mysql

数据库三大范式

第一范式：每个列都不可以再拆分

第二范式：在第一范式的基础上，非主键列完全依赖于主键，而不能是依赖于主键的一部分。

第三范式：在第二范式的基础上，非主键列只依赖于主键，不依赖于其他非主键。

权限的表

权限表分别user，db，table_priv，columns_priv和host

user权限表：记录允许连接到服务器的用户帐号信息，里面的权限是全局级的。

db权限表：记录各个帐号在各个数据库上的操作权限。

table_priv权限表：记录数据表级的操作权限。

columns_priv权限表：记录数据列级的操作权限。

host权限表：配合db权限表对给定主机上数据库级操作权限作更细致的控制。这个权限表不受GRANT和REVOKE语句的影响。

权限表存放在mysql数据库里，由mysql_install_db脚本初始化

事务

ACID特性

原子性

一致性

隔离性

行隔离

条件中使用了索引（主键），那么只隔离该条记录

表隔离

没有索引，系统是通过全表检索，则会锁定整个表

持久性

对数据库的改变是永久性的，其他操作或故障不应该对其有影响

概念：访问并可能更新数据库中各数据项的一个程序执行单元

存储引擎必须是INNODB

将用户所在的操作暂时保存起来，不直接放到数据表（更新），等到用户确认结果之后再进行操作

自动事务

autocommit,当客户端发送一条sql指令（增删改）给服务器的时候，服务器在执行之后，不用等待用户反馈结果，会自动将结果同步到数据表。

show variables like ‘autommit%’；
关闭自动事物：set autocommit =off;

一旦事务关闭，那么需要用户提供是否同步的命令
commit：提交（同步数据表，事物也会被清空）
rollback：回滚(清空之前的操作）

手动事务

不管是开始还是结果都需要用户发送指令来操作

start transaction; //从这条语句开始，后面的所有语句都将保持到事务日志中而不是直接写入数据表

事务提交：commit/rollback,到这个时候所有的事务才算是就结束了。

回滚点（savepoint）

增加回滚点：savepoint 回滚点名字；//字母数字和下划线构成

回到回滚点：rollback to 回滚点名字//回滚点之后的操作被清楚了

术语

脏读

某个事务已更新一份数据，另一个事务在此时读取了同一

份数据，由于某些原因，前一个RollBack了操作，则后一个事务所读取的数据就会是不正确的。

不可重复读

在一个事务的两次查询之中数据不一致（可能是两次查询过程中间插入了一个事务更新的原有的数据）

幻读

在一个事务的两次查询中数据笔数不一致（例如有一个事务查询了几列(Row)数，而另一个事务却在此时插入了新的几列数据，先前的事务在接下来的查询中，就会发现有几列数据是它先前所没有的）

锁

共享锁: 又叫做读锁。 当用户要进行数据的读取时，对数据加上共享锁。共享锁可以同时加上多个

排他锁: 又叫做写锁。 当用户要进行数据的写入时，对数据加上排他锁。排他锁只可以加一个，与其他的排他锁、共享锁都互斥

并发控制

乐观锁

假设不会发生并发冲突，只在提交操作时检查是否违反数据完整性

作时检查是否违反数据完整性

多读场景

悲观锁

假定会发生并发冲突，屏蔽一切可能违反数据完整性的操作

在查询完数据的时候就把事务锁起来，直到提交事务

多写的场景下

视图

本质上是一种虚拟表，在物理上是不存在的。其内容与真实的表相似

数据类型

整数类型

tinyInt

很小的整数(8位二进制)

1字节

smallint

小的整数(16位二进制)

2字节

mediumint

中等大小的整数(24位二进制)

3字节

int（integer）

普通大小的整数(32位二进制)

4字节

bigint

8字节整数

任何整数类型都可以加上UNSIGNED属性，表示数据是无符号的，即非负整数

长度：整数类型可以被指定长度

INT(11)表示长度为11的INT类型。长度在大多数场景是没有意义的，它不会限制值的合法范围，只会影响显示字符的个数，而且需要和NSIGNEDZEROFILL属性配合使用才有意义。

假定类型设定为INT(5)，属性为UNSIGNEDZEROFILL，如果用户插入的数据为12的话，那么数据库实际存储数据为00012

小数类型

float

单精度浮点数

double

双精度浮点数

decimal（m,d）

压缩严格的定点数

DECIMAL可以用于存储比BIGINT还大的整型，能存储精确的小数

DECIMAL你可以理解成是用字符串进行处理

日期类型

year

YYYY 1901~2155

time

HH:MM:SS -838:59:59~838:59:59

date

YYYYMM-DD 1000-01-01~9999-12-3

datetime

YYYYMM-DDHH:MM:SS 1000-01-0100:00:00~9999-12-3123:59:59

timestamp

timestamp

文本、二进制类型

CHAR(M)

M为0~65535之间的整数

CHAR是定长的，根据定义的字符串长度分配足够的空间。

CHAR存储的内容超出设置的长度时，内容同样会被截断。

VARCHAR(M)

