Sample Course Materials

Inside Berkeley DB

Don Anderson

Berkeley DB consultant
dda@ddanderson.com

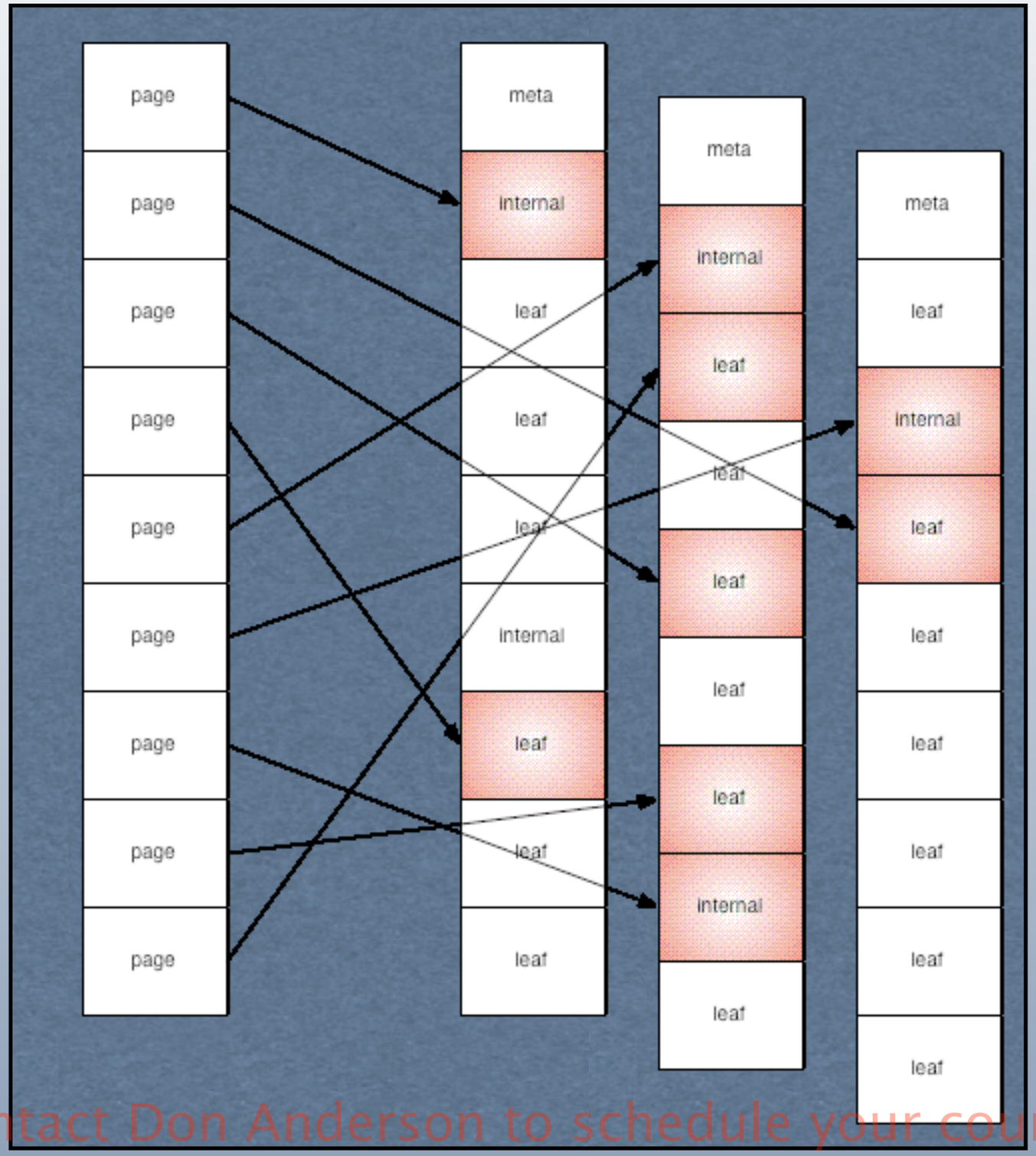
Btree Internal nodes

```
(header)
12:          4084       (first leaf)
14:          4000       (Idaho)
16:          4036       (Maine)
18:          4056       (North Carolina)
20:          3980       (South D)
22:          4020       (Wi)
24:          ♦ ♦
       ...
       ...
       ...
3980:  0  7   1  0   0 0 0 7   0 0 0 0   S o u t h   D ♦
4000:  0  5   1  0   0 0 0 6   0 0 0 0   I d a h o ♦ ♦ ♦
4020:  0  2   1  0   0 0 0 5   0 0 0 0   W i ♦ ♦
4036:  0  5   1  0   0 0 0 4   0 0 0 0   M a i n e ♦ ♦ ♦
4056:  0 14   1  0   0 0 0 3   0 0 0 0   N o r t h   C a r o l i n a ♦ ♦
4084:  0  0   1  0   0 0 0 2   0 0 0 0
```

