数据库设计使用规约

php-include<haikun2012@qq.com>

2017/7/3

目录

[表引擎 2](#_Toc486920591)

[表命名 2](#_Toc486920592)

[字段命名 2](#_Toc486920593)

[字段类型 3](#_Toc486920594)

[索引 3](#_Toc486920595)

[日常使用： 4](#_Toc486920596)

# 表引擎

默认使用InnoDB，其他按照实际场景选用，但需要在表描述里作出必要说明

# 表命名

采用分层递进命名，上下层有从属关系，非特殊情况（日志模块），不跳层设计：

模块( goods )

模块\_自定义  (goods\_image)

模块\_自定义\_自定义(goods\_image\_extend)

# 字段命名

字段和表的注释都要正确填写，要完整，可读，字段命名使用英文单词，不可以使用不通用的缩写。表命名和字段命名，使用单数，非复数： store 非 stores；废弃使用统一的表前缀。

每个表必须包含三个必要字段：

唯一标识id（可以为非主键，使用唯一索引）,

created\_time（程序插入）,

update\_time（自动更新）

主键：

不能直接使用id，

模块主表为：模块名\_id ：goods\_id,store\_id,user\_id，

其他表如：goods\_image 即为：gi\_id

如果一个模块里面有重名id,则最新一个主键按照最后一个单纯取前面几个字母，一直到能区分为止：

api\_interface\_require ：air\_id(第一次定义)

api\_interface\_result ：aire\_id(第二次定义)

创建时间：created\_time

最后更新时间：update\_time

状态：status 0: 默认 4: 删除',

类型：type

名字直接使用name，如果存在多个名字，可以使用全名英文

# 字段类型

int默认使用int(10)

char 与varchar合理使用，注意长度和默认值

时间尽量存储为int

type和status等类型的用tinyint

钱相关的使用有符号int 分为单位存储

任何字段如果为非负数，必须是 unsigned

非特殊情况字段需要设置默认值，使用not null

# 索引

索引数量为不超5个，

命中率不高的字段不要建索引。例如 status 、type

mysql5.7以后多个单索引会自动组合为联合索引，创建索引时尽量少创建联合索引

# 日常使用：

1. 模块化结构，所以不能跨模块链表查询
2. 不能[慎重]使用 like '%关键词%'
3. 不能使用\*来查询所有字段，但统计时只能使用count(\*)查询，不能使用count(字段名)或count(1)
4. 查询一般都要使用limit指定查询数量，不能全表查取
5. 坚持写完SQL后用explain分析其性能、扫描行数及索引使用情况
6. 注意SQL注入，参数使用之前要安全过滤
7. 能拆分查询时，尽量避免链表查询，且链表数量控制在3个以内
8. 事务使用注意高并发导致的锁表和避开在事务中存在大量的查询操作。
9. 能明确查询的数据处理在数据库处理还是在PHP中处理的利弊
10. 在主从服务器的框架开发业务时，请注意主从延迟可能会带来的BUG，关键业务建议还是查主库。
11. 日志，统计等数据量大的，高并发的不重要的业务场景开发时，数据建议存储在mongodb中。
12. 避免循环单条插入，应该组装为合适长度数量的一条批量插入。
13. 慎用批量更新，update操作一定要注意后面的where条件是否有效。
14. 慎用子查询，需要组长审核
15. 慎用删除操作，业务上的删除非真正的数据删除，当真的要删除数据的时候，提交给运维人员处理。
16. 减少循环单条查询，能合并查询的建议写修改为用in（ids）的方式查询，减少查询次数。
17. 不可避免的大规模更新，需要避开站点高峰期。
18. 增加，修改等操作索引，如果操作表数据量大，避免高峰期操作。
19. 难于优化的SQL，建议拆分业务，拆分为多条查询，减轻一次性查询数据库压力，或者使用临时表，或者使用搜索引擎。
20. 使用mysql，如果预计三年后的单表行数超过 500 万行或者单表容量超过 2GB，才推荐进行分库分表。
21. 关于非关系数据库 mongodb字段命名大致和MySQL风格一致。
22. 不能使用存储过程。