**Migration of Data from MS Sql server to Postgres**

In the process of migrating the entire database from “SourceFuse\_DB” database in MS Sql server “Sourcefuse\_DB” database in Postgres, there are 2 major process involved as follows.

1. **Migration of the Database Schema**
2. **Migration of the data from Source to Target database**

For the above process we can achieve by using multiple tools like **AWS-SCT, Ispirer, QuerySurge, DBConvert, SSIS, SQL Server Import and Export Wizard, sqlserver2pgsql** etc. In this exercise I have used **AWS-SCT for the schema conversion process** and also used **Import and Export Wizard in Sql server for Data migration process**. (I have not used DMS, due to the permission issue).

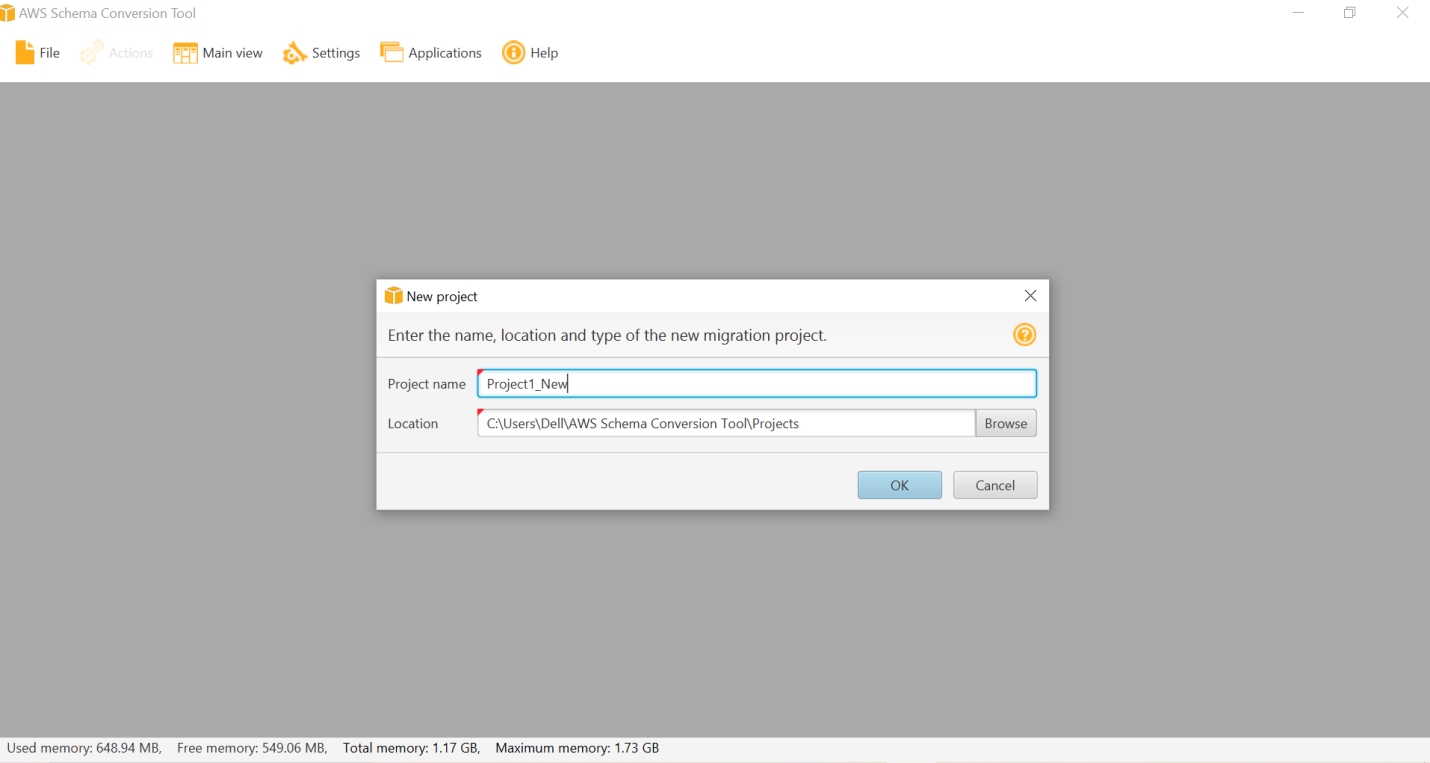
**Process1 : Migration of the Schema from Sql server to Postgres using AWS-SCT**

**Step1**: Sql server database Configuration

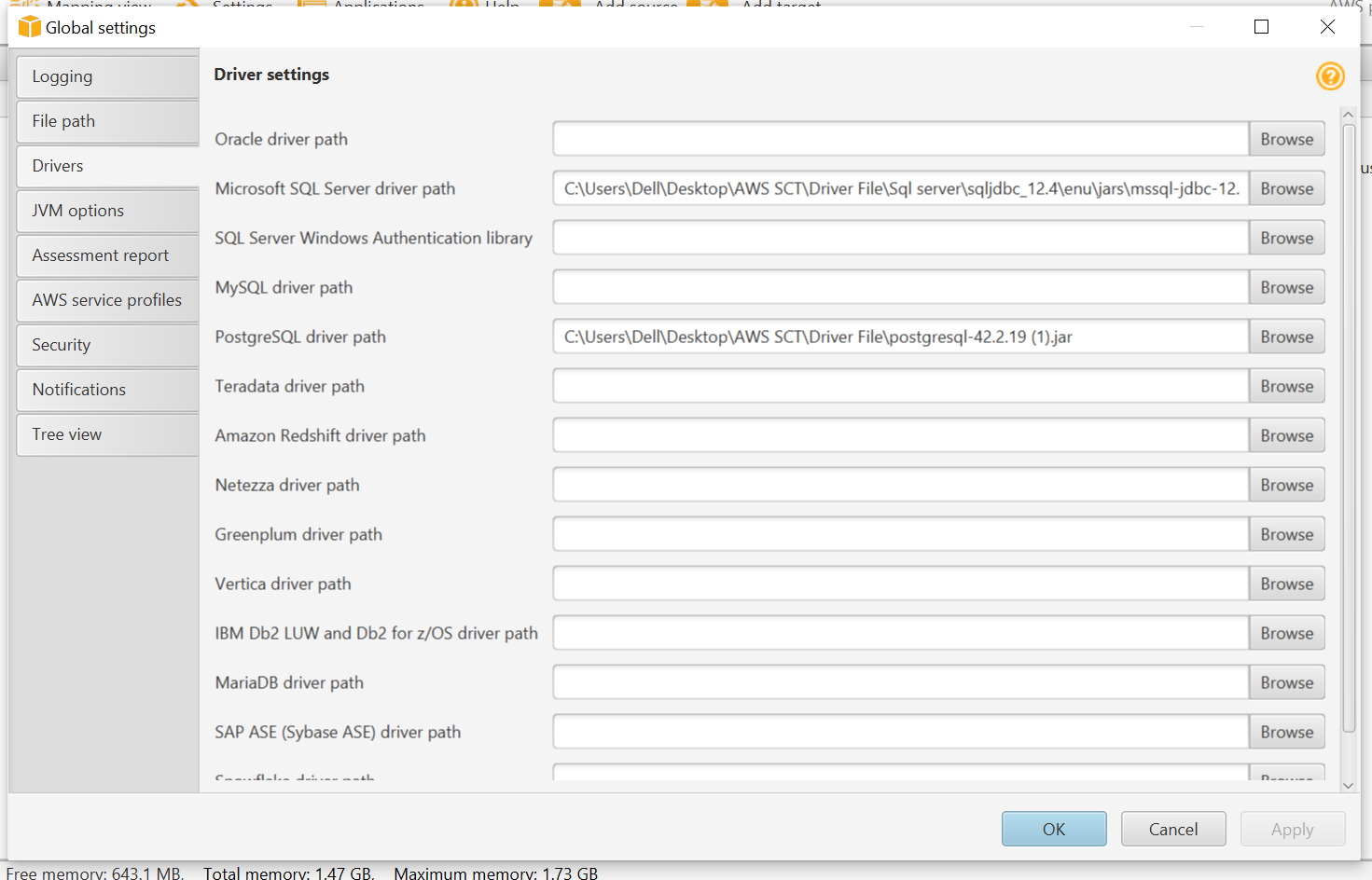
Steps involved in the Sql server database Configuration:

1. Downloaded the sql server script file from given Github URL.
2. Open SSMS with Sql server Authentication and created a new DB called SourceFuse\_DB.
3. Run the downloaded DB script which contains the creation of the DB objects script along with constraints.
4. Validated the objects creation after the successful run of the script.

