a **transaction log** (also **transaction journal**, **database log**, **binary log** or **audit trail**) is a history of actions executed by a database management system used to guarantee ACID properties over crashes or hardware failures. Physically, a log is a file listing changes to the database, stored in a stable storage format.

If, after a start, the database is found in an inconsistent state or not been shut down properly, the database management system reviews the database logs for uncommitted transactions and rolls back the changes made by these transactions. Additionally, all transactions that are already committed but whose changes were not yet materialized in the database are re-applied. Both are done to ensure atomicity and durability of transactions.

This term is not to be confused with other, human-readable logs that a database management system usually provides.

In database management systems, a journal is the record of data altered by a given process.[1][2][3][4]

Anatomy of a general database log

A database log record is made up of:

- *Log Sequence Number*: A unique ID for a log record. With LSNs, logs can be recovered in constant time. Most LSNs are assigned in monotonically increasing order, which is useful in recovery algorithms, like ARIES.
- *Prev LSN*: A link to their last log record. This implies database logs are constructed in linked list form.
- *Transaction ID number*: A reference to the database transaction generating the log record.
- *Type*: Describes the type of database log record.
- Information about the actual changes that triggered the log record to be written.

Types of database log records

All log records include the general log attributes above, and also other attributes depending on their type (which is recorded in the *Type* attribute, as above).

- **Update Log Record** notes an update (change) to the database. It includes this extra information:
  - *PageID*: A reference to the Page ID of the modified page.
  - *Length and Offset*: Length in bytes and offset of the page are usually included.
  - *Before and After Images*: Includes the value of the bytes of page before and after the page change. Some databases may have logs which include one or both images.
- **Compensation Log Record** notes the rollback of a particular change to the database. Each corresponds with exactly one other Update Log Record (although the corresponding update log record is

not typically stored in the Compensation Log Record). It includes this extra information:

  - *undoNextLSN*: This field contains the LSN of the next log record that is to be undone for transaction that wrote the last Update Log.

- **Commit Record** notes a decision to commit a transaction.

- **Abort Record** notes a decision to abort and hence roll back a transaction.

- **Checkpoint Record** notes that a checkpoint has been made. These are used to speed up recovery. They record information that eliminates the need to read a long way into the log's past. This varies according to checkpoint algorithm. If all dirty pages are flushed while creating the checkpoint (as in PostgreSQL), it might contain:
  - *redoLSN*: This is a reference to the first log record that corresponds to a dirty page. i.e. the first update that wasn't flushed at checkpoint time. This is where redo must begin on recovery.
  - *undoLSN*: This is a reference to the oldest log record of the oldest in-progress transaction. This is the oldest log record needed to undo all in-progress transactions.

- **Completion Record** notes that all work has been done for this particular transaction. (It has been fully committed or aborted)

See also

- Data logging
- Error correction and detection
- Hash function
- Journaling file system
- Log-structured file system
- Write-ahead logging
- Redo log