M为0~65535之间的整数

使用额外1或2个字节存储字符串长度。列长度小于255字节时，使用1字节表示，否则使用2字节表示

VARCHAR存储的内容超出设置的长度时，内容会被截断。

TINYBLOB

允许长度0~255字节

BLOB

允许长度0~65535字节

MEDIUMBLOB

允许长度0~167772150字节

LONGBLOB

允许长度0~4294967295字节

TINYTEXT

允许长度0~255字节

TEXT

允许长度0~65535字节

MEDIUMTEXT

允许长度0~167772150字节

LONGTEXT

允许长度0~4294967295字节

VARBINARY(M)

允许长度0~M个字节的变长字节字符串

BINARY(M)

允许长度0~M个字节的定长字节字符串

枚举类型（ENUM）

把不重复的数据存储为一个预定义的集合

引擎（存储引擎）

定义：MySQL中的数据、索引以及其他对象是如何存储的，是一套文件系统的实现

Innodb引擎

提供了对数据库ACID事务的支持。并且还提供了行级锁和外键的约束

设计的目标就是处理大数据容量的数据库系统

四大特性

插入缓冲（insertbuffer)

二次写(doublewrite)

自适应哈希索引(ahi)

预读(readahead)

更新（删除）操作频率也高，或者要保证数据的完整性；并发量高，支持事务和外键。比如OA自动化办公系统

MyIASM引擎(原本Mysql的默认引擎)

不提供事务的支持，也不支持行级锁和外键。

以读写插入为主的应用程序，比如博客系统、新闻门户网站。

MEMORY引擎

所有的数据都在内存中，数据的处理速度快，但是安全性不高

索引

定义：包含着对数据表里所有记录的引用指针

原理：把无序的数据变成有序的查询

1. 把创建了索引的列的内容进行排序

2. 对排序结果生成倒排表

3. 在倒排表内容上拼上数据地址链

4. 在查询的时候，先拿到倒排表内容，再取出数据地址链，从而拿到具体数据

设计索引的原则

1.适合索引的列是出现在where子句中的列，或者连接子句中指

定的列

2.基数较小的类，索引效果较差，没有必要在此列建立索引

3.使用短索引，如果对长字符串列进行索引，应该指定一个前缀长度，这样能够节省大量索引空间

4.不要过度索引

较频繁作为查询条件的字段才去创建索引

更新频繁字段不适合创建索引

若是不能有效区分数据的列不适合做索引列(如性别，男女未知，多也就三种，区分度实在太低)

尽量的扩展索引，不要新建索引。比如表中已经有a的索引，现在要加(a,b)的索引，那么只需要修改原来的索引即可

数据库索引，是数据库管理系统中一个排序的数据结构，以协助快速查询、更新数据库表中数据。索引的实现通常使用B树及其变种

使用场景：

1.查询

2.orderby排序

3.join

对join语句匹配关系（on）涉及的字段建立索引能够提高效率

索引覆盖

如果要查询的字段都建立过索引，那么引擎会直接在索引表中查询而不会访问原始数据（否则只要有一个字段没有建立索引就会做全表扫描）

需要尽可能的在select后只写必要的查询字段，以增加索引覆盖的几率

类型

主键索引

数据列不允许重复，不允许为NULL，一个表只能有一个主键

唯一索引

数据列不允许重复，允许为NULL值，一个表允许多个列创建唯一索引

ALTER TABLE table_name ADD UNIQUE(column);

ALTER TABLE table_name ADD UNIQUE(column1，column2);

普通索引

基本的索引类型，没有唯一性的限制，允许为NULL值。

ALTER TABLE table_name ADD index index_name(column);

全文索引：是目前搜索引擎使用的一种关键技术。

ALTER TABLE table_name ADD FULLTEXT(column);

术语

DML(Data Manipulation Language)数据操纵语言

对数据库中的数据进行一些简单操作，如insert,delete,update等

DDL(Data Definition Language)数据定义语言

对数据库中的某些对象(例如，database,table)进行管理，如Create,Alter和Drop.

DQL（Data Query Language）

以select关键字。各种简单查询，连接查询等

DCL（Data Control Language）

数据库安全性完整性等有操作的权限控制： grant，revoke，commit，rollback

binlog

它记录了所有的DDL和DML语句（除了数据查询语句select），以事件形式记录，还包含语句所执行的消耗的时间

录入格式

statement模式下，每一条会修改数据的sql都会记录在binlog中。不需要记录每一行的变化，减少了binlog日志量，节约了IO，提高性能。由于sql的执行是有上下文的，因此在保存的时候需要保存相关的信息，同时还有一些使用了函数之类的语句无法被记录复制。

row级别下，不记录sql语句上下文相关信息，仅保存哪条记录被修改。记录单元为每一行的改动，基本是可以全部记下来但是由于很多操作，会导致大量行的改动(比如altertable)，因此这种模式的文件保存的信息太多，日志量太大。

mixed，一种折中的方案，普通操作使用statement记录，当无法使用statement的时候使用row。

在设计数据库结构的时候，要尽量遵守三范式，如果不遵守，必须有足够的理由。比如性能。事实上我们经常会为了性能而妥协数据库的设计。