erwin Data Modeler

Git Support

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

You can connect erwin DM to source control repositories via erwin Mart Portal. This enables you to save Forward Engineering (FE) scripts for a Mart model to your source control repository.

> You cannot store FE scripts or DDL on a erwin Mart Portal but only erwin models.

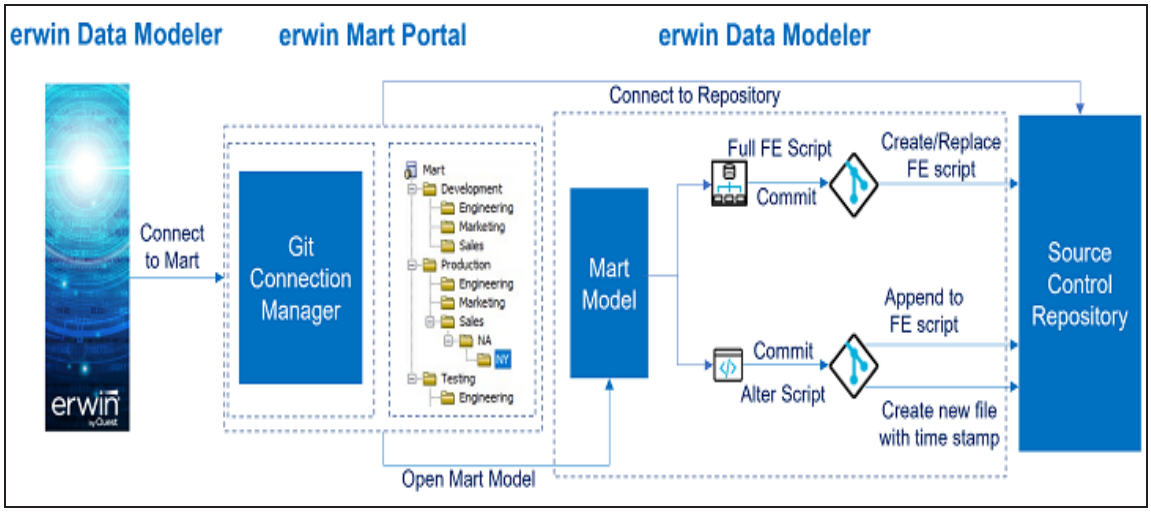
With source control support you can implement:

- DevOps adoption
- collaboration with team members
- version control
- workflow management
- data integrity

Pushing FE scripts to a repository involves:

1. Connecting erwin DM to erwin Mart Portal
2. Connecting erwin DM to a repository
3. Opening a Mart Model and committing FE scripts

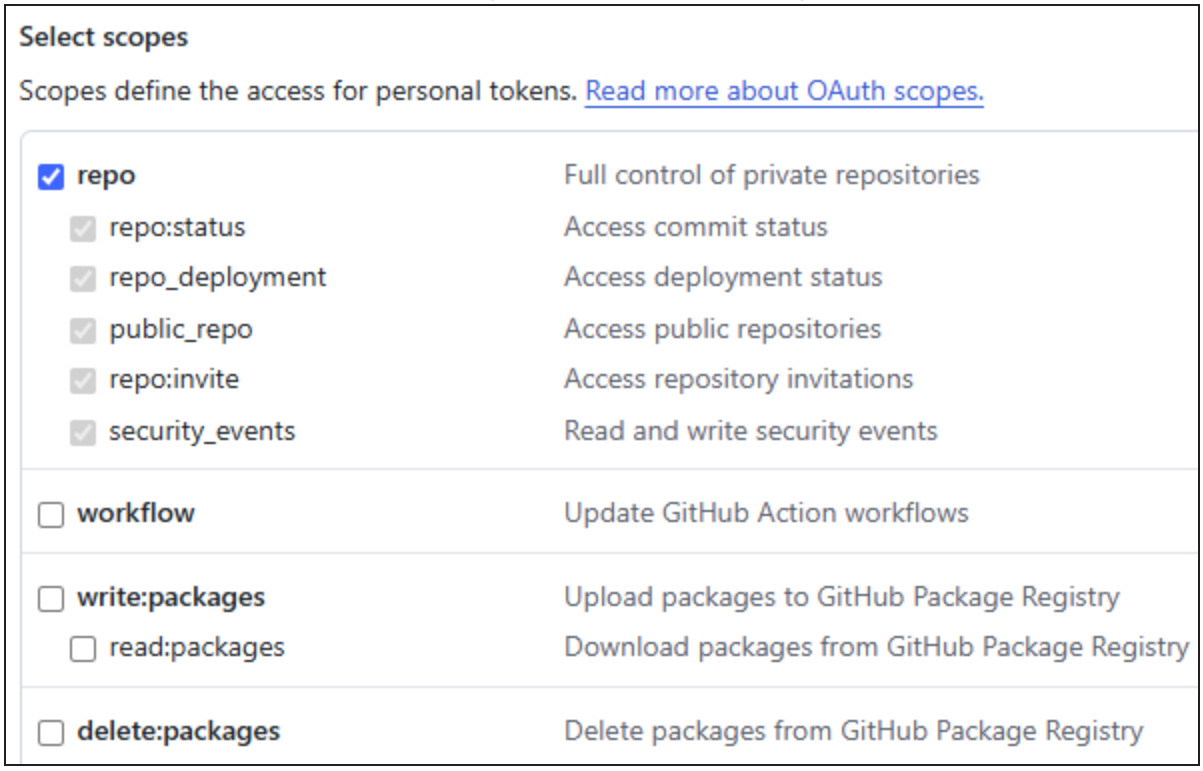
To summarize, following is the workflow to commit FE scripts.



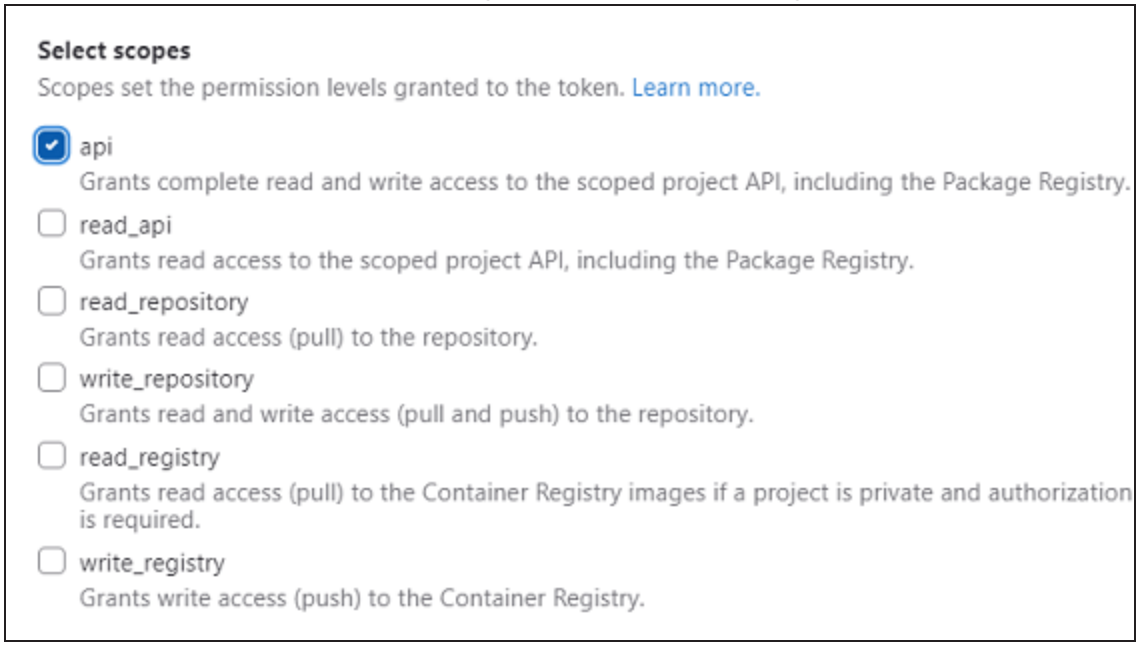
Connecting to Source Control Repositories

A repository may be hosted on GitLab, GitHub, Bitbucket, or Azure DevOps. For a successful connection to these repositories, following are the prerequisites:

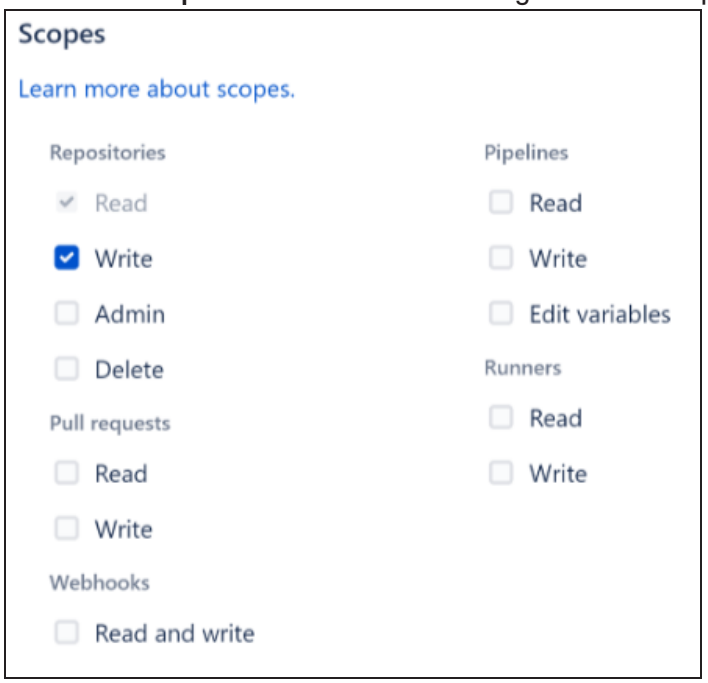
- **GitHub Scope**: Ensure that the following minimum scope is configured.



