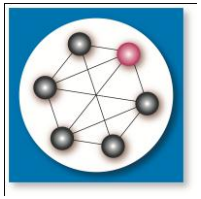


www.dbtechnet.org

Basics of SQL Transactions

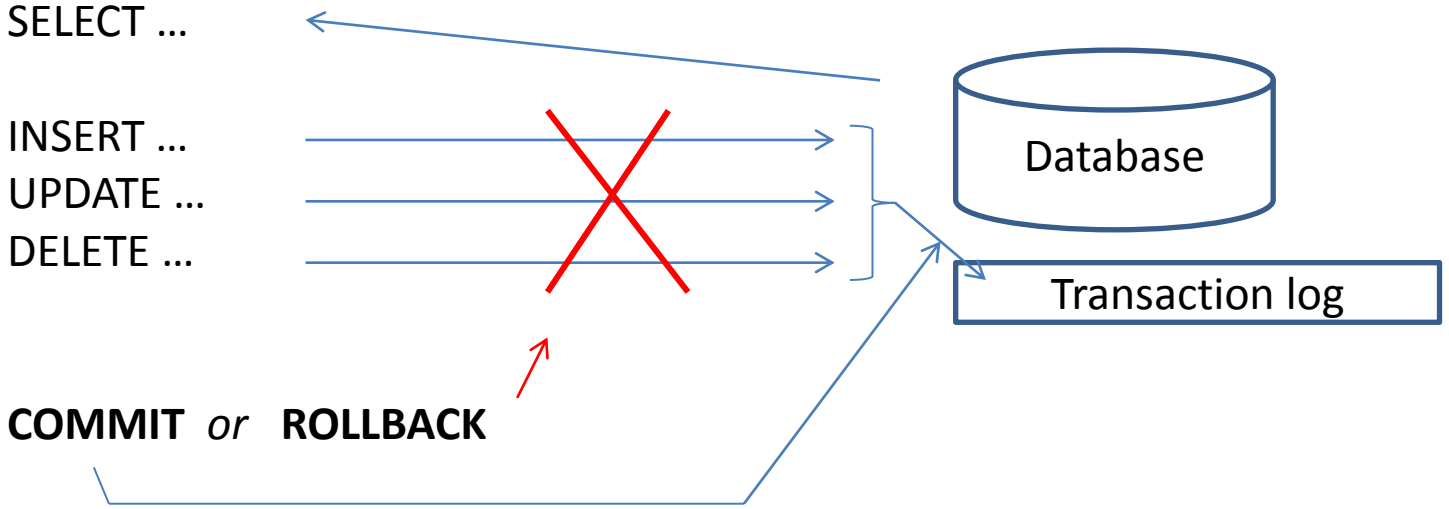
"Big Picture" for understanding COMMIT and ROLLBACK of SQL transactions

Files, Buffers, Listener, Service Threads, and Transactions

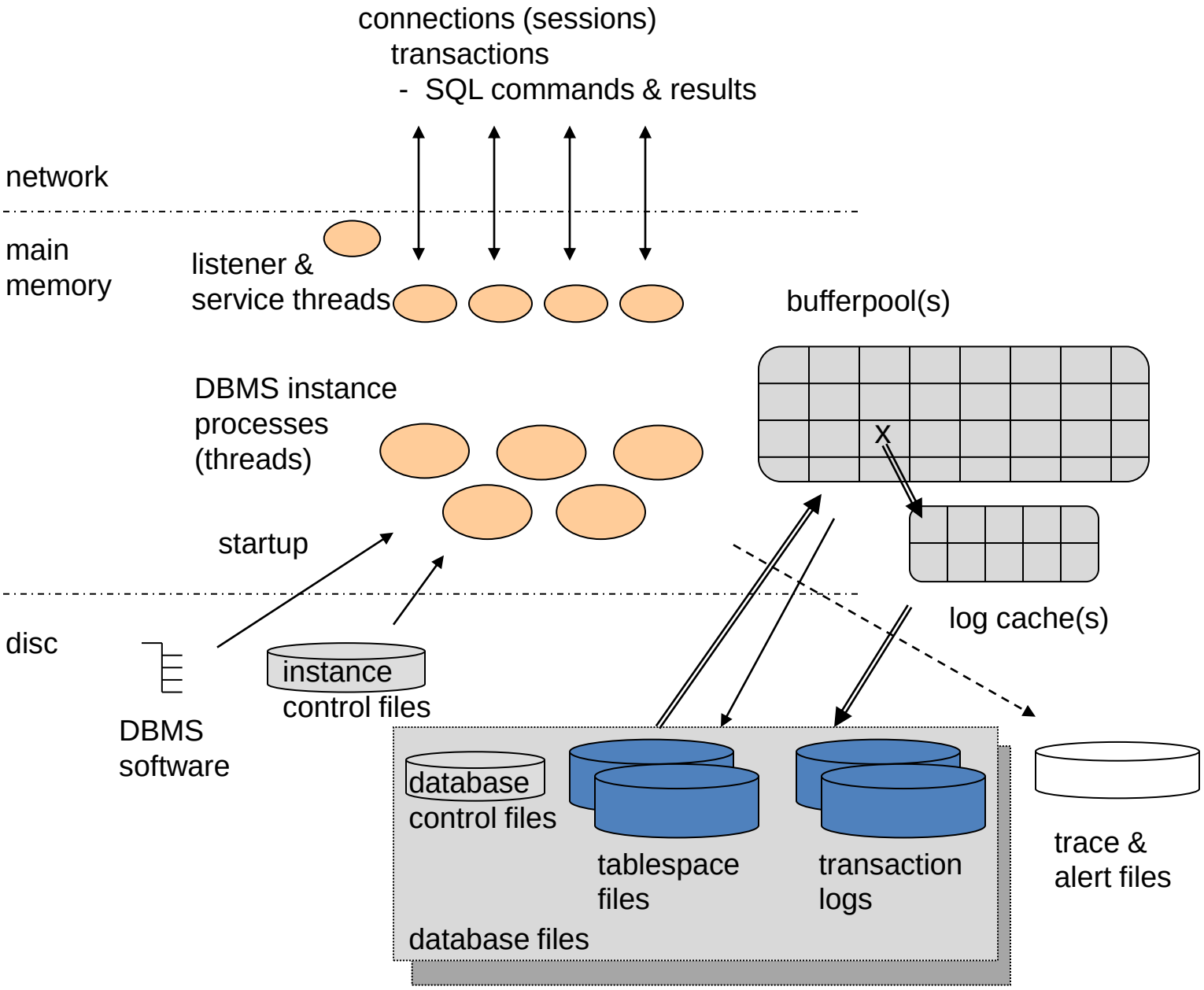
Martti Laiho 2012-04-02

(Flat) SQL Transaction

[BEGIN TRANSACTION]
[SET TRANSACTION ISOLATION ..]

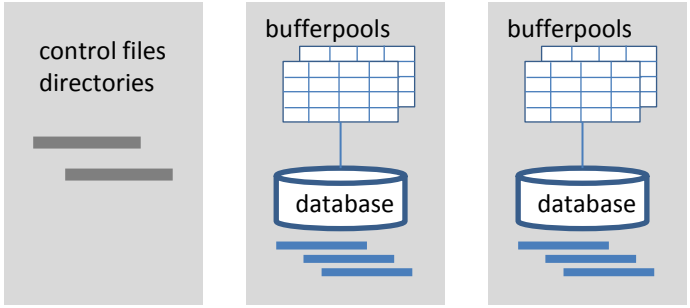


Overview of a Database Server Instance



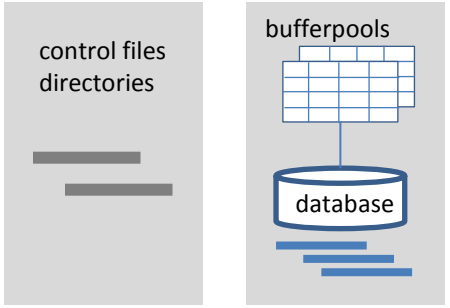
Database Instance Architectures

DB2 Instance



For every database the active transaction logs build a circular chain

Oracle Instance

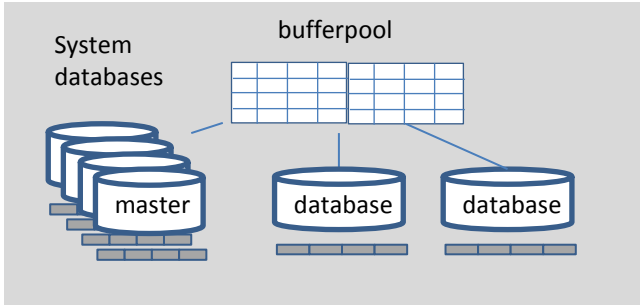


Oracle calls the chained active transaction logs as redo-logs

A server Instance may include just one or multiple databases depending on the DBMS product.