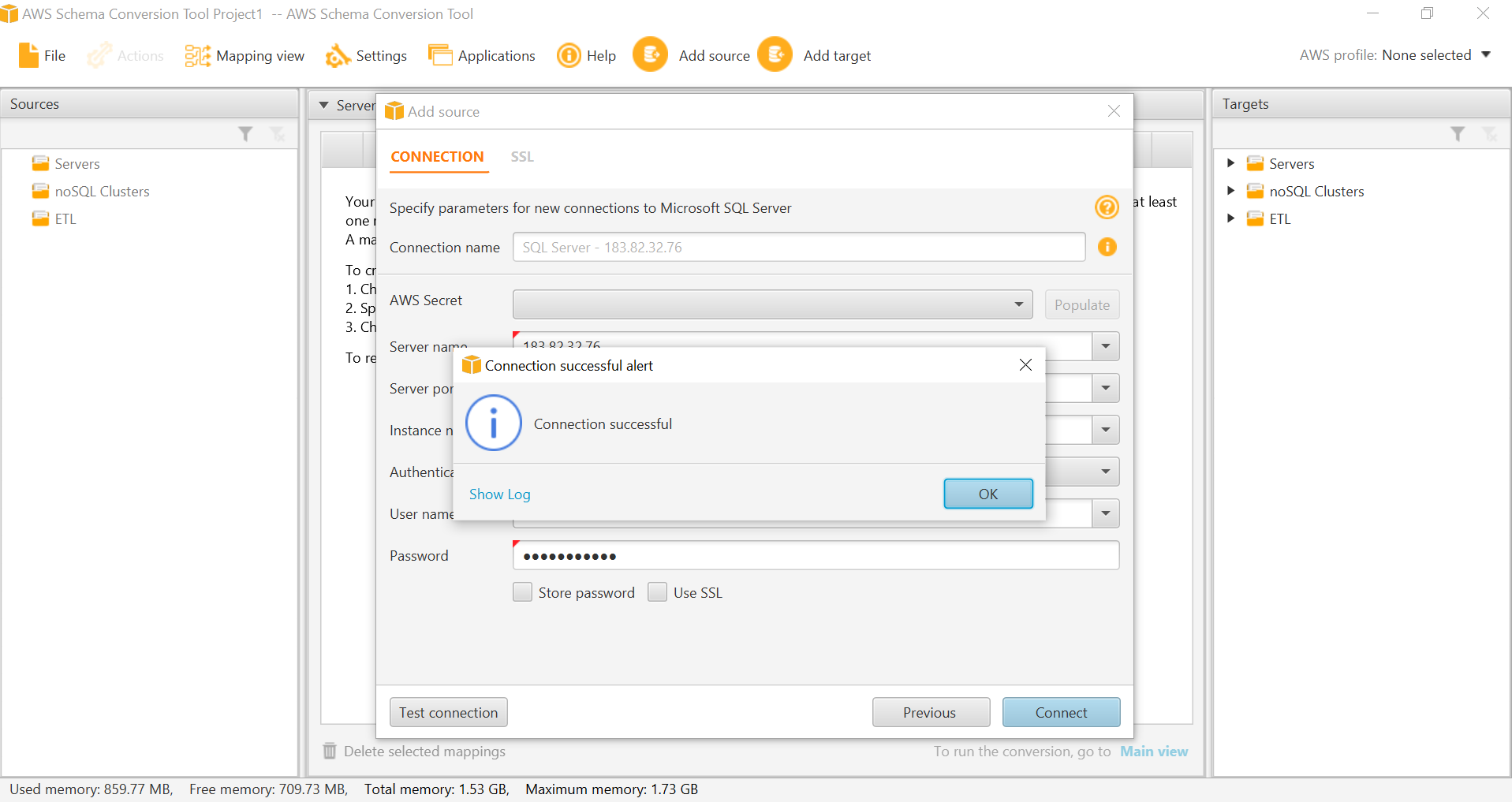
**Step2**:Create a new Schema conversion project called “Project\_New1” by selecting the “Create New Project” option in File menu.



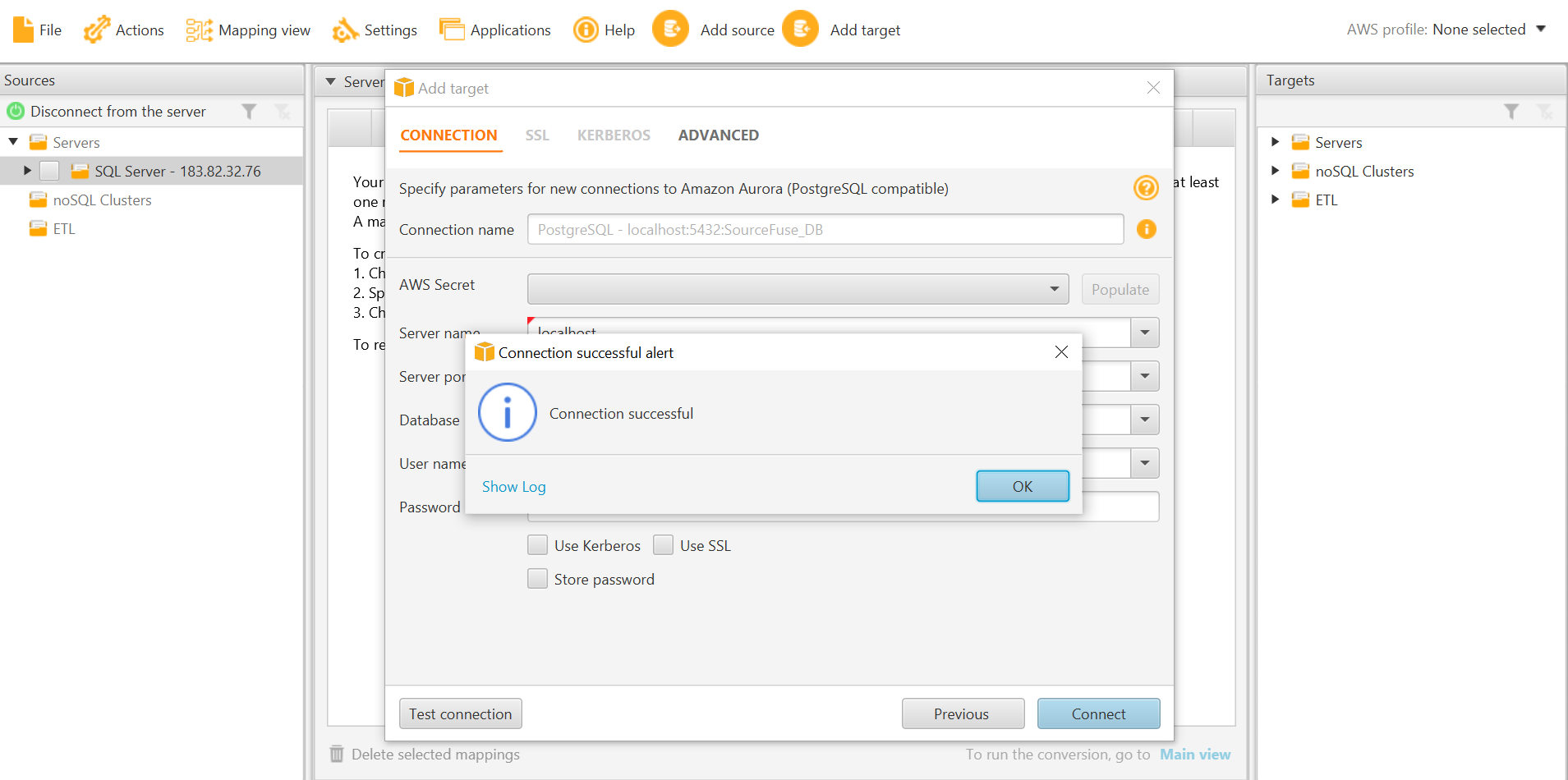
**Step 2.1:** Download and Configure the relevant JDBC JAR files for the source and Target systems into SCT in the Settings 🡪 Global settings 🡪 Driver as prerequisite for the selection of the Source and Target systems.



