MySQL学习笔记（Day021：InnoDB_2–SpaceID.PageNumber/压缩表）

MySQL学习

MySQL学习笔记（Day021：InnoDB_2–SpaceID.PageNumber/压缩表）
一. InnoDB文件（二）
1.Undo表空间文件
2. 重做日志文件
二. 表空间内部组织结构
1. 表空间 – 区
2. 表空间 – 页
1）.页的定义
2）.如何定位到页
3. 压缩表

一. InnoDB文件（二）

1.Undo表空间文件

- `innodb_undo_tablespaces = 3`
  - 创建3个undo表空间

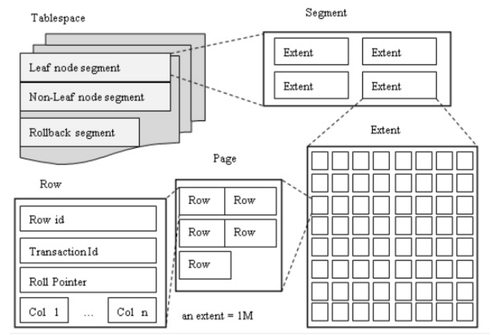
```
shell> ll undo*
-rw-r-----. 1 mysql mysql 7340032 Jan  3 17:21 undo001
-rw-r-----. 1 mysql mysql 7340032 Jan  3 17:40 undo002
-rw-r-----. 1 mysql mysql 7340032 Jan  3 17:21 undo003
```

2. 重做日志文件

- `innodb_log_file_size`
  - 该文件设置的 尽可能的大 ，模板中给出的大小是 4G
  - 设置太小可能会导致 脏页刷新 时hang住

二. 表空间内部组织结构

表空间
内部有多个 段对象（Segment） 组成
每个段（Segment）由 区（Extent） 组成
每个区（Extent）由页（Page）组成
每个页里面报存 数据（ *或者叫记录* Row ）



1. 段 对用户来说是 透明 的
2. 段 也是一个 逻辑概念
3. 目前为止在 `information_schema` 中无法找到段的概念
4. 重点需要理解 区（Extent） 和 页（Page） 的概念

1. 表空间 – 区

- 区是最小的空间申请单位
- 区的大小 固定为**1M**
  - page_size= 16K 就是1M * 1024 / 16 = 64 个页
  - 同理 page_size= 8K 就是 128个页
  - 同理 page_size= 4K 就是 256个页
- 通常说来， 一次申请4个区（4M） 的大小（存在一次申请5个区的时候，但是绝大部分情况就是申请4个区）
- 单个区的 1M 空间内， 物理上是连续 的（一次申请的4个区的空间之间 *（1M和1M之间）* 不保证连续）

2. 表空间 – 页

1）.页的定义

- 页 **是** 最小的**I/O操作** **单位**
  - data 的最小单位不是页，而是页中的 记录（row）
- **普通用户表中MySQL 默认 的每个页为 16K**
  - 从MySQL5.6开始使用 innodb_page_size 可以控制页大小（模板中设置为 8K ）
  - 一旦数据库通过 innodb_page_size 创建完成，则后续无法更改
  - innodb_page_size 是针对 普通表 的， 压缩表 不受其限制

2）.如何定位到页

- **每个 表空间 都 对应 一个 SpaceID ，而 表空间 又对应一个 ibd文件 ，那么一个 ibd文件 也对应一个 SpaceID**
  - 因为 表空间 <-> idb文件 ， 表空间 <-> SpaceID ，所以 ibd文件 <-> SpaceID
  - ibdata1 对应的 SpaceID 为 0
  - 每创建一个 表空间（ibd文件） ， SpaceID 自增长（全局）
- **PageNumber**
  - 在一个表空间中，第 几 个16K的页 *（假设 innodb_page_size = 16K ）* 即为 PageNumber


```
# 每次读取 Page 时，都是通过 SpaceID 和 PageNumber 进行读取；

# 可以简单理解为从表空间的开头读多少个 PageNumber * PageSize 的字节（偏移量）

# 想像成数组，数组的名字就是 SpaceID， 数组的下标就是PageNumber
```

