Labs 5: Dimension, measures, star schema, snow flake, shared connection managers and package tasks

1. What are the different database design techniques for OLAP?

**ROLAP (Relational OLAP):**

ROLAP operates directly on relational tables without reorganizing data into a cube.

SQL is commonly used for multidimensional queries, reporting, and analysis1.

**MOLAP (Multidimensional OLAP):**

MOLAP stores data in a multidimensional cube format.

Efficient for complex calculations and aggregations.

Examples include Microsoft Analysis Services (SSAS) and IBM Cognos TM1.

**HOLAP (Hybrid OLAP):**

Combines features of both ROLAP and MOLAP.

Stores summary data in a cube while detailed data remains in relational tables.

Provides a balance between performance and flexibility.

**DOLAP (Desktop OLAP):**

Designed for individual users or small workgroups.

Runs on desktop machines and doesn’t require a separate server.

Limited scalability but suitable for ad hoc analysis.

**WOLAP (Web OLAP):**

Web-based OLAP accessible via browsers.

Enables collaboration and sharing of insights across teams.

Examples include web-based dashboards and reporting tools.

**IOLAP (Intelligent OLAP):**

Integrates OLAP with artificial intelligence (AI) techniques.

Enhances decision-making by providing intelligent recommendations.

1. Why is normalisation design not good for OLAP database?

The debate over whether to normalize or denormalize databases for OLAP (Online Analytical Processing) workloads has been ongoing for years. Let’s explore the reasons behind this discussion:

OLTP (Online Transaction Processing) vs. OLAP:

OLTP databases are designed for transactional systems, where data integrity and operational efficiency are critical. In OLTP systems, normalization is highly recommended to eliminate redundancy and maintain data consistency1.

OLAP databases, on the other hand, serve analytical purposes. They focus on flexible and rapid data analysis. Here, the trade-off between normalization and denormalization becomes more nuanced.

**Normalization in OLAP**:

Advantages:

Data Integrity: Normalization ensures data consistency and reduces anomalies.

Storage Efficiency: Normalized tables occupy less space due to reduced redundancy.

Easier Maintenance: Updates and modifications are straightforward.

Challenges:

Complex Queries: Highly normalized schemas can lead to complex joins and queries, impacting performance.

Aggregation Overhead: Aggregating data across multiple normalized tables can be resource-intensive.

Read Performance: Normalized structures may not be optimal for read-heavy OLAP workloads.

**Denormalization in OLAP:**

Advantages:

Query Performance: Denormalization simplifies queries by reducing joins and improving read performance.

Aggregation Efficiency: Pre-aggregated data in denormalized tables speeds up complex calculations.

User Experience: Faster query response times enhance user satisfaction.

Trade-offs:

Data Redundancy: Denormalization introduces redundancy, which can lead to data inconsistencies if not managed carefully.

Maintenance Complexity: Updates become more complex due to redundant data.

Storage Overhead: Denormalized tables consume more storage space.

Striking the Balance:

The ideal strategy depends on the context:

Highly Transactional OLTP Systems: Prioritize high normalization for data integrity and operational efficiency.

Analytical OLAP Systems: Lean towards denormalization for flexible analysis, with some level of normalization for efficient storage2.

Hybrid Approaches (HOLAP): Combine elements of both normalization and denormalization to achieve a balance.

In summary, while normalization is essential for OLTP systems, OLAP databases often benefit from a judicious mix of both approaches. The choice should align with your specific use case, performance requirements, and data modeling goals.

1. What are dimensions and measures?

Dimension: Qualitative Information: Dimension tables contain qualitative or categorical data. Examples include names, dates, or geographical data.

Quantitative Information: Measure tables provide quantitative or numerical data. They contain numeric values that you can measure.

* Population: Dimension, because it is used to categorise a group of persons and cannot be measured.
* Currency Code: It is a dimension because it is to categorise a specific group of money currency.
* Age (Hint all numbers are not measures): In this context, Age is a dimension because Age is used to categorise a specific group of age (ex:18-30). In case the user would like to know the average of people’s age within a company then Age is measure.
* Employee Name: It is a dimension because it is qualitative, and describes a specific individual. It can not be measured.

1. What is the difference between star schema and snow Flake?

Two common data warehouse modeling techniques:

**Star Schema:**

Definition: Star schema is a type of multidimensional model used for data warehouses.

**Structure:**

Contains a central fact table representing the business transactions or events (e.g., sales, orders).

Dimension tables surround the fact table, providing context and descriptive attributes (e.g., time, product, location).

**Join Complexity:**

In star schema, fewer foreign-key joins are used.

Forms a star shape with the fact table at the center and dimension tables radiating outwards.

**Design Approach:**

Top-down model.

Space Usage:

Consumes more space due to denormalization.

Query Execution Time:

Queries execute faster in star schema.

**Normalization:**

Not used in star schema.

**Design Simplicity:**

Star schema has a simple design.

**Query Complexity:**

Query complexity is low.

**Understanding:**

Easy to understand.

**Foreign Keys:**

Has fewer foreign keys.

**Data Redundancy:**

High data redundancy.

**Snowflake Schema:**

Definition: Snowflake schema is another type of multidimensional model used for data warehouses.

**Structure:**

Contains the same central fact table and dimension tables as star schema.

Additionally, it includes sub-dimension tables (normalized tables) that further break down attributes (e.g., subcategories within a product category).

**Join Complexity:**

In snowflake schema, more joins are involved due to normalization.

Forms a snowflake shape with fact tables, dimension tables, and sub-dimension tables.

**Design Approach:**

Bottom-up model.

**Space Usage:**

Uses less space due to normalization.

**Query Execution Time:**

Takes more time than star schema for query execution.

**Normalization:**

Both normalization and denormalization are used.

**Design Complexity:**

Snowflake schema design can be complex.

**Query Complexity:**

Query complexity is higher than star schema.

**Understanding:**

Requires more effort to understand.

**Foreign Keys:**

Has more foreign keys.

**Data Redundancy:**

Low data redundancy due to normalization12.

In summary, choose the schema that best aligns with your specific data analysis needs, performance requirements, and understanding of the data

1. How to share connection manager across different tasks?

We share connection manager across different tasks by creating a connection manager on the solution explorer then we click on the connection manager.

1. How to disable and enable Tasks on a package ?

In order to disable or enable a task, we should right click the task and select disable or enable options (on the displayed menu).

1. How to execute a package ?

To execute a package it is necessary to double left click the package that we want to execute, and then click on the run button.

In case we have more than one package, we want to run a specific package as a start up, we should right click the package that we would like to run first, then select the menu RUN AS START UP OBJECT.