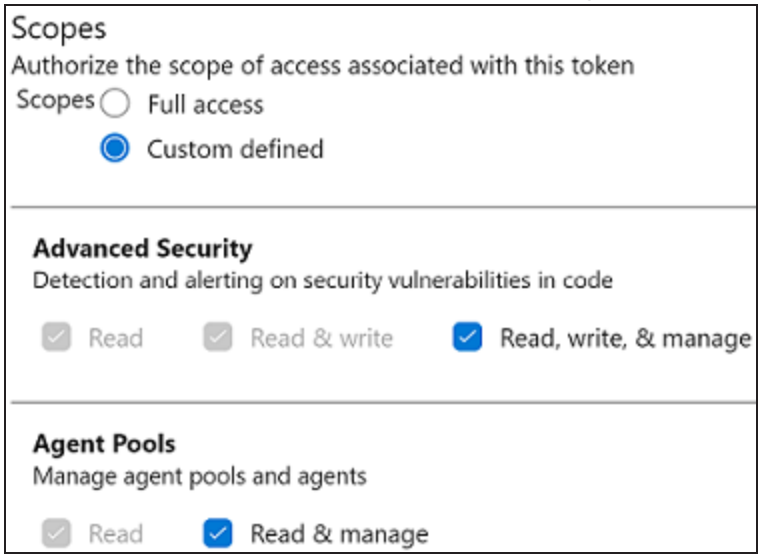
- **GitLab Scope**: Ensure that the following minimum scope is configured.



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

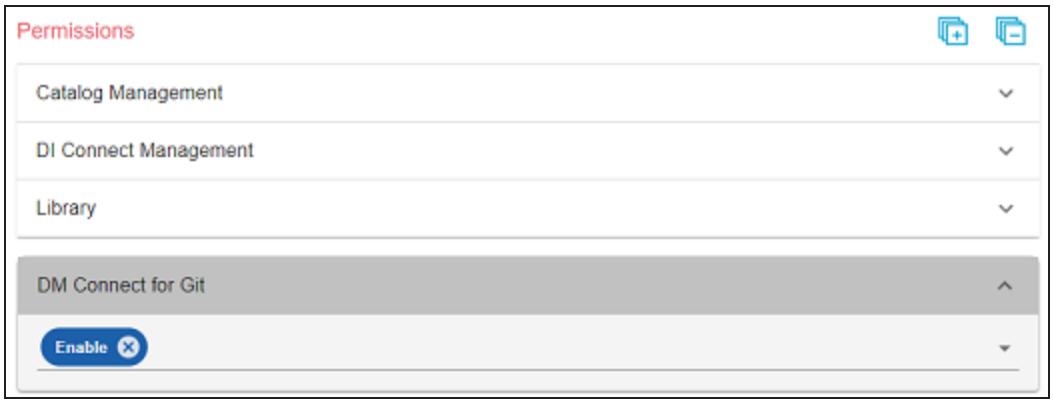


- **Azure DevOps Scope**: Ensure that the following minimum scope is configured.



- **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.

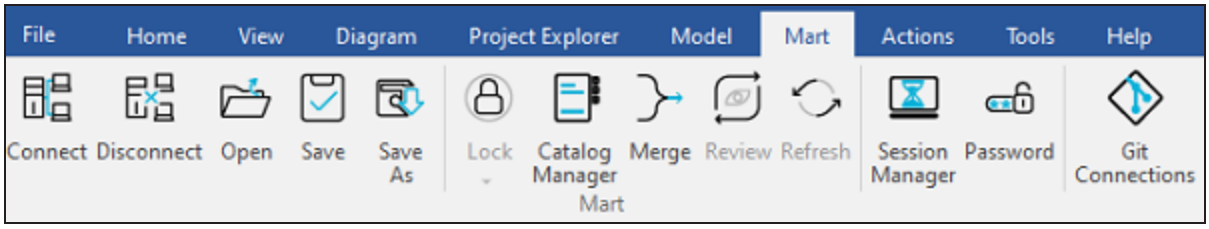


> This permission is not available for Viewer profile.

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

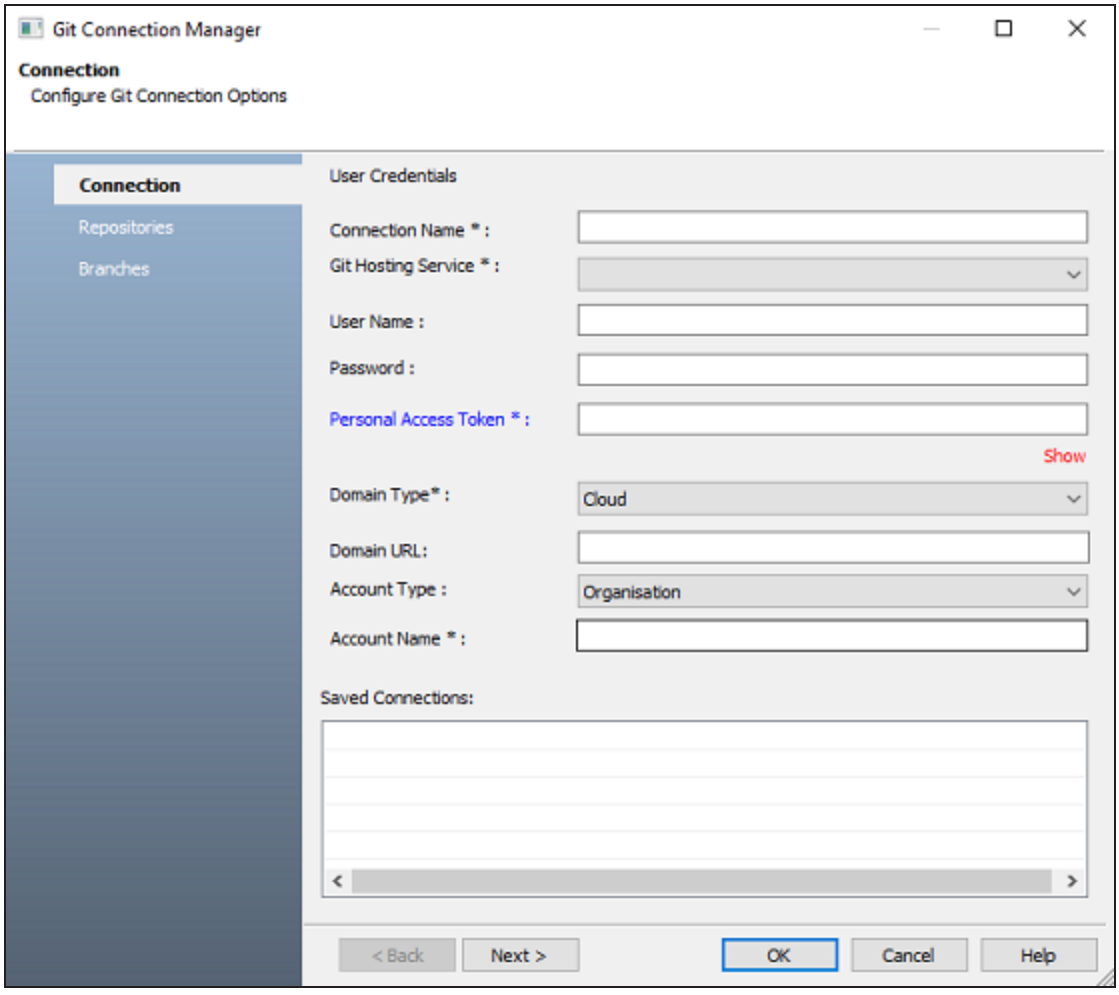
Once, these prerequisites are in place, to connect your repositories to erwin DM, follow these steps:

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. Based on your Git Hosting Service, the available connection parameters differ.

   Refer to the following table for field descriptions.

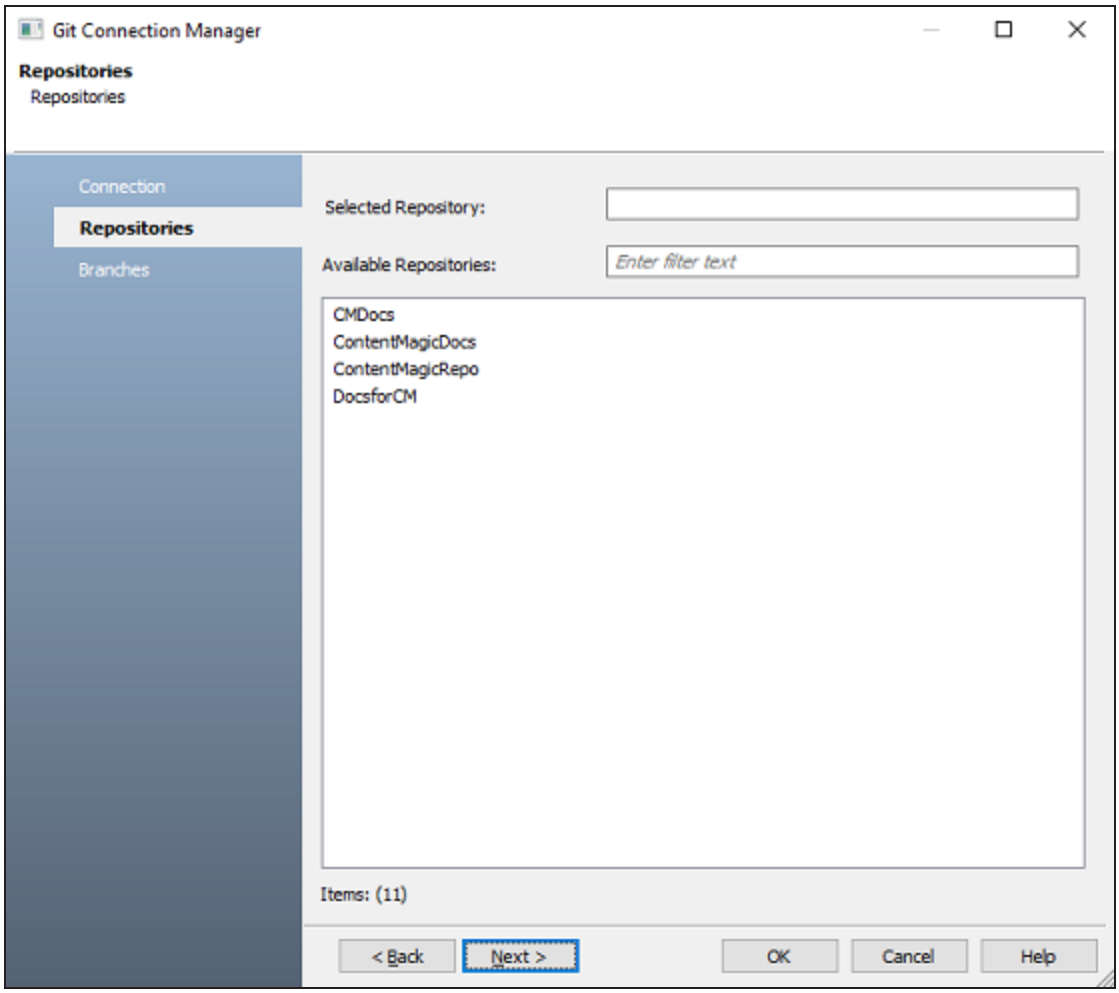
Field Name	Description	Additional Information
Connection Name	Specifies a user-defined connection name	For example, TechPubsConnect. You can create multiple connections one for each repository.
Git Hosting Service	Specifies the source control hosting service to which erwin DMconnects	You can connect to a repository hosted on one of the following services: • GitLab (on-premises/cloud) • GitHub (on-premises/cloud) • Bitbucket (cloud) • Azure DevOps (cloud)
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	
Access Token	Specifies the personal access token to connect to the Bitbucket hosting service	This option is available only when the Git Hosting Service option is set to Bitbucket. You can click the Access Token field label to learn about creating tokens and their types.

