## 

## About SQL Migration

At this stage you should already be using the migrated .NET application with your original Btrieve database. Everything works great and it's time to migrate the data as well. Firefly provides a fast and easy way to migrate all your data from Btrieve to SQL server. In addition, the migrated .NET application is already designed to work with SQL server.

## Preparing for SQL migration

The migrated application has been designed to work in both Btrieve and in SQL with the same code base. This allows you to migrate your customers to SQL gradually, while you can continue developing the application and serve both Btrieve and SQL customers.

Firefly has invested a lot in making the application behave in SQL exactly the same as it was in Btrieve. This includes:

1. Optimal SQL Schema design

2. Object naming

3. Primary key violations detecting

4. Identical transactions and locking behavior (a Main issue when migrating in Magic)

5. Identical sorting behavior

6. Code that runs on both databases, configured by a feature flag.

7. Superior tuning tools

## What You Can Do To Prepare

There are special cases that are unique to your application, which may need some manual adjustment.

In general, anything in the application that is tied to Btrieve technology might need adjustment or at least be considered.

Please find below a list of usages that might exist in your application and need to be considered. If you think you have any of the following cases in your application, **please provide an example of how you use it from the user perspective.** Firefly is here to help and guide you through this process. We will examine each scenario and advise about the right solution for your specific application.

With the delivery of the SQL version of the application, we will show you how to apply the solution and make sure that it works in SQL using the same tests cases you provide.

## Referencing data tables as regular files

In Btrieve, all tables are files on disk that could be copied, renamed, deleted just like any other file. In SQL the tables are stored in SQL server database and are not accessed from the file system.

These are the most common functions that manipulate files in the application, which you should search for in the code:

|  |  |
| --- | --- |
| Function | What to search for |
| IORen | u.IORen( |
| IODel | u.IODel( |
| IOCopy | u.IOCopy( |
| IOExist | u.IOExist( |

#### Migrating from UniPaas may also include the following:

|  |  |
| --- | --- |
| Function | What to search for |
| FileRename | u.FileRename( |
| FileDelete | u.FileDelete( |
| FileCopy | u.FileCopy( |
| FileExist | u.FileExist( |

#### Searching for usages in Visual Studio

In Visual Studio, search for "u.IO" to find all of the above functions at once, as follows:

1. Open the full solution of the migrated application in Visual Studio

2. Press Ctrl+Shift+F

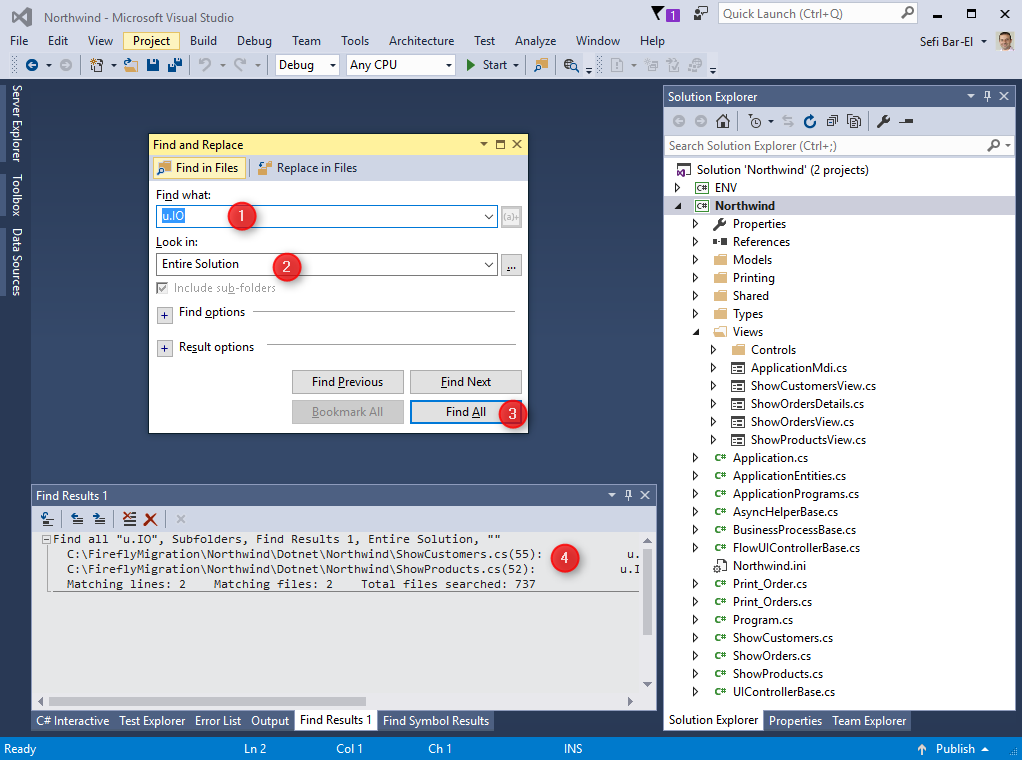
3. In Find What: enter 'u.IO'

4. In Look In: select 'Entire Solution'

5. Press 'Find All'

6. The result wll display in "Find Result #" window

7. Click on each result to go to the code



2017 03 09 16H52 36

#### Example for code that is harmless

protected override OnStart()  
{  
+ u.IODel("logo.bmp");  
}

#### Example for code that should be considered

protected override OnStart()  
{  
+ u.IODel("Orders.DAT");  
}

### OS Command

**Search for "ENV.Windows.OSCommand("** to find all the usages of the OSCommand and check if the command has anything to do with data files.