Labs 6: SCD, Type 0, Type 1, OLEDB command and Unicode conversions

In summary:

* Fixed attributes remain constant.
* Changing attributes are updated over time.
* Historical attributes save changed values in new records while marking previous records as outdated.

Unicode and Non-unicode

Certainly! Let’s explore the differences between Unicode and non-Unicode characters:

**Unicode:**

Definition: Unicode is a universal character encoding standard that represents characters from various languages and scripts.

Character Set: Unicode includes roughly 100,000 characters, covering a wide range of languages, symbols, and special characters.

Storage Size: Unicode uses multiple bytes (typically 2 or 4 bytes) to represent each character.

Variants: It has three common variants: UTF-8, UTF-16, and UTF-32.

Usage:

Widely used in web development, databases, and international applications.

Default encoding for many programming languages.

Supports emojis and diverse character sets.

**Non-Unicode:**

Definition: Non-Unicode (also known as ANSI or legacy encoding) is an older character encoding system.

Character Set: It supports a limited set of characters (usually 256) based on the specific encoding (e.g., ISO-8859-1, Windows-1252).

Storage Size: Non-Unicode uses single bytes (8 bits) per character.

Limitations: Due to its limited character set, it cannot represent all languages and symbols.

Usage:

Legacy systems that predate Unicode adoption.

Local applications with specific language requirements.

Slowly Changing Dimension (SCD) task is used when we want to insert or update a database (To avoid errors that don’t allow insertion of records that contains the same Business Key).

To use SCD for auditoria purposes, it is important to remove the primary key on the destination table, to allow the insertion of two records with the same business key to audit changes between them. Setting type 2 (Historical attribute) is crucial to ensure auditing. Additionally, it is imperative to create a Boolean column within the destination table that would be used to show the current and expired records.

In Microsoft Visual Studio 2022, when using Slowly Changing Dimension Wizard, in the historical Attribute Options, the “variable to set date values” option does not act as users would expect due to the fact it does not give the possibility to use the drop-down menu. The alternative option is to navigate with the mouse’s pad to solve the issue. Additionally, to connect to the local SQL server 2022 database, it is important to use Microsoft OLE DB Provider for SQL Server. Version: Microsoft Visual Studio Community 2022 (64-bit) – Current Version 17.9.7.

1. What is SCD and why is the name slowly changing dimensions?

Slowly Changing Dimensions (SCD) refer to how data in a data warehouse changes over time. These dimensions have the same natural key (such as a product ID or customer ID), but other data columns associated with them may or may not change over time. The term “slowly changing” emphasizes that these changes occur gradually rather than frequently.

Here are some key points about SCDs:

Purpose of SCDs:

SCDs are essential for tracking how records evolve over time.

They allow us to maintain historical context while accommodating changes.

By understanding SCDs, businesses can analyze trends, monitor performance, and make informed decisions.

Types of Slowly Changing Dimensions:

There are different types of SCDs based on how they handle changes:

Type 0: These dimensions never change. Examples include mapping tables (e.g., states, zip codes) and static data like social security numbers.

Type 1: Data is overwritten by new data without keeping a historical record. No tracking of changes over time occurs. For instance, customer addresses often fall into this category.

Type 2: Historical records are preserved by creating new rows for each change. This type allows tracking of changes over time. For example, product features or customer attributes.

Type 3: Only limited history is maintained by adding new columns to represent changes. It’s a compromise between Type 1 and Type 2.

Avoiding Costly Mistakes:

Choosing the wrong SCD type can impact business metrics.

For instance, if historical attributes are not tracked, gaps in reporting may occur.

Understanding SCDs helps prevent such mistakes.

In summary, SCDs play a crucial role in data analytics by allowing us to observe how data evolves over time, ensuring accurate reporting and informed decision-making.

1. What is the need of business key in SCD?

The business key (also known as the natural key) plays a crucial role in Slowly Changing Dimensions (SCD). Let’s explore why it is needed:

Uniqueness and Identification:

The business key uniquely identifies each dimension record.

It ensures that each entity (such as a customer, product, or location) has a distinct representation in the data warehouse.

Without a business key, it would be challenging to differentiate between similar records.

Tracking Changes Over Time:

SCDs capture how data changes over time.

By using the business key, we can track historical changes for a specific entity.

For example, if a customer’s address changes, the business key (such as the customer ID) allows us to maintain a history of both old and new addresses.

Joining and Integration:

When integrating data from multiple sources, the business key serves as a common link.

It enables joins between fact tables and dimension tables.

For example, in a sales transaction, the business key (e.g., product ID) connects the sales fact table with the product dimension.

Efficient Lookups and Queries:

Using the business key for lookups (e.g., finding a customer’s details) is more efficient than searching by other attributes.

Indexes are often created on business keys to optimize query performance.

In summary, the business key ensures uniqueness, facilitates historical tracking, supports data integration, and improves query efficiency in SCDs.

1. Explain the difference between fixed, changing and historical attributes?

Certainly! Let’s delve into the differences between fixed attributes, changing attributes, and historical attributes in the context of Slowly Changing Dimensions (SCD):

Fixed Attributes:

Definition: Fixed attributes represent characteristics of a dimension that remain constant over time.

Purpose: These attributes provide stable information about a dimension member.

Examples:

In a product dimension, the SKU (Stock Keeping Unit) is a fixed attribute. Once assigned, the SKU rarely changes.

Fixed attributes are typically used for descriptive or categorization purposes.

Changing Attributes:

Definition: Changing attributes are those that can be updated or modified over time.

Purpose: These attributes capture dynamic information about a dimension member.

Examples:

In the same product dimension:

The price of a product is a changing attribute. Prices can fluctuate due to promotions, inflation, or other factors.

Changing attributes are often used for performance metrics or operational data.

Historical Attributes:

Definition: Historical attributes represent changes in a dimension member over time while preserving historical context.

Purpose: These attributes allow us to track changes and maintain a history of values.

Examples:

Consider a customer dimension:

The address of a customer is a historical attribute.

When a customer moves to a new address, a new record is created with the updated address, while the previous address remains in the system (marked as outdated).

Historical attributes are crucial for trend analysis and accurate reporting.

In summary:

Fixed attributes remain constant.

Changing attributes are updated over time.

Historical attributes save changed values in new records while marking previous records as outdated

1. What is the difference between Unicode and Non-unicode types ?

Unicode provides a broader character repertoire and better international support, while non-Unicode systems are more limited in scope.

1. What is the difference between type1 and type 0 SCD’s?

Type 0 SCDs retain the original data without any updates.