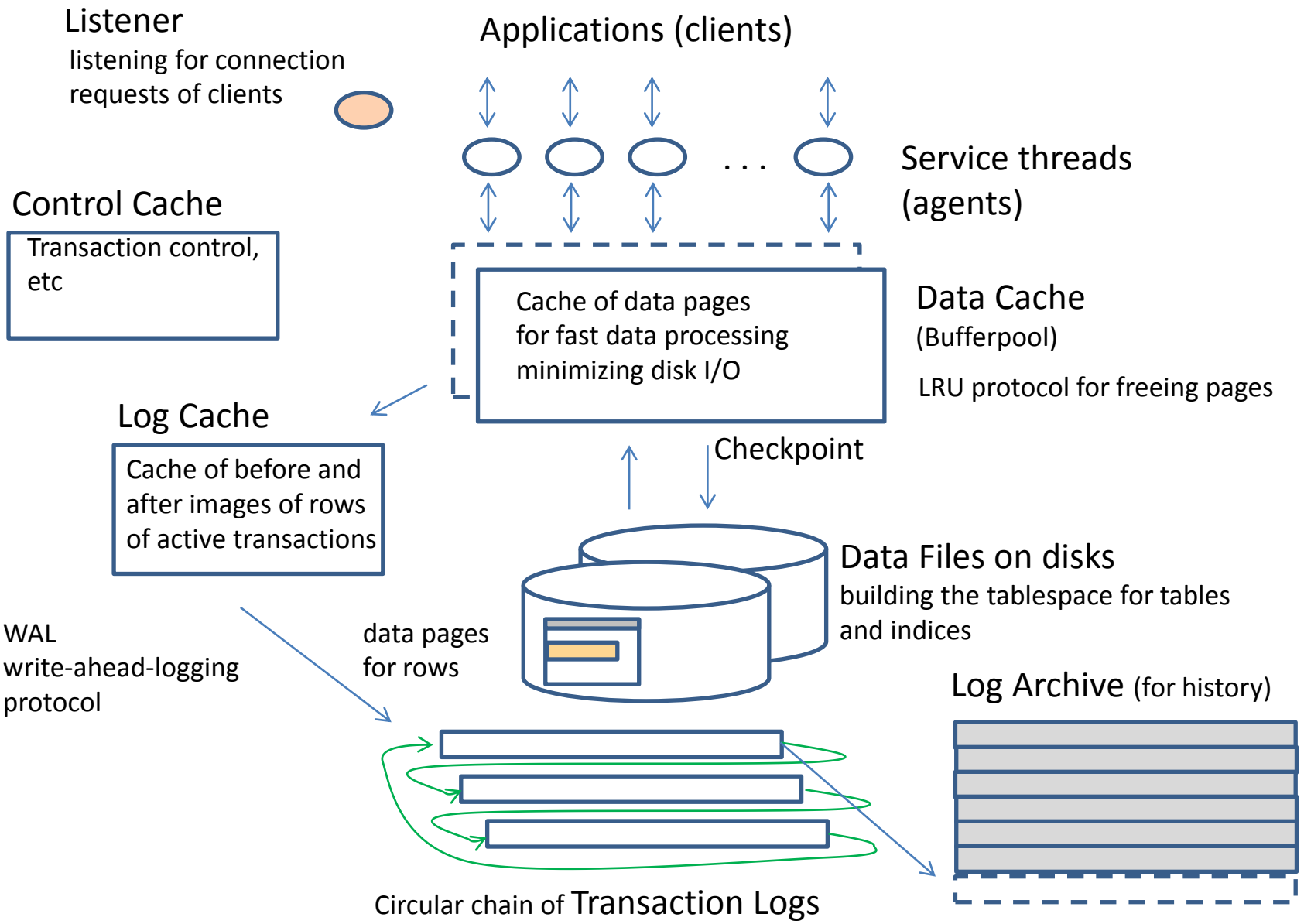
In the following we will focus only on the simple case of a single database

SQL Server Instance

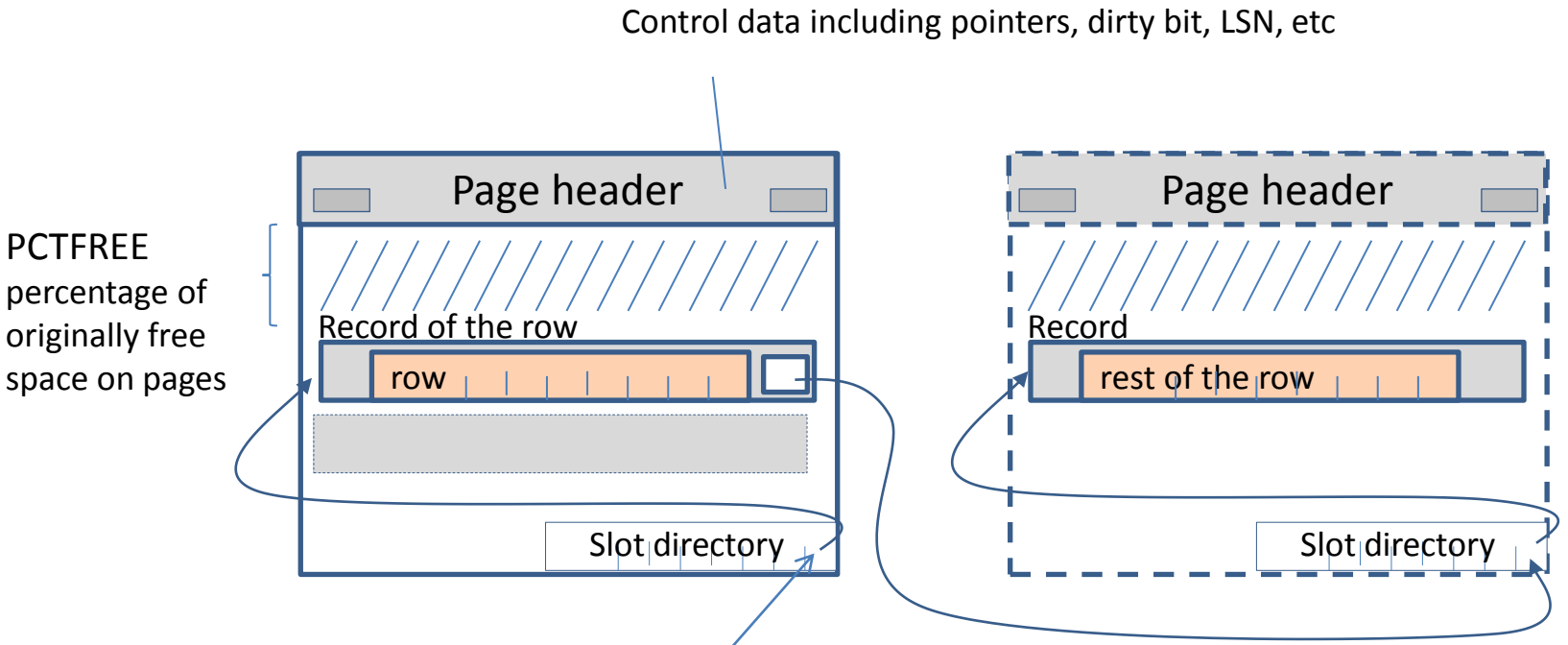


The active database transaction logs are allocated as Virtual Log Files in a single file building a circular chain