1. 可以通过（ SpaceID ， PageNumber ）定位到某一个页
2. 在一个 SpaceID（ibd文件）中， PageNumber 是 唯一且自增 的
3. 这里的 区（extent） 的概念已经弱化。在这个例子中，第一个区的PageNumber是 （0~63） 且这 64个页在物理上是连续 的；第二个区的PageNumber是 （64~127） 且这 64个页在物理上也是连续 的；但是 （0~63） 和 （64~127）之间 在物理上则 不一定是连续的 ，因为区和区之间在物理上不一定是连续的。
4. 删除表的时候， SpaceID不会回收 ，SpaceID是全局自增长的。

```
mysql> select * from information_schema.innodb_sys_tables limit 1\G  -- INNODB_SYS_TABLES 表
*************************** 1. row ***************************
     TABLE_ID: 14
         NAME: SYS_DATAFILES
         FLAG: 0
       N_COLS: 5
        SPACE: 0   -- 这个就是SpaceID，由于这个表存放在ibdata1中，所以SpaceID是0
  FILE_FORMAT: Antelope
   ROW_FORMAT: Redundant
ZIP_PAGE_SIZE: 0
   SPACE_TYPE: System
1 row in set (0.00 sec)

mysql> select name, space, table_id from information_schema.innodb_sys_tables where space=0 ;
+-----------------+-------+----------+
| name            | space | table_id |
+-----------------+-------+----------+
| SYS_DATAFILES   |     0 |       14 |
| SYS_FOREIGN     |     0 |       11 |
| SYS_FOREIGN_COLS |     0 |       12 |
| SYS_TABLESPACES |     0 |       13 |
| SYS_VIRTUAL     |     0 |       15 |
+-----------------+-------+----------+
5 rows in set (0.00 sec)

mysql> select name, space, table_id from information_schema.innodb_sys_tables where space<>0 limit 5 ;
+--------------------+-------+----------+
| name               | space | table_id |
+--------------------+-------+----------+
| burn_test/Orders   |    77 |       89 |
| burn_test/Orders_MV |    79 |       91 |
| burn_test/child    |    37 |       52 |
| burn_test/parent   |    33 |       49 |
| burn_test/t1       |    58 |       78 |
+--------------------+-------+----------+
5 rows in set (0.00 sec)

-- 独立表空间的table_id 和 SpaceID 一一对应
-- 共享表空间是多个table_id 对应 一个 SpaceID
```

3. 压缩表

> 官方文档

- 基于页的压缩
- 每个表的页大小可以不同（针对压缩表来讲）

```
-- ger3_space的file_block_size=4096，不是innodb_page_size的大小
-- 所在在创建 普通表 的时候，报错了
mysql> create table  test_ger (a int) tablespace=ger3_space;
ERROR 1478 (HY000): InnoDB: Tablespace `ger3_space` uses block size 4096 and cannot contain a table with physical page size 8192

-- 使用压缩表的方式
mysql> create table comps_test1 (a int) row_format=compressed, key_block_size=4; -- 1K, 2K, 4K, 8K, 16K 只有这几个页大小可以选择
Query OK, 0 rows affected (0.13 sec)

-- 在之前的ger3_space中创建压缩表
mysql> create table comps_test2 (a int)tablespace=ger3_space row_format=compressed, key_block_size=4;
-- 由于ger3_space是4K的，所以这里页大小也只能是4K
Query OK, 0 rows affected (0.09 sec)

-- 修改存在的表变成压缩表
mysql> alter table t1 row_format=compressed,key_block_size=4;
Query OK, 0 rows affected (0.17 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

> **注意：**
> 虽然SQL语法中写的是 `row_format=compressed` ，但是 压缩是针对页 的，而 不是记录 ；即 读页 的时候 解压 ， 写页 的时候 压缩 ，并不会在读取或写入单个记录（row）时就进行解压或压缩操作。

- **key_block_size的含义**
  - `key_block_size` 的可选项是1k，2k，4k，8k，16k（是页大小，不是比例）
  - 不是将原来 `innodb_page_size` 页大小的数据压缩成 `key_block_size` 的页大小，因为有些数据可能不能压缩，或者压缩不到那么小
  - 压缩是将原来的页的数据通过压缩算法压缩到一定的大小，然后用 `key_block_size` 大小的页去存放。
    - 比如原来的 `innodb_page_size` 大小是 `16K` ，现在的 `key_block_size` 设置为 `8K` ；
    - 某表的数据大小是 `24k` ，原先使用 `2` 个 `16k` 的页存放；
    - 压缩后，数据从 `24k —> 18k` ；
    - 由于现在的 `key_block_size=8k` ，所以需要 `3` 个 `8K` 的页存放压缩后的 `18k` 数据
    - 多余的空间可以留给下次插入或者更新

> 压缩比和设置的 `key_block_size` 没有关系。压缩比看数据本身和算法（zlib）， `key_block_size` 仅仅是设置 存放压缩数据的页大小
>
> **不解压也能插入数据**，通过在剩余空间直接存放 `redo log` ，然后页空间存放满后，再解压，利用 `redo log` 更新完成后，最后再压缩存放（此时就没有redo log 了）。减少解压和压缩的次数。

- **查看压缩比**

