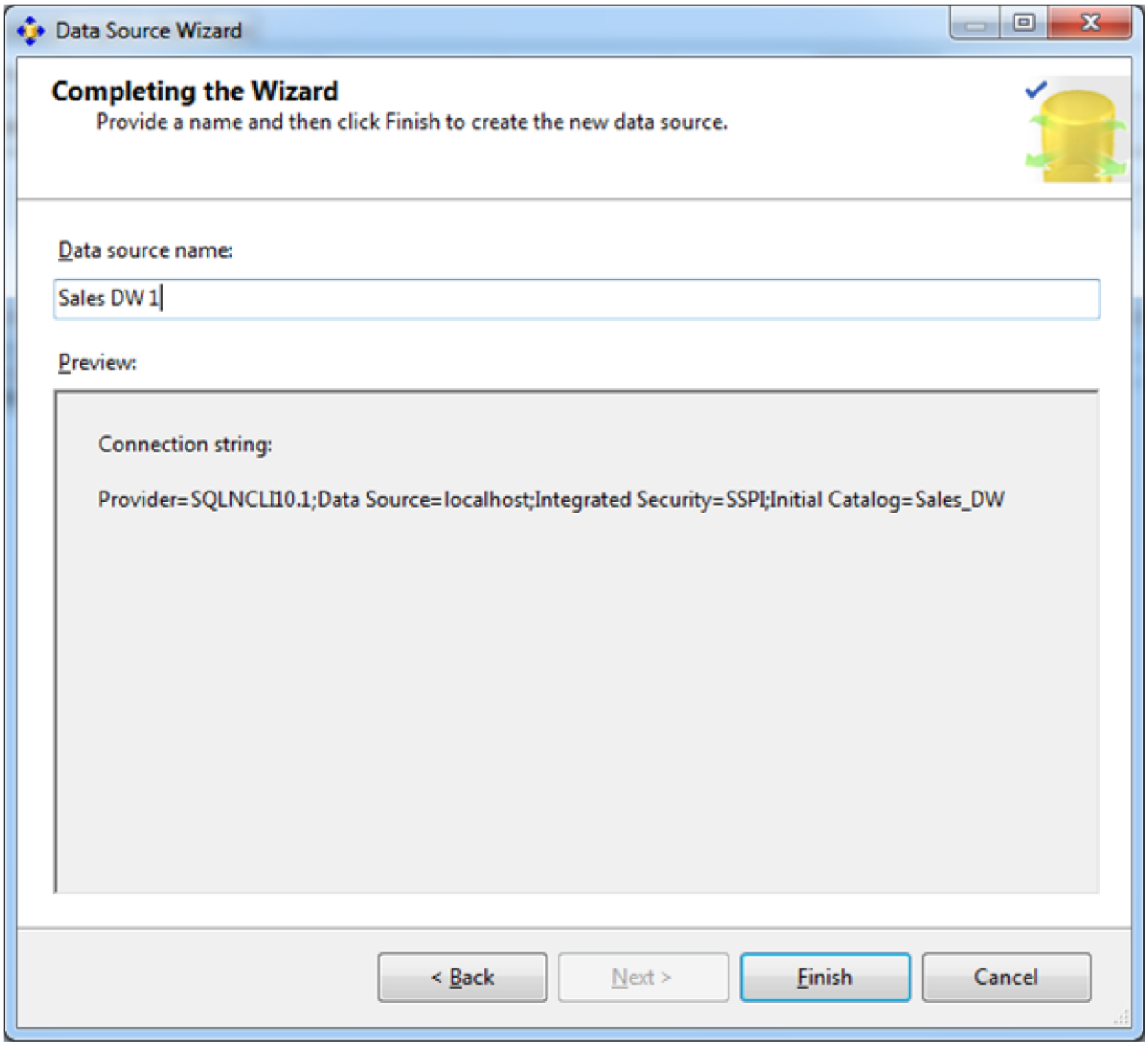
3. Create new data source by right-click on **Data Sources** in Solution Explorer-



Now chose available connections or create new connection and click on next button-

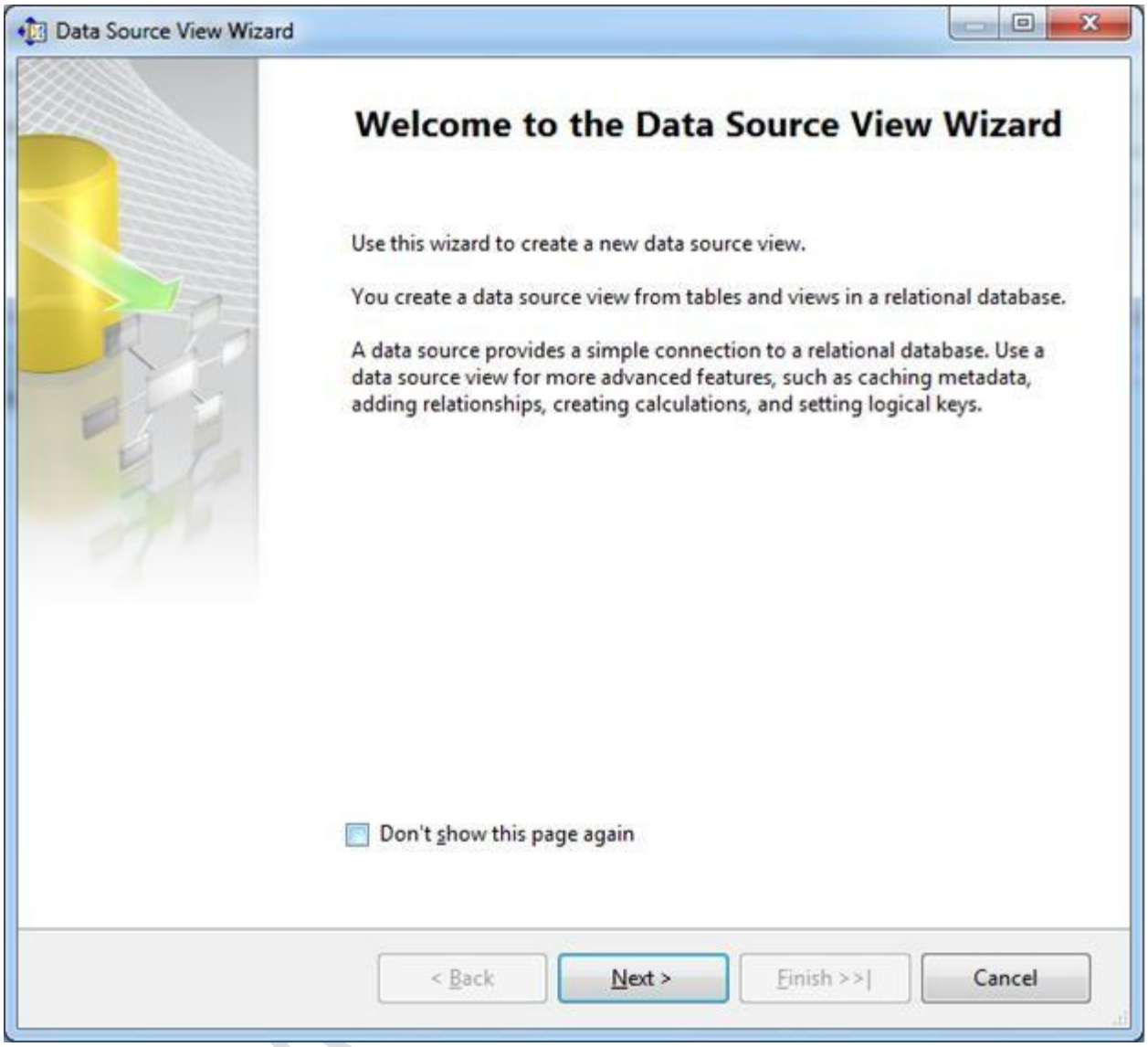


Select **Inherit** option and click on Next button-

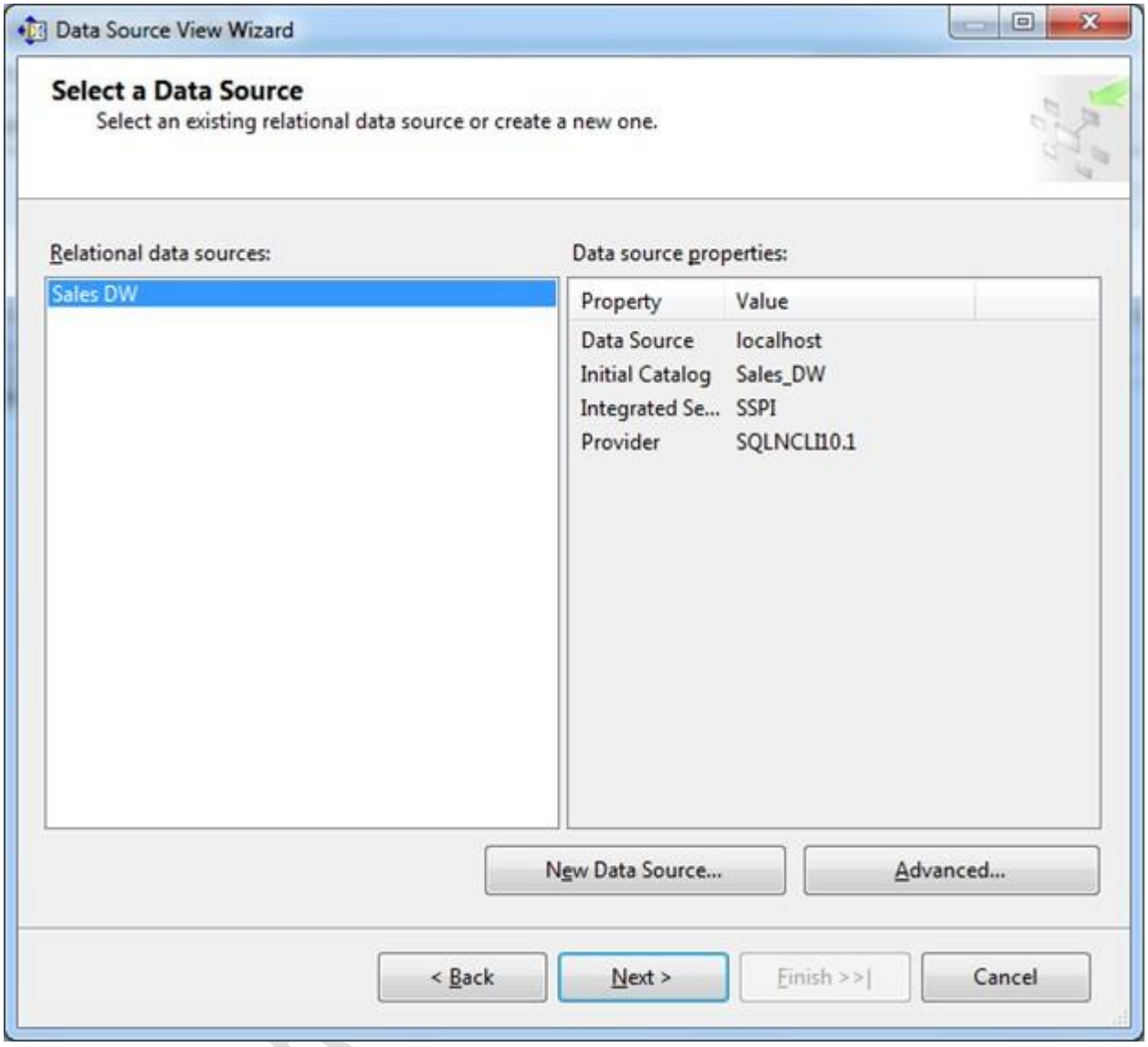


Click on Finish button. The data source will be created.