Files and Caches

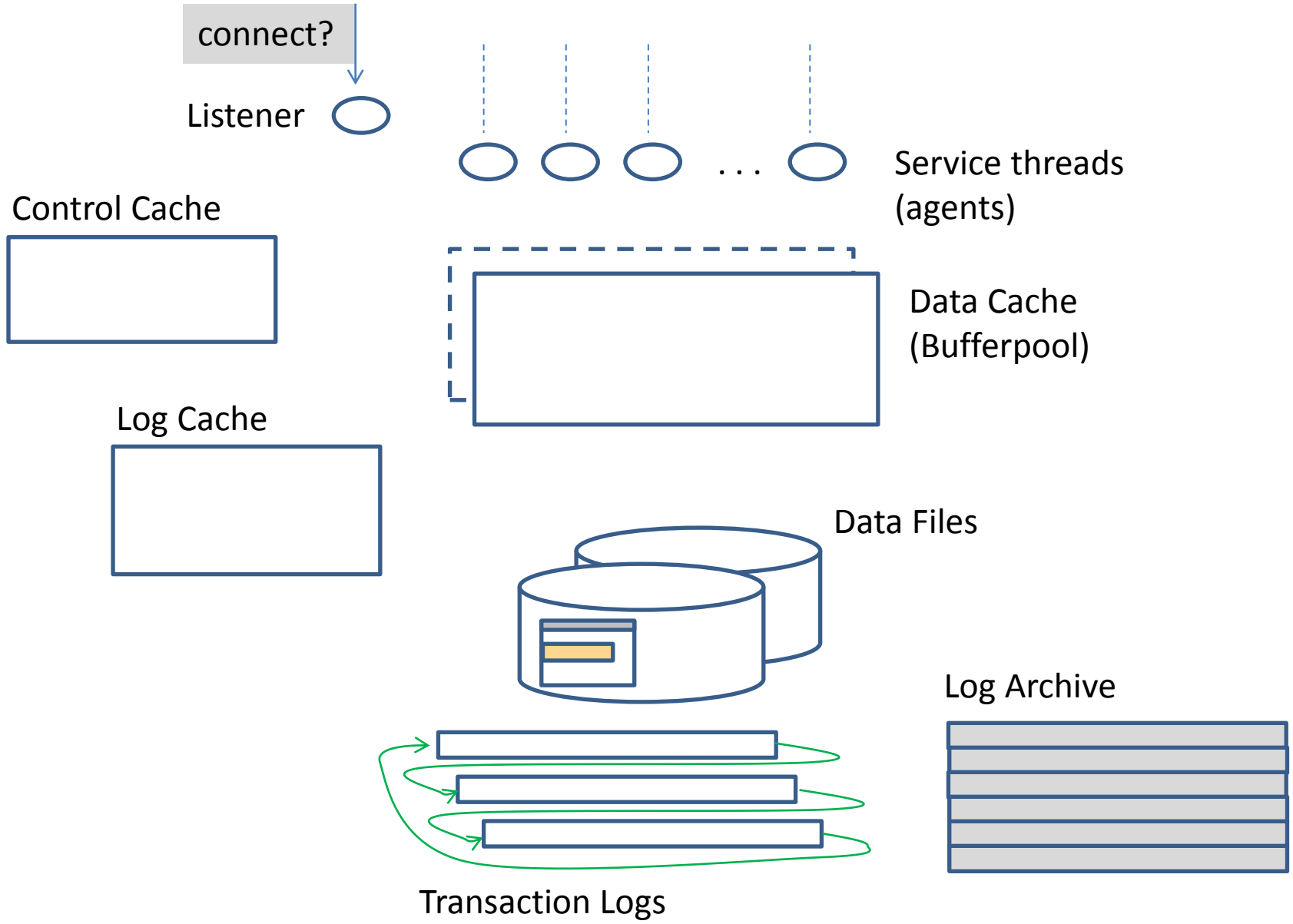


Structures of Data Pages

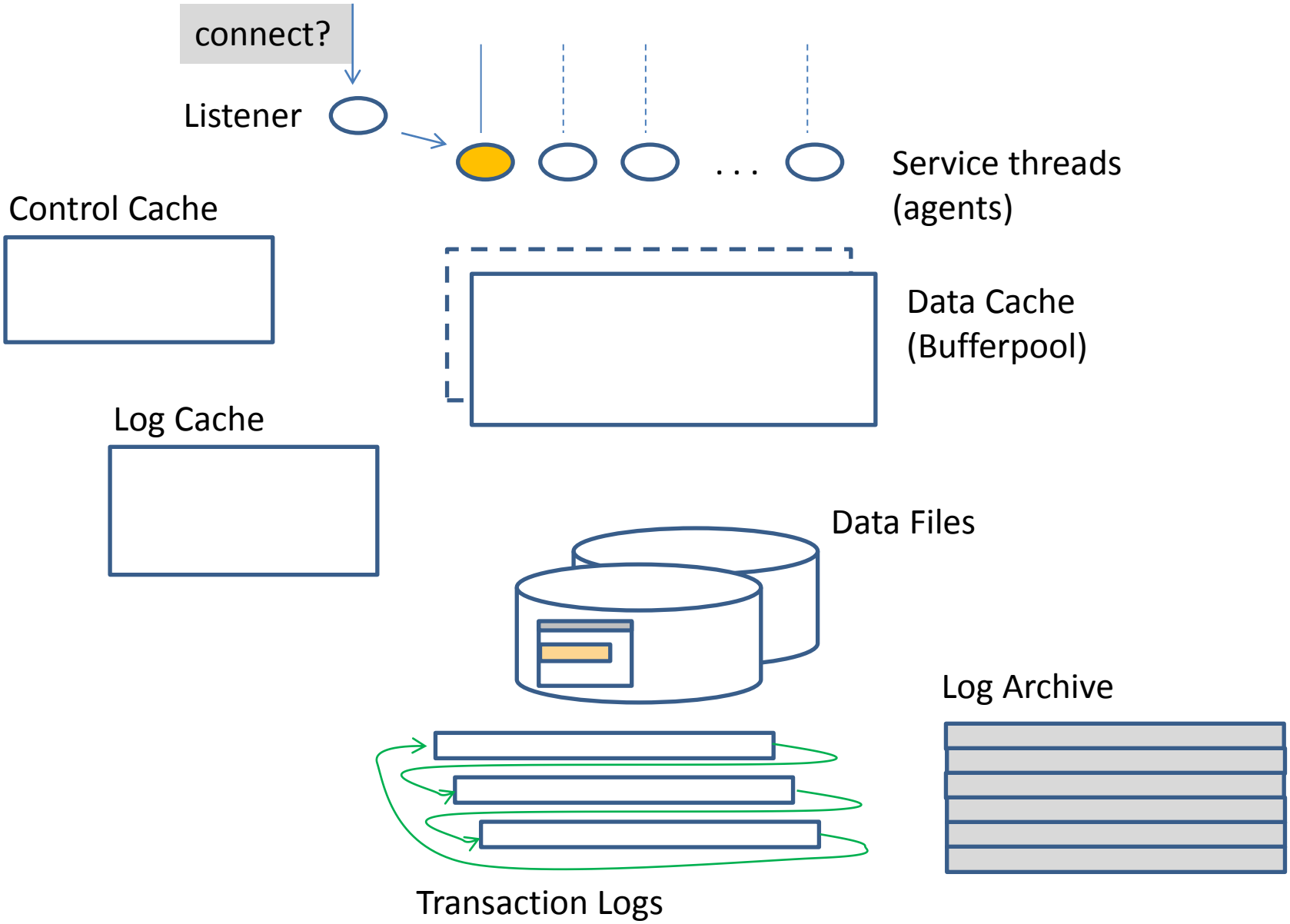


Every data file has an internal **file#** in the database and the file space is managed as a sequence of pages.
Page# is the ordinal number of the page in the file.
A typical page size nowadays is 4, 8, 16 or 32 KB.

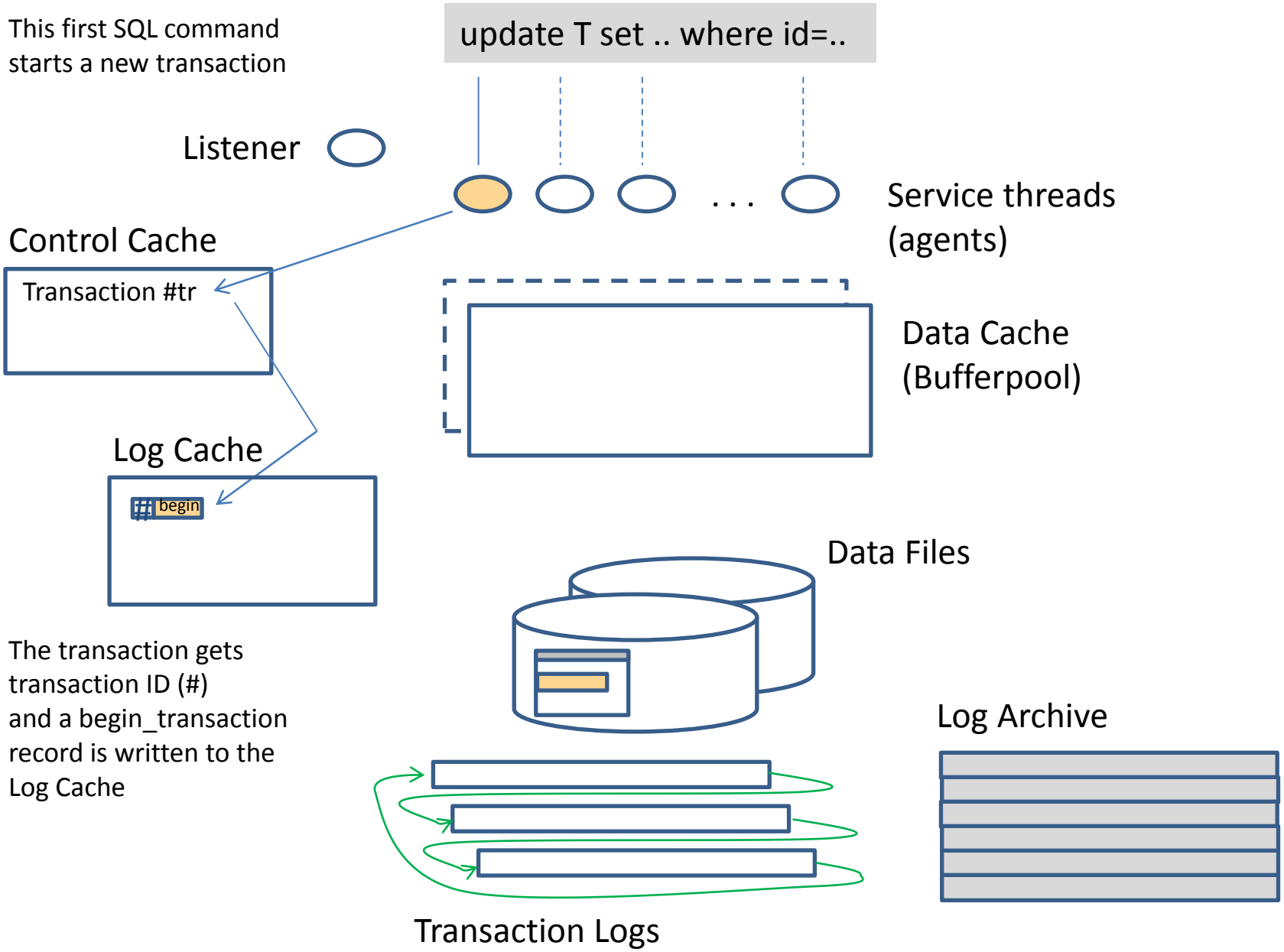
SQL Session: A client starts its SQL connection ..



