

36 | 数据库没有备份，没有使用Binlog的情况下，如何恢复数据？

2019-09-16 陈旸

SQL必知必会

[进入课程 >](#)



讲述：陈旸

时长 12:28 大小 14.28M



我们上节课讲解了 MySQL 的复制技术，通过主从同步可以实现读写分离，热备份，让服务器更加高可用。MySQL 的复制主要是通过 Binlog 来完成的，Binlog 记录了数据库更新的事件，从库 I/O 线程会向主库发送 Binlog 更新的请求，同时主库二进制转储线程会发送 Binlog 给从库作为中继日志进行保存，然后从库会通过中继日志重放，完成数据库的同步更新。这种同步操作是近乎实时的同步，然而也有人误操作情况的发生，比如 DBA 人员为了方便直接在生产环境中对数据进行操作，或者忘记了当前是在开发环境，还是在生产环境中，就直接对数据库进行操作，这样很有可能会造成数据的丢失，情况严重时，误操作还有可能同步给从库实时更新。不过我们依然有一些策略可以防止这种误操作，比如利用延迟备份的机制。延迟备份最大的作用就是避免这种“手抖”的情况，让我们在延迟从库进行误操作前停止下来，进行数据库的恢复。

当然如果我们对数据库做过时间点备份，也可以直接恢复到该时间点。不过我们今天要讨论的是一个特殊的情况，也就是在没做数据库备份，没有开启使用 Binlog 的情况下，尽可能地找回数据。

今天的内容主要包括以下几个部分：

1. InnoDB 存储引擎中的表空间是怎样的？两种表空间存储方式各有哪些优缺点？
2. 如果.ibd 文件损坏了，数据该如何找回？
3. 如何模拟 InnoDB 文件的损坏与数据恢复？

InnoDB 存储引擎的表空间

InnoDB 存储引擎的文件格式是.ibd 文件，数据会按照表空间 (tablespace) 进行存储，分为共享表空间和独立表空间。如果想要查看表空间的存储方式，我们可以对 `innodb_file_per_table` 变量进行查询，使用 `show variables like 'innodb_file_per_table';`。ON 表示独立表空间，而 OFF 则表示共享表空间。

```
mysql> show variables like 'innodb_file_per_table';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| innodb_file_per_table | ON |
+-----+-----+
1 row in set, 1 warning (0.02 sec)
```

如果采用共享表空间的模式，InnoDB 存储的表数据都会放到共享表空间中，也就是多个数据表共用一个表空间，同时表空间也会自动分成多个文件存放到磁盘上。这样做的好处在于单个数据表的大小可以突破文件系统大小的限制，最大可以达到 64TB，也就是 InnoDB 存储引擎表空间的上限。不足也很明显，多个数据表存放到一起，结构不清晰，不利于数据的找回，同时将所有数据和索引都存放到一个文件中，也会使得共享表空间的文件很大。

采用独立表空间的方式可以让每个数据表都有自己的物理文件，也就是 `table_name.ibd` 的文件，在这个文件中保存了数据表中的数据、索引、表的内部数据字典等信息。它的优势在于每张表都相互独立，不会影响到其他数据表，存储结构清晰，利于数据恢复，同时数据表还可以在不同的数据库之间进行迁移。

如果.ibd 文件损坏了，数据如何找回

如果我们之前没有做过全量备份，也没有开启 Binlog，那么我们还可以通过 .ibd 文件进行数据恢复，采用独立表空间的方式可以很方便地对数据库进行迁移和分析。如果我们误删除 (DELETE) 某个数据表或者某些数据行，也可以采用第三方工具回数据。

