

LinearBase

How-To Digitally Transform an Existing Database into a Databank Program and Code

Release 0.1

Included

- Transform.zip
- Host.zip
- Docker image linearbase/lineardb-amd64:latest available from <https://hub.docker.com>

Auto-Generated Artifacts

- Databank files
- Databank Plain Old CLR Object (POCO) code
- LinearBase.Host test project
- Profiles supporting Standalone, Centralised, Decentralized, and Distributed deployments
- Initializers for use with LinearBase.Host.exe

Summary

The purpose of Digitally Transforming an existing database is to create a new Databank with optional content and add it to a Realm with the minimum of effort. The auto-generated code allows for relational queries and transactional updates across Windows, Android, iOS, Mac, and Linux with Docker-on-Linux for server, simultaneously.

A Realm contains one or more Realm Identity folders defined by a unique RealmId.

Each RealmId folder contains system Databank files that define a transaction boundary and describe each individual Model's schema, behaviour, and interaction with other transaction boundaries in other Realms.

Download the Transformation and Host Programs

Create a folder to store the programs, it can be any location including a flash drive for example 'E:\LinearBase'.

The transformation process writes Databank files to a Realm folder. For this example, and to keep all the files in one place, create another folder called Realms under E:\LinearBase resulting in E:\LinearBase\Realms. This folder is not fixed and can be moved to another location, if necessary, post transformation.

Navigate to <https://linearbase.com/#download>

Transformation Program

Click or tap the 'Download Transformer' button. This will download a 64Mb compressed file called Transform.zip to your device's download folder.



Digital Transformation Program

Download Transformer

Figure 1 Download Digital Transformation Program

Extract the compressed files to the folder created earlier by selecting Transform.zip and choosing the 'Extract All...' option from the context menu.

Unless you change the default options this will create a new folder E:\LinearBase\Transform.

There are nine files in the zip archive, once decompressed the file we're interested in is LinearBase.Transform.exe. The program requires the other eight support files to run.

How-To Digitally Transform an Existing Database into a Databank Program and Code

Run the program by double clicking or tapping the executable file.

Host Program

Click or tap the 'Download Host' button. This will download a 31Mb compressed file called 'Host.zip' to your device's download folder.



Windows Host Program

Download Host

Figure 2 Download Windows Host Program

Extract the compressed files to the folder created earlier by selecting Host.zip and choosing the 'Extract All...' option from the context menu.

Unless you change the default options this will create a new folder E:\LinearBase\Host.

There are three files in the zip archive, once decompressed the file we're interested in is LinearBase.Host.exe. The program requires the other two support files to run.

The LinearBase.Host.exe program is initiated using launchOptions.json configuration file or by command line arguments. The transformation process auto-generates these settings for both options and writes them to the relevant Realm folder upon completion.

The 'Start in Console on Finish' option described later uses the command line method to initiate a standalone 'Centralized' topology instance for use in development.

Transformation Process

The current implementation of LinearBase.Transform is a development mule Windows desktop program. In its present form the Windows desktop Transformation program limits selection to Microsoft SQL Server databases version 2005 onwards.

Beta test program participants can suggest the priority of database providers to include next.

The Transformation Program can complete a digital data transformation in as few as 10 clicks!

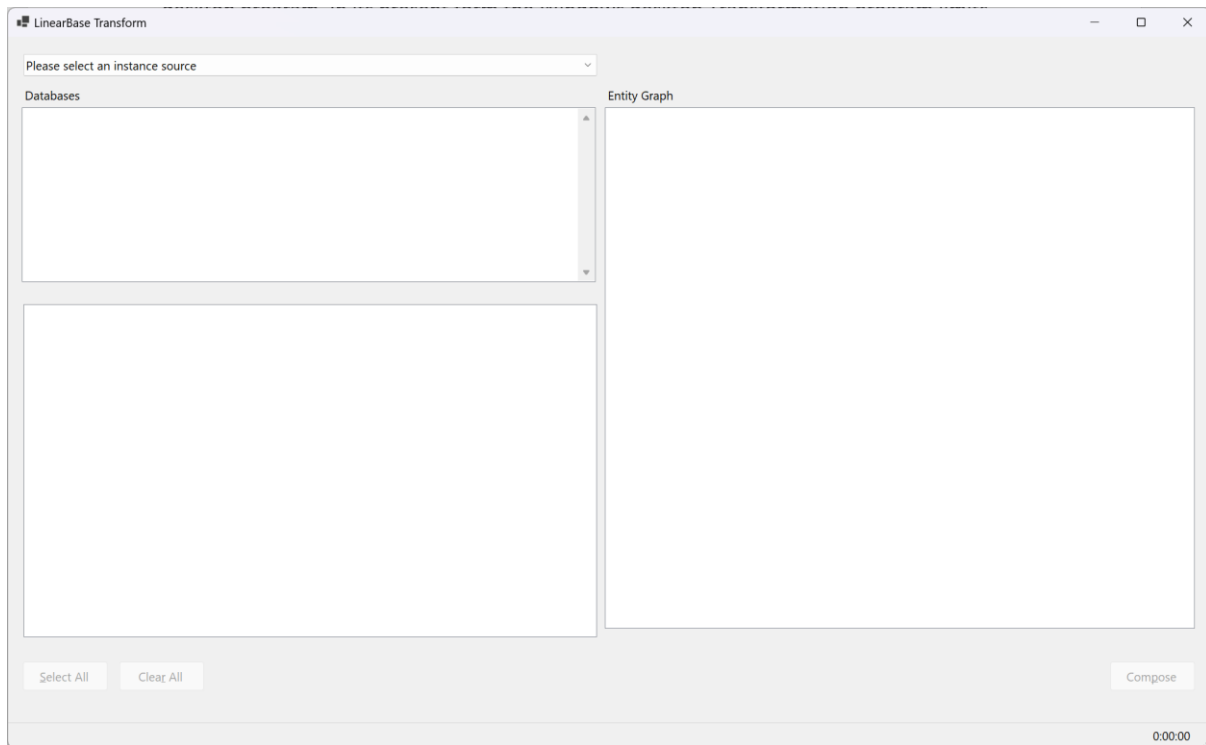


Figure 3 LinearBase Transform

Under development is a Web based program and an App accessed via the built-in Admin Portal. The new version introduces Domain Modelling, Data Modelling, and Transformation capabilities across other latent data sources, e.g., RDBMS, Document Stores, Avro Data Lakes, etc.

