1 Introduction of Data Warehousing

Topic Covered

- What is Data Warehouse?
- Data warehousing Today
  - Business intelligence
  - Customer Relationship Management
  - Data Mining
- Future Trend in Data Warehousing
  - Unstructured Data
  - Search
  - service oriented architecture (SOA)
  - Real-time data warehousing
- Data Warehouse Architecture
  - Data Flow Architecture
  - System Architecture

What is Data Warehouse?

- A data warehouse is a powerful database model that significantly enhance the user ability to quickly analyses large, multidimensional dataset.
- It cleanses and organizes data to allow users to make business decisions based on facts. Data warehouse must have strong analytical characteristics.

- A data warehousing is a technique for collecting and managing data from varied sources to provide meaningful business insights. It is a combine of technologies and components, which allows the strategic use of data.
- It is electronic storage of a large amount of information by a business, which is design for query and analysis instead of transaction processing. It is a process of transforming data into information and making it available to users in a timely manner to make a difference.

Data Warehouse Features [Characteristics]
The key features of a data warehouse are discuss below:

1. **Subject Oriented**
   - A data warehouse is subject oriented because it provides information around a subject rather than the organization's ongoing operations.
   - These subjects can be product, customers, suppliers, sales, revenue, etc. A data warehouse does not focus on the ongoing operations; rather it focuses on modelling and analysis of data for decision-making.

2. **Integrated**
   - Integration is closely relate to subject orientation. Data warehouses need to have the data from different sources put into a consistent format.
   - This means that naming conflicts have to be resolved and problems like data being in different units of measure must be resolved.

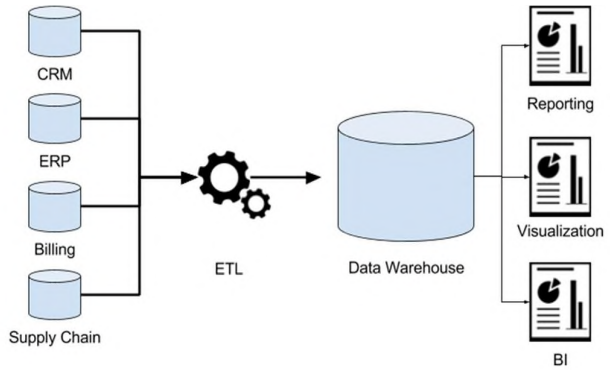
- A data warehouse integrates data from multiple data sources. For example, source A and source B may have different ways of identifying a product, but in a data warehouse, there will be only a single way of identifying a product.
- This integration enhances the effective analysis of data.

3. Nonvolatile

- *Nonvolatile* means that the data should not change once entered into the warehouse. This is logical because the purpose of a warehouse is to analyze what has occurred.
- Non-volatile means the previous data is not erase when new data is been added to it.

4. Time Variant

- The data collected in a data warehouse is been identified with a particular time. The data in a data warehouse provides information from the historical point of view.
- Historical data is been kept in a data warehouse. For example, one can retrieve data from 3 months, 6 months, 12 months, or even older data from a data warehouse.
- This contrasts with a transactions system, where often only the most recent data is been kept.
- For example, a transaction system may hold the most recent address of a customer, where a data warehouse can hold all addresses associated with a customer.



- **Advantages of Data Warehouse**
  It is a common process for the new implementations in a business that is based on various intelligent plans. Out of the many advantages of data warehousing, some of them discussed below.

1. Cleans data:

It mainly follows in data cleansing of removing errors that are inconsistent to improve the data and its respective quality. It emerges as a database containing many files. It has a variety of resources that made by using creativeness. It undergoes a process that enables one to deal with data cleaning substances.
Metadata reflects in sufficient quantity that especially means for all the constraints and even the system translation.

2. Indexes multiple types:

Indexing has created multiple database tables and created to speed up the accessing of information.
It can handle a large quantity of data and iterative queries before building the aligned form of data using OLTP applications. It has a huge number of existence within the modulated database system management queries.

3. Secured data and its access:

Security is the best way to mitigate the self publish breaches on rapid warehousing and that has to apply for all aspects as tradeoffs into potential warehousing behavior.
It has consolidated layered form of data with the objectives enabled and database enforced as to improve its values and gains.
It has critical compromising of sequential data within the unauthorized access.

4. Query processing with multiple options:

Query processes caries out in a parallel manner that helps in defining the unthinkable state of technology. These query tools design to process and load the data into various modules.
It accesses using simple logics along with a parallel repository of data. It enhances the defined field of routes and queries. It has a large number of query tools that manage heterogeneous resources. It handles requests from the tools online.

5. Enhanced business intelligence:

These insights develop within the information access and free from decision making. It limits the gut feelings and also defines each strategic credible fact of the evidence and backup.
It has personal needs that are varied within the better involved decision makings that are more competent with that of the limited data. It has warehouse related business tactics that measure within the informed facts. Financial management plays a vital role within the inventory management.

6. Increased system and query performance:

It mainly constructs to enhance and find the retrieval of data. It has the speed of performing different warehouses and the corresponding storage on large volumes. It has credible facts that involve storing large values.
It enables within the sequential information mediated within the business intelligence and has defined the modules that are matched with personal needs.
It constructs the operations of multiple subsystems. It concludes business intelligence and to alleviate the business repository. It gathers efforts for extracting the information.

7. Business Intelligence:

Many enterprises from a detailed log of multiple subsystems. It has different platforms that physically build within the data sources and access to a single phase of data.
It defines platforms that made different multiple sources and imagined to have a consolidated enterprise.
It enables a single data repository on a detailed subject to ensure that there is no duplication of data.

8. Timely access to data:

It helps the users to access different resources to analyze the data for the retrieval process. It spends time on schedule information on data that sequenced into routines. It has multiple resources that hold time for information technology.
It sustains for the queries and the consuming of data on query language. It has lesser information about the ability to generate standard reports that define with a special performance. It also has professional queries that diminish against warehouse reports.

9. Enhanced data consistency and quality:

It manages and sequences the illuminated data with the standardization of unique system resources. It has individual sales and utilization of a repository of data.
It has different and consistent units of substantially increased business. It accounts for the repository of operations and manages unique resources.

10. Return on investment is high:

Here the ROI made as a revenue part and with decreased expenses. It is a business that enables realize the project capital within the generates revenues and the cost savings.
The study of the business and substantial impact upon the analytics of the financial status can divide into various business studies.

11. Increase revenues:

It manages similar investigation systems that joined up for approach that might link to the stability of work and modulate within the deploy data on the database. It exists among the isolated warehouse departing from the cross checks and manage with the central point of each database.
It also follows a proactive approach within the link database to detect and prevent the summarized reports. It proactively minimizes the corporate investigators that match with increased streams.

12. Standardizes data across the organization:

Data standards are followed on different secured sharing of data. It has a particular standard within the modulated and visualized knowledge about connectivity. It contributes to numerous applications and is organized within the delivered data management systems. The conflict between data sharing avoided. It has critical applications that sequenced.

13. Database normalization:

The data can be stored and extracted in various forms that are stored in warehouse reports. It is a process of organizing the data in the relational database to minimize redundancy and that is more helpful in organizing the data. It emerges as a sequential flow of all the required data that are minimized.

- **Disadvantages of Data Warehouse**
  Even though there are a lot of advantages, people involve in implementing time and cost with high sequences that involve data translation, long time implementation of processes, lack of flexibility in the data transfer. Here are some of the disadvantages of data warehousing explained:

1. Raising ownership:

The majority of the data that are passed are held from the data resources and are represented within multiple efforts of a data warehouse. It intimates long term implementation of the schema and its resources.
It has its issues with raising ownership, privacy and secured results. It is associated with long term owners and with high costs.

2. Extra reporting:

The data warehouse will be run depending on the risks of the organization. It has typically generated teams that help in business negotiations. It manages to duplicate the data exist within the sequencing of the long term database. It consumes more time when the extra reporting is done.

3. **Data flexibility:**
It is arranged when the data that is imported has many static complaints and abilities that are mapped with the same schema and enumerated filtered displays. It is often recognized leaks between customers of an organization.
It generates analysis reports within the related privacy of the customer and is defined with minimal ability. It has limited value and constant transition that are mapped within the sequential processing of data.

4. **Compatibility with the existing system:**
The data warehouse system can be managed within the regular extract of the data that are loaded into the system. The usage of technology requires modification of data that has foremost concerns. All the existing system functionalities that are engaged are considered to be complex.

5. **Keeping data online:**
Softwares do not allow keeping the entire repository online after a certain duration. It maintains the data online and is enlarged by its textual means and large data online. It records and analyses the data for future reference.

6. **Dimensional technique:**
This technique contains all the information with specific events. It has a limit amount of information that identified with the proper understanding of all the events. It uses for many of the practical applications that are redundant.
The process of updating, deletion, and insertion process here. It accounts for the detailed description of the undesirable characteristics of data warehousing.

7. **Costs:**
Nowadays the maximum of the business started using techniques of the data warehouse. So the price range has fallen under the price range that most of the products towards design.
It complicated because even the small business details form when the situations are capable of designing the data provided. It manages the price range between the people in the company.
Thus, most of the tools that users begin with the transactions which in case accounts to the techniques of data warehousing. It groups all the transactions and signifies each operation that reports in detail.
It can access a large amount of information and will enable a neutral network that is replaced with the warehouse. Users supposed to train before using warehouse techniques.

- **Types of Data Warehouse**

Information processing, analytical processing and data mining are the three types

of data warehouse applications that are discussed below:

- **Information Processing**

A data warehouse allows processing the data stored in it. The data can be process by means of querying, basic statistical analysis, reporting using crosstabs, tables, charts, or graphs.

- **Analytical Processing**

A data warehouse supports analytical processing of the information stored in it. The data can be analyze by means of basic OLAP operations, including slice-and-dice, drill down, drill up, and pivoting.

- **How Data Warehouse Works?**

A data warehouse is a place where data collects by the information which flew from different sources. Usually, the data pass through relational databases and transactional systems. The data from here can assess by users as per the requirement with the help of various business tools, SQL clients, spreadsheets, etc.

The data flown will be in the following formats

- Structured
- Semi-structured
- Unstructured data

- **Difference between OLAP and OLTP**

Data Warehouse (OLAP) **OnLine Analytical Processing**	**Operational Database(OLTP)** **OnLine Transaction Processing**
It involves historical processing of information.	It involves day-to-day processing.
Knowledge workers such as executives, managers, and analysts use OLAP systems.	Clerks, DBAs, or database professionals use OLTP systems.
It is use to analyze the business.	It is use to run the business.
It focuses on Information out.	It focuses on Data in.
It is base on Star Schema, Snowflake Schema, and Fact Constellation Schema.	It is base on Entity Relationship Model.
It focuses on Information out.	It is application oriented.
It contains historical data.	It contains current data.
It provides summarized and consolidated data.	It provides primitive and highly detailed data.
It provides summarized and multidimensional view of data.	It provides detailed and flat relational view of data.
The number of users is in hundreds	The number of users is in thousands.
The number of records accessed is in millions.	The number of records accessed is in tens.
The database size is from 100GB to 100 TB.	The database size is from 100 MB to 100 GB.
These are highly flexible.	It provides high performance.

- **Applications of Data Warehouse:**

The business executives help in performing various other businesses to organize and analyze the detailed data description. These instances execute within the loop and monitor within a closed loop. Data warehousing mainly follow in the following fields:

- Airline
- Banking services
- Healthcare

- Public sector
- Investment and Insurance sector
- Telecommunication
- Hospitality Industry
- Financial services
- Retail sectors
- Consumer goods
- Controlled manufacturing

Data Warehouse Today

- A **Data Warehousing** (DW) is process for collecting and managing data from varied sources to provide meaningful business insights. A Data warehouse is typically use to connect and analyze business data from heterogeneous (different) sources.
- As the **size** of the databases grows, the estimates of what constitutes a very large database continue to grow. It is complex to build and run data warehouse systems, which are always increasing in size. The hardware and software resources are available today do not allow to keep a large amount of data online.
- Today most data warehouses are use for business intelligence to enhance CRM and for data mining. Some are also used for reporting, and some are used for data integration

1. **Business intelligence(BI)**
   - Business intelligence, as we know it today, would not be possible without the data warehouse.
   - In business intelligence, data warehouses serve as the backbone of data storage.
   - At its core, business intelligence is the ability to answer complex questions about your data and use those answers to make informed business decisions.
   - Data warehouse helps in achieving the vision for the managers and business executives.
   - In order to do this well, you need a data warehouse, which not only provides a safe way to centralize and store all your data but also a method to quickly find the answers you need, when you need them.
   - Companies use data warehouses to manage transactions, understand their data, and keep it all organized. In short, data warehouses make large amounts of information more usable for organizations of all sizes and types.

2. **Customer Relationship Management**
   - Customer Relationship Management (CRM) is the activity of establishing contact and managing communications with customers, analyzing information about customers, campaigning to attract new customers, performing business transactions with customers, servicing customers, and providing support to customers.
   - Data warehouses that fulfill CRM's basic requirements of granular customer transaction data are termed as customer data repositories (CDR).
   - Effective CRM collect data at every customer interaction and then analyze it for future improvements. The data warehouse becomes the repository for all customer information from all sources including operational systems, interaction data (solicitations, call centers), demographic data, behavioral data and customer-provided bio data.

- The key focus of a data warehouse is to support enterprise decision support system and is not restricted to a specific LOB (Line of Business).

3. Data Mining

- Data mining is the process of analyzing massive volumes of data to discover business intelligence that helps companies solve problems, mitigate risks, and seize new opportunities.
- Data mining can answer business questions that traditionally were too time consuming to resolve manually. Using a range of statistical techniques to analyze data in different ways, users can identify patterns, trends and relationships they might otherwise miss.
- Data mining is been used in many areas of business and research, including sales and marketing, product development, healthcare, and education.
- When used correctly, data mining can provide a profound advantage over competitors by enabling you to learn more about customers develop effective marketing strategies, increase revenue, and decrease costs.
- The insights extracted via Data mining can be use for marketing, fraud detection, and scientific discovery, etc.

Future Trend of Data Warehousing

Several future trends in data warehousing today are unstructured data, search, service and real-time data warehousing.

1. Unstructured Data

- Unstructured data makes a unique challenge for companies wishing to utilize their data for analysis.
- If your company is trying to deal with its unstructured data now, that issue will weaken over time.
- In the business world, most unstructured information lies in customer-related content, which is plentiful and accessible.
- Organizations utilize big data tools and software such as Hadoop to process, mine, integrate, store, track, index and report business insights from raw unstructured information.
- Data that is in databases is structured; it is been organized in row and columns in data warehouse.
- How do you store unstructured data in the data warehouse? Well, the answer to this is for each unstructured data item you define the attributes and then organize these items according to the attributes. You can store the unstructured data items in a relational database as a binary object column, with the attributes as other columns or you can store the unstructured data items in the file systems and just store the pointer to the file in the database.

2. Search

- How do you get the information out after store data warehouse? The answer is by searching. To get the information out of structured data.
- Search Engine refers to a huge database of internet resources such as web pages, newsgroups, programs, images etc. It helps to locate information on World Wide Web.

- User can search for any information by passing query in form of keywords or phrase. It then searches for relevant information in its database and return to the user.
- Search Technologies has been a leader in implementing search-based applications for quite some time with a "who's who" list of customers.

3. Service Oriented Architecture (SOA)

- Service-oriented architecture is a software architecture that describes a component-based approach to allow loosely coupled components to address business requirements.
- This new IT solution will connect the SOA integration layer to near-real-time data warehouse and consider the near-real-time data warehouse as a data stage source for other business and system environments.
- The expected outcomes from data will be combining near-real-time data warehousing with SOA architecture to deliver integrated tactical and strategic data.
- More and more data warehousing applications on all fronts are been built using SOA: ETL, reporting, analytics, BI applications, data mining, metadata, data quality and data cleansing.
- In the future, it would be easier to update one component without affecting the others and to connect different components that are been made using different technologies.

4. Real-time data warehousing

- Real-time Data Warehousing describes a system that reflects the state of the warehouse in real time. If a query run against the real-time data warehouse to understand a particular fact about the business or entity described by the warehouse, the answer reflects the state of that entity at the time the query run.
- A real-time data warehouse is a data warehouse that is update (by the ETL) the moment the transaction happens in the source system.
- The other approach of implementing a real-time data warehouse is to modify the operational source application to write to the data warehouse staging area, immediately after it writes the data in its internal database.

Data Warehouse Architectures

➢ Data warehouse S/W and H/W Architecture:

- The different methods used to construct/organize a data warehouse specified by an organization are numerous. The hardware utilized, software created and data resources specifically required for the correct functionality of a data warehouse are the main components of the data warehouse architecture.
- Production applications such as payroll, accounts, payable, product, purchasing and inventory control are design for online transaction processing **(OLTP)**. Such applications gather detailed data from day to day operations.
- Data Warehouse applications are design to support the user ad-hoc data requirements, an activity recently dubbed online analytical processing (OLAP). These include applications such as forecasting, profiling, summary reporting, and trend analysis.
- Basic technical infrastructure includes operating system, hardware platform, database management system, and network. The DBMS selection becomes a little more complicated than a straightforward operational system because of the unusual

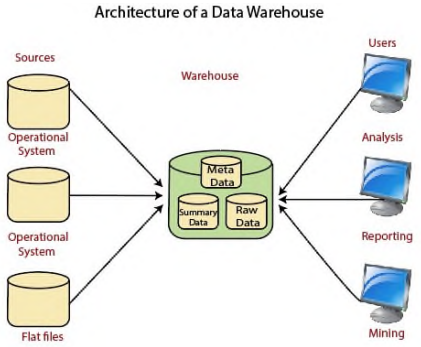
challenges of the data warehouse, especially in its capability to support very complex queries that cannot been predicted in advance.

- **Basic steps to develop DW architecture:**
  1. The first and the most important step of developing effective DW architecture are to enlist the full support and commitment of project sponsor/executive of the company.
  2. Appointed staff in architecture team must be strongly skilled personnel. It is not necessarily the technology you choose for your architecture, it is the personnel you have designing and developing the architecture that makes the project successful.
  3. Design and develop prototype that can be used to test all of the different technologies that are being considered
  4. The architecture team should be give enough time to build the architecture infrastructure before development begins.
  5. The development staff must be train on the architecture before development begins.
  6. Provide freedom to the architecture team to enhance and improve the architecture as the project moves forward.

- Data warehouses and their architectures vary depending upon the specifics of an Organization's situation.
- Three common architectures are:
  - Data Warehouse Architecture (Basic)
  - Data Warehouse Architecture (with a Staging Area)
  - Data Warehouse Architecture (with a Staging Area and Data Marts)

Data Warehouse Architecture (Basic)

- Figure shows a simple architecture for a data warehouse. End users directly access data derived from several source systems through the data warehouse.



- In above Figure, the metadata and raw data of a traditional OLTP system is present, as is an additional type of data, summary data. Summaries are very valuable in data warehouses because they pre-compute long operations in advance.
- For example, a typical data warehouse query is to retrieve something such as August sales.

Data Warehouse Architecture (with a Staging Area):

- You need to cleanse and process your operational data before putting it into the warehouse, as shown in Figure. You can do this programmatically, although most data warehouses use a staging area instead.
- A staging area simplifies building summaries and general warehouse management.

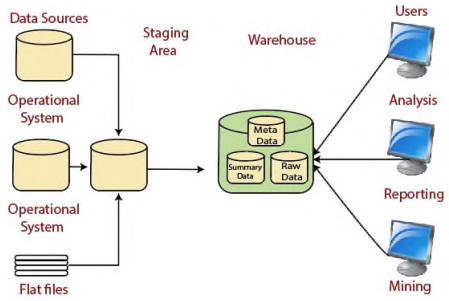


Figure: Architecture of a Data Warehouse with a Staging Area

Data Warehouse Architecture (with a Staging Area and Data Marts)

- Although the architecture in above figure is quite common, you may want to customize your warehouse's architecture for different groups within your organization.
- You can do this by adding data marts, which are systems designed for a particular line of business.(LOB)
- Following Figure illustrates an example where purchasing, sales, and inventories are separate. In this example, a financial analyst might want to analyze historical data for purchases and sales.

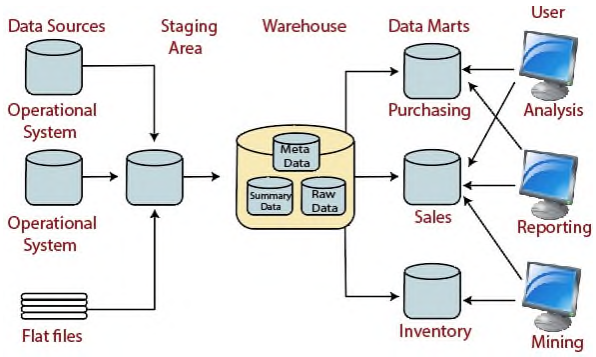
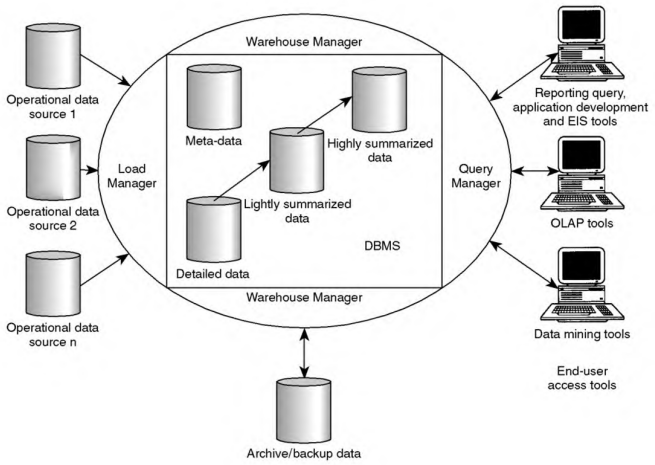


Figure: Architecture of a Data Warehouse with a Staging Area and Data Marts

Architectural component of Data warehouse:

- **Architecture** is the proper arrangement of the elements. We build a data warehouse with software and hardware components. To suit the requirements of our organizations, we arrange these building we may want to boost up another part with extra tools and services. All of these depend on our circumstances.
- The architecture includes everything that is need to prepare the data and store it.
- The architecture is further composed of the rules, procedures and functions that enable data warehouse to work and fulfill the business requirements.

- Finally, the architecture made up of the technology that empowers the data warehouse.
- The architecture made up of a number of interconnected components.



- The figure shows the essential elements of a typical warehouse.
- We see the Source Data component shows on the left. The data in a data warehouse comes from operational systems of the organization as well as from other external sources.
- The Data staging element serves as the next building block.
- In the middle, we see the Data Storage component that handles the data warehouses data. This element not only stores and manages the data; it also keeps track of data using the metadata repository.
- The Information Delivery component shows on the right consist of all the different ways of making the information from the data warehouses available to the users.

OPERATIONAL DATA	The sources of data for the data warehouse is supplied from: 1.The data from the mainframe systems in the traditional network and hierarchical format. 2.Data can also come from the relational DBMS like Oracle, Informix. 3.In addition to these internal data, operational data also includes external data obtained from commercial databases and databases associated with supplier and customers.
LOAD MANAGER	• The load manager performs all the operations associated with extraction and loading data into the data warehouse. • These operations include simple transformations of the data to prepare the data for entry into the warehouse. • The size and complexity of load manager will vary between data warehouses. Load manager may be constructed using a combination of vendor data, loading tools and custom built programs (off-the-shelf tools, bespoke coding, c programs and shell scripts)

WAREHOUSE MANAGER	• The warehouse manager performs all the operations associated with the management of data in the warehouse. • This component is built using vendor data, management tools and custom built programs. • The operations performed by warehouse manager include: 1. Analysis of data to perform consistency. 2. Transformation and merging the source data from temporary storage into data warehouse tables 3. Create indexes and views on base tables 4. Denormalization 5. Generate aggregation 6. Backing up and archiving of data • In certain situations, the warehouse manager also analyzed query profiles to determine which indexes and aggregations are appropriate.
QUERY MANAGER	• The query manager performs all operations necessary to support the query management process. • The work of query manager is that it directs queries to the appropriate tables and schedules the execution of user queries. • This component is usually constructed using vendor user access tools, data warehousing monitoring tools, database facilities and custom built programs. • The complexity of a query manager is determined by facilities provided by the user access tools and database.
DETAILED DATA	• This area of the warehouse stores all the detailed data in the database schema. In most cases detailed data is not stored online but aggregated to the next level of details. • However the detailed data is added regularly to the warehouse to supplement the aggregated data.
LIGHTLY AND HIGHLY SUMMERIZED DATA	• The area of the data warehouse stores all the predefined lightly and highly summarized (aggregated) data generated by the warehouse manager. • This area of the warehouse is transient (volatile) as it will be subject to change on an ongoing basis in order to respond to the changing query profiles. The purpose of the summarized information is to speed up the query performance. • The summarized data is updated continuously as new data is loaded into the warehouse.
ARCHIVE AND BACK UP DATA	• This area of the warehouse stores detailed and summarized data for the purpose of archiving and back up. • The data is transferred to storage archives such as magnetic tapes or optical disks.
META DATA	• Meta data stores all the Meta data definitions used by all processes within the data warehouse. • In order to provide universal data access, it is necessary to maintain some form of data directory of Meta data information. • Meta data is data about data within the enterprise for example record description in a COBOL program, info in the ER diagram is Meta data. Meta data is used for variety of purpose including:

	1. **The extraction and loading process**- Metadata is used to map data sources to a common view of information within the warehouse. 2. **The warehouse management process**- Metadata is used to automate the production of summary tables. 3. **As part of query management process** - Metadata is used to direct a query to the most appropriate data source. • The structure of Meta data will differ in each process, because the purpose is different.

Types of Architecture

- **A** data warehouse system has two main architectures: the **data flow architecture and the system architecture.**

DATA FLOW ARCHITECTURE

- The data flow architecture is about how the data stores are arranged within a data warehouse and how the data flows from the source systems to the users through these data stores.
- In data warehousing, the data flow architecture is a configuration of data stores within a data warehouse system, along with the arrangement of how the data flows from the source systems through these data stores to the applications used by the end users.
- This include show the data flows are controlled, logged, and monitored, as well as the mechanism to ensure the quality of the data in the data stores.
- **Data stores** are important components of data flow architecture. A data store is one or more databases or files containing data warehouse data, arranged in a particular format and involved in data warehouse processes.
- Based on the user accessibility, you can classify data warehouse data stores into three types:

  - A **user-facing data store** is a data store that is available to end users and is queried by the end users and end-user applications.
  - An **Internal data store** is a data store that is used internally by data warehouse components for the purpose of integrate, cleansing, logging, and preparing data, and it is not open for query by the end users and end-user applications.
  - A **hybrid data store** is used for both internal data warehouse mechanisms and for query by the end users and end-user applications.

- A master data store is a user-facing or hybrid data store containing a complete set of data in a data warehouse, including all versions and all historical data. Based on the data format,

- You can classify data warehouse data stores into four types:
  - A stage is an **internal data store** used for transforming and preparing the data obtained from the source systems, before the data is loaded to other data stores in a data warehouse.

- A **normalized data store (NDS)** is an internal master data store in the form of one or more normalized relational databases for the purpose of integrating data from various source systems captured in a stage, before the data is loaded to a user-facing data store.
- An **operational data store (ODS)** is a hybrid data store in the form of one or more normalized relational databases, containing the transaction data and the most recent version of master data, for the purpose of supporting operational applications.
- A **dimensional data store (DDS)** is a user-facing data store, in the form of one or more relational databases, where the data is arranged in dimensional format for the purpose of supporting analytical queries.

System Architecture

- The **syst**em architecture is about the physical configuration of the servers, network, software, storage, and clients.
- System architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.
- System architecture can consist of system components and the sub-systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called Architecture Description Language (ADL)
- System architecture makes use of elements of both software and hardware and is used to enable design of such a composite system.
- System architecture primarily concentrates on the internal interface among the systems components or subsystem and on the interface between the system and its external environment with user.
- There are 3 approaches for constructing Data Warehouse layers: single tier, two tiers and three tiers. This 3 tier architecture of Data Warehouse is explained as below.

Single-tier architecture

The objective of a single layer is to minimize the amount of data stored. This goal is to remove data redundancy. This architecture is not frequently used in practice.

Two-tier architecture

Two-layer architecture is one of the Data Warehouse layers which separates physically available sources and data warehouse. This architecture is not expandable and also not supporting a large number of end-users. It also has connectivity problems because of network limitations.

Three-Tier Data Warehouse Architecture

This is the most widely used Architecture of Data Warehouse.

It consists of the Top, Middle and Bottom Tier.

- **Bottom Tier:** The database of the Data warehouse servers as the bottom tier. It is usually a relational database system. Data is cleansed, transformed, and loaded into this layer using back-end tools.
- **Middle Tier:** The middle tier in Data warehouse is an OLAP server which is implemented using either ROLAP or MOLAP model. For a user, this application tier presents an abstracted view of the database. This layer also acts as a mediator between the end-user and the database.
- **Top-Tier:** The top tier is a front-end client layer. Top tier is the tools and API that you connect and get data out from the data warehouse. It could be Query tools, reporting tools, managed query tools, Analysis tools and Data mining tools.

Use of Data Warehouse

Here, are most common sectors where Data warehouse is used:

Airline:

- In the Airline system, it is used for operation purpose like crew assignment, analyses of route profitability, frequent flyer program promotions, etc.

Banking:

- It is widely used in the banking sector to manage the resources available on desk effectively. Few banks also used for the market research, performance analysis of the product and operations.

Healthcare:

- Healthcare sector also used Data warehouse to strategize and predict outcomes, generate patient's treatment reports, share data with tie-in insurance companies, medical aid services, etc.

Public sector:

- In the public sector, data warehouse is used for intelligence gathering. It helps government agencies to maintain and analyze tax records, health policy records, for every individual.

Investment and Insurance sector:

- In this sector, the warehouses are primarily used to analyze data patterns, customer trends, and to track market movements.

Retain chain:

- In retail chains, Data warehouse is widely used for distribution and marketing. It also helps to track items, customer buying pattern, promotions and also used for determining pricing policy.

Telecommunication:

- A data warehouse is used in this sector for product promotions, sales decisions and to make distribution decisions.

Hospitality Industry:

- This Industry utilizes warehouse services to design as well as estimate their advertising and promotion campaigns where they want to target clients based on their feedback and travel patterns.

Definations

1) A Relational database is a database that consists of entity tables with parent-child relationships between them.
2) A Normalized database is a database with little or no data redundancy in third normal form or higher.

3) A Denormalized database is a database with some data redundancy that has not gone through a normalization process.
4) A Dimensional database is a denormalized database consisting of fact tables and common dimension tables containing the measurements of business events, categorized by their dimensions.

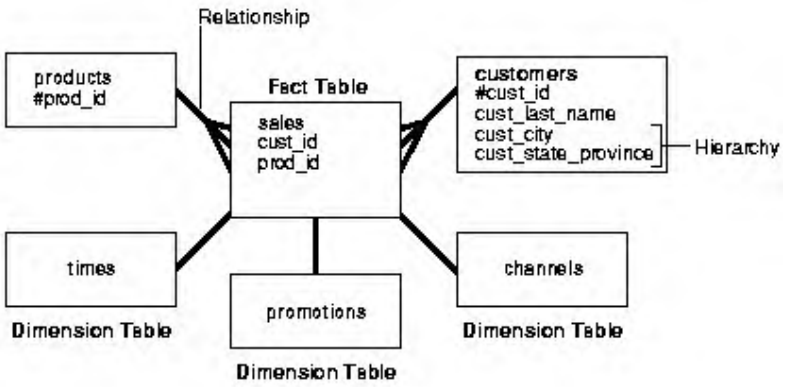
2 Designing & Implementation of Data warehousing

Topic Covered

1. Logical Design for data warehouse
   - Data Warehousing Schemas
     - Star Join (Star Schema) and Snowflake structure
2. Physical Design for data warehouse
   - Physical Design Structures
     - Tablespaces
     - Tables and Partitioned Tables
     - Views
     - Integrity Constraints
     - Dimensions
3. Design dimension table fact table for data warehouse
4. Design and implement effective physical data structure for data warehouse

1. Logical Design For Data Warehouse

- A logical design is conceptual and abstract. You do not deal with the physical implementation details yet. You deal only with defining the types of information that you need.
- One technique you can use to model your organization's logical information requirements is entity-relationship modelling. Entity-relationship modelling involves identifying the things of importance (entities), the properties of these things (attributes), and how they are related to one another (relationships).
- The process of logical design involves arranging data into a series of logical relationships called entities and attributes. An entity represents a mass of information. In relational databases, an entity often maps to a table. An attribute is a component of an entity that helps define the uniqueness of the entity. In relational databases, an attribute maps to a column.
- To be sure that your data is consistent, you need to use unique identifiers. A unique identifier is something you add to tables so that you can differentiate between the same item when it appears in different places. In a physical design, this is usually a primary key.
- While entity-relationship diagramming has traditionally been associated with highly normalized models such as OLTP applications, the technique is still useful for data warehouse design in the form of dimensional modelling.
- In dimensional modelling, instead of seeking to discover atomic units of information (such as entities and attributes) and all of the relationships between them, you identify which information belongs to a central fact table and which information belongs to its associated dimension tables.
- You identify business subjects or fields of data, define relationships between business subjects, and name the attributes for each subject.



- A logical data warehouse is an architectural layer that sits a top the usual data warehouse (DW) store of persisted data. The logical layer provides (among other things) several mechanisms for viewing data in the warehouse store and elsewhere across an enterprise without relocating and transforming data ahead of view time.
- These views also serve as interfaces into disparate data and its sources. In other words, the logical data warehouse complements the traditional core warehouse (and its primary function of a priori data aggregation, transformation, and persistence) with functions that fetch and transform data, in real time (or near to it), thereby instantiating non-persisted data structures, as needed.
- The advantage of the logical layer is that data is fresher (as required by time-sensitive business processes) and the structure of delivered data is created at run time (as required by discovery oriented analytics), without limiting data to the pre-built structures of the DW's persisted store.

- **Data Warehouse Dimensional Modeling (Types of Schemas)**
  - A schema is a collection of database objects, including tables, views, indexes, and synonyms. You can arrange schema objects in the schema models designed for data warehousing in a variety of ways. Most data warehouses use a dimensional model.

  - Before detailing each of the architectures, there are two concepts that need to be clarified:

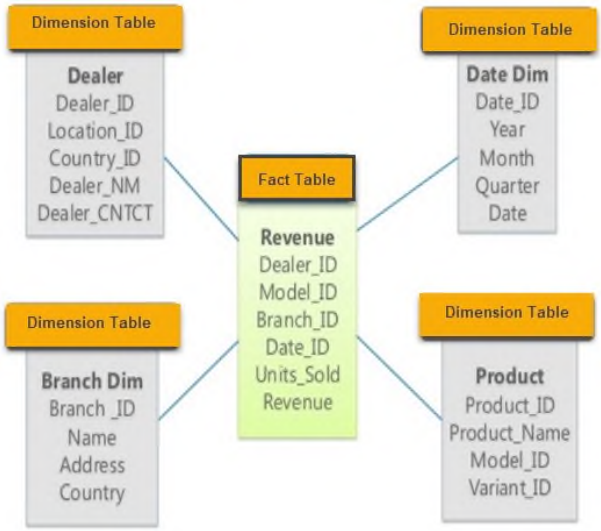
    - **Fact table** :-
      A fact table typically has two types of columns: foreign keys to dimension tables and measures those that contain numeric facts. A fact table can contain fact's data in detail or aggregated level;

    - **Dimension table** :-
      A dimension is a structure, usually composed of one or more hierarchies that categorizes data. If a dimension hasn't got hierarchies and levels it is called a flat dimension or list. The primary keys of each of the dimension tables are part of the composite primary key of the fact table. Dimensional attributes help to describe the dimensional value. They are normally descriptive, textual values. Dimension tables are generally smaller in size than fact table.

- There are three types of schemas are available in data warehouse. Out of which the star schema is mostly used in the data warehouse designs. The second mostly used data warehouse schema is snow flake schema.

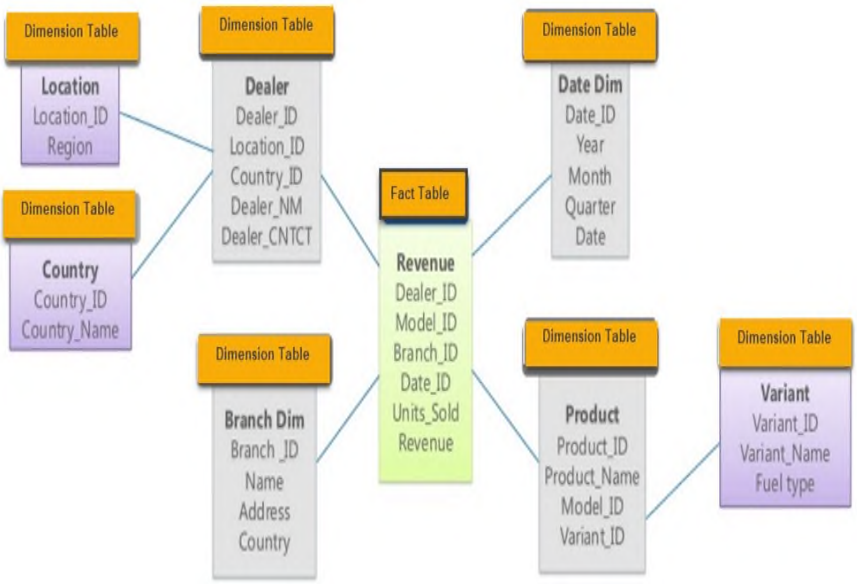
1) Star Schema:

- Star Schema in data warehouse, in which the center of the star can have one fact table and a number of associated dimension tables. It is known as star schema as its structure resembles a star. The Star Schema data model is the simplest type of Data Warehouse schema. It is also known as Star Join Schema and is optimized for querying large data sets.
- In the following Star Schema example, the fact table is at the center which contains keys to every dimension table like Dealer_ID, Model ID, Date_ID, Product_ID, Branch_ID & other attributes like Units sold and revenue.



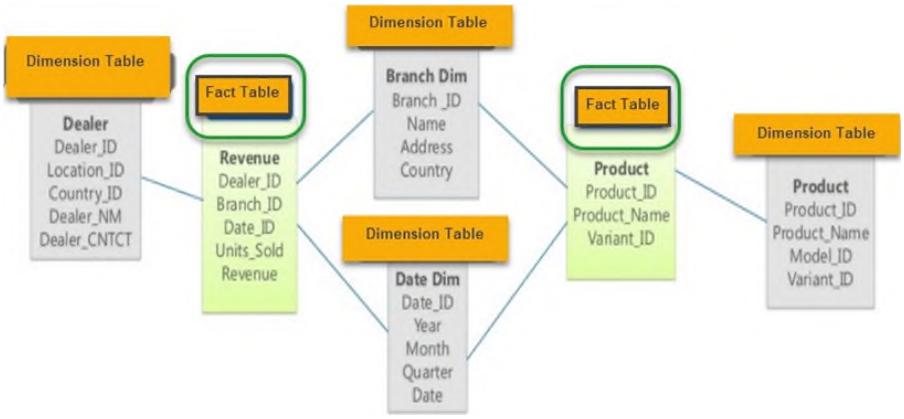
2) Snow Flake Schema:

- A snow flake schema is an enhancement of star schema by adding additional dimensions.
- Snowflake Schema in data warehouse is a logical arrangement of tables in a multidimensional database such that the ER diagram resembles a snowflake shape. A Snowflake Schema is an extension of a Star Schema, and it adds additional dimensions. The dimension tables are normalized which splits data into additional tables.
- In the following Snowflake Schema example, Country is further normalized into an individual table.



3) Galaxy Schema [Fact Constellation]:

- A **Galaxy Schema** contains two fact table that share dimension tables between them. It is also called Fact Constellation Schema. The schema is viewed as a collection of stars hence the name Galaxy Schema.
- This schema is a combination of many data marts.



2. Physical Design For Data Warehouse

- During the physical design process, you convert the data gathered during the logical design phase into a description of the physical database structure.
- During the logical design phase, you defined a model for your data warehouse consisting of entities, attributes, and relationships. The entities are linked together using relationships. Attributes are used to describe the entities. The unique identifier (UID) distinguishes between one instance of an entity and another.
- Figure illustrates a graphical way of distinguishing between logical and physical designs.
- Figure Logical Design Compared with Physical Design

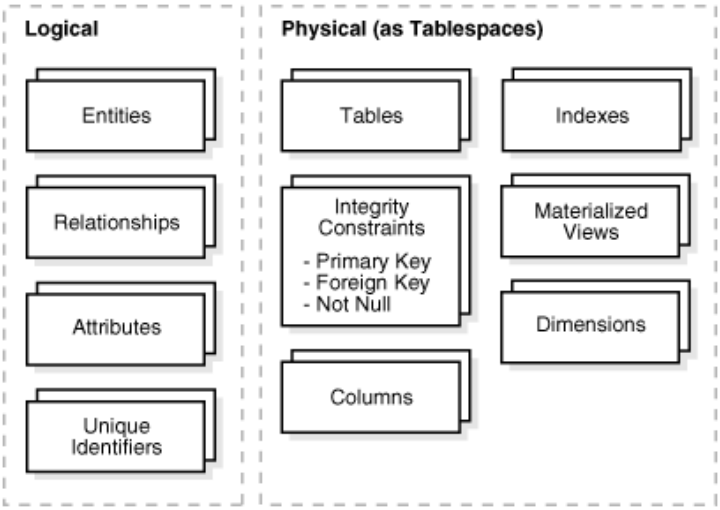


Figure :- Logical Design Compared with Physical Design

- During the physical design process, you translate the expected schemas into actual database structures. At this time, you must map:
  - Entities to tables
  - Relationships to foreign key constraints
  - Attributes to columns
  - Primary unique identifiers to primary key constraints
  - Unique identifiers to unique key constraints

- **Physical Design Structures**
- Once you have converted your logical design to a physical one, you must create some or all of the following structures:
  1. Tablespaces
  2. Tables and Partitioned Tables
  3. Views
  4. Integrity Constraints
  5. Dimensions
- Some of these structures require disk space. Others exist only in the data dictionary. Additionally, the following structures may be created for performance improvement:

  6. Indexes and Partitioned Indexes
  7. Materialized Views

1) Tablespace :

- Tablespace is a storage location where the actual data underlying database objects can be kept. It provides a layer of abstraction between physical and logical data and serves to allocate storage for all DBMS managed segments. (A database segment is a database object which occupies physical space such as table data and indexes.) Once created, a tablespace can be referred to by name when creating database segments.
- Tablespaces specify only the database storage locations, not the logical database structure, or database schema.Sometimes it can be used to specify schema so as to form a bond between logical and physical data.

- A tablespace consists of one or more datafiles, which are physical structures within the operating system you are using. A datafile is associated with only one tablespace. From a design perspective, tablespaces are containers for physical design structures.
- Tablespace is the core small piece for backup and recovery or the transportable tablespaces mechanism, the logical business design affects availability and maintenance operations.

2) <u>Table & Partitioned Tables :</u>

- Tables are the basic unit of data storage. They are the container for the expected amount of raw data in your data warehouse.
- Using partitioned tables instead of non partitioned ones addresses the key problem of supporting very large data volumes by allowing you to divide them into smaller and more manageable pieces.
- The main design criterion for partitioning is manageability, though you also see performance benefits in most cases because of partition intelligent parallel processing.
- For example, you might choose a partitioning strategy based on a sales transaction date and a monthly small piece. If you have four years' worth of data, you can delete a month's data as it becomes older than four years with a single, fast DDL statement and load new data while only affecting 1/48th of the complete table. Business questions regarding the last quarter only affect three months, which is equivalent to three partitions, or 3/48ths of the total volume.
- Partitioning large tables improves performance because each partitioned piece is more manageable. Typically, you partition based on transaction dates in a data warehouse. For example, each month, one month's worth of data can be assigned its own partition.

3) <u>Views</u>

- A view is a tailored presentation of the data contained in one or more tables or other views. A view takes the output of a query and treats it as a table. Views do not require any space in the database.

4) <u>Integrity Constraints</u>

- Integrity constraints are used to enforce business rules associated with your database and to prevent having invalid information in the tables. Integrity constraints in data warehousing differ from constraints in OLTP environments. In OLTP environments, they primarily prevent the insertion of invalid data into a record,
- Which is not a big problem in data warehousing environments because accuracy has already been guaranteed. In data warehousing environments, constraints are only used for query rewrite. NOT NULL constraints are particularly common in data warehouses. Under some specific circumstances, constraints need space in the database. These constraints are in the form of the underlying unique index.

5) <u>Dimensions</u>

- A dimension is a schema object that defines hierarchical relationships between columns or column sets. A hierarchical relationship is a functional dependency from one level of a hierarchy to the next one. A dimension is a container of logical relationships and does not require any space in the database. A typical dimension is city, state (or province), region, and country.

- **<u>Indexes and Partitioned Indexes</u>**

- Indexes are optional structures associated with tables or clusters. In addition to the classical B-tree indexes, bitmap indexes are very common in data warehousing environments. Bitmap indexes are optimized index structures for set-oriented operations. Additionally, they are necessary for some optimized data access methods such as star transformations.
- Indexes are just like tables in that you can partition them, although the partitioning strategy is not dependent upon the table structure. Partitioning indexes makes it easier to manage the data warehouse during refresh and improves query performance.

- **Materialized Views**

  - Materialized views are query results that have been stored in advance so long-running calculations are not necessary when you actually execute your SQL statements. From a physical design point of view, materialized views resemble tables or partitioned tables and behave like indexes in that they are used transparently and improve performance.

3. Design dimension & fact table of DataWarehouse

Dimension

- A **dimension** is a structure that categorizes facts and measures in order to enable users to answer business questions. Commonly used dimensions are people, products, place and time.
- In a data warehouse, dimensions provide structured labeling information to otherwise unordered numeric measures. The dimension is a data set composed of individual, non-overlapping data elements. The primary functions of dimensions are three fold: to provide filtering, grouping and labeling.

- These functions are often described as "slice and dice". A common data warehouse example involves sales as the measure, with customer and product as dimensions. In each sale a customer buys a product. The data can be sliced by removing all customers except for a group under study, and then diced by grouping by product.

Dimension Table

- A Dimension Table is a table in a star schema of a data warehouse. Data warehouses are built using dimensional data models which consist of fact and dimension tables. Dimension tables are used to describe dimensions; they contain dimension keys, values and attributes.
- For example,

  The time dimension would contain every hour, day, week, month, quarter and year that has occurred since you started your business operations.

  Product dimension could contain a name and description of products you sell, their unit price, color, weight and other attributes as applicable. Attributes would be a customer's first and last name, age, gender etc.

- Dimension table rows are uniquely identified by a single key field. It is recommended that the key field be a simple integer because a key value is meaningless, used only for joining fields between the fact and dimension tables.
- A dimension table data element is similar to a categorical variable in statistics.

- Typically dimensions in a data warehouse are organized internally into one or more hierarchies. "Date" is a common dimension, with several possible hierarchies:
  - "Days (are grouped into) Months (which are grouped into) Years",
  - "Days (are grouped into) Weeks (which are grouped into) Years"
  - "Days (are grouped into) Months (which are grouped into) Quarters (which are grouped into) Years"

- In data warehousing, a dimension **table** is one of the set of companion tables to a fact table.
- The fact table contains business facts (or *measures*), and foreign keys which refer to candidate keys (normally primary keys) in the dimension tables.
- Contrary to fact tables, *dimension* tables contain descriptive attributes (or fields) that are typically textual fields (or discrete numbers that behave like text). These attributes are designed to serve two critical purposes: query constraining and/or filtering, and query result set labeling.

- Dimension attributes should be:
  - Verbose (labels consisting of full words)
  - Descriptive
  - Complete (having no missing values)
  - Discretely valued (having only one value per dimension table row)
  - Quality assured (having no misspellings or impossible values)
- The use of surrogate dimension keys (surrogate key [SK] is sequentially generated meaningless unique number attached with each and every record in a table in any data warehouse) brings several advantages, including:
  - Performance. Join processing is made much more efficient by using a single field (the surrogate key)
  - Buffering from operational key management practices. This prevents situations where removed data rows might reappear when their natural keys get reused or reassigned after a long period of dormancy.
  - Mapping to integrate disparate sources
  - Handling unknown or not-applicable connections
  - Tracking changes in dimension attribute values

Types of Dimensions in data warehouse

- A dimension table consists of the attributes about the facts. Dimensions store the textual descriptions of the business. Without the dimensions, we cannot measure the facts. The different types of dimension tables are explained in detail below.

1. **Conformed Dimension:**

   A conformed dimension is the dimension that is shared across multiple data mart or subject area. Company may use the same dimension table across different projects without making any changes to the dimension tables.
   Conformed dimension example would be *Customer* dimension, i.e. both *marketing* and *sales* department can use Customer dimension for their reporting purpose.

Eg: The *date* dimension table connected to the *sales* facts is identical to the date dimension connected to the *inventory* facts.

2. **Junk Dimension:**

   A junk dimension is a collection of random transactional codes flags and/or text attributes that are unrelated to any particular dimension. The junk dimension is simply a structure that provides a convenient place to store the junk attributes.

   Eg: Assume that we have a *gender* dimension and *marital* status dimension. In the fact table we need to maintain two keys referring to these dimensions. Instead of that create a junk dimension which has all the combinations of gender and marital status (cross join gender and marital status table and create a junk table). Now we can maintain only one key in the fact table.

3. **Degenerated Dimension:**

   A degenerate dimension is a dimension which is derived from the fact table and doesn't have its own dimension table.
   For example, receipt number does not have dimension table associated with it. Such details are just for information purpose.

4. **Role-playing Dimension:**

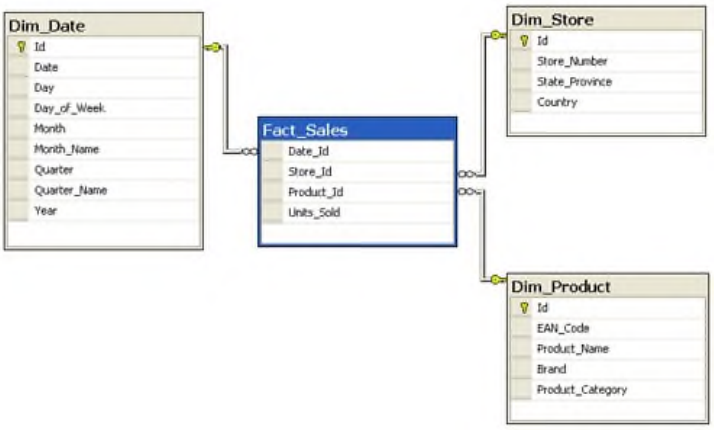
   Dimensions which are often used for multiple purposes within the same database are called role-playing dimensions.
   Role-playing occurs when a single physical dimension appears several times in a fact table, each represented as a separate logical table with unique column names through views.
   A single dimension which is expressed differently in a fact table using views is called as role-playing dimension.
   Single dimension appears several times in the same fact table like Orderdate, date, shipdate.
   For example, a date dimension can be used for “date of sale", as well as "date of delivery", or "date of hire".

5. **Audit Dimension:**
   A table which stores statistical information about data warehousing objects.

Definition of Fact Table

- A fact table is used in the dimensional model in data warehouse design. A fact table is found at the center of a star schema or snowflake schema surrounded by dimension tables. A fact table consists of facts of a particular business process e.g., sales revenue by month by product.

sales volume by day by product by store



- The fact table mainly consists of business facts and foreign keys that refer to primary keys in the dimension tables. A dimension table consists mainly of descriptive attributes that are textual fields. A dimension table contains a surrogate key, natural key, and a set of attributes.
- The fact table contains business facts (or measures), and foreign keys which refer to candidate keys (normally primary keys) in the dimension tables. Contrary to fact tables, dimension tables contain descriptive attributes (or fields) that are typically textual fields (or discrete numbers that behave like text).

- A fact table typically has two types of columns: those that contain facts and those that are a foreign key to dimension tables.
- The primary key of a fact table is usually a composite key that is made up of all of its foreign keys.
- Fact tables contain the content of the data warehouse and store different types of measures like additive, non additive, and semi additive measures.
- Fact tables provide the (usually) additive values that act as independent variables by which dimensional attributes are analyzed. Fact tables are often defined by their grain.
- The grain of a fact table represents the most atomic level by which the facts may be defined. The grain of a sales fact table might be stated as "sales volume by day by product by store".
- Each record in this fact table is therefore uniquely defined by a day, product and store. Other dimensions might be members of this fact table (such as location/region)

Type Of Fact Table

- There are four fundamental measurement events, which characterize all fact tables.

1. **Transactional**

- A transactional table is the most basic and fundamental. The grain associated with a transactional fact table is usually specified as "one row per line in a transaction", e.g., every line on a receipt. Typically a transactional fact table holds data of the most detailed level, causing it to have a great number of dimensions associated with it.

2. **Periodic snapshots**

- The periodic snapshot, as the name implies, takes a "picture of the moment", where the moment could be any defined period of time, e.g. a performance summary of a salesman

over the previous month. A periodic snapshot table is dependent on the transactional table, as it needs the detailed data held in the transactional fact table in order to deliver the chosen performance output.

3. accumulate snapshots

- This type of fact table is used to show the activity of a process that has a well-defined beginning and end, e.g., the processing of an order. An order moves through specific steps until it is fully processed. As steps towards fulfilling the order are completed, the associated row in the fact table is updated. An accumulating snapshot table often has multiple date columns, each representing a milestone in the process. Therefore, it's important to have an entry in the associated date dimension that represents an unknown date, as many of the milestone dates are unknown at the time of the creation of the row.

4. Temporal snapshots

- By applying temporal database theory and modeling techniques the *temporal snapshot fact table* allows to have the equivalent of daily snapshots without really having daily snapshots. It introduces the concept of time Intervals into a fact table, allowing to save a lot of space, optimizing performances while allowing the end user to have the logical equivalent of the "picture of the moment" they are interested in.

6. Design & Implement effective Physical data structure for data ware house

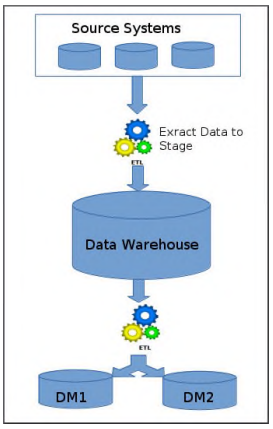
- The choice of an implementation approach is influenced by such factors as the current IT infrastructure, the resources available, the architecture selected, the scope of the implementation, the need for more business-wide data access across the organization, the return-on-investment requirements, and the speed of implementation. The selection of an implementation approach is a very important part of data warehouse design; the decision needs to be made at an early stage of the data warehouse project.

Implementation using design approach

- Data warehouse design is one of the key technique in building the data warehouse. Choosing a right data warehouse design can save the project time and cost. Basically there are two data warehouse design approaches are popular.
- There are two different Data Warehouse Design Approaches normally followed when designing a Data Warehouse solution and based on the requirements of your project you can choose which one suits your particular scenario. These methodologies are a result of research from Bill Inmon and Ralph Kimball.

Bill Inmon – Top-down Data Warehouse Design Approach

- "Bill Inmon" is sometimes also referred to as the "father of data warehousing"; his design methodology is based on a top-down approach. In the top-down approach, the data warehouse is designed first and then data mart are built on top of data warehouse.

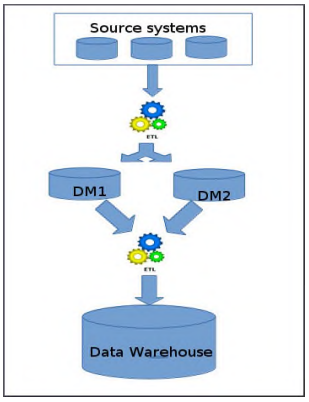


- The above image depicts how the top-down approach works.

- Below are the steps that are involved in top-down approach:
    - Data is extracted from the various source systems. The extracts are loaded and validated in the stage area. Validation is required to make sure the extracted data is accurate and correct. You can use the ETL tools or approach to extract and push to the data warehouse.
    - Data is extracted from the data warehouse in regular basis in stage area. At this step, you will apply various aggregation, summerization techniques on extracted data and loaded back to the data warehouse.
    - Once the aggregation and summerization is completed, various data marts extract that data and apply the some more transformation to make the data structure as defined by the data marts.

Ralph Kimball – Bottom-up Data Warehouse Design Approach

- Ralph Kimball is a renowned author on the subject of data warehousing. His data warehouse design approach is called dimensional modelling or the Kimball methodology. This methodology follows the bottom-up approach.
- As per this method, data marts are first created to provide the reporting and analytics capability for specific business process, later with these data marts enterprise data warehouse is created.



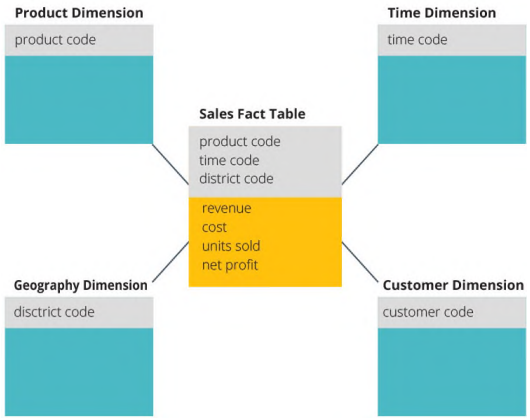
- The above image depicts how the bottom-up approach works.
- Basically, Kimball model reverses the Inmon model i.e. Data marts are directly loaded with the data from the source systems and then ETL process is used to load in to Data Warehouse.
- Below are the steps that are involved in bottom-up approach:

- The data flow in the bottom up approach starts from extraction of data from various source system into the stage area where it is processed and loaded into the data marts that are handling specific business process.
- After data marts are refreshed the current data is once again extracted in stage area and transformations are applied to create data into the data mart structure. The data is the extracted from Data Mart to the staging area is aggregated, summarized and so on loaded into EDW and then made available for the end user for analysis and enables critical business decisions.

Implementation using Dimensional Data Model

- Dimensional modelling is used in Data Warehouse to organize data effectively and assist analytical operations on huge volumes of data. The concept of Dimensional Modelling was developed by Ralph Kimball and is comprised of "fact" and "dimension" tables
- Data Warehouse uses denormalized tables (flat tables). (Whereas, database or OLTP systems use Normalized tables). So, to create this denormalized model effectively you will use a modelling technique called "Dimensional Modelling". A dimensional model is a data structure technique optimized for Data warehousing

Elements of Dimensional Data Model



Fact

Fact table consists of the measurements, metrics or facts of a business process. It is located at the center of a star schema or a snowflake schema surrounded by dimension tables

Eg. Monthly sales volume, Average Customer Balance etc...

A Fact Table contains

1. Measurements/facts
2. Foreign key to dimension table

Dimension

A category of information. For example, the time dimension. In simple terms, they give who, what, where of a fact. In the Sales business process, for the fact quarterly sales number, dimensions would be

- Who - Customer Names
- Where - Location
- What - Product Name
- When - Time Dimension

Dimension table properties:

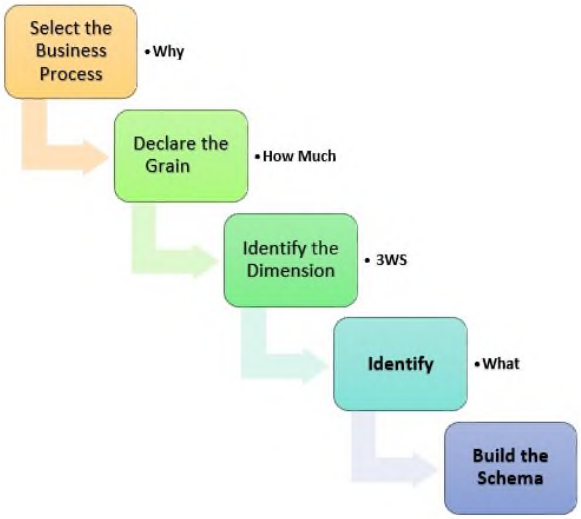
- A dimension table contains dimensions of a fact
- They are joined to fact table via a foreign key

- Dimension tables are de-normalized tables
- The dimension can also contain one or more hierarchical relationships

Attributes
The Attributes are the various characteristics of the dimension.In the Location dimension, the attributes can be
- State
- Country
- Zipcode etc

Attributes are used to search, filter, or classify facts. Dimension Tables contain Attributes

Steps of Dimensional Modelling
A good dimensional model determines the success of your data warehouse implementation
Here are the steps in dimenstional data modelling
1. Understanding the Business Problem [Identify Business Process]
2. Identify Grain (level of detail)
3. Identify Dimensions
4. Identify Facts
5. Build Star
   The model should describe the Why, How much, When/Where/Who and What of your business process



Step 1: Understanding the Business Problem
Identifying the actual business process a datawarehouse should cover. This could be Marketing, Sales, HR, etc. as per the data analysis needs of the organization. It is the most important step of the Data Modelling process, and a failure here would have cascading and irreparable defects

Step 2: Identify the grain
The Grain describes the level of detail for the business problem/solution. It is the process of identifying the lowest level of information for any table in your data warehouse. If a table contains sales data for every day, then it should be daily granularity. If a table contains total sales data for each month, then it has monthly granularity. During this stage, you answer questions like
- Do we need to store all the available products or just a few types of products?

- Do we store the product sale information on a monthly, weekly, daily or hourly basis? This decision depends on the nature of reports requested by executives
- How do the above two choices affect the database size?

Step 3: Identify the dimensions
Dimensions are nouns like date, store, inventory, etc. These dimensions are where all the data should be stored. For example, the date dimension may contain data like a year, month and weekday

Example of Dimensions:
The CEO at an MNC wants to find the sales for specific products in different locations on a daily basis
Dimensions: Product, Location and Time Attributes: For Product: Product key (Foreign Key), Name, Type, Specifications Hierarchies: For Location: Country, State, City, Street Address, Name

Step 4: Identify the Fact
This step is co-associated with the business users of the system because this is where they get access to data stored in the data warehouse. Most of the fact table rows are numerical values like price or cost per unit, etc.
Example of Facts:
The CEO at an MNC wants to find the sales for specific products in different locations on a daily basis
The fact here is Sum of Sales by product by location by time

Step 5: Build Schema
The Database schema is designed and finalized in this step. There are two popular schemas

1. **Star Schema**
   The star schema architecture is easy to design. It is called a star schema because diagram resembles a star, with points radiating from a center. The center of the star consists of the fact table, and the points of the star is dimension tables
   The fact tables in a star schema which is third normal form whereas dimensional tables are de-normalized

2. **Snowflake Schema**
   The snowflake schema is an extension of the star schema. In a snowflake schema, each dimension are normalized and connected to more dimension tables

Dimensional Modelling Rules
- Load atomic data into dimension tables
- Design dimensional models around business processes
- Ensure that every fact table has an associated dimension table
- Ensure that all facts in a single fact table are at the same grain or level of detail
- Need to ensure that dimension tables use a surrogate key

Benefits of dimensional modeling
- Bring together data from many different sources and create a single, consistent user view
- Support the ad hoc queries that arise from real business questions
- Maximize flexibility and scalability
- Optimize the end-user experience

Deference Between Logical & Physical Design :-

Logical design :
Logical design deals with the logical relationships between objects. Entity-relationship (ER) modeling technique can be used for logical design of data warehouse. ER modeling involves identifying the entities (important objects), attributes (properties about objects) and the relationship among them.

An entity is a chunk of information, which maps to a table in database. An attribute is a part of an entity, which maps to a column in database.
A unique identifier can be used to make sure the data is consistent.

Physical design:
Physical design deals with the effective way of storing and retrieving the data. In the physical design, the logical design needs to be converted into a description of the physical database structures.

Physical design involves creation of the database objects like tables, columns, indexes, primary keys, foreign keys, views, sequences etc.

- **Extraction Methods in Data Warehouse**

The extraction methods in data warehouse depend on the source system, performance and business requirements. There are two types of extractions, Logical and Physical.

1. **Logical extraction**

   There are two types of logical extraction methods:

   a) Full Extraction: Full extraction is used when the data needs to be extracted and loaded for the first time. In full extraction, the data from the source is extracted completely. This extraction reflects the current data available in the source system.

   b) Incremental Extraction: In incremental extraction, the changes in source data need to be tracked since the last successful extraction. Only these changes in data will be extracted and then loaded. These changes can be detected from the source data which have the last changed timestamp. Also a change table can be created in the source system, which keeps track of the changes in the source data.

   Physical extraction

   The data can be extracted physically by two methods:

   a) Online Extraction: In online extraction the data is extracted directly from the source system. The extraction process connects to the source system and extracts the source data.

   2) Offline Extraction: The data from the source system is dumped outside of the source system into a flat file. This flat file is used to extract the data. The flat file can be created by a routine process daily.

- **Types of Facts in Data Warehouse**

A fact table is the one which consists of the measurements, metrics or facts of business process. These measurable facts are used to know the business value and to forecast the future business. The different Four types of facts are explained in detail below.

1) Additive
Additive facts are facts that can be summed up through all of the dimensions in the fact table. A sales fact is a good example for additive fact.

2) Semi-Additive
Semi-additive facts are facts that can be summed up for some of the dimensions in the fact table, but not the others.
Eg: Daily balances fact can be summed up through the customers dimension but not through the time dimension.

3) Non-Additive
Non-additive facts are facts that cannot be summed up for any of the dimensions present in the fact table.
Eg: Facts which have percentages, ratios calculated.

4) Factless Fact Table
In the real world, it is possible to have a fact table that contains no measures or facts. These tables are called "Factless Fact tables".
Eg: A fact table which has only product key and date key is a factless fact. There are no measures in this table. But still you can get the number products sold over a period of time. A fact tables that contain aggregated facts are often called summary tables.

- **Defination of Normalization**

Normalization is the standard relational method of database organization. In this method, the rows are collapsed into a single row, hence it reduce space. Take a look at the following tables that show how normalization is performed.

Table before Normalization

Product_id	Qty	Value	sales_date	Store_id	Store_name	Location	Region
30	5	3.67	3-Aug-13	16	sunny	Bangalore	S
35	4	5.33	3-Sep-13	16	sunny	Bangalore	S
40	5	2.50	3-Sep-13	64	san	Mumbai	W
45	7	5.66	3-Sep-13	16	sunny	Bangalore	S

Table after Normalization

Store_id	Store_name	Location	Region
16	sunny	Bangalore	S
64	san	Mumbai	W

Product_id	Quantity	Value	sales_date	Store_id
30	5	3.67	3-Aug-13	16
35	4	5.33	3-Sep-13	16
40	5	2.50	3-Sep-13	64

45	7	5.66	3-Sep-13	16

Row Splitting

Row splitting tends to leave a one-to-one map between partitions. The motive of row splitting is to speed up the access to large table by reducing its size.

Note − While using vertical partitioning, make sure that there is no requirement to perform a major join operation between two partitions.

Definition Of Data Mart

A data mart is a subset of the data warehouse, which concentrates on a specific business unit. A data mart, may or may not derived from a data warehouse and is aimed at meeting an immediate requirement.

3 Creating ETL solution with SSIS Implementing Control Flow in SSIS

Topic Covered

1. Introduction to ETL with SSIS
2. Introduction to SQL Server Data Tools
3. Implementing data flow using SSIS
   - Configuring the Data Flow Sources
     - ♣ Transformations
       - ▪ Derived Column
       - ▪ Conditional Split
   - Destinations
4. Introduction to Control Flow
   - SSIS Tasks
5. SSIS Containers
   - For Loop Container
   - Foreach Loop Container
     - ♣ Foreach File enumerator
     - ♣ Foreach Item enumerator
     - ♣ Foreach SMO enumerator
     - ♣ Foreach Nodelist enumerator
     - ♣ Foreach ADO enumerator
     - ♣ Foreach ADO.NET Schema
     - ♣ Foreach From Variable enumerator
   - Sequence Container
6. Create Dynamic Packages Using Containers

Introduction to ETL with SSIS

- ETL stands for Extract, Transform and Load. It is a process in data warehousing to extract data, transform data and load data to final source. ETL covers a process of how the data are loaded from the source system to the data warehouse.
- Every data integration expert is familiar with the three basic steps of ETL:

- **Extract of Data:**
  - Extraction is the first step of ETL process.
  - First, data is *extract* from a variety of internal and external sources. These data sources may include CSV, JSON, and XML/HTML files; Excel spreadsheets; relational and non-relational databases; and data from websites obtained using APIs or web scraping tools.
  - A staging area is required during ETL load.
  - The source systems are only available for specific period to extract data. This period is less than the total data-load time. Therefore, staging area allows you to extract the data from the source system and keeps it in the staging area before the time slot ends.
  - Staging area is required when you want to get the data from multiple data sources together or if you want to join two or more systems together. For example, you will not be able to perform a SQL query joining two tables from two physically different databases.

- For these reason it is necessary to extract the relevant data from the operational database before bringing into the data warehouse. Many commercial tools are available to help with the extraction process for data junction is one of the commercial products.

- **Transform:**
  - Second, data is transform in order to make it suitable for, and compatible with, the schema of the target data warehouse. This may include; cleansing the data to remove duplicate or out-of-date entries; converting data from one format to another; joining and aggregating data; sorting and ordering data; and many more processes.
  - Transformation is the second step of ETL process where all collected data is been transformed into same format i.e. format can be anything as per our requirement before loading it to data-warehouse i.e. it may be data-type format, data merge format, splitting format, alphabet joining format, currency format etc.
  - In data transformation, you apply a set of functions on extracted data to load it into the target system. Data, which does not require any transformation known as direct move or pass through data.
  - You can apply different transformations on extracted data from the source system. For example, you can perform customized calculations. If you want sum-of-sales revenue and this is not in database, you can apply the SUM formula during transformation and load the data.
  - For example, if you have the first name and the last name in a table in different columns, you can use concatenate before loading.

  - **Cleansing**

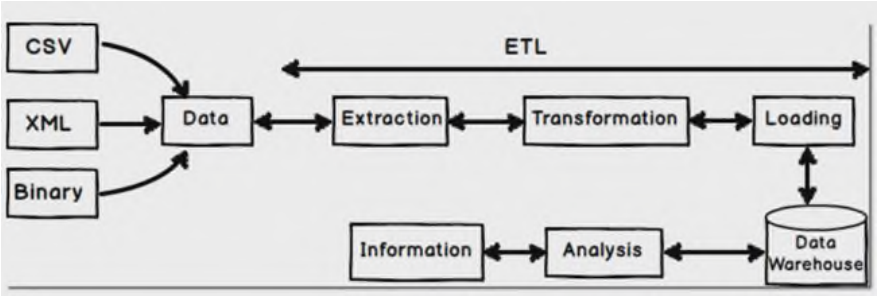
    - "Data cleaning is one of the three biggest problems in data warehousing."
    - Ralph Kimball also according to the survey "Data cleaning is the number one problem in data warehousing."
    - There are some important tasks defined as data cleaning tasks, they are like fill in missing values, identify outlier and smooth out noisy data, correct inconsistent data and to resolve redundancy caused by data integration.
    - Simply it can explained by the words "removal of noisy, inconsistent data".

- **Load:**
  - Third, and finally, data is loaded into the target data warehouse. Once inside the data warehouse, data can efficiently queried and used for business intelligence and analytics.
  - Final step of ETL process, the big chunk of data, which collected from various sources and transformed then, finally load to our data warehouse.
  - During Load phase, data is loaded into the end-target system and it can be a flat file or a Data Warehouse system.
  - Loading often implies physical movement of the data from the computer storing the source database to that which will store the data warehouse database.
  - This takes place immediately after the extraction phase.
  - The most common channel for data movement is a high-speed communication link.
  - E.g. Oracle warehouse builder is the API from oracle, which provides the features to perform the ETL task on oracle data warehouse.
  - Most DBMS provide a bulk loader function, the bulk loader provides the ability to load data more quickly as it takes a number of shortcuts, for exa: the bulk loader does not insert records into recovery log on a record-by-record basis.

Example

- Let us assume there is a manufacturing company having multiple departments such as sales, HR, Material Management, EWM, etc. All these departments have separate databases, which they use to maintain information related to their work and each database has a different technology, landscape, table names, columns, etc. Now, if the company wants to analyze historical data and generate reports, all the data from these data sources should be extracted and loaded into a Data Warehouse to save it for analytical work.

- **What is ETL Architecture?**
  - ETL architecture is a "blueprint" for how your ETL processes will execute from start to finish. This includes a description of how data will flow from the source to target locations, as well as a list of the transformations that will passed on this data.



- ETL architecture often consists of a diagram that outlines the flow of information from start to finish in your ETL pipeline. Data originates from sources files and databases, where it has taken up and entered into the ETL transformation engine. From here, the data may be loaded into any or all of the following locations:

- **Landing area:**
  The landing area is where data first arrives after being extract from a source location. Few, if any, transformations are apply to the data in the landing area. If you are performing ETL batch processing, the landing area may store multiple batches of data before moving it through the ETL pipeline.

- **Staging area:**
  The staging area is a temporary, intermediate location for performing ETL transformations. This area may take the form of a relational database, or binary or text files. The transformations performed in this area include joining and consolidating multiple data sources, cleansing and validating source data, and standardizing and aligning information.

- **Data warehouse area:**
  The data warehouse area is the final destination for data in an ETL pipeline. From here, your data can easily queried and analyzed in order to obtain valuable insights and make better business decisions. This area may consist of an enterprise data warehouse spanning entire organization, or a data mart that has been set up to serve the needs of a single team or department.

- To achieve **data integration** via ETL there are several **tools** which are use,
  1. SSIS
  2. Informatica
  3. Oracle Data Integrator
  4. AWS Glue

SSIS

- SSIS is a platform for data integration and workflow applications. It features a data-warehousing tool used for data extraction, transformation, and loading (ETL).
- SQL Server Integration Service (SSIS) is a component of the Microsoft SQL Server database software that can use to execute a wide range of data migration tasks. SSIS is a fast & flexible data-warehousing tool used for data extraction, loading and transformation like cleaning, aggregating, merging data, etc.
- It makes it easy to move data from one database to another database. SSIS can extract data from a wide variety of sources like SQL Server databases, Excel files, Oracle and DB2 databases, etc.
- SSIS also includes graphical tools & wizards for performing workflow functions like sending email messages, FTP operations, data sources, and destinations.
- Microsoft provides a suite of BI tools for data manipulation and reporting. Following are the main tools in the Microsoft BI Suite (MSBI),
  - **SSIS -SQL Server Integration Services**
  - **SSAS-SQL Server Analysis Services**
  - **SSRS-SQL Server Reporting Services**

SSIS Architecture

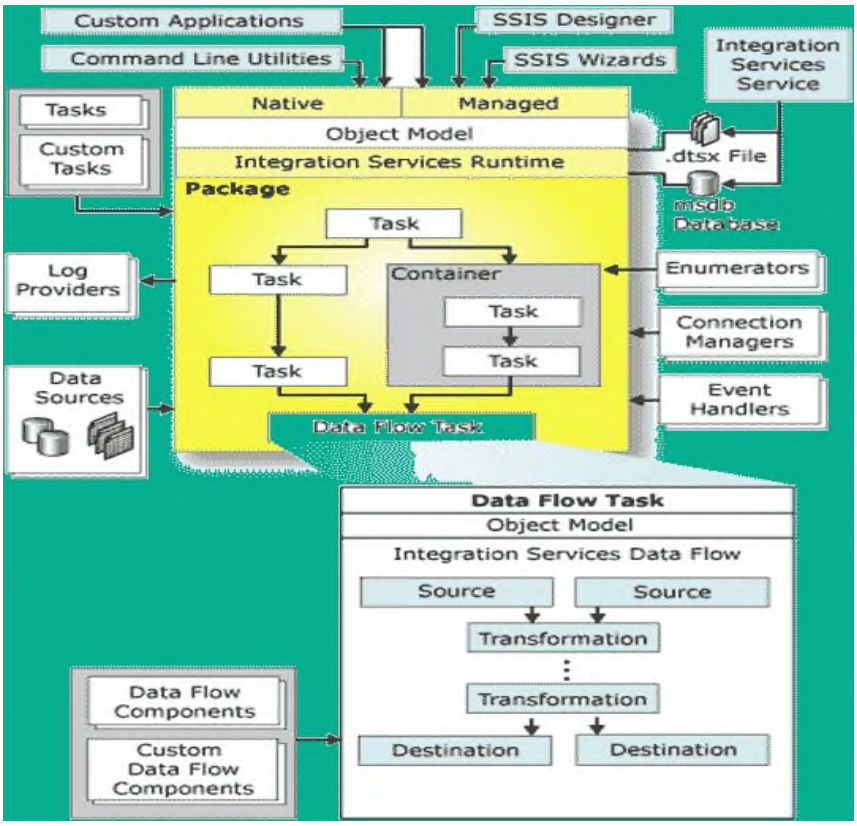


Figure :SSIS Architecture

Following are components of/ SSIS architecture:

- Control Flow (Stores containers and Tasks)
- Data Flow (Source, Destination, Transformations)
- Event Handler (sending of messages, Emails)
- Package Explorer (Offers a single view for all in package)
- Parameters (User Interaction)

Advantages and Disadvantages of using SSIS

SSIS tool the offers the following advantages:

- Broad documentation and support
- Ease and speed of implementation
- Tight integration with SQL Server and visual studio
- Standardized data integration
- Offers real-time, message-based capabilities
- Support for distribution model
- Helps you to remove network as a bottleneck for insertion of data by SSIS into SQL
- SSIS allows you to use the SQL Server Destination instead of OLE DB to load the data faster

Disadvantages of SSIS

Few drawbacks of using SIS tools are as follows:

- Sometimes create issues in non-windows environments
- Unclear vision and strategy
- SSIS lacks support for alternative data integration styles
- Problematic integration with other products

- To manage your database, you need a tool. Whether your databases run in the cloud, on Windows, on macOS, or on Linux, your tool does not need to run on the same platform as the database.

The following tools provide a graphical user interface (GUI).

RECOMMENDED TOOLS		
Tool	**Description**	**Operating system**
Azure Data Studio	A light-weight editor that can run on-demand SQL queries, view and save results as text, JSON, or Excel. Edit data, organize your favorite database connections, and browse database objects in a familiar object browsing experience.	**Windows macOS Linux**
SQL Server Management Studio(SSMS)	Manage a SQL Server instance or database with full GUI support. Access, configure, manage, administer, and develop all components of SQL Server, Azure SQL Database, and Azure Synapse Analytics.	**Windows**
SQL Server Data Tools (SSDT)	A modern development tool for building SQL Server relational databases, Azure SQL databases, Analysis Services (AS) data models, Integration Services (IS) packages, and Reporting Services (RS) reports. With SSDT, you can design and deploy any SQL Server content type with the same ease, as you would develop an application in **Visual Studio**.	**Windows**

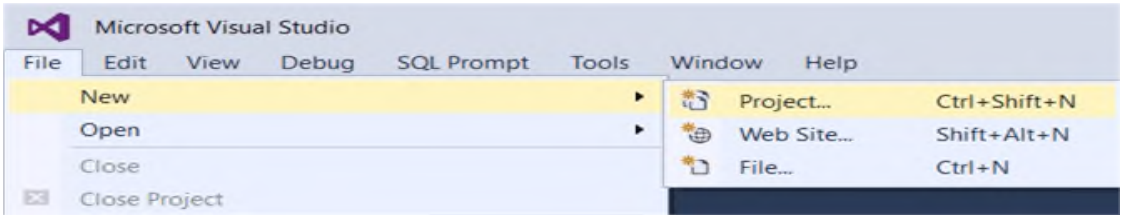
Visual Studio Code	The **mssql extension** for Visual Studio Code is the official SQL Server extension that supports connections to SQL Server and rich editing experience for T-SQL in Visual Studio Code. Write T-SQL scripts in a lightweight editor.	**Windows macOS Linux**

Introduction to SQL Server Data Tools [SSDT]

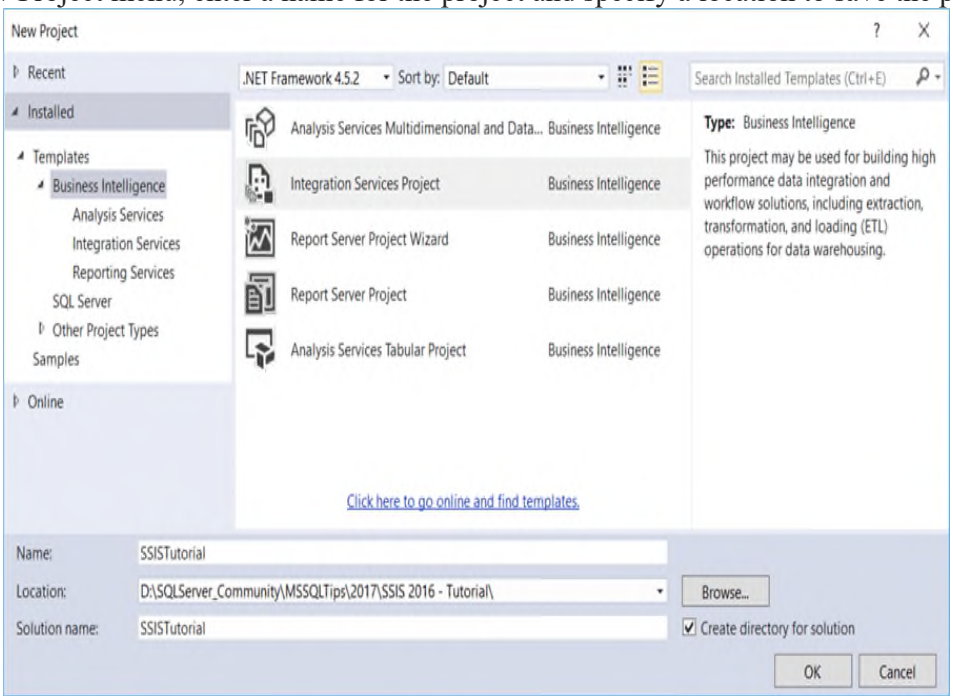
- Microsoft SQL Server Data Tools is introduce with SQL Server 2012 and is the successor of the Business Intelligence Development Studio (BIDS). It is a, Visual Studio based, integrated development environment, used for all your BI-related development.
- One of the biggest changes is the fact that now all the development can done in one environment. Where BIDS could 'only' be used for developing Integration Services, Analysis Services or Reporting Services solutions, and SQL Server Management Studio was needed for your database development, Data Tools makes it possible to also develop your database solutions in the same environment as your SSIS, SSAS and SSRS development.
- SQL Server Data Tools (SSDT) transforms database development by introducing a global, declarative model that spans all the phases of database development inside Visual Studio. You can use SSDT Transact-SQL design capabilities to build, debug, maintain, and refactor databases. You can work with a database project, or directly with a connected database instance on or off-premise.
- SSDT provides an integrated environment for database developers to carry out all their database design work for any SQL Server platform (both on and off premise) within Visual Studio. Database developers can use the SQL Server Object Explorer in VS to easily create or edit database objects and data, or execute queries.
- SQL Server Data Tools, SSDT, is a plugin to Visual Studio 2010 and later, with the purpose of enabling an integrated development environment, including database development in Visual Studio.
- The ultimate goal of SSDT is to create a more integrated development environment. Traditionally, database development has often been done outside of Visual Studio, for instance in SQL Server Management Studio. With SSDT, database development instead done inside Visual Studio. Databases are included as projects in the Visual Studio solutions.
- Using the Visual Studio SQL Server Data Tools, you can create databases in pretty much the same way you can in SQL Server Management Studio (SSMS).
- The release of SSDT provides a single environment hosted in Visual Studio, with database tooling that specifically targets the development process. Thus, you can now design and build your databases without constantly toggling between Visual Studio and other tools.