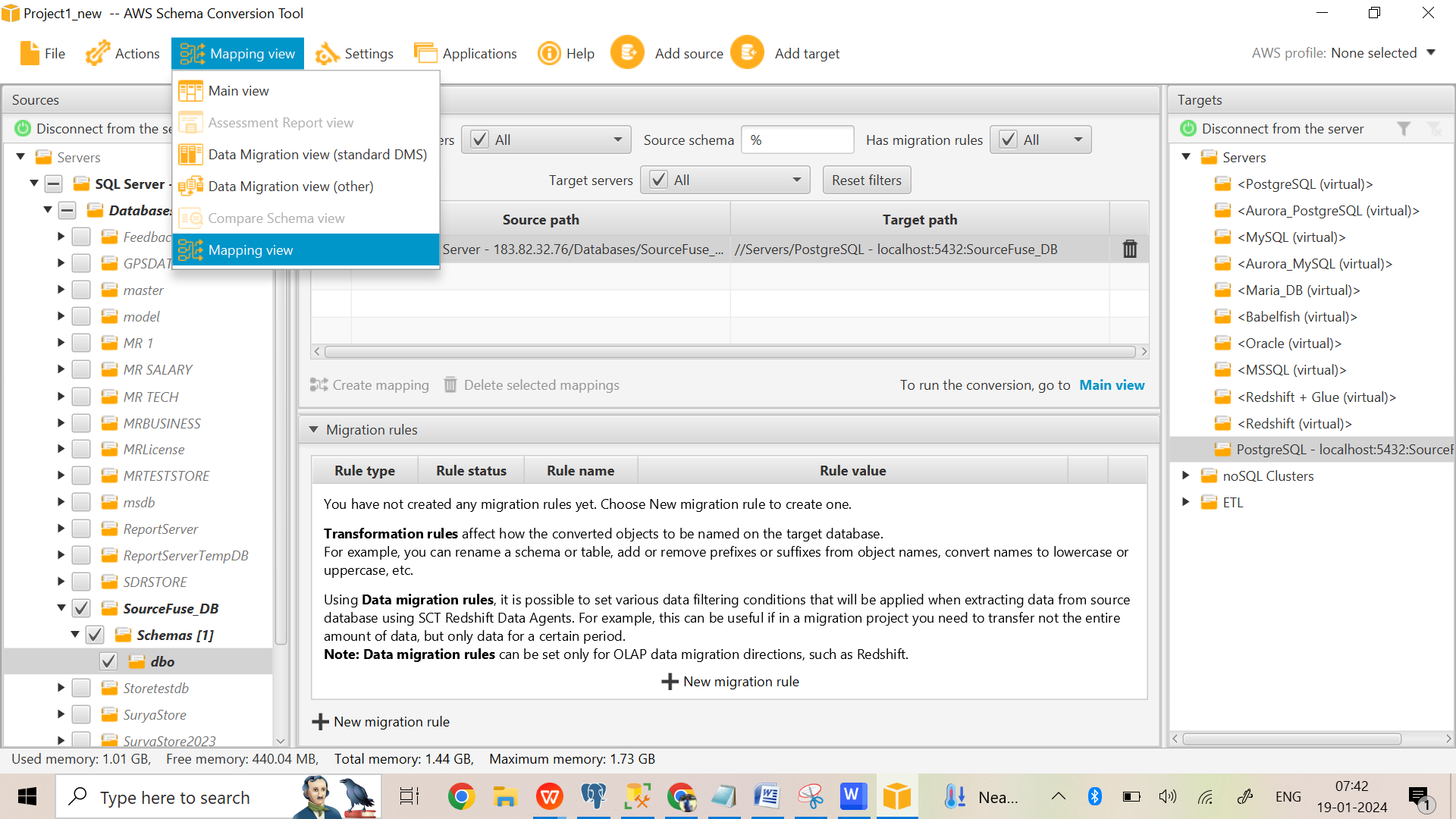
**Step3**: Add the Source database as MS SQL SERVER with proper connection string by selecting Add Source Option in top bar and test the connection.



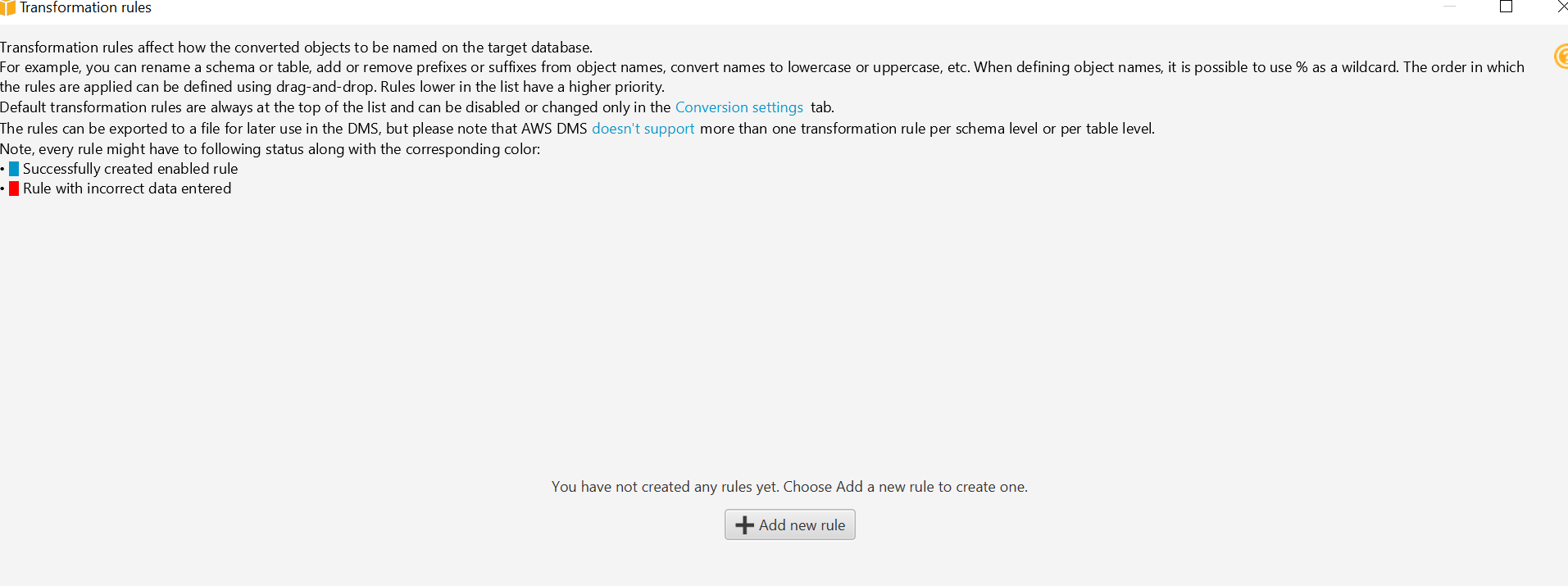
**Step4**: Once the Source DB connectivity done, add the target system as POSTGRESQL with proper connection string by selecting Add Target Option in top bar and test the connection.