Select a Data Instance Source

Identify the SQL Instance to analyse.

The three options are

- Network Microsoft SQL Instance
- ODBC File DSN (disabled in this release)
- Any Local SQL Instances

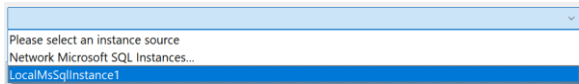


Figure 4 Select Data Instance Source

Selecting an option opens the MSSQL Connection Criteria input dialogue.

MSSQL Connection Criteria

Choose the security model to use when connecting to the chosen MSSQL instance.

The three options are

- Trusted Connection, use the logged-on user's credentials
- Impersonate Trusted Connection using supplied credentials
- SQL Authentication using supplied credentials

Impersonate and SQL both prompt for User Identity and Password.

Test the connection prior to connecting. Timeout is set to seven seconds.

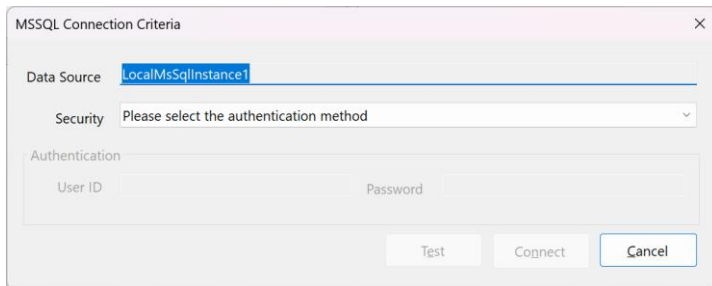


Figure 5 MSSQL Connection Criteria

Database Selection

Upon a successful connection all databases in the chosen MSSQL Instance appear in the Database list

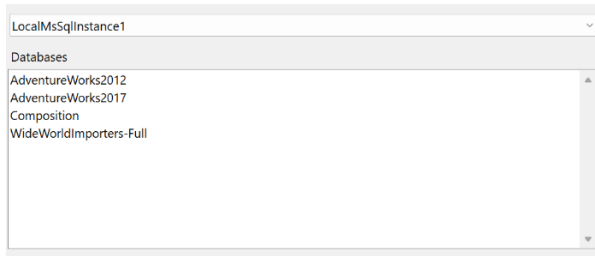


Figure 6 Database List

Select a Database

Select a database from the list to model.

Upon selection the Transformation program analyses the tables and relationships. A Cancel button, progress bar, and timer appear at the bottom right of the form.

Once complete the database tables appear in the Table list below the Database list.

Database implementations vary enormously. It may not be possible to transform some databases. The modelling process looks for Primary Keys, Identity fields, and Foreign Key associations.

The intention is to follow the Domain Driven Design philosophy, modelling related tables as separate Programs. In a future release the modelling process implements cross domain and external Program associations helping build complex object graphs.

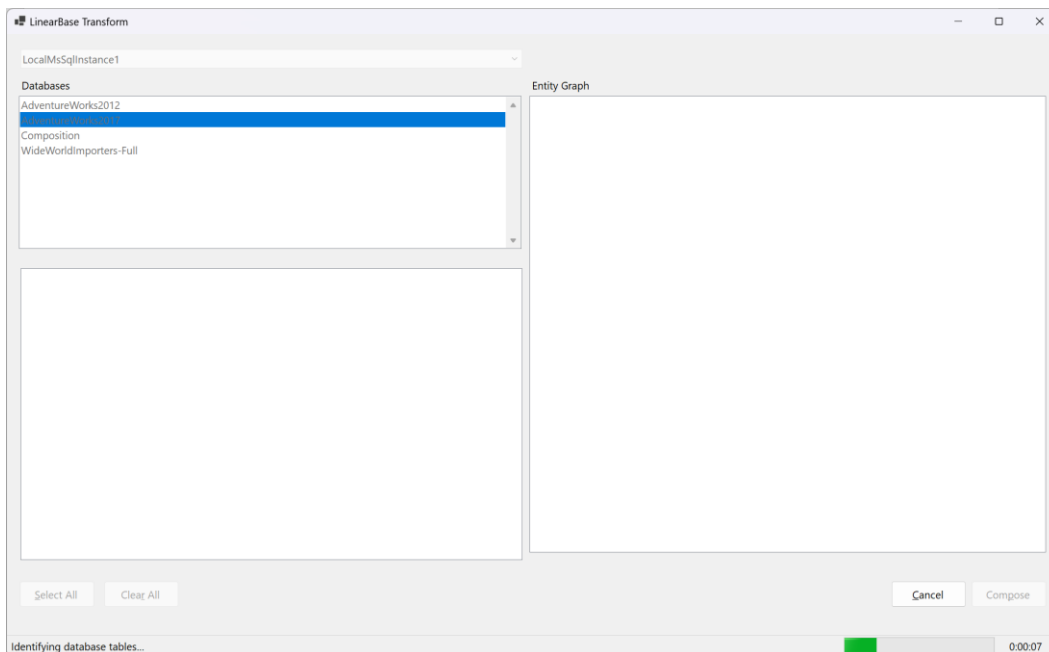


Figure 7 Select a Database

Select the Model

Build a Model of the Program to transform.

