Oracle® GoldenGate for Base24

T24 Tokenized Data Supplemental Guide

Version 3.0

Oracle® GoldenGate for Base24 T24 Tokenized Data Supplemental Guide, version 3.0

Contents

CHAPTER 1

Introducing T24

This chapter introduces T24, a supplemental module that facilitates data replication between BASE24 and other databases/platforms. Topics include:

Contents

Overview

T24 moves unstructured data from BASE24 into the structured format of your choice. Historically, data replication to structured targets required custom coding. In contrast, T24 reorganizes and reformats transaction log (TLF and PTLF) tokens for all transaction types into a configurable order.

T24 also reorganizes and reformats PBF and CAF segments into a defined flat file structure. Replicat then uses this layout to propagate data into any supported database.

Components

T24 has the following components that run on your source system:

- A user exit, which is a program extension to the GoldenGate Extract process
- The T24 DDL token definitions
- The DDL layout of the output transaction log
- The Extract parameter file column mapping of the token header fields

Understanding tokenized data

Implementing T24 requires that you define the target structure that will receive tokenized data from BASE24. Tokenized data is a record that has a fixed structure data area and a dynamic token area, which defines the record. In short, tokenized data is metadata that defines the record and its use.

For example, each record can be defined as a record type through the use of tokens. So, a data string can have a token that identifies it as a customer, administrative, or exception record. Tokens can be customized based on the type of transaction logged, so that withdrawal tokens differ from deposit tokens. Ultimately, this means that each transaction can have a unique set of tokens, whose size differs from record to record.

Historically, it has been difficult to capture such variable data and write it in a structured format for easy querying. When you implement T24, you are able to address this issue in the following ways:

- Design a data structure that works with your record types
- Define which tokens you want to capture
- Determine the length each token value can be, and
- Specify the order you want data written to GoldenGate

Understanding your tokenized data becomes critical when you must define your T24 Structured Record Definition, a key step in your implementation.

T24 processing

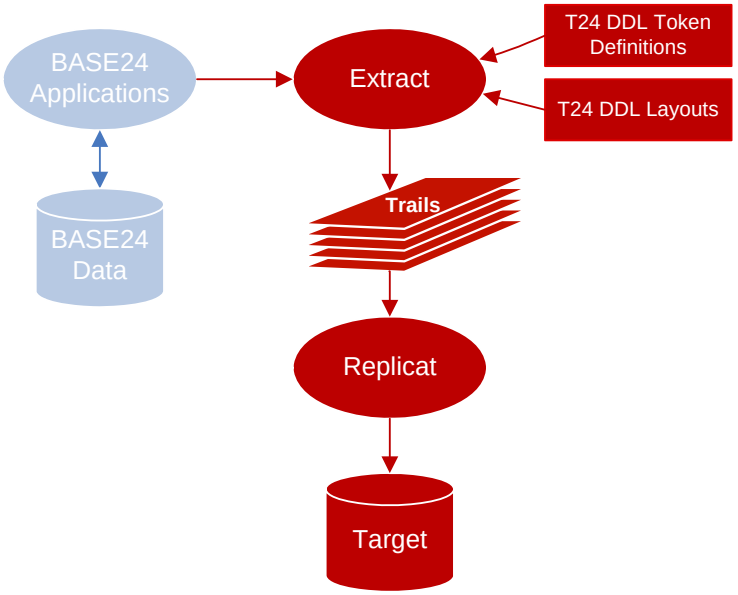
The following table lists the components needed for T24 processing.

Component	Description
BASE24	Application processing ATM and POS records
BASE24 Data	Source data for GoldenGate's T24 processing
Extract Parameter File	Contains parameters that map token headers to column headings and control Extract behavior.
Extract/Extract Trail	Reads BASE24 records and writes them to the extract trail.
T24 DDL Token Definitions	Defines the tokens you wish to capture
T24 DDL Layout	Maps the tokens you wish to capture to a specific data structure.
Replicat Parameter File	Contains parameters that control Replicat behavior.
Target Database	Receives restructured BASE24 data.

Component	Description
T24 User Exit	Program extension to capture, parse, and organize tokenized data from the source system.

To understand how T24 affects your BASE24 and GoldenGate implementation, you must understand its logical dataflow, illustrated in Figure 1.

Figure 1 T24 dataflow



T24 implementation overview

Before you begin installing T24 code, it is important to understand and plan each step of your implementation. This section outlines the basic T24 implementation project; customize it to fit your own business needs.

- Planning for T24
  - Analyze your source data
  - Decide which data you wish to capture, and its order
- Installing and Configuring T24
  - Install T24 code to appropriate directories
  - Determine token area requirements
  - Edit T24 DDL files
  - Create T24 template files
  - Generate source definitions
  - Generate the target table schema
  - Bind the T24 user exits
- Configuring Change Capture
  - Add and configure Extract and the extract trail
  - Determine your capture technique
  - Configure capture checkpointing
  - Create your Extract parameter file
- Configure Data Delivery
  - Create Replicat parameter file
  - Configure Replicat checkpointing

CHAPTER 2

Preparing for T24

Before you can install and configure T24, you must make some decisions regarding your source data. These considerations are discussed in the following topics:

Contents

Analyzing source data

Before you can prepare your T24 structured record definition, you must understand how your business uses its tokenized data areas in the types of transactions you log. This section reviews how to read your transaction records and identify token types, in preparation for selecting the tokens you want to replicate.

In a BASE24 transaction log record, a fixed-length data string is followed by variable length data tokens (see Figure 2).

Figure 2 TLF record structure

Each header token contains an “eye-catcher”, represented by 2620, the hexidecimal equivalent of an ampersand (&) followed by a blank space. Each subsequent data token, represented in Figure 3, contains:

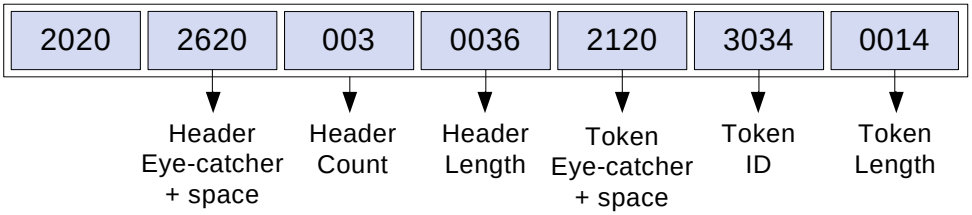
- An “eye-catcher” character, represented by 2120, the hexidecimal equivalent of an exclamation point (!) followed by a blank space.
- Token Identifier
- Token Length
- Token Data

Figure 3 TLF data token structure

Identifying your tokens

Each of your tokens will have the components discussed above: header tokens, eye-catchers, ids, lengths, and values. The following example (Figure 4) shows the last 100 bytes of a typical PTLF record, which is part of a POS type transaction.

Figure 4 Sample PTLF tokenized record



In Figure 5, an example of a data log shows the distribution of tokens, and the types of characters you must identify. Header tokens, eye-catchers with spaces, token IDs, and token lengths are in bold.