我们这里可以使用 Percona Data Recovery Tool for InnoDB 工具，能使用工具进行修复是因为我们在使用 DELETE 的时候是逻辑删除。我们之前学习过 InnoDB 的页结构，在保存数据行的时候还有个删除标记位，对应的是页结构中的 delete_mask 属性，该属性为 1 的时候标记了记录已经被逻辑删除，实际上并不是真的删除。不过当有新的记录插入的时候，被删除的行记录可能会被覆盖掉。所以当我们发生了 DELETE 误删除的时候，一定要第一时间停止对误删除的表进行更新和写入，及时将 .ibd 文件拷贝出来并进行修复。

如果已经开启了 Binlog，就可以使用闪回工具，比如 mysqlbinlog 或者 binlog2sql，从工具名称中也能看出来它们都是基于 Binlog 来做的闪回。原理就是因为 Binlog 文件本身保存了数据库更新的事件 (Event)，通过这些事件可以帮我们重现数据库的所有更新变化，也就是 Binlog 回滚。

下面我们就来看下没有做过备份，也没有开启 Binlog 的情况下，如果 .ibd 文件发生了损坏，如何通过数据库自身的机制来进行数据恢复。

实际上，InnoDB 是有自动恢复机制的，如果发生了意外，InnoDB 可以在读取数据表时自动修复错误。但有时候 .ibd 文件损坏了，会导致数据库无法正常读取数据表，这时我们就需要人工介入，调整一个参数，这个参数叫做 `innodb_force_recovery`。

我们可以通过命令 `show variables like 'innodb_force_recovery';` 来查看当前参数的状态，你能看到默认为 0，表示不进行强制恢复。如果遇到错误，比如 ibd 文件中的数据页发生损坏，则无法读取数据，会发生 MySQL 宕机的情况，此时会将错误日志记录下来。

```
mysql> show variables like 'innodb_force_recovery';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| innodb_force_recovery | 0 |
+-----+-----+
1 row in set, 1 warning (0.01 sec)
```

`innodb_force_recovery`参数一共有 7 种状态，除了默认的 0 以外，还可以为 1-6 的取值，分别代表不同的强制恢复措施。

当我们需要强制恢复的时候，可以将`innodb_force_recovery`设置为 1，表示即使发现了损坏页也可以继续让服务运行，这样我们就可以读取数据表，并且对当前损坏的数据表进行分析和备份。

通常`innodb_force_recovery`参数设置为 1，只要能正常读取数据表即可。但如果参数设置为 1 之后还无法读取数据表，我们可以将参数逐一增加，比如 2、3 等。一般来说不需要将参数设置到 4 或以上，因为这有可能对数据文件造成永久破坏。另外当`innodb_force_recovery`设置为大于 0 时，相当于对 InnoDB 进行了写保护，只能进行 SELECT 读取操作，还是有限制的读取，对于 WHERE 条件以及 ORDER BY 都无法进行操作。

当我们开启了强制恢复之后，数据库的功能会受到很多限制，我们需要尽快把有问题的数据表备份出来，完成数据恢复操作。整体的恢复步骤可以按照下面的思路进行：

1. 使用`innodb_force_recovery`启动服务器

将`innodb_force_recovery`参数设置为 1，启动数据库。如果数据表不能正常读取，需要调大参数直到能读取数据为止。通常设置为 1 即可。

2. 备份数据表

在备份数据之前，需要准备一个新的数据表，这里需要使用 MyISAM 存储引擎。原因很简单，InnoDB 存储引擎已经写保护了，无法将数据备份出来。然后将损坏的 InnoDB 数据表

备份到新的 MyISAM 数据表中。

3. 删除旧表，改名新表

数据备份完成之后，我们可以删除掉原有损坏的 InnoDB 数据表，然后将新表进行改名。

4. 关闭innodb_force_recovery，并重启数据库

innodb_force_recovery大于 1 的时候会有很多限制，我们需要将该功能关闭，然后重启数据库，并且将数据表的 MyISAM 存储引擎更新为 InnoDB 存储引擎。