Memory pool acts like VM

Memory pool

Database files



No hidden threads

- Memory cache is synchronous, though you can create your own *trickle thread*.
- Locking and deadlock detection generally synchronous but you can have a *deadlock detection thread*.
- Logging is synchronous though you can create your own *checkpoint thread*.
- Transactions are synchronous.
- Replication is synchronous. Exception is the *repmgr*, the higher level replication framework.

Exercise: show all entries

```
Exercise 03 (03_showall/ex03.c):

Create a cursor and use it to iterate through
the database, showing each entry.
Use show_film() to print each entry.

Extra credit: is there anything odd about the order
items are shown?  How might you fix that?
```

Show all entries solution (part 1)

```
DBC *cursor;
            ...

/*
 * Create a cursor
 */
if ((ret = filmdb->cursor(filmdb, NULL, &cursor, 0)) != 0) {
    fprintf(stderr, "DB->cursor failed: %sn", db_strerror(ret));
    exit(1);
}
            ...
```

Show all entries solution (part 2)

```
/*
 * Iterate using the cursor
 */
while ((ret = cursor->get(cursor, &keydbt, &valdbt, DB_NEXT)) == 0) {
    show_film(&key, &val);
}

/*
 * DB_NOTFOUND is a normal return, it breaks us out of the loop above.
 * Any other return is an error, and also exits the loop above.
 */
if (ret != DB_NOTFOUND) {
    fprintf(stderr, "DBC->get failed: %sn", db_strerror(ret));
    exit(1);
}

if ((ret = cursor->close(cursor)) != 0) {
    fprintf(stderr, "DBC->close failed: %sn", db_strerror(ret));
    exit(1);
}
```

Show all entries solution (part 3)