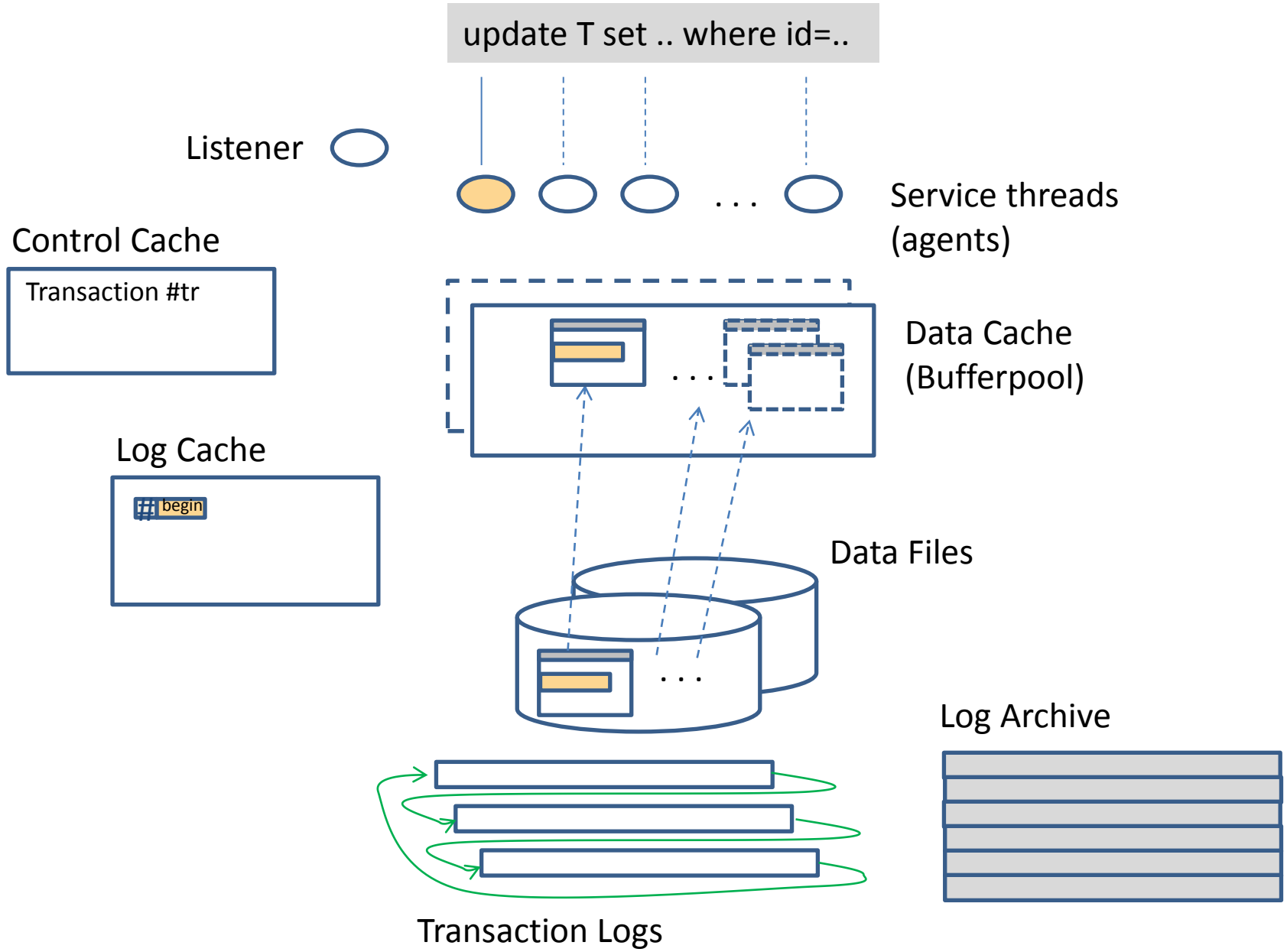
.. gets a service thread



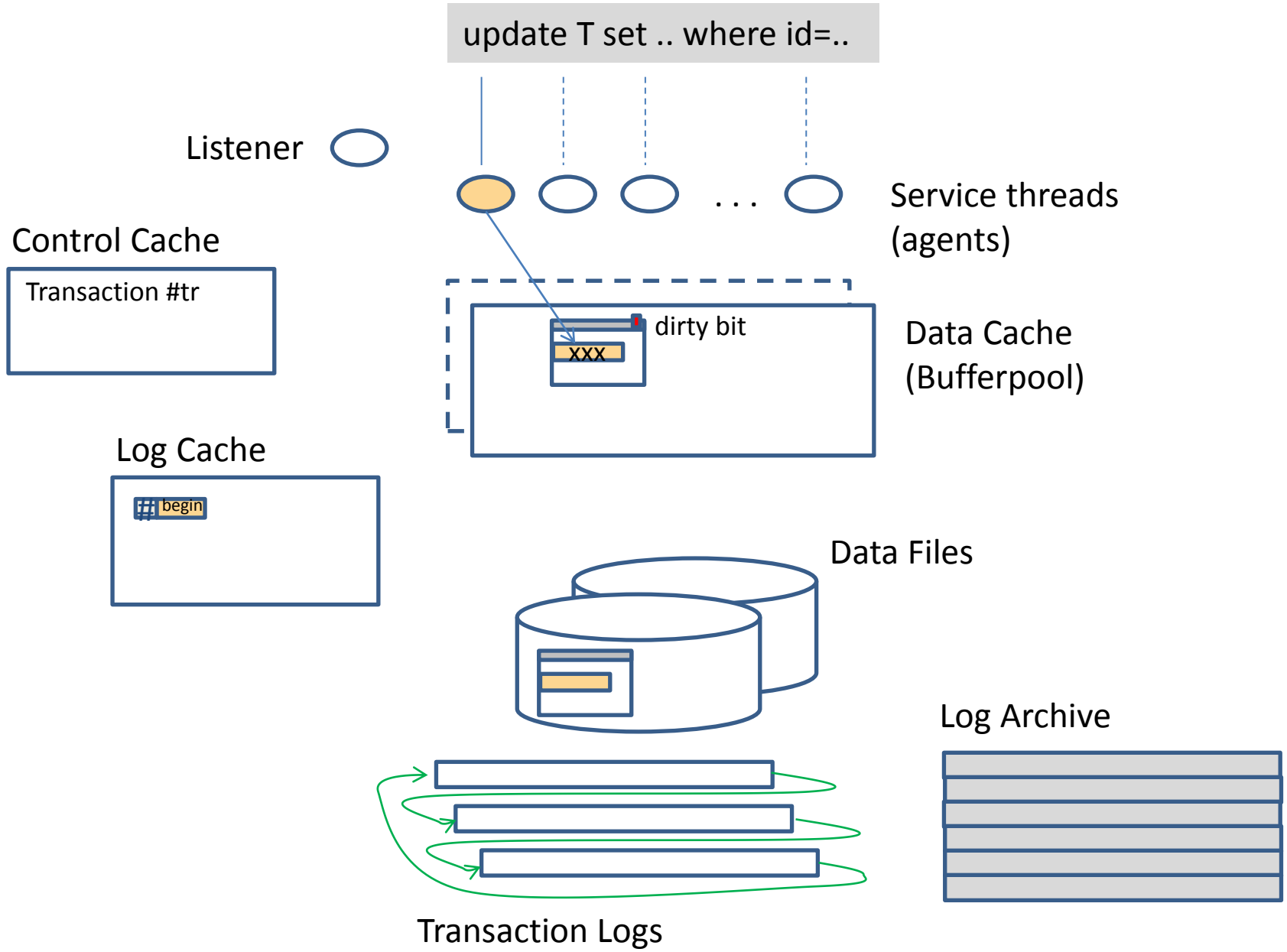
.. Client enters a SQL command to update a row (still on disk)



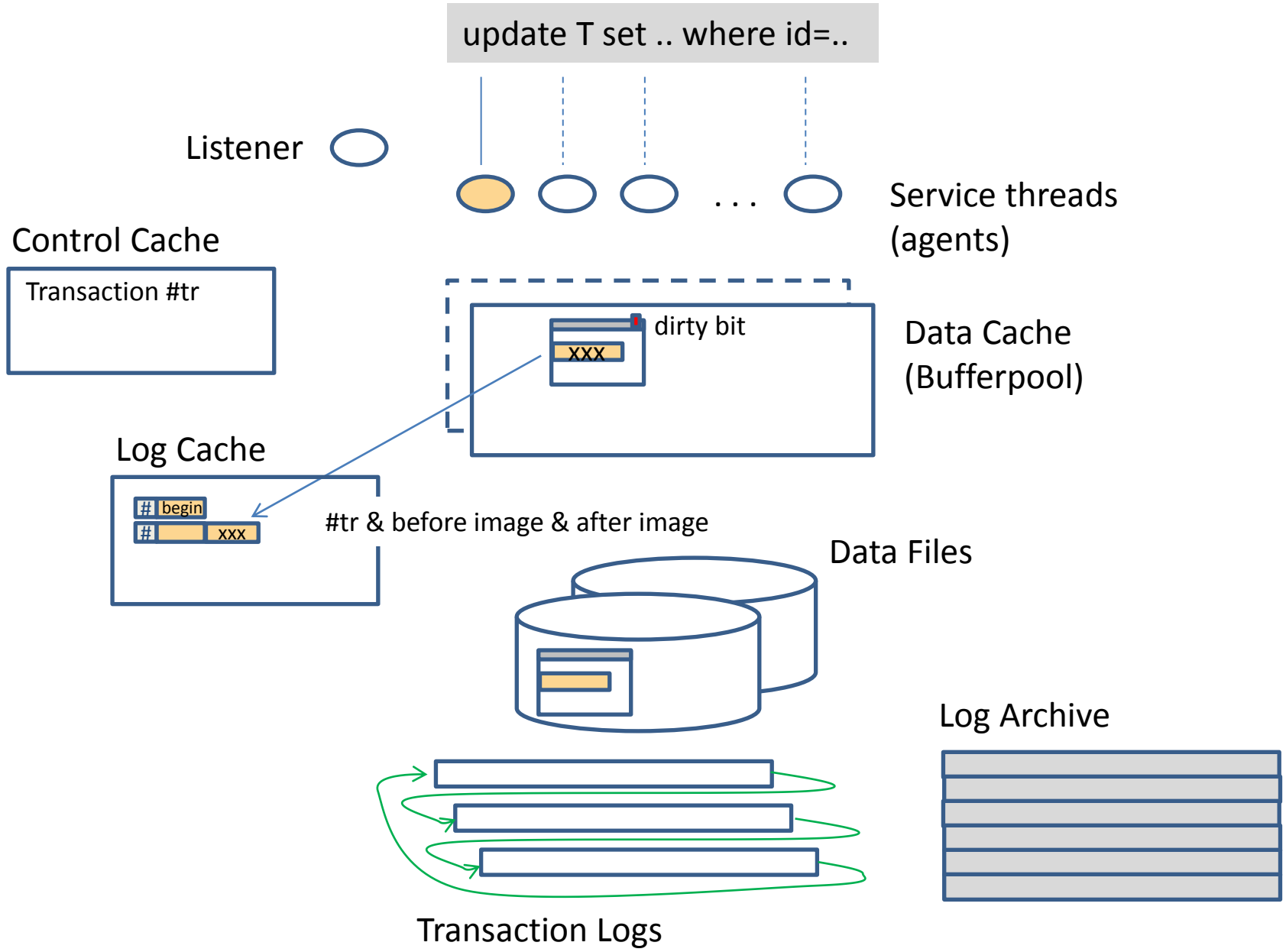
DBMS reads the page of the row to bufferpool



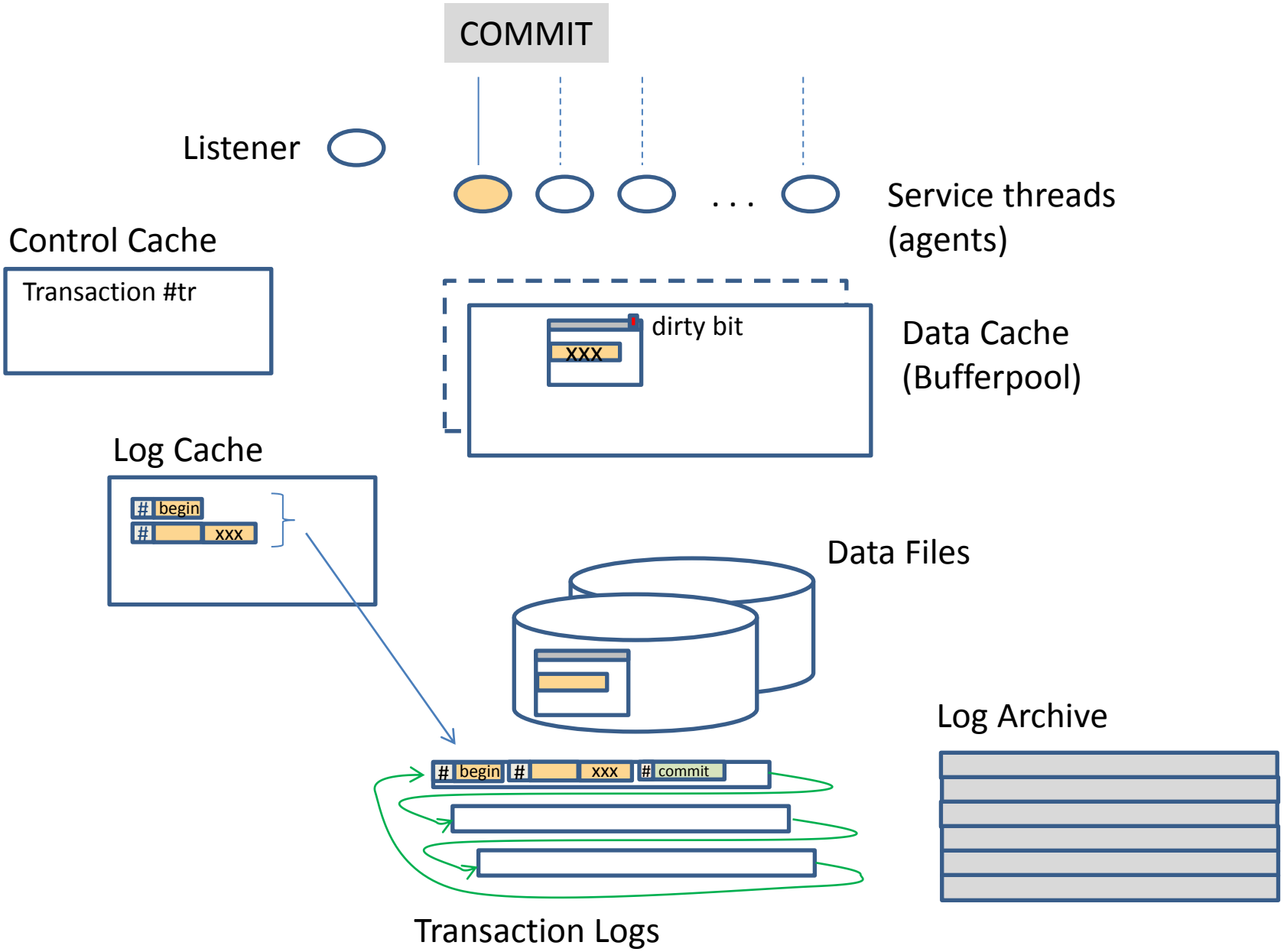
The row is updated and the updated page is marked with the Dirty Bit



