erwin Data Modeler

Feature Tour

Release 14.0

Legal Notices

Contact erwin

Understanding your Support

Review support maintenance programs and offerings.

Registering for Support

Access the erwin support site and register for product support.

Accessing Technical Support

For your convenience, erwin provides easy access to "One Stop" support for all editions of erwin Data Modeler, and includes the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- erwin Support policies and guidelines
- Other helpful resources appropriate for your product

For information about other erwin products, visit erwin by Quest Products page.

Provide Feedback

If you have comments or questions, or feedback about erwin product documentation, you can send a message to techpubs@erwin.com.

News and Events

Visit News and Events to get up-to-date news, announcements, and events. View video demos and read up on customer success stories and articles by industry experts.

Contents

Introduction

The Feature Tour guide walks Data Architects, Data Administrators, Application Administrators, Database Administrators, and Partners through the features introduced in erwin Data Modeler (erwin DM) 14.0 release.

The features and enhancements introduced in this release are:

- Data Vault 2.0 Modeling
- Models in JSON Format
- PostgreSQL 16.2
- Azure DevOps Source Control Support
- Reverse Engineer Configuration
- MongoDB Enhancements
- Normalization-Denormalization
- Productivity and UI Enhancements
- erwin Mart Portal Features and Enhancements
- erwin ER360 Features and Enhancements

For additional information about a feature, in erwin Data Modeler, click **Help** > **Help Topics** or press **F1**.

Data Vault 2.0 Enhancements

erwin Data Modeler (erwin DM) supports Data Vault 2.0 as a modeling technique across all target databases and brings new enhancements to the feature. The key principle of Data Vault Modeling is separating business keys, contexts, and relationships in distinct tables as hubs, satellites, and links.

This enhancement provides all Data Vault components by default for new and older models, and ability to clone components. Apart from this, erwin DM 14.0 also brings the following enhancements to Data Vault 2.0:

- Changing component theme
- Specifying primary key

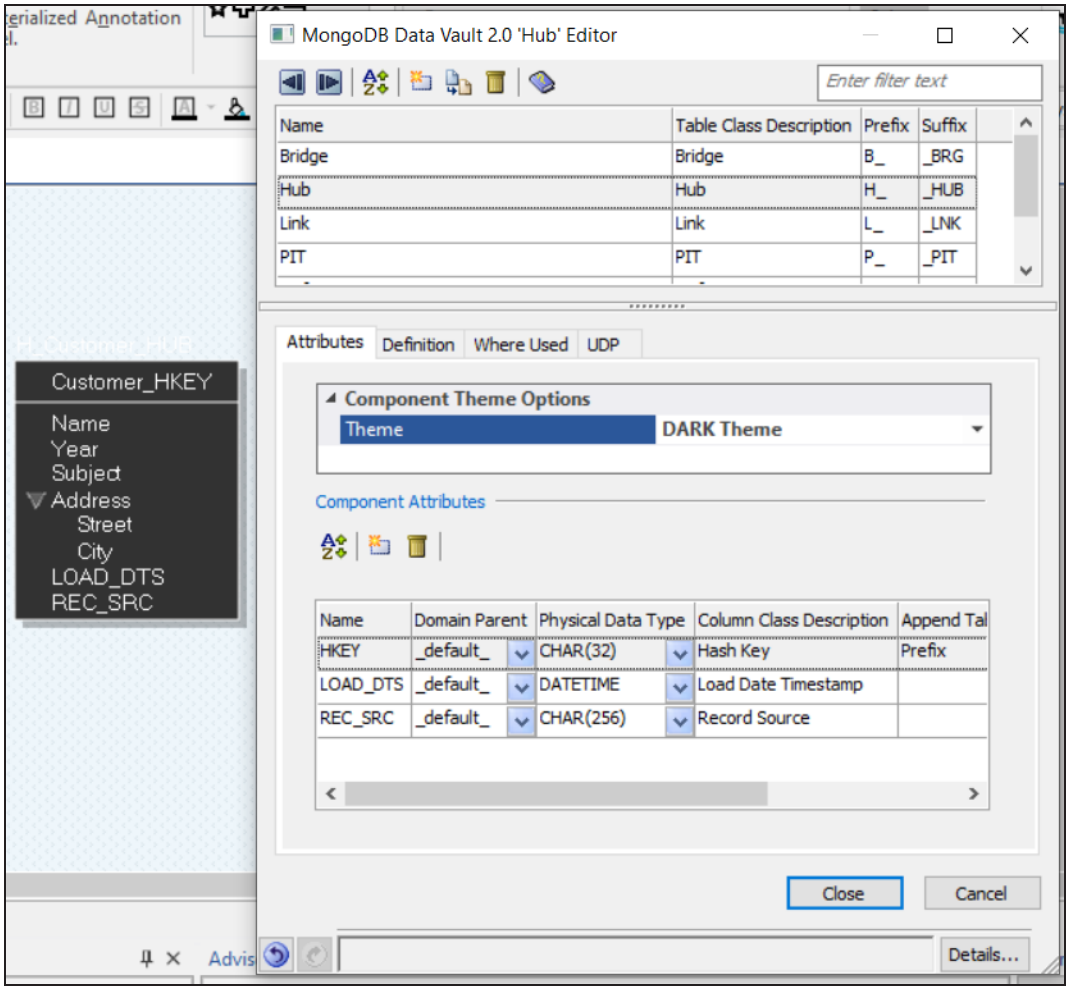
Changing Component Theme

You can either use an existing theme or create a new one and apply to the data vault components. To change the theme of the data vault components, follow these steps:

1. Open a model and select a table.
2. Click the appropriate Data Vault component on the ribbon.
   The selected table is converted to the selected Data Vault component type.
   For example, click **Hub** on the application ribbon after selecting a table. The selected table is converted to the Hub component.
3. On the Model Explorer, expand the Data Vault 2.0 node.
4. Right-click the Data Vault component, and then click **Properties**.
   The Data Vault 2.0 Hub Editor appears.
5. On the Attributes tab, you can change the theme of the selected table under **Component Theme Options**.
6. Under the Components Theme Options, you can choose classic, dark, default, or create custom themes for a Data Vault component.

   > You can also configure and save custom themes. For more information about configuring a custom themes, refer to The Theme Editor topic.

   The below screenshot displays the selected table in a dark theme. Similarly, you can switch between the themes mentioned above.

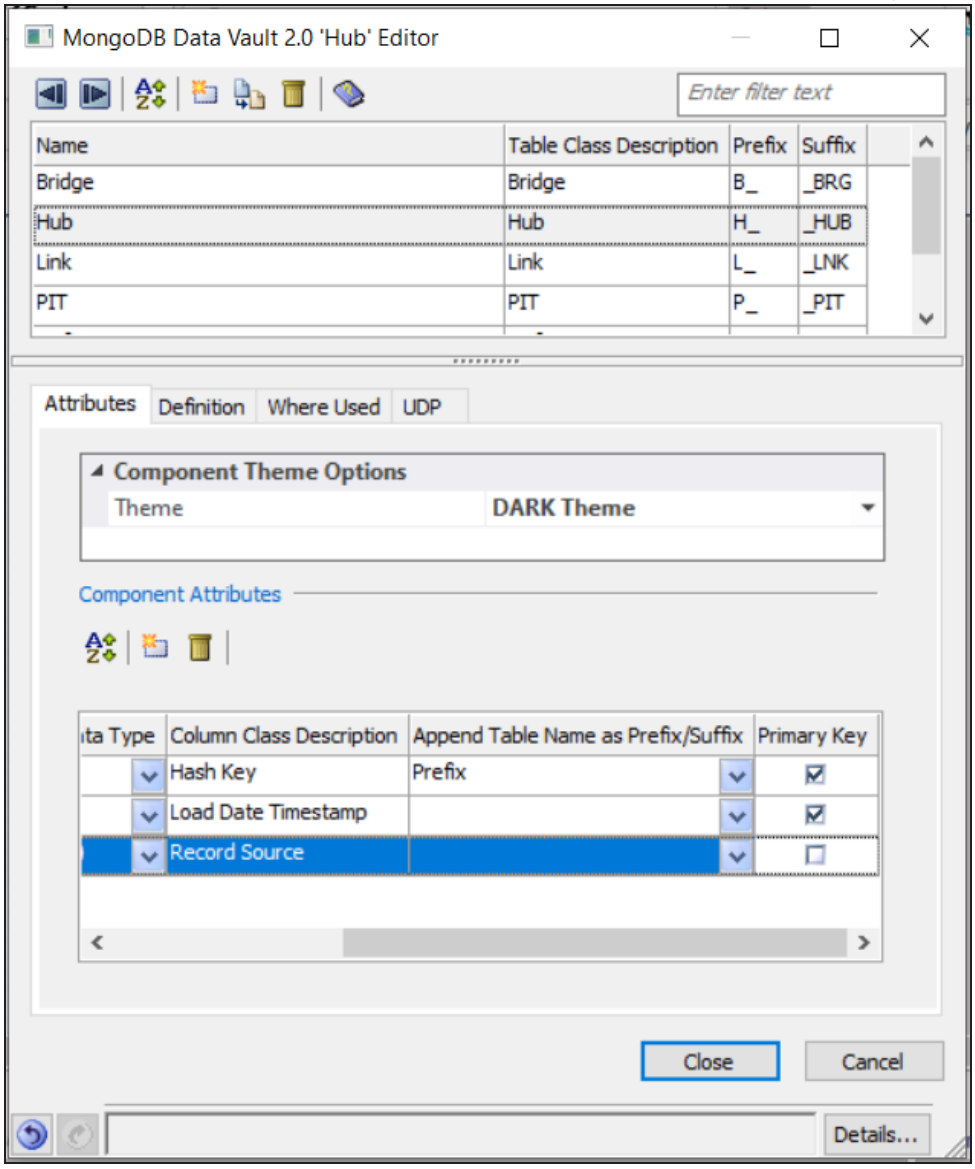


Specifying Primary Key

You can specifiy primary keys to the Data Vault component tables to uniquely identify the records. To specify primary keys for a table, follow the steps below:

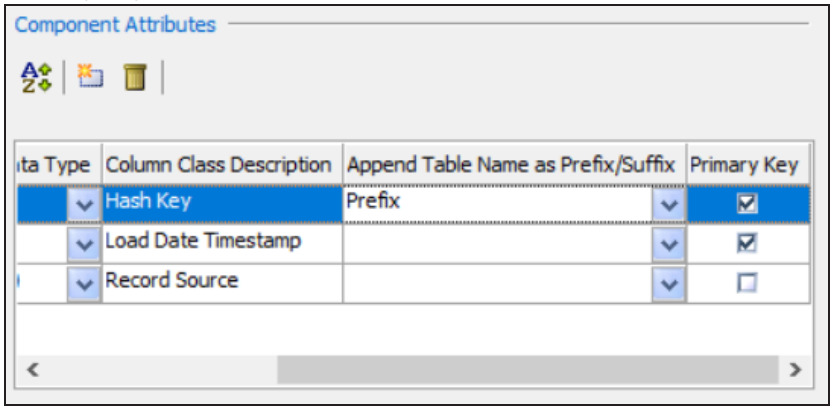
1. Open a model and select a table.
2. On the Model Explorer, expand the Data Vault 2.0 node.

3. Right-click the Data Vault component, and then click **Properties**.
   The Data Vault 2.0 Hub Editor appears, and the Attributes tab is shown by default.



4. Under the Components Attributes section, scroll to the right to the Primary Key column.

5. Select the relevant check box under the Primary Key column to specify an attribute as a primary key.



Save Models as JSON

erwin Data Modeler (erwin DM) allows you to save a model in a JSON format. This ability enables you to open a data model in cloud-based web applications, data localization, and more. To save a model as JSON, follow these steps:

1. Open a model and select **Save As** from the File menu.
   The Save As window opens.
2. Enter the name of the file in the File name field.
3. Click Save as type drop-down list, and select **JSON Files (*.json)**.
4. Click **Save**.
   The model is saved in JSON.

PostgreSQL Support Summary

erwin Data Modeler (DM) now supports PostgreSQL 16.2 as a target database. This implementation supports the following objects:

- Access Methods
- Casts
- Collations
- Roles
- Databases
- Domains
- Event Triggers
- Extensions
- Functions
- Languages
- Materialized Views
  - Column
  - Statistics
- Schemas
- Sequences
- Servers
- Permissions
- Procedure
- Tables
  - Column
  - Index
  - Statistics
  - Triggers
- Tablespaces

- Transforms
- User-Defined Types
- Views
  - Column

The following table lists the supported data types:

Category	Datatypes
Numeric Data Types	• smallint • integer • bigint • decimal • numeric • real • double precision • smallserial • serial • bigserial
Monetary Data Types	• money
Character Data Types	• char, character • character varying, varchar • text
Binary Data Types	• bytea
Boolean Data Types	• boolean
Date and Time Data Types	• date • timestamp [without time zone] • timestamp with time zone

PostgreSQL Support Summary

Category	Datatypes
	• time [without time zone] • time with time zone • interval
Geometric Data Types	• point • line • lseg • box • path • polygon • circle
Network Address Types	• inet • cidr • macaddr • macaddr8
Bit String Types	• bit • bit varying
Text Search Types	• tsvector • tsquery
UUID Type	• uuid
XML Type	• xml
JSON Type	• json • jsonb
pg_lsn Type	• pg_lsn
Other Datatypes	• pg_snapshot • txid_snapshot

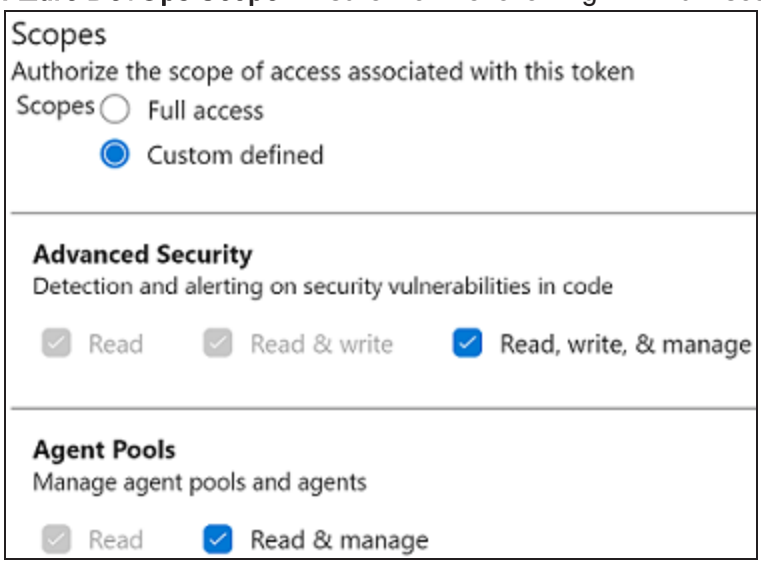
PostgreSQL Support Summary

Category	Datatypes
Composite Types	• User Defined Composite Types
Range Types	• User Defined Range Types

Azure DevOps Source Control Support

A new source control provider, Azure DevOps, has been added to erwin DM. This enables you to connect erwin DM to the Azure DevOps repositories and branches to save forward engineering scripts for a Mart model. For a successful connection to this repository, following are the pre-requisites:

- **Azure DevOps Scope**: Ensure that the following minimum scope is configured.



- **erwin Mart**: Ensure that,
  - erwin DM is connected to erwin Mart Portal. For more information, refer to the Connect to Mart topic.

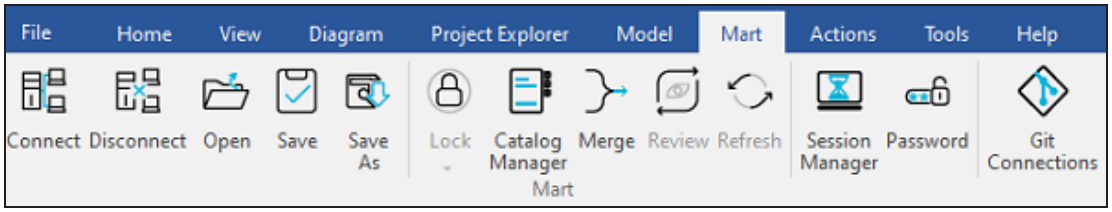
    Ensure that the following minimum permission is configured.

This permission is not available for Viewer profile.

- **Personal Access Token**: Ensure that you have created the required personal access token. To know how to create personal access tokens for Azure DevOps, refer to the Azure DevOps documentation.

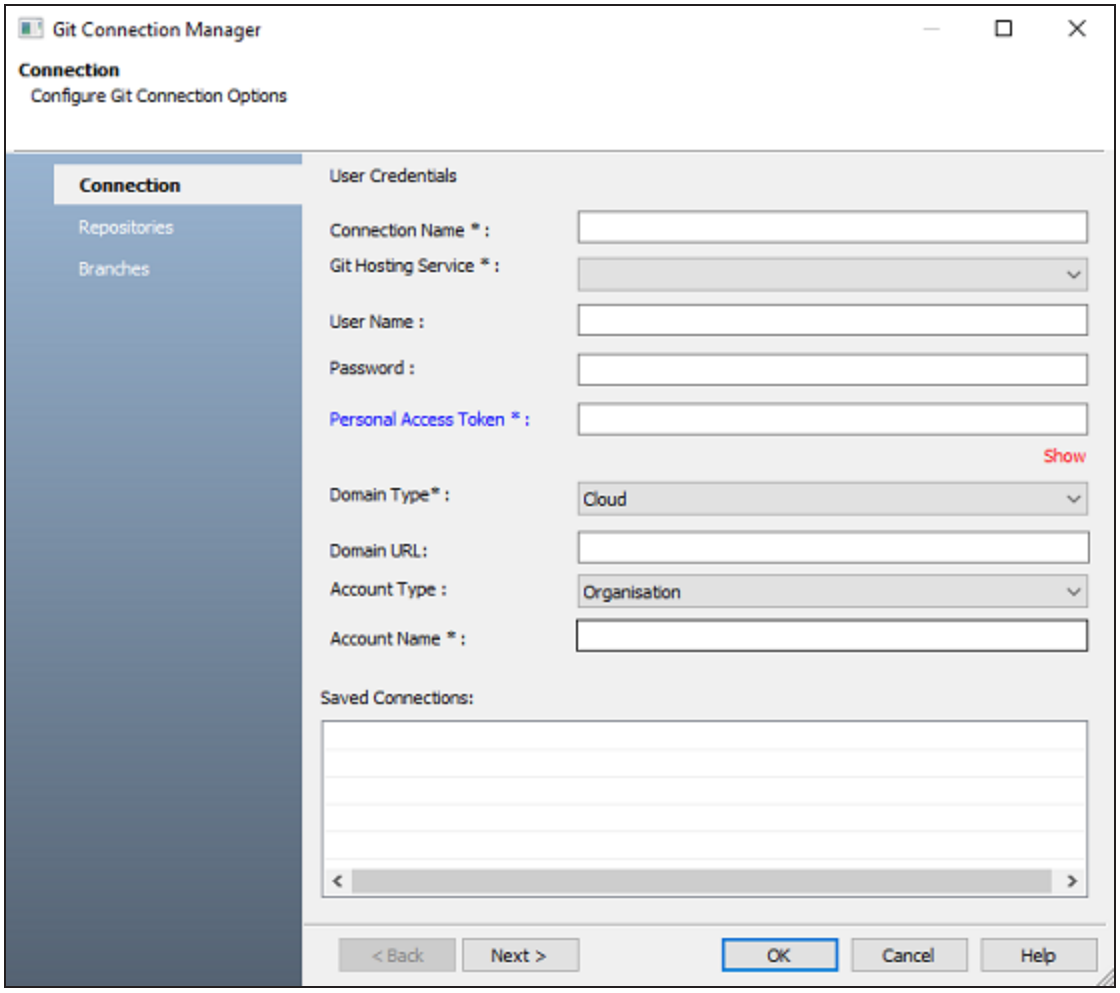
Once, these prerequisites are in place, to connect your repositories to erwin DM, follow these steps:

1. On the ribbon, click **Mart**.



2. Click **Git Connections**.

   The Git Connection Manager opens.



By default, the Connection tab opens.

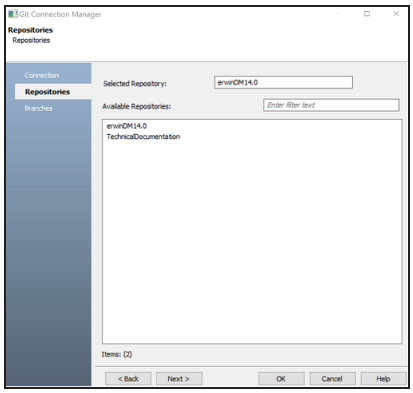
3. Enter appropriate values in the fields. Refer to the following table for field descriptions.

Field Name	Description	Additional Information
Connection Name	Specifies a user-defined connection name	For example, TechPub-sConnect. You can create multiple connections one for each repository.

Field Name	Description	Additional Information
Git Hosting Service	Specifies the source control hosting service to which erwin DM connects	Select a Azure DevOps from the drop-down.
User Name	Specifies the username to log on to the hosting service	This field is not mandatory.
Password	Specifies the password to log on to the hosting service	This field is not mandatory.
Personal Access Token	Specifies the personal access token to connect to the hosting service	
Org/User Option	Specifies whether the organization name or username should be use for your connection	
Org/User Name	Depending on your selection in Org/User Option field, enter organization name or username	

4. Click **Next**.

   The **Repositories** tab appears and displays the list of repositories available to your source control account.

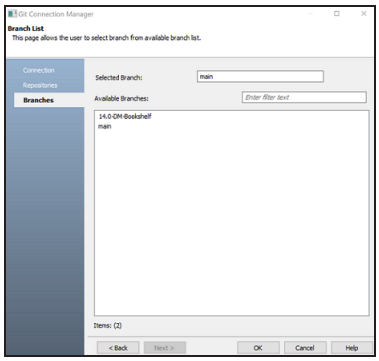


5. Select the repository where you want to push forward engineering scripts.

   You can also filter the list of repositories using the Available Repositories field.