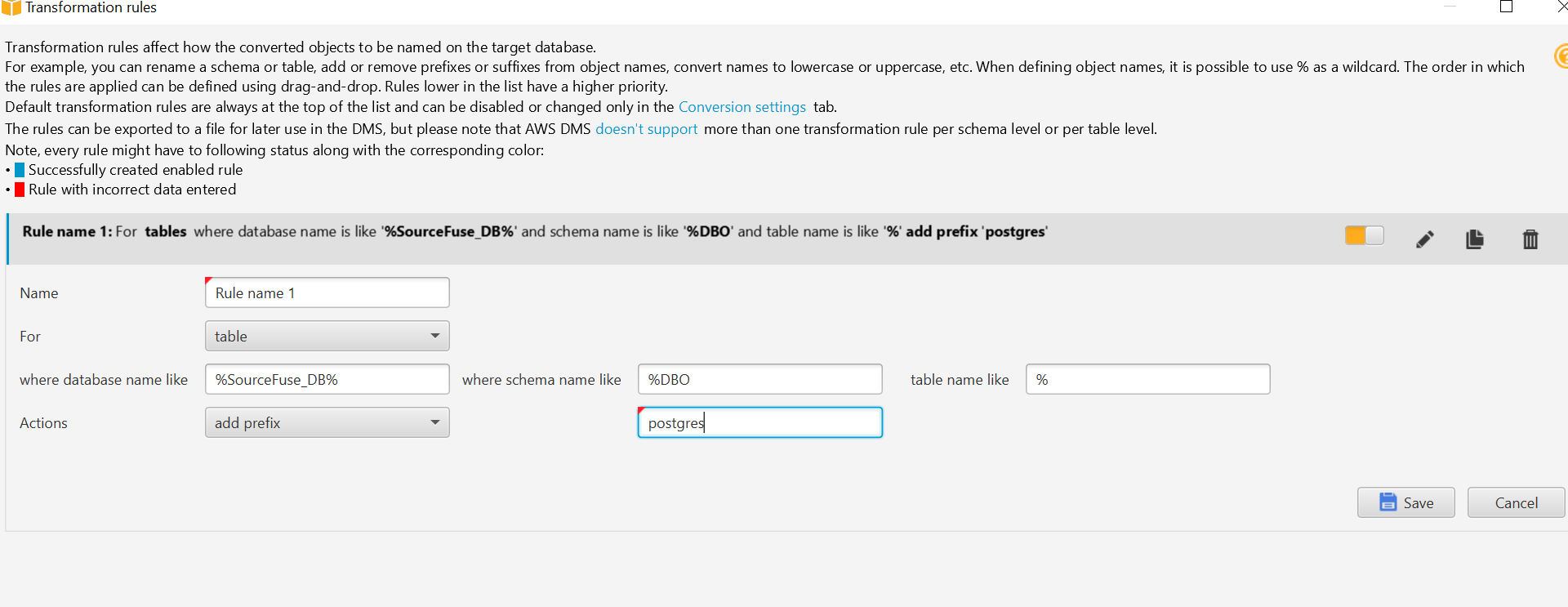


**Step5**: Once the Source and Target system selection and connectivity done, Create the Mapping with Needed Mapping rules to avoid the incompatibility issues.



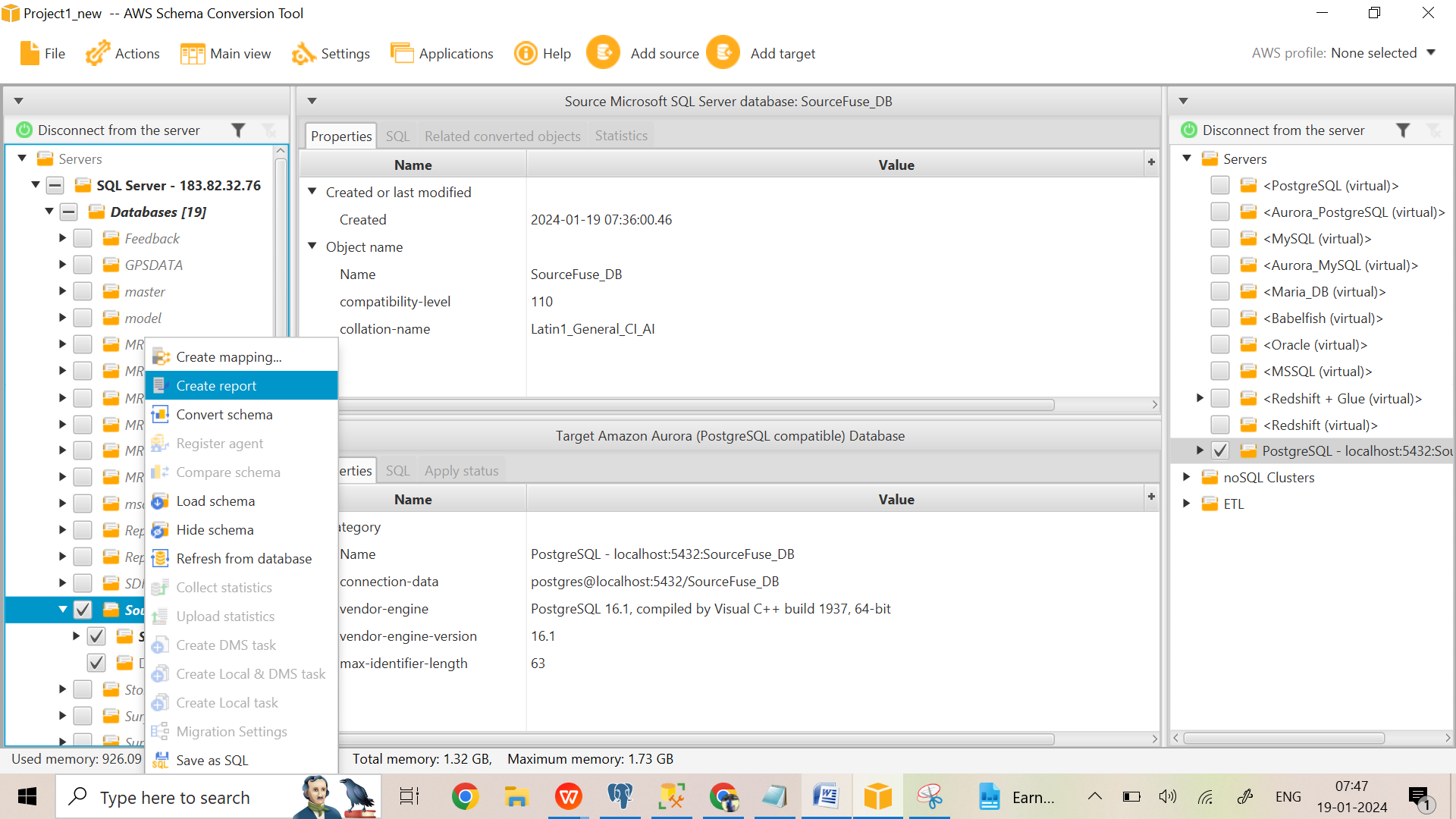
Based on the incompatibility issues in terms of data types / case sensitivity / etc, we can create the mapping rules which specified the transformation rules between source and target as below.