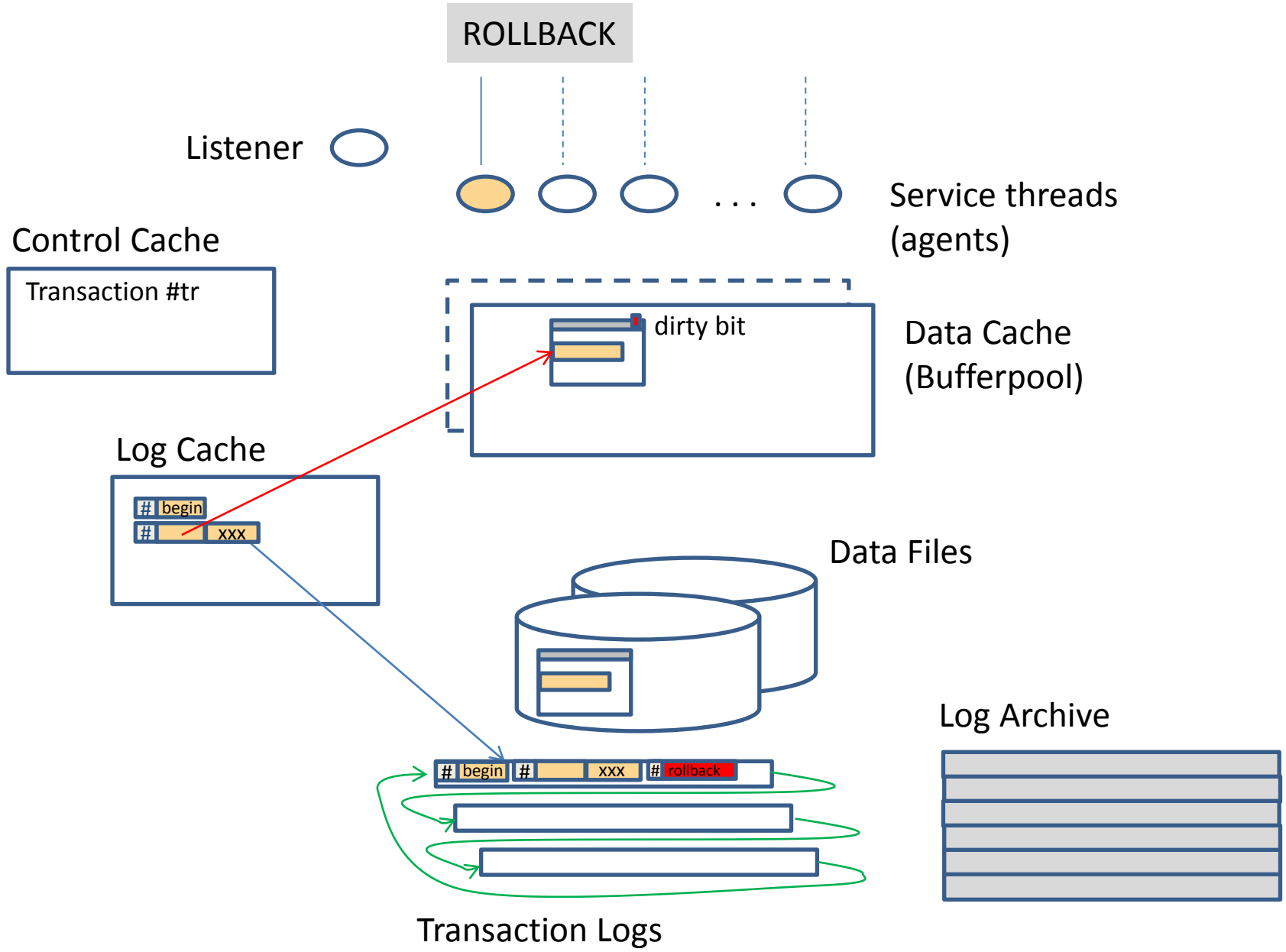
A log record of the before and after image of the row is written to log cache



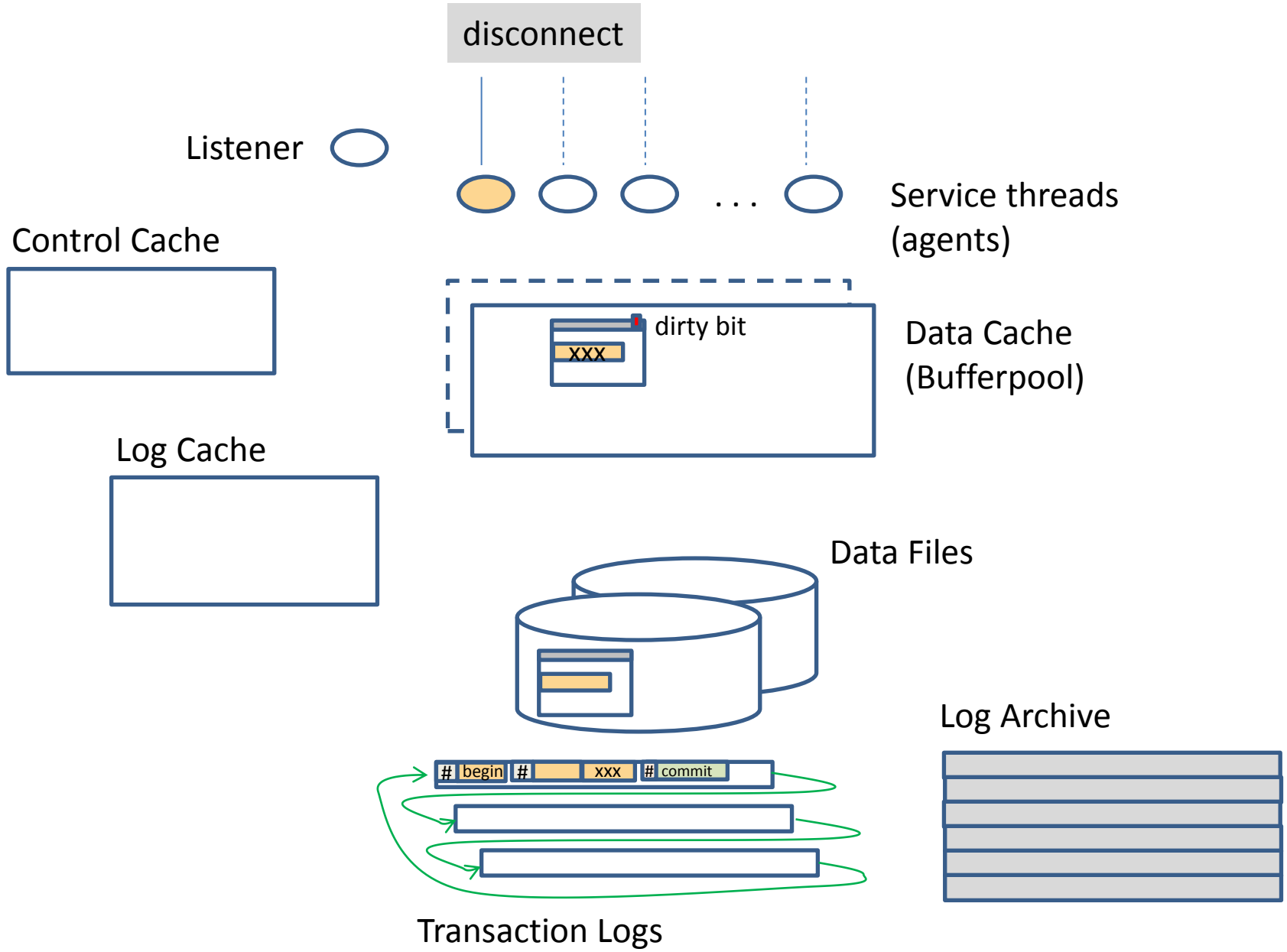
On commit the log records of the transaction are written to the transaction log



