



## ***Getting Started with tRelational and DPS***

### **ADABAS-to-RDBMS Data Transfer**

This document is applicable to the tRelational and Data Propagation System (DPS) product set from Treehouse Software, Inc.

Comments pertaining to this document are encouraged. Please direct all comments to:

**Treehouse Software, Inc.**  
409 Broad St., Suite 140  
Sewickley, PA 15143  
Phone: (412) 741-1677  
Fax: (412) 741-7245  
E-mail: [tsi@treehouse.com](mailto:tsi@treehouse.com)  
<http://www.treehouse.com>

Worldwide marketing of tRelational and DPS and other Treehouse products is handled through the Sewickley office.

Reproduction of any portion of this document without the written consent of Treehouse Software, Inc. is prohibited.

Copyright November 2003, by Treehouse Software, Inc. in Sewickley, Pennsylvania.

This page intentionally left blank.

## Table of Contents

<b>GETTING STARTED WITH TRELATIONAL AND DPS .....</b>	<b>4</b>
1 Preparing To Install tRelational And DPS.....	4
1.1 Outline Of Installation And Configuration Procedure .....	4
2 Getting Started With tRelational.....	5
2.1 Enhanced Modeling Using tRelationalPC, TRA, And TRASTATUS .....	6
2.2 Complex Modeling And Mapping .....	7
2.3 Instantiating The Target RDBMS .....	7
2.4 Understanding DPS Parameter Files.....	7
3 Getting Started With DPS Materialization.....	9
4 Getting Started With DPS Propagation.....	10
5 Understanding Transaction Volume And Characteristics .....	10

## Getting Started with tRelational and DPS

### 1. Preparing to Install tRelational and DPS

Before embarking on installation, you should study the following documents in order to understand the installation and operational requirements, prerequisites, supported platforms, and product versions:

- *tRelational Release Notes*
- *tRelationalPC Release Notes*
- *tRelational Manual* (particularly the sections on Installation, Operations, Treehouse Remote Access, and tRelationalPC Installation Instructions)
- *DPS Release Notes*
- *DPS Manual* (particularly the sections on Operations, Installation, and DPS Supplemental Utilities)

You should identify the ADABAS files that will be used as the source of data, as well as the target RDBMS(s) and any existing schemata that are to be modeled as targets.

#### 1.1 Outline of Installation and Configuration Procedure

Each major component has a separate installation procedure that is independent of the installation of the other major components. As such, the sequence of component installation is not important, and component installation activities may proceed in parallel. However, the following sequence will facilitate timely analysis and decision-making:

Procedure	Document Reference(s)
1. Install tRelational (mainframe)	<i>tRelational Manual</i>
2. Use tRelational to implement and analyze selected source ADABAS files	<i>tRelational Manual</i>
3. Use tRelational to generate the preliminary data model	<i>tRelational Manual</i>
4. Use tRelational to generate DPS parameters for the preliminary data model for use with TRANSACT	<i>tRelational Manual</i>
5. Install Treehouse Remote Access	<i>tRelational Manual</i>
6. Install tRelationalPC	<i>tRelational Manual</i>
7. (optional) Use tRelationalPC to import existing RDBMS schema(ta)	<i>tRelationalPC Help facility</i>
8. Use tRelational/tRelationalPC to develop/map to RDBMS schema(ta)	<i>tRelational Manual</i> and <i>tRelationalPC Help facility</i>
9. Use tRelational to generate RDBMS DDL and instantiate the RDBMS schema(ta) physically	<i>tRelational Manual</i>
10. Use tRelational to generate DPS parameters for revised data model	<i>tRelational Manual</i> and <i>DPS Manual</i>
11. Install DPS (including the DPSSPLIT utility)	<i>DPS Manual</i>
12. Configure and test DPS materialization	<i>DPS Manual</i>
13. Configure and test DPS propagation	<i>DPS Manual</i>
14. Use DPS TRANSACT utility to gain insight into PLOG transaction volume, transaction types, and arrival rates over time for selected ADABAS files	<i>DPS Manual</i>

## 2. Getting Started with tRelational

tRelational is principally a mainframe-based NATURAL application providing a range of analysis, modeling, and mapping functions in both online and batch modes. Installation of the basic mainframe tRelational facilities enables commencement of the process of identification of ADABAS data sources and resolution of issues pertaining to them.