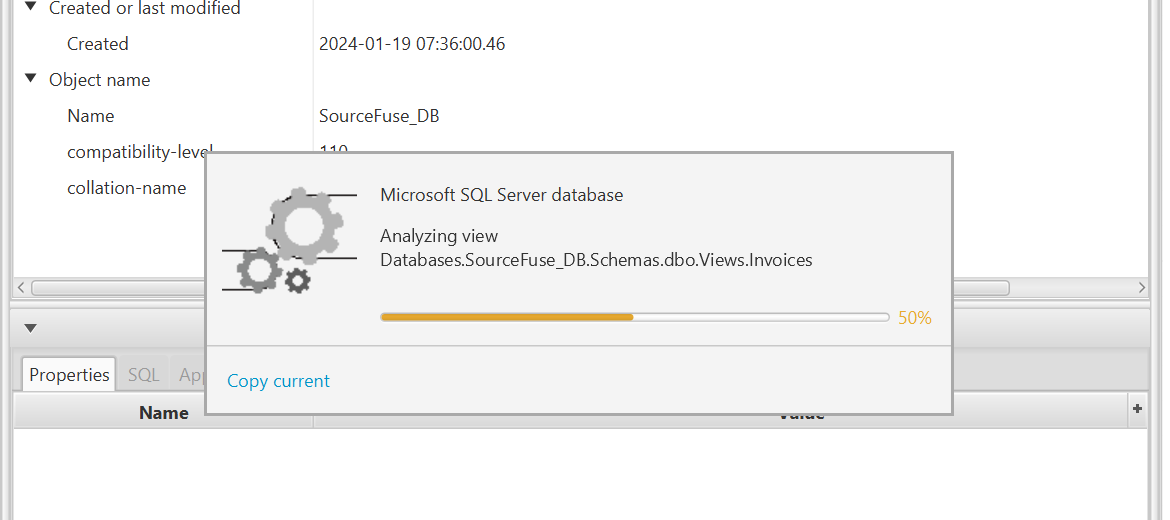


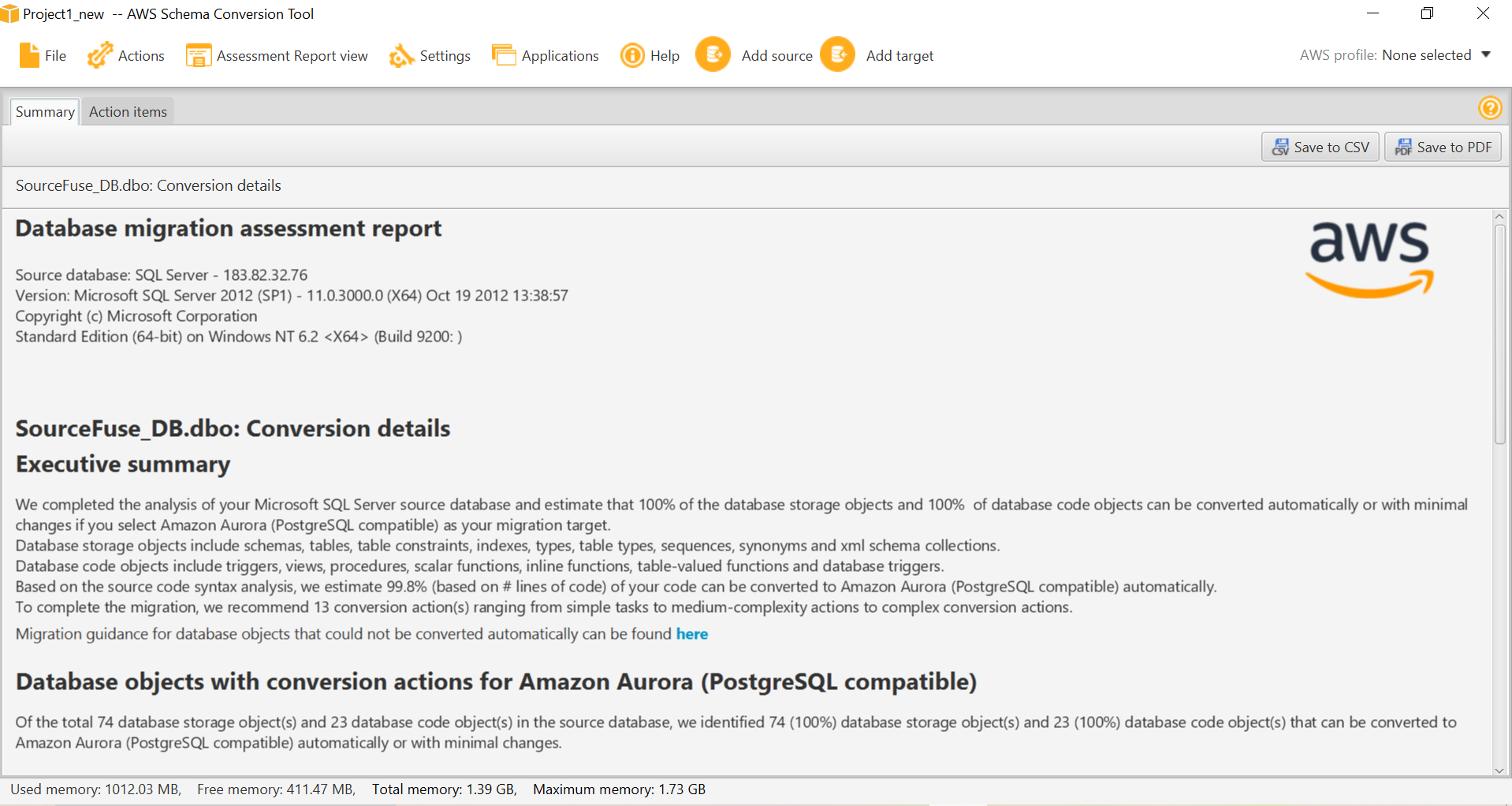
**Step6**: Once the Mapping rule created, generate the DB migration assessment report which describes the compatible and incompatible scenarios/cases to do for proper migration.

We can generate this report by selecting the Create Report option by right click on the source database as follows.

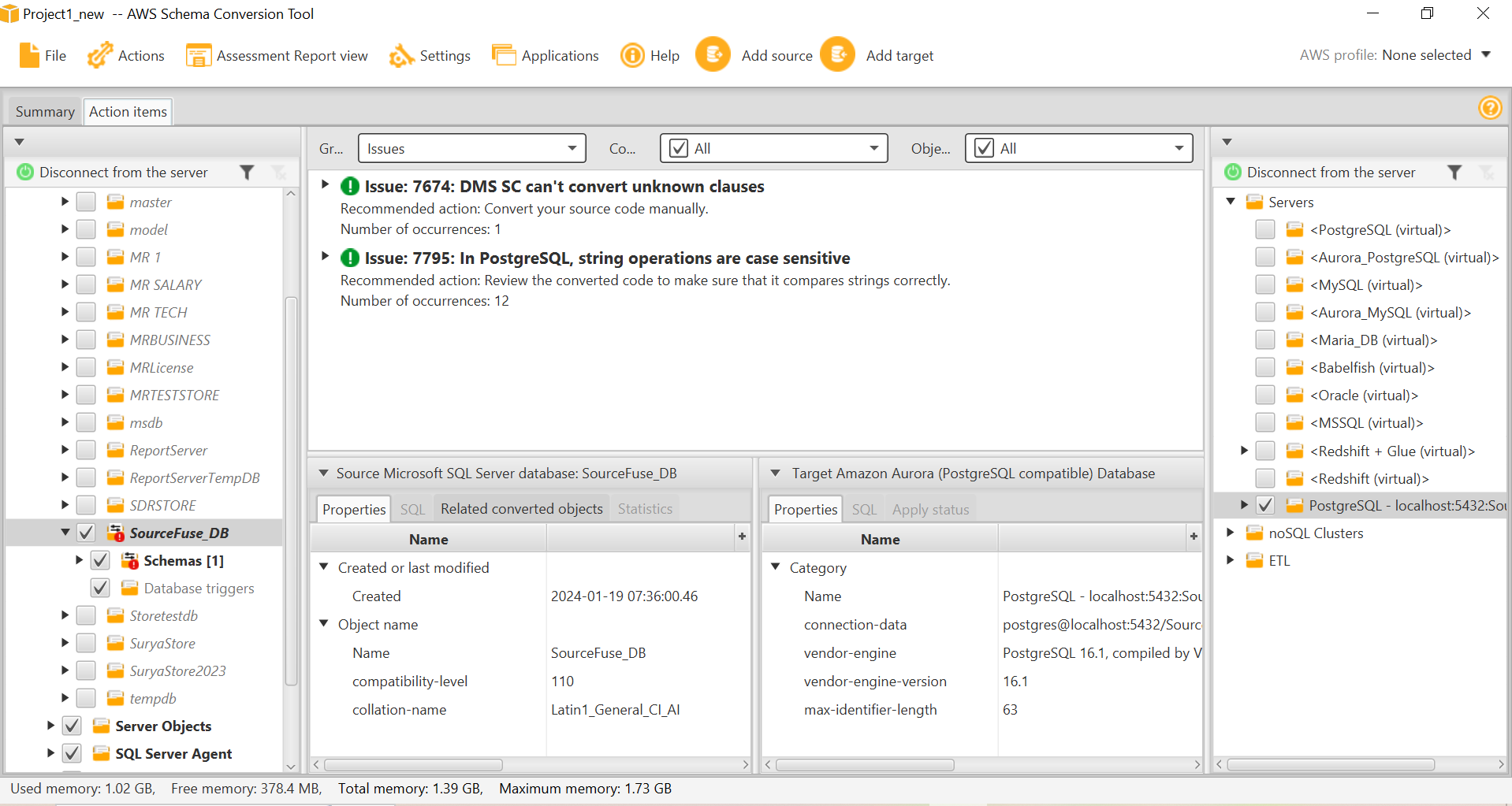


Report generation in progress:



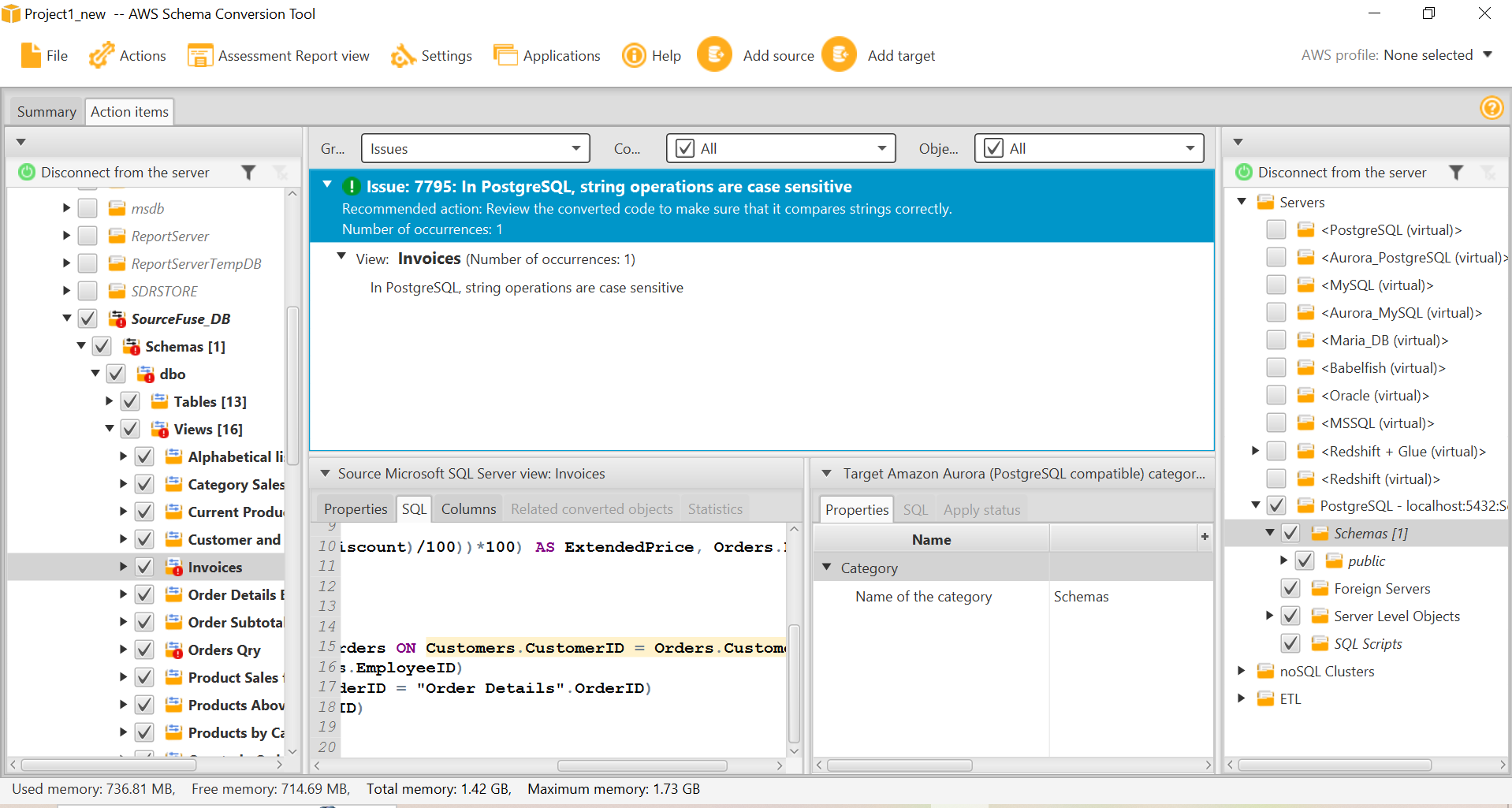
Generated Report view:

Action items to be taken:

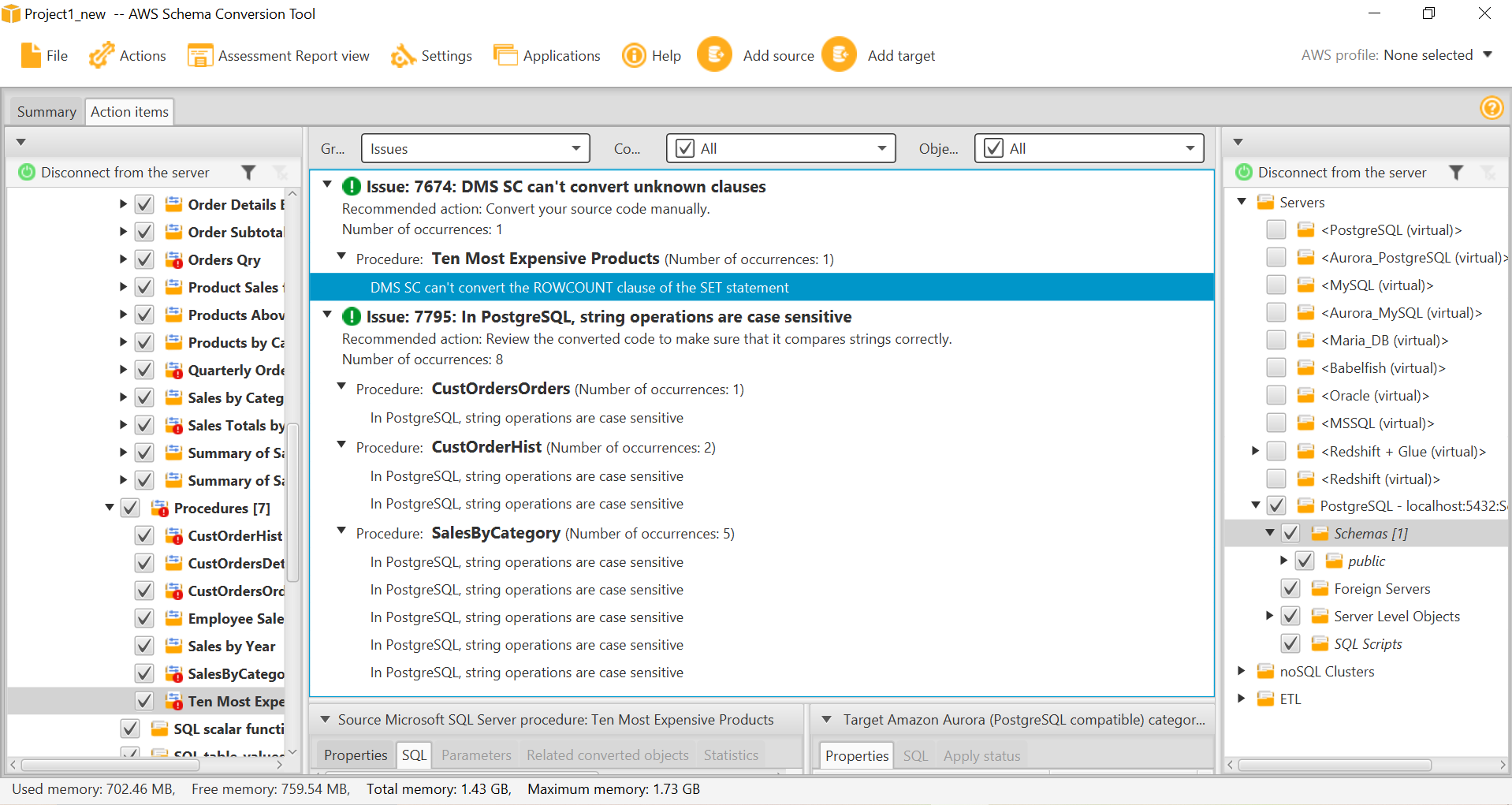


Manual conversion to be done for this migration:

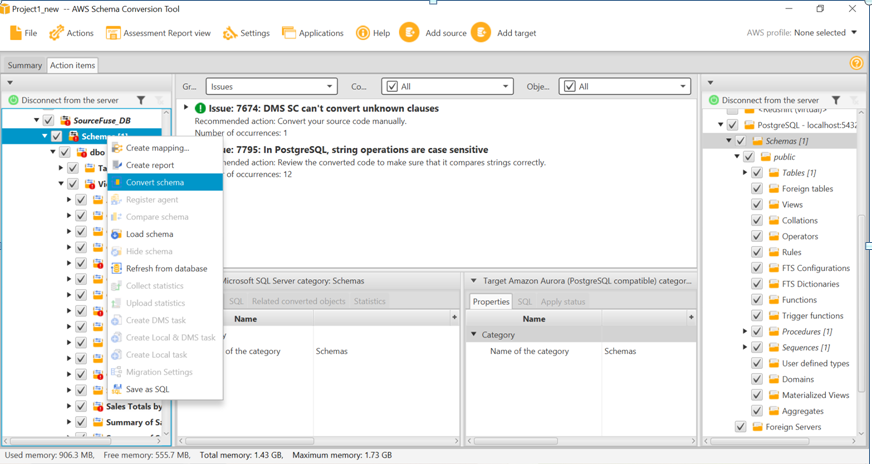
1. **View**: Case sensitivity Issue.(In postgres, String Operations are case sensitive, but in Sql server it is not case sensitive)



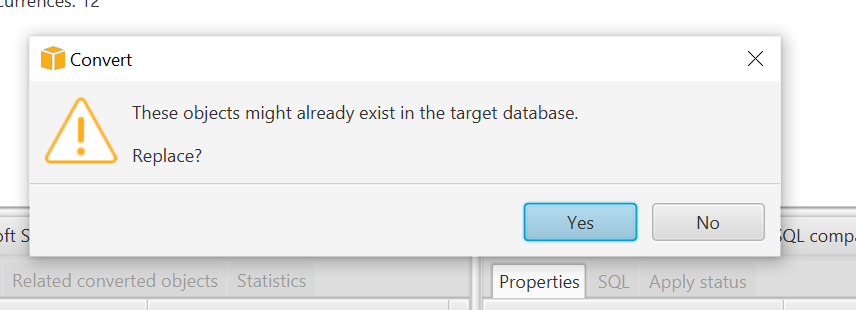
1. **Stored Procedure:**
   1. ROWCOUNT Clause Issue.(In SCT cant convert the ROWCOUNT clause of the SET Statement)
   2. String Operations are Case Sensitive.



