# Project

Zhengwei QI

2013.12

# 社交网络和大数据时代

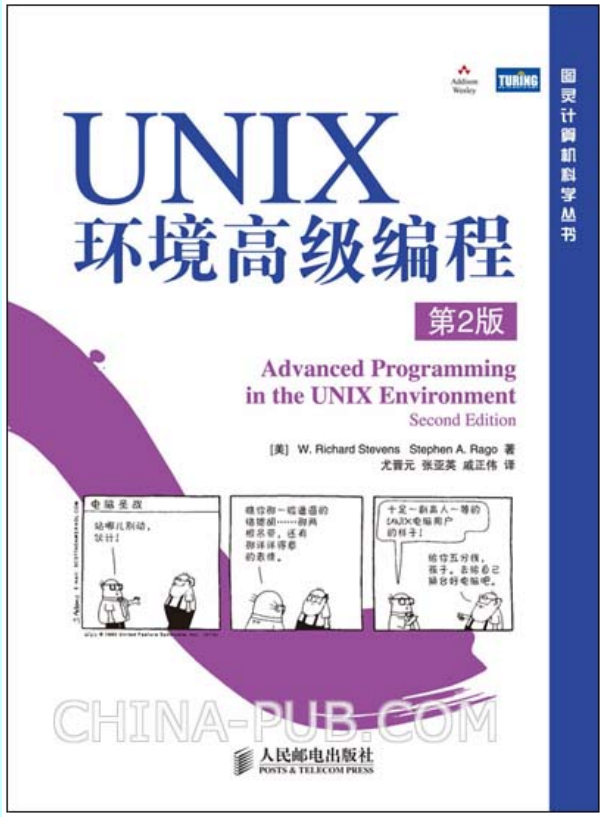
- 【2012天猫双11】开始第一分钟，超过1000万人涌入天猫！
  - 10分钟后，天猫和淘宝总交易金额就突破了2.5亿元人民币
  - 37分钟时，交易量突破10亿
  - 截至2：19分，支付宝总交易额超33.7亿！刷新去年天猫双11全天支付宝交易额。

*淘宝系统在狂欢节开始10分钟之后就出于瘫痪状态，不仅是支付页面打不开，连已买到的商品也无法查看。*

# 淘宝大数据

- 注册用户已达3.7亿，千万级卖家
- 2000多个维度，结构化和非结构化数据
- 日交易400万笔，日新增交易数据达10TB
- 典型应用：淘宝卖家的数据分析和信用评级