#### Example for code that is harmless

protected override OnStart()  
{  
+ ENV.Windows.OSCommand("excel \"" + fileName + "\"");  
}

#### 

#### Example for code that should be considered

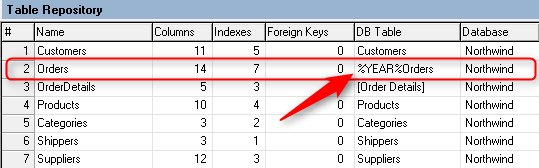
protected override OnStart()  
{  
+ ENV.Windows.OSCommand("cmd /C xcopy " + "Orders.DAT Orders2016.DAT");  
}

## Dynamic names of tables

The table names can be changed in runtime in many ways:

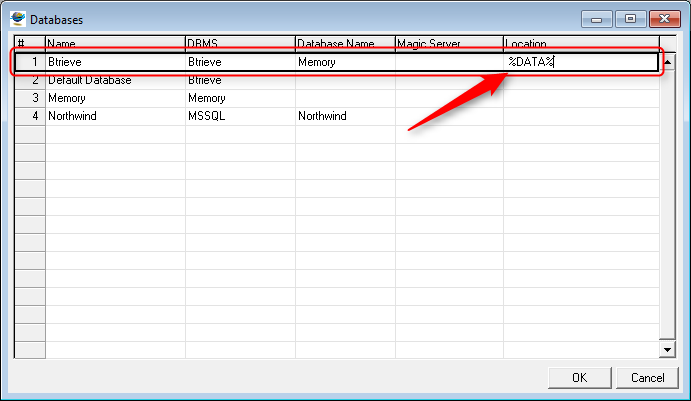
### Using Logical Names

Logical names can be used in two places:

1. Using logical names for table names in the tables repository (Shift + F2) and set the logical name in runtime using the IniPut() function or by using different ini file for different systems / users. 

**Look for table with Logical Names in their DB Name in the Table Repository**

1. In the dabases settings, using logical name as the location



2017 03 16 11H01 12

### Using Expresssions

Using expressions for table names in DB tables (Ctrl + D). For example, a program can change the table name so that it ends with the terminal number (using the Term() function).  
**To trace table name changes in Ctrl+D search for the text "EntityName = ".**

protected override void OnLoad()  
{  
+ Orders.EntityName = "Orders2016.DAT";  
}

## Pervasive (a.k.a Actian) PSQL and SQL Commands (Ctrl + Q)

If you have used PSQL to enable SQL commands (Direct SQL) to your tables via ODBC, the SQL code may need to be adjusted. **Search for “new DynamicSQL” to find all the usages in your application.**

void InitializeDataView()  
{  
+ sqlEntity = new DynamicSQLEntity(Northwind.Shared.DataSources.SQL,   
+@"SELECT ""OrderID"", Count(\*) as OrderItems  
FROM ""Order Details""   
WHERE ""CustomerID""=:1";  
}

### Changing Table Names

In Magic, each table has a name (caption) and a DB Table name. In Btrieve the DB Table name usually refer to the file name, which might be invalid SQL name. In this case, we use the captions as SQL names for the table and fix them so that there will be SQL style without any spaces for example "my table name" will be changed "MyTableName". This can affect existing SQL Commands that use different table names

### Changing Column Names

As with table names, Btrieve has no notion of Column names. The only thing that matters in the table structure is the columns position. In SQL, column names are important and are based on the columns captions of the original application. If your application had SQL commands which uses different columns names, they could be affected.

### Changing Column Storage

Most applications store dates as char(8) in the database and store times as numbers. In Pervasive PSQL, there could be non-standard ways to store data values for date, time or boolean columns. For example, dates might be stored as number instead of char(8).

## Multiple Entries in Table Repository for the same Data File

### Using the same table structure with different column names

In Btrieve, it is possible to use the same data file with different entries in the table repository (Shift + F2) that have the same structure and differ only by the column names.

### 

### Using the same table with different structures

In Btrieve it is possible to refer to the same data file with different table structure. For example, if a table has the following fields: 1. FirstName, alpha 50 2. LastName, alpha 50 3. Address, alpha 100 4. Phone, alpha 50

It is possible to create a second table structure for the same data file, which has the following structure: 1. FirestName, alpha 50 2. LastName, alpha 50  
3. **Filler, alpha 150**  
Both structures will open the file and show the data in it.

## Dll calls to Btrieve (W32mkde)

This is not very common but we have seen some applications that has calls to the Btrieve dll. **Search for "W32mkde"** to check if such calls exists in your application.

# Summary

The most important purpose of this article is to list the special cases that need to be considered when migrating the data from Btrieve to SQL. Now that you are aware of all them, please make sure you have use cases in your application that you can demonstrate to Firefly. Together we will decide on the best solution that you can apply to your specific application. Firefly will deliver a version with some adjustments to the data migration engine, based on these decisions. Once the application is ready for the migration, you can run the migration and start testing the application in SQL using the use cases that were prepared in advance. While testing the SQL version, you might find some performance differences. This is because Btrieve and SQL are very different technologies designed with different approach. Firefly can help in improving any performance issue that might be found in the SQL version. As always, please prepare use cases to demo these issues and consult with us.