Unanet
University

UNANET AE:

Creating Analytic Models using Query Builder

PARTICIPANT GUIDE

Last Revised March 2024

TABLE OF CONTENTS

ABOUT THIS PARTICIPANT GUIDE

This document serves as a supplement to the training courses offered through Unanet University. Participants should use this workbook to complete exercises in the student Unanet system while attending the course. Additionally, participants should record notes and actions for reference when working within their own Unanet AE system.

Symbols used in this guide

- 🖱 Activity
- 💻 Demo
- 🗺 Tour
- ⚠ Important Note
- ? Question

Unanet Contact Information

- VirtualUU@unanet.com

Send questions here for information about courses, course schedule, unanetuniversity.com, certifications, and enrollments. This is a mailbox that is monitored daily.

Support Portal

To improve your experience in accessing our support resources, all Knowledge Center articles, and FAQs have been relocated to the Unanet Support Portal, located at support.unanet.com. You will need a Support ID to access the portal, which you can acquire using the steps below:

> To obtain a login to the Unanet support portal or to create a new ticket, send an email to support@unanet.com with "Support Portal ID request" in the Subject line.
>
> The following conditions must be met to be granted a support account:
>
> - The "from" address must match the domain of your company.
> - It must NOT be a generic email account (e.g., it cannot be from gmail.com, aol.com, or msn.com).

The company must have an active support agreement with Unanet.

COURSE INTRODUCTION

In this course, you will gain a comprehensive understanding of the Unanet AE Query Builder. You will learn how to get up and running using Query Builder, to export queries as a .sql file, or use them to build an Analytic Model. You will also learn how to employ them in custom Analytic Dashboards and reports.

Learning Objectives

After this session of **Unanet AE: Creating Analytic Models Using Query Builder**, participants will be able to:

- Explain the query data process.
- Summarize how to build queries using the Query Build applet.
- Summarize how to build queries the Analytic Dashboard Designer.

LESSON 1: QUERY BUILDER OVERVIEW

Learning Objectives

Explain the query data process.

- Describe a query developer’s responsibilities.
- Recall necessary permissions for building queries.
- Identify the data sources for building queries in Unanet A/E.

Overview

Querying data with an analytic dashboard provides a powerful means to extract actionable information from complex datasets, empowering users to unlock the full potential of their data resources.

Query Designer Roles and Responsibilities

A query designer for an analytic dashboard is an individual responsible for designing, building, and optimizing the queries that retrieve and process data for the dashboard's visualizations. This person typically has expertise in data querying, data modeling, and understanding the underlying data sources.

The role of a query designer for an analytic dashboard may involve the following responsibilities:

- **Data Analysis and Requirements Gathering:** The query designer works closely with stakeholders, such as business analysts or dashboard users, to understand their data requirements. They analyze the desired metrics, KPIs, and dimensions that need to be visualized and make informed decisions on how to structure and retrieve the relevant data.
- **Query Design and Development:** The query designer translates the data requirements into actionable queries. They use query design tools or programming languages, such as SQL (Structured Query Language) or specialized query builders, to construct queries that retrieve the necessary data from the underlying data sources. They optimize the queries for performance and efficiency, considering factors such as data volume, complexity, and response time requirements.
- **Data Transformation and Modeling:** The query designer may be involved in data transformation tasks, such as aggregating, filtering, joining, or creating

calculated fields. They understand the data model and relationships between tables to design effective queries that deliver accurate and meaningful results for visualization.

- **Query Optimization:** The query designer optimizes queries to improve performance and minimize execution time. They analyze query execution plans, identify bottlenecks, and suggest optimizations like index creation, query rewriting, or caching strategies to enhance query performance.
- **Data Validation and Testing:** The query designer validates query results to ensure accuracy and data integrity. They perform data quality checks, verify data against the expected outcomes, and troubleshoot any discrepancies or anomalies. They also conduct testing and validation of queries in different scenarios or edge cases to ensure reliable data retrieval.
- **Collaboration and Communication:** The query designer collaborates with other members of the dashboard development team, such as dashboard designers, data engineers, or business analysts. They communicate effectively to understand requirements, gather feedback, and iterate on query design based on changing needs or evolving data sources.
- **Documentation and Documentation Maintenance:** The query designer maintains documentation of queries, data sources, and transformations. They document the query logic, data mapping, and any relevant information necessary for future reference or troubleshooting.

In summary, a query designer for an analytic dashboard plays a crucial role in designing and optimizing queries that retrieve and transform data for visualization. They ensure the accuracy, efficiency, and performance of the queries to deliver valuable insights through the dashboard.

Query Writer Roles and Responsibilities

The role of a query writer for an analytic dashboard involves writing, optimizing, and maintaining queries that retrieve and transform data for visualization. They play a critical role in ensuring efficient data retrieval, accuracy, and optimal performance of the dashboard.

The query writer is responsible for writing queries that retrieve the required data from underlying data sources. They have expertise in query languages like SQL (Structured Query Language) or other query builders specific to the dashboard platform. They understand the data model and structure of the underlying databases or systems and write queries that extract the necessary data efficiently. This responsibility distinguishes the Query Writer from the Query Designer.

Configuration and Permissions

Unanet AE relies on permissions to grant access to write and design queries for Analytic Dashboards. You will need permission to the following applets that are involved with Query building:

- Analytic Dashboard Designer
- Analytic Dashboards
- Dashboard Queries Manager
- Query Builder

Data Sources

Each Analytic Model relies on a data source for the data it contains. Analytic Dashboards support both native (Unanet A/E) and third-party data sources - selected when building the design. This means that, in addition to your Unanet A/E data, you can now visualize outside data across other areas of business, making Unanet A/E a central hub for business analysis.