Type 1 SCDs overwrite existing data with new values, without tracking historical changes.

1. What is the difference between DT\_STR and DT\_WSTR?

DT\_STR:

Definition: The DT\_STR data type represents a null-terminated ANSI character string.

Character Set: It uses the single-byte character encoding (such as ASCII or ISO-8859-1).

Storage Size: The length is variable, and the maximum length is typically 8000 bytes.

Usage:

Used for non-Unicode (ASCII) character data.

Suitable for legacy systems or when dealing with specific code pages.

Example: Storing names, addresses, or other text data.

DT\_WSTR:

Definition: The DT\_WSTR data type represents a null-terminated Unicode character string.

Character Set: It uses the double-byte character encoding (UTF-16).

Storage Size: The length is variable, and the maximum length is also typically 8000 characters.

Usage:

Used for Unicode character data (supporting multiple languages and special characters).

Suitable for modern applications, internationalization, and data exchange.

Example: Storing multilingual text, emojis, or any data requiring Unicode support.

Implications:

When working with expressions in SSIS:

You can use either DT\_STR or DT\_WSTR as the return type.

Inside an expression, only DT\_WSTR is supported, and DT\_STR values are automatically converted to DT\_WSTR values.

In summary:

DT\_STR is for non-Unicode (ASCII) character data.

DT\_WSTR is for Unicode character data (UTF-16) supporting multiple languages and special characters.

1. Can destination component do UPDATES and DELETES?

Certainly! Let’s explore how destination components in SQL Server Integration Services (SSIS) handle updates and deletes:

Updates:

Destination components can indeed handle updates. When using this approach, you typically follow these steps:

Save the rows to temporary tables (created specifically for this purpose) within the destination component.

After the data flow task, use an Execute SQL task to perform bulk update operations against the destination table from the temporary tables.

This allows you to efficiently update existing records in the destination based on changes from the source.

Deletes:

Similarly, destination components can handle deletes. Here’s how:

Save the rows to temporary tables (again, created for this purpose) within the destination component.

After the data flow task, use an Execute SQL task to perform bulk delete operations against the destination table from the temporary tables.

This ensures that any records marked for deletion in the source are removed from the destination.

Overall Approach:

The general pattern involves identifying inserts, updates, and deletes in the source data.

You then push the updates and deletes to staging tables.

Finally, you use set-based SQL statements (UPDATE and DELETE) to apply these changes to the destination from the staging tables.

In summary, destination components can handle both updates and deletes, allowing you to efficiently synchronize data between source and destination systems in SSIS.

1. What is the need of OLEDB command component?

The OLE DB Command component in SQL Server Integration Services (SSIS) serves a specific purpose and is useful in certain scenarios. Let’s explore its need and functionality:

Purpose:

The OLE DB Command component is used for row-level operations within an SSIS data flow.

Unlike other destination components (such as OLE DB Destination or Flat File Destination), which handle bulk inserts, the OLE DB Command component allows you to execute individual SQL commands for each row of data.

Use Cases:

The OLE DB Command component is typically used when:

You need to perform customized updates, deletes, or other operations on a row-by-row basis.

The transformation logic cannot be achieved using standard SSIS transformations.

You want to apply business rules or complex calculations during data flow execution.

How It Works:

For each input row, the OLE DB Command component executes a parameterized SQL command against the specified database connection.

You define the SQL command (usually an UPDATE or DELETE statement) with placeholders for parameters.

The component maps input columns to these parameters, allowing you to customize the command for each row.

Performance Considerations:

The OLE DB Command component can be inefficient for large datasets because it executes individual SQL commands sequentially.

It involves round-trips to the database for each row, which can impact performance.

Use it judiciously and consider alternatives (such as staging tables and set-based operations) for better performance.

In summary, the OLE DB Command component is valuable when you need fine-grained control over row-level operations during data flow execution. However, due to its potential performance impact, use it selectively and optimize your SSIS packages accordingly.

1. What are params and sequence in OLEDB command?  
   Params allow you to customize SQL commands for each row using placeholders.

Sequence determines the order in which SQL commands are executed for individual rows in the OLE DB Command component.

Labs 7

What is the use of the lookup component?

The **Lookup Transformation** in **SQL Server Integration Services (SSIS)** is a powerful tool used for comparing data between the source and destination. Let’s explore its use cases and functionality:

1. **Purpose of the Lookup Transformation**:
   * The Lookup Transformation allows you to:
     + Compare data from the input (source) with data in a reference dataset (such as a table or query).
     + Filter out matched and unmatched rows based on specified conditions.
     + Perform lookups to access additional information in related tables.
2. **Common Use Cases**:
   * Here are some scenarios where the Lookup Transformation is useful:
     + **Data Enrichment**:
       - You have a fact table with foreign keys (e.g., product IDs, customer IDs).
       - You want to enrich the fact table by looking up additional information (e.g., product names, customer details) from dimension tables.
     + **Data Cleansing**:
       - You need to validate data against reference data (e.g., validating customer ZIP codes against a master ZIP code table).
       - Rows with invalid or unmatched data can be redirected to an error output.
     + **Incremental Data Loading**:
       - When loading data incrementally (e.g., from a staging table to a data warehouse), you can use the Lookup Transformation to identify new, updated, or deleted records.
3. **Configuration**:
   * In the Lookup Transformation, you configure:
     + The connection manager to the reference dataset (usually an OLE DB connection).
     + The columns to compare (input columns vs. reference columns).
     + The lookup mode (full cache, partial cache, or no cache).
     + The output (matched, unmatched, or both).
4. **Performance Considerations**:
   * The Lookup Transformation can be resource-intensive, especially when using full cache mode.
   * Consider using partial cache or no cache for large datasets.
   * Optimize your package design to minimize lookups when possible.

In summary, the Lookup Transformation is essential for data enrichment, validation, and incremental loading scenarios in SSIS. [It allows you to compare data efficiently and make informed decisions during ETL processes](https://www.sqlshack.com/an-overview-of-the-lookup-transformation-in-ssis/)

What is the difference between compiled and debug mode in SSIS?