- **Creating a Project**

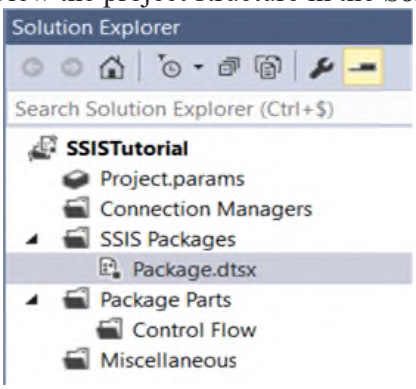
  Start Visual Studio. If it is the first time, you might get a prompt asking which settings Visual Studio should use. You can pick the Business Intelligence settings. When Visual Studio has started, go to File → New →Project.



In the New Project menu, enter a name for the project and specify a location to save the project.



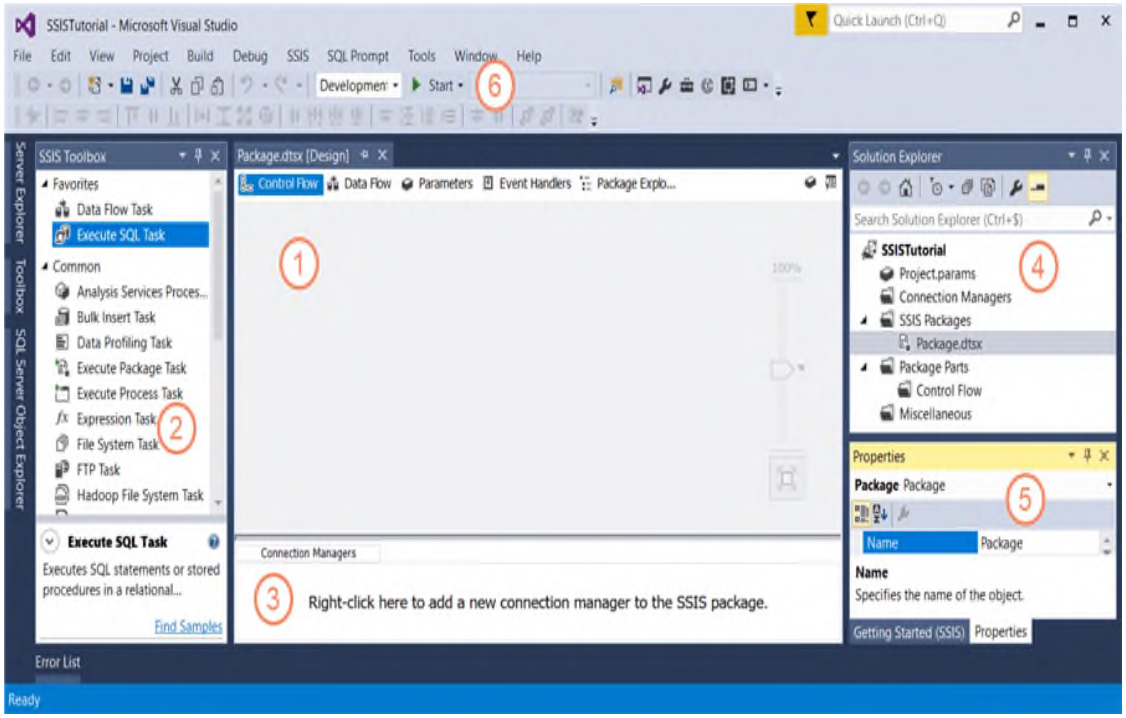
- When you create a project, Visual Studio will create a solution first and add the project to that solution. By default, the solution has the same name as the project.
- If you want to add multiple projects to one solution, you might want to change the solution name. If you have source control integrated into Visual Studio, you will have an extra checkbox asking you if you want to add the project to source control.
- When you click OK, the solution and the project will be created and an empty package will be added to the project. You can view the project structure in the **Solution Explorer** window:



- When there is only one project, the solution will not displayed.

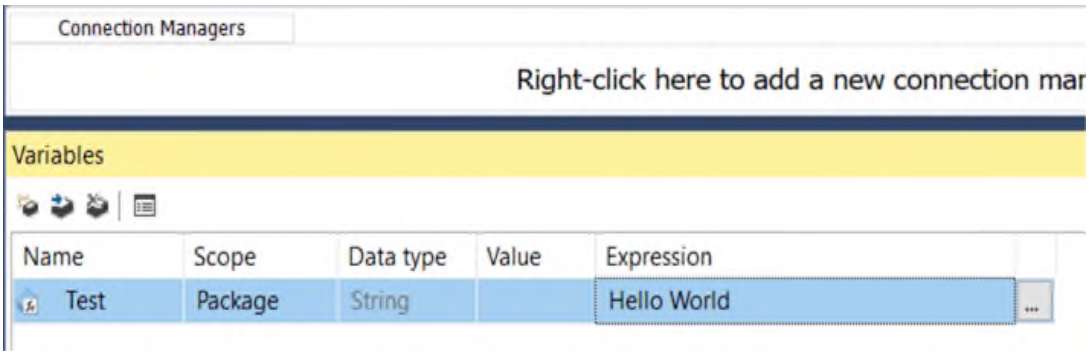
- **The SQL Server Data Tools Interface for Integration Services**

  Let's take a look at our development environment for creating SSIS packages. Keep in mind that most of the windows are dockable, which means you can move them around, so it's possible you do not have the exact same view as in this screenshot.



1. This is your canvas. Here you drag items from the toolbox and you connect them with each other to create a workflow. The package canvas has multiple tabs:
    1. The control flow. Here you can have multiple tasks, which you can connect with each other. The control flow is important as it defines what your package actually does.
    2. The data flow. This is a special task of the control flow. Here you move data around between sources and destinations, and you can transform the data while it is in memory.
    3. Parameters. You can define parameters to make your package more flexible.
    4. Event Handlers. This are special "control flow"-like canvasses where you can define tasks that will only execute if, a specific event occurs.
    5. Package Explorer. A tree-view of all the objects inside your package.
2. The SSIS Toolbox. Here you can find all tasks and transformations for the control and data flow. You can drag them from the toolbox into the canvas. There is also another window just called "Toolbox". It is used for other types of projects such as Reporting Services, so don't confuse it for the SSIS Toolbox. If you cannot find the SSIS Toolbox, right-click on the canvas and select *SSIS Toolbox* from the context menu.
3. The connection managers. A connection manager defines a connection to a specific object. This can be a flat file, a database, a folder and so on. Tasks and transformations use a connection manager to create a connection to the object.
4. The solution Explorer. A tree view of all the objects in the project or solution.
5. The properties window. Here you can view and change the properties of almost all objects within an SSIS package.
6. The toolbars. The most important item is the green arrow, which you can use to start the debugger. The debugger will execute the SSIS package within Visual Studio.

There's only one window missing from this view: the variables. When you create your first SSIS project, this window is hidden. You can right-click on the canvas and select Variables to open the window.



Variables are used to make your package more flexible and change properties on the fly when a package is running. The difference between parameters and variables is that parameters cannot change value once the package has started executing, while variables can. Parameters are use as input for the package before it starts.

Implementing data flow using SSIS

Adding a Data Flow task to the control flow of a package is the first step in implementing a data flow in a package. A package can include multiple Data Flow tasks, each with its own data flow. After the control flow includes a Data Flow task, you can begin to build the data flow that a package uses.

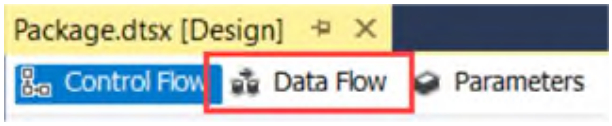
Creating a data flow includes the following steps:

- Adding one or more sources to extract data from files and databases, and add connection managers to connect to the sources.
- Adding the transformations that meet the business requirements of the package. A data flow is not required to include transformations.
- Some transformations require a connection manager. For example, the Lookup transformation uses a connection manager to connect to the database that contains the lookup data.
- Connecting data flow components by connecting the output of sources and transformations to the input of transformations and destinations.
- Adding one or more destinations to load data into data stores such as files and databases, and adding connection managers to connect to the data sources.
- Configuring error outputs on components to handle problems.
  For example, a data column with a string value cannot converted to an integer, or an expression tries to divide by zero. Both operations generate errors, and the rows that contain the errors can processed separately using an error flow.

- **Configuring the Data Flow**

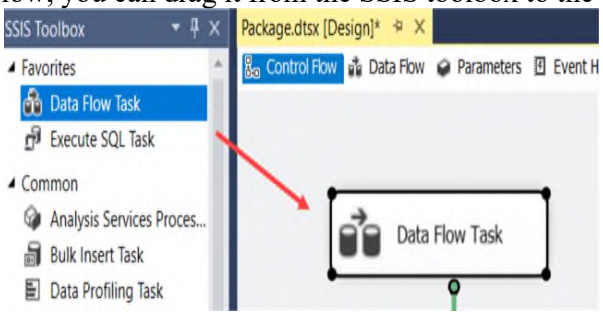
  When the **Data Flow** tab is active, the Toolbox contains the sources, transformations, and destinations that you can add to the data flow.

  - The data flow is a special task of the control flow. It needs a canvas of its own, so there is an extra tab for the data flow.



- The data flow is a construct where you can read data from various sources into the memory of the machine that is executing the SSIS package.
- While the data is in memory, you can perform different kinds of transformations. Because it is in memory, these are very fast. After the transformations, the data written to one or more destinations (a flat file, an Excel file, a database, etc.).
- In most cases, not all data read into the memory at once - although this is possible if you use certain kind of transformations – but the data read into buffers. Once a buffer filled by the source component, it is passed on to the next transformation, which does it logic on the buffer.
- Then the buffer passed to the following transformation and so on until it written to the destination. You can imagine the data flow is like a pipeline, with data flowing through.

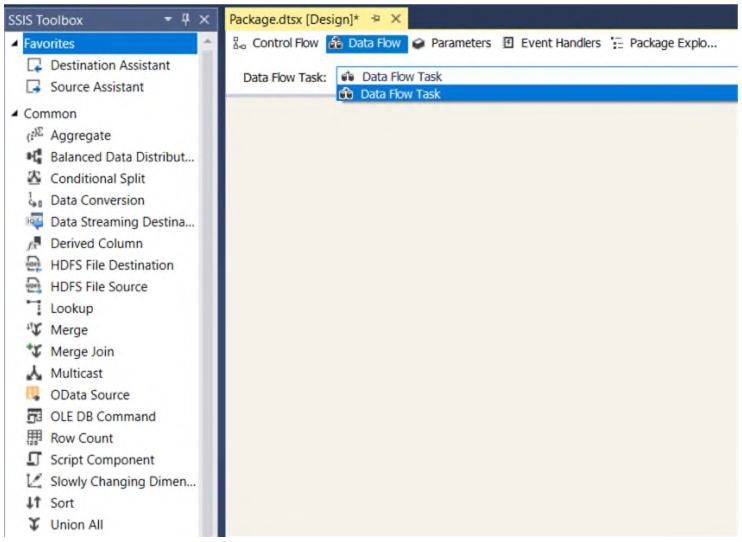
- To create a data flow, you can drag it from the SSIS toolbox to the control flow canvas.



Another option is simply go to the data flow tab, where you will received with the following message:

No Data Flow tasks have been added to this package. Click here to add a new Data Flow task.

- Clicking the link will create a new data flow task for you. You end up with an empty canvas, just like in the Control Flow.



- As you can see in the screenshot above, the SSIS toolbox will change once you go to the data flow canvas.
- All the tasks are now replace with transformations, sources and destinations for the data flow. At the top, you also have a dropdown box that lets you easily switch between multiple data flows if you have any.

- There are three types of SSIS data flow components: Sources, Transformations, and Destinations. Each data flow component has an output, and you can use the output to connect with other components. For instance, connect the source output to SSIS transformations and then connect transformation output to destination.
- While working with the SSIS data flow components, you always have to check the columns mappings and this is a place whether you might end up with errors. I mean wrong column mapping or no column mapping, etc.

➢ SSIS Data Flow Sources

The main motive of the SSIS package is to transfer data from heterogeneous sources to a destination. It means you need a Source to get
the data from and a Destination to load data into it. The following are the list of common sources used in the SSIS package development.

1. Source Assistant

The Source Assistance in SSIS will help you to choose the source from the available list. If you are confused to select your source or if you do not know which one to select, Source Assistance can be beneficial.

2. ADO.NET Source

The ADO.NET Source in SSIS (SQL Server Integration Services) is use to extract data from the Database using .Net provider. The ADO.NET Source in SSIS uses ADO.NET Connection Manager to connect with the Database.

3. Excel Source

The EXCEL Source in SSIS (SQL Server Integration Services) is use to extract data from the Excel worksheet. The SSIS Excel Source uses Excel Connection Manager in SSIS to connect with the Excel files.

4. OLE DB Source

The OLE DB Source in SSIS is use to extract data from a wide variety of databases that supports OLE DB Connections. The SSIS OLE DB Source uses the OLE DB Connection Manager to connect with the Database Tables or Views

5. ODBC

If you have an ODBC connection defined on your machine, you can use it to read or write data. Keep in mind Visual Studio is a 32-bit application, so you might want to have both 32-bit and 64-bit versions of the ODBC connection installed. You can give them the same name to make the transition smoother.

6. Flat File Source

A Flat File Source in SSIS is use to extract or reads data from text files. Flat File Source uses the Flat File Connection Manager to connect with the text files.

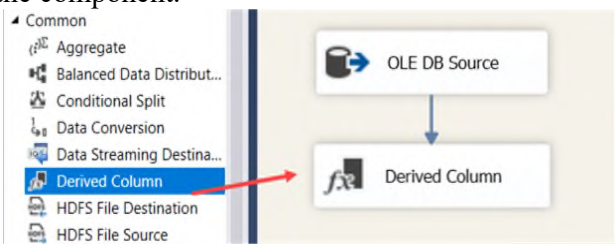
➢ SSIS Data Flow Transformations

Some of the more common transformations:

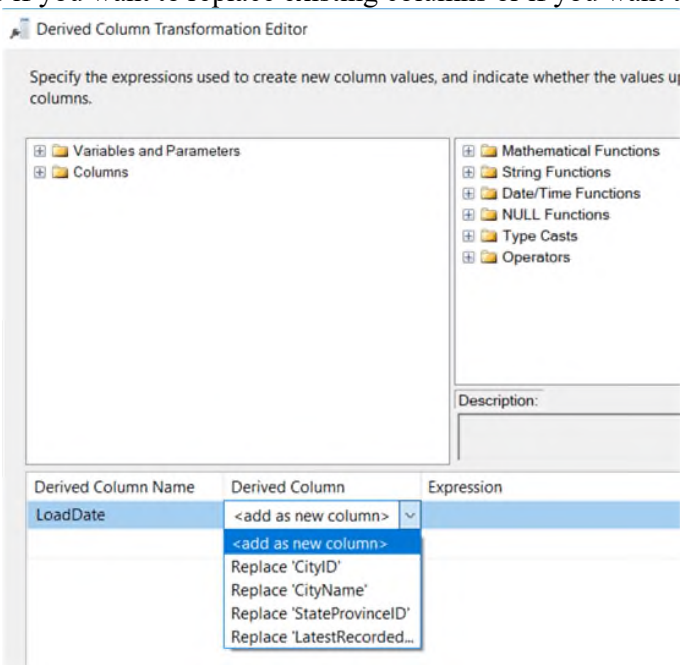
- **Aggregate:** Like a GROUP BY in T-SQL.
- **Conditional Split:** Splits out streams based on one or more conditions.

- **Data Conversion:** Allows you to convert columns from one data type to another.
- **Derived Column:** Allows you to manipulate existing columns or create new columns using expressions.
- **Lookup:** Similar to LOOKUP in Excel. You match a row against a reference data set and retrieve one or more columns.
- **Merge:** Merges two streams together. It needs sorted inputs.
- **Merge Join:** Like a JOIN in T-SQL, it can do inner, left join and full outer joins. It needs sorted inputs as well.
- **Multicast:** Duplicates a stream into multiple streams.
- **Script Component:** With the component, you can write your own transformations in .NET. When you open the editor, you have to choose if the component is a source, destination or a transformation.
- **Sort:** This transformation allows you to sort data and remove duplicates.
- **Union All:** Merges two streams, but does not need sorted input.

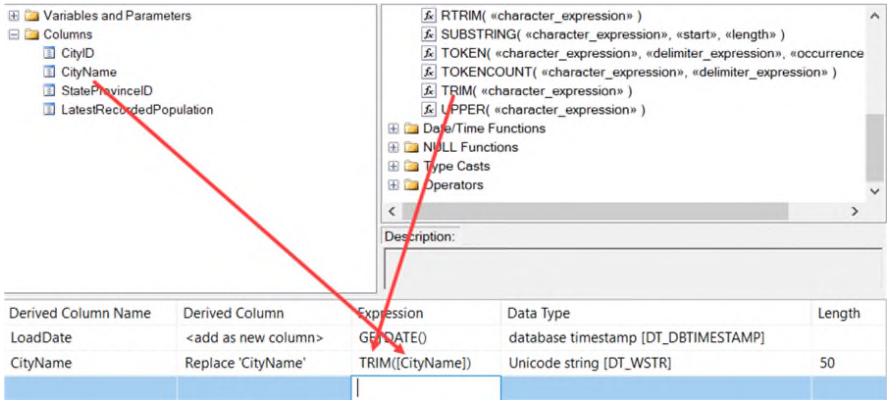
- **<u>SSIS Derived Column Transformation</u>**
- The Derived Column transformation has the ability to either modify existing columns or add new columns to the buffer. You can open the editor by double-clicking the component.



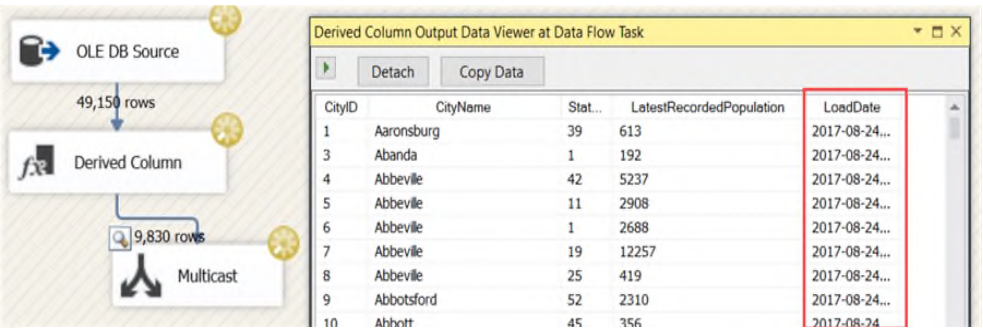
- In the editor, you can drag variables, parameters and existing columns to the expression. At the right, you have also a library of functions available. You can also choose if you want to replace existing columns or if you want to add a new column:



- Let's add a new column that contains the date of today (using the GETDATE() function) and trim the existing CityName column. You can drag the column name and the Trim function to the editor:

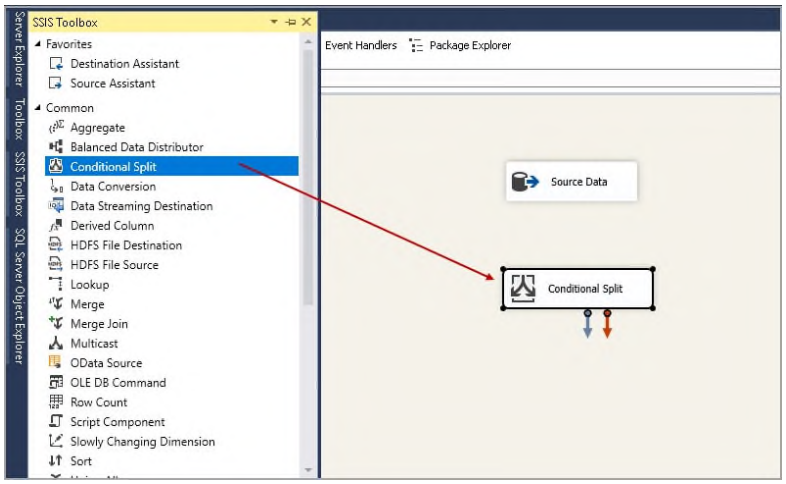


- The Derived Column transformation is very powerful, but the one-line expression editor can be frustrating. Let's add a multicast and a data viewer to inspect the results:

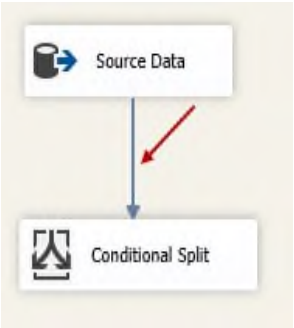


- **Conditional Split Transformation in SSIS**
- A Conditional Split Transformation in SSIS is just like the IF condition or CASE statement. It checks the given condition and based on the condition result, the output will send to the appropriate destination path. It has ONE input and MANY outputs.
- For example, if we want to store the students in a class who have marks greater than 40 in one Table and the students who score less than 40 to another table, then we can use this SSIS Conditional Split Transformation to split the data using the condition. Note that SSIS Conditional Split is **case-sensitive**.

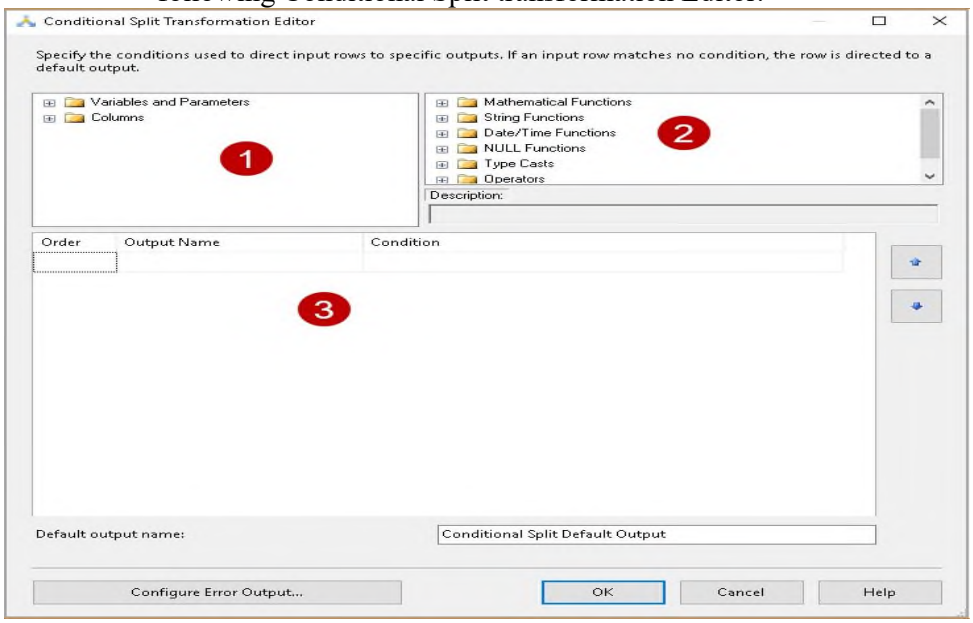
- **Add a SSIS Conditional Split Transformation to the Data Flow**
  - We need to add a Conditional Split Transformation task in the SSIS package to split the data. The Conditional Split Transformation task checks for the specified condition. It moves the data to an appropriate destination depending upon the condition.
  - Drag the SSIS Conditional Split task from the SSIS Toolbox.



- Now, Connect the Source Data (OLE DB Source) to the Conditional Split transformation task. To connect, drag the green arrow from the Source Data to the SSIS Conditional Split as shown in the following image.

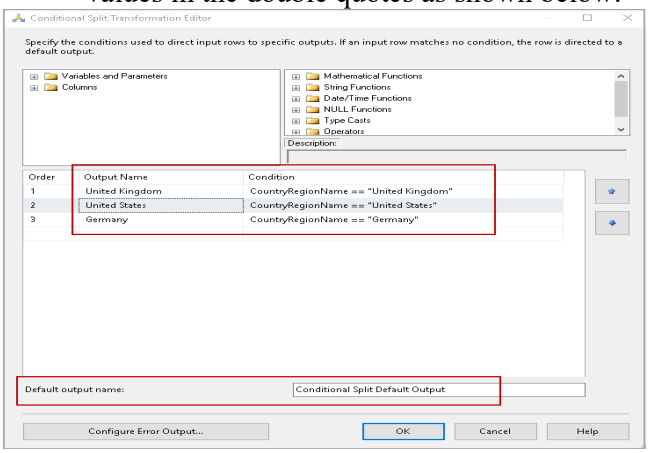


- Now, we need to define the conditions in the SSIS Conditional Split transformation task. Double click on the Conditional Split and it opens the following Conditional Split transformation Editor.



- **The Conditional Split transformation Editor has three sections.**

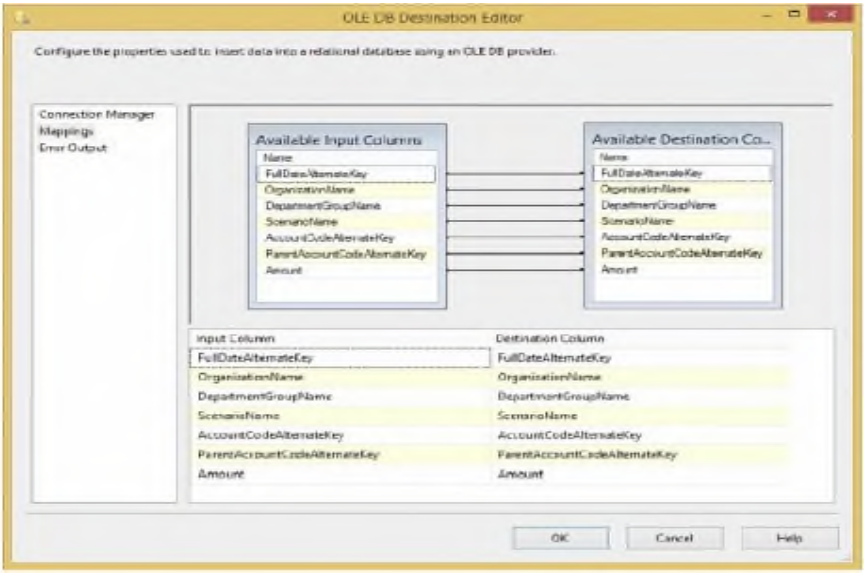
1. We can use system variables and query output columns in the expressions to split the data flow
2. We can define the various functions such as Mathematical, Date & Time function, Logical function, Null function
3. In this section, we define the conditions to define the split of data. We should be careful in defining the conditions and these conditions should not overlap with each other. If the data satisfy one, it should not satisfy other conditions
   - Specify the conditions in the SSIS Conditional Split transformation Editor. We need to use equal operator (==) in the split conditions and values in the double quotes as shown below.



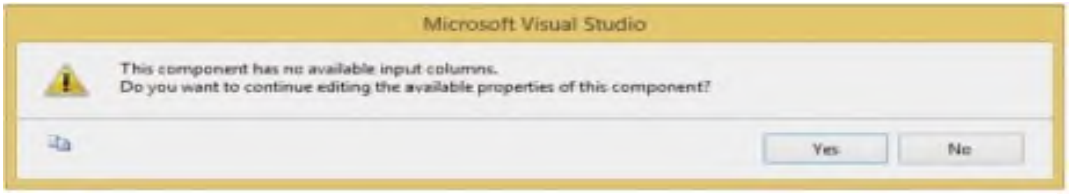
- The specified conditions are as follows:
  For United Kingdom employees, insert data into the table [United Kingdom]
  CountryRegionName == "United Kingdom"
  For United States employees, insert data into the table [United States]
  CountryRegionName == " United States "
  For Germany employees, insert data into the table [Germany]
  CountryRegionName == " Germany"
- The rest of the data (not satisfying above conditions) should go to the [default] table

➢ SSIS Data Flow Destinations

- The **Data Flow Destination Component** is an SSIS Data Flow Component for exposing data from a data flow to the Data Flow Source component.
- A destination is the data flow component that writes the data from a data flow to a specific data store, or creates an in-memory dataset. You can load data into flat files, process analytic objects, and provide data to other processes. You can also load data by accessing tables and views in databases and by running queries.
- Inside the Data Flow, destinations accept the data from the Data Sources and from the transformations.
- Destinations are manage through Connection Managers.
- The configuration difference between sources and destinations is that in destinations, you have a Mappings page, where you specify how the inputted data from the Data Flow map to the destination.
- As shown in the Mappings page in this figure, the columns are automatically map based on column names.



- Select the source or a transformation and drag the blue arrow to the destination. If you want to output bad data or data that has had an error to a destination, you would drag the red arrow to that destination. If you try to configure the destination before attaching it to the transformation or source, you will see the error in Figure below. In SQL Server 2014, you can still proceed and edit the component.

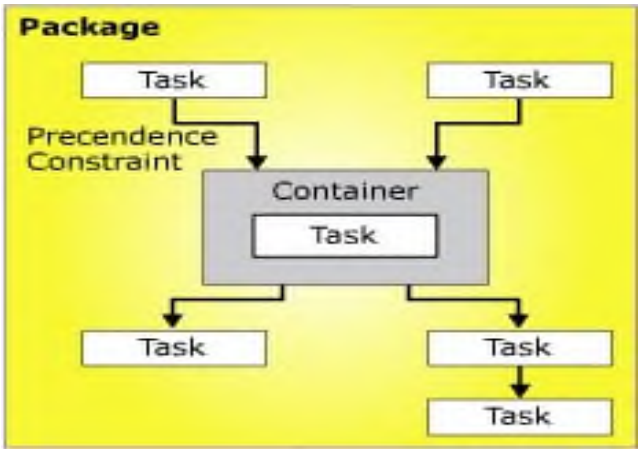


Types of Destination

No.	Name of Destination	Description
1.	**Excel Destination**	The Excel Destination is identical to the Excel Source except that it accepts data rather than sends data. To use it, first select the Excel Connection Manager from the Connection Manager page, and then specify the worksheet into which you wish to load data.
2.	**Flat File Destination**	The Flat File Destination sends data to a flat file, and it can be fixed-width or delimited based on the Connection Manager.
3.	**OLE DB Destination**	The SSIS OLE DB Destination is use to load data into a variety of database tables or views or SQL Commands. It can write data from the source or transformation to OLE DB–compliant Data Sources such as Oracle, DB2, Access, and SQL Server.
4.	**Raw File Destination**	The Raw File Destination is an especially speedy Data Destination that does not use a Connection Manager to configure. Instead, you point to the file on the server in the editor. Once written to, other packages can read the data in by using the Raw File Source.
5.	**Recordset Destination**	The Recordset Destination populates an ADO recordset that can be use outside the transformation.
6.	**DataReader Destination**	The DataReader Destination provides a way to extend SSIS Data Flows to external packages or programs that can use the DataReader interface, such as a .NET application.

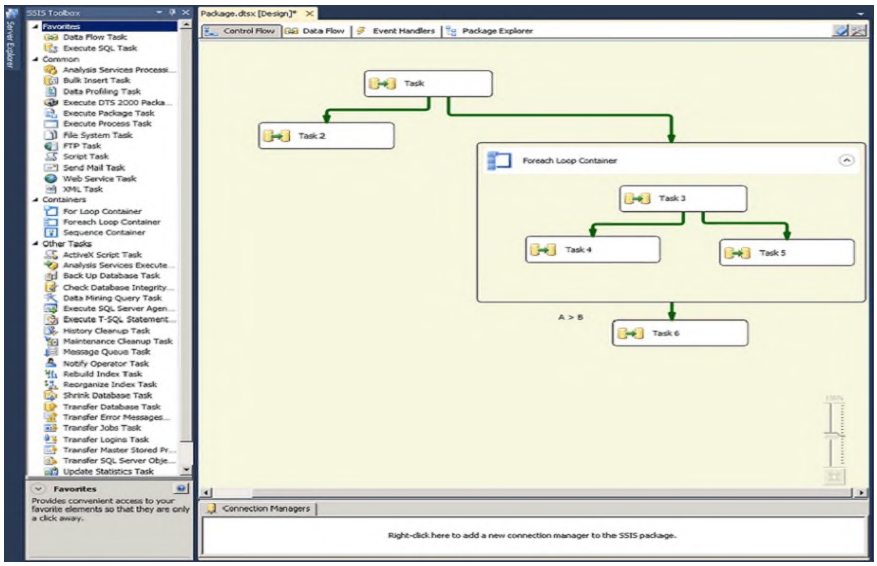
Introduction of Control Flow Using SSIS

- Data **flows** between your selected entities (sources, transformations, destinations). Moreover within a data **flow** task, you cannot perform tasks such as iteration, component execution, etc. A **control flow** defines a workflow of tasks to executed, often a particular order (assuming your included precedence constraints)
- A package consists of a control flow and, optionally, one or more data flows. SQL Server Integration Services provides three different types of control flow elements: **containers** that **provide structures in packages**, tasks that provide functionality, and precedence constraints that connect the executables, containers, and tasks into an ordered control flow.
- The following diagram shows a control flow that has one container and six tasks. Five of the tasks are define at the package level, and one task is define at the container level. The task is inside a container.



Control Flow Implementation

- You create the control flow in a package by using the **Control Flow** tab in SSIS Designer. When the **Control Flow** tab is active, the Toolbox lists the tasks and containers that you can add to the control flow.
- The following diagram shows the control flow of a simple package in the control flow designer. The control flow shown in the diagram made up of three package-level tasks and one package-level container that contains three tasks. The tasks and container are connect by using **precedence constraints.**



Creating a control flow includes the following tasks:

- Adding containers that implement repeating workflows in a package or divide a control flow into subsets.
- Adding tasks that support data flow, prepare data, perform workflow and business intelligence functions, and implement script.
- Integration Services includes a variety of tasks that you can use to create control flow that meets the business requirements of the package. If the package has to work with data, the control flow must include at least one Data Flow task. For example, a package might have to extract data, aggregate data values, and then write the results to a data source. For more information, see Integration Services Tasks and Add or Delete a Task or a Container in a Control Flow.
- Connecting containers and tasks into an ordered control flow by using precedence constraints.
- After you add a task or container to the design surface of the **Control Flow** tab, SSIS Designer automatically adds a connector to the item. If a package includes two or more items, tasks or containers, you can join them into a control flow by dragging their connectors from one item to another.
- The connector between two items represents a precedence constraint. A precedence constraint defines the relationship between the two connected items. It specifies the order in which tasks and containers are execute at run time and the conditions under which tasks and containers run. For example, a precedence constraint can specify that a task must succeed for the next task in the control flow to run.
- Adding connection managers.
- Many tasks require a connection to a data source, and you have to add the connection managers that the task requires to the package. Depending on the enumerator type it uses, the Foreach Loop container may also require a connection manager. You can add the connection managers as you construct the control flow item by item or before you start to construct the control flow.
- SSIS Designer also includes many design-time features that you can use to manage the design surface and make the control flow self-documenting.
- The Integration Services architecture supports the nesting of containers, and a control flow can include multiple levels of nested containers. For example, a package could contain a container such as a Foreach Loop container, which in turn could contain another Foreach Loop container and so on.
- Event handlers also have control flows, which built using the same kinds of control flow elements.

The control flow allows you to execute different tasks and organize a workflow between the tasks.

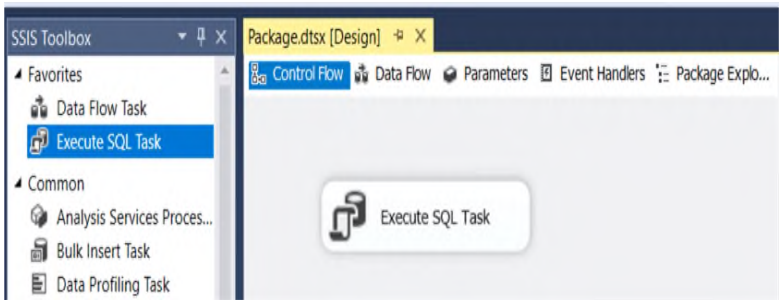
- **<u>SSIS Tasks</u>**

In an SSIS package, you can add tasks to the control flow. A task is a unit of work and you have different kinds of tasks to perform different kinds of work. Explaining all tasks would take us too far in this tutorial, so here is an overview of some of the most common tasks:

- **Execute SQL Task:** These tasks will execute a SQL statement against a relation database.
- **Data Flow Task:** These special tasks can read data from one or more sources transform the data while in memory and write it out against one or more destinations.
- **Analysis Services Processing Task:** You can use this task to process objects of an SSAS cube or Tabular model.
- **Execute Package Task:** With this task, you can execute other packages from within the same project. You can also pass variable values to the called package.

- **Execute Process Task:** Allows you to call an executable (.exe). You can specify command line parameters. With this task, you can for example unzip files; execute batch scripts and so on.
- **File System Task:** This task can perform manipulations in the file system, such as moving files, renaming files, deleting files, and creating directories etc.
- **FTP Tasks:** Allows you to perform basic FTP functionalities. However, this task is limited because it does not support FTPS or SFTP.
- **Script Task:** This task is essentially a blank canvas. You can write .NET code (C# or VB) that performs any task you want.
- **Send Mail Task:** Here you can send an email. Ideal for notifying users that your package has done running or that something went wrong.

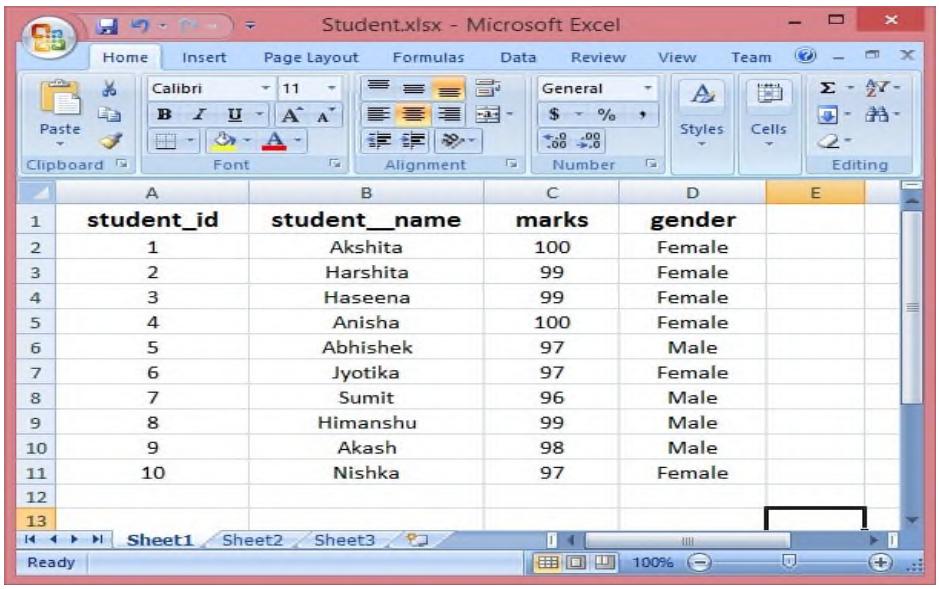
In the screenshot, you can see an Execute SQL Task that has been added to the control flow:



ETL process with SSIS Step by Step using example

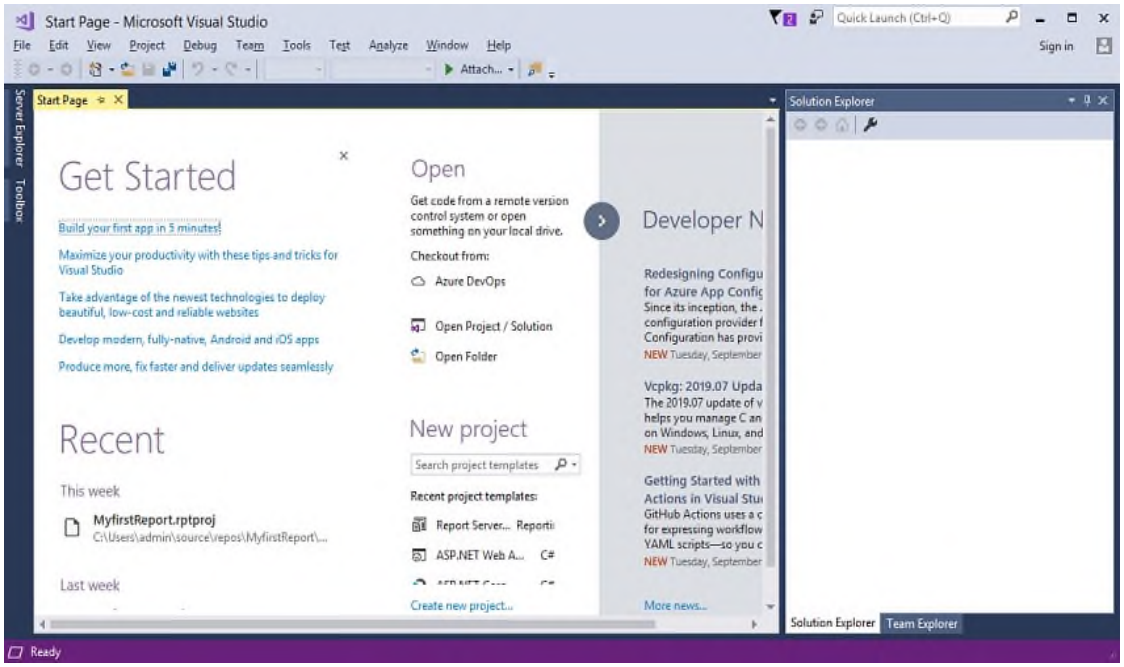
Perform ETL process to load excel sheet to SQL server table

Step 1: First, we create an excel file in Microsoft Excel. Create the student worksheet in Microsoft Excel, and fields in student table are student_id, student_name, marks, and gender.

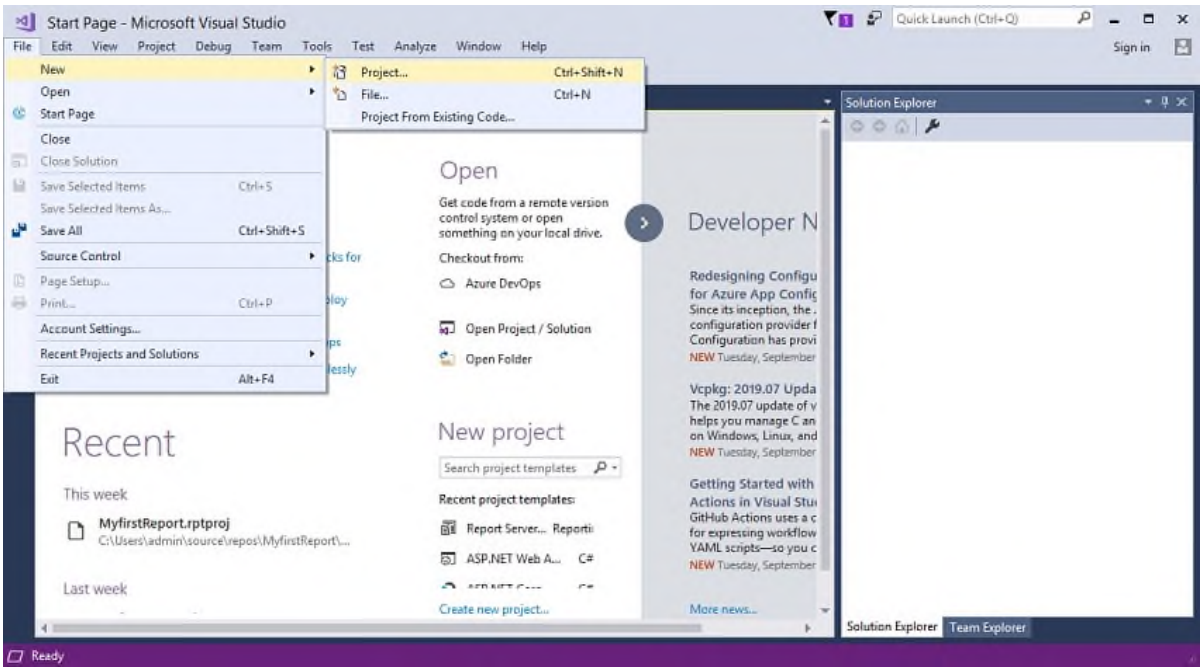


	A	B	C	D
1	student_id	student__name	marks	gender
2	1	Akshita	100	Female
3	2	Harshita	99	Female
4	3	Haseena	99	Female
5	4	Anisha	100	Female
6	5	Abhishek	97	Male
7	6	Jyotika	97	Female
8	7	Sumit	96	Male
9	8	Himanshu	99	Male
10	9	Akash	98	Male
11	10	Nishka	97	Female

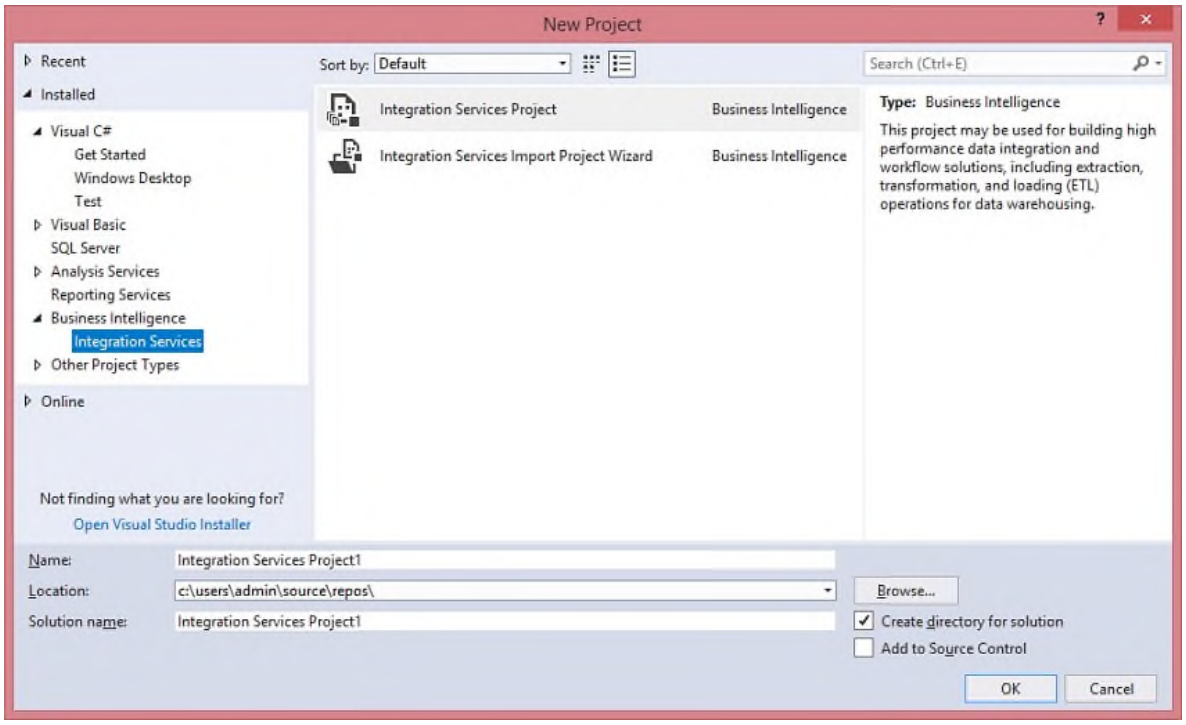
Step 2: Open the visual studio (SQL Server Data Tools).



Step 3: Click on the File option and then move to New->Project.

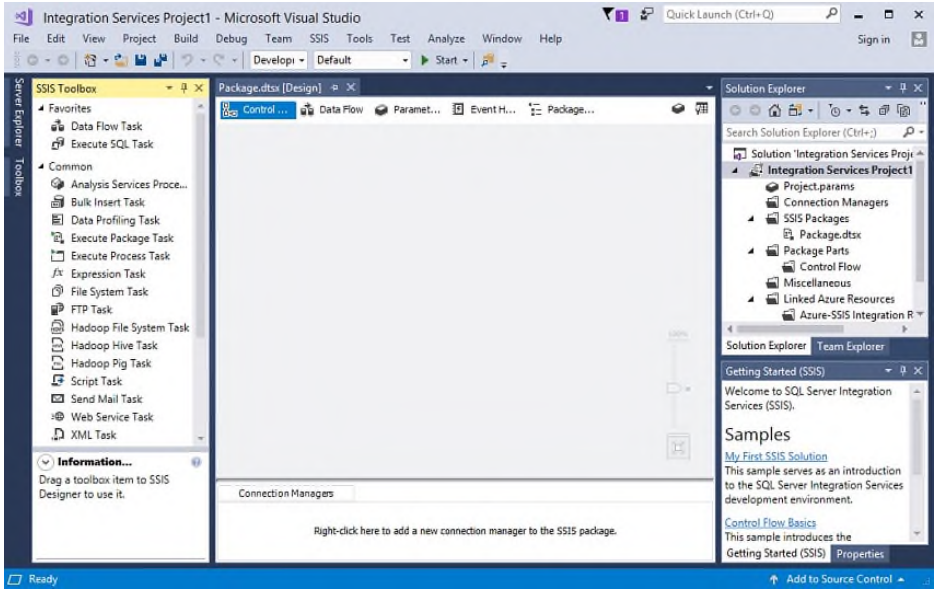


Step 4: When you click on the Project, the screen appears which is shown as below:



In the above screen, click on the Integration Services appearing at the leftmost side of the panel and then click on the Integration Services Project.

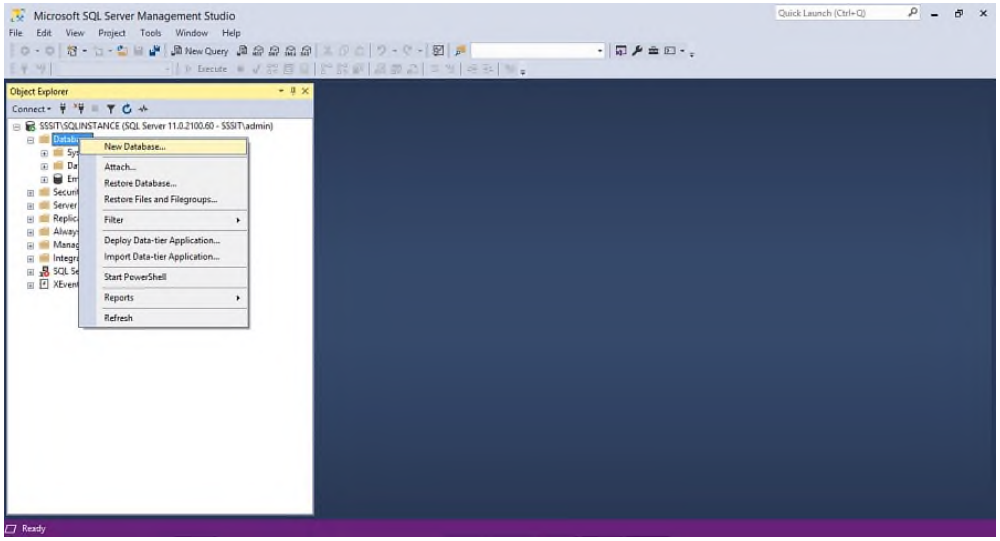
Step 5: After clicking on **the Integration Services Project** option, the screen appears shown as below:



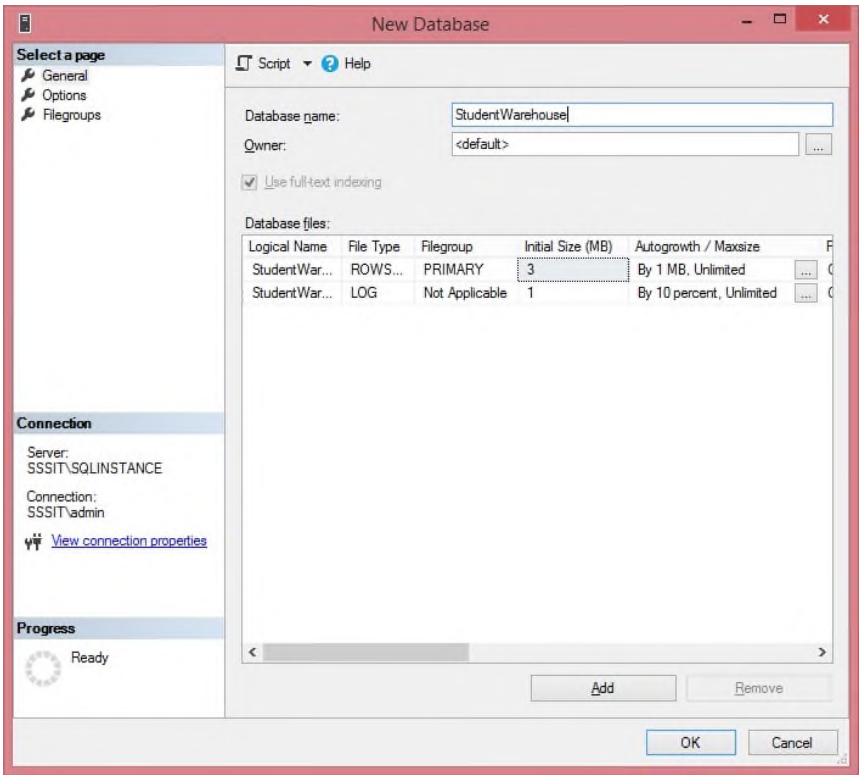
The above screen consists of five tabs such as Control flow, Data flow, Parameters, Event Handler, and Package Explorer. We have already discussed the Control flow and Data flow, both the control flow and data flow exist inside the SSIS package.

Step 6: To import the data, we need first to create the database in which we create the table in SQL Server database. As we know that student data is available inside the Excel file, and we want to import the data in SQL Server database. In order to achieve this, we need first to create the table in SQL Server. **Open the SQL Server Management studio.**

Step 7: Create the **student** database. Right-click on the database, and then click on the **New Database** option.



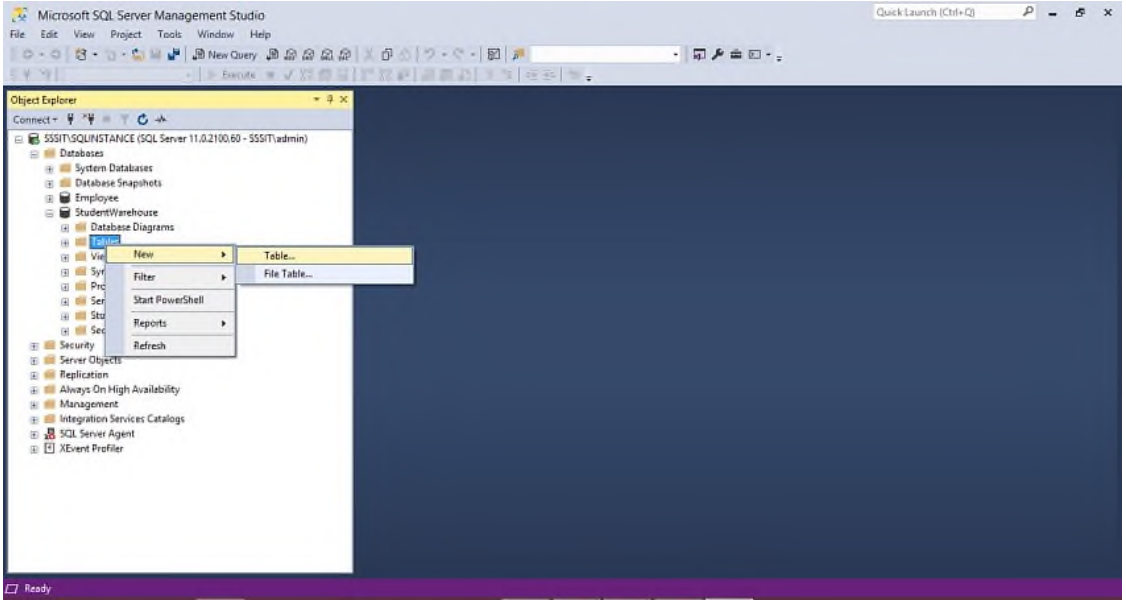
Step 8: Enter the database name in the below screenshot:



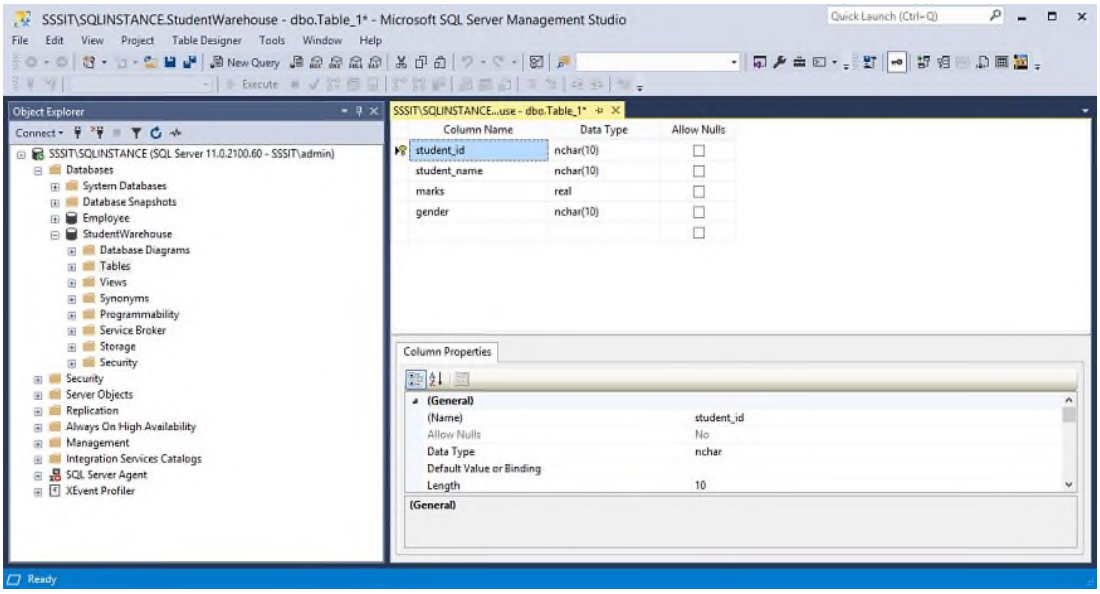
I have given the **StudentWarehouse** as a database name.

Step 9: Click on the **OK** button.

Step 10: Now, we create the table in a **StudentWarehouse** database. Right-click on the Tables, and then click on the New Table.

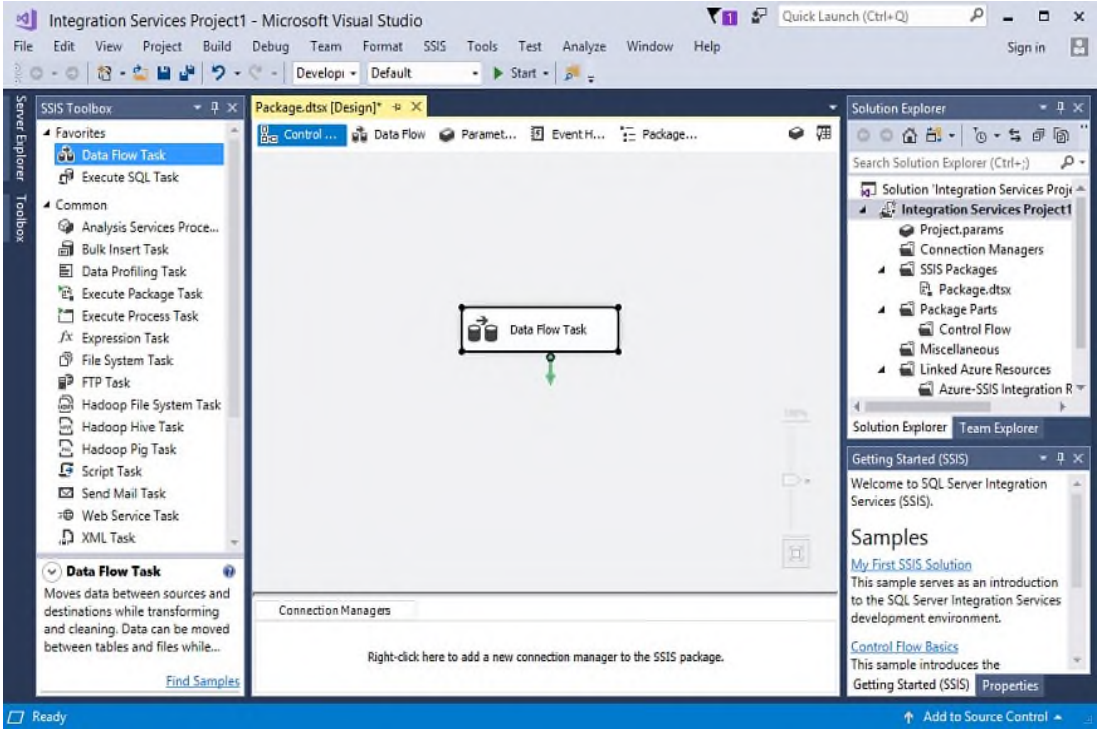


Step 11: Now we add the fields in a table such as **student_id, student_name, marks**, and **gender**. We set the **student_id** as a primary key, which uniquely identifies each row.



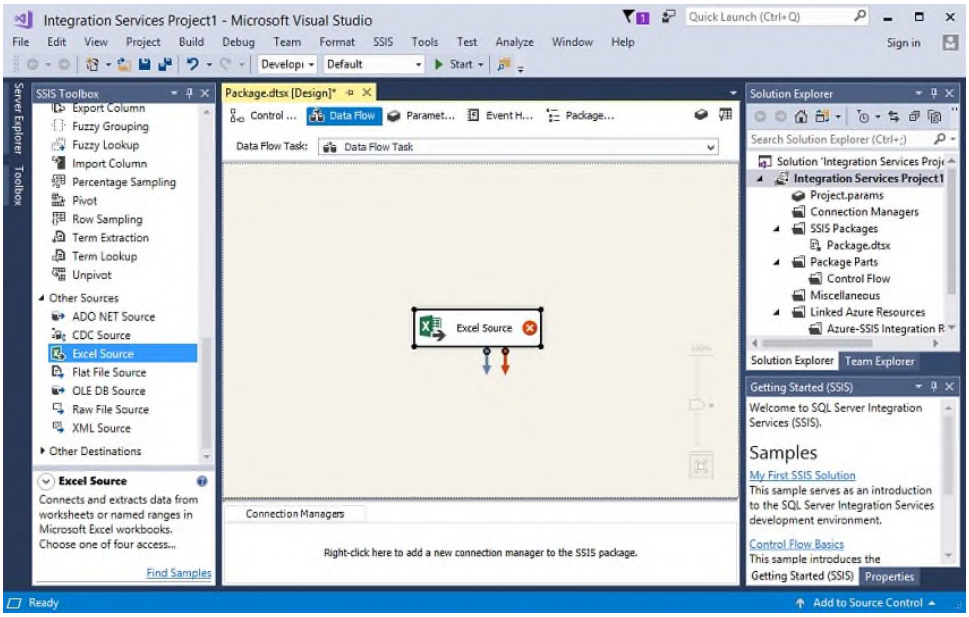
The above screen shows that we add four fields in a table. After adding all the fields in a table, we save the table by providing some useful name. Suppose I have provided the **student** as a table name.

Step 12: Move back to the visual studio. Control flow contains the Data Flow Task, drag and drop the Data Flow Task on the main window.



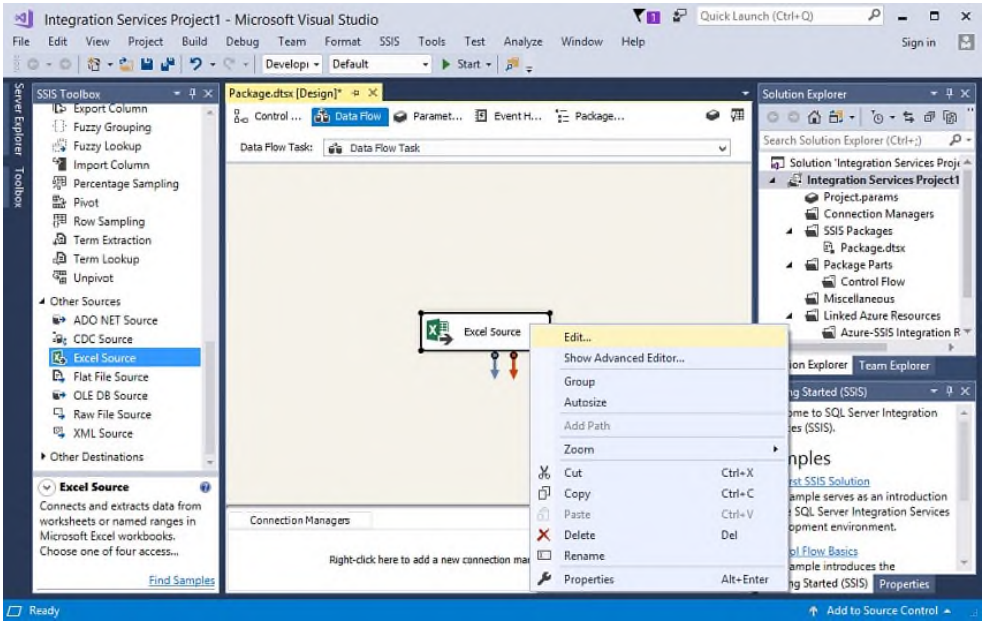
Step 13: When we double-click on the Data Flow Task, the control automatically moves to the Data Flow from the Control Flow, so we can say that Control Flow is a container of Data Flow.

Step 14: Now, we can perform the transformations in Data Flow. We want to extract the data from Excel file that we created earlier, so drag and drop the **Excel source** from the other sources appearing at the leftmost side of the panel.

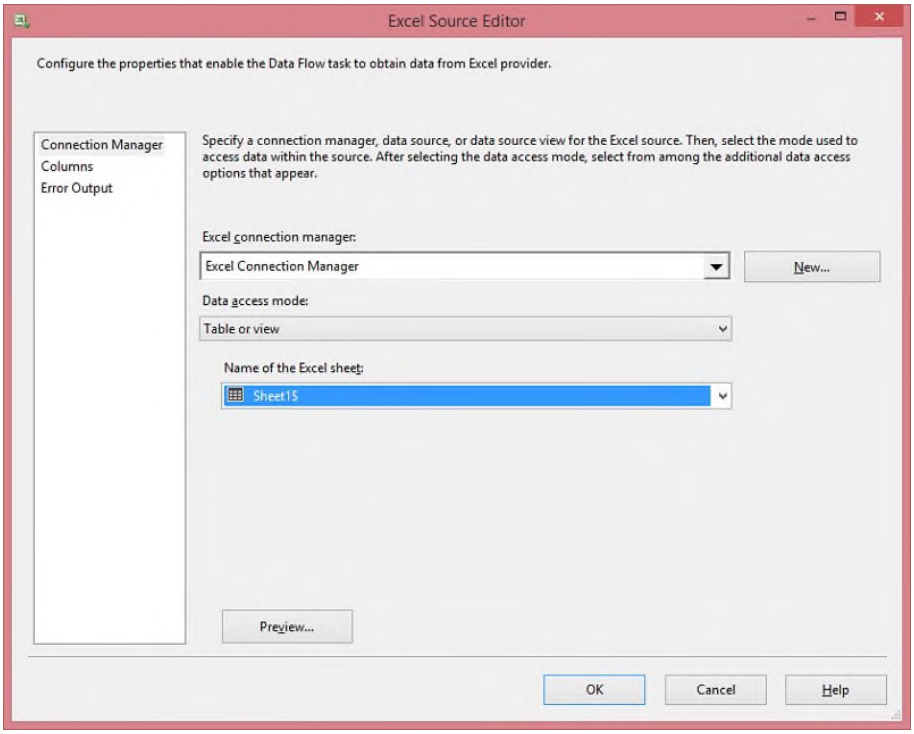


From the above screen, we observe that the red cross appears inside the Excel Source, it means that the component is configured with the Excel Source.

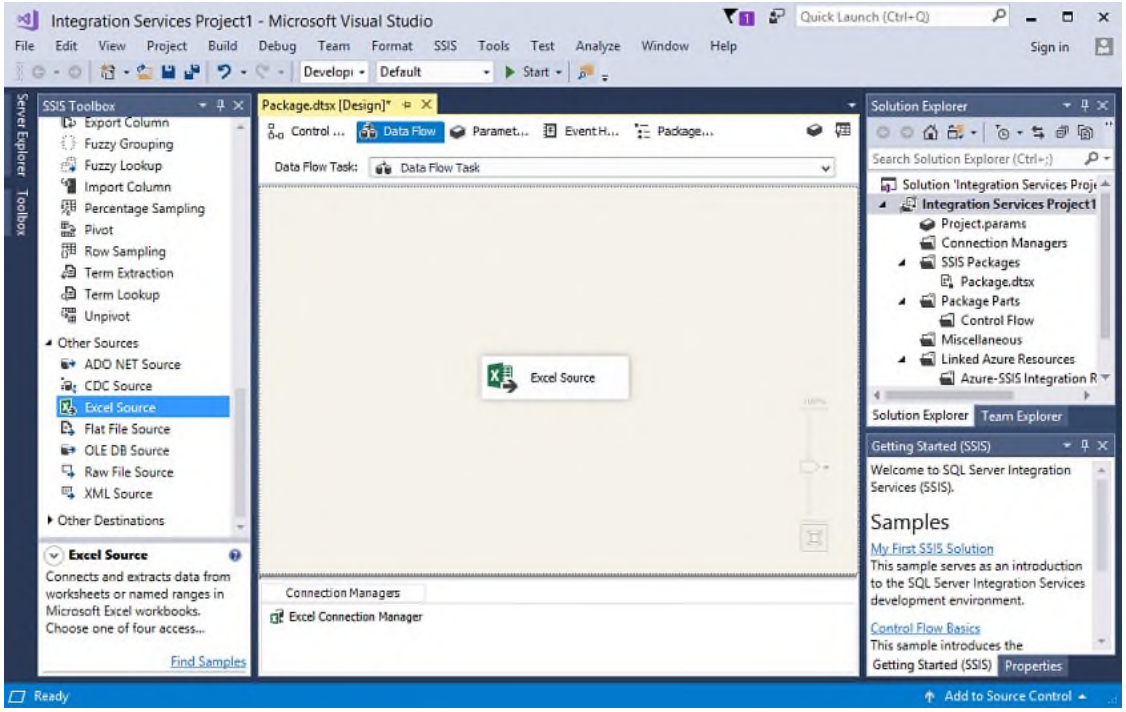
Step 15: To configure the component, right-click on the Excel source and click on the Edit option as shown in the below screenshot:



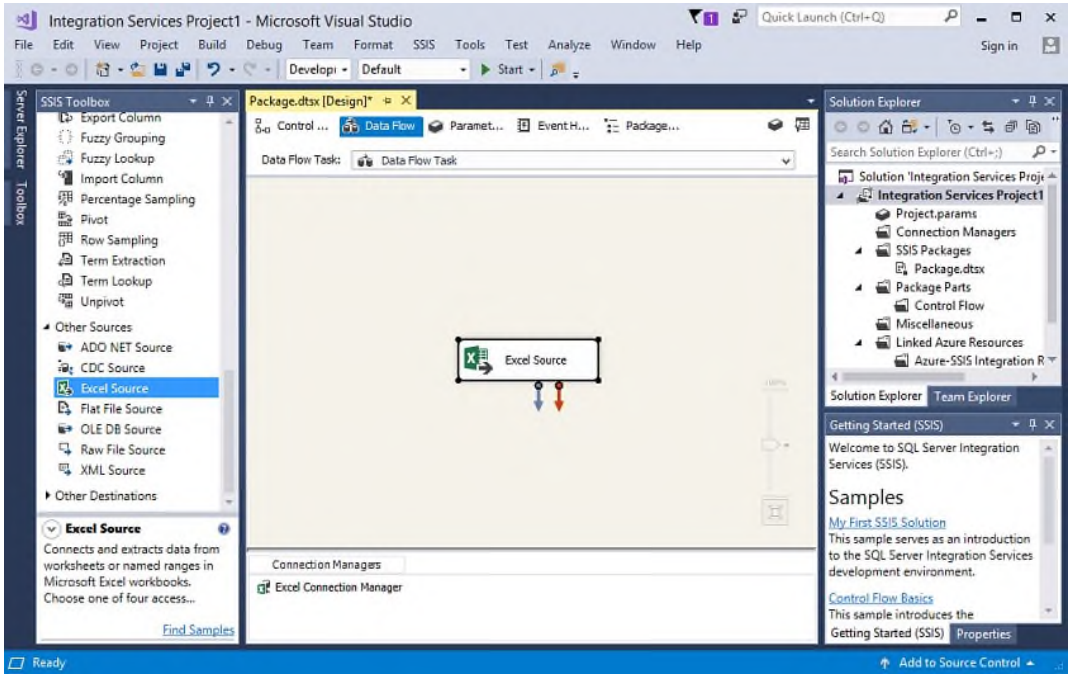
Step 16: Fill the details shown in the below screenshot:



Step 17: After entering all the details, the screen appears shown as below:

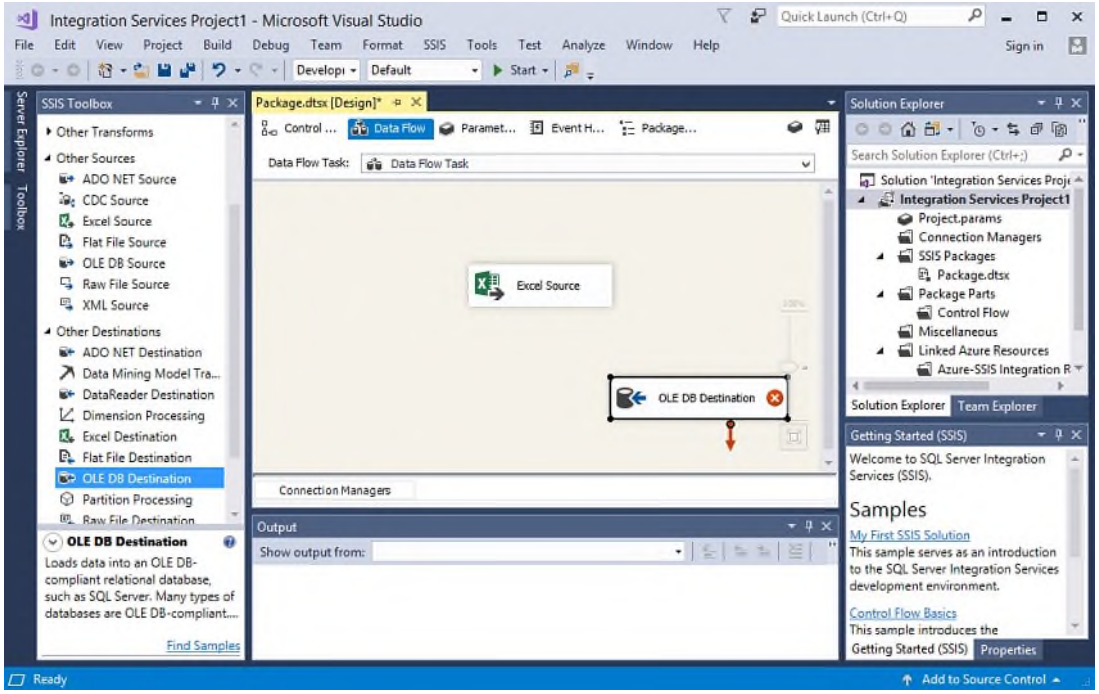


Step 18: Click on the Excel Source.

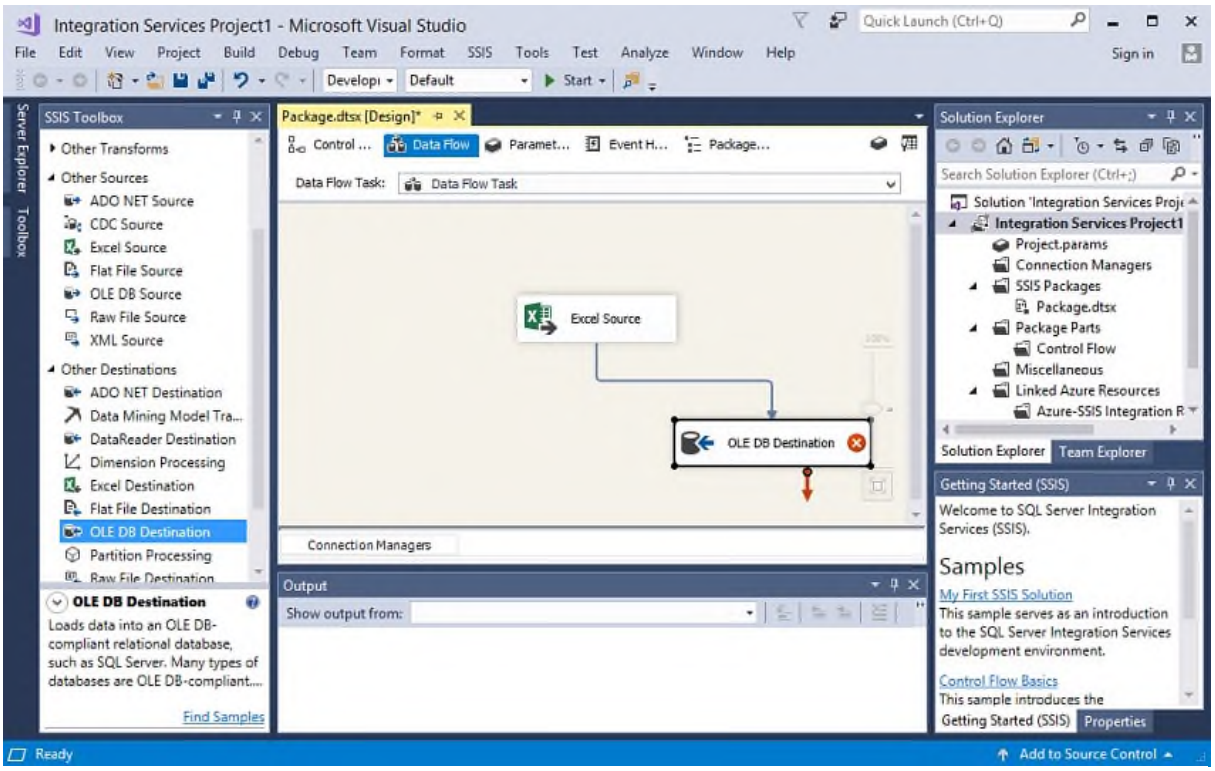


The above screen shows two arrows, red and blue arrow. The blue arrow defines the actual data that we receive from the Excel file, and the red arrow denotes the errors that we have. Now we have to perform the transformations, and before performing the transformations, we need to load the database.

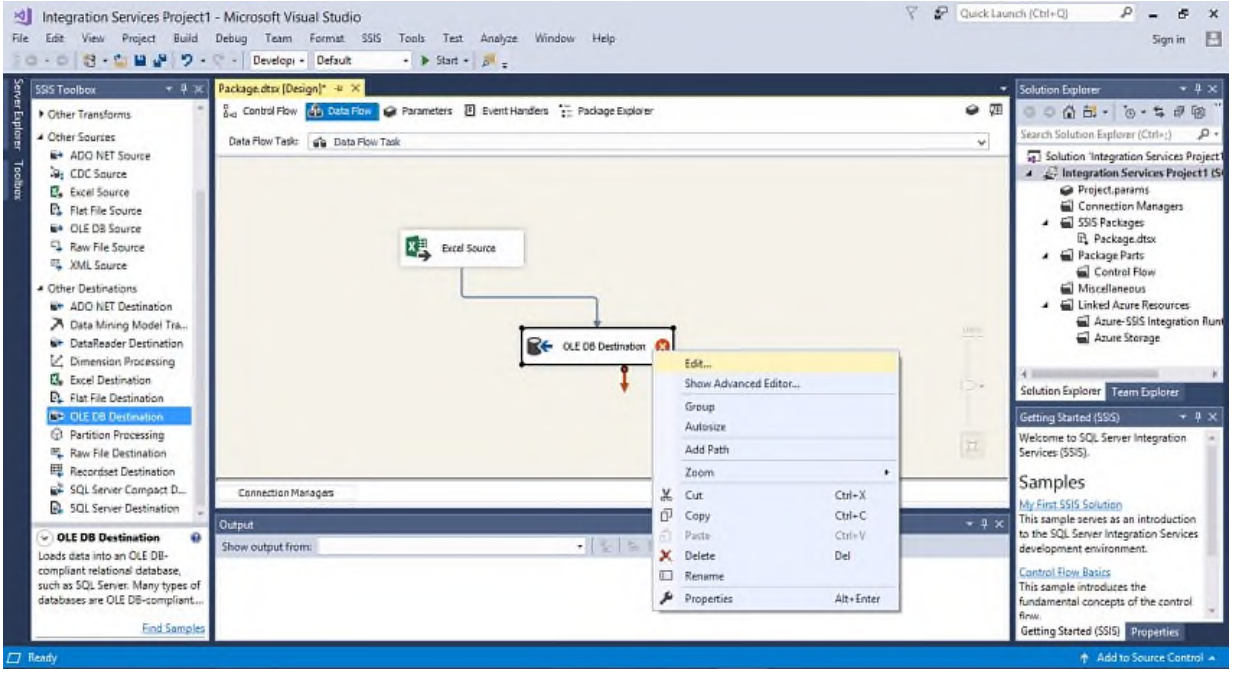
Step 19: To load the database, click on the Other Destinations and then click on the Destination.



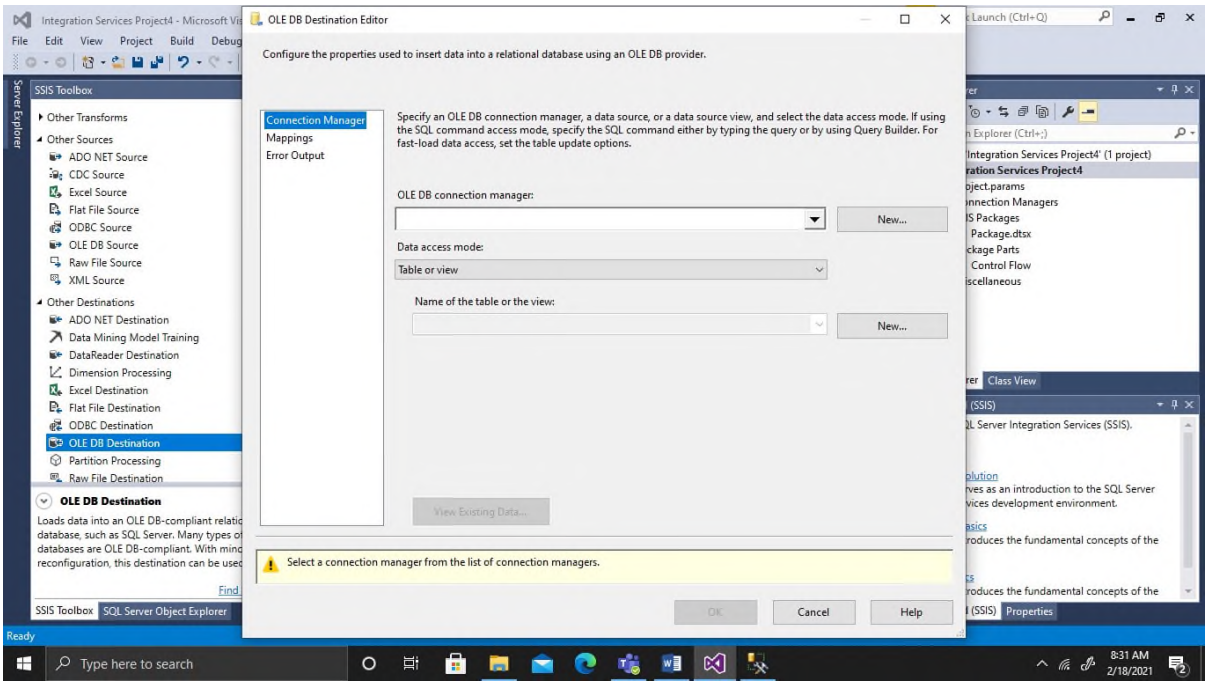
Step 20: After adding the destination, connect the **source to the Destination**.