Field Name	Description	Additional Information
Domain Type	Specifies whether the connection connects to a cloud account or an on-premise account	
Domain URL	Specifies the URL of your organization's source control account	The value of this field depends on the value set for Domain Type as follows: • **Domain Type is set to On-Premise**: This field is mandatory. Specify your organization's Git hosting service URL till the organization name. Do not include user account or repository name in the URL. For example, https://github.com/quest-TechPubs/, where quest-TechPubs is the organization on github.com. • **Domain Type is set to Cloud**: This field is not mandatory. However, you can specify the common URL of your Git hosting service. For example, https://github.com
Account Type	Specifies whether the organization name or username should be used for your connection	
Account Name	Depending on your selection in Account Type option, specifies the organ-	For example, if https://github.com/quest-TechPubs/ is your GitHub organization and you log in to it using a username, TechWriter: • If Account Type is Organisation,

Field Name	Description	Additional Information
	ization name or user-name	Account Name is quest-TechPubs • If Account Type is User, Account Name is TechWriter

4. Click **Next**.

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

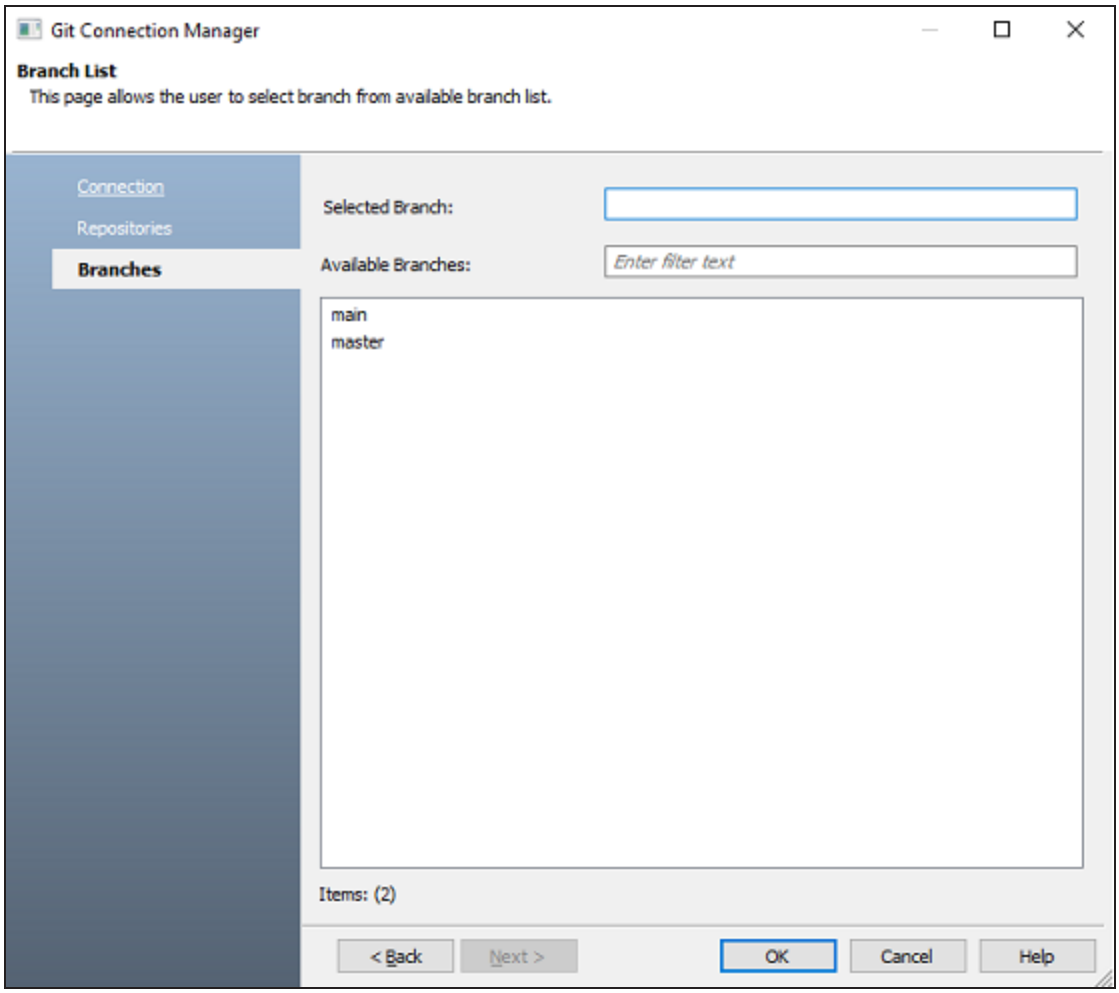


5. Select the repository where you want to push forward engineering scripts.

You can also use the Available Repositories field to filter the list of repositories by name.

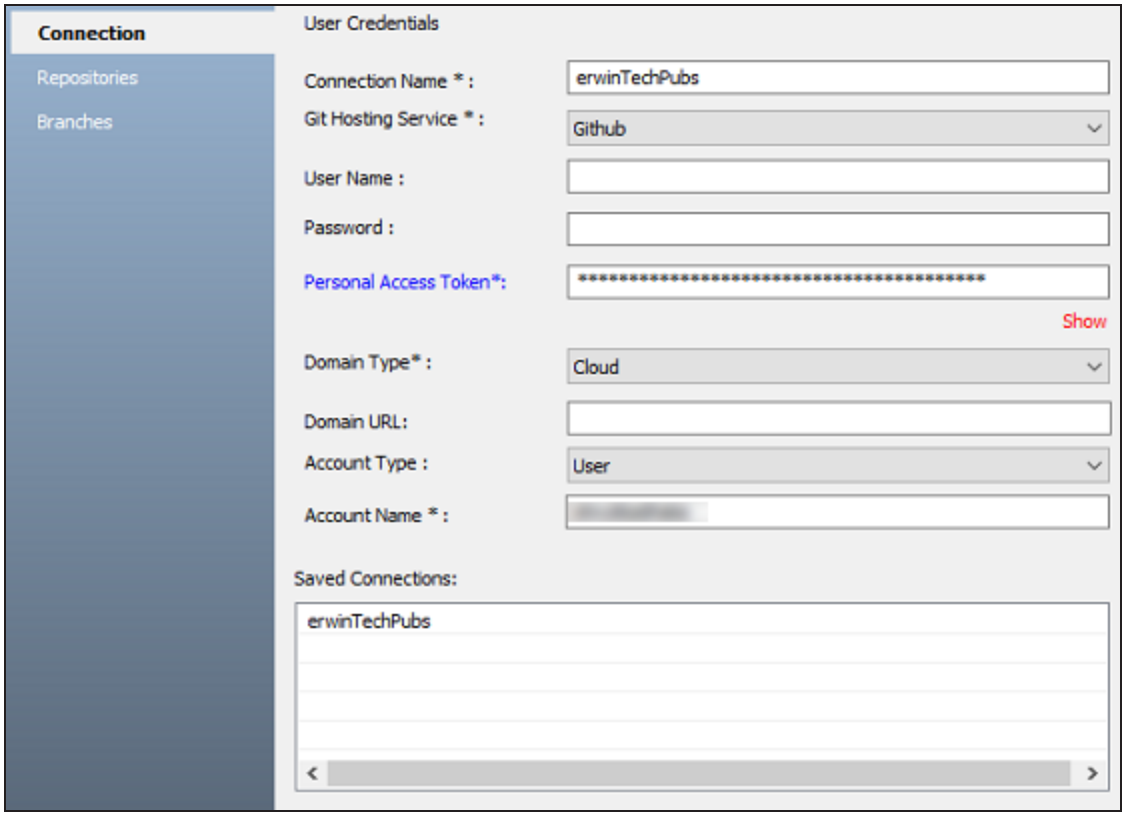
6. Click **Next**.

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



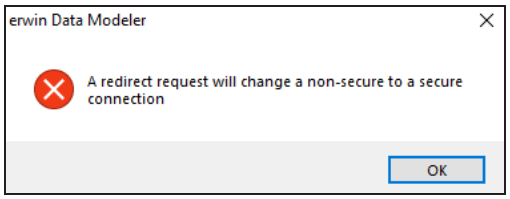
7. Select the branch where you want to push forward engineering scripts and then click **OK**.

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



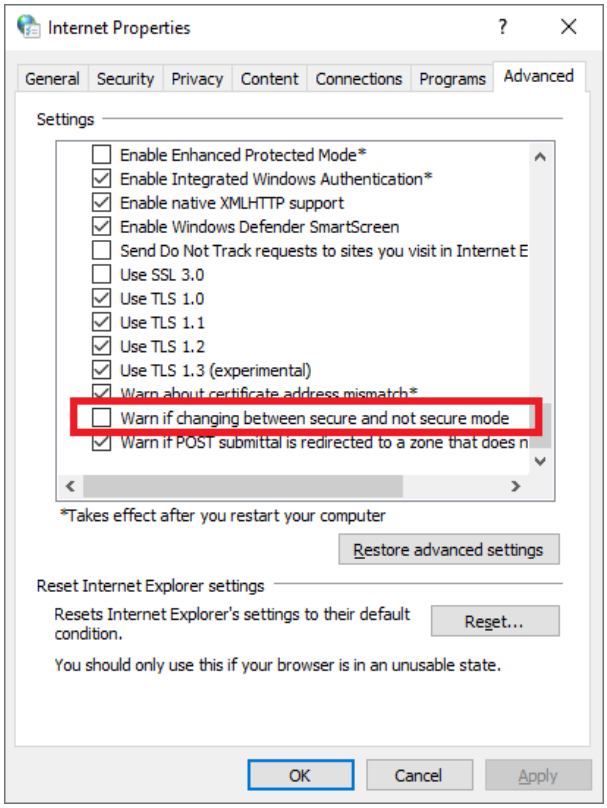
Troubleshooting

While setting up your connection, you may encounter the following error:



To resolve this error, follow these steps:

1. On your system, go to **Control Panel** > **Internet Options** > **Advanced** Tab.
2. Under the Security category, clear the **Warn if changing between secure and not secure mode** check box.