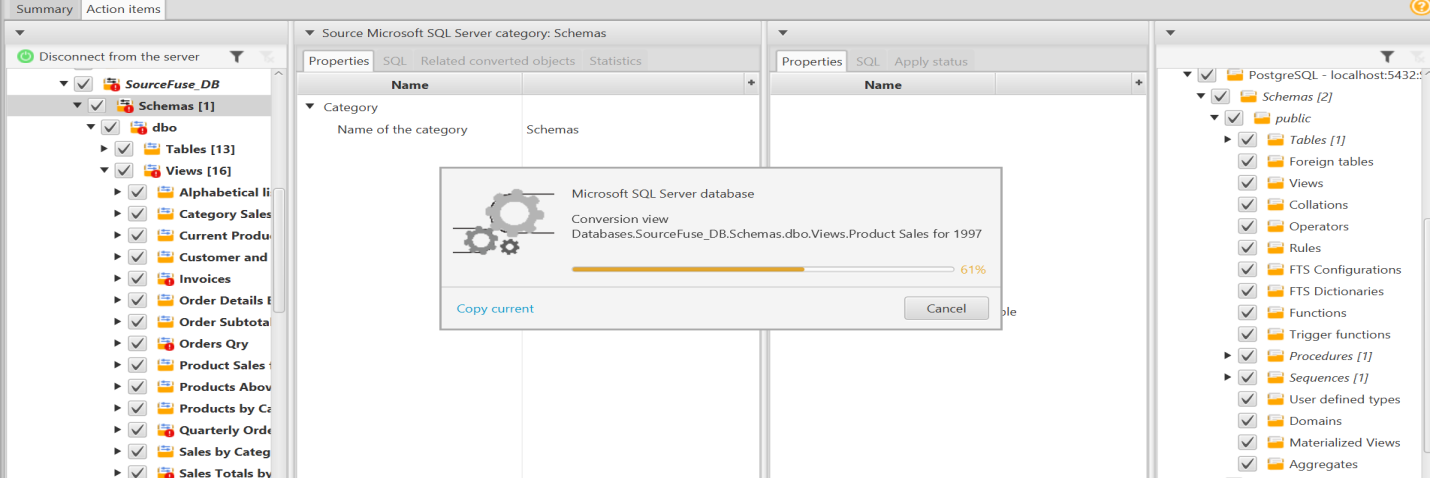
**Step7**: We can do given correction or resolve the incompatibility then convert schema by selecting convert schema option by right click on the source database schema as follows.



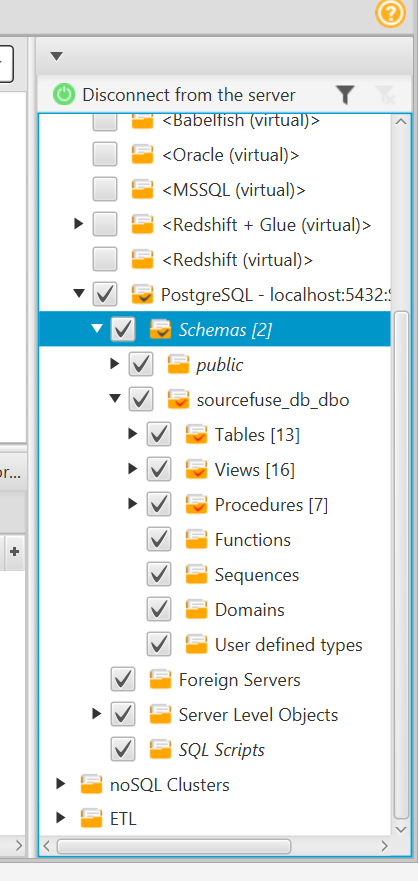
If the objects are already available in target schema, system asks our permission to replace as follows.



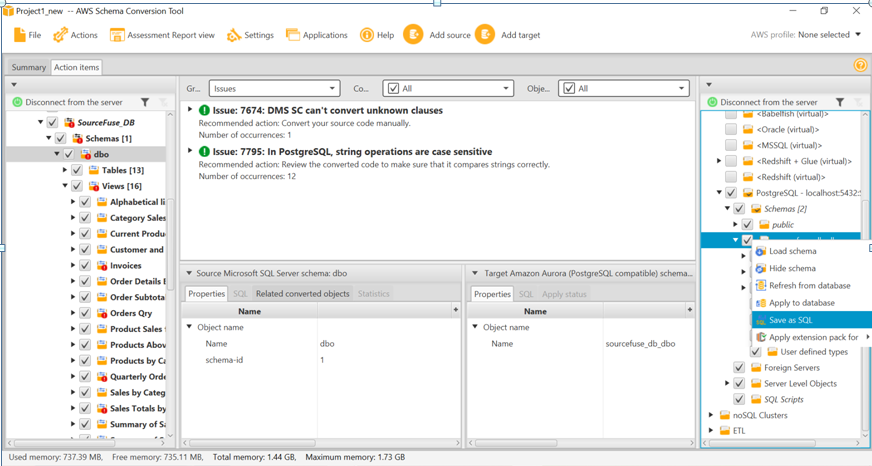
Once we confirmed to replace the existing same objects in target, Schema conversion will takes place as below.

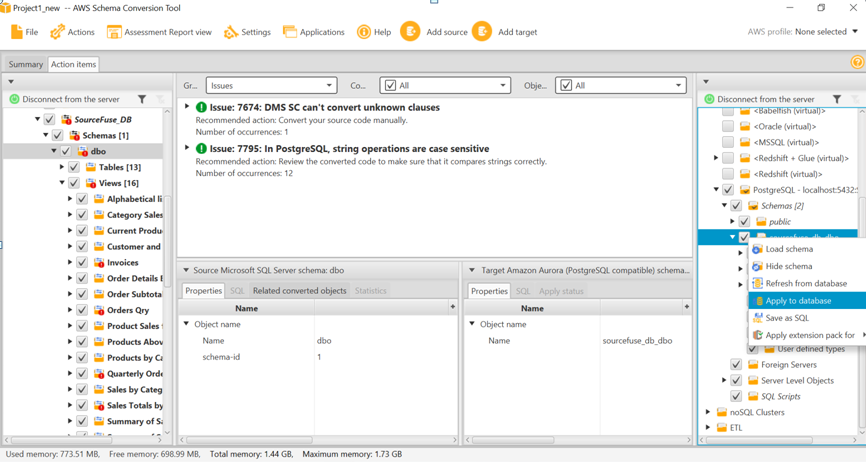


Once the schema conversion done, make sure all the source objects are moved into target as below.



**Step8**: Once the conversion complete then right click on Target database and “Save SQL” or you can also do “Apply to database”.





**Step9**: Once the Migration of the schema has done, need to correct the incompatibility issues between source and target database by using the incompatibility report issue items.

For the data type incompatibility issues, we can modify the script by Alter table script and other also do the manual conversion to those objects which are occurred in AWS SCT Assessment Report. Those database objects need the manual efforts for conversion.