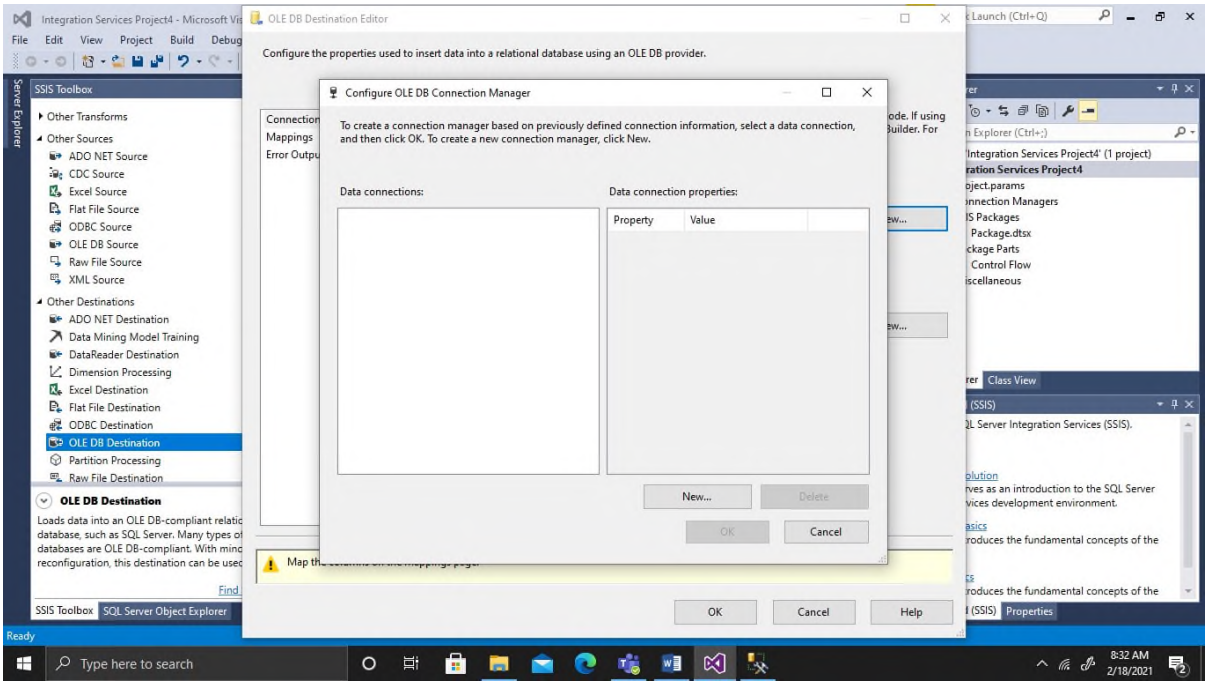
Step 21: In the above step, we noticed the red cross in OLE DB Destination, which means that it is not configured with a component. Right-click on the OLE DB Destination and then click on the Edit option as shown in the below screenshot:



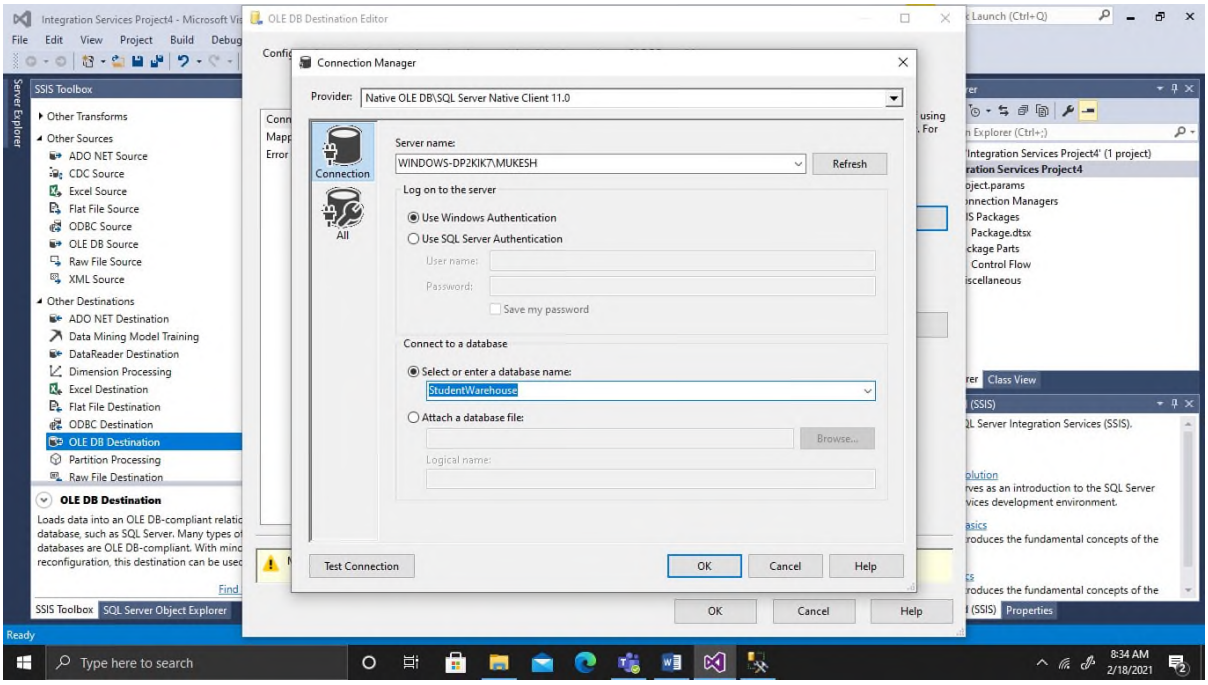
Step 22: You need to specify the configuration manager.



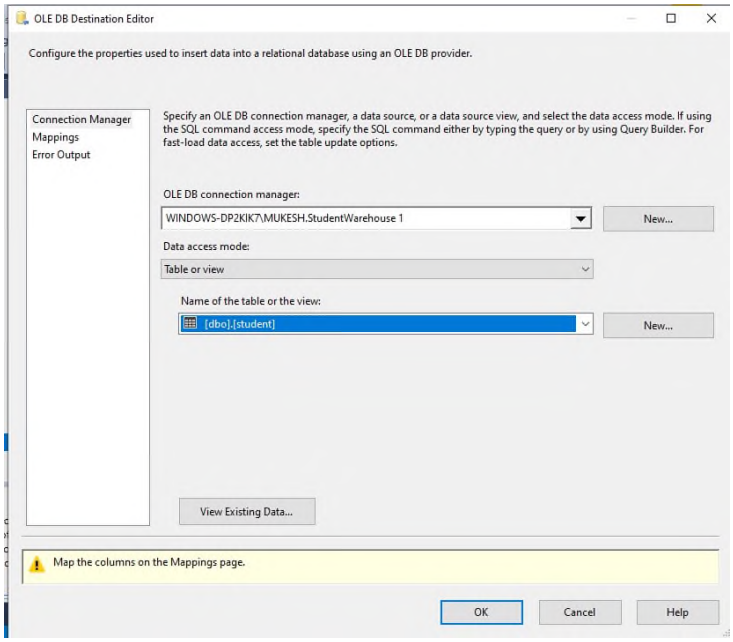
Step 23: Click on the **New** button to add the connection manager. When you click on the **New** button, the screen appears shown below:



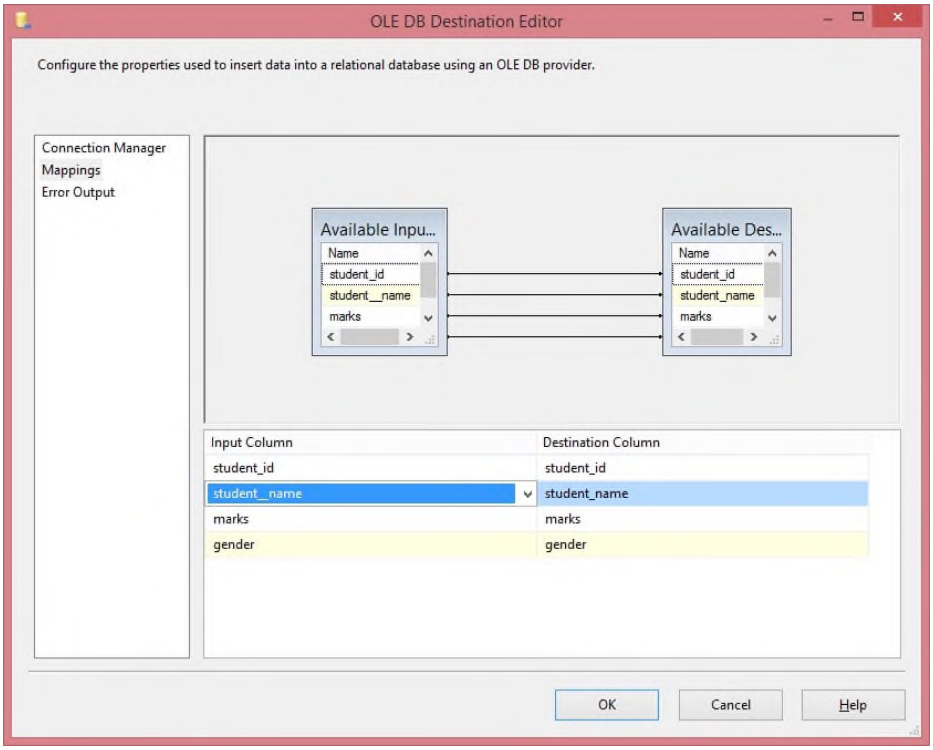
Step 24: Add the Server name and select the database name from the SQL Server.



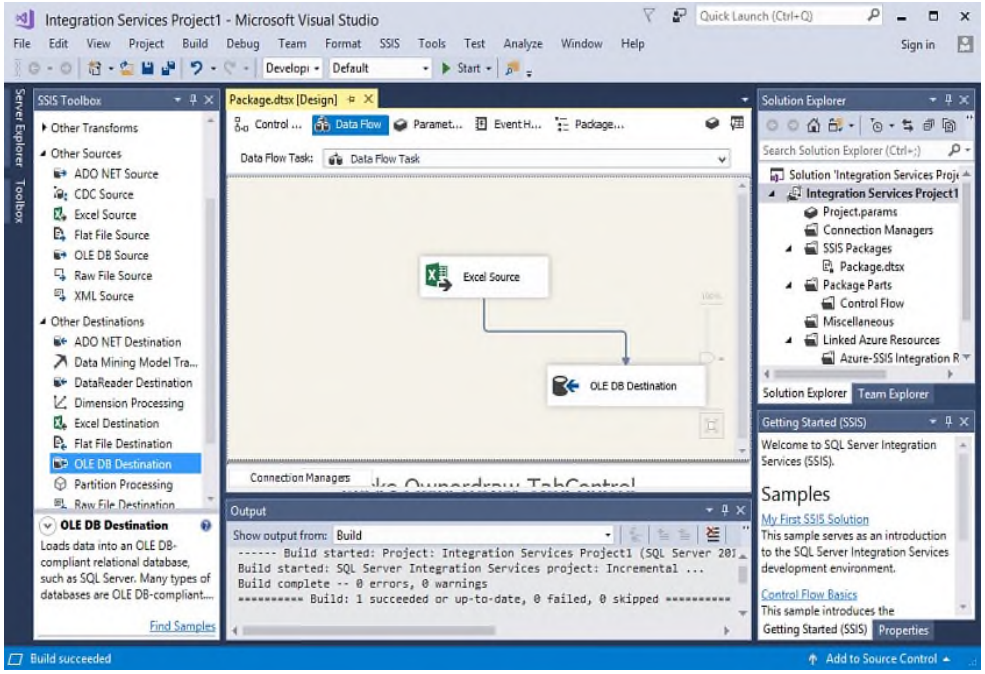
Step 25: Click on the OK button.
Step 26: Add the table name in which you want to load the data.



Step 27: In the below screen, map the field of input source to the field of the destination source.



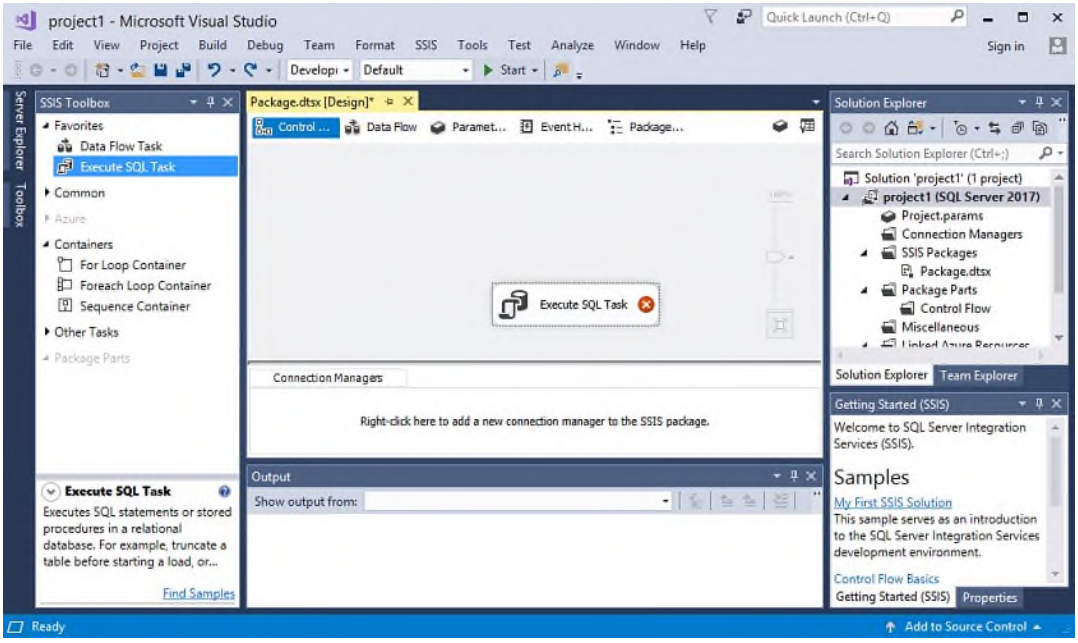
Step 28: After adding a mapping, the source is connect to the Destination, as shown in the below screenshot:



Step 29: Click on the **Start** button to run the Integration Services Project. When you run the project, the data gets loaded to the destination, i.e., **SQL Server**

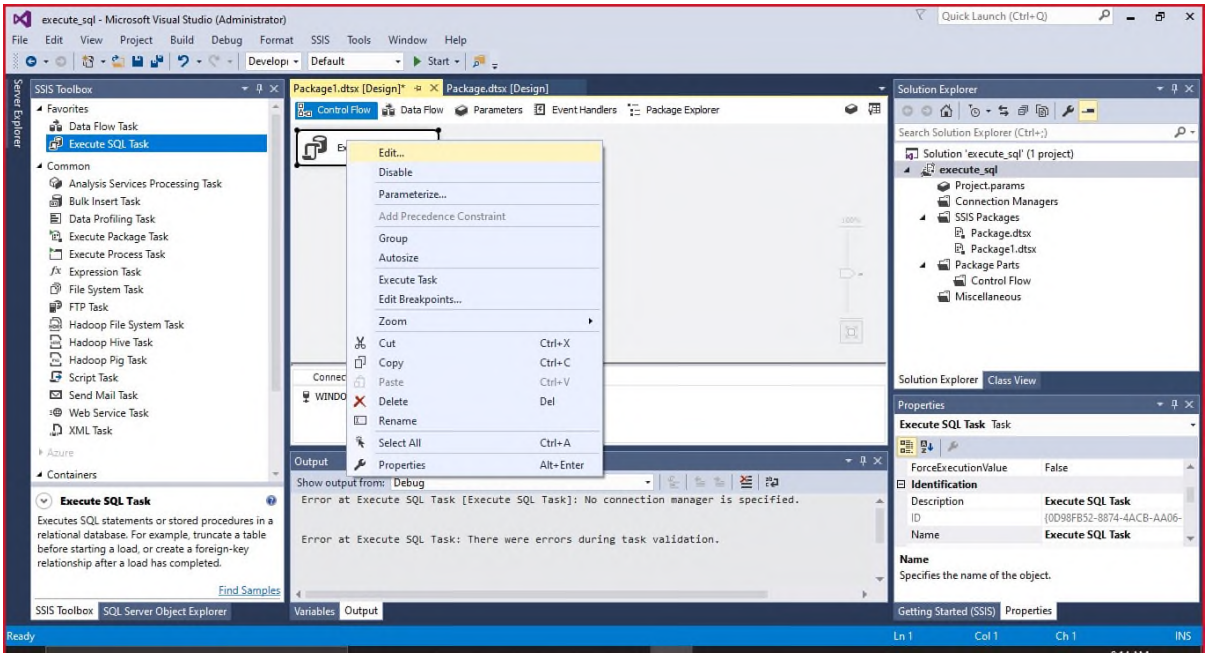
Example of Execute SQL Task

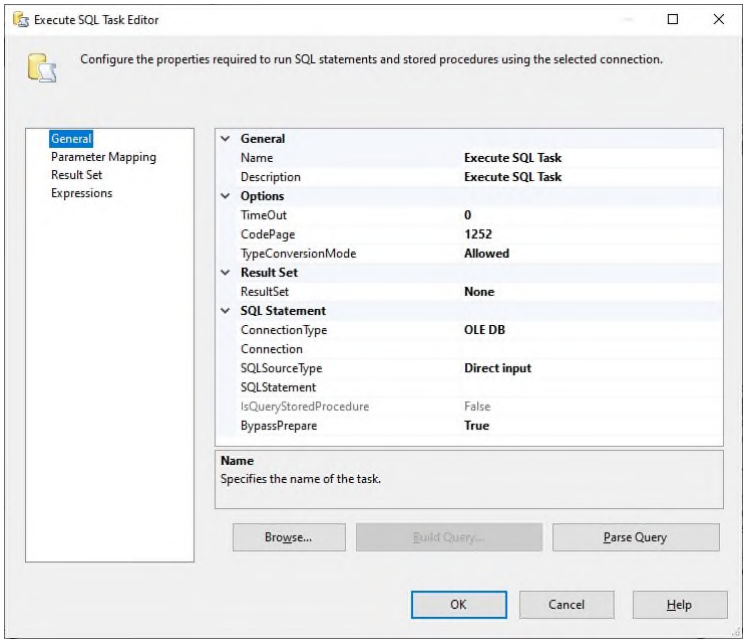
- Follow the steps from step 2 to step 5 of the previous example.
- Drag and drop the **Execute SQL Task** appearing at the leftmost panel to the main window.



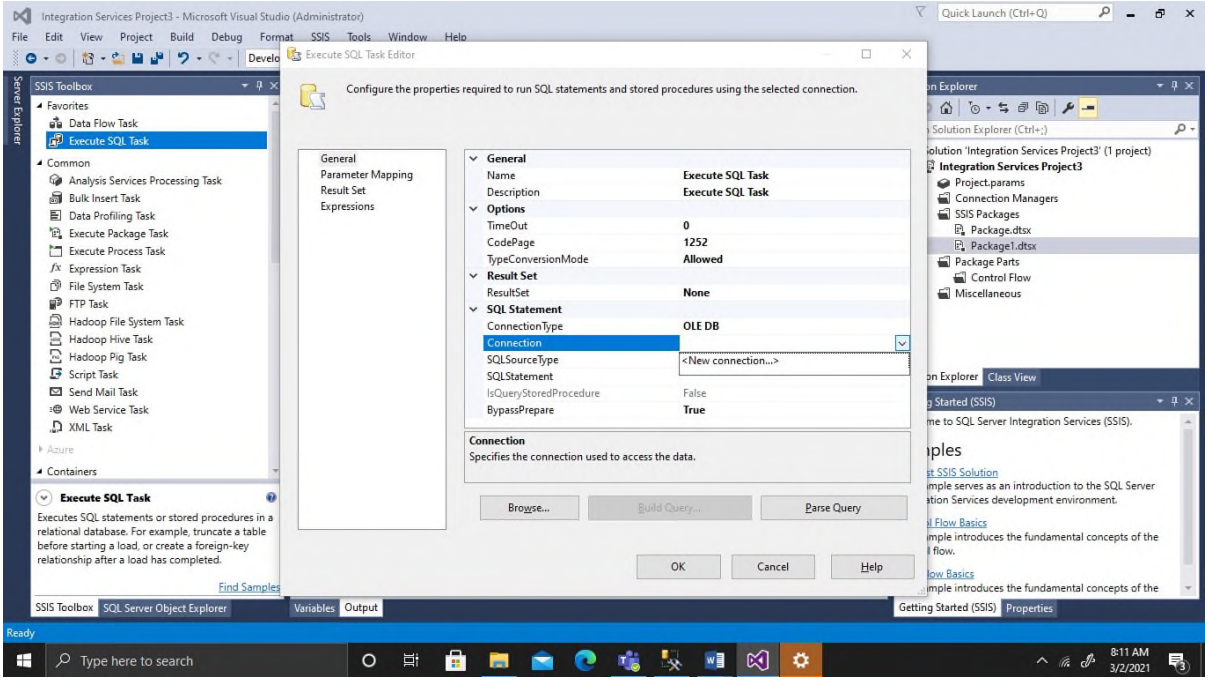
- In the above screen, we observe that the red-cross appears in the Execute SQL Task, which means that the **Connection Manager** is not configured with the **Execute SQL Task**. To configure it, we need to add the Connection Manager.

- Right-click on the Execute SQL Task and then click on the edit as shown in the below screenshot:

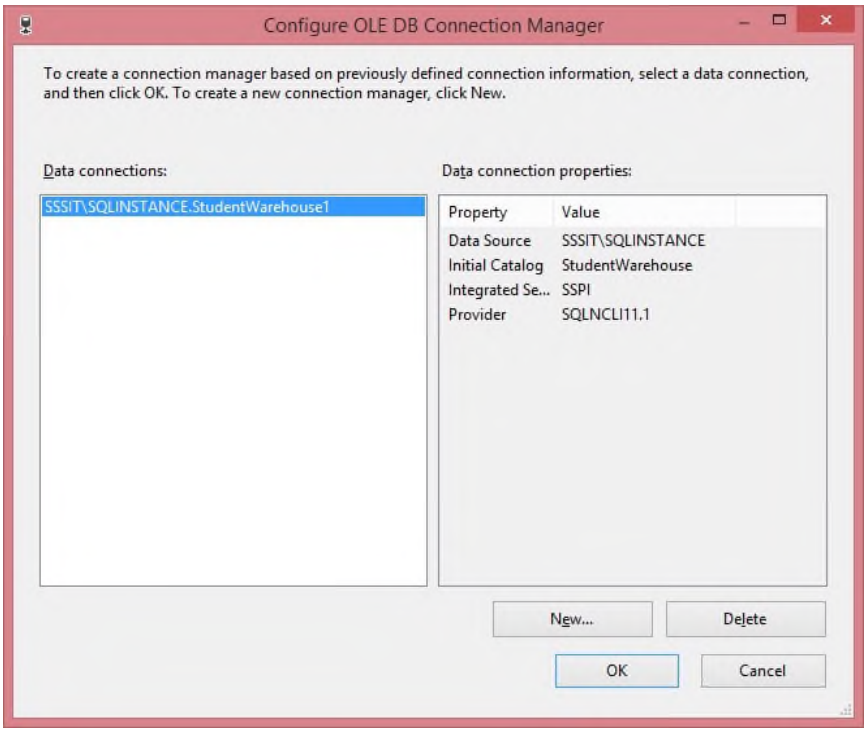




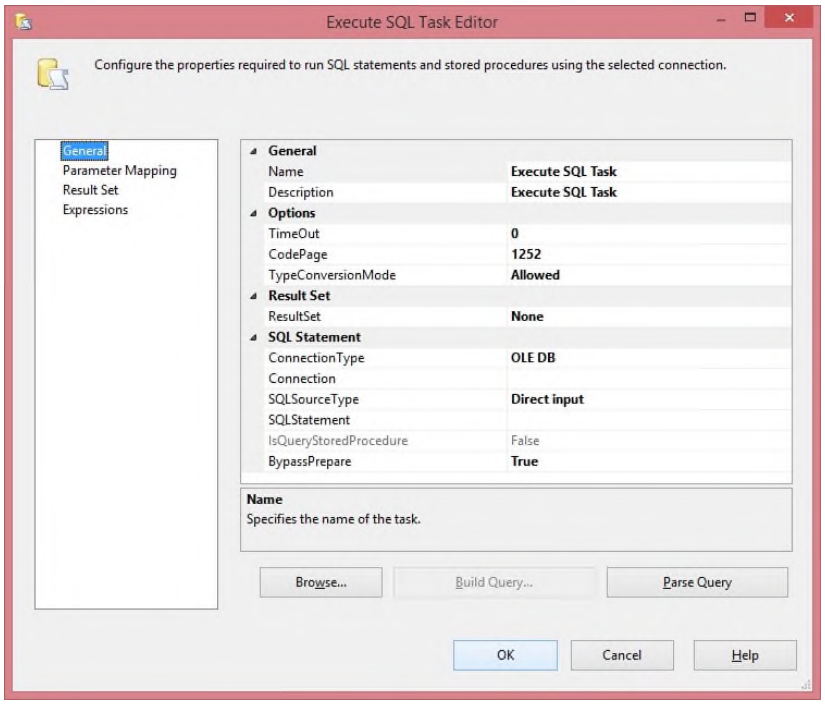
- Click in connection textbox, in appeared screen appears shown in the below screenshot:



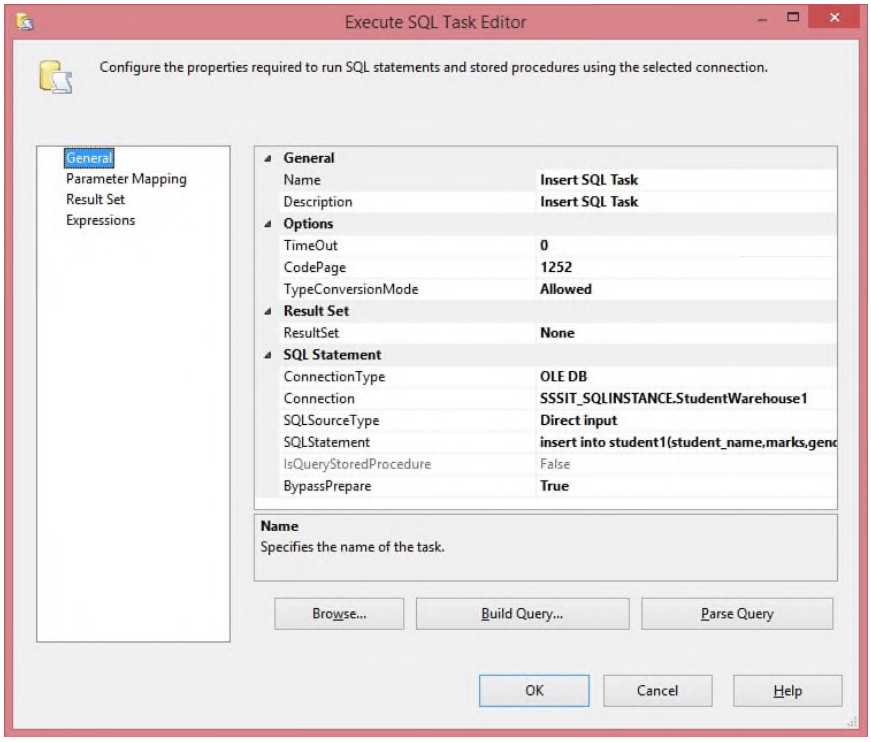
Select new connection.



- In the above screen, we observe that the data connection of **StudentWarehouse** database is already available as we created in the previous step. Click on the **OK** button.
    - When you click on the OK button, the screen appears as shown below:

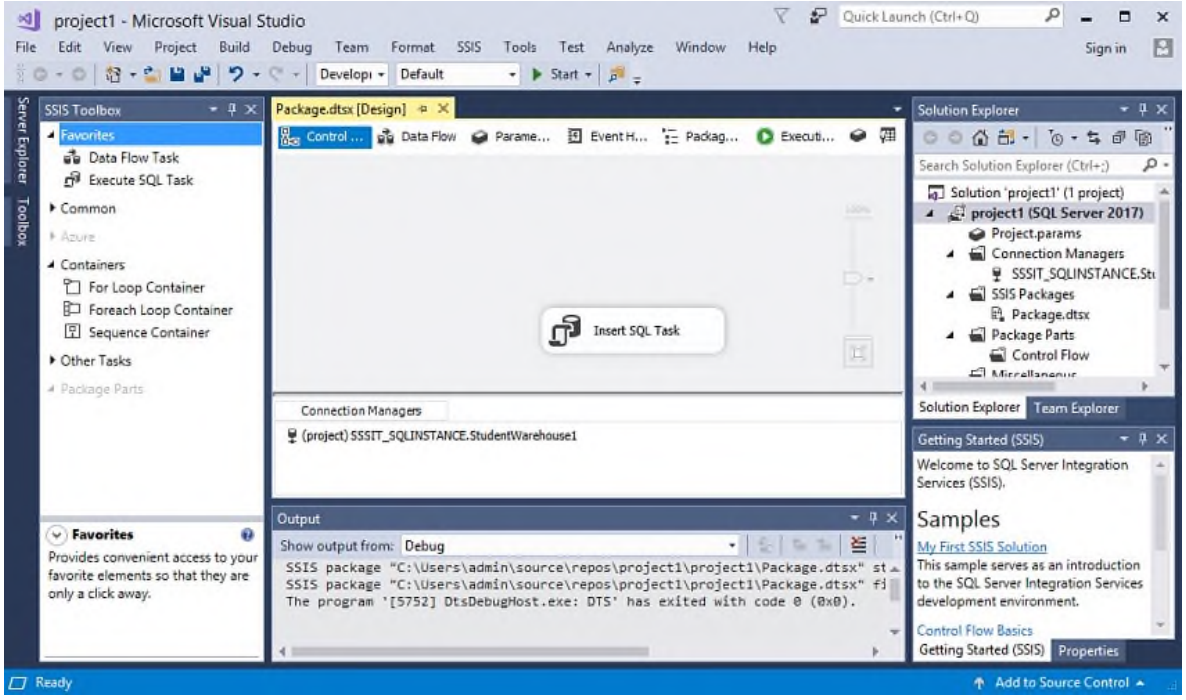


- In the above screen, you need to enter the details such as the name of your task, description, SQL statement (SQL task that you need to perform on the table).
  - I entered the following details in the above screenshot:

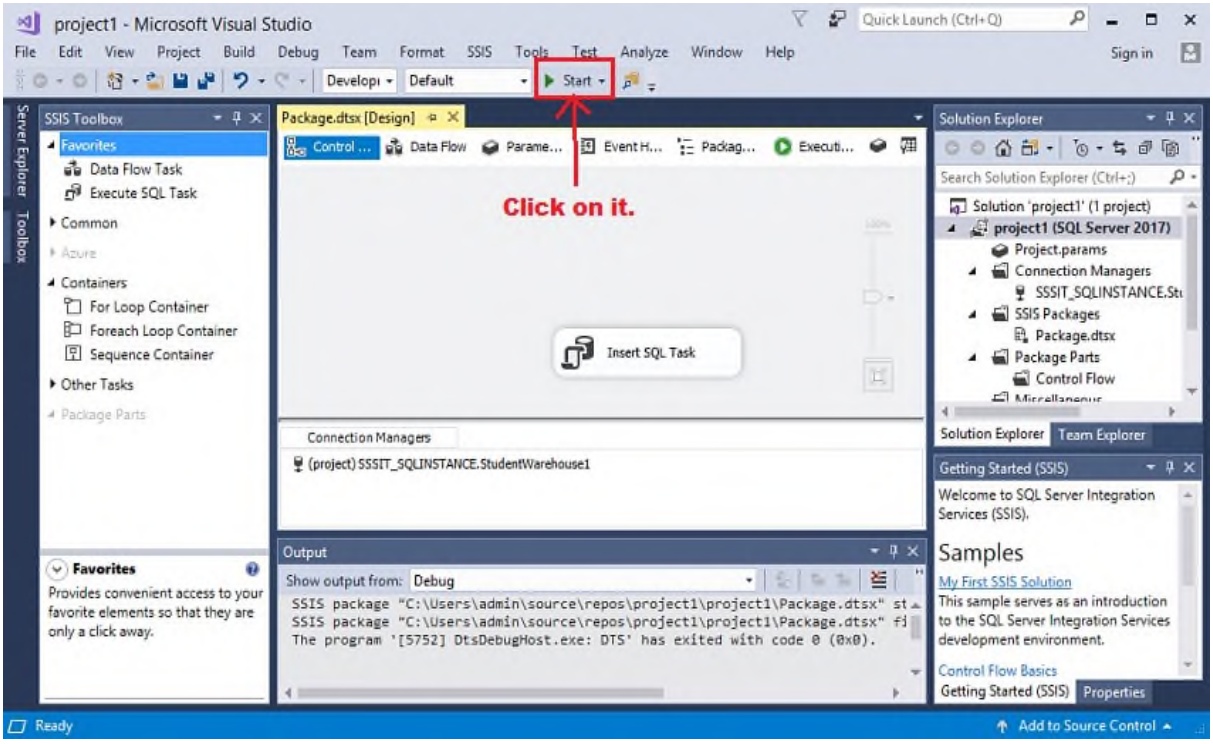


In the above case, I have provided the insert command in SQLStatement.
Write insert query[**insert into student(student_name,marks,gender) values('rohan',77,'male')**]

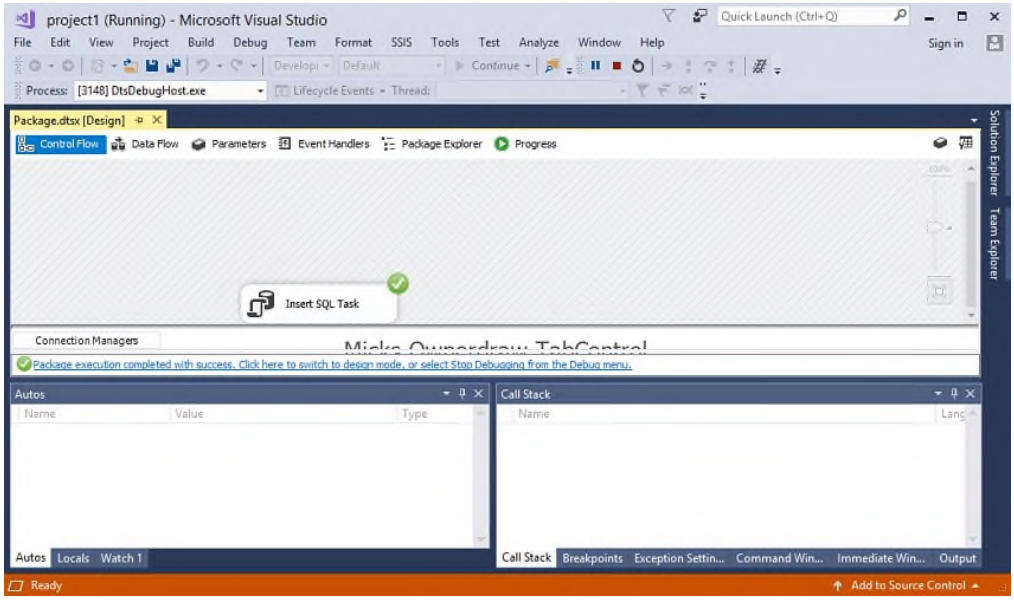
- Click on the **OK** button.
- When you configure the component with the Execute SQL Task, the red cross disappears as shown in the below screenshot:



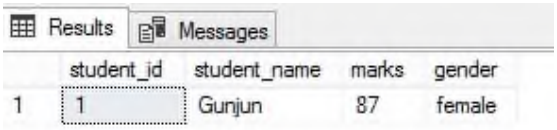
- Run the project by clicking on the **Start**.



- When you click on the **Start**, the screen appears shown as below:



- In the above screen, the tick mark shows that your SQL Task has been successfully completed. The insert statement, which we added in the previous steps, has been executed, and the data is add in table. i.e., student1.
  - Output of this task is shown below



Results | Messages

	student_id	student_name	marks	gender
1	1	Gunjun	87	female

SSIS Containers

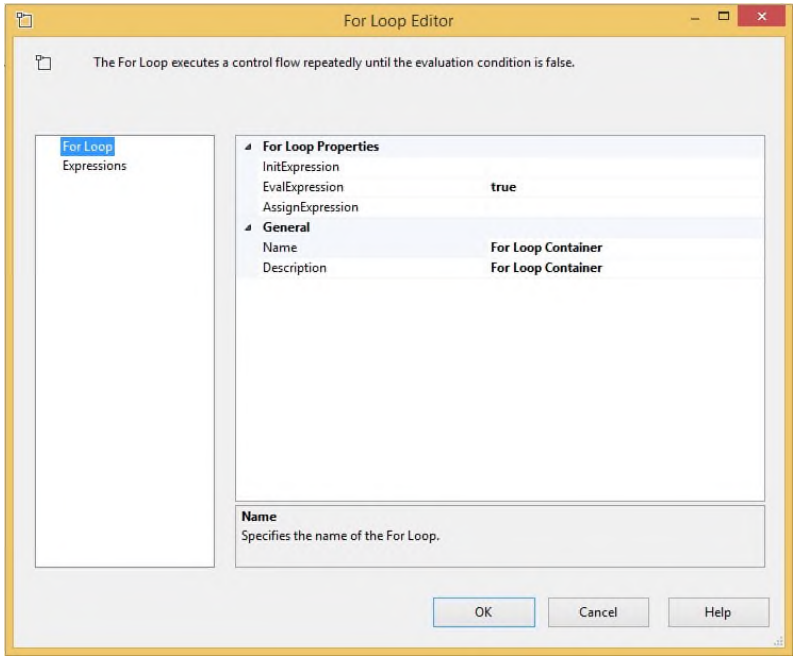
Next to tasks, you also have **containers**. These give you more power over how tasks executed. You can add one or more tasks to a single container.

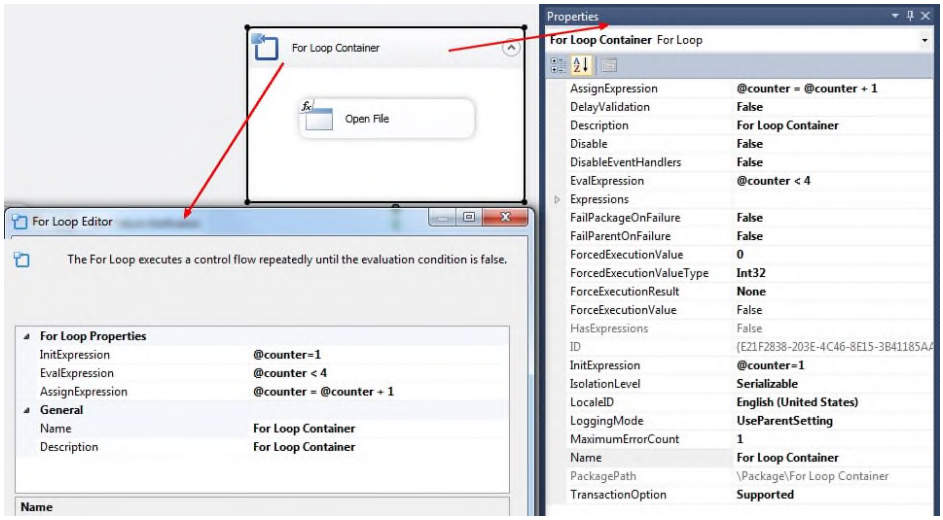
- **<u>For Loop Container:</u>**

  The For Loop Container enables you to create looping in your package similar to how you would loop in nearly any programming language. In this looping style, SSIS optionally initializes an expression and continues to evaluate it until the expression evaluates to false. With this container, you can execute all tasks inside for a fixed number of executions. For Loop container executes its ETL logic zero to n times. It has three functions that control how many times the loop will be executed:

  - **InitExpression:** Used for the initial setting of a starting variable (such as setting a counter variable to zero).
  - **EvalExpression:** This is the expression that evaluates whether the loop should continue. Of the three functions described here, this is the only one that requires a value – the others are optional.
  - **AssignExpression**: This allows the use of an assignment expression, such as incrementing a loop counter.

  The only value that is required for the For Loop container is the EvalExpression. A very simple For Loop configuration is shown below.





The defined variable can be used with the following syntax:

@[User::counter]

- **<u>For each Loop Container:</u>**

  This container does not execute a fixed number of times like for loop, but the number of executions determined by a collection.
  This can be for example the number of files in a directory or the number of rows in a table.
  This makes the container more flexible than a for loop container.
  Foreach Loop containers performs a loop with a collection.
  The SSIS Foreach Loop Container is more complicated than the For Loop Container since it has many use cases and requires a more complex configuration:

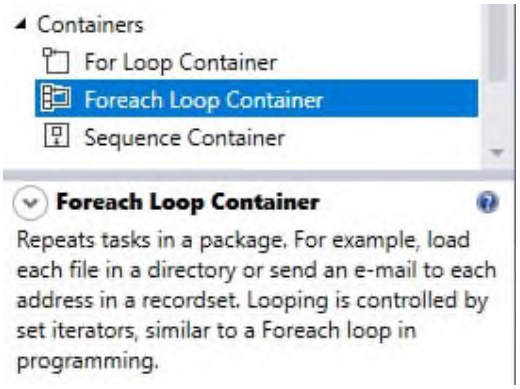


Figure – SSIS Foreach Loop Container description from the toolbox

There are different types of enumerators in the SSIS Foreach Loop Container. You can select the enumerator type from the collection tab within the SSIS Foreach Loop Container editor form:

- **Foreach File enumerator: -**
  This is the most common type of enumerator .Performs an action for each file in a directory with a given file extension. You will loop through a list of files and simulate some type of action that has occurred inside the container.
  Foreach File enumerator to enumerate files in a folder. The enumerator can traverse subfolders. For example, you can read all the files that have the *.log file name extension in the Windows folder and its subfolders. Note that the order in which the files are retrieve cannot be specify.

- **Foreach Item enumerator: -**
  Foreach Item enumerator to enumerate items that are collections. For example, you can enumerate the names of executables and working directories that an Execute Process task uses.

- **Foreach SMO enumerator: -**
  Foreach SMO enumerator to enumerate SQL Server Management Objects (SMO) objects. For example, you can enumerate and get a list of the views in a SQL Server database.

- **Foreach Nodelist enumerator: -**
  Elements and attributes in an XML document. Foreach Nodelist enumerator to enumerate the result set of an XML Path Language (XPath) expression. For example, this expression enumerates and gets a list of all the authors in the classical period: /authors/author[@period='classical'].

- **Foreach ADO enumerator: -**
  Foreach ADO enumerator to enumerate rows in tables. For example, you can get the rows in an ADO recordset.
  The Recordset destination saves data in memory in a recordset that is stored in a package variable of **Object** data type. You typically use a Foreach Loop container with the Foreach ADO enumerator to process one row of the recordset at a time.

- **Foreach ADO.NET Schema: -**

Tables in a dataset or rows in a table. Foreach ADO.NET Schema Rowset enumerator to enumerate the schema information about a data source. For example, you can enumerate and get a list of the tables in the AdventureWorks2012 SQL Server database.

- **Foreach from Variable enumerator: -**
  Foreach from Variable enumerator to enumerate the enumerable object that a specified variable contains. The enumerable object can be an array, an ADO.NET **DataTable**, an Integration Services enumerator, and so on. For example, you can enumerate the values of an array that contains the name of servers.

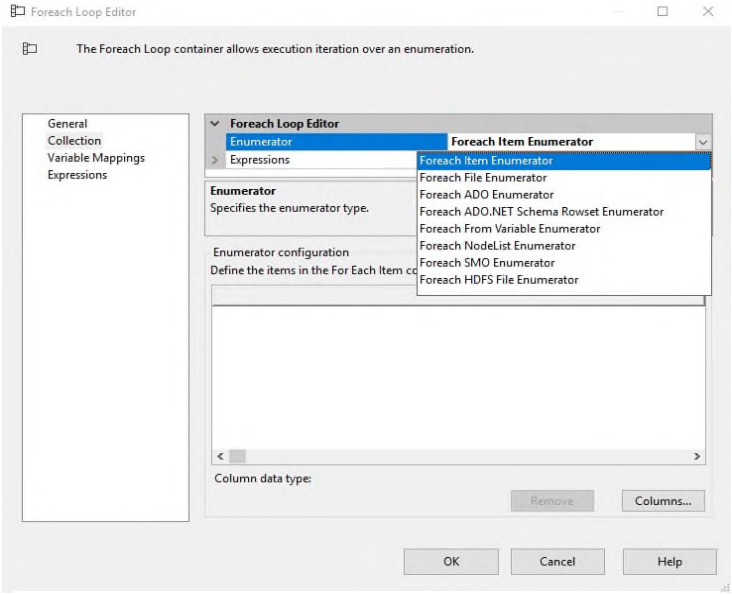


Figure – SSIS Foreach Loop Container collection tab page

- Each enumerator has its own properties that we must configure.
- To catch the current item while looping over a collection, we must add an SSIS package and map this variable to the item within the SSIS Foreach Loop Container variable mappings tab page by specifying the item index within the current row and the variable name.

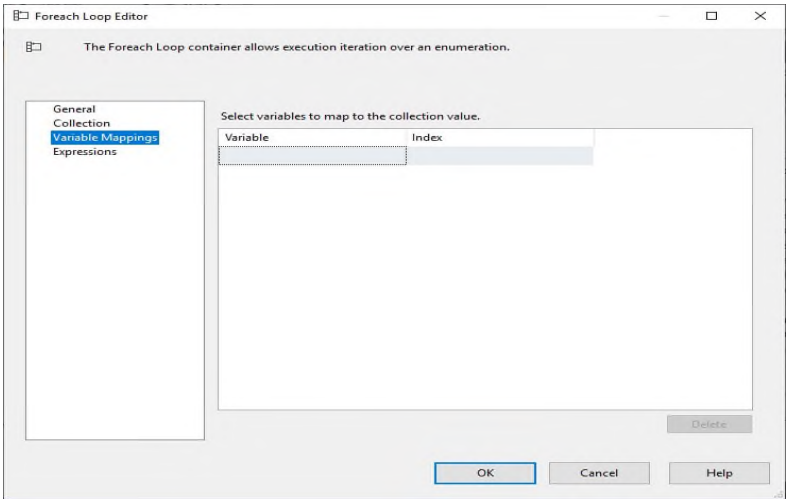
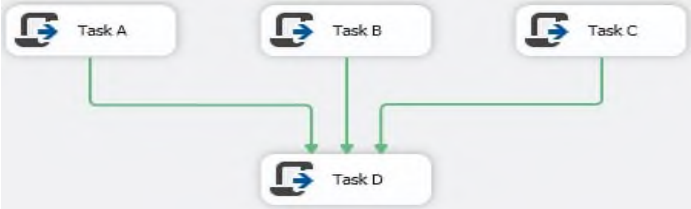


Figure – SSIS Foreach Loop Container variable mappings tab page

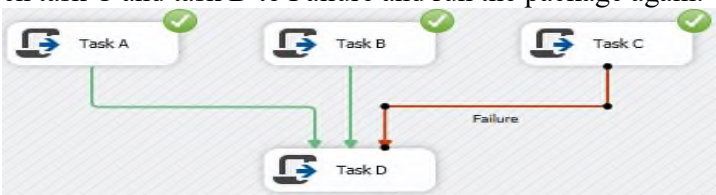
- **Sequence Container:**
  - We can consider a Sequence container as a subset of an SSIS package. It acts as a single control point for the tasks defined inside a container. This container simply groups tasks together. The tasks will execute together. This container is useful to split your control flow into logical units of work.
  - This container used in areas where the process needs to follow certain tasks sequentially. Sequential Task groups the tasks into multiple control flows and executes the process sequentially.
  - We can use this task widely based on our requirements; such as disabling a sequence when it should not process or using it when managing multiple tasks in one location. If there are many tasks in our package, we can group them based on their sequence and used to collapse and expand them and to enable and disable them easily.
  - The Sequence container has no custom user interface and you can configure it only in the Properties window of SQL Server Data Tools (SSDT) or programmatically.

Let's see the example below.

1. Create a new package named as "SequenceContainer.dtsx", drag and drop 4 script tasks, renamed and linked them as follows.

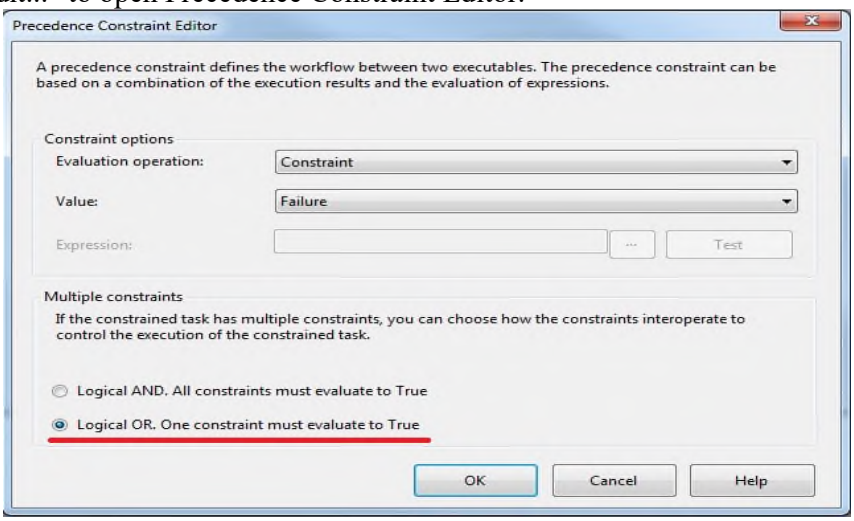


2. Run the package and you'll see all the tasks run successfully. Then change the precedence constraint between task C and task D to Failure and run the package again.

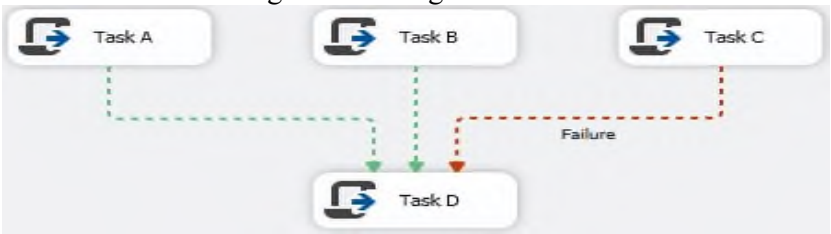


   You can see the task D has never run because the precedence constraint between task c and task D return false and the multiple precedence was set as Logical AND by default.

3. Click "Stop Debugging" or press shift F5. Right click one of the precedence constraints, choose "Edit..." to open Precedence Constraint Editor.



4. Set the multiple precedences as Logic OR and click "OK". Then all the lines were changed to dotted lines which stand for Logic OR among them.



5. Run the package again and you will find the task D runs great because of Logic OR. If one of the constraints linked to Task D returns true, the task D will run.

If the condition is changed to either both task A and task B returning success or task C returning failure, the task D will run. How can we realize this condition? That means if the current condition were A || B || C, the changed condition would be (A && B) || C. The solution is to add a sequence container.

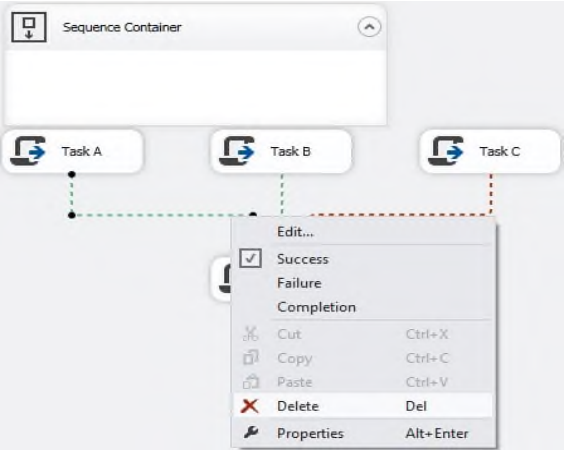
Sequence Container

A container is a subset of the SSIS package in the control flow and it can contain zero or more tasks and containers. A sequence container likes a sub package and you can add, edit or remove tasks as you do in the package. It can be configured through its property window. The benefits of using a sequence container are summarized below.

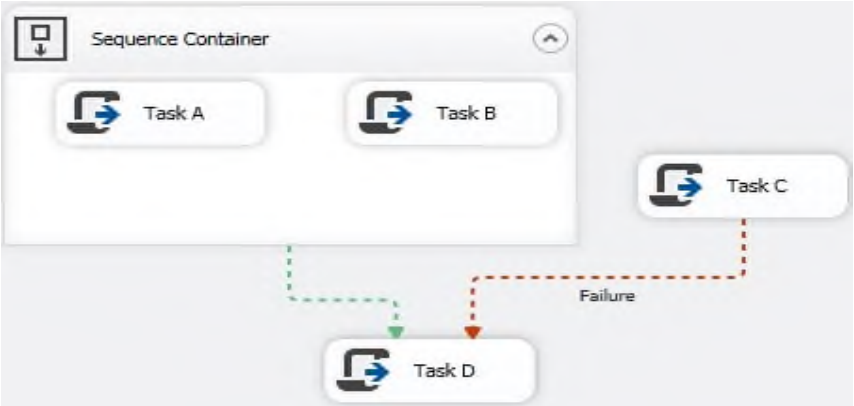
- One location Management: Change a property of a sequence container instead of changing the same property of all the tasks in the container.
- Variable Scope: Variables defined in the container scope are only accessed by the tasks or containers in the sequence container.
- Collapsing and Expanding: Have this ability to group related tasks in a container to be managed easily.
- Transaction: Easily to make the related tasks be finished in one transaction.

Now it is time to add a sequence container to the package above. Please follow the steps below.

1. Open the package SequenceContainer.dtsx if it is closed. Then drag and drop a sequence container to the control flow of the package.
2. Right click the precedence constraint between task A and D, then choose "Delete" to delete the constraint.



3. Repeat the step 2 again to delete the constraint between task B and D too.
4. Drag and drop the task A and B to the sequence container and link the precedence constraint of the container to the task D.



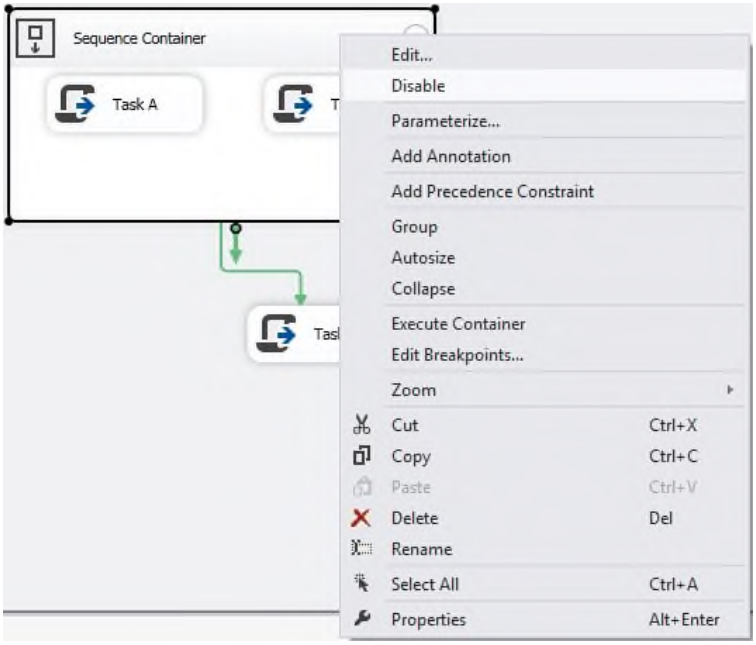
5. Run the package and you will see the task D will run OK because both task A and B running successfully makes the whole container return success. At last, the logic OR makes the task D execute.
6. Click "Stop Debugging".Then change the code of task B like we did before to make it return failure instead of success.
7. Run the package again.

Task B failed running makes the whole sequence container return failure so the task D has never run.

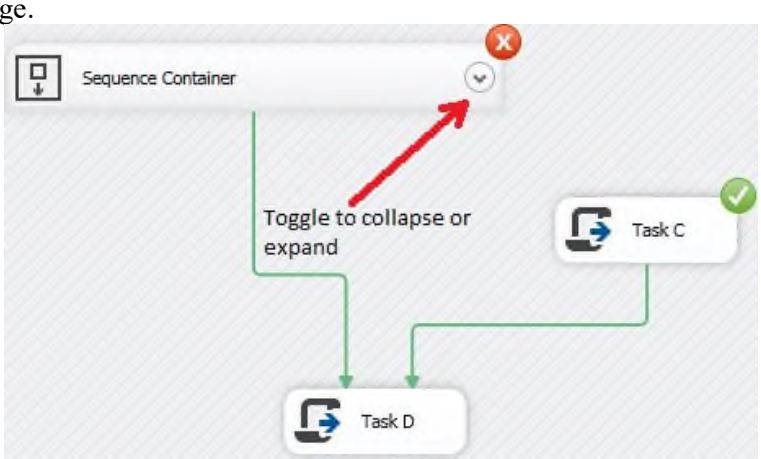
8. Click "Stop Debugging".Then change the constraint between task C and D to Success and logic AND. Run the package again.

The task D still cannot run because of the failure the container returns.

9. Click "Stop Debugging".Then right click the sequence container and choose "Disable" to disable the container.



10. Run the package and you will see the task D runs but the sequence container has never run because it was disabled. Click "Stop Debugging".
11. Right the container and choose "Enable" to enable it. Then click upper arrow at the top right corner of the container to Collapse it.
12. Run the package.



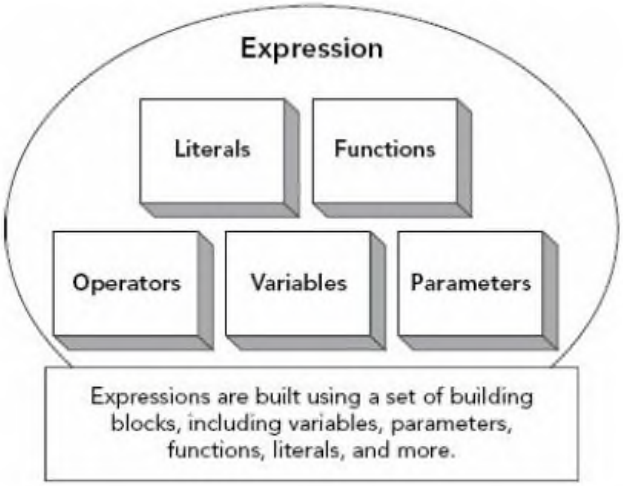
You can click the arrow to toggle collapsing and expanding the sequence container.

13. Click "Stop Debugging" to terminate running the package.

Create Dynamic Package Using Container in SSIS

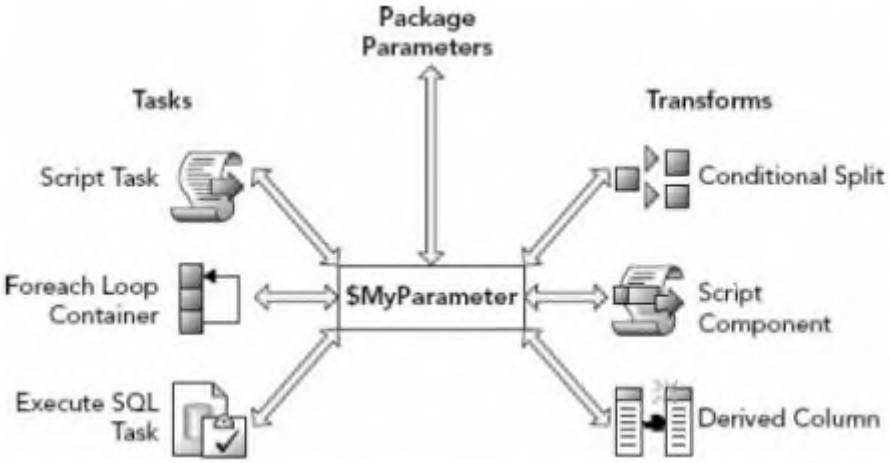
- Containers are objects in SQL Server Integration Services that provide structure to packages and services to tasks. They support repeating control flows in packages, and they group tasks and containers into meaningful units of work. Containers can include other containers in addition to tasks.
- SSIS includes multiple objects that used to create dynamic packages. Below Screen Shoot provides a graphical description of this model in SSIS. SSIS can dynamically set a property in a task or component using an expression, which can be built using a set of building blocks,

including variables, parameters, functions, literals, and more. When the package is run, the expression is evaluated and used in the property as each task or component is accessed.



- **Variable Overview**
  Variables are a key feature in the SSIS package development process. This object contains a value that can be hardcoded, dynamically set once, or modified multiple times throughout the execution of the package. Principally, variables provide a method for objects in a package to communicate with each other. Similar to their use in programming languages, variables used in specific types of logic, such as iterating through a loop, concatenating multiple values together to create a file directory, or passing an array between objects./

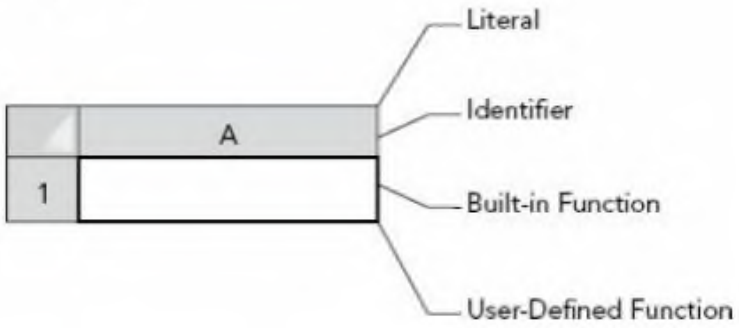
- **Parameter Overview**
  Parameters first introduced in SQL Server 2012. While similar to a variable in that a parameter can store information and used in an expression, it has a few different properties and uses that you will want to understand. As demonstrated in below Screen Shot, parameters are set externally. The parameter can then be used in an expression to affect different properties in the package.



SSIS uses two types of parameters: project parameters and package parameters. Project parameters created at the project level and used in all packages that are included in that project. On the other hand, package parameters created at the package level and used only in that

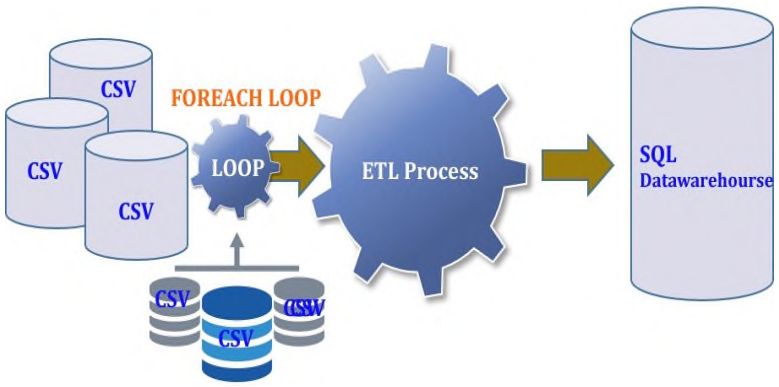
package. Project parameters best used for values that shared among packages, such as e-mail addresses for error messages. Package parameters best used for values specific to that package, such as directory locations.
When using the project deployment model, parameters are the best choices to replace package configurations to create a dynamic and more flexible SSIS solution. Using the Required property of the parameter, you can also necessitate the caller of the package to pass in a value for the parameter. If you want, to set either the value of a property from outside the package, required or not, parameters are the object to use. On the other hand, if you want to create or store values only within a package, variables are the object to use.

- **Expression Overview**
  Expressions are the key to understanding how to create dynamic packages in SSIS. One way to think about expressions is to compare them to the familiar model of a spreadsheet cell in a program like Microsoft Excel. A spreadsheet cell can hold a literal value, a reference to another cell on the spreadsheet, or functions ranging from simple to complex arrangements. In each instance, the result is a resolved value displayed in the cell. Figure shows these same capabilities of the expression, which can hold literal values, identifiers available to the operation (references to variables or columns), or functions (built-in or user-defined). The difference in the SSIS world is that these values can substituted directly into properties of the package model, providing powerful and dynamic workflow and operational functionalities.



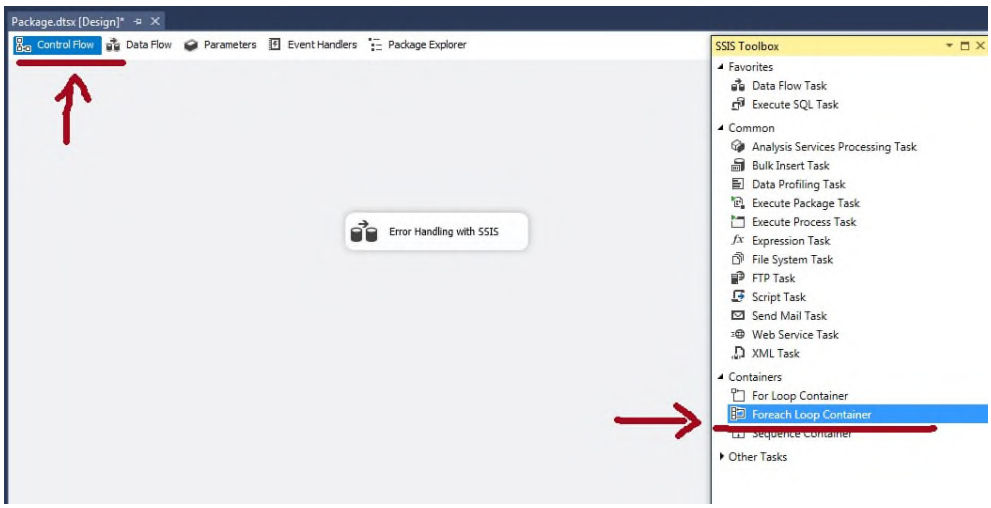
Create dynamic package using SSIS

- The Foreach Loop Container can be used to assist with conditional splitting of a data stream bound for a large number of destinations. We know that each of the output files will be structured identically, we can use one of the principles of basic programming: reusability. With the help from a dynamic expression, I will show you how to reuse the destination object to avoid the unnecessary creation and maintenance of multiple SSIS objects.
- For this example, I will create output files containing a list of invoices, with each file containing a single vendor's invoices. The methodology will be as such:
  - First, retrieve a list of vendors to be extracted as part of our ETL package, and store that list in an SSIS object variable.
  - Next, create a Foreach Loop container, and set the enumerator to use that vendor list object variable as its control collection.
  - Within the loop container, place a Data Flow Task, which will contain an OLE DB Source object and a Flat File Destination object.
  - For each iteration of the loop, the OLE DB source will query the database to retrieve invoices for the current vendor in the list, and will use an expression to dynamically set the output filename based on the vendor number.



- ETL process activities can be done in "Data Flow" Tab and Non-ETL process activites can be done in "Control Flow" Tab.
- There is also one more conatiner i.e. FOR LOOP but to use FOR LOOP we should know the number of source files with file-name. FOR LOOP loops through fixed count
- When there are collection of source files in a folder then it is better to use FOREACH LOOP CONTAINER because FOREACH loop loops through items in floder or recordset.
- When there is no fixed count of files or when we dont no number of recordset it is better to implement FOREACH LOOP. In this scenario, we want to loop through a folder with number source files so we will use FOREACH LOOP CONTAINER.
- In this example, we will implement FOREACH LOOP CONTAINER in a CONTROL TAB on a bulk folder, which will have multiple source files and that with help of FOREACH loop we will iterate each file and pass that to DATA FLOR TAB for ETL process.

- **<u>FOREACH Loop in SSIS step by step using example</u>**
  FOREACH LOOP Container is only available at Control Tab in SSIS tool box.

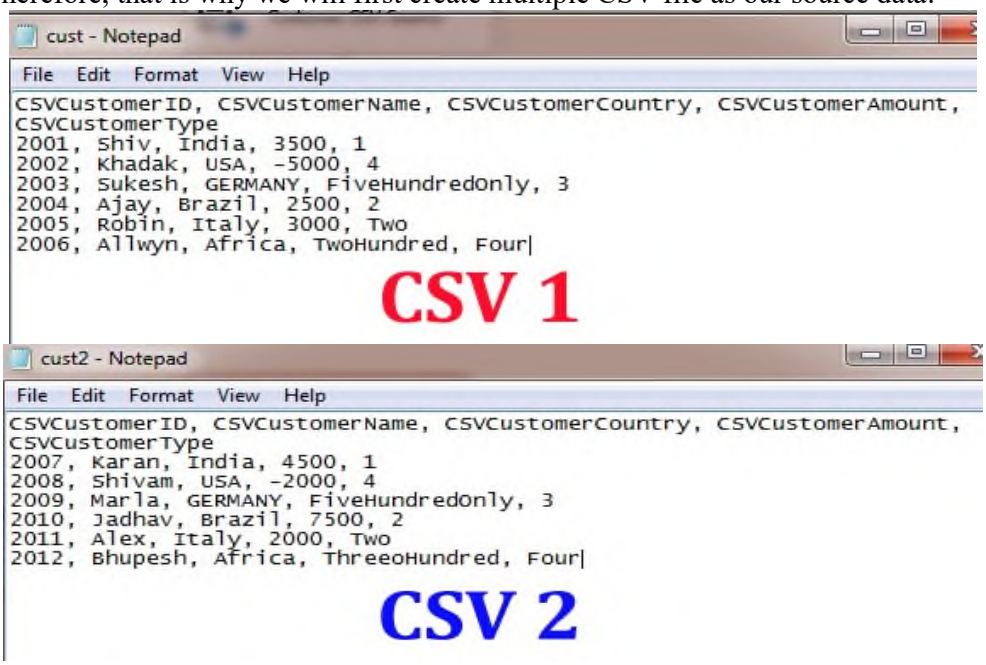


So we will be implementing FOREACH LOOP Container at control flow stage.

Step 1

Let us create multiple CSV files for Customer records as shown below.

FOREACH LOOP container can be used if there are multiple items to iterate. Therefore, that is why we will first create multiple CSV file as our source data.

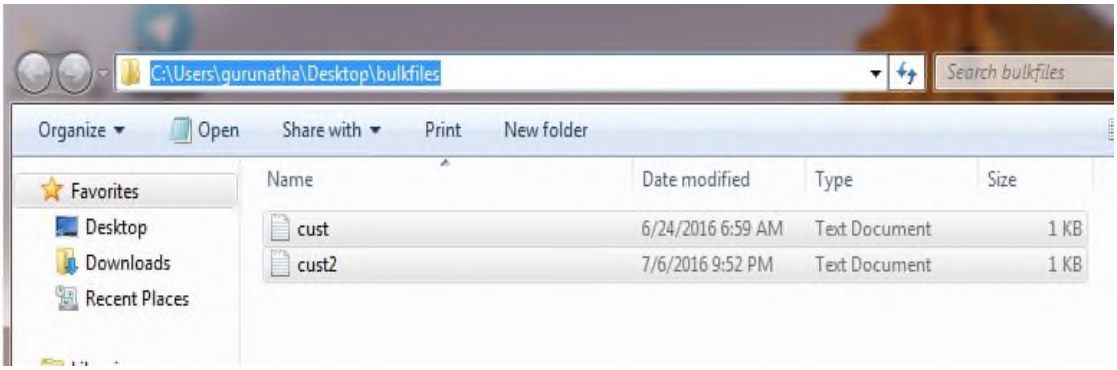


Step 2

In order to do this example first we need to create ETL process for a single file.
So create an ETL process for a single CSV file and proceed to next step.

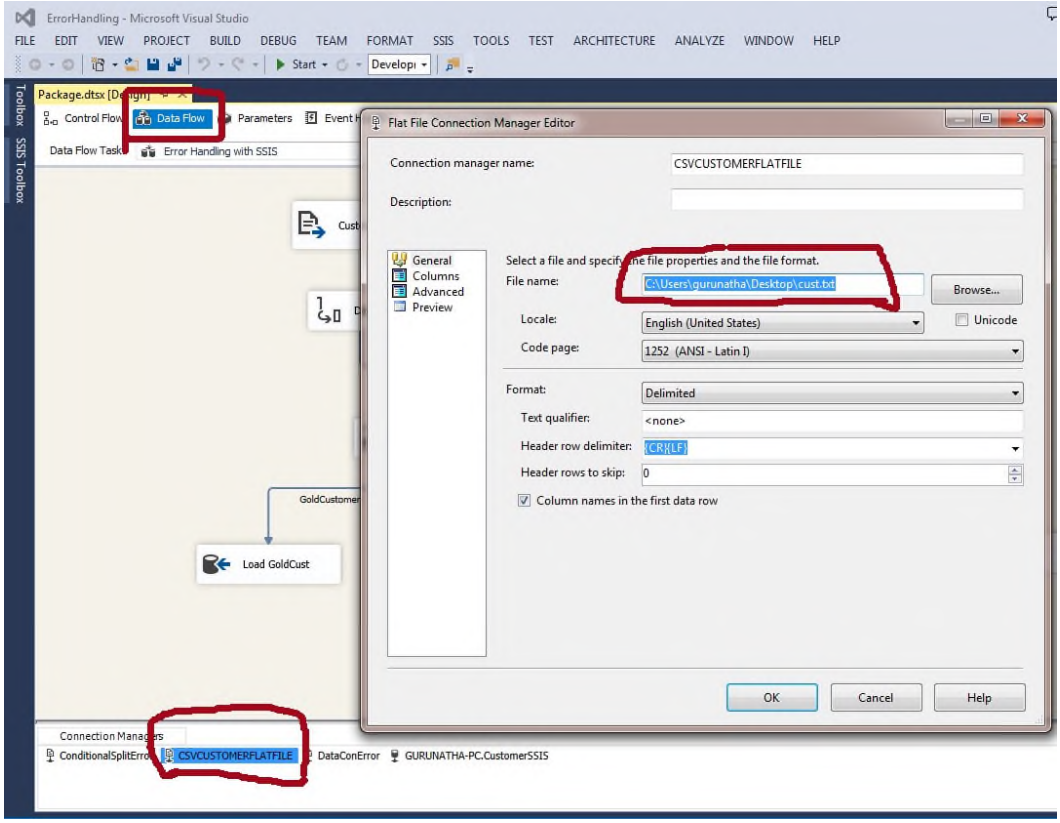
Step 3

After ETL process is completed now to make it easy we will save both our CSV files to a folder called "BULKFILES".
Create a new folder called "BULKFILES" and move both CSV files here.

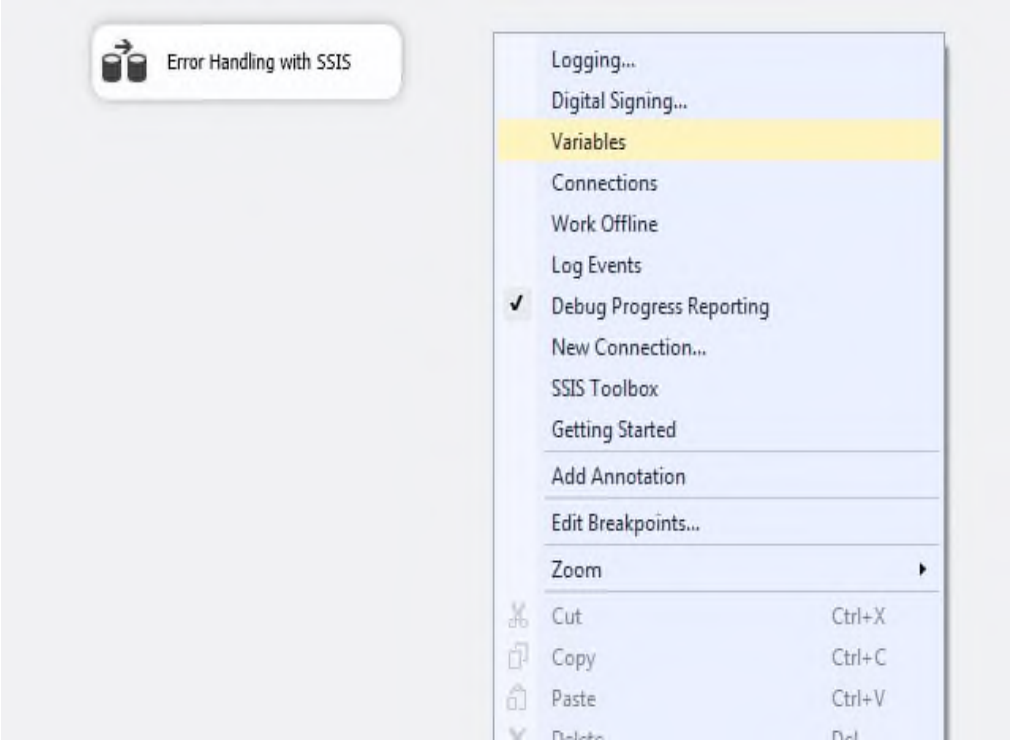


Step 4

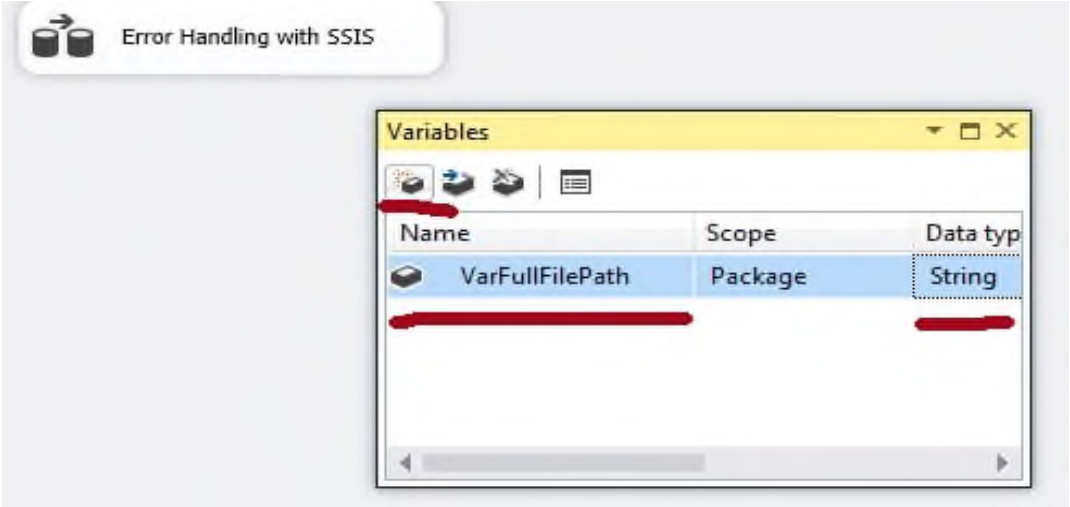
By "Control Flow" tab we will create a variable because if you see on "Data Flow" tab our source i.e. in Flat file Connection we have physical file name i.e. "cust.txt" which is one single file name but since we are working with multiple files so that file path should be dynamic so in order to make it dynamic.
We will create a variable in "CONTROL FLOW" tab and pass that full file name through this variable name to "Data Flow" using foreach loop dynamically.



At this moment, let us create a variable.
In order to create a variable goto -> Control Flow -> right click -> Add Variable as shown in below image.



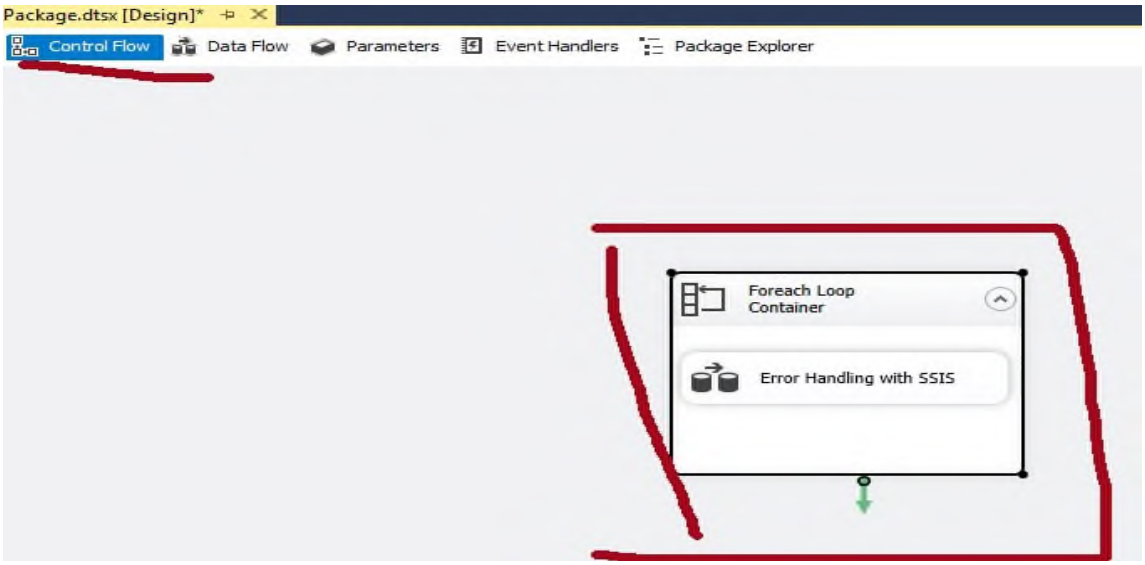
Add variable and give the suitable name and since we are passing file name so our variable data-type should be "String" as shown in below image.



This above variable will be used to pass to "Data Flow" tab which will be containing "FullFilePath".

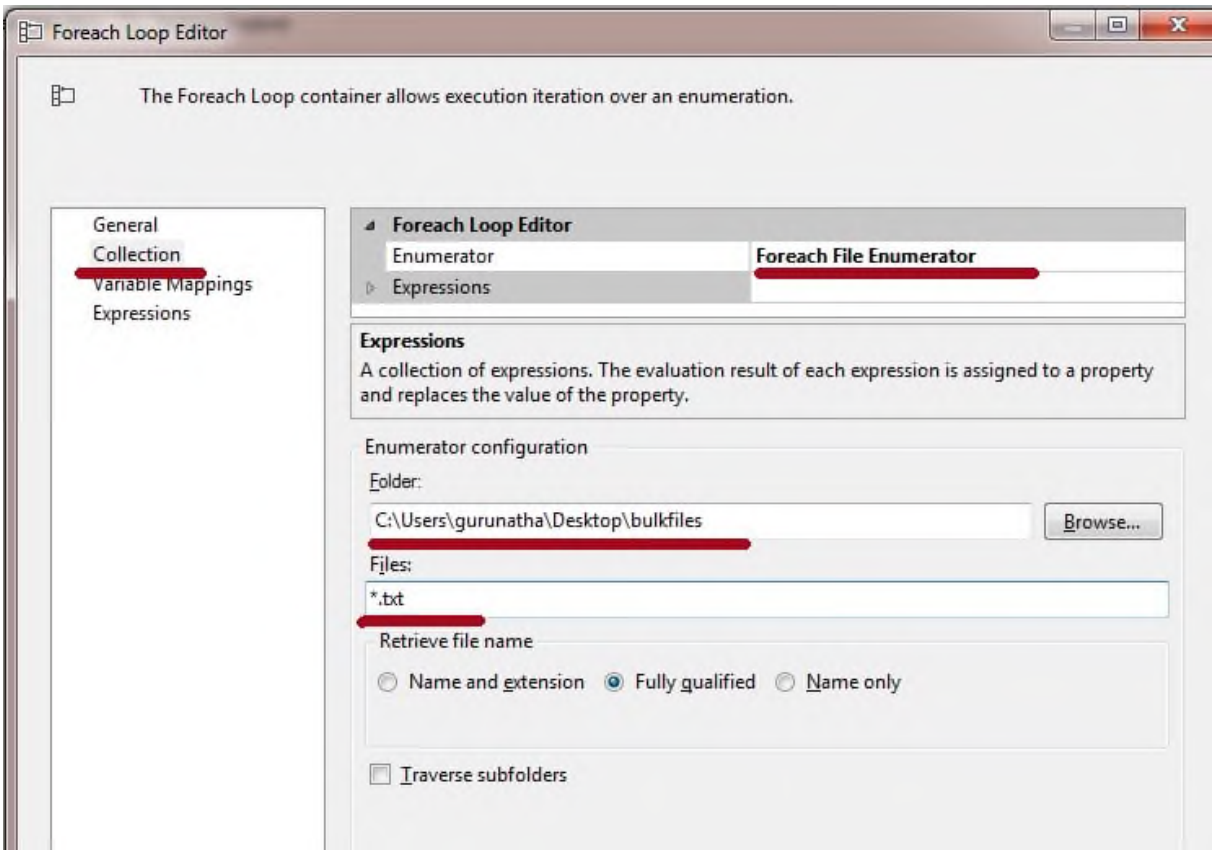
Step 5

In this step we will add a FOREACH LOOP container so in order add go to -> SSIS toolbox -> Now drag and drop data flow task container inside the FOREACH LOOP container as shown in below image.