3. Click **OK**.
4. Close and reopen erwin DM.
5. Connect erwin DM to erwin Mart Portal.
6. Open the Git Connection Manager and configure your source control connection.

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

Committing Forward Engineering Scripts

There are two scenarios in which you commit Forward Engineering (FE) scripts to a Git repository:

- **Scenario 1: Committing new or full FE scripts**:

  Use the Forward Engineer Schema Generation Wizard to commit a physical database schema or FE script from a Mart Model.

  > To avoid script files from being overwritten, ensure that you use unique file names.

- **Scenario 2: Committing alter scripts**:

  Use the Forward Engineer Alter Script Schema Generation Wizard to commit an alter script after you make changes to a Mart Model. You can commit an alter script in two ways:

  - **Commit and append an alter script to an existing script file**
  - **Commit and create a new alter script file in the Git repository**

  For more information, refer to the Scenario 2: Committing Alter Scripts topic.

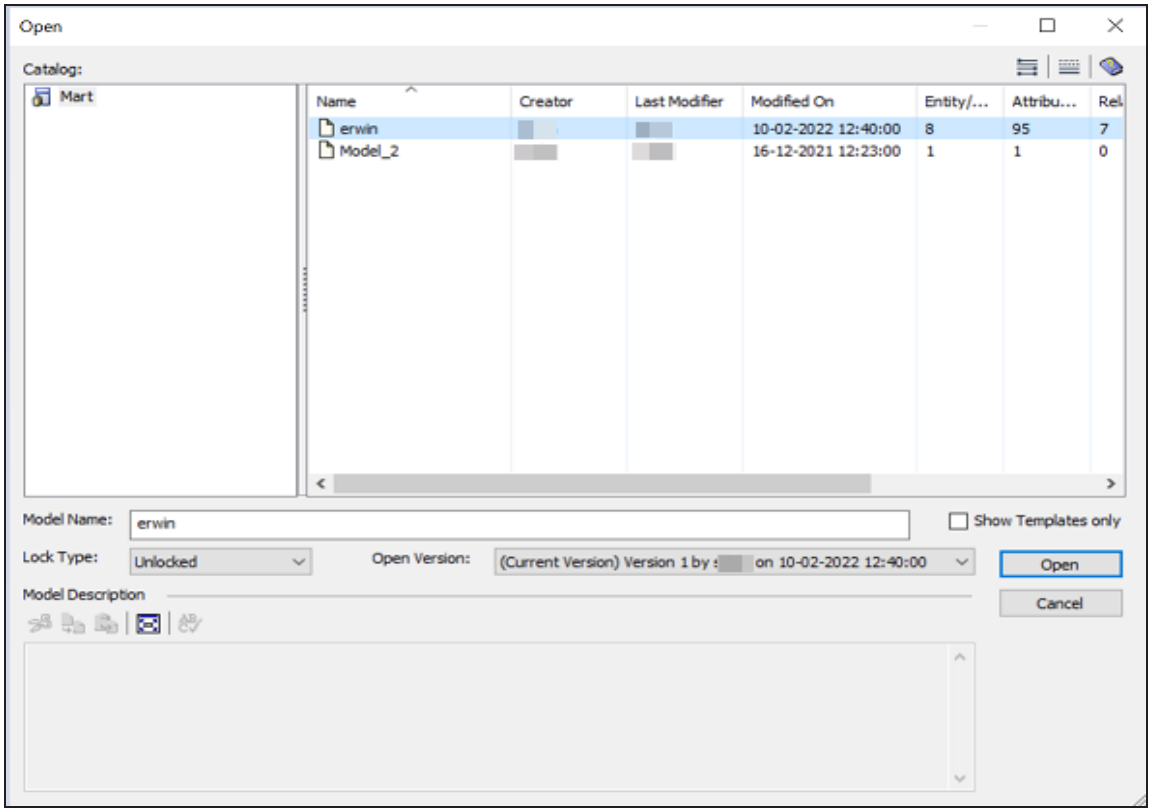
Scenario 1: Committing New or Full FE Scripts

The Forward Engineer Schema Generation Wizard generates a physical database schema or Forward Engineering (FE) script. For a Mart Model, you can push the FE script to a Git repository.

To commit new or full FE scripts to Git repositories, follow these steps:

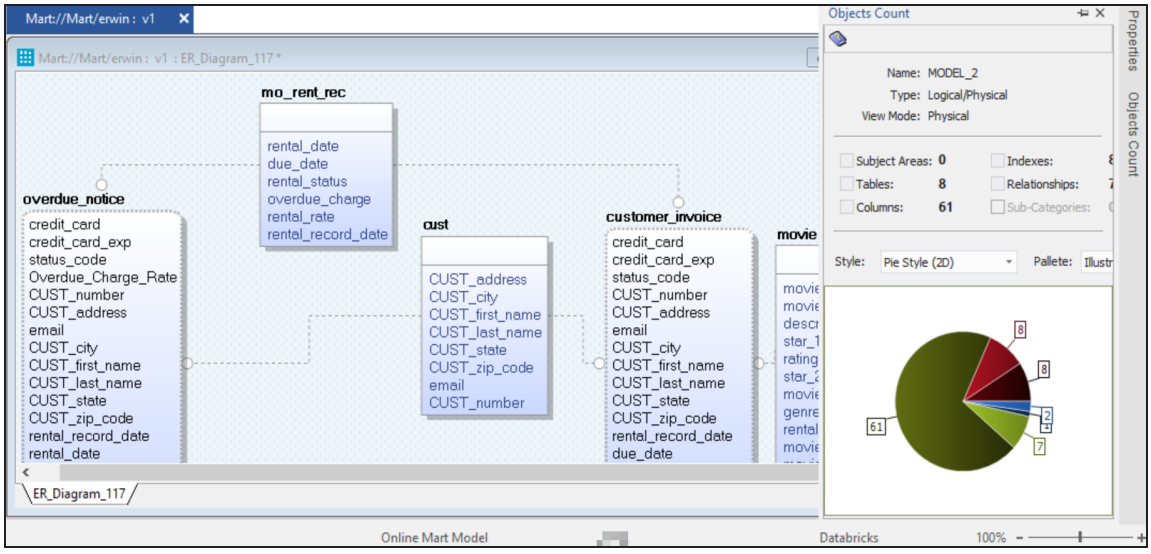
1. On the ribbon, go to **Mart** > **Open**.

   The Open page appears.



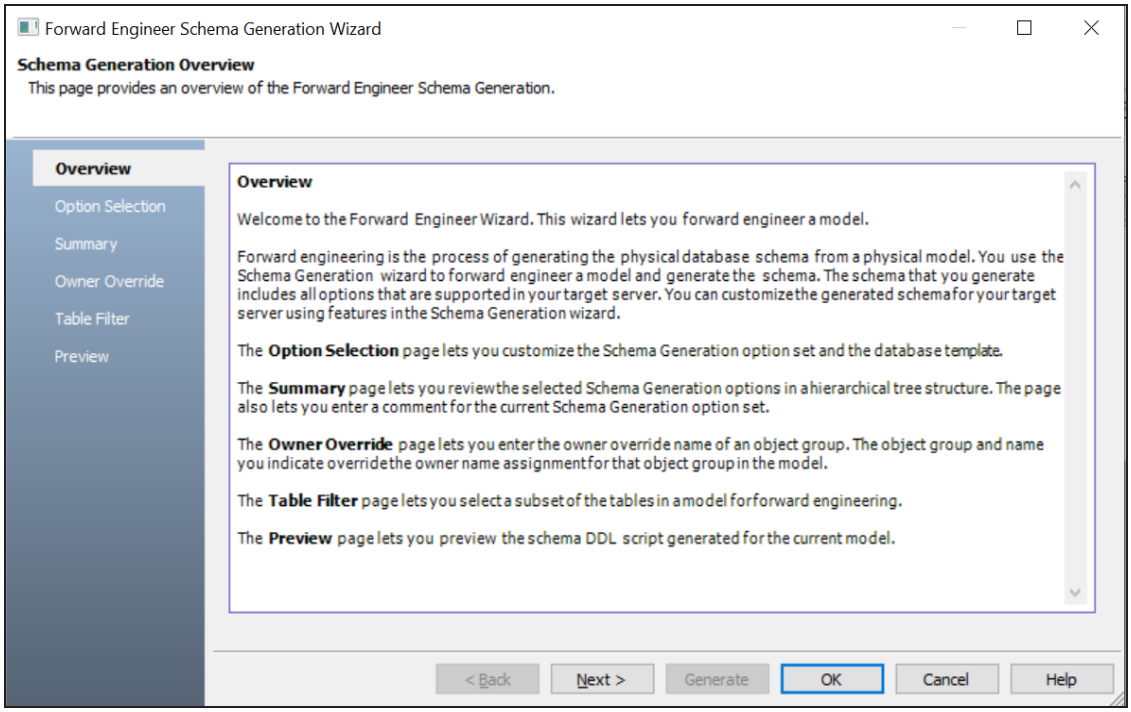
2. Select a model, and then click **Open**.

   The Mart Model opens.