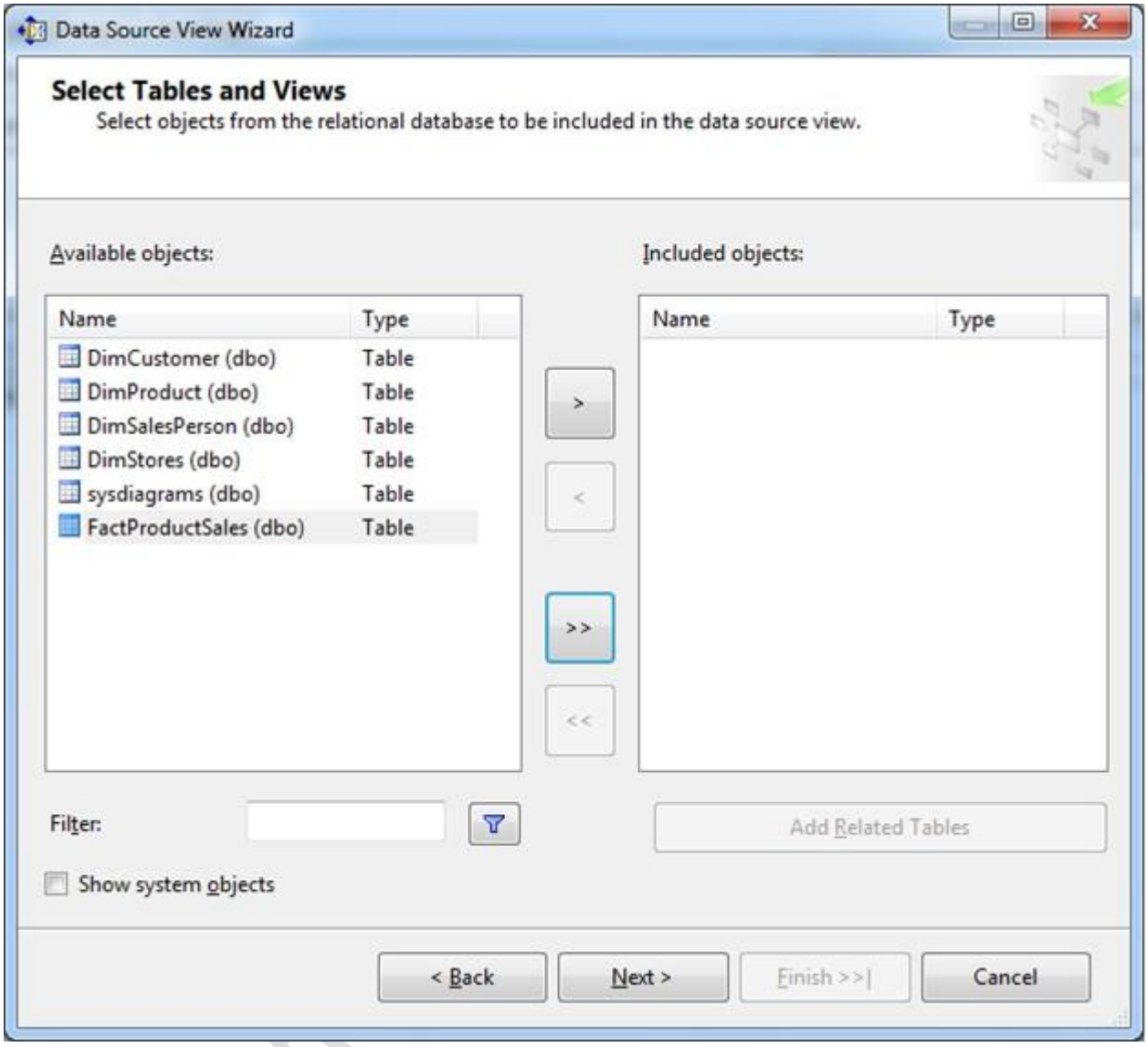
4. Create new data source view by right click on Data Source Views in Solution Explorer-



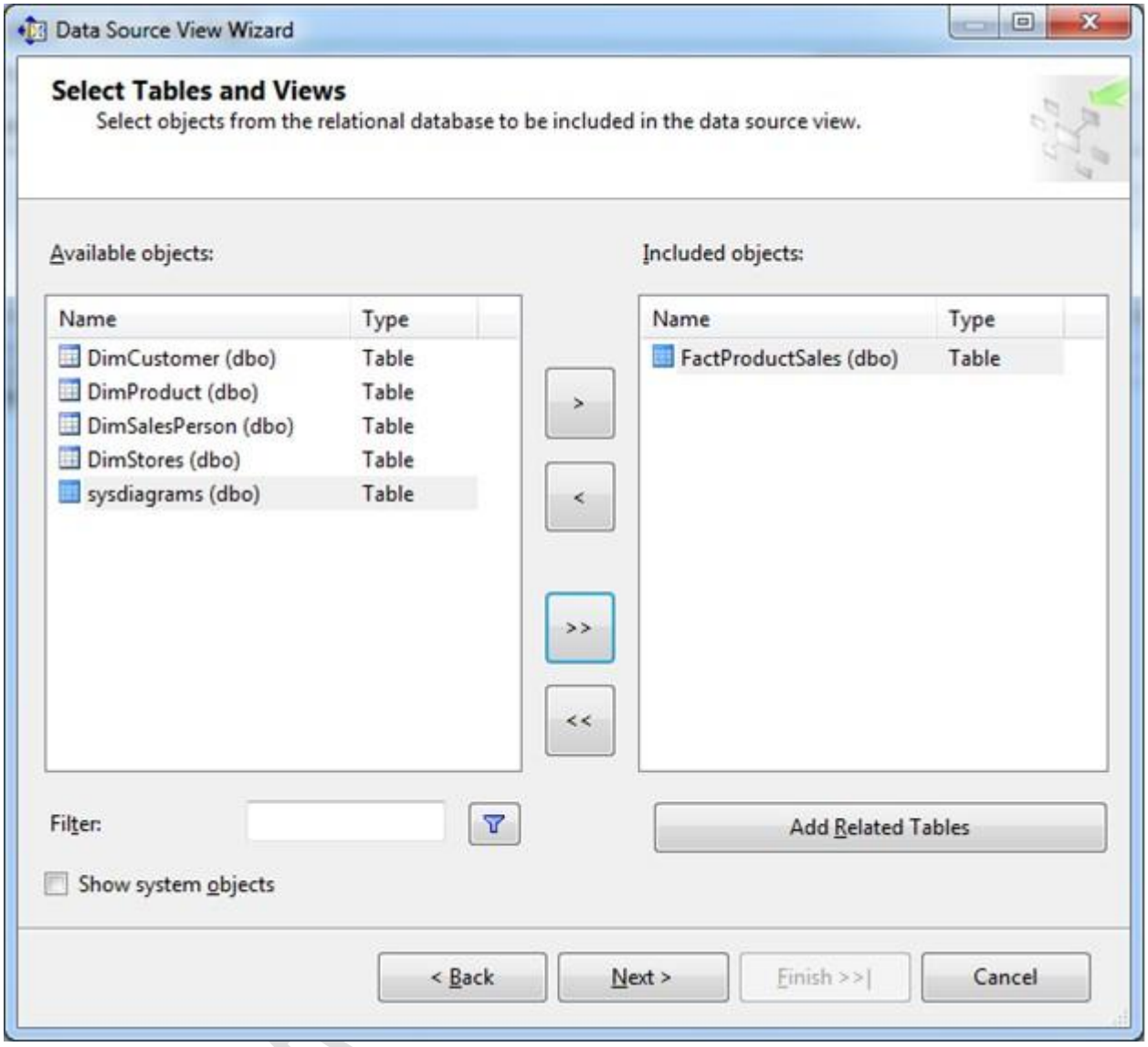
Click on Next button-



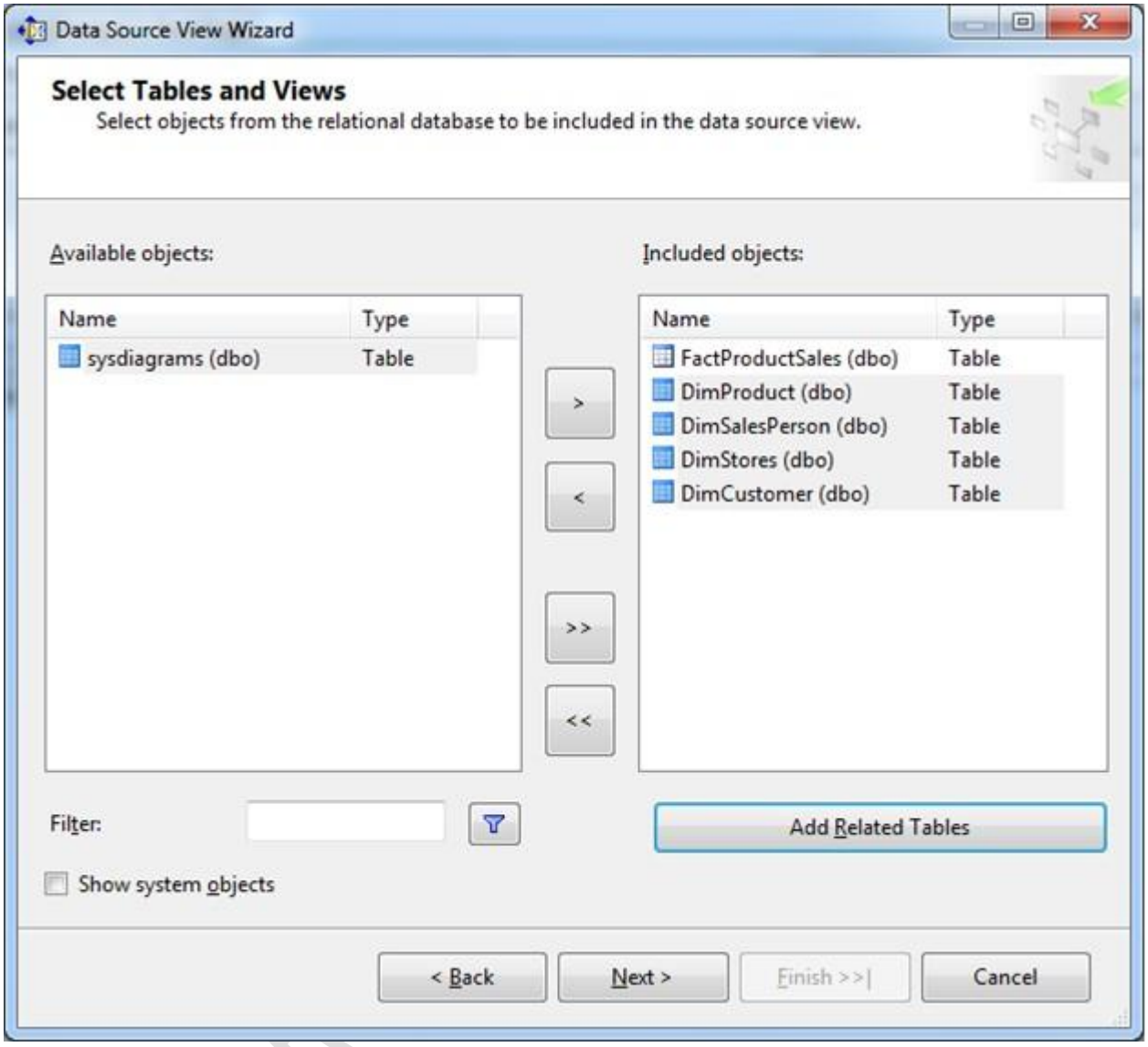
Select data source and click on Next button-