```
mysql>  use employees ;
Database changed

mysql> create table employee_comps_1 like employees;
Query OK, 0 rows affected (0.16 sec)

mysql> alter table employee_comps_1 row_format=compressed,key_block_size=4;
Query OK, 0 rows affected (0.19 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> show create table employee_comps_1\G
*************************** 1. row ***************************
       Table: employee_comps_1
Create Table: CREATE TABLE `employee_comps_1` (
  `emp_no` int(11) NOT NULL,
  `birth_date` date NOT NULL,
  `first_name` varchar(14) NOT NULL,
  `last_name` varchar(16) NOT NULL,
  `gender` enum('M','F') NOT NULL,
  `hire_date` date NOT NULL,
  PRIMARY KEY (`emp_no`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 ROW_FORMAT=COMPRESSED KEY_BLOCK_SIZE=4
1 row in set (0.00 sec)

-- 插入数据
mysql> insert into employee_comps_1 select * from employees;
Query OK, 300024 rows affected (8.10 sec)
Records: 300024  Duplicates: 0  Warnings: 0

-- 查看压缩比
mysql> use information_schema;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from INNODB_CMP;
+-----------+-------------+----------------+---------------+----------------+-----------------+
| page_size | compress_ops | compress_ops_ok | compress_time | uncompress_ops | uncompress_time |
+-----------+-------------+----------------+---------------+----------------+-----------------+
|      1024 |           0 |              0 |             0 |              0 |               0 |
|      2048 |           0 |              0 |             0 |              0 |               0 |
|      4096 |       12687 |          11451 |             1 |           1236 |               0 |
|      8192 |           0 |              0 |             0 |              0 |               0 |
|     16384 |           0 |              0 |             0 |              0 |               0 |
+-----------+-------------+----------------+---------------+----------------+-----------------+
5 rows in set (0.00 sec)

mysql> select 11451/12687;  -- compress_ops_ok / compress_ops
+-------------+
| 11451/12687 |
+-------------+
|      0.9026 |  -- 压缩比在90%
+-------------+
1 row in set (0.00 sec)


mysql> select * from INNODB_CMP_RESET;
-- 查询INOODB_CMP_RESET，会把INNODB_CMP表中的数据复制过来，并清空INNODB_CMP
+-----------+-------------+----------------+---------------+----------------+-----------------+
| page_size | compress_ops | compress_ops_ok | compress_time | uncompress_ops | uncompress_time |
+-----------+-------------+----------------+---------------+----------------+-----------------+
|      1024 |           0 |              0 |             0 |              0 |               0 |
|      2048 |           0 |              0 |             0 |              0 |               0 |
|      4096 |       12687 |          11451 |             1 |           1236 |               0 |
|      8192 |           0 |              0 |             0 |              0 |               0 |
|     16384 |           0 |              0 |             0 |              0 |               0 |
+-----------+-------------+----------------+---------------+----------------+-----------------+
5 rows in set (0.00 sec)