Figure 5 Sample data log

```
$DATA PRO1PTLF 13> FUP COPY POyymmdd,, H, Share, Count 5
```

180:	2020	2020	2020	2020	2020	2020	2020	2020		
188:	2020	**2620**	0003	0036	**2120**	**3034**	**0014**	2020	&..6!	04..
190:	2020	2020	2020	2020	2020	2020	2020	2020		
198:	5920	**2120**	**4331**	**0010**	5031	425E	4745	4E53	Y	!
1A0:	494D	5E30	3120	2020					IM^01	

Sample header tokens

The header token identifies the start of the token area in the Transaction Log record; its definition is located in the DDLBATKN file.

```
DEFINITION HEADER-TKN
   02 EYE-CATCHER       PIC X.
   02 USER-FLD1         PIC X.
   02 CNT               TYPE BINARY 16.
   02 LGTH              TYPE BINARY 16.
END
```

Sample data token header

The following is an example of the data token header definition:

```
DEFINITION TKN-HEADER
   02 EYE-CATCHER       PIC X.
   02 USER-FLD1         PIC X.
   02 TOKEN-ID          PIC X(2).
   02 LGTH              TYPE BINARY 16.
END
```

Sample data token definitions

The data token data definition contains all the data fields in the token. The data token definitions are found in the following BASE24 files:

- **DDLBATKN** - BASE tokens
- **DDLATTKN** - ATM tokens
- **DDLPSTKN** - POS tokens

The following two examples are for the BASE24 ATM PIN Non-Currency Dispense token (BASE24 token id A5) and the PIN Change token (Base24 token id 06). Both tokens are defined in the DDLATTKN file DEFINITION NCD-TKN.

```
* The number of items being purchased.

   02 ITEM-QTY             PIC XX.

* Identifier of the non-currency item dispensed at the ATM.

   02 HOPR-CONTENT         PIC XX.
END
```

The following example is of the BASE24 the PIN Change token DEFINITION PINC-TKN.

```
* The format of the new PIN field.  Valid values are:
* 0 = No encryption, clear PIN
* 1 = Encrypted ANSI PIN block
* 3 = Encrypted PIN/PAD PIN block

02 NEW-PIN-FRMT         PIC X.
```

```
* The PIN offset for the new PIN.

02 NEW-PIN-OFST         PIC X(16)

* The number of new PINs present. Valid values are:
* 1 = One new PIN present
* 2 = Two new PINs present

02 PIN-CNT              PIC X.

* The length of the new PIN (for example, 04 = 4 digits).
* If the new PIN is encrypted, this field contains the
* value 16.  Valid values are in the range from 4 to 12
* and the value 16.

02 NEW-PIN-SIZE         PIC 9(2).

* The new PIN.

02 NEW-PIN-1            PIC X(16).

* The new PIN (second entry).  This PIN is compared to
* the value in the NEW-PIN-1 field to ensure that the
* user has entered the same new PIN twice (that is, that
* the user did not make an error in entering the new PIN)

02 NEW-PIN-2            PIC X(16).

END
```

When you look at your data logs, these are the types of tokens you must identify.

Selecting data for replication

Part of implementing T24 is defining your T24 Structured Record Definition. To do this, you must select the data you wish to replicate. This means you must:

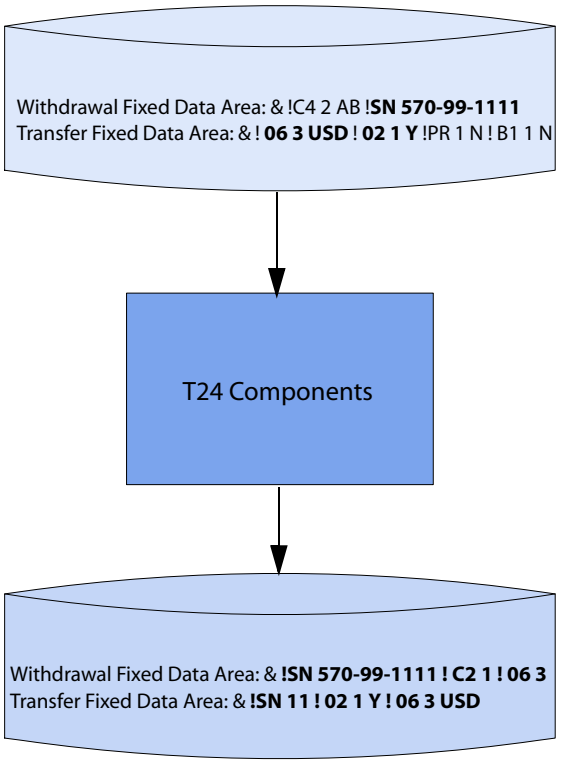
- Identify which tokens you want to place in a fixed structure
- Determine the order your tokens will be written to the target.

For example, you decide you want to capture withdrawal and transfer transactions. Upon examining the transaction log, you determine that withdrawals contain tokens C4, D3, and SN, while transfers contain tokens 06, 02, and B1. You further decide you want final reports to pull data from tokens SN, 02, and 06, in that order. You now have all the details required to create your T24 record definition, as well as configure mappings and the capture process.

Data before and after

This section illustrates how the tokens you selected are extracted from an unstructured BASE24 record and loaded to a fixed structure you have defined. Using the example above, Extract captures tokens SN, 02, and 06 from your source data. The T24 components then examine the extracted records, place the tokens in the order you have specified, and write the output to a GoldenGate trail. If a record does not contain a desired token, (e.g. it has SN but does not have 02 or 06) T24 assigns a default value to the missing token. The default value is determined by the datatype of the missing token (see Figure 6 - captured tokens are in bold).

Figure 6 Before and after tokens



CHAPTER 3

Installing and Configuring T24

This chapter guides you through installing T24. This procedure is discussed in the following topics:

Contents

Installation prerequisites

Before you upload GoldenGate for T24, you must install GoldenGate for HP NonStop in its own subvolume. Instructions and code downloads are available at http://support.goldengate.com.

Uploading T24

To upload required T24 files to your HP NonStop server, you must first download the appropriate zip file. Select this from GoldenGate support based on your HP NonStop operating system version.

Unzip the file on your workstation. The file is in PAK format. The file name will include information such as the:

- Version number of the GoldenGate release (e.g. GGv10)
- Operating system of the NonStop system that will host GoldenGate represented as a letter and number (e.g. G06).

Transfer the file to the HP NonStop Server in binary mode. Use the <GGS volume>.T24 as the destination location.

Locate X24UNPAK. This macro is used to restore Base24 modules using the syntax:

```
TACL> RUN X24UNPAK <module>
```

Where <module> may be D24, T24, N24, or M24. If <module> is left blank, HELP is displayed. If multiple modules are entered, only the last is installed.

To restore the files, run the X24UNPAK macro using T24 as the <module>.

```
TACL> RUN X24UNPAK T24
```

The macro restores the following files to $<GGS volume>.T24 .