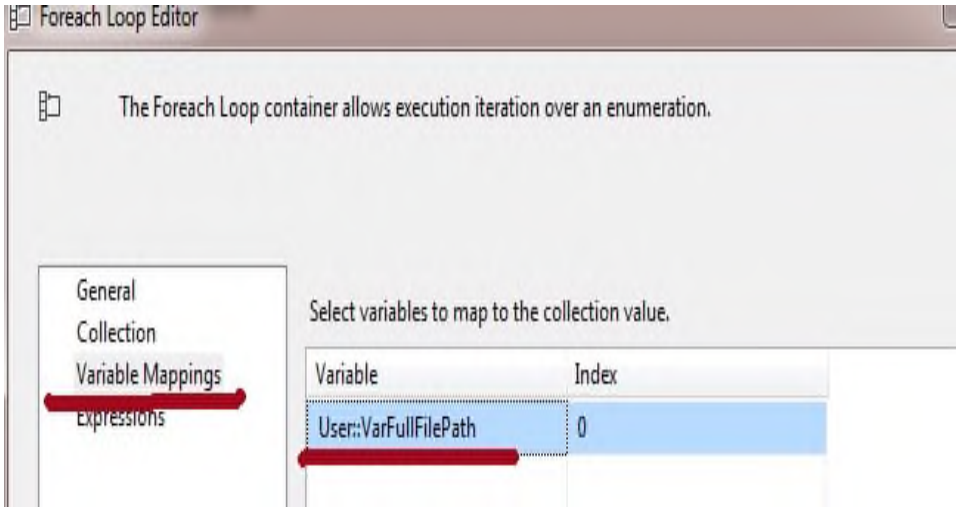


Step 6

We will configure FOREACH LOOP container. So just select container and right click -> edit.From left-hand side menu select collection and from enumerator select "FOREACH FILE Enumerator".In the folder section folder path i.e. "Bulk folder" which we have created earlier and below that, give extension as ".txt" since our data is in text format as shown in below image.



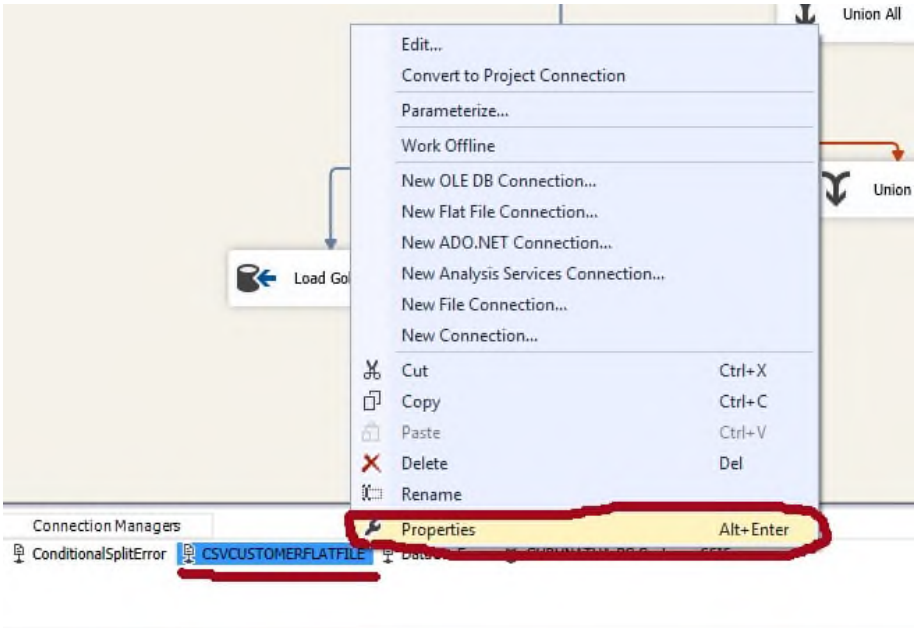
Do not close that modal now we have map our variable name to "FOREACH FILE Enumerator" to handle "File path" and same to pass to "Data Flow" tab.
So just below collection tab select Variable mapping from left-hand side menu and choose variable name which we have created earlier i.e. "VarFullFilePath" as shown in below image.



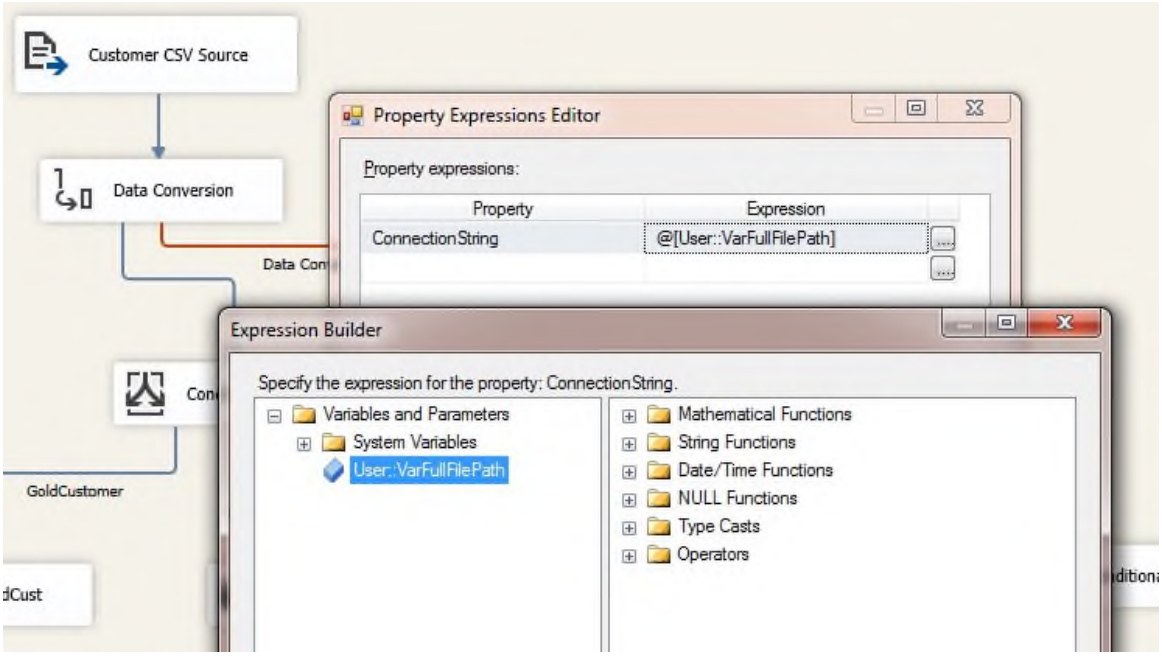
By this what will happen that everytime "FOREACH FILE Enumerator" loops and get a file it will assign that file path to this variable "VarFullFilePath".

Step 7

After configuring variable at control flow tab now we should also confirgure at Data Flow table at source end because it is where file name is needed for ETL process.
Data Flow tab and -> Connection Manage -> right click on CSVConnectionManager -> Properties as shown in below image.

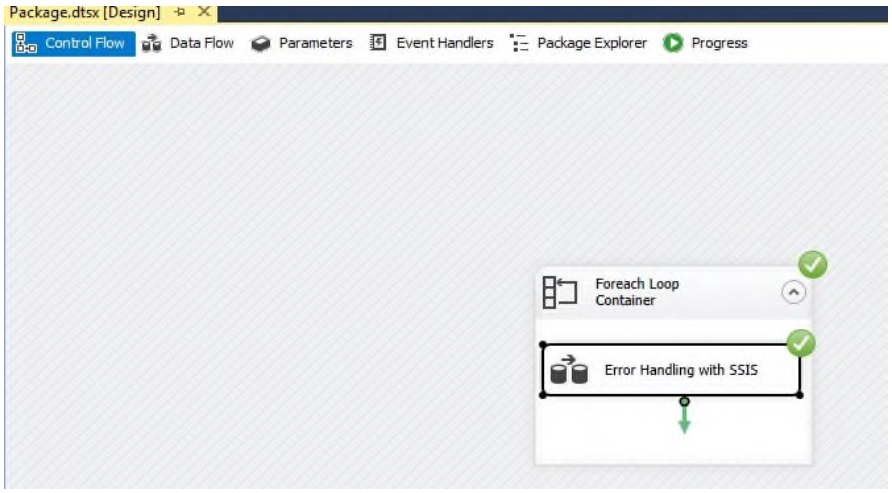


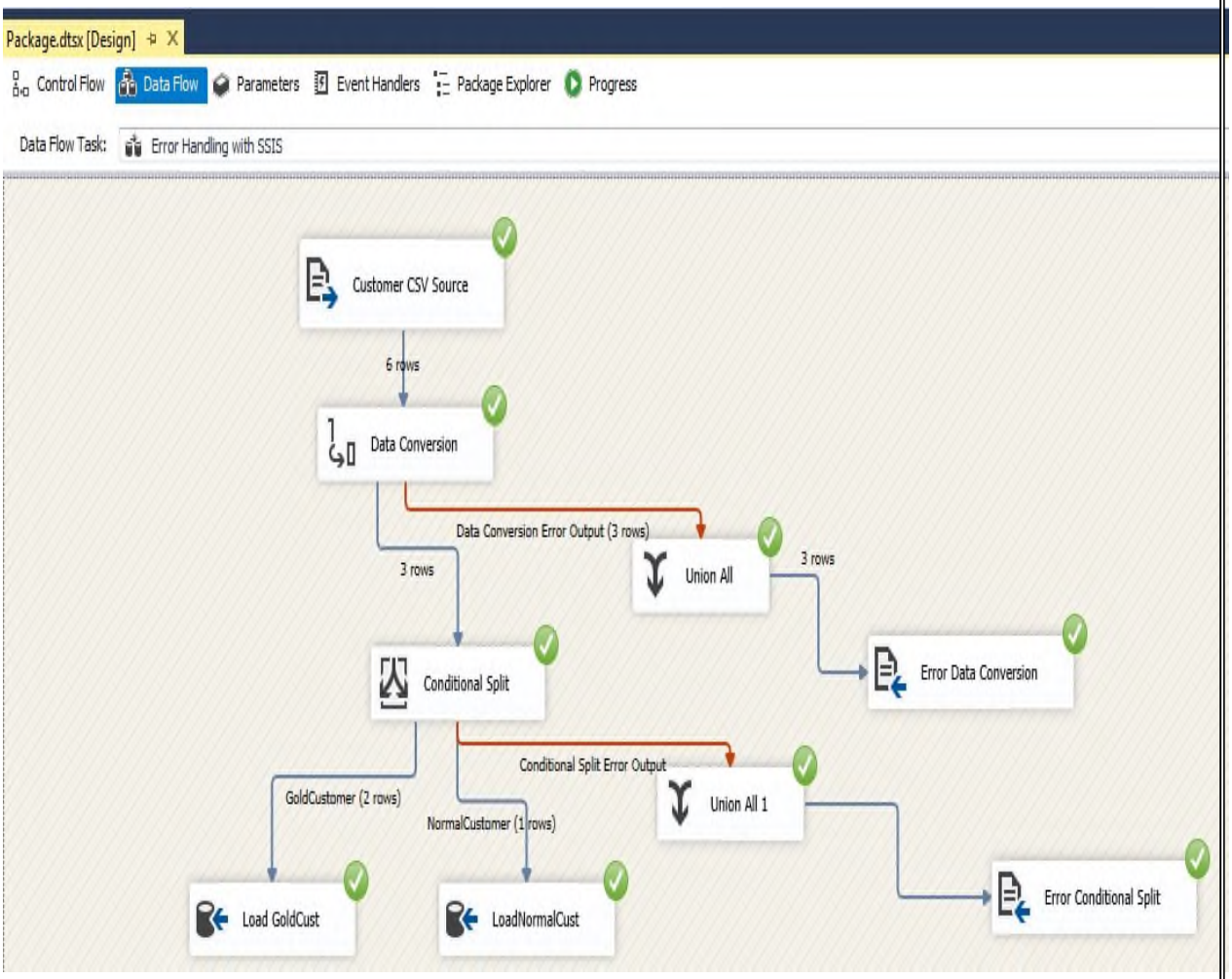
In the properties window if you see current "ConnectionString" it consist of single file path that we want to make it dynamic by assigning "VarFullFilePath".
So to do that in the same property window there "Expression" menu -> configure -> click. In the property select "ConnectionString" and click on expression select variable "VarFullFilePath" and drag it to expression and click on OK as shown in below image.



Step 8

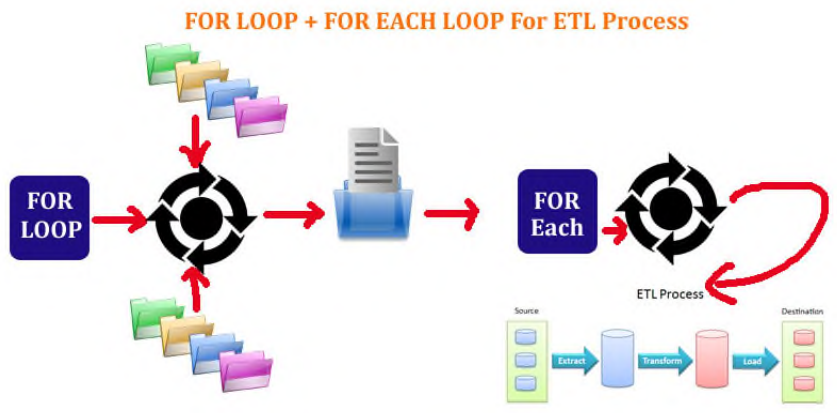
This is final step just run this project to see the output.





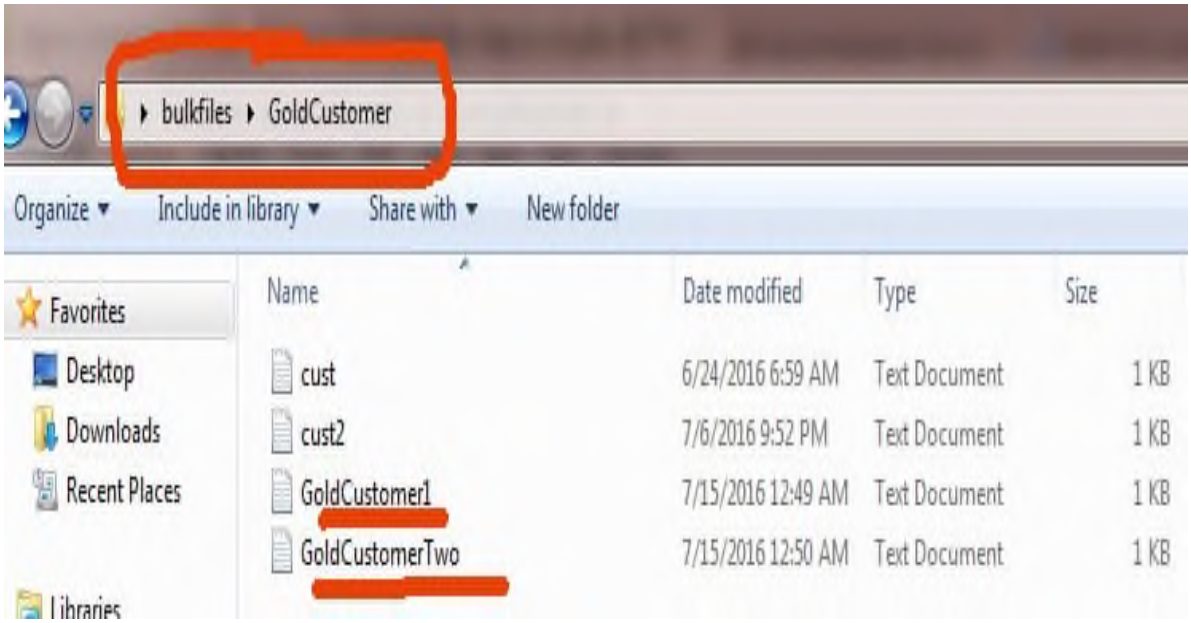
As you see, we have successfully executed the project and we successfully loaded data from both sources and same update to destination data source.

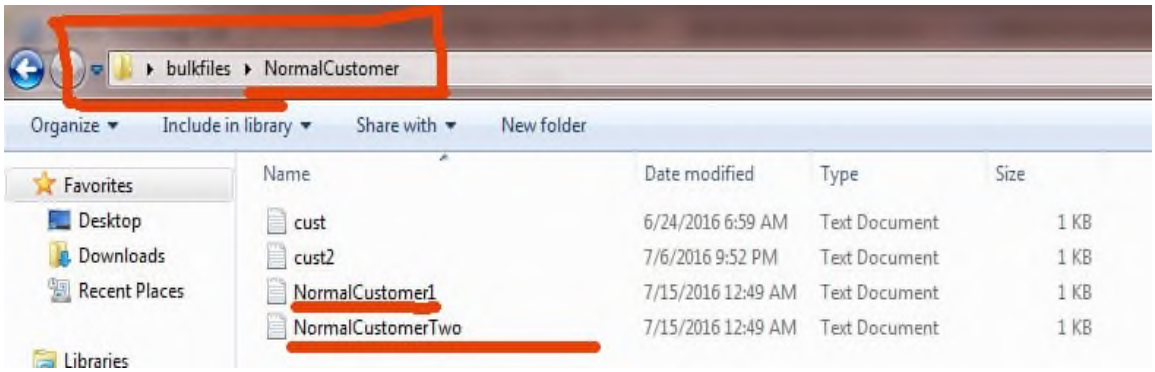
FOR Loop in SSIS step by step using example



Implement For Loop in SSIS

- For example, we have two source folders one for "GoldCustomer" and another one for "NormalCustomer" and inside these folder there are multiple data source files combined with "GoldCustomer" and "NormalCustomer" source files.
- So here what we want to do that we want to loop both the folders but we want to consider only "GoldCustomer" and "NormalCustomer" naming multiple source files for ETL process and rest other non-related source files we don't to take it.
- If we take foreach loop container for both folders then it could only loop source files inside that one folder only with a single FileSpec (GoldCustomer *).
- It cannot able to jump back to root folder to loop again another folder with a different FileSpec (NormalCustomer *). For this only we considered "FOR LOOP" to pass folder path and FileSpec i.e. (GoldCustomer *) and (NormalCustomer *).
- Because of this reason, we have used FOR LOOP container to loop both folders and to get their source files path and naming Goldcustomer and Normalcustomer.





- There are multiple source files naming "GoldCustomer * .txt" and "NormalCustomer * .txt" inside "GoldCustomer" and "NormalCustomer" folders so to loop that files we need to use FOREACH loop container here we cannot use FOR LOOP why because naming is not in sequence format or names are in different format with a starting "GoldCustomer * .txt" and "NormalCustomer * .txt" so for that we have to FOREACH LOOP inside the FORLOOP container.
- Then responsibility of FOREACH LOOP container to pass each source file name for ETL process. If you want to know, more about ETL process then click to read here.
- Responsibility of "FOR LOOP" to pass the folder path and FileSpec i.e. "goldcustomer *" and "normalcustomer *" respectively and responsibility of "FOREACH LOOP" to get that "FolderPath and FileSpec from "FOR LOOP" and using that FileSpec to loop each file inside that Folderpath starting with "goldcustomer *" and "normalcustomer *" then passing same file name for ETL process. Hope you have understood scenario so let us start our example step by step.

Step 1

Prepare a same example, which we did it our previous article i.e. FOREACH LOOP to know more visit here.

Step 2

Here in this step we will create variable in "Control Flow". Since we have two folders along with their file name so for this we need to create two variables to store two folder-path and two variables to store file-name "goldcustomer *" and "normalcustomer *" respectively. We will add two more variables i.e. FileName and FolderPath to pass values to ForEach Loop container.

Since we are using "FOR LOOP" container it requires two variables i.e. once for Initialization and another for total number of loops to Evaluate Expression variable. So let us add two more variable and name it as IncrementValue and TotalCount.

Assign values like

- FolderPathOne = "C:\Users\gurunatha\Desktop\bulkfiles\GoldCustomer\"
- FolderPathTwo="C:\Users\gurunatha\Desktop\bulkfiles\NormalCustomer\"
- FileNameOne = "GoldCustomer*.* (* means consider anything after GoldCustomer word and .* consider any extension)"
- FileNameTwo = "NormalCustomer*.* (* means consider anything after GoldCustomer word and .* consider any extension)"
- IncrementValue = 1 (Our Initial Value)
- TotalCount = 2 (Since we have two folders)
- FolderPath = Keep it empty we will set up in step 5

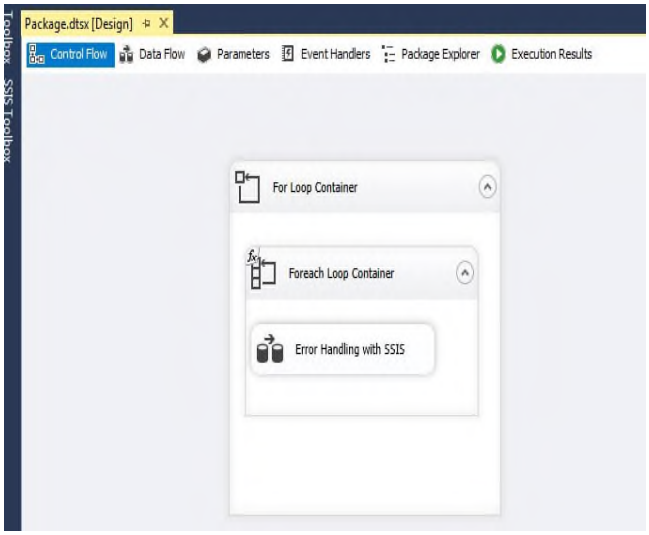
- FileName = Keep it empty we will set up in step 5

Variables

Name	Scope	Data type	Value	Expression
FileName	Package	String	GoldCustomer*.*	@[User::IncrementValue] ...
FileNameOne	Package	String	GoldCustomer*.*	
FileNameTwo	Package	String	NormalCustomer*.*	
FolderPath	Package	String	C:\Users\gurunatha\Desktop\bulkfiles\...	@[User::IncrementValue] ...
FolderPathOne	Package	String	C:\Users\gurunatha\Desktop\bulkfiles\...	
FolderPathTwo	Package	String	C:\Users\gurunatha\Desktop\bulkfiles\...	
IncrementValue	Package	Int32	1	
TotalCount	Package	Int32	2	
VarFullFilePath	Package	String		

Step 3

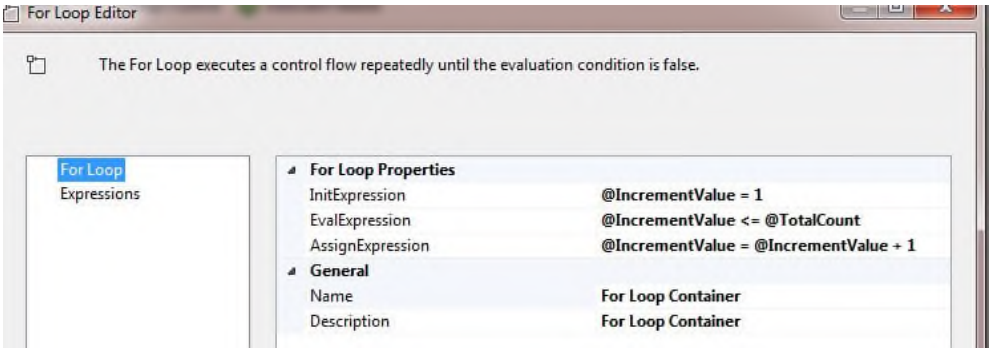
Now just drap and drop FOR LOOP container to control tab and drag complete foreach loop inside the for loop container as shown in below image.



Step 4

Time to configure For Loop so select For loop container and right click on it → Edit → For Loop Properties.

Here we need to specify InitExpression, EvalExpression and AssignExpression so specify it as shown in below image. InitExpression will be IncrementValue (Initial Value) and EvalExpression (Evaluating expression i.e. Initial Value less than or equal to TotalCount) and finally AssignExpression (To increment Initial Value everytime it loops).

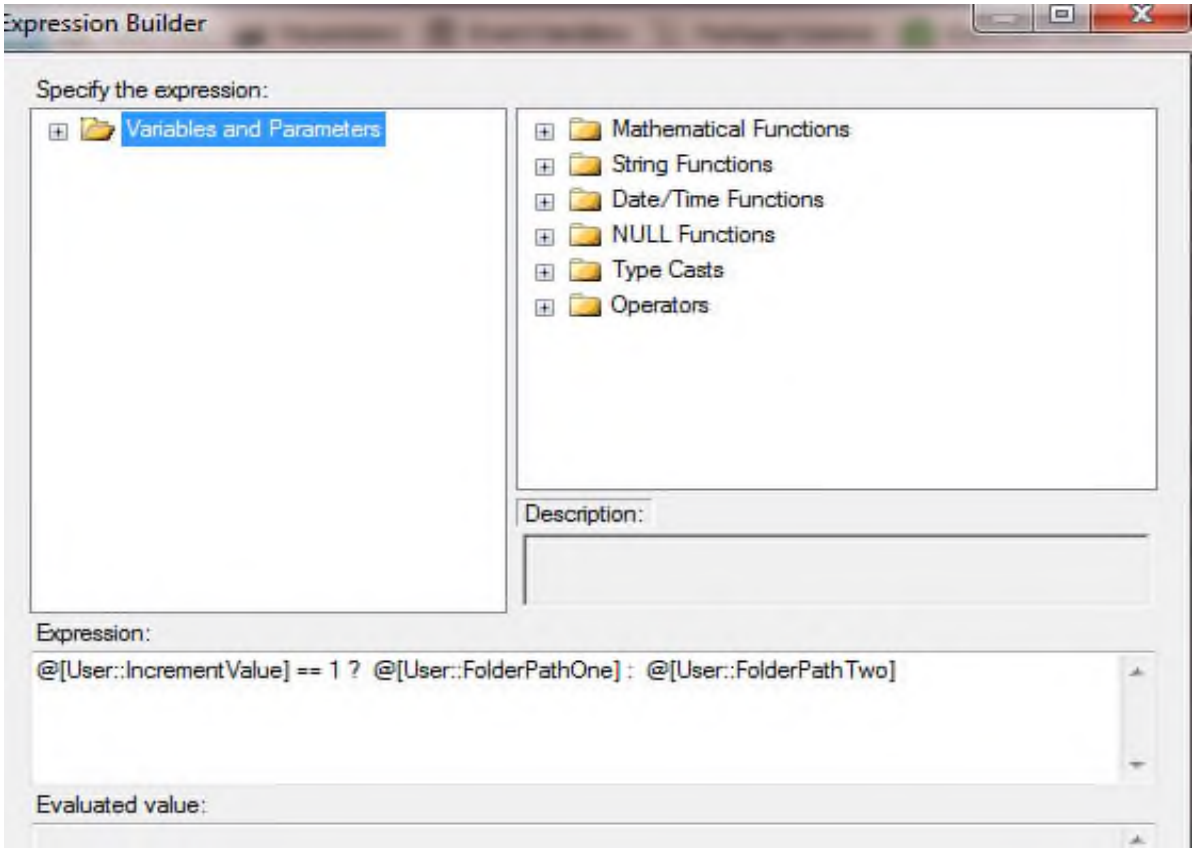


What we are saying above is to loop for loop until Evaluating expression false it means here it will loops two times because our starting intial value is 1 and loop first round it increments to 2 and it cannot loop third round coz Evaluating expression returns false (3 <= 2).

Step 5

In this step, we need to set up two variables i.e. FolderPath and FileName.

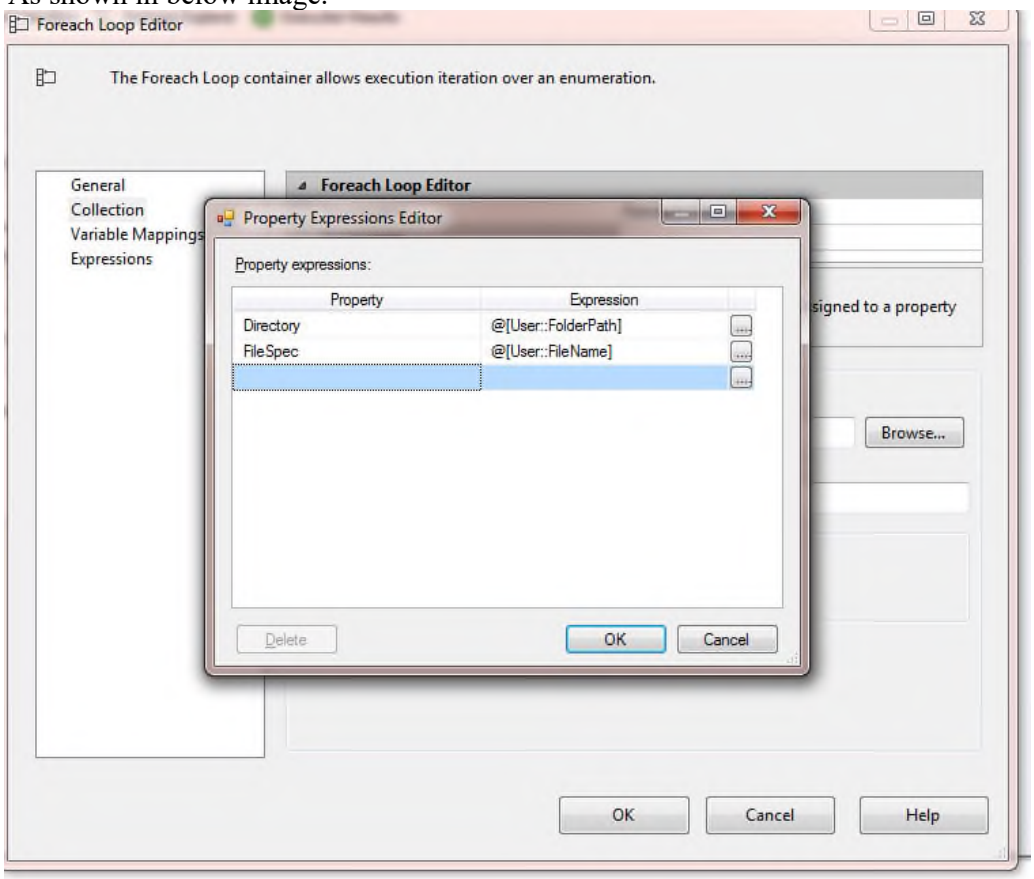
Set up FolderPath variable so go to variable window and choose FolderPath row at end there is expression button so just click on that and in the expression textarea define this expression "@[User::IncrementValue] == 1 ? @[User::FolderPathOne] : @[User::FolderPathTwo]" means when for loop starts looping if Initial Value (IncrementValue) == 1 then consider "FolderPathOne" or else consider "FolderPathTwo".



Set up FileName variable so go to variable window and choose FileName row at end there is expression button so just click on that and in the expression textarea define this expression "@[User::IncrementValue] == 1 ? @[User::FileNameOne] : @[User::FileNameTwo]" means when for loop starts looping if Initial Value (IncrementValue) == 1 then consider "FileNameOne" or else consider "FileNameTwo".

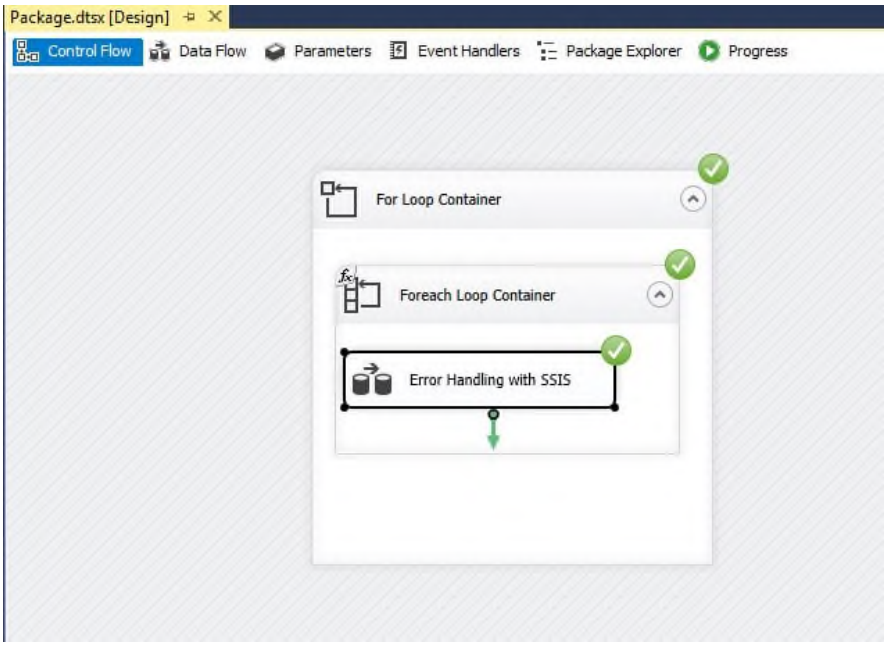
Step 6

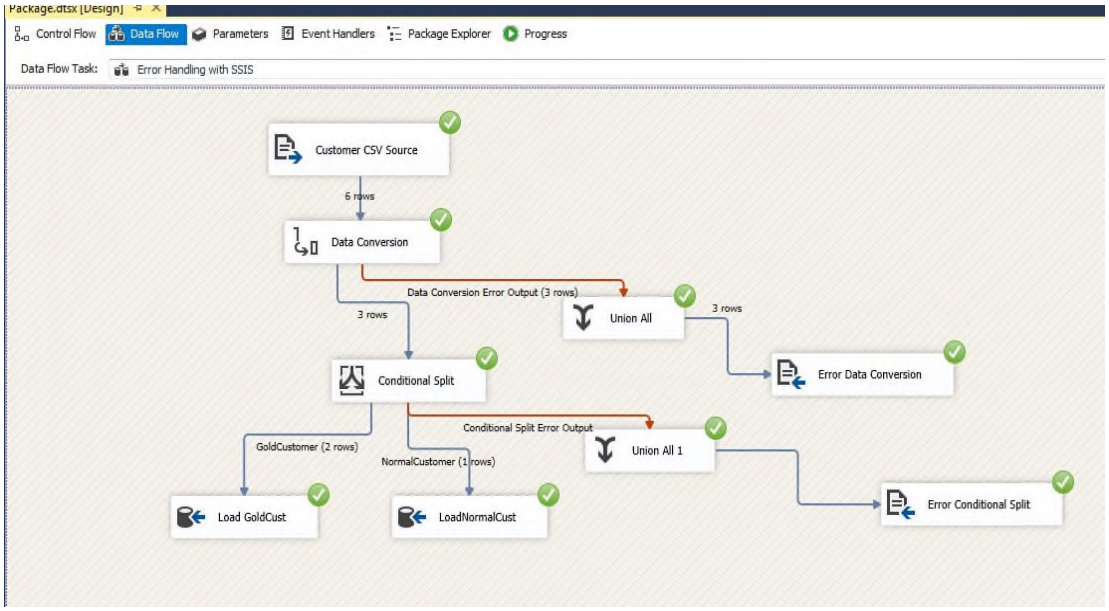
In this step, we need to configure "FOREACH LOOP" container so select on "FOREACH LOOP" right click and edit -> select collection (Left side) and choose Expression and click. A window will popup in the property column specify "Directory" and Expression as FolderPath and same way specify another property "FileSpec" and Expression as FileName. As shown in below image.



Step 7

Save everything because we have confiured both containers properly. Now its time to run the project and check the output.


Package.dtsx [Design]
Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
For Loop Container
Foreach Loop Container
Error Handling with SSIS


Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer
Progress
Data Flow Task:
Error Handling with SSIS
Customer CSV Source
6 rows
Data Conversion
Data Conversion Error Output (3 rows)
3 rows
Union All
3 rows
Error Data Conversion
Conditional Split
Conditional Split Error Output
GoldCustomer (2 rows)
NormalCustomer (1 rows)
Union All 1
Load GoldCust
LoadNormalCust
Error Conditional Split

SQLQuery2.sql - GU...PC\gurunatha (54))* SQLQuery1.sql - GU...PC\gurunatha (53))*

```
/****** Script for SelectTopNRows command from SSMS  ******/
select *
  FROM [CustomerSSIS].[dbo].[NormalCustomer]
```

100 %

Results | Messages

	CustomerID	CustomerName	CustomerCountry	CustomerAmount	CustomerType
1	2002	G_Khadak	USA	-5000.00	4
2	2008	G_Shivam	USA	-2000.00	4
3	3002	N_Khadak	USA	-5000.00	4
4	3008	N_Shivam	USA	-2000.00	4

SQLQuery2.sql - GU...PC\gurunatha (54))* SQLQuery1.sql - GU...PC\gurunatha (53))*

```
/****** Script for SelectTopNRows command from SSMS  ******/
select *
  FROM [CustomerSSIS].[dbo].[GoldCustomer]
```

100 %

Results | Messages

	CustomerID	CustomerName	CustomerCountry	CustomerAmount	CustomerType
1	2001	G_Shiv	India	3500.00	1
2	2004	G_Ajay	Brazil	2500.00	2
3	2007	G_Karan	India	4500.00	1
4	2010	G_Jadhav	Brazil	7500.00	2
5	3001	N_Shiv	India	3500.00	1
6	3004	N_Ajay	Brazil	2500.00	2
7	3007	N_Karan	India	4500.00	1
8	3010	N_Jadhav	Brazil	7500.00	2

As you see we have successfully executed the project executed two folders using "FOR LOOP" and then files inside each folder executed via "FOREACH LOOP" and passing it to destination data source via ETL process.

4 Enforcing Data Quality, Extending SQL Server Integration Services

Topic Covered

1. Introduction to Data Quality
   - Installing and setting up the DQS
   - Creating a Knowledge Base
2. Using Data Quality Service to Cleanse data
3. Using Data Quality Service to match data
4. Using Scripts in SSIS
   - Script Task
     - The Script Task Dts Object
   - Script Component
5. Using Custom components in SSIS

Introduction to Data Quality

- Amount of data is increasing on a regular basis and so is the responsibility to ensure the quality of data remains intact.
- We have incorrect data in our application, which can result from user entry errors, corruption in transmission or storage, mismatched data dictionary definitions, and other data quality and process issues.
- Aggregating data from different sources that use different data standards can result in inconsistent data. Incorrect data affects the ability of a business to perform its business functions and to provide services to its customers, resulting in a loss of credibility and revenue, customer dissatisfaction, and compliance issues.
- Automated systems often do not work with incorrect data, and bad data wastes the time and energy of people performing manual processes. Incorrect data can wreak havoc with data analysis, reporting, data mining, and warehousing
- An organization of any size can use DQS to improve the information value of its data, making the data more suitable for its intended use.
- A data quality solution can make data more reliable, accessible, and reusable. It can improve the completeness, accuracy, conformity, and consistency of data, resolving problems caused by bad data in business intelligence or data warehouse workloads.
- Data Quality Services (DQS) in SQL Server 2012 is new feature used for improving and maintaining the quality of your data across enterprise.
- DQS is a new feature in SQL Server 2012 that provides data quality solutions to the data steward (someone who ensures data quality and makes the data useful for the business) or IT Professionals.
- DQS is knowledge-based solution that provides both interactive or automated/batch mode approaches to manage the integrity and quality of the data.

- **DQS provides the following features to resolve data quality issues.**
  - **Knowledge based** - You can create a reusable Data Quality Knowledge Base (DQKB) which can be used to improve the integrity and quality of the data.
  - **Semantics** - Data is mapped into data domains, which you create and can capture semantics.

- **Knowledge Discovery** - DQS can acquire additional knowledge from sample data and/or from user feedback.
- **Open and Extensible** - Not only you can use your own knowledge base, but you can use third party reference data providers which ensures the quality of your data by comparing it to the data guaranteed by a third-party company. Integration with SSIS is possible for automated match mode execution.
- **Easy to use** - An intuitive and compelling user experience designed for increased productivity.

Data before DQS data cleansing process:

Name	Gender	Street	House Number	Zip code	City	State	Date of Birth
John Doe	Male	60th streat	45		New York	New York	08/12/64
Jane Doe	Male	Jonathan ln	36	10023	Poughkeepsy	NY	21-dec-1954

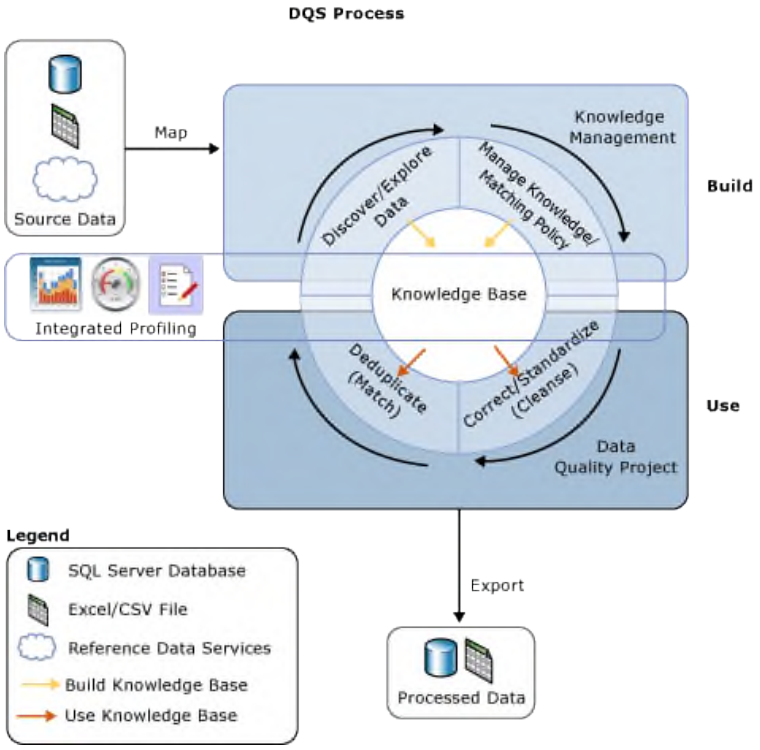
Data after DQS data cleansing process:

Name	Gender	Street	House Number	Zip code	City	State	Date of Birth
John Doe	Male	E 60th St	45W	10022	New York	NY	08/12/64
Jane Doe	Female	Jonathan Lane	36	10023	Poughkeepsie	NY	12/21/54

What are the different SQL Server DQS processes?

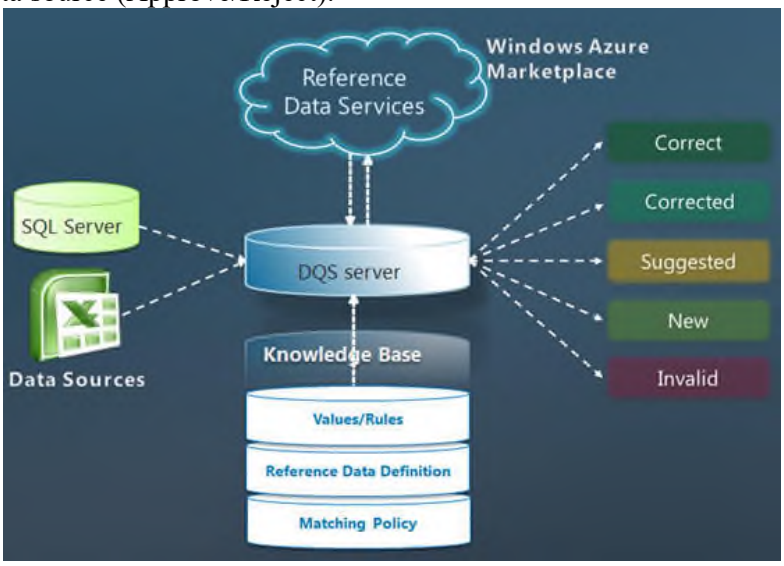
DQS is a knowledge-based tool that allows us to create a knowledge base about the data and then it uses the data in the knowledge base to improve the integrity and quality of the data.

- The following illustration displays the DQS process:



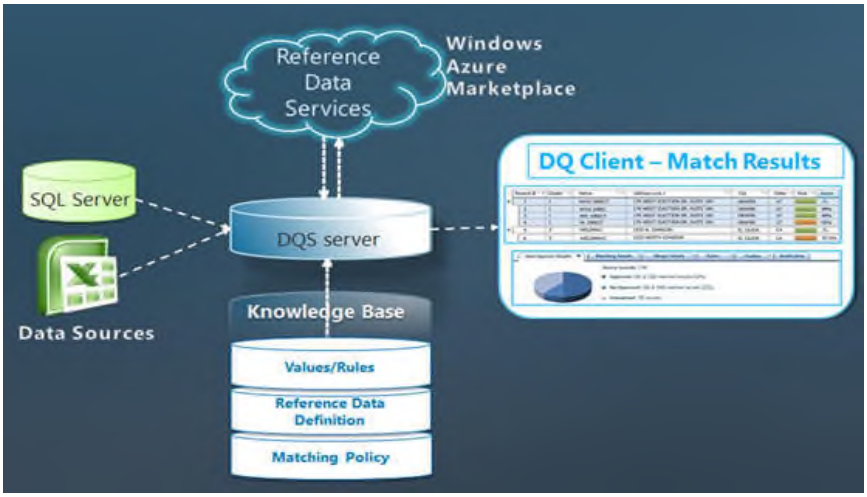
Here are some of the operations that DQS performs or allows us to perform:

- **Monitoring** - It allows tracking and monitoring the state of data quality activities and quality of data; this way you can monitor DQS activities and its progress.
- **Profiling** - It does the analysis of the data source to provide insight into the quality of the data and helps to identify data quality issues at every stage in knowledge discovery, domain management, matching and data cleansing processes.
- **Cleansing** - It can be performed interactively via the DQS Client tool or in batch mode via use of an SSIS component. It updates, amends, removes or enriches data that is incorrect or incomplete. This includes correction, standardization and enrichment. The cleansing process classifies the data to different categories as follows:
    - Correct - Terms/data values that were found correct; for instance, matched a domain value or returned from the RDS (Reference Data Service) as 'correct'.
      Morning
      Automatically activitias
    - Corrected - Terms/data values that were automatically corrected by the DQS engine by using the knowledge base that it uses or the RDS (Reference Data Service) provider.
    - Not Corrected - Terms/data values that were not recognized by the system as 'correct' or terms/ data values that violate a domain rule.
    - Auto suggested - Terms/data values that the DQS engine has found a correction for, but with medium or low confidence and require user intervention to move ahead (Approve/Reject).
    - New - Terms/data values that the DQS engine has identified as new values at the data source (Approve/Reject).



- **Matching** - Identifying, linking or merging related entries within or across sets of data. The goal of matching is to remove duplicates from the source. It can be done intra-source (matching a source against itself) or inter-source (matching a source against a lookup table). A matching process includes four steps:
    - Matching Policy Training - To define and test the set of policy rules that govern the matching process. The matching policy rules help identifying the relevant duplicates.

- Matching - To run a matching project by matching the data at source against itself or against a lookup table.
- Auto-Approve - To define policy rules for auto-approving matching result clusters based on the match results obtained.
- Merge/Survivorship - To define the policy rules for selecting one approved record from each cluster of matched records.



DQS Installation

Applies to: SQL Server (all supported versions) - Windows only
SQL Server Data Quality Services (DQS) contains the following two components: **Data Quality Server** and **Data Quality Client**.

DQS Component	Description
Data Quality Server	Data Quality Server is install on top of the SQL Server Database Engine, and includes three databases: DQS_MAIN, DQS_PROJECTS, and DQS_STAGING_DATA. • DQS_MAIN contains DQS stored procedures, the DQS engine, and published knowledge bases. • DQS_PROJECTS contains the data quality project information. DQS_STAGING_DATA is the staging area where you can copy your source data to perform DQS operations, and then export your processed data.
Data Quality Client	Data Quality Client is a standalone application that enables you to connect to Data Quality Server, and provides you with a highly intuitive graphical user interface to perform data-quality operations, and other administrative tasks related to DQS.

DQS installation is a three-part process:

- Pre-Installation Tasks: Verify system requirements before you install DQS.
- Data Quality Services Installation Tasks: Install DQS by using SQL Server Setup.
- Post-Installation Tasks: Perform these tasks after finishing with the SQL Server Setup to finish installing DQS.

Pre-Installation Tasks
Before installing DQS, make sure that your computer meets the minimum system requirements. The following table provides information about the minimum system requirements for the DQS components:

Pre-Installation Tasks	
DQS Component	**Minimum System Requirements**
Data Quality Server	Memory (RAM): Minimum: 2 GB / Recommended: 4 GB or more SQL Server Database Engine.
Data Quality Client	.NET Framework 4.0 (installed during the Data Quality Client installation, if not already installed) Internet Explorer 6.0 SP1 or later

Data Quality Services Installation Tasks

You have to use the SQL Server Setup to install DQS components. When you run the SQL Server Setup, you have to go through a series of installation wizard pages to select appropriate options based on your requirements. The following table lists only those pages in the installation wizard where the options that you select will affect your installation of DQS:

Data Quality Services Installation Tasks	
Page	**Action**
Feature Selection	Select: **Data Quality Services** under **Database Engine Services** to install the Data Quality Server. If you select the **Data Quality Services** check box, SQL Server Setup will copy an installer file, DQSInstaller.exe, under the SQL Server instance directory on your computer. You must run this file after you have completed the SQL Server Setup to complete the Data Quality Server installation. Also, you must perform some additional steps to configure your Data Quality Server before you can use it. **Data Quality Client** to install Data Quality Client. (Recommended) **Management Tools - Complete** under **Management Tools - Basic** to install SQL Server Management Studio. It provides you a graphical user interface to manage your SQL Server instance, and will aid you in performing additional tasks post installation as listed in the next section.
Database Engine Configuration	Click **Add Current User** to add your user Windows account to the sysadmin fixed server role. This is required for you to be able to run the DQSInstaller.exe file later to complete the Data Quality Server installation.

Post-Installation Tasks

After you complete the SQL Server installation wizard, you must perform additional steps mentioned in this section to complete your Data Quality Server installation, configure it, and then use it.

1. **To complete the Data Quality Server installation, run the DQSInstaller.exe file. On running the DQSInstaller.exe file:**
    - The DQS_MAIN, DQS_PROJECTS, and DQS_STAGING_DATA databases are create.
    - The ##MS_dqs_db_owner_login## and ##MS_dqs_service_login## logins are created.
    - The dqs_administrator, dqs_kb_editor, and dqs_kb_operator roles are created in the DQS_MAIN database.

- The DQInitDQS_MAIN stored procedure is created in the master database.
- DQS_install.log file is typically created in the C:\Program Files\Microsoft SQL Server\MSSQL13.<instance_name>\MSSQL\Log folder. The file contains information about the actions performed on running the DQSInstaller.exe file.
- If a Master Data Services database is present in the same SQL Server instance as Data Quality Server, a user mapped to the Master Data Services login is created, and is granted the dqs_administrator role on the DQS_MAIN database.

This completes the Data Quality Server installation.

2. Grant DQS Roles to Users:

To log on to Data Quality Server using Data Quality Client, a user must have any of the following three roles on the DQS_MAIN database:

- **dqs_administrator**
- **dqs_kb_editor**
- **dqs_kb_operator**

By default, if your user account is a member of the sysadmin fixed server role, you can log on to Data Quality Server using Data Quality Client even if none of the DQS roles are granted to your user account.

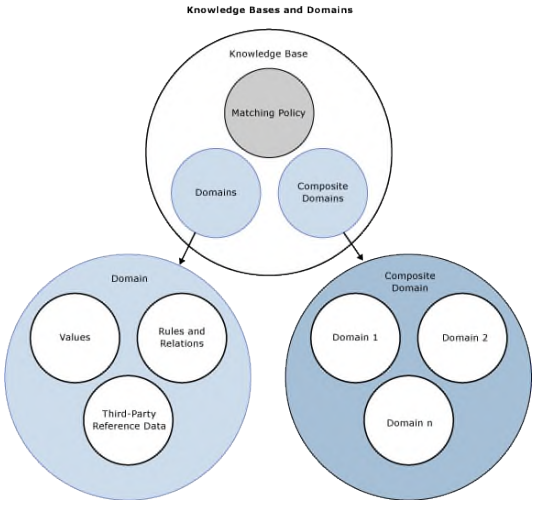
Make your data available for DQS operations. Make sure that you can access your source data for the DQS operations, and can export the processed data to a table in a database.

Creating a Knowledge Base

- To prepare knowledge for a data quality project, you build and maintain a knowledge base (KB) that DQS can use to identify incorrect or invalid data.
- DQS enables you to use both computer-assisted and interactive processes to create, build, and update your knowledge base.
- Knowledge in a knowledge base is maintain in domains, each of which is specific to a data field. The knowledge base is a source of knowledge about your data that enables you to understand your data and maintain its integrity.
- **DQS knowledge bases have the following benefits:**
  - Building knowledge about data is a detailed process. The DQS process of extracting knowledge about data automatically, from sample data, makes the process much easier.
  - DQS enables you to see its analysis of the data, and to augment the knowledge in the knowledge base by creating rules and changing data values. You can do so repeatedly to improve the knowledge over time.
  - You can leverage pre-existing data quality knowledge by basing a knowledge base on an existing KB, importing domain knowledge from files into the KB, importing knowledge from a project back into a KB, or using the DQS default KB, DQS Data.
  - You can ensure the quality of your data by comparing it to the data maintained by a reference data provider.
  - There is a clear separation between building a knowledge base and applying it in the data correction process, which gives you flexibility in how you build and update the knowledge base.

The data steward uses the Data Quality Client application both to execute and control the computer-assisted steps, and to perform the interactive steps.

The following illustration displays various components in a knowledge base and a domain in DQS:



How to Create and Build a DQS Knowledge Base

Building a DQS knowledge base involves the following processes and components:

Knowledge Discovery a computer-assisted process that builds knowledge into a knowledge base by processing a data sample

Domain Management An interactive process that enables the data steward to verify and modify the knowledge that is in knowledge base domains, each of which is associated with a data field. This can include setting field-wide properties, creating rules, changing specific values, using reference data services, or setting up term-based or cross-field relationships.

Reference Data Services a process of domain management that enables you to validate your data against data maintained and guaranteed by a reference data provider.

Matching Policy a policy that defines how DQS processes records to identify potential duplicates and non-matches, built into the knowledge base in a computer-assisted and interactive process.

Using Data Quality Service to cleanse data

- Data Quality Services is a Cleansing transformation in the SSIS.
- The complexity in data is bad data entered by the End-user. The data is good or bad it depends on the end-user.
- The DQS is ensuring to the End-user or programmer that data quality is good. DQS is a knowledge base Tool it has an ability to detect the good data.
- DQS is some kind of tool or power or service give to the End-user or programmer so they enter good data.
- As DQS allows us to create a knowledge base by discovering, building and managing the information or knowledge about the data, we will first create a knowledge base, then we will use that knowledge base for cleansing the data. Like Creating a Data Quality Knowledge Base (DQKB)

Steps for the Data Quality Services as Follows -

- There are two types of Data Quality Services we want to open. First is Data Quality Server Installer and the second one is the Data Quality Client.

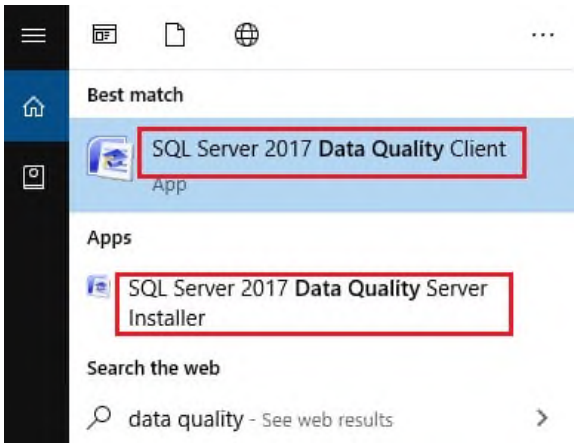


Figure: Data Quality Services

- Firstly, Enable the DQS Server Installer→for that first install the DQS.

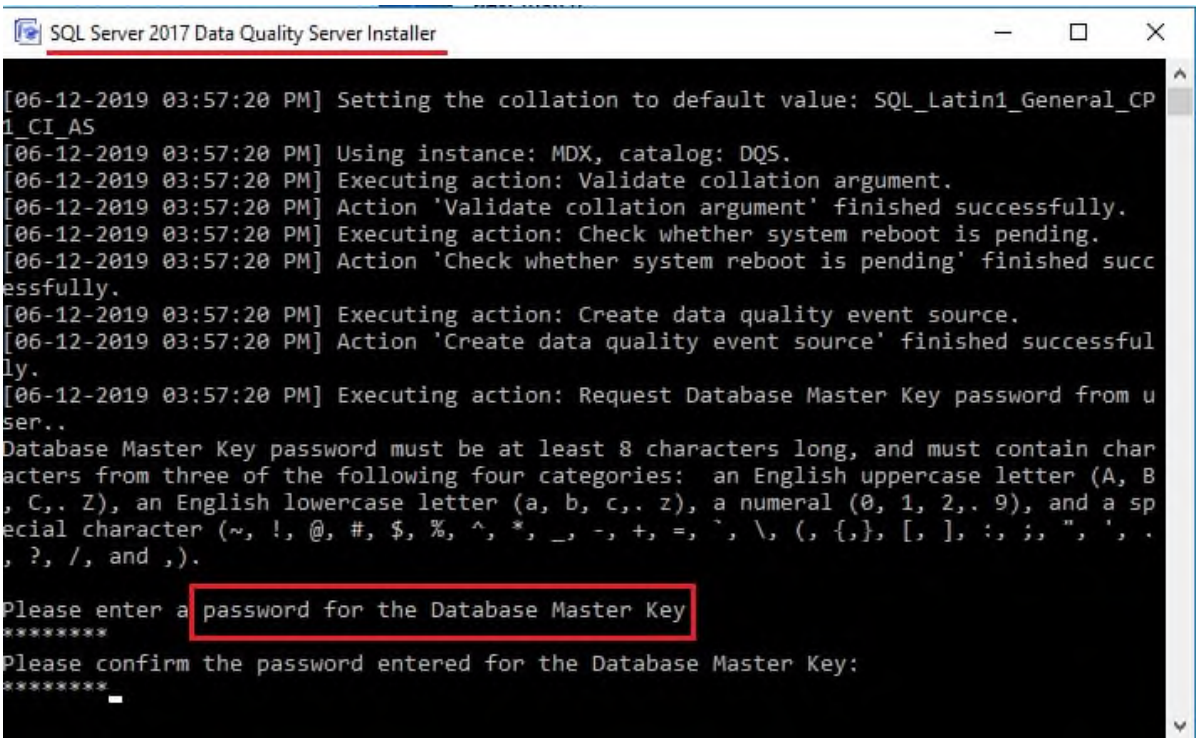


Figure: Installation of DQS Server

- After that, It asks the password for the database master key →Set your password →Confirm the password →After some time it will be Executed →It will show a message like Follows.

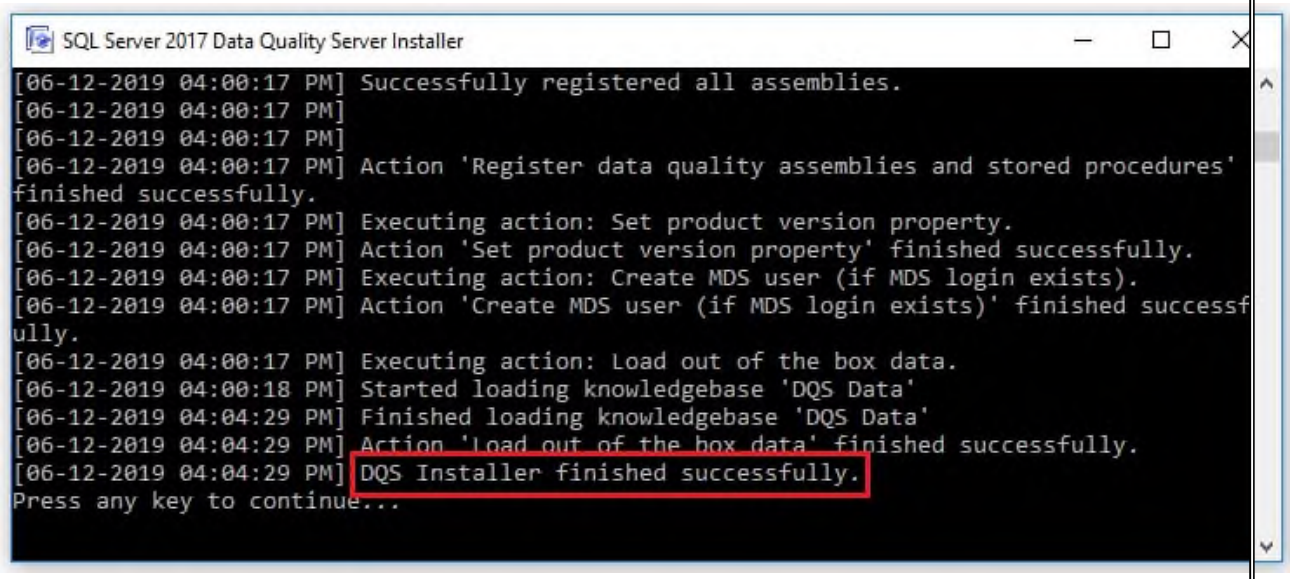


Figure: Installation Completed

- It creates a new three databases in the SQL as follows-

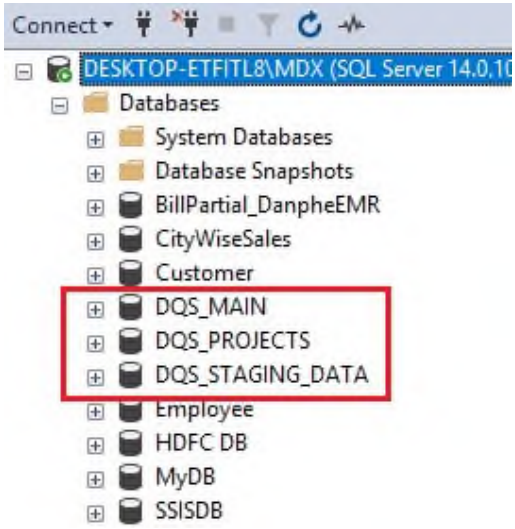


Figure: Data Quality Services Databases

- Thereafter open the DQS Client →We need to connect to the DQS server and choose the knowledge base that we will be using for our source input data →Choose the server name →Click on Connect.

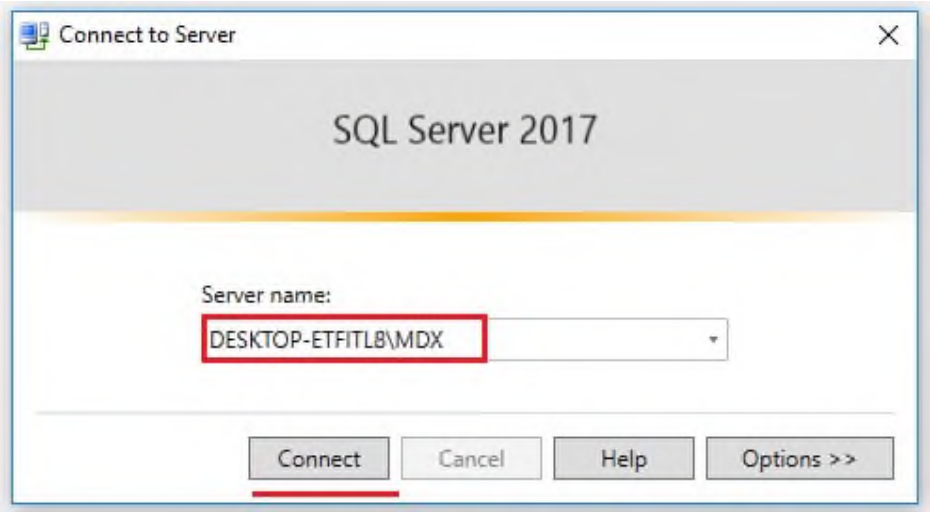


Figure: Connection Manager

- My source data is like follows it is bad data.

	PName	Price
1	Shooes	200
2	Puma	100
3	Cloth	400
4	item7	380
5	Shoes	150
6	1234	300
7	NULL	400

Figure:Original data

- Then we can perform three primary tasks with DQS Client→One is Knowledge Base Management → Second one is Data Quality Projects and→The last one is Administration

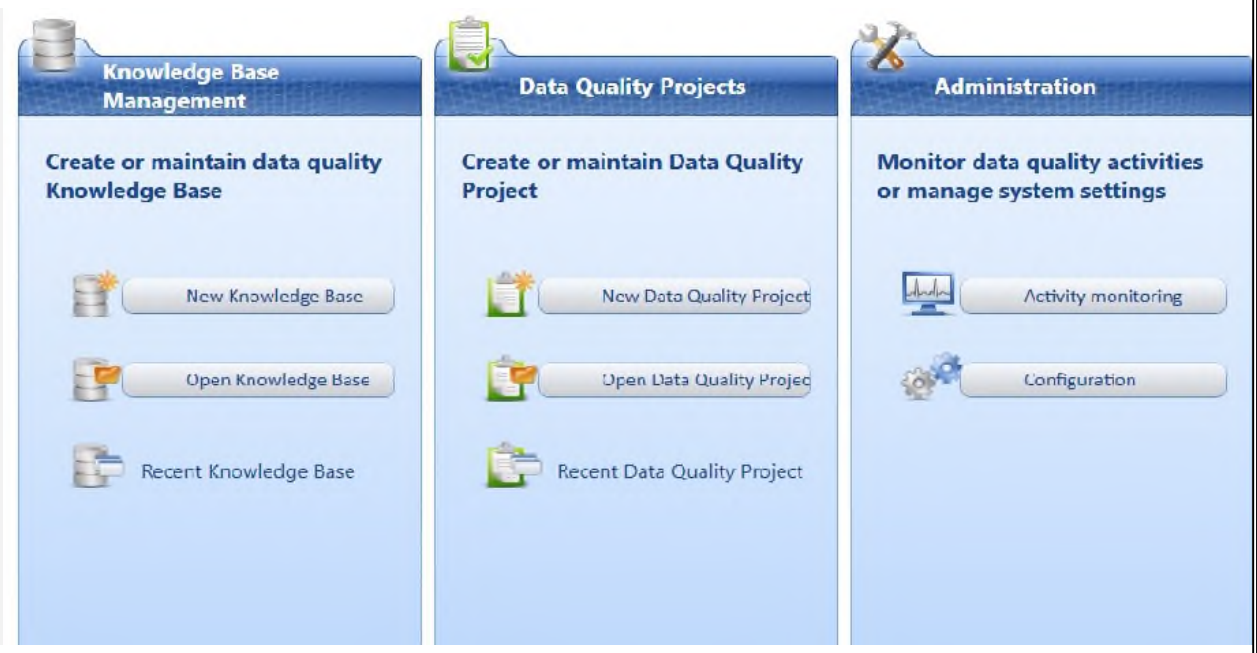


Figure: DQS Client primary Three Tasks

- In the Knowledge Base Management, we define the data cleansing rules and policies →Click on New Knowledge Base.

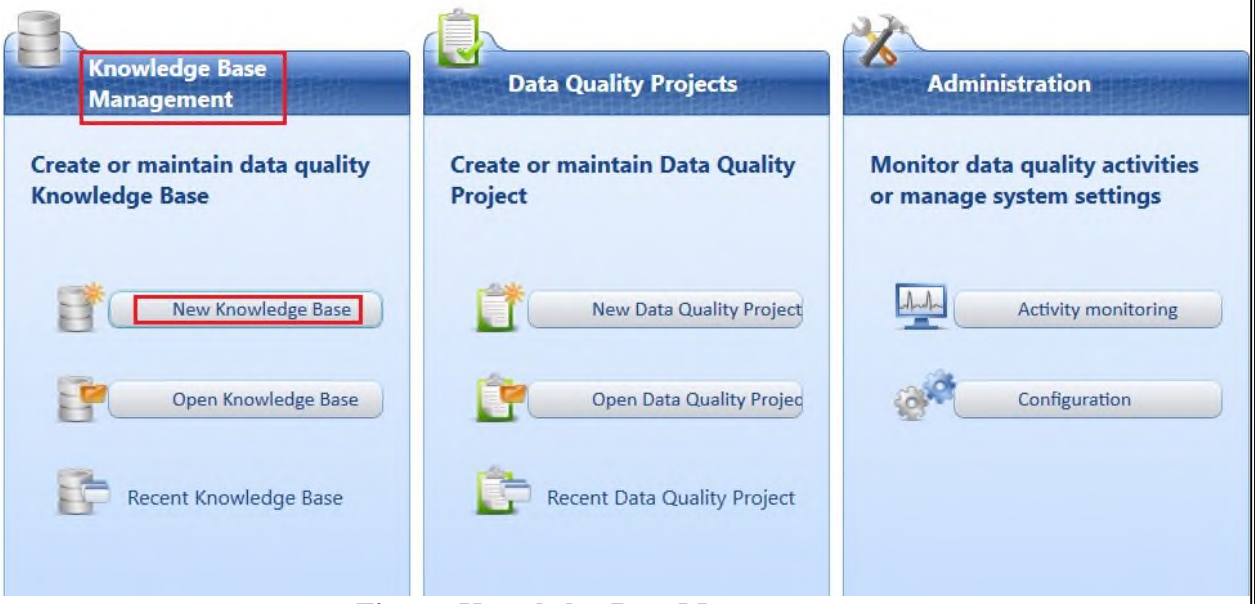


Figure: Knowledge Base Management

- Give the name → Select the activity Domain Management →Click on Next.

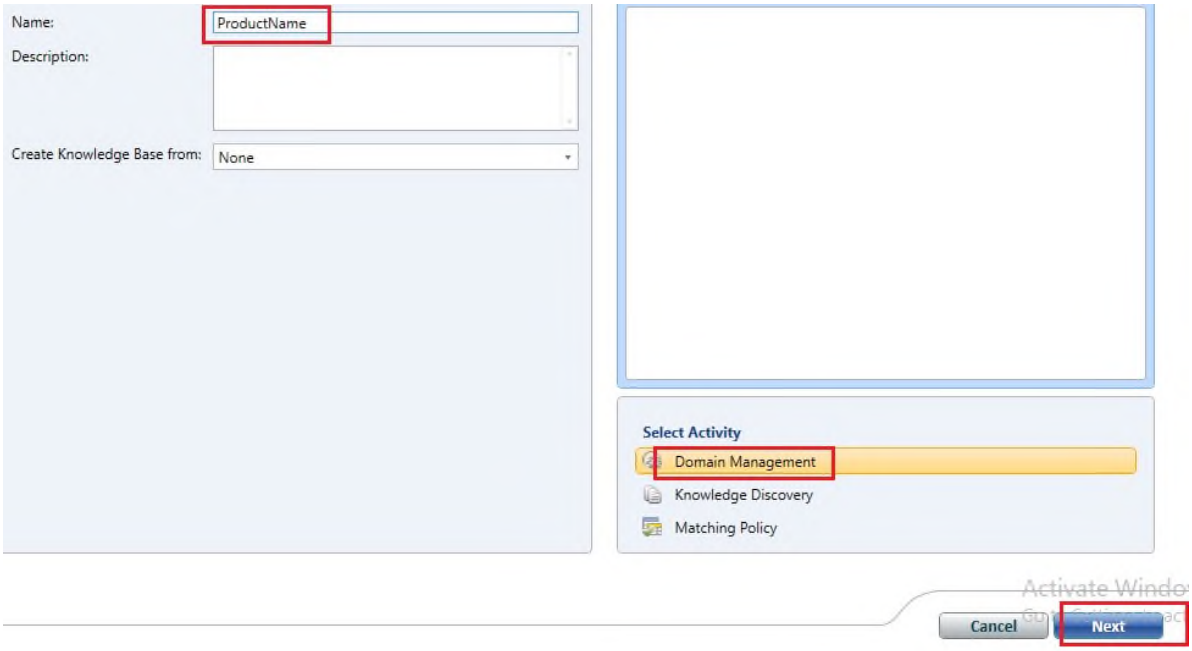


Figure: Data Quality Client

- Then it opens the following Domain Management window →Create a Domain.

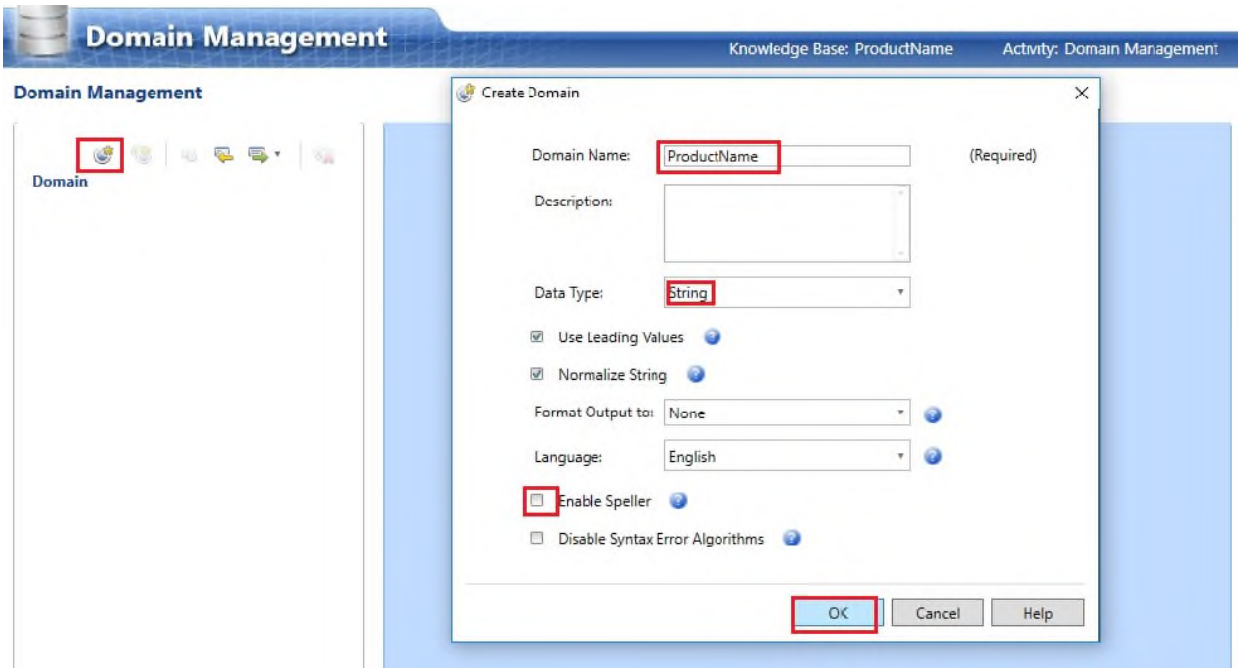
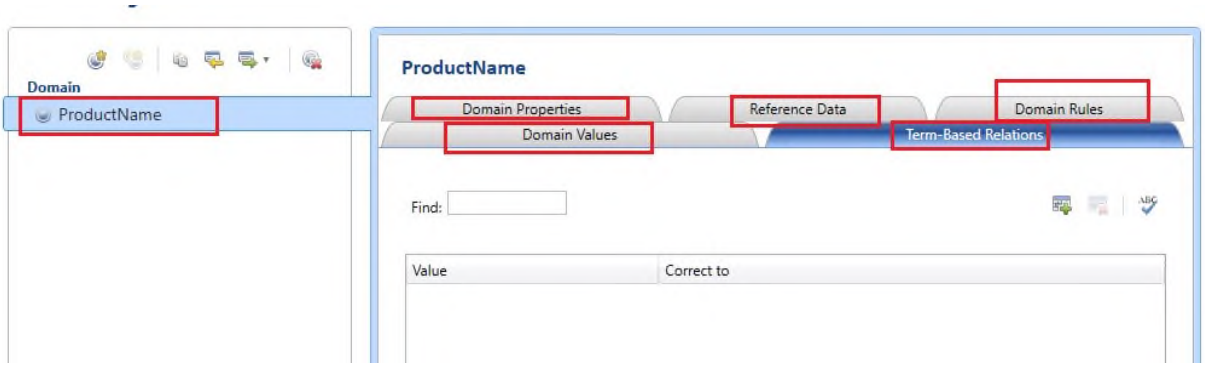


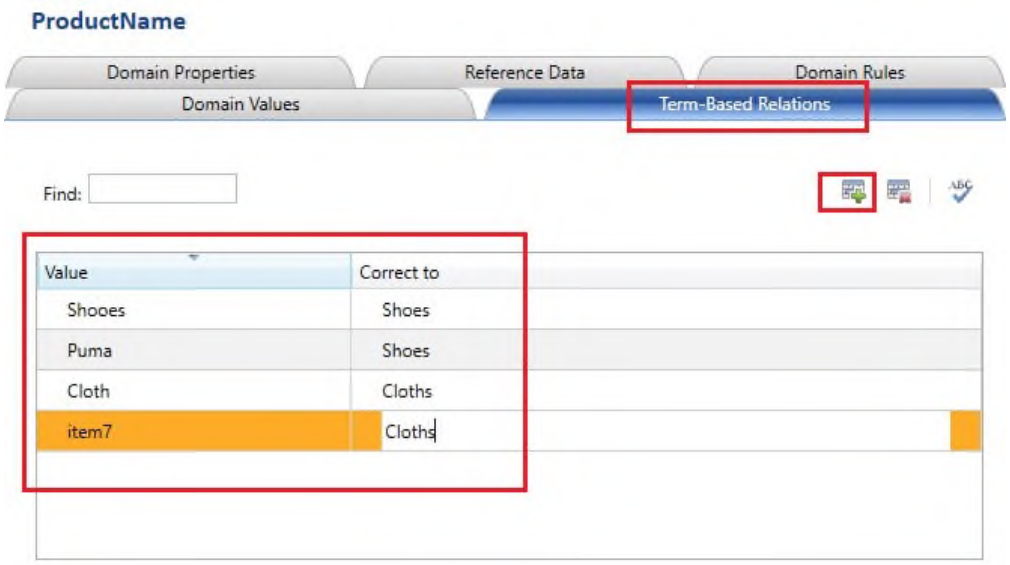
Figure: Create Domain

- A DQS KB(Knowledge base) is a grouping of related data quality definitions and rules it is called domains that are defined up front.
- They are Five Tabs like follows screenshot. These definitions and rules are applied against data with various outcomes such as corrections, exceptions, etc.



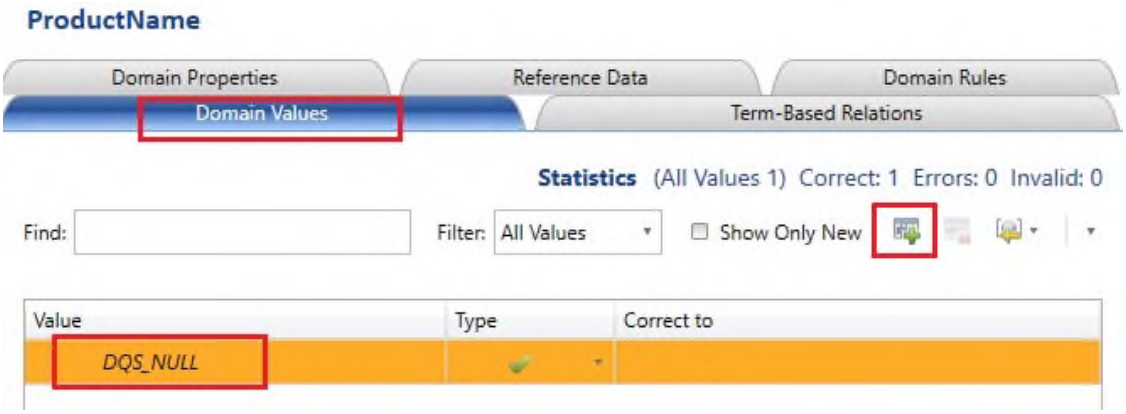
Domains

- A domain defines what the DQS engine should do with data it receives→Is it valid?→Does it need to be corrected?→Should it look at external services to cleanse the data?



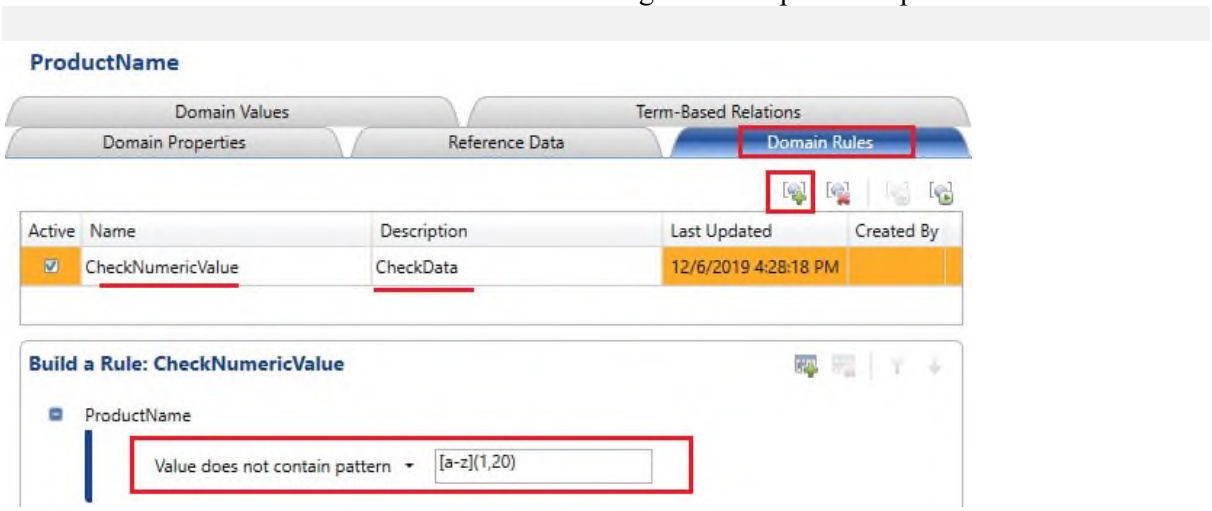
Term-Based Relations

- In the Domain Values, we can correct the values like city name, country name, etc.