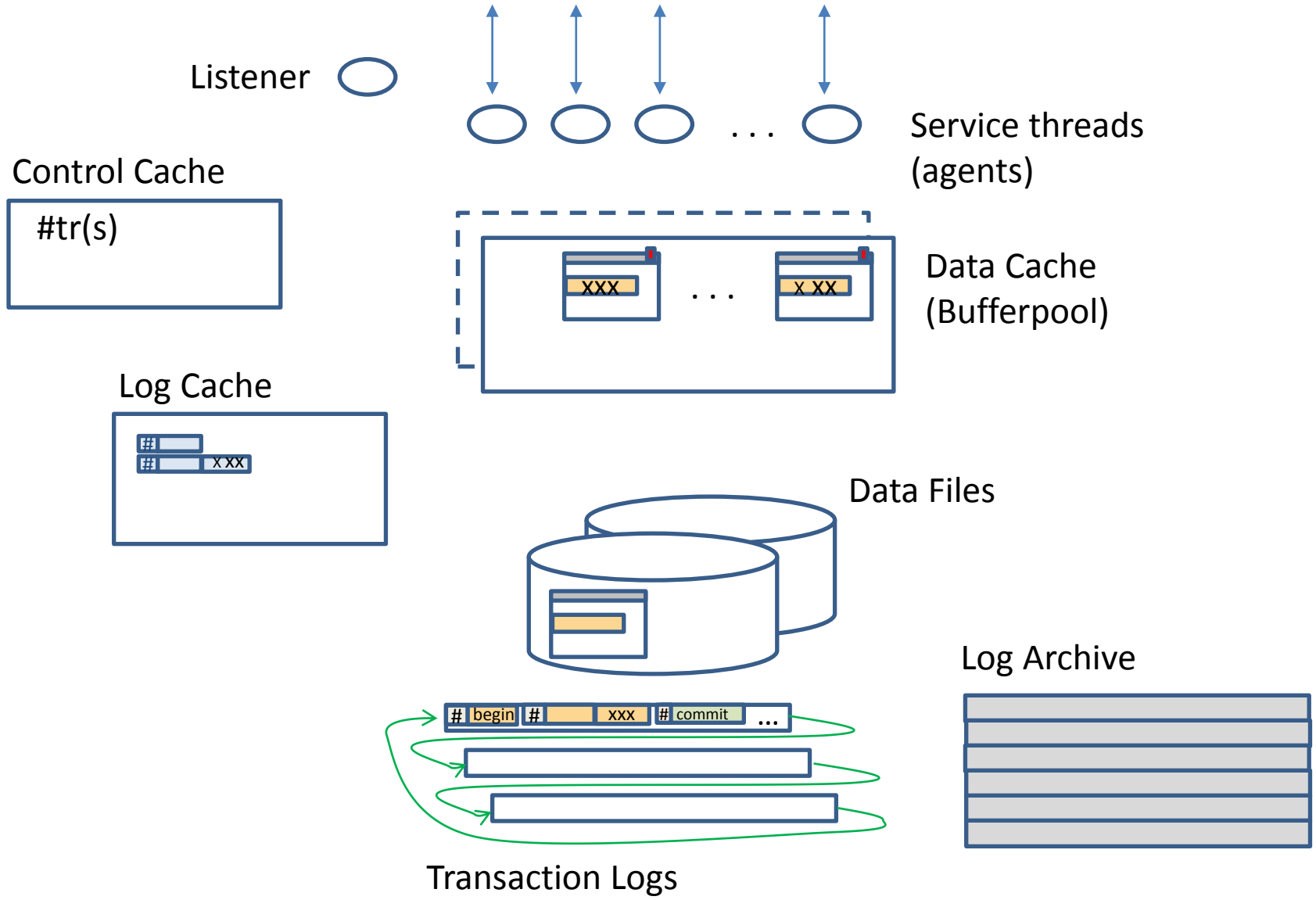
.. but in case of ROLLBACK the before images are returned to original addresses



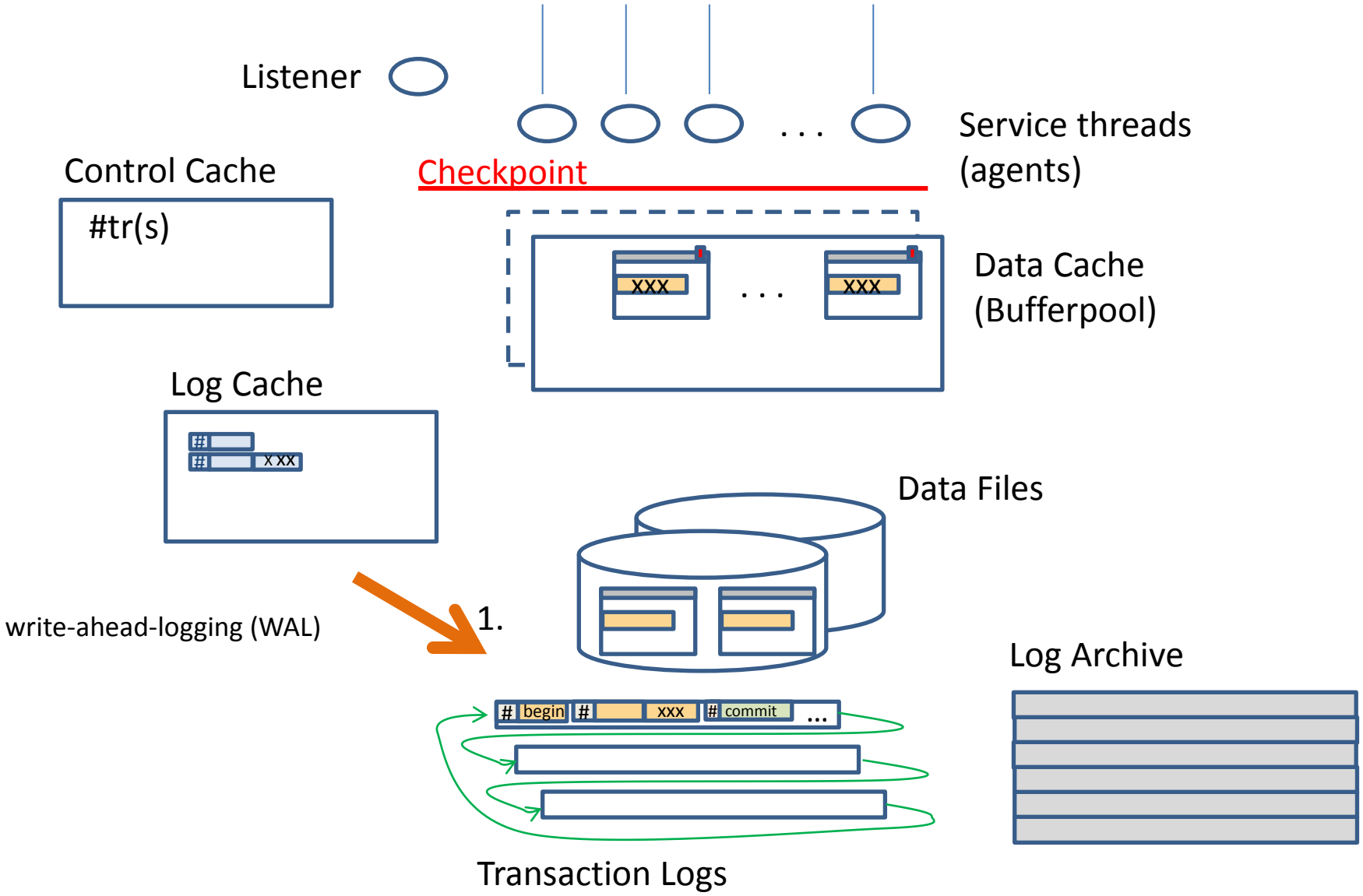
.. Later the client closes its SQL connection



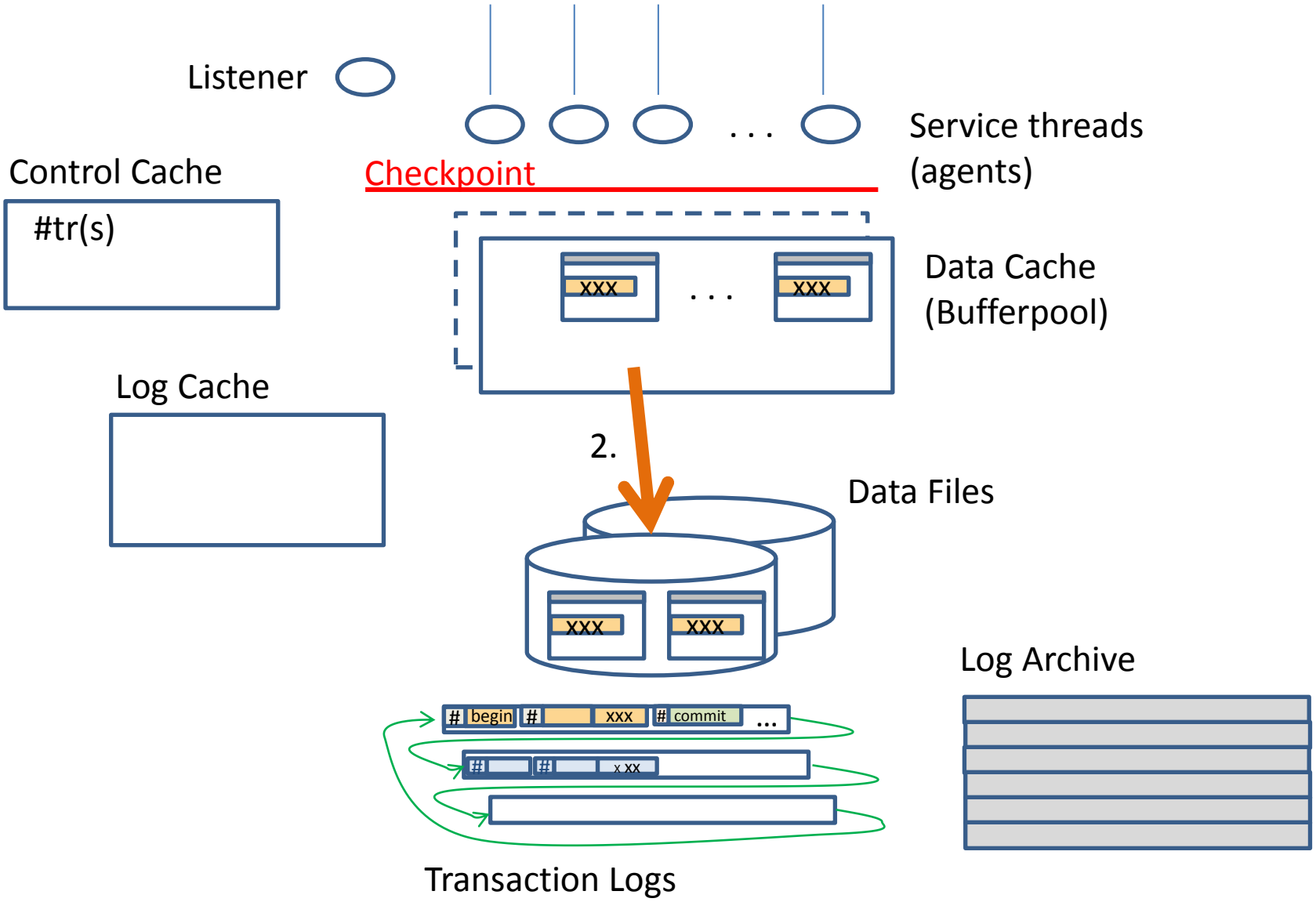
.. Some other pages may also get updated by SQL sessions of clients