Analytic Models (Dashboard Query Manager Applet)

Analytic Models (or queries) are used to feed dashboards designed with the Analytic Dashboard Designer. Generally, models are written for performing business analysis and should be written with the end user in mind (e.g. using decipherable return names, etc.). In addition to the query, you can leverage Parameters, Field/List Descriptions and Permissions.

While users with SQL expertise can build an Analytic Model by writing the SQL query from scratch, the Query Builder makes building Analytic Models accessible to users who are unfamiliar with writing SQL.

There are two Query Builders within Unanet A/E. One is the Query Builder Utility, which allows users to build SQL Queries using available datasets and save them as Analytic Models. There is also a built-in Query Builder within the Analytic Dashboard Designer, which builds local queries using available tables and views. Queries built using the built-in Query Builder are available only for use on a single dashboard.

Check Your Understanding

What is the purpose of querying data with an analytic dashboard?

a) To write the programming language
b) To extract actional information from complex datasets
c) To structure visualizations

Refer to Appendix A for answers to the Check Your Understanding questions.

LESSON 2: BUILDING QUERIES IN THE QUERY BUILDER APPLET

Learning Objectives

Summarize how to build queries using the Query Build applet.

- Identify the benefits of the Query Builder applet.
- Explain the role of the Datasets in the Query Builder utility.
- Define the components of the Query Builder.
- Summarize the functionality of SQL statements within Query Builder.
- Describe how to manage and edit a dashboard query.

Purpose and Benefits

The Query Builder applet provides you with a visual interface for constructing queries, dynamically assembling them for application in Analytic Dashboard Designer and Custom Reports modules.

Even if you're not proficient with SQL, this tool simplifies the process of building queries and Analytic Models, making it accessible to all users.

Moreover, it permits the exporting of your query either as a .sql file or as an Analytic Model directly within Unanet A/E. This exported query can then be utilized in the Analytic Dashboard Designer and Custom Reports applets.

Datasets and Fields

Select a Dataset to produce a list of Available Fields within that dataset. Fields to be used in the query are selected from the Available Fields list. Each query can only reference one dataset.

- **Datasets:** Lists available data sets. Each data set contains a list of available fields that can be selected for use in the query.
- **Available Fields:** Displays a list of fields per dataset. Fields can be filtered by typing in the search box and selected for use in the query by checking the box next to each field. Field descriptions are displayed at the bottom of the pane when clicking the field.
- **Clear Selected:** Clears all sections made in the Datasets field list.

Overview of Query Builder Components

The Query Builder applet offers a robust set of features for manipulating and extracting insights from the dataset. It comprises several essential sections:

- **Selected Fields:** Displays a running list of selected fields from the dataset.
- **Calculated Columns:** Adds calculated fields using selected fields from the dataset. Calculated columns use standard SQL formatting. These calculations are added as new fields in the query using the specified title.
- **Filters (requires SQL formatting):** Add filters to the query using selected fields.
- **Toolbar Options**
    - **Build:** Generates the query based on the dataset selections, filters, and calculated columns. Selecting this will override previous build.
    - **Build & Execute:** Generates the Query as described above, then runs the SQL query as displayed in the Editor. **Note:** SQL Queries directly affect the Unanet database. Use caution with this functionality.
    - **Execute:** Executes the query in the Editor only. **Note:** SQL Queries directly affect the Unanet database. Use caution with this functionality.
    - **Copy:** Copies the text generated in the query editor.
    - **Clear:** Clears the generated query.
    - **Export Query:** Exports the text of the query to a .sql file.
    - **Save as Model:** Saves the query as an Analytic Model, which can be accessed in the Dashboard Queries manager and used on dashboards and reports.
- **Query Editor Window:** Displays the query once built.
- **Results:** Displays the results of the query once executed from the dataset.

SQL Statements

Simplified SQL statements are built on the following commands: SELECT, FROM and WHERE. Query Builder compiles these as follows:

- **SELECT:** Built from Selected Fields and Calculated Columns based on the dataset
- **FROM:** Dynamically built from the selected Dataset
- **WHERE:** Built from defined Filters

Additional keywords such as GROUP BY and ORDER BY are dynamically built from the dataset. Generated SQL can be further customized as necessary in the editor. Once edited, selecting Rebuild will override any manual adjustments to the query.

Activity 2.1 – Build and Save a Query

In this activity, you will build, execute, and save a query using the various components of the Query Builder.

Activity Steps

Part 1: Select fields from a dataset

1. Select **Utilities > Query Builder**.
2. Select **Project Transactions** in the **Datasets** field.
3. Select the following field check boxes. **Note:** You can use the search field to locate a specific field in the list.
    - Effort Amount
    - Billed Amount
    - Recvd Amount
    - Units (**Note**: This is actual hours)
    - Budget Hours
    - Budget Labor
    - Budget ODC
    - Budget OCC
    - Budget ICC
    - Contract Amount Labor
    - Contract Amount ODC
    - Contract Amount OCC
    - Contract Amount ICC
    - Charge Type
    - Project Code 1
    - Project Code 2
    - Project Code 3
    - Project is Active
    - Project Name 1
    - Project Name 2
    - Project Name 3

Part 2: Add calculated fields to a query

1. Select the **Calculated Columns** tab.
2. Select **Add new calculated field** in the **Field Name** column. A text field displays.
3. Type *Total Budget* in the text field.
4. Verify **Number** is selected in the **Data Type** field.
5. Type the following formula in the Formula (SQL Format).
   Note: Do not include spaces in your formula.

   Budget_Labor+Budget_ICC+Budget_OCC+Budget_ODC

6. Select **Add new calculated field** in the **Field Name** column. A text field displays.
7. Type *Total Contract* in the text field.
8. Verify **Number** is selected in the **Data Type** field.
9. Type the following formula in the Formula (SQL Format).
   Note: Do not include spaces in your formula.