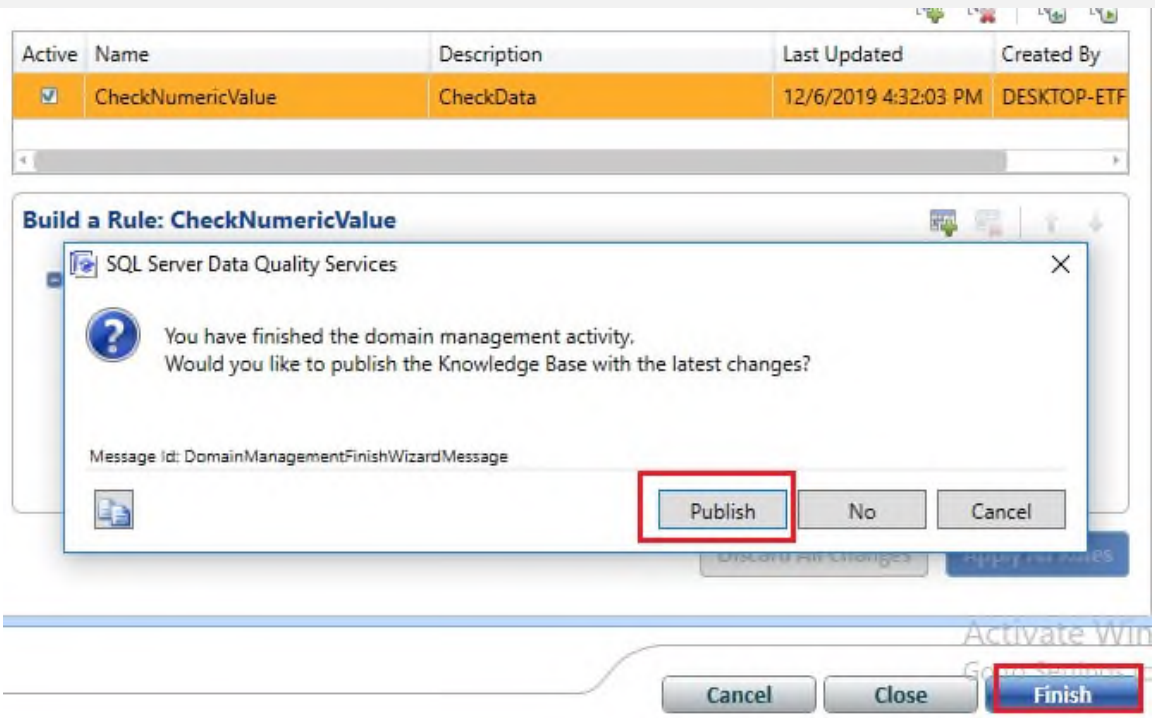
Domain Values

- Domain rules include range rules to check that numeric data values fall within a given range and value lists to make sure that the data coming in meets specific requirements.



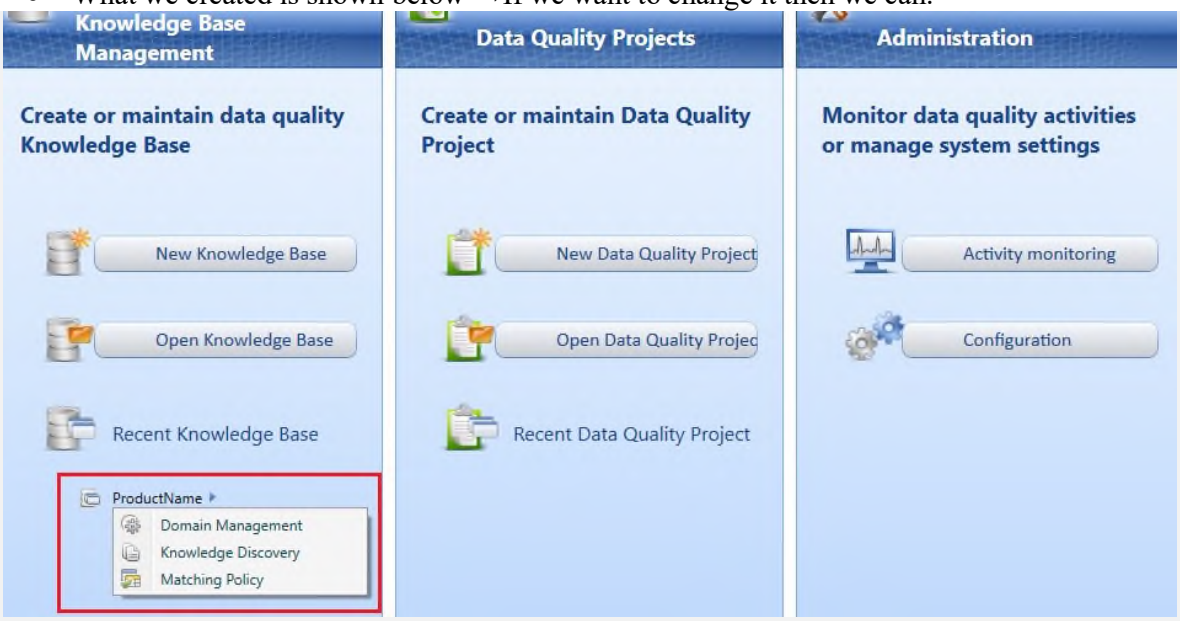
Domain Rules

- Subsequently, Publish these rules →Click on Publish →Click on the Finish.



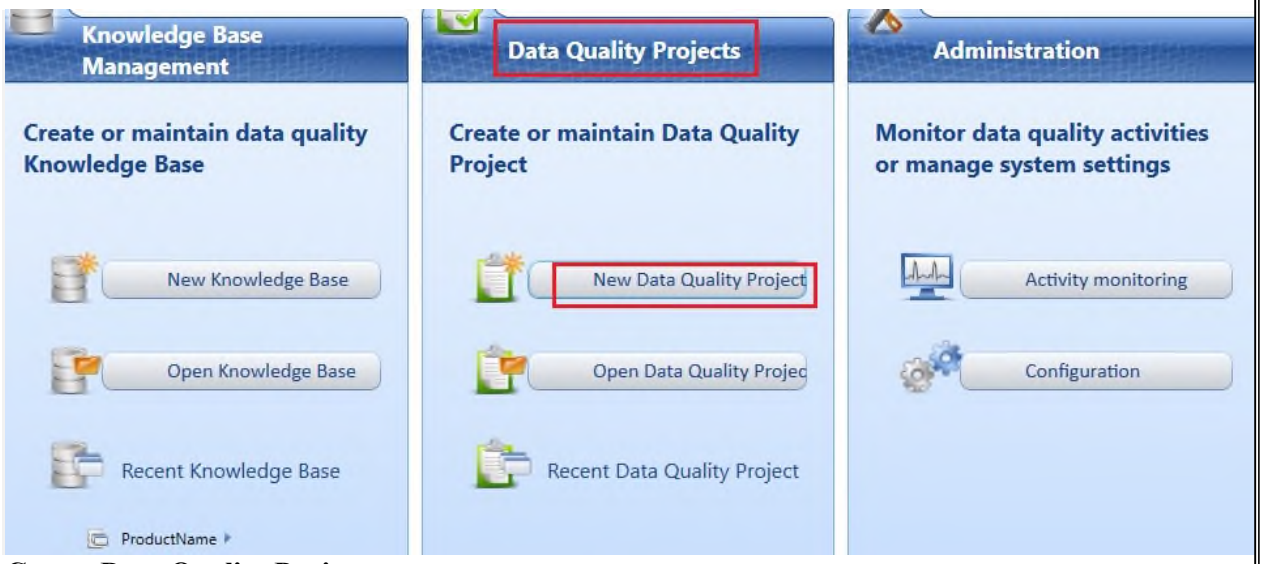
Publishing Rules

- What we created is shown below →If we want to change it then we can.



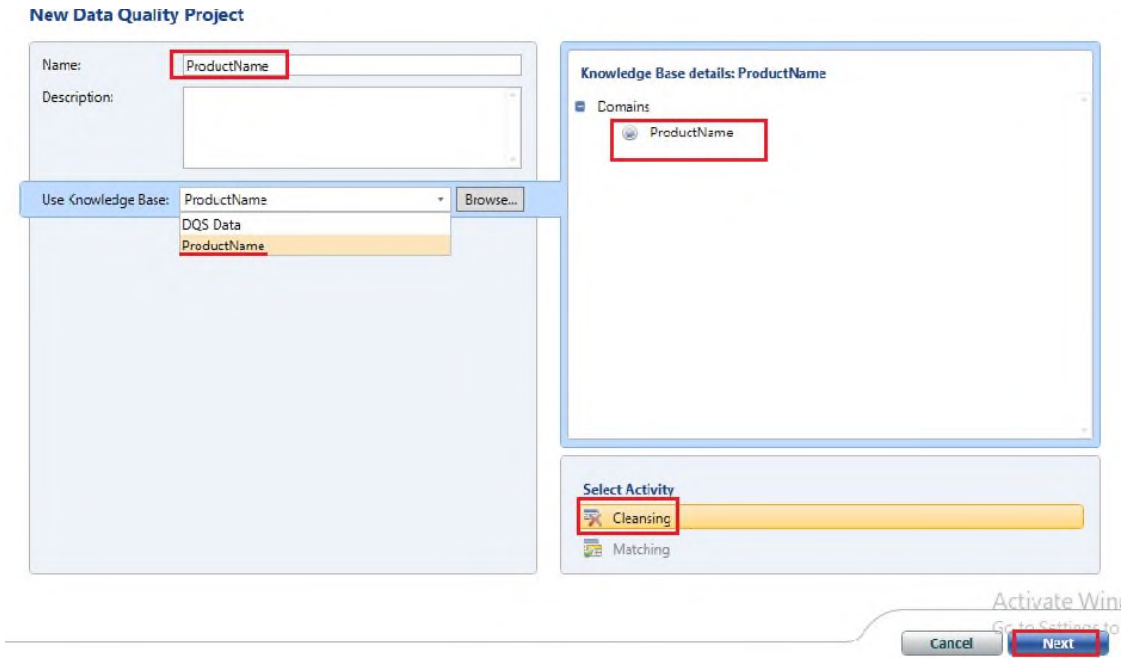
Rules are Published

- Data Quality Projects, we use for applying the data quality definitions from the knowledge base against real data.



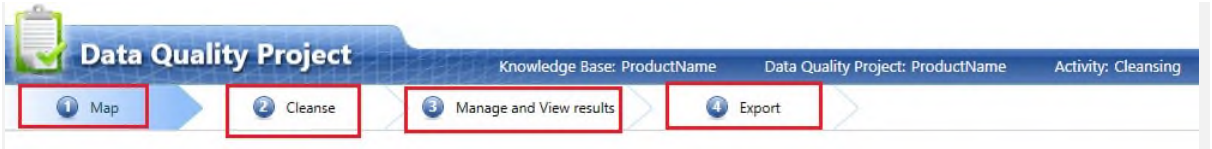
Create Data Quality Projects

- Create a New Data Quality Project →Give a Name →Select the Cleansing activity →Click on Next.



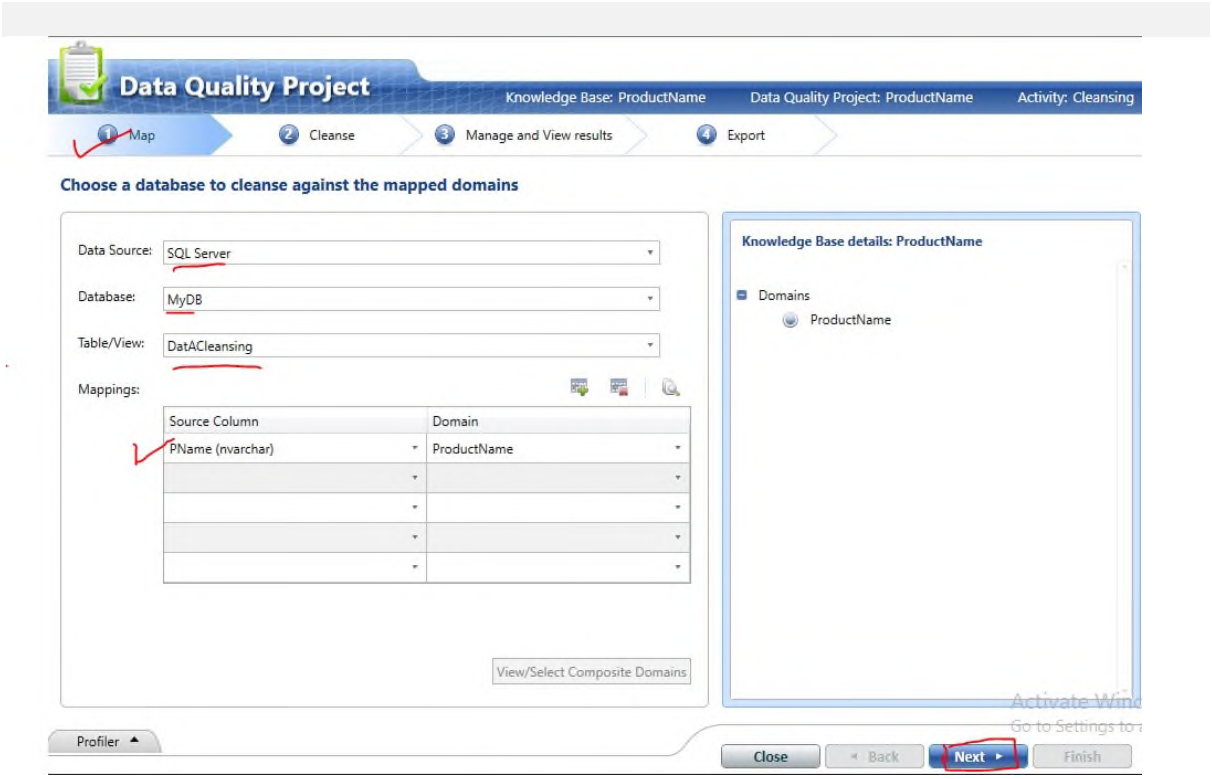
Create a New Data Quality Project

- It will be Processing in the Four steps →1. Map →2. Cleanse →3. Manage and View Results →Export.



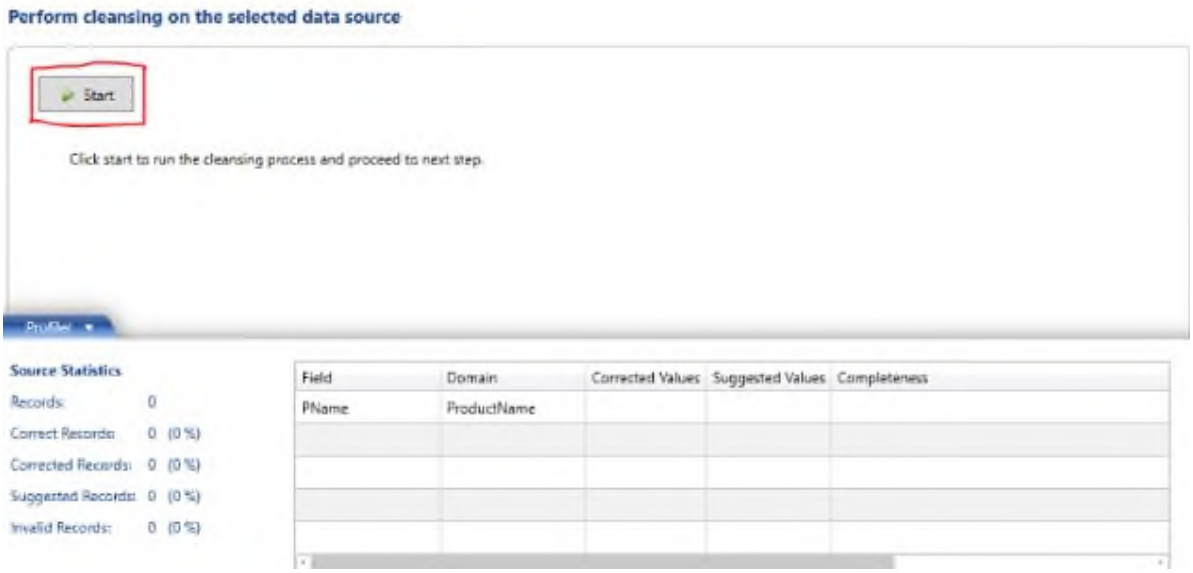
Data Quality Project

- Then select the Source →In the Mapping →Select source column and Domain →Click on Next.



Mapping

- Click on Start →It will start to perform a cleansing on the selected data source.



Perform Cleansing

- It shows the result →In the new there are 5 columns.

ProductName

Suggested (0) | New (5) | Invalid (0) | Corrected (1) | Correct (1)

Search Value:

Value	# Records	Correct to	Confidence	Reason	Approve
Puma	1	Shoes	100%	Term based relation	
item7	1	Cloths	100%	Term based relation	
Cloth	1	Cloths	100%	Term based relation	
1234	1		0%	New value	

Records containing the value:

Correct to	Confidence	Reason	Approve	Reject	PName

OUTPUT

- If data confidence is 100% then we can approve the data so it can go into the corrected data.

ProductName

Suggested (0) | New (3) | Invalid (0) | Corrected (3) | Correct (1)

Search Value:

Value	# Records	Correct to	Confidence	Reason	Approve
Shoes	1		0%	New value	
Cloth	1	Cloths	100%	Term based relation	
1234	1		0%	New value	

OUTPUT

- If data is invalid then in the correct to fields we can correct the data and then click on approve.

ProductName

Suggested (0) | New (0) | Invalid (2) | Corrected (5) | Correct (0)

Search Value:

Value	# Records	Correct to	Confidence	Reason	Approve
DQS_NULL	1	Cloths	100%	Modified by user	
Shoes	1		0%	Modified by user	

OUTPUT

- Finally, all data is a load to the destination for this →Give the destination type →Browse the path →and click on Export →Finish.

Browse the DESTINATION

- We can see that good data is generate in the destination file.

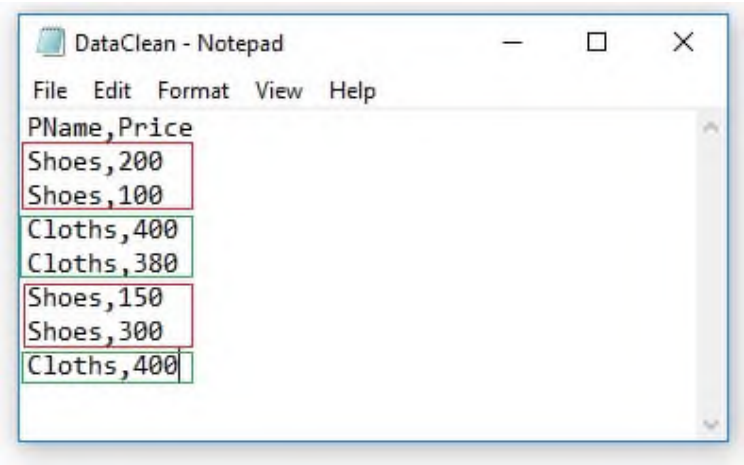


Figure: Destination File

- If we want to change this then we can able to like ProductName in the screenshot →Click on ProductName.

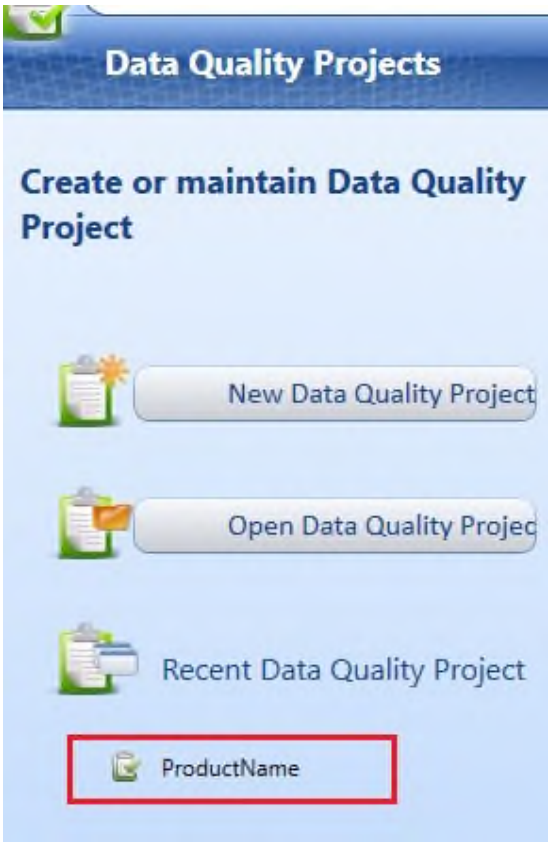


Figure: Data Quality Project

- SSIS provides a transformation component DQS Cleansing. With the help of this, we can create good data.

Figure: SSIS Component

- Right-click on DQS Cleansing →It opens the DQS Cleansing Transformation Editor →Click on Mapping →Choose the column name →Select the Domain.

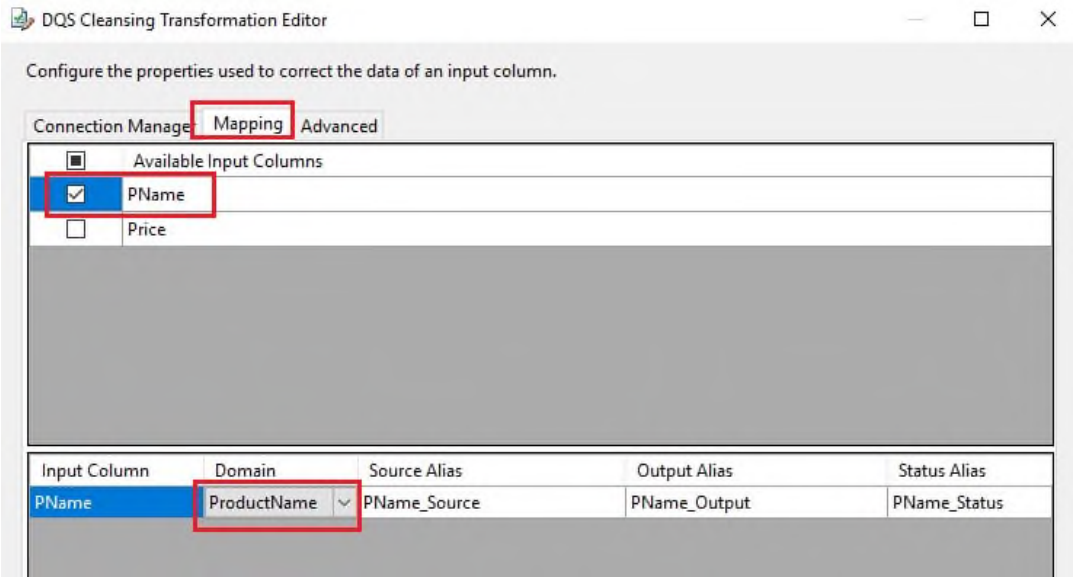


Figure: DQS Cleansing Transformation Editor

- IN THE DQS Cleansing Transformation Editor →Go to the connection manager →select the Data Quality connection manager and → Data Quality Knowledge Base →Click on OK.

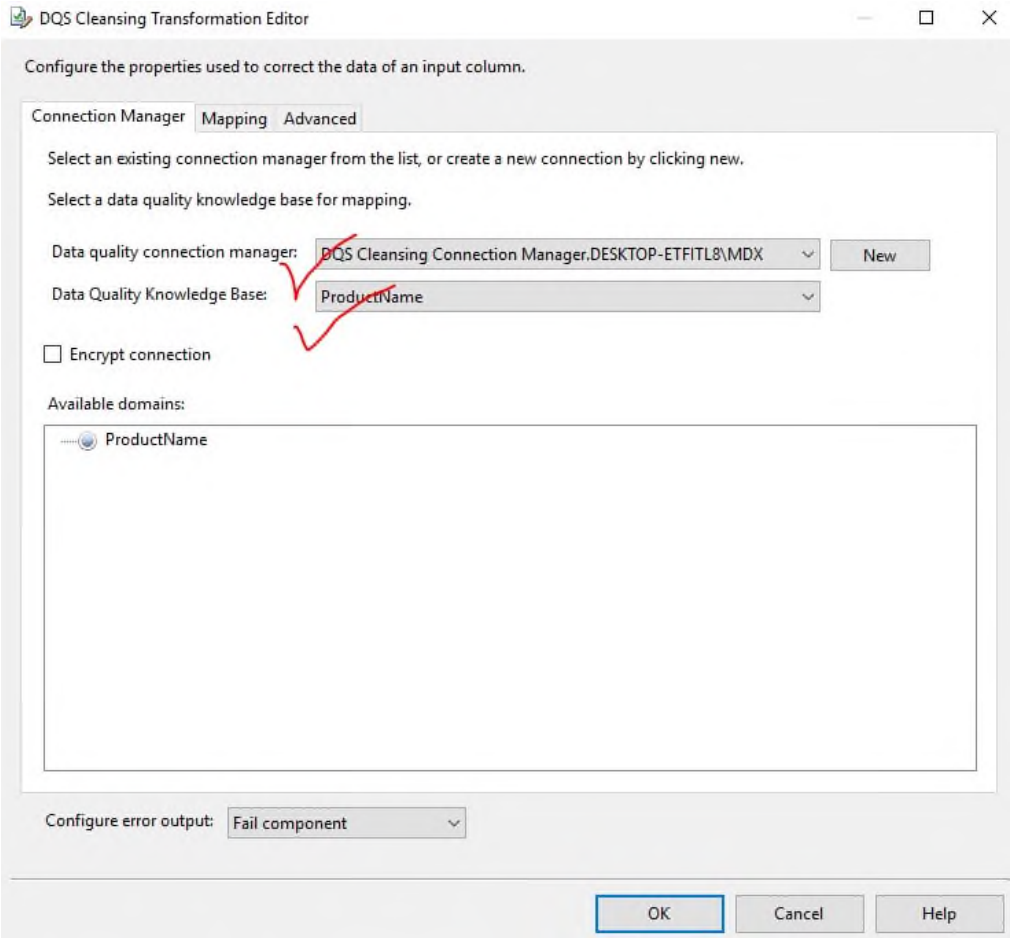


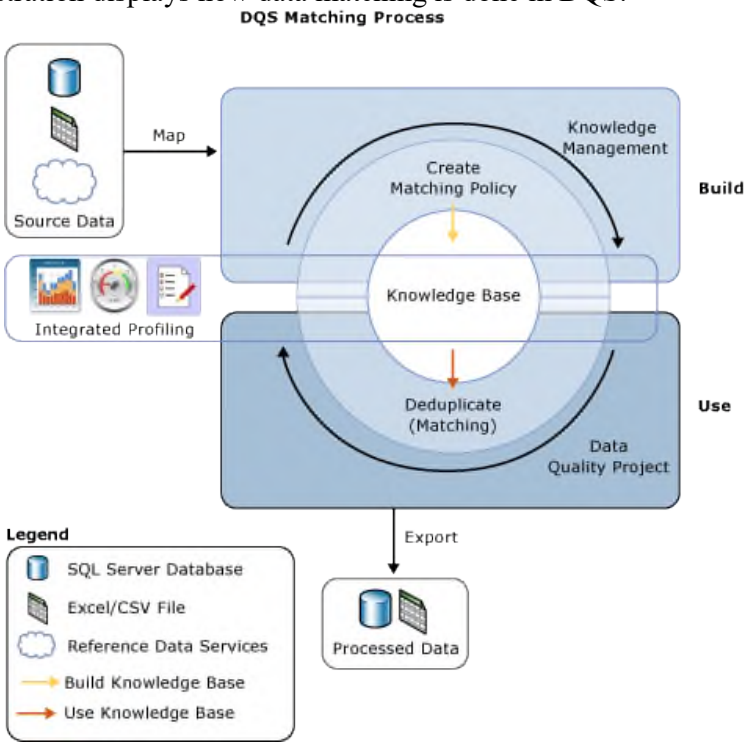
Figure: DQS Cleansing Transformation Editor

- Click on Start →It Execute the package →We can see the output in the destination file with the good data.

Figure: Package Execution

Using Data Quality Service to Match data

- The Data Quality Services (DQS) data matching process enables you to reduce data duplication and improve data accuracy in a data source. Matching analyzes the degree of duplication in all records of a single data source, returning weighted probabilities of a match between each set of records compared. You can then decide which records are matches and take the appropriate action on the source data.
- To prepare DQS for the data deduplication process, you must create a matching policy that DQS will use to calculate the probability of a match.
- The DQS matching process has the following benefits:
  - Matching enables you to eliminate differences between data values that should be equal, determining the correct value and reducing the errors that data differences can cause. For example, names and addresses are often the identifying data for a data source, particularly customer data, but the data can become dirty and weaken over time. Performing matching to identify and correct these errors can make data use and maintenance much easier.
  - Matching enables you to ensure that values that are equivalent, but were entered in a different format or style, are rendered uniform.
  - Matching identifies exact and approximate matches, enabling you to remove duplicate data as you define it. You define the point at which an approximate match is in fact a match. You define which fields are assessed for matching, and which are not.
  - DQS enables you to create a matching policy using a computer-assisted process, modify it interactively based upon matching results, and add it to a knowledge base that is reusable.
  - You can re-index data copied from the source to the staging table, or not re-index, depending on the state of the matching policy and the source data. Not re-indexing can improve performance.
- You can perform the matching process in conjunction with other data cleansing processes to improve overall data quality. You can also perform data de-duplication using DQS functionality built into Master Data Services.
- The following illustration displays how data matching is done in DQS:



How to Perform Data Matching

Require

- You must have created a knowledge base with a matching policy consisting of one or more matching rules.
- Microsoft Excel must be installed on the Data Quality Client computer if the source data to be matched is in an Excel file. Otherwise, you will not be able to select the Excel file in the mapping stage. The files created by Microsoft Excel can have an extension of .xlsx, .xls, or .csv. If the 64-bit version of Excel is used, only Excel 2003 files (.xls) are supported; Excel 2007 or 2010 files (.xlsx) are not supported. If you are using 64-bit version of Excel 2007 or 2010, save the file as an .xls file or a .csv file, or install a 32-bit version of Excel instead.

Security Permissions

- You must have the dqs_kb_editor or the dqs_administrator role on the DQS_MAIN database to run a matching project.
  You perform the matching activity in a data quality project that you create in the DQS client application.

Mapping Stage

In the mapping stage you identify the source of the data that you will run the matching analysis on, and you map source columns to domains to make the domains available for the matching activity.

1. On the **Map** page, to run matching on a database, leave **Data Source** as **SQL Server**, select the database that you want to run matching on, and then select the table. The source database must be present in the same SQL Server instance as the DQS server. Otherwise, it will not appear in the drop-down list.

Matching Stage

In this stage, you perform a computer-assisted matching process that shows you how many matches there are in the source data based upon the matching rules. This process will generate a matching results table that shows the clusters that DQS has identified, each record in the cluster with its record ID and its matching score, and the initial leading record for the cluster.

Survivorship and Exporting Stage

In the survivorship process Data Quality Services determines a survivor record for each cluster, which will replace the other records that match it in the cluster. It then exports the matching and/or survivorship results to a table in the SQL Server database, a .csv file, or an Excel file. Survivorship is optional.

Profiler Tab

Click the **Profiler** tab to display statistics for the source database and for each field included in the policy rule. The statistics will be updated as the policy rule is run. Profiling will help you assess the effectiveness of the de-duplication process, helping determine the extent to which the process is able to improve the quality of the data. Accuracy in profiling is not important for a matching project.

The source database statistics include the following:

- **Records**: The total number of records in the database
- **Total Values**: The total number of values in the fields

- **New Values**: The total number of values that are new since the previous run, and their percentage of the whole
- **Unique Values**: The total number of unique values in the fields, and their percentage of the whole
- **New Unique Values**: The total number of unique values that are new in the fields, and their percentage of the whole

The field statistics include the following:

- **Field**: Name of the field that was included in the mappings.
- **Domain**: Name of the domain that was mapped to the field.
- **New**: The number of new matches found and their percentage of the total
- **Unique**: The number of unique records in the field and their percentage of the total
- **Completeness**: The percentage that the rule run is complete.

Matching Policy Notifications

For the matching policy activity, the following conditions result in notifications:

- The field is empty in all records; it is recommended that you eliminate it from mapping.
- The field completeness score is very low; you may want to eliminate it from mapping.
- All values in a field are invalid; you should verify the mapping and the relevancy of domain rules to the field contents.
- There is a low level of valid values in the field; you should verify the mapping and the relevancy of domain rules to the field contents.
- There is a high level of uniqueness in this field. Using this field in matching policy can decrease the matching results.

Matching Rules Tab

Click this tab to display a list of the rules in the matching policy and the conditions in a rule.

RulesList

Displays a list of all matching rules in the matching policy. Select one of the rules to display the conditions in the rule in the Matching Rule table.

Matching Rule Table

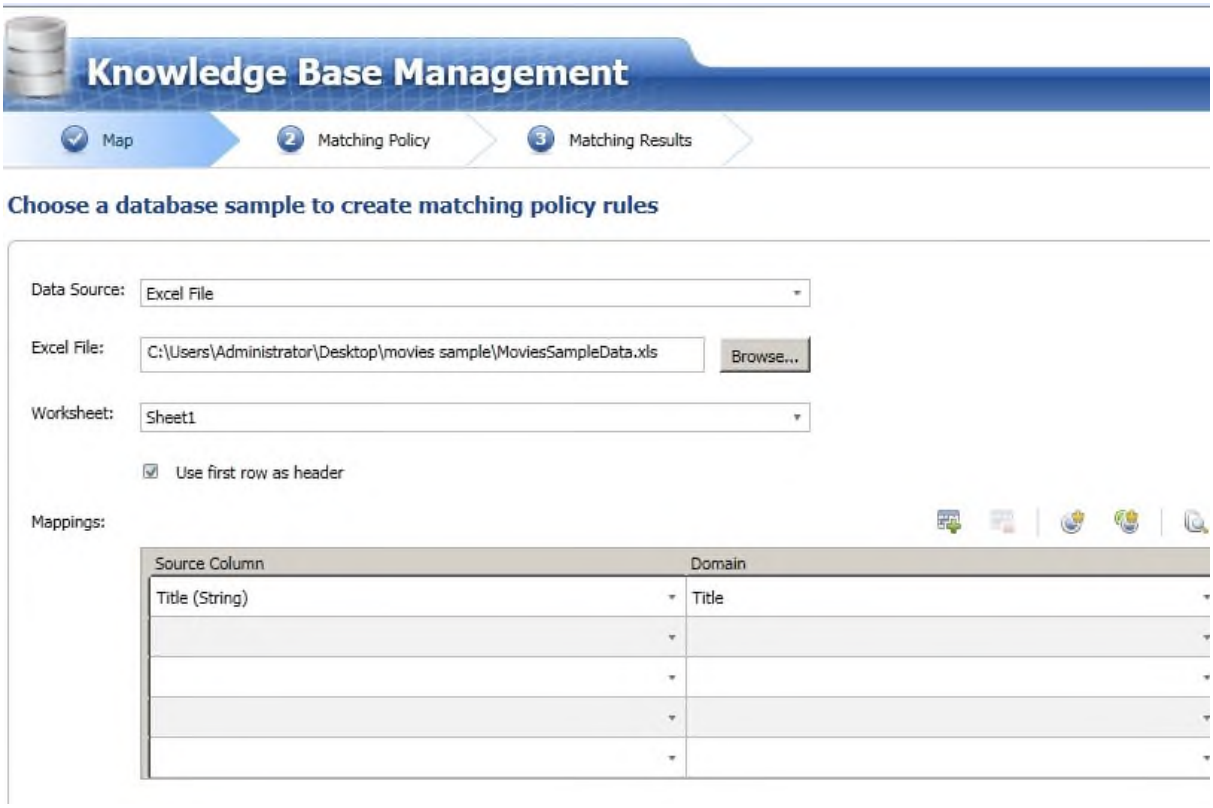
Displays each condition in the selected rule, including domain, similarity value, weight, and prerequisite selection.

Matching Results Tab

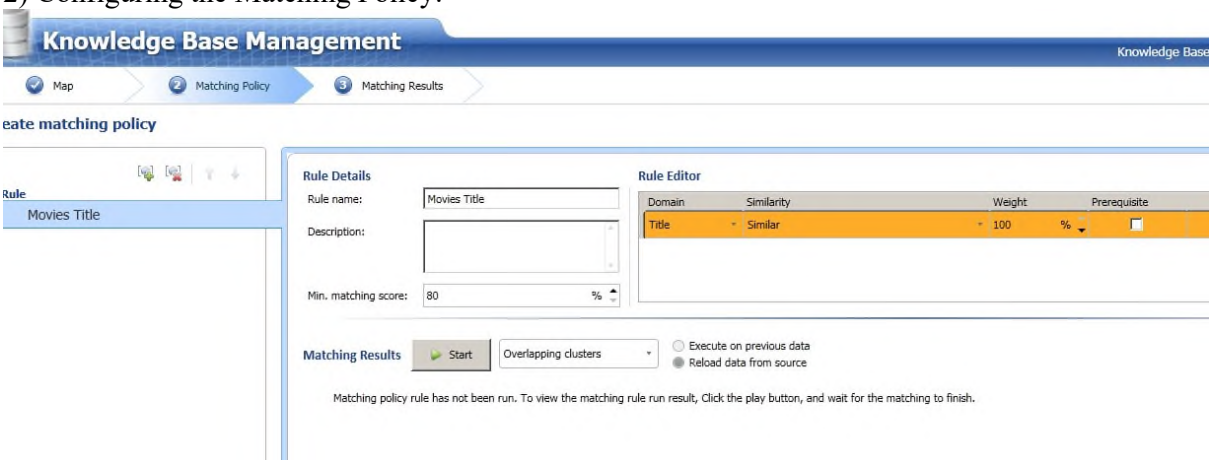
Click the **Matching Results** tab to display statistics for the analysis of the data source using the knowledge selected for the project and the matching rule or rules in that knowledge base. The statistics include the following:

- The total number of records in the database
- The total number of matching records in the database
- The number of records in the database that are not considered to be duplicates
- The number of clusters discovered
- The average cluster size (number of duplicate records divided by number of clusters)
- The fewest number of duplicates in a cluster
- The greatest number of duplicates in a cluster

Example: we've a list of movies and we want to identify "matching" movie titles.
Create a Matching Policy
1) Mapping the Domains:

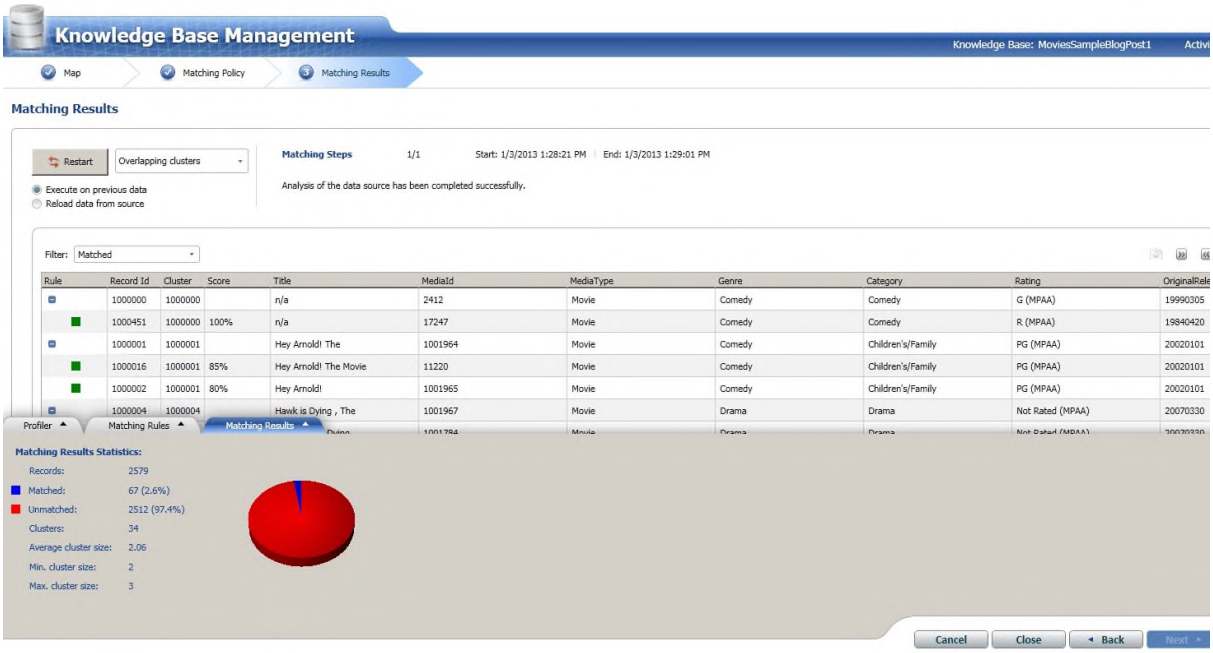


2) Configuring the Matching Policy:



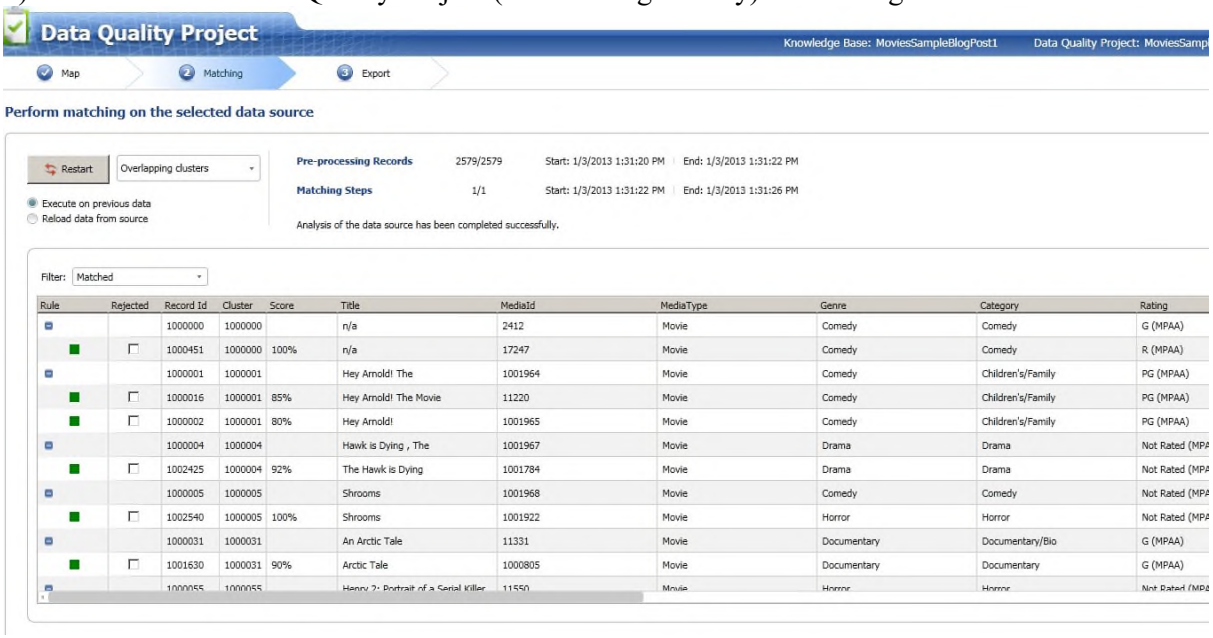
Note: You can have Matching Rules on more than one domain. I used one domain for demo purposes.

3) View Matching results:



4) Publish the KB (which stores the matching policy)

Once you have the matching policy, you can use this in a Data Quality Project:
5) See How I ran a Data Quality Project (w/ matching activity) in the image below.



Note: You can export the clean data-set via Data Quality Project.

Using Script in SSIS

- The Script task provides code to perform functions that are not available in the built-in tasks and transformations that SQL Server Integration Services provides. The Script task can also combine functions in one script instead of using multiple tasks and

transformations. You use the Script task for work that must be done once in a package (or once per enumerated object), instead than once per data row.

- You can use the Script task for the following purposes:
  - Access data by using other technologies that are not supported by built-in connection types. For example, a script can use Active Directory Service Interfaces (ADSI) to access and extract user names from Active Directory.
  - Create a package-specific performance counter. For example, a script can create a performance counter that is updated while a complex or poorly performing task runs.
  - Identify whether specified files are empty or how many rows they contain, and then based on that information affect the control flow in a package. For example, if a file contains zero rows, the value of a variable set to 0, and a precedence constraint that evaluates the value prevents a File System task from copying the file.

The Script Task Dts Object

- The Dts object is actually a property on your package that is an instance of the class. The **Dts** object provides a window into the package in which your script executes. Although you can't change properties of the package as it executes, the **Dts** object has seven properties and one method that allow you to interact with the package.
- The following is an explanation of these members:
  - **Connections**: A collection of Connection Managers defined in the package. You can use these connections in your script to retrieve any extra data you may need.
  - **Events**: A collection of events that are defined for the package. You can use this interface to fire off these predefined events and any custom events.
  - **ExecutionValue**: A read-write property that enables you to specify additional information about your task's execution using a user-defined object. This can be any information you want.
  - **TaskResult**: This property enables you to return the success or failure status of your Script Task to the package. This is the main way of communicating processing status or controlling flow in your package. This property must be set before exiting your script.
  - **Transaction**: Obtains the transaction associated with the container in which your script is running.
  - **VariableDispenser**: Gets the VariableDispenser object, which you can use to retrieve variables when using the Script Task.
  - **Variables**: A collection of all the variables available to any script.
  - **Log**: You can use this method to write to any log providers that have been enabled.

Script Component

- The Script Component provides another area where programming logic can be applied in an SSIS package. This component, which can be used only in the Data Flow portion of an SSIS package, allows programmatic tasks to occur in the data stream. This component exists to provide, consume, or transform data using .NET code.
- SSIS Script component is one data transformation tasks in SQL Server Integration Services (SSIS). SSIS is an integration tool in the Microsoft BI family to extract data from heterogeneous data sources and transform it to your need. Apart from the standard data

sources such as databases, text files, excel files, and web services, there can be instances where you need to retrieve non-traditional data sources.

- For example, let us say you want to extract the details of text files such as file sizes, created date, etc. In these types of scenarios, traditional data sources cannot be used.

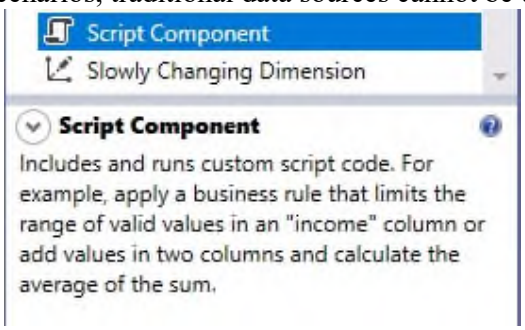


Figure – SSIS Script Component description from toolbox

- SSIS Script component can be used in three ways:
  1. As a source component
  2. As a destination component
  3. As a transformation component

  When adding a Script component to the data flow task, you will be asked how the script will be used, as shown in the image below:

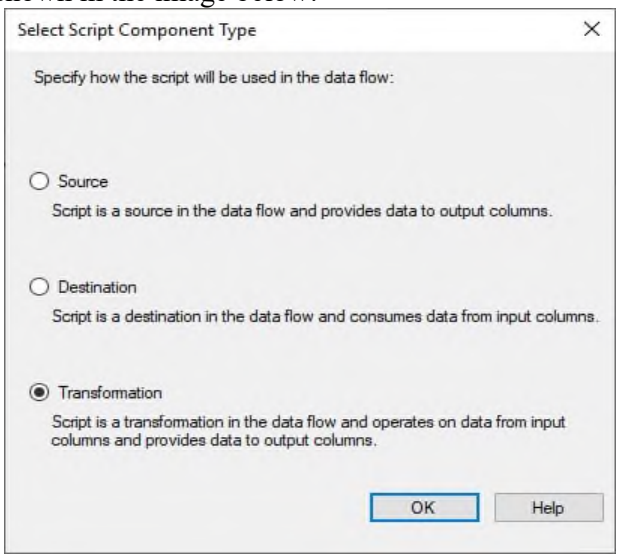


Figure – Selecting SSIS Script component type

SSIS Script component editor

- Before describing each approach, we will illustrate the options that can be configured in the SSIS script component editor. If you open the SSIS Script component editor, you will see four tabs:
  - **Script:** Here, you can open the script editor where you can write your script, select the script programming language (C# or VB.NET) and select the SSIS variables you need within the script (for read or write purposes)
  - **Input Columns:** Here, you can select the input columns for use within the script
  - **Inputs and Outputs:** Here, you can configure the outputs generated by the script (number of outputs, type of each output, output columns, etc.)
  - **Connection Managers:** Here, you can select the connection managers for use within the script

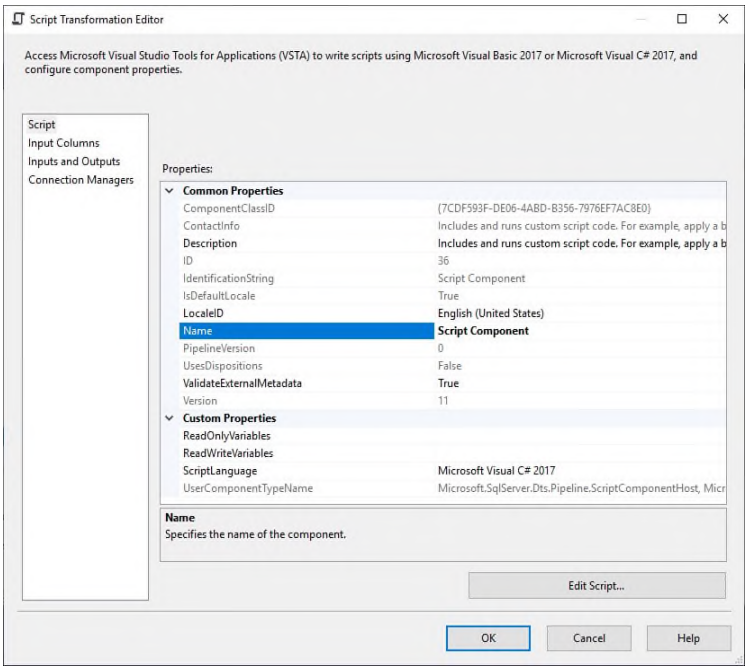


Figure – SSIS Script component editor

- Note, that when using the SSIS script component as a source, there are no input columns to select. In addition, when using this component as a destination, no output configuration is needed.

SSIS Script component as source

- The first way to use a script component is as a source. This approach is useful when reading from a data source that doesn't have its own source component (Flat File, Excel, OLE DB, etc.,) such as when reading JSON data from a Web API or reading an unstructured Flat File that can't be handled using a Flat File Connection Manager.
- The role of this Script Component is to provide data to your Data Flow Task. You can define outputs and their types and use script code to populate them. An example would be reading in a complex file format, possibly XML or something that requires custom coding to read, like HTTP or RSS Sources.

SSIS Script component as destination

- The second way to use an SSIS Script component is as a destination. This approach is useful when exporting the data into a file that is not supported by other destination components, or for applying some processing without the need for a destination component.
- This type of Script Component consumes data much like an Excel or Flat File Destination. This component is the end of the line for the data in your data stream. Here, you'll typically put the data into a DataSet variable to pass back to the Control Flow for further processing, or send the stream to custom output destinations not supported by built-in SSIS components. Examples of these output destinations can be web service calls, custom XML formats, and multi-record formats for mainframe systems. You can even programmatically connect and send a stream to a printer object.

SSIS Script component transformation

- The third and most popular way to use a script component is for transformations.

- This type of Script Component can perform custom transformations on data. It consumes input columns and produces output columns. You would use this component when one of the built-in transformations just isn't flexible enough.
- In this approach, the script component reads input values from the data flow and generates outputs. There are two types of outputs that can be generated:
  - Synchronous output
  - Asynchronous output

Synchronous output

- With this output type, there is an output row for each input row. All output columns defined in the output buffer are added to the input row. As shown in the image below, in the Inputs and Outputs tab page, you must select the synchronous input property for the output buffer created:

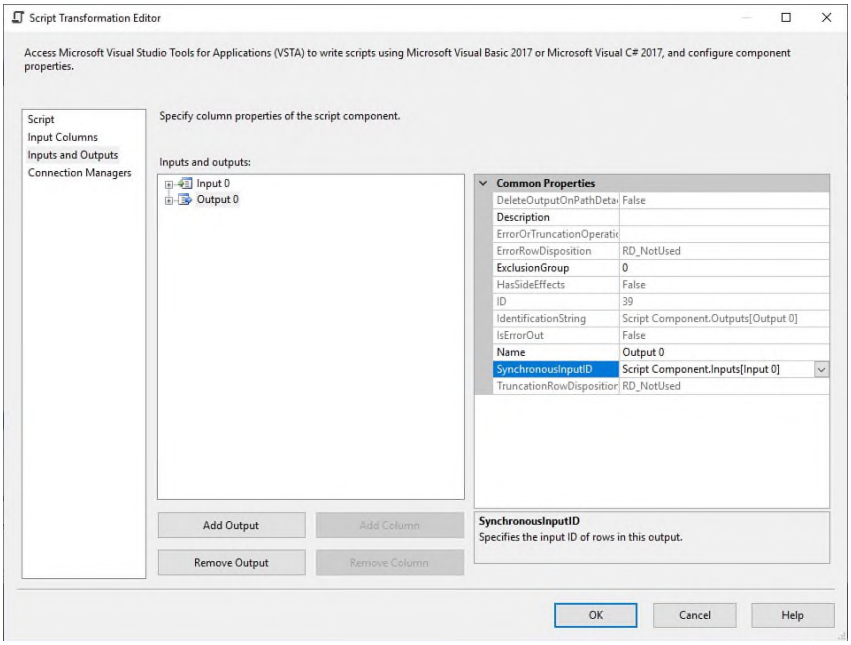


Figure – Define synchronous Input for the output buffer

- When performing a synchronous transformation, all input and output columns are accessed using the Row class. As an example:
- Row.outColumn = Row.InColumn + "_1";

Asynchronous transformation

- With this transformation type, each input row may have 0 or multiple output buffers. You can set the output type as asynchronous by setting the SynchronousInput property to None:

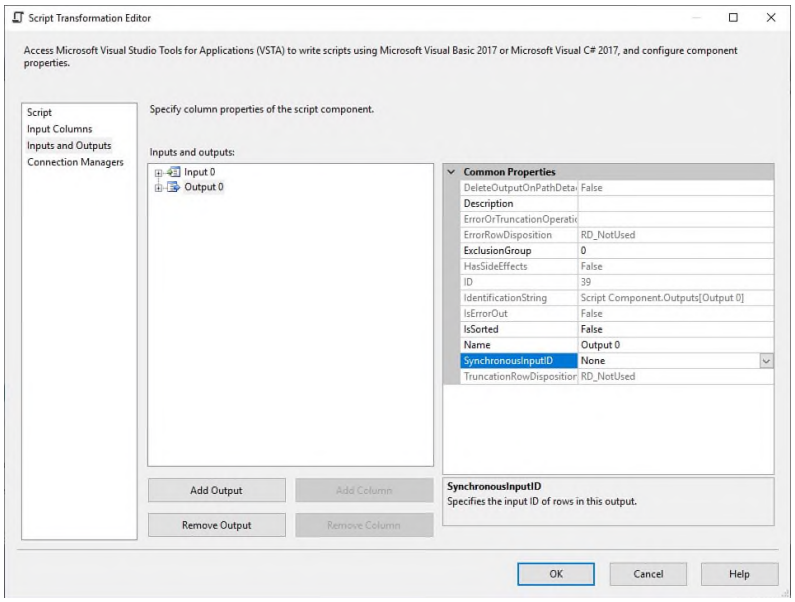


Figure – Set the output buffer as asynchronous

- You must use the output buffer wrapper class to add an output row and to assign values to columns. As an example, if the output buffer name is "Output1," then you must use the following code:
- Output1Buffer.AddRow();
  Output1Buffer.outColumn = "abc";

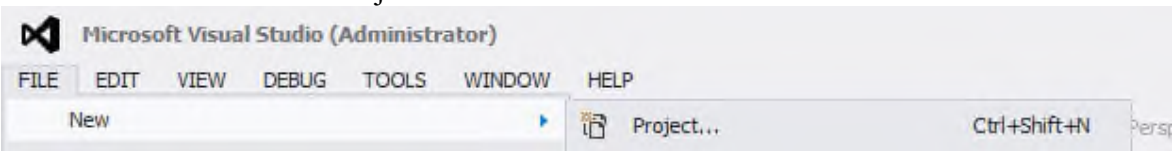
Single/multiple outputs

- From the Inputs and Outputs tab, you can add multiple output buffers, and you can configure each one as synchronous or asynchronous based on the logic you want to implement.

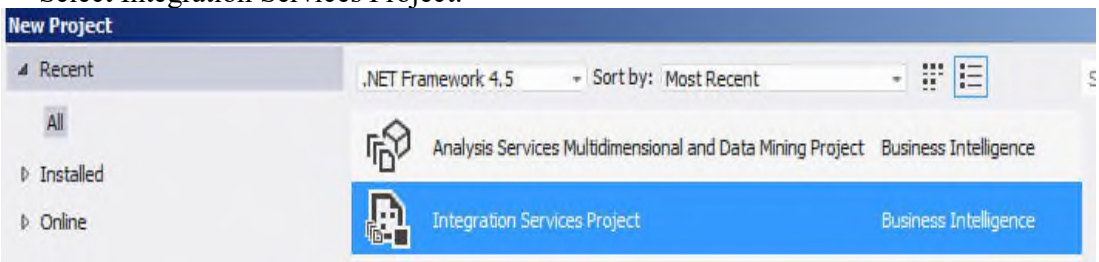
<u>Example 1 - Hello World</u>

- Let's start with the Hello World example using a simple Script Task.
- In order to start, open the SQL Server Data Tools for Visual Studio.

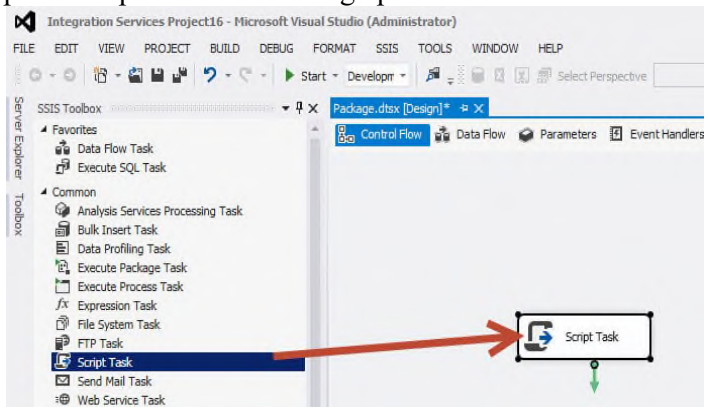
- Go to File > New > Project



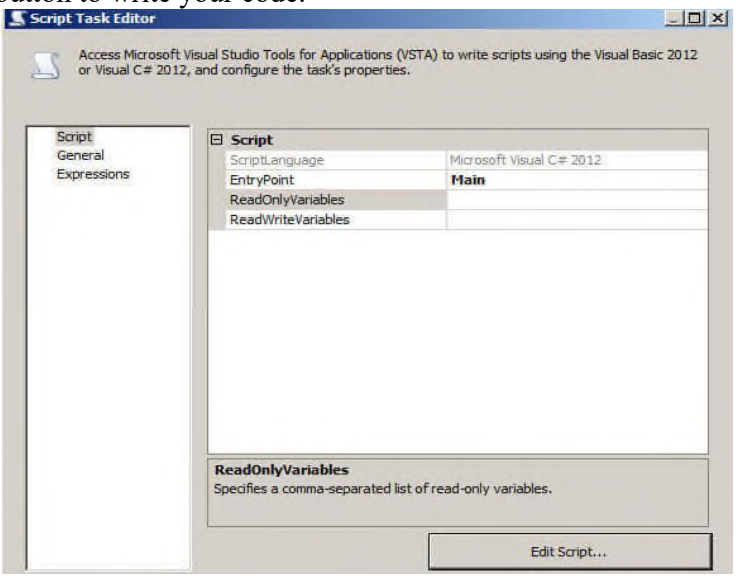
- Select Integration Services Project.



- Drag and drop the Script Task to the design pane and double click on it.



- The following window will open. The Script Language is use to select which language to use, either C# or Visual Basic.
- Entry Point is use to select where to start in the code, by default it starts in Main. Press the Edit Script button to write your code.

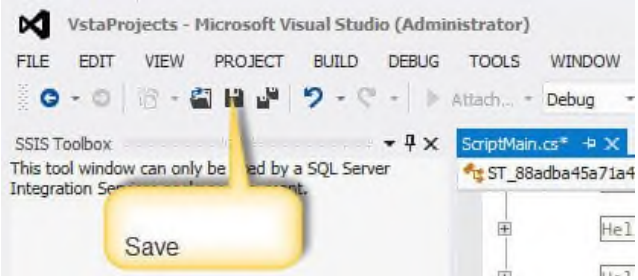


- A new Window will be display to allow you to write the code. Go to the main procedure, by default you will create your code there.

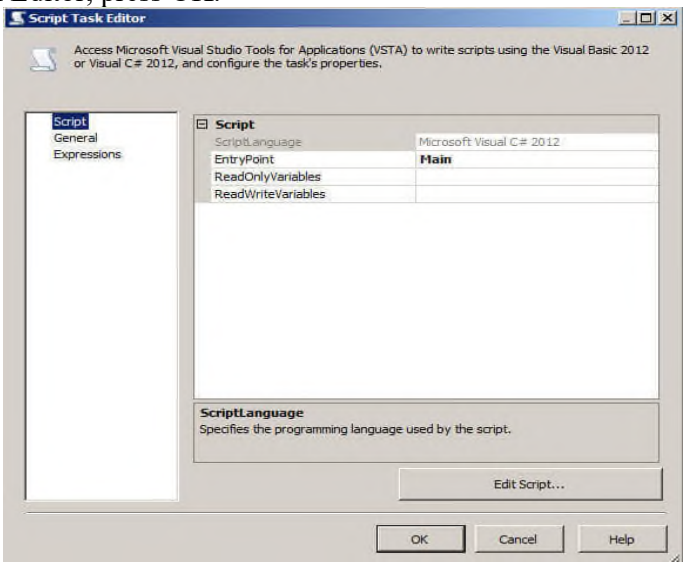
- Add the following code in the Main section. This code will display a message with the Hello World message.

```
public void Main()
{    // TODO: Add your code here
   MessageBox.Show("Hello World");
}
```

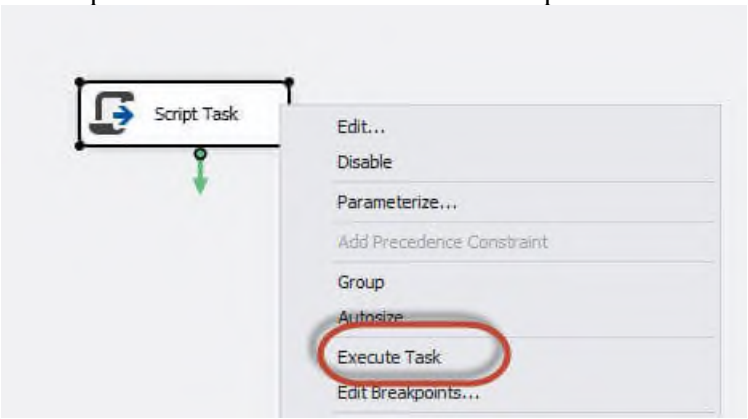
- Save the code.



- In the Script Task Editor, press OK.



- Right click on the Script Task and select the Execute Task option.



- If everything is done correctly, you will receive the following pop-up message:

- Once finished, you can stop the package as shown below.

Using Custom Component in SSIS

- SQLServer Integration Services (SSIS) is the Microsoft platform for building enterprise-level data integration and data transformation solutions. Integration Services can extract and transform data from a wide variety of sources such as XML data files, flat files, and relational data sources, and then load the data into one or more destinations. Integration Services is so versatile because it has the ability to connect to disparate data sources built into its object model. In this article, we will look at how to extend this object model by building a custom data source, using a working example.
- The data flow task consists of components that connect to a variety of data sources, then transform, and route that data at high speed. Microsoft SQL Server Integration Services provides an extensible object model that lets developers create custom sources, transformations, and destinations that you can use in SQL Server Data Tools (SSDT) and in deployed packages.

What is an SSIS component?

- SSIS components are one of the basic building blocks of the SSIS framework, and can be found in the SSIS toolbox in SQLServer Data Tools.These can read data from external data sources, transform it, and write it back to other data destinations. SQLServer Data Tools have some examples of data flow components, including the OLE DB Source, Lookup transformation and the Character Map transformation components.
- Under the hood, an SSIS data flow component is a .Net class that inherits from the PipelineComponent class, which is part of the overall SSIS Object Model. It overrides the various methods of PipelineComponent, such as those for creating inputs/outputs, adding custom properties to the component etc. The compiled class library is place in a subfolder in the SQLServer installation folder. SQLServer Data Tools automatically recognizes any class libraries in this folder that implement PipelineComponent, and makes the component available in the SSIS Toolbox.

Why Custom Component ?

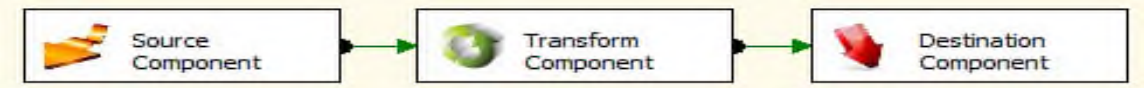
- Reusability
  - Component is a DLL
  - Single code base
  - Can be used multiple times in a single project
  - Can be shared across multiple projects
  - Easy to test Component version
- Performance
  - Faster than scripting
- Well documented
  - Though not a how-to guide

Types Of Component

- Data Connections
- Log Providers
- For Each Loops
- Control Flow Tasks
- Data Flow Pipeline Component

Pipeline Component Types

- Sources
- Transforms
- Destinations



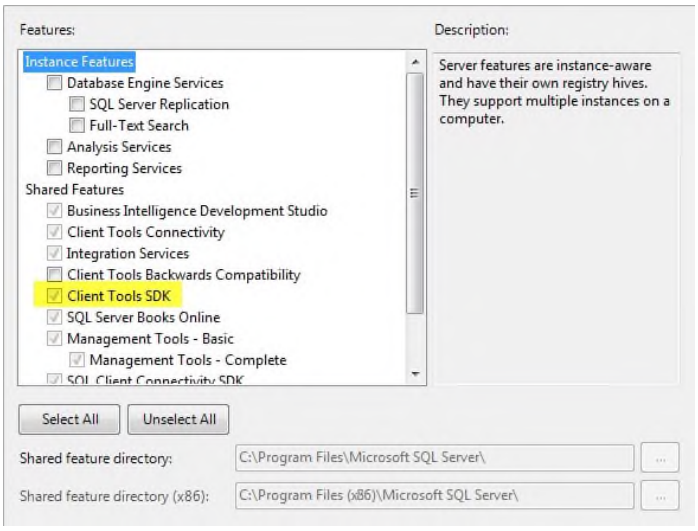
Design/Run Time

- Design Time
  - Work done in BIDS
  - attachments / detachments
  - Validation
  - Column usage
- Run Time
  - Metadata interrogation
  - DTEXEC
  - Flow of data

To create custom component

Requirements

- Visual Studio – BIDS is not enough
- Or Visual Basic / C# Express
- Client Tools SDK



- Target Framework 3.5 (Advanced compile options)
- Sign the assembley
- Add References(Program file(x86)/<SqlServer>/100/sdk/Assemblies)
  - Microsoft.SqlServer.DTSPipelineWrap
  - Microsoft.SqlServer.DTSRuntimeWrap
  - Microsoft.SqlServer.ManagedDTS
  - Microsoft.SqlServer.PipeLine Host

Class Creation

- Inherits PipelineComponent
- Uses attribute DtsPipelineComponent

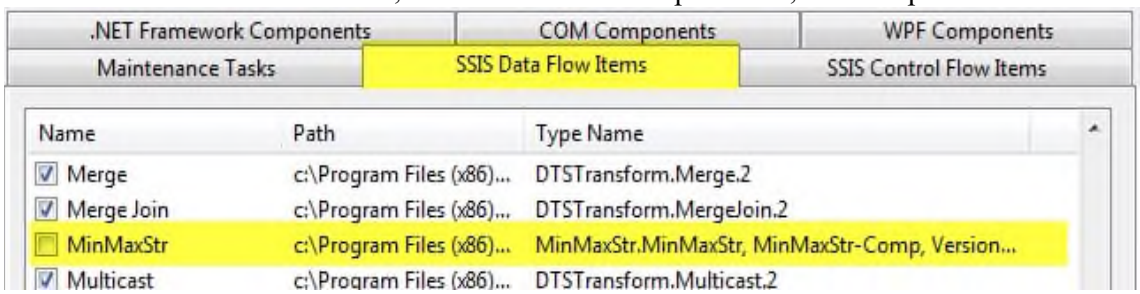
```
Imports System
Imports System.Data
Imports Microsoft.SqlServer.Dts.Pipeline
Imports Microsoft.SqlServer.Dts.Pipeline.Wrapper
Imports Microsoft.SqlServer.Dts.Runtime.Wrapper

<DtsPipelineComponent(ComponentType:=ComponentType.Transform, _
                      DisplayName:="Basic Component")> _
Public Class BasisComponent
    Inherits PipelineComponent

End Class
```

Post Build

- Copy DLL to "C:\Program Files (x86)\Microsoft SQL Server\100\DTS\PipelineComponents"
- Register to Global Assembley Cache using GACUTIL
- Must Restart BIDS
- For first use "Choose Items","SSIS Data Flow Components" ,tick Component



Design Time Methods

- Methods
  - ProvideComponentProperties
    - Define initial metadata of component
  - Validate
    - Tests the metadata is correct
  - ReinitializeMetaData
    - Fix the metadata

5 Deploying and configuring SSIS Packages, Consuming Data in Data Warehouse

Topic Covered

- Overview of SSIS Development
  - Packages
  - Control Flow vs. Data Flow
  - Package Functionality Extension Objects
- Deploying SSIS Projects
  - Deploying Packages to SQL Server Integration Services Catalog
  - Deployment Models In SSIS
- Planning SSIS Packages Execution
- Introduction to Business Intelligence
- Introduction to Reporting
  - Basic Reports in SQL Reporting Services
  - Parameters in Reporting Services
  - Charts in SSRS
- SQL Server Reporting Services Configuration Manager (For Deploy Project in Report Server)
- Introduction to Data Analysis
- Create Cube and Dimensions Using Data Analysis

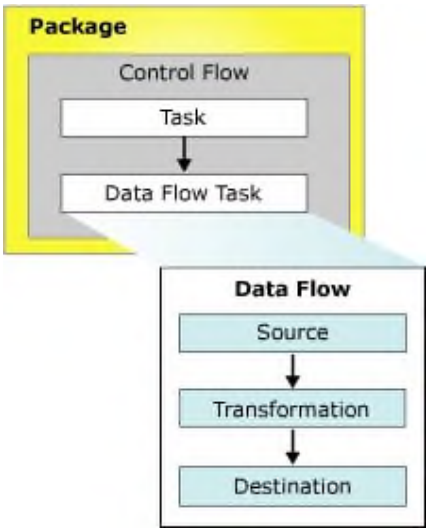
Overview of SSIS Development

- 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 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 Packages

- Another core component of SSIS is the notion of a package. It is a collection of tasks, which execute in an orderly fashion. Here, precedence constraints help manage the order in which the task will execute.
- A package can help you to saves files onto a SQL Server, in the msdb or package catalog database. It can save as a .dtsx file, which is a structured file very similar to .rdl files are to Reporting Services.

- A **package** is an organized collection of connections, control flow elements, data flow elements, event handlers, variables, parameters, and configurations, that you assemble using either the graphical design tools that SQL Server Integration Services provides, or build programmatically. You then save the completed package to SQL Server, the SSIS Package Store, or the file system, or you can deploy the ssISnoversion project to the SSIS server. The package is the unit of work that is retrieved, executed, and saved.
- When you first create a package, an empty object does nothing. To add functionality to a package, you add a control flow and, optionally, one or more data flows to the package.
- The following diagram shows a simple package that contains a control flow with a Data Flow task, which in turn contains a data flow.



Here are some of the key terms and phrases

Term	Description
Component	A component is part of a data pipeline, representing either a source from which data is retrieve, a destination to which data is written, or a transformation that manipulates or reshapes the data.
Task	A task is a single operation within a package. There are dozens of different types of tasks available in SSIS.
Project	Source code in SSIS is arrange into functional units called projects. A project can contain one package or many packages. In most cases, when deploying SSIS code, the entire project is deploy to the server.
Control flow	A control flow consists of the tasks and containers. The tasks perform specific types of work such as executing SQL statements or sending email messages, and the containers define repeating subsets of the control flow or

	group subsets of the control flow to make the package easier to manage. The tasks and containers are usually connected by precedence constraints that specify the sequence in which tasks and containers are execute and the conditions that must be satisfied to run the next task or container in the control flow.
Data flow	A data flow consists of sources that extract data, transformations that modify data, and the destinations that load the data into data stores.
Connections	A connection includes the information necessary to connect to a particular data source. Tasks can reference the connection by its name, allowing the details of the connection to be changed or configured at run time.
Connection manager	To connect to the data stores, a package uses connection managers. The connection managers are defined when you create the package. From the definition, the Integration Services runtime creates a connection at run time.
Variables	SSIS provides support for strongly typed part–Common Type System (CTS)–compliant variables as a part of any container or package.
Container	The level beneath the package can be either a container or some kind of executable (a Control Flow executable). A container is a logical construct that can hold other containers or components grouped together under the "super-container."
Precedence constraints	Control Flow design surface may have multiple Control Flow items, in this situation we need to decide which item will execute first. Using Precedence Constraints, we can define processing flow for Control Flow items. We can define different flow based on success or failure of any item.
Property Expression	You can set property values of packages and package objects in two different ways: directly by setting the value of each property, or indirectly by using property expressions. An expression, mapped to a property, is called a property expression. You build property expressions by using the operators and functions that the Integration Services expression language provides and variables. When the package is validated, which occurs when you save the package, the evaluation results of the property expressions replace the original values of properties.
Data Source	A data source is a connection reference that you create and save outside a package, and then use as a source when adding new connection managers to a package. A data source represents a simple connection to a data store and therefore makes all tables and views in the data store available to the package.
Data source view	A data source view build on a data source. It can contain only selected database objects. It can be extend with calculated columns that populated by custom expressions, new relationships between tables, and queries. You can also apply a filter to a data source view to specify a subset of the data selected. In Integration Services, data sources and data source views are saved within the package definitions of the packages in which they are used.

Execution	This is the act of invoking, or running, the logic in an SSIS package. Packages can be execute from within the SQL Server Data Tools (SSDT) development environment or directly on a properly configured instance of SQL Server.
Deployment	A deployment occurs when the fully developed SSIS project is push from the development workstation to an instance of SQL Server, where it can then be executed either manually or through a scheduling tool such as SQL Server Agent. Deployment is usually more complex than copying code from one machine to another, although SQL Server does a good job of hiding that complexity for most deployments.
Solution	A solution is a logical grouping of related projects.
The SSIS runtime engine	This logic allows a package to run. When you are working with SSIS packages in SSDT, the packages will be execute using the SSIS runtime on your development machine. After the code is deploy to SQL Server, any execution run on that server will use the server's SSIS runtime.

- **CONTROL FLOW VS DATA FLOW**

  Control Flow:
  - Process is the key: precedence constraints control the project flow based on task Completion, success or failure
  - Task 1 needs to complete before task 2 begins
  - Smallest unit of the control flow is a task
  - Control flow does not move data from task to task
  - Tasks are run in series if connected with precedence or in parallel
  - Package control flow is made up of containers and tasks connected with
  - Precedence constraints to control package flow

  Data Flow:
  - Unlike control flow, multiple components can process data at the same time
  - Smallest unit of the data flow is a component
  - Data flows move data, but are also tasks in the control flow, as such, their success or failure effects how your control flow operates
  - Data is moved and manipulated through transformations
  - Data is passed between each component in the data flow
  - Data flow is made up of source(s), transformations, and destinations.

- **Package Functionality Extension Objects**

  Packages can include additional objects that provide advanced features or extend existing functionality, such as event handlers, configurations, logging, and variables.

  Event Handlers

  An event handler is a workflow that runs in response to the events raised by a package, task, or container. For example, you could use an event handler to check disk space when a pre-execution event occurs or if an error occurs, and send an e-mail message that reports the available space or error information to an administrator. An event handler is constructed like a package, with a control flow and optional data flows. Event handlers can be added to individual tasks or containers in the package.

Configurations
A configuration is a set of property-value pairs that defines the properties of the package and its tasks, containers, variables, connections, and event handlers when the package runs. Using configurations makes it possible to update properties without modifying the package. When the package is run, the configuration information is loaded, updating the values of properties. For example, a configuration can update the connection string of connection.The configuration is saved and then deployed with the package when the package is installed on a different computer. The values in the configuration can be updated when the package is installed to support the package in a different environment.

Logging and Log Providers
A log is a collection of information about the package that is collected when the package runs. For example, a log can provide the start and finish time for a package run. A log provider defines the destination type and the format that the package and its containers and tasks can use to log run-time information. The logs are associated with a package, but the tasks and containers in the package can log information to any package log. Integration Services includes a variety of built-in log providers for logging. For example, Integration Services includes log providers for SQL Server and text files. You can also create custom log providers and use them for logging.

Variables
Integration Services supports system variables and user-defined variables. The system variables provide useful information about package objects at run time, and user-defined variables support custom scenarios in packages. Both types of variables can be used in expressions, scripts, and configurations.
The package-level variables include the pre-defined system variables available to a package and the user-defined variables with package scope.

Parameters
Integration Services parameters allow you to assign values to properties within packages at the time of package execution. You can create *project parameters* at the project level and *package parameters* at the package level. Project parameters are used to supply any external input the project receives to one or more packages in the project. Package parameters allow you to modify package execution without having to edit and redeploy the package.

Deploying SSIS Projects

- Integration Services supports two deployment models, the project deployment model and the legacy package deployment model. The project deployment model enables you to deploy your projects to the Integration Services server.
- The project deployment model was introduce in SQL Server 2012 Integration Services (SSIS). With this deployment model, you were not able to deploy one or more packages without deploying the whole project. SQL Server 2016 Integration Services (SSIS) introduced the package deployment model, which lets you deploy one or more packages without deploying the whole project.
- At the center of the project, deployment model is the project deployment file (.ispac extension). The project deployment file is a self-contained unit of

deployment that includes only the essential information about the packages and parameters in the project. The project deployment file does not capture all of the information contained in the Integration Services project file (.dtproj extension). For example, additional text files that you use for writing notes are not stored in the project deployment file and thus are not deploy to the catalog.

To convert a project to the project deployment model

1. Open the project in Visual Studio, and then in Solution Explorer, right-click the project and click **Convert to Project Deployment Model**.

   OR

   From Object Explorer in Management Studio, right-click the **Projects** node and select **Import Packages**.
2. Complete the wizard.

Start the Integration Services Deployment Wizard

1. In Object Explorer, with the **Integration Services Catalogs** node and the **SSISDB** node expanded, expand a folder.
2. Select the **Projects** node.
3. Right-click on the **Projects** node and select **Deploy project**. The Integration Services Deployment Wizard opens. You can deploy a project from the current catalog or from the file system.

Deploying Packages to SQL Server Integration Services Catalog

- Starting with SQL Server 2012, Integration Services (SSIS) packages can now be deploy to a single source for managing execution in multiple environments. The SSIS Catalog is a single database container for all deployed packages. The configuration files are replace with Environments. Deployed versions are track historically and a package can be reverted to a previous deployment.
- To use this feature, the SSIS Catalog must be create. The System Administrator of the SQL Server instance needs to create the Catalog or a user with elevated permissions for creating databases. Figure shows the menu after right-clicking on the Integration Services Catalog folder in SQL Server Management Studio (SSMS).

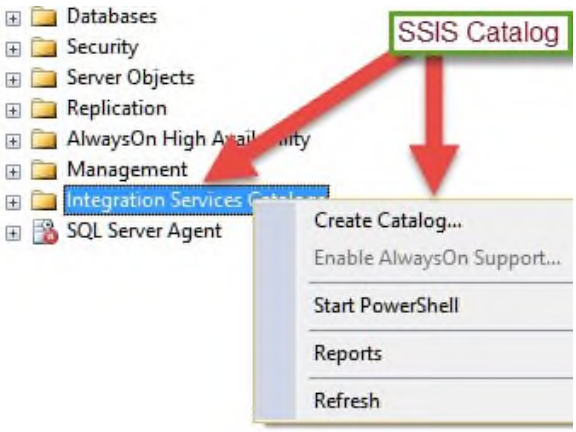


Figure – Integration Services Catalog

- The Create Catalog… meu option will launch the Create Catalog window. Some configuration takes place during its creation. CLR Integration will need to be enabled on this instance. On SQL Server Startup, the automatic execution of SSIS

stored procedure(s) can be enable or not. The most important part is to provide a strong password for the management of this new database container.

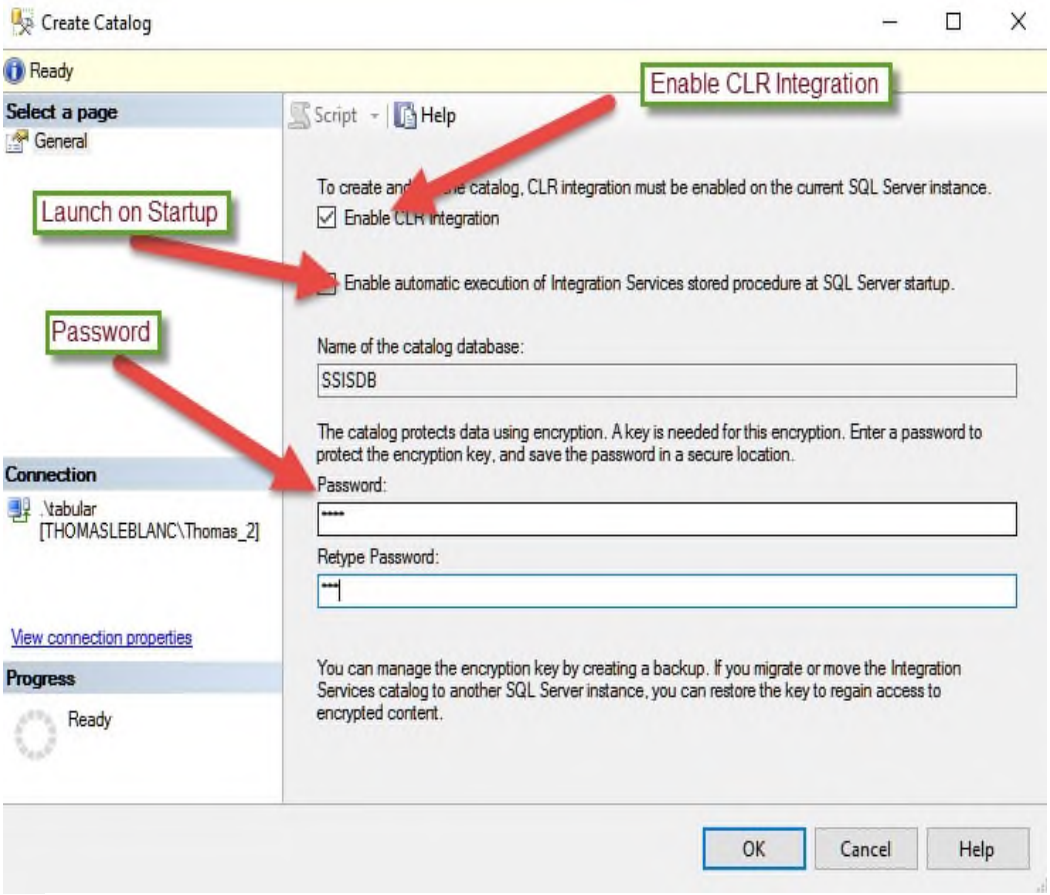


Figure: Create Catalog window

- When the configuration is complete, a new database is created on this instance: SSISDB. The database needs the same maintenance as any other production database on this system. Maintenance items include backups, index rebuild/reorganize and update statistics. The recovery mode of the database inherits the properties from the Model database just like all new databases. Simple recovery mode is adequate for SSISDB because of the infrequent deployments.