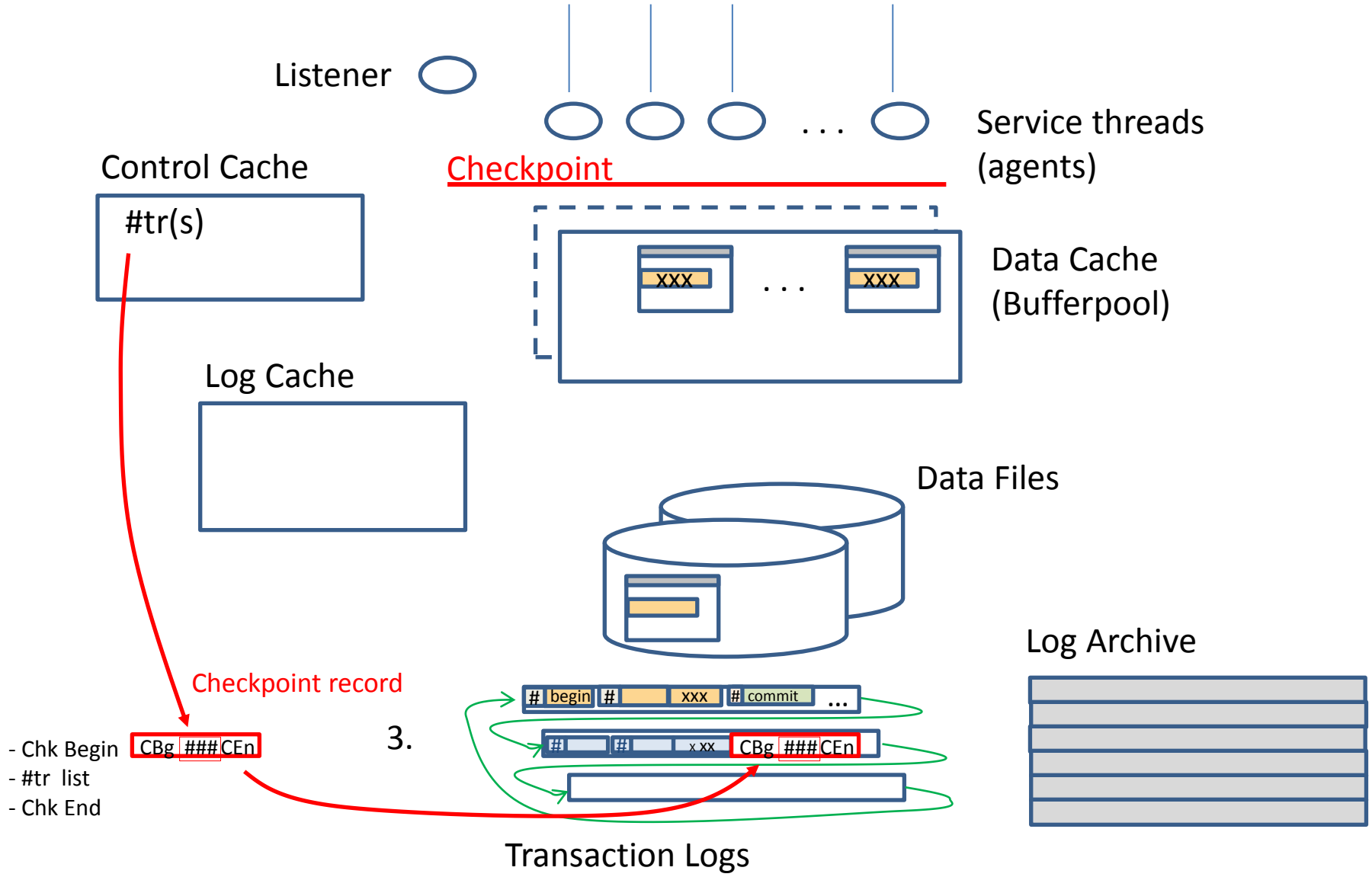
From time to time a checkpoint stops the services ..



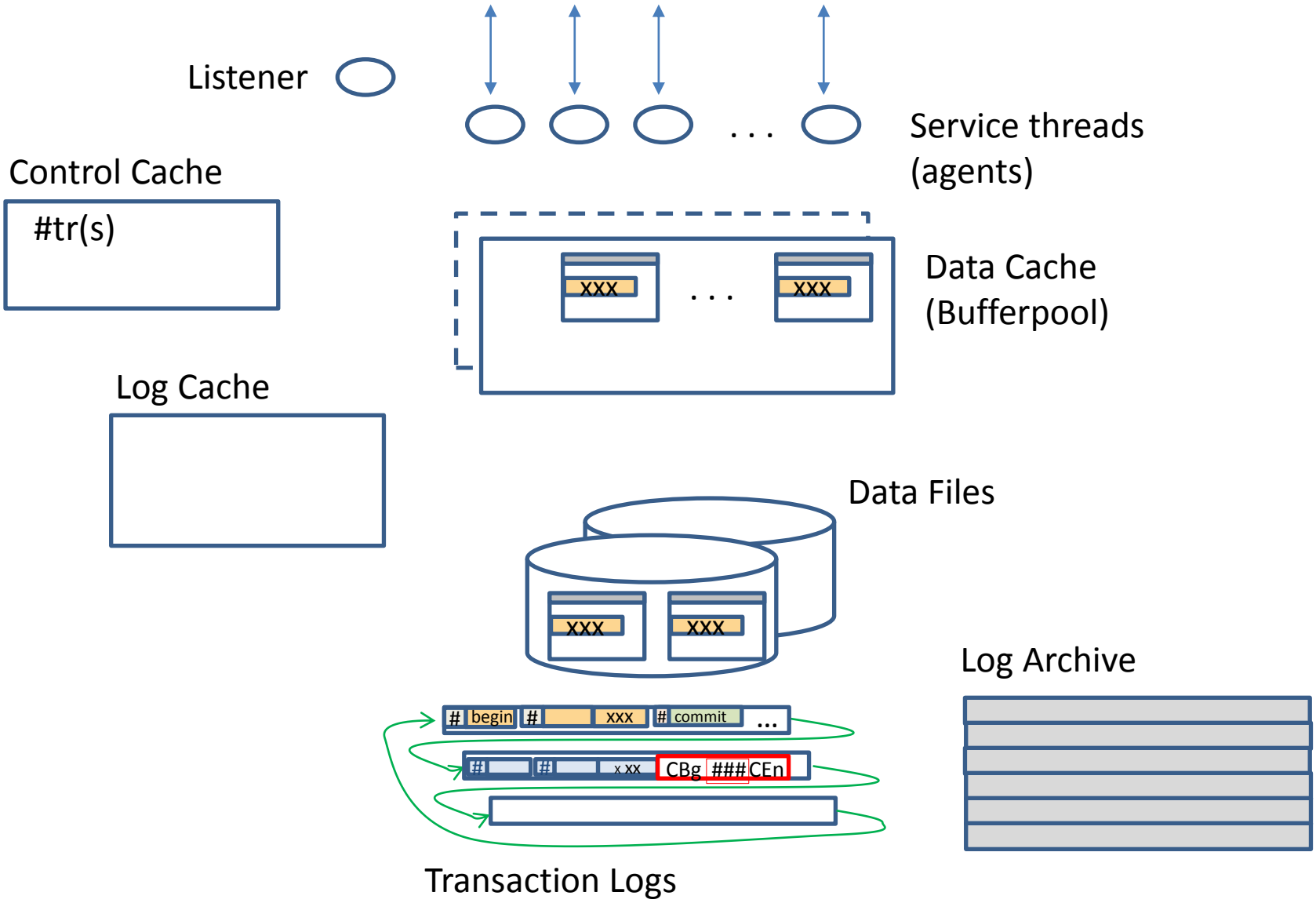
and all dirty pages are written to data files clearing the dirty bits