InnoDB 文件的损坏与恢复实例


我们刚才说了 InnoDB 文件损坏时的人工操作过程，下面我们用一个例子来模拟下。

生成 InnoDB 数据表

为了简便，我们创建一个数据表 t1，只有 id 一个字段，类型为 int。使用命令 `create table t1(id int);` 即可。

```
mysql> create table t1(id int);
Query OK, 0 rows affected (0.07 sec)
```

然后创建一个存储过程帮我们生成一些数据：

 复制代码

```
1 BEGIN
2 -- 当前数据行
3 DECLARE i INT DEFAULT 0;
4 -- 最大数据行数
5 DECLARE max_num INT DEFAULT 100;
6 -- 关闭自动提交
7 SET autocommit=0;
8 REPEAT
9 SET i=i+1;
10 -- 向 t1 表中插入数据
11 INSERT INTO t1(id) VALUES(i);
12 UNTIL i = max_num
13
```

```
14 END REPEAT;  
15 -- 提交事务  
16 COMMIT;  
    END
```

然后我们运行 `call insert_t1()`，这个存储过程帮我们插入了 100 条数据，这样我们就有了 `t1.ibd` 这个文件。

模拟损坏.ibd 文件

实际工作中我们可能会遇到各种各样的情况，比如 .ibd 文件损坏等，如果遇到了数据文件的损坏，MySQL 是无法正常读取的。在模拟损坏 .ibd 文件之前，我们需要先关闭掉 MySQL 服务，然后用编辑器打开 `t1.ibd`，类似下图所示：

```
1  f8b9 2c71 0000 0000 0001 388d 0000 0001  
2  0000 0000 02f4 8d08 0008 0000 0000 0000  
3  0000 0000 0050 0000 0050 0000 0000 0000  
4  0007 0000 0040 0000 4021 0000 0005 0000  
5  0000 ffff ffff 0000 ffff ffff 0000 0000  
6  0001 0000 0000 009e 0000 0000 009e 0000  
7  0000 ffff ffff 0000 ffff ffff 0000 0000  
8  0000 0000 0005 0000 0000 ffff ffff 0000  
9  ffff ffff 0000 0000 0001 0000 0002 0026  
10 0000 0002 0026 0000 0000 0000 0000 ffff  
11 ffff 0000 ffff ffff 0000 0000 0002 aafe  
12 ffff ffff ffff ffff ffff ffff ffff 0000  
13 0000 0000 0000 0000 0000 0000 0000 0000  
14 0000 0000 0000 0000 0000 0000 0000 0000  
15 0000 0000 0000 0000 0000 0000 0000 0000  
16 0000 0000 0000 0000 0000 0000 0000 0000  
17 0000 0000 0000 0000 0000 0000 0000 0000
```

文件是有二进制编码的，看不懂没有关系，我们只需要破坏其中的一些内容即可，比如我在 t1.ibd 文件中删除了 2 行内容（文件大部分内容为 0，我们在文件中间部分找到一些非 0 的取值，然后删除其中的两行：4284 行与 4285 行，原 ibd 文件和损坏后的 ibd 文件见 [GitHub 地址](#)。其中 t1.ibd 为创建的原始数据文件,t1- 损坏.ibd 为损坏后的数据文件，你需要自己创建 t1 数据表，然后将 t1- 损坏.ibd 拷贝到本地，并改名为 t1.ibd）。

然后我们保存文件，这时.ibd 文件发生了损坏，如果我们没有打开 innodb_force_recovery，那么数据文件无法正常读取。为了能读取到数据表中的数据，我们需要修改 MySQL 的配置文件，找到[mysqld]的位置，然后再下面增加一行 innodb_force_recovery=1。

```
73 [mysqld]
74 innodb_force_recovery=1
75 # The next three options are mutually exclusive to SERVER_PORT below.
76 # skip-networking=
77 # enable-named-pipe=
78 # shared-memory=
```