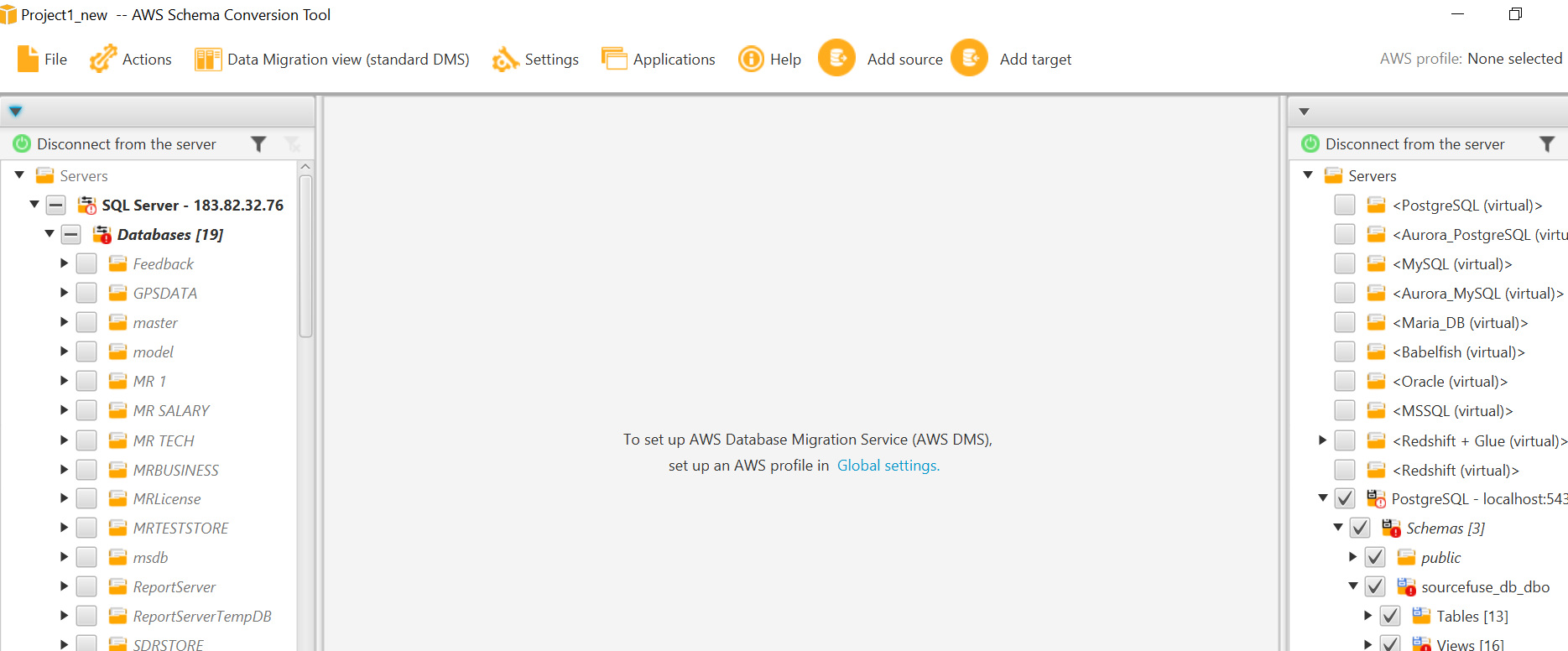
Once the manual conversion done for the incompatibility issue, generate the script and run once again to validate the schema and get the list of objects in source and target databases and confirm the schema conversion was done successfully.

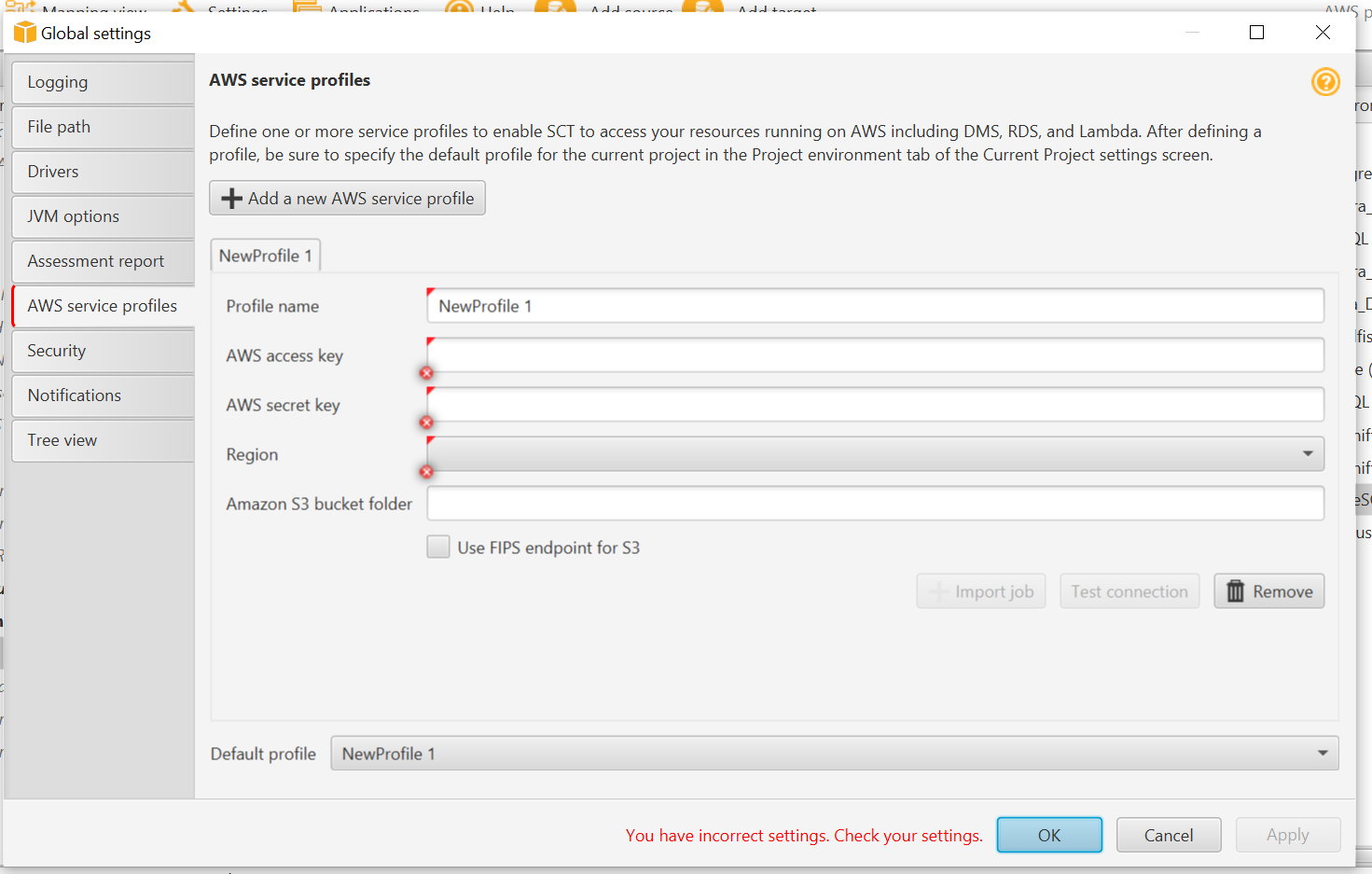
--------------------------------------- End of Schema Conversion using AWS-SCT-------------------------------------------

**Process 2: Data Migration from Sql server DB “SourceFuse\_DB” to Postgres DB “SourceFuse\_DB”**

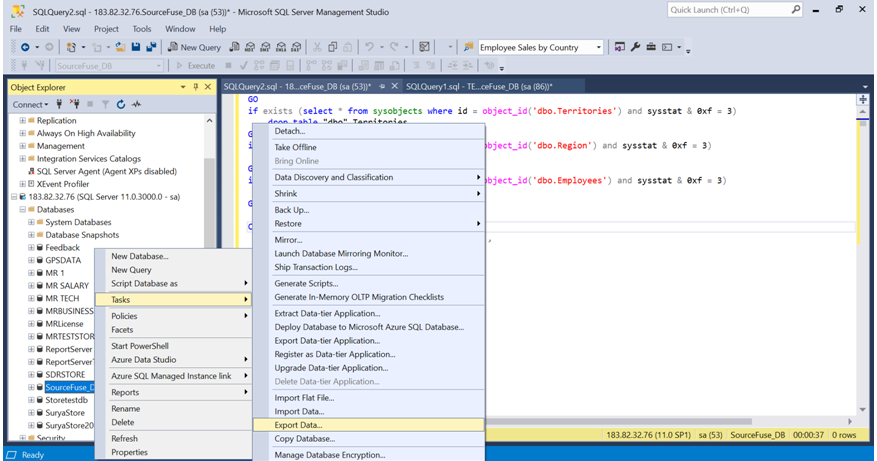
I Have used Import and Export Wizard in Sql server for Data migration process, since I have not used DMS, due to the permission issue.

**DMS Configuration**:





**Step10**: Data Import Using Import Export Wizard:



**Step11**: Validation of the data and the outputs.

Once the Data has been imported into PostgreSQL by the Import and Export wizard in sql server, need to check the Data validation at the target database that is PostgreSQL Database to make sure the data integrity. Data validation to be checked as follows

Validation at no of records count

Validation at Data Type of source and target database tables.

Validation at Data accuracy and consistency.

Validation of the Other Database objects like function, views and stored procedures by passing the relevant input parameters to the objects in target system.

**Step12**: Code commit in Repository

Once the schema conversion and data migration activity done, we have to commit the code into **code repository either in Git** or others for the further review as well as deployment into the testing/production server environment.

----------------------------------------------------------- End of Doc ---------------------------------------------------------------