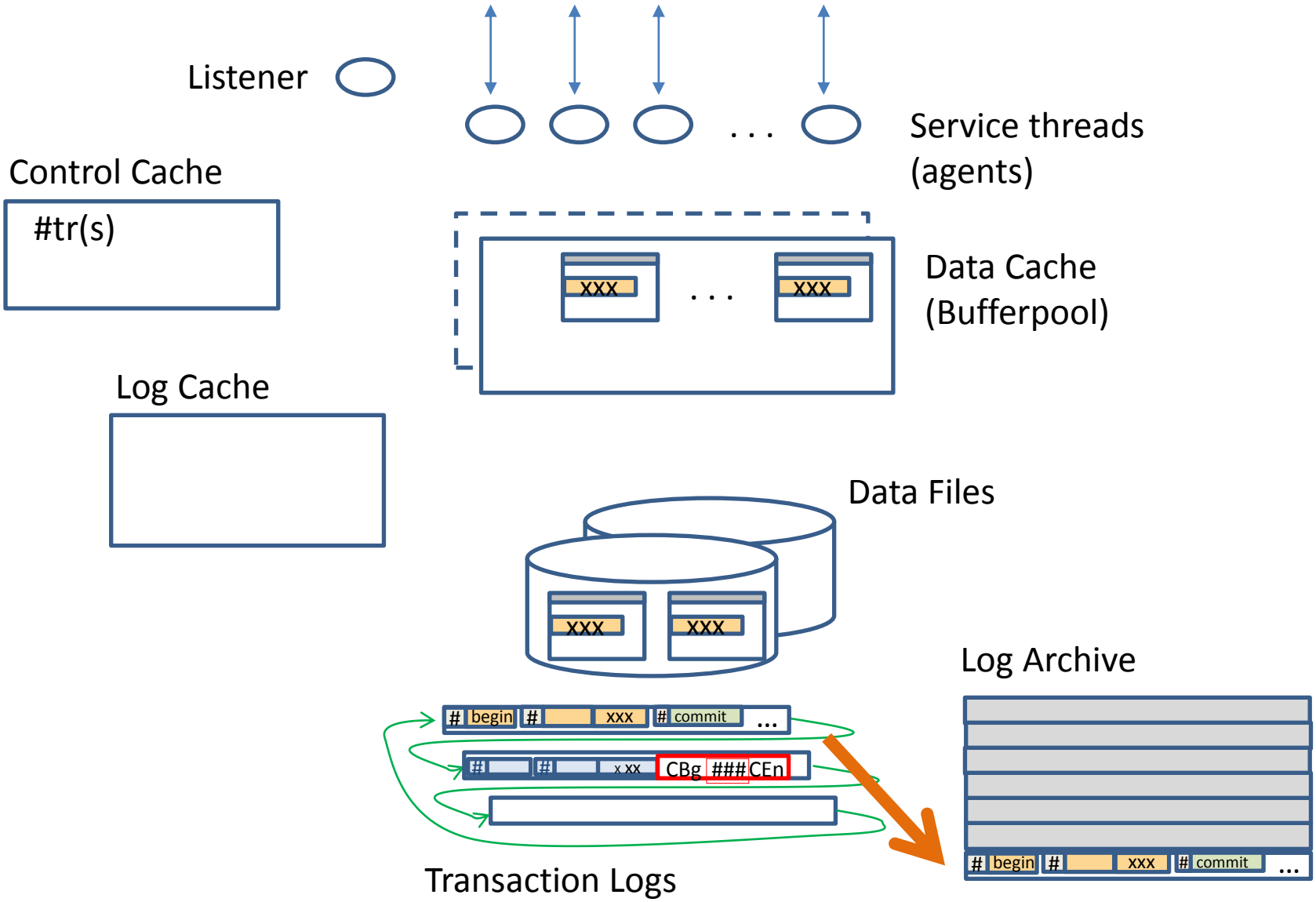
and writes checkpoint record



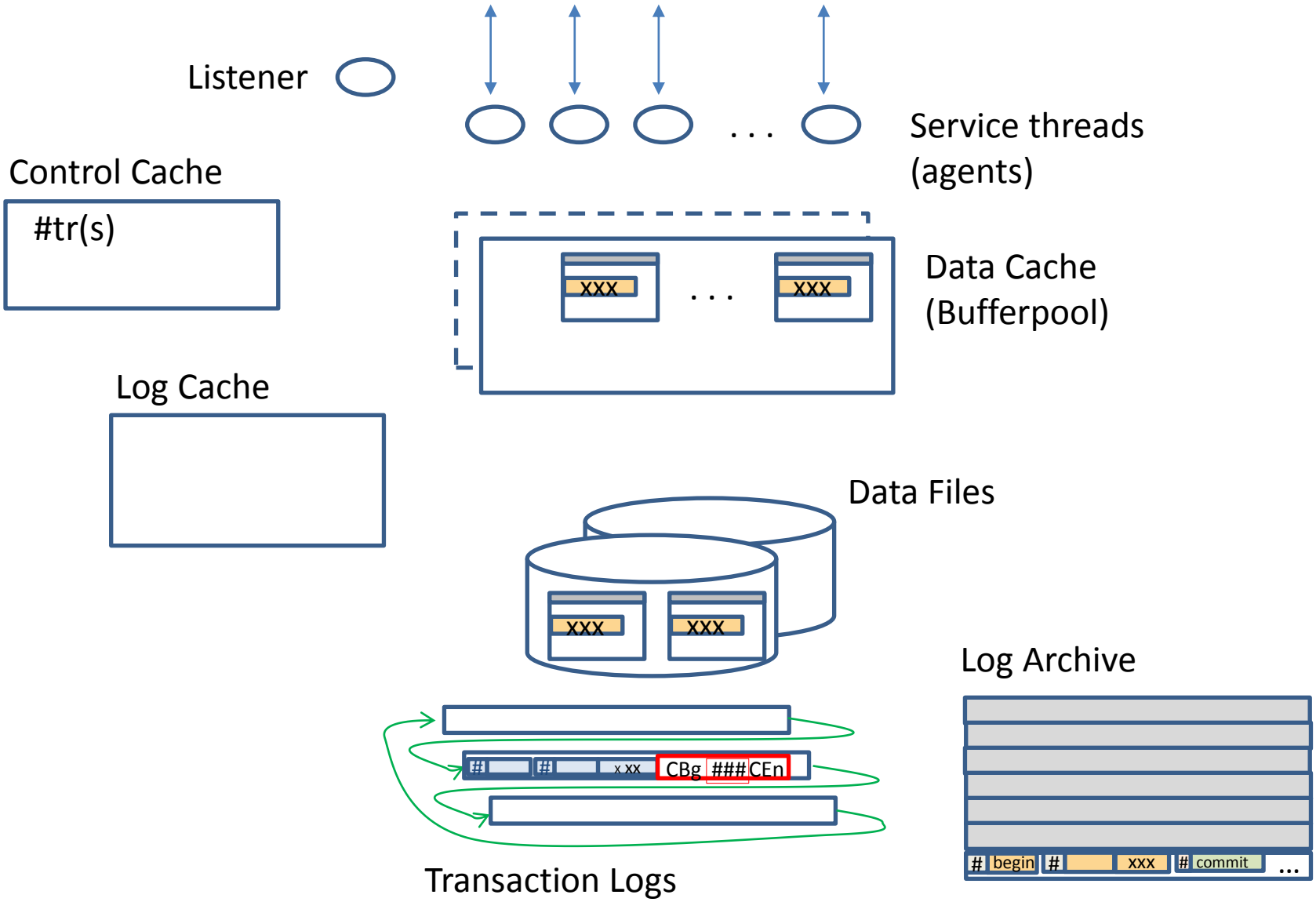
After the checkpoint the serving of clients can continue



Contents of full transaction log files are copied to archive ..



.. and the space of the copied transaction log file can be reused



A logging test using SQL Server 2012

based on the original exercise by Kari Silpiö

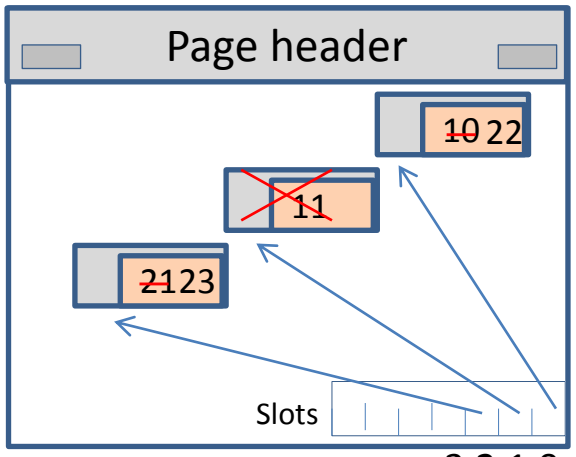
```
CREATE TABLE Test (col INT);
-- currently we have AUTOCOMMIT mode on
INSERT INTO Test VALUES (10);
INSERT INTO Test VALUES (11);
-- now we will change to transactional mode
BEGIN TRANSACTION;
INSERT INTO Test VALUES (21);
UPDATE Test SET col = 22 WHERE col = 10;
DELETE Test WHERE col = 11;
GO
-- forcing a checkpoint manually
CHECKPOINT;
GO
-- our transaction continues
UPDATE Test SET col = 23 WHERE col = 21;
ROLLBACK;
-- Now, let's see the transaction log
SELECT [Current LSN], [Previous LSN], Operation,[Num Transactions], SPID, [Transaction ID],
       AllocUnitName, [Page ID],[Slot ID],SUBSTRING([Begin Time],12,12) AS Time
FROM fn_dblog(NULL, NULL)  -- active log
WHERE [Slot ID] IS NULL OR AllocUnitName NOT LIKE 'sys.%';
```

21

10

11

21



Results | Messages

Current LSN	Previous LSN	Operation	Num Transactions	SPID	Transaction ID	AllocUnitName	Page ID	Slot ID	Time
00000020:00000114:0001	00000000:00000000:0000	LOP_BEGIN_XACT	NULL	54	0000:0000030a	NULL	NULL	NULL	18:11:49:713
00000020:00000114:0002	00000020:00000114:0001	LOP_INSERT_ROWS	NULL	NULL	0000:0000030a	dbo.Test	0001:0000004d	2	NULL
00000020:00000114:0003	00000020:00000114:0002	LOP_MODIFY_ROW	NULL	NULL	0000:0000030a	dbo.Test	0001:0000004d	0	NULL
00000020:00000114:0004	00000020:00000114:0003	LOP_DELETE_ROWS	NULL	NULL	0000:0000030a	dbo.Test	0001:0000004d	1	NULL
00000020:00000114:00a0	00000020:0000001b:002b	LOP_BEGIN_CKPT	NULL	NULL	0000:00000000	NULL	NULL	NULL	NULL
00000020:00000155:0001	00000000:00000000:0000	LOP_XACT_CKPT	1	NULL	0000:00000000	NULL	NULL	NULL	NULL
00000020:00000155:0002	00000000:00000000:0000	LOP_XACT_CKPT	0	NULL	0000:00000000	NULL	NULL	NULL	NULL
00000020:00000156:0001	00000020:00000114:00a0	LOP_END_CKPT	NULL	NULL	0000:00000000	NULL	NULL	NULL	NULL
00000020:00000157:0001	00000020:00000114:0004	LOP_MODIFY_ROW	NULL	NULL	0000:0000030a	dbo.Test	0001:0000004d	2	NULL
00000020:00000157:0002	00000020:00000114:0004	LOP_MODIFY_ROW	NULL	NULL	0000:0000030a	dbo.Test	0001:0000004d	2	NULL
00000020:00000157:0003	00000020:00000114:0003	LOP_INSERT_ROWS	NULL	NULL	0000:0000030a	dbo.Test	0001:0000004d	1	NULL
00000020:00000157:0004	00000020:00000114:0002	LOP_MODIFY_ROW	NULL	NULL	0000:0000030a	dbo.Test	0001:0000004d	0	NULL
00000020:00000157:0005	00000020:00000114:0001	LOP_DELETE_ROWS	NULL	NULL	0000:0000030a	dbo.Test	0001:0000004d	2	NULL
00000020:00000157:0006	00000020:00000114:0001	LOP_ABORT_XACT	NULL	NULL	0000:0000030a	NULL	NULL	NULL	NULL

Some Notes ..

- The ***Least Recently Used*** (LRU) procedure is needed when DBMS needs to recover free space in the bufferpool(s). The pages which have been unused for the longest time and have dirty bit reset, will be cleared for new pages to be read from the data files.
- The **active, circular transaction log files** are the most important files of a database, since they contain data of the latest committed transactions! Dual logging (replicated) is recommended for these, and on dedicated disks.
- Based on the last checkpoint record and the circular transaction log a DBMS can recover the database after a **soft crash** automatically by ***roll-back recovery*** of failed and ***roll-forward recovery*** of the committed transactions to the level of the latest committed transasction before the soft crash.
- In case of **hardware crash** the database content need to be **restore**d from the **database backup** and the applying roll-forward recovery of the transaction history from the archive and the latest circular logs, and finally rollback recovery of the latest failed transactions.
- These topics are covered in separate tutorials.