# 文件型数据库




dbm（3）是一个在UNIX系统中很流行的数据库函数库，它由Ken Thompson开发，使用了动态散列结构。

BSD的开发者扩充了dbm函数库，并将它称为ndbm。

4.4BSD提供了一个新的库--db(3)，该库支持三种不同的访问方式：面向记录、散列和B-树。

# SQLite

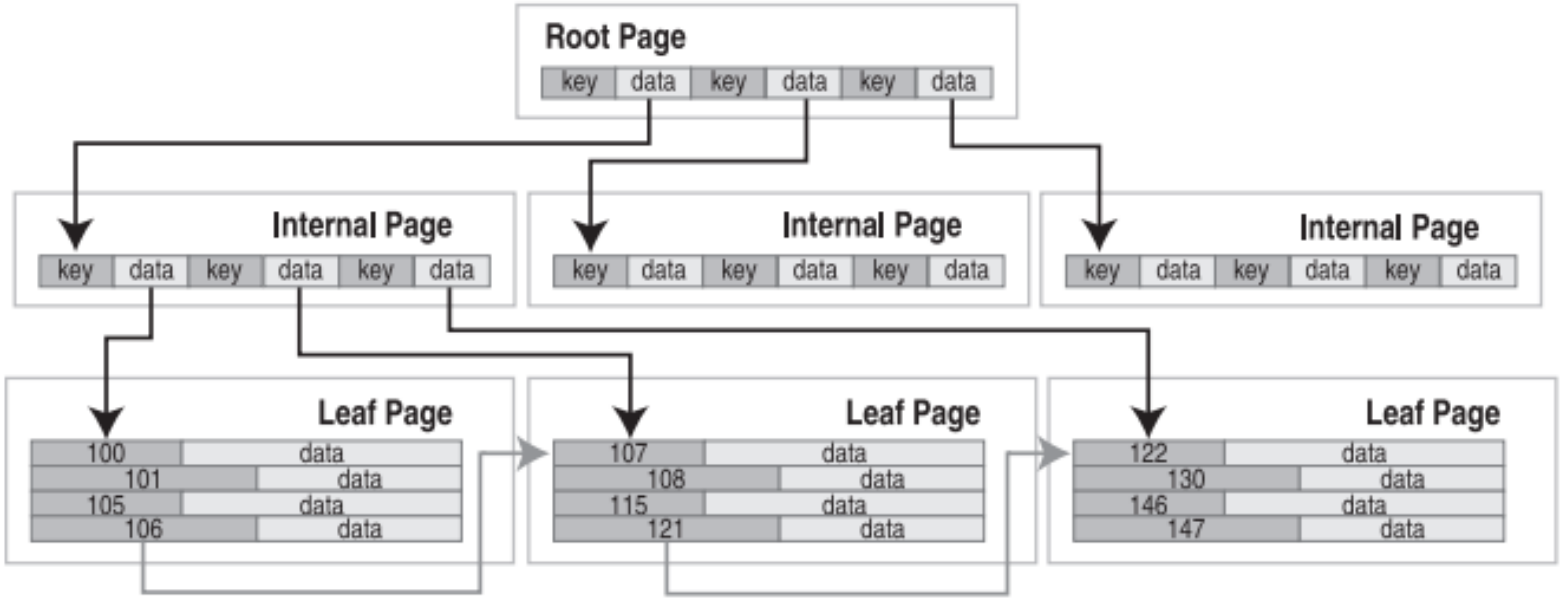



***Figure 11-1.*** *B+tree organization (tables)*

The root page and internal pages in B+trees are all about navigation. The data fields in these pages are all pointers to the pages below them—they contain keys only. All database records are stored on the leaf pages. On the leaf level, records and pages are arranged in key order so it is possible for B-tree cursors to traverse records (horizontally), both forward and backward, using only the leaf pages. This is what makes traversal possible in O(1) time.