   Contract_Amount_Labor+Contract_Amount_ICC+Contract_Amount_OCC
   +Contract_Amount_ODC

Part 3: Select filters

1. Select the **Filters** tab.
2. Select **Project_Is_Active** in the first row of the **Field** column.
3. Type *1* in the **Value** field.
4. Select **Charge_Type** in the next row of the **Field** column.
5. Type ‘*Billable’* in the **Value** field.
   Note: You must include the single quote marks.

Part 4: Build and execute the query

1. Select the **Build** button in the toolbar above the Editor. The query displays.
2. Select the **Execute** button. The **Results** tab is automatically selected to display the results.
   Note: The quantity of data determines how long the execute process will take.

Part 5: Save the query

1. Select the **Export Query** button in the toolbar above the Editor. The **Save Query** window opens.
2. Type *Project Budget Analysis* in the **File name** field.
3. Select the **Save** button. A confirmation message displays.
4. Select the **OK** button.

Part 6: Save query as a model

1. Select the **Save as Model** button in the toolbar above the Editor. The **Enter Name** window opens.
2. Type *Project Budget Analysis* in the **Enter an Analytic Model** name field.
3. Select the **OK** button. A confirmation message displays.
4. Select the **OK** button.

Manage and Edit a Dashboard Query

Once the query has been saved as an Analytic Model you can assign permissions within the Dashboard Queries Manager utility for Dashboard designers to access the Analytic Model in the Analytic Dashboard Designer. Also in the Dashboard Queries Manager, an **Edit in Query Builder** link opens the query in the Query Builder applet to make any necessary changes. You can also manually edit the SQL code in the editor. This should be done once all edits in the Query Builder are complete, as editing the SQL directly in the Dashboard Queries Manager “breaks” the link to the Query Builder.

Activity 2.2 – Manage the Analytic Model

In this activity, you will assign model permissions to the Analytic Model and add it to a dashboard.

Activity Steps

1. Select **Utilities > Dashboard Queries Manager**.
2. Verify **Analytic Models** is selected in the drop-down field.
3. Double-click **Project Budget Analysis**. The query displays.
4. Select the **Model Permissions** tab.
5. Select the following check boxes in the **Groups** section.
   - Accounting
   - Administrator
6. Select the **Save** button.
7. Select **Utilities > Analytic Dashboard Designer**.
8. Select the **New** button. An **Add Model?** message displays.
9. Select the **Yes** button. The **Analytic Models** window opens.
10. Double-click the **Project Budget Analysis** model. The fields display, including the two calculated fields (Total Budget and Total Contract).
11. Select **Save.** The **Save Dashboard** window opens.
12. Select **Projects** in the **Grouping** field.
13. Type *Project Budget Analysis* in the **Dashboard Name** field. The **Title** field will auto-populate.
14. Select the **Save** button.

Activity 2.3 – Edit the Query

In this activity, you will query from within Dashboard Queries Manager.

Activity Steps

1. Select **Utilities > Dashboard Queries Manager**.
2. Verify **Analytic Models** is selected in the drop-down field.
3. Double-click **Project Budget Analysis**. The query displays.
4. Select the **Edit in Query Builder** link (located under the tabs). The **Query Builder** opens. **Note:** If you make any manual edits to the query in the **Dashboard Queries Manager**, the link will break and no longer be available to select.
5. Select the **Schedule Hours** check box in the **Available Fields** section.
6. Select the **Build** button.
7. Select the **X** for the **Effort_Amount** field to remove it from the query.
8. Select the **Build** button.
9. Select the **Execute** button.
10. Select the **Save as Model** button. A confirmation message displays.
11. Select the **OK** button.
12. Select **Utilities > Analytic Dashboards**.
13. Double-click **Project Budget Analysis** in the **Analytic Dashboards** list.
14. Select **Edit** from the toolbar. The **Analytic Dashboard Designer** applet opens and loads the dashboard. **Schedule_Hours** is now in the fields list and **Effort_Amount** has been removed.

Check Your Understanding

How many datasets can each query reference in the Query Builder?

a) One
b) Two
c) Multiple

Refer to Appendix A for answers to the Check Your Understanding questions.