The Table list allows control over which tables to include or exclude from the Model. Double click or use the space bar to check or uncheck each checkbox.

Analyse each table and its relationships by selecting the table in the table list. The table structure appears in the Entity Graph control on the right.

The Entity Graph allows control over which columns to include or exclude from the Model.

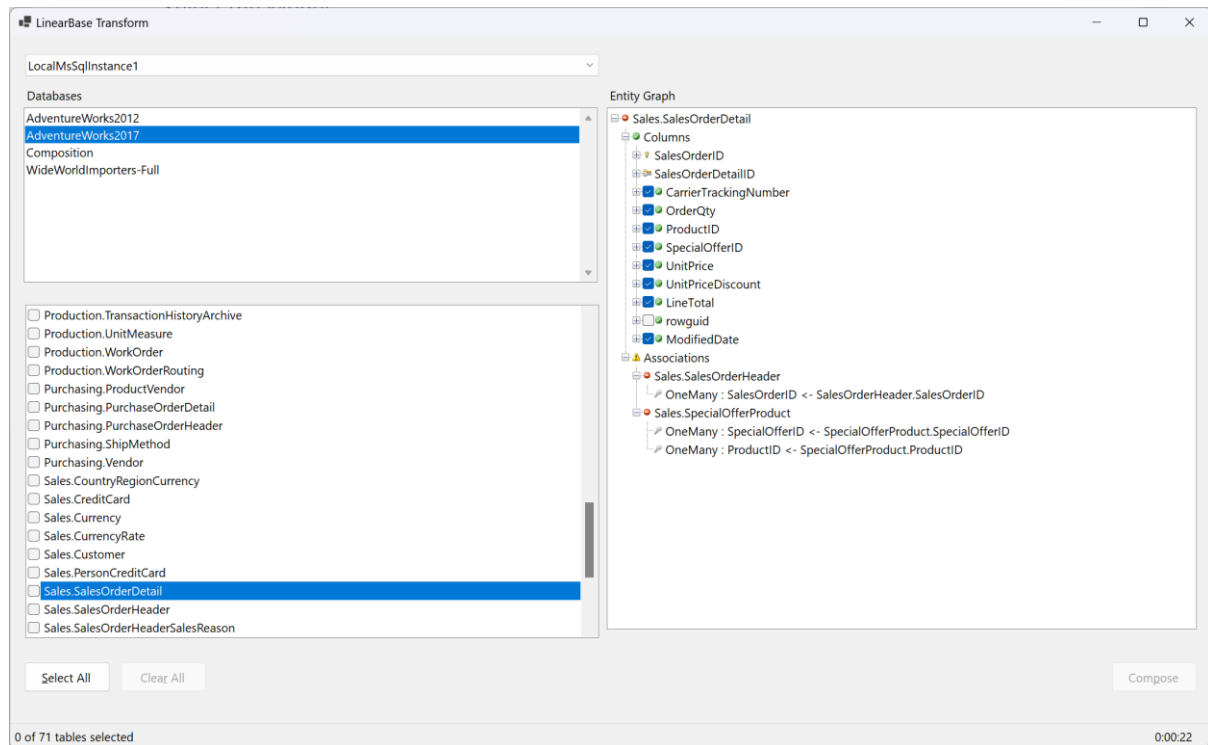


Figure 8 Select the Model

This example includes all Sales tables from the AdventureWorks2017 database as the Model.

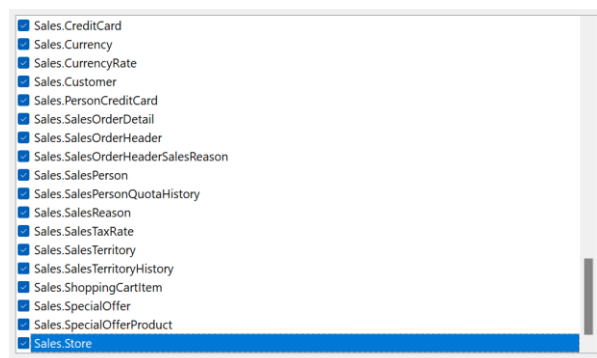


Figure 9 Sales Model

Once selection is complete and more than one table is selected the Compose button becomes available at the bottom right of the form.

Compose the Model

Compose transforms the Model schema, and optionally data, and saves the product as a format-neutral binary file known as a Databank ending with a .data extension.

To implement a transition the Model requires a principal object, known as a root aggregate object in Domain Driven Design parlance. The process remodels associations to flow out from the principal object in effect reversing associations. SQL queries define their own join associations so do not affect and are not affected by the transition.

Each Databank is associated to Realm. A Realm represents the transaction boundary and contains system Databank files that describe schemas, associations, transactions, and more. If creating a new Realm, as is the case with the first transformation, enter a descriptive name in the Realm field, for example Adventure. This name is appended to the file location path along with an automatically generated unique Realm Id. A generated example maybe E:\LinearBase\Realms\Adventure\NEWREALM.

Once a Realm exists, appending more Databanks to it requires the existence of a Primary Active Realm Service (PARS) instance. This is the Realm controller, intended as a Docker image for production but for development the 'Start in Console on Finish' option will initiate a PARS instance to help understand the different deployment scenarios. Alternatively, initiate any Program at any time in any configuration using the LinearBase.Host.exe program downloaded earlier.

To append a Databank use the Windows File Explorer to identify existing Realms, for example navigating to E:\LinearBase\Realms will 'sniff' and populate the dropdown list with any unique previously created realm names. Once an existing Realm is selected from the list the PARS Host field is enabled. This requires the URL and port number of the active PARS instance, for example <http://127.0.0.1:60001>. On completing this field, the transformation process communicates with the PARS instance to confirm it is active and represents the correct Realm in the dropdown.