### File Implementation

After tRelational installation, the first step in building tRelational/DPS applications is to execute (online or in batch) tRelational file implementation for each ADABAS file (or userview, if desired) to be used as a data source. This retrieves file structure information from PREDICT data definition entries (DDEs) and the ADABAS Field Definition Table (FDT) and creates an "implemented file" entry in the TRE-DICT part of the tRelational repository. Discrepancies between PREDICT and the FDT are flagged in the Implemented Field Summary and should be investigated and resolved in order to ensure accurate representation of the data structures.

### File Analysis

tRelational file analysis should be run (typically in batch) for each implemented file. The results of file analysis are important not only to guide the RDBMS modeling and mapping, but to uncover potential data quality issues—issues that may have disastrous effects if they are not identified and resolved early.

Note that, by default, file analysis processes all records in physical sequence. However, subsets of records may be analyzed by specifying a descriptor (and optionally a descriptor value range). This is particularly useful when multiple logical "record types" exist within a physical ADABAS file. Only the statistics for the latest-run file analysis are available from the tRelational repository.

The following table illustrates the value of these analyses:

Analysis	Primary purpose(s)	Potential data quality issues that may be identified
Repeating Field	<ul style="list-style-type: none"> <li>Support normalization vs. denormalization modeling decisions</li> <li>Estimate number of rows in normalized "child tables"</li> </ul>	<ul style="list-style-type: none"> <li>Unused MU/PE fields</li> <li>MU/PE fields that are populated differently from expectations</li> </ul>
Alphanumeric	<ul style="list-style-type: none"> <li>Support RDBMS datatype (CHAR vs. VARCHAR or other) and length selection in modeling</li> </ul>	<ul style="list-style-type: none"> <li>Unused fields</li> <li>Fields where maximum length exceeds defined length</li> </ul>
Descriptor	<ul style="list-style-type: none"> <li>Support selection of primary key for RDBMS table(s)</li> </ul>	<ul style="list-style-type: none"> <li>Duplicate values in descriptors and superdescriptors assumed to be unique</li> <li>Records lacking a value in a required descriptor or superdescriptor component</li> </ul>

### **Creating a Data Model**

Creation of a first-cut data model (which may be thought of as a DPS “unit of work”—a collection of RDBMS tables related to ADABAS sources) enables the use of TRANSACT, as well as initial testing of the materialization and propagation processing. The most expedient way to create this data model is through the use of tRelational’s auto-generation function in the RDBMS Modeling and Mapping Subsystem. Content can be added to the data model on a file-by-file basis and a fully-functional data model can be completed in minutes. All resulting metadata is stored in the TRE-TRANS portion of the tRelational repository.

Before using tRelational’s batch GENDPS function to create the parameter set (“DDPARM” files) required by DPS processes (including propagation extraction required by TRANSACT), the ADABAS Version value for the DBID(s) used in the data model should be reviewed in the DPS Parameters Subsystem and modified if necessary. Filters and record processing limits may be applied if desired.

#### **2.1 Enhanced Modeling using tRelationalPC, TRA and TRASatus**

tRelationalPC, a native Windows application, is an alternative to the mainframe 3270- and batch-oriented interface. tRelationalPC offers a point-and-click, drag-and-drop environment to accomplish the same modeling and mapping functions as can be done with tRelational on the mainframe.

tRelationalPC maintains a local (PC file) representation of implemented files (.imp) and data models (.tre), but communicates with mainframe tRelational to exchange data and invoke mainframe tRelational functions. This communication, through designated TCP/IP ports, is effected using Treehouse Remote Access (TRA).

TRA is a mainframe Remote Procedure Call (RPC) server, executing as a long-running batch job or started task, that listens for requests from clients on an identified “data” port and executes tRelational functions via an attached NATURAL subtask, passing data to and from the client via TCP/IP. tRelationalPC issues requests and receives data via the same port on the PC side.

TRASatus is another native Windows application that may be used to monitor TRA availability and activities from the PC via the designated “status” port.

Once TRA is installed and configured and TRASatus and tRelationalPC are installed on one or more client PCs, implemented files may be downloaded using the “Server/Load Implemented Files” function and data models using the “Server/Remote Open” function. Modeling activities (including auto-generation) may then be undertaken on the PC side. Ultimately, new or revised data models must be uploaded to the mainframe tRelational repository (via the “Server/Remote Save or Server/Remote Save As functions) to make them available for GENDDL, GENDPS, and DPS processing.

tRelationalPC adds one additional significant feature that cannot be accomplished using mainframe tRelational: importation of an existing RDBMS schema. If ODBC connectivity is available from the client PC to the RDBMS instance, and authorization to read the RDBMS catalog has been granted to the user, tRelationalPC can list and capture selected table structure information, including columns definitions, constraints, and physical DDL parameters. The imported schema can then be mapped to ADABAS sources as usual. This feature avoids time-consuming and error-prone manual duplication of table structure details when an existing RDBMS schema is available.