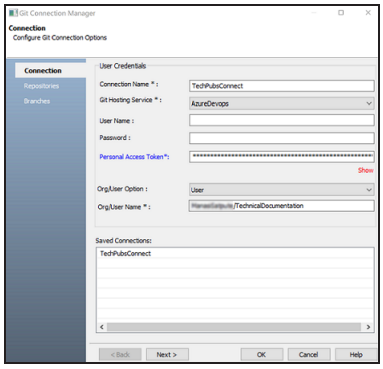
6. Click **Next**.

The **Branches** tab appears and displays the list branches available in the selected repository.



7. Select the branch where you want to push forward engineering scripts and then click **OK**.

   On successful connection, the connection name appears under Saved Connections.



Once you are connected to a repository, you can commit FE scripts.

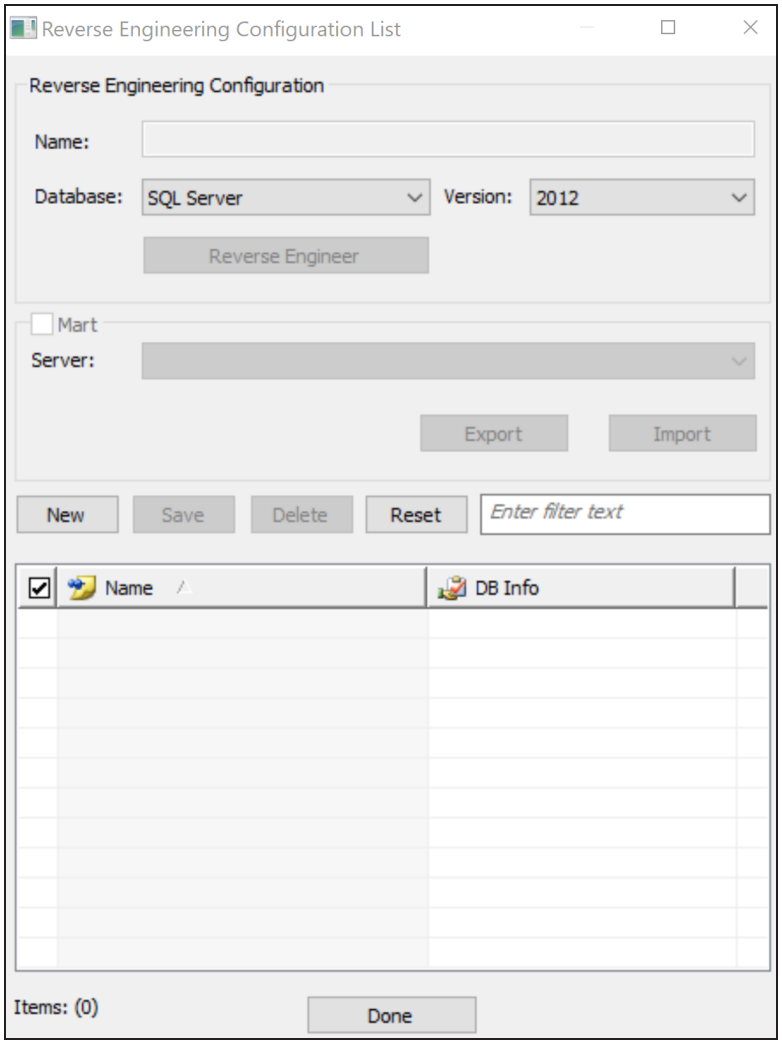
Setting Predefined Reverse Engineering Options

You can now configure and save predefined reverse engineering (RE) options, which enables you to eliminate manual option selections and reuse these configurations for future RE jobs. You can also save these configurations to your mart and use them as a base for new configurations. Additionally, you can synchronize RE configurations between erwin DM and erwin DM Scheduler using predefined lists.

To create predefined RE configurations, follow these steps:

1. In erwin DM, click **Actions** > **Reverse Engineer Configuration**.

   The Reverse Engineering Configuration List screen appears.



2. Click **New**.
3. Use the following options to configure RE options:

Option	Description
Name	Enter a name for the configuration.
Database	Select the database type.
Version	Select the relevant database version.

Option	Description
Reverse Engineer	Click Reverse Engineer to specify RE options. The Reverse Engineering Wizard appears. On the Reverse Engineering Wizard, do the following: 1. Configure RE options according to the selected database. For more information on RE options, refer to the corresponding database in the Database Support topic. 2. Click **Connections** to set up database connections. For more information on database specific connection parameters, refer to the Database Connection Parameters topic.
Mart	Select the Mart check box to import or export RE configuration in your mart.
Export (Optional)	Click Export to export the newly created configuration to erwin Mart Portal.
Import (Optional)	Alternatively, click Import to import an existing RE configuration saved in your mart. This option is available only when you are connected to erwin Mart Portal. To connect to a erwin Mart Portal, refer to the Connecting to Mart topic.

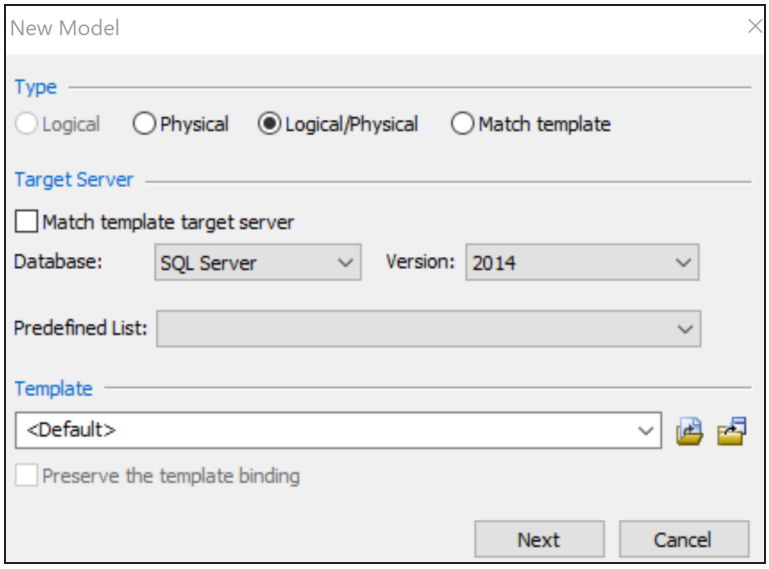
Once you have created a configuration, you can view it in the configuration list. In the Reverse Engineering Configuration List, use one of the following options:

- **Save**: Use this option to save the changes to a selected configuration.
- **Delete**: Use this option to delete the selected configuration.
- **Reset**: Use this option to reset the data in the Reverse Engineering Configuration section

4. Click **Done**.

   Your RE configuration is saved as predefined configuration.

   When you reverse engineer a model, you can select this configuration under Predefined List on the New Model screen.


New Model
Type
Logical
Physical
Logical/Physical
Match template
Target Server
Match template target server
Database:
SQL Server
Version:
2014
Predefined List:
Template
<Default>
Preserve the template binding
Next
Cancel

MongoDB Enhancements

Several enhancements have been implemented for MongoDB, which are supported for both REDB and RES. These enhancements are:

- Multiple Datatype Support and Union of Attributes in Arrays
- Enhanced Array Display
- Array Handling for Arrays with Heterogeneous Elements

Multiple Datatype Support and Union of Attributes in Arrays

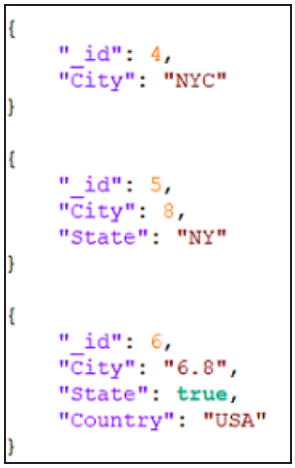
Multiple datatype support has been enhanced for the following scenarios:

- **Scenario1**: Objects within an array have an attribute with different datatypes

  For example, the following image shows an array of objects, address, that contains an attribute, city, with data that has string, integer, and double datatypes.

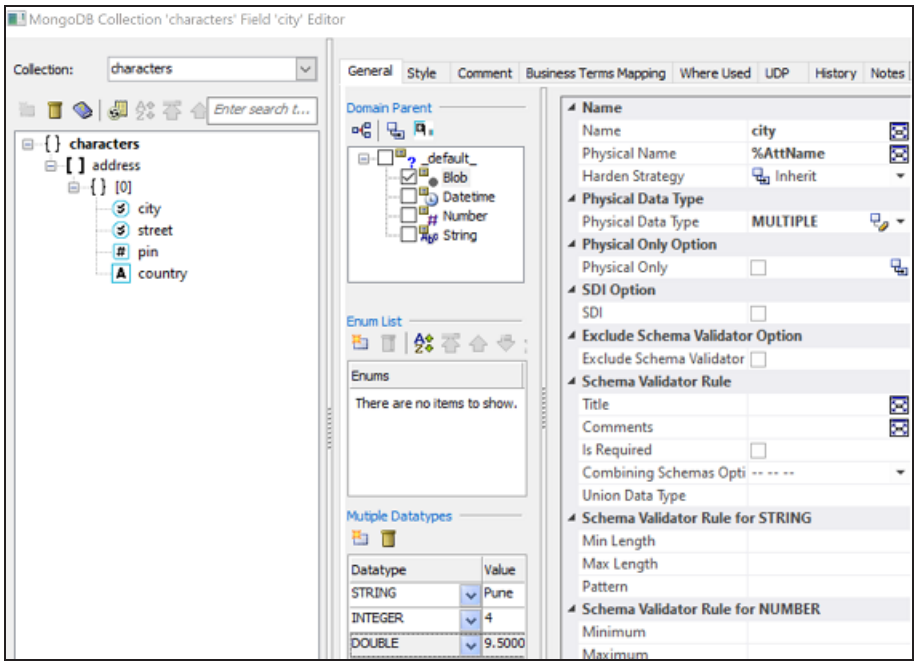
- **Scenario2**: Multiple objects (documents) in a collection have attributes with the same name but different datatypes

  For example, the following image shows a collection with multiple objects. Each object contains an attribute, city, with data that has string, integer, and double datatypes.



These scenarios are supported via the Multiple Datatypes section of the MongoDB Collection Field Editor. In the scenarios explained above, after REDB or RES, a union of attributes is done, and the

Physical Data Type of such attributes is set to Multiple. The actual datatypes are listed in the Multiple Datatypes section. For example, in the following image, the array, address, has an attribute, city with multiple datatypes. The Physical Data Type for city is set to Multiple and the Multiple Datatypes section lists that it has values with String, Integer, and Double datatypes.



Enhanced Array Display

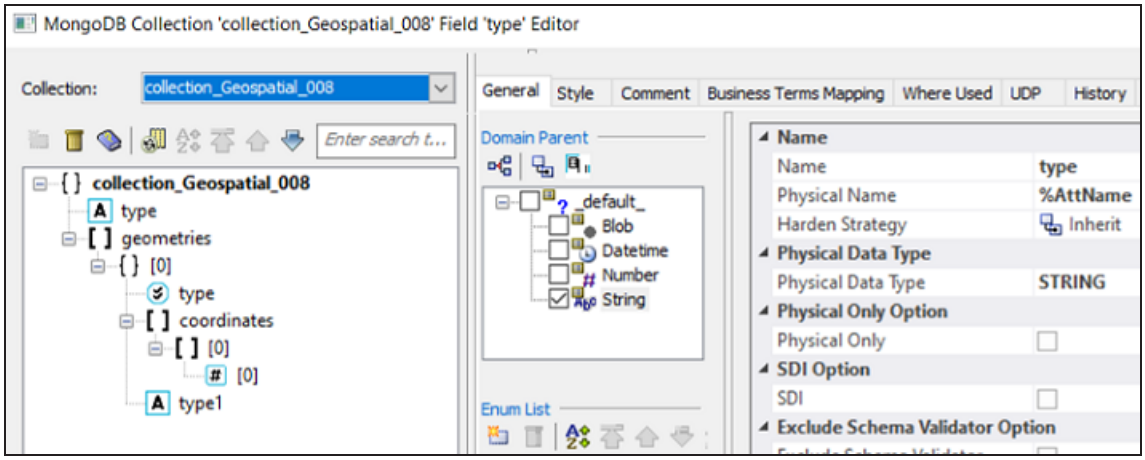
For collections with multiple arrays of geometric or geospatial like data, REDB or RES fetches the collection's schema instead of array elements and displays only the 0th element of such arrays in the ER diagram.

For example, consider a collection, GeometryCollection, which has multiple arrays.

```
type: "GeometryCollection",
geometries: [
  {
    type: "MultiPoint",
    coordinates: [
      [ -73.9580, 40.8003 ],
      [ -73.9498, 40.7968 ],
      [ -73.9737, 40.7648 ],
      [ -73.9814, 40.7681 ]
    ]
  },
  {
    type: 90,
    coordinates: [
      [ [ -73.96943, 40.78519 ], [ -73.96082, 40.78095 ] ],
      [ [ -73.96415, 40.79229 ], [ -73.95544, 40.78854 ] ],
      [ [ -73.97162, 40.78205 ], [ -73.96374, 40.77715 ] ],
      [ [ -73.97880, 40.77247 ], [ -73.97036, 40.76811 ] ]
    ],
```

Reverse engineering this collection results into the following ER diagram, where the schema of the collection is reverse engineered instead of array elements. Also, only the 0th element is displayed for each array.

The following image shows that Collection Field Editor for the reverse engineered collection.


MongoDB Collection 'collection_Geospatial_008' Field 'type' Editor
Collection:
collection_Geospatial_008
Enter search t...
collection_Geospatial_008
type
geometries
[0]
type
coordinates
[0]
[0]
type1
General
Style
Comment
Business Terms Mapping
Where Used
UDP
History
Domain Parent
default
Blob
Datetime
Number
String
Enum List
Name
Name
type
Physical Name
%AttName
Harden Strategy
Inherit
Physical Data Type
Physical Data Type
STRING
Physical Only Option
Physical Only
SDI Option
SDI
Exclude Schema Validator Option

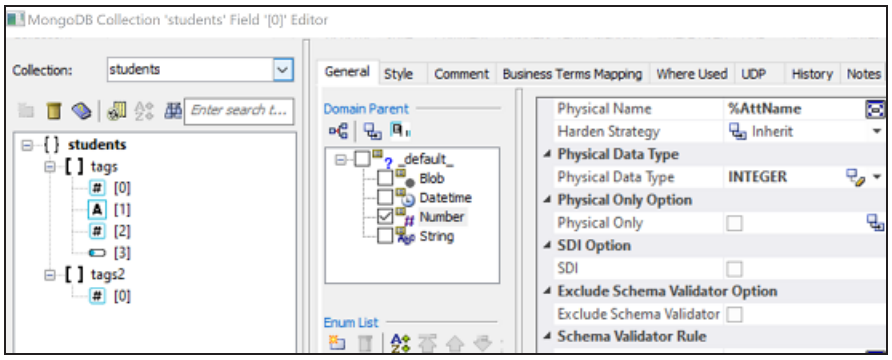
Handling Arrays with Heterogeneous Elements

For heterogeneous arrays, REDB or RES fetches only distinct datatypes and presents unique data-types in the model.

For example, the following image shows a heterogeneous array, tags, with one Boolean, one Double, two Integer, and two String elements.

```
{ "insertOne": { "document":
{ "_id": 3,
"name": "Alice1",
"year": "NumberInt(2013)",
"major": 4,
"gpa": 4.3,
"tags": [
  6.5,
  4,
  "str6",
  2,
  "str2",
  true
],
```

Reverse engineering this collection fetches distinct array element datatypes and is presented as shown in the following image. The array, tags, has single instance of elements with String, Integer, Double, and Boolean datatypes.



Normalizing and Denormalizing NoSQL Models

Denormalization is a database optimization technique that adds data redundancy to tables in data models to improve query performance. Using denormalization, you can also denormalize the structure of logical models such that you can build physical models that are designed effectively for target databases.

To summarize, denormalization is the process of converting normalized schema to non-normalized form for data optimization and to support time-sensitive database operations.

> Denormalization is not the reverse of normalization. It is an optimization technique that you can apply after normalizing a data model.

Normalization is a process that reorganizes data in a relational construct to minimize redundancy and non-relational constructs. Normalization enables you to control and eliminate data redundancy by removing model structures that provide multiple ways to know the same fact.

The normalization process follows a bottom-up approach, whereas the denormalization process follows a top-down approach. Based on the target database type and the process that you perform, following changes occur in your data models:

- **Denormalization**: Converting SQL models to NoSQL models
    - One-to-one relationships are converted to Object datatype.
    - One-to-many relationships are converted to an ArrayOfObject datatype.
- **Normalization**: Converting NoSQL models to SQL models
    - Objects datatype is converted to one-to-one relationships.
    - Array of Objects datatype is converted to one-to-many relationships.

Starting erwin Data Modeler (erwin DM) 14.0, the normalization-denormalization process offers complete logical and physical separation, where the logical side does not contain hierarchical data. This capability is available for MongoDB, DynamoDB, Couchbase, Google BigQuery, AVRO, JSON, and Parquet data models. For these models, you can view the logical side as a logical model and the physical side as a database model, and you can modify both the logical and the physical side of the model.

The following topics explain the normalization and denormalization processes in detail:

- Denormalizing NoSQL Models
- Normalizing NoSQL Models

Denormalizing Models

The denormalization process (SQL to NoSQL model conversion) embeds one or more objects into other objects based on the following logic:

- One-to-one relationships are converted to Object datatype.
- One-to-many relationships are converted to an ArrayOfObject datatype.

> You can also denormalize models when you derive a new model from an existing model.

Denormalizing SQL Models to NoSQL Models

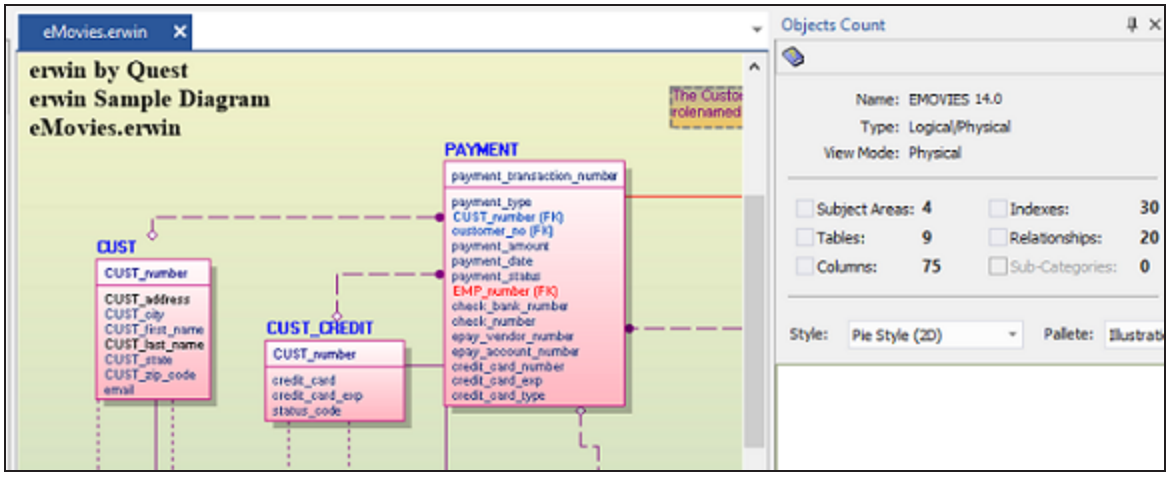
This section walks you through the denormalization process and its outcome for the conversion of SQL models to NoSQL models. For example, the following process demonstrates the conversion of the standard SQL model, eMovies to a MongoDB model.

To denormalize models, follow these steps:

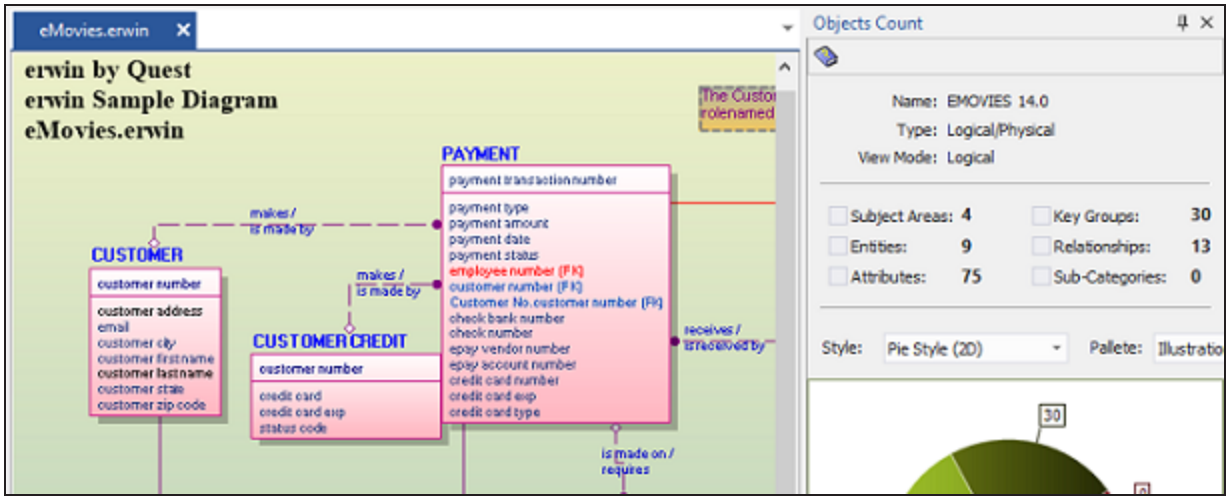
1. Open your SQL model.

   In this example, navigate to C:\Program Files\erwin\Data Modeler r10\BackupFiles\Samples\Standard and open the eMovies model.