LESSON 3: BUILDING QUERIES IN THE ANALYTIC DASHBOARD DESIGNER

Learning Objectives

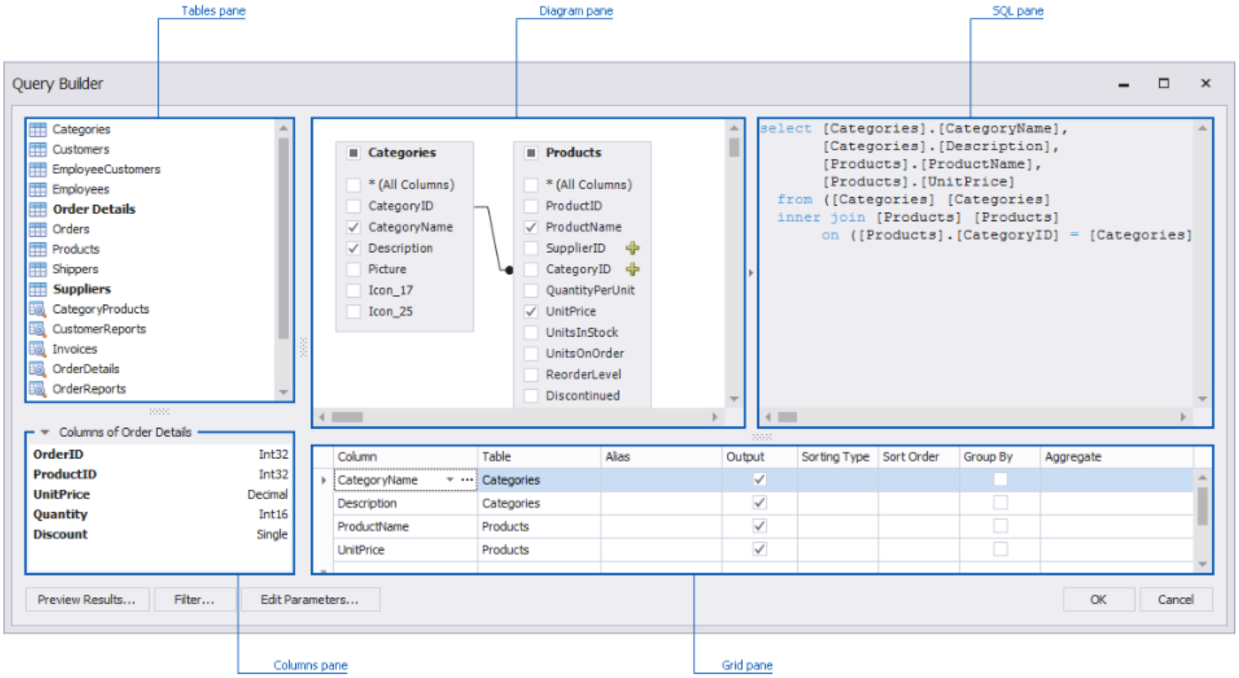
Summarize how to build queries the Analytic Dashboard Designer.

- Identify the benefits of the Query Builder within the Analytic Dashboard Designer applet.
- Explain how to build a query using the Analytic Dashboard Designer applet.

Purpose and Benefits

In addition to building queries in the Query Builder Applet, queries can be built directly inside the Analytic Dashboard Designer. Instead of adding an analytic model, the built-in Query Builder creates a local query, available only to the dashboard for which it is built. This Query Builder uses tables and views to build the SQL Query.

The Data Dictionary Report in the Utilities module provides a list of available tables and fields within each table.



Categories
Customers
EmployeeCustomers
Employees
Order Details
Orders
Products
Shippers
Suppliers
CategoryProducts
CustomerReports
Invoices
OrderDetails
OrderReports

Categories

- □ * (All Columns)
- □ CategoryID
- ☑ CategoryName
- ☑ Description
- □ Picture
- □ Icon_17
- □ Icon_25

Products

- □ * (All Columns)
- □ ProductID
- ☑ ProductName
- □ SupplierID
- □ CategoryID
- □ QuantityPerUnit
- ☑ UnitPrice
- □ UnitsInStock
- □ UnitsOnOrder
- □ ReorderLevel
- □ Discontinued

```
select [Categories].[CategoryName],
       [Categories].[Description],
       [Products].[ProductName],
       [Products].[UnitPrice]
  from ([Categories] [Categories]
  inner join [Products] [Products]
       on ([Products].[CategoryID] = [Categories]
```

Columns of Order Details

OrderID	Int32
ProductID	Int32
UnitPrice	Decimal
Quantity	Int16
Discount	Single

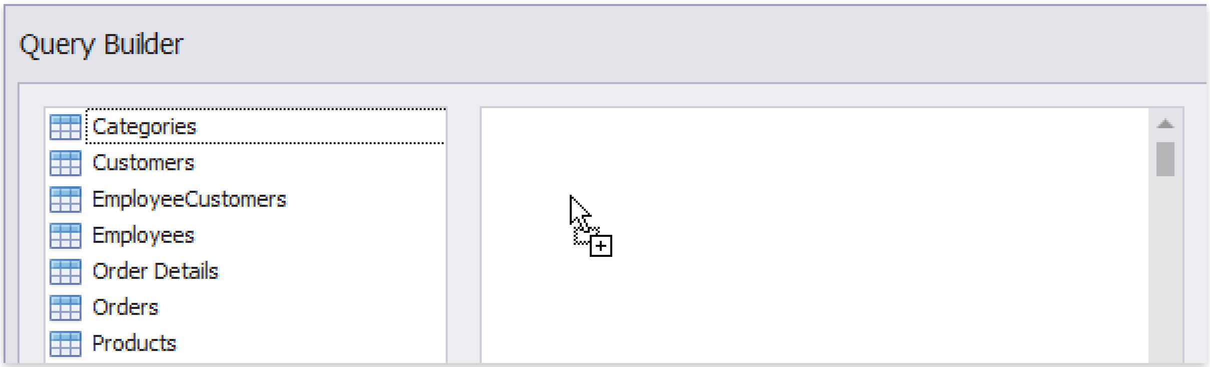
Column	Table	Alias	Output	Sorting Type	Sort Order	Group By	Aggregate
CategoryName	Categories		☑			□	
Description	Categories		☑			□	
ProductName	Products		☑			□	
UnitPrice	Products		☑			□	