## **2.2 Complex Modeling and Mapping**

Developing tRelational data models is often an iterative process. Fortunately, the high-productivity tRelational environment and high-performance DPS engine facilitate iterative use.

While tRelational's auto-generation feature produces a fully-functional data model in a very short period of time, such a data model is intended to serve as a starting point, since most applications will require significant customization. New columns may need to be added, columns may be unnecessary and can be deleted, and various transformations may be needed in order to appropriately populate the RDBMS with ADABAS-derived data. It is important to have gathered the data source and transformation requirements prior to refining the first-cut data model.

Designing an effective data model requires an understanding of the source data, the target data, and relationships between them. Very complex mapping relationships can be defined with tRelational and realized by DPS. Examples of mapping relationships that can easily be implemented include:

- Denormalization of MU/PE occurrences into discrete columns
- Concatenation of MU/PE occurrences into a single column
- Normalization of MU/PE occurrences into "child tables" with foreign-key relationships to the "parent" table
- Concatenation of multiple fields into a single column
- Mapping substrings of fields
- Converting ADABAS-specific datatype values to RDBMS-specific datatype values
- Scrubbing unprintable characters
- Decoding of encoded values

Some of these transformations require the application of External Transformation Routines (ETRs), which are central to the transformational capabilities of DPS. A number of ETRs are provided to meet common transformation requirements, and user-written routines can be developed to meet site-specific requirements. The modeler needs to have a good understanding of tRelational modeling techniques and available transformations in order to optimize the data model.

## **2.3 Instantiating the Target RDBMS**

tRelational provides facilities to capture a large number of RDBMS-specific physical data definition language (DDL) parameters. These should be reviewed and values assigned prior to generating the RDBMS-native DDL for the revised data model using the batch tRelational GENDDL function.

The output file(s) from GENDDL can be transferred to the RDBMS host and executed using the RDBMS's interpretive SQL processing utility (e.g., ORACLE SQL\*Plus).

Note that it is not necessary to run GENDDL if the RDBMS schema has been previously defined and instantiated. However, it is a good practice to ensure that any physical DDL parameters being used in the instantiated schema are reflected in tRelational.

## **2.4 Understanding DPS Parameter Files**

The DPS parameter files are the "instructions" to DPS describing the ADABAS sources, the RDBMS targets, and the transformations required. tRelational generates parameter file sets based on the contents of the data model. These file sets include the table, column, and mapping definitions specified in the RDBMS Modeling and Mapping Subsystem. Additionally, tRelational provides the DPS Parameters Subsystem for specification of parameter values to adjust and tune DPS processing.

Once the data model's schema, mappings, and associated DPS parameter specifications are completed, tRelational's batch GENDPS function can be used to create the DPS parameter files. Specific references are made to ADABAS DBID and FNR within the parameter files, since it is critical that DPS be able to properly interpret the data source (which is in a compressed format). It may be useful to employ the Translate DBID (TD) and/or Translate File (TF) functions in conjunction with GENDPS to facilitate testing with specific data sources.

GENDPS creates four parameter files:

The *DDPARM* file contains the File Parameters that describe the source ADABAS database(s), field(s), and file(s) at a physical level.

The *DDPARME* file contains Extraction Parameters that name the files to be included in a given DPS process and file-specific filter criteria that apply.

The *DDPARML* file contains Logical Transformation (LTR) Parameters that describe how the ADABAS sources are to be transformed and data created for the RDBMS targets.

The *DDPARMA* file, which applies to the optional-use ADABAS PLOG Consolidation (APC) facility. This feature may be used to reduce the volume of SQL statements produced by DPS for applications with suitable characteristics. Refer to the *DPS Manual* for a full description of APC.

Any change to data model specifications (or to the associated implemented files) generally necessitates regeneration of the DDPARMs, and certain changes may necessitate rematerialization and reloading of the target RDBMS tables.

### **3. Getting Started with DPS Materialization**

DPS runs as a set of batch jobs using ADABAS input and creating RDBMS output.