mysql> select * from INNODB_CMP;  -- 查询该表，数据已经被清空了
+-----------+-------------+----------------+---------------+----------------+-----------------+
| page_size | compress_ops | compress_ops_ok | compress_time | uncompress_ops | uncompress_time |
+-----------+-------------+----------------+---------------+----------------+-----------------+
|      1024 |           0 |              0 |             0 |              0 |               0 |
|      2048 |           0 |              0 |             0 |              0 |               0 |
|      4096 |           0 |              0 |             0 |              0 |               0 |
|      8192 |           0 |              0 |             0 |              0 |               0 |
|     16384 |           0 |              0 |             0 |              0 |               0 |
+-----------+-------------+----------------+---------------+----------------+-----------------+
5 rows in set (0.00 sec)
-- 注意，这个表里面的数据是累加的，是全局信息，没法对应到某一张表


shell> ll -h employee*.ibd  # 可以看出磁盘占用还是有明显减小的
-rw-r-----. 1 mysql mysql 14M Jan  4 13:41 employee_comps_1.ibd
-rw-r-----. 1 mysql mysql 22M Dec  2 21:32 employees.ibd
```

```
mysql> show variables like "%innodb_cmp_per_index%";
+----------------------------------+-------+
| Variable_name                    | Value |
+----------------------------------+-------+
| innodb_cmp_per_index_enabled | OFF   |  -- 该功能目前是关闭的
+----------------------------------+-------+
1 row in set (0.00 sec)

mysql> set global innodb_cmp_per_index_enabled=1;
Query OK, 0 rows affected (0.00 sec)

mysql> show variables like "%innodb_cmp_per_index%";
+----------------------------------+-------+
| Variable_name                    | Value |
+----------------------------------+-------+
| innodb_cmp_per_index_enabled | ON    |
+----------------------------------+-------+
1 row in set (0.00 sec)

mysql> use employees
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> create table employee_comps_2k like employees;
Query OK, 0 rows affected (0.13 sec)

mysql> alter table employee_comps_2k row_format=compressed,key_block_size=2;  -- 设置成2K的页大小
Query OK, 0 rows affected (0.18 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> insert into employee_comps_2k select * from employees; -- 插入数据
Query OK, 300024 rows affected (9.68 sec)
Records: 300024  Duplicates: 0  Warnings: 0

mysql> use information_schema;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from INNODB_CMP;
+-----------+--------------+-----------------+---------------+-----------------+------------------+
| page_size | compress_ops | compress_ops_ok | compress_time | uncompress_ops | uncompress_time |
+-----------+--------------+-----------------+---------------+-----------------+------------------+
|      1024 |            0 |               0 |             0 |               0 |                0 |
|      2048 |        34676 |           23729 |             2 |           10947 |                0 |
|      4096 |            0 |               0 |             0 |               0 |                0 |
|      8192 |            0 |               0 |             0 |               0 |                0 |
|     16384 |            0 |               0 |             0 |               0 |                0 |
+-----------+--------------+-----------------+---------------+-----------------+------------------+
5 rows in set (0.00 sec)

mysql> select 23729/34676;
+-------------+
| 23729/34676 |
+-------------+
|      0.6843 |  -- 2K时，压缩比是68%
+-------------+
1 row in set (0.00 sec)

mysql> select * from INNODB_CMP_PER_INDEX;  -- 开启innodb_cmp_per_index_enabled才有数据
+---------------+------------------+------------+--------------+-----------------+---------------+-----------------+------------------+
| database_name | table_name       | index_name | compress_ops | compress_ops_ok | compress_time | uncompress_ops | uncompress_time |
+---------------+------------------+------------+--------------+-----------------+---------------+-----------------+------------------+
| employees     | employee_comps_2k | PRIMARY    |        34676 |           23729 |             2 |           10947 |                0 |
+---------------+------------------+------------+--------------+-----------------+---------------+-----------------+------------------+
1 row in set (0.00 sec)

-- 可以看到employees.employee_comps_2k这个表的 索引的压缩比（在INNODB中索引即数据）；
-- 在page_size=2K只有一个压缩表的时候，INNODB_CMP和INNODB_CMP_PER_INDEX的值是一样的，并且能够知道是哪个表的情况
```

> **innodb_cmp_per_index_enabled 这个参数默认关闭，开启对性能有影响**