Certainly! Let’s explore the differences between **compiled mode** and **debug mode** in **SQL Server Integration Services (SSIS)**:

1. **Compiled Mode**:
   * **Definition**: Compiled mode refers to the state of an SSIS package after it has been successfully built (compiled) without any errors.
   * **Purpose**:
     + In compiled mode, the SSIS package is ready for deployment and execution.
     + It ensures that the package is syntactically correct and all components are properly connected.
   * **Characteristics**:
     + No breakpoints are active.
     + No debugging information is generated.
     + The package is optimized for execution.
2. **Debug Mode**:
   * **Definition**: Debug mode is used during development and testing to identify and fix issues in the SSIS package.
   * **Purpose**:
     + Debugging allows you to step through the package, inspect variables, and identify errors.
     + It helps locate logic errors, data flow issues, and unexpected behavior.
   * **Characteristics**:
     + Breakpoints can be set to pause execution at specific points.
     + Debugging information (such as variable values) is available.
     + The package runs slower due to additional overhead for debugging.
3. **Workflow**:
   * During development, you work in debug mode to identify and resolve issues.
   * Once the package is stable and error-free, you switch to compiled mode for deployment.
4. **Switching Modes**:
   * In Visual Studio, you can switch between compiled mode and debug mode:
     + Use the **Start Debugging** button (or press F5) to run the package in debug mode.
     + Use the **Start Without Debugging** button (or press Ctrl+F5) to run the package in compiled mode.

In summary, compiled mode is for production deployment, while debug mode is for development and troubleshooting. [Debugging provides detailed insights into package behavior, whereas compiled mode ensures optimal execution](https://stackoverflow.com/questions/2651243/what-is-the-difference-between-debug-and-compile)

Lab 8L Sort, Merge and Merge joins.

Lab 9: Creating SSAS Cube (SSAS)

Issue: When Installing SQL server for first time, I chose tabular mode on my MSQLSERVER instance for Analysis Services, which made it impossible for me to launch my SSAS cube (which is set to Multidimensional mode). I had to create a new instance named SSAS\_MultiDimen, select analysis services again and finally choosing multidimensional mode.

I created new directories (just like the first installation for good practices for saving data about log, data, temp and backup files) with SSAS as a suffix at the end.

I went to the SSASCustomer projects, clicked on properties, then the deployment tab, I changed the server from localhost to EMMAN\SSAS\_MultiDimen (YourServer\SSAS\_MultiDimen). I rebuilt the projects and clicked on process which managed to solve the issue.

Here’s the steps I followed which were recommended by chatGPT:

To verify the mode of your SQL Server Analysis Services (SSAS) instance in SQL Server Management Studio (SSMS), follow these steps:

1. **Open SQL Server Management Studio (SSMS)**:
   * Launch SSMS from your Start menu or desktop.
2. **Connect to the Analysis Services Instance**:
   * In the "Connect to Server" dialog box, select "Server type" as "Analysis Services".
   * Enter the server name (e.g., localhost if it's installed locally).
   * Click "Connect".
3. **Check the Server Mode**:
   * After connecting, right-click on the Analysis Services instance name in the Object Explorer.
   * Select "Properties" from the context menu.
4. **View the Server Mode**:
   * In the "Properties" window, go to the "Information" section.
   * Look for the "Server Mode" property. This will indicate the current mode of the SSAS instance (e.g., Multidimensional, Tabular)

Since your existing SQL Server Analysis Services (SSAS) instance is set to "Tabular" mode and it’s not possible to change it directly, you’ll need to install a new SSAS instance in "Multidimensional and Data Mining" mode. Here are the steps to do this:

**Installing a New SSAS Instance in Multidimensional Mode**

1. **Launch SQL Server Installation Center**:
   * Open the SQL Server Installation Center from your Start menu or search for "SQL Server Installation Center".
2. **Start a New Installation**:
   * Click on "New SQL Server stand-alone installation or add features to an existing installation".
3. **Setup Support Rules**:
   * The installer will run a setup support rules check. Ensure all checks pass, then click "Next".
4. **Specify a New SQL Server Instance**:
   * On the "Instance Configuration" page, select "Named instance" and provide a unique name for this new instance (e.g., SSAS\_Multidimensional).
5. **Select Analysis Services**:
   * On the "Feature Selection" page, select "Analysis Services" and click "Next".
6. **Server Configuration**:
   * On the "Server Configuration" page, set the appropriate service account and collation settings, then click "Next".
7. **Select Multidimensional Mode**:
   * On the "Analysis Services Configuration" page, select "Multidimensional and Data Mining Mode".
8. **Configure SSAS Security**:
   * Add users who will have administrative rights to the Analysis Services instance.
9. **Complete the Installation**:
   * Follow the remaining prompts to complete the installation. Once the installation is complete, you will have a new SSAS instance running in Multidimensional mode.

**Update Visual Studio to Use the New SSAS Instance**

1. **Open Your Visual Studio Project**:
   * Open the project containing your cube.
2. **Update Deployment Server**:
   * Right-click on your project in Solution Explorer and select "Properties".
   * Go to the "Deployment" tab.
   * Change the "Server" property to the new SSAS instance name you specified during installation (e.g., YourServerName\SSAS\_Multidimensional).
3. **Deploy the Cube**:
   * Right-click the project and select "Deploy". This will deploy the cube to the new SSAS instance running in Multidimensional mode.
4. **Process the Cube**:
   * After deploying, right-click the cube and select "Process". Ensure the processing completes without errors.

Second Encountered Issue: Internal error: The operation terminated unsuccessfully. The following system error occurred: The user name or password is incorrect.

Solution: I opened the data source, I went to impersonation information, and edit the login detail by providing my email address (which is the admin) and my password. In conclusion, set the impersonation information tab within the data source designer dialog box with the admin credential for the appropriate privilege.

Lab 9 Time series and Excel

1. What does the time series wizard do?

The Time Series Wizard in SQL Server Analysis Services (SSAS) is a feature designed to simplify the creation and configuration of time series models, particularly when working with data mining projects. Here's a detailed look at its functionalities and usage:

### Purpose of the Time Series Wizard

The Time Series Wizard helps in setting up time series models that forecast future values based on historical data. This is particularly useful for scenarios like sales forecasting, inventory management, financial forecasting, and other applications where predicting future trends based on past data is crucial.

### Key Features and Functions of the Time Series Wizard

1. \*\*Data Selection\*\*:

- The wizard allows you to select and prepare the data that will be used for the time series analysis. This includes choosing which data columns (attributes) to analyze.

2. \*\*Algorithm Selection\*\*:

- It guides you in choosing the appropriate algorithm for your time series analysis. SQL Server supports multiple algorithms for time series, but the most commonly used is Microsoft Time Series Algorithm.

3. \*\*Model Configuration\*\*:

- You can configure various parameters of the model, such as the number of time buckets to forecast, handling of missing values, periodicity of the data (e.g., daily, monthly, yearly), and whether to include cross-predicted columns for multivariate time series.