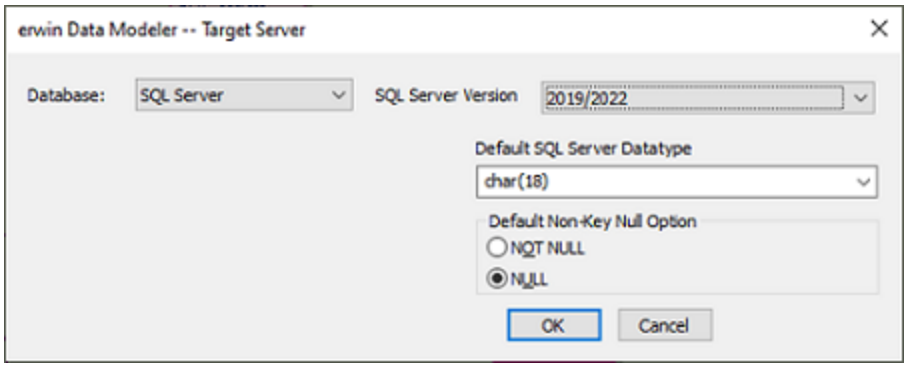
   Observe that the physical model contains 9 tables and 20 relationships.



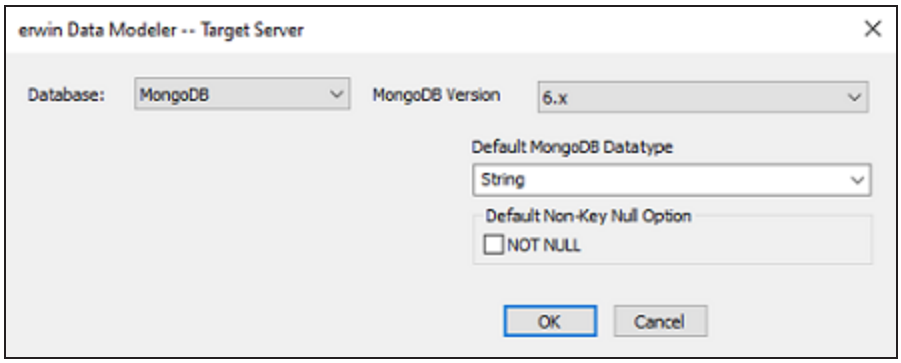
   The logical model has 9 entities and 13 relationships.



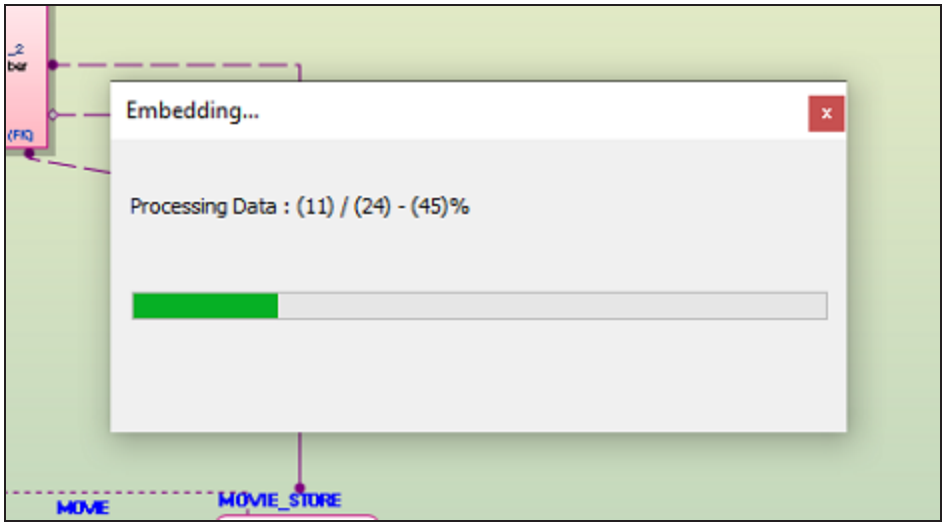
2. In the Physical view mode, on the ribbon, click **Actions** > **Target Database**.



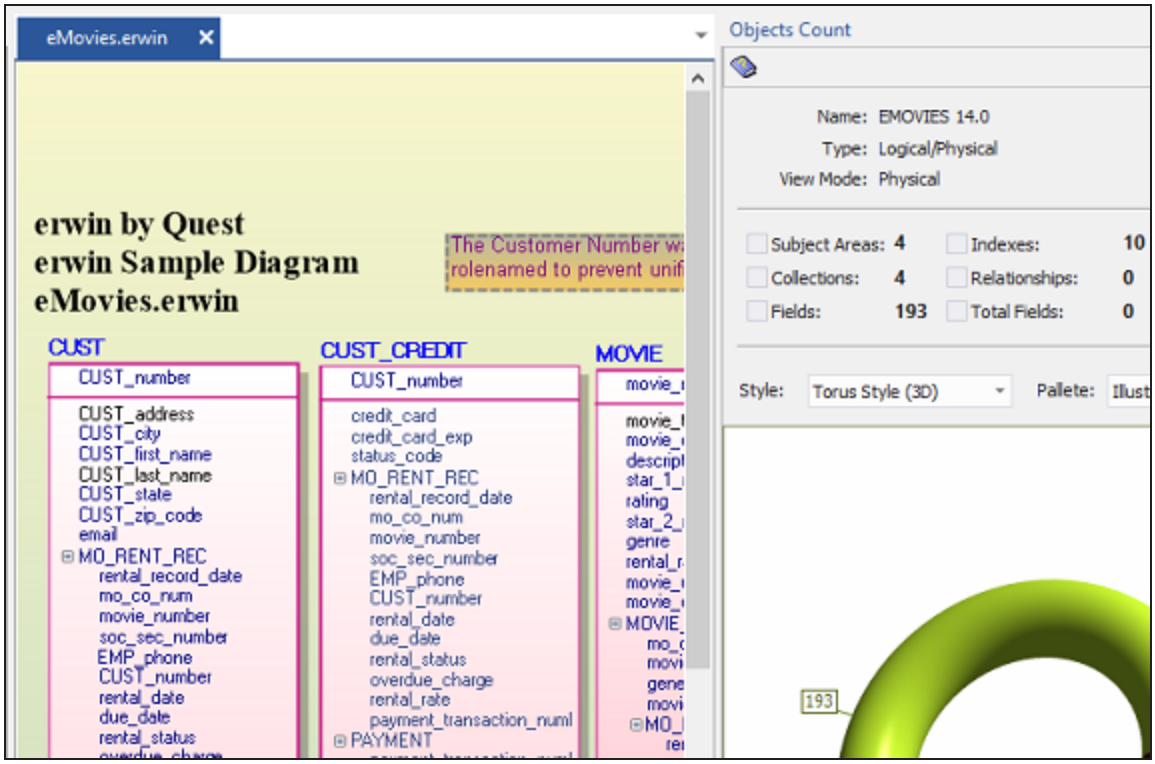
3. Change the Database from SQL Server to MongoDB and click **OK**.



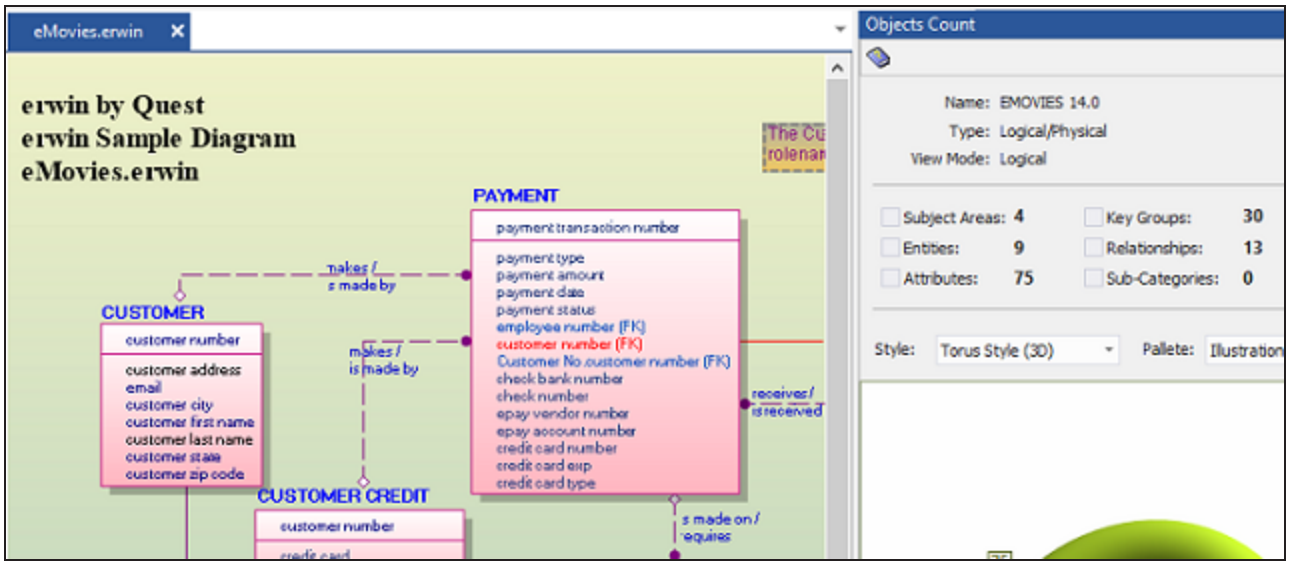
The embedding process starts.



After the embedding process completes, the eMovies model is converted to a MongoDB model. Based on the relationships in the SQL model, the parent-child embedding process occurs. Based on this process, the MongoDB physical model has 4 collections and 0 relationships.



The logical model retains relationships and does not contain any hierarchical datatypes. Thus, keeping the logical and physical models completely separate.

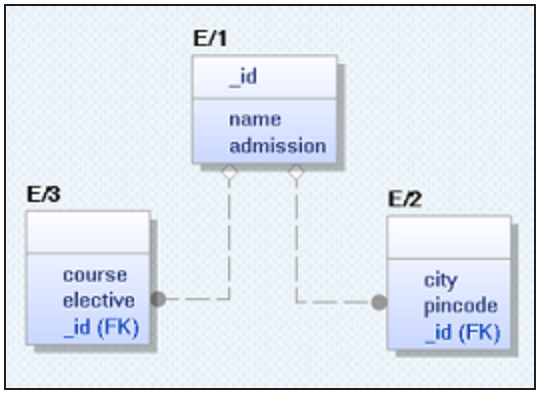


To summarize, post denormalization process, the logical model remains as a normalized model, whereas the physical model is converted to it denormalized form.

Logical to Physical Model Denormalization

Due to the complete logical-physical model separation, the logical mode always maintains the normalized form and the physical side always maintains the denormalized form. Whenever you switch from the logical model to the physical model, the model is auto-denormalized, and based on parent-child relationships in the logical model, the embedding process runs.

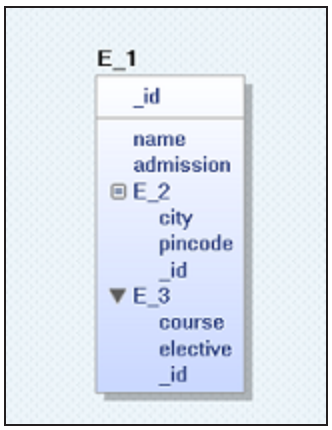
For example, consider the following logical MongoDB model.



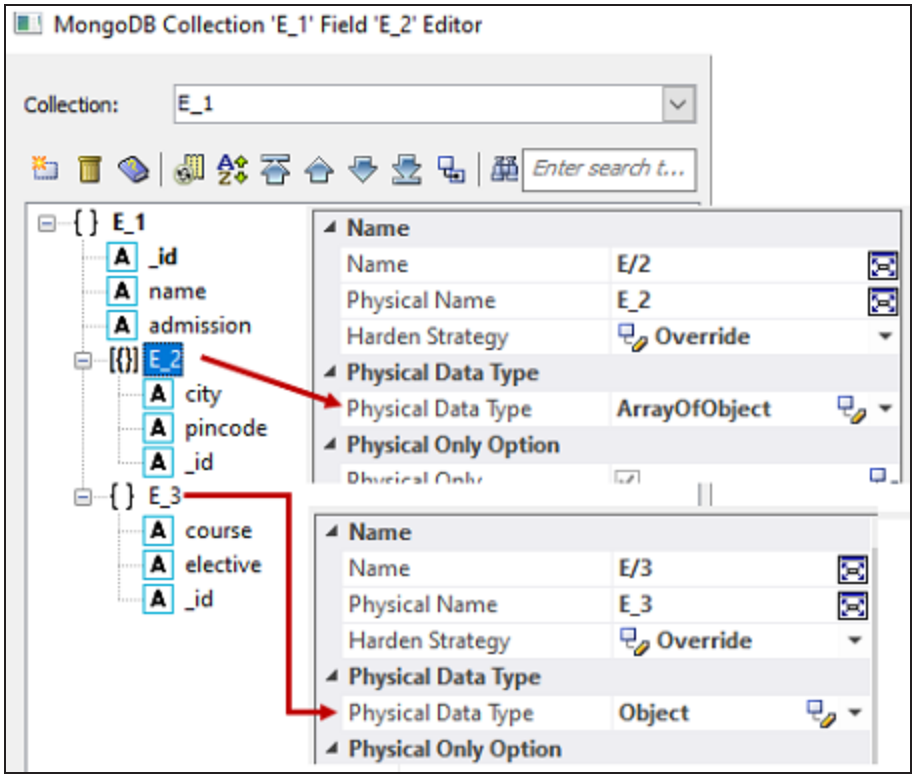
In this model:

- E/1 and E/2 have a non-identifying parent-child relationship with its cardinality set to Zero, One or More.
- E/1 and E/3 have a non-identifying parent-child relationship with its cardinality set to Zero or One (Z).

Now, when you switch to the physical model, auto-denormalization occurs. Based on the parent-child relationships in the logical model, E/2 and E/3 are embedded into E/1.



One-to-one relationship is converted to Object datatype and one-to-many relationship is converted to an ArrayOfObject datatype.



Also, any changes you make to the logical model are maintained in the physical model and vice-a-versa.

DynamoDB

The conversion logic applied to DynamoDB models is:

- One-to-one relationships are converted to Map datatype.
- One-to-many relationships are converted to List datatype.

Parquet

The conversion logic applied to Parquet models is:

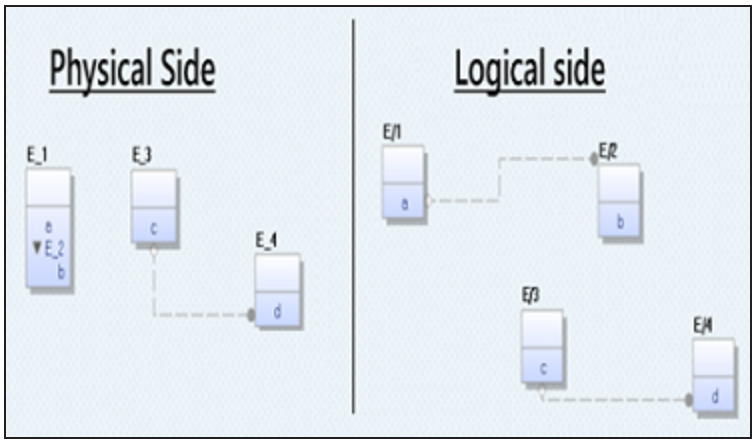
- One-to-one relationships are converted to Record/Struct datatype.
- One-to-many relationships are converted to Record/Struct datatype with the Mode set to Repeated/Array

Exceptions in Google BigQuery

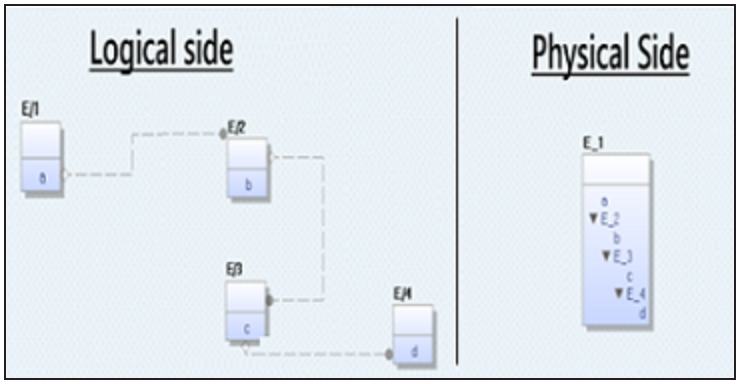
The conversion logic applied to Google BigQuery models is:

- One-to-one relationships are converted to Record/Struct datatype.
- One-to-many relationships are converted to Record/Struct datatype with the Mode set to Repeated/Array.

Also, as Google BigQuery supports relationships when you switch from the logical model to physical model, auto-denormalization is not performed. However, this scenario has an exception. For example, consider a physical model as shown in the following image.



Switch to the logical model and add a relationship between E/2 and E/3. In this scenario, when you switch to physical model, auto-denormalization will be performed as E_2 does not exist in the physical model as a separate table.



Denormalizing Cassandra and Amazon Keyspaces

Similar to denormalizing SQL models to NoSQL models by changing the target database, you can denormalize Amazon Keyspaces and Cassandra models using the Denormalization feature.

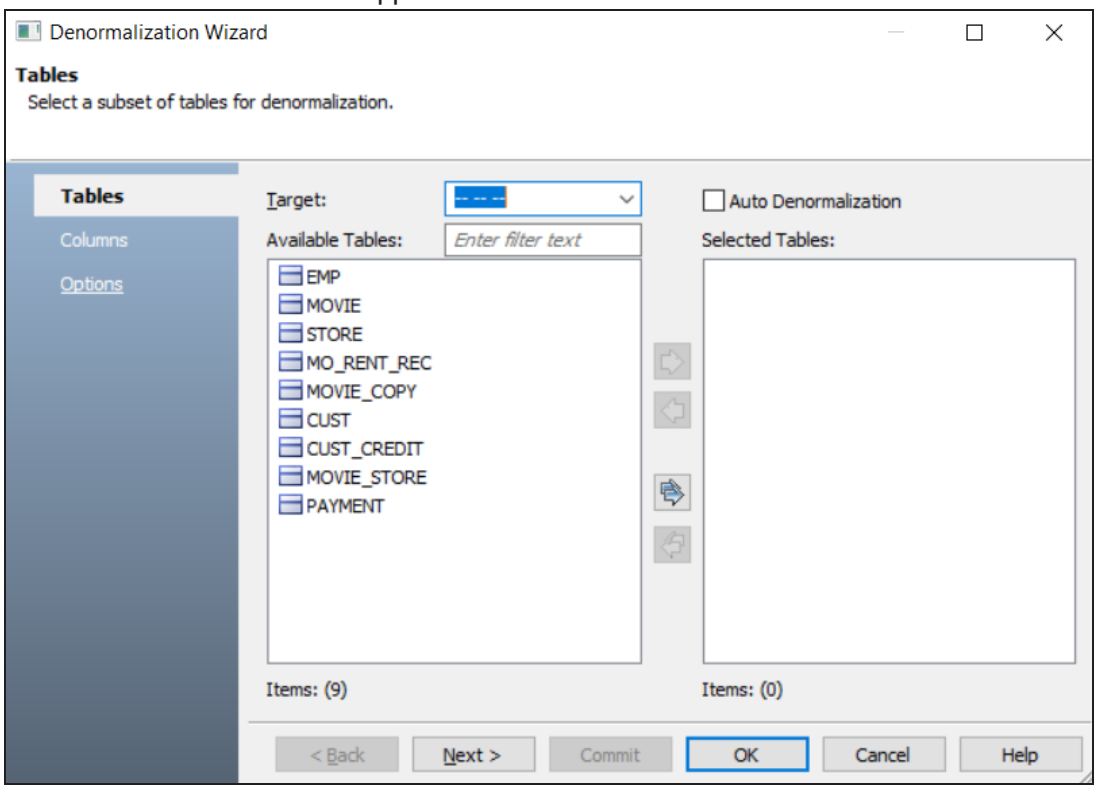
Denormalizing NoSQL Database Models

This section walks you through the denormalization options for a Cassandra model. To denormalize models, follow these steps:

1. Open your model.
2. On the ribbon, click **Action** > **Denormalization**.

   > This option is supported for Amazon Keyspaces and Cassandra.

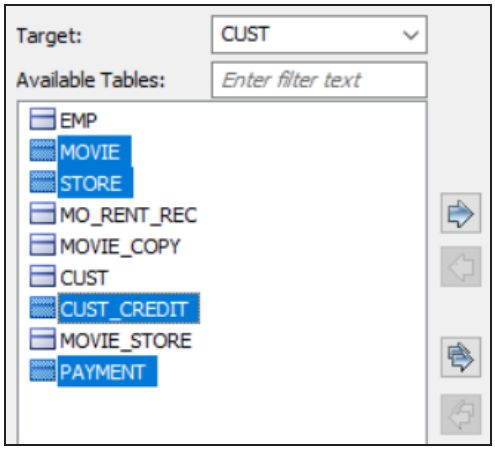
   The Denormalization Wizard appears.



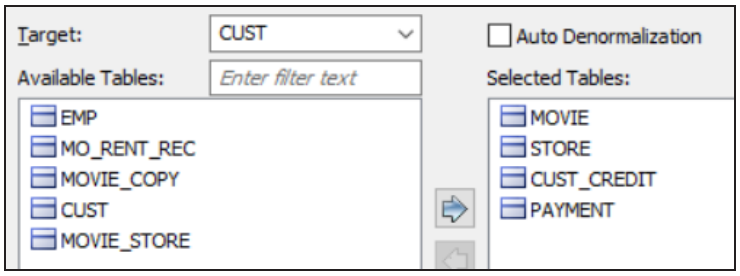
3. In the Tables section, click the **Target** drop-down to select a target table. All the tables will be merged into the selected target table.

   > Select the **Auto Denormalization** option to merge tables with the target automatically. This embeds the tables in the model with one-to-one relationships as User Defined Type and one-to-many relationships as normal columns. If you use this option, do not configure any options on the Columns and Options tab. Click **OK**.