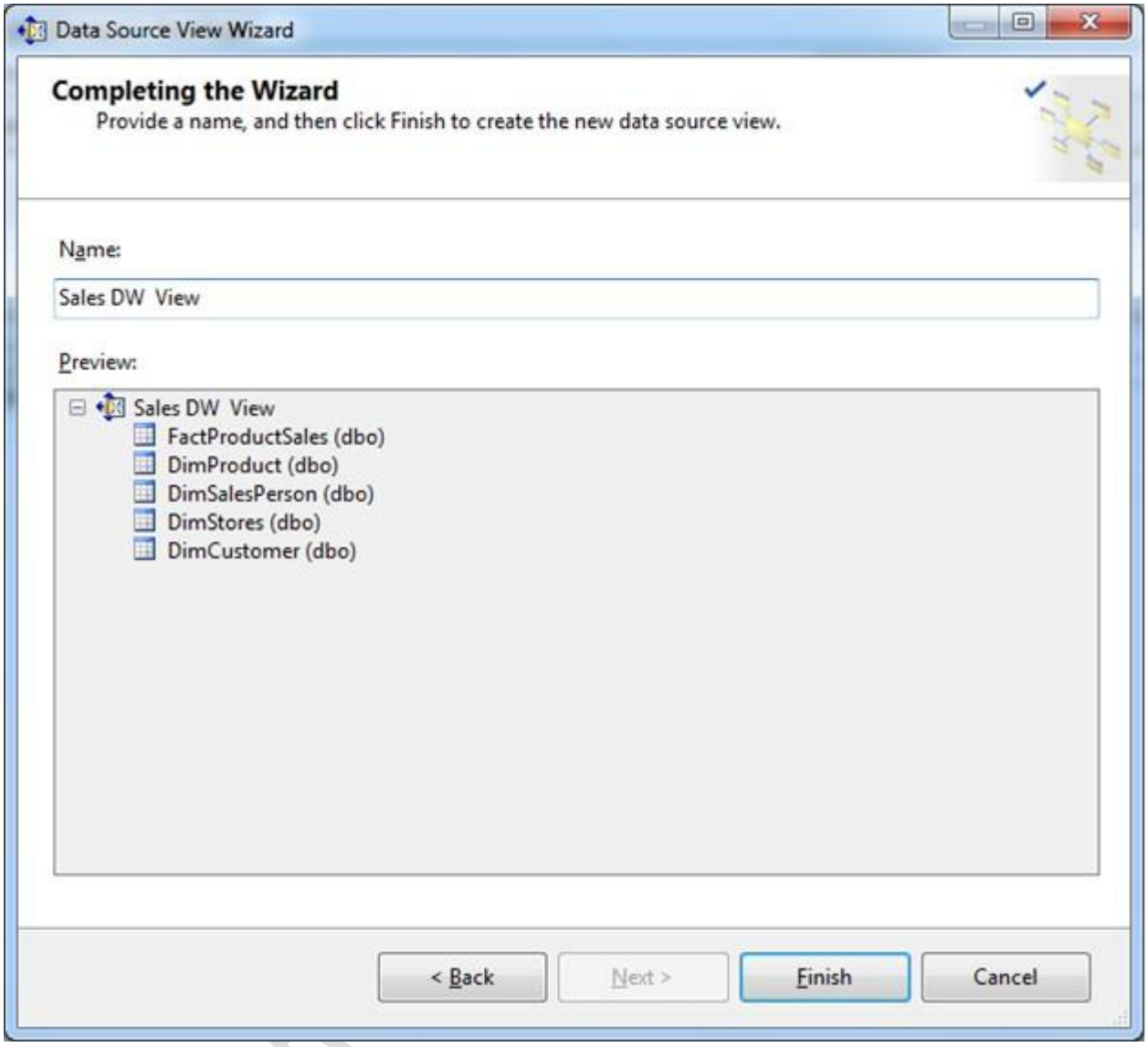
Move Fact table on right pane-



Click on Add Related Tables button-

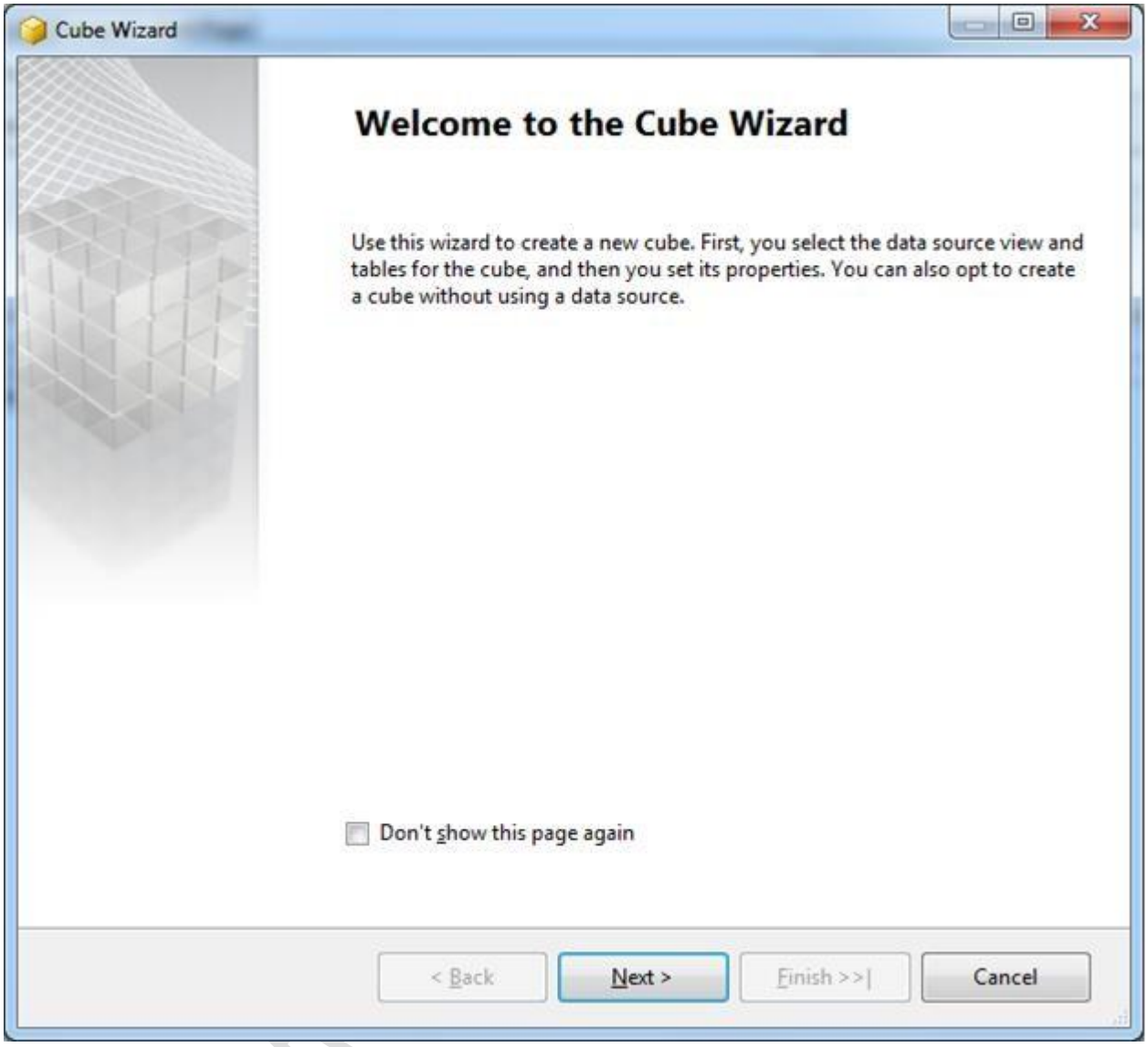


Click on Next button-

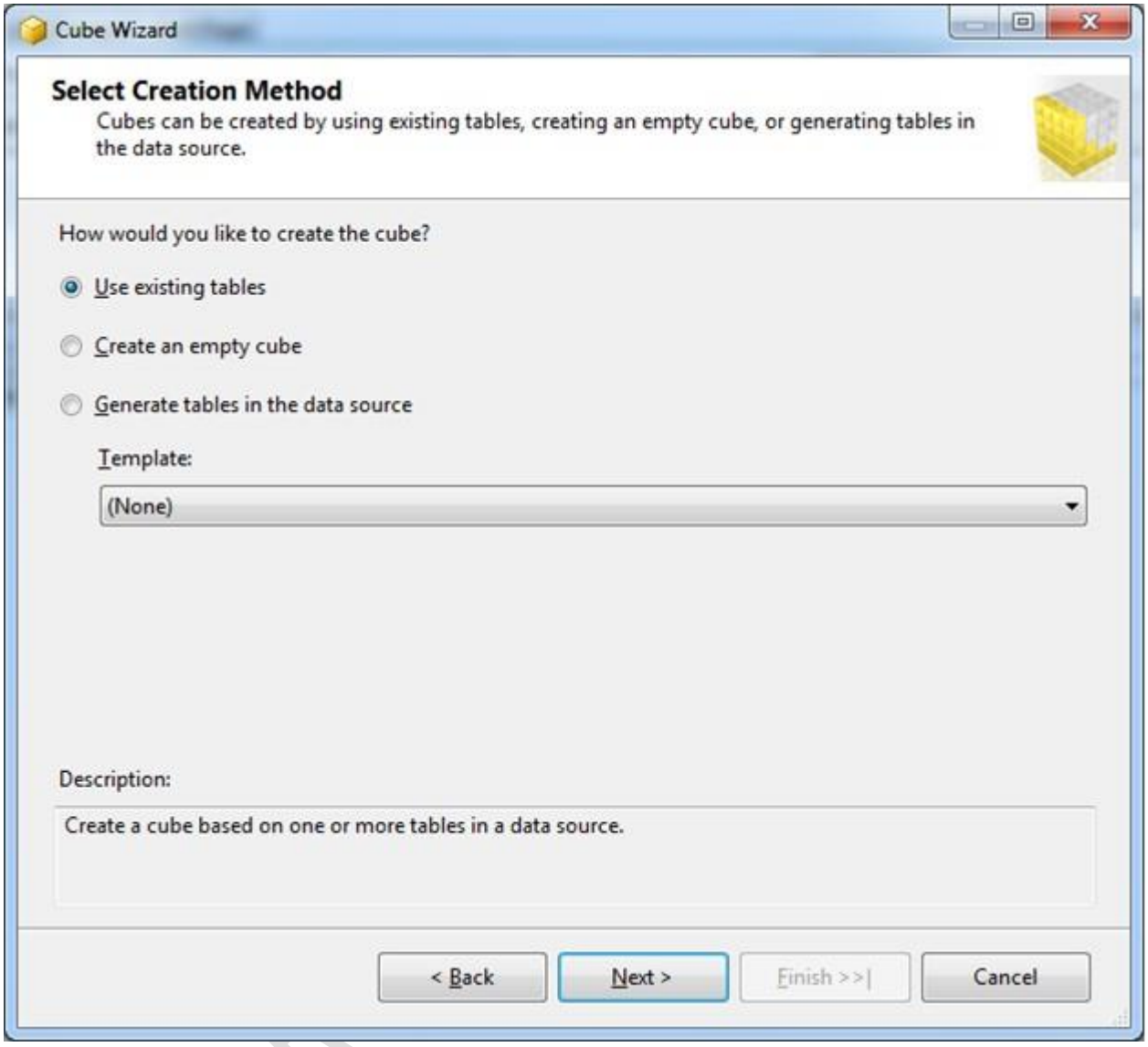


Enter data source view name and click on Finish button, the data source view will be created.

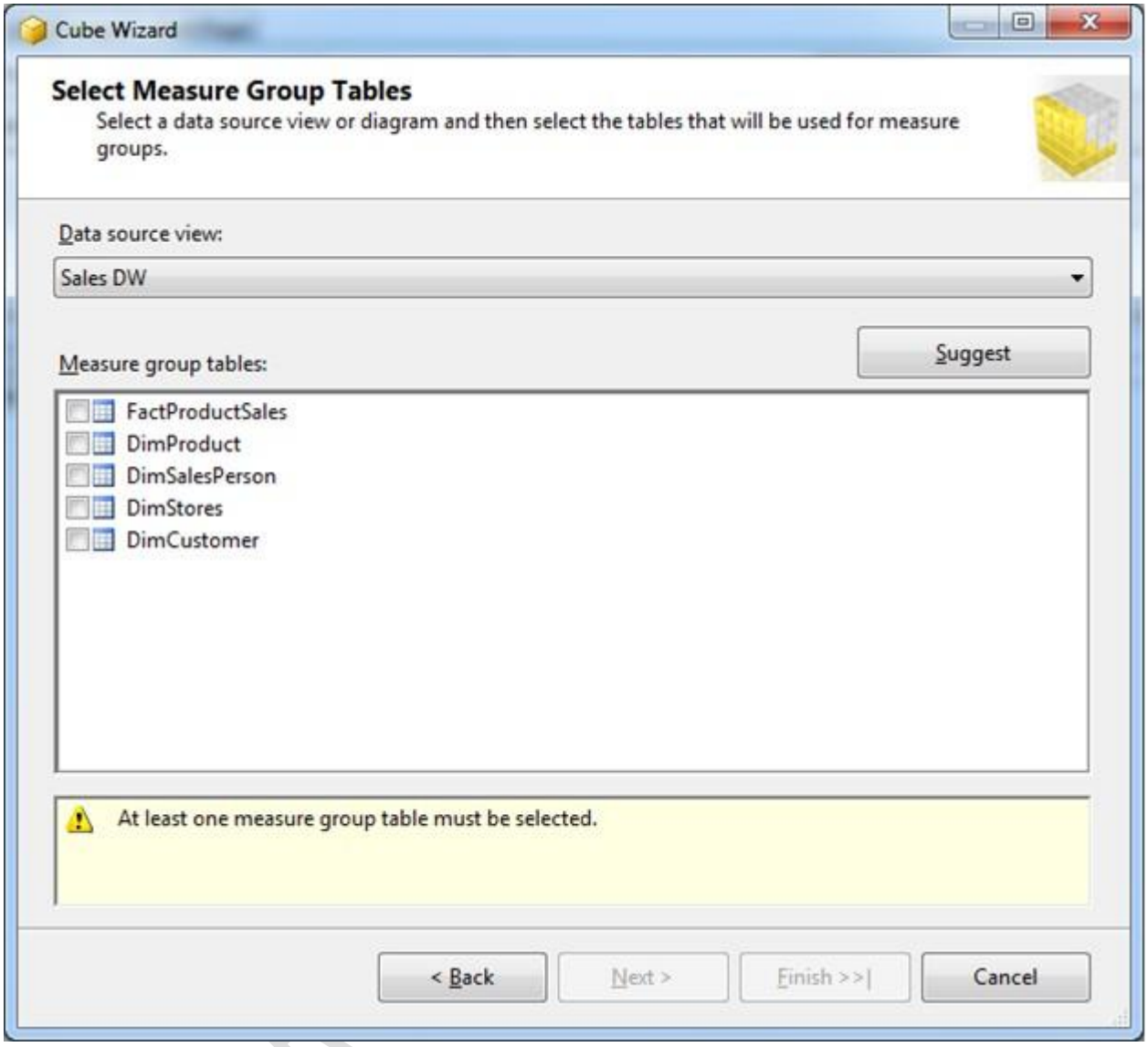
5. Create new cube by right click on Cubes in Solution Explorer-