4. \*\*Automated Model Creation\*\*:

- The wizard automates the process of creating the data mining structure and the mining model, reducing the need for manual setup and configuration.

5. \*\*Model Validation and Testing\*\*:

- It provides tools and capabilities to validate and test the model against historical data to assess its accuracy and effectiveness. This often involves splitting the data into training and testing sets to evaluate the model's predictive power.

6. \*\*Visualization and Analysis\*\*:

- Once the model is created, the wizard can generate visualizations and outputs that help in understanding the results, such as line charts of actual vs. predicted values over time.

### Using the Time Series Wizard

To use the Time Series Wizard in SQL Server Analysis Services, follow these general steps:

1. \*\*Open SQL Server Data Tools (SSDT)\*\*:

- Launch the SSDT environment where your SSAS project is located.

2. \*\*Create a New Data Mining Project\*\*:

- Start a new data mining project and add a data source that contains your historical time-series data.

3. \*\*Launch the Wizard\*\*:

- Right-click on the 'Mining Structures' node in your project and select 'New Mining Structure'. Then choose the 'Use the wizard to create a new mining structure' option and select the 'Time Series Model'.

4. \*\*Follow the Wizard Steps\*\*:

- The wizard will guide you through selecting your data source view, specifying the key time column, choosing the predictable column(s), and configuring the algorithm and its parameters.

5. \*\*Deploy and Process the Model\*\*:

- After configuring the model, deploy it to your Analysis Services server and process the model to generate predictions.

6. \*\*Analyze the Results\*\*:

- Use the built-in viewers in SSDT to analyze the prediction results and validate the model's performance.

The Time Series Wizard is a powerful tool that significantly eases the process of developing and deploying predictive models in SSAS, especially for users who may not be experts in data mining or statistical analysis.

1. What is the importance of the dimension usage tab in SSAS?

The Dimension Usage tab in SQL Server Analysis Services (SSAS) plays a critical role in configuring how dimensions are related to measure groups in a cube. Its importance is multi-fold, affecting the overall design, functionality, and performance of OLAP cubes. Here's a detailed breakdown of its significance:

### Key Aspects of the Dimension Usage Tab in SSAS

1. \*\*Defining Relationships\*\*:

- The Dimension Usage tab is where you define and manage the relationships between dimensions and measure groups within a cube. These relationships are crucial for correctly aggregating and slicing data when executing queries.

2. \*\*Types of Relationships\*\*:

- SSAS allows for several types of relationships in the Dimension Usage tab, including:

- \*\*Regular Relationship\*\*: This is the most common type, where a dimension is directly related to a measure group via a foreign key.

- \*\*Fact Relationship\*\*: Used when a dimension attribute is derived directly from a measure group table (fact table).

- \*\*Many-to-Many Relationship\*\*: Complex and allows a dimension to relate to a measure group through another intermediate fact table, useful for scenarios like classifying sales into multiple categories.

- \*\*Referenced Relationship\*\*: Used when there is an indirect relationship between a dimension and a measure group through another dimension.

- \*\*Data Mining Relationship\*\*: Used for integrating data mining models directly into the cube.

3. \*\*Granularity\*\*:

- The tab allows you to specify the granularity of each relationship, which defines at what level of detail the dimension attributes relate to the measures in the measure group. Correct granularity settings are essential to ensure accurate results in cube queries.

4. \*\*Performance Optimization\*\*:

- Proper configuration in the Dimension Usage tab can significantly impact the performance of cube processing and query execution. Optimally configuring how dimensions relate to measure groups can lead to more efficient storage and faster retrieval of data.

5. \*\*Semantic Layer of the Cube\*\*:

- It provides a clear, visual representation of how different parts of the business data model interact. This makes it easier to understand the cube structure, especially for complex cubes with multiple dimensions and measure groups.

6. \*\*Impact on MDX Queries\*\*:

- The relationships and configurations set in the Dimension Usage tab directly influence how Multidimensional Expressions (MDX) queries are interpreted and executed by SSAS. They determine how dimensions filter and slice the data in the cube, which in turn affects the accuracy and relevance of query results.

### Practical Usage of the Dimension Usage Tab

When you open a cube in SQL Server Data Tools (SSDT), the Dimension Usage tab presents a matrix of dimensions and measure groups. Here you can:

- \*\*Add or remove dimensions\*\*: Adjust which dimensions are used in the cube.

- \*\*Edit relationships\*\*: Change how dimensions relate to each measure group.

- \*\*View and modify properties\*\*: Such as relationship type and granularity, to fine-tune cube behavior.

### Conclusion

The Dimension Usage tab is foundational for ensuring that a cube's data model is logically correct and performs well. It allows cube designers to accurately map the real-world relationships of the data into the multidimensional model, ensuring that users can analyze data in a way that is consistent with business rules and expectations. Proper use of this tab is crucial for designing efficient and effective analytical solutions in SSAS.

1. What does reconnect do in the SSAS browser tab?

In SQL Server Analysis Services (SSAS), the "Reconnect" button found in the browser tab within SQL Server Data Tools (SSDT) or SQL Server Management Studio (SSMS) plays a specific and useful role, especially during the cube development and testing phases. Here's what it does and why it's important:

### Functionality of the "Reconnect" Button

1. \*\*Restores Connection\*\*:

- The primary function of the "Reconnect" button is to re-establish a connection to the SSAS database. This is particularly useful if the session has timed out, the connection has been lost, or the server was temporarily unavailable.

2. \*\*Refreshes Metadata\*\*:

- When you make changes to the structure of a cube—such as adding or removing dimensions, changing measures, or modifying properties—these changes might not immediately reflect in an existing session. Clicking "Reconnect" ensures that the browser tab reflects the most current state of the cube's metadata.

3. \*\*Updates Data Views\*\*:

- In scenarios where data or metadata might have been updated since the last connection (like after processing the cube or applying structural changes), using "Reconnect" ensures that you are querying the latest data. This helps avoid inconsistencies and errors that might occur due to querying stale data.

### Practical Use Cases

- \*\*After Processing\*\*: If you've just processed a cube or dimension and want to test the updates, clicking "Reconnect" ensures that the browser is querying the latest version of the cube.

- \*\*Error Recovery\*\*: If there are connectivity issues or errors during a session, such as network interruptions or server restarts, "Reconnect" can restore the session without the need to restart SSMS or SSDT.

- \*\*Development and Testing\*\*: During cube development, you often make changes to cube definitions. "Reconnect" is handy for quickly refreshing the connection to load these changes in the browser for immediate testing.