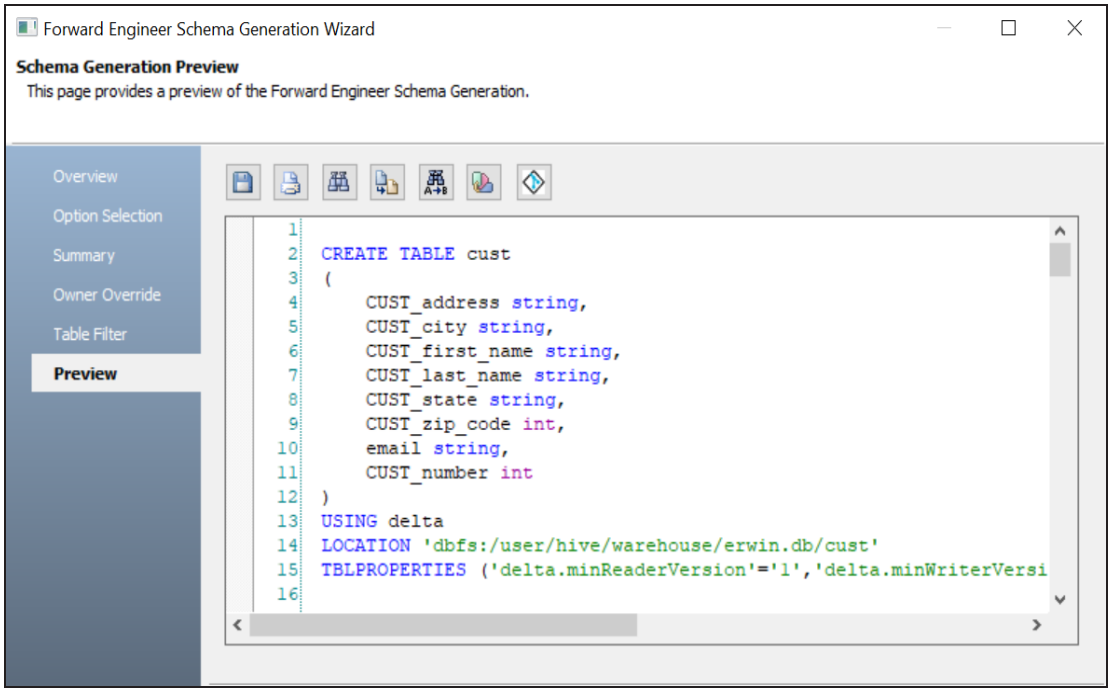
3. Go to **Actions** > **Schema**.

   The Forward Engineer Schema Generation Wizard appears.



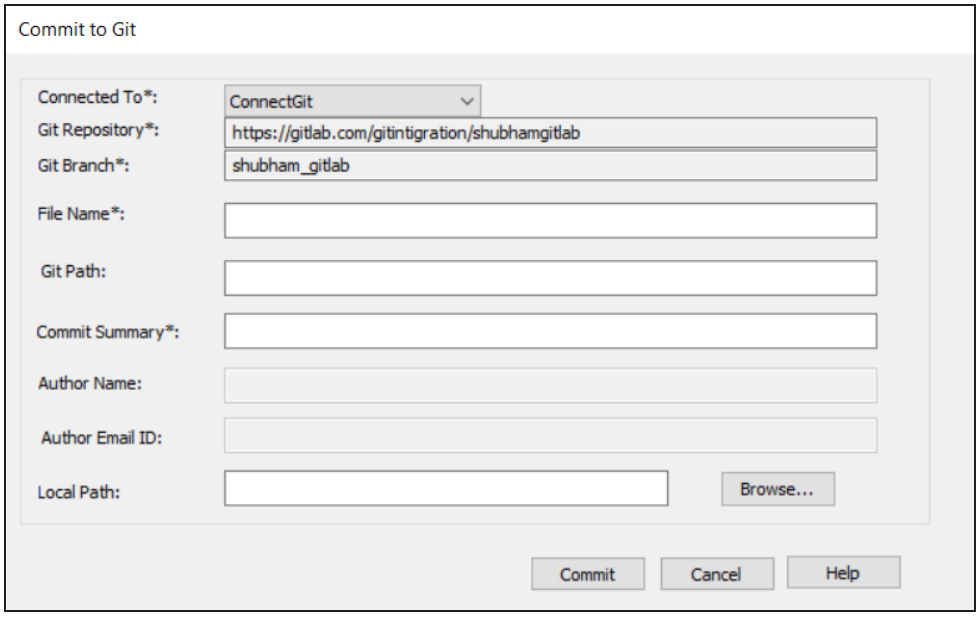
4. On the **Forward Engineer Schema Generation Wizard**, click the **Preview** section.

   The FE script appears. For example, in the following image the Preview section displays FE script of a Databricks database. For more information on generating FE scripts, refer to the Forward Engineering/Schema Generation for Databases topic.



5. Click  .

   The Commit to Git screen appears.



6. Enter appropriate values in the fields. Fields marked with an asterisk (*) are mandatory. Refer to the following table for field descriptions.

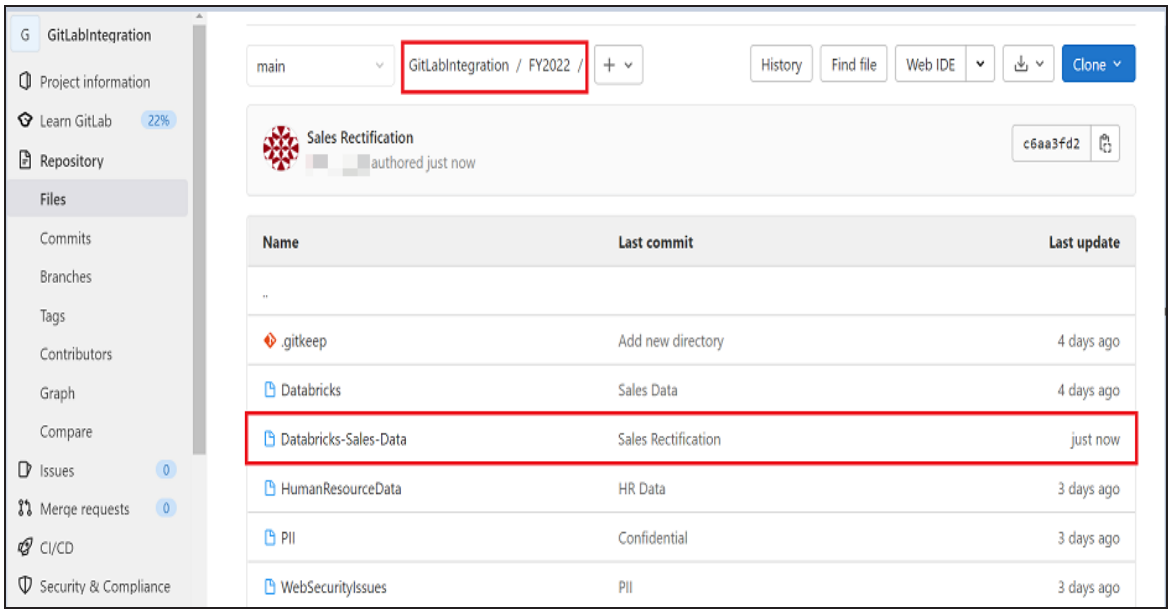
Field Name	Description	Additional Information
Connected To	Specifies the connection that connects erwin DM to a Git repository	For example, ConnectGit.
Git Repository	Specifies the Git repository configured for the connection	For example, https://-gitlab.com/d4215/GitLabIntegration is set for the ConnectGit connection. This field autopopulates based on the repository configured in the Git Connection Manager.
Git Branch	Specifies the Git branch configured for the connection	For example, main is set for the ConnectGit connection. This field autopopulates based on the repository configured in the Git Connection Manager.
File Name	Specifies the user-defined name of the FE script file being committed to a Git repository	For example, Databricks-Sales-Data.sql To avoid script files from being overwritten, ensure that you use unique file names.
Git Path	Specifies the location in the Git repository where the FE script is committed	For example, FY2022/ The FE script is committed to the FY2022 folder inside the root folder of your Git repository.
Commit Summary	Specifies the summary of the push commit	For example, Sales Rectification.
Author Name	Specifies the name of the author pushes the commit	
Author Email ID	Specifies the email address of the author pushes the commit	

Field Name	Description	Additional Information
Local Path	Specifies the location on your local machine where the FE script is saved	C:\Users\SO\Documents\Databricks

7. Click **Commit**.

   The FE script file is saved on the local path and committed to the Git repository.

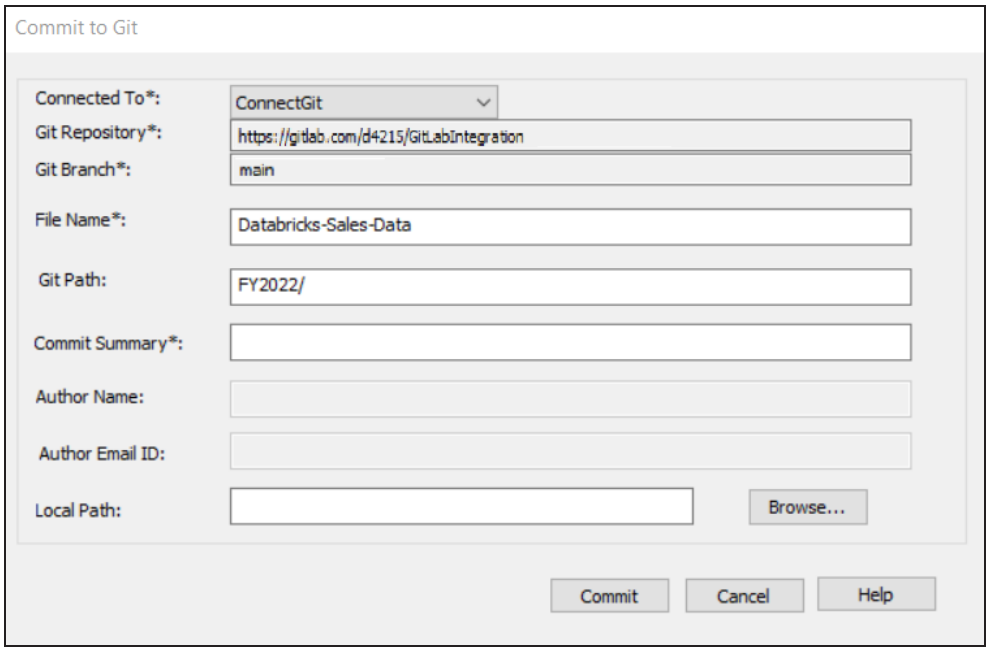
   For example, in the following image, FE script is committed to a GitLab repository in a file, Databricks-Sales-Data, with a commit summary, Sales Rectification using the main branch.



   You can click the file to review its content. For example, in the following image, Databricks-Sales-Data's content is visible.