Functionality

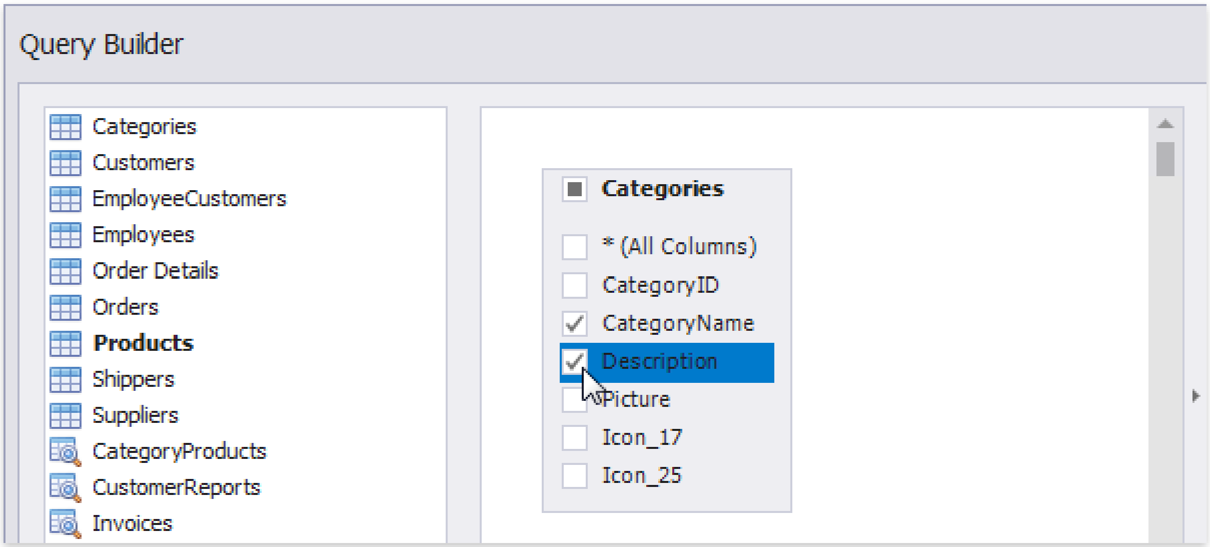
In the Query Builder dialog, you can add data tables and views to the data source, and select which columns to include. The Query Builder automatically joins the related tables, so all you need to do is drag and drop or double click on the table or view you wish to select.

Add Tables

To add tables/views to a data source, double-click the table (or view) or drag and drop it from the Tables pane onto the Diagram pane.



Then, select the required columns.

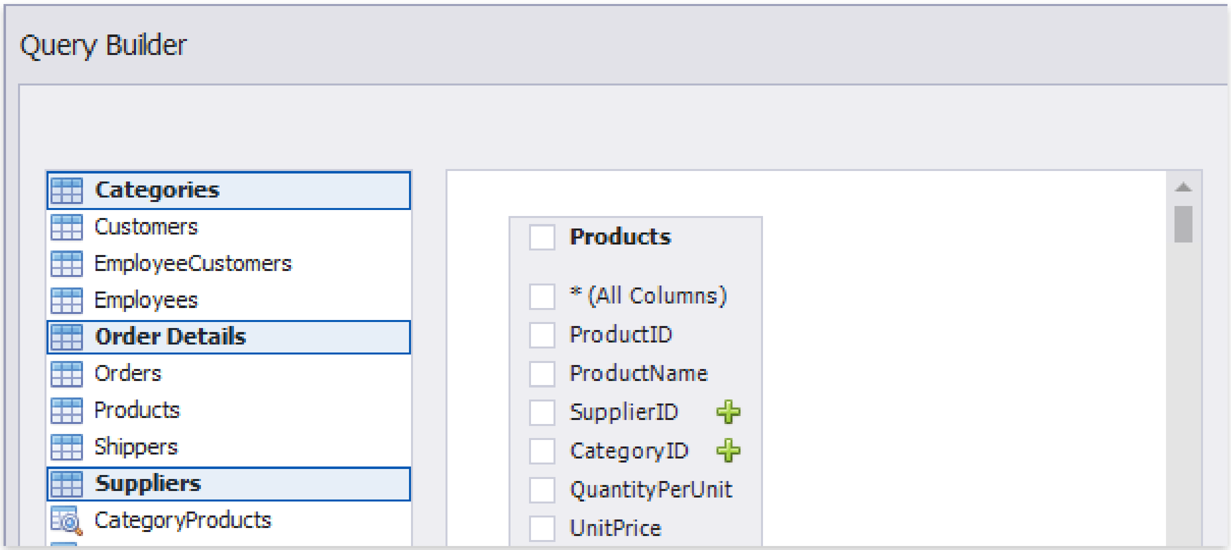


Categories

- □ * (All Columns)
- □ CategoryID
- ☑ CategoryName
- ☑ Description
- □ Picture
- □ Icon_17
- □ Icon_25

Join Tables

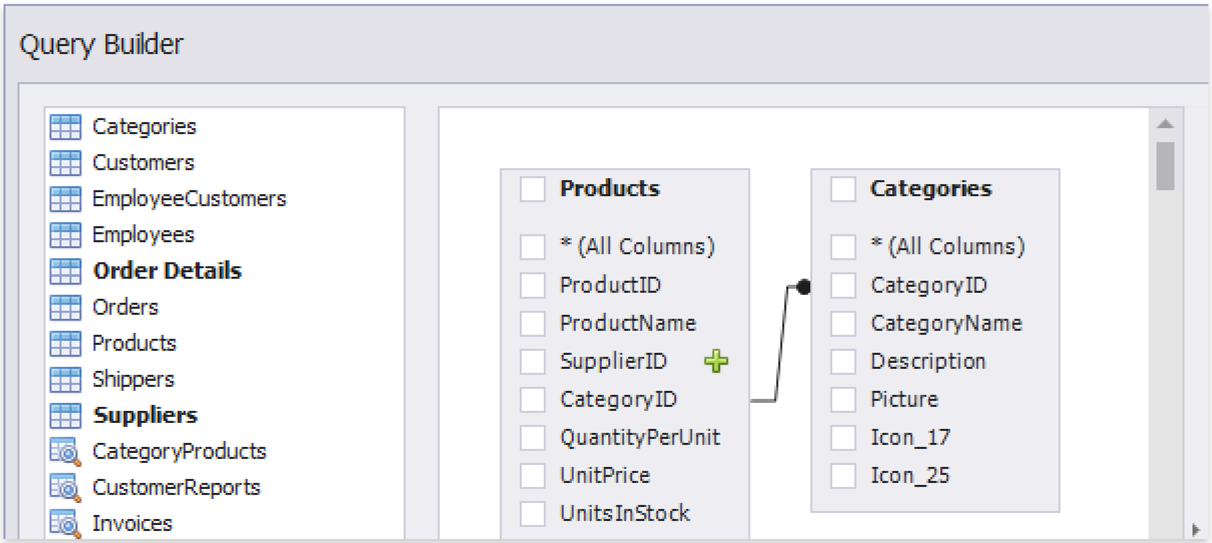
If at least one table has been added to the Diagram pane, the Tables pane marks in **Bold** lettering any tables that have a relationship with any of the recently added tables.