### How to Use "Reconnect"

When using the SSAS browser tab:

1. Open the cube you want to browse in either SSDT or SSMS.

2. If you suspect that the data or cube structure displayed is outdated, or if you have lost the connection, click the "Reconnect" button usually located at the top of the browser tab.

3. The tool will re-establish the connection to the SSAS server, refreshing both the data and the metadata in the browser.

The "Reconnect" feature ensures that the cube data you are viewing and interacting with in the browser is current and accurately reflects the underlying database state. This functionality is essential for effective development workflows and accurate data analysis within SSAS.

1. What is the importance of hierarchical dimensions?

Hierarchical dimensions in SQL Server Analysis Services (SSAS) and other OLAP (Online Analytical Processing) environments play a crucial role in organizing data in a way that reflects real-world relationships and enhances data analysis capabilities. They are essential for providing context, detail, and a structured way to drill down into data during analysis. Here’s an overview of their importance and benefits:

### Importance and Benefits of Hierarchical Dimensions

1. \*\*Natural Data Representation\*\*:

- Hierarchical dimensions represent data in a way that mirrors real-life hierarchies. For example, a geographical hierarchy might consist of Country > State > City, while a time hierarchy could be Year > Quarter > Month > Day. This structure is intuitive for users, making data exploration and analysis more accessible and meaningful.

2. \*\*Improved Query Performance\*\*:

- Hierarchies help optimize query performance through a feature called "attribute relationships." Properly defined relationships between levels in a hierarchy can help the SSAS engine optimize storage and efficiently retrieve data, especially during aggregations and calculations.

3. \*\*Enhanced Data Analysis\*\*:

- Hierarchies allow users to perform detailed data analysis at various levels of granularity. Analysts can easily drill down, drill up (roll up), or slice and dice the data across different levels of the hierarchy to discover trends and patterns that are not immediately visible at a single level.

4. \*\*Facilitates Aggregations\*\*:

- In SSAS, hierarchies help in defining and pre-calculating aggregations at each level of the hierarchy. This pre-calculation speeds up the retrieval of aggregated data during analysis, as the server can serve pre-computed totals rather than recalculating from the lowest level each time.

5. \*\*Business Intelligence (BI) Tools Compatibility\*\*:

- Hierarchical dimensions are highly compatible with various BI tools and reporting platforms. They provide a structured way to interact with data, enabling dynamic reports and dashboards that adjust according to the selected hierarchy level.

6. \*\*Improved User Experience\*\*:

- For end users, particularly those less familiar with the underlying data structures, hierarchies provide a straightforward and familiar way to navigate data. This improves the user experience, making analytical applications more user-friendly.

### Practical Implementations and Considerations

- \*\*Dimension Design in SSAS\*\*:

- When designing a dimension in SSAS, it’s important to define attribute relationships correctly to ensure that the hierarchy behaves as expected. This involves setting up relationships that define how each level of the hierarchy relates to the others, ensuring logical data roll-up and consistent query responses.

- \*\*User Interface and Interaction\*\*:

- In the user interface of BI tools, hierarchical dimensions are often represented in a tree-like structure, allowing users to expand or collapse levels as needed for detailed analysis or high-level overviews.

- \*\*Security and Data Access\*\*:

- Hierarchies also play a role in implementing data security. Security roles in SSAS can be defined to limit data access at certain levels of a hierarchy, allowing sensitive data to be protected while still providing useful insights at higher aggregation levels.

### Conclusion

Hierarchical dimensions are foundational in data warehousing and business intelligence for modeling complex data structures in an understandable, efficient, and scalable manner. They enhance both the analytical power and performance of OLAP solutions, making them indispensable in many data-intensive applications.

1. How to use Excel to see SSAS cube data?

Using Excel to view and analyze data from an SQL Server Analysis Services (SSAS) cube is a popular choice because of its familiar interface and powerful analytical capabilities. Here’s a step-by-step guide on how to connect Excel to an SSAS cube and start analyzing your data:

### Step-by-Step Guide to Connect Excel to an SSAS Cube

#### Step 1: Open Excel

Start by opening Microsoft Excel. You can use any recent version of Excel, as they all support connecting to an SSAS database.

#### Step 2: Establish a Data Connection

1. \*\*Go to the Data Tab\*\*:

- In Excel, click on the "Data" tab on the Ribbon.

2. \*\*Get External Data\*\*:

- Click on "Get Data" (in newer versions of Excel) or "From Other Sources" (in older versions).

- Select “From Database” and then choose “From Analysis Services”.

![Get Data from Analysis Services](https://docs.microsoft.com/en-us/sql/reporting-services/report-data/media/ssrs-2016-report-data-data-source-properties-analysis-services-connection-string.png?view=sql-server-ver15)

3. \*\*Enter Server Details\*\*:

- In the dialog box that appears, enter the name of the server where the SSAS cube is hosted.

- Choose how you want to authenticate (Windows Authentication is common in corporate environments). Enter your credentials if required.

4. \*\*Select Database and Cube\*\*:

- After connecting to the server, Excel will display a list of available databases and cubes. Select the database and then the cube you wish to analyze.

#### Step 3: Import Data

1. \*\*Choose Import Type\*\*:

- You can choose to import data as a PivotTable Report or PivotChart, which are ideal for interactive analysis.

- Select "PivotTable Report" for a flexible, drag-and-drop interface within Excel.

2. \*\*Place PivotTable\*\*:

- Decide where you want to place your PivotTable in your Excel workbook—either in a new worksheet or an existing one.

#### Step 4: Analyze Data

1. \*\*Use PivotTable Fields\*\*:

- The PivotTable Field List pane will show the cube’s dimensions, measures, and hierarchies.

- Drag and drop fields to different areas of the PivotTable to start analyzing your data. You can place dimensions in the Rows or Columns, measures in the Values, and so forth.

2. \*\*Filter and Slice\*\*:

- Use slicers or filters to refine your data analysis. Slicers provide buttons that you can click to filter data, while the regular filters offer more traditional dropdown menus to select specific elements.

3. \*\*Create Visualizations\*\*:

- If you chose to use a PivotChart, configure your chart by arranging fields and formatting the chart for better visualization.

#### Step 5: Refresh Data

- \*\*Refresh as Needed\*\*:

- Whenever the data in the SSAS cube updates, you can refresh your Excel report by right-clicking on the PivotTable and selecting "Refresh". This ensures your analysis reflects the most current data.

#### Step 6: Save and Share

- \*\*Save Your Workbook\*\*:

- Save your Excel workbook as usual. You can also share it with others who have access to the same SSAS cube.