During the Publish phase described later, the transformation process downloads and interprets the binary Databank file and constructs a volatile in-memory representation of the Model.

Instead of database, LinearBase uses the term Program to describe an instance of a Databank. This is because the in-memory representation is a Strongly Typed Object Graph in programmatic form, in contrast to rectangular database Entity Relationship form; there is no impedance-mismatch.

There will likely be many Realms from many internal and external sources that subsequently use modelling to form additional associations between Programs. Once modelled, distributed Programs appear as a single programme object graph and can be queried as such.

Profiles are key in LinearBase. They're stored in the system Registry Databank and control how each instance interprets Databank files. The transformation process auto-generates four default Profiles for topologies covering Standalone, Centralized, Distributed, and Decentralized Program scenarios. Standalone is used to implement data-privacy on devices. A future release will include a Profile manager to create and modify Profiles.

The Compose input dialogue fields are mandatory

How-To Digitally Transform an Existing Database into a Databank Program and Code

- Databank ID – read-only auto generated unique Databank identifier also known as a Program Id
- Alias – a friendly name for the Program for use in SQL queries (omitted in this release)
- Principal – the Program object graph’s root aggregate
- Path (1) – directory to save the Databank file to, click to open the Windows File Explorer
- Realm – New name or selected from the dropdown list
- PARS – the URL and port number of the PARS host (append Databank to Realm only)
- Meta Data – include database schema with the output (disabled in this release)
- Content – include database content with the output
- Case Sensitive Meta Data – Create profiles specifying case-sensitive schema
- Case Sensitive Content – Create profiles specifying case-sensitive content
- Culture - Create profiles specifying specific culture
- Start in Console on Finish – Load Databank as new instance on completion
- Path (2) – File path to LinearBase.Host.exe program, click to open the Windows File Explorer
- Port – The Port the new instance should listen on locally
- Generate Code – Option to generate associated Databank programme code (disabled in this release, set to true)

The screenshot shows a 'Compose' dialog box with the following fields and options:

- DataBank**
 - DataBank Id: LFWMKUZD
 - Alias: User friendly alias, max 15 alpha alpha-numeric characters
 - Principal: Sales.SalesOrderHeader
- Publish to Realm**
 - Path: Realm network share or local folder to write Databank file to, e.g. \\LinearBase\Realms
 - Realm: [Dropdown]
 - PARS Host: http://127.0.0.1:60001
 - Meta Data (Discovery):
 - Content:
 - Case Sensitive Meta Data:
 - Case Sensitive Content:
 - Culture: Program Culture...
- Start in Console on Finish
- Path: Filepath to LinearBase.Host.exe | Port: Listen on
- Generate Code
 - Language: Microsoft .NET CSharp v11
 - Use new Databank file path (recommended):
 - Zip: Filepath to write zipped code files to, e.g. \\LinearBase\CodeBase
 - Include Host Console Demonstration Code:
 - Include docker-compose.yml File:

Buttons: Compose, Cancel

Figure 10 Compose Input Dialogue

Sales.SalesOrderHeader is the principal object.

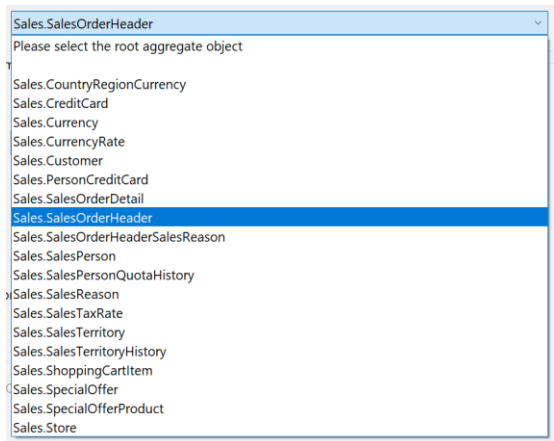


Figure 11 Principal Object

In this example:

- Databank Id – LFWMKUZD (auto-generated)
- Alias - Sales
- Principal - Sales.SalesOrderHeader
- Realm Path - E:\LinearBase\Realms
- Realm Name – Adventure
- Culture – English – United Kingdom
- Start in Console on Finish – True
- Host.exe Path - E:\LinearBase\Host
- Host Port - 60002