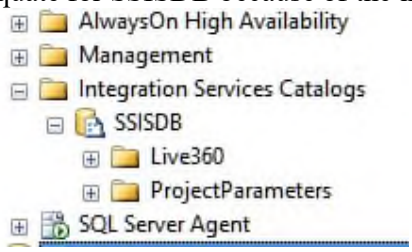


Figure: SSIS Catalog Projects Deployed

- A user does not need to be a SysAdmin to deploy projects (and packages) to the Catalog. There is a Database Role named ssis_admin in the SSISDB database. This role contains the permissions to deploy projects from Visual Studio. A developer's Active Directory (AD) account or an AD group can be added to this role.

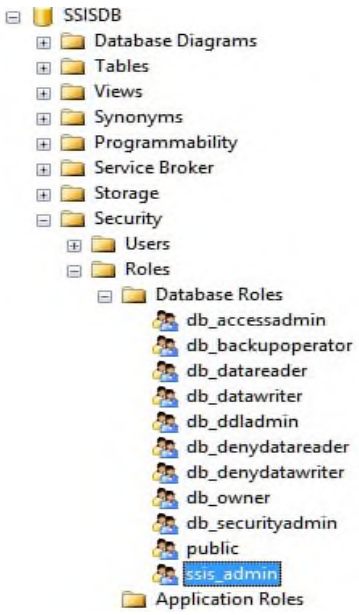


Figure: ssis_admin Database Role in SSISDB

- Deploying SSIS projects, along with the project's package(s), was added in SQL Server 2012 as well as the SSIS Catalog. This organization of packages in a SSIS Project enables objects, properties and values to be shared among the packages in a project. Figure shows the project. Param file associated with a SSIS Project.

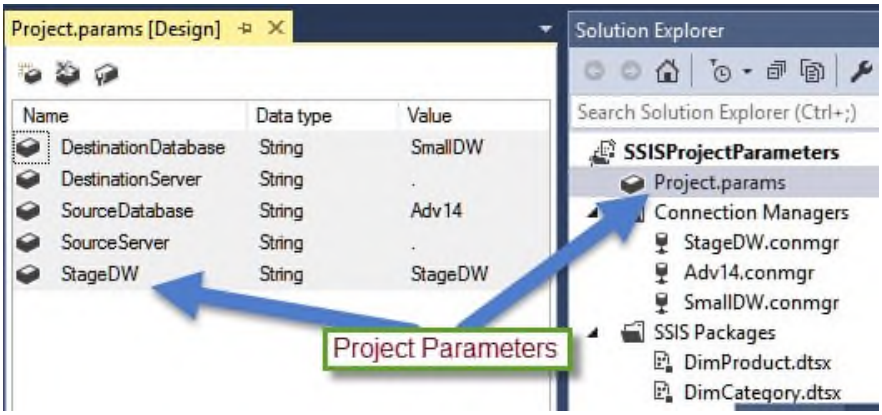


Figure: Project Parameters

- This project has multiple packages. The Source Database and Source Server is shared with packages like DimProduct and DimCategory. The connection is created under the Connection Manager in the Project's solution as seen in Figure. Not only can the packages can share parameter values, it also shares these database connections. This example contains staging, source and destination databases. The prefix (project) is added to the Package Connection Managers' connections as seen in Figure under the objects of the package.

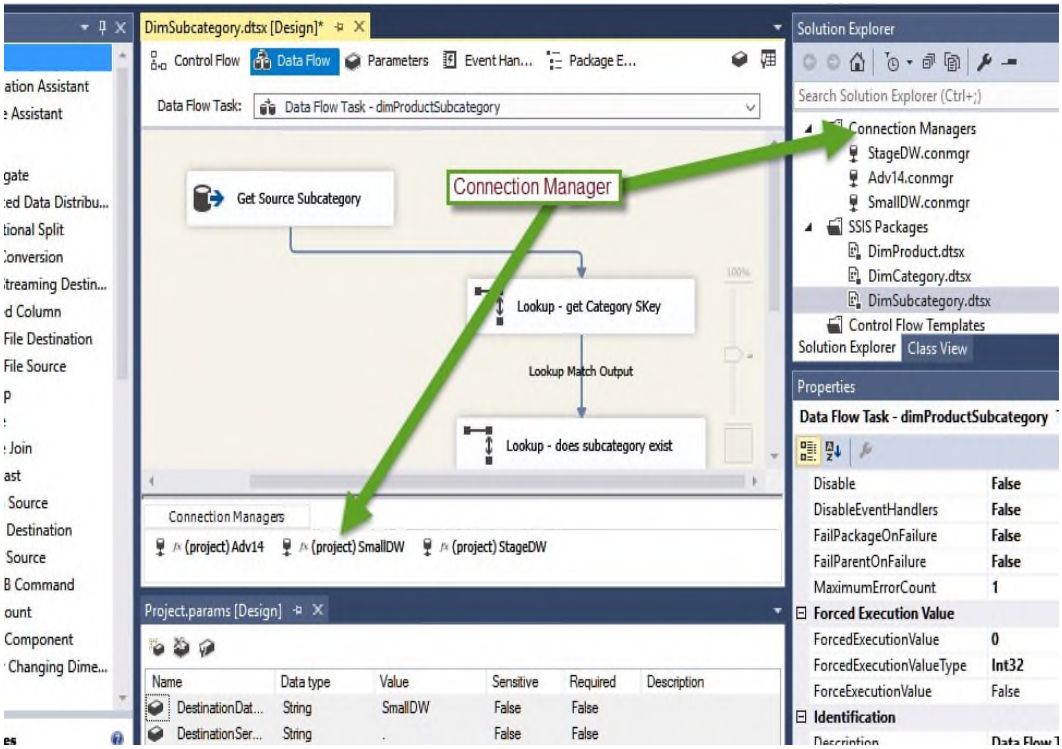


Figure: Project Connection Managers

- In this environment, the development team uses a separate server as the testing team and production system. The use of Project Parameter enables a single change to effect multiple packages in the project. To deploy the project to the SSIS Catalog, right-click the project in the solution explorer of Visual Studio.

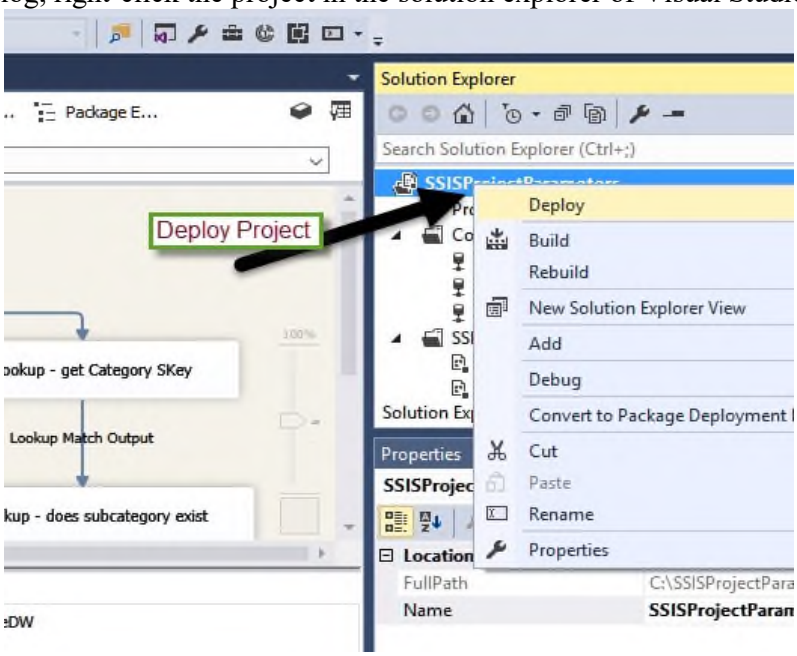


Figure: Deploy Project and Packages

- The Integration Services Deployment Wizard will show a welcome screen. You can disable this screen for future deployments. The Select Source will default to

the project selected while in Visual Studio. The wizard first brings you to the Select Destination page, but you can use the back button to go to the Select Source page. The Source Selection can be changed, but usually the deployment wizard is launched from the project being deployed. Figure shows the Select Destination screen where the Server Name and Path in the SSIS Catalog are selected.

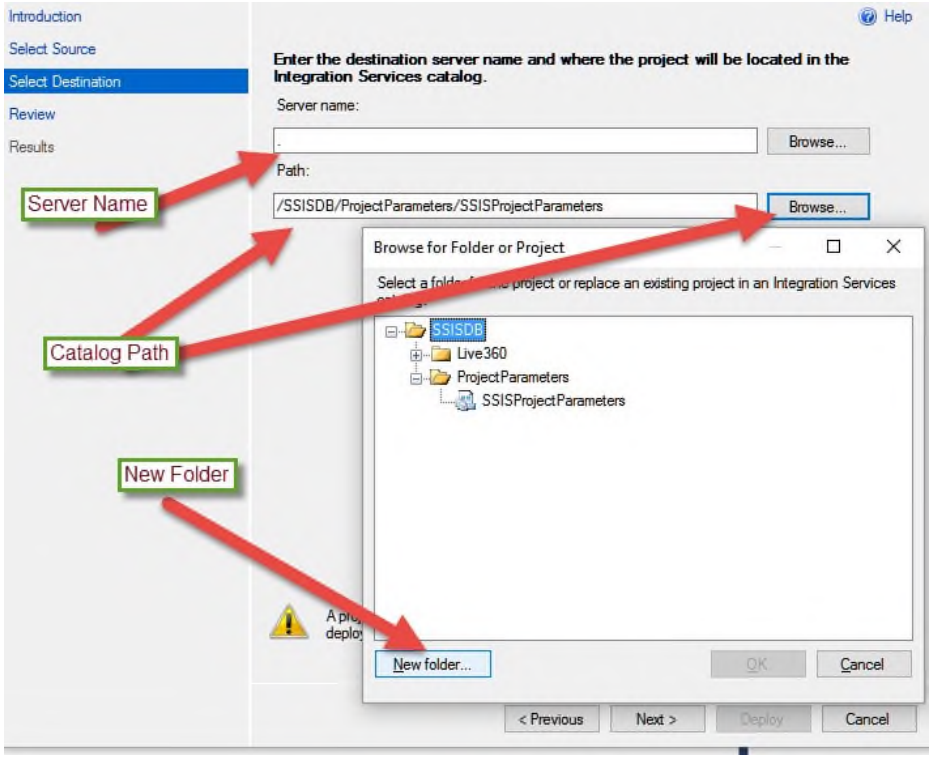
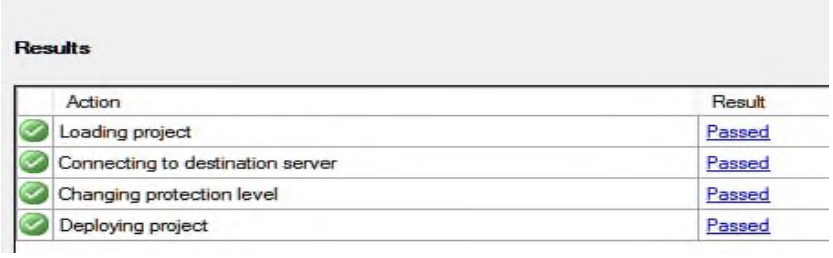


Figure: Select Destination Screen of SSIS Deployment Wizard

- The Server name will be the instance the Catalog was created. The Path can be an existing one or a New Path can be created. The same path can be used for different projects. All project packages will be together under the project in the Path.
- The Review screen gives the options to view the selected source and destination. Once the Deploy button is selected on the Review screen, the Results screen will show the success list or an indication of what failed during deployment.



Results

Action	Result
Loading project	Passed
Connecting to destination server	Passed
Changing protection level	Passed
Deploying project	Passed

Figure: Results screen of SSIS Deployment Wizard

- A package can be launched in several ways. In SQL Server Management Server (SSMS), the Integration Services Catalog folder can be expanded to see the Paths created in the Catalog. Figure shows the menu from right-clicking the package DimCategory in the SSIS Catalog.

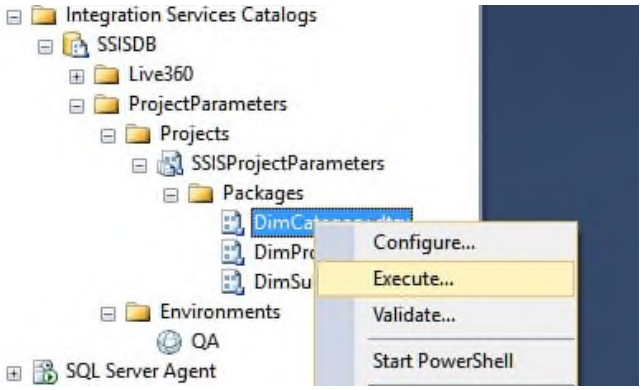


Figure: Executing a Package from SSIS Catalog

- After selecting Execute… from the menu, the execution prompts for Parameters, Connection Managers and Advanced options. Figure show the Parameters that can be changed before the DimCategory package is executed. This is where we can change the server or database names for the Project Parameters before the package is executed.

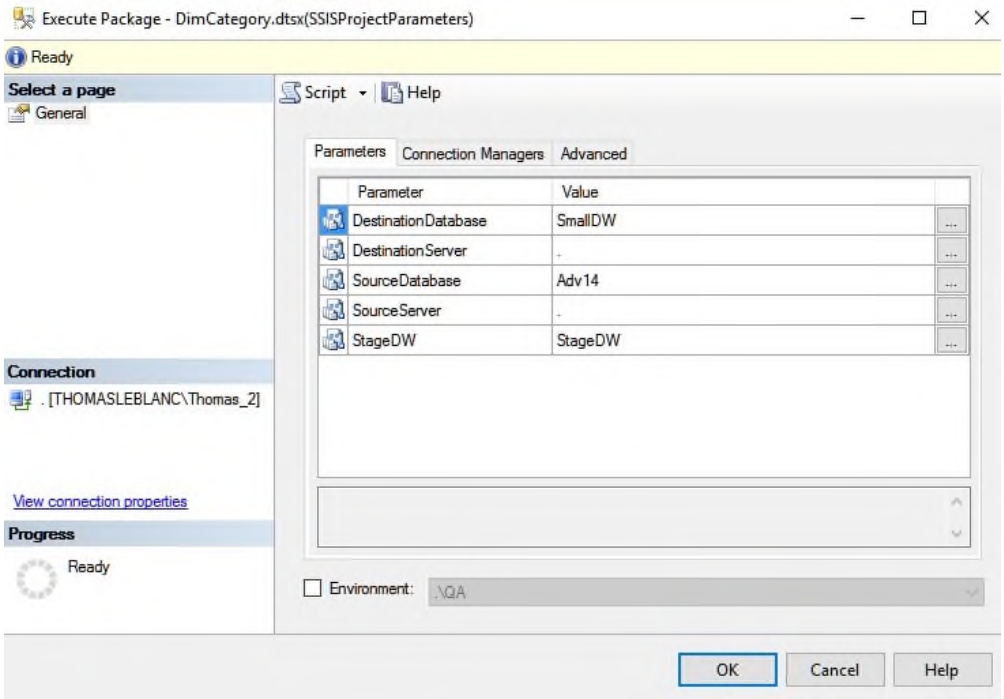


Figure: Executing a Package from SSIS Catalog

- One of the cool features of the SSIS Catalog are free reports. Figure shows the execution report that can be displayed after launching the package. There is a prompt to view or not view the report. There are two additional links to drill down into Messages and Performance. Individual Tasks can be drilled into to show messages from different steps of the package.

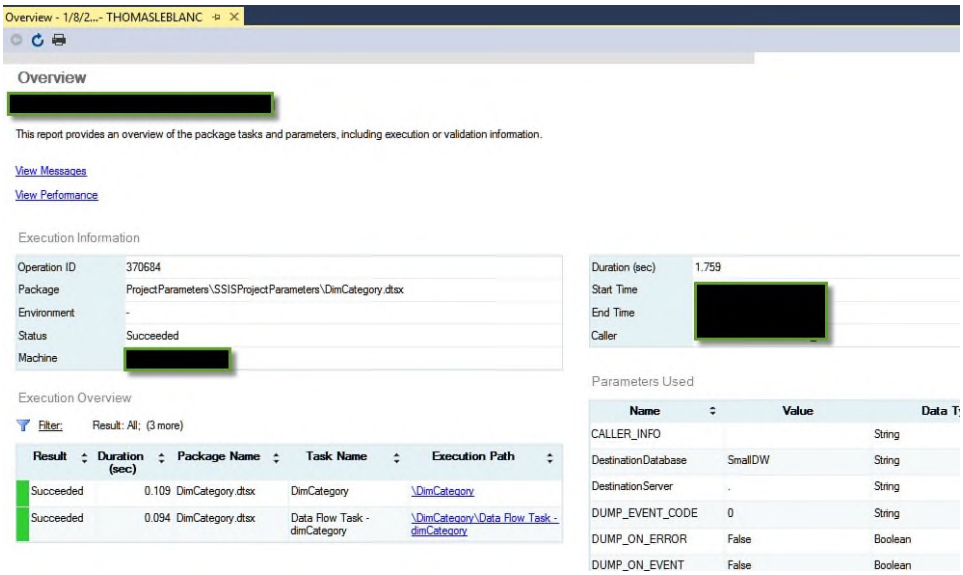


Figure: Report Overview of Package Execution

- If there is an error, the messages are the best place to start to debug a package. To see the performance of this package over time, click the View Performance drill down report.
- There is a main Dashboard that can be used to see an overview of all packages running from the Catalog. This dashboard is launched from SSMS by right-clicking on the SSISDB folder under the Integration Services Catalog. Figure shows the path to the report.

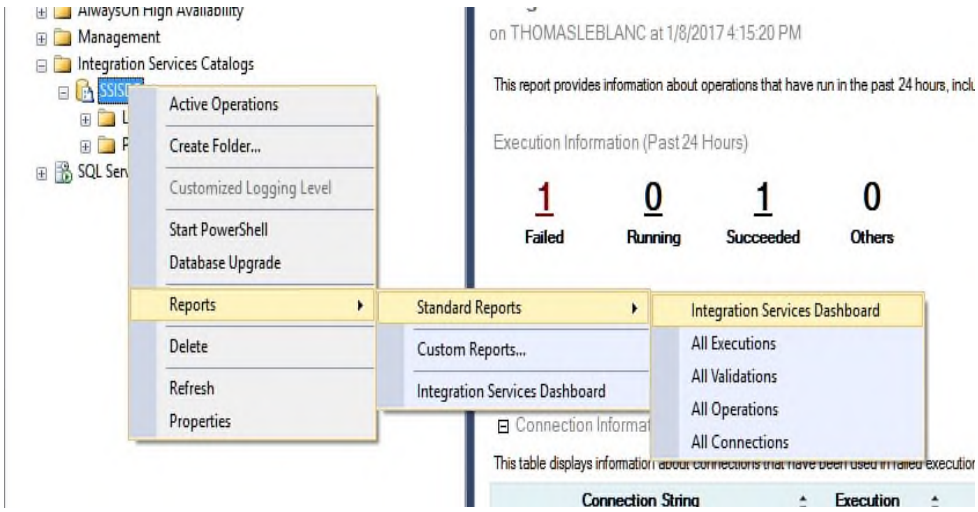


Figure: Launching the Integration Services Dashboard

- Figure shows for today one package has executed successfully and another has failed.

Integration Services Dashboard

on THOMASLEBLANC at 1/8/2017 4:15:20 PM

This report provides information about operations that have run in the past 24 hours, including executions that are currently running.

Execution Information (Past 24 Hours)

1	0	1	0
Failed	Running	Succeeded	Others

Package Information (Past 24 Hours)

2 out of 9 packages have executed.

Other Integration Services Reports
All Executions View all package executions.
All Validations View all package validations.
All Operations View all operations.
All Connections View information for connections used in failed executions.

Connection Information (Past 24 Hours)

This table displays information about connections that have been used in failed executions.

Connection String	Execution Occurrences	Last Failed Time	Last Failed Package	
Data Source=.; Initial Catalog=Adv14; Provider=SQLNCLI11.1; Integrated Security=SSPI; Auto Translate=False;	1	1/8/2017 4:15:06 PM	DimProduct.dtsx	Import Product Dimension:Error: SSIS processing input "Lookup Input" (119 task to stop running. There may be e
Data Source=.; Initial Catalog=SmallDW; Provider=SQLNCLI11.1; Integrated Security=SSPI;	1	1/8/2017 4:15:06 PM	DimProduct.dtsx	Import Product Dimension:Error: SSIS processing input "Lookup Input" (119 task to stop running. There may be e

Figure: Integration Services Dashboard

- By clicking on the Failed number, the next report gives the ability to drill into the Messages returned from the success and fail steps of the package like in next Figure

View Messages

View Performance

Execution Information

Operation ID	370686
Package	ProjectParameters\SSISProjectParameters\ParentPackage.dtsx
Environment	-
Status	Failed
Machine	THOMASLEBLANC

Execution Overview

Filter: Result: All; (3 more)

Result	Duration (sec)	Package Name	Task Name	Execution Path
Succeeded	0.093	DimCategory.dtsx	DimCategory	\ParentPackage\Category\DimCategory
Succeeded	0.078	DimCategory.dtsx	Data Flow Task - dimCategory	\ParentPackage\Category\DimCategory\Data Flow Task - dimCategory
Failed	0.391	DimProduct.dtsx	DimProduct	\ParentPackage\Product\DimProduct
Failed	0.235	DimProduct.dtsx	Import Product Dimension	\ParentPackage\Product\DimProduct\Import Product Dimension
Succeeded	0.156	DimProduct.dtsx	Truncate Staging tables	\ParentPackage\Product\DimProduct\Truncate Staging tables

Figure: Failed Package Messages report

Deployment Models in SSIS

Integration Services supports 2 deployment models i.e.

1. Package Deployment Model
2. Project Deployment Model

Prior SSIS 2012, in all versions like SSIS 2005, 2008 or 2008 R2 we had '**Package Deployment Model**'. With the introduction of SQL Server 2012 or 2014, a new Deployment model introduced named '**Project Deployment Model**'.

Package Deployment Model

SQL Server Integration Service (SSIS) 2005, 2008 and 2008 R2 versions were using this deployment model.

- In this model, package is the unit of deployment. At a time we can deploy single package not multiple like in SSIS 2012.
- Packages and configuration saved in the file system. Package with extension '**.dtsx**' and configuration file with extension '**.dtsConfig**'.
- Packages can be deployed either in File system or under **msdb** database.
- Packages are validated just before execution. You can also validate a package with **dtExec** or managed code.
- Packages are run in a separate window process.
- During execution, events that are produced by a package are not captured automatically. A log provider must be added to the package to capture events.
- Under this model, package configuration is required for each and every package under the project.
- When we deploy same package again, it'll overwrite the old one and due to this reason there was no way to check the previous history that how much time we've deployed our packages.

The following is the example showing how we can configure our SSIS 2008 created package for deployment on another computer.

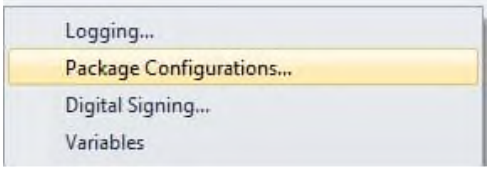
In this deployment model, to deploy any packages, we need to go through the following four steps:

- **Step 1:** Create Package Configuration File.
- **Step 2:** Create a deployment utility.
- **Step 3:** Copy Deployment folder on destination.
- **Step 4:** Package Installation.

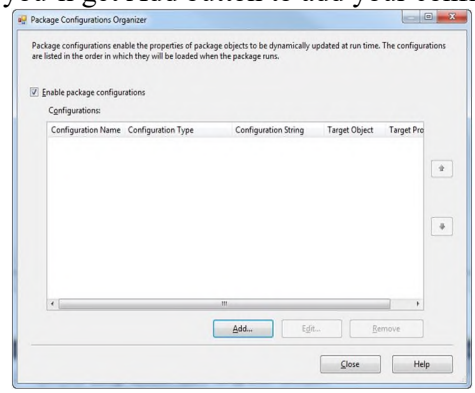
Let's take a look at each step.

Step 1: To create Package Configuration File:

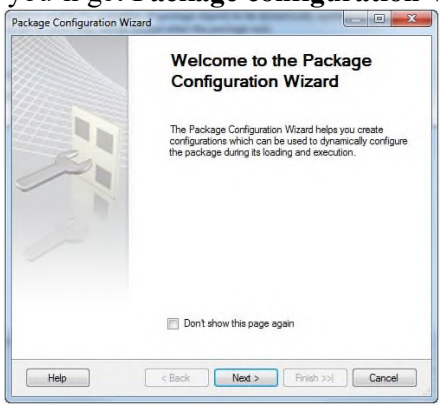
In this step, create a package configuration file that updates properties of package elements at runtime. To create a package configuration file, right click anywhere on the Control Flow area and select '**Package Configuration**' as shown below:



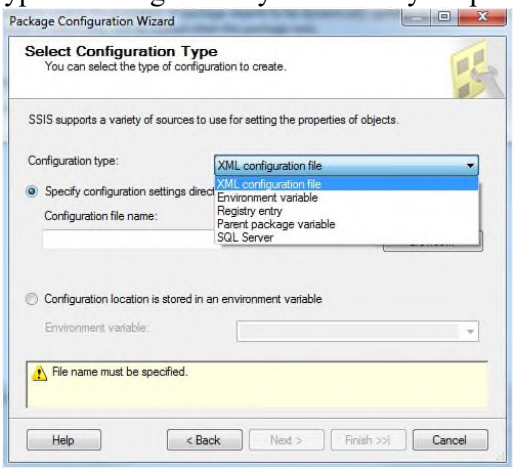
On clicking this you'll get **Package Configuration Organizer**, check '**Enable package configuration**' option and you'll get Add button to add your configuration as shown below.



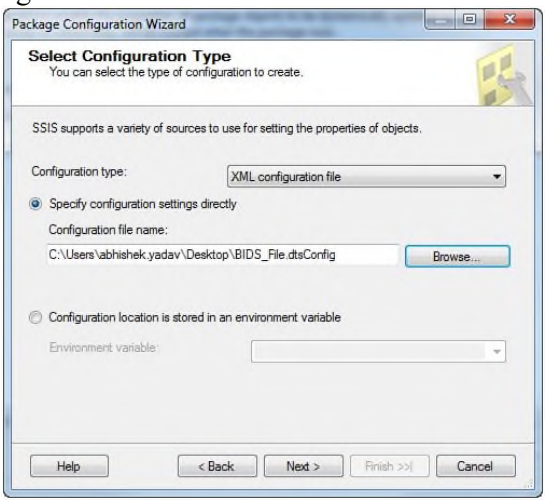
Click on the **Add** button and you'll get **Package configuration Wizard** as shown below.



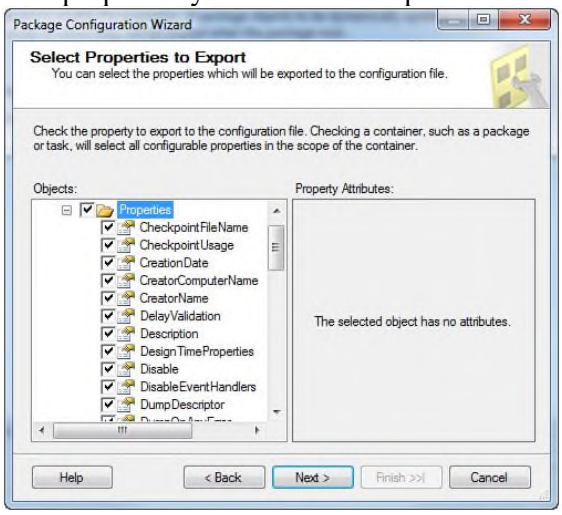
Click on **Next** and select type of configuration you want in your package.



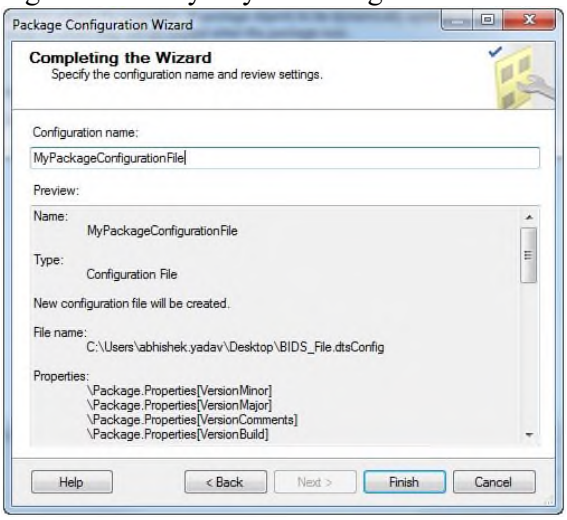
I'm selecting XML configuration file and will save the file at desired location.



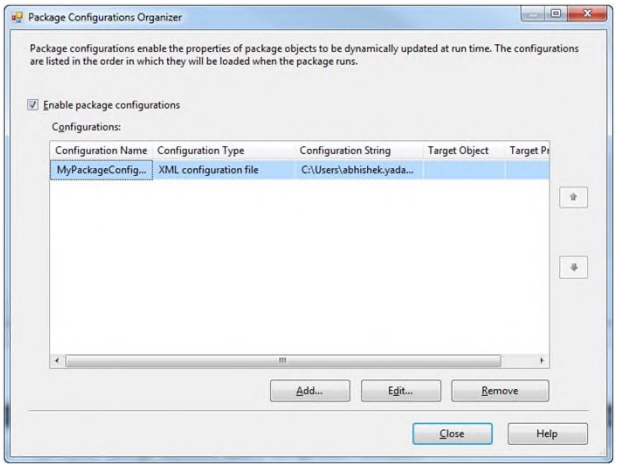
Click on Next and select the properties you want to be exported to the configuration file.



Click on **Next** and you'll get a summary of your configuration file.



Click on **Finish** to finalize your configuration file creation. After doing the above steps, you'll get something like below.

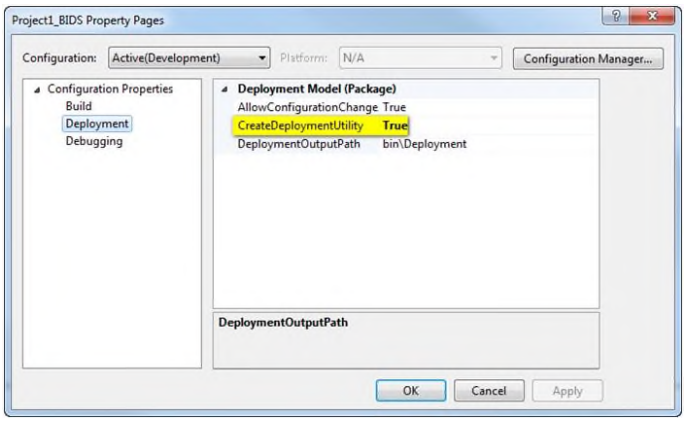


Now we're done with Step 1 where we created the package configuration file.

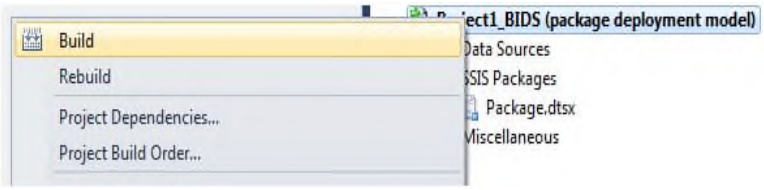
Step 2: To create a deployment utility

At this step, we require a package deployment utility for our project, which contains the package that we want to deploy.

To create Deployment Utility, right click on your package and select Properties and you'll get the following window:



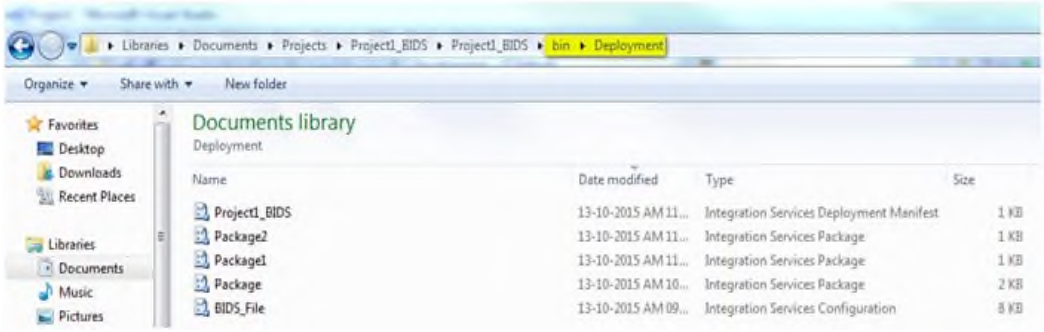
On this window, set **CreateDeploymentUtility** to '**True**' and click on OK.
Now build your Project as shown below.



On successful build, you will get the following message in the output window.

1. -----Build started: Project: Project1_BIDS (package deployment model), Configuration: Development------
2. Build started: SQL Server Integration Services project: Incremental...
3. Creating deployment utility...
4. Deployment Utility created.
5. Build complete -- 0 errors, 0 warnings
6. ========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========

If you go to **\bin folder** of your project, you will find a newly created directory named 'Deployment' with some files.



These files are nothing but your configuration file, Manifest file and your packages you want to deploy.

Step 3: Copy Deployment folder

Copy your Deployment folder under which you had built your project to the target computer on which you want to deploy your package.

Step 4: Package Installation

Install your package with help of the '**Package Installation Wizard**' to the file system or to an instance of SQL Server.

As we can see, we have added 3 packages in our project and if we want to deploy those packages, we need to do it one-by-one.

Therefore, this was some information regarding package deployment model. Now, let us move to the next deployment model.

Project Deployment Model

The Project Deployment Model was introduced in SQL Server 2012 Integration Services. This model provides variety of features than Package Deployment Model.

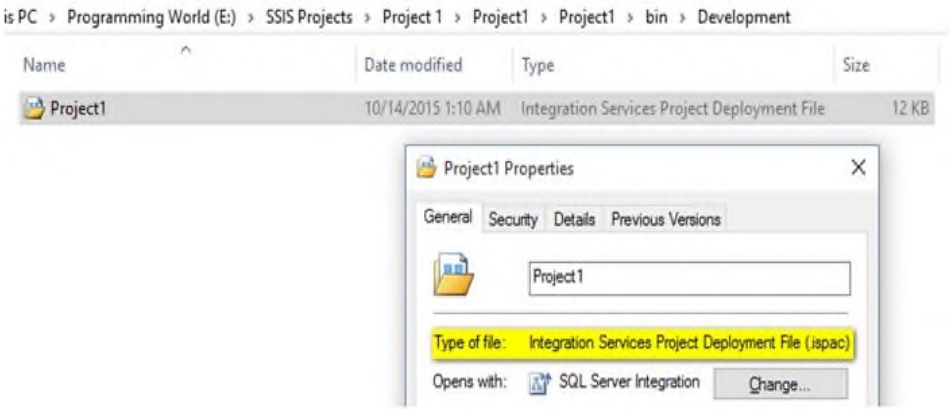
- In this model, the project is considered as a unit of deployment. This means we can deploy whole project.

- Projects containing packages and parameters are deployed to SSISDB catalog on SQL Server instance. During execution, events that are produced by the package are captured automatically and saved in the catalog.

- One disadvantage I found under this deployment model is that, you cannot deploy one or more packages without deploying the whole project. However, SSIS 2016 introduced an Incremental Package Deployment feature that allows you to deploy one or more packages without deploying the whole project.

- Deployment version or deployment history can be easily maintained in this model.

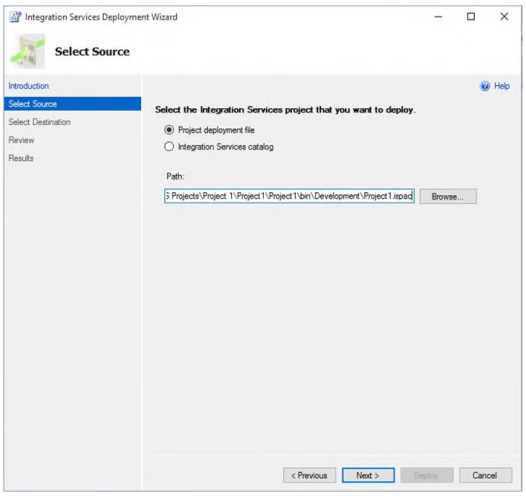
- When you build a project under this deployment model, it'll create a deployment file with .ispac extension as shown below.



The project deployment file shown above is a self-contained unit of deployment that includes only the essential information about the packages and the parameters of the project.

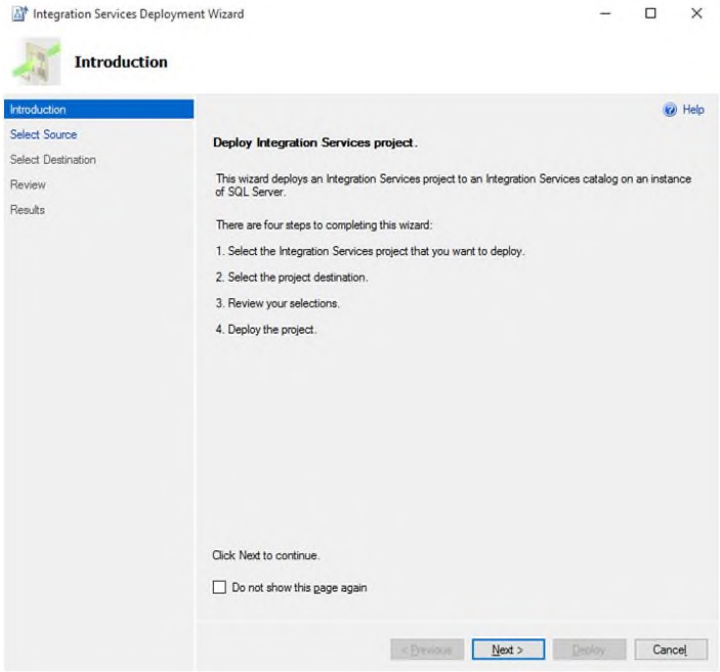
Project Deployment Model does not require any package configuration file as we did in Package Deployment. Here you can build and deploy your package under your catalog. Let us deploy our package, which we have created under Project deployment model.

After successful build, you will get '**Integration Services Project Deployment File (.ispac)**' file under a bin/Deployment folder. Double click on that file and it will open Deployment Wizard as shown below.
Click next and select your Project Deployment File as Source.

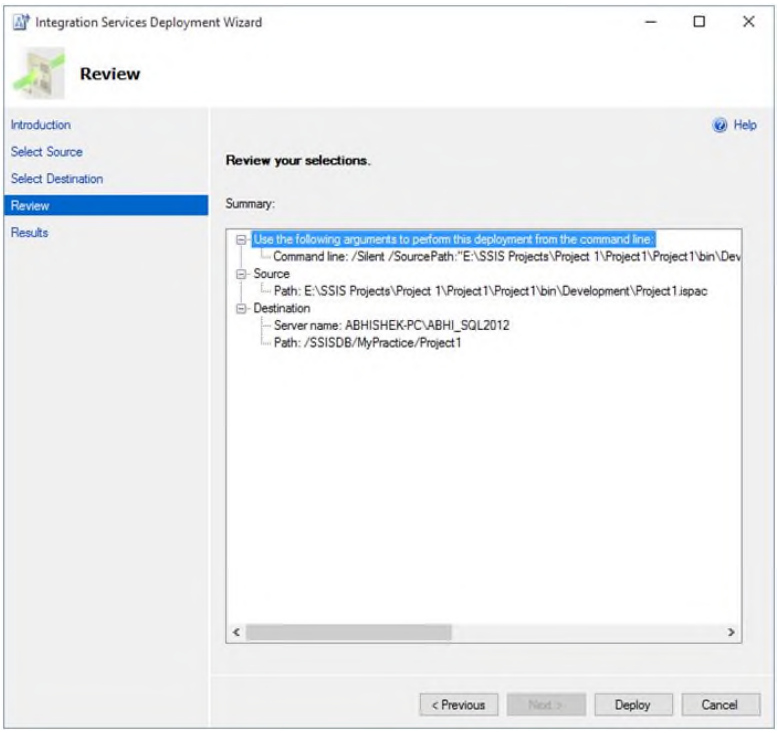


Click Next to choose Destination where you want to deploy your SSIS packages.

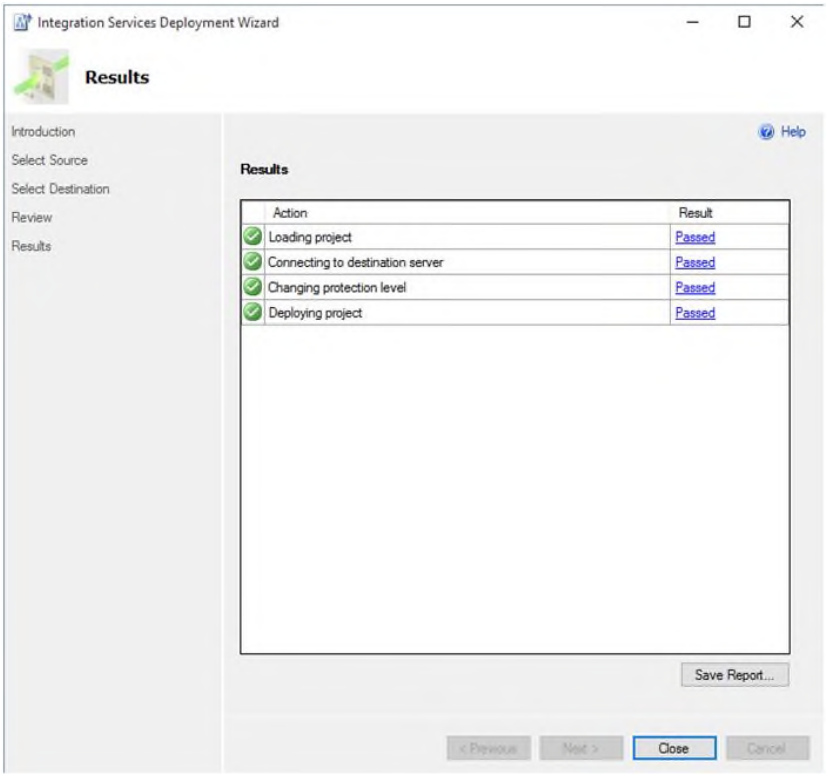
I have already created my Integration Service Catalog and I will deploy my package over there.



Click Next and you will get the summary of the deployment i.e. Source, Destination, etc.

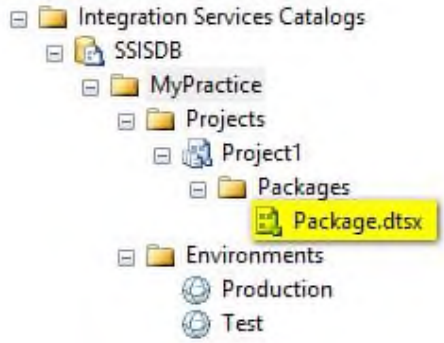


Click on **Deploy** to finalize the setup. On successful deployment, you will get the following window.



We have successfully deployed our package in this Model, which seems very simple as compared to Package Deployment Model.

Now, if you open Integration Services Catalog under SQL Server, you will find your deployed package as shown below.



Therefore, with this, we are done with Deployment models available in SSIS.

Planning SSIS Packages Execution

A common requirement is to execute several SSIS packages as part of one logical process. This is especially true when one is transferring existing data from an external system.
If the individual packages are placed in a folder and named in such a way that a batch can be executed via a wildcard file mask, then they can be altered without changing any code, and

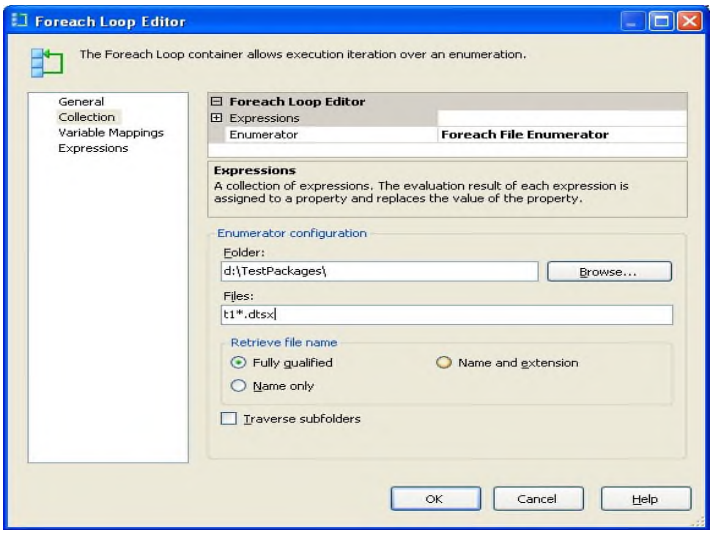
can be used for different processes by changing the file mask. This article presents three different methods of executing all packages in a folder:

1. **SSIS control package**
2. **Stored procedure**
3. **DOS batch file**

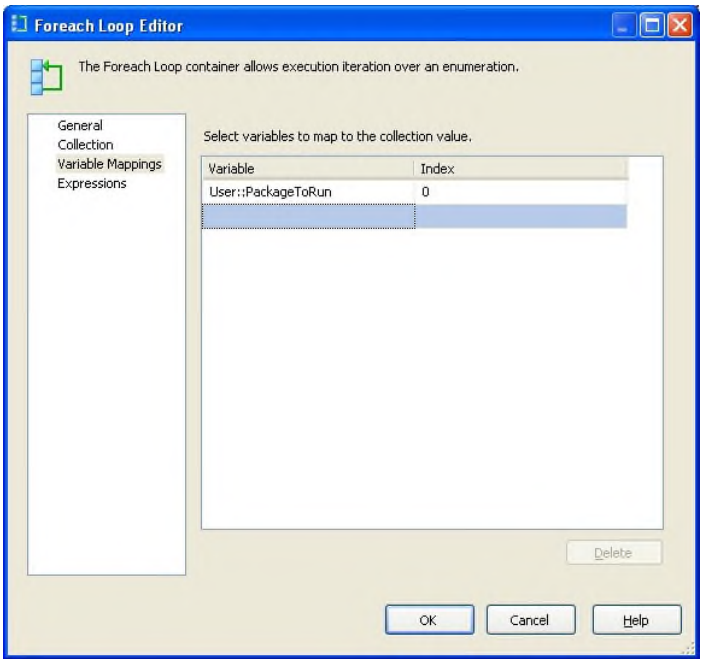
The methods are of differing complexity and flexibility, the SSIS package being by far the most difficult to code, and the DOS command being by far the simplest but also the least flexible.
The folder that contains the packages is d:\TestPackages\.
All the packages to be run have names of the format t1…….dtsx.

SSIS control package
For this we create a package that uses a for each loop task to call an execute package task for all packages in the folder.
The first task is to create an SSIS package with a Foreach Loop container that will loop round the packages in the folder, setting a variable called "PackageToRun" to the file name for each package.

1. Load Business Intelligence Development Studio and start a SSIS project.
2. Create a new package.
3. Add a Foreach Loop container to the package.
4. Right-click on the **Foreach Loop** container and select **Edit**.
5. Click on **Collection**.
6. Set the Enumerator to **Foreach File Enumerator**.
7. In the Enumerator configuration:
    a. Set **Folder** to "d:\TestPackages\"
    b. Set **Files** to "t1*.dtsx"
    c. Under **Retrieve file name** select **Fully qualified**.
8. Click **OK**.

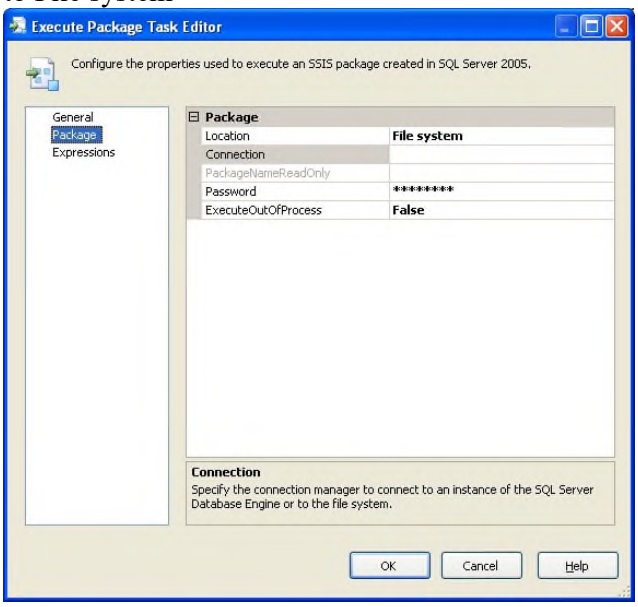


1. Click on **Variable Mappings**.
2. Click on the **Variable** drop-down list and select **New Variable**.
3. Set Name to PackageToRun.
4. Click **OK**.



Next, we need to add the **Execute Package** task to the Foreach Loop container so that this task will be executed for each package that we wish to run. We then set the variable value to be the name of the package to be executed by the **ExecutePackage** task.

1. Drag an **Execute Package** task into the Foreach Loop container.
2. Right-click on the **Execute Package** task and select **Edit**.
3. Select **Package**.
4. Set Location to File system



1. Click on the **Connection** drop-down list and select <New connection…>.
2. Set the File to an existing package.

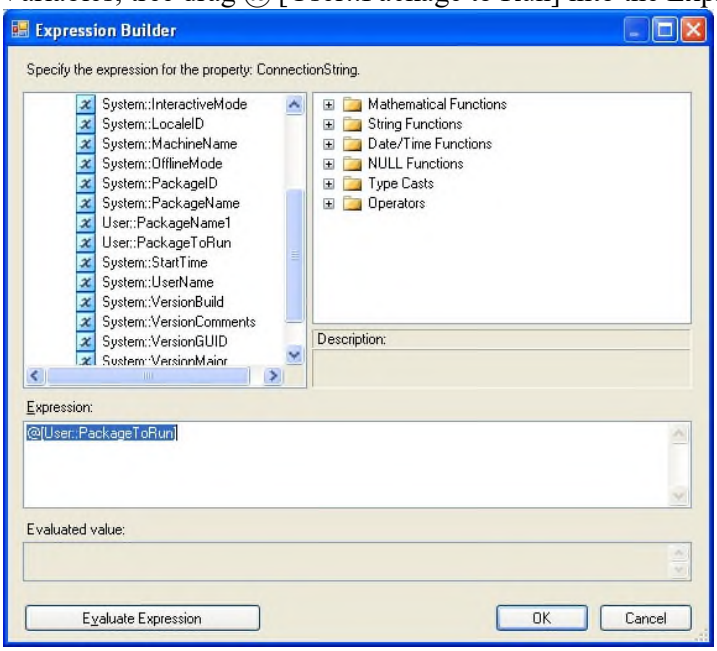
1. Click **OK** to save the connection.
2. Click **OK** to complete the Execute Package task configuration.

Finally, we configure the connection to use the variable package name:

1. Right-click on the connection and select **Properties**.
2. In the Properties window change the name to PackageToRunConnection.

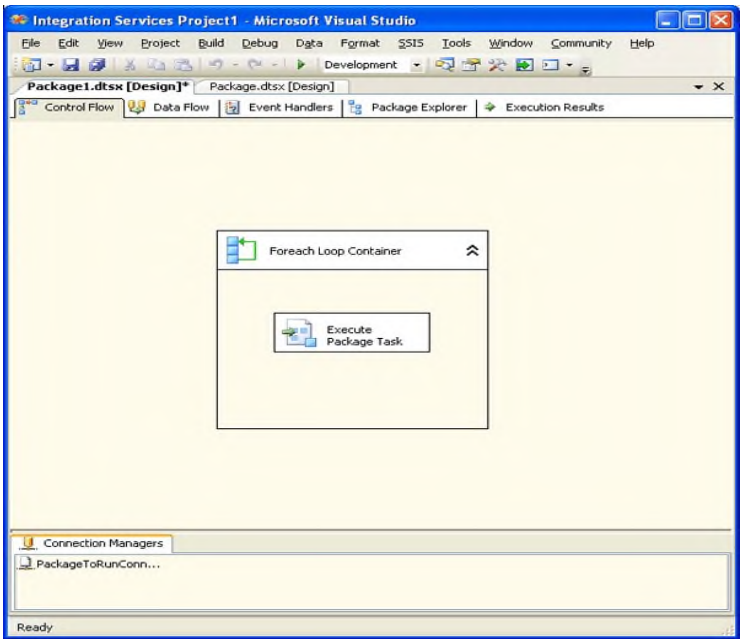
***Note**: this will automatically update the connection name in the Execute Package Task.*

1. Select Expressions and add a new expression.
2. In the property drop-down list, select **ConnectionString**.
3. Click on the **Expression Editor** Button.
4. From the Variables, tree drag @ [User::Package to Run] into the Expression window.



1. Click **OK** twice, to save the expression.

The package should now look like this:



If you now run the package, it will execute each package in the folder with the correct filename mask.

Stored Procedure

This involves the following steps:

1. Retrieve a list of the package names to run from the folder
2. Loop through each package name
3. Execute each package

First, we need to get a list of all packages in the folder. The following code executes the DOS command **dir /B d:\TestPackages\ t1*.dtsx** and places the output in the temporary table **#dir**. The DOS command gets a list of the filenames in the folder, which satisfy the filemask.

```
Create table #dir (Filename varchar(1000))
Insert #dir
Exec master..xp_cmdshell 'dir /B d:\TestPackages\ t1*.dtsx'
delete #dir where Filename is null or Filename like '%not found%'
```

The following is a useful technique for looping through values in a table and here we use it to loop through the filenames in **#dir**:

```
Declare @Filename varchar(1000)
Select @Filename = ''
While @Filename < (select max(Filename) from #dir)
Begin
   Select @Filename = min(Filename) from #dir where Filename > @Filename
End
```

Now we just need to create the command to execute the package:

```
Declare @cmd varchar(1000)
select @cmd = 'dtexec /F " d:\TestPackages\' + @FileName + '"'
exec master..xp_cmdshell @cmd
```

This takes the package name and uses the **dtexec** command to execute it. Putting this all together, we get:

```
Declare @Filename varchar(1000)
```

```
Declare @cmd varchar(1000)
Create table #dir (Filename varchar(1000))
Insert #dir
Exec master..xp_cmdshell 'dir /B d:\TestPackages\t1*.dtsx'
delete #dir where Filename is null or Filename like '%not found%'
Select @Filename = ''
While @Filename < (select max(Filename) from #dir)
Begin
   Select @Filename =  min(Filename) from #dir where Filename > @Filename
   select  @cmd =  'dtexec /F "d:\TestPackages\' + @Filename + '"'
   exec master..xp_cmdshell @cmd
end
drop table #dir
```

The above completes the code needed for the requirement but to simplify the implementation I like to keep a utility stored procedure, which calls another stored procedure for each file (SSIS package) in a folder. The following utility stored procedure, **ProcessAllFilesInDir**, uses the above techniques to obtain a list of files from a folder and calls another stored procedure for each one. The file path and mask are input as parameters together with the **@ProcSP** parameter that defines the stored procedure to be called for each file (in this case this will be the **ExecutePackage** procedure that executes each SSIS package).

```
Create procedure [dbo].[s_ProcessAllFilesInDir]
@FilePath varchar(1000) ,
@FileMaskvarchar(100) ,
@ProcSp varchar(128)
as
   set nocount on

declare @File varchar(128) ,
      @MaxFile varchar(128) ,
      @cmd     varchar(2000)

   create table #Dir (s varchar(8000))

   select @cmd = 'dir /B ' + @FilePath + @FileMask
   insert #Dir exec master..xp_cmdshell @cmd

   delete #Dir where s is null or s like '%not found%'

   select @File = '', @MaxFile = max(s) from #Dir
   while @File < @MaxFile
   begin
      select @File = min(s) from #Dir where s > @File

      select @cmd = @ProcSp + ' "' + @FilePath + '" , "' + @File + '"'

      exec (@cmd)

   end
   drop table #Dir
```

```
go
```

With this, the only coding needed is to create the procedure to execute the package and to call it. The **ExecutePackage** procedure is as follows:

```
Create procedure [dbo].[s_ExecutePackage]
@FilePath   varchar(1000) ,
@Filename   varchar(128)
as
Declare @cmd varchar(1000)
  select @cmd =  'dtexec /F "' + @FilePath + @Filename + '"'
  exec master..xp_cmdshell @cmd
go
```

And the call:

```
Exec [dbo].[s_ProcessAllFilesInDir]
  @FilePath = 'd:\TestPackages\' ,
  @FileMask = 't1*.dtsx' ,
  @ProcSp = '[dbo].[s_ExecutePackage]'
```

DOS batch file

The DOS command required is **dtexec /F "d:\TestPackages\mypackagename"**as above. We can make use of a DOS For loop to execute this command for all files in a folder:

```
for %1 in (d:\TestPackages\t1*.dtsx) do dtexec /F "%1"
```

This uses the for loop construct:

- **(d:\TestPackages\t1*.dtsx)** returns a list of the files, which meet the path (similar to a dir command).
- **For %1 in (d:\TestPackages\t1*.dtsx)** loops through the file list setting the variable %1 to the filename.
- **Do dtexec /F %** executes **dtexec /F** with each filename in turn.

Note that you can place this command in a **bat** file to execute.

Introduction to Business Intelligence

- 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.

Benefit of Business Intelligence:

1) 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.

2) 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.

3) 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.

4) 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.

Introduction to Reporting

SQL Server Reporting Services (SSRS) is a reporting software that allows you to produce formatted reports with tables in the form of data, graph, images, and charts. These reports are hosted on a server that can be executed any time using parameters defined by the users. It is part of Microsoft SQL Server Services suite.

Types of reporting services

Three types of SSRS reporting services are:

1. Microsoft SQL Server Integration services which integrate data from different sources.
2. Microsoft SQL Server Analytical service which helps for the analysis of the data
3. Microsoft SQL Server Reporting service allows for generating a visual report of the data.

Why SSRS?

Here, are prime reasons for using SSRS tool:

- SSRS is an enhanced tool compared to Crystal Reports
- Faster processing of reports on both relational and multidimensional data
- Allows better and more accurate Decision-making mechanism for the users
- Allows users to interact with information without involving IT professionals
- It provides a World Wide Web-based connection for deploying reports. Hence, reports can be accessed over the internet
- SSRS allows reports to be exported in different formats. You can deliver SSRS reports using emails
- SSRS provides a host of security features, which helps you to control, who can access which report

Features of SSRS

- Offers a Simple Object Access Protocol (SOAP) application and pluggable architecture
- Retrieve data from managed, OLE ODBC, and DB connections
- Allows you to create adhoc reports and save them to the server
- Display data in a variety of formats which includes tabular, free-form, and charts

- Create custom controls by using report-processing extensionEmbed graphics, images to the reports. You can also integrate with external content using SharePoint
- You can store and manage custom reports
- Chart and Gauge control feature allows you to display KPI data

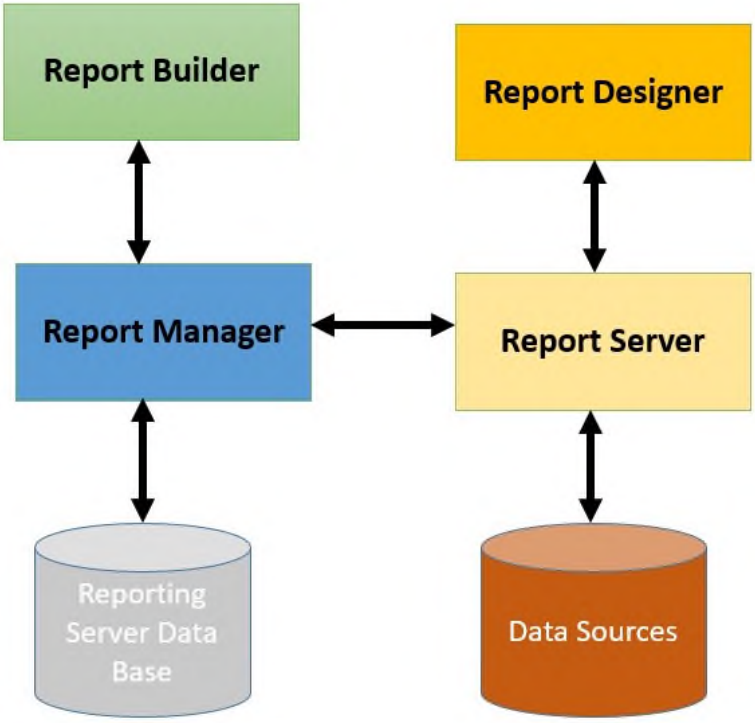
How SSRS Works?

Now in this SSRS tutorial, we will learn how SSRS works:

How SSRS Works

1. The report users are the peoples who work with the data as well as want some insights from data. They send a request to the SSRS server
2. SSRS server finds the metadata of the report and sends a request for data to the data sources.
3. Data returned by the data source is merged with the report definition into a report.
4. When the report is generated, it is returned to the client.

SSRS Architecture



SSRS Architecture
SSRS has quite a complex architecture. The report services architecture includes development tools, administration tools, and report viewers.
Here, are important components of SSRS
Report Builder
It is an ad-hoc report publishing tool that is executed on a client's computer. It has a drag and drop interface that is easy to use.
Report Designer
The Report designer tool helps to develop all types of reports. It is a publishing tool, that is hosted in Visual Studio or Business Intelligence Development Studio (BIDS).
Report Manager
The report managers check the report, matching it with the given requirements. They make decisions based on those reports.
Report Server
It is a server which use the SQL Server database engine to store metadata information
Report server database
It stores metadata, report definitions, resources, security settings, delivery data, etc.
Data sources
Reporting services retrieve data from data sources like relational and multidimensional data sources.

Reporting Life Cycle
Every organization follows a standard reporting lifecycle which can be classified as follows:

Authoring: In this phase, the report author defines the layout and syntax of the data. The tools used in this process are the SQL Server Development Studio and SSRS tool.
Management: This phase involves managing a published report which is mostly part of the websites. In this stage, you need to consider access control over report execution.
Delivery: In this phase, you need to understand when the reports need to be delivered to the customer base. Delivery can be on-demand or pre-defined schedule. You can also add an automation feature of subscription which creates reports and sends to the customer automatically.

What is RDL?
Report Definition Language is shortly known as RDL. It describes all possible elements of a report using an XML grammar which is validated by an XML schema.
The report definition of an individual report is based on RDL. It contains instructions for rendering the design of the report at the run time.

Type of SSRS reports
Here, are types of reports which you can develop using SSRS tool.

Report Type Name	Detail
Parameterized reports	This type of report uses input values to complete report or data processing.

Linked Reports	A linked report offers a point to an existing report. This type of report is derived from an existing report and retains the original's report definition.
Snapshot reports	A snapshot report contains layout information and query results which can be retrieved at a specific point in time.
Cached Reports	The cashed report allows you to create a copy of the processed reports. They are used to enhance the performance by reducing the number of processing requests and time to retrieve large reports.
Drill down Reports	Drill down reports helps you to hide complexity. It allows users toggle between hidden reports items to control how much detail data you want to see. It must retrieve all possible information that can be shown in the report.
Drillthrough Reports	Drillthrough reports are standard reports which can be accessed through a hyperlink on a text box in the original report. It works with the main report and is the target of a drill through an action for a report item like a placeholder text or a chart.
Subreports	As the name suggests, sub-reports is a report which displays another report inside the body of the main report.

Advantages of using SSRS

- It is faster and cheaper.
- Efficient reporting access to information that is residing in both MS SQL Server database or Oracle
- No need for expensive specialist skills
- In SSRS the default report designer is integrated with Visual Studio .NET. This allows us to create an application and reports in the same environment.
- The security is managed in a role-based method which can be applied to folders and reports.
- Subscription-based reports are automatically sent to the users.
- Faster production of reports on both relational and cube data
- Real time information to the business, providing better decision support

Disadvantages of using SSRS

Some limitations of SSRS are given below:

- There is no print button. So if you want to print something you need to export PDF, excel, word or other formats.
- All reports need parameters to be accepted by users.
- It is difficult to make changes in the custom code and debug expressions.
- Does not allow you to add page number or total pages in the report body
- Does not offer any method to pass values form sub-reports to the main report
- Page header always creates extra spaces on every new page.

Basic Reports in SQL Reporting Services

SQL Server Reporting Services (SSRS) is part of Microsoft SQL Server services since version 2000. If you are an ordinary user with a basic knowledge of SQL, now create first SSRS report.

Requirements for creating a SSRS report

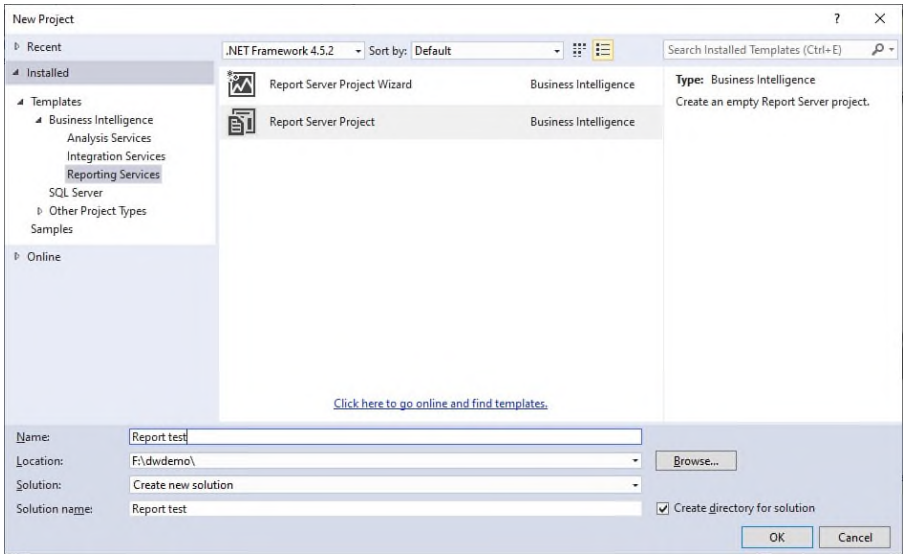
Before you begin using SSRS and creating reports, you must have:

- Microsoft SQL Server 2014 database engine
- SQL Server 2014 reporting service (SSRS)
- SQL Server data tools
- AdventureWorks2014 database is used for this example, as **data source**.

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, find **New** 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 type **Report test** in **Name**
- The 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 the **Reports** 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.

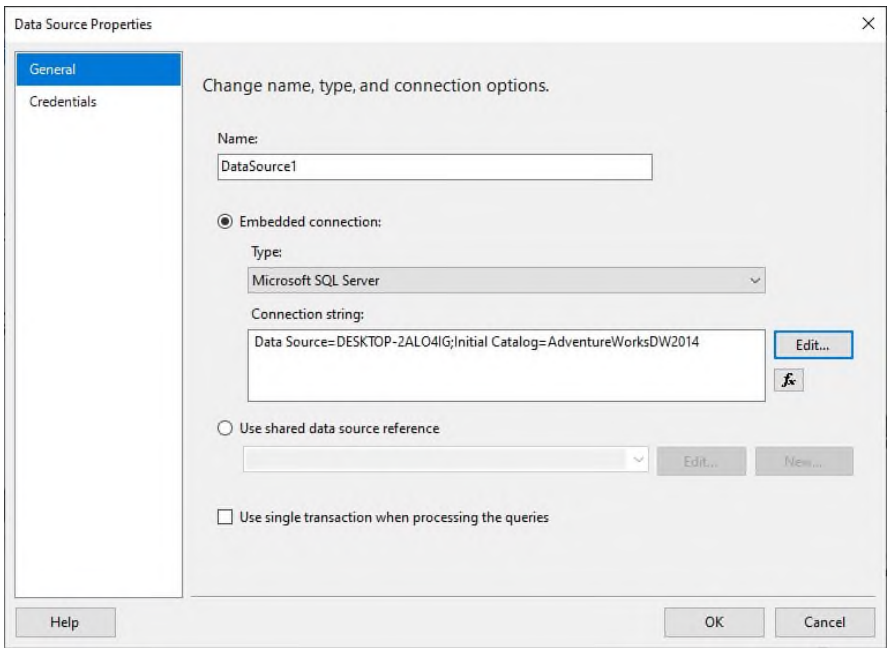
Defining data source and dataset

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 AdventureWorks2014 database as an example.

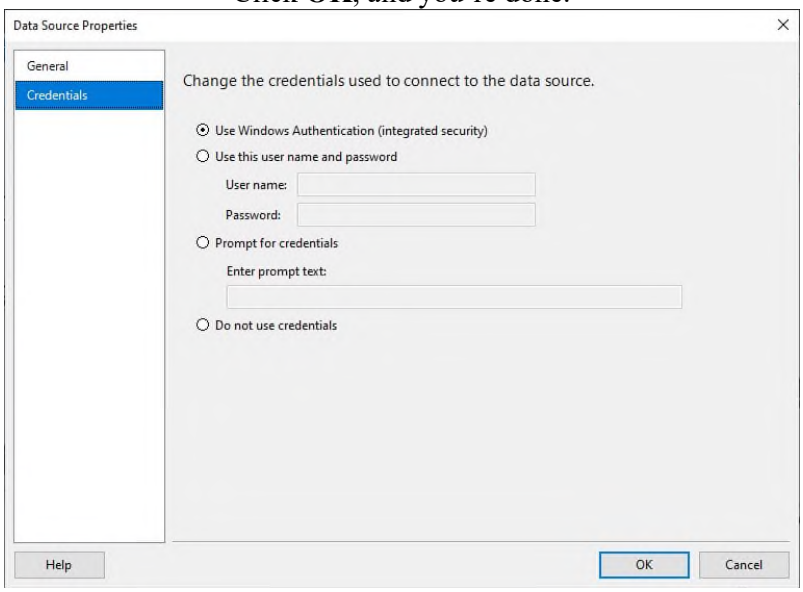
- In the **View** menu find and click Report1 Data, then **New** and after that ***Data Source***
- Type *AdventureWorks2014* in the field **Name**
- Select **Embedded connection**
- As Type select **Microsoft SQL Server**
- In **Connection string** field type:

- Data source=localhost; initial catalog=AdventureWorks2014



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.

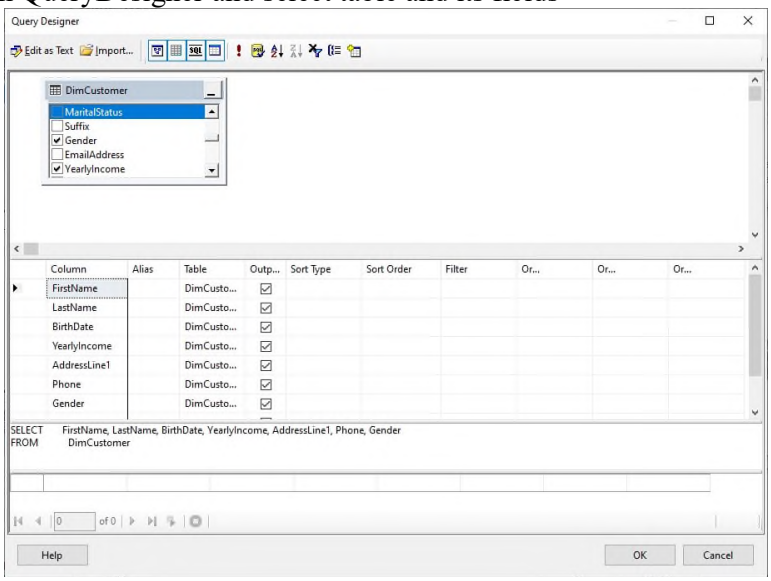


Define a SQL query for report data

Write 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

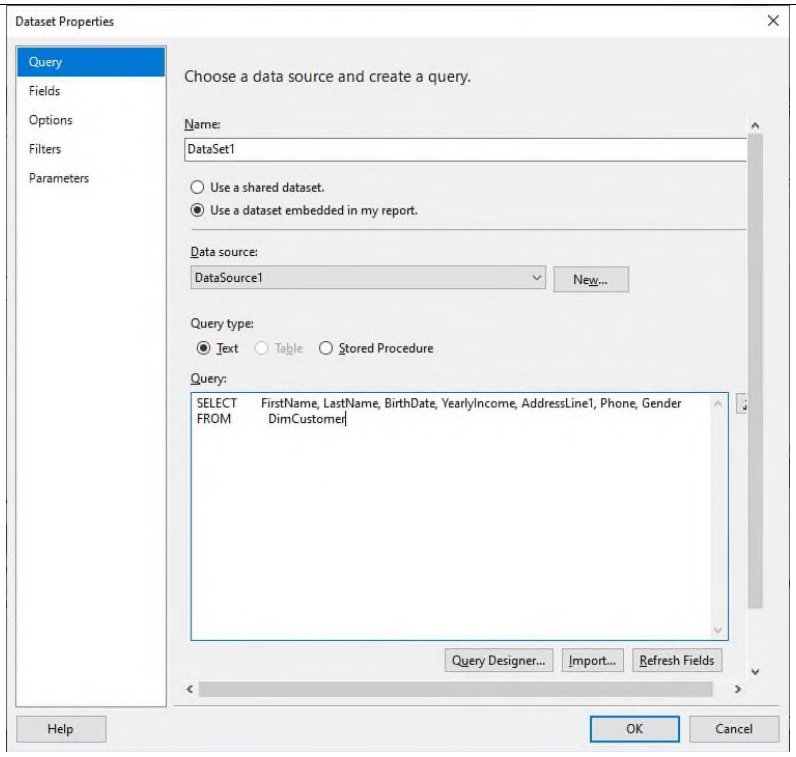
- Make sure to check **Use a dataset embedded in my report**.
- Then select a AdventureWorks2014 as your **data source**,
- Click on QueryDesigner and select table and its fields



OR

- Check **Text** as a **Type**, and type this into the Query input:

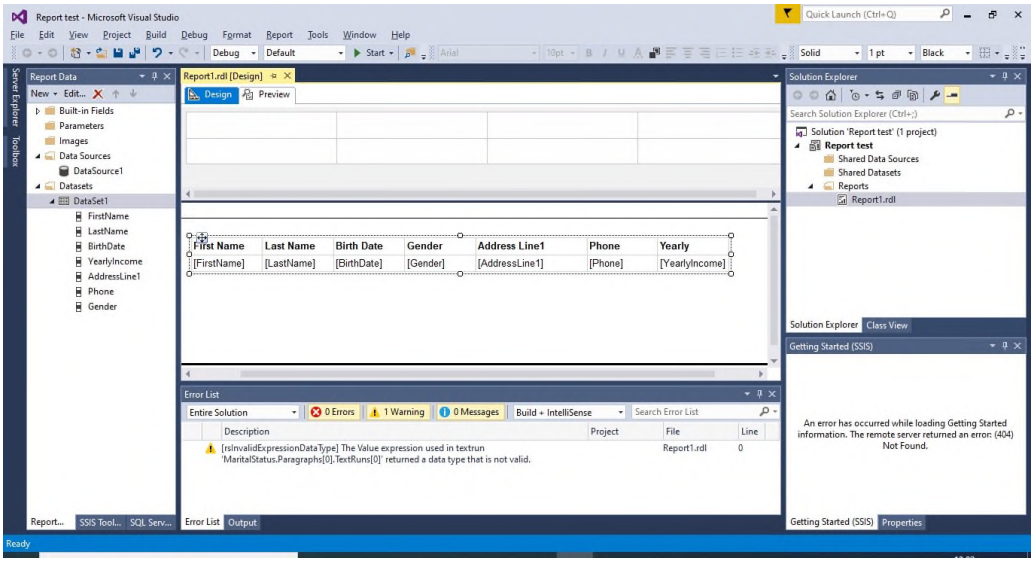
```
SELECT      FirstName, LastName, BirthDate, YearlyIncome, AddressLine1,
Phone,Gender
FROM        DimCustomer
```



How to add a table and fields to a report layout

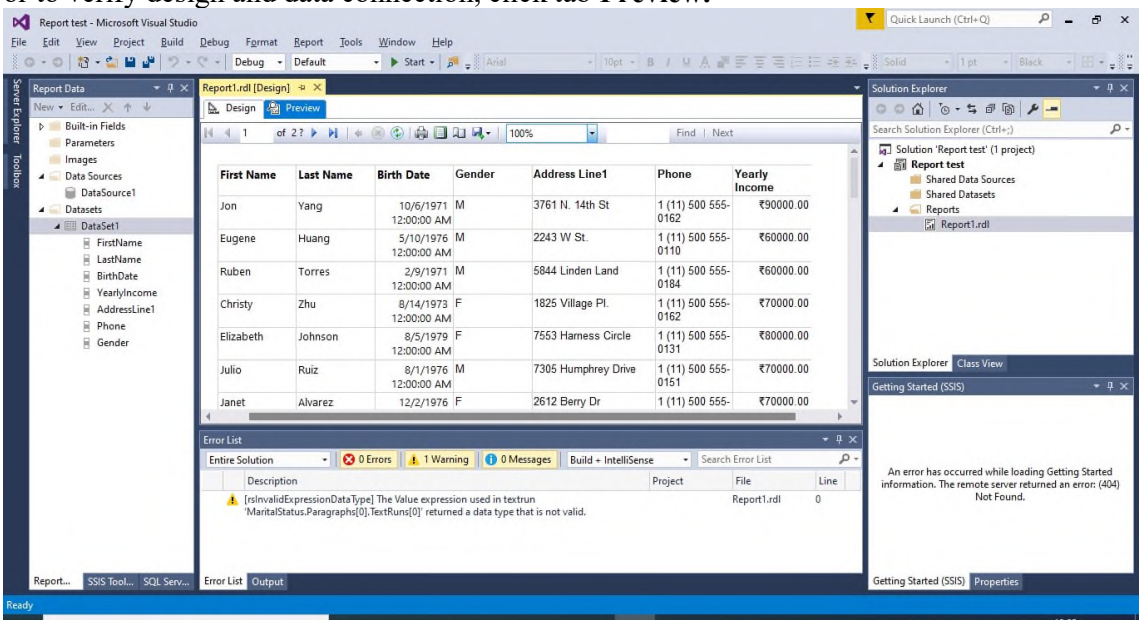
After finishing with previous technical details, we can start with more interesting things, like designing your first SSRS report. This part is easy because instead of writing code, you can drag-and-drop graphic icons into the report format. This section will show you how to add Table and Fields to your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

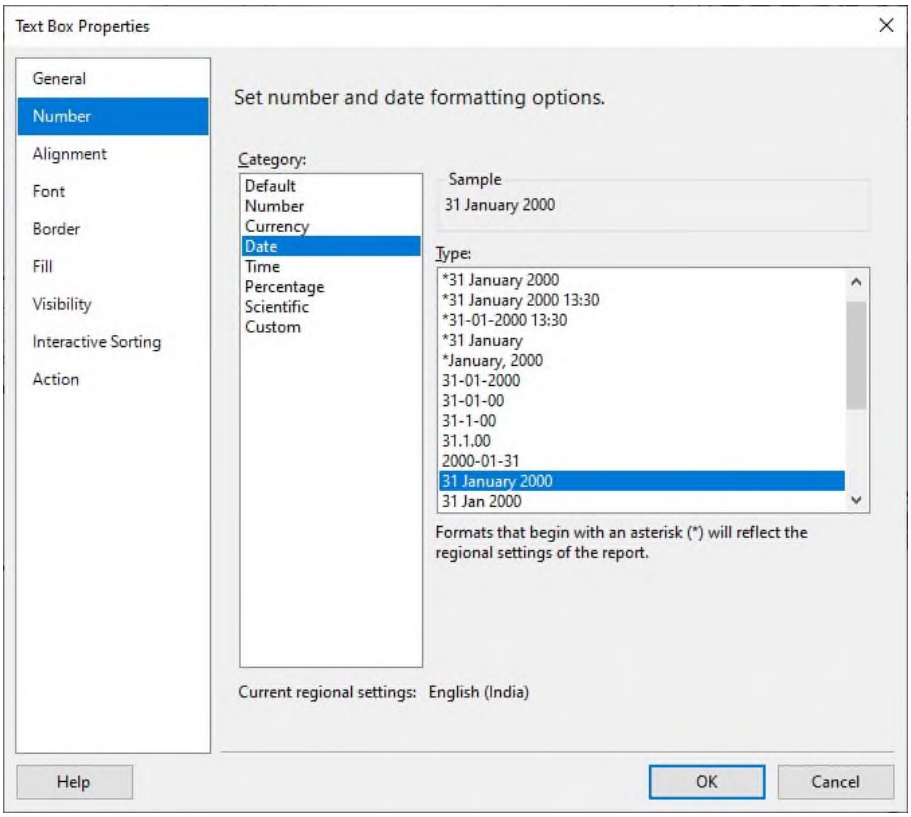
An easier way to add a table is to right-click on the design surface, click **Insert** and then click **Table.**

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:

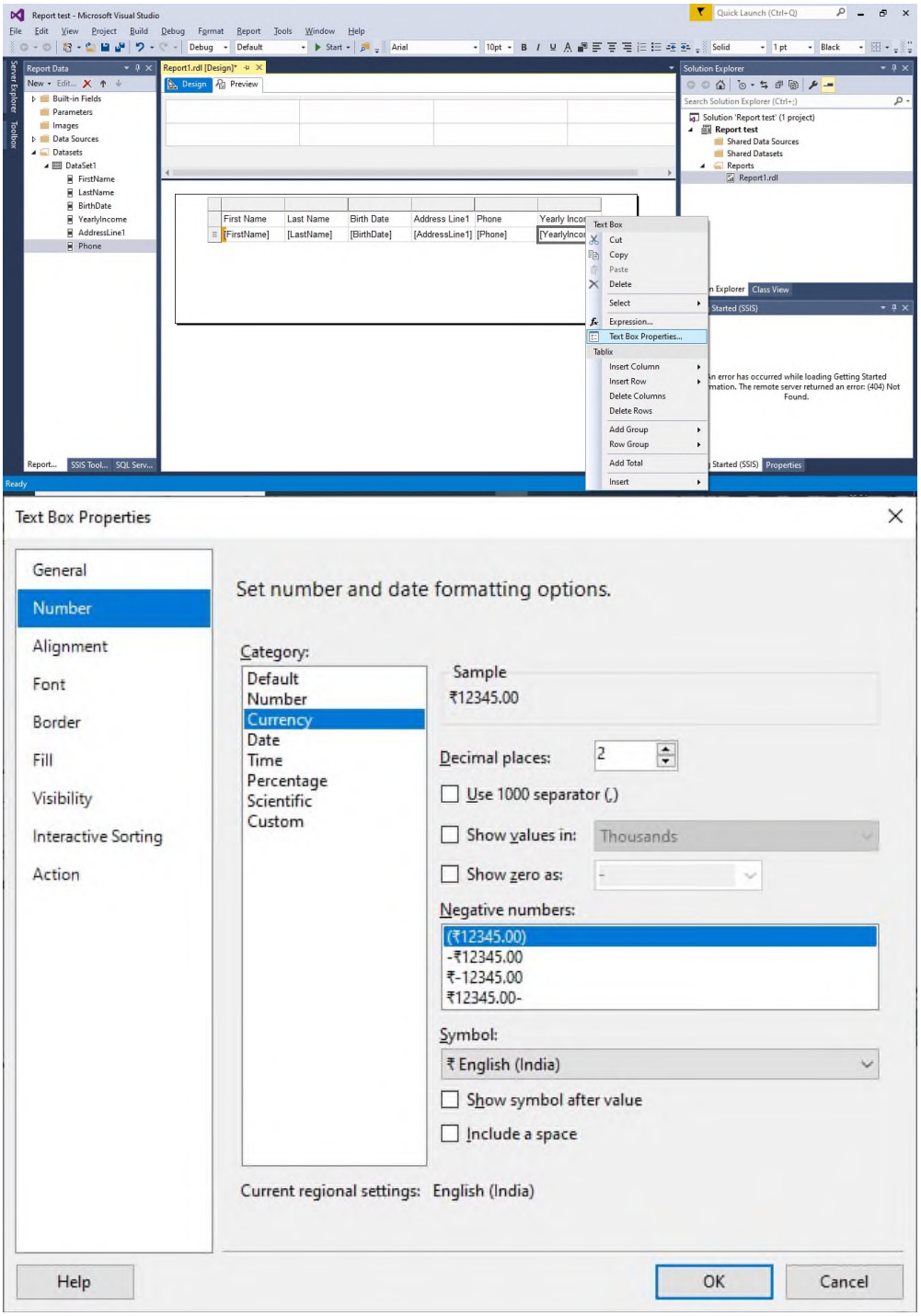
- In the **Design** tab, right-click the desired cell, then click **Text Box Properties.**
- Find and click **Number**, then in the field **Category** click **Date**
- As **Type** select date format you want, and click **OK** to finish.