You can use FE Schema Generation Wizard to commit FE script using the same connection again. The Commit to Git screen autopopulates the previously set values in File Name and Git Path.

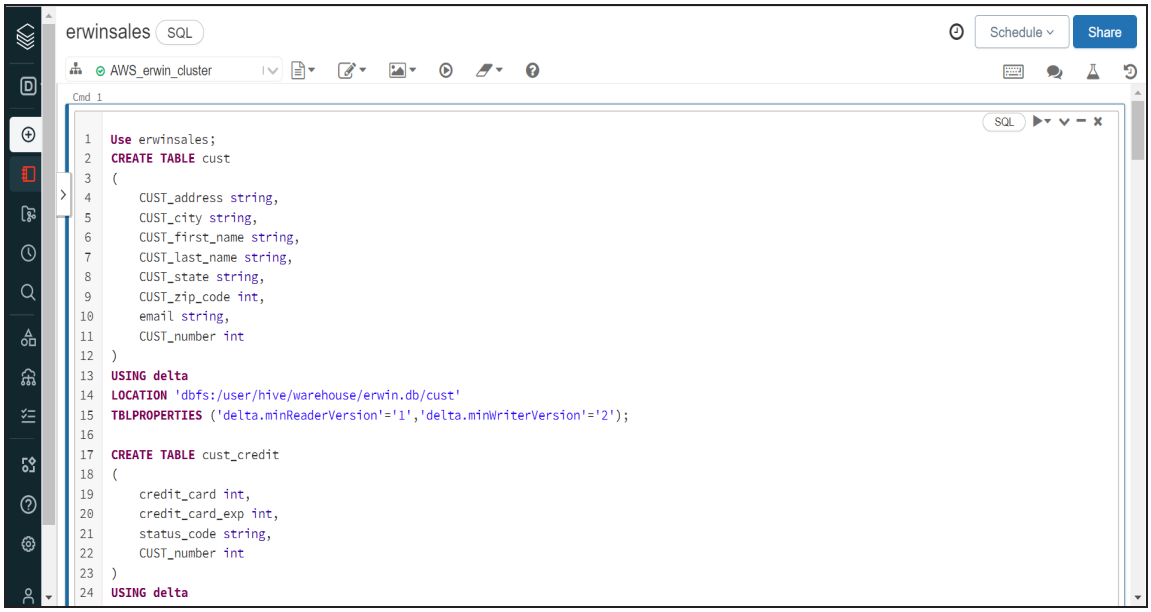
For example, in the following image File Name is set to Databricks-Sales-Data and Git Path is set to FY2022/.



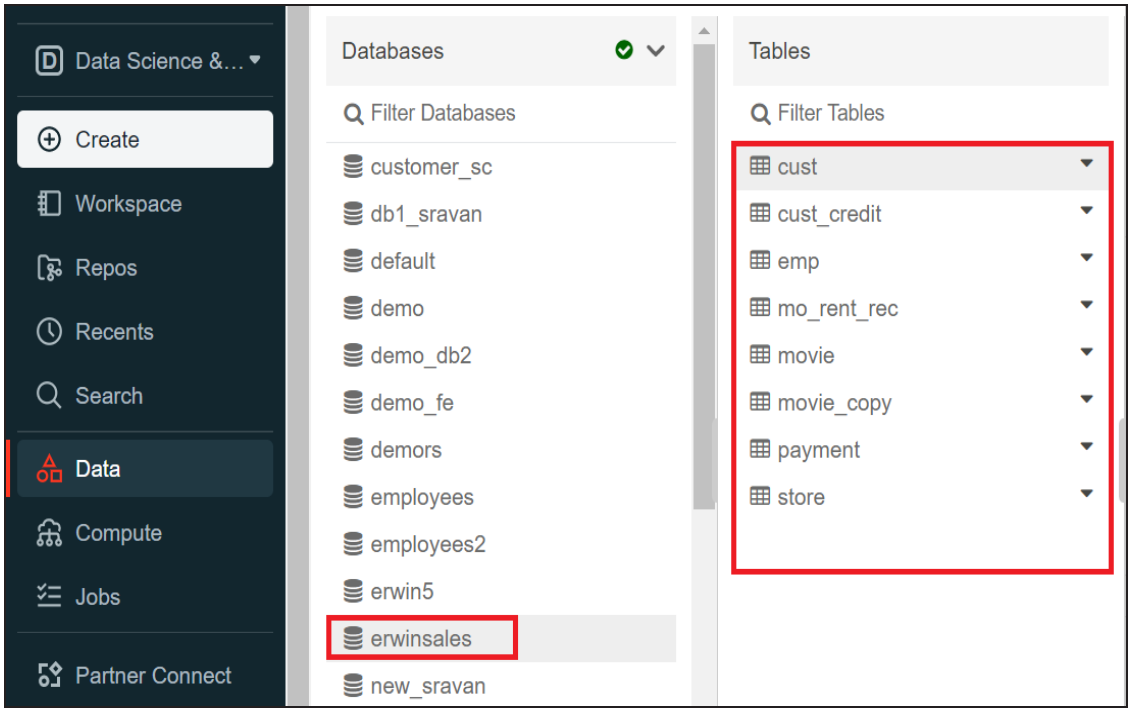
Committing the FE script again with the same File Name and Git Path overwrites the previous file in the Git repository.

Once the FE script is committed, you can run it on your database to generate and verify the physical schema.

For example, in the following Databricks database, the FE script copied from the Git repository is run.



After running the FE script, the required database objects are created. You can access these objects from the database. For example, the following tables can be accessed in a Databricks database.



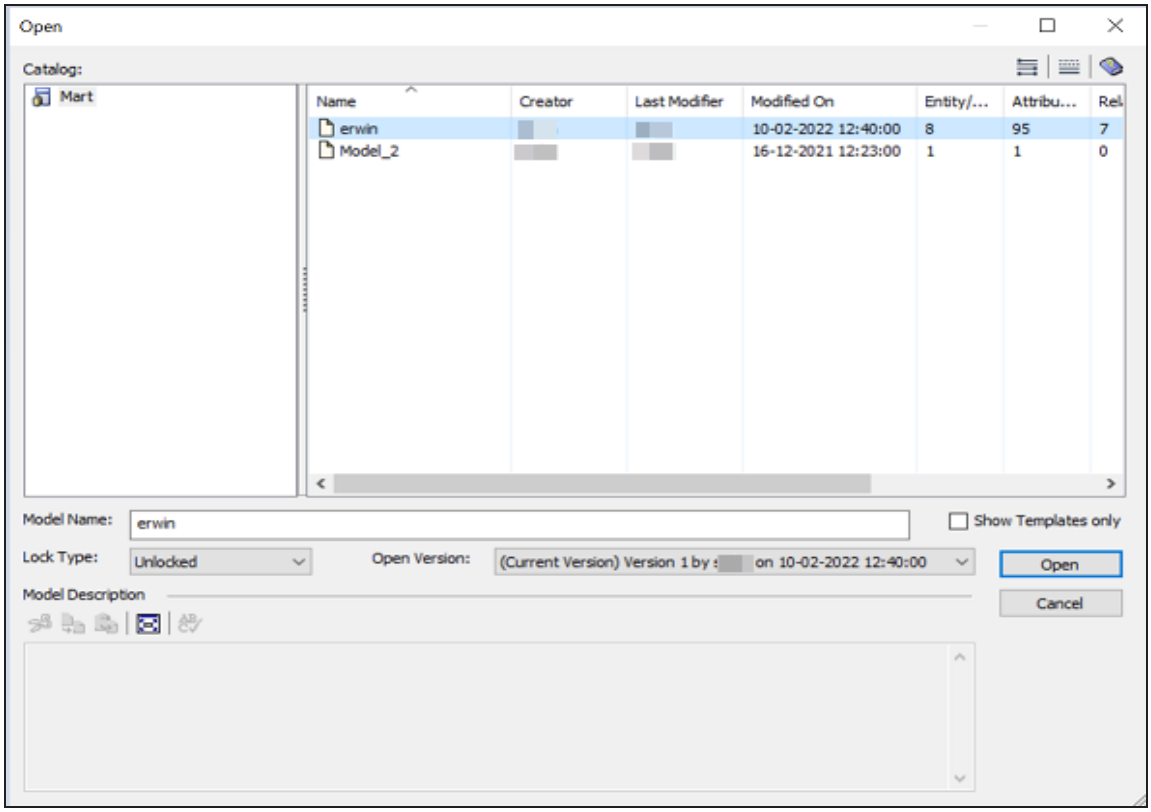
Scenario 2: Committing Alter Scripts

The Forward Engineer Alter Schema Generation Wizard generates an alter script for a database after you make changes to a model. For a Mart Model, you can push the alter script to a Git repository.

To commit alter scripts to Git repositories, follow these steps:

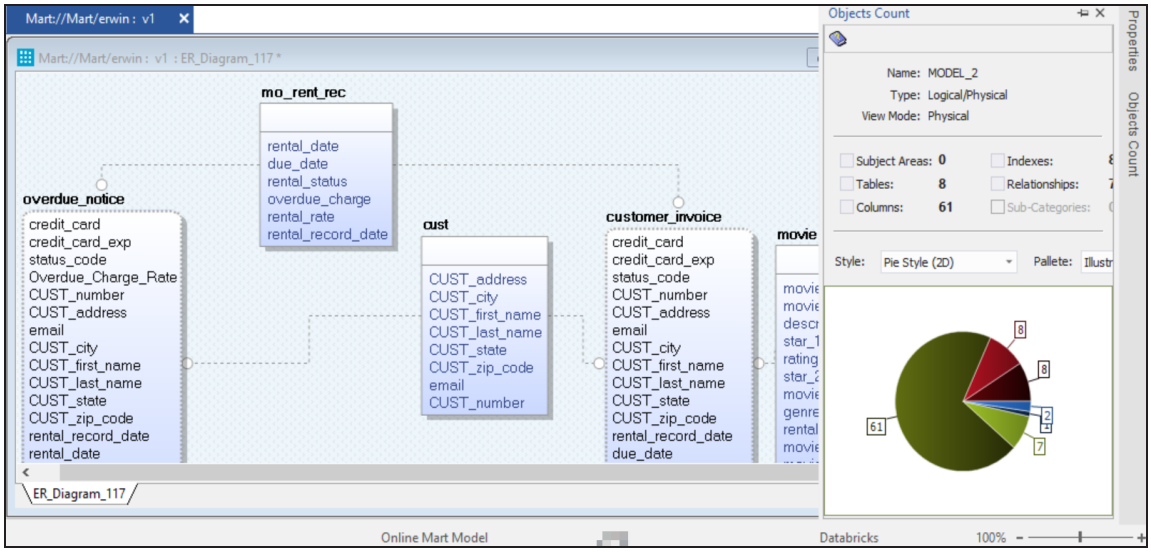
1. On the ribbon, go to **Mart** > **Open**

   The Open page appears.