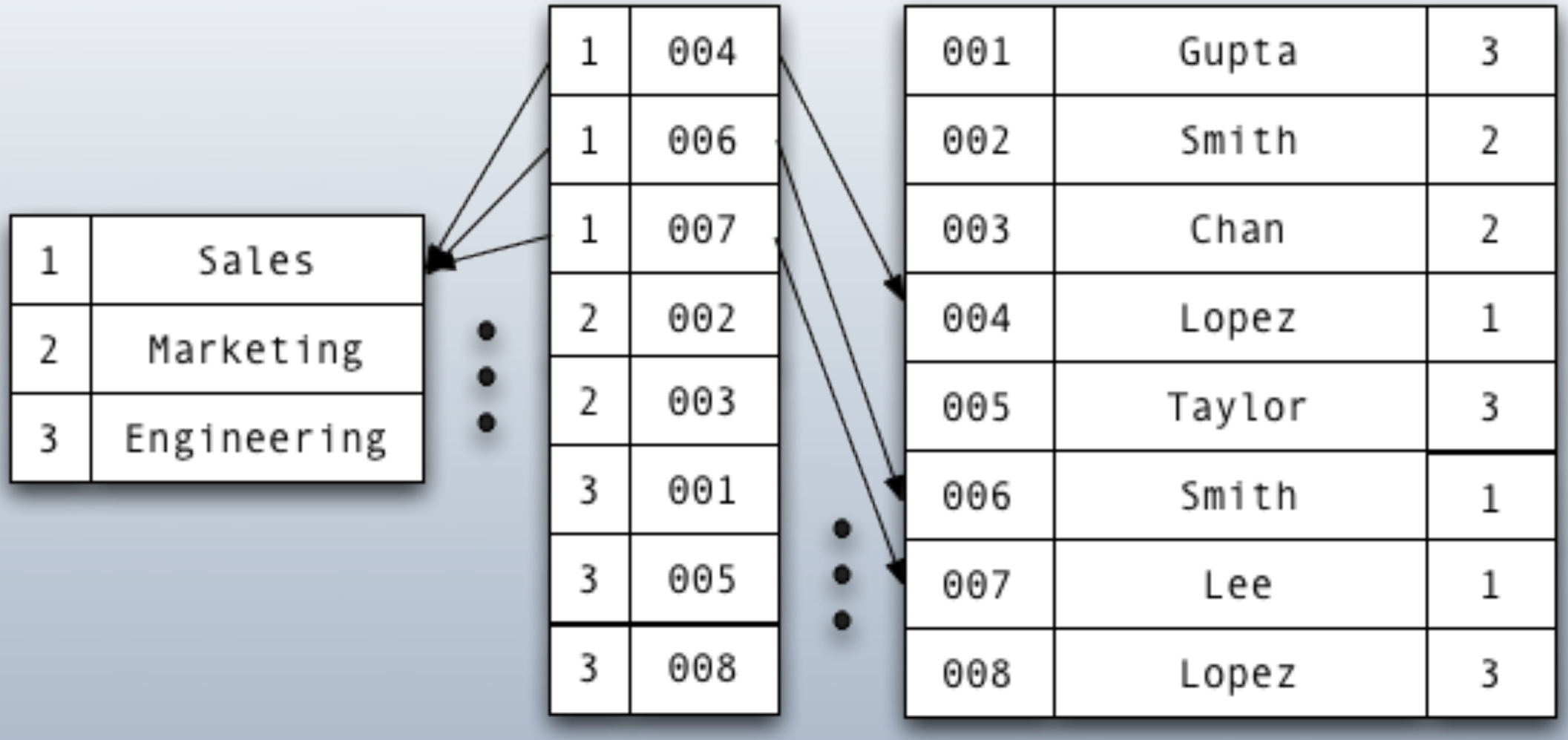
```
#include <netinet/in.h>    /**** defines htonl(), ntohl() ****/
                ...

    /**** one way to fix the byte ordering ****/
    printf("Film %d:n", ntohl(keyp->filmid));
                ...

    /**** one way to fix the byte ordering ****/
    key.filmid = htonl(filmid);
                ...
```

