MariaOzawa
博客等级 码龄8年
55 原创 120 点赞 275 收藏 37 粉丝
关注 私信

热门文章
java把map转成url参数拼接 14605
java8 LocalDateTime 获取当前是第几周 14441
五百万数据 mysql count（*）优化 12162
feign自定义请求拦截器、编码器、解码器 8943
nginx配置SSL(https)证书 8726

分类专栏
spring 2篇
设计模式 14篇
笔记 47篇

上一篇：mysql基本架构及select语句执...
下一篇：mysql的redo log和bin log

grant之后为什么要flush privileges

原创 MariaOzawa 于 2020-06-28 17:04:01 发布 阅读量1k 收藏 点赞数 1 CC 4.0 BY-SA版权
分类专栏：笔记 文章标签：mysql

笔记 专栏收录该内容 47 篇文章 订阅专栏

摘要 本文深入解析MySQL权限管理系统，包括用户创建、权限赋予权限回收等操作的底层逻辑，以及flushprivileges命令的作用和使用场景。

摘要生成于 C知道 ，由 DeepSeek-R1 满血版支持， 前往体验 >

grant 之后真的需要执行 flush privileges 吗？如果没有执行这个 flush 命令的话，赋权语句真的不能生效吗?

先创建一个用户：

AI写代码 登录复制

```
1 create user 'ua'@'%' identified by 'pa';
```

这条语句的逻辑是创建一个用户'ua'@'%'，密码是 pa。注意，在 MySQL 里面，**用户名 (user)+ 地址 (host) 才表示一个用户**，因此 ua@ip1 和 ua@ip2 代表的是两个不同的用户。

这条命令做了两个动作：

1. 磁盘上，往 mysql.user 表里插入一行，由于没有指定权限，所以这行数据上所有表示权限的字段的值都是 N；
2. 内存里，往数组 acl_users 里插入一个 acl_user 对象，这个对象的 access 字段值为 0。

下图就是这个时刻用户 ua 在 user 表中的状态。

```
mysql> select * from mysql.user where user='ua'\G
*************************** 1. row ***************************
                  Host: %
                  User: ua
           Select_priv: N
           Insert_priv: N
           Update_priv: N
           Delete_priv: N
           Create_priv: N
             Drop_priv: N
           Reload_priv: N
         Shutdown_priv: N
          Process_priv: N
             File_priv: N
            Grant_priv: N
       References_priv: N
            Index_priv: N
            Alter_priv: N
          Show_db_priv: N
            Super_priv: N
 Create_tmp_table_priv: N
      Lock_tables_priv: N
          Execute_priv: N
       Repl_slave_priv: N
      Repl_client_priv: N
      Create_view_priv: N
        Show_view_priv: N
   Create_routine_priv: N
    Alter_routine_priv: N
      Create_user_priv: N
            Event_priv: N
          Trigger_priv: N
Create_tablespace_priv: N
                      https://blog.csdn.net/MariaOzawa
```

在 MySQL 中，用户权限是有不同的范围的。接下来，我就按照用户权限范围从大到小的顺序依次和你说明。

全局权限

一个SQL查询花费35秒，如何排查?

最新文章

14441

五百万数据 mysql count（*）优化 12162

feign自定义请求拦截器、编码器、解码器 8943

nginx配置SSL(https)证书 8726

分类专栏

spring 2篇

设计模式 14篇

笔记 47篇

上一篇：mysql基本架构及select语句执…

下一篇：mysql的redo log和bin log

最新评论

java把map转成url参数拼接
weixin_42476498: 不是说是哪个类?

五百万数据 mysql count（*）优化
星落的小镇: 清晰明了，帮帮哒

mysql为什么对字段使用函数就不能走索引
luckue: 1.B+树同一层的兄弟节点是有序的，非叶子节点只存储了索引值，叶子节…

2022 oracle账号，下载jdk账号
JavaPope: 我愿称你为“神”

SpringCloud Alibaba(Nacos)整合SpringB…
七月流萤: 大佬，你的这个问题怎么解决的?

大家在看

P1352 没有上司的舞会 (树型dp)

如何一键压缩并裁剪百张图片?

从0到1：C++语法之命名空间 117

【前端八股文面试题】【JavaScript篇7】什么是JavaScript的原型、原型链? 有什么特点

一个SQL查询花费35秒，如何排查?

