

## **Migrate Once!**

### **NonStop S-series to Integrity NonStop platform, Enscribe to NonStop SQL in One (Easy) Step**

Migrating an application to the newest industry-leading hardware platform supporting the latest compilers and Open Source tools is an exciting prospect, but does it really provide any benefit if your database is left in a less-than-open format?

In this migration example, we use Carr Scott Software's Escort SQL product (Escort) to demonstrate how it is possible to migrate an Enscribe fileset on a NonStop S-series platform to a well-designed NonStop SQL database residing on the new Intel® Itanium® 2 Processor based HP Integrity NonStop platform—without even so much as recompiling a program. This evolutionary step results in an industry-leading platform along with the most reliable and scalable Open format database in the world.

### **Project Overview**

The goal of this migration example is to take an Enscribe employee database accessed by an application running on a NonStop S-series system, and migrate both application and database to a well-designed NonStop SQL table set running on the new Integrity NonStop platform. Note that the example below uses a COBOL language application for ease of explanation and reading, but Escort SQL supports conversion of all supported language applications, including TAL, PTAL, C, and all native compilers.

Escort SQL provides a runtime library that allows programs developed with Enscribe database calls to access a NonStop SQL/MP database without modifying or recompiling any of the application code.

This conversion process consists of a number of small steps as outlined below:

1. Install the Escort SQL user library into the application object files.
2. Design the new SQL database.
3. Create Escort SQL Maps to tell Escort how to translate the data as it is written to and read from the tables.
4. Create the tables in the Integrity NonStop machine and load them from the S-series machine.
5. Test the application and database on the Integrity NonStop system.
6. Complete the migration from the S-series to the new Integrity NonStop platform.

Using a real-life employee database as an example, we will walk through the life cycle of the migration process.

### **Migrating the Application**

Duplicating the application environment is the first step toward a complete migration. Depending on the size of the environment, it may be a good idea to use a product such as HP's NonStop AutoSYNC (NS AutoSYNC) to synchronize all source code, object, and configuration files to the new Integrity NonStop system. Having NS AutoSYNC

running will ensure that any changes to the current S-series system are replicated to the new environment during the conversion, testing, and burn-in period.

Before using the Escort SQL product, it is necessary to PREPARE the object files on both systems. This preparation is a simple, one-time process that enables the Escort intercept library technology without recompiling a program, and typically takes less than one second per binary object file. Prepared programs work equally well with a mixed environment of traditional Enscribe files as well as SQL tables, so it is recommended that all programs be run through the preparation process.

### Designing the New SQL Database

The first step in the database redesign process is to ensure you have an accurate DDL describing the file(s) to be converted. Once a dictionary has been created, the new table (or tables if a file is to be normalized) should be architected. This is a critical step and should be undertaken by persons who are familiar with the application and comfortable designing an SQL database.

When designing the new database, the following should be considered:

- Convert Proprietary Data Formats
  - Convert dates (e.g., YYMMDD) to SQL DATETIME
  - Expand compressed and encoded fields to readable format
  - Eliminate duplicate fields used to support Enscribe Alternate Keys
  - Appropriately size items for the real world including reasonable growth (i.e., expand numeric fields if likely to be exceeded in coming months/years; change fixed-length text fields to VARCHAR, etc.)
- Convert (Normalize) Non-Relational Record Formats
  - Multiple record definitions in one file
  - Redefined groups
  - Arrays (OCCURS)

### Creating the Escort Mapping Rules

The dictionary describing an existing file along with the design concept for the new table(s) provide two of the three pieces of input to the Escort SQL conversion process. The third piece is a set of commands that describe how data is to be cleansed or transformed in the transition from Enscribe to SQL and—a significant concept—**back** to Enscribe. It is important to remember that the end result of this exercise is to create tables **plus** a set of rules that the Escort runtime library can use to allow existing applications to access the new SQL tables, through Enscribe API calls with no programming changes. Obviously, new SQL-based programs can easily and without restriction access the new SQL tables, but the “trick” that Escort SQL provides is allowing Enscribe programs to read and write directly to/from the NS SQL tables as well, without requiring any changes to existing programs (no need to even recompile).