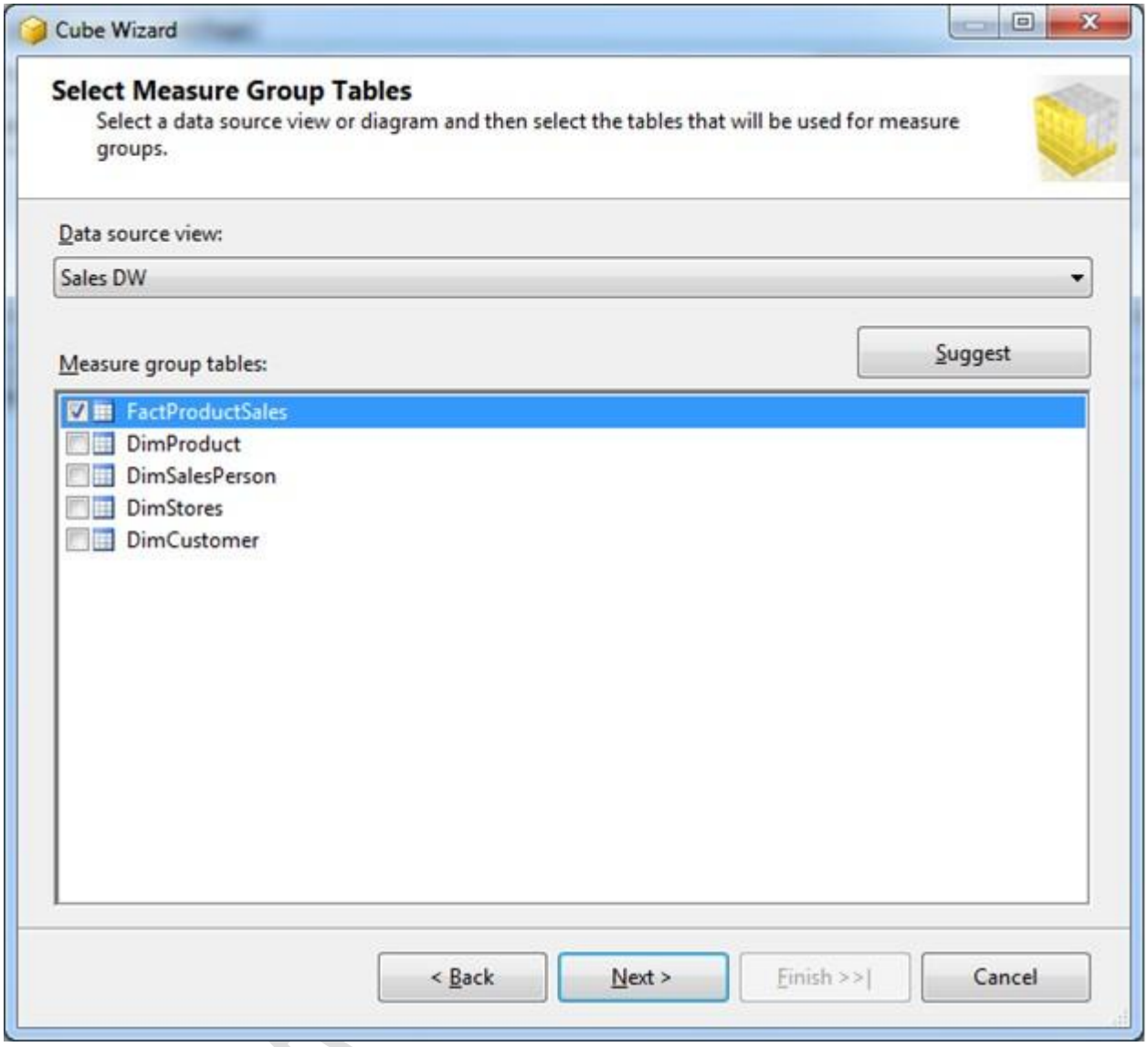
Click on Next button-



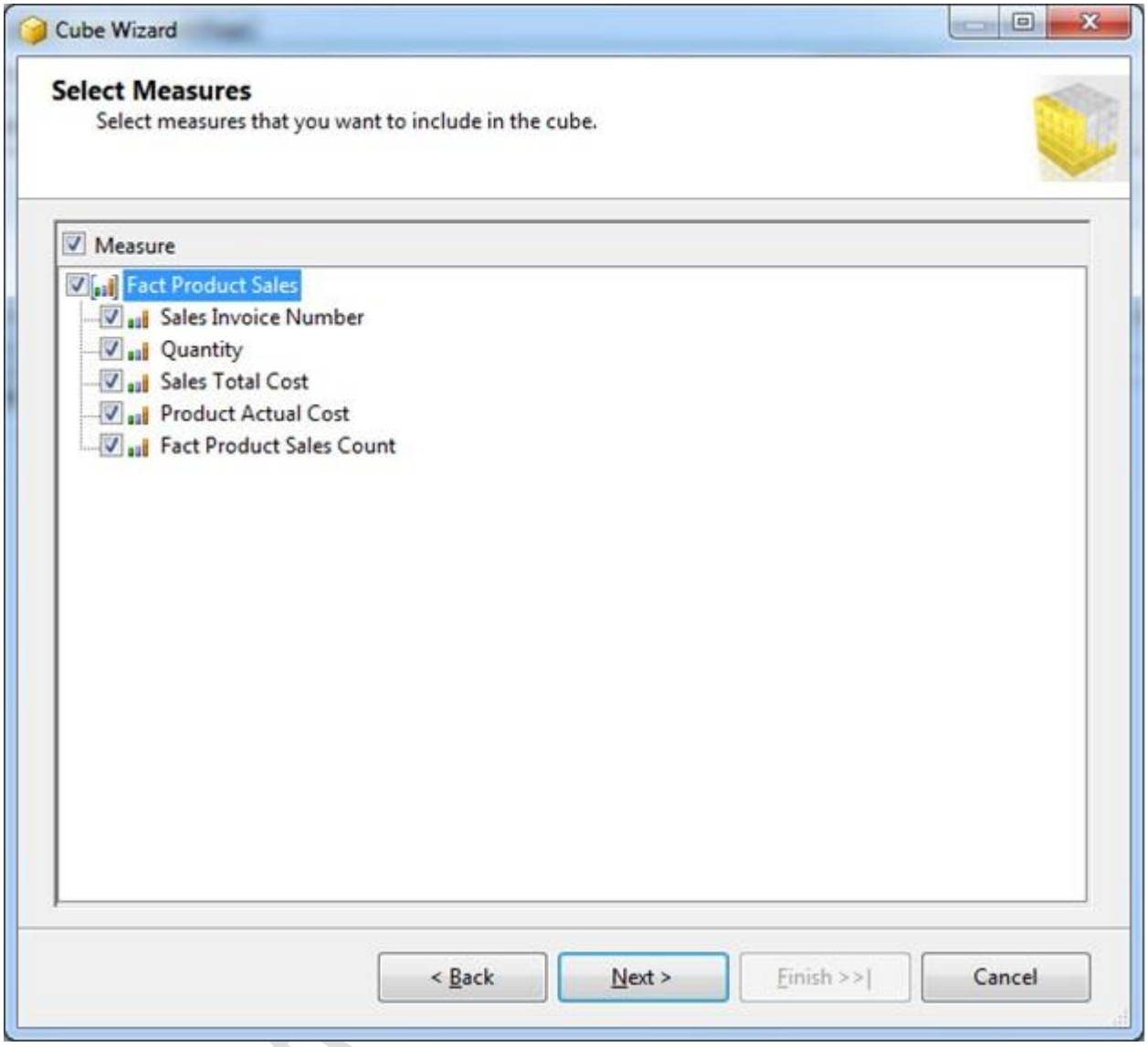
Click on Next button-



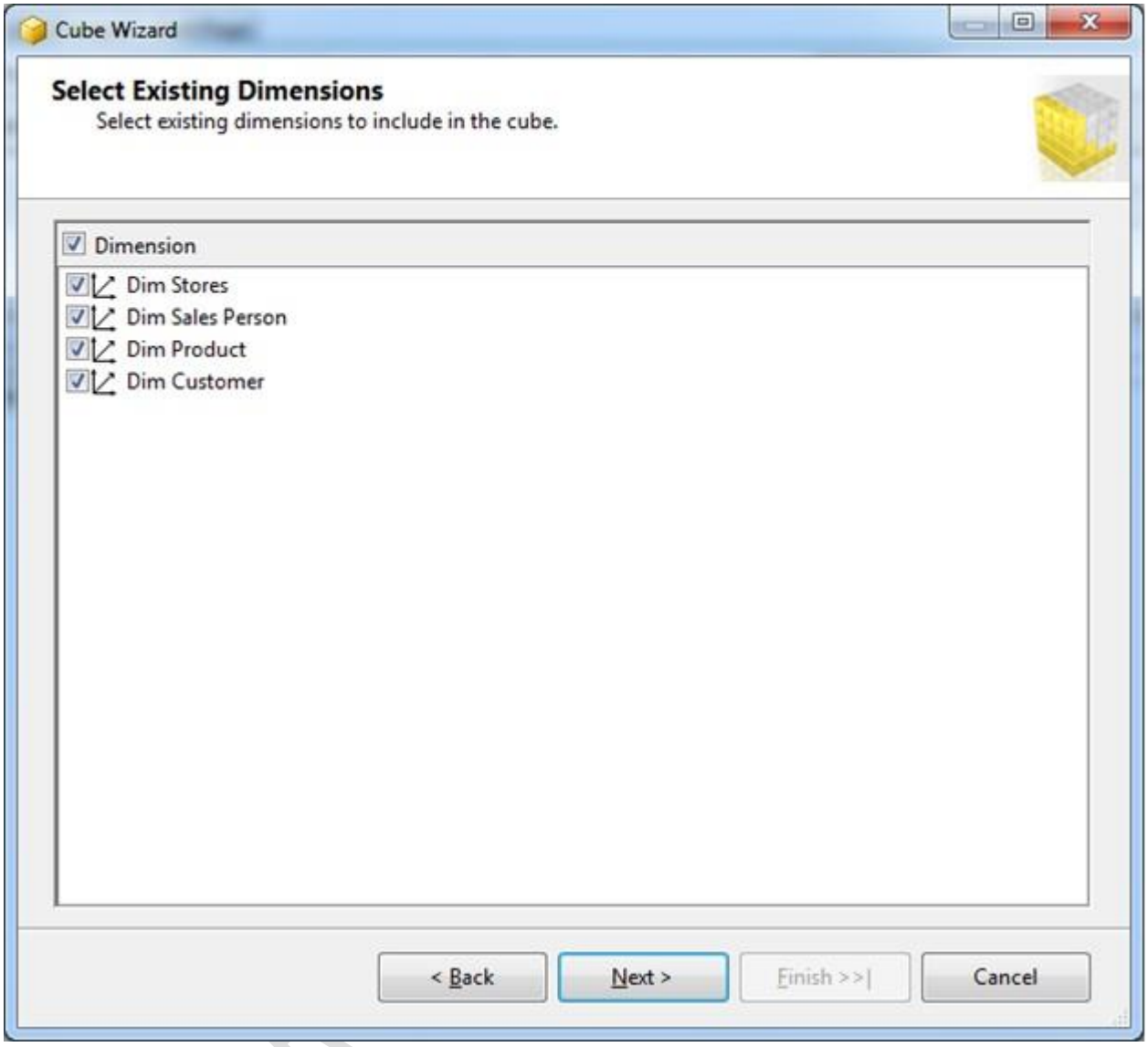
Select Fact table and click on Next button-



