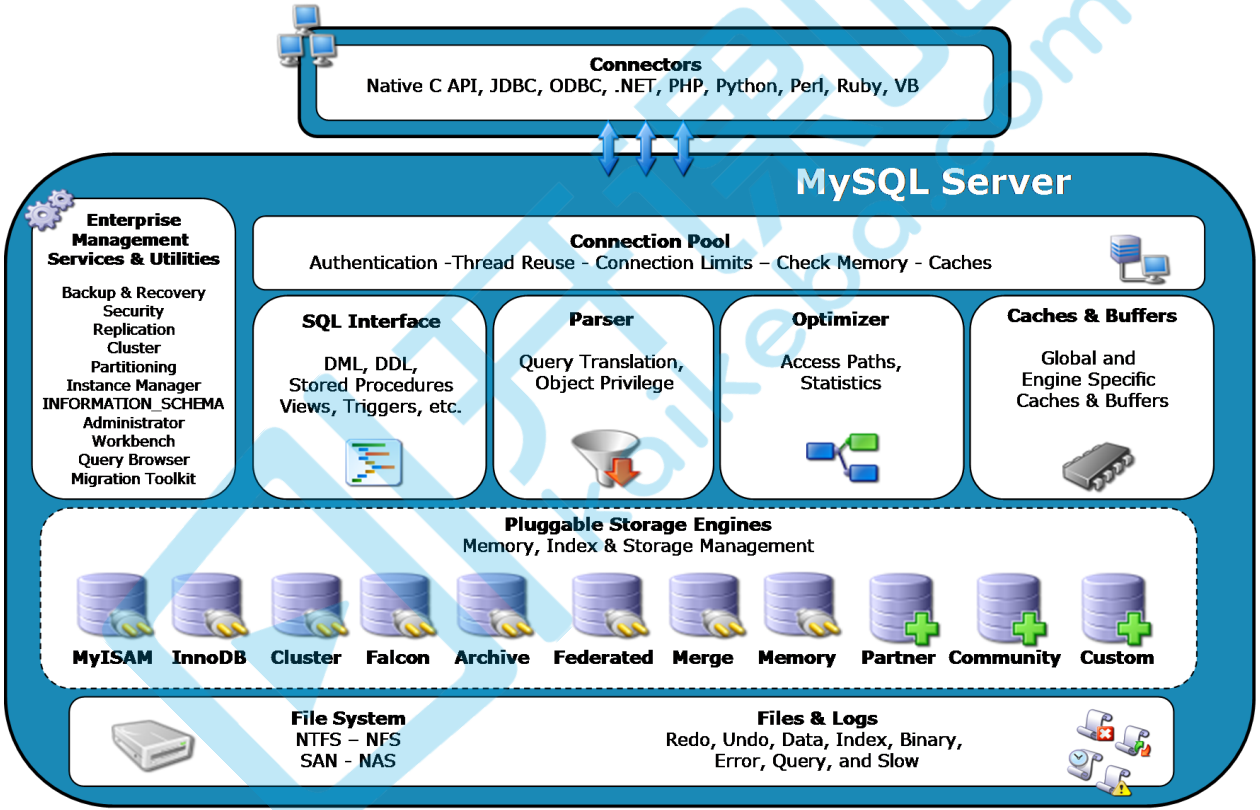
目录

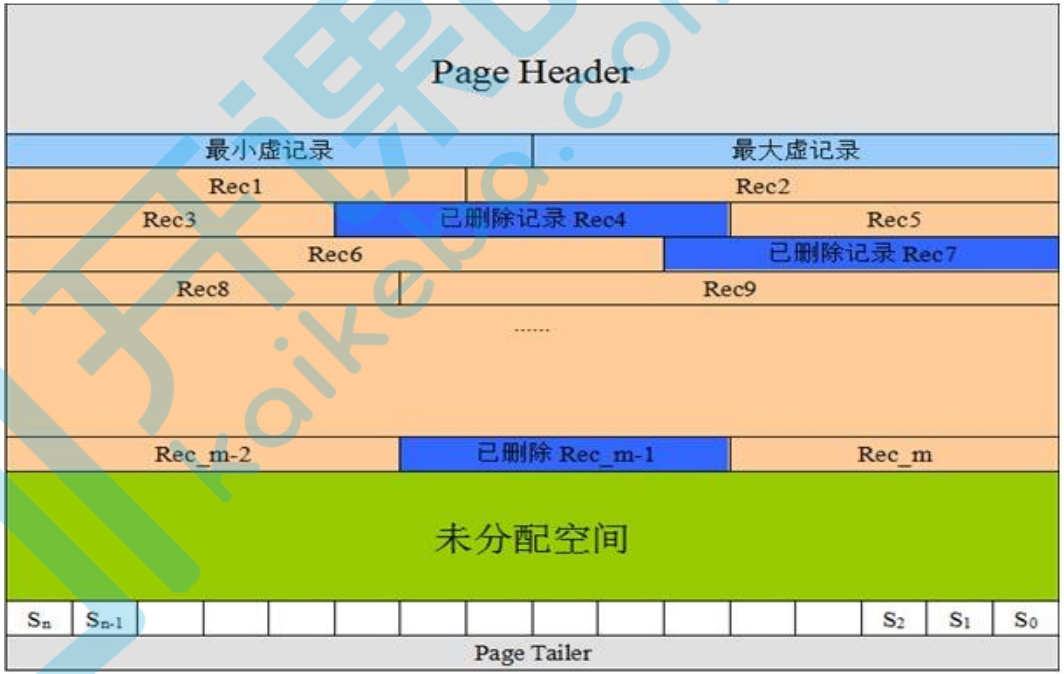
- MySQL存储原理介绍
- 某著名企业MySQL表设计及分表方案
- 某著名企业数据库开发规范解读
- 某著名企业MySQL部署方式&运维知识

MySQL体系结构


Connectors
Native C API, JDBC, ODBC, .NET, PHP, Python, Perl, Ruby, VB
MySQL Server
Enterprise Management Services & Utilities
Backup & Recovery
Security
Replication
Cluster
Partitioning
Instance Manager
INFORMATION_SCHEMA
Administrator
Workbench
Query Browser
Migration Toolkit
Connection Pool
Authentication -Thread Reuse - Connection Limits – Check Memory - Caches
SQL Interface
DML, DDL,
Stored Procedures
Views, Triggers, etc.
Parser
Query Translation,
Object Privilege
Optimizer
Access Paths,
Statistics
Caches & Buffers
Global and
Engine Specific
Caches & Buffers
Pluggable Storage Engines
Memory, Index & Storage Management
MyISAM
InnoDB
Cluster
Falcon
Archive
Federated
Merge
Memory
Partner
Community
Custom
File System
NTFS – NFS
SAN - NAS
Files & Logs
Redo, Undo, Data, Index, Binary,
Error, Query, and Slow