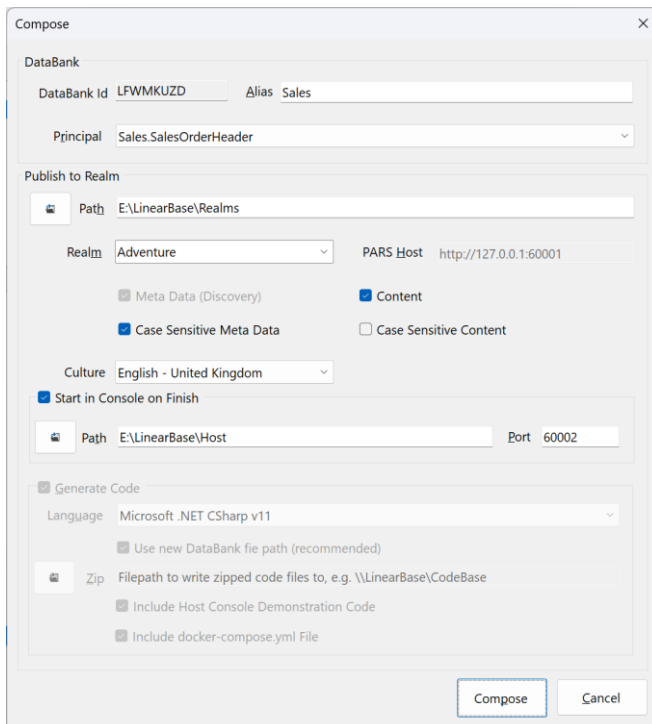
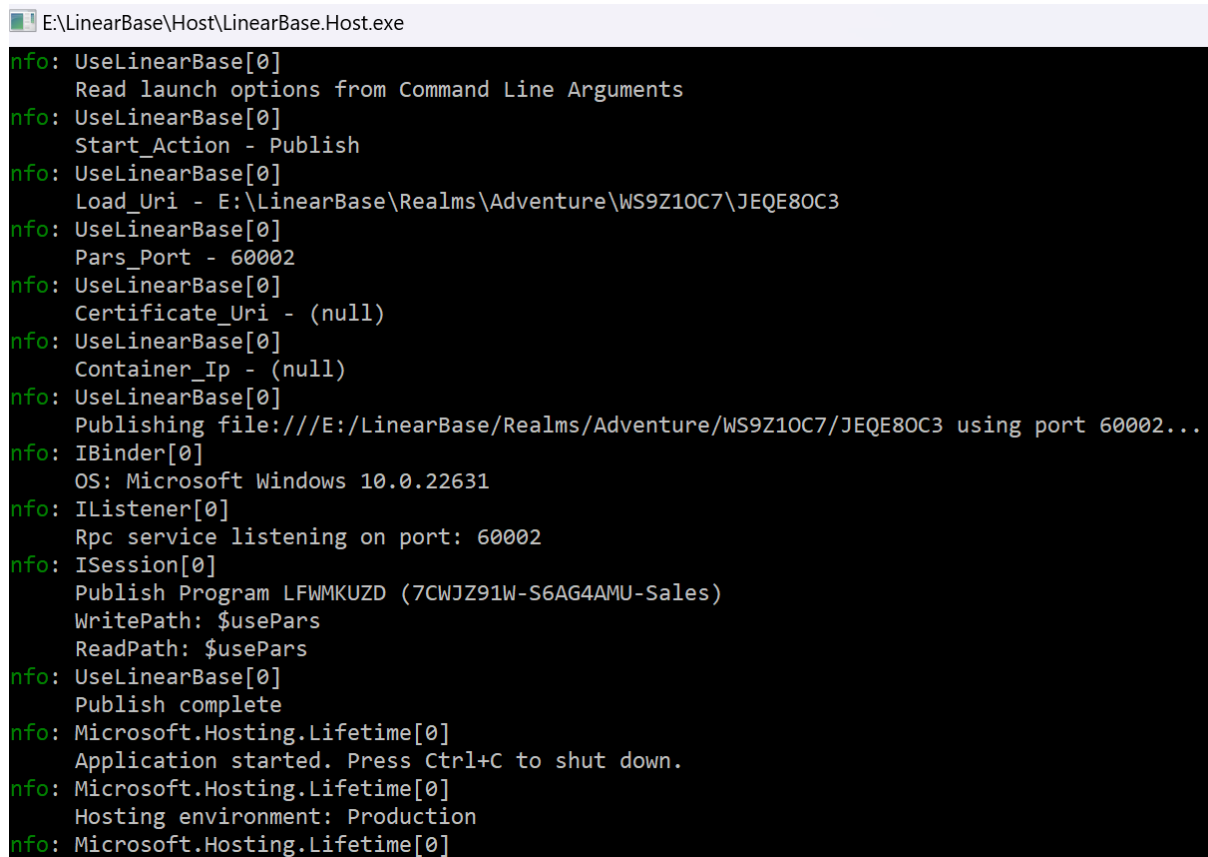


Figure 12 - Completed Compose Form

Once all fields are complete click the Compose button to complete the transformation.

The progress bar bottom right provides feedback. The Compose dialogue reports success or failure, closes automatically, and clears the model.

A separate console opens and loads an active PARS instance for the new 'Adventure' Realm we just created.



```
E:\LinearBase\Host\LinearBase.Host.exe
nfo: UseLinearBase[0]
    Read launch options from Command Line Arguments
nfo: UseLinearBase[0]
    Start_Action - Publish
nfo: UseLinearBase[0]
    Load Uri - E:\LinearBase\Realms\Adventure\WS9Z1OC7\JEQE8OC3
nfo: UseLinearBase[0]
    Pars_Port - 60002
nfo: UseLinearBase[0]
    Certificate Uri - (null)
nfo: UseLinearBase[0]
    Container_Ip - (null)
nfo: UseLinearBase[0]
    Publishing file:///E:/LinearBase/Realms/Adventure/WS9Z1OC7/JEQE8OC3 using port 60002...
nfo: IBinder[0]
    OS: Microsoft Windows 10.0.22631
nfo: IListener[0]
    Rpc service listening on port: 60002
nfo: ISession[0]
    Publish Program LFWMKUZD (7CWJZ91W-S6AG4AMU-Sales)
    WritePath: $usePars
    ReadPath: $usePars
nfo: UseLinearBase[0]
    Publish complete
nfo: Microsoft.Hosting.Lifetime[0]
    Application started. Press Ctrl+C to shut down.
nfo: Microsoft.Hosting.Lifetime[0]
    Hosting environment: Production
nfo: Microsoft.Hosting.Lifetime[0]
```

Figure 13 -Primary Active Realm Server (PARS) instance hosted in a console application

DO NOT CLOSE THIS CONSOLE APPLICATION it's hosting our PARS instance on <http://localhost:60002> and we need it in this example to append additional Models to the Realm.

The line '**Load Uri - E:\LinearBase\Realms\Adventure\WS9Z1OC7\JEQE8OC3**' identifies this as a Centralized PARS instance as it's loading from a file location not a URL as used in Distributed and Decentralized deployments we'll see later.

