[**MySql 表设计原则与注意事项**](http://blog.csdn.net/xundh/article/details/42833849)

1. 字段尽量设置为Not Null
2. 认真规范字段大小，越小越好；数据类型越简单越好
3. 表中不应该有重复值和字段
4. 表中记录应有唯一标志符
5. 表名规范前缀
6. 一个表尽量存储一个对象本身
7. 小数空间占用可能比整数大，精度高时会消耗更多CPU资源。可能的情况下，把数据存储为整数，由客户程序再转换运算。
8. 一些短且定长的，使用char比varchar更高效
9. 使用Memory引擎要避免Blog、TEXT类型（因为Memory不支持BLOG、TEXT，所以查询过程需要使用MyISAM磁盘临时表）
10. DATETIME比TIMESTAMP大一倍，但TIMESTAMP时间区间支持1970-2038。
11. 避免字符串作ID
12. 不要过度索引
13. 规范命名  
    所有的库名、表名、域名必须遵循统一的命名规则，并进行必要说明，以方便设计、维护、查询。
14. 控制字段的引用  
    在设计时，可以选择适当的数据库设计管理工具，以方便开发人员的分布式设计和数据小组的集中审核管理。采用统一的命名规则，如果设计的字段已经存在，可直接引用；否则，应重新设计。
15. 库表重复控制  
    在设计过程中，如果发现大部分字段都已存在，开发人员应怀疑所设计的库表是否已存在。通过对字段所在库表及相应设计人员的查询，可以确认库表是否确实重复。
16. 并发控制  
    设计中应进行并发控制，即对于同一个库表，在同一时间只有一个人有控制权，其他人只能进行查询。
17. 必要的讨论  
    数据库设计完成后，数据小组应与相关人员进行讨论，通过讨论来熟悉数据库，从而对设计中存在的问题进行控制或从中获取数据库设计的必要信息。
18. 数据小组的审核  
    库表的定版、修改最终都要通过数据小组的审核，以保证符合必要的要求。
19. 头文件处理  
    每次数据修改后，数据小组要对相应的头文件进行修改（可由管理软件自动完成），并通知相关的开发人员，以便进行相应的程序修改。
20. **三、设计技巧**
21. 分类拆分数据量大的表  
    对于经常使用的表（如某些参数表或代码对照表），由于其使用频率很高，要尽量减少表中的记录数量。例如，银行的户主账表原来设计成一张表，虽然可以方便程序的设计与维护，但经过分析发现，由于数据量太大，会影响数据的迅速定位。如果将户主账表分别设计为活期户主账、定期户主账及对公户主账等，则可以大大提高查询效率。
22. 索引设计  
    应挑选重复值较少的字段；在对建有复合索引的字段进行检索时，应注意按照复合索引字段建立的顺序进行。例如，如果对一个5万多条记录的流水表以日期和流水号为序建立复合索引，由于在该表中日期的重复值接近整个表的记录数，用流水号进行查询所用的时间接近3秒；而如果以流水号为索引字段建立索引进行相同的查询，所用时间不到1秒。因此在大型数据库设计中，只有进行合理的索引字段选择，才能有效提高整个数据库的操作效率。
23. 数据操作的优化  
    在大型数据库中，如何提高数据操作效率值得关注。例如，每在数据库流水表中增加一笔业务，就必须从流水控制表中取出流水号，并将其流水号的数值加一。正常情况下，单笔操作的反应速度尚属正常，但当用它进行批量业务处理时，速度会明显减慢。经过分析发现，每次对流水控制表中的流水号数值加一时都要锁定该表，而该表却是整个系统操作的核心，有可能在操作时被其他进程锁定，因而使整个事务操作速度变慢。对这一问题的解决的办法是，根据批量业务的总笔数批量申请流水号，并对流水控制表进行一次更新，即可提高批量业务处理的速度。另一个例子是对插表的优化。对于大批量的业务处理，如果在插入数据库表时用普通的Insert语句，速度会很慢。其原因在于，每次插表都要进行一次I/O操作，花费较长的时间。改进后，可以用Put语句等缓冲区形式等满页后再进行Ｉ/Ｏ操作，从而提高效率。对大的数据库表进行删除时，一般会直接用Delete语句，这个语句虽然可以进行小表操作，但对大表却会因带来大事务而导致删除速度很慢甚至失败。解决的方法是去掉事务，但更有效的办法是先进行Drop操作再进行重建。
24. 数据库参数的调整  
    数据库参数的调整是一个经验不断积累的过程，应由有经验的系统管理员完成。以Informix数据库为例，记录锁的数目太少会造成锁表的失败；逻辑日志的文件数目太少会造成插入大表失败等，这些问题都应根据实际情况进行必要的调整。
25. 必要的工具  
    在整个数据库的开发与设计过程中，可以先开发一些小的应用工具，如自动生成库表的头文件、插入数据的初始化、数据插入的函数封装、错误跟踪或自动显示等，以此提高数据库的设计与开发效率。
26. 避免长事务  
    对单个大表的删除或插入操作会带来大事务，解决的办法是对参数进行调整，也可以在插入时对文件进行分割。对于一个由一系列小事务顺序操作共同构成的长事务（如银行交易系统的日终交易），可以由一系列操作完成整个事务，但其缺点是有可能因整个事务太大而使不能完成，或者，由于偶然的意外而使事务重做所需的时间太长。较好的解决方法是，把整个事务分解成几个较小的事务，再由应用程序控制整个系统的流程。这样，如果其中某个事务不成功，则只需重做该事务，因而既可节约时间，又可避免长事务。
27. 适当超前  
    计算机技术发展日新月异，数据库的设计必须具有一定前瞻性，不但要满足当前的应用要求，还要考虑未来的业务发展，同时必须有利于扩展或增加应用系统的处理功能。
28. 与小型数据库相比，大型数据库的设计与开发要复杂得多，因此在设计、开发过程中，除了要遵循数据库范式理论、增加系统的一致性和完整性外，还要在总体上根据具体情况进行分布式设计，紧紧把握集中控制、统一审核的基本原则，保证数据库设计结构紧凑、分布平衡、定位迅速。在数据库操作上，要采用一定的技巧提高整个应用系统的执行效率，并注意适当超前，以适应不断变化的应用及系统发展的要求。

# 大项目如何设计mysql表结构

1、这个数据库设计关系不大，但是有关系。首先，你要考虑数据库的选型，大型项目是否用MySQL。

2、用MySQL是否对事务完整性有要求。InnoDB和MyISAM的区别。

3、了解一下DP（设计模式），在架构上考虑缓存、读写缓冲、读写分离等手段。

4、MySQL目前尚不支持表分区等功能，那么就要求要考虑是否分表存储。表存储的物理介质是否要分开。

5、索引。

6、即时数据和历史数据分开。

大体上先做这些，后期根据实际情况酌情优化。