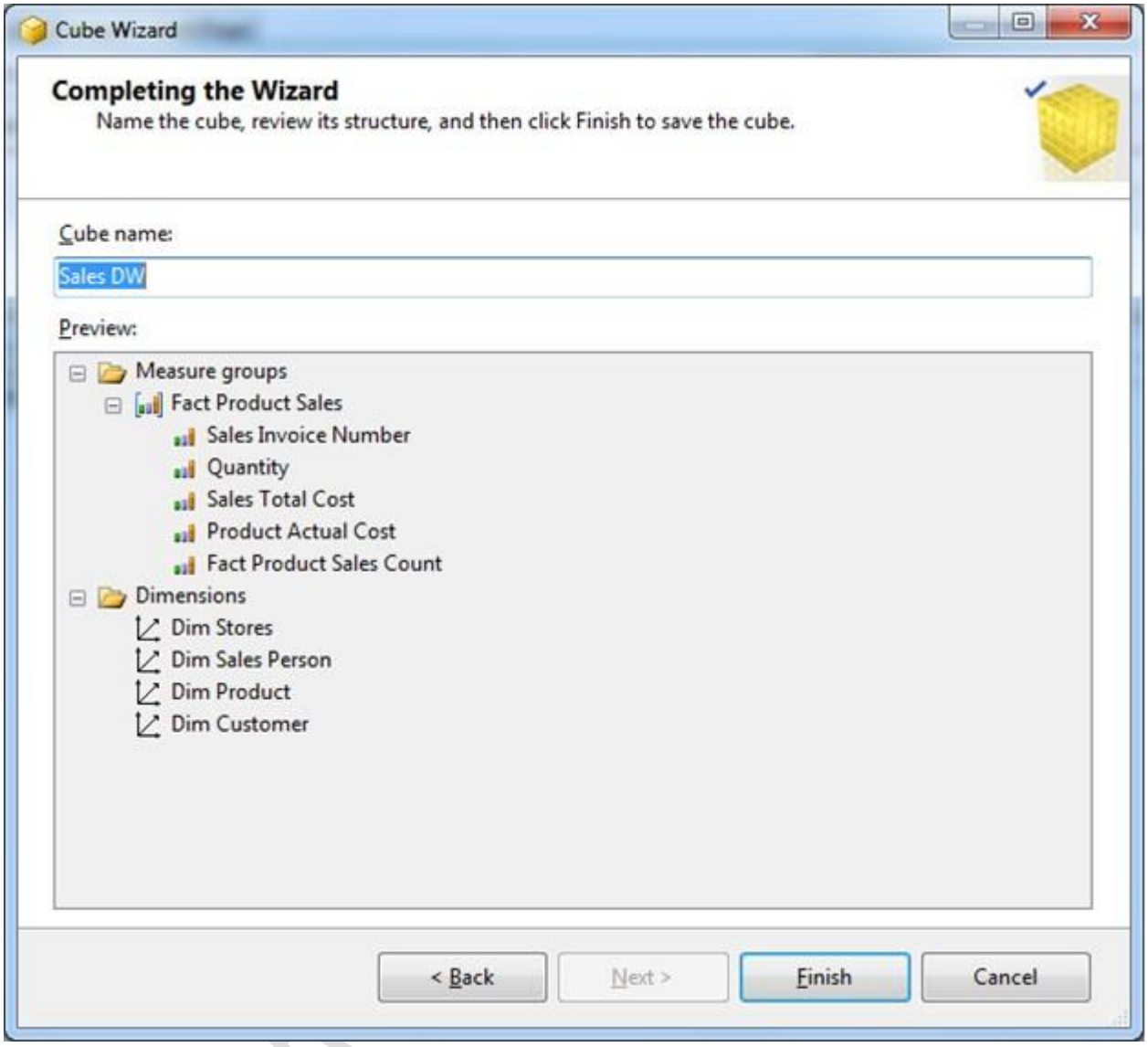
Select measures for fact table which you want-



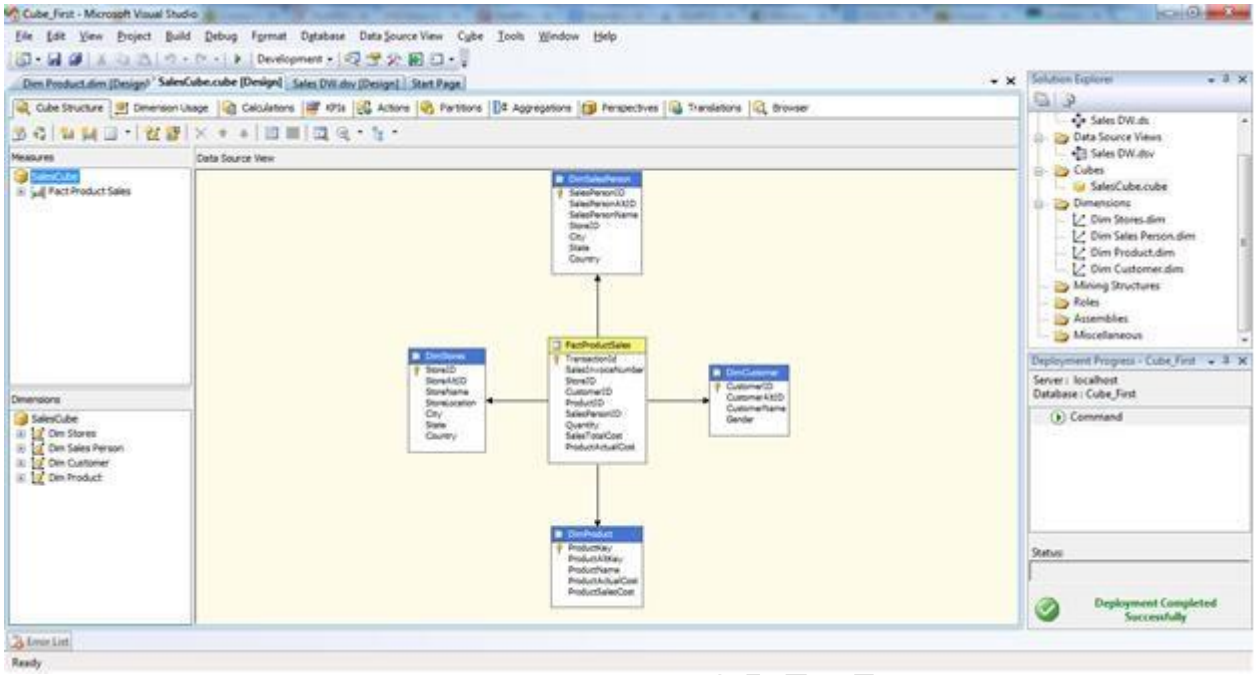
Click on Next button-



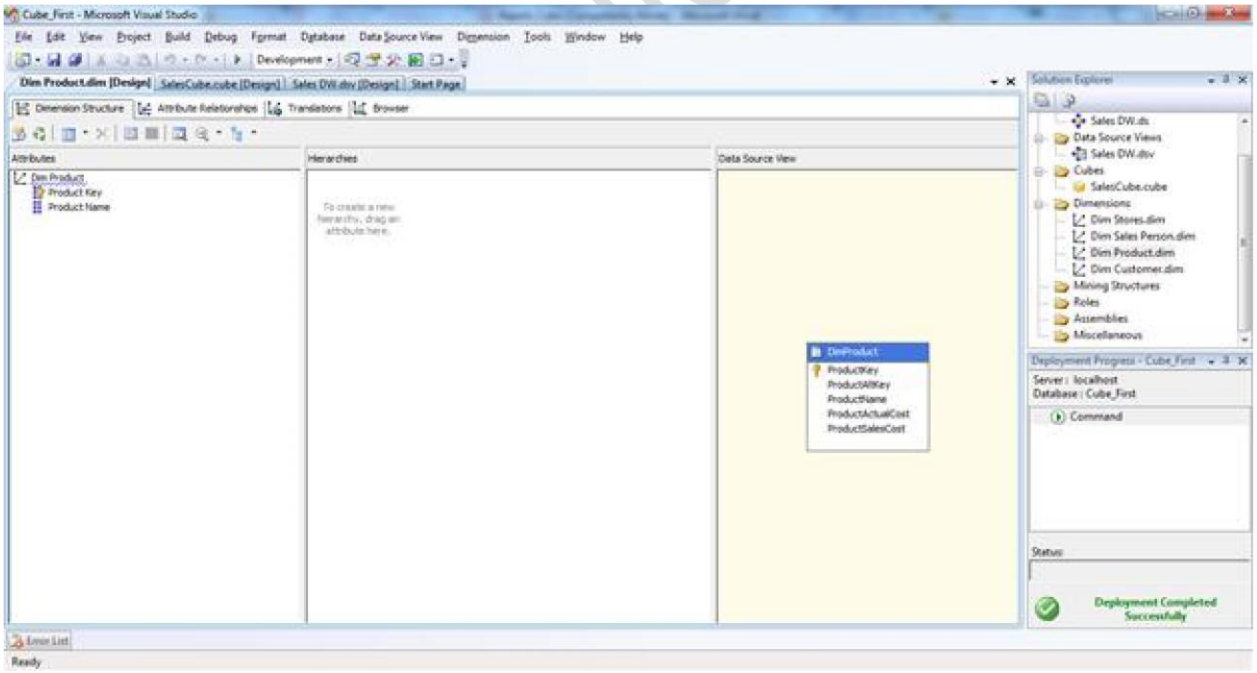
Select Dimension tables and click on Next-



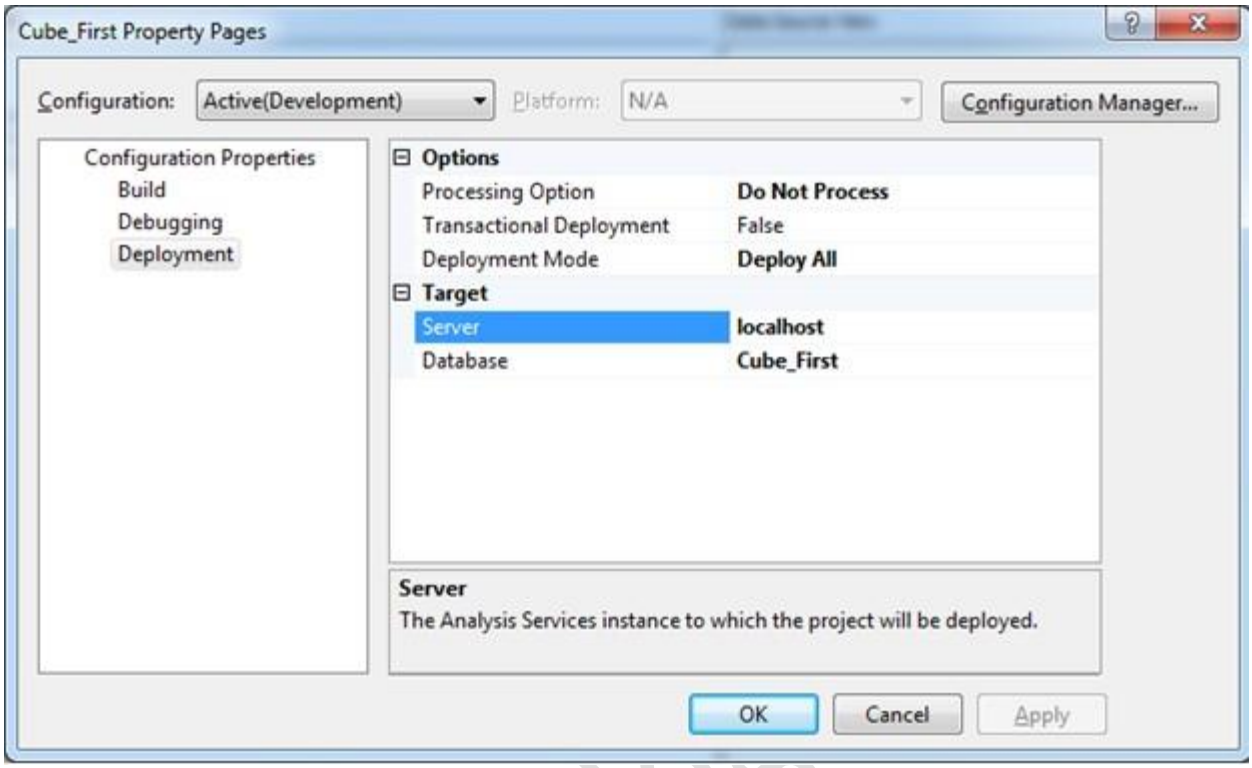
Enter cube name and click on Finish button. The cube will be created like this-



6. Now modify dimensions for queries. In Solution Explorer, double click on dimension **Dim Product ->** Drag and Drop Product Name from Table in Data Source View and Add in Attribute Pane at left side.