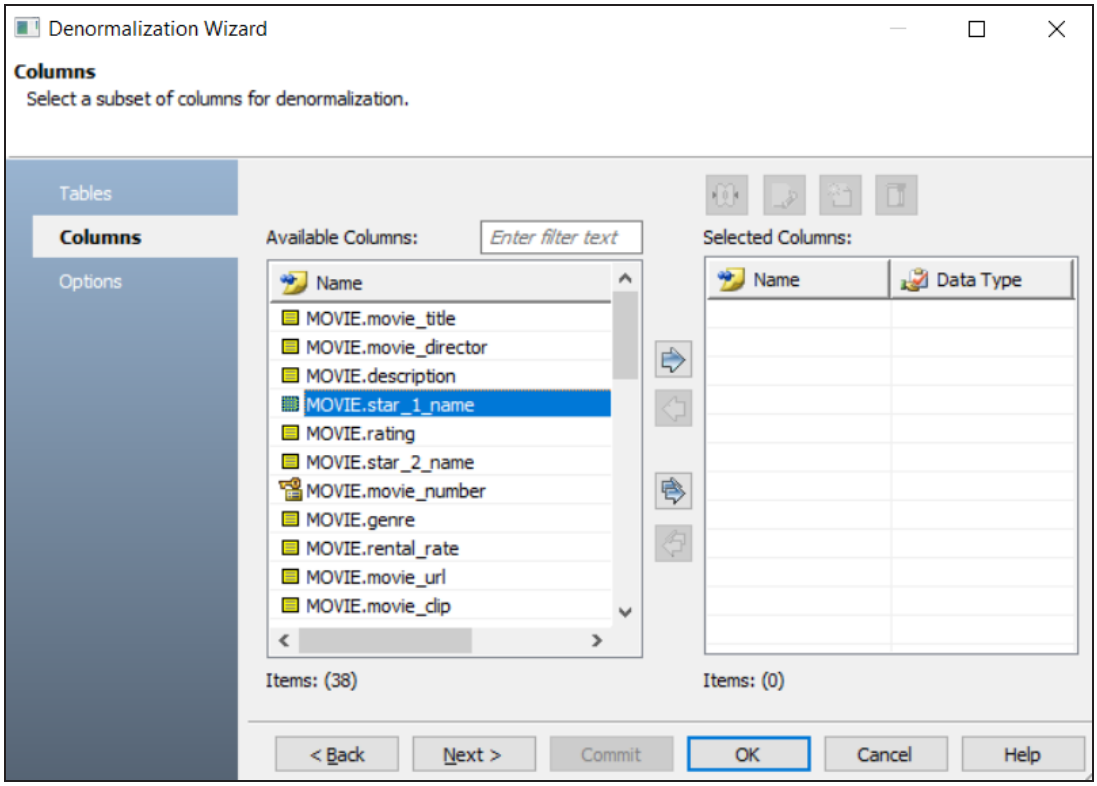
4. Under **Available Tables**, select one or more tables to merge. Then, click  .



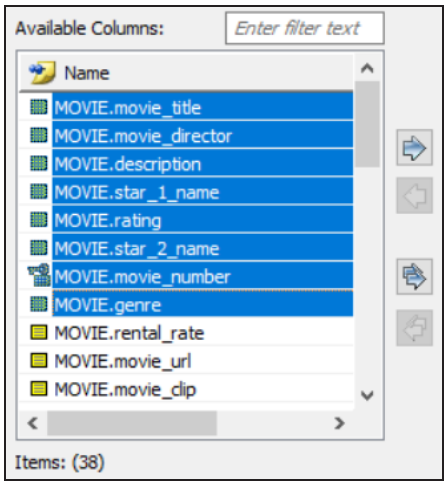
This moves the selected tables under Selected Tables.



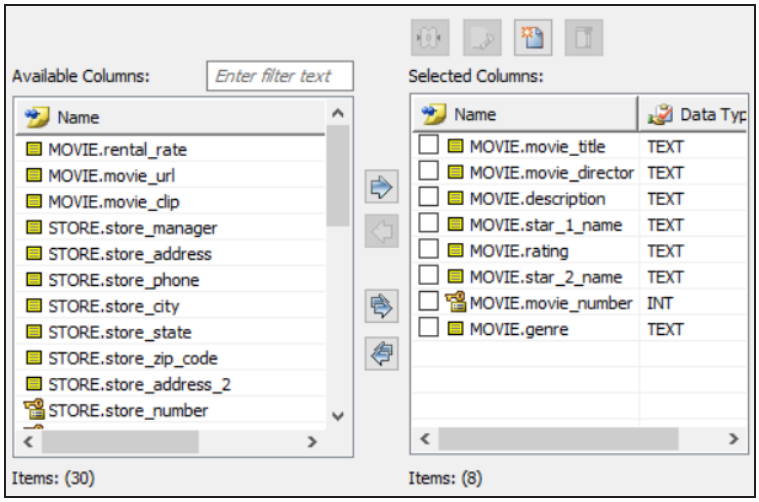
5. Click **Next**.
   The Columns section appears. It displays a list of available columns.



6. Under **Available Columns**, select the columns that you want to merge. Then, click  .



This moves the selected columns under Selected Columns section.



Once you have added the selected columns, you can use any of the following options:

Merge ( )

Use this option to merge the selected columns and create a new column under Selected Columns.

Update ( )

Use this option to edit column details such as column name, domain parent, and data type for a selected column.

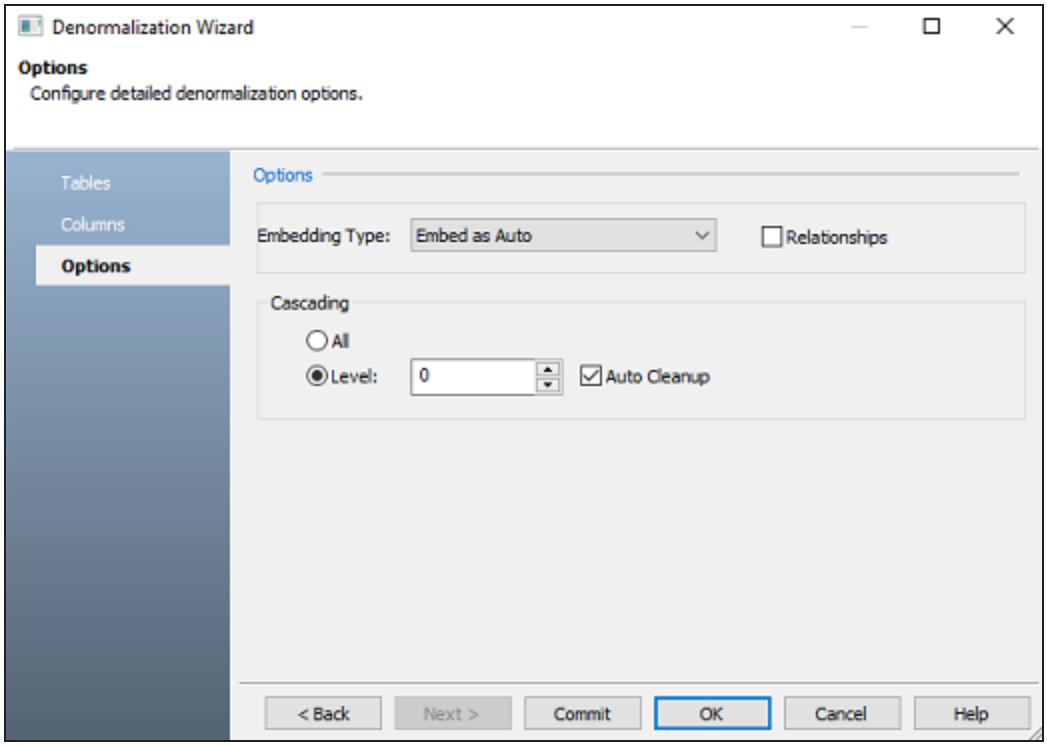
New ( )

Use this option to add a new column under Selected Columns.

Delete ( )

Use this option to delete the selected columns.

7. Click **Next**.
   The Options section appears.



8. Select an **Embedding Type**.

   You can select the following embedding options:

   - **Embed as Auto**: Use this option to embed tables through an auto-mechanism based on one-to-many and one-to-one relationships
     - One-to-many relationships are converted to normal columns.
     - One-to-one relationships are converted to User Defined Type (UDT) columns.
   - **Embed as Normal**: Use this option to embed collections using normal column styles.
   - **Embed as UDT**: Use this option to embed collections using UDT styles.

9. Select **Relationships** option to include table relationships to the model.

   > Relationships and Auto Cleanup options are mutually exclusive. As a best practice, always select only one of the options.

10. Select **Cascading** options to determine how multiple collections are merged into a single collection.
    Use the following cascading options:

- **All**: Use this option to denormalize all relationship levels in a collection into a single collection.
- **Levels**: Use this option to specify the number of levels up to which collections are denormalized into one collection. For example, if you set Level to 1, all the collections up to level 1 in the relationship hierarchy will be denormalized into a single collection.
- **Auto Cleanup**: This option removes child collections after denormalization.

> Relationships and Auto Cleanup options are mutually exclusive. As a best practice, always select only one of the options.

11. Click **OK**.

    The denormalization process starts and displays objects based on selected options. Alternatively, you can click **Commit** to apply changes to the model without exiting the Denormalization Wizard.

Normalizing Models

The normalization process creates one or more objects and relationships based on the following logic:

- Object datatype is converted to one-to-one relationships.
- ArrayOfObject datatype is converted to one-to-many relationships.

> Normalization is not the reverse of denormalization.

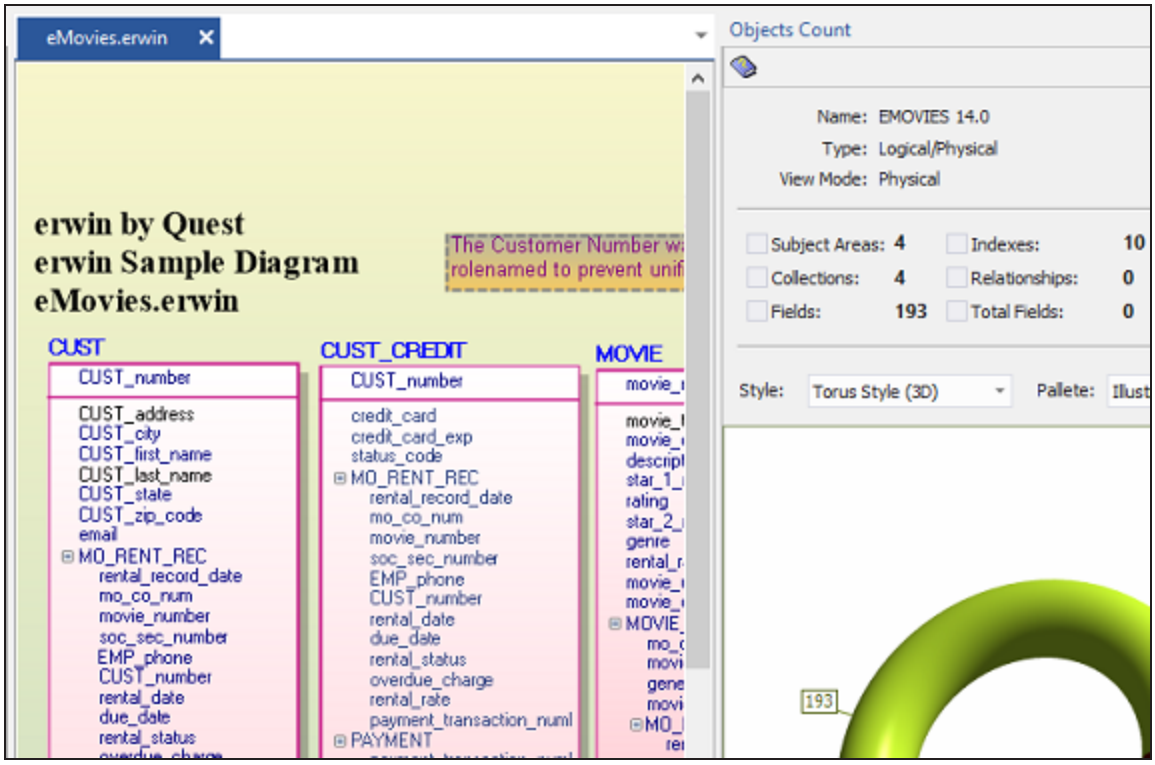
Normalizing NoSQL Models to SQL Models

This section walks you through the normalization process and its outcome for the conversion of NoSQL models to SQL models. For example, the following process demonstrates the conversion of a MongoDB model to SQL model.

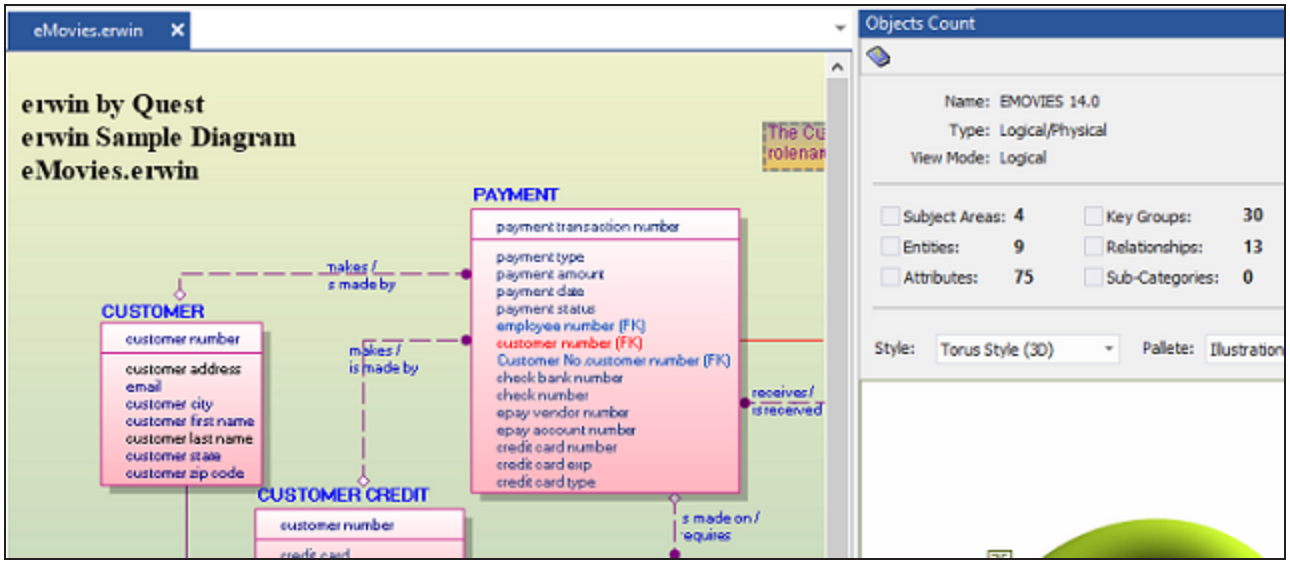
To normalize models, follow these steps:

1. Open your NoSQL model.

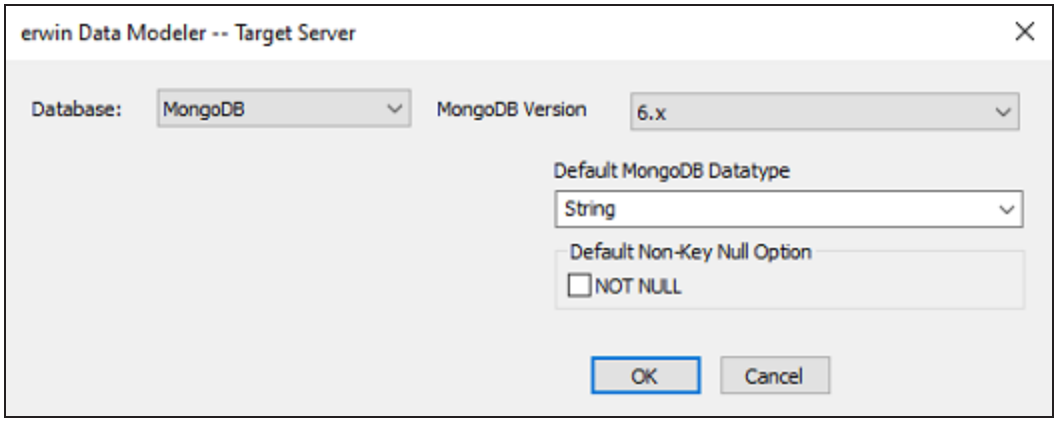
   Observe that the MongoDB physical model has 4 collections and 0 relationships. Each of the four collections have objects or array of objects embedded within them.



The logical model contains relationships and does not contain any hierarchical datatypes.

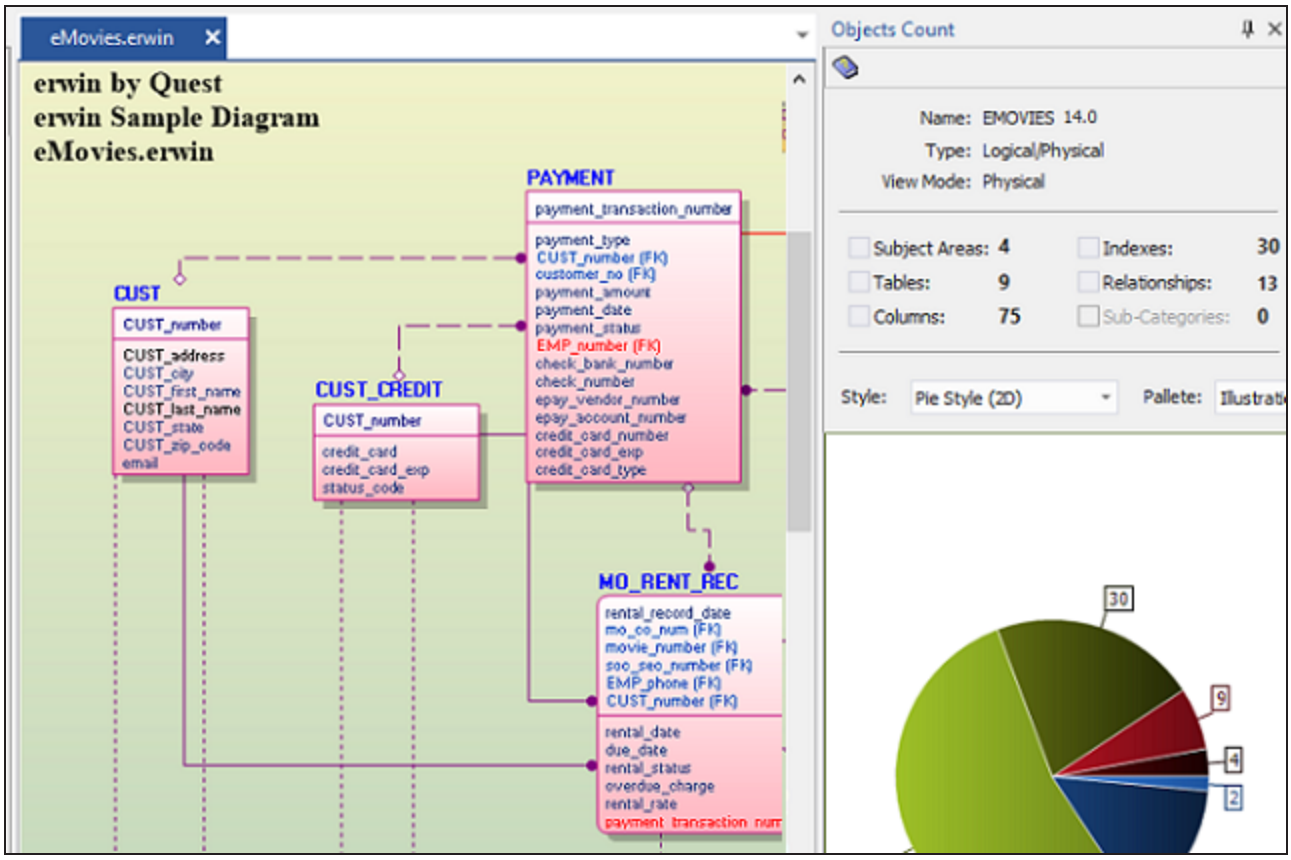


2. In the Physical view mode, on the ribbon, click **Actions** > **Target Database**.

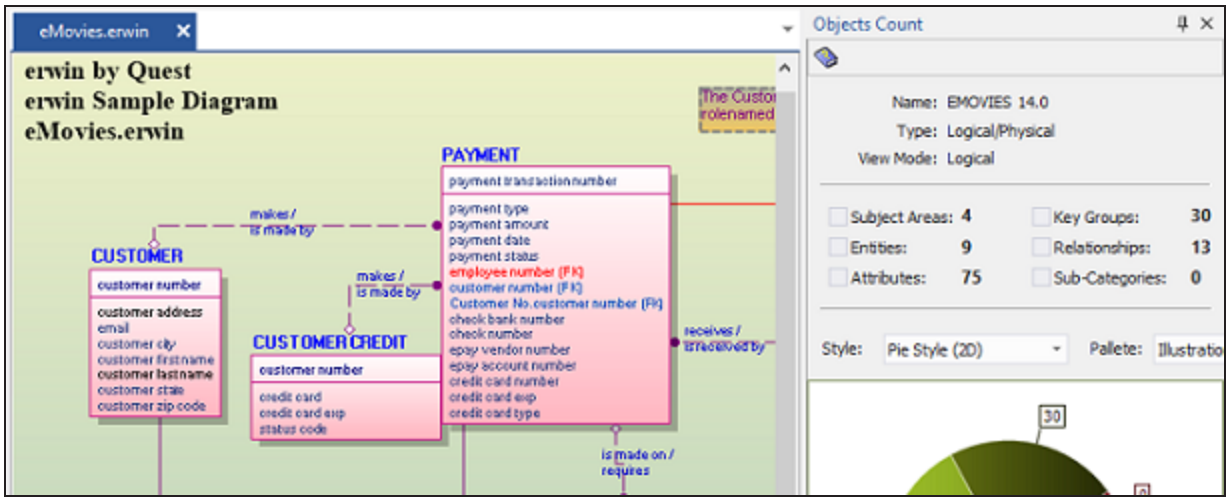


3. Change the Database from MongoDB to SQL Server and click **OK**.

   The normalization process is performed and tables and relationships are created based on the embedded objects. Observe that the physical model contains 9 tables and 13 relationships.