# SQLite

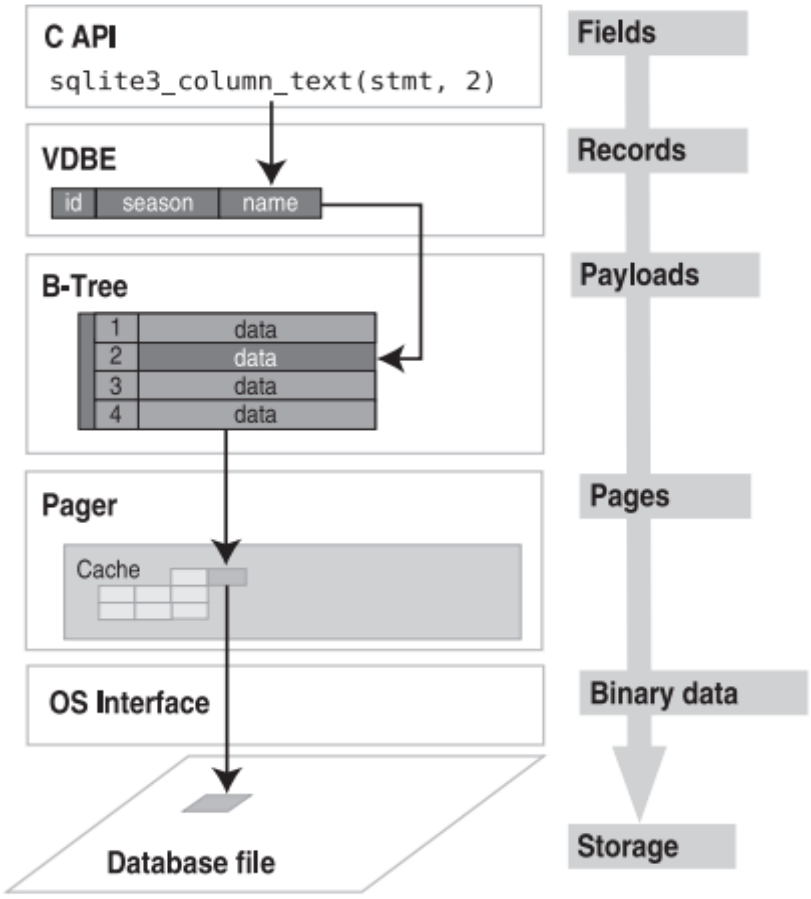
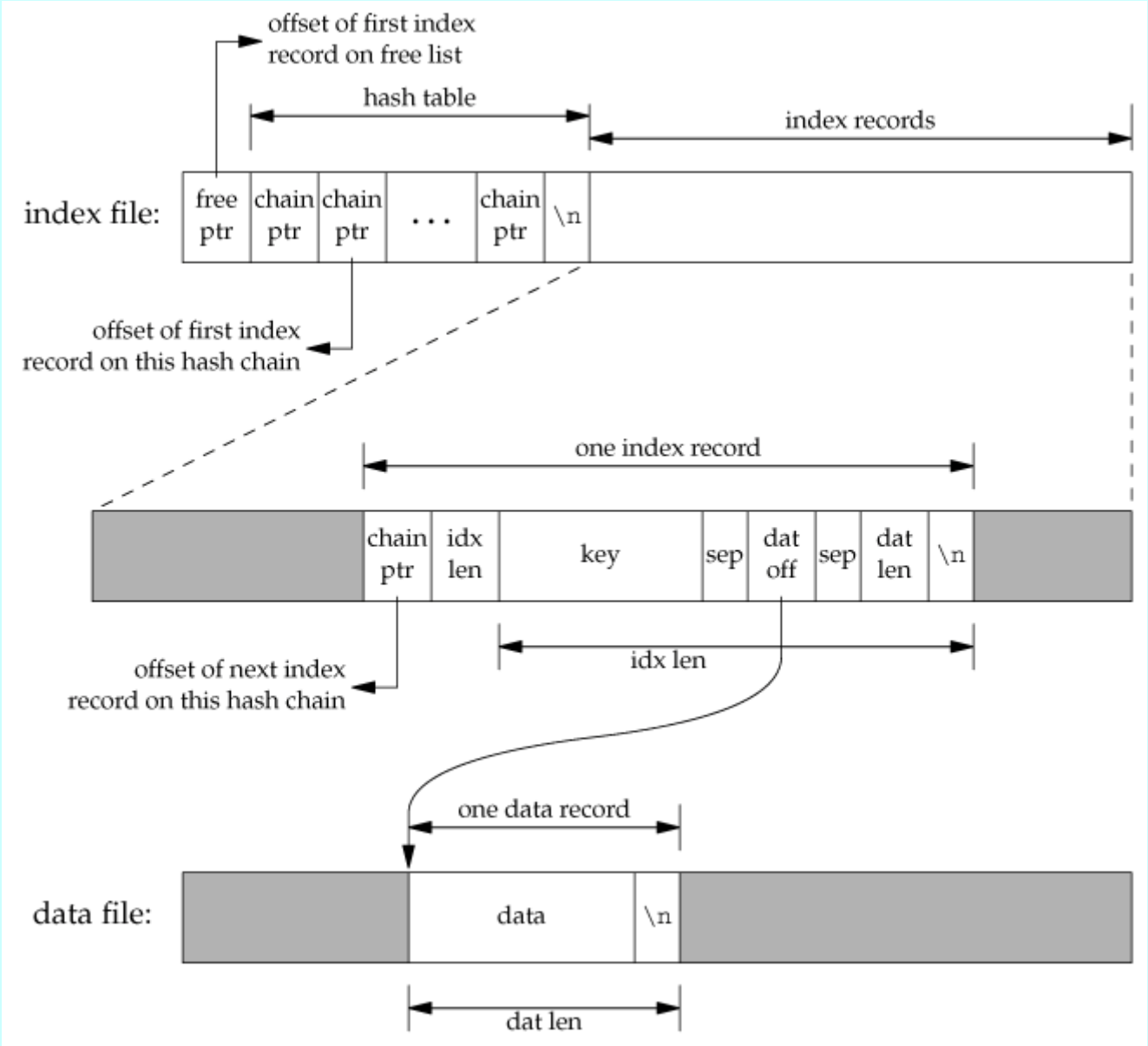