备份数据表

当我们设置innodb_force_recovery参数为 1 的时候，可以读取到数据表 t1 中的数据，但是数据不全。我们使用SELECT * FROM t1 LIMIT 10;读取当前前 10 条数据。

```
mysql> SELECT * FROM t1 LIMIT 10;
+-----+
| id    |
+-----+
| 1     |
| 2     |
| 3     |
| 4     |
| 5     |
| 6     |
| 7     |
| 8     |
| 9     |
| 10    |
+-----+
10 rows in set (0.00 sec)
```

但是如果我们想要完整的数据，使用 `SELECT * FROM t1 LIMIT 100;` 就会发生如下错误。

```
mysql> SELECT * FROM t1 LIMIT 100;
ERROR 2013 (HY000): Lost connection to MySQL server during query
```

这是因为读取的部分包含了已损坏的数据页，我们可以采用二分查找判断数据页损坏的位置。这里我们通过实验，可以得出只有最后一个记录行收到了损坏，而前 99 条记录都可以正确读出（具体实验过程省略）。

这样我们就能判断出来有效的数据行的位置，从而将它们备份出来。首先我们创建一个相同的表结构 t2，存储引擎设置为 MyISAM。我刚才讲过这里使用 MyISAM 存储引擎是因为

在innodb_force_recovery=1的情况下，无法对 innodb 数据表进行写数据。使用命令
CREATE TABLE t2(id int) ENGINE=MyISAM;。

然后将数据表 t1 中的前 99 行数据复制给 t2 数据表，使用：

```
1 INSERT INTO t2 SELECT * FROM t1 LIMIT 99;
```

 复制代码

```
mysql> CREATE TABLE t2(id int) ENGINE=MyISAM;
Query OK, 0 rows affected (0.10 sec)

mysql> INSERT INTO t2 SELECT * FROM t1 LIMIT 99;
Query OK, 99 rows affected (0.02 sec)
Records: 99  Duplicates: 0  Warnings: 0
```

我们刚才讲过在分析 t1 数据表的时候无法使用 WHERE 以及 ORDER BY 等子句，这里我们可以实验一下，如果想要查询 id<10 的数据行都有哪些，那么会发生如下错误。原因是损坏的数据页无法进行条件判断。

```
mysql> SELECT * FROM t1 WHERE id < 10;
ERROR 2013 (HY000): Lost connection to MySQL server during query
```

删除旧表，改名新表

刚才我们已经恢复了大部分的数据。虽然还有一行记录没有恢复，但是能找到绝大部分的数据也是好的。然后我们就需要把之前旧的数据表删除掉，使用 DROP TABLE t1;。

```
mysql> DROP TABLE t1;
Query OK, 0 rows affected (0.08 sec)
```

更新表名，将数据表名称由 t2 改成 t1，使用 RENAME TABLE t2 to t1;。

```
mysql> RENAME TABLE t2 to t1;
Query OK, 0 rows affected (0.05 sec)
```

将新的数据表 t1 存储引擎改成 InnoDB，不过直接修改的话，会报如下错误：

```
mysql> ALTER TABLE t1 engine = InnoDB;
Query OK, 99 rows affected (0.08 sec)
Records: 99 Duplicates: 0 Warnings: 0
```

关闭innodb_force_recovery，并重启数据库

因为上面报错，所以我们需要将 MySQL 配置文件中的innodb_force_recovery=1删除掉，然后重启数据库。最后将 t1 的存储引擎改成 InnoDB 即可，使用ALTER TABLE t1 engine = InnoDB;。