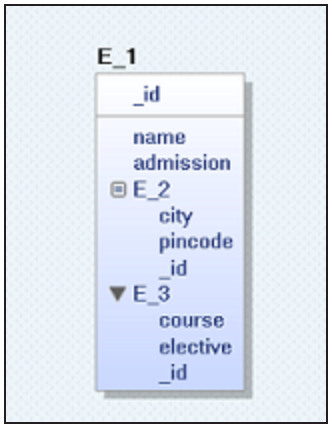
The logical model has 9 entities and 13 relationships.



Physical to Logical Model Normalization

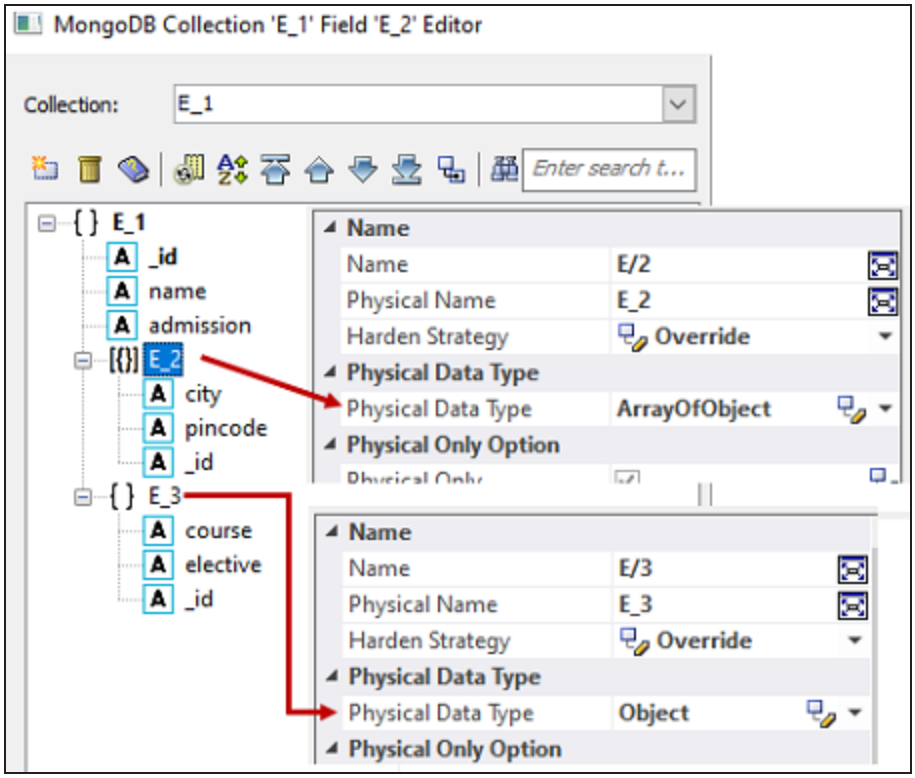
Due to the complete logical-physical model separation, the logical mode maintains the normalized form and the physical side always maintains the denormalized form. Whenever you switch from the physical model to the logical model, the model is auto-normalized and based on the embedding in the physical model, parent-child relationships are retained in the logical model.

For example, consider the following physical MongoDB model.

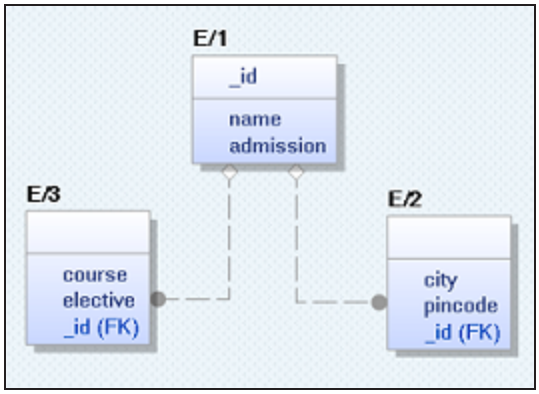


In this model:

- E_2 is embedded as an ArrayOfObject.
- E_3 is embedded as an Object.



Now, when you switch to the logical model, the model is normalized and based on the embedding, parent-child relationships are created. Object datatype is converted to one-to-one relationship and ArrayOfObject datatype is converted to one-to-many relationship.



Productivity and UI Enhancements

Several additions and enhancements have been implemented to improve erwin Data Modeler's productivity and usage experience. These enhancements are:

- Complete Compare
- Multiline Tabs
- Monolithic UI for Property Editors
- Properties Pane Enhancements
- Color Themes for Models
- Field Editor

Complete Compare

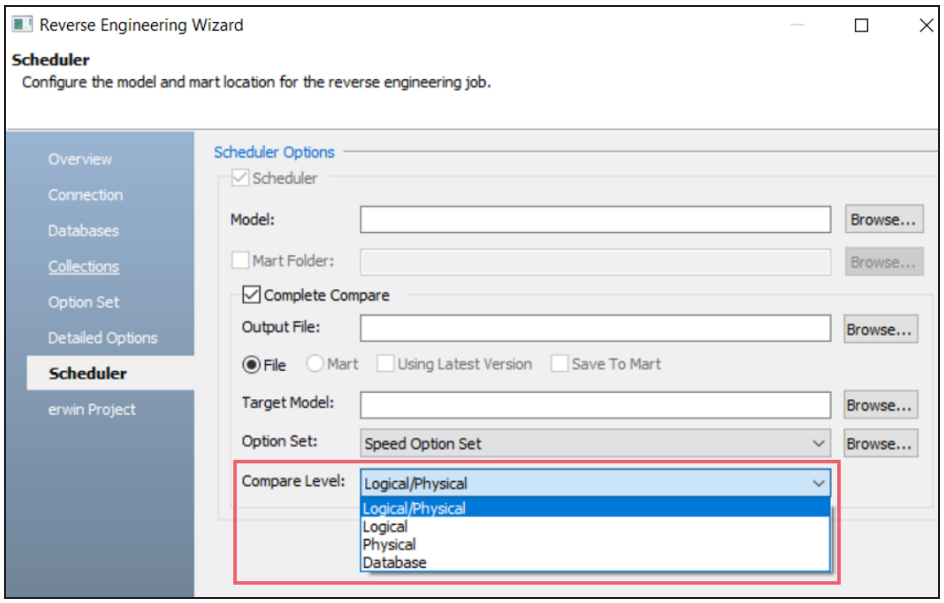
erwin DM has the following enhancement to the Complete Compare:

- **SCAPI Support**: You can view the Compete Compare results on the Command Prompt, bypassing the time it takes to process data into a UI.
- **Complete Compare Levels**:

  The Complete Compare can now compare two models based on levels during the reverse engineering. You can compare the models based on the levels such as Logical/Phycal, Logical, Physical, or Database. The Scheduler supports the complete compare level, reducing the time to compare models.

  The screenshot below displays the Compare Level option for Scheduler on the Reverse

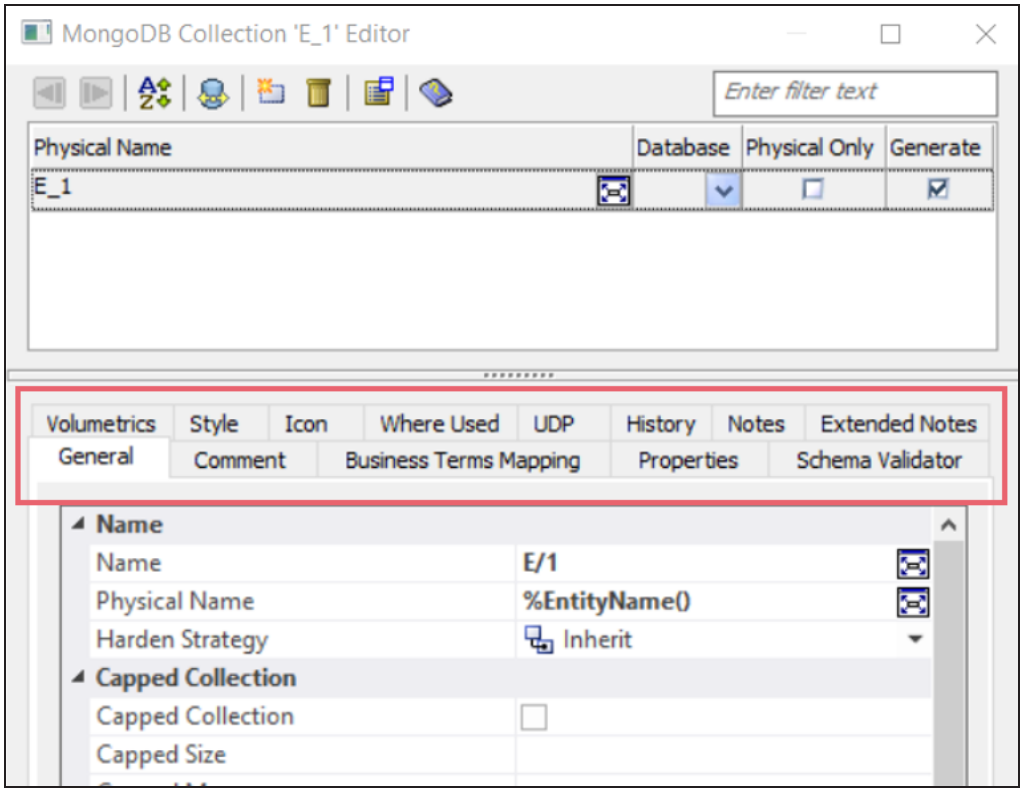
Engineering Wizard.



Multiline Tabs

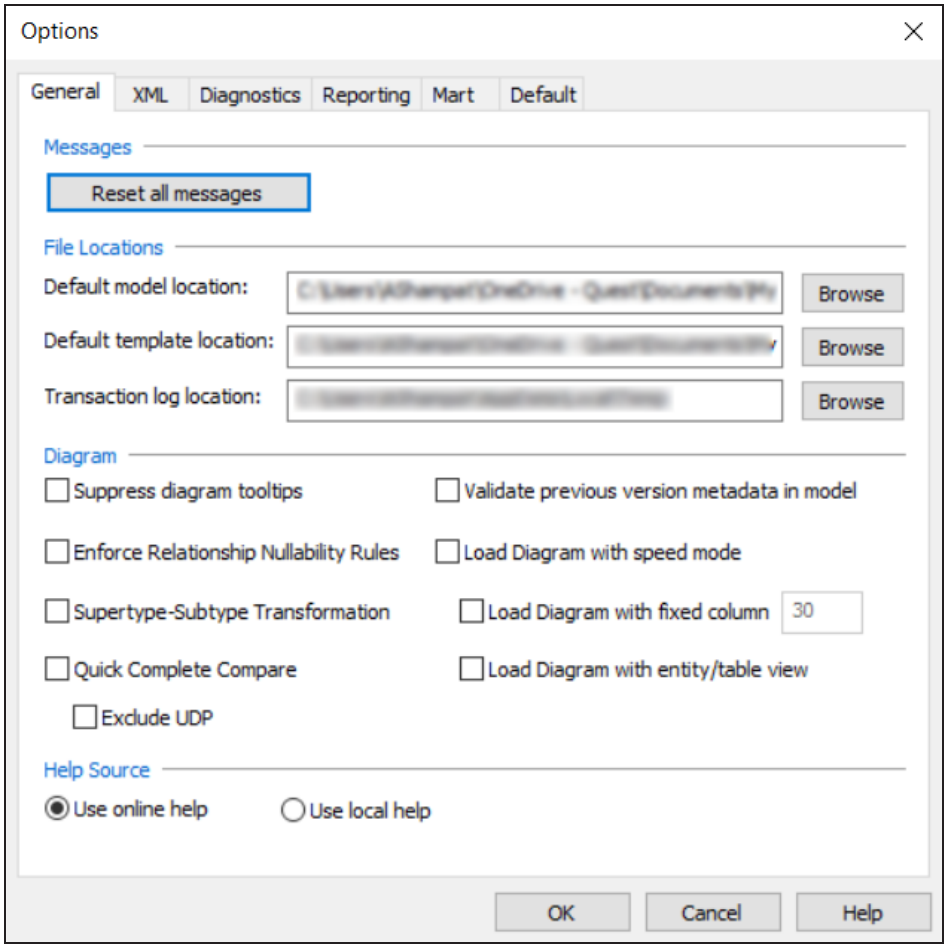
The multiline tab feature lets you to view all property tabs in an editor across multiple rows. This enhancement provides immediate access to all tabs without horizontal scrolling, streamlines navigation, and improves overall efficiency.

The screenshot below is an example of the MongoDB Collection Editor, which displays how all the available property tabs are arranged in multiple rows across the editor.

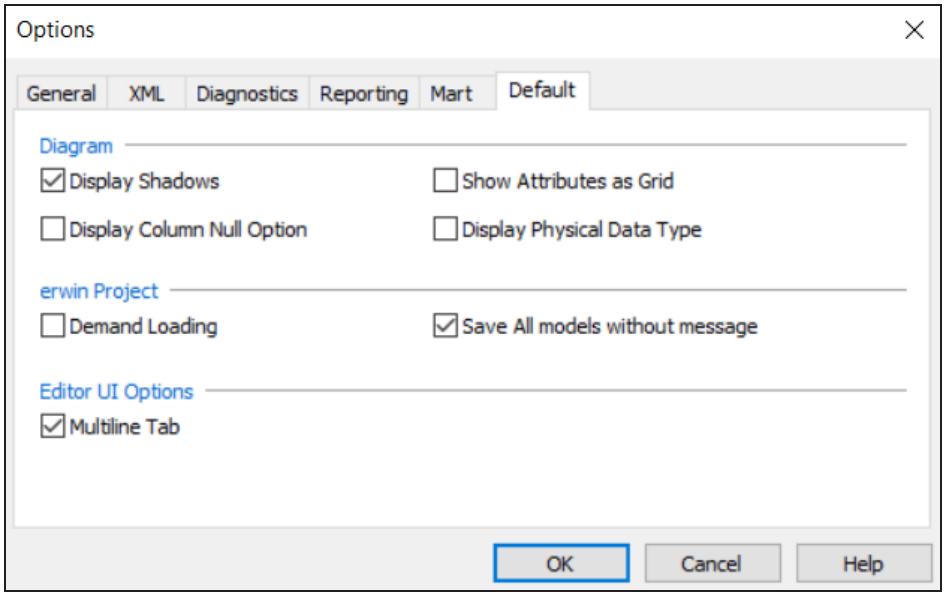


To enable the multiline tab, follow these steps:

1. Click **Tools** > **Options**.
   The Options screen appears.


Options
General
XML
Diagnostics
Reporting
Mart
Default
Messages
Reset all messages
File Locations
Default model location:
Browse
Default template location:
Browse
Transaction log location:
Browse
Diagram
Suppress diagram tooltips
Validate previous version metadata in model
Enforce Relationship Nullability Rules
Load Diagram with speed mode
Supertype-Subtype Transformation
Load Diagram with fixed column
30
Quick Complete Compare
Load Diagram with entity/table view
Exclude UDP
Help Source
Use online help
Use local help
OK
Cancel
Help

2. On the Default tab, under the Editor UI Options, select **Multiline Tab**.



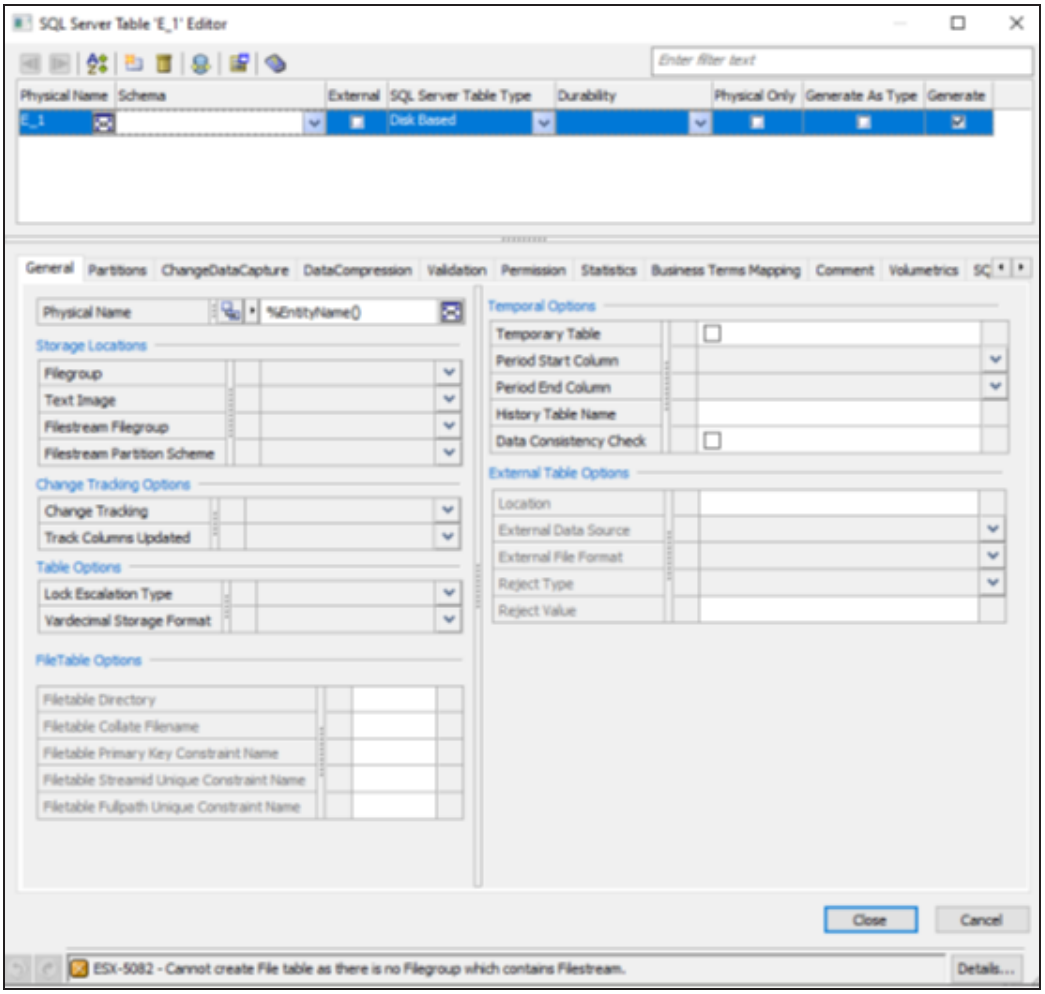
3. Click **OK**.

Monolithic UI for Property Editors

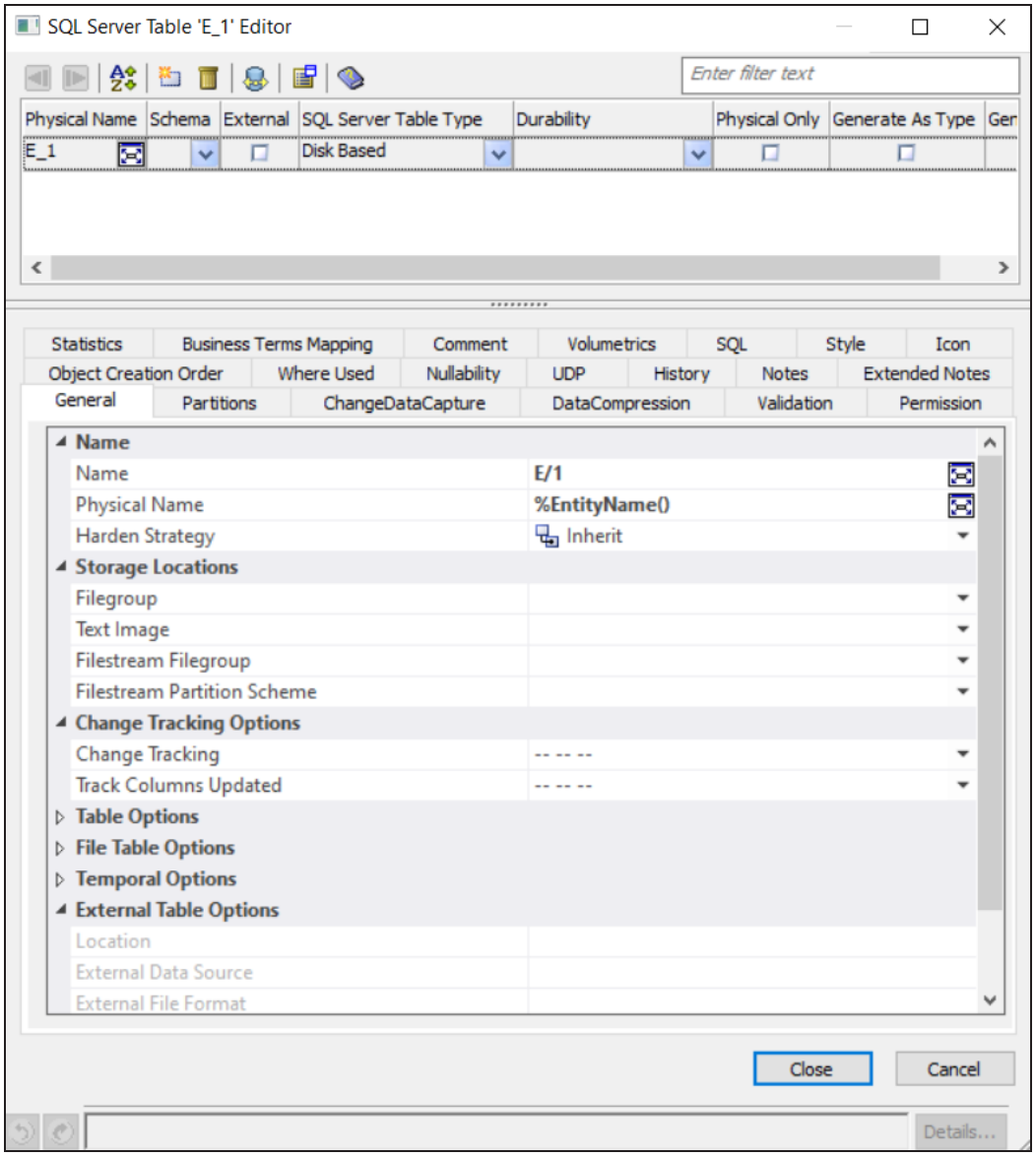
erwin DM 14.0 introduces a new monolithic UI design for all property editors. This new UI now adapts to any screen resolution, eliminates data truncation, and displays clutter-free data. Additionally, the collapsible and expandable category sections enables you to view and edit only necessary properties in an editor based on your requirements.