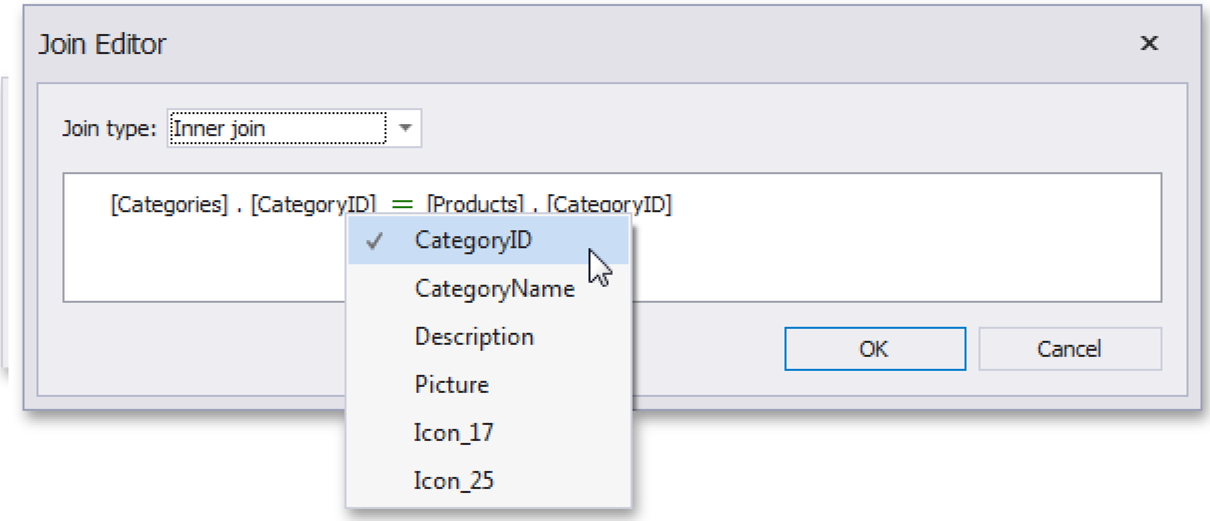


To join the already added table with another table, do one of the following.

- Click the green plus button next to the foreign key column (*SupplierID* and *CategoryID* in the image above).
- Drag and drop the highlighted table from the **Tables** pane to **Diagram** pane.

The Query Builder will display a relationship between tables.





Join Editor

Join type: Inner join

[Categories] . [CategoryID] = [Products] . [CategoryID]

To edit this relation, select it and use its context menu.

The following commands are available.

- **Edit Relation** - Allows you to edit the selected relation. Clicking this menu item opens the **Join Editor** dialog.

First, check the join type. You can specify it in the **Join type** combo box (*Inner join* or *Left outer join*). To edit column and table names in the existing condition, click the name you wish to replace and choose a different name from the popup menu.

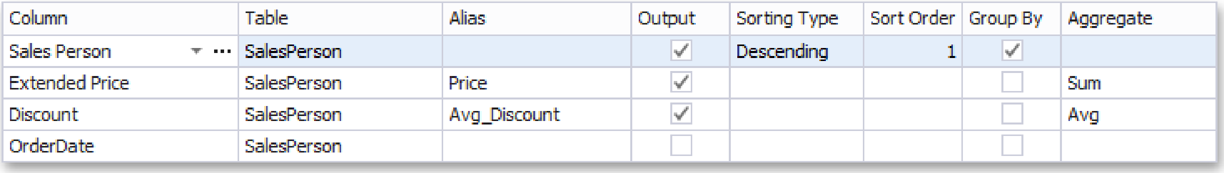
Note: The Join Editor dialog will be triggered automatically if you join tables that do not have a relationship at the database level.

Delete Relation - Removes the selected relation. Note that this action removes the joined table(s).

- Reverse Relation – Reverses the relationship between the tables.