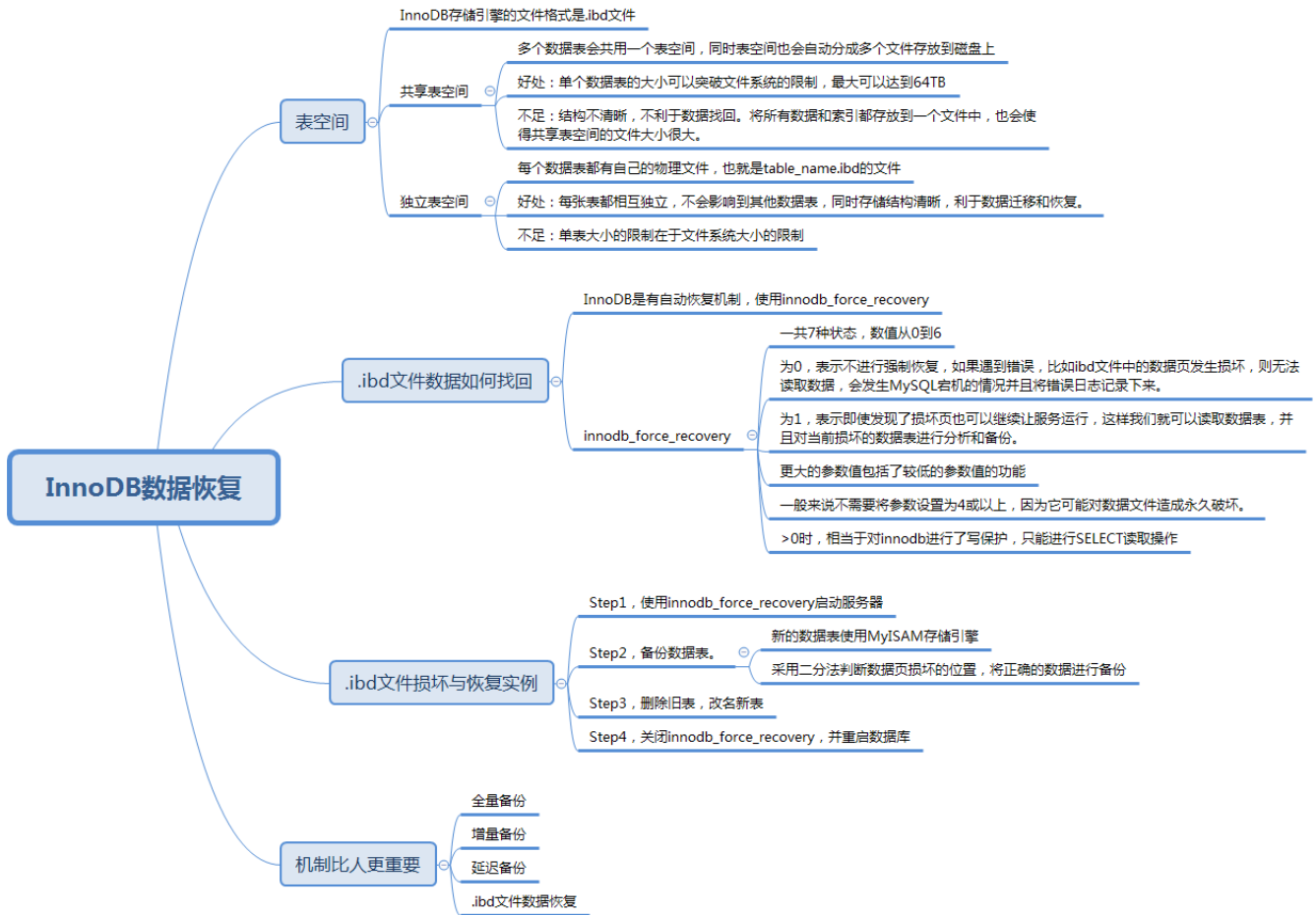
```
mysql> ALTER TABLE t1 engine = InnoDB;
Query OK, 99 rows affected (0.08 sec)
Records: 99 Duplicates: 0 Warnings: 0
```