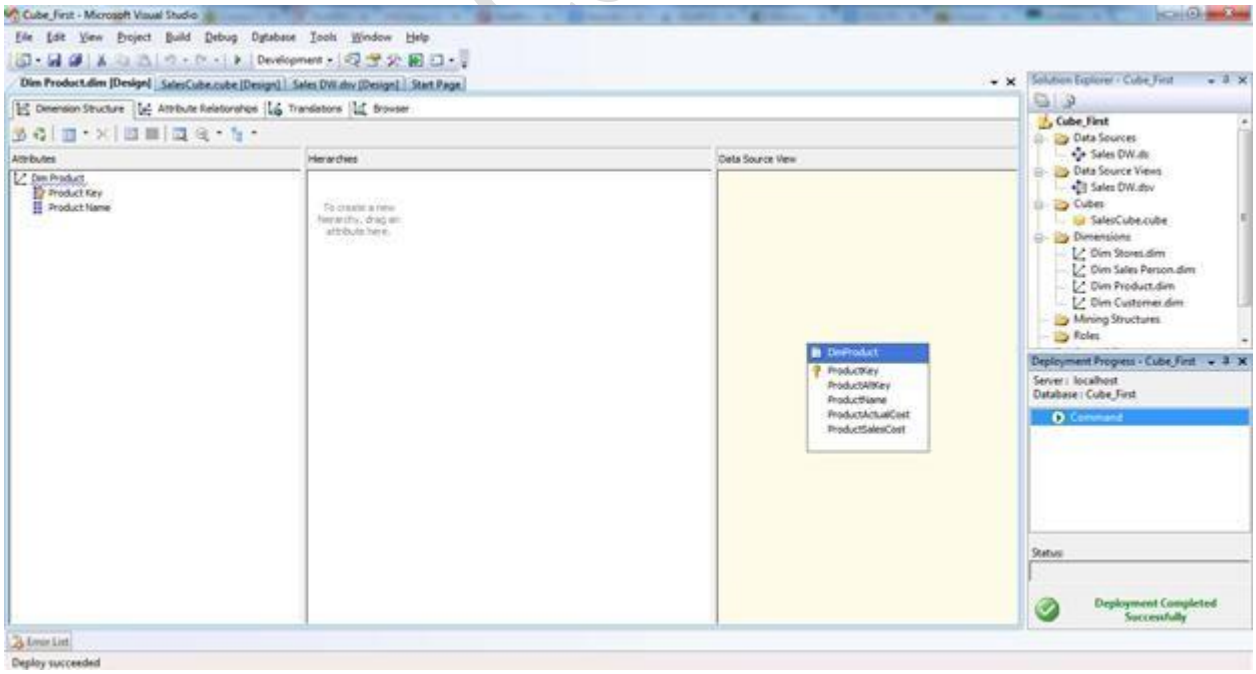


7. Now deploy the project. First right click on project name in solution explorer and click on properties –

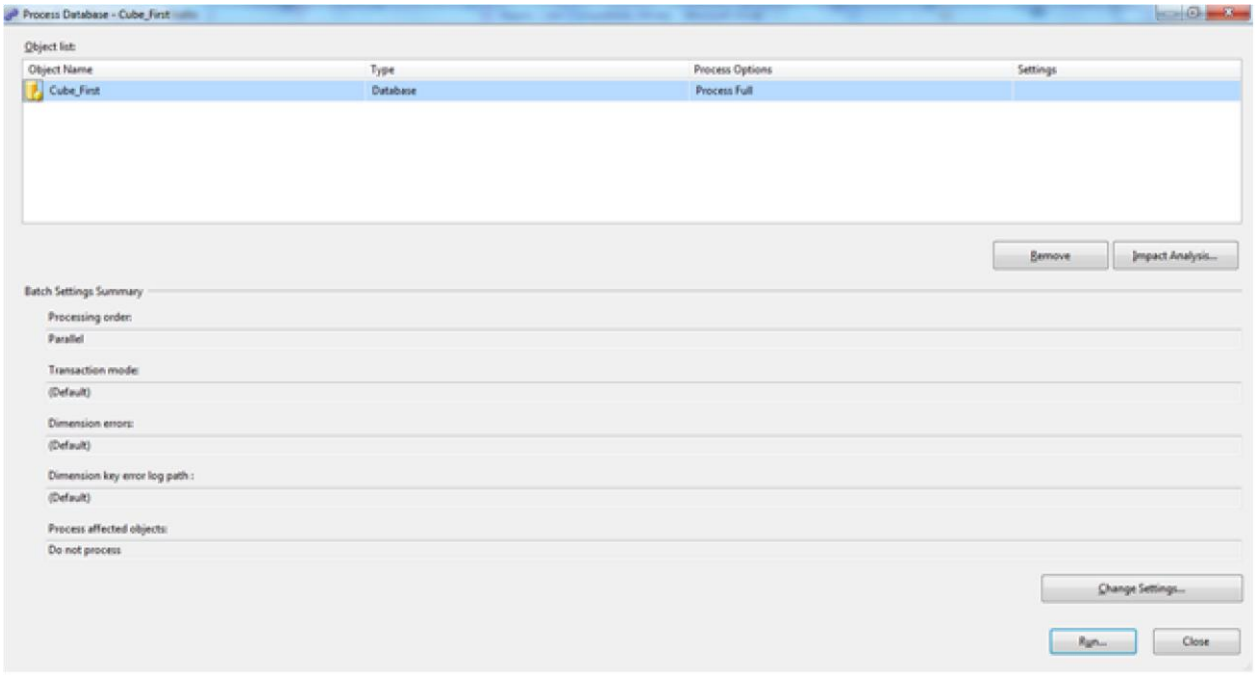


Now click on **OK** button

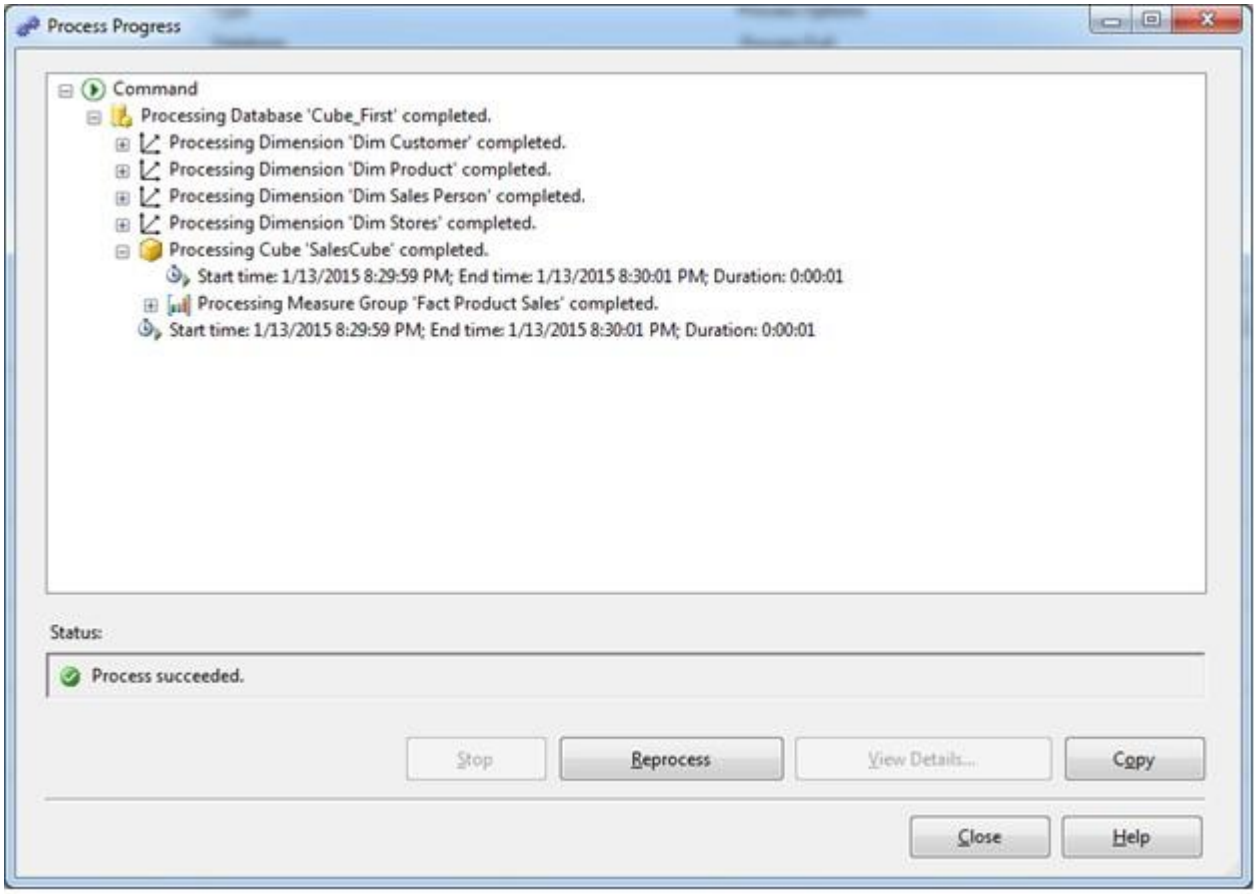
8. Now right click on project name in solution explorer and click on deploy; the project will deploy and the message will be displayed –



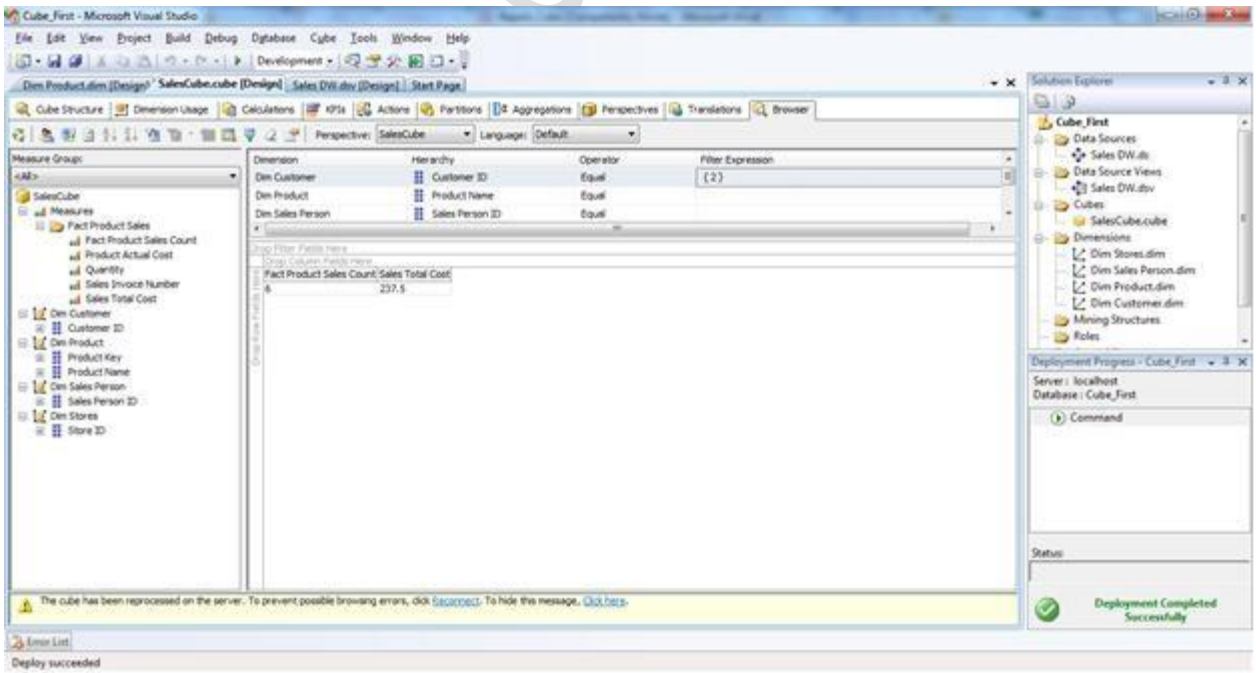
9. Now right click on project name in solution explorer and click on process –