This is just one approach

Foreign Keys



1	Sales
2	Marketing
3	Engineering

1	004
1	006
1	007
2	002
2	003
3	001
3	005
3	008

001	Gupta	3
002	Smith	2
003	Chan	2
004	Lopez	1
005	Taylor	3
006	Smith	1
007	Lee	1
008	Lopez	3

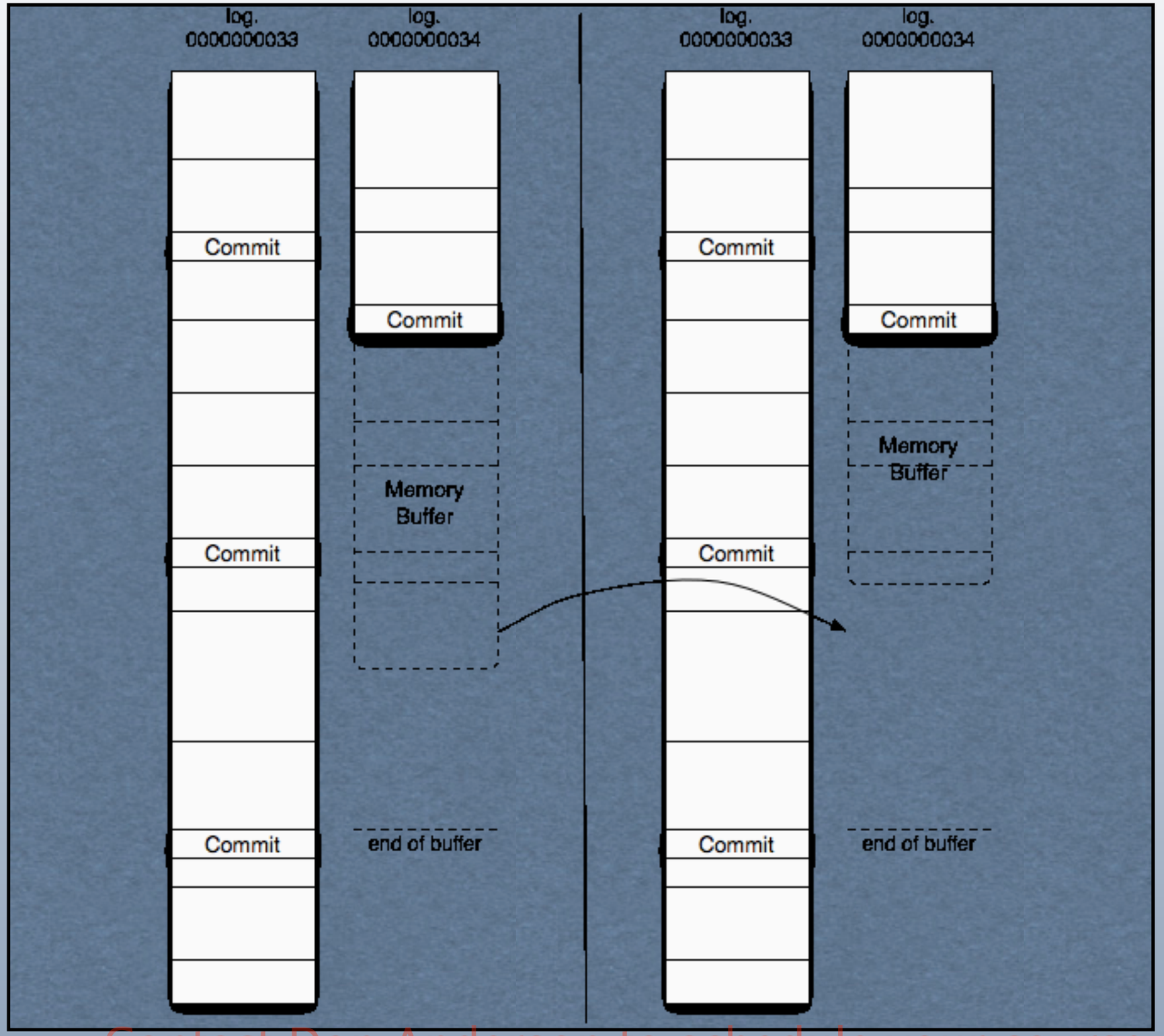
Adds a constraint at the cost of a lookup

What's in an env directory?

```
__db.001 __db.rep.diag00   census-2010.db   log.0000000001
__db.002 __db.rep.diag01   lastname.db      log.0000000002
__db.003 __db.rep.egen     by-state.db      log.0000000003
__db.004 __db.rep.gen      by-education.db  log.0000000004
```

Log files
Data files
Environment files
Replication Metadata
Snapshot 'freezer' files

Logging in BDB - replication



✔ Logging performance

- put log file on separate disk (databases can go an separate disks, too
- lg_bsize (log buffer size)
- lg_max (max size of log files)
- DB_DIRECT_LOG (if available with OS)
- Do replication (!) with DB_TXN_NOSYNC
- DB_LOG_IN_MEMORY (if non-durable, or with replication)