The screenshots below display the difference between the old and new monolithic UI with collapsible sections, using SQL Server Table Editor as an example.

Before:



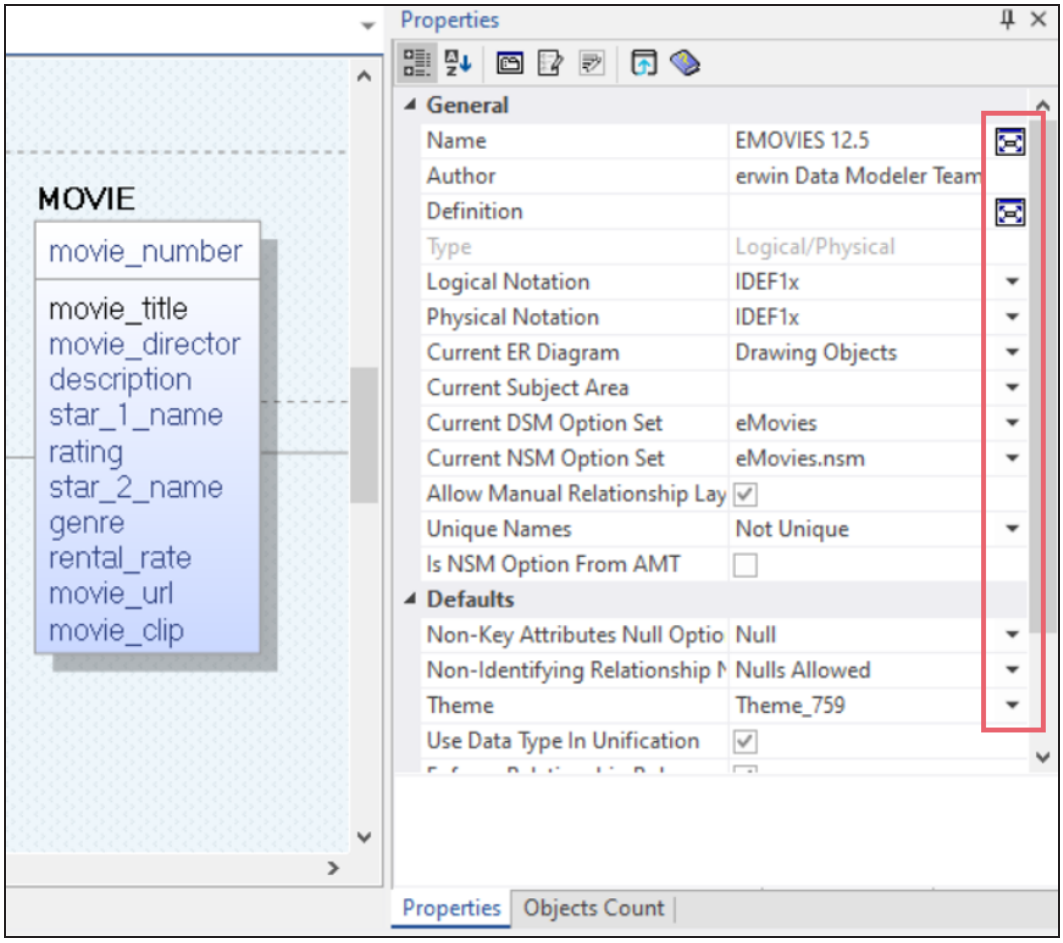
Monolithic UI for erwin DM 14.0:



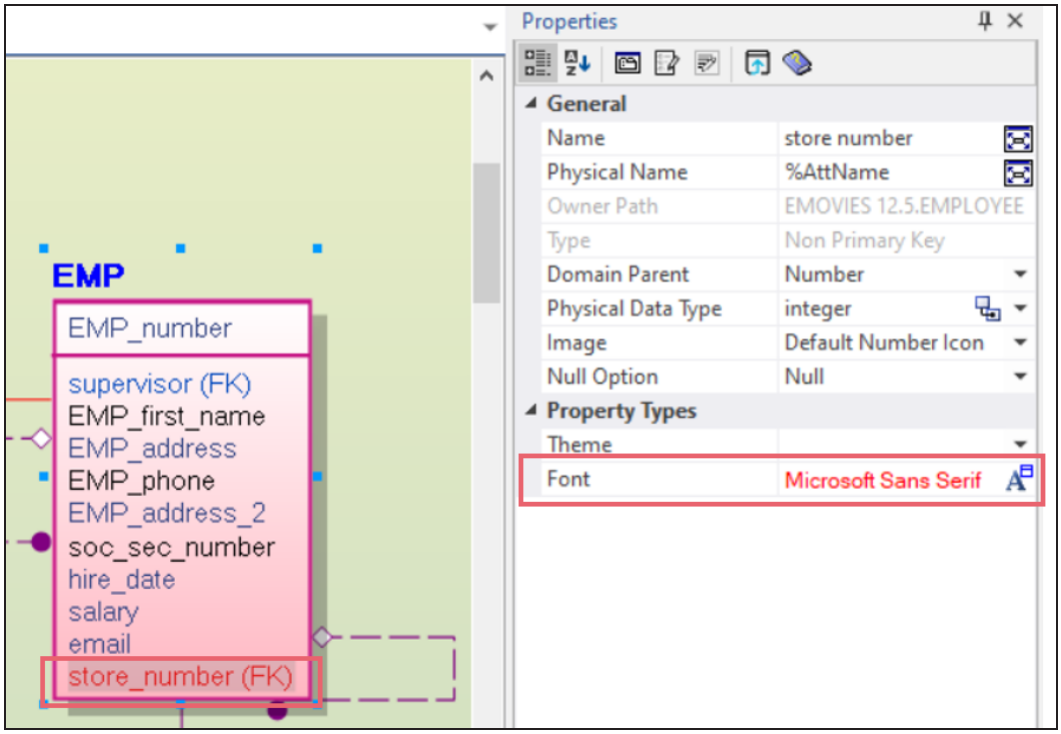
Properties Pane Enhancements

The model properties pane now displays buttons (such as editor and drop-down buttons) next to each property, which lets you to easily identify that the properties are editable.

The screenshot below displays the new buttons added next to the property editor by default.



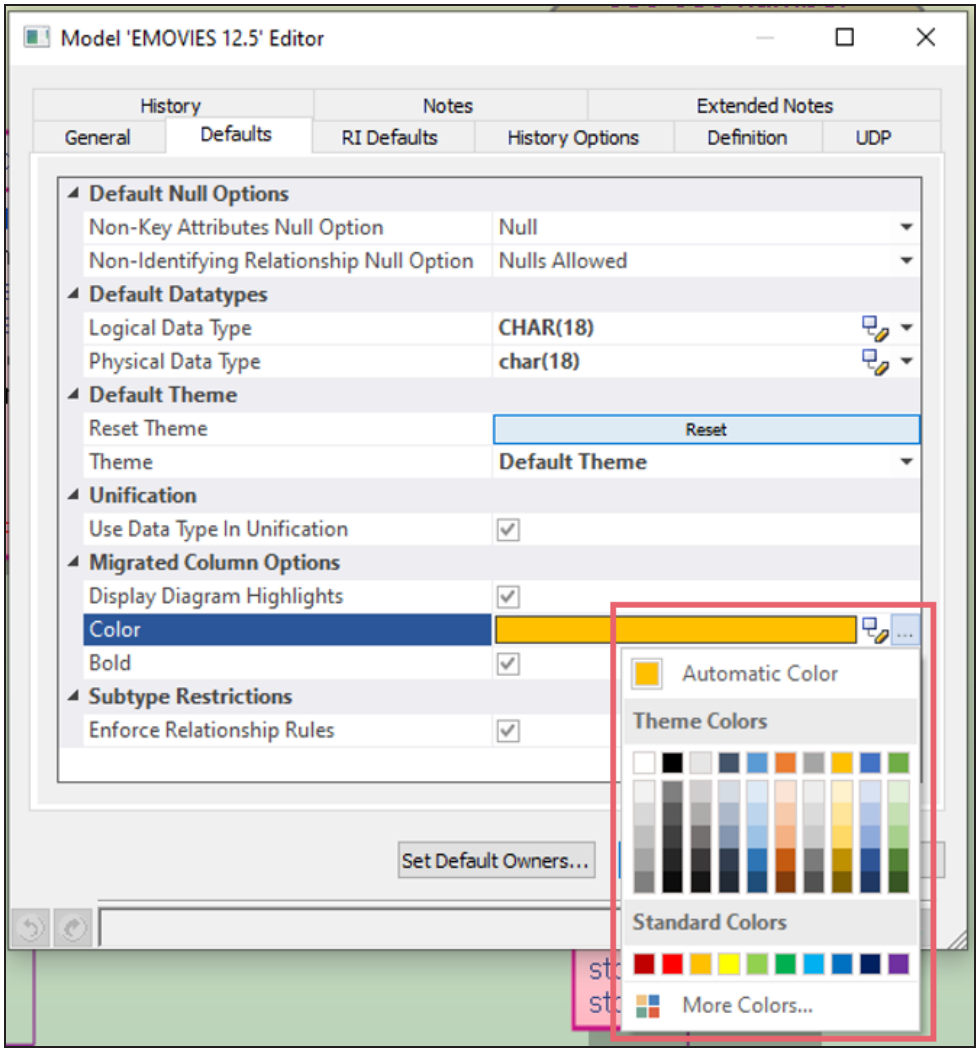
For example, the screenshot below displays the color and font style icon of the column on the Properties pane based on the selected column.



Color Themes for Migrated Columns

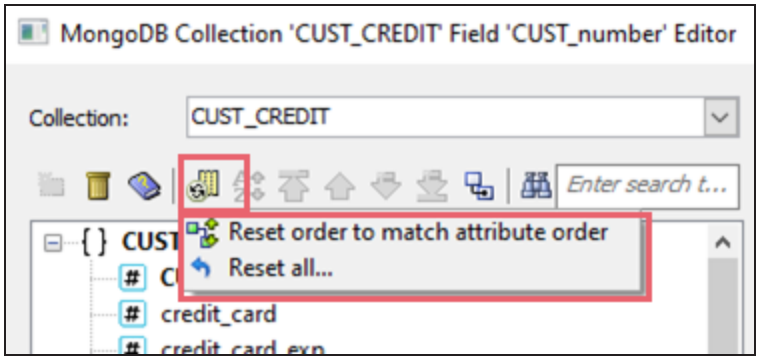
You can now set theme and custom colors for the migrated columns in addition to the standard colors.

To set the theme and custom colors, click **Model** > **Model Properties** on the ribbon. Then click **Defaults** tab, and use the color options under **Migrated Column Options**.



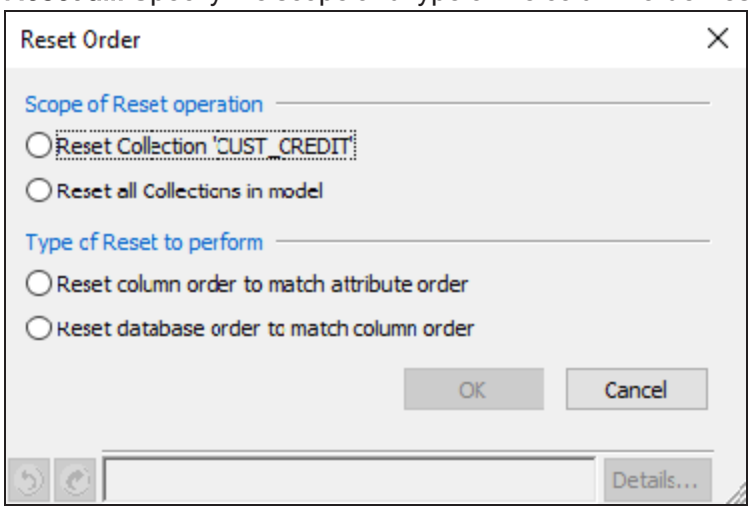
Field Editor

The Reset Order option has been added to the Field Editor for NoSQL models.



Using this option, you can do one of the following:

- **Reset order to match attribute order**: Reset field order to match the order of the corresponding attributes in the logical model.
- **Reset all:** Specify the scope and type of the column order reset actions.



erwin Mart Portal Enhancements

erwin Mart Portal has undergone several enhancements:

- The UI has been redesigned to provide a modern experience with better usability.
- erwin Mart Portal configuration:
  - Authentication: You can now create a list of up to 10 Active Directory domains. This enables you to access and add users from multiple domains.
  - Database:
    - SQL Server mart now supports Windows Authentication.
    - PostgreSQL mart can now be based on any schema unlike earlier, where you could connect only to the public schema.
- The DM Connect for DI feature has been upgraded to support erwin DI v14.0.

erwin ER360 Features and Enhancements

erwin ER360 offers two new modules:

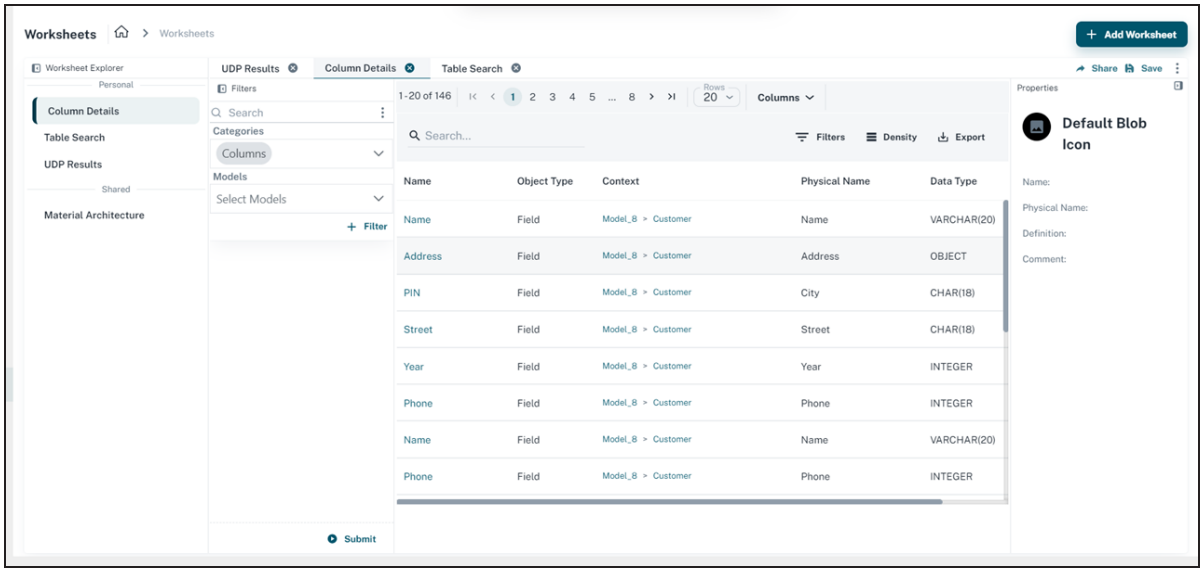
- Worksheet is an extension of the Global Search feature and offers advanced filters and search configurations. Worksheet search results enable you to navigate to the resultant objects via links.
- Collection enables you to save metadata objects of interest in a collection of objects and watch them for any updates.

Worksheet

Worksheets enhance metadata reporting by offering flexible search and browse capabilities. You can create worksheets, search metadata, apply filters, and select from various object types to optimize the data in the worksheets. Worksheets support detailed filtering using multiple properties and object types, providing you the ability to manage and derive results. The detailed filtering option supports all objects in a database and lets you create custom worksheets with multiple parameters to show detailed combined results.

The Worksheets module uses multiple organized panes that lets you explore the available worksheets, apply filters, view filtered data and properties, and share worksheets with other users. Shared worksheets are dynamically updated as and when they are edited and submitted.

To view worksheets, on the application menu click (**Worksheet**).
The Worksheet module appears and displays different panes that let you select, filter, and manage your worksheets.



The following list explains the usage of different panes in this module:

Worksheet Explorer

This pane displays the custom worksheets you created and those shared with you. Selecting a worksheet displays filter options and detailed results. To add a worksheet, refer to the Adding Worksheets section.

Filters

This pane enables you to search for data available in the erwin ER360, and categorize your search results based on database objects, selected models, common and object attributes, and user defined properties. Based on the filters you set, the results are displayed in the details pane next to it. To know more about filtering data, refer to the Filtering Worksheet Data section.

Properties

The Properties pane displays properties of the objects selected in the details pane. This pane is collapsible, and the list of data displayed here varies based on the selected object type.

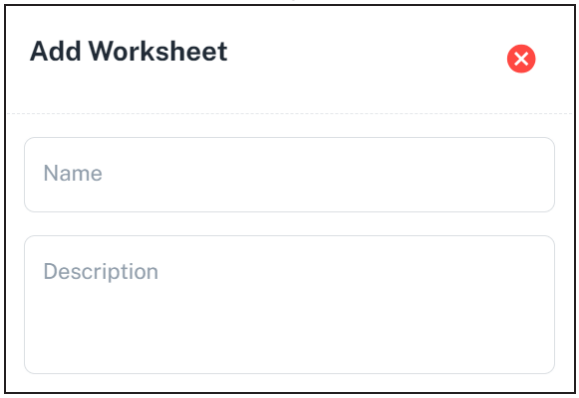
Adding and managing the worksheets involves the following:

- Adding Worksheets
- Filtering Worksheet Data
- Sharing Worksheets

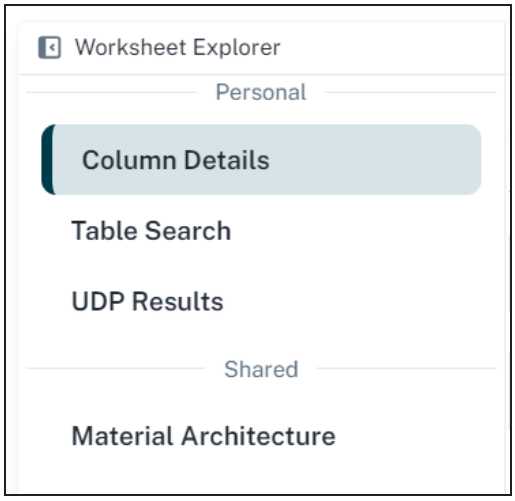
Adding Worksheets

To add a worksheets, follow these steps:

1. On the Worksheets page, click **Add Worksheet**.
   The Add Worksheet page appears.



2. Enter a name and description for the worksheet.
3. Click **Save**.
   The new worksheet is added to the Worksheet Explorer pane.



You can also add and open multiple worksheets simultaneously and toggle between them.

Filtering Worksheet Data

Once you have created a new worksheet, you can set up custom filters and save the worksheet. The Filters pane provides detailed filter options to derive optimized results by searching available data, filtering using database objects and different categories, common and object attributes, and user-defined properties.

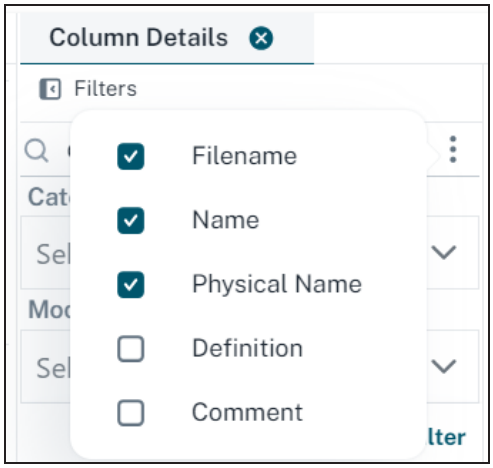
To filter worksheet data, follow these steps:

1. Select a worksheet.
2. In the Filters pane, use the following options to filter the data:

   Search

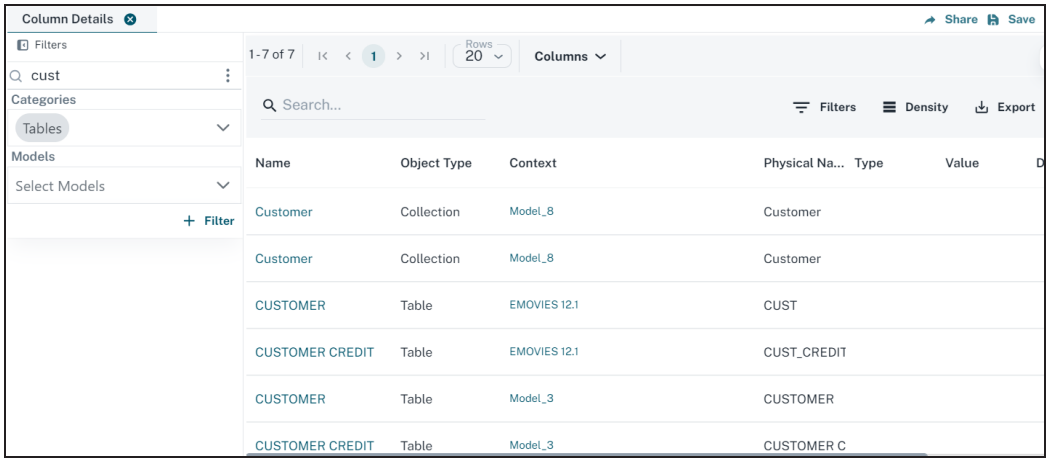
   Use this option to search for any data in your catalogs. You can search columns, tables, definitions, comments, or physical names of any models in the repository based on the search keyword.

   Additionally, click ⋮ to view more search filters to refine your search results. Select required objects from the list to search only based on the selected objects.



Categories

Use this option to filter search results based on the available objects in a model. Selecting a category narrows the search results within one or more selected objects. For example, when you select Tables in the Categories list, the search results display only tables that contain the search keyword.
The screenshot below displays table specific search results with the keyword "cust" in it.



If you do not select categories, results are filtered for all the objects by default.