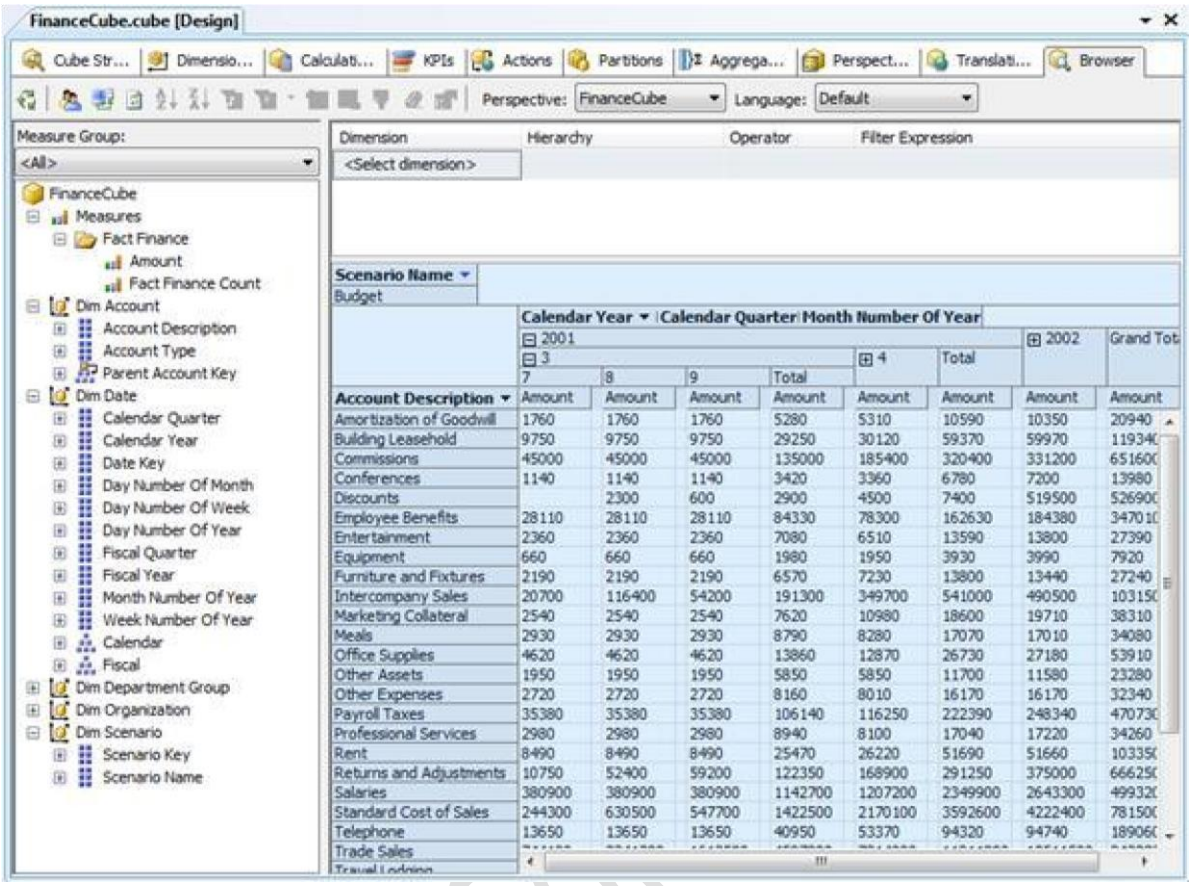
10. Now click on Run button and process has been completed –



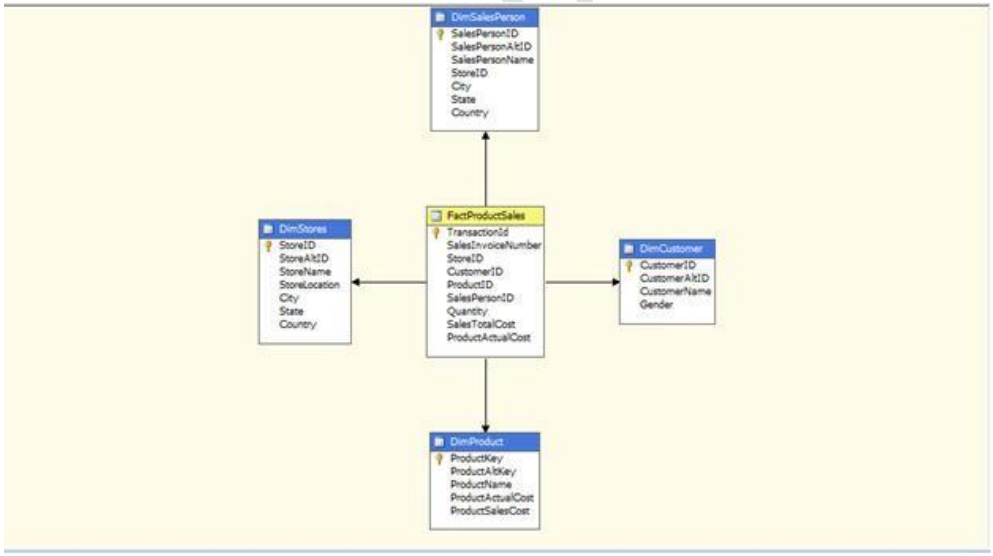
11. Now right click on Cube and click on Browse–



The Cube Browser displays data in following manner



12. Add dimensions and fact fields like above image. The cube is ready to get quick results like below-



Data Warehousing Architecture

Data Warehouse builds by integrating large amount of data from multiple heterogeneous or homogeneous sources. These integrated data needs to be cleaned and transformed and store it in a

structure that is easy to access, understand and use. These data is then used for query, reporting and data analysis. The access, usage, technology and performance requirement are different from those in OLTP environment. For this, data warehouse must have an architecture that allows gathering, organization, manipulation and presentation of data quickly and efficiently. The generic model of Data warehouse is shown in following figure.

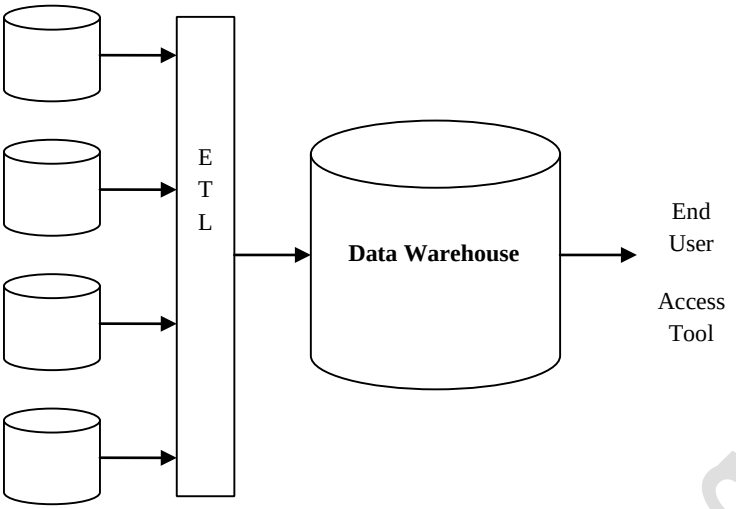


Figure. Generic model for Data Warehouse

To build a data warehouse, Inmon proposed architecture which follows top-down approach and Kimball proposed architecture which follows bottom-up approach to build data warehouse. Hybrid approach is also being used to build data warehouse which is a combination of top-down and bottom-up approach. Inmon's top-down architecture collects all the data from data source systems and store it to the data warehouse. Then, if necessary, it builds data marts.

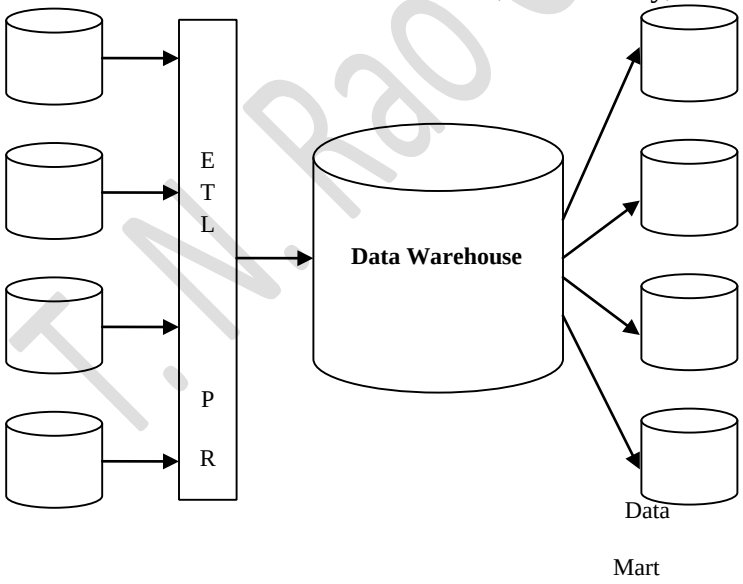


Figure. Top-Down approach architectural structure

Kimball's bottom-up architecture starts with building data marts from individual department's data. Then, it uses these individual data marts to build the data warehouse.

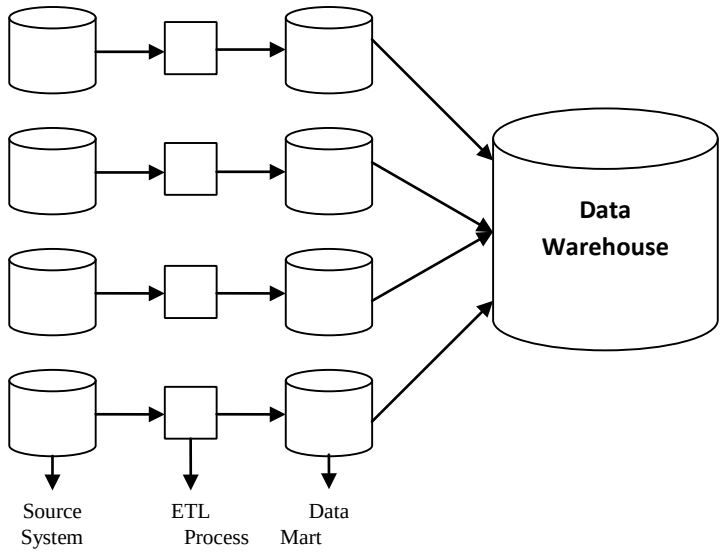


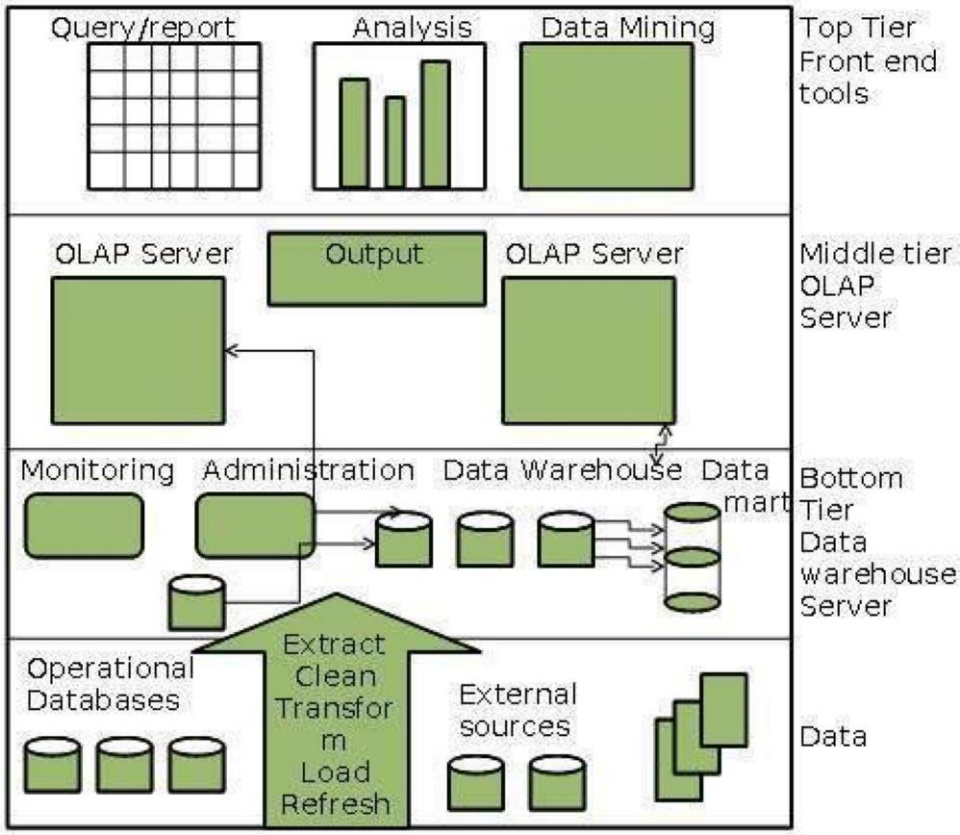
Fig. Bottom-Up approach architecture

Three-Tier Data Warehouse Architecture

Generally a data warehouses adopts a three-tier architecture. Following are the three tiers of the data warehouse architecture.