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.)

Publish your Report to the report server

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

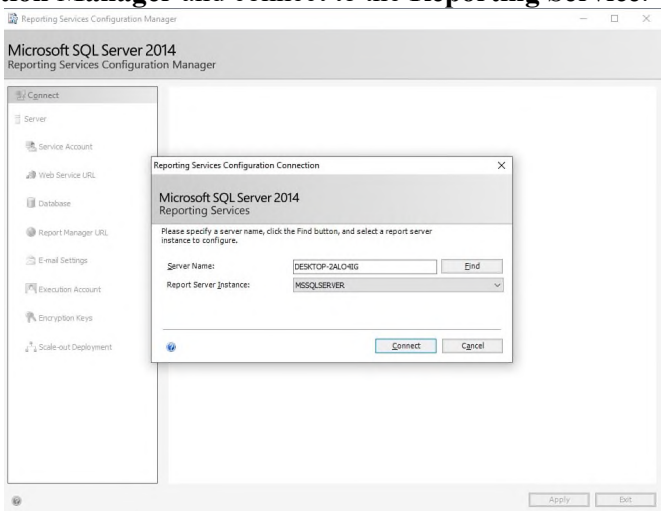
- In the **Project** menu click **Test report Properties**
- Put your report server's name in the field TargetServerURL, then click OK.
- In the **Build** menu click **Deploy Test report**. You will get a message that indicates whether you have successful or unsuccessful deployment.

Note: A problem will occurred if you don't have permissions on the report server or if you have been using SSDT with administrator privileges.

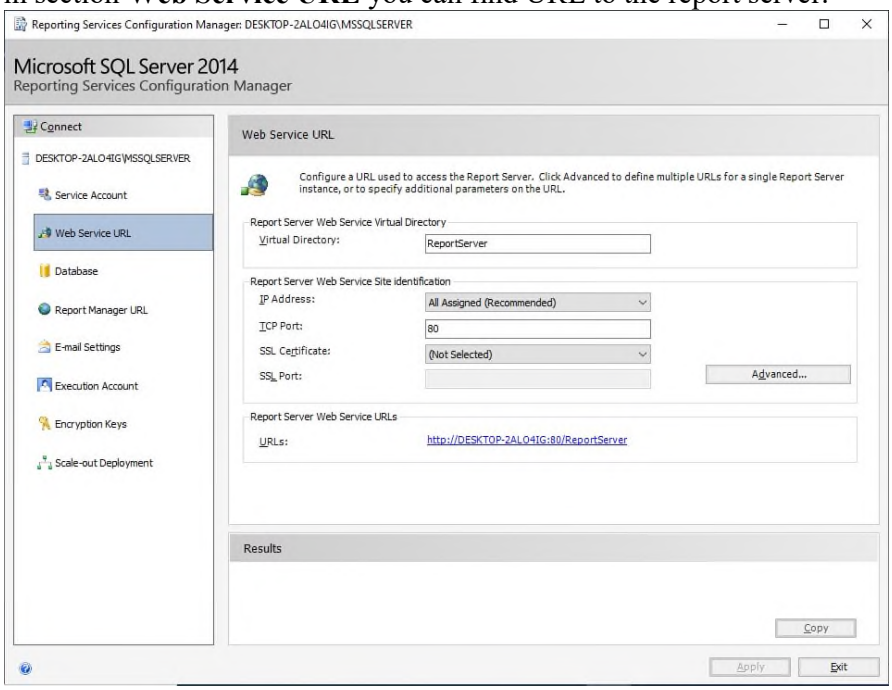
Most common problems

The most common problem in publishing the report to the report server is configuring the target server URL.

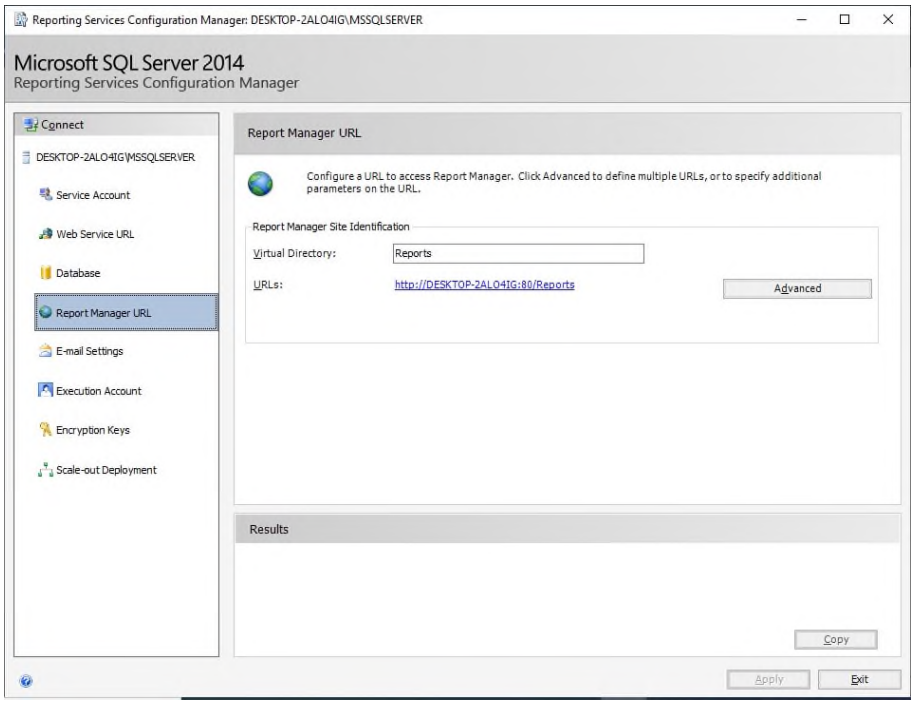
The first step that you need to do in configuration is to launch the **SQL Server Reporting Service Configuration Manager** and connect to the **Reporting Service**.



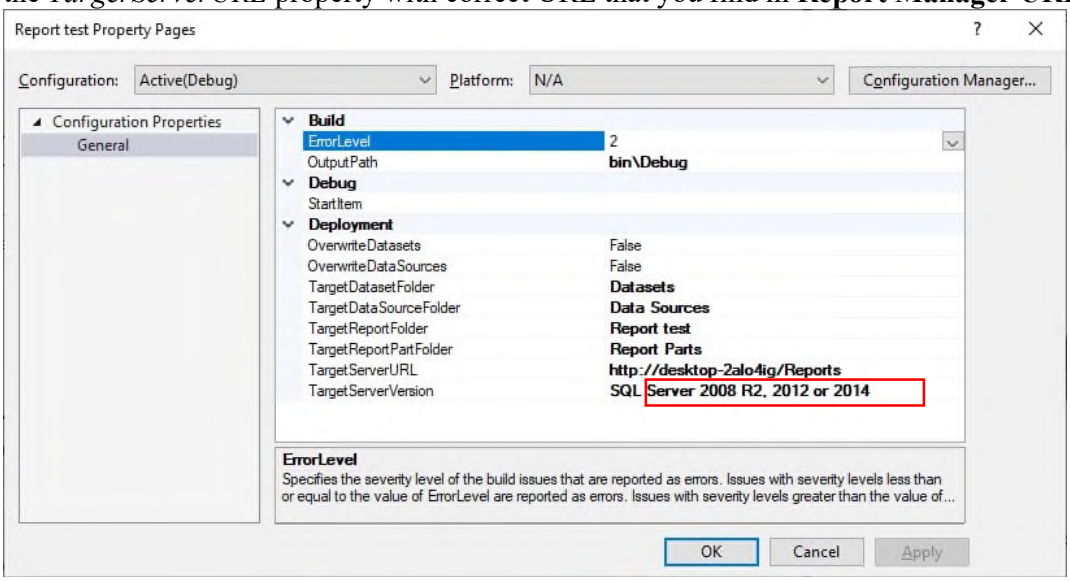
After that in section **Web Service URL** you can find URL to the report server.



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



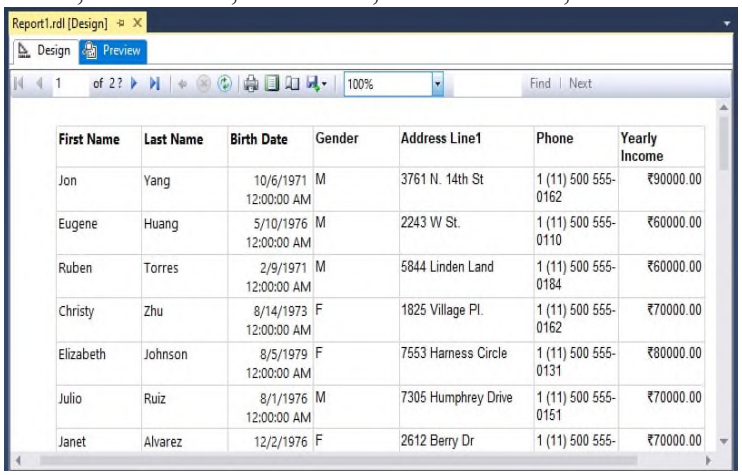
In the **Project** menu open the **Test report Properties**.
Note: Prefix "Test report" is name of your report.
Final step in configure target server URL is to fill the Web Service URL in the *TargerServerURL* property with correct URL that you find in **Report Manager URL**.



Congratulation, you have successfully created your first SSRS report!

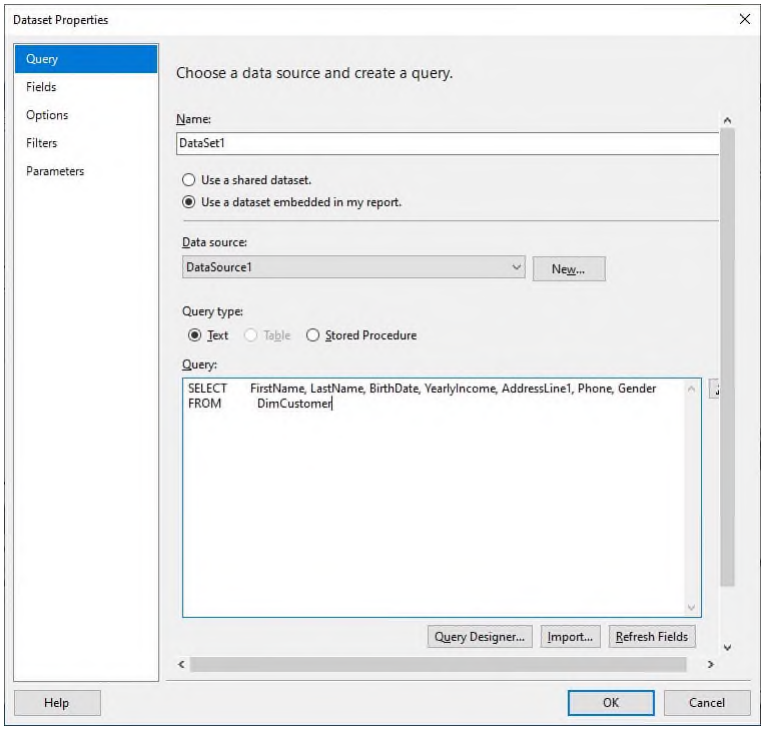
Parameters in Reporting Services

- The SSRS Report Parameters allows the users to enter a specific value in the textBox and then, SSRS will filter the Report data using the user-specified value. OR Report Parameters enables the users to Filter the SSRS Reports Dynamically.
- **how to add Report parameters in SSRS or SQL Server Reporting Services with an example.**
- We are going to use the below-shown report to explain, SSRS Report parameters in Reports.
- First we have create report project.
- If you observe the below screenshot, It was a normal Table report with First Name, Last Name, Birth Date, Address Line1,Phone and Yearly Income columns.



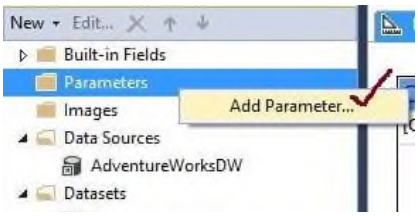
First Name	Last Name	Birth Date	Gender	Address Line1	Phone	Yearly Income
Jon	Yang	10/6/1971 12:00:00 AM	M	3761 N. 14th St	1 (11) 500 555-0162	₹90000.00
Eugene	Huang	5/10/1976 12:00:00 AM	M	2243 W St.	1 (11) 500 555-0110	₹60000.00
Ruben	Torres	2/9/1971 12:00:00 AM	M	5844 Linden Land	1 (11) 500 555-0184	₹60000.00
Christy	Zhu	8/14/1973 12:00:00 AM	F	1825 Village Pl.	1 (11) 500 555-0162	₹70000.00
Elizabeth	Johnson	8/5/1979 12:00:00 AM	F	7553 Harness Circle	1 (11) 500 555-0131	₹80000.00
Julio	Ruiz	8/1/1976 12:00:00 AM	M	7305 Humphrey Drive	1 (11) 500 555-0151	₹70000.00
Janet	Alvarez	12/2/1976	F	2612 Berry Dr	1 (11) 500 555-	₹70000.00

The below screenshot will show you the Shared Data Source and Dataset we used for this report.



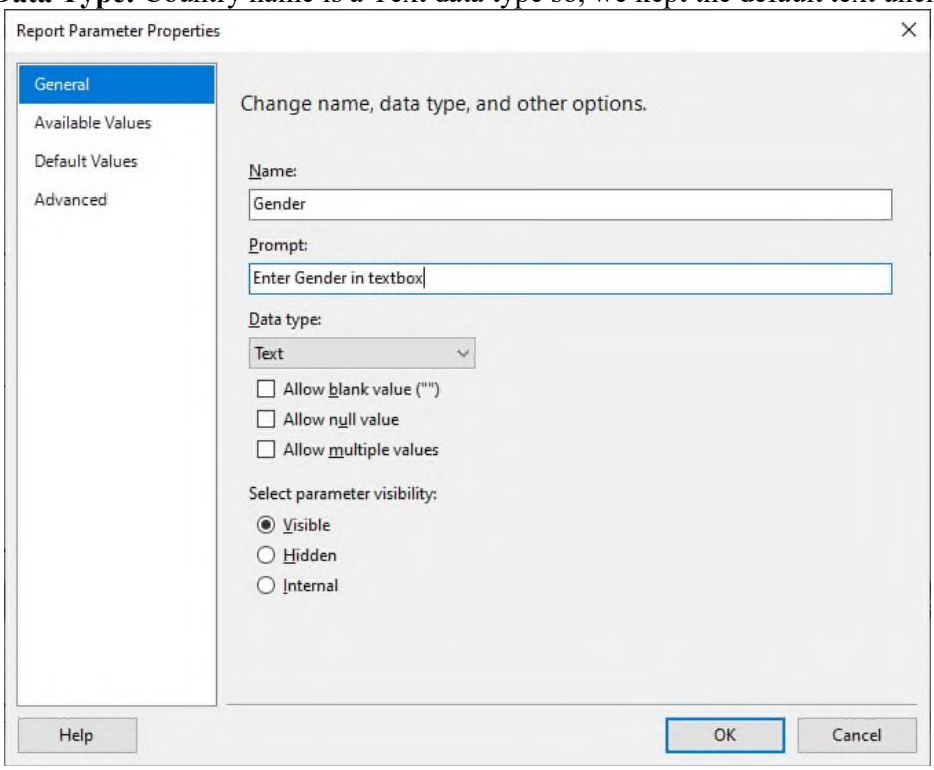
Adding SSRS Report Parameters

To add SSRS Report Parameters, Right Click on the **Parameters** Folder present in the **Report Data** tab will open the Context Menu to select **Add parameters..** option.



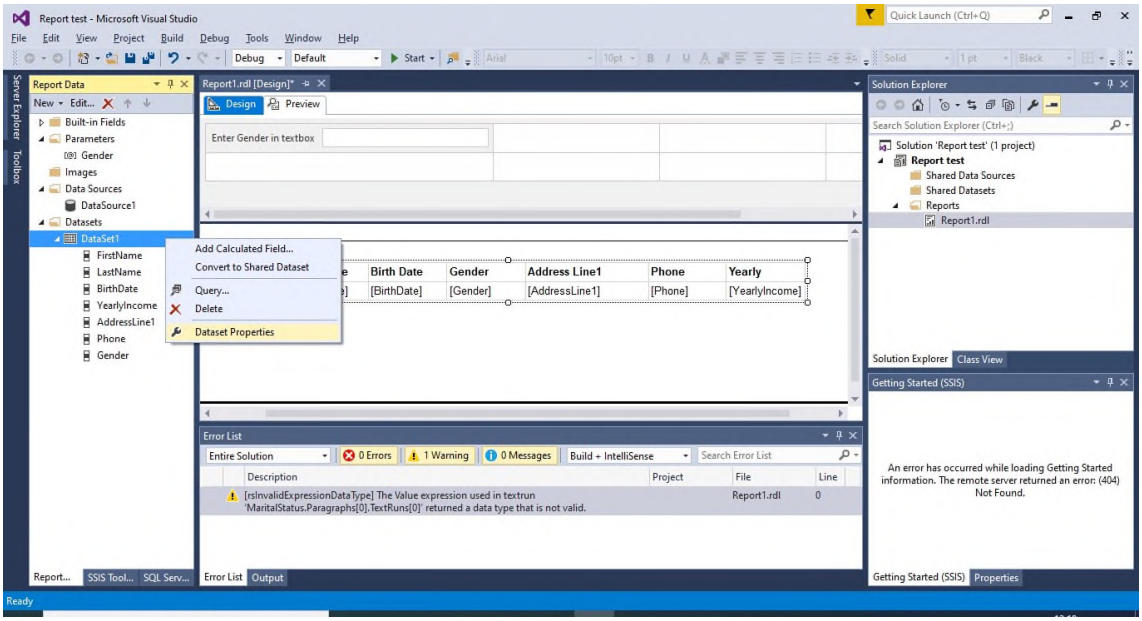
Once you click on **Add parameters..** option opens a new window called Report parameter Properties to configure the parameter properties in SSRS.

- **Name:** Please specify the valid Parameter name as per your requirement. Here, we defined it as Gender
- **Prompt:** The text you specified here will display as a label before the text box
- **Data Type:** Country name is a Text data type so, we kept the default text unchanged



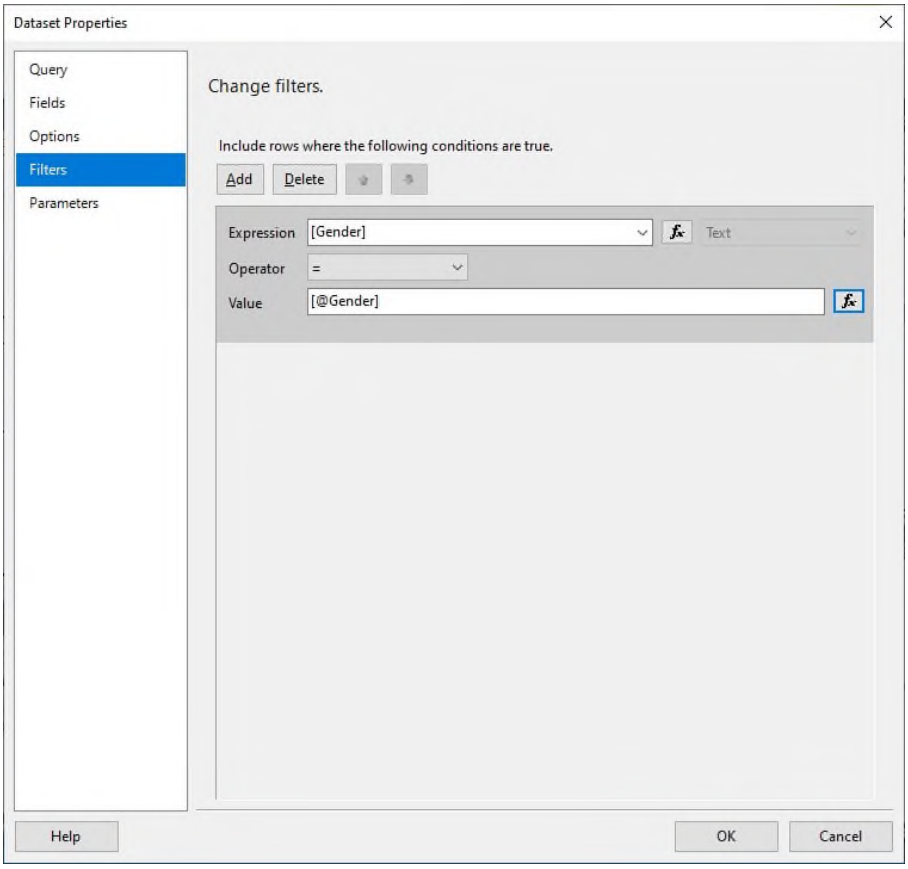
Click Ok to finish configuring the Report parameters in SSRS. Now, We have to apply filter condition on the Dataset

Select the Dataset from the Report data tab, and right-click on it will open the context menu. Please select the **Dataset Properties..** option from it.

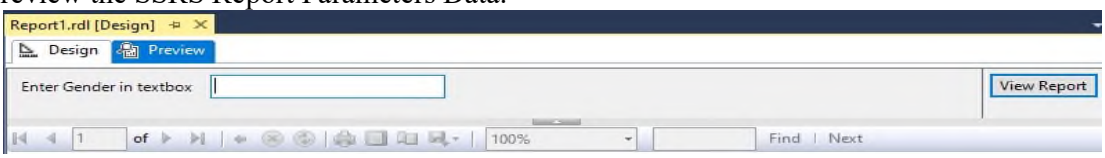


In this SSRS report parameters example, We are going to display the records whose [Gender] is equal to [Gender] we just created. So, Please select the Country as **Expression** and Operator as Like Operator and Value as Parameter name.

If you find any difficult to write the parameter name, Please click on the **fx** button and select the parameter name from the GUI.

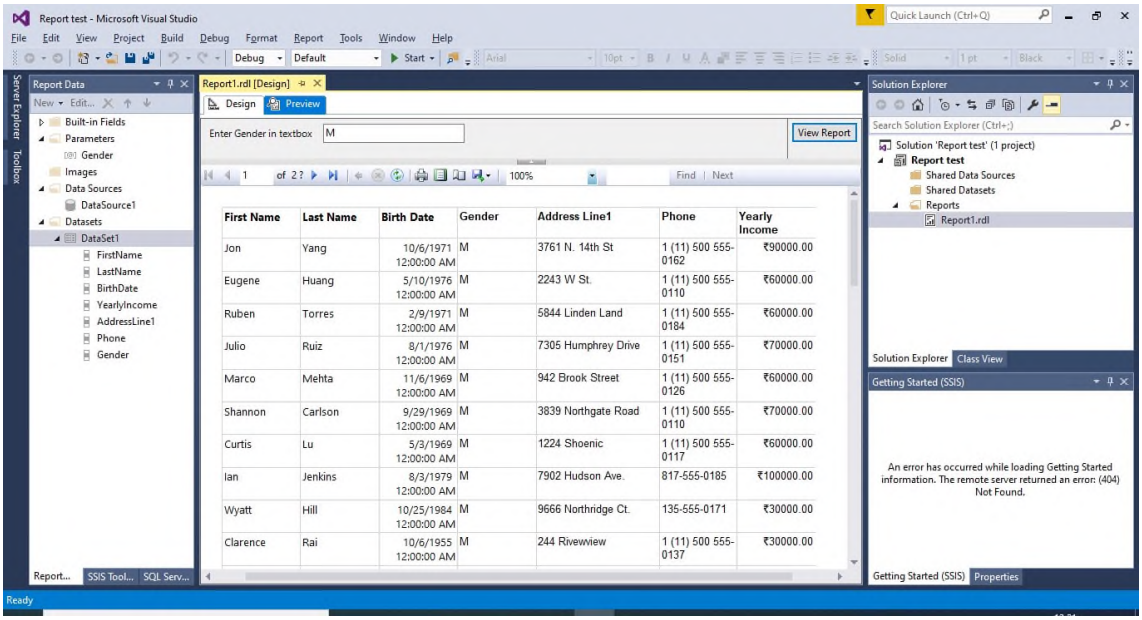


Click Ok to finish configuring the Filters at Dataset Level. Let's click on the Preview Tab to preview the SSRS Report Parameters Data.



From the above screenshot, you can observe that It is displaying a Blank Report with Empty Textbox on Top. To view the records, we have to enter the Gender in that Textbox and Press Enter.

For now, we are entering M as Gender.



First Name	Last Name	Birth Date	Gender	Address Line1	Phone	Yearly Income
Jon	Yang	10/6/1971 12:00:00 AM	M	3761 N. 14th St	1 (11) 500 555-0162	₹90000.00
Eugene	Huang	5/10/1976 12:00:00 AM	M	2243 W St.	1 (11) 500 555-0110	₹60000.00
Ruben	Torres	2/9/1971 12:00:00 AM	M	5844 Linden Land	1 (11) 500 555-0184	₹60000.00
Julio	Ruiz	8/1/1976 12:00:00 AM	M	7305 Humphrey Drive	1 (11) 500 555-0151	₹70000.00
Marco	Mehta	11/6/1969 12:00:00 AM	M	942 Brook Street	1 (11) 500 555-0126	₹60000.00
Shannon	Carlson	9/29/1969 12:00:00 AM	M	3839 Northgate Road	1 (11) 500 555-0110	₹70000.00
Curtis	Lu	5/3/1969 12:00:00 AM	M	1224 Shoenic	1 (11) 500 555-0117	₹60000.00
Ian	Jenkins	8/3/1979 12:00:00 AM	M	7902 Hudson Ave.	817-555-0185	₹100000.00
Wyatt	Hill	10/25/1984 12:00:00 AM	M	9666 Northridge Ct.	135-555-0171	₹30000.00
Clarence	Rai	10/6/1955 12:00:00 AM	M	244 Rivewview	1 (11) 500 555-0137	₹30000.00

Charts in SSRS

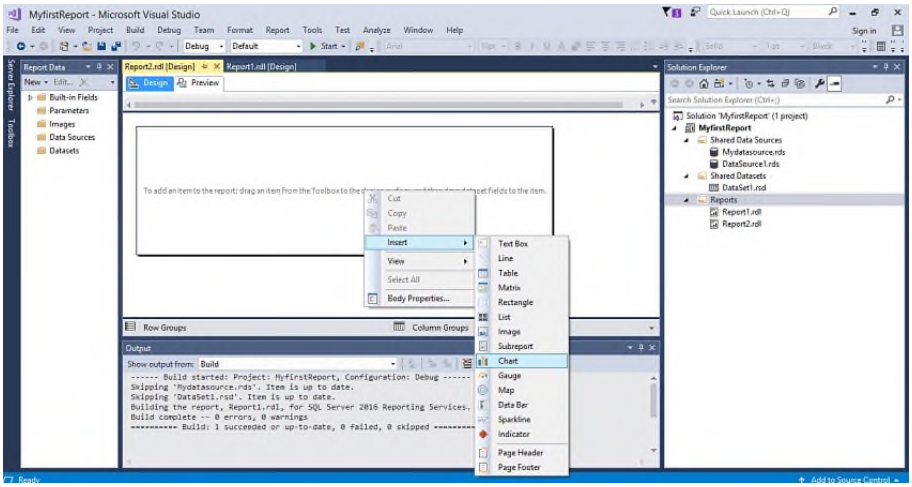
- **How to create charts**

  Step 1: I created a new table, i.e., employee having three fields, i.e., employee_id, employee_name, and gender in SQL Server. Now, we need to create a new data source in a similar way we created in other reports.

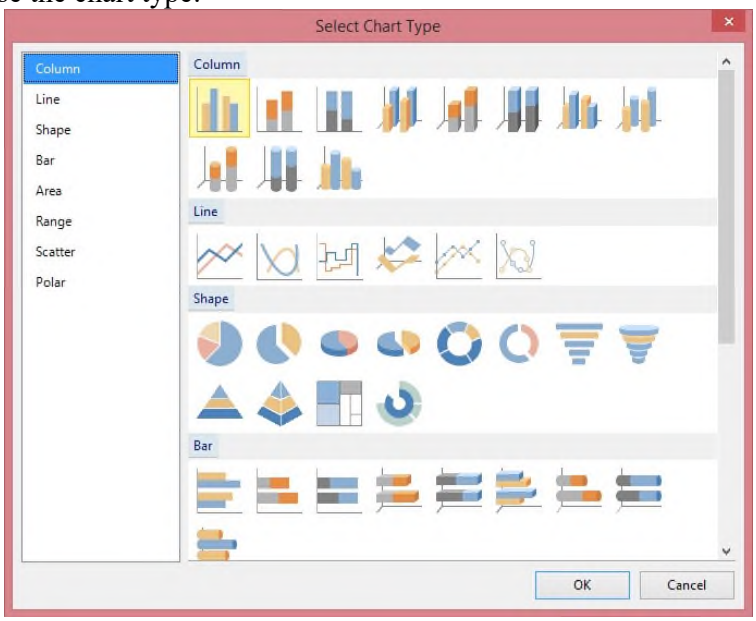
  Step 2: Create a new dataset.

  Step 3: Add a new item in Reports.

  Step 4: Right-click on the Report workspace, select the **insert** option and then click on the **Charts**.

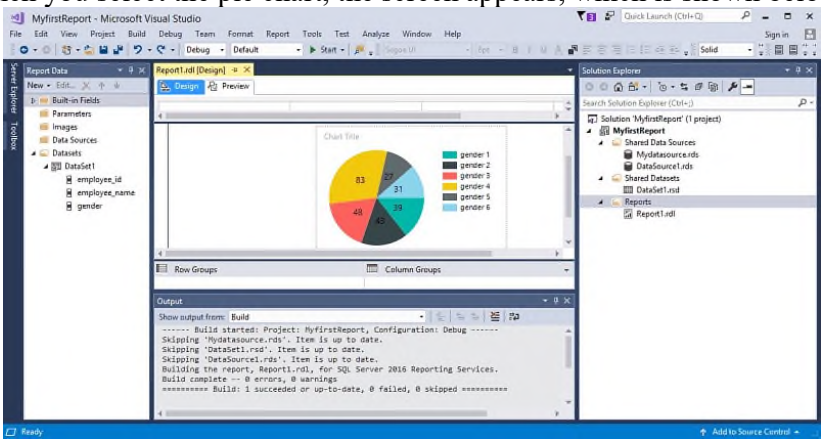


Step 5: Choose the chart type.

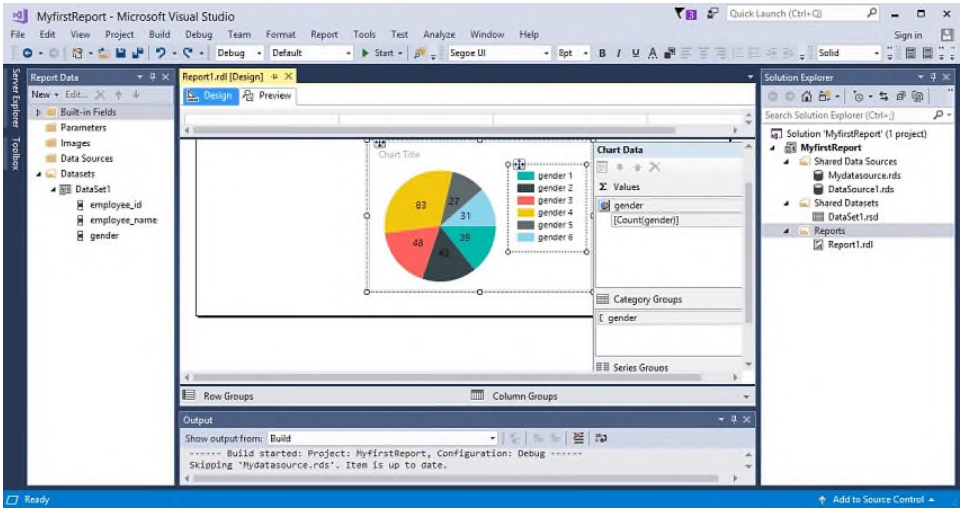


I selected the Pie chart.

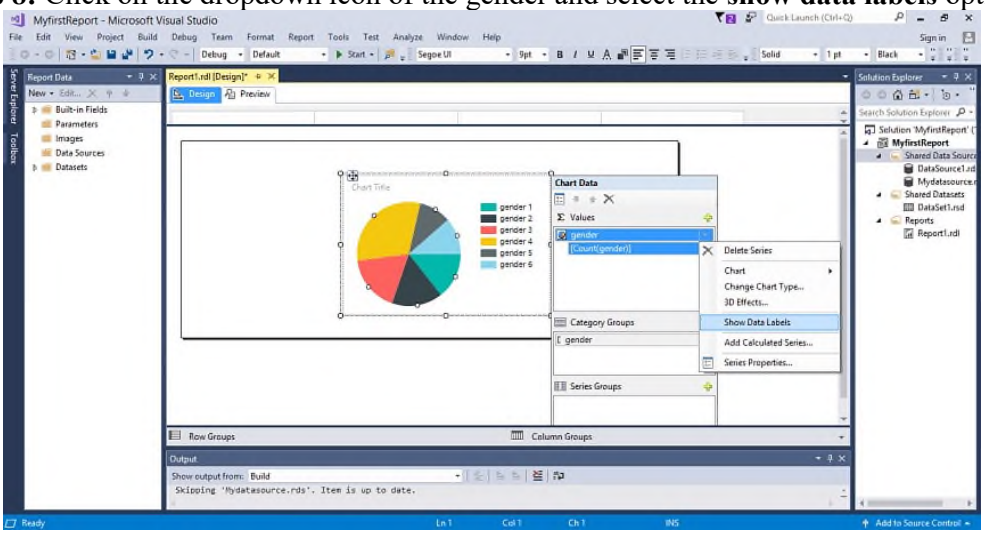
Step 6: When you select the pie chart, the screen appears, which is shown below:



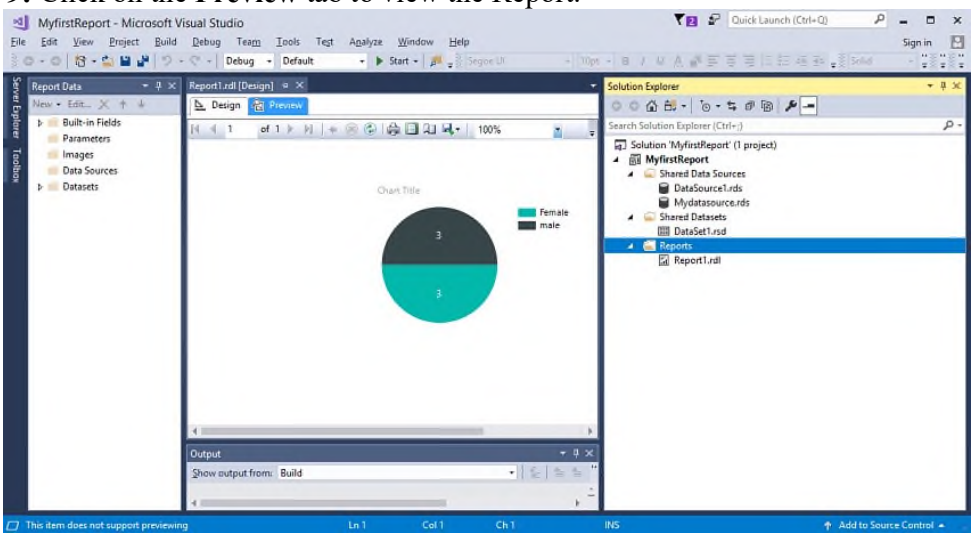
Step 7: Add the gender column to the categories group and values in the chart data.



Step 8: Click on the dropdown icon of the gender and select the **show data labels** option.



Step 9: Click on the **Preview** tab to view the Report.



In the above screen, the pie chart shows that 3 are males and 3 are females.

SQL Server Reporting Services Configuration Manager (For Deploy Project in Report Server)

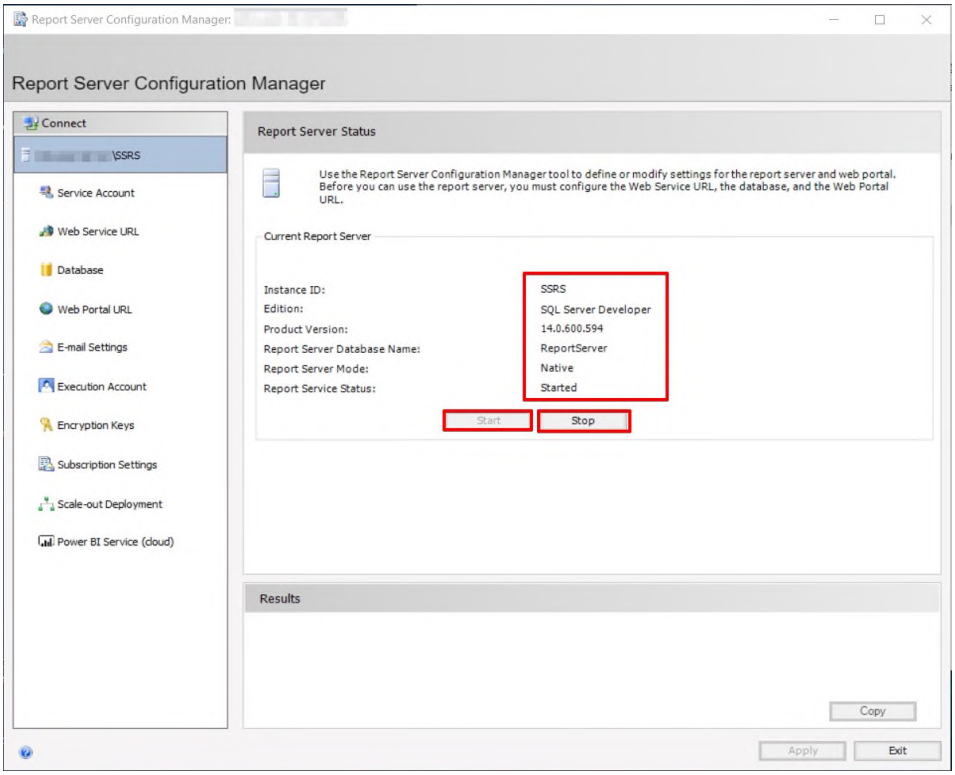
After the native installation of the SQL Server Reporting Service, we may need to customize some settings of Report Server. SQL Server Reporting Service provides a tool that is named as such; Report Server Configuration Manager. This tool helps us to customize SQL Server Reporting Service parameters and configurations. With the Report Server Configuration Manager, we can change and customize various parameters of the SQL Server Reporting Service. We can find the below settings in Report Server Configuration Manager:

- Service Account
- Web Service URL
- Database
- Web Portal URL
- E-Mail Settings
- Execution Account
- Encryption Keys
- Subscription Settings
- Scale-out Deployment
- Power BI Service (cloud)

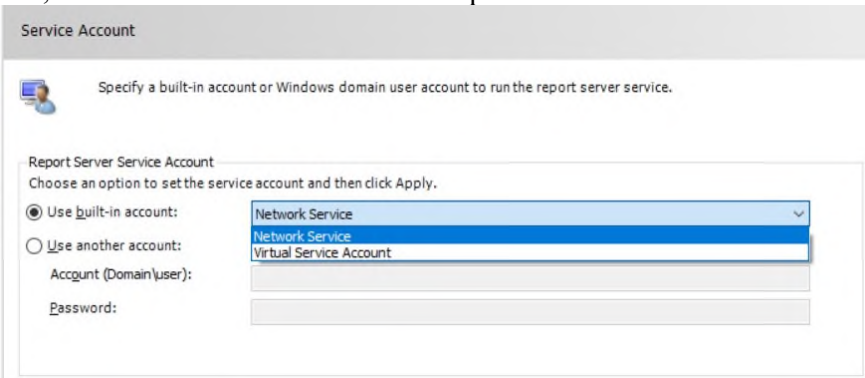
We can find and run the Report Server Configuration Manager on the Windows start screen.

- Type **Reporting** in Windows start screen
- Click Report Server Configuration Manager.

In the main tab of configuration manager, we can start and stop Report Server service and at the sametime we can find out Report Server edition, product version and other main information about SQL Server Reporting Service.

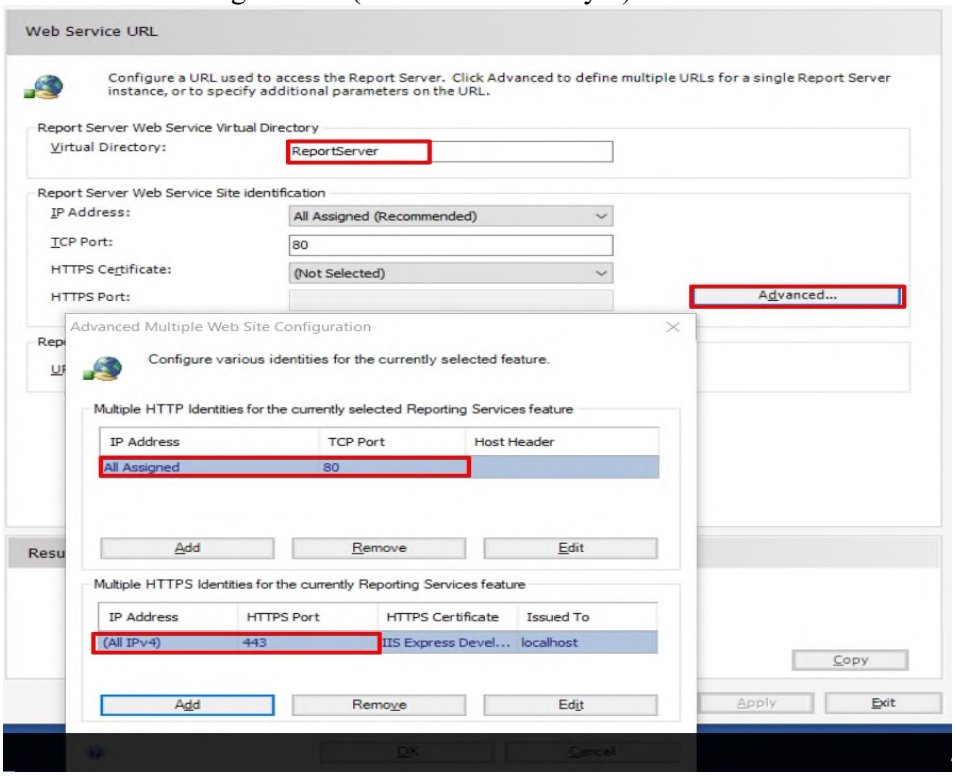


Service Account: In this tab, we can configure Report Server Windows service account. In the built-in account selection box; we can find Network Service and Virtual Service Accounts. These accounts have minimum privileges which they require to run Report Server and have network login permissions. Microsoft recommends that if we do not have any domain account we can use a Virtual Service Account. Because this account allows us to avoid password expiration or Report Server breakdowns, but if we have a domain account, we can create a dedicated account for Report Server, can customize required security policies to this account, and can use this account to run Report Server.

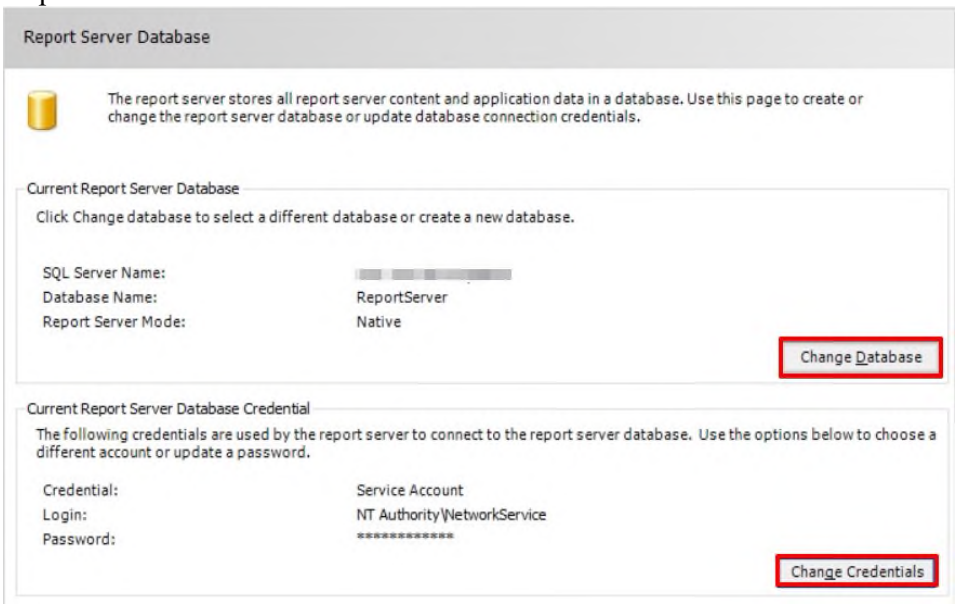


Web Service URL: In this setting tab, we can customize network policies, which have access to Report Server web URL. Therefore, we can control incoming request to Report Server. In this configuration tab, we can configure:

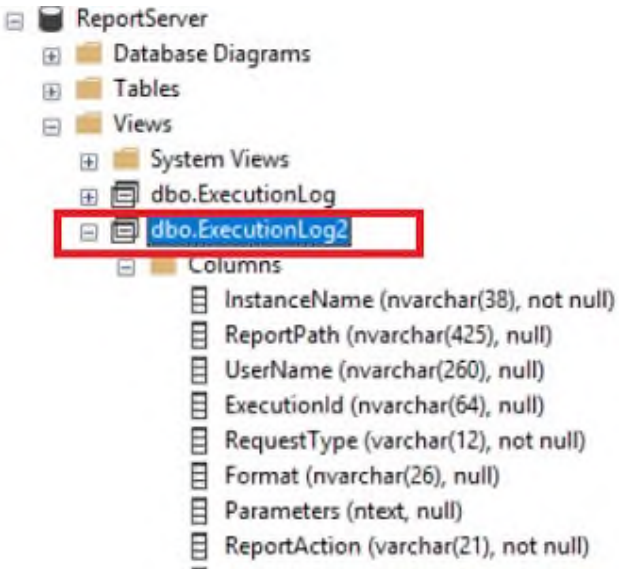
- Can filter the web service request
- Can change web service request ports for HTTP and https in **Advanced** tab
- Can change virtual directory name for the ReportServer. Default virtual directory name is **ReportServer**.
- Can bind and configure SSL (Secure Sockets Layer) certification.



Database: The Report Server database stores all report contents, subscriptions and other several data (Dataset, Data source, Roles etc.) which required by Report Server. In this tab, we can configure the Report Server database connection login and password setting. In the default Report Server installation, two databases are created which are named as **ReportServer** and **ReportServerTempDB.** At the same time, this tab provides us to change Report Server database.

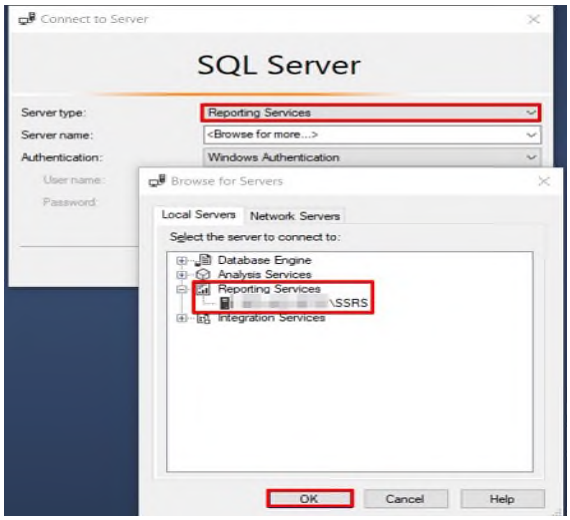


In Report Server database, we can find a handy view, which is named as ExecutionLog2.

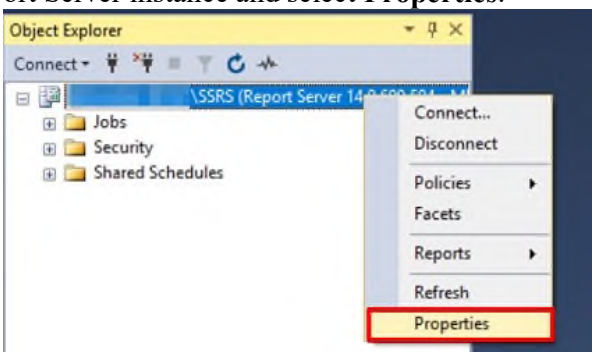


This view helps us to find out the usage frequency of reports, execution times and what parameters are used in the reports. By default, this log is kept for 60 days. We can change this setting to connect the SSRS instance with the management studio.

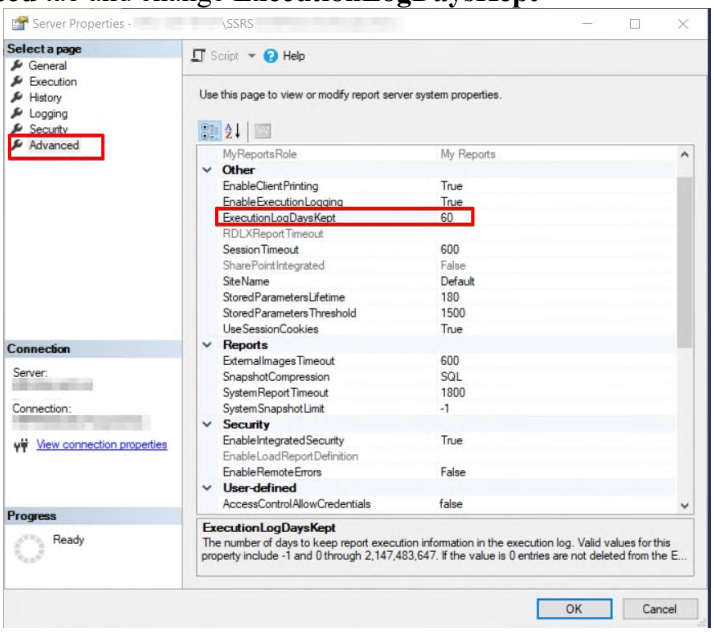
- Open Microsoft SQL Server Management Studio and run as “Administrator”
- Select **Server type** as **Reporting Services** and click **OK**



- Right click on Report Server instance and select **Properties**.

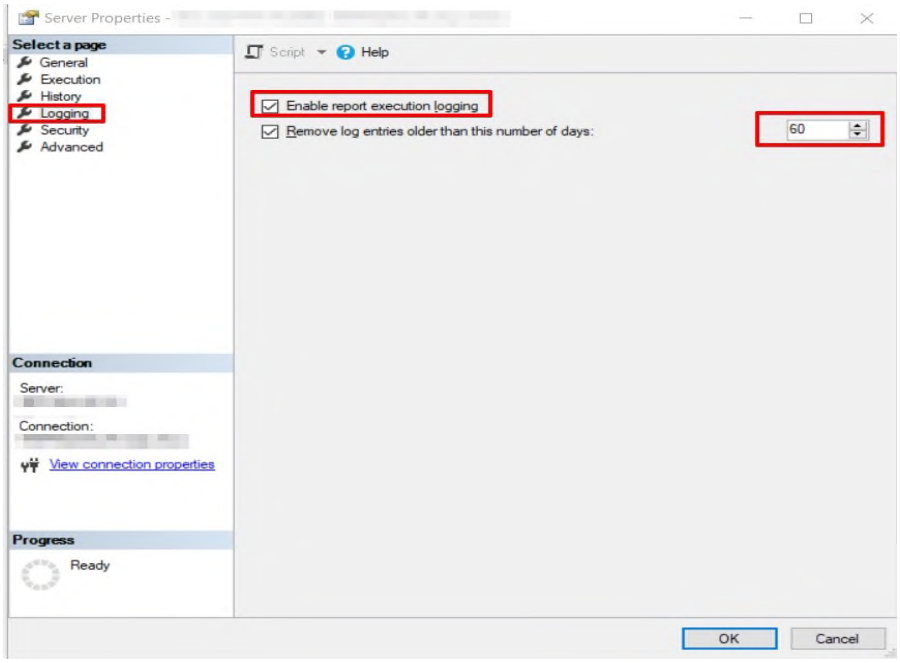


- Click **Advanced** tab and change **ExecutionLogDaysKept**

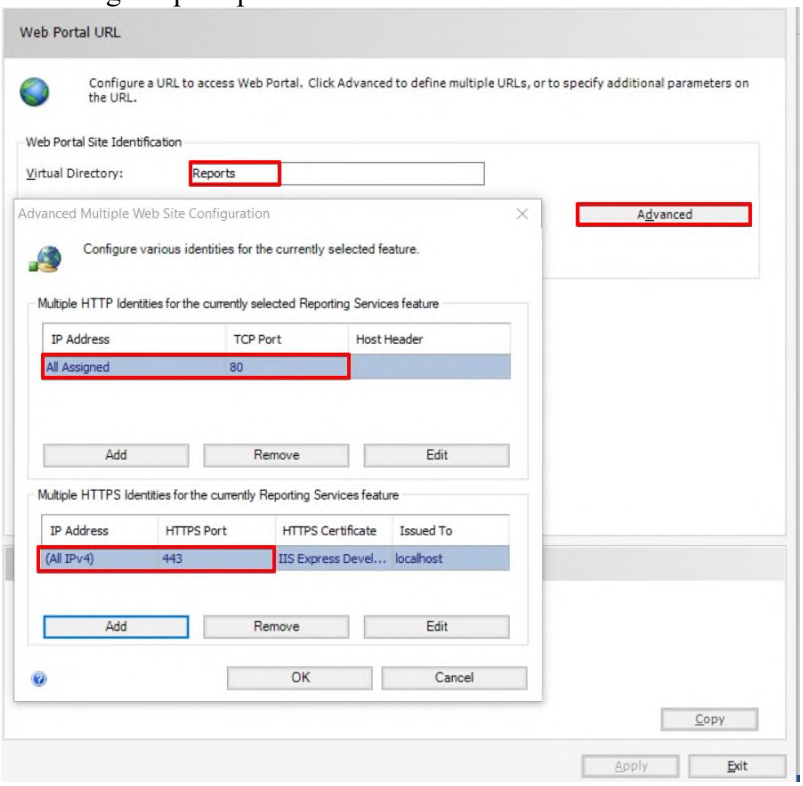


The other method is:

- Click **Logging** tab and change default number. In this tab execution, logging option can disable.

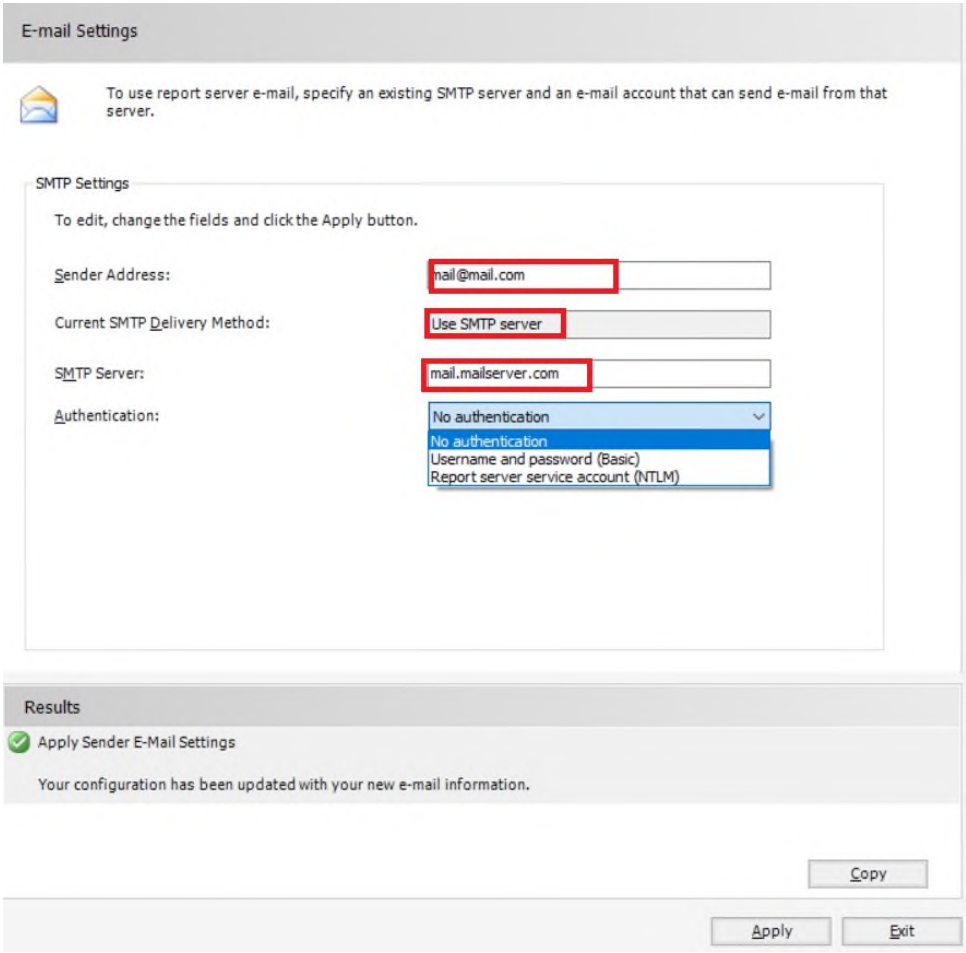


Web Portal URL: In this configuration tab, we can configure web portal virtual directory name and we can change request ports and filter IP address like Web Service URL.



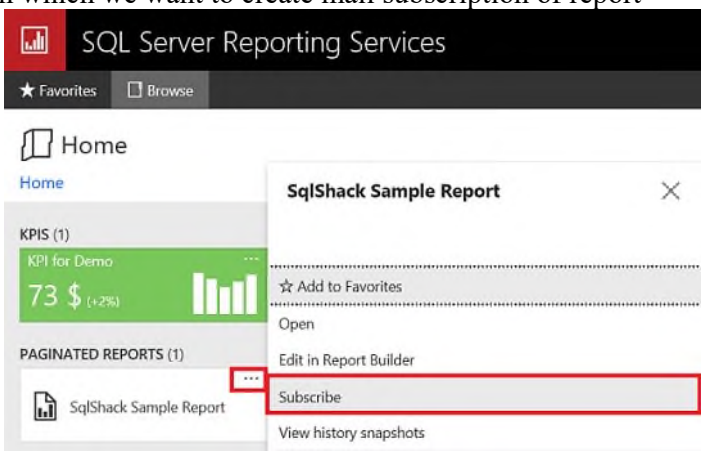
E-Mail Settings: In this configuration tab, we can configure e-mail account for mail subscriptions. Report Server allows us three type of authentication methods:

- No authentication
- Username and password for SMTP
- Report server service account NTLM

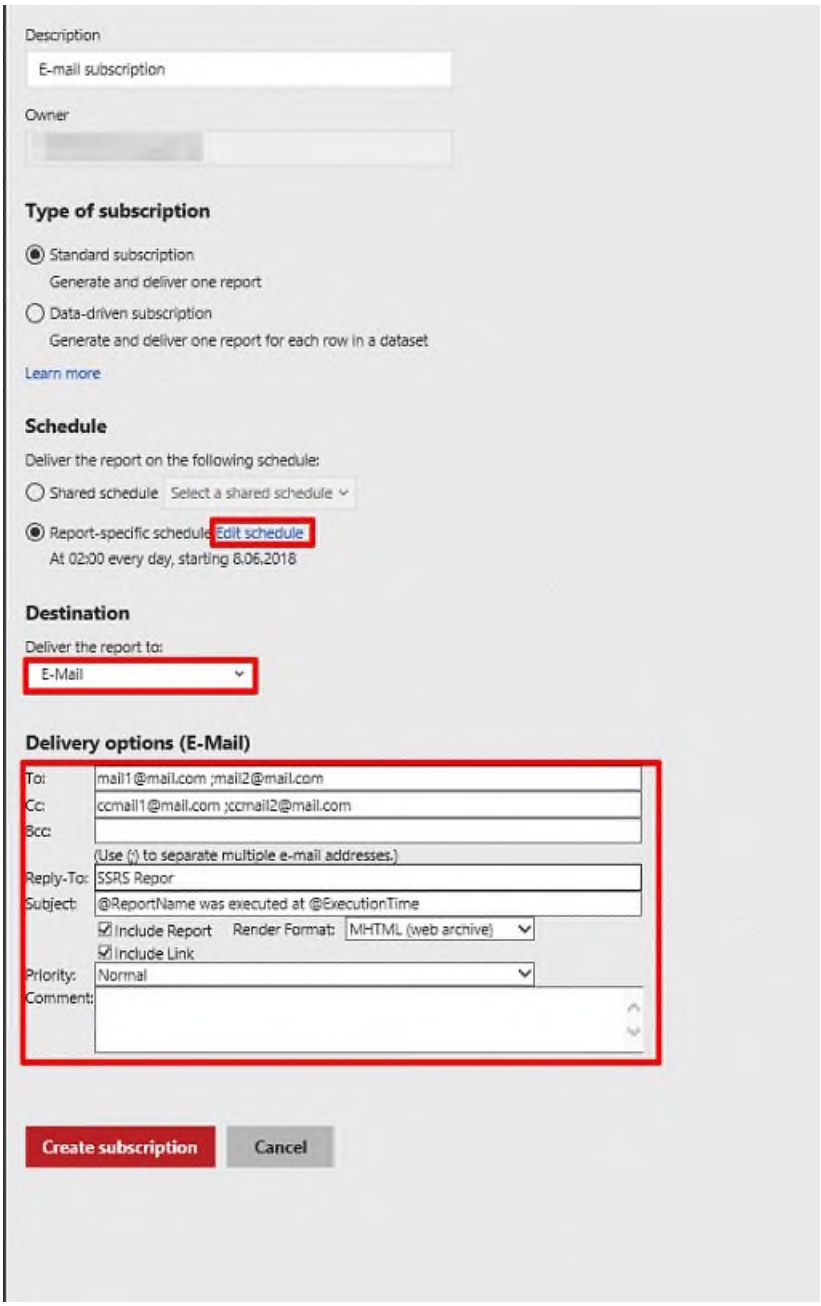


Now, we will look at how to create an e-mail subscription on SQL Server Reporting Service

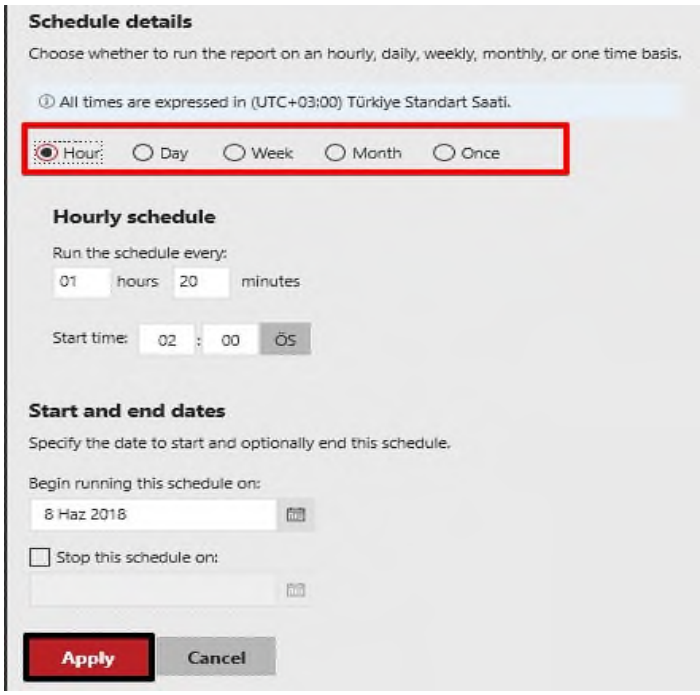
- Open report server web portal
- Click **(…)** icon which we want to create mail subscription of report



- Chose to **Deliver the report to** as **E-Mail**
- Fill **Delivery options (E-Mail)**

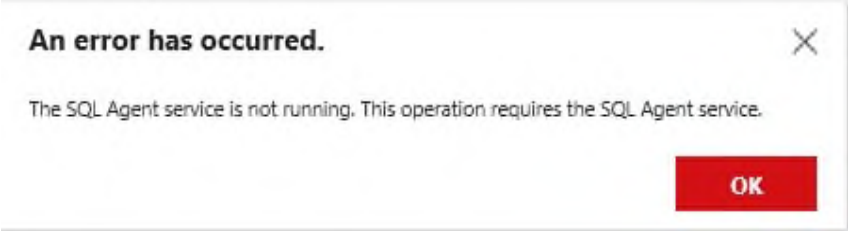


- Click **Edit schedule** and create required subscription schedule
- Click **Apply**

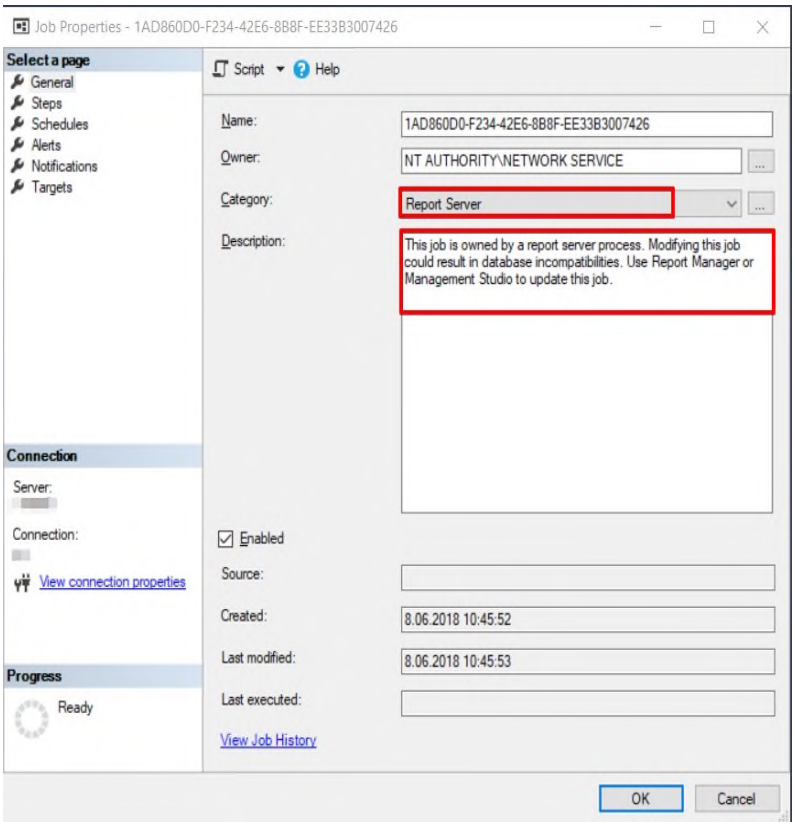


- Click **Create Subscription**

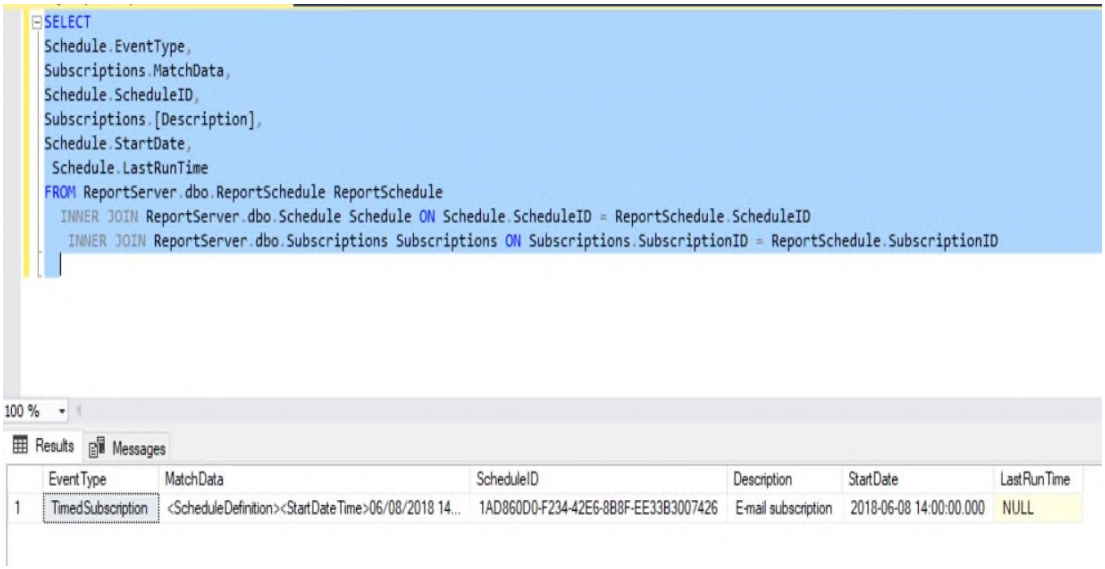
After completing all these steps about email subscriptions, Report Server will create a job on SQL Agent. If our SQL Agent does not run, we get an error about this.



When we look at the SQL Agent jobs, we can find out a job, which represents Report Server email subscription.



At the same time, we find some information about this job in Report Server database, the below query returns the following information about subscription.



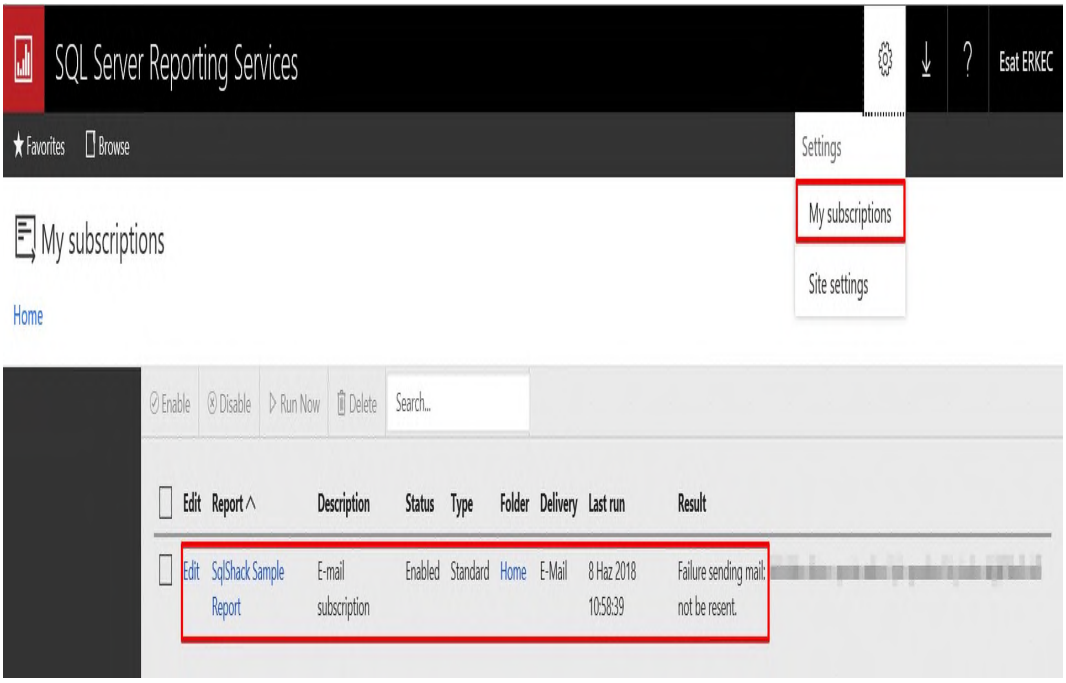
```
SELECT
Schedule.EventType,
Subscriptions.MatchData,
Schedule.ScheduleID,
Subscriptions.[Description],
Schedule.StartDate,
 Schedule.LastRunTime
FROM ReportServer.dbo.ReportSchedule ReportSchedule
  INNER JOIN ReportServer.dbo.Schedule Schedule ON Schedule.ScheduleID = ReportSchedule.ScheduleID
   INNER JOIN ReportServer.dbo.Subscriptions Subscriptions ON Subscriptions.SubscriptionID = ReportSchedule.SubscriptionID
```

100 %

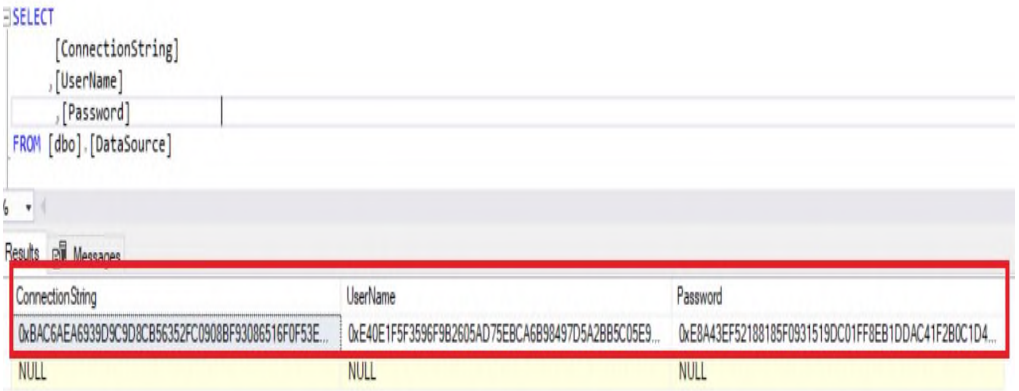
Results | Messages

	EventType	MatchData	ScheduleID	Description	StartDate	LastRunTime
1	TimedSubscription	<ScheduleDefinition><StartDateTime>06/08/2018 14...	1AD860D0-F234-42E6-8B8F-EE33B3007426	E-mail subscription	2018-06-08 14:00:00.000	NULL

At last, you can find logs of subscriptions in **My Subscription** page.



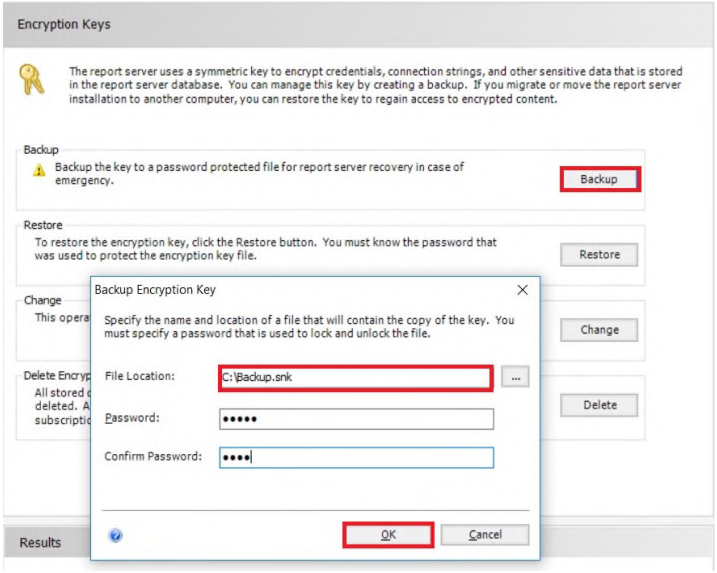
Encryption Keys: In this configuration page, we can find out encryptions keys configurations. Encryption keys are most important for Report Server stored connection strings and other sensitive data in Report Server database and this data is encrypted by these symmetric keys.



For this reason, our first task after the installation of the Report Server should be the backup of this encryption key. Because when we need to move or change the location of the Report Server location. This encryption key provides us to recreate connection strings username and password successfully. If we delete the encryption key Report Server it will clear all connection string usernames, passwords, and other sensitive data. We will need to redefine this data.

Backup encryption key

- Click **Backup**
- Define file location for encryption key
- Set password for encryption key
- Click **OK**



Subscription Settings: We mentioned about file subscriptions under the email set heading. If we do not set this setting, the Report Server tries to access file paths with their service account privileges. However, in some special security policies service accounts cannot access defined file subscription keys. To handle this issue, we can use file share account. This setting allows us to use different accounts to access file paths.

Scale-out Deployment: In this model, we can run multiple Report Server and can connect one Report Server Database. The benefit of Scale-out Deployment;

- Increase high availability of Report Server
- Handle more concurrent users at the same time without performance issue
- Network load balancing can configure

Introduction to Data Analysis

- **Data analysis** and **data mining** are a subset of **business intelligence** (BI), which also incorporates **data warehousing**, **database** management systems, and Online Analytical Processing (OLAP). The technologies are frequently used in customer relationship management (CRM) to **analyze** patterns and query customer databases.
- Data Analytics is often used for processing data, whether from a single or multiple sources, using statistical and mathematical tools in order to generate insights. Both data warehousing and data analytics can be seen as parts or stages of Business Intelligence, although BI and DA are often used interchangeably.
- A career in data warehousing becomes more promising when you have a degree in Data Analytics. Your knowledge of both the worlds (of data analytics, which is related to business intelligence) and data warehousing (related to data management) sets you apart.

What is SSAS?

- **SQL Server Analysis services (SSAS)** is a multi-dimensional OLAP server as well as an analytics engine that allows you to slice and dice large volumes of data. It is part of Microsoft SQL Server and helps perform analysis using various dimensions. It has two variants Multidimensional and Tabular. The SSAS full form is SQL Server Analysis Services.
- **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.

- **Important SSAS Terminology**

Data Source:	Data Source is a kind of connection string. It establishes a connection between the analysis database and RDBMS.
Data Source View:	Data source view is a logical model of Database
Cube:	A cube is a basic unit of storage. It is a collection of data which has been aggregated to allow queries to return data quickly.
MOLAP:	The MOLAP is made of data cube which contains of measures and dimensions. It includes all the members who may be in a hierarchical relationship. It is specific set of rule, which helps you to determine how certain cells are computed in a sparse cube, and measures values rolled up inside those hierarchies.
Dimension Table	• A dimension table contains dimensions of a fact. • They are joined to fact table using a foreign key. • Dimension tables are de-normalized tables. • Dimensions offer characteristics of the facts with the help of their attributes. • Not offers limit set for a given number of dimensions • The dimension contains one or more hierarchical relationships.
Dimension:	Dimension offers the context surrounding a business process event. In simple terms, they give who, what, where of a fact. In the Sales business process, for the fact sales number, dimensions would be who customer Names. • Where – Location • What – Product Name • In other words, you can say that a dimension is a window to view the information in the facts.
Level:	Each type of summary, which can be retrieved from the single dimension, is called label.
Fact Table:	A fact table is the most important table in a dimensional model. A Fact Table contains Measurements/fact and Foreign key to the dimension table. For example, payroll operations.
Measure:	Each fact table contains one or more table, which should be analyzed. For example, a book sells information table. It can be a profit or loss for the number of book's sold.
Schema:	The database schema of a database system and its structure described in a formal language. It supports the database management system. The term "schema" means to the organization of data as a blueprint of a manner of which database is constructed.

Create Cube and Dimensions Using Data Analysis

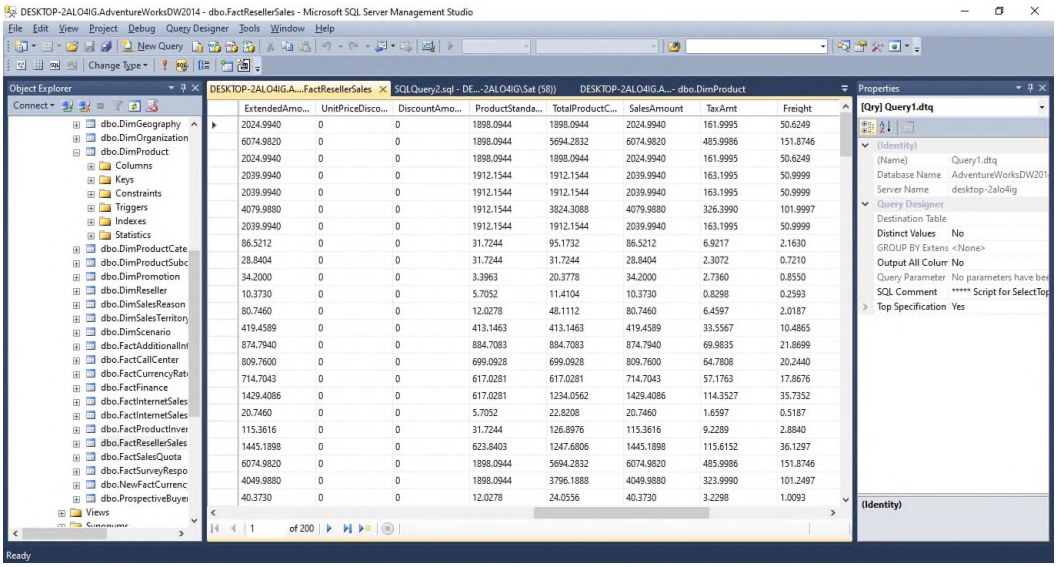
- SQL Server Analysis Services (SSAS) is the technology from the Microsoft Business Intelligence stack, to develop Online Analytical Processing (OLAP) solutions. In simple terms, you can use SSAS to create cubes using data from data marts / data warehouse for deeper and faster data analysis.
- Cubes are multi-dimensional data sources, which have dimensions and facts (also known as measures) as its basic constituents. From a relational perspective, dimensions can be thought of as master tables and facts can be thought of as measureable details.
- These details are generally stored in a pre-aggregated proprietary format and users can analyze huge amounts of data and slice this data by dimensions very easily. Multi-dimensional expression (MDX) is the query language used to query a cube, similar to the way T-SQL is used to query a table in SQL Server.
- Simple examples of dimensions can be product / geography / time / customer, and similar simple examples of facts can be orders / sales. A typical analysis could be to analyze sales in Asia-pacific geography during the past 5 years. You can think of this data as a pivot table where geography is the column-axis and years is the row axis, and sales can be seen as the values. Geography can also have its own hierarchy like Country->City->State. Time can also have its own hierarchy like Year->Semester->Quarter. Sales could then be analyzed using any of these hierarchies for effective data analysis.
- We will learn how to create your first cube from scratch.