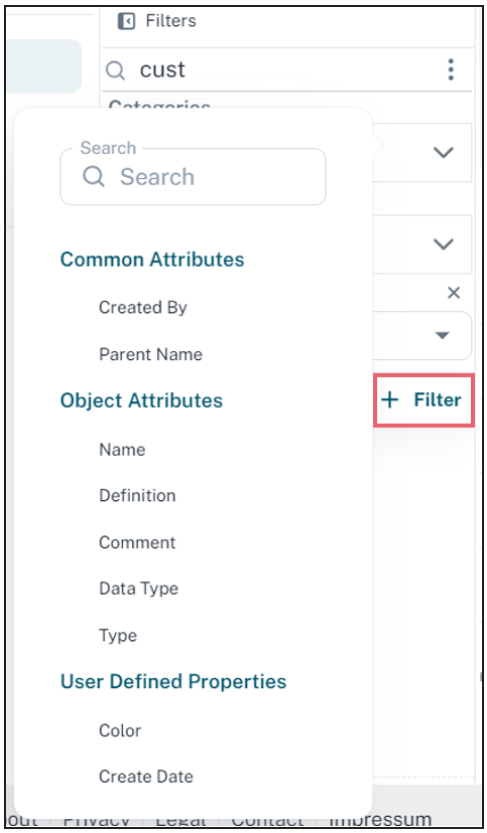
Models

Use this option to filter search results based on the selected models. By default, the results are filtered for all the models available in the erwin ER360.

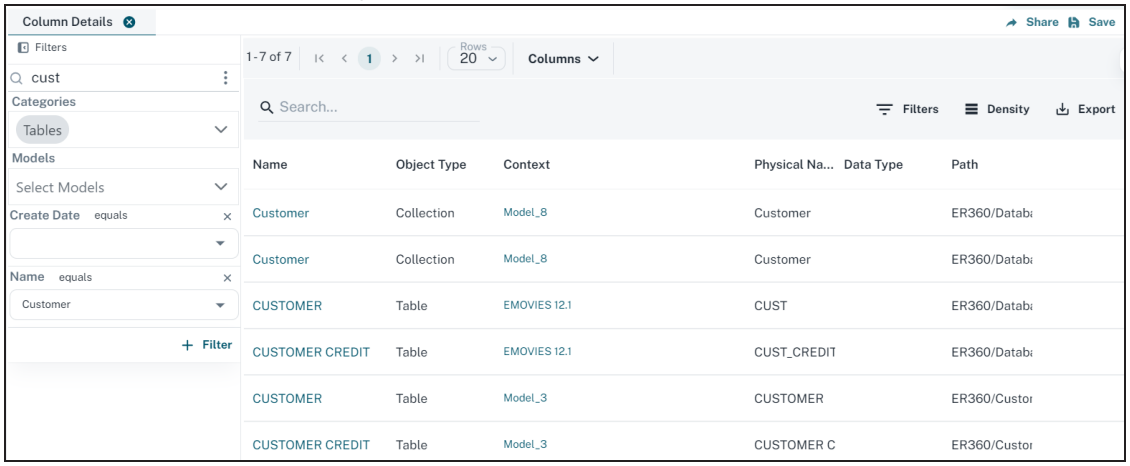
Additional Filters

Use this option to filter search results based on the selected common and object attributes, and user defined properties. You can add multiple filters based on your requirement and use operators to filter results.

To add additional filters, click **+ Filter**. Then, select the required options from the list.



3. Click **Submit**.
   The filtered results are displayed in the pane next to the Filters pane.

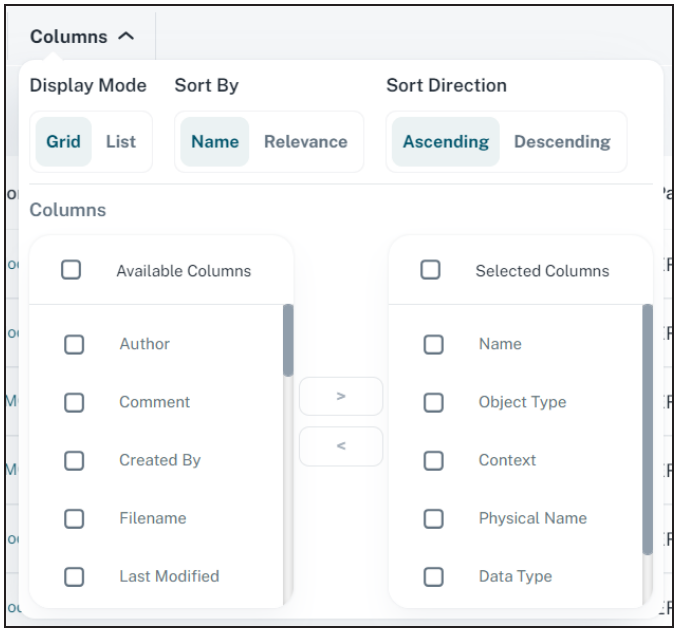


4. Click **Save**.
   This saves the worksheet with the filter that you have configured.

Once the search results are filtered, in the details pane, you can use the following options to refine the results:

Columns

Use this option to view filtered results in a list or grid view. Additionally, you can sort the data and choose to display the relevant columns for the filtered results.

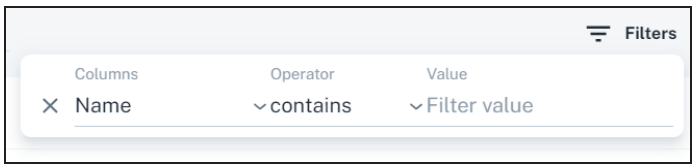


Search

Use this option to search within filtered search results. Enter a keyword in the search box to view results based on the keyword.

Filters

Use this option to filter search results based on a column using operators.



Density

Use this option to view the filter results based on you accessibility needs. You can choose Compact, Standard, or Comfortable view. The Compact view reduces the space between the rows, while the Comfortable view uses more space to display the results.

Export

Use this option to export search results to a CSV file or print them.

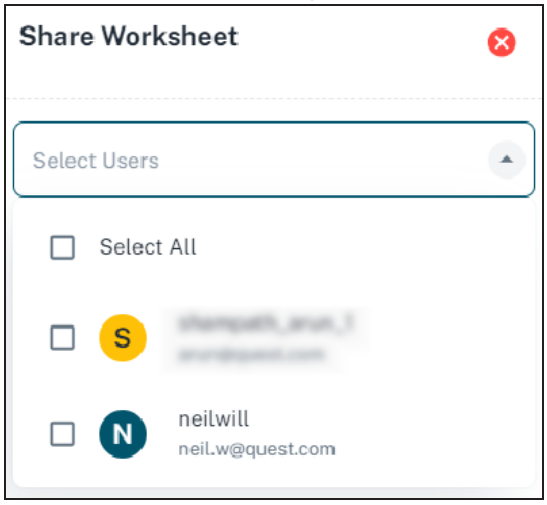
Sharing Worksheet

Worksheet

You can share the custom worksheet you created with other users and view worksheets that are shared with you.

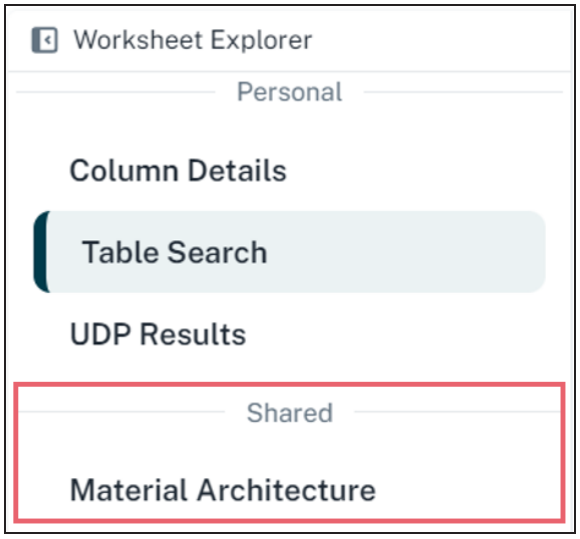
To share worksheets, follow these steps:

1. Select a worksheet, and click **Share**.
   The Share worksheet page appears.



2. Select required users.
3. Click **Save**.
   The worksheet is shared to the selected users and displays under the Shared section.

The worksheets that are shared with you by the other users are displayed under the Shared section in the Worksheet Explorer pane.

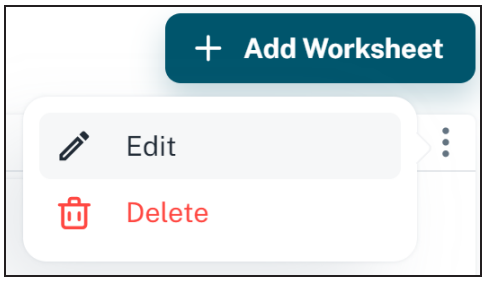


Apart from creating and sharing worksheets, you can edit the worksheet's name and description or delete them.

To edit or delete worksheets, click ⋮.

The following options appear.

- **Edit**: Use this option to edit the name and description of the worksheet.
- **Delete**: Use this option to delete the worksheet.



Collections

Collection enables you to save metadata objects of interest from your model in a collection of objects. For example, if you want to keep track of an attribute, email, and all related objects in your model, you can create a collection of all such attributes and related objects. You can share collections with your team members and collaborate to add objects to them.

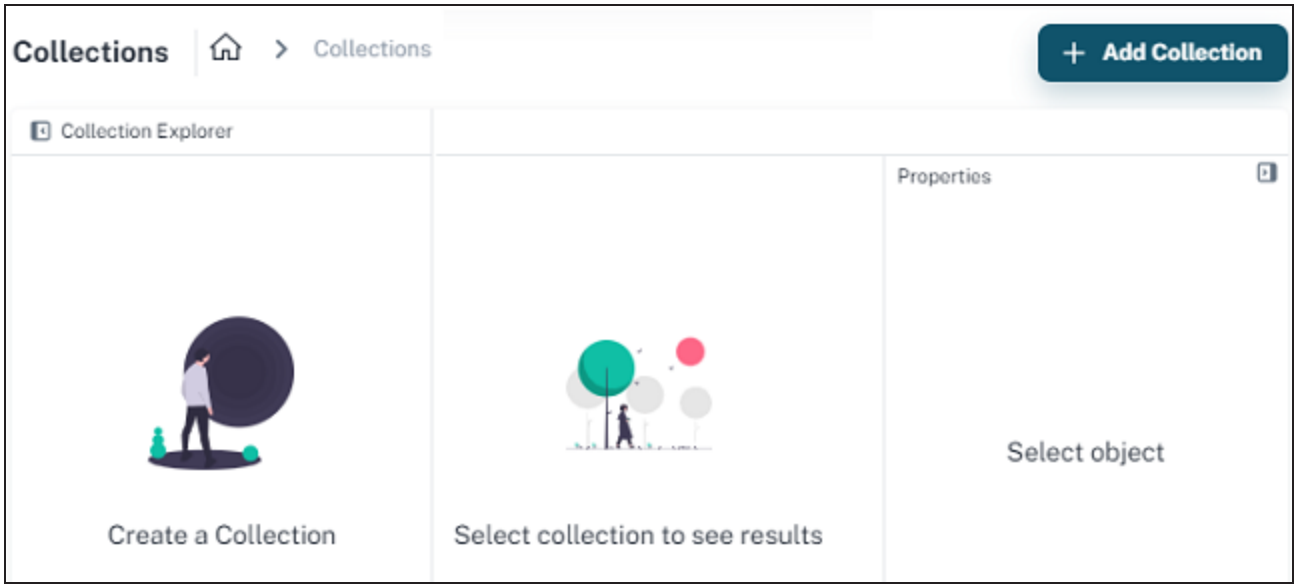
Creating collections involves:

1. Creating collections.
2. Adding objects to collections.
3. Sharing collections for collaboration.

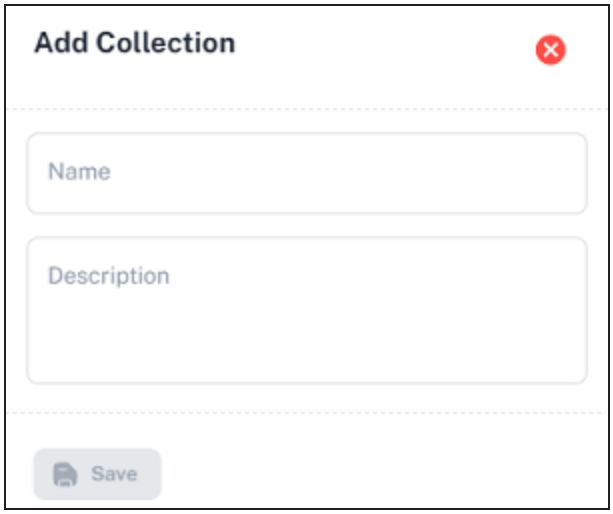
Creating Collections

To create collections, follow these steps:

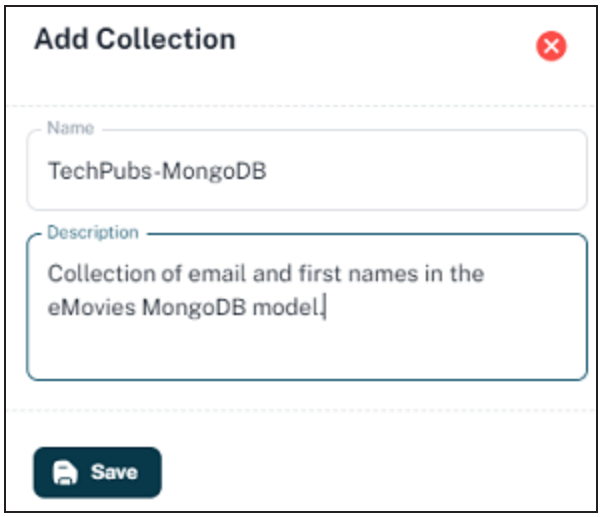
1. Go to **Application Menu** > **Collection**.
   The Collections page appears.



2. Click **Add Collection**.
   The Add Collection section appears.

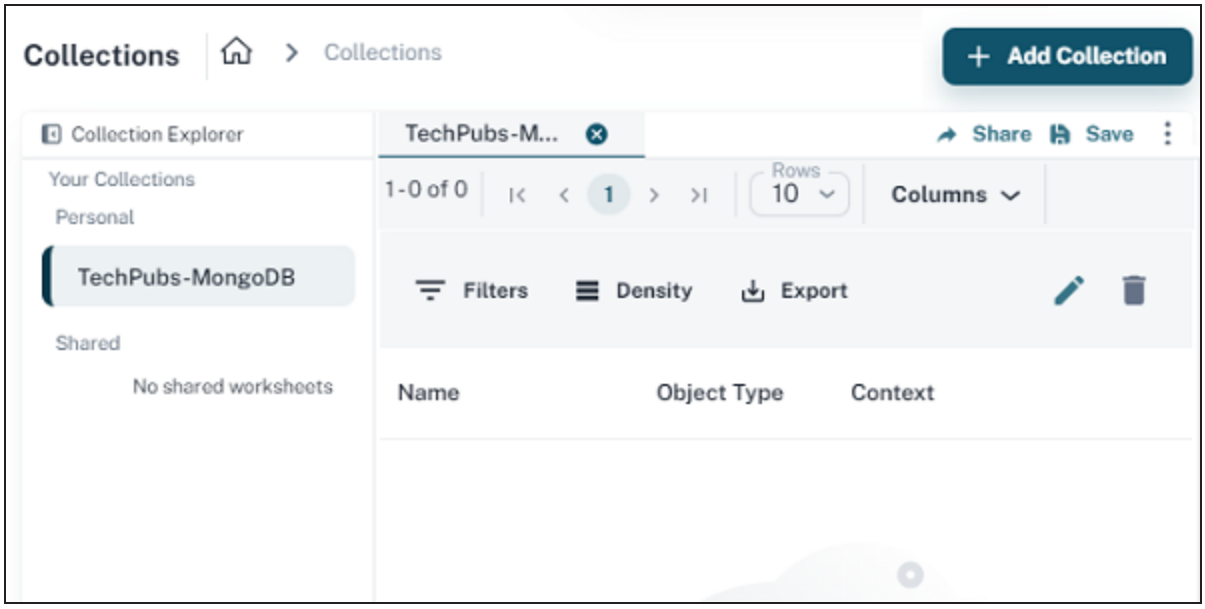


3. Enter the **Name** and **Description** of the collection.



4. Click **Save**.

   An empty collection is created. You can now add objects to the collection.



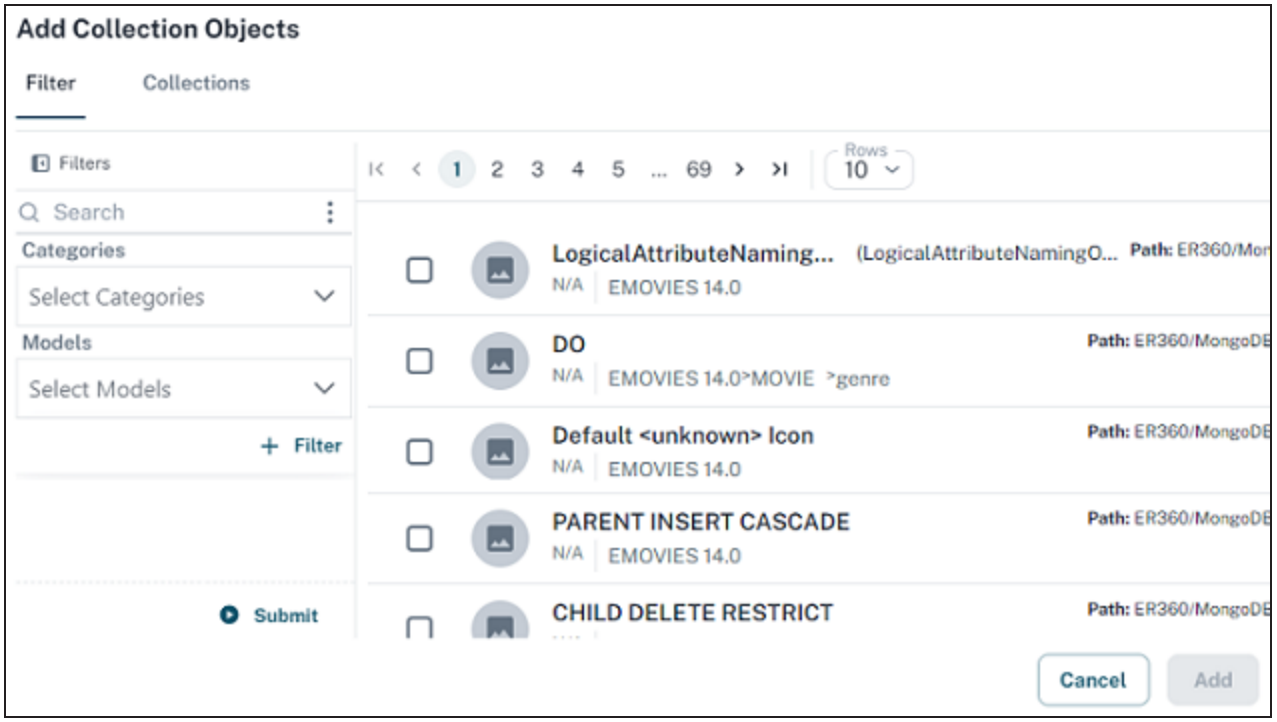
Adding Objects to Collections

To add objects to collections, follow these steps:

1. In the Collection Explorer, select the required collection and click .

   The collection opens in edit mode.

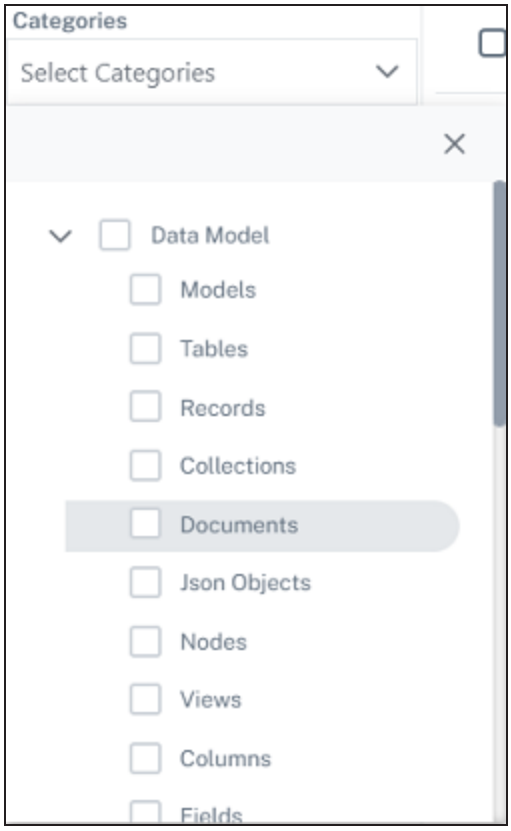
2. Click .
   The Add Collection Objects page opens.



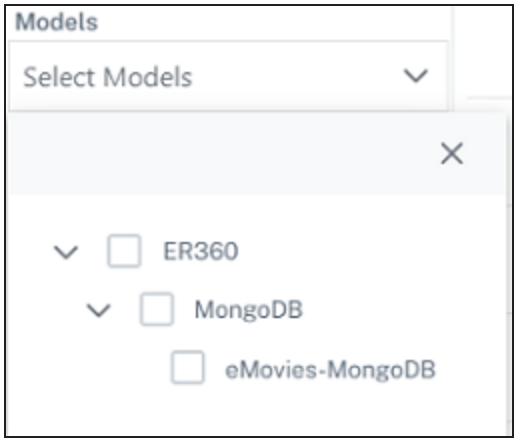
It has the following sections:

- Filter options to search for objects based on selected criteria
- Collections list to select objects from other collections to add to the current collection
- List of objects

3. Under Filters, in the Search box, enter the object name that you want to add to the collection.
4. Under Categories, select the type of object.

