PERCONA
Performance Consulting Experts

Data Recovery for MySQL

Istvan Podor

FOSDEM – 5 / February / 2011

Brussels

Introduction

- Consultant at Percona
- Performance engineer previously
- Replacing Ewen
- Data recovery for dummies :)

About the presentation

- InnoDB recovery
- Not explaining deep innodb architecture
- Showing you an example instead
- Feel free to interrupt

Data set

- Sample database available at http://downloads.mysql.com/docs/sakila-db.zip
- Table: `store`
- 2 rows only

```
Create Table: CREATE TABLE `store` (
  `store_id` tinyint(3) unsigned NOT NULL auto_increment,
  `manager_staff_id` tinyint(3) unsigned NOT NULL,
  `address_id` smallint(5) unsigned NOT NULL,
  `last_update` timestamp NOT NULL default CURRENT_TIMESTAMP on update CURRENT_TIMESTAMP,
  PRIMARY KEY  (`store_id`),
  UNIQUE KEY `idx_unique_manager` (`manager_staff_id`),
  KEY `idx_fk_address_id` (`address_id`),
  CONSTRAINT `fk_store_address` FOREIGN KEY (`address_id`) REFERENCES `address` (`address_id`) ON UPDATE CASCADE,
  CONSTRAINT `fk_store_staff` FOREIGN KEY (`manager_staff_id`) REFERENCES `staff` (`staff_id`) ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8
1 row in set (0.00 sec)
```

Some basic things about InnoDB

- InnoDB not storing tables, but primary key (index) Ids
- Split common table space (ibdata) in to pages
- Find the belonging index id to your table
- Create a table definition file
- Extract 'lost' data
- Questions before take off?

Table content

- As simple as:

```
debianvm:~/recovery# mysql sakila -e 'select * from store';
+----------+-----------------+------------+---------------------+
| store_id | manager_staff_id | address_id | last_update         |
+----------+-----------------+------------+---------------------+
|        1 |               1 |          1 | 2006-02-15 04:57:12 |
|        2 |               2 |          2 | 2006-02-15 04:57:12 |
+----------+-----------------+------------+---------------------+
```

Let's drop it

- Just drop the table from mysql
- Grab the ibdata file on it's own, make a copy

Get the tool

- Download and compile first:
  - https://launchpad.net/percona-innodb-recovery-tool
- Just simply run `make` to compile

Run page_parser

- Nothing needed to be done
- Print a lots of output

```
debianvm:~/recovery# percona-innodb-recovery-tool/page_parser -4 -f ibdata1
Opening file: ibdata1
Read data from fn=3...
Read page #0.. saving it to pages-1296031385/0-26083328/0-00000000.page
Read page #1.. saving it to pages-1296031385/0-0/1-00000001.page
Read page #2.. saving it to pages-1296031385/4294967295-65535/2-00000002.page
Read page #3.. saving it to pages-1296031385/0-0/3-00000003.page
Read page #4.. saving it to pages-1296031385/4294967295-0/4-00000004.page
Read page #5.. saving it to pages-1296031385/65