ATM Database Management

ET Data Storage

ET Data Storage in the Transaction Manager’s Recovery File

By default, when Adabas Transaction Manager is in use with the runtime parameter setting `TMETDATA=ATM`, `ET` data is stored in and read from the transaction manager’s recovery file. (This can be overridden by the client runtime control `Application controls ET data`.)

If your applications need their current `ET` data to be established in the transaction manager’s recovery file before they can execute, refer to section Copy ET Data for more information.

ET Data Storage with External Transaction Coordinators

When running with the CICS Syncpoint Manager or Recoverable Resource Management Services (RRMS), it is not possible to synchronize the storage of `ET` data when an unsolicited syncpoint occurs, because CICS and RRMS syncpoints have no knowledge of `ET` data.

If an application stores `ET` data and runs in a CICS/RMI or RRMS environment, you can ensure that the storing of its `ET` data is synchronized with the two-phase commit process by conforming to the following rules:

- Any syncpoint for which `ET` data is to be stored must be triggered by an `ET` or `CL` command.
- The `ET` or `CL` command that triggers the syncpoint must also supply the `ET` data; that is, if the application issues a series of `ET` commands to different databases, the first `ET` must supply the `ET` data.

In an IMS TM system whose transactions are coordinated by RRMS, it is not possible to store `ET` data synchronously with an RRMS syncpoint. IMS allows an RRMS commit syncpoint to take place only at the successful completion of message processing, and this syncpoint cannot be triggered by an `ET` or `CL` command.

Pool/Queue Usage Control

Pools and queues used by ATM will expand dynamically as required. The high-water marks can be displayed using the Online Services application.

Excluding DBMSs from Global Transaction Processing

Rigorous management of global transactions inevitably creates overhead, which can be minimized by careful exclusion of certain databases. For example:

- A development database generally does not require the same guarantees of transaction integrity as a production database and can therefore be excluded from two-phase commit processing.
- System files generally do not require the same guarantees of transaction integrity as application data files. If you maintain application data files and system files in different databases, those containing system files can often be excluded from two-phase commit processing.

The ADARUN DTP parameter is used to include an Adabas database (DTP=RM) or exclude it from (DTP=NO) participation in two-phase commit processing. Remember, however, that ATM generates ET or BT commands to changed databases that run with DTP=NO, when a user's global transaction terminates. For more information about the way ATM handles commands directed at databases running with DTP=NO, see the section Adabas Transactional Commands.

Disengaging a Database from Two-Phase Commit Processing

If a database running with the ADARUN parameter setting DTP=RM is terminated, a message is issued by the transaction manager job indicating

- that an RM has signed off;
- whether or not the RM had unresolved transactions; and
- if so, whether the unresolved transactions were prepared or unprepared.

If you restart the database with DTP=NO, you must also specify IGNDTP=YES for its first execution. However, you may lose the integrity of incomplete prepared transactions if you restart the database with DTP=NO and IGNDTP=YES. To avoid this, you must resolve any incomplete transactions before switching to DTP=NO.

If you restart a database with a different DTP parameter value, the change will not be recognized by any ATM client proxy components, for clients who are already in session with that database. This could cause errors if such a client tries to change the database. However, new client sessions will recognize the new setting, and clients who close the database and issue a new OP command will be able to carry on processing normally.

If you need to change a database's DTP parameter, the safest procedure is as follows:

- ensure that the database is closed cleanly, with no incomplete global transactions in flight;

- restart all application/client environments which use the database;
- restart the database.