***Figure 11-4.*** *Modules and their associated data*

# 接口定义

- DBHANDLE db_open(const char *pathname, int oflag, ... /* int mode */);
- void db_close(DBHANDLE db);
- int db_store(DBHANDLE db, const char *key, const char *data, int flag);
- char *db_fetch(DBHANDLE db, const char *key)
- void db_rewind(DBHANDLE db);
- char *db_nextrec(DBHANDLE db, char *key);


offset of first index
record on free list
hash table
index records
index file:
free
ptr
chain
ptr
chain
ptr
. . .
chain
ptr
\n
offset of first index
record on this hash chain
one index record
chain
ptr
idx
len
key
sep
dat
off
sep
dat
len
\n
idx len
offset of next index
record on this hash chain
one data record
data file:
data
\n
dat len

# 性能测试

(1) 向数据库写nrec条记录。
(2) 通过关键字读回nrec条记录。
(3) 执行下面的循环nrec×5次。
    (a) 随机读一条记录。
    (b) 每循环37次，随机删除一条记录。
    (c) 每循环11次，随机添加一条记录并读取这条记录。
    (d) 每循环17次，随机替换一条记录为新记录。在连续两次替换中，一次用同样大小的记录替换，一次用比以前更长的记录替换。
(4) 将此进程写的所有记录删除。每删除一条记录，随机地寻找10条记录。