Cloudless enabled Apps and programs can now interact with this instance in Centralized, Distributed, and Distributed data topologies simultaneously. For more information on topologies, Profiles, and interacting with an active PARS instance read the article ['How-To Host a Databank Program'](#)

Next, we'll use the Command Line option to append the address.json Model to the new 'Adventure' Realm we just created.

First, we need to identify the auto-generated RealmId containing the system Databank files.

Navigate to E:\LinearBase\Realms\Adventure. Here we find three new folders.

Name	Date modified	Type
Initializers	09/04/2024 15:25	File folder
LFWMKUZD	09/04/2024 15:25	File folder
WS9Z1OC7	09/04/2024 15:25	File folder

Figure 14 - Adventure Realm Databank Folders

Folder LFWMKUZD contains the newly Transformed Sales Program and associated code. LFWMKUZD is the new Program's unique auto-generated ProgramId. This is the Databank Id from the Compose form.

Name	Date modified	Type	Size
Schema	09/04/2024 15:25	File folder	
7CWJZ91W-S6AG4AMU-Sales.data	09/04/2024 15:25	DATA File	11,647 KB

Figure 15 - Sales Program Databank & Code

Folder GZ5HEJP3 contains the system Databank files, it's this Id we need to append additional Models to the new RealmId.

Name	Date modified	Type
BINDBASE	09/04/2024 15:25	File folder
COMPOSER	09/04/2024 15:25	File folder
INCIDENT	09/04/2024 15:25	File folder
REGISTRY	09/04/2024 15:25	File folder
SERVICES	09/04/2024 15:25	File folder
TRANSACT	09/04/2024 15:25	File folder
TRIGGERS	09/04/2024 15:25	File folder

Figure 16 - Realm System Databanks

Append Additional Transformations to an Existing Realm

To append additional Transformations, repeat the process up to the Compose stage. Let's choose AdventureWorks2107 HumanResources this time.

How-To Digitally Transform an Existing Database into a Databank Program and Code

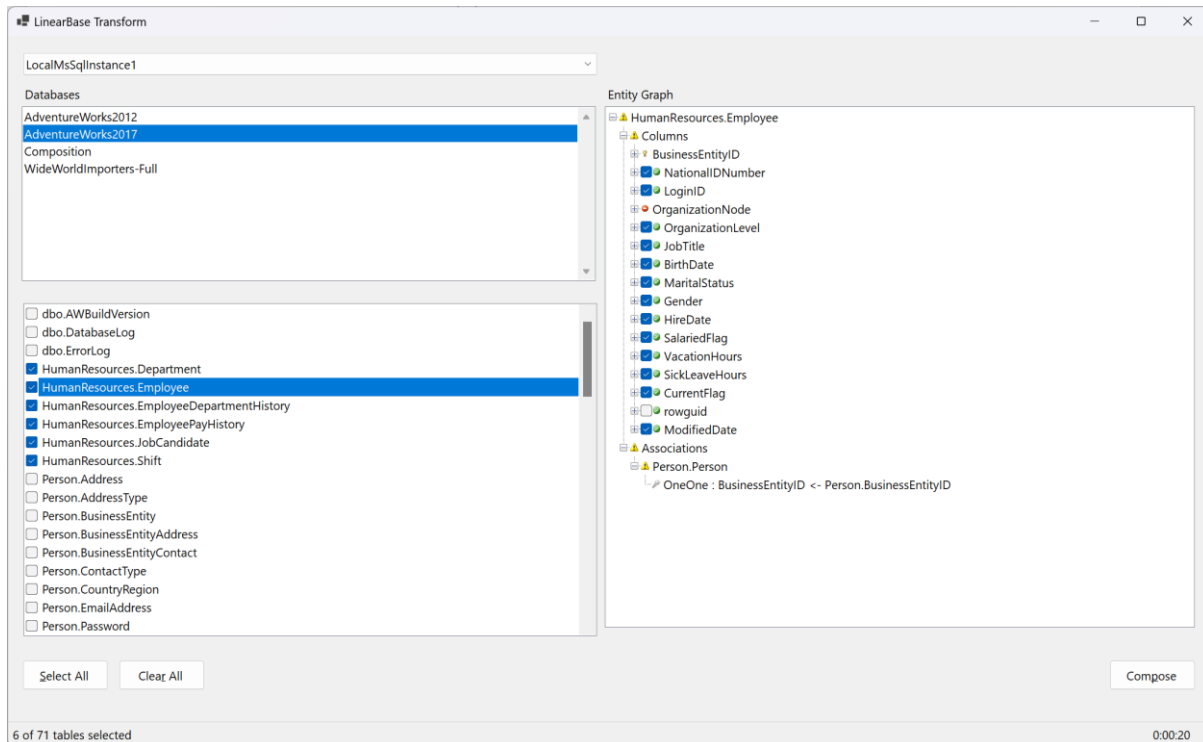


Figure 17 - AdventureWorks HumanResources

To append, instead of entering a new Realm Name, we select the path of the existing Realm, in this case Adventure’.

When selecting the Realm Path, the process checks the location for existing Realm Identities containing system Databank files and lists the associated Realm Name in the dropdown list. The only option available in this example is Adventure.

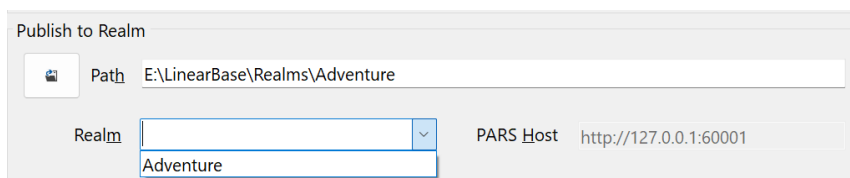


Figure 18 - Adventure Realm Option

Select this and the PARS Host field is enabled to enter the URL of the active PARS instance we started earlier: <http://localhost:60002>. On exiting this field, the process contacts the active PARS instance and checks the details are correct and reports any issues.