In our example, we migrate the employee file EMP and the department file DEPT. An abbreviated DDL source for these two files is shown below.

```

Record EMP.
File is "EMP"                                Key-sequenced.
?Talbound 1
  02 EMP-KEY.
    03 EMP-NUM-KEY.
      04 EMP-NUM                               Pic "9(6)".
  02 EMP-WORK-NAME.
    03 LAST-NAME                               Pic "X(15)".
    03 NAME-LAST Redefines LAST-NAME.
      04 LI                                     Pic "X(1)".
      04 FILLER                                 Pic "X(14)".
    03 FIRST-NAME                               Pic "X(10)".
    03 NAME-FIRST Redefines FIRST-NAME.
      04 FI                                     Pic "X(1)".
      04 FILLER                                 Pic "X(9)".
    03 MI                                       Pic "X(1)".
... (deleted for clarity)
  02 EMP-FILLER                               Pic "X(37)".
  66 FULL-NAME                                Renames EMP-WORK-NAME.
Key is EMP-KEY Duplicates not allowed.
Key "EA" is NAME-PART.
Key "EC" is EMP-CORRESPONDENT-NAME.
Key "ED" is EMP-DEPT-NUM-KEY.
... deleted for clarity
Key "EP" is EMP-WORK-PHONE.
Key "EW" is EMP-WORK-NAME.
End

```

**Figure 1 – EMP DDL source definition**

```

Record DEPT.
File is "DEPT"                                Key-sequenced.
? 02 DEPT-NUM                                 Pic "9(4)".
  02 DEPT-COMPANY-NUM                          Pic "999".
  02 DEPT-FUNC-CD                              Pic "X(2)".
  02 DEPT-DISCIPLINE-CD                       Pic "X(3)".
  02 DEPT-MANAGER-NUM                          Pic "9(6)".
  02 DEPT-MAILSTATION-ID                      Pic "9(6)".
  02 DEPT-MAIL-FILLER                         Pic "X(2)".
  02 DEPT-NAME                                 Pic "X(25)".
  02 DEPT-EFFECTIVE-DATE.
    03 YY                                       Pic "99".
    03 MM                                       Pic "99".
    03 DD                                       Pic "99".
  02 DEPT-OBSOLETE-DATE.
    03 YY                                       Pic "99".
    03 MM                                       Pic "99".
    03 DD                                       Pic "99".
  02 DATE-FISCAL Redefines DEPT-OBSOLETE-DATE.
    03 YYYYMM                                 Pic "9(4)".
    03 FILLER                                 Pic "X(2)".
... (deleted for clarity)
  02 FILLER                                    Pic "X(6)".

Key is DEPT-NUM Duplicates not allowed.
Key "DC" is DEPT-COMPANY-NUM.
Key "DT" is DEPT-TYPE-CD.
End

```

**Figure 2 – DEPT DDL source definition**

Unlike Enscribe, SQL supports data types, so there are a few changes we would like to see in our migrated data as well as some naming cleanup we would like to perform. In particular, we would like to convert the YYMMDD dates in the DEPT file to an SQL DATETIME data type; and in the interest of saving space, we will convert a number of text fields to type VARCHAR. For improved readability, we will also drop the old-style data item name prefixes EMP and DEPT.

Below is an example of the MAP command files that will be used by Escort for handling data conversion between the Enscribe I/O calls in the application and the new SQL database. Note the following entries in particular:

**dictionary migrdict;**—Points to the compiled Enscribe DDL Dictionary, which is one of the inputs Escort requires along with MAP commands and a file name to produce a set of bidirectional data mapping rules and create the resulting tables.

**catalog migrcat;**—Identifies the SQL catalog where the new tables will be registered.

**convert record**—Instructs Escort to convert the specified DDL record into a set of map rules and a new SQL table(s).

**file is**—Points to the original Enscribe file where Escort will pick up useful information such as extent size and partition settings to be used when creating the corresponding SQL table(s).

**delprefix**—This mapping command causes the new SQL column to be created without the redundant prefixes EMP and DEPT.

**trim-spaces**—The mapping command that converts a PIC X field to a type VARCHAR and removes trailing spaces.

**migr-yyymmdd-68**—Converts a PIC X(6) field of the format YYMMDD into an SQL DATETIME data type and adds the century based on the year being prior to 68. Any number of date and time conversions can be handled by using the variety of built-in or custom data conversion routines available.

**redefine**—While this has a number of different uses, in this case it is used to convert a field to FILLER, which is then automatically deleted from the SQL table because the reserving of filler space is not needed in SQL tables.

```

-- MAP command script for EMP
dictionary migrdict;
catalog migrcat;
convert record emp to table migrsql.emp
file is \nskned.$prpc.migrens.emp
rename (filler AS info)
delprefix emp_
trim-spaces (emp-work-name.last-name
             ,emp-work-name.first-name
             ,emp-filler
             )
;

-- MAP command script for DEPT
dict migrdict;
catalog migrcat;
convert record dept to table migrsql.dept
file is \nskned.$prpc.migrens.dept

migr-yymmdd-68 (dept-effective-date
               ,dept-obsolete-date
               )
delprefix dept_
redefine (dept-mail-filler FILLER)
trim-spaces (dept-name)
;

```

**Figure 3 – Sample Escort SQL MAP script**

While the files EMP and DEPT in this simple example contain only a single record type and no arrays, it is often common practice to store multiple record types with a single Enscribe file. When designing the new SQL database, the best practice would be to normalize the variant data into individual tables. Escort SQL provides a complete set of tools for generating these normalized tables; the application can then use them seamlessly with no programming changes.