Edit Column Settings

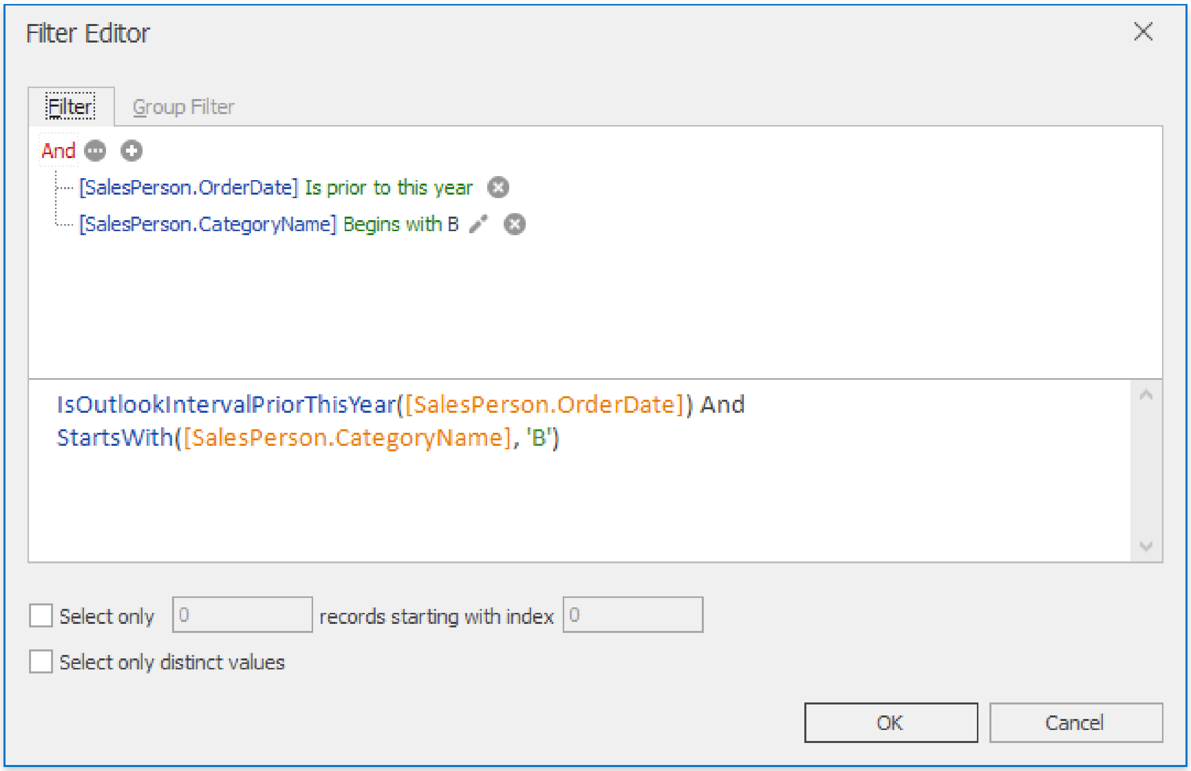
After you have added the tables and selected the required columns, you can change settings for each column in the **Grid** pane.



Column	Table	Alias	Output	Sorting Type	Sort Order	Group By	Aggregate
Sales Person	SalesPerson		☑	Descending	1	☑	
Extended Price	SalesPerson	Price	☑			☐	Sum
Discount	SalesPerson	Avg_Discount	☑			☐	Avg
OrderDate	SalesPerson		☐			☐	

The following settings are available for each column.

Use **Column** to select the required column from the combo box or add a new column. If



And
- [SalesPerson.OrderDate] Is prior to this year
- [SalesPerson.CategoryName] Begins with B

```
IsOutlookIntervalPriorThisYear([SalesPerson.OrderDate]) And
StartsWith([SalesPerson.CategoryName], 'B')
```

□ Select only 0 records starting with index 0

□ Select only distinct values

Use **Column** to select the required column from the combo box or add a new column. If necessary, you can customize a column expression. To do this, click the ellipsis button for the required column and specify the expression in the open dialog.

- The **Table** column displays corresponding table names.
- The **Alias** column allows you to specify the column alias, or display name.
  Note: Aggregated columns should always have an alias.
- The **Output** column allows you to choose whether to include specific columns to the query.
- Use the **Sorting type** combo box to specify the sort order of column values. The **Sort order** column allows you to specify the order in which several columns are sorted.
- The **Group By** statement is used in conjunction with the aggregate functions to group the result-set by one or more columns.
- The **Aggregate** option allows you to specify the aggregate function used to aggregate column values.
  Note: You should apply aggregation/grouping either to all columns or to none of them.

Filter Data

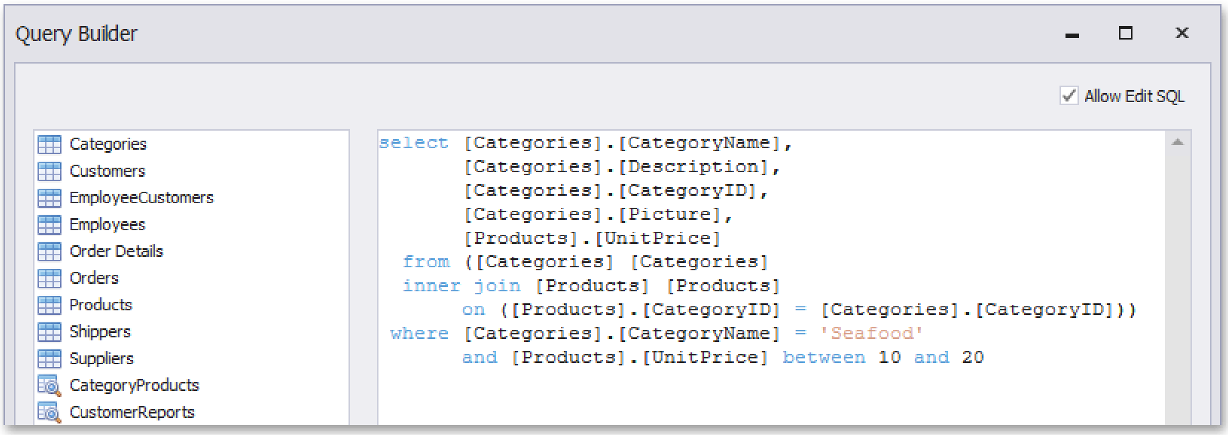
To filter data in the Query Builder, click the **Filter…** button. This opens the **Filter Editor** dialog that allows you to build filter criteria:

Edit Query Parameters

The Query Builder allows you to add query parameters when creating a filter criteria. To specify settings of added query parameters, click the **Edit Parameters…** button.