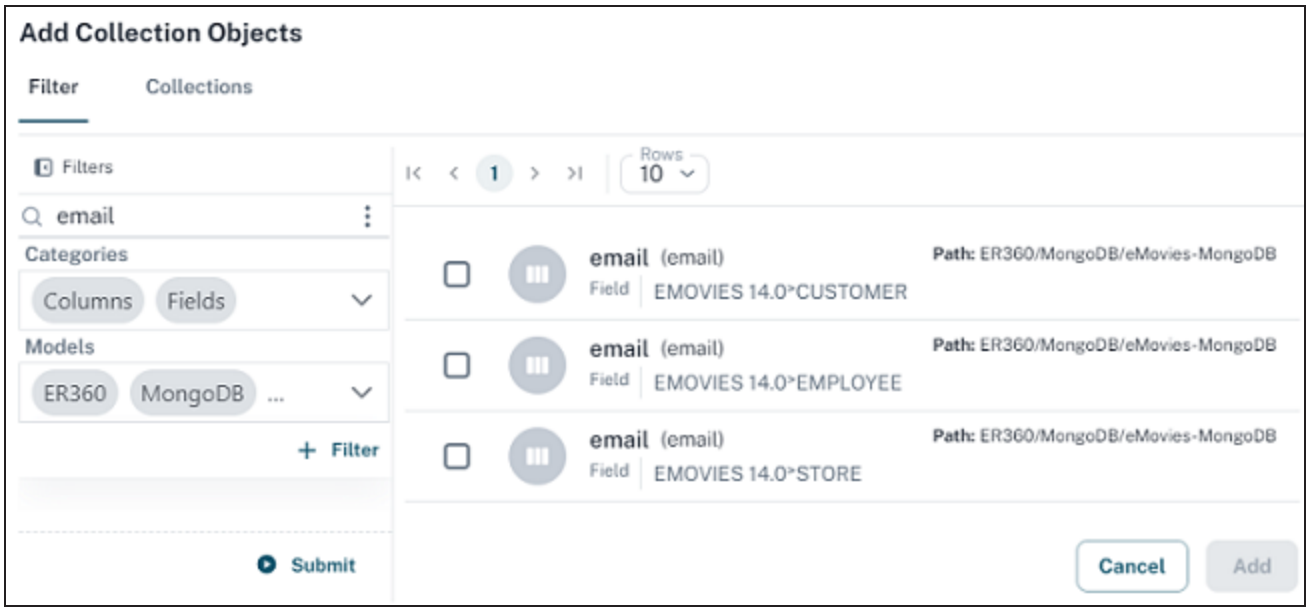


5. Under Models, select the models in which you want to search for objects.



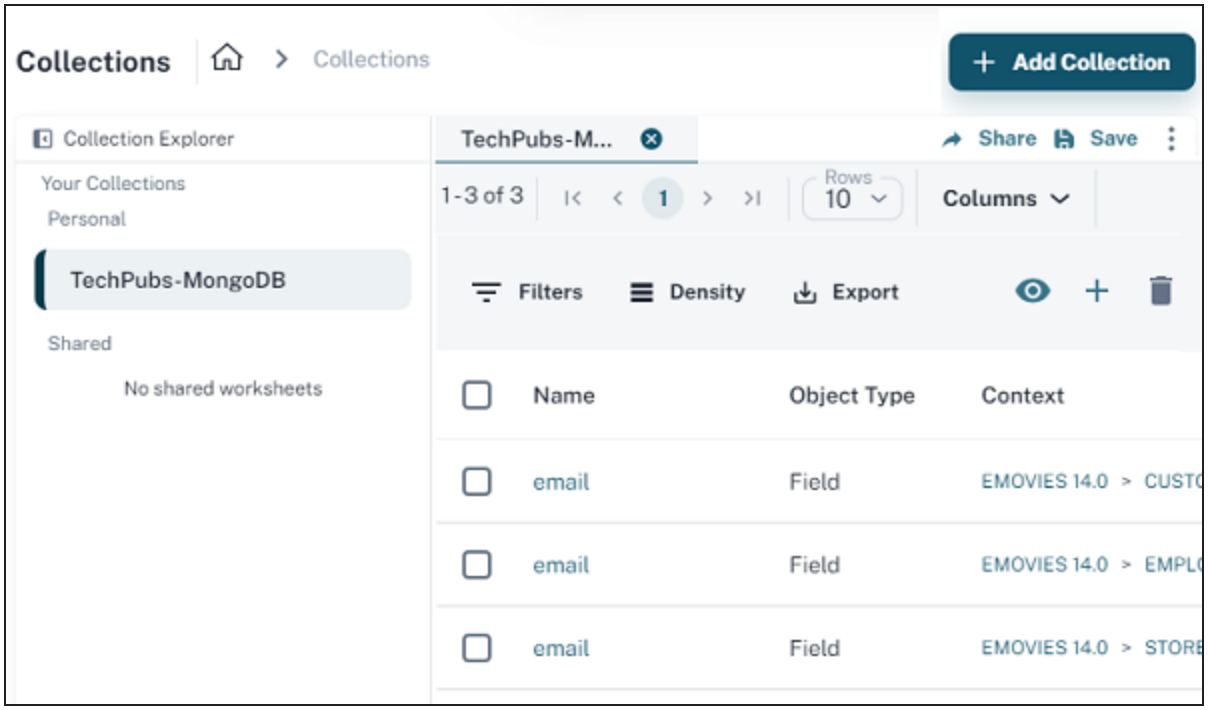
6. Click **Submit**.

Based on the Search input, category, and model that you configured, search results are displayed.



7. Select the objects that you want to add to the collection and click **Add**.

   The selected objects are added to your collection.



The object name and path are linked to the object and the respective model/table. You can click the links to navigate to the Metadata browser and view them. Selecting any object in the collection list displays its properties in the Properties pane.

You can also export the collection to a CSV file or print it.

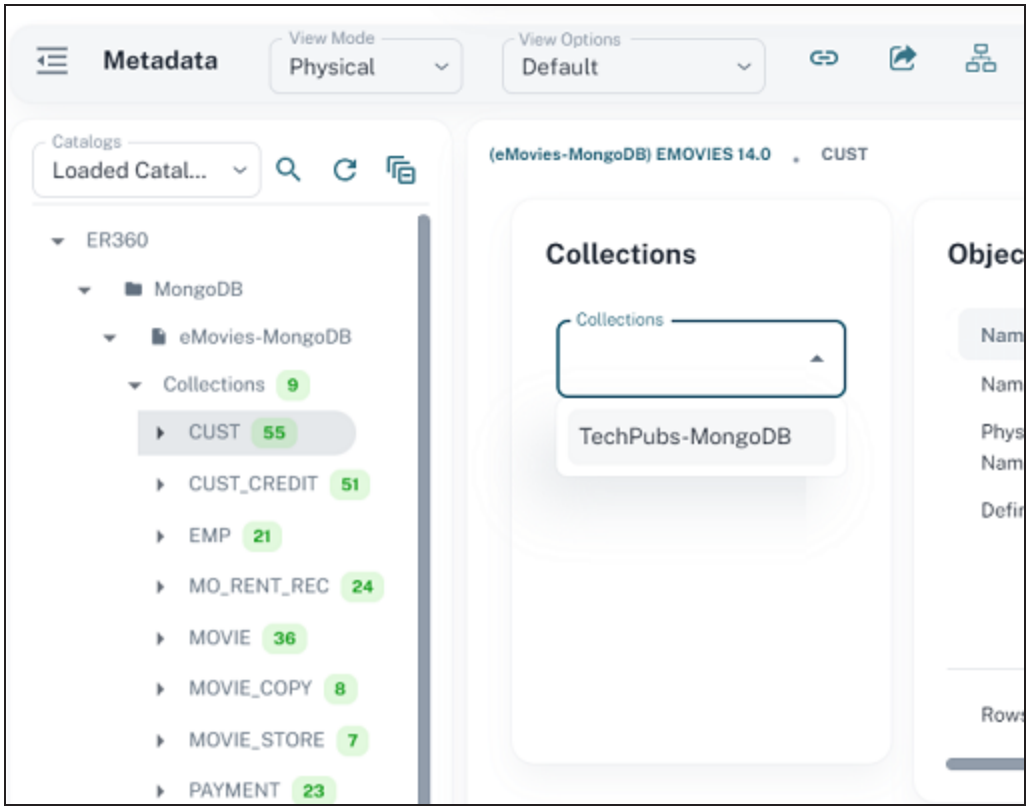
Alternatively, you can also add objects to collections via the Metadata browser.

To add objects to collections via Metadata browser, follow these steps:

1. Go to **Application Menu** > **Browse**.

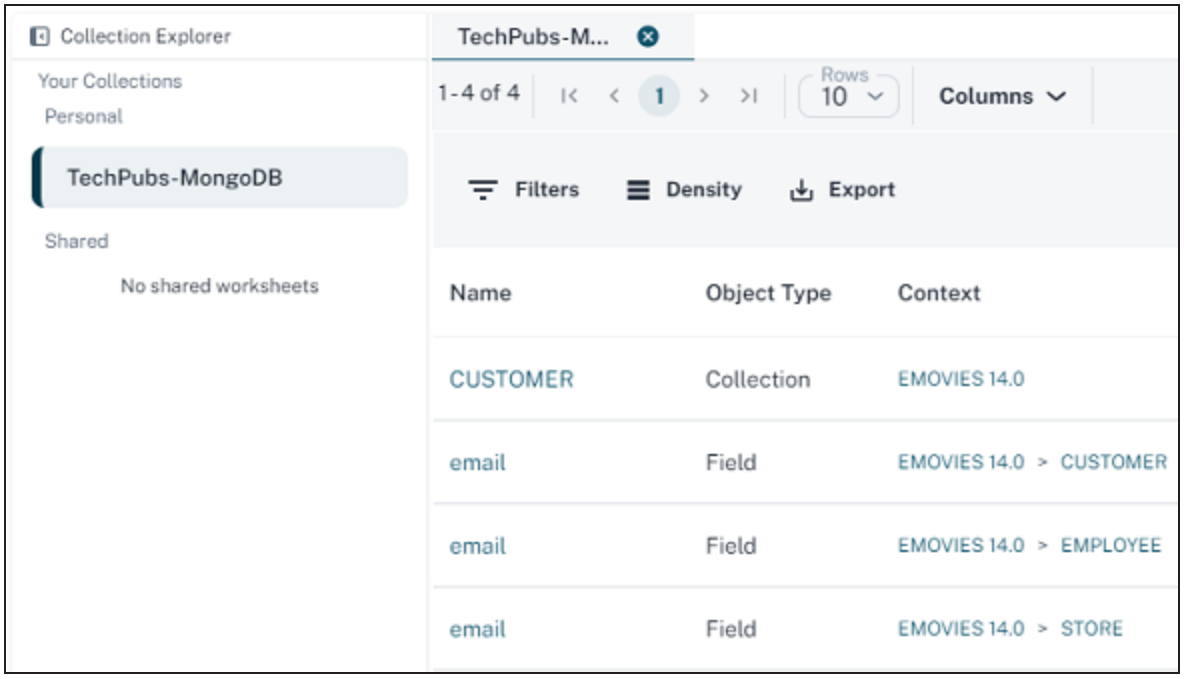
   The Metadata page appears.

2. In the catalogs tree, select the object that you want to add to a collection.



3. Then, under Collections, select the collection to which you want to add the object.
4. Navigate to the Collection module and select your collection.

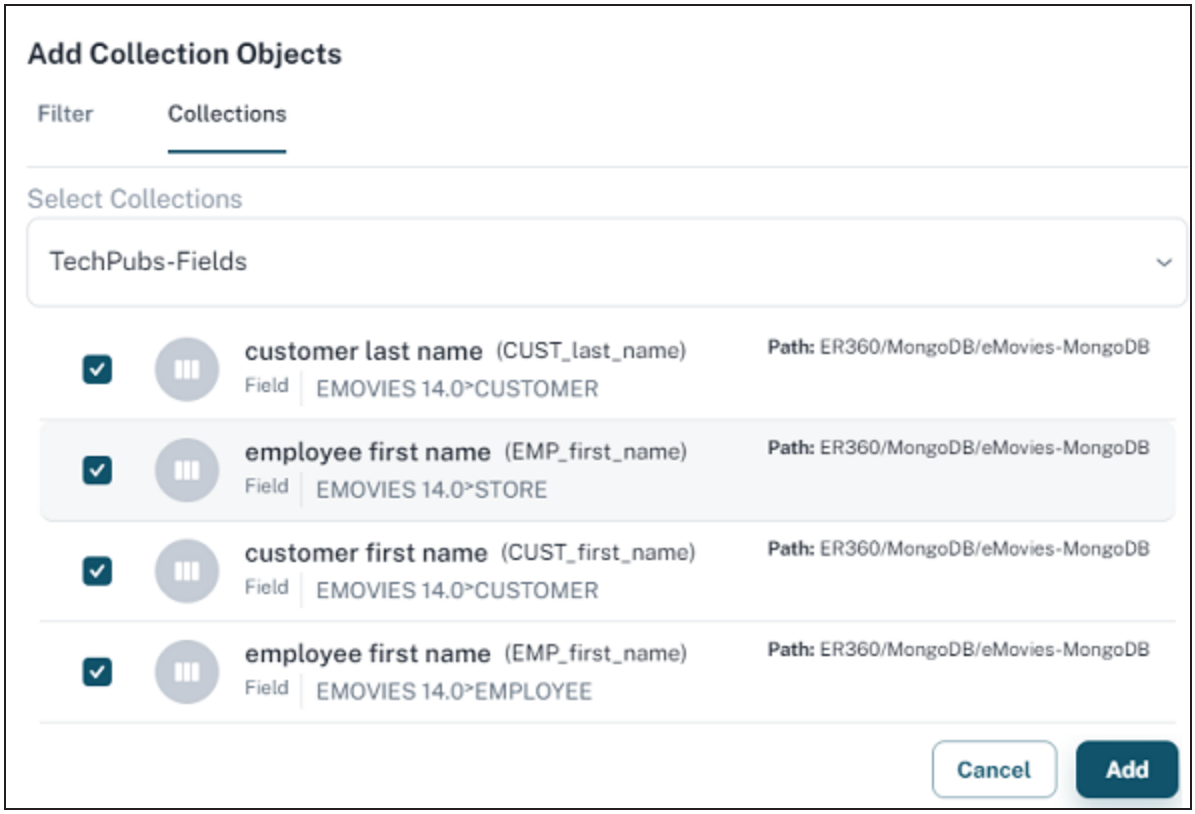
   The Customer object is added to your collection.



You can also add objects from one collection to another. To do so, follow these steps:

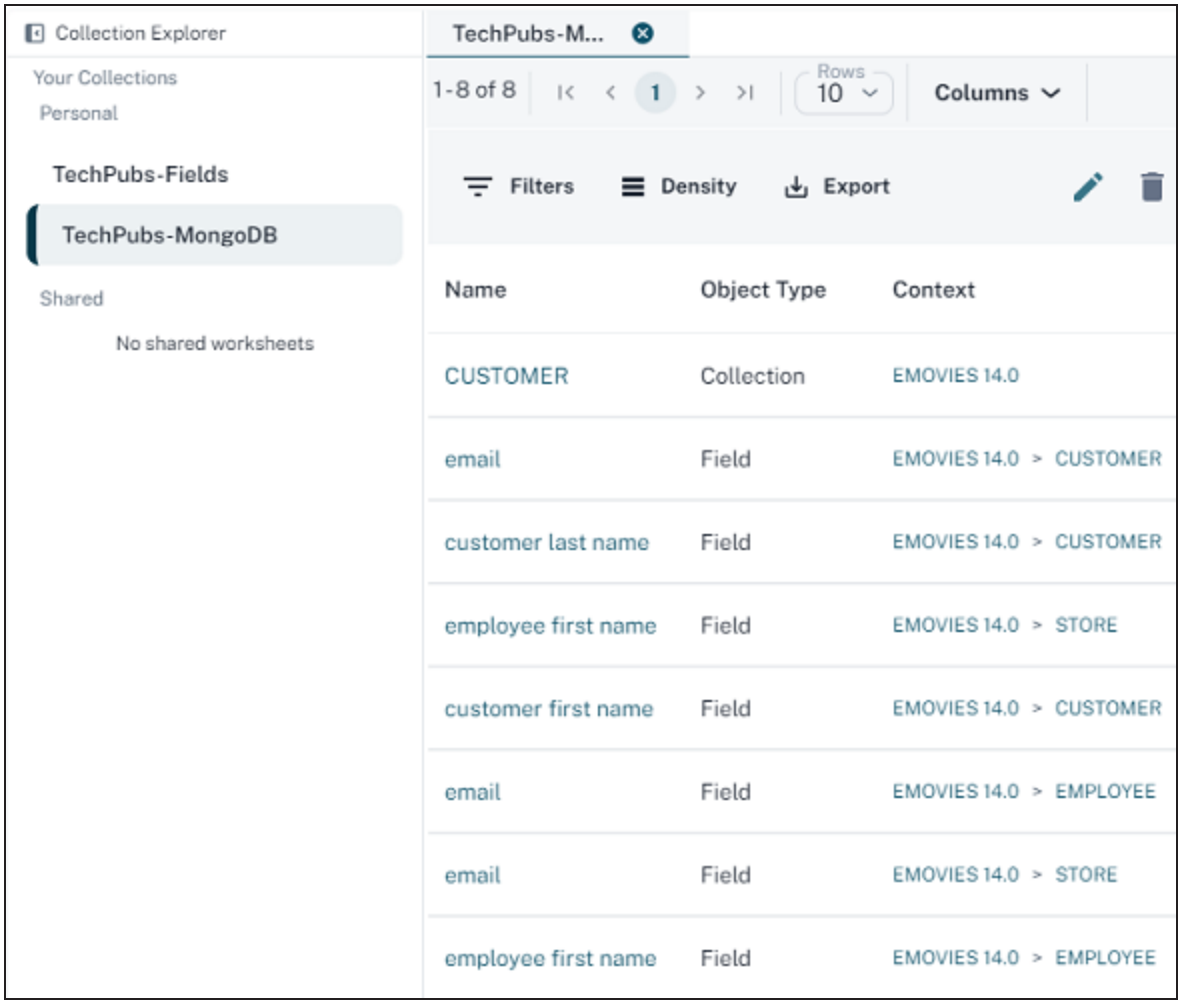
1. On the Add Collection Objects page, click the **Collections** tab.
2. Under Select Collections, select the collection from which you want to add objects.

   A list of objects in the selected collections appears.



3. Select the required objects and click **Add**.

   The objects from the selected collection are added to your collection.

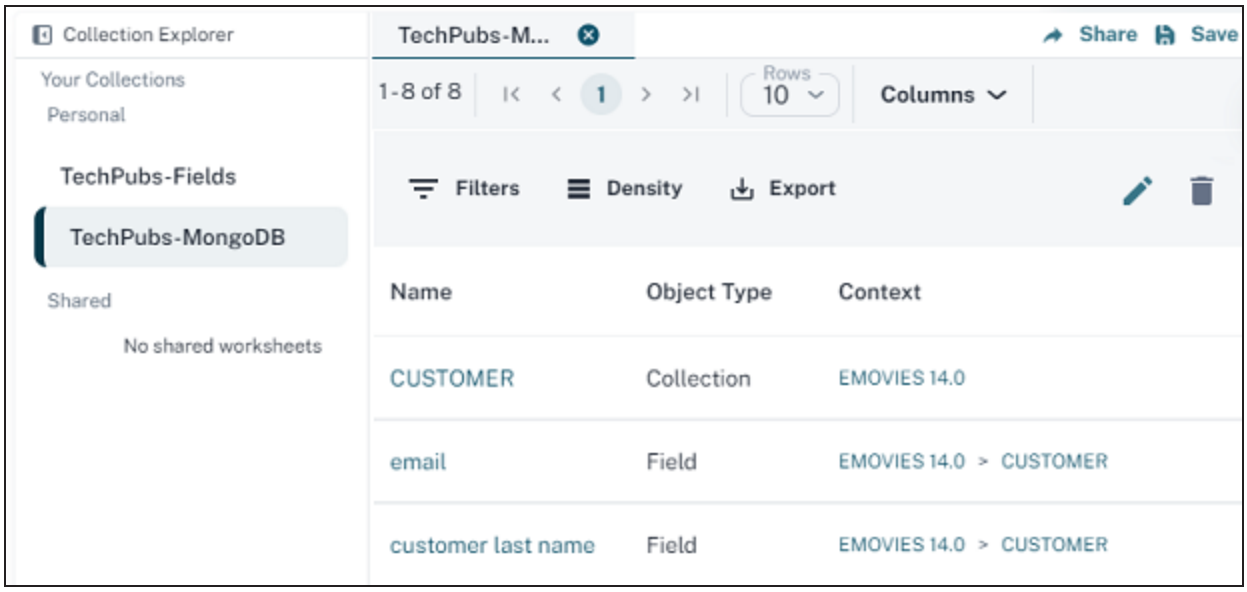


Sharing Collections for Collaboration

You can share your collection with your team members for collaboration. They can then add, edit, or view your collection.

To share collections, follow these steps:

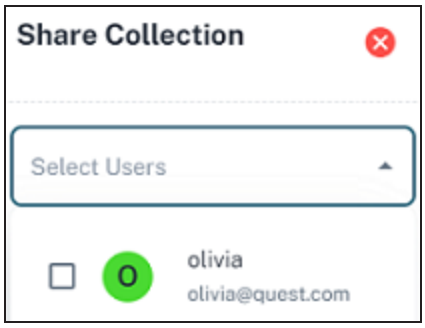
1. Open your collection.



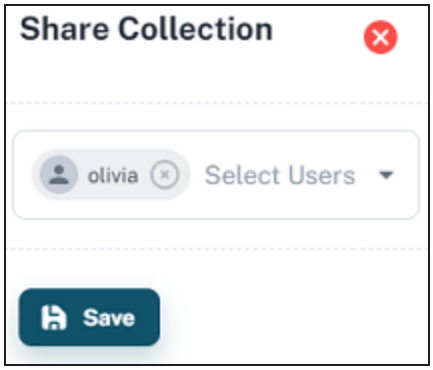
2. Click **Share**.

   The Share Collection section appears.

3. Click **Select Users**.



4. Select your team members.



5. Click **Save**.

   The shared collection appears under Collection Explorer > Shared section for your team members.