总结

我们刚才人工恢复了损坏的 ibd 文件中的数据，虽然没有 100% 找回，但是相比于束手无措来说，已经是不幸中的万幸，至少我们还可以把正确的数据页中的记录成功备份出来，尽可能恢复原有的数据表。在这个过程中相信你应该对 ibd 文件，以及 InnoDB 自身的强制恢复（Force Recovery）机制有更深入的了解。

数据表损坏，以及人为的误删除都不是我们想要看到的情况，但是我们不能指望运气，或者说我们不能祈祷这些事情不会发生。在遇到这些情况的时候，应该通过机制尽量保证数据库的安全稳定运行。这个过程最主要的就是应该及时备份，并且开启二进制日志，这样当有误操作的时候就可以通过数据库备份以及 Binlog 日志来完成数据恢复。同时采用延迟备份的策略也可以尽量抵御误操作。总之，及时备份是非常有必要的措施，同时我们还需要定时验证备份文件的有效性，保证备份文件可以正常使用。

如果你遇到了数据库 ibd 文件损坏的情况，并且没有采用任何的备份策略，可以尝试使用 InnoDB 的强制恢复机制，启动 MySQL 并且将损坏的数据表转储到 MyISAM 数据表中，尽可能恢复已有的数据。总之机制比人为更靠谱，我们要为长期的运营做好充足的准备。一旦发生了误操作这种紧急情况，不要慌张，及时采取对应的措施才是最重要的。



今天的内容到这里就结束了，我想问问，在日常工作中，你是否遇到过误操作的情况呢？你又是如何解决的？除了我上面介绍的机制外，还有哪些备份的机制可以增强数据的安全性？

欢迎你在评论区写下你的思考，也欢迎把这篇文章分享给你的朋友或者同事，一起交流一下。

更多课程推荐


MySQL 实战 45 讲

从原理到实战，丁奇带你搞懂 MySQL

林晓斌

网名丁奇
前阿里资深技术专家



新版升级：点击「 请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 35 | 数据库主从同步的作用是什么，如何解决数据不一致问题？

下一篇 37 | SQL注入：你的SQL是如何被注入的？

精选留言 (7)

 写留言



Monday

2019-09-16

开启innodb_force_recovery只能进行有限制的select操作，那后续的四步中，怎么还能再删除旧表？

上网查的资料都是innodb_force_recovery>0时，可以select, create, drop但是不可以insert, update, delete。。。恨windows系统下安装的mysql没找到在哪里设置innodb_force_recovery的值，所以没验证。。。

展开 ∨

作者回复: 可能文件隐藏了，需要先将隐藏文件设置为课件，然后在C:\ProgramData\MySQL\MySQL Server 8.0中找到my.ini，然后再添加innodb_force_recovery数值



隰有荷

2019-11-15

老师，我在使用ibd文件进行数据恢复时，进入了my.cnf文件，然后设置了innodb_force_recovery = 1，再重新启动Mysql发现无法启动，然后在log里面看到下面这句话：

```
[ERROR] /usr/sbin/mysqld: unknown variable 'innodb_force_recovery = 1
```

...

展开 ▾



爱思考的仙人球

2019-10-25

原来丢失数据连接这个错误是由于数据损坏造成的

展开 ▾



Venom

2019-10-17

代码可不可以都放上来呀 少一句创建存储过程的语句也很麻烦的。。。



空知

2019-09-17

如果 ibd文件损坏的数据在开头，那会都select不出来吗？

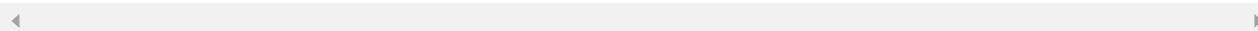


许童童

2019-09-16

磁盘也是逻辑删除，只要文件还没有被覆盖写，也是可以通过物理的方式把数据找回来的。

作者回复: 使得 可以使用恢复工具



蒙开强

2019-09-16

老师，你好，那个存储引擎是可以针对表级设定的么
展开 ∨

作者回复: 对 MySQL中存储引擎是针对数据表的