- BLDDICT - Obey file for creating the dictionary.
- DDLFT24 - T24 Token DDL file
- EXTPTLF - Run time Extract parameter file PTLF example
- EXTTLF - Run time Extract parameter file TLF example

- INITPTLF - Initial load Extracct parameter file PTLF example
- INITTLF - Initial load Extracct parameter file TLF example
- T24UE - User exit object
- T24UEN - User exit native object

Binding required code

T24 requires Extract to use a user exit to capture, parse, and organize tokenized data from your source system. Before this process can run, the user exit must be bound into Extract using either BINDEXIT for the TNS version of Extract or NLDEXIT for the native version.

Binding for TNS version

The bolded type below is a sample of what can be entered when running BINDEXIT. While running this macro, you may also specify a SQL catalog to use for SQLCOMP processes.

```
TACL> RUN BINDEXIT

BINDEXIT Utility

Creates a new Extract or Replicat object file with bound-in USER EXIT
routines. Enter X at any prompt to quit.

Enter type of object to create, EXTRACT or REPLICAT:      EXTRACT
Enter name of USER EXIT object file:                      T24
Enter name of the NEW EXTRACT object file:                EXTT24
SQL Catalog for the SQLCOMP (or N to avoide SQL compile): GGSCAT
Accelerate code when BIND finished (Y/N)?                 Y
```

Binding for native version

The bolded type below is a sample of what can be entered when running NLDEXIT. This step takes a few minutes to complete. Make sure there are no warnings or errors.

```
TACL> Run NLDEXIT

Creates a new Native EXTRACT or REPLICAT object file linked with a
USEREXIT module.
Enter X at any prompt to quit.

Enter type of GGS object to create
Extract or Replicat or N (nonpriv Replicat):
GGS Object Type:                                  EXTRACT
Enter $Vol.Subvol for EXTRACT Relinkable :        $DATA.GGS8020
Enter location of userexit object :               $DATA.GGS8020.T24N
Enter name for new object file :                  EXTT24

SQL Catalog for SQLCOMP (or N to avoid SQL compile):N
```

Rename Extract

If you plan to run Extract as a continuous online group (rather than a batch), you must rename your Extract object file to run properly. This can be done in either of two ways:

1. Execute the following to rename the BASE24 EXTT24 program as the default Extract program, EXTRACT:

   ```
   TACL> RENAME EXTRACT, EXTRACTO
   TACL> RENAME EXTT24, EXTRACT
   ```

2. Add a PROGRAM statement to the parameter file to point to the BASE24 EXTT24 Extract instead of the default.

   ```
   ADD EXTRACT EXTT24, PROGRAM <volume>.<subvolume>.EXTT24
   ```

Building T24 DDLs

You must create DDL definition files for the tokens you wish to map and replicate. These definitions are placed in the DDLFT24 file, to be used by the Extract output transaction log record.

To build your DDLs, you must:

- Transfer required files
- Create the T24 dictionary
- Update DDLFT24

Transfer required files

Several BASE24 definition files are required to build the T24 dictionaries; move the following to your GGST24 subvolume:

- DDLATTKN
- DDLPSTKN
- DDLFPTFL
- DDLFTFL
- DDLGDEFS
- CUSTCNST (BASE24 version 6.x only)

Create the T24 dictionary

It is necessary to create a T24 DDL subvolume. This subvolume contains the dictionary files, the T24 definitions, and the BASE24 TLF, PTLF & token DDL files.

The following files, which come as part of the T24 product, are required for the DDL subvolume:

File	Description
BLDDICT	This obey file is used to create the dictionary files. Edit this file as needed to set the T24 volume and subvolume, and to add new DDL files.
DDLFT24	This file will contain all the T24 token definitions and output transaction log DDL for TLF and PTLF records.
EXTPTLF	Example online Extract PTLF parameter file.
EXTTLF	Example online Extract TLF parameter file
INITPTLF	Example initial load Extract PTLF parameter file
INITTLF	Example initial load Extract TLF parameter file
T24UE or T24UEN	T24 TNS or native user exit object file

Edit DDL files

You must edit each BASE24 file copied to your T24 subvolume before you can generate your full T24 dictionary. This requires three steps.

1. If your source data is audited, insert the DDL command ?DICT. If your source data is non-audited, insert the DDL command ?DICTN. This command must be the first line in the following DDL files:
   - CUSTCNST
   - DDLATT
   - KNDDL
   - BATKNDDL
   - FPTLFDDL

- FTLFDDL
- GDEFS
- DDLPSTKN

2. Comment out all references that contain an =define_name for the following files:
   - CUSTCNST
   - DDLFPTLF
   - DDLFTLF
3. In the DDLFPTLF file:
   - Change all references for the HEAD definition to PHEAD.
   - Change all references for the AUTH definition to PAUTH. This is required since both the PTLF and TLF records contain AUTH and HEAD definitions. Since there is only one dictionary for both DDLFTLF and DDLFPTLF, one set of definitions must be changed. This includes specifying the alternate key fields.

Generate T24 DDL

Execute BLDDICT to build the GGST24 data dictionaries:

```
TACL> OBEY BLDDICT
```

Create TLF and PTLF T24 template files

The files T24TLF and T24PTLF must exist so that daily captures can map transaction files to your specified T24 format. Create the files by using the FUP output generated when you compiled the DDL:

```
TACL> FUP /IN GGST24.GGT24FUP/
```

Update DDLFT24

GoldenGate provides a sample DDLFT24 file; see Appendix A. The DDLFT24 file contains all the T24 token definitions and the TLF/PTLF Output Transaction Log records. This file should be updated with the token definitions you selected

while preparing for installation. The following examples show how to create the definitions.

Sample T24 token definition for PIN change token TKN06.

The DDLFT24 file contains T24 token definitions and output transaction log records. The PIN Change token TKN06 is defined as DEFINITION TKN06.

```
DEFINITION TKN06.
   02 TKN-HEADER     TYPE *.
   02 PINC-TKN       TYPE *.
END
```

Generating source definitions

To successfuly transfer BASE24 from an unstructured to a structured format, you must define both the source and target layouts, including field names and datatypes. To create these definitions, use the DEFGEN utility, then replicate its output to all target systems in text or ASCII format.

Note Never modify the output of DEFGEN, as unpredictable results may occur.

To execute DEFGEN:

Execute the following:, answering the prompts with the supplied values.

```
TACL>RUN DEFGEN EXPANDDDL RESOLVEDUPGROUP OMITREDEFS

Enter definitions file name (or Exit):            GGSDEF.T24TLF

File/Table to create definition for (or Exit):    GGST24.T24TLF
Include DDL record definition (Y/N)?              Y
DDL Dictionary:                                   GGST24
DDL record of definition name:                    T24-TLF
Definition retrieved.

File/Table to create definition for (or Exit):    GGST24.T24PTLF
Include DDL record definition (Y/N)?              Y
DDL dictionary (default $DATA.GGST24):            GGST24
```

```
DDL record or definition name:                    T24-PTLF
Definition retrieved.

File/Table to create definition for (or Exit):    EXIT
```

Generating target schemas

Because you have defined your T24-TLF and T24-PTLF files and created templates, generating your target schemas is straightfoward. Run DDLGEN and specify your T24 record definitions and file to generate your table create statements in the syntax of your choice. The following is an example:

```
TACL> RUN DDLGEN -D GGSDEF.T24DEF
Output file for table DDL (or Exit):          GGSDEF.T24SQL
DDL template file name (or Exit):             TMPLMSS
Source File/Table (or Exit):                  GGST24.T24TLF
Source File/Table (or Exit):                  GGST24.T24PTLF
Source File/Table (or Exit):                  EXIT
```

When DDLGEN finishes compiling your target schema, transfer the resulting text file in ASCII format to your target system.