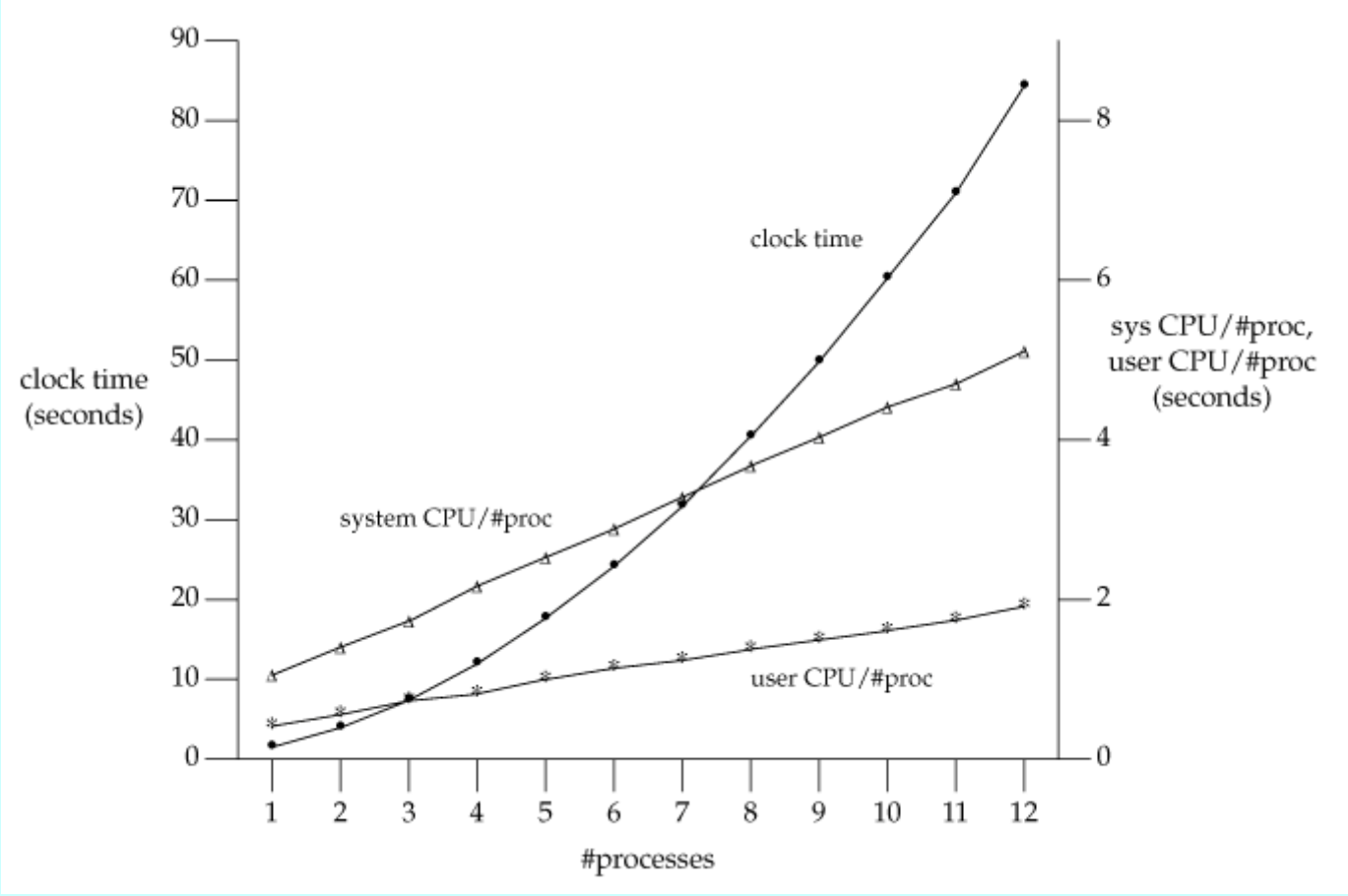
# Benchmark

表20-2　nrec为500时，每个子进程执行的操作的典型计数

| 操　作 | 计　数 |
|---|---|
| db_store，DB_INSERT，无空白记录，添加 | 678 |
| db_store，DB_INSERT，重用空白记录 | 164 |
| db_store，DB_REPLACE，数据长度不同，添加 | 97 |
| db_store，DB_REPLACE，数据长度相同 | 109 |
| db_store，没有找到记录 | 19 |
| db_fetch，找到记录 | 8 114 |
| db_fetch，没有找到记录 | 732 |
| db_delete，找到记录 | 842 |
| db_delete，没有找到记录 | 110 |

读取的次数大约是存储或删除的10倍，这可能是许多数据库应用程序的典型情况。

# Values for advisory fine-grained locking



# 代码风格

## Bjarne Stroustrup，C++之父：

我喜欢优雅、高效的代码：

- 逻辑应该是清晰的，bug难以隐藏；
- 依赖最少，易于维护；
- 错误处理完全根据一个明确的策略；
- 性能接近最佳化，避免代码混乱和无原则的优化；
- 整洁的代码只做一件事。

# 代码风格

Grady Booch，《面向对象分析与设计》作者：

- 整洁的代码是简单、直接的；
- 整洁的代码，读起来像是一篇写得很好的散文；
- 整洁的代码永远不会掩盖设计者的意图，而是具有少量的抽象和清晰的控制行。

# 代码风格

Dave Thomas，OTI公司创始人，Eclipse战略教父：

- 整洁的代码可以被除了原作者之外的其他开发者阅读和改善；
- 具备单元测试和验收测试；
- 有一个有意义的名字；
- 使用一种方式来做一件事情；
- 最少的依赖，并明确定义；
- 提供了一个清晰的、最小的API；
- 应该根据语言特性，在代码中单独显示必要的信息，而不是所有的信息。

# 代码风格

Michael Feathers，《修改代码的艺术》作者：

- 整洁的代码看起来总是像很在乎代码质量的人写的；
- 没有明显的需要改善的地方；
- 代码的作者似乎考虑到了所有的事情。

# 代码风格

Ward Cunningham，Wiki和Fit创始人，极限编程联合创始人，Smalltalk和面向对象的思想领袖：

- 当你读代码时，你发现每个程序都如你期待的那样
- 你可以称之为漂亮的代码
- 代码完美展现了该编程语言的设计目的

# 代码风格参考

1. Google C++ Style Guide: http://google-styleguide.googlecode.com/svn/trunk/cppguide.xml
2. JPL Coding Standard
3. 华为编程规范

# 答辩PPT目录

- 总体架构
- 数据结构设计
- 功能设计
- 性能分析
- 总结

新年快乐！