最新文章

ArrayList iterator源码分析，为什么迭代器删除元素不会报错

ArrayList扩容源码解析

2022 oracle账号，下载jdk账号

2023年 1篇	2022年 2篇
2021年 5篇	2020年 47篇
2019年 9篇	2018年 2篇

全局权限，作用于整个 MySQL 实例，这些权限信息保存在 mysql 库的 user 表里。给用户 ua 赋一个最高权限的话是这么写的：

AI写代码 登录复制

```
grant all privileges on *.* to 'ua'@'%' with grant option;
```

这个 grant 命令做了两个动作：

1. 磁盘上，将 mysql.user 表里，用户'ua'@'%'这一行的所有表示权限的字段的值都修改为'Y'；
2. 内存里，从数组 acl_users 中找到这个用户对应的对象，将 access 值（权限位）修改为二进制的“全 1”。

在这个 grant 命令执行完成后，如果有新的客户端使用用户名 ua 登录成功，MySQL 会为新连接维护一个线程对象，然后**从 acl_users 数组里查到这个用户的权限，并将权限值拷贝到这个线程对象中。之后在这个连接中执行的语句，所有关于全局权限的判断，都直接使用线程对象内部保存的权限位。**

基于上面的分析我们可以知道：

1. grant 命令对于全局权限，同时更新了磁盘和内存。命令完成后即时生效，接下来新创建的连接会使用新的权限。
2. 对于一个已经存在的连接，它的全局权限不受 grant 命令的影响。

db权限

除了全局权限，MySQL 也支持库级别的权限定义。如果要让用户 ua 拥有库 db1 的所有权限，可以执行下面这条命令：

AI写代码 登录复制

```
grant all privileges on db1.* to 'ua'@'%' with grant option;
```

基于库的权限记录保存在 mysql.db 表中，在内存里则保存在数组 acl_dbs 中。这条 grant 命令做了如下两个动作：

1. 磁盘上，往 mysql.db 表中插入了一行记录，所有权限位字段设置为“Y”；
2. 内存里，增加一个对象到数组 acl_dbs 中，这个对象的权限位为“全 1”。

下图就是这个时刻用户 ua 在 db 表中的状态。

```
mysql> select * from mysql.db where user='ua'\G
*************************** 1. row ***************************
                 Host: %
                   Db: db1
                 User: ua
          Select_priv: Y
          Insert_priv: Y
          Update_priv: Y
          Delete_priv: Y
          Create_priv: Y
            Drop_priv: Y
           Grant_priv: Y
      References_priv: Y
           Index_priv: Y
           Alter_priv: Y
Create_tmp_table_priv: Y
     Lock_tables_priv: Y
     Create_view_priv: Y
       Show_view_priv: Y
  Create_routine_priv: Y
   Alter_routine_priv: Y
         Execute_priv: Y
           Event_priv: Y
         Trigger_priv: Y
1 row in set (0.00 sec)
```


每次需要**判断一个用户对一个数据库读写权限的时候，都需要遍历一次 acl_dbs 数组**，根据 user、host 和 db 找到匹配的对象，然后根据对象的权限位来判断。

也就是说，grant 修改 db 权限的时候，是同时对磁盘和内存生效的。

目录
全局权限
db权限
表权限和列权限
flush privileges 使用场景
小结

grant 操作对于已经存在的连接的影响，在**全局权限和基于 db 的权限效果是不同的。**接下来，我们做一个对照试验来分别看一下。

	session A	session B	session C
T1	connect(root,root) create database db1; create user 'ua'@'%' identified by 'pa'; grant super on *.* to 'ua'@'%'; grant all privileges on db1.* to 'ua'@'%';		

T2		connect(ua,pa)
		set global sync_binlog=1;
		(Query OK)
		create table db1.t(c int);
		(Query OK)
			connect(ua,pa)
			use db1;

T3	revoke super on *.* from 'ua'@'%';		

T4		set global sync_binlog=1;
		(Query OK)
		alter table db1.t engine=innodb;
		(Query OK)
			alter table t engine=innodb;
			(Query OK)

T5	revoke all privileges on db1.* from 'ua'@'%';		

T6		set global sync_binlog=1;
		(Query OK)
		alter table db1.t engine=innodb;
		(ALTER command denied)
			alter table t engine=innodb;
			(Query OK)

需要说明的是，图中 set global sync_binlog 这个操作是需要 super 权限的。

可以看到，虽然用户 ua 的 super 权限在 T3 时刻已经通过 revoke 语句回收了，但是在 T4 时刻执行 set global 的时候，权限验证还是通过了。这是因为 super 是全局权限，这个**权限信息在线程对象中**，而 revoke 操作影响不到这个线程对象。

而在 T5 时刻去掉 ua 对 db1 库的所有权限后，在 T6 时刻 session B 再操作 db1 库的表，就会报错“权限不足”。这是因为 acl_dbs 是一个全局数组，所有线程判断 db 权限都用这个数组，这样 revoke 操作马上就会影响到 session B。