Materialization provides a first-time initialization and point-in-time full synchronization of ADABAS with the RDBMS. Most applications require that a body of data exists in the RDBMS prior to propagation of changes. For example, SQL UPDATE statements based on ADABAS record updates processed in propagation will fail if the table has not been pre-populated with the appropriate corresponding rows.

The source of data for materialization is a backup of the ADABAS database(s) or the pertinent file(s), commonly known as an ADASAV backup. For integrity purposes, this backup needs to be done while the database is down or while the files being backed up are locked against updating.

Template JCL is provided with DPS for various DPS processes. These can be easily customized to create site-specific jobs. Multiple materialization examples are provided to demonstrate different phases of DPS operation (extraction and transformation), as well as run types.

A materialization run can produce a large volume of output data, so adequate disk space and/or tape devices should be available.

The principal outputs from materialization are the RDBMS control file (DDCTL) and the data file (DDMAT). The contents of these files span all target tables, so it is necessary to split the contents on table-by-table basis. This is the function of the DPSSPLIT utility, which is provided in mainframe, UNIX, and Windows implementations. The choice of platform for executing DPSSPLIT is at the user's discretion. If the target RDBMS runs on a mainframe host, then mainframe DPSSPLIT is the logical choice. Otherwise, the files may be split on the mainframe and transferred individually to the RDBMS host, or the pre-split files may be transferred to the RDBMS host and split there.

Once the control files and data files have been split into table-specific files, they can be used immediately by the native RDBMS load utility (e.g., ORACLE SQL\*Loader) to populate the target on a table-by-table basis.

It is important to test materialization and load the results into the RDBMS early in the development of the application, in order to assess parameter effectiveness (e.g., filters, DPS Significance assignments), identify and respond to data quality issues that may arise and understand the materialization time requirements. Testing logistics may be greatly facilitated through the use of record processing limits and/or filters.

#### **4. Getting Started with DPS Propagation**

PLOG data must be considered as a continuous stream that begins at the materialization point (when the ADASAV backup is taken). However, the slice of this stream that is processed through a given propagation run may contain in-flight ADABAS transactions whose disposition is uncertain until the End Transaction (ET) occurs in the stream or a Backout Transaction (BT) indication is found. DPS automatically cycles incomplete transactions from a given run through its “restart” datasets so that they may be input to the next run. Any backed-out transactions are automatically discarded and produce no output, while output for committed (ETed) transactions is produced only when the ET is encountered.

The very first run of DPS propagation following materialization requires either that the input “restart” transaction dataset is empty or that the IGNORE RESTART parameter is coded in DDPARME.

As with materialization, template JCL is provided for various propagation functions. Applications will generally use the standard PARM='PROPAGATION' to combine extraction with transformation and produce the standard SQL output.

The default output from propagation is a set of SQL statements (INSERT/UPDATE/DELETE) that reflect, per the transformation specifications described in the DPS parameters, the changes required in the RDBMS to resynchronize it with ADABAS. SQL COMMITs are included and by default have the same scope as the corresponding ADABAS transactions. However, this behavior is tunable in the DPS parameters.

It is important to “unit test” DPS propagation on an end-to-end basis (i.e., including applying the SQL to the RDBMS) in order to assess overall throughput. Ordinary PLOG archives may be used for this, and filters and/or record processing limits may facilitate testing logistics.

#### **5. Understanding Transaction Volume and Characteristics**

The volume of PLOG transactions and activity peaks at points in time are important in configuring for optimum throughput and problem avoidance.

Typically, propagation needs to involve only a subset of production operational data originating in ADABAS. In order to limit the scope of analysis appropriately, the DPS TRANSACT utility takes input from a DPS propagation extract (i.e., transactions that have been extracted from source PLOGs based on file and filter selection criteria). TRANSACT enumerates these transactions by ADABAS database ID (DBID), file number (FNR), and update type (insert, update, delete).

Each of these utilities may be used to create spreadsheet-ready outputs for detailed analysis. The results of such analysis may be used to guide various configuration and planning decisions, including but not limited to:

- DPS SQLOUT/DDIMAGE dataset space
- DPS RESTART dataset space
- RDBMS tablespace and growth physical DDL parameters
- RDBMS undo/redo log size
- Network bandwidth requirements between mainframe and RDBMS host

This page intentionally left blank.



Treehouse Software, Inc.

409 Broad Street, Suite 140  
Sewickley, PA 15143  
Phone: (412) 741-1677  
Fax: (412) 741-7245  
E-mail: [tsi@treehouse.com](mailto:tsi@treehouse.com)  
[www.treehouse.com](http://www.treehouse.com)