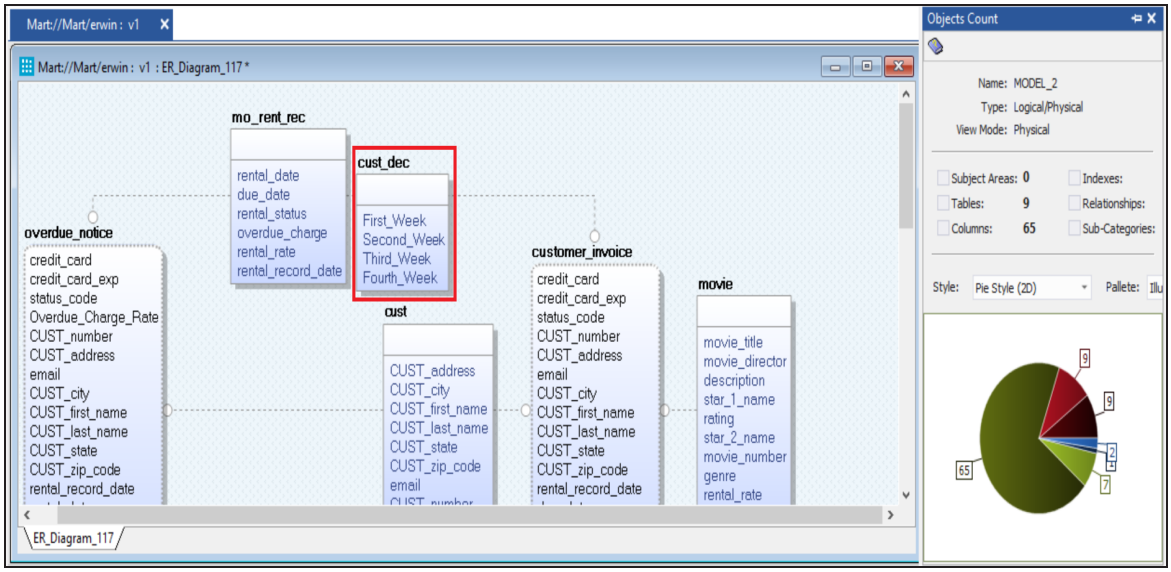
2. Select a model, and then click **Open**.

   The Mart Model opens.



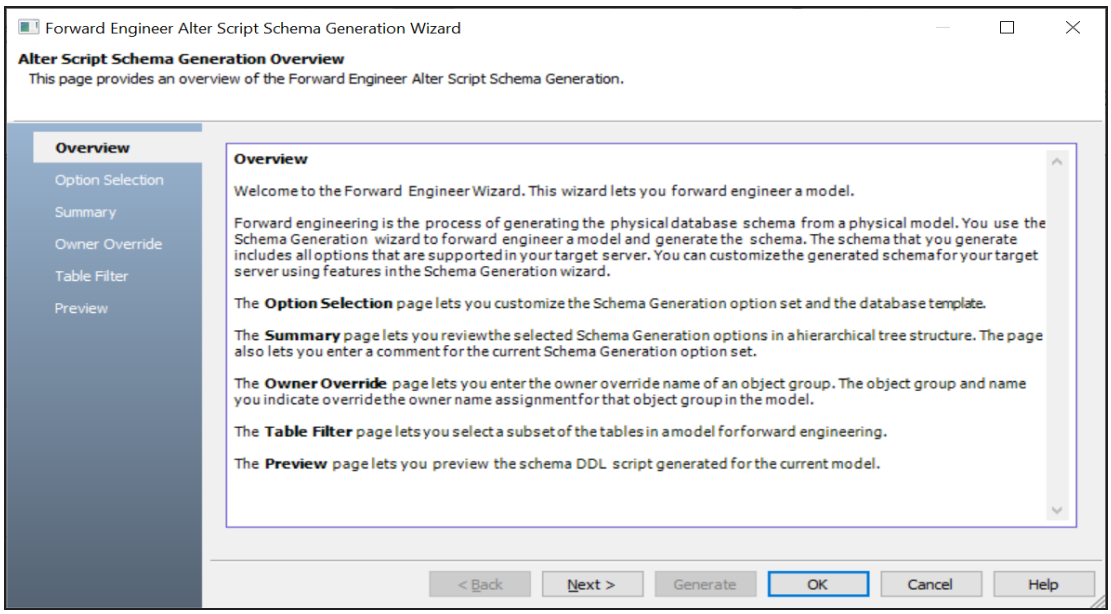
3. Make the required changes in the model.

   For example, in the following model, a new table, cust_dec with four columns is added.



4. Go to **Actions** > **Alter Script**.

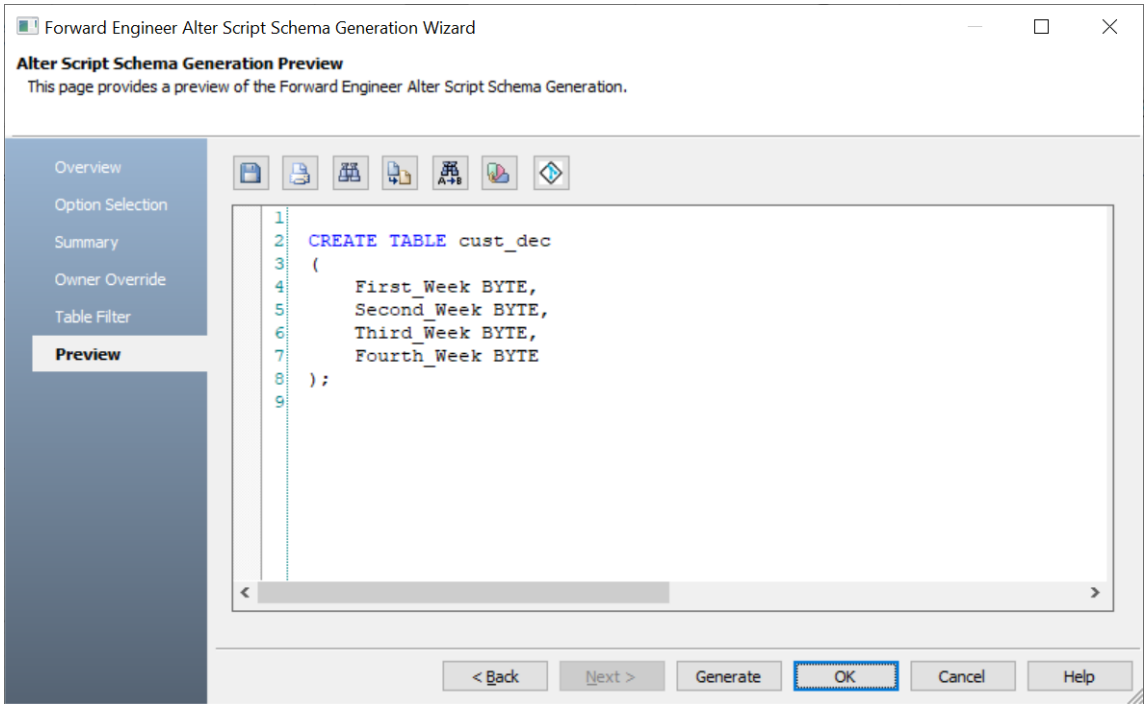
   The Forward Engineer Alter Script Schema Generation Wizard appears.



5. On the **Forward Engineer Alter Schema Generation Wizard**, click the **Preview** section.

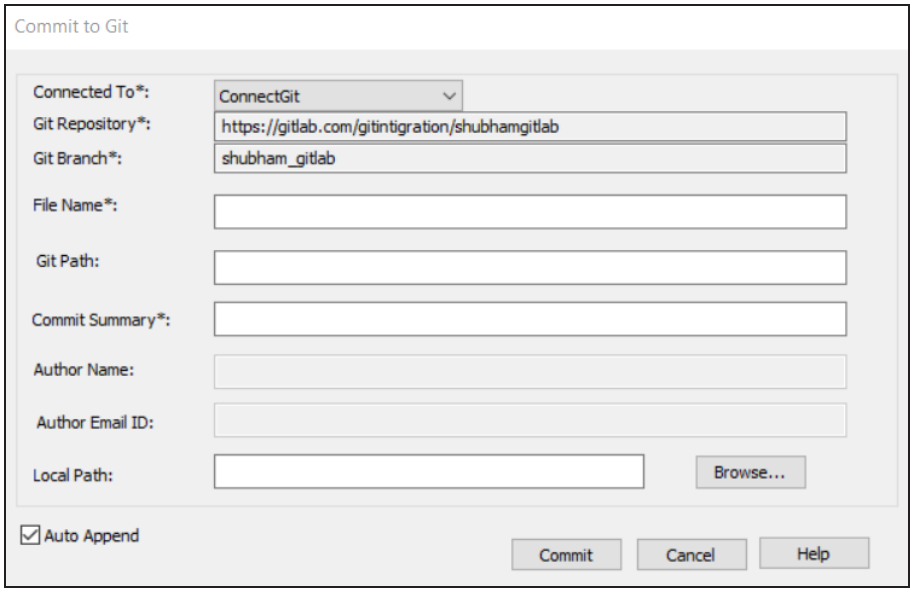
The alter script appears. For more information on generating alter scripts, refer to the Generating Alter Script for Databases topic.

For example, in the following image the Preview section displays an alter script of a Databricks database.



6. Click  .

The Commit to Git screen appears. The File Name and Git Path values autopopulates with the values configured in the previous commit. You can update the File Name and Git Path as per the requirement.



7. Enter appropriate values in the fields. Fields marked with an asterisk (*) are mandatory. Refer to the following table for field descriptions.