- **Bottom Tier** - The bottom tier of the architecture is the data warehouse database server. It is the relational database system. We use the back end tools and utilities to feed data into the bottom tier. These back end tools and utilities perform the Extract, Clean, Load, and refresh functions.
- **Middle Tier** - In the middle tier, we have the OLAP Server that can be implemented in either of the following ways.
    - By Relational OLAP (ROLAP), which is an extended relational database management system. The ROLAP maps the operations on multidimensional data to standard relational operations.
    - By Multidimensional OLAP (MOLAP) model, which directly implements the multidimensional data and operations.
- **Top-Tier** - This tier is the front-end client layer. This layer holds the query tools and reporting tools, analysis tools and data mining tools.

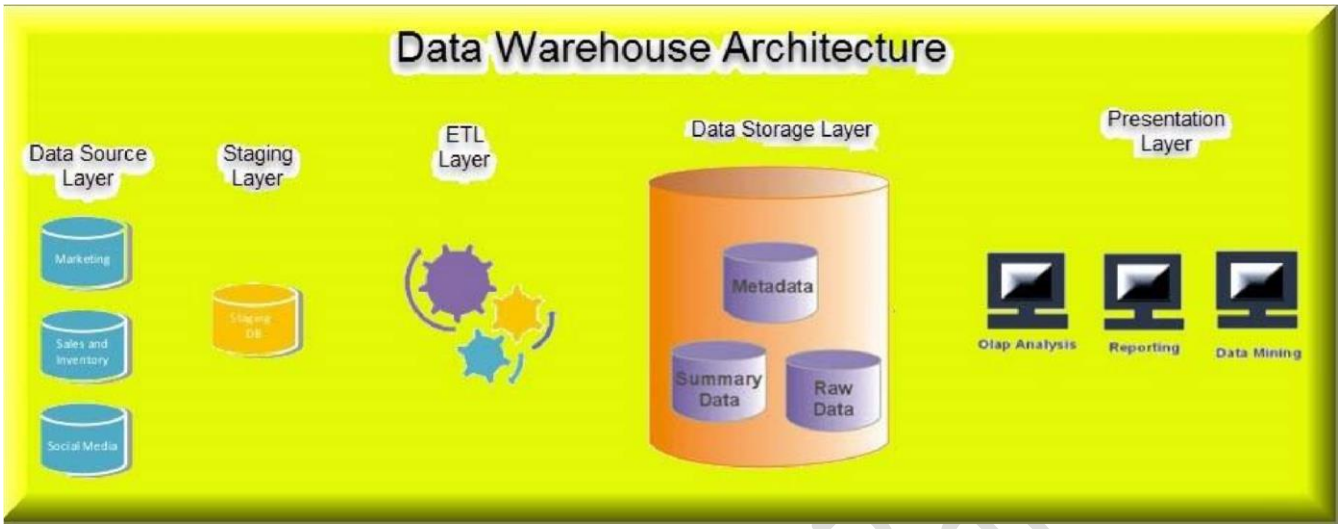
The following diagram depicts the three-tier architecture of data warehouse:



Data Warehouse Architecture

A data warehouse is the defacto source of business truth developed by combining data from multiple disparate sources. It supports analytical reporting, and both structured and ad hoc queries. Data warehousing systems, like home designs, have many different architectural options. Some have Operational Data Stores (ODS), others are deployed with data marts. Some data warehouse may reference finite set of source data, or as with most enterprise data warehouses, reference a variety of internal and external data sources. Because constructing a data warehouse is unique to the business use, we will look at the common layers found in all data warehouse architecture. All data warehouse architecture includes the following layers:

- Data Source Layer
- Data Staging Layer
- Data Storage Layer
- Data Presentation Layer



Data source layer

The data source layer of data warehouse architecture is where original data, collected from a variety internal and external sources, resides in the relational database. Examples of source data types include but are not limited to: These stores disparate data types including:

- Operational Data—Product data, inventory data, marketing data, or HR data.
- Social Media Data—Web site hits, content popularity, contact page completion. Third-party data—Demographic data, survey data, census data.

While most data warehouse architecture deals with structured data, consideration should be given to the future use of unstructured data sources, such as voice recordings, scanned images, and unstructured text. These streams of data are valuable silos of information and should be considered when developing your data warehouse.

Data Staging Layer

The data staging layer resides between data sources and the data warehouse. In this layer, data is extracted from different internal and external data sources. Because source data comes in many different formats, the data extraction layer will utilize multiple technologies and tools to extract the required data. Once the extracted has been loaded, it will be subjected to high-level data quality checks. The final result will be clean and organized data that you will load into your data warehouse. The staging layer contains the following components: Landing Database and Staging Area

- Data Integration Tool (ETL)

Landing Database and Staging Area

The landing database stores the data retrieved from the data source. Staging is used to apply quality checks on the data before moving it to the data warehouse. Staging is an essential step in data warehouse architecture. Poor data will amount to inadequate information and result is poor business decision making. The staging layer s also where you want to make adjustments to the schema to handle unstructured data sources.

Data integration tool
Extract, Transform and Load tools (ETL) are the data integration tools used to extract data from source systems, transform and prepare data and load into the data warehouse. Panoply.io product provides this entire process, easily and quickly. All you need to do is point it to your data source(s).

Data Storage Layer
The data storage layer is where data that was cleansed in the staging area is stored as a single central repository. Depending on your business and your data warehouse architecture requirements, your data storage may be a data warehouse, data mart (data warehouse partially replicated for specific departments), or an Operational Data Store (ODS).

Data Presentation Layer
The presentation layer is where users interact with the cleansed and organized. This layer of the data warehouse architecture provides users with the ability to query the data for product or service insights, analyze the information to conduct hypothetical business scenarios, and develop automated or ad-hoc reports. You may employ an OLAP or reporting tool with a user-friendly Graphical User Interface (GUI) to help users build their queries, perform analysis, or design their reports.

When planning your data warehouse, create one that will handle both structured and unstructured data and is cross-functional. It should also provide a long-term foundation for data provision and decision support.

Characteristics of Data warehouse

A data warehouse has following characteristics:

- Subject-Oriented
- Integrated
- Time-variant
- Non-volatile

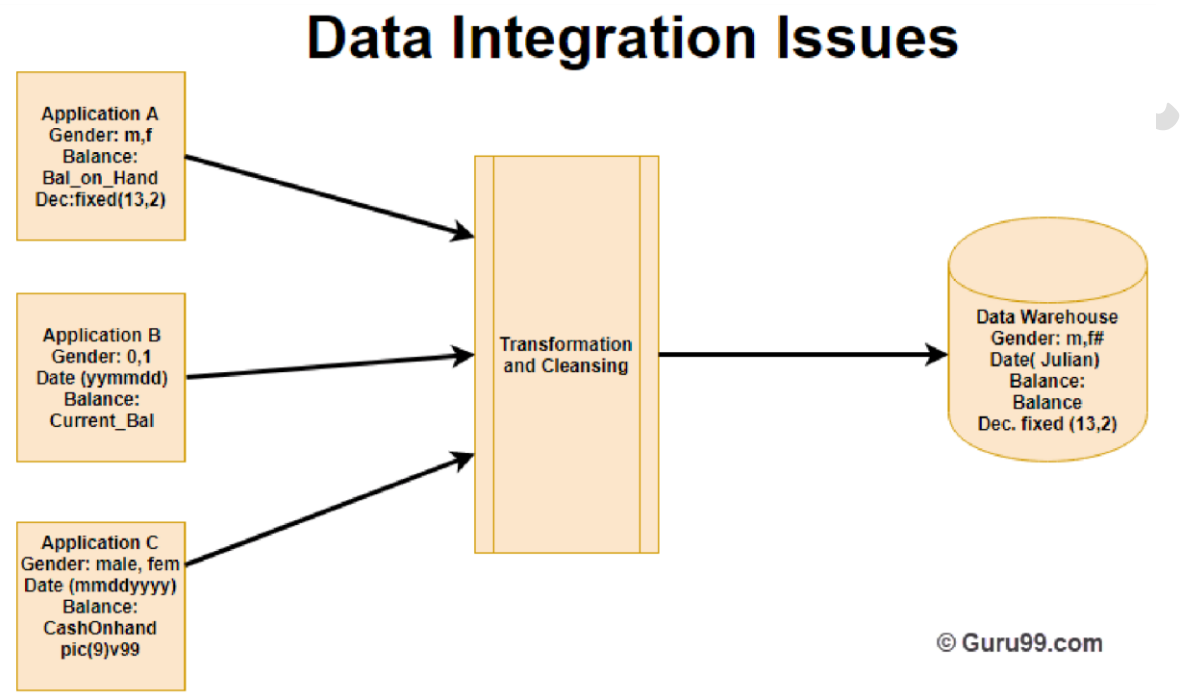
Subject-Oriented
A data warehouse is subject oriented as it offers information regarding a theme instead of companies' ongoing operations. These subjects can be sales, marketing, distributions, etc.

A data warehouse never focuses on the ongoing operations. Instead, it put emphasis on modeling and analysis of data for **decision making**. It also provides a simple and concise view around the specific subject by excluding data which not helpful to support the decision process.

Integrated
In Data Warehouse, integration means the establishment of a common unit of measure for all similar data from the dissimilar database. The data also needs to be stored in the Datawarehouse in common and universally acceptable manner.

A data warehouse is developed by integrating data from varied sources like a mainframe, relational databases, flat files, etc. Moreover, it must keep consistent naming conventions, format, and coding.

This integration helps in effective analysis of data. Consistency in naming conventions, attribute measures, encoding structure etc. have to be ensured. Consider the following example:



In the above example, there are three different application labeled A, B and C. Information stored in these applications are Gender, Date, and Balance. However, each application's data is stored different way.

- In Application A gender field store logical values like M or F
- In Application B gender field is a numerical value,
- In Application C application, gender field stored in the form of a character value. Same is the case with Date and balance

However, after transformation and cleaning process all this data is stored in common format in the Data Warehouse.

Time-Variant

The time horizon for data warehouse is quite extensive compared with operational systems. The data collected in a data warehouse is recognized with a particular period and offers information from the historical point of view. It contains an element of time, explicitly or implicitly. One such place where Data warehouse data display time variance is in in the structure of the record key. Every primary key contained with the DW should have either implicitly or explicitly an element of time. Like the day, week month, etc.

Another aspect of time variance is that once data is inserted in the warehouse, it can't be updated or changed.

Non-volatile

Data warehouse is also non-volatile means the previous data is not erased when new data is entered in it.

Data is read-only and periodically refreshed. This also helps to analyze historical data and understand what & when happened. It does not require transaction process, recovery and concurrency control mechanisms.

Activities like delete, update, and insert which are performed in an operational application environment are omitted in Data warehouse environment. Only two types of data operations performed in the Data Warehousing are

1. Data loading
2. Data access

Here, are some major differences between Application and Data Warehouse

Operational Application	**Data Warehouse**
Complex program must be coded to make sure that data upgrade processes maintain high integrity of the final product.	This kind of issues does not happen because data update is not performed.
Data is placed in a normalized form to ensure minimal redundancy.	Data is not stored in normalized form.
Technology needed to support issues of transactions, data recovery, rollback, and resolution as its deadlock is quite complex.	It offers relative simplicity in technology.

Data Marts

In a simple word Data mart is a subsidiary of a data warehouse. The data mart is used for partition of data which is created for the specific group of users.

Data marts could be created in the same database as the Data warehouse or a physically separate Database.