### Conclusion

Connecting Excel to an SSAS cube allows business analysts and other users to leverage Excel’s powerful analytical tools to explore and visualize data stored in a multidimensional format. This process enables detailed data analysis and decision-making based on large and complex datasets managed by SSAS.

LAB 11 What are transactions and checkpoints in SSIS?

I did not practically do the steps because there was a need to modify my file. But it is self-sufficient to watch the video to understand.

1. What is the importance of transactions?
2. Which component helps us to enable transactions in SSIS?
3. Are transactions enabled by default in SSIS
4. What is the difference between Required, Supported and Not supported?
5. What are checkpoints?
6. What is written inside the checkpoint file?
7. Explain Never, IfExists and Always on Checkpoint usage?
8. What are GUID’s ?
9. When does the checkpoint file get deleted?
10. Is the checkpoint tracked at row level or component level?

LAB 12 Simple SSRS report & implementing Matrix, Tabular, Parameters, Sorting.

ERROR

TITLE: Microsoft Analysis Services

------------------------------

Query (1, 10) Column [Sales Person Name] is part of composite key, but not all columns of the composite key are included in the expression or its dependent expression.

------------------------------

BUTTONS:

OK

In investigating the error, a need to understand the structure of the dimension is significant. After questioning co-pilot on the matter, it seems the error was raised due to the fact I did not include Sales Person ID along with Sales Person Name within the query builder (hypothetically). After including the Sales Person ID, the query was successful. From this previous test, I concluded the issue was mostly based on the dimension architecture that does not allow me to query the sales person name without the primary key (which is the sales person ID).

Note: The lab lecturer managed to query the sales person name without the sales person ID successfully.

Problem:

* **Is my dimension architecture the same as the lab lecturer's?**
* **Why in SSRS, do my query builders require me to include sales person ID?.**
* **Why in the cube, querying the sale person name without the sales person ID is successful?**
* **Is there any settings to deactivate this error?**

Analysis:

* I need to understand the difference of his dimension architecture with mine to solve the issue.
* I have to review lab 9 to understand the cause. Min:25:00

Findings:

1. **Is my dimension architecture the same as the lab lecturer's?**

My dimension architecture is the same in Sql server Analysis service, due to the fact country and product table are query tables in the data source view. In database engine, however, the primary key constraints were unable on the country tables and product tables to show the use of SCD for audit purposes (Creating a column to track changes in each records), which naturally led to duplication. Differences to keep in mind are (For future references):

Product 1 in my fact customer table does not exist. When watching the lab 9, the lecturer faced some difficulties (31:50) concerning product 1 which was in the fact customer table (real table) but not included in product table (query table). He solved it by setting dateExpired to null so that it can be included to query inside the product query table (FMI, the query takes only records where dateexpired is null).

1. **Why in SSRS, do my query builders require me to include sales person ID?.**

Because the connection type was set to DAX instead MDX query langage: Connection Type: Additionally, consider the connection type you’re using. Sometimes, selecting the wrong connection type (e.g., MDX instead of DAX) can lead to this error1.

1. **Why in the cube, querying the sale person name without the sales person ID is successful?**

Because the cube uses MDX as a query language.

1. **Is there any settings to deactivate this error?**

Yes, setting MDX language in the query builder interface.

Query builder from matrix. Connection failed due to credential.An alternative approach would be to use SQL to query the tables.

**Differentiate between Matrix and Tabular report ?**

In SQL Server Reporting Services (SSRS), the main differences between Matrix and Tabular reports are:

Matrix Report: Also known as a cross-tab report, it allows for both row and column grouping. It’s dynamic, meaning the number of columns can change based on the data. It’s used to display summarized data.

Tabular Report: Displays detail data with a fixed number of columns and allows only row grouping. It’s more static compared to a matrix report.

In essence, if you need to summarize data with dynamic columns based on the data content, use a Matrix report. If you need to list detailed records with specific columns, use a Tabular report1

**Explain the importance of the three sections in a report (Page, Group and details).**

In report design, the three sections—Page, Group, and Details—serve distinct purposes:

* **Page Section**: Controls the layout of content that repeats on every page, such as headers and footers. It’s crucial for including report titles, page numbers, or date stamps that provide context and ease of navigation.
* **Group Section**: Allows you to organize data into meaningful categories or groups. This is important for summarizing data and creating subtotals or counts based on a specific field.
* **Details Section**: The core area where individual records are displayed. It’s essential for showing the granular level of data that makes up the report.

Each section plays a role in structuring the report to make it informative, easy to read, and navigable.

**What is the importance of RDL file?**

An RDL file, which stands for Report Definition Language file, is crucial in SQL Server Reporting Services (SSRS) because:

* It is an XML representation of a report definition that contains data retrieval and layout information for a report.
* It defines the report’s structure, including the layout, data sources, datasets, and other parameters necessary for report execution.
* RDL files enable developers and report authors to design, modify, and manage reports effectively.

[Understanding and utilizing RDL files is essential for creating accurate and comprehensive reports in SSRS1](https://learn.microsoft.com/en-us/sql/reporting-services/reports/report-definition-language-ssrs?view=sql-server-ver16).

**What is the internal format of RDL file?**

The internal format of an RDL (Report Definition Language) file is XML-based. It contains all the necessary information about how a report should be rendered, including:

* **Report Data**: Connection strings, queries, fields, etc., to obtain the data to display in the report.
* **Calculations**: Grouping, sorting, filtering, aggregations, scalar formulas, and other transformations on the report data.
* **Layout**: Design elements like tables, charts, images, and text boxes.
* **Parameters and Expressions**: Dynamic elements that control report behavior or appearance based on user input or other variables.

[Each RDL file consists of various XML elements that define these aspects of a report in SSRS1](https://shallbd.com/understanding-the-rdl-format-in-ssrs-a-comprehensive-guide/)[2](https://www.sqlines.com/articles/ssrs/rdl_manually).

Explain the importance of report data tool bar?

The report data toolbar is an essential feature in SQL Server Reporting Services (SSRS) for several reasons:

* It provides **navigation** and **zoom** controls, allowing users to easily move through the report and focus on areas of interest.
* The toolbar includes **refresh**, **search**, **export**, and **print** functionalities, enhancing the interactivity and usability of the report.
* [It displays **report parameters**, enabling users to select values that control the report data or appearance, thus customizing the report for specific needs or audiences1](https://learn.microsoft.com/en-us/sql/reporting-services/html-viewer-and-the-report-toolbar?view=sql-server-ver16)[2](https://learn.microsoft.com/en-us/sql/reporting-services/report-data/report-data-ssrs?view=sql-server-ver16).