**Creating and Loading the New SQL Database**

Once the Escort MAP command files have been built, they are used as input to the Escort Command Interpreter (CI). The result is the creation of sets of SQL tables and mapping rules that are automatically stored in the Escort configuration database.

Looking at the Enscribe version of our database, we see the following:

	CODE	EOF	LAST MODIF	OWNER	RWEP		
\$PRPC.MIGRENS							
DEPT		753664	26Aug2005 14:13	170,46	NUNU	KA	
DEPTO		929792	24Aug2005 12:35	170,46	NUNU	K	
EMP	2000	1708032	26Aug2005 13:35	170,46	NUNU	KA	
EMPO	2000	2535424	24Aug2005 11:08	170,46	NUNU	K	

**Figure 4 – Enscribe database files**

After running the MAP commands through the Escort CI and using the Escort LOAD command to transfer the data from the Enscribe files to the new NS SQL tables, we see the following on the Integrity NonStop system:

	CODE	EOF	LAST	MODIF	OWNER	RWEP	TYPE	REC
BL								
\$DATA5.MIGRSQL								
DEPT	A+	552960	26Aug2005	12:21	170,46	NUNU	K	
DEPTDC	A+	94208	26Aug2005	12:21	170,46	NUNU	K	In
DEPTDT	A+	94208	26Aug2005	12:21	170,46	NUNU	K	In
EMP	2000A+	1458176	26Aug2005	12:21	170,46	NUNU	K	Ta
EMPEA	2000A+	229376	26Aug2005	12:21	170,46	NUNU	K	In
EMPEC	2000A+	655360	26Aug2005	12:21	170,46	NUNU	K	In
EMPED	2000A+	163840	26Aug2005	12:21	170,46	NUNU	K	In
EMPEF	2000A+	233472	26Aug2005	12:21	170,46	NUNU	K	In
EMPEJ	2000A+	188416	26Aug2005	12:21	170,46	NUNU	K	In
EMPEL	2000A+	188416	26Aug2005	12:21	170,46	NUNU	K	In
EMPEM	2000A+	188416	26Aug2005	12:21	170,46	NUNU	K	In
EMPEN	2000A+	241664	26Aug2005	12:21	170,46	NUNU	K	In
EMPEP	2000A+	380928	26Aug2005	12:22	170,46	NUNU	K	In
EMPEW	2000A+	352256	26Aug2005	12:22	170,46	NUNU	K	In

Figure 5 – SQL database tables

Note that an SQL INDEX has been automatically created for each Enscribe ALTERNATE KEY on the original files. For ease of management, we chose to add the ALT KEY SPECIFIER as a suffix to each of the indices.

After loading the new Integrity NonStop system–based SQL tables, we ran a QA test cycle of our application to confirm that it worked flawlessly on the new Integrity NonStop machine and that our new SQL tables were transparently accessible by our unchanged application.

### Migration: The Final Step

After completing a QA cycle and confirming that everything works as it should, we need to reload our Integrity NonStop system–based SQL tables with the latest data from our NonStop S-series production application. In order to minimize the outage window for the production system, we decided to utilize the Escort Journaling product.

Escort Journaling (Journaling) allows us to load from the production Enscribe files in SHARED mode while the application is still running and being accessed by the user community. While the load is in process, Journaling keeps track of all updates to the database in a separate Journal fileset resident on the S-series system. After the load is complete, Journaling will apply all updates to the new SQL tables and continue to apply new changes until the actual cutover of production work to the Integrity NonStop platform.

A big advantage of this method of data migration is that the S-series production Enscribe files and the Integrity NonStop SQL database can be kept completely synchronized until

the migration team is prepared to shut down production, move communication lines, and bring up the production processing on the Integrity NonStop platform.

Once the cutover decision is made, the S-series production application is stopped and the Escort Journaling updater processes are allowed to apply the last updates. After all communication is redirected to the Integrity NonStop system, the new production application is started using the newly loaded NonStop SQL database.

### **Summary**

In our migration example, we chose to take the full evolutionary step because the tools we used provided proof-points where we could easily verify the stability of our application. For some, migrating an application to a new hardware platform is a big task and may not necessarily be the ideal time to also migrate your database. However, moving to the new open Integrity NonStop platform certainly is the time to be considering how to evolve your database to the new Open standards.



<http://www.CarrScott.com>