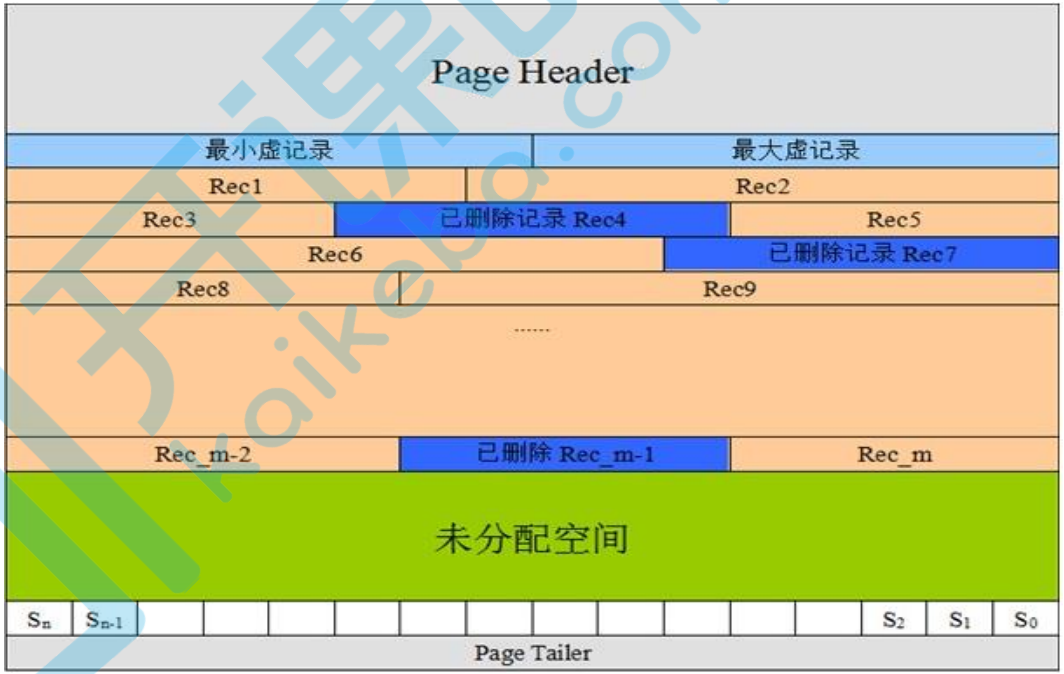
MySQL存储原理介绍

- 记录存储方式
  - 记录按行存储在页内；
  - 按照主键“顺序”
  - 页内单项链表
  - 页之间双向链表



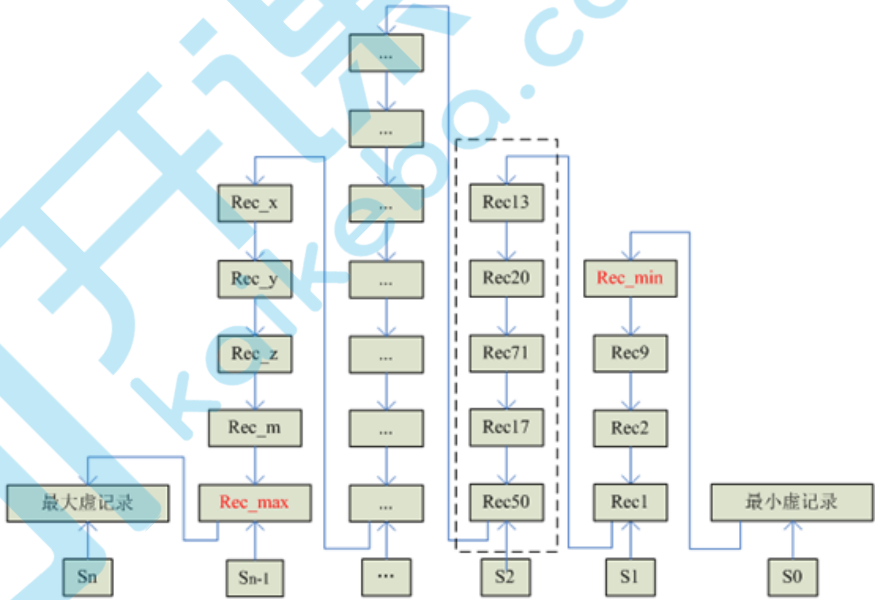
MySQL存储原理介绍

- 页内记录维护
  - 保证“有序”
    - 物理有序 VS 逻辑有序
  - 插入策略
    - 追加写入?
  - 页内查询
    - 是否支持二分查找



MySQL存储原理介绍

- 页面记录逻辑组织结构
  - slot指向链表的一个（支链）
  - slot指向的记录有序
  - 每个slot大小固定



MySQL存储原理介绍

- 基本数据类型
  - 整数类型
    - TINYINT，SMALLINT，MEDIUMINT，INT，BIGINT
  - 浮点类型
    - FLOAT，DOUBLE，DECIMAL（精确数字）
    - DECIMAL是存储类型不是数据类型，CPU不支持直接计算
  - 字符串类型
    - CHAR，VARCHAR