In this example:

- Databank Id – DZ7B7WM1 (auto-generated)
- Alias – HumanRes- it is important this is unique within the Realm
- Principal – HumanResources.Employee
- Realm Path - E:\LinearBase\Realms\Adventure
- Realm Name – Adventure
- Culture – English – United Kingdom
- Start in Console on Finish – True

How-To Digitally Transform an Existing Database into a Databank Program and Code

- Host.exe Path - E:\LinearBase\Host
- Host Port – 60002 (disabled when appending a Model)

The screenshot shows the 'Compose' dialog box with the following configuration:

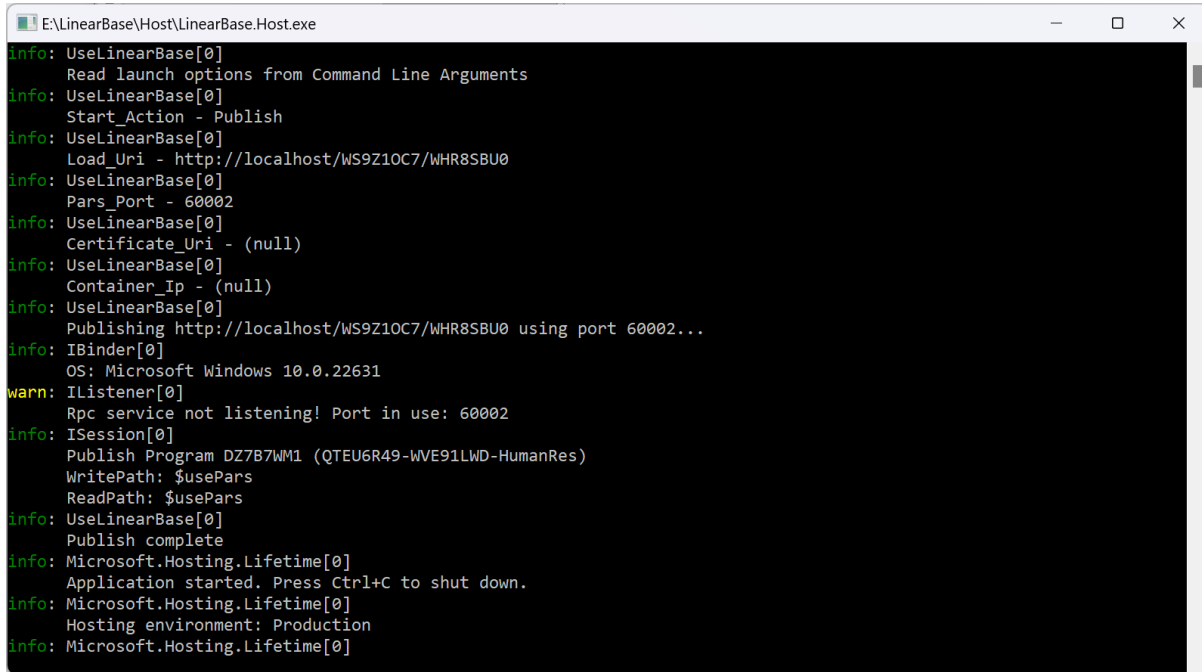
- DataBank:**
 - DataBank Id: DZ7B7WM1
 - Alias: HumanRes
 - Principal: HumanResources.Employee
- Publish to Realm:**
 - Path: E:\LinearBase\Realms\Adventure
 - Realm: Adventure
 - PARS Host: http://localhost:60002
 - Meta Data (Discovery)
 - Content
 - Case Sensitive Meta Data
 - Case Sensitive Content
 - Culture: Program Culture...
 - Start in Console on Finish
 - Path: E:\LinearBase\Host
 - Port: 60002
- Generate Code:**
 - Generate Code
 - Language: Microsoft .NET CSharp v11
 - Use new DataBank file path (recommended)
 - Zip: Filepath to write zipped code files to, e.g. \\LinearBase\CodeBase
 - Include Host Console Demonstration Code
 - Include docker-compose.yml File

Figure 19 - Completed Compose Form

Once all fields are complete click the Compose button to complete the transformation.

The progress bar bottom right provides feedback. The Compose dialogue reports success or failure, closes automatically, and clears the model.

On successful completion a separate console application opens similar to...



```
E:\LinearBase\Host\LinearBase.Host.exe
info: UseLinearBase[0]
      Read launch options from Command Line Arguments
info: UseLinearBase[0]
      Start_Action - Publish
info: UseLinearBase[0]
      Load Uri - http://localhost/WS9Z1OC7/WHR8SBU0
info: UseLinearBase[0]
      Pars_Port - 60002
info: UseLinearBase[0]
      Certificate Uri - (null)
info: UseLinearBase[0]
      Container_Ip - (null)
info: UseLinearBase[0]
      Publishing http://localhost/WS9Z1OC7/WHR8SBU0 using port 60002...
info: IBinder[0]
      OS: Microsoft Windows 10.0.22631
warn: IListener[0]
      Rpc service not listening! Port in use: 60002
info: ISession[0]
      Publish Program DZ7B7WM1 (QTEU6R49-WVE91LWD-HumanRes)
      WritePath: $usePars
      ReadPath: $usePars
info: UseLinearBase[0]
      Publish complete
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Production
info: Microsoft.Hosting.Lifetime[0]
```

Figure 20 - HumanRes Distributed Instance

The significance of this instance is it's using the Distributed Profile to initiate the configuration as can be seen by the line '**Load Uri - http://localhost/WS9Z1OC7/WHR8SBU0**'. This reads ProfileId **WHR8SBU0** from the active PARS instance describing where the relevant resources are located and how to initialize them.