CHAPTER 4

Configuring T24 Capture and Delivery

This chapter guides you through configuring the different capture and delivery options that make up GoldenGate for T24 processing. This procedure is discussed in the following topics:

Contents

Capture prerequisites

Before configuring change capture, you must satisfy the following prerequisites:

- Select a change capture method
- Add capture checkpoints
- Define GoldenGate trails

Select a change capture method

Before you can configure change capture for T24, you must select a change capture method. Choices include:

- Logger capture: best used if your BASE24 output is non-audited. Requires a GoldenGate intercept library and disk space for local trails.

  For information on how to set up capture using logger, refer to the chapter on configuring change synchronization in the *GoldenGate for HP NonStop Administrator Guide*.

- Direct read: best used if your BASE24 output is audited. Does not require an intercept library or disk space for local trails.

  The examples that follow explain how to set up this capture method.

Add capture checkpoints

Checkpoints allow you to restart change capture at a specific RBA, instead of having to resync your entire database. The following commands set checkpoints for your TLF and PTLF files.

For direct read:

1. Execute the following commands to set a TLF checkpoint:

```
GGSCI> ADD EXTRACT ET24AT1, FILETYPE ACITLF $DATA.PRO1ATLF.TL*
EXTSEQNO 040622
GGSCI> ADD EXTRACT ET24AT2, FILETYPE ACITLF $DATA.PRO1ATLF.TL*
EXTSEQNO 040623
```

2. Execute the following commands to set a PTLF checkpoint:

```
GGSCI> ADD EXTRACT ET24PS1, FILETYPE ACIPTLF $DATA.PRO1X4TLF.PO*
EXTSEQNO 040622
GGSCI> ADD EXTRACT ET24PS2, FILTEYPE ACIPTLF $DATA.PRO1ATLF.PO*
EXTSEQNO 040623
```

For log trails:

Execute the following command for both TLF and PTLF checkpoints:

```
GGSCI> ADD EXTRACT ET24, LOGTRAILSOURCE $DATA.GGSLOG.AA
```

Define GoldenGate trails

Add the following trails to capture your T24 data.

```
GGSCI> ADD RMTTRAIL C:\GGS\DIRDAT\A1, EXTRACT ET24AT1
GGSCI> ADD RMTTRAIL C:\GGS\DIRDAT\A2, EXTRACT ET24AT2
GGSCI> ADD RMTTRAIL C:\GGS\DIRDAT\P1, EXTRACT ET24PS1
GGSCI> ADD RMTTRAIL C:\GGS\DIRDAT\P2, EXTRACT ET24PS2
```

Preparing the Extract parameter file

GoldenGate capture behavior is controlled through the Extract parameter file. This is where you can specify the range of files to process, the dictionary containing TLF and PTLF record definitions, how the T24 user exit is configured, and how tokens are mapped.

Sample Extract parameter file

The following is a sample Extract parameter file for the Direct Read capture method.

Figure 7 Sample T24 Extract parameter file

```
EXTRACT ET24AT1
ALTINPUT RANGE (1 OF 2) TEMPLATE $DATA.PRO1ATLF.TL*
DICTIONARY $DATA.GGST24
CUSEREXIT
```

```
RMTHOST HOUSTON, MGRPORT 7832
RMTTRAIL C:\GGS\DIRDAT\A1
FILE $DATA.PRO1ATLF.TL*,
   TARGET $DATA.GGST24.T24TLF,
   EXITPARAM “AT0024”,
   DEF TLF,
   TARGETDICT $DATA.GGST24,
   TARGETDEF T24-TLF,
   USETARGETDEFLENGTH,
   COLMAP (USEDEFAULTS,
          FILE-NAME = “    “,
          HEADER-TKN.EYE-CATCHER = “&”,
          HEADER-TKN.USER-FLD1 = “ “,
          HEADER-TKN.CNT = 2,
          HEADER-TKN.LGTH = 22,
          TKN24.TKN-HEADER.EYE-CATCHER = “!”,
          TKN24.TKN-HEADER.USER-FLD1 = “ ”,
          TKN24.TKN-HEADER.TKN-ID = “24”,
          TKN24.TKN-HEADER.LGTH = 10 ),
   WHERE ( REC-TYP <> “04” AND REC-TYP <> “00”);
```

Table 1 Parameters explained

Parameter	Description
ALTINPUT	Specifies the range of files this Extract group will process.
RANGE (1 of 2)	Specifies that this Extract group processes every other day’s file.
TEMPLATE	The set of files to evaluate to identify the next file to process.
DICTIONARY	Specifies the location of your data dictionary containing the TLF and PTLF record definitions.
CUSEREXIT	Instructs Extract to call the T24 user exit

Parameter	Description
RMTHOST	Identifies where to find the remote trail. Supply this value as an IP address or a host name that can be resolved to an IP address.
MGRPORT	Tells Extract which port Manager uses. This must match the port number assigned to the Manager during installation, so Manager can communicate with Extract.
RMTTRAIL	Specifies where captured data will be written. Changes detected on any file specified in the FILE parameter are output to a remote trail.
FILE	Specifies the file set to monitor for new data. Takes a variety of options, including: ◆ **TARGET:** Maps data to a fixed target format and modifies the name in a trail to a standard file name. ◆ **TARGETDEF, TARGETDICT:** Specifies the location of the T24 DDLs ◆ **USETARGETDEFLENGTH:** Specifies the record definition to use for the fixed target length.

Table 2 Sample token fields explained

Token field	Description
HEADER-TKN.EYE-CATCHER	A single character that should be set to an ampersand (&) to identify the beginning of the token header.
HEADER-TKN.USER-FLD1	A single character that must be set to a blank space.

Table 2 Sample token fields explained

Token field	Description
HEADER-TKN.CNT	Two digits that specify the number of tokens plus one for the header.
HEADER-TKN.LGTH	Two digits that store the length of the token header area. This is calculated as: the 6 byte header area, plus each token's length, plus 6 bytes for the token header. In the sample the calculation would be: 6 + (10 + 6) = 22.
TKN24.TKN-HEADER.EYE-CATCHER	A single character that should be set to an exclamation mark (!) to identify the beginning of each token. **Note:** The beginning of the token field name is built from the characters TKN and the identifier of the token. In the sample the token is 24, so the identifier is TKN24. TKNCB would be the identifier for the CB token.
TKN24.TKN-HEADER.USER-FLD1	A single character that must be set to a blank space
TKN24.TKN-HEADER.TKN-ID	The two character identifier of the token.
TKN24.TKN-HEADER.LGTH	The two digit length of the token data area as calculated from the DDL definition.