MySQL存储原理介绍

- 变长记录存储
  - 分两部分存储
    - 行内+溢出页
    - 768字节数据+20字节溢出页指针
    - 页内单行长度不能超过页大小一半 16K/2 = 8K
    - 大字段最多10个
  - VARCHAR占用空间
    - 1到2个字节存储长度 >255需要两个字节
    - 最大接近65535
    - VARCHAR(255)存一个字符？
    - VARCHAR(256)存一个字符？
  - 手机号/MD5如何存储

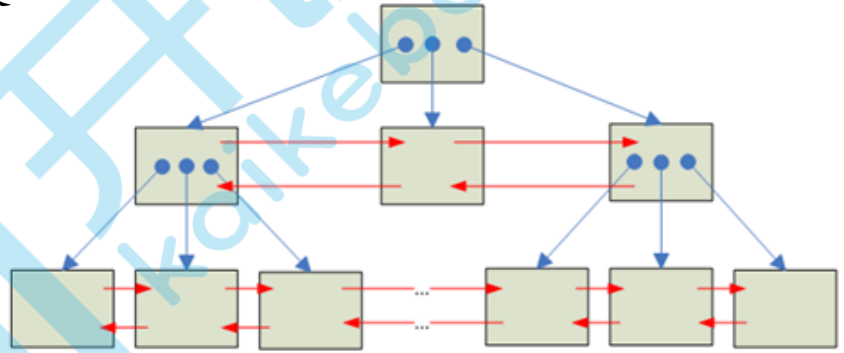
MySQL存储原理介绍

- 主键
  - 唯一确定一条记录
  - 默认创建索引——B+Tree聚簇索引

- 问题
  - 如何选择主键？自增 or 随机 or业务or联合主键

MySQL存储原理介绍

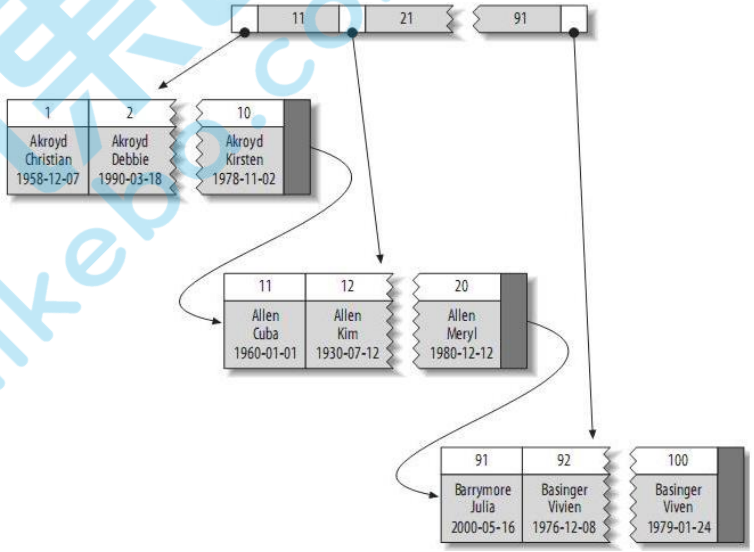
- B+Tree
  - 平衡多叉树
  - 叶子节点存储数据
  - 叶子节点组成顺序双向链表



MySQL存储原理介绍

➢ 主键索引—聚簇索引

- 存储方式，行数据存储在叶子节点
- 自增主键，顺序写入，效率高
- 随机主键，页内碎片，索引大，随机I/O



➢ 主键大小对索引的影响

- 索引文件大小
- 页内节点数->树的层数?