Inspect Source Files

Assuming successful transformation, navigate to the path chosen in the Compose input dialogue, for example E:\LinearBase\Realms\Adventure.

The create and append transformation examples have created four new folders

- DZ7B7WM1 – HumanRes Databank folder
- Initializers – Initialization instruction folder
- LFWMKUZD – Sales Databank folder
- WS9Z1OC7 – System Databank folder

It is important Realm Databank folders remain under the relevant Realm folder, in this case 'Adventure'. It is equally important only a single version of the Realm exist at any one time although it can be moved anywhere. Moving a Realm requires a PARS instance restart to reflect the new location.

Expand the Sales Databank folder E:\LinearBase\Realms\Adventure\LFWMKUZD.

The process created a 'Schema' folder and a .data Databank file containing compressed, format-neutral binary files.

Expand the E:\LinearBase\Realms\Adventure\LFWMKUZD\Schema folder.

The Transform process created files

How-To Digitally Transform an Existing Database into a Databank Program and Code

- 7CWJZ91W-S6AG4AMU-Sales.zip – C# Program poco files unique to this transformation
- ExampleHostConsole.zip – the demonstration project
- Expand.ps1 – PowerShell script to unzip the Program files and a demonstration project
- Extensions.zip – Generic host code

Name	Date modified	Type	Size
7CWJZ91W-S6AG4AMU-Sales.zip	09/04/2024 15:25	Compressed (zipp...	44 KB
ExampleHostConsole.zip	09/04/2024 15:25	Compressed (zipp...	7 KB
Expand.ps1	09/04/2024 15:25	PowerShell Source...	1 KB
Extensions.zip	09/04/2024 15:25	Compressed (zipp...	2 KB

Figure 21 Schema files

Select Expand.ps1, right click and choose 'Run with PowerShell'. This generates another folder named ExampleHostConsole containing the zip file contents.

Expand the E:\LinearBase\Realms\Adventure\LFWMKUZD\Schema\ExampleHostConsole folder to inspect unzipped files...

Name	Date modified	Type	Size
Classes	10/04/2024 09:34	File folder	
Interfaces	10/04/2024 09:34	File folder	
AddServiceExtension.cs	09/04/2024 15:25	C# Source File	4 KB
Example.Host.csproj	09/04/2024 15:25	C# Project File	1 KB
ExampleHostedService.cs	09/04/2024 15:25	C# Source File	4 KB
ExampleReadWrite.cs	09/04/2024 15:25	C# Source File	4 KB
Program.cs	09/04/2024 15:25	C# Source File	2 KB

Figure 22 ExampleHostConsole Project

Expand the Classes folder to view auto-generated poco files corresponding to the Table and Field selections made earlier. The example generates AdventureWorks2017.Sales classes and interfaces.

Name	Date modified	Type	Size
CountryRegionCurrency.cs	09/04/2024 15:25	C# Source File	3 KB
CreditCard.cs	09/04/2024 15:25	C# Source File	3 KB
Currency.cs	09/04/2024 15:25	C# Source File	3 KB
CurrencyRate.cs	09/04/2024 15:25	C# Source File	3 KB
Customer.cs	09/04/2024 15:25	C# Source File	3 KB
PersonCreditCard.cs	09/04/2024 15:25	C# Source File	3 KB
SalesOrderDetail.cs	09/04/2024 15:25	C# Source File	3 KB
SalesOrderHeader.cs	09/04/2024 15:25	C# Source File	5 KB
SalesOrderHeaderSalesReason.cs	09/04/2024 15:25	C# Source File	3 KB
SalesPerson.cs	09/04/2024 15:25	C# Source File	4 KB
SalesPersonQuotaHistory.cs	09/04/2024 15:25	C# Source File	3 KB
SalesReason.cs	09/04/2024 15:25	C# Source File	3 KB
SalesTaxRate.cs	09/04/2024 15:25	C# Source File	3 KB
SalesTerritory.cs	09/04/2024 15:25	C# Source File	3 KB
SalesTerritoryHistory.cs	09/04/2024 15:25	C# Source File	3 KB
ShoppingCartItem.cs	09/04/2024 15:25	C# Source File	3 KB
SpecialOffer.cs	09/04/2024 15:25	C# Source File	3 KB
SpecialOfferProduct.cs	09/04/2024 15:25	C# Source File	3 KB
Store.cs	09/04/2024 15:25	C# Source File	3 KB

Figure 23 Classes

Profiles control everything in LinearBase. More on this in the next article ['How-To Host a Databank Program'](#).

Figure 1 Download Digital Transformation Program	2
Figure 2 Download Widows Host Program	4
Figure 3 LinearBase Transform	5
Figure 4 Select Data Instance Source	6
Figure 5 MSSQL Connection Criteria	6
Figure 6 Database List.....	7
Figure 7 Select a Database.....	7
Figure 8 Select the Model.....	8
Figure 9 Sales Model.....	8
Figure 10 Compose Input Dialogue	10
Figure 11 Principal Object.....	11
Figure 12 - Completed Compose Form.....	11
Figure 13 -Primary Active Realm Server (PARS) instance hosted in a console application.....	12
Figure 14 - Adventure Realm Databank Folders.....	13
Figure 15 - Sales Program Databank & Code.....	13
Figure 16 - Realm System Databanks	13
Figure 17 - AdventureWorks HumanResources	14
Figure 18 - Adventure Realm Option.....	14
Figure 19 - Completed Compose Form.....	15
Figure 20 - HumanRes Distributed Instance	16
Figure 21 Schema files	17
Figure 22 ExampleHostConsole Project.....	17
Figure 23 Classes.....	17