Field Name	Description	Additional Information
Connected To	Specifies the connection that connects erwin DM to a Git repository	For example, ConnectGit.
Git Repository	Specifies the Git repository configured for Connection	For example, https://-gitlab.com/d4215/GitLabIntegration is set for the ConnectGit connection. This field autopopulates based on the repository configured in the Git Connection Manager.
Git Branch	Specifies the Git branch that was set for connection in the Git Connection Manager	For example, main is set for the ConnectGit connection. This field autopopulates based on the repository configured in the Git Connection Manager.

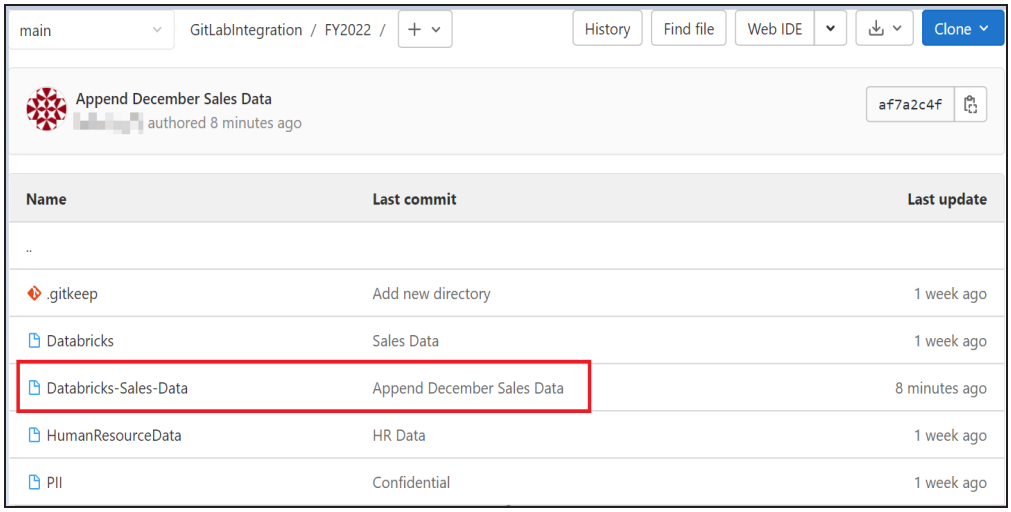
Field Name	Description	Additional Information
File Name	Specifies the user-defined name of the FE script file being committed to a Git repository	For example, Databricks-Sales-Data.sql
Git Path	Specifies the location in the Git repository where the FE script is committed	For example, FY2022/ The FE script is committed to the FY2022 folder inside the root folder of your Git repository.
Commit Summary	Specifies the summary of the push commit	For example, Append December Sales.
Author Name	Specifies the name of the author pushes the commit	
Author Email ID	Specifies the email address of the author pushes the commit	
Local Path	Specifies the location on your local machine where the Alter script is saved	C:\Users\SO\Documents\Databricks
Auto Append	Specifies whether the alter script is appended to the file set in File Name and Git	By default, the Auto Append check box is selected. To create a new script file, clear the Auto Append check box and set the File Name and File Path belonging to an existing file. A new file with the following naming convention: <File Name>_YYYY-MM-DD_HH-MM-SS is created.

Field Name	Description	Additional Information
	Path	Ensure that you use this check box consistently every time you commit an alter script.

8. Click **Commit**.

   The alter script file is saved on the local path and committed to the Git repository.

For example, in the following image, an alter script file is committed to a GitLab repository and appended to an existing file, Databricks-Sales-Data, with a commit summary, Append December Sales using the main branch.

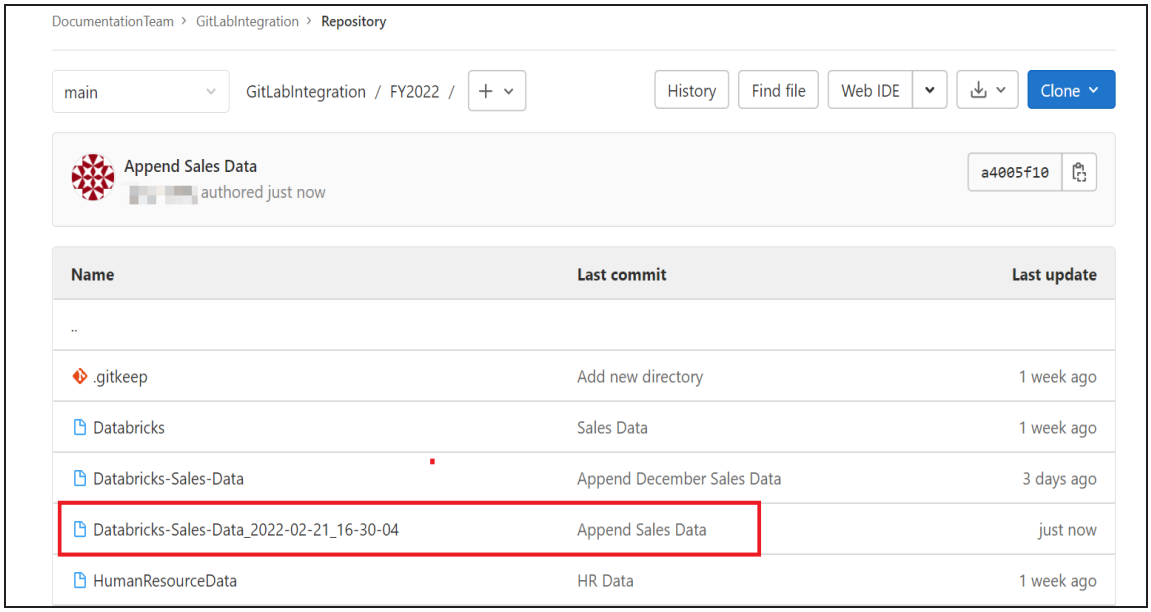


You can click the file to review its content. For example, in the following image, Databricks-Sales-Data contains the alter script.

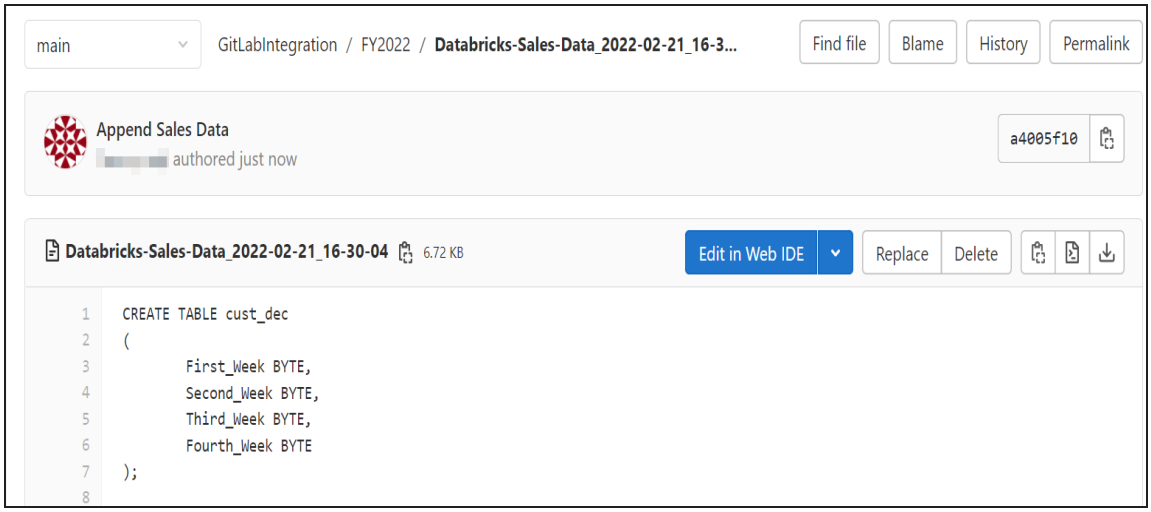
```
        status_code string,
        CUST_number int
)
USING delta
LOCATION 'dbfs:/user/hive/warehouse/erwin.db/cust_credit'
TBLPROPERTIES ('delta.minReaderVersion'='1','delta.minWriterVersion'='2');

CREATE TABLE cust_dec
(
        First_Week BYTE,
        Second_Week BYTE,
        Third_Week BYTE,
        Fourth_Week BYTE
);

CREATE TABLE emp
(
        EMP_first_name string,
        EMP_address string,
        EMP_phone int,
        EMP_address_2 string,
        email string,
        salary int,
        hire_date timestamp,
        soc_sec_number int,
        EMP_number string
```

Clearing the Auto Append check box and setting the File Name and File Path belonging to an existing file creates a new file with the following naming convention: <File Name>_YYYY-MM-DD_HH-MM-SS.

For example, in the following image, a file is created with a time stamp in a Git repository.

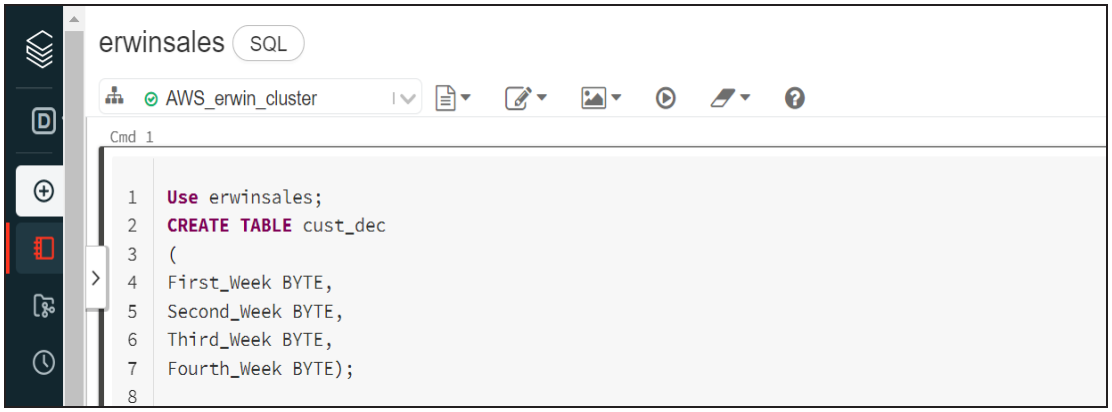


This file contains only the alter script.



Use the committed FE script to generate a physical schema in your database. To generate schema, copy the FE script from your Git repository and run the script in the database.

For example, in the following Databricks database, the FE script copied from the Git repository is run.



The cust_dec table is created in a Databricks database.