The EXITPARAM

The GoldenGate Extract parameter file must contain an EXITPARAM, as it maps the tokens in your transaction log to your fixed structure. For the source and

target layouts to match up, the order of the tokens in the output transaction log record must be the same as the order of token ids in this parameter.

The syntax for the EXITPARAM is:

```
EXITPARAM "<TLF type><sequence number><target file name indicator
flag><token id, token id, …>"
```

Argument	Description
`TLF Type`	Two characters are used to identify the type of transaction log file: ◆ AT is used for ATM transaction log files (TLF) ◆ PS is used for POS transaction log files (PTLF)
`Sequence Number`	This has two applications: ◆ A character that requests an action. **W** - Displays a warning message for any token that is larger than defined in the output definition. Example: EXITPARAM "PSW04C04". **D** - Calls Debug after the output of any critical message. Example: EXITPARAM "PSD04C04". **Note:** This should not be used in a production system unless directed to do so by GoldenGate. **T** - Displays the input and output size and a trace message for all tokens. Example: EXITPARAM "PST04C04". **Note:** This should not be used in a production system unless directed to do so by GoldenGate.

Argument	Description
	◆ A numeric character that identifies the sequence number for this transaction log file. The sequence number allows for the possibility of having different token layouts for different financial institutions in the same BASE24 Logical Network. Or different token layouts for financial transactions and administrative transactions.
`Target File Name Indicator Flag`	A one-character flag that indicates the source file name should be used as the target file name. This allows the file name to change the database on the remote platform. ◆ **0** - Zero means to use the target name in the FILE statement. ◆ **1** - One means to use the source file name as the target name.
`Token ID`	The two character token id. Up to 50 token ids can be configured.

EXITPARAM samples

This section outlines samples of EXITPARAM with different types of token mapping.

Example In the example below, the tokens are for the PTLF file, in the following order in the token area of the PTLF record, 04, A1, CB.

```
EXITPARAM "PS0004A1CB"
```

Example The following example is for PTLF files that require a user data token.

```
EXITPARAM "PS00QZ04A1CB"
```

In the example above the first token id must be the special user data token QZ. This is followed by the tokens 04, A1, CB.

Example An ATM TLF example would be:

```
EXITPARAM "AT00A506"
```

In the example above the tokens are for the TLF file. The tokens are in the following order in the token area of the TLF record, A5 and 06.

Example An ATM TLF example with target filename changed to the source filename would be:

```
EXITPARAM "AT01A506"
```

In the example above the tokens are for the TLF file. The tokens are in the following order in the token area of the TLF record, A5 and 06.

Configuring delivery

Once you have configured your capture processes, configuring delivery is quite simple: create delivery checkpoints, and configure the Replicat parameter file.

Create delivery checkpoints

Execute the following to replicate your captured data to your target.

```
TACL> RUN GGSCI
GGSCI> ADD REPLICAT RT24AT1, EXTTRAIL C:\GGS\DIRDAT\A1
GGSCI> ADD REPLICAT RT24AT2, EXTTRAIL C:\GGS\DIRDAT\A2
GGSCI> ADD REPLICAT RT24PS1, EXTTRAIL C:\GGS\DIRDAT\P1
GGSCI> ADD REPLICAT RT24PS2, EXTTRAIL C:\GGS\DIRDAT\P2
```

Configure the Replicat parameter file

The Replicat parameter file defines your target environment and maps your source data, stored on extract trails, to your target. The following is a sample Replicat parameter file.

Figure 8 Sample T24 Replicat file

```
REPLICAT RT24AT1
TARGETDB database, USERID GoldenUser, PASSWORD “password”
PURGEOLDEXTRACTS
SOURCEDEFS C:\GGS\GGSMSS7\DIRDEF\T24.DEF
DISCARDFILE C:\GGS\GGSMSS7\DIRRPT\RT24AT1.TXT, PURGE
MAP $DATA.GGST24.T24TLF, TARGET T24TLF;
```

Table 3 Replicat Parameters explained

Parameter	Description
TARGETDB	Establishes the ODBC data source for the destination database. Required if you are replicating to a SQL Server, DB2, or other ODBC-compliant databases. **Note**: If your target is Oracle, you only need the USERID and PASSWORD options.
PURGEOLDEXTRACTS	Directs Replicat to delete GoldenGate trails once data has been processed.
SOURCEDEFS	Identifies the source definition file, with all metadata, on the source system.
DISCARDFILE	Determines where to write failed operation messages.

Parameter	Description
MAP	Defines a relationship between source and target. Notice that in our sample, a generic T24TLF source is specified, instead of a TLF file from a specific system. Extract renamed all of the source TLF files it extracted to this generic name as part of T24 processing. By making the MAP statement generic, you only need one. The more specific your MAP statement, the more statements you need to cover every type of data that may be captured.

APPENDIX 1

Sample DDLFT24 File

This sample DDL file contains all the T24 Tokens used in the TLF and PTLF output transaction logs. Three sample T24 TLF/PTLF records are defined:

- T24-TLF TLF record
- T24-PTLF PTLF record without a user data field
- T24-PTLF-UD PTLF record with a user data field

```
* The following table list all the standard BASE24 tokens for the BASE,
* ATM and POS products.

*  BASE24 6.0 DATA TOKENS

*  BASE - DDLBATKN

*                      BASE24     T24
*    BASE24            TOKEN      TOKEN
*  TOKEN NAME            ID       NAME   DESCRIPTION
*  ----------          --------   -----  -----------
*  ACCT-QUAL-TKN         18       TKN18  Account Qualifier Token
*  ACQ-RTE-TKN           BA       TKNBA  Acquirer Routing Token
*  CR-LINE-TKN           13       TKN13  Credit Line Token
*  CRD-POSTAL-CDE-TKN    27       TKN27  Cardholder Postal Code Token
*  DATA-ENCRYPTION-KEY-TKN  BN    TKNBN  Data Encryption Key Token
*  EMV-DISCR-TKN         B3       TKNB3  EMV Discretionary Data Token
*  EMV-ISS-SCRIPT-RSLTS-TKN BJ    TKNBJ  EMV Issuer Scripts Results
*  EMV-RQST-TKN          B2       TKNB2  EMV Request Data Token
*  EMV-RESP-TKN          B5       TKNB5  EMV Response Data Token
*  EMV-SCRIPT-TKN        B6       TKNB6  EMV Script Data Token
*  EMV-STAT-TKN          B4       TKNB4  EMV Status Token
*  ISSUER-FEE-REBATE-TKN 30       TKN30  Issuer Fee Rebate Token
*  MICR-DATA-TKN        12      TKN12  Magnetic Ink Char Recognition
*  MULT-CRNCY-TKN        BD       TKNBD  Multi-Currency Token
*  MULT-LN-TKN           BK       TKNBK  Multiple LN Token
*  NAM-TKN               08       TKN08  Customer Short Name Token
*  ORIG-CRNCY-60-TKN     BE       TKNBE  Original Currency 60 Token
*  PRISM-TKN             28       TKN28  Prism Token
*  PSEUDO-CRD-NUM-TKN    BL       TKNBL  Pseudo Card Number Token
*  RVSL-DAT-TIM-TKN      BH       TKNBH  Reversal Date Time Token
*  SURCHARGE^DATA^TKN    25       TKN25  Surcharge Data Token
*  SWI-TKN              B0      TKNB0  Acquirer Generic Switch Token
*  SWI-TKN               B1       TKNB1  Issuer Generic Switch Token
*  TLF-TKN              B7      TKNB7  Transaction Log File Name Tkn
*  TRACK1-TKN            23       TKN23  Track1 Token
*  TRK3-TKN              BG       TKNBG  Track3 Token
*  TXN-DESCR-TKN        B9      TKNB9  Transaction Description Token
*  TXN-PRFL-TKN          B8       TKNB8  Transaction Profile Token
*  TXN-SUBTYP-TKN        BM       TKNBM  Transaction Subtype Token
```

```
*  ATM - DDLATTKN

*                      BASE24     T24
*    BASE24            TOKEN      TOKEN
*  TOKEN NAME            ID       NAME   DESCRIPTION
*  ----------          --------   -----  -----------
*  ADDL-HOPR-TKN         22       TKN22  Additional Hopper Token
*  AT-FLG1-TKN           24      TKN24  ATM Flag1 (Misc fields) Token
*  AT50-TKN              03       TKN03  BASE24-atm Release 5.0 Token
*  ATM-BAL-TKN           AB       TKNAB  BASE24-atm Balances Token
*  CASH-ACCPT-TERM-SETL-TKN AD    TKNAD  Cash Accept Term Setl Token
*  ICHG-COMPLIANCE-ATM-TKN  A6    TKNA6  Interchange Compliance Token
*  MBC-BD-TKN            A8      TKNA8  Merch Bank Center Bag Deposit
*  MBC-MX-TKN            A9      TKNA9  Merch Bank Center Money Exchg
*  MBC-SETL-TKN          AA       TKNAA  MBC Settlement Token
*  MULT-ACCT-TKN                         Multiple Account Token
*  NCD-TKN               A5       TKNA5  Non-Currency Dispense Token
*  PINC-TKN              06       TKN06  PIN Change Token
*  PS2000-ATM-TKN        21      TKN21  Payment service 2000 ATM Token
*  SM-PRI-TKN            A0       TKNA0  Smart Card Primary Token
*  SM-REFR-TKN           A2       TKNA2  Smart Card Refresh Token
*  SM-TERM-SETL-TKN      A4      TKNA4  Smart Card Terminal Settlement
*  SM-VISA-TKN           A3       TKNA3  Smart Card Visa Token
*  SSBB-TKN              07       TKN07  Self-Service Bank Base Token
*  SSBC-TKN              14      TKN14  Self-Service Bank Check Token
*  SSBC-TERM-SETL-TKN    15      TKN15  Self-Serv Bank Check Term Setl
*  STMT-PRNT-TKN         02       TKN02  Statement Print Token
*  MULT-ACCT-TKN         A7       TKNA7  Multiple Account Token

*  POS - DDLPSTKN

*                      BASE24     T24
*    BASE24            TOKEN      TOKEN
*  TOKEN NAME            ID       NAME   DESCRIPTION
*  ----------          --------   -----  -----------
*  ACH-DB-TKN            11      TKN11  Automated Clearing House Debit
*  ADDR-VER-TKN          01       TKN01  Address Verification Token
*  ALT-MERCH-ID          16       TKN16  Alternate Merchant ID Token
*  AMEX-TKN              10       TKN10  American Express Token
*  AUTHN-DATA-TKN        CE       TKNCE  Authentication Data Token
*  CERT-TKN              C3       TKNC3  Certificate Token
```

```
*  CHK-AUTH-TKN          05      TKN05  Check Authorization Token
*  CHK-AUTH2-TKN         29      TKN29  Check Authorization Token
*  CHK-CALLBACK-TKN      31      TKN31  Check Callback Token
*  CRDHLDR-SERIAL-NUM-TKN C8    TKNC8  Cardholder Serial Number Token
*  DUKPT-DATA-TKN        CA      TKNCA  Derived Unique Key Per Trans
*  EBT-AVAIL-BAL-TKN     U1      TKNU1  EBT Available Balance Token
*  EBT-VOUCHER-NUM-TKN   U0      TKNU0  EBT Voucher Number Token
*  IAVS-DATA-TKN         CF      TKNCF  IAVS Data Token
*  ICHG-COMPLIANCE-TKN   20      TKN20  Interchange Compliance Token
*  MHI-ADDL-DATA-TKN     C6      TKNC6  Merchant Host Interface
*                                       Additional DataToken
*  MRCH-SERIAL-NUM-TKN   C9      TKNC9  Merchant serial Number Token
*  OPT-DATA-TKN          C5      TKNC5  Increased Optional Data Token
*  POS-BAL-TKN           CB      TKNCB  POS Balances Token
*  POS-DATA1-TKN         CH      TKNCH  POS Data1 Token
*  POS-MRCH-TKN          CI      TKNCI  POS Merchant Token
*  PS2000-OFFL-TKN       19      TKN19  VISA Pmt Serv 2000 Offline
*  PS2000-TKN            17      TKN17  VISA Payment Service 2000
*  PS50-TKN              04      TKN04  POS 5.0 Token
*  PS51-TKN              C0      TKNC0  POS 5.1 Token
*  PT-SRV-DATA-TKN       C4      TKNC4  Point of Service Data Token
*  PURCHASE-TKN          C2      TKNC2  Purchasing Card & Fleet Card
*  STA-ID-TKN            C1      TKNC1  Station ID Token
*  STORED-VALUE-TKN      U2      TKNU2  Stored Value Token
*  TRANS-STAIN-XID-TKN   C7      TKNC7

* ATM - T24-TLF

* This example shows how to create the TLF Output Transaction Log DDL
* 'T24-TLF'.

* The TLF Output Transaction Log DDL contains the PIN Change token
* (PINC-TOKEN)and the Non-Currency Dispense token (NCD-TOKEN)

* 1) Create the TKN06  PIN Change Token

* DEFINITION TKN06.
*   02 TKN-HEADER   TYPE *.  DDLBATKN - Data Token Header definition
*   02 PINC-TKN     TYPE *.  DDLATTKN - the NCD Token definition
* END
```

```
DEFINITION TKN06.
  02 TKN-HEADER     TYPE *.
  02 PINC-TKN       TYPE *.
END

* 2) Create the TKNA5  Non-Currency Dispense

* DEFINITION TKNA5.
*   02 TKN-HEADER   TYPE *.  DDLBATKN - Data Token Header definition
*   02 NCD-TKN      TYPE *.  DDLATTKN - the NCD Token definition
* END

DEFINITION TKNA5.
  02 TKN-HEADER      TYPE *.
  02 NCD-TKN         TYPE *.
END

* 3) Create the FILE-NAME Definition

*DEFINITION FILE-NAME  PIC X(8).

* 4) Create the T24 TLF Output Transaction Log DDL

* RECORD T24-TLF.
*   02 FILE-NAME    TYPE *.  DDLFT24 - the FILE-HEAD definition
*   02 HEAD         TYPE *.  DDLFTLF - the HEAD definition
*   02 AUTH         TYPE *.  DDLFTLF - the AUTH definition
*   02 HEADER-TKN   TYPE *.  DDLBATKN - the HEADER Token definition
*   02 TKNA5        TYPE *.  DDLFT24 - the TKNA5 definition
*   02 TKN06        TYPE *.  DDLFT24 - the TKN06 definition
* END

RECORD T24-TLF.
  02 FILE-NAME      TYPE *.
  02 HEAD           TYPE *.
  02 AUTH           TYPE *.
  02 HEADER-TKN     TYPE *.
  02 TKNA5          TYPE *.
  02 TKN06          TYPE *.
END

* POS - T24-PTLF (without a user data field)
```

```
* This example shows how to create the PTLF Output Transaction Log DDL
* 'T24-PTLF' that does not contain a user date field.

* The PTLF Output Transaction Log DDL contains:
*  1) Multi-Currency token (MULT-CRNCY-TKN)          BD
*  2) POS 5.1 token (PS51-TKN)                       C0
*  3) Station ID token (STA-ID-TKN)                  C1
*  4) Point of Service Data token (PT-SRV-DATA-TKN)  C4

* 1) Create the TKNBD  Multi-Currency token

* DEFINITION TKNBD.
*  02 TKN-HEADER     TYPE *.  DDLBATKN - Data Token Header definition
*   02 MULT-CRNCY-TKN  TYPE *. DDLPSTKN - Multi-Currency definition
* END

DEFINITION TKNBD.
  02 TKN-HEADER       TYPE *.
  02 MULT-CRNCY-TKN   TYPE *.
END

* 2) Create the TKNC0  POS 5.1 token

* DEFINITION TKNC0.
*   02 TKN-HEADER   TYPE *.  DDLBATKN - Data Token Header definition
*   02 PS51-TKN     TYPE *.  DDLPSTKN - the POS 5.1 Token definition
* END

DEFINITION TKNC0.
  02 TKN-HEADER      TYPE *.
  02 PS51-TKN        TYPE *.
END

* 3) Create the TKNC1  Station ID token

* DEFINITION TKNC1.
*   02 TKN-HEADER   TYPE *.  DDLBATKN - Data Token Header definition
*   02 STA-ID-TKN   TYPE *.  DDLPSTKN - Station ID token definition
* END

DEFINITION TKNC1.
  02 TKN-HEADER      TYPE *.
```

```
  02 STA-ID-TKN      TYPE *.
END

* 4) Create the TKNC4  Point of Service Data token

* DEFINITION TKNC4.
*   02 TKN-HEADER      TYPE *.  DDLBATKN - Data Token Header definition
*   02 PT-SRV-DATA-TKN  TYPE *.  DDLPSTKN - Point of Service definition
* END

DEFINITION TKNC4.
  02 TKN-HEADER       TYPE *.
  02 PT-SRV-DATA-TKN  TYPE *.
END

* 5) Create the T24 PTLF Output Transaction Log DDL

* RECORD T24-PTLF.
*    02 FILE-NAME    TYPE *.  DDLFT24 - the FILE-HEAD definition
*    02 PHEAD        TYPE *.  DDLFTLF- the HEAD definition
*    02 PAUTH        TYPE *.  DDLFTLF - the AUTH definition
*    02 HEADER-TKN   TYPE *.  DDLBATKN -the HEADER Token definition
*    02 TKNBD        TYPE *.  DDLFT24 - the TKNBD definition
*    02 TKNC0        TYPE *.  DDLFT24 - the TKNC0 definition
*    02 TKNC1        TYPE *.  DDLFT24 - the TKNC1 definition
*    02 TKNC4        TYPE *.  DDLFT24 - the TKNC4 definition
* END

RECORD T24-PTLF.
  02 FILE-NAME      TYPE *.
  02 PHEAD          TYPE *.
  02 PAUTH          TYPE *.
  02 HEADER-TKN     TYPE *.
  02 TKNBD          TYPE *.
  02 TKNC0          TYPE *.
  02 TKNC1          TYPE *.
  02 TKNC4          TYPE *.
END

* POS - T24-PTLF-UD  (with user data)

* This example shows how to create the PTLF Output Transaction Log DDL
```

```
* 'T24-PTLF-UD' when the PTLF has a user data field.

* The PTLF Output Transaction Log DDL contains:
*  1) User Data token (USER-DATA-TKN)               QZ

* The following tokens have already been defined:
*     Multi-Currency token (MULT-CRNCY-TKN)         BD
*     POS 5.1 token (PS51-TKN)                      C0
*     Station ID token (STA-ID-TKN)                 C1
*     Point of Service Data token (PT-SRV-DATA-TKN) C4

* 0) Create the User Data token  USER-DATA-TKN

DEFINITION USER-DATA-TKN.
  02 DATA-LEN         TYPE BINARY 16.
  02 FIELD-1          PIC 9(2).
  02 FIELD-2          TYPE BINARY 16.
  02 FIELD-3          PIC X(34).
  02 FIELD-4          TYPE BINARY 64.
END

* 1) Create the TKNQZ User Data token

* DEFINITION TKNQZ.
*   02 TKN-HEADER      TYPE *.  DDLBATKN - the Data Token Header def
*   02 USER-DATA-TKN   TYPE *.  DDLFT24  - the User Data token def
* END

DEFINITION TKNQZ.
  02 TKN-HEADER       TYPE *.
  02 USER-DATA-TKN    TYPE *.
END

* 3) Create the T24 PTLF with User Data Output Transaction Log DDL

* RECORD T24-PTLF-UD.
*   02 FILE-NAME    TYPE *.  DDLFT24 - the FILE-HEAD definition
*   02 PHEAD        TYPE *.  DDLFPTLF - the HEAD definition
*   02 PAUTH        TYPE *.  DDLFPTLF - the AUTH definition
*   02 HEADER-TKN   TYPE *.  DDLBATKN - the HEADER Token definition
*   02 TKNQZ        TYPE *.  DDLFT24 - the TKNQZ definition
*   02 TKNBD        TYPE *.  DDLFT24 - the TKNBD definition
```

```
*   02 TKNC0        TYPE *.  DDLFT24 - the TKNC0 definition
*   02 TKNC1        TYPE *.  DDLFT24 - the TKNC1 definition
*   02 TKNC4        TYPE *.  DDLFT24 - the TKNC4 definition
* END

RECORD T24-PTLF-UD.
  02 FILE-NAME      TYPE *.
  02 PHEAD          TYPE *.
  02 PAUTH          TYPE *.
  02 HEADER-TKN     TYPE *.
  02 TKNQZ          TYPE *.
  02 TKNBD          TYPE *.
  02 TKNC0          TYPE *.
  02 TKNC1          TYPE *.
  02 TKNC4          TYPE *.
END
```

APPENDIX 2
T24 Messages

The messages in this appendix write to the GoldenGate report file, which can be reviewed as part of regular system maintenance, as well as for troubleshooting. Currently, T24 only writes error messages to the report file, which can be accessed by executing the following:

```
TACL> Volume <GoldenGate volume and subvolume>
TACL> Run GGSCI
GGSCI> View Report <Extract Group Name>
```

Error messages

UE 300 ERROR: GET_RECORD FAILED

Cause An Internal error occurred. The retrieval of the input record failed.

Recovery Contact support at Golden Gate.

UE 305 ERROR: GET_SYSKEY_LENGTH FAILED

Cause An internal error occurred. The retrieval of the system key length failed.

Recovery Contact support at Golden Gate.

UE 310 ERROR: GET_FILENAME FAILED

Cause An internal error occurred. The retrieval of the source file name failed

Recovery Contact support at Golden Gate.

UE 315 ERROR: THE EXITPARAM MUST NOT CONTAIN INVALID CHARACTERS

Cause The EXITPARAM contains invalid values.

Recovery Check the EXITPARAM. The format is a quotation mark, a two character constant AT or PS followed by the Sequence Number as 0, W, T, or D, the Indicator Flag as 0 or 1, the list of two character token ids with no separating characters and a final quotation mark. The total length for the EXITPARAM must be an even number of characters. The values must be contiguous without any spaces or other characters between the values. Restart Extract processing as appropriate.

UE 320 ERROR: THE EXITPARAM MUST CONTAIN EITHER 'AT' OR 'PS'.

Cause The BASE24 product must be specified in the first 2 characters of the EXITPARAM. Currently, only two products are supported: 'AT' for ATM and 'PS' POS.

Recovery Correct the EXITPARM to include the two-character product id. Restart Extract processing as appropriate.

UE 325 ERROR: THE EXITPARAM MUST CONTAIN A SEQ# OR 'D', 'T', OR 'W'.

Cause The EXITPARAM does not have a valid Sequence Number after the first two characters. This must be a numeral, such as 0, or one of the valid special characters: D, T, or W. The characters are used for special displays and the numeric is for future processing enhancements.

Recovery Check the third character of the EXITPARAM. It must be a sequence number or one of the special characters D, T, or W. The total length for the EXITPARAM must be an even number of characters. The values must be contiguous without any spaces or other characters between the values. Correct the EXITPARM to include a valid third character, "AT0..." or "ATW..." for instance. Restart Extract processing as appropriate.

UE 327 ERROR: THE EXITPARAM MUST CONTAIN AN INDICATOR FLAG OF '0' OR '1'.

Cause The EXITPARAM indicator flag number must be specified after the first 3 characters. This is for Future processing enhancements and currently includes the TARGETFILE override.

Recovery Check the EXITPARAM. The fourth character must be a character constant 0 or 1. The total length for the EXITPARAM must be an even number of characters with the Indicator Flag being one character. The previous parameter sequence number must also be one character. The values must be contiguous without any spaces between each parameter value. Correct the EXITPARM to include a digit for the Indicator Flag, for example "AT00..." or "ATW1..." (the fourth digit is the Indicator Flag). Restart Extract processing as appropriate.

UE 330 ERROR: THE EXITPARAM MUST HAVE AT LEAST 1 TOKEN ID SPECIFIED

Cause The EXITPARAM contains invalid values. Token ids are two characters with no separating characters.

Recovery Check the EXITPARAM. The last characters must represent at least one token id. The total length for the EXITPARAM must be an even number of characters. The values must be contiguous without any spaces or other characters between the values, for example "AT00C4" where C4 is a token id. Restart Extract processing as appropriate.

UE 335 ERROR: THE EXITPARAM MUST HAVE AN EVEN NUMBER OF CHARACTERS

Cause The EXITPARAM contains invalid values. Token Ids are two characters with no separating characters. The values must be contiguous without any spaces or other characters between the values, for example "PS00C4B9".

Recovery Correct the invalid values and restart Extract processing as appropriate.

UE 340 ERROR: EXITPARAM TOKEN IDS OUT OF ORDER DOES NOT MATCH RECORD

Cause The EXITPARAM token id order is incorrect when compared to the DDL output record.

Recovery Correct the output DDL or correct the EXITPARAM to match each other. Also, check the initialization from the COLMAP in the parameter files, this includes the length fields and the token ids. Restart Extract processing as appropriate.

UE 343 ERROR: OUTPUT TOKEN ID LENGTH (LGTH> IS INVALID FOR <TOKEN ID>

Cause The mapping for the token id size is incorrect compared to the output record. This can occur when there are multiple fields that have the same field name and the default value has not been set.

Recovery Correct the DDL definition or correct the EXITPARAM to initialize any fields that have the same field names. Also check the initialization defined for COLMAP in the parameter files. This

includes the length and token ids and there are usually fields that need to have defaults declared to ensure the proper default mapping. Restart Extract processing as appropriate.

UE 345 ERROR: EXITPARM TOKEN IDS OR # SPECIFIED DOES NOT MATCH RECORD

Cause The EXITPARAM is incorrect when compared to the DDL output record.

Recovery Correct the output DDL or correct the EXITPARAM to match with the same number and order of tokens. Also, check the initialization from the COLMAP in the parameter files this includes the length fields and the Token Ids. Restart Extract processing as appropriate.

UE 350 ERROR: INTERNAL ERROR. THE TOKEN TABLE HAS BEEN CORRUPTED

Cause The internal Token Table contains invalid characters.

Recovery Contact support at Golden Gate.

UE 355 ERROR: EXITPARM TOKEN IDS SPECIFIED DO NOT MATCH THE RECORD

Cause The EXITPARAM is incorrect when compared to the DDL output record.

Recovery Correct the output DDL or correct the EXITPARAM to match with the same number and order of tokens. Also, check the initialization from the COLMAP in the parameter files. Check for duplicate token ids. Restart Extract processing as appropriate.

UE 360 WARNING: TOKEN <TOKEN ID> SIZE <LGTH> DOES NOT MATCH OUTPUT TOKEN SIZE <LGTH>

Cause The actual length of the data (LGTH) is larger than that defined in the DDL. By default the data will be truncated to the maximum

size defined. This message is displayed only when the Sequence Number (third character) of the EXITPARAM is set to W.

Recovery Correct the output DDL, allow the data to be truncated or change the Sequence Number of EXITPARAM to 0 to suppress the message. Restart Extract processing as appropriate.

UE 365 TRACE: TOKEN <TOKEN ID> SIZE <LGTH> OUTPUT TOKEN SIZE <LGTH>

Cause This message is triggered by entering T in the Sequence Number (third character) of the EXITPARAM. It displays the mapped data size and the output size.

Recovery Change the Sequence Number of EXITPARAM to 0 to suppress the message. Restart Extract processing as appropriate.