这里在代码实现上有一个特别的逻辑，**如果当前会话已经处于某一个 db 里面，之前 use 这个库的时候拿到的库权限会保存在会话变量中。**

你可以看到在 T6 时刻，session C 和 session B 对表 t 的操作逻辑是一样的。但是 session B 报错，而 session C 可以执行成功。这是因为 session C 在 T2 时刻执行的 use db1，拿到了这个库的权限，在切换出 db1 库之前，session C 对这个库就一直有权限。

表权限和列权限

除了 db 级别的权限外，MySQL 支持更细粒度的表权限和列权限。其中，表权限定义存放在表 mysql.tables_priv 中，列权限定义存放在表 mysql.columns_priv 中。这两类权限，组合起来存放在内存的 hash 结构 column_priv_hash 中。

这两类权限的赋权命令如下：

AI写代码 登录复制

```
1 create table db1.t1(id int, a int);
2
3 grant all privileges on db1.t1 to 'ua'@'%' with grant option;
4 GRANT SELECT(id), INSERT (id,a) ON mydb.mytbl TO 'ua'@'%' with grant option;
```

跟 db 权限类似，这两个权限每次 grant 的时候都会修改数据表，也会同步修改内存中的 hash 结构。因此，**对这两类权限的操作，也会马上影响到已经存在的连接。**

看到这里，你一定会问，看来 grant 语句都是即时生效的，那这么看应该就不需要执行 flush privileges 语句了呀。

答案也确实是这样的。

flush privileges 命令会清空 acl_users 数组，然后从 mysql.user 表中读取数据重新加载，重新构造一个 acl_users 数组。也就是说，以数据表中的数据为准，会将全局权限内存数组重新加载一遍。

同样地，对于 db 权限、表权限和列权限，MySQL 也做了这样的处理。

也就是说，如果内存的权限数据和磁盘数据表相同的话，不需要执行 flush privileges。而如果我们都是用 grant/revoke 语句来执行的话，内存和数据表本来就是保持同步更新的。

因此，正常情况下，grant 命令之后，没有必要跟着执行 flush privileges 命令。

flush privileges 使用场景

那么，flush privileges 是在什么时候使用呢？显然，当数据表中的权限数据跟内存中的权限数据不一致的时候，flush privileges 语句可以用来重建内存数据，达到一致状态。

这种不一致往往是由不规范的操作导致的，比如直接用 DML 语句操作系统权限表。我们来看一下下面这个场景：

	client A	client B
T1	connect(root, root) create user 'ua'@'%' identified by 'pa';	
T2		connect(ua,pa) (connect ok) disconnect
T3	delete from mysql.user where user='ua';	
T4		connect(ua,pa) (connect ok) disconnect
T5	flush privileges;	
T6		connect(ua,pa) (Access Denied)

可以看到，T3 时刻虽然已经用 delete 语句删除了用户 ua，但是在 T4 时刻，仍然可以用 ua 连接成功。原因就是，这时候内存中 acl_users 数组中还有这个用户，因此系统判断时认为用户还正常存在。

在 T5 时刻执行过 flush 命令后，内存更新，T6 时刻再要用 ua 来登录的话，就会报错“无法访问”了。

直接操作系统表是不规范的操作，这个不一致状态也会导致一些更“诡异”的现象发生。比如，前面这个通过 delete 语句删除用户的例子，就会出现下面的情况：

	client A
T1	connect(root, root) create user 'ua'@'%' identified by 'pa';
T2	
T3	delete from mysql.user where user='ua';
T4	grant super on *.* to 'ua'@'%' with grant option; ERROR 1133 (42000): Can't find any matching row in the user table
T5	create user 'ua'@'%' identified by 'pa'; ERROR 1396 (HY000): Operation CREATE USER failed for 'ua'@'%'

可以看到，由于在 T3 时刻直接删除了数据表的记录，而内存的数据还存在。这就导致了：

1. T4 时刻给用户 ua 赋权限失败，因为 mysql.user 表中找不到这行记录；
2. 而 T5 时刻要重新创建这个用户也不行，因为在做内存判断的时候，会认为这个用户还存在。

小结

grant 语句会同时修改数据表和内存，判断权限的时候使用的是内存数据。因此，规范地使用 grant 和 revoke 语句，是不需要随后加上 flush privileges 语句的。

flush privileges 语句本身会用数据表的数据重建一份内存权限数据，所以在权限数据可能存在不一致的情况下再使用。而这种不一致往往是由于直接用 DML 语句操作系统权限表导致的，所以我们尽量不要使用这类语句。

权限	磁盘存储	内存存储	修改策略
全局权限	表 mysql.user	数组 acl_users	已存在的连接不生效，新建连接立即生效
db权限	表 mysql.db	数组 acl_dbs	所有连接立即生效
表权限	表 mysql.tables_priv	和列权限组合的hash结构 column_priv_hash	所有连接立即生效
列权限	表 mysql.columns_priv	和表权限组合的hash结构 column_priv_hash	所有连接立即生效