BIGINT类型主键3层可以存储约10亿条数据

16KB/(8B(key)+8B(指针))=1K 10^3 * 10^3 * 10^3 = 10亿

32字节主键3层可以存储6400W

MySQL存储原理介绍

二级索引

- 除了主键索引以外的索引
- 索引的Data是主键
- 索引的key包含主键（保证key唯一）
- 一次查询至少走两次索引
- 主键大小会影响所有二级索引大小

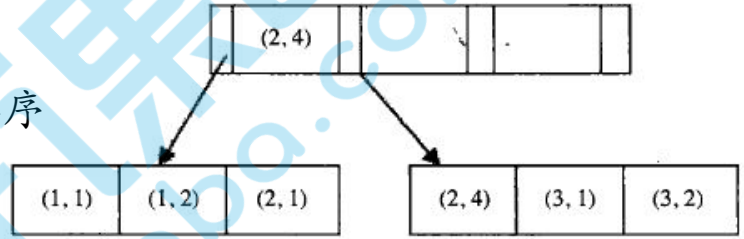
字符串建索引

- 从头部开始截取部分数据作索引
- 不支持%开头的模糊查询

MySQL存储原理介绍

➢ 联合索引——多列索引

- 一个索引只创建一棵树
- 按第一列排序，第一列相同按第二列排序
- 前缀索引



➢ 前缀索引特点

- 按照最左开始查找，否则无法使用索引
- 不能跳过中间列
- 某列使用范围查询，后面的列不能使用索引

MySQL存储原理介绍

- **覆盖索引**
  - 不需要读取数据行
  - 减少数据访问量
  - 减少磁盘I/O
  - 避免对主索引的二次查找

```
mysql> EXPLAIN SELECT store_id, film_id FROM sakila.inventory\G
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: inventory
         type: index
possible_keys: NULL
          key: idx_store_id_film_id
      key_len: 3
          ref: NULL
         rows: 4673
        Extra: Using index
```

MySQL存储原理介绍

➢ 如何选择主键

- 自增主键：写入、查询效率和磁盘利用率都高，每次查询走两级索引
- 随机主键：写入、查询效率和磁盘利用率都低，每次查询走两级索引
- 业务主键：写入、查询效率和磁盘利用率都低，可以使用一级索引，覆盖索引
- 联合主键：影响索引大小，不易维护，DBA已经禁止

MySQL存储原理介绍

➢ 索引使用技巧

- 联合索引：优于多列独立索引
- 索引顺序：选择性高的在前面
- 覆盖索引：二级索引存储主键值更有利
- 索引排序：索引同时满足查询和排序
- 使用“独立”列：索引不能是表达式，不能是函数的参数
- 使用EXPLAIN分析查询

目录

- MySQL存储原理介绍
- 某著名企业MySQL表设计及分表方案
- 某著名企业数据库开发规范解读
- 某著名企业MySQL部署方式&运维知识

MySQL表设计与分表方案

➢ 表设计原则

- 主键选择
  - uid，infoid，msgid(服务生成全局唯一)
- 索引个数
  - 不超过5个
- 列类型选择
  - 通常更小的更好 **时间戳秒还是毫秒**
  - 简单就好 **时间、金额 long是否可行？枚举是否可以用TINYINT代替？**

➢ 实际案例

用户表：uid，nickname，mobile，addr，image.....,switch(bigint,开关量1位表示一个用户状态,频繁被查询)

- 垂直拆分
- 覆盖索引

MySQL表设计与分表方案

- **是否分表（水平拆分）**
  - **数据库出现性能都是在数据量到达一定程度以后！！**
  - 做好预估，不要等需要分再分，拆分一步到位；
  - 控制表的数据量 千万级别 以内

- **分表策略**
  - 按key取模，读写均匀；
  - 按时间分，冷热数据明确；

MySQL表设计与分表方案

- 商品表拆分案例

  主键：infoId，索引：uid 等等其他列

  查询需求：按infoId查商品详情，按uid查用户发布的商品

  key选择：infoId or uid ?

  解决方案

  infoId中包含uid，infoId分表

  按uid查询：直接计算

  按infoid查询：取出uid部分，然后计算

目录

- MySQL存储原理介绍
- 某著名企业MySQL表设计及分表方案
- 某著名企业数据库开发规范解读
- 某著名企业MySQL部署方式&运维知识

某著名企业数据库开发规范解读

- 具体内容详见word文档《**转转数据库使用规范解读.docx**》