Customize SQL Query

After you add the tables, you can customize the automatically generated SQL query. To do this, enable the **Allow Edit SQL** checkbox and edit the SQL query displayed in the **SQL** pane.



☑ Allow Edit SQL

```
select [Categories].[CategoryName],
       [Categories].[Description],
       [Categories].[CategoryID],
       [Categories].[Picture],
       [Products].[UnitPrice]
  from ([Categories] [Categories]
  inner join [Products] [Products]
       on ([Products].[CategoryID] = [Categories].[CategoryID]))
 where [Categories].[CategoryName] = 'Seafood'
       and [Products].[UnitPrice] between 10 and 20
```

For instance, you can add a WHERE clause to the SQL expression.

Note: If you edit the automatically generated query and uncheck the **Allow Edit SQL** check box, your changes will be discarded, and the generated query will be restored.

Preview Data

The Query Builder allows you to preview data for the created SQL query. To do this, click the **Preview Results…** button.

This opens the Data Preview window containing data returned after executing the query.

Activity 3.1 – Build a Query

In this activity, you will build a query using the Analytic Dashboard Designer Query Builder.

Activity Steps

Part 1: Select tables

1. Select **Utilities > Analytic Dashboard Designer**.
2. Select the **New** button. An **Add Model?** message displays.
3. Select the **No** button.
4. Select the **Data Source** tab.
5. Select the **Add Query** button. The **Query Editor** opens.
6. Select the **Run Query Builder** button. The **Query Builder** opens.
7. Double-click the **SV_Projects** table in the upper left window.
8. Double-click the **SV_ProjectFacts** table in the upper left window. The **Join Editor** displays.

Part 2: Join tables

1. Verify that **Inner join** displays in the **Join Type** field.
2. Select the **<Select a column>** link. A list displays.
3. Select **FactID**.
4. Select the **<Select a table>** link. A list displays.
5. Select **SV_Projects**.
6. Select the **<Select a column>** link. A list displays.
7. Select **FactID**.
8. Select the **OK** button.

Part 3: Build the query

1. Select the following fields in the **SV_Projects** table.
   - ProjectName
   - Project Path
2. Select the following fields in the **SV_ProjectsFacts** table.
   - ChargeTypename
   - IsActive
   - PICEmpname
   - PMEmpname
3. Select the **Filter...** button. The **Filter Editor** opens.
4. Hover your cursor over the **And** button. Additional icons display.
5. Select the **Add** (plus) icon.
6. Select **SV_ProjectFacts.FactID**. A list displays.
7. Select **IsActive**. **Note:** You can filter the list by typing in the text field.

8. Select **Enter a value**.
9. Select the **check box**. This indicates the value is True.
10. Select the **OK** button.
11. Select the **Preview Results** button. The **Data Preview** window opens.
12. Select the **Close** button.
13. Select the **OK** button. The **Query Editor** opens.
14. Select the **Run Query Builder** button. The **Query Builder** opens.

Part 4: Continue building the query

1. Double-click the **EV_ProjectMetrics** table in the upper left window. The **Join Editor** displays.
2. Verify that **Inner join** displays in the **Join Type** field.
3. Select the **<Select a column>** link. A list displays.
4. Select **projectID**.
5. Select the **<Select a table>** link. A list displays.
6. Select **SV_Projects**.
7. Select the **<Select a column>** link. A list displays.
8. Select **projectID**.
9. Select the **OK** button.
10. Select the following fields in the **EV_ProjectMetrics** table.
    - fixedfeebilled
    - hourlybilled
    - laboreffort
    - laborcost
    - hourlycontract
    - fixedfeecontract
    - hoursworked
    - hoursbilled
    - laborbudget
    - hoursbudget
11. Select the **Preview Results** button. The **Data Preview** window opens.
12. Select the **Close** button.
13. Select the **OK** button. The **Query Editor** opens.
14. Select the **Finish** button.

Part 5: Validate the query

1. Select the **Home** tab.
2. Select the **Grid** button.
3. Expand the following tables in the **Data Source** section.
    - EV_ProjectMetrics
    - SV_ProjectFacts
    - SV_Projects

4. Click and drag **ChargeTypename** to the **New Column** field.
5. Click and drag the following fields to the **New Column** field. The data displays in the grid.
   - ChargeTypename
   - ProjectName
   - fixedfeebilled
   - hourlybilled
   - laboreffort
   - laborcost
   - hourlycontract
   - fixedfeecontract
   - hoursworked
   - hoursbilled
   - laborbudget
   - hoursbudget

CHECK YOUR UNDERSTANDING

Which one of the following options do you complete to add tables to a data source?

a) Select the New button.
b) Right-click the table.
c) Drag and drop to the table.

Refer to Appendix A for answers to the Check Your Understanding questions.

APPENDIX A: Check Your Understanding Answer Key

Lesson 1: Query Builder Overview

? What is the purpose of querying data with an analytic dashboard?

a) To write the programming language
b) To extract actional information from complex datasets
c) To structure visualizations

Lesson 2: Building Queries in the Query Builder Applet

How many datasets can each query reference in the Query Builder?

[✓ a) One]
b) Two
c) Multiple

Lesson 3: Building Queries in the Analytic Dashboard Designer

Which one of the following options do you complete to add tables to a data source?

a) Select the New button.
b) Right-click the table.
[✓ c) Drag and drop to the table.]

COMPLETION CERTIFICATE AND EVALUATION

When the course has finished, Unanet University staff will mark the attendance in the Learning Management System (LMS.) This marking is percentage based and may be prorated if a participant was absent for any period of time during the course.

The LMS will generate a completion certificate (including **CPE credits**, where applicable) which will be available to the participant under their Transcript section of the LMS.

We appreciate your feedback on our courses and encourage you to complete the course evaluation.