Overall, the report data toolbar improves the user experience by providing tools to interact with, customize, and manage the displayed report data effectively.

**What does dataset and datasource do in report data?**

In the context of report data, particularly in SQL Server Reporting Services (SSRS):

* **Data Source**: It contains the details about the database server you will be connecting to, including the login and the database to use. It’s essentially the connection information that tells the report where to fetch the data from.
* **Dataset**: It contains the specific query that will be used to fetch data for a particular report. [A dataset represents the result set from running a query command on a data source, including the fields (columns) and data (rows) that will be used in the report1](https://mindmajix.com/ssrs/data-sources-and-data-sets-in-reporting-services)[2](https://learn.microsoft.com/en-us/sql/reporting-services/report-data/report-datasets-ssrs?view=sql-server-ver16).

Datasets and data sources are fundamental components in SSRS that work together to retrieve and present data in reports.

**How to create parameters in SSRS?**

Creating parameters in SQL Server Reporting Services (SSRS) involves the following steps:

1. In the **Report Data** pane, expand the **Datasets** folder, right-click your dataset, and select **Query**.
2. Modify your query to include a **WHERE** clause with a parameter.
3. Run the query to ensure it prompts for a parameter value.
4. The report will automatically create a parameter under the **Parameters** folder in the **Report Data** pane.
5. [You can then set properties for the parameter, such as name, data type, available values, and default values1](https://learn.microsoft.com/en-us/sql/reporting-services/tutorial-add-a-parameter-to-your-report-report-builder?view=sql-server-ver16).

Parameters allow users to filter and customize the report data dynamically.

**How can we convert the parameter to a dropdown?**

To convert a parameter to a dropdown list in SSRS:

1. In the **Report Data** pane, right-click on the parameter and select **Parameter Properties**.
2. Under **Available Values**, select **Get values from a query**.
3. Choose the dataset that will provide the values for the dropdown and select the value and label fields.
4. [If you want to allow multiple selections, check the **Allow multiple values** checkbox1](https://shristi08.medium.com/ssrs-report-with-dropdown-7a664e8ae849)[2](https://stackoverflow.com/questions/4851800/using-dropdown-parameters-in-ssrs-report).

This will create a dropdown list for users to select from when running the report.

**How to do sorting and interactive sorting in SSRS?**

To implement sorting and interactive sorting in SSRS:

**Sorting:**

1. In the **Report Data** pane, select your dataset and add a **Sort by** expression to the dataset properties.
2. You can also add sorting to a tablix data region by editing the **Tablix Properties** and adding a sort expression.

**Interactive Sorting:**

1. Right-click on a column header in the tablix data region and select **Textbox Properties**.
2. Go to the **Interactive Sorting** tab.
3. Enable interactive sorting and specify the field to sort by.
4. [Users can then click on the column header to sort the data in ascending or descending order1](https://learn.microsoft.com/en-us/sql/reporting-services/report-design/interactive-sort-report-builder-and-ssrs?view=sql-server-ver16)[2](https://www.mssqltips.com/sqlservertip/3599/interactive-sorting-for-a-sql-server-reporting-services-report/).

This allows users to dynamically sort report data without modifying the dataset or report design.

Differentiate between expression and textbox properties.

In SSRS:

**Expressions:**

* Begin with an equal sign (=).
* Used to calculate values, set property values, and define conditions.
* Can include built-in functions, custom code, and variables.

**Textbox Properties:**

* Define the appearance and behavior of a textbox.
* Include settings for font, color, size, alignment, and visibility.
* [Can contain expressions to dynamically set these properties1](https://medium.com/@smitagudale712/expression-and-textbox-in-sql-server-reporting-services-ssrs-41b1a1c1ab6a)[2](https://learn.microsoft.com/en-us/sql/reporting-services/report-design/expressions-report-builder-and-ssrs?view=sql-server-ver16).

Expressions are used within textbox properties to control content and formatting dynamically.

**How is the IIF syntax of expression written?**

The IIF function syntax in SSRS is written as follows:

IIF(condition, true\_value, false\_value)

* **condition**: A Boolean expression that evaluates to either true or false.
* **true\_value**: The value returned if the condition is true.
* [**false\_value**: The value returned if the condition is false1](https://hatchjs.com/ssrs-iif-multiple-conditions/)[2](https://www.sqliz.com/sqlserver-ref/iif/).

For example, to set text style based on a condition:

=IIF(Fields!location.Value = "CA", "Bold", "Italic")

[This sets the text style to bold if the location is CA, otherwise it sets it to italic](https://hatchjs.com/ssrs-iif-multiple-conditions/)[3](https://www.educba.com/ssrs-iif/).

**How to implement running total on a table?**

To implement a running total in a table in SSRS, you can use the RunningValue function. Here’s the syntax:

=RunningValue(Fields!YourFieldName.Value, Sum, "Scope")

* **Fields!YourFieldName.Value**: The field for which you want to calculate the running total.
* **Sum**: The aggregate function (in this case, we’re summing the values).
* **“Scope”**: The scope can be a dataset, group, or data region name.

For example, if you want to calculate the running total of sales in a dataset named SalesDataSet, you would write:

=RunningValue(Fields!SalesAmount.Value, Sum, "SalesDataSet")

[This expression would be placed in the textbox where you want the running total to appear1](https://www.tutorialgateway.org/calculate-running-total-in-ssrs-grouped-report/).

**How to achieve cascading drop downs?**

To achieve cascading drop-downs in SSRS, you need to:

1. Create datasets for each drop-down list, with the child dataset including a parameter that references the parent selection.
2. Define report parameters that will be used by the drop-down lists.
3. Set up the report parameters to get their available values from the appropriate datasets.
4. Ensure the child parameter’s available values are filtered based on the parent parameter’s selection.

For example, if you have two parameters, @Country and @City, where @City depends on the selection of @Country, you would:

* Create a dataset for @Country that retrieves all countries.
* Create a dataset for @City that includes a query parameter for the country selection.
* Configure the @City report parameter to use the @Country parameter to filter its available values.

[This way, when a country is selected, the city drop-down will automatically update to show only cities from that country1](https://www.mssqltips.com/sqlservertip/3466/cascaded-parameters-in-sql-server-reporting-services/)[2](https://blogs.perficient.com/2022/08/01/how-to-create-cascading-parameters-in-reporting-services-ssrs/)[3](https://learn.microsoft.com/en-us/sql/reporting-services/report-design/add-cascading-parameters-to-a-report-report-builder-and-ssrs?view=sql-server-ver16).

Lab 13: Using data profiling task to check data quality.(SSIS)