Requirements

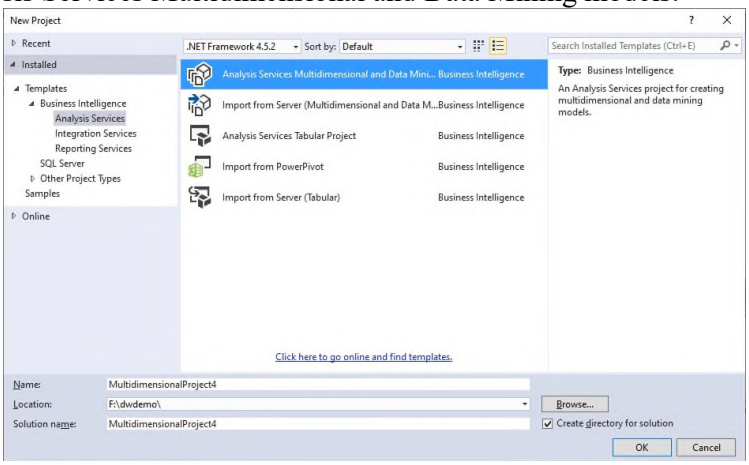
1. SSAS multidimensional database installed
2. SSDT
3. The AdventureworksDW2014 database
4. SSMS 2014 installed

Getting started

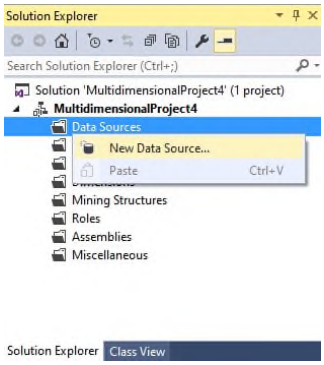
- A cube in SSAS requires a data warehouse. The data warehouse is a special database designed to store enterprise information from different sources like Excel, ERPs, CRMs, flat files, legacy data and more.
- If you want to have an SSAS cube, creating the data warehouse is a process that takes a bit of time and resources. You can use SSIS to extract the information from sources to the data warehouse or use other ETL (Extract Transformation and load) tools like Talend Open Studio, SpagoBI Business Intelligence, Jaspersoft ETL and several other external tools.
- We will use the AdventureworksDW2014 sample ready to be used. However, in the real-life, you will need to create some ETLs to automatically extract your business information to the Data Warehouse.
- We will create a cube based on the fact table FactResellerSales:



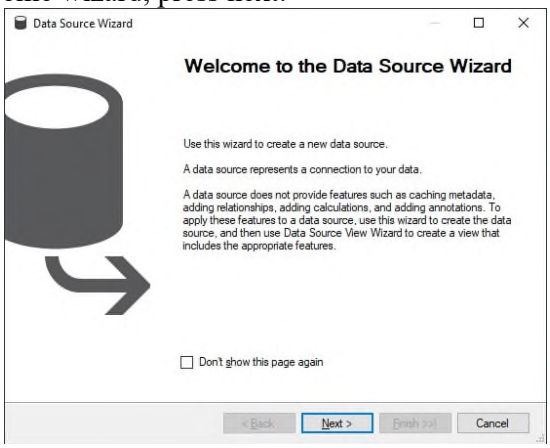
- FactResellerSales shows information about resellers' sales like discount amounts, tax amounts, freight, sales amount and other related information.
- We will create reports about the fact information combined with 2 dimensions:
  - currency
  - sales territory
- The **Currency dimension** will help us to detect amounts in different currencies around the world and **Sales territory** will help to detect sales, discounts, etc. per region.
- In order to start get started in SQL Server Data Tools aka SSDT, open a new project and select Analysis Services Multidimensional and Data Mining models:



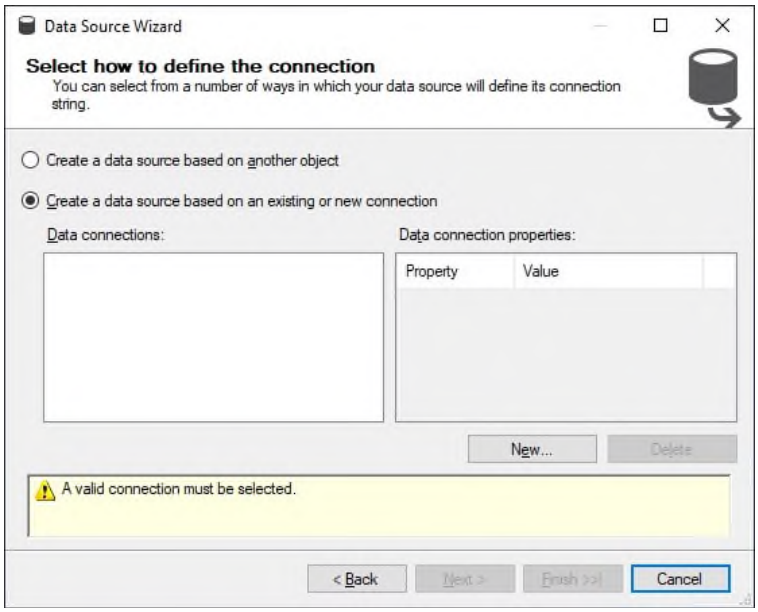
In the Solution Explorer, right-click Data Sources and select New Data Source:



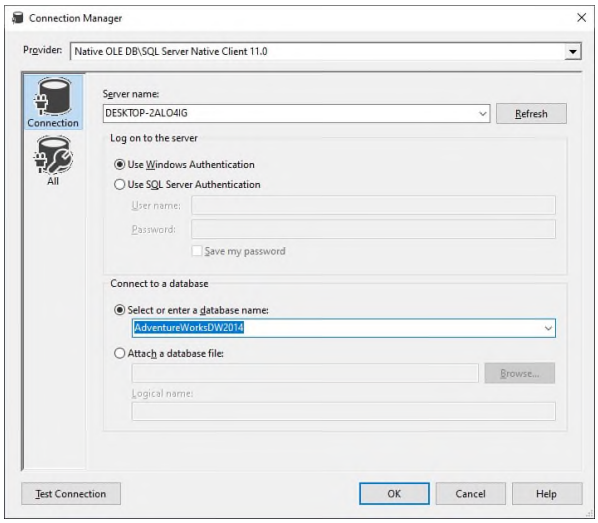
You will receive a Welcome wizard, press next:



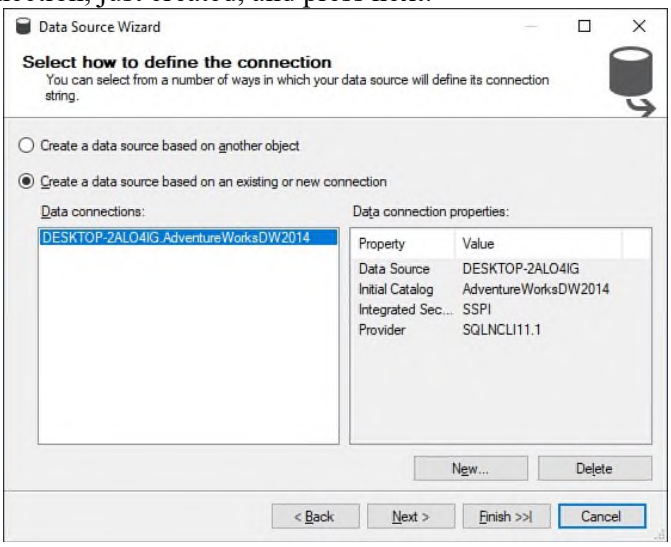
Select the option to create a new Data Source based on existing or new connection and press the new button:



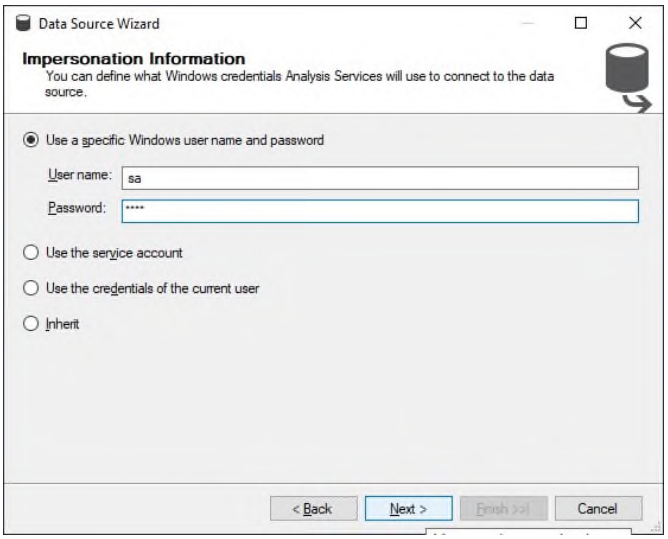
In the **Provider**, select a Native OLEDB\SQL Server Native Client 11.0. For **Server name**, select the SQL Server with the AdventureworksDW2014 database:



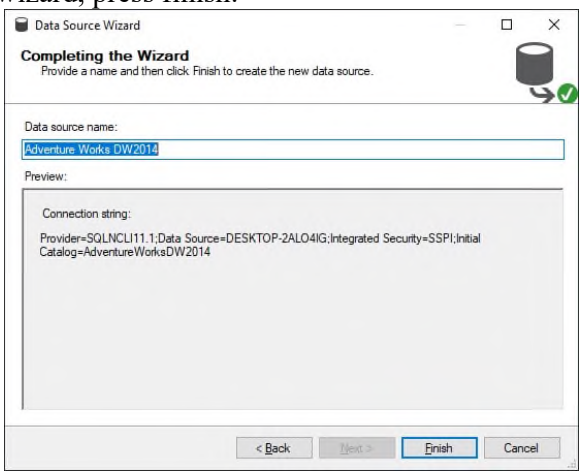
Select the data connection, just created, and press next:



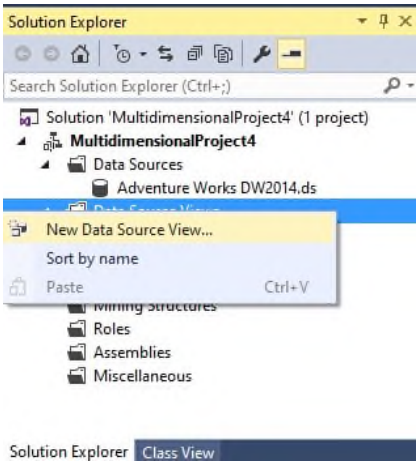
In **Impersonation information,** specify a user with access to the AdventureworksDW2014 database:



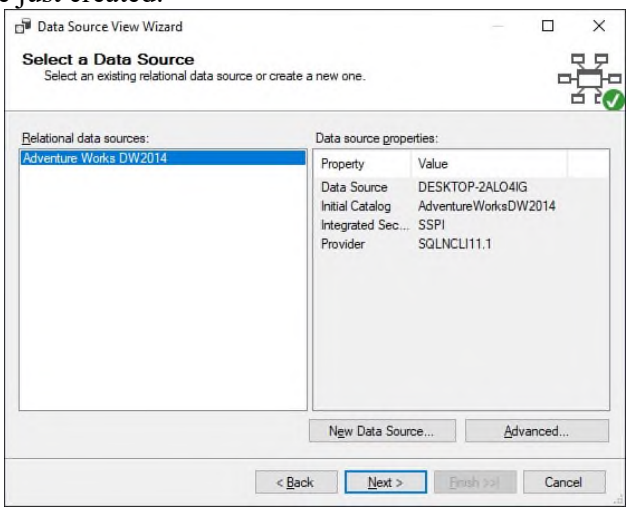
In the Completing the wizard, press finish:



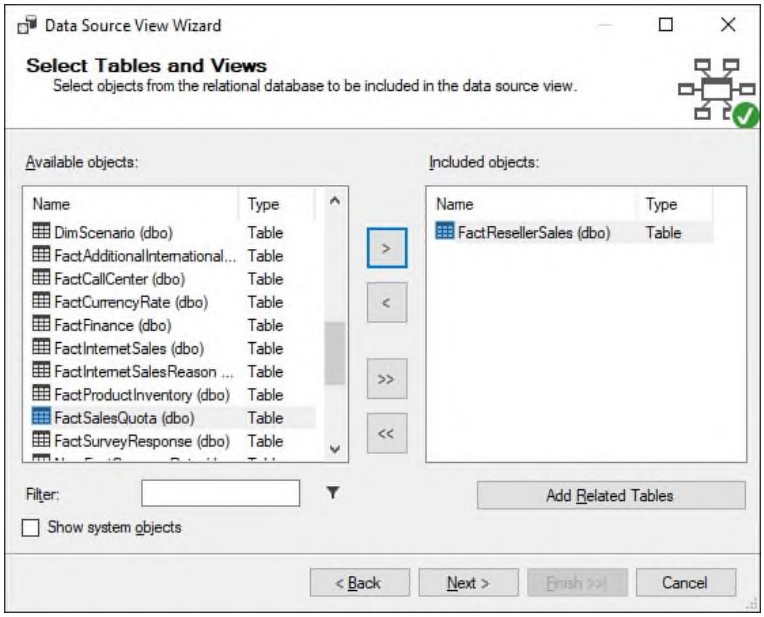
We created the credentials to connect to the AdventureworksDW2014 database. We will create now the views. In this process, we will select the tables to be included in the cube. In solution explorer, select New Data Source View:



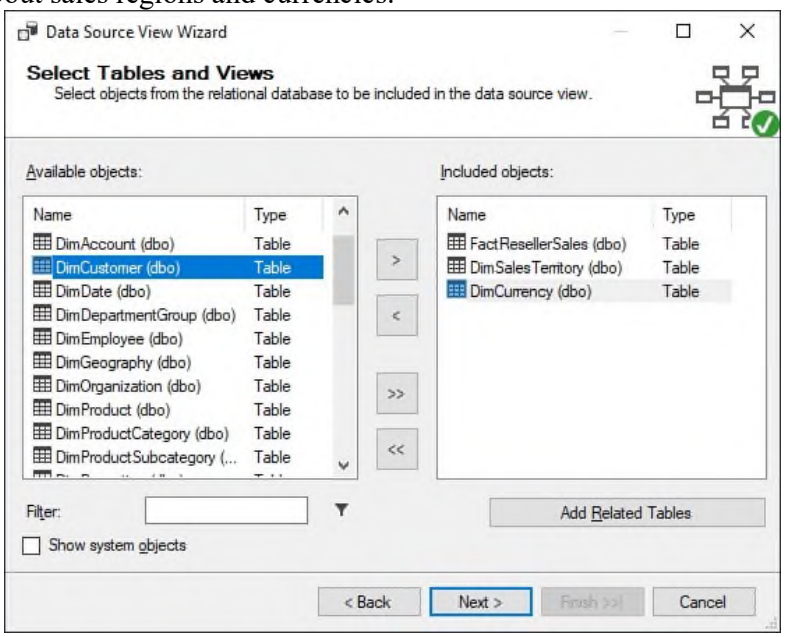
Select the data source just created:



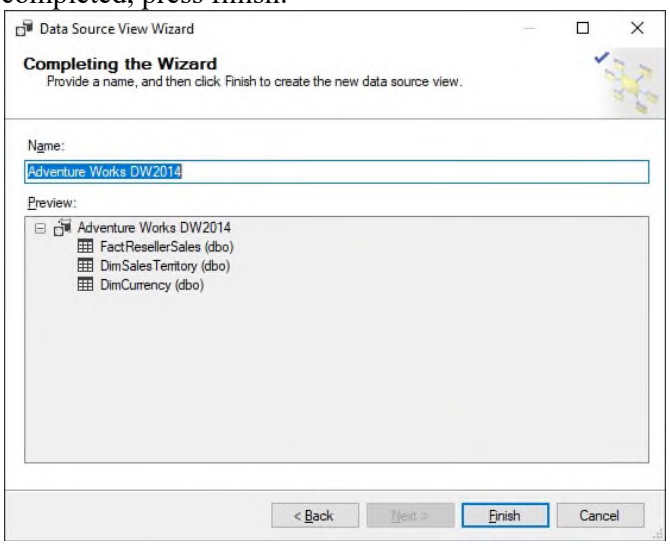
Select the table FactResellerSales:



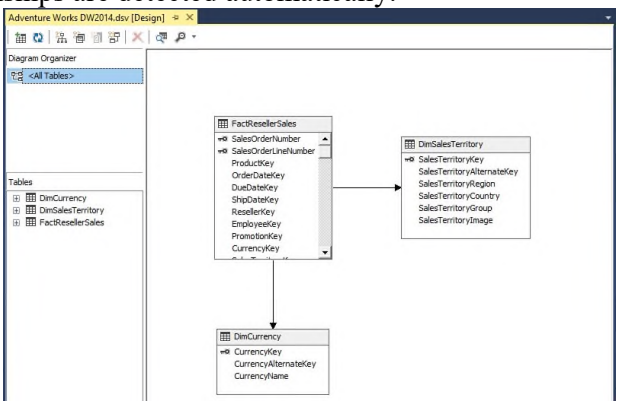
Include the dimensions DimSalesTerritory and DimCurrency. These dimensions include information about sales regions and currencies:



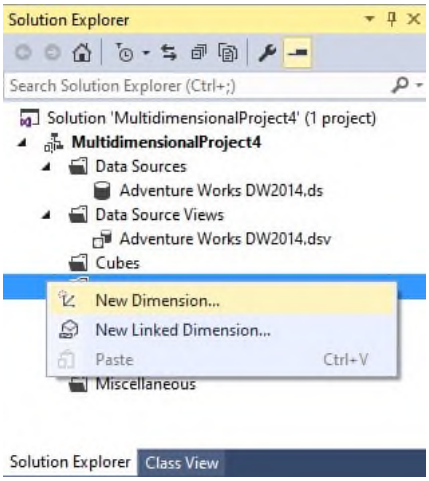
Once the wizard is completed, press finish:



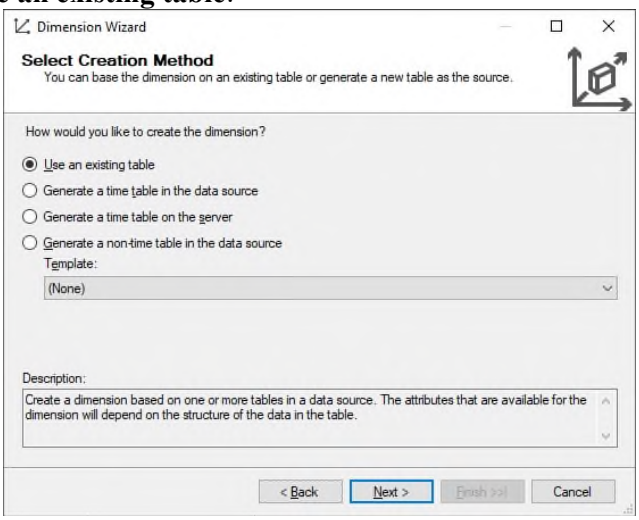
As you can see, the fact table and the dimensions have a relationship. If the common columns are equal, the relationships are detected automatically:



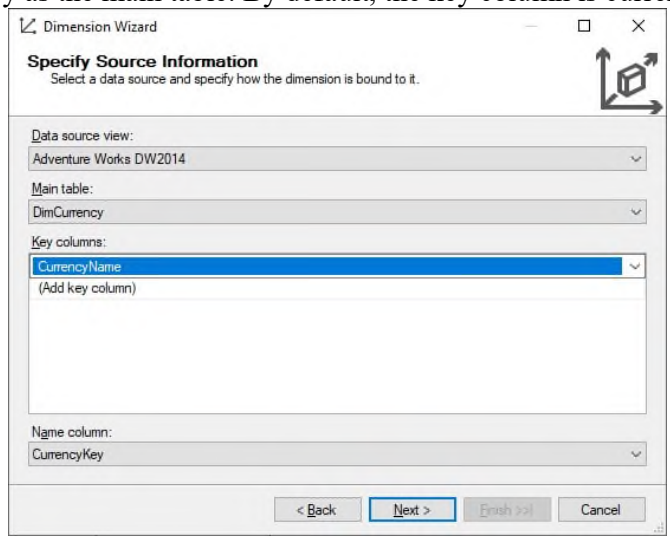
To create a cube, we will add dimensions to the cube.
In the Solution Explorer, right click and select New Dimension:



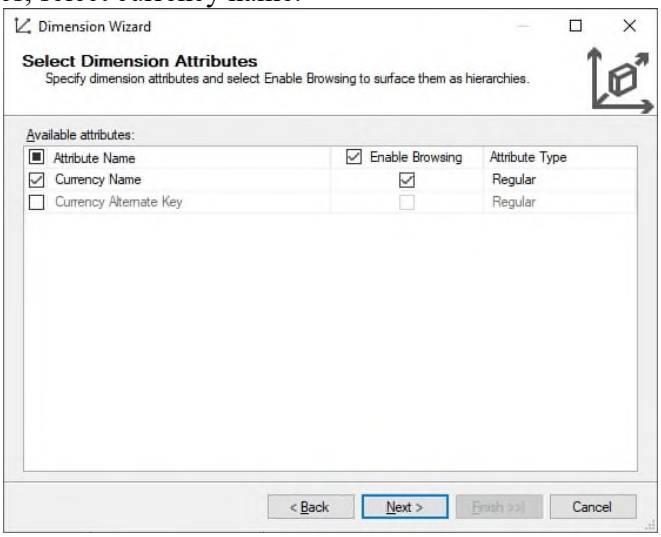
You can create a dimension based on templates, time tables or based on existing tables.
Select the option **Use an existing table**:



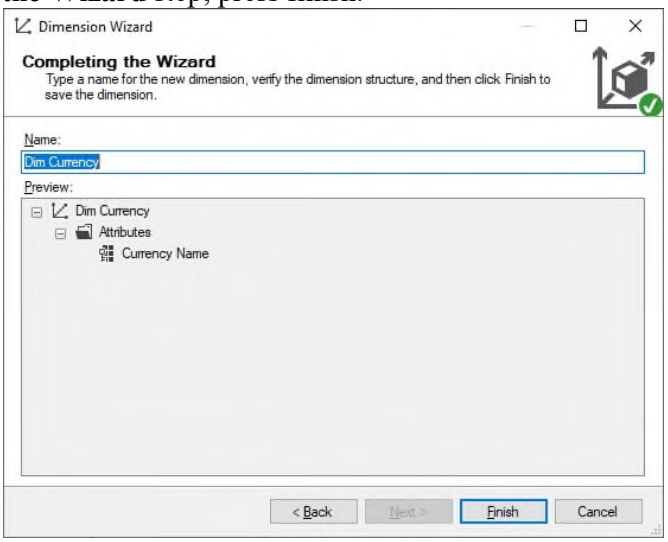
Select DimCurrency as the main table. By default, the key column is currency key:



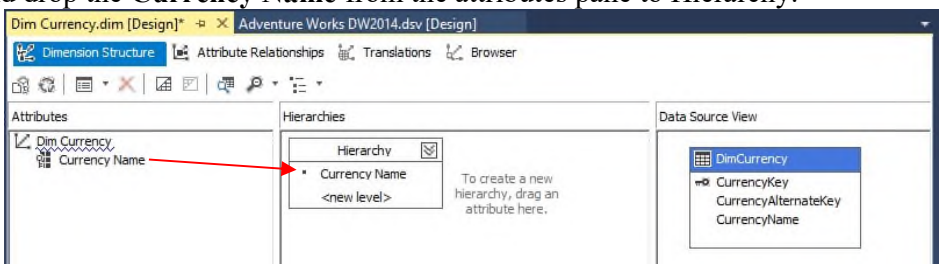
In available attributes, select currency name:



In the **Completing the Wizard** step, press finish:

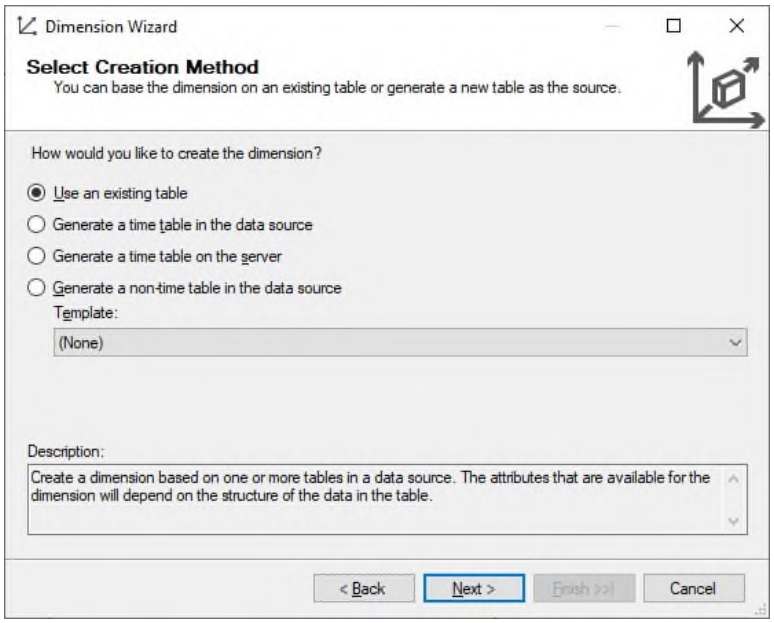


Now, you will have a dimension created like this one.
Drag and drop the **Currency Name** from the attributes pane to Hierarchy:

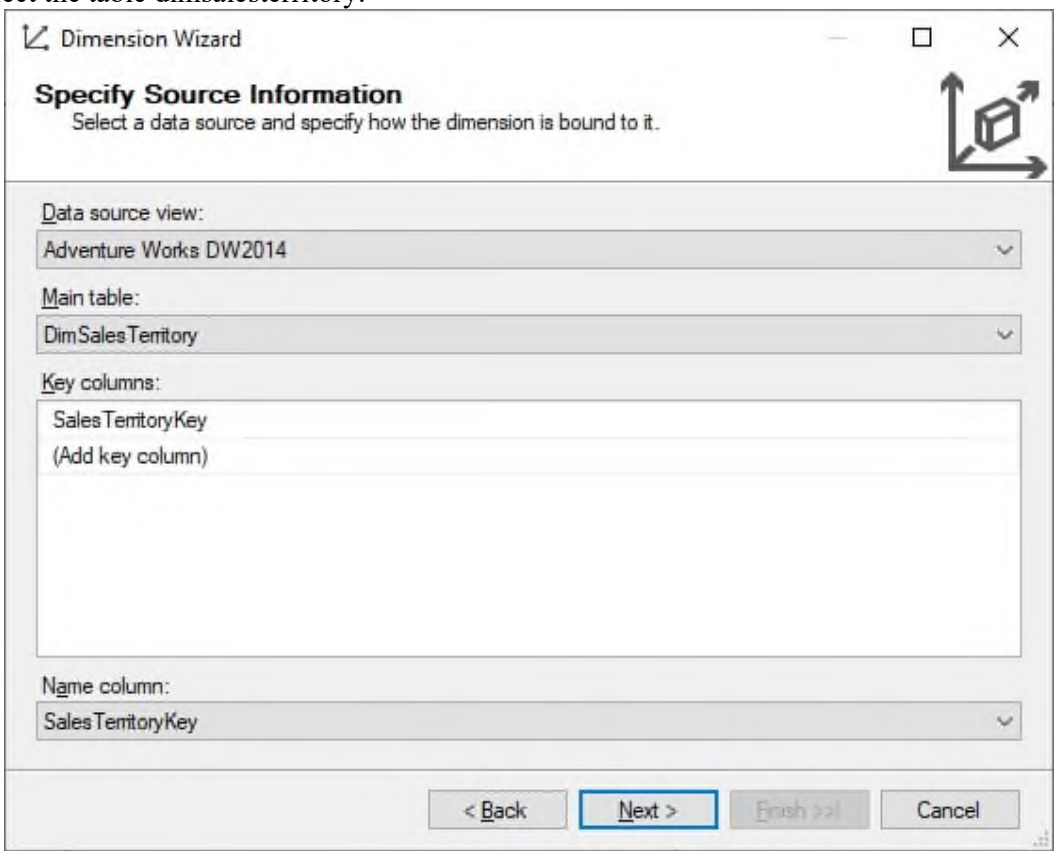


Once that you have the first dimension created, **Currency name**, we will create the sales territory dimension. In solution explorer, right click Dimensions and select New dimension:

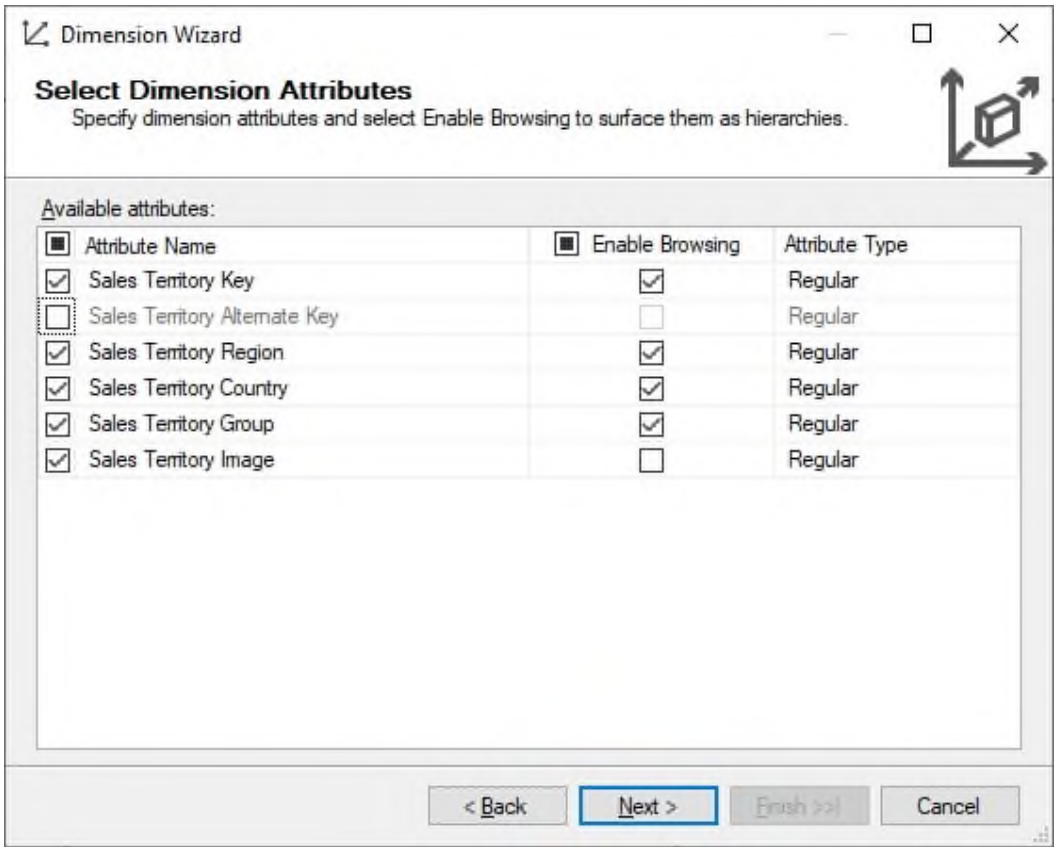
Select the option use an existing table:



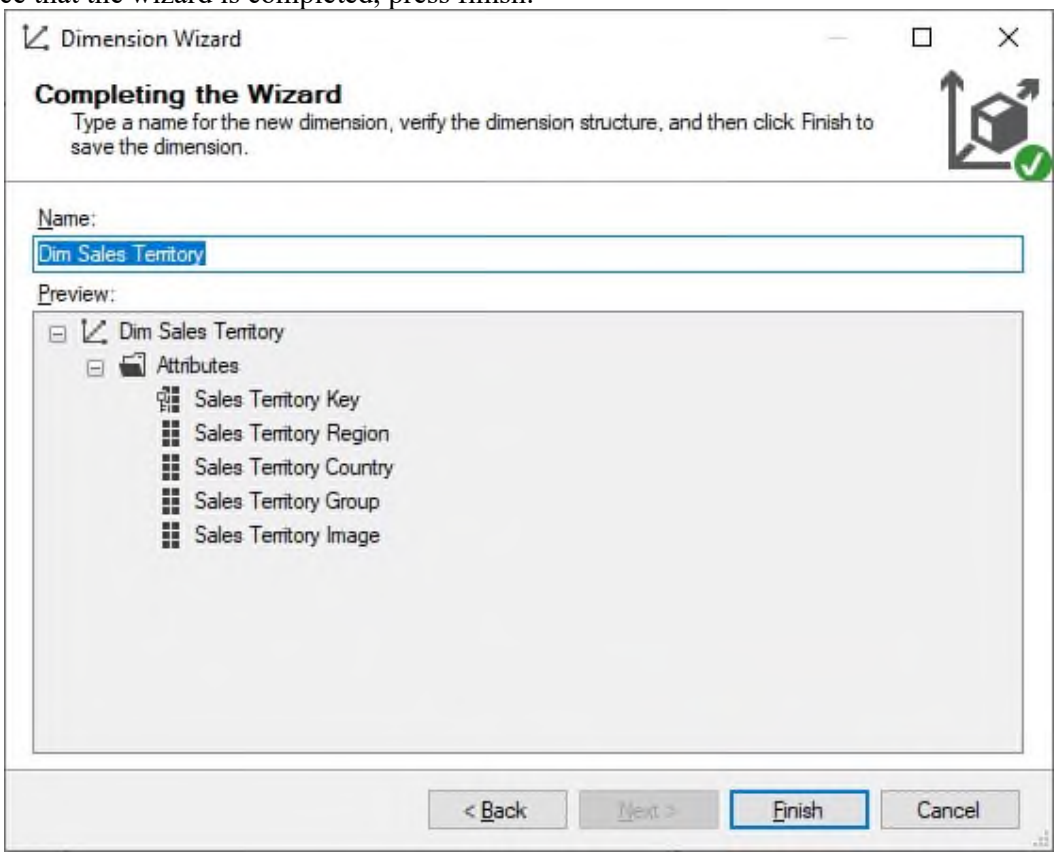
Select the table dimsalesterritory:



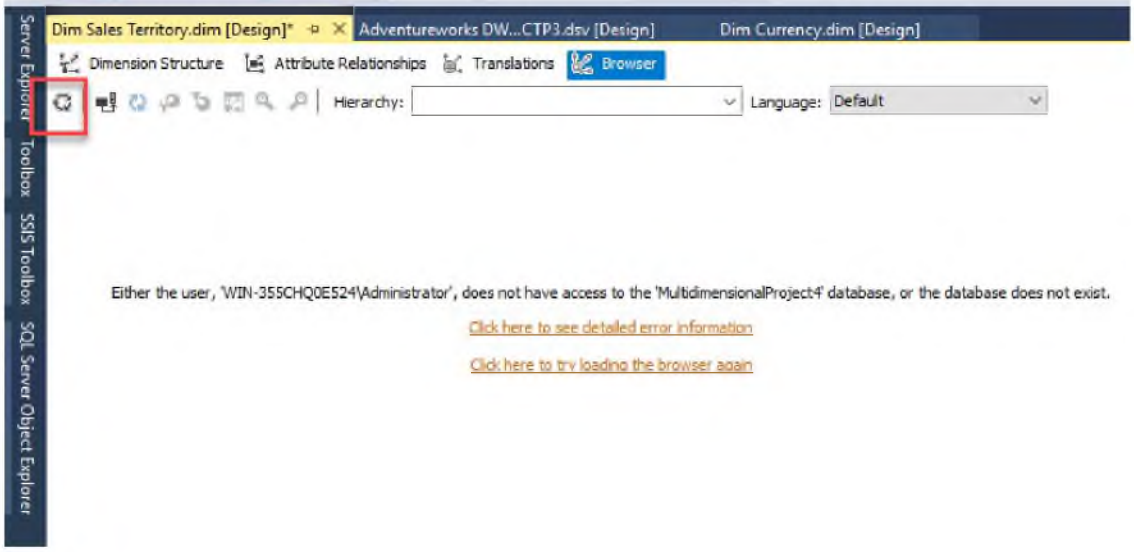
Select the attributes that you want to be brow sable in the reports:



Once that the wizard is completed, press finish:

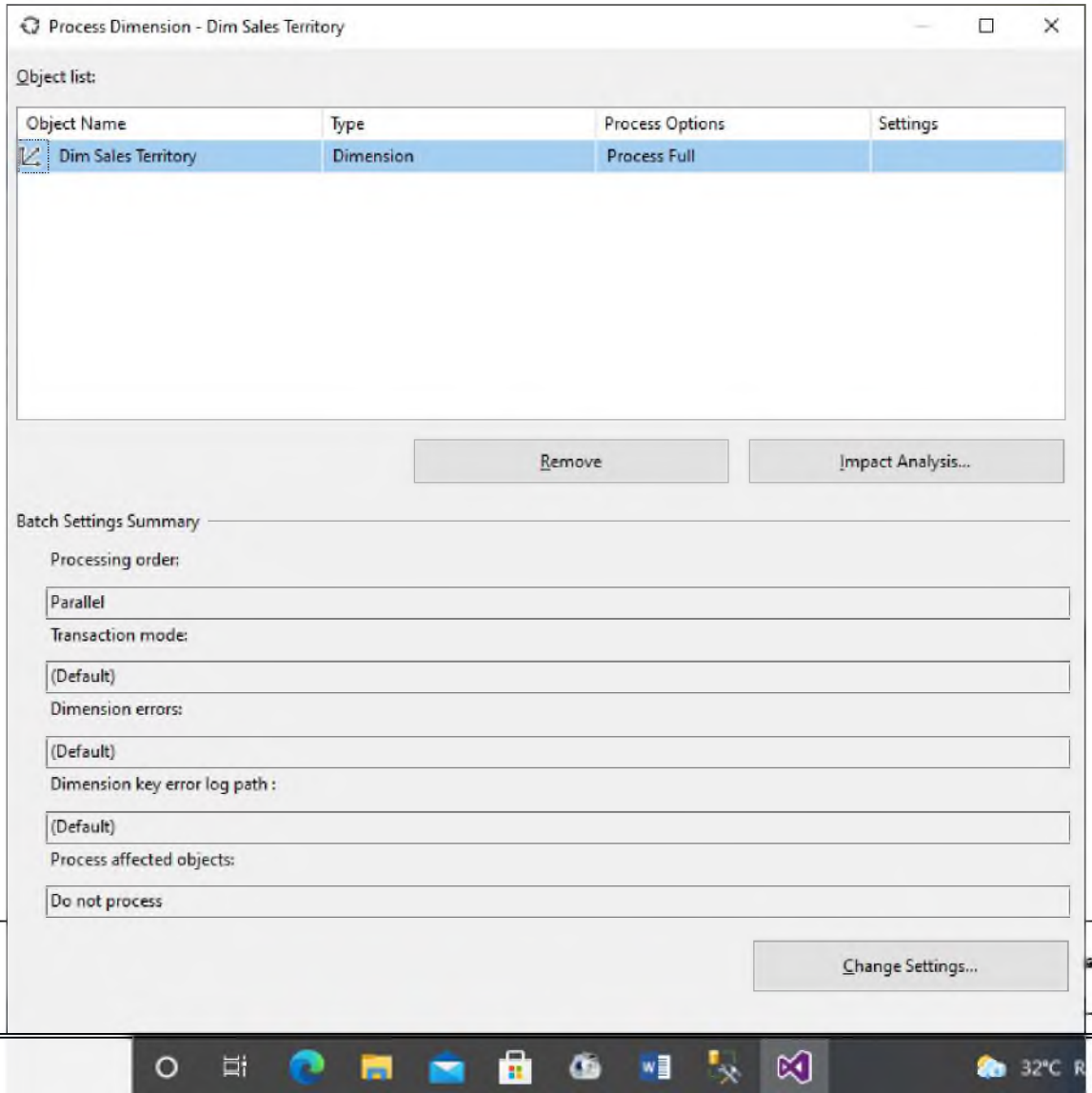


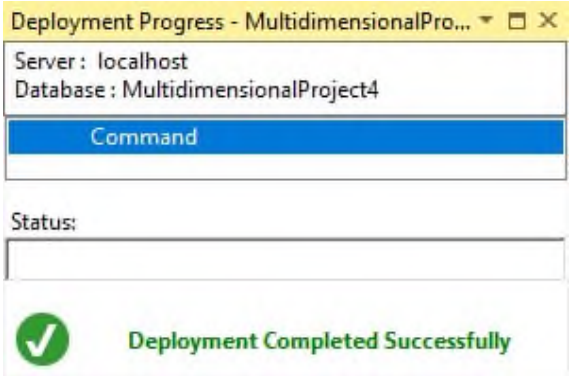
In the new dimension created, click the process icon to process the dimension. The process will generate the structure and load data to the dimension from the SQL Server table:

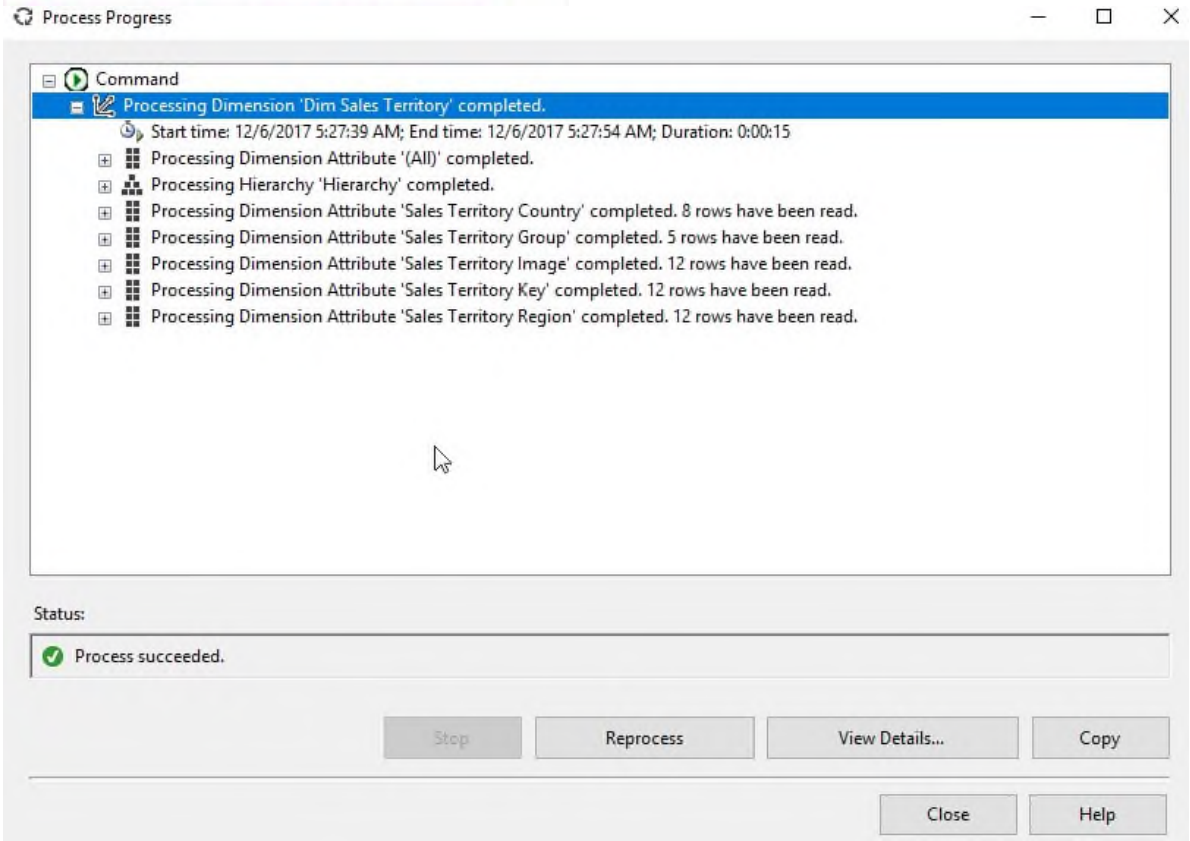


Press run to process the dimension:

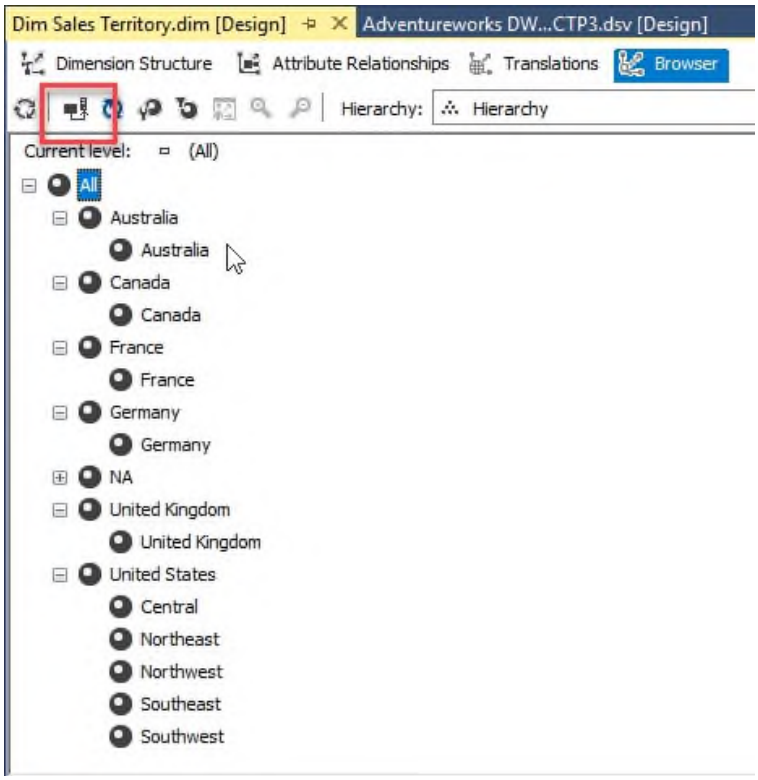
In the Process Progress, close the process once that the process is succeed:



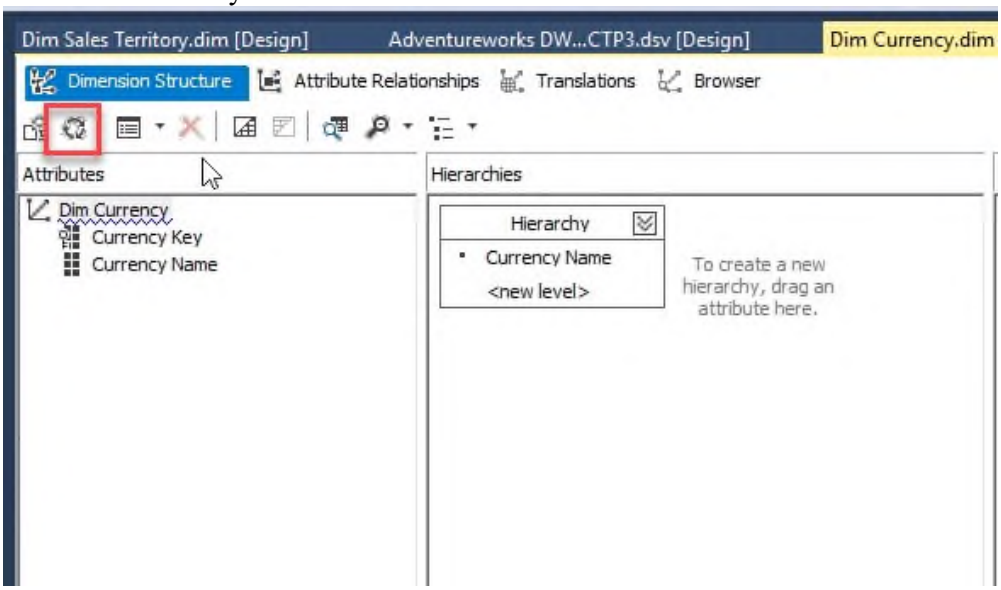




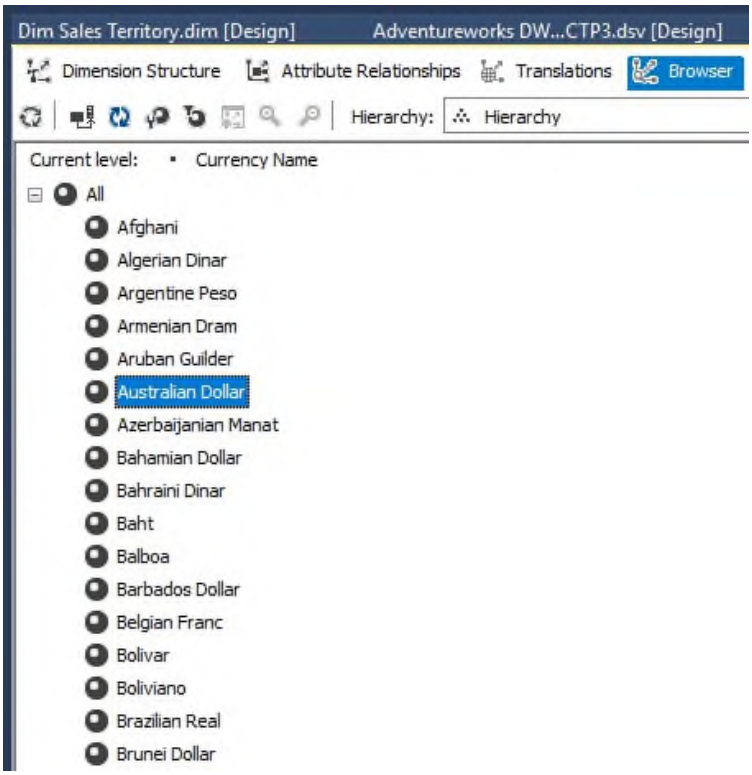
You can refresh the dimension to view the dimension just processed:



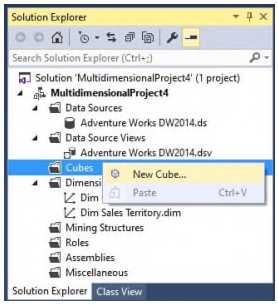
Process the Dim currency dimension also:



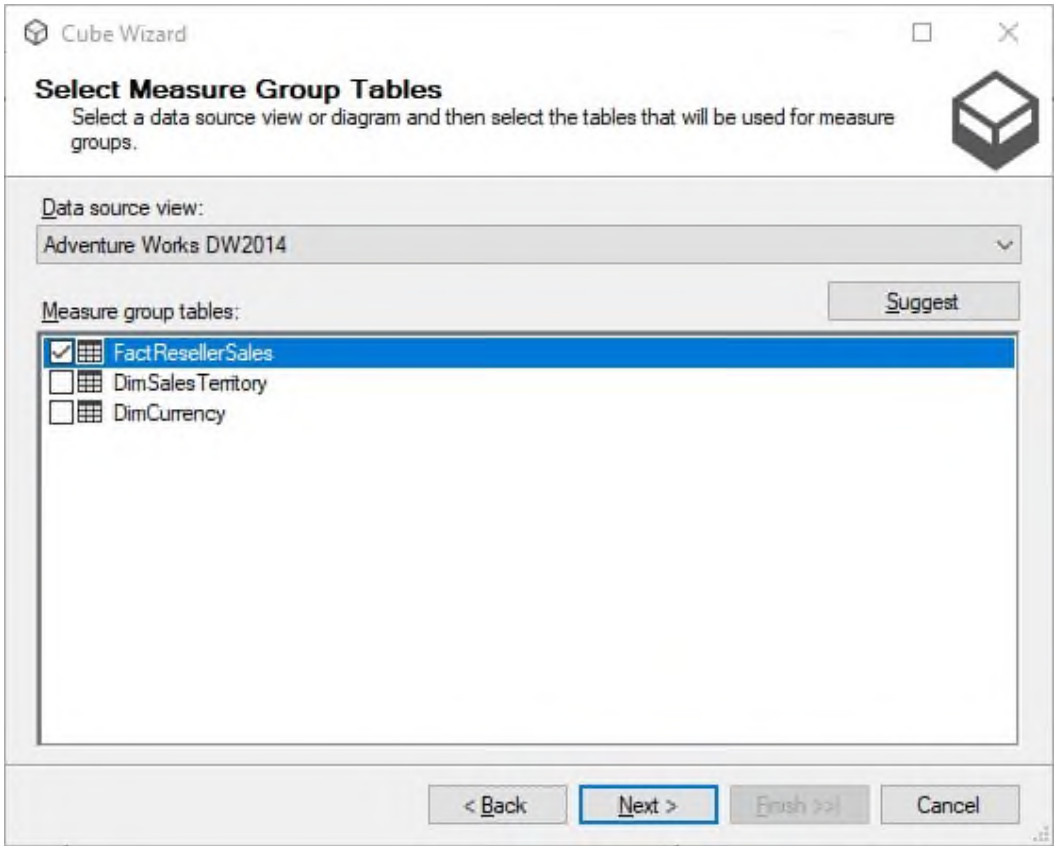
Once processed, you can verify in the Browser tab:



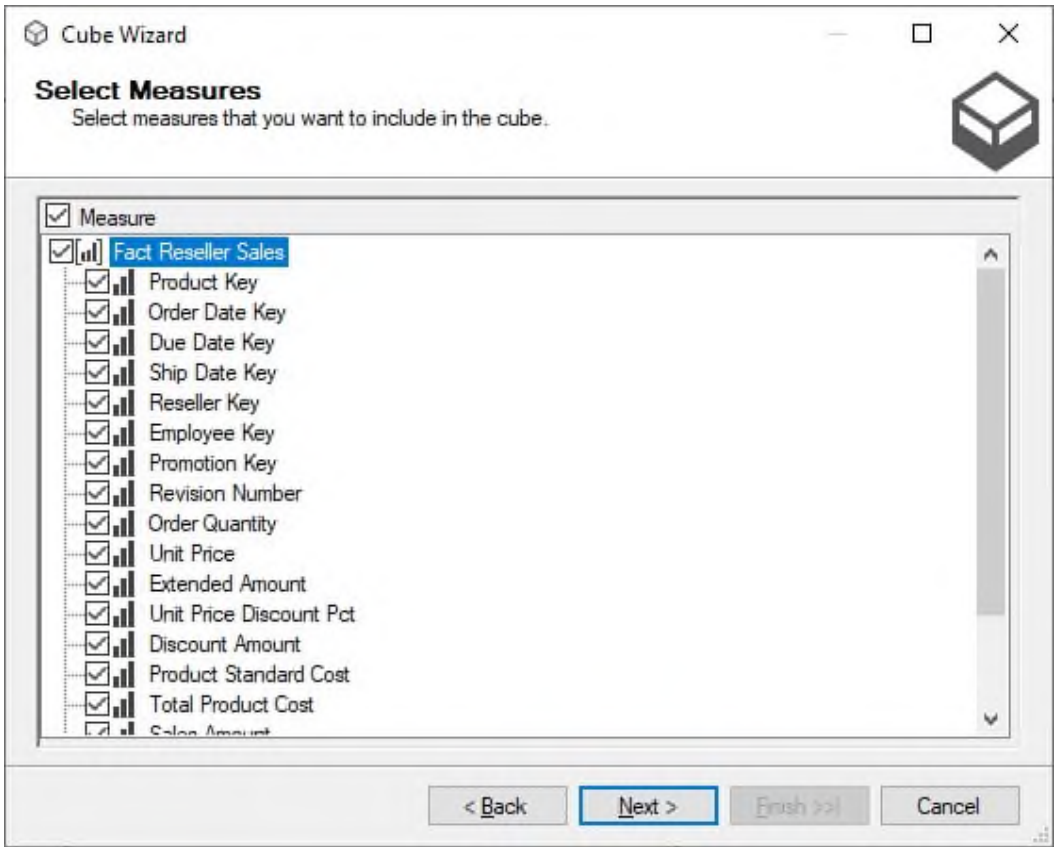
We will now create the cube. In the Solution Explorer, right click **Cubes** and select **New Cube:**



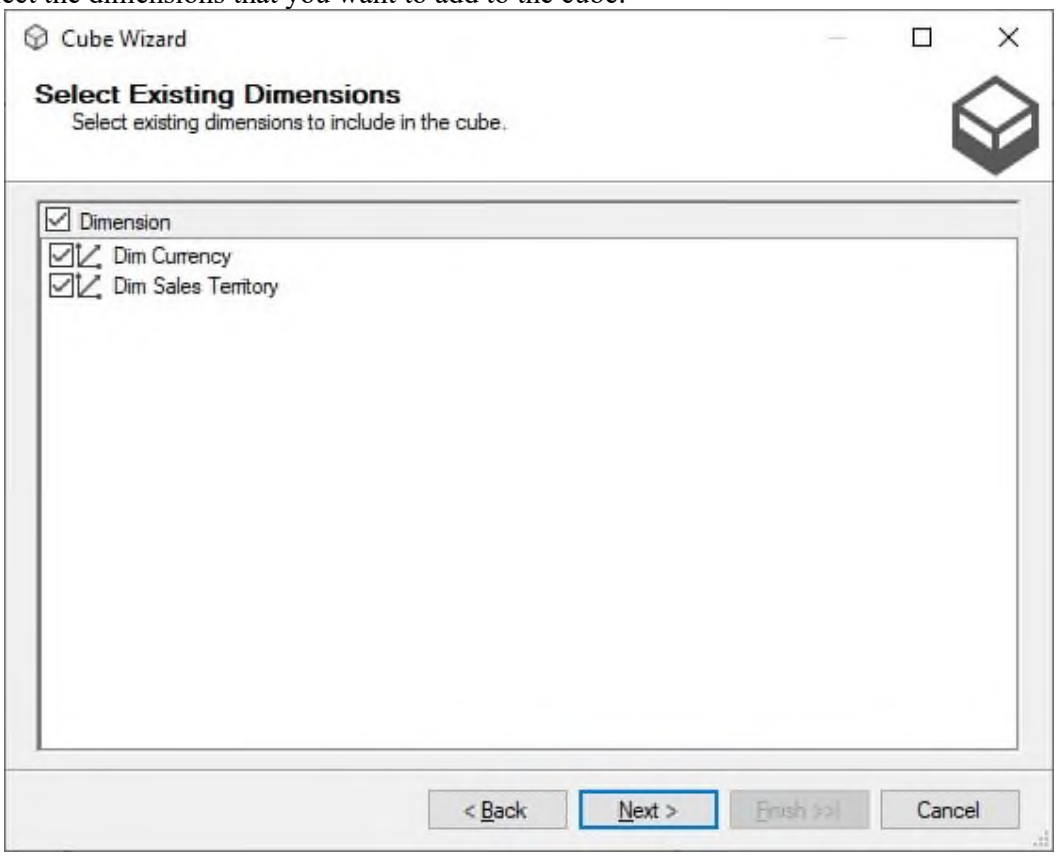
In **Select Measure Groups Tables** select FactResellerSales table. Measure group tables are used to include the table with data to measure. A measure can be the number of sales, amount sold, freight, etc.:



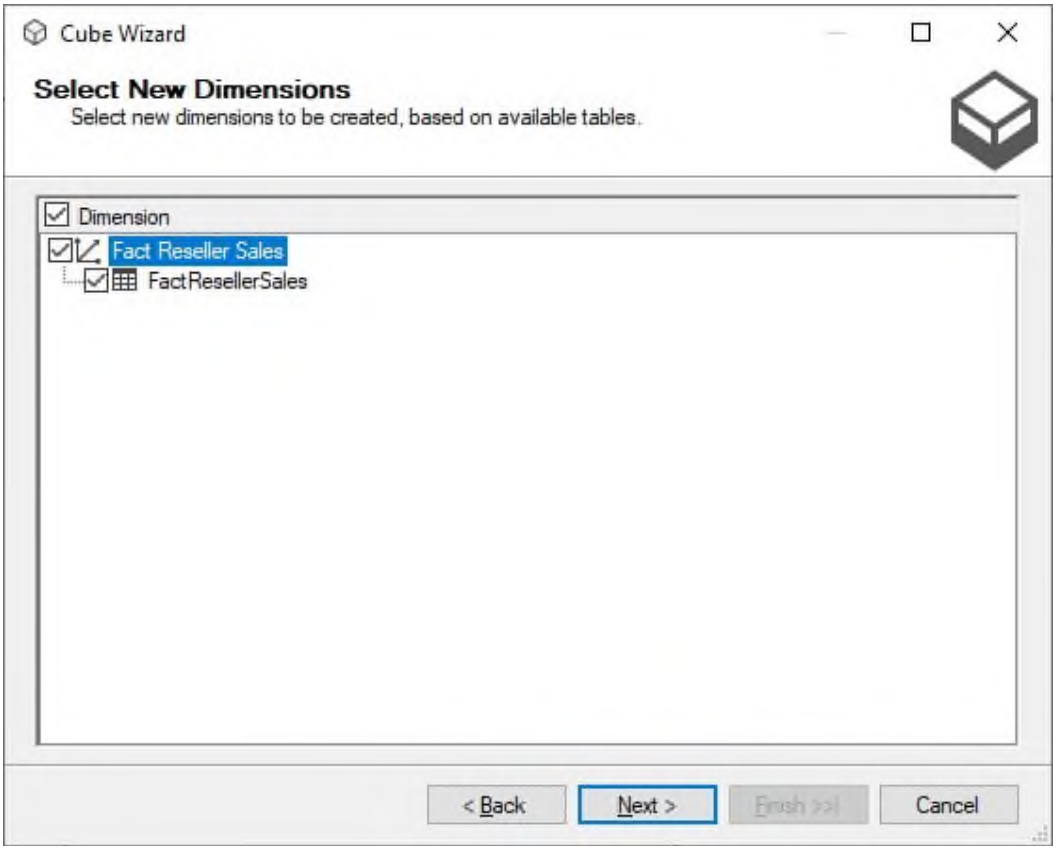
Select the data to measure. We will uncheck the keys and check the other attributes to measure:



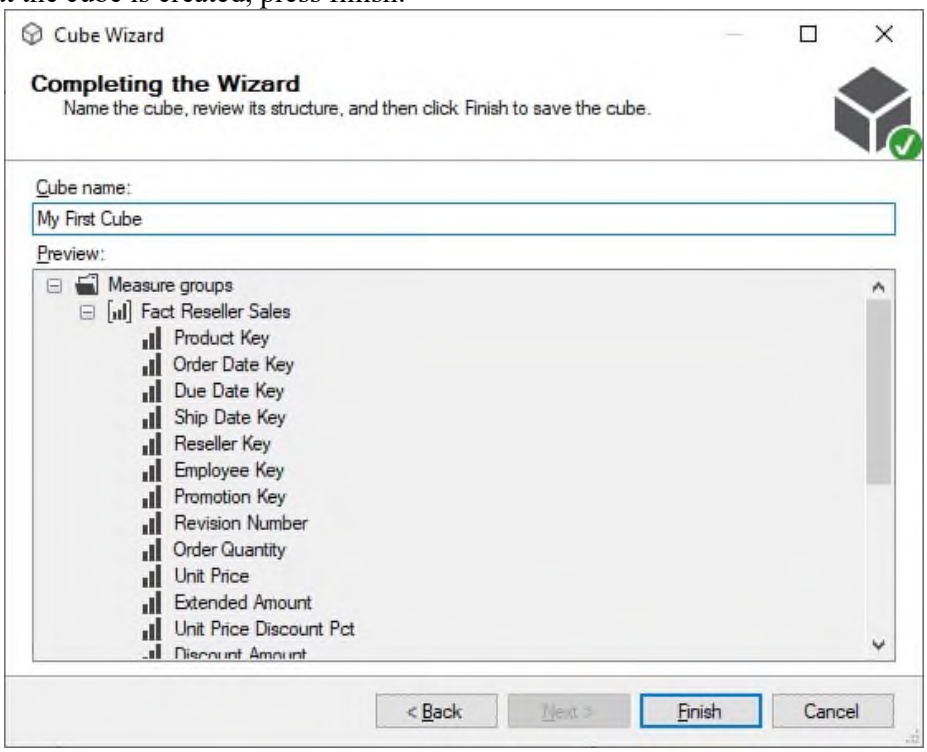
Select the dimensions that you want to add to the cube:



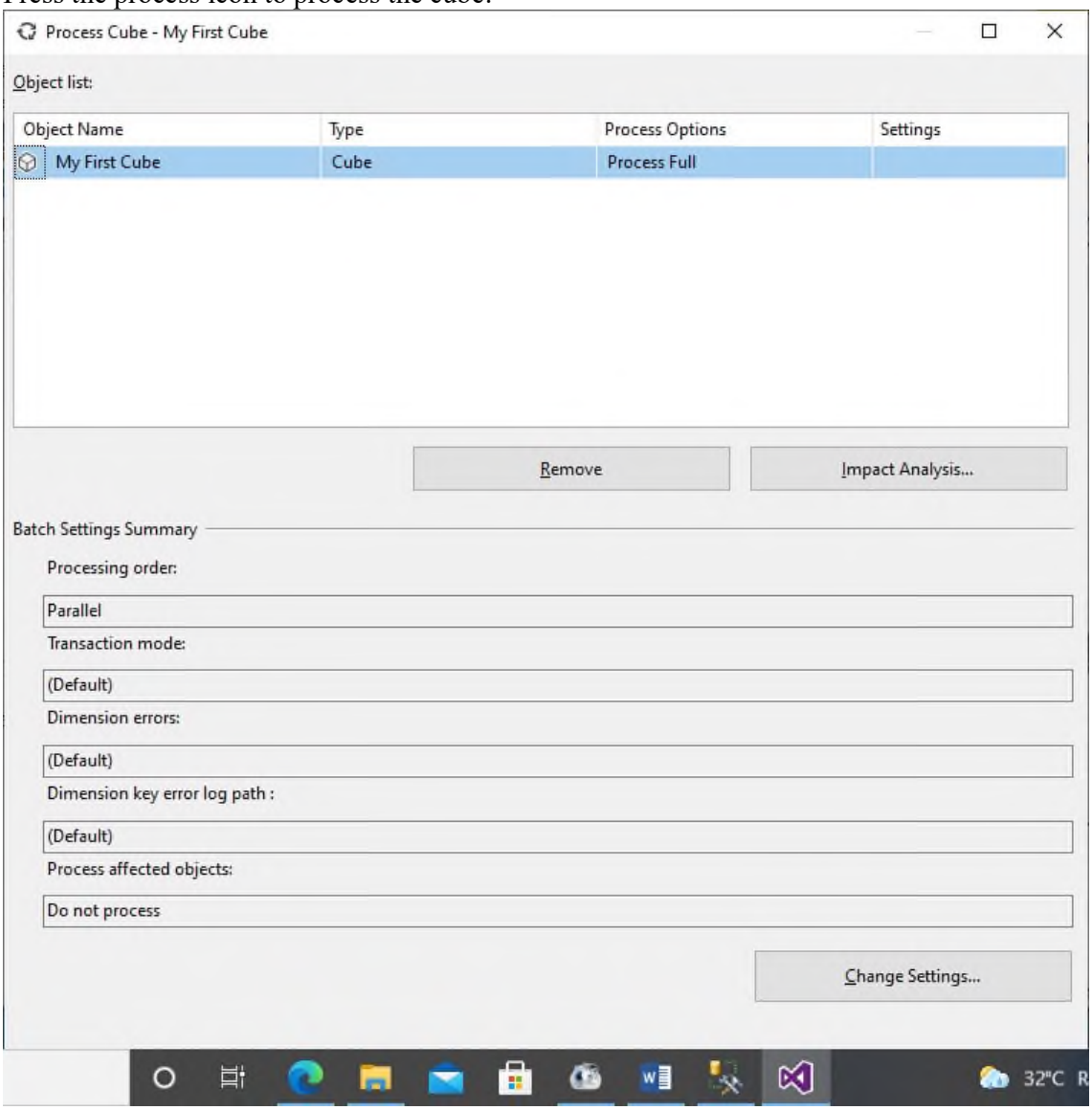
You can also add the fact table as a dimension (degenerate dimension). In this example, we will not add it:



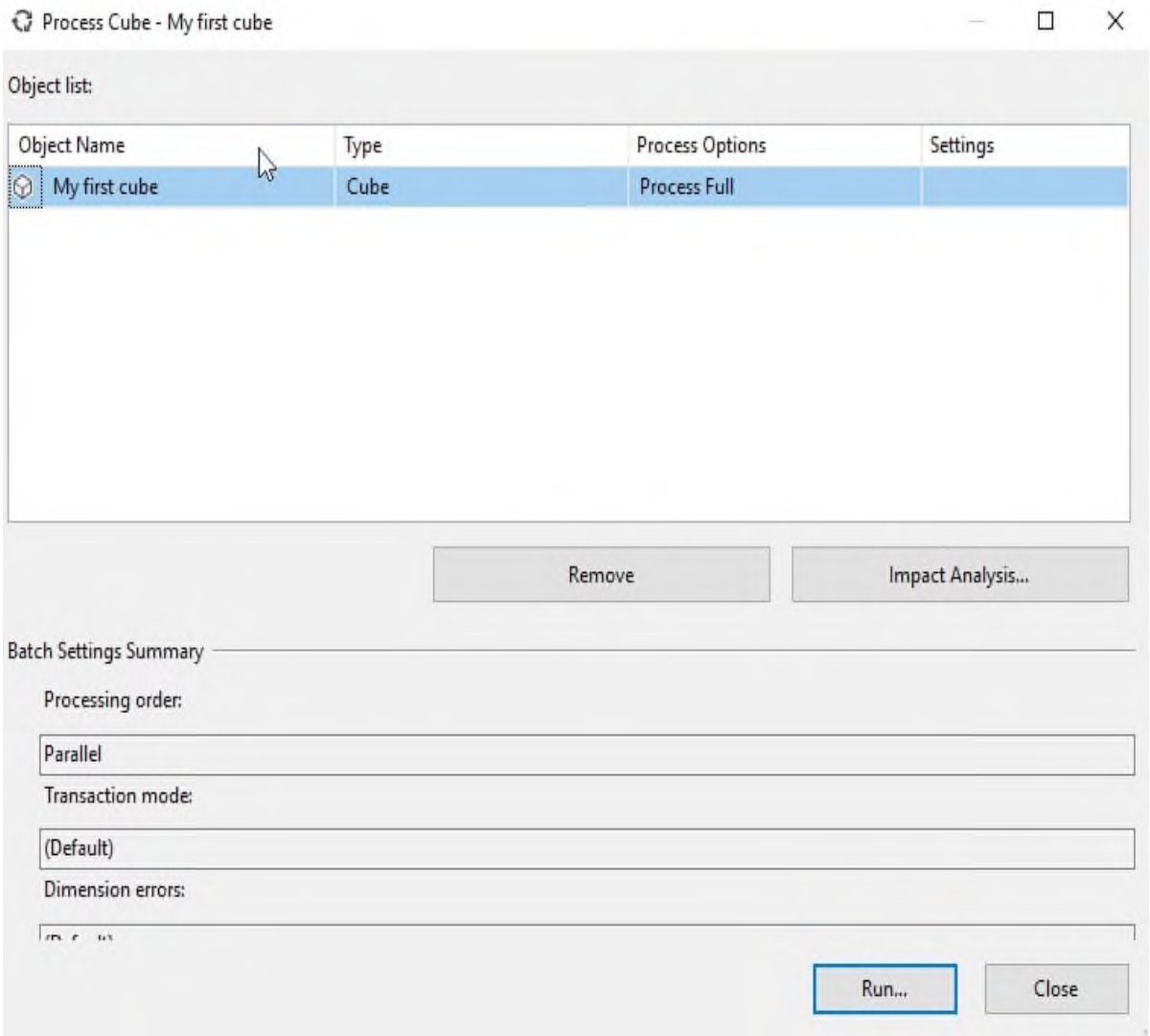
Once that the cube is created, press finish:



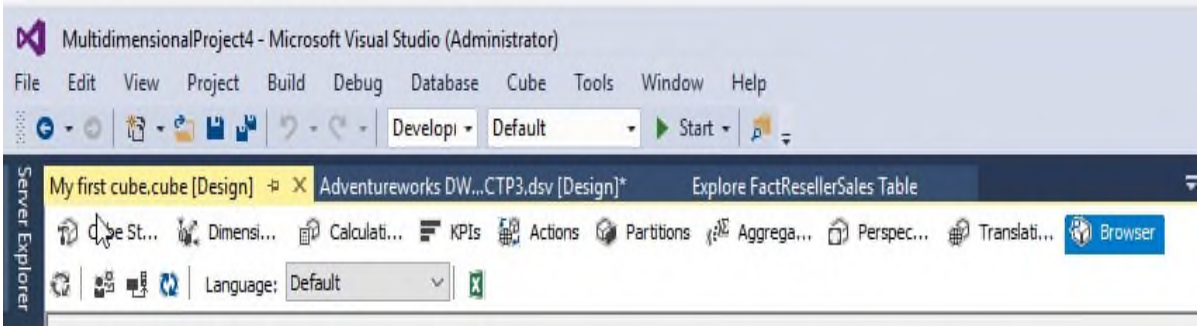
You will have the cube created with the fact table (in yellow) and the dimensions (in blue). Press the process icon to process the cube:



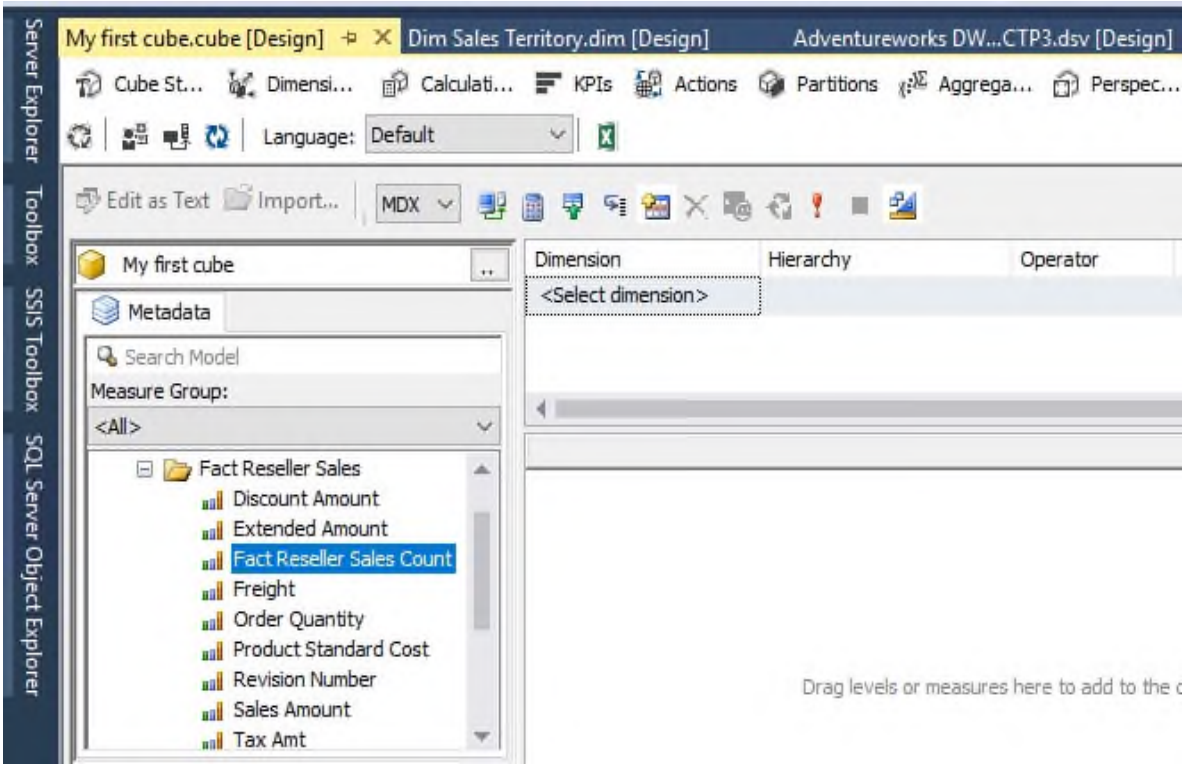
In **Process Cube**, press run to process the cube:



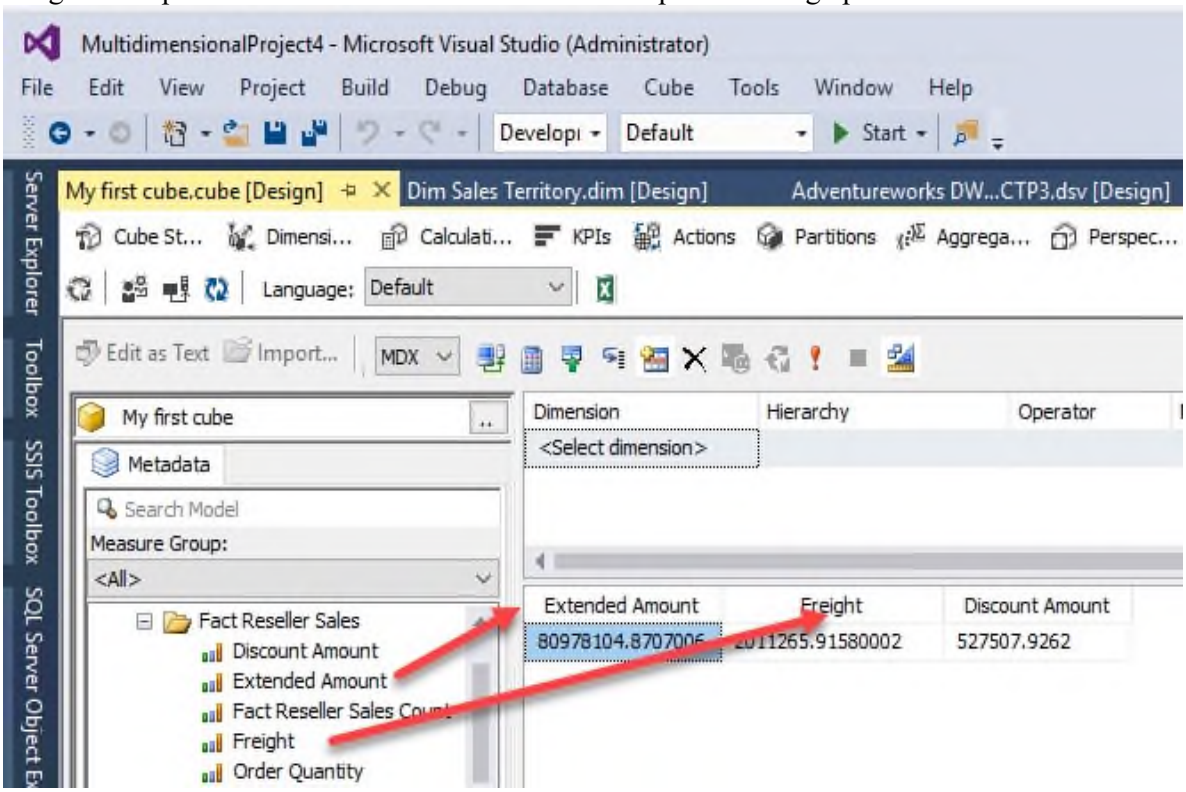
Once processed, go to the browser:



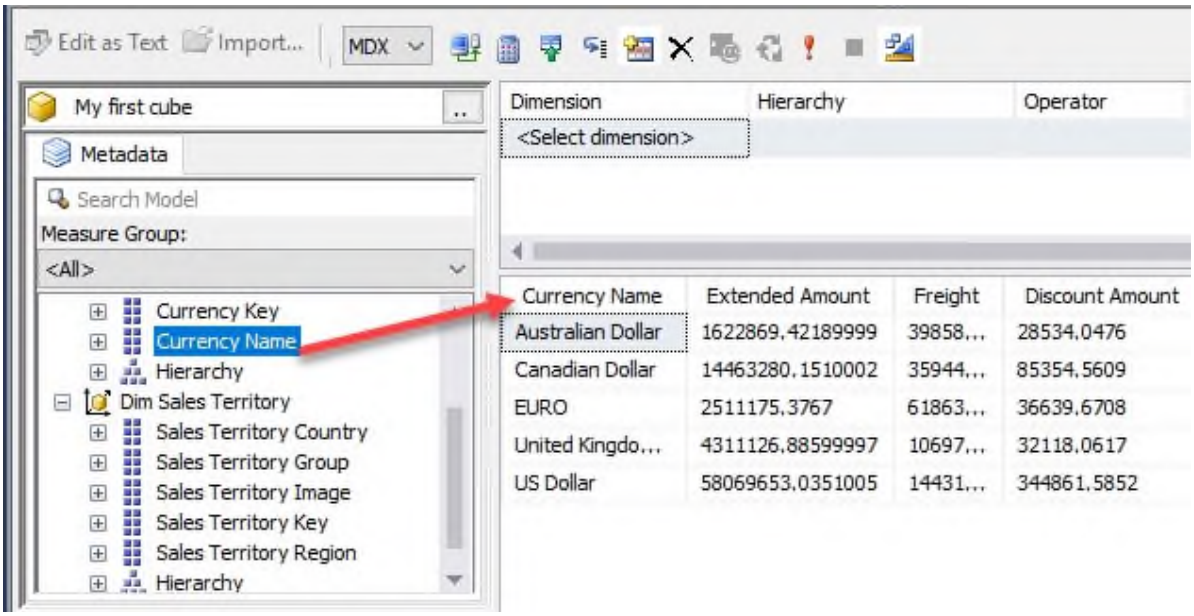
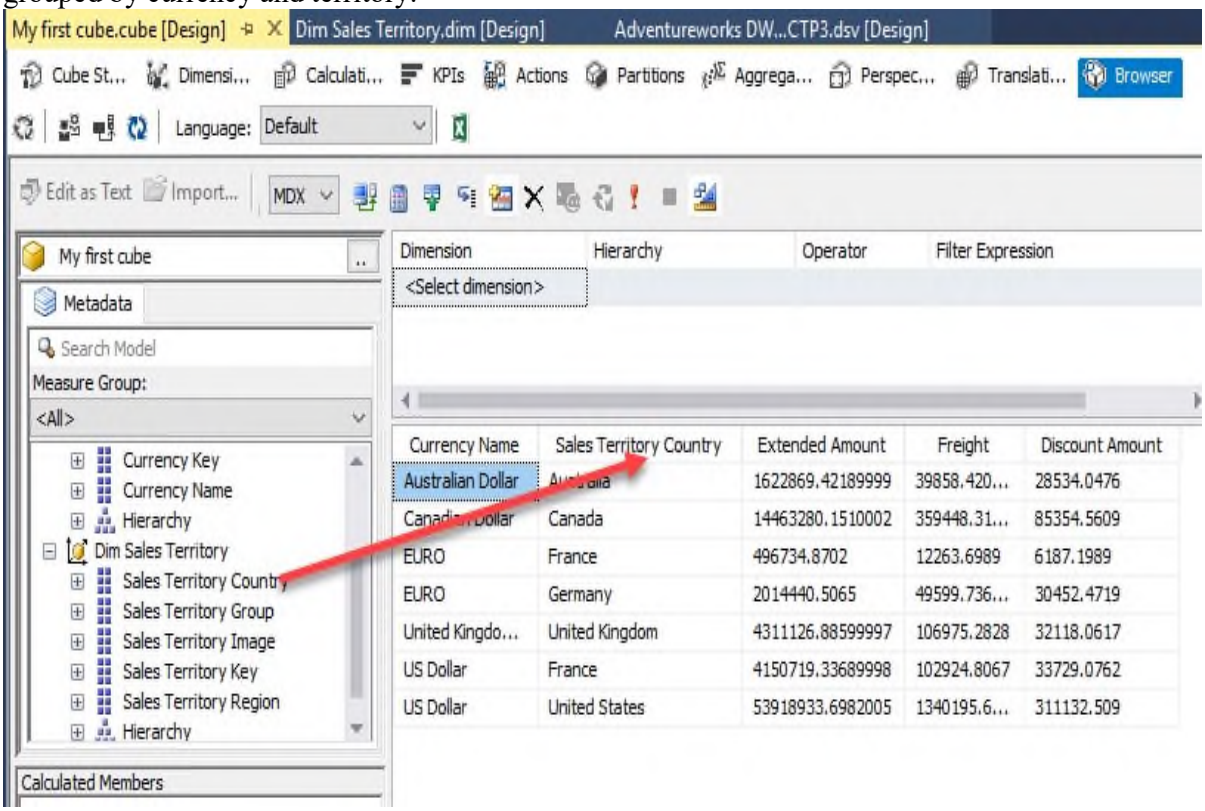
Press the refresh icon:



Drag and drop some measures from the Measure Group to the design pane:



Now drag and drop the currency name to the graph pane, you will be able to see the measures grouped by currency:



Drag and drop the Sales Territory Country. You will now be able to see the measures grouped by currency and territory: