Recover EDB and Export Exchange Database to PST 2010

***Overview*:**

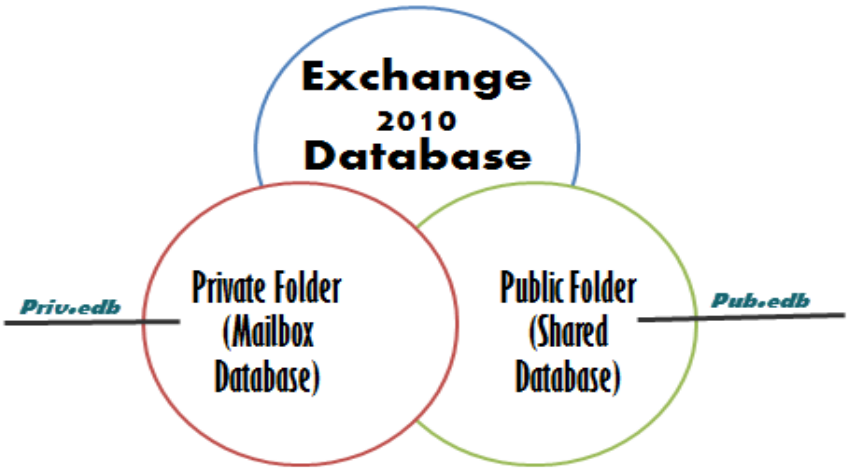
The Exchange Store (store.exe) is the main repository of Exchange Server 2010 edition. In this article, the infrastructure of store.exe along with its components and other important details are covered up along with solution to recover and export Exchange database to PST 2010 in corruption circumstances.

Exchange 2010 Store and Its Components:

There are two major components that make up database of Exchange Server:

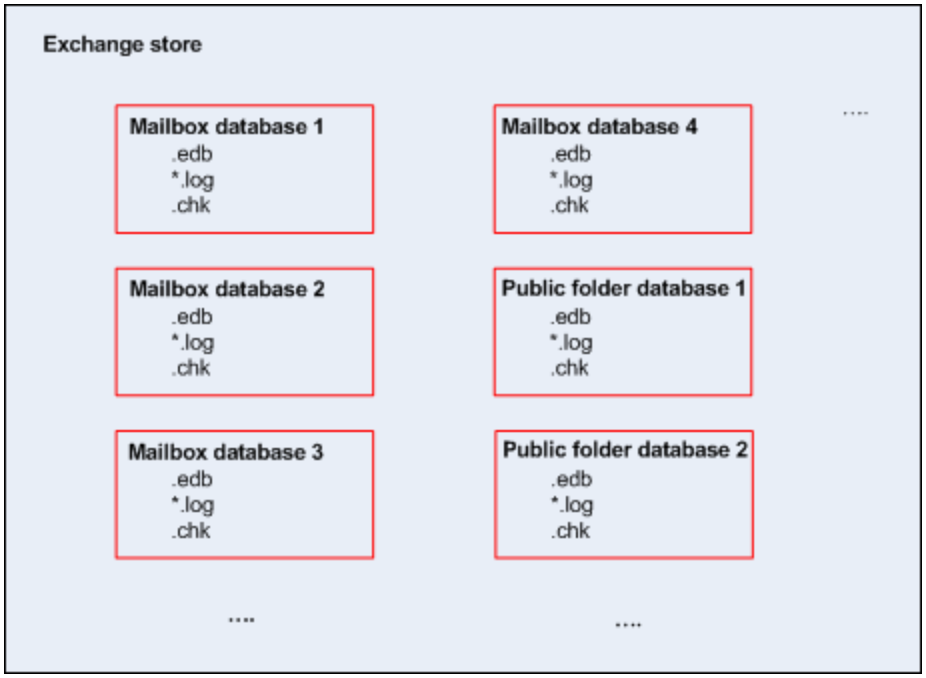
Private Folders as the name suggests stores private information of an individual user. It comprises if data and its definitions, flags, indexes etc. For every private folder, an individual mailbox is created that get saved in priv.edb file.

Public Folders again comprises of same information as that of private folders but is shared amongst various users within Exchange environment. The accessibility of this folder depends upon permissions that can be set through Exchange Management Shell conveniently or through Exchange Management Shell.



Structure of Exchange Server 2010 Store is managed through three special kind of data files: the Exchange database files (.edb), the transaction log files (.log), and the checkpoint files (.chk). However, you won't come across these files until operations like online backup, soft/hard recovery has to be performed. The following table describes all these data files in short:

Data file	Description
Exchange database (.edb)	These files are the repository for mailbox data. They're accessed by the Extensible Storage Engine (ESE) directly and have a B-tree structure designed for quick access. This enables users to access any page of data within one input/output (I/O) cycle, which is a four-fold increase compared to Microsoft Exchange Server 2007. Exchange databases are composed of multiple B-trees, with ancillary trees that work with the main tree by holding indexing and views.
Transaction log (.log)	These files are the repository for database operations such as creating or modifying a message. Committed operations are later written to the database itself (in an .edb file). This approach guarantees that all complete and incomplete transactions are logged to maintain data integrity in case of a service interruption. Each database has its own set of transaction logs.
Checkpoint (.chk)	These files are the repository for data that indicates when an operation is successfully saved to the database on the hard disk. Exchange 2010 uses .chk files so an instance of the ESE can automatically replay log files into an inconsistent database when recovering from a service interruption, starting with the next unwritten operation. The .chk files are placed in the same log location as the .log files.



Exchange Transaction Logging: Exchange Server is popularly known as transaction based email client. A transaction is a set of operations that are performed on a database (example: deletion, insertion, or up-dation) and follows ACID properties:

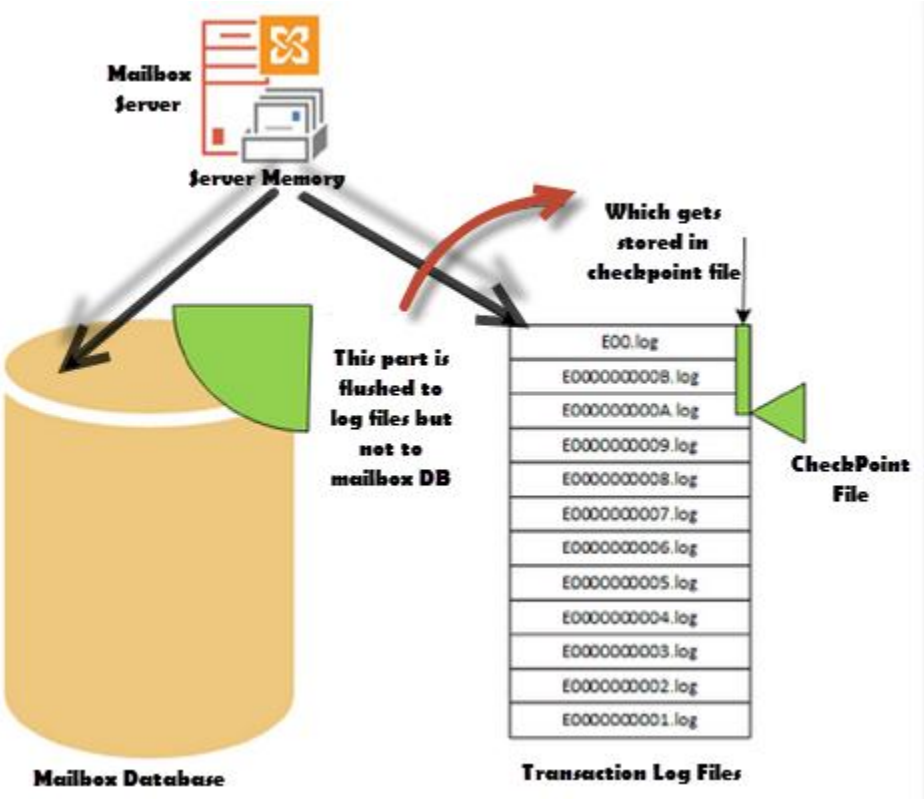
Atomic: For a transaction to be complete, it is necessary that all of the operations occur successfully. For example: if an email is to be sent to another user, all operations included in email send/receive process should take place effectively.

Consistent: When a transaction is committed, it is transformed from an inconsistent state to consistent state. This means, a change in state of DB takes place.

Isolated: Changes done in the database will not be visible at user's end only when the transaction is complete (i.e. executed properly one after another).

Durable: This ensures that when a transaction is committed, it will be reserved on the system and will exist even if the system crashes.

A transaction is committed only when it is it passes the durability test which means the changes and the DB are protected against any kind of loss. And the database engine (ESE) will only commit a transaction when there is surety that transaction is flushed from memory to the disk (into LOG files).



Most of the Exchange DBA's would agree to the fact that for data recovery case of disaster, transaction logs are very important. Log file contains information that database (.edb) file does not and if they are missing, the entire database will be inconsistent or unusable. Extensible Store Engine (ESE) uses the concept of ***Write-Ahead Logging*** where the transactions are first written to the dis or log files before being written to the Information Store. This mechanism is to avoid hardware failures and the database will remain in consistent state even if the system crashes.

LOG files are generated in a sequential manner and store data according to order the transaction occur. Depending upon the version, each log file can contain a specific amount of data and once that much space is occupied, new log files with corresponding series get

generated. LOG file for first storage group is named automatically as E00.log and when it gets filled, a new file will get created with changes in prefix like E01.log, E02.log,…..E0n.log. However, it has to be noticed that the numbering is done is hexadecimal format.

What is Checkpoint File?

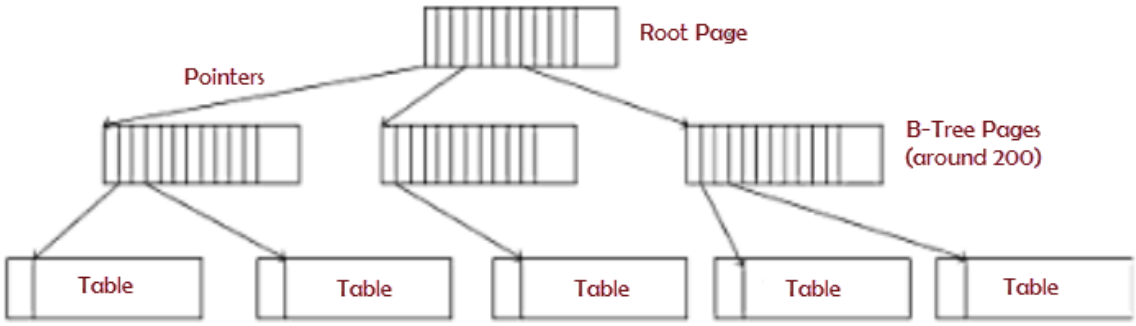
For database recovery, not all log files are required and Checkpoint file is the mechanism that tells from which transaction log file the recovery should start. A checkpoint (.chk) file keeps a track of transactions that are still not committed to database. Corresponding to a log file, a checkpoint file gets generated with name like E0n.CHK that is meant to maintain a status between the memory and database files on disk. This means that an indication will be received as from which uncommitted transaction the recovery process has to be started.

Extensible Storage Engine (ESE):

JET database engine has two variants: JET "Red" and JET "Blue". While the JET Red was meant for single database engine (example MS Access), the JET Blue is well suited for multi-user access and thus is used in Exchange Server. The Blue variant of JET was renamed as Exchange Extensible Engine (ESE).

This database engine technique stores Exchange Server database in a B-tree. Every table in Information Store is a collection of B-Trees. The index of a balanced-tree helps to reduce I/O operations required to access the database.

Example of B-Tree



Testing Exchange Store Health: Exchange Server 2010 is integrated with facility to detect and fix store database if it is found unhealthy. Using Exchange Troubleshooters or Isinteg utility, Information Store health can be checked out. Poison (corrupt) mailboxes can be detected and repaired using the PowerShell cmdlets. To repair a corrupt mailbox, the *New-MailboxRepairRequest* command can be used. The quality of these cmdlets is you can use them while the database is running live.

To detect corruption in mailbox, following cmdlet can be used:

```
New-MailboxRepairRequest -Mailboxname-CorruptionType ProvisionedFolder,SearchFolder,FolderView –DetectOnly
```

In order to repair a poison mailbox via PowerShell cmdlets, use the below mentioned syntax:

```
New-MailboxRepairRequest -MailboxName -CorruptionType ProvisionedFolder,SearchFolder,FolderView
```

Exchange 2010 Database Recovery to PST:

If Exchange Server database in .edb file is corrupt, then recovering mailboxes to PST file is possible using Exchange Recovery software. The solution has power to work around consequences of physical and logical EDB file corruption and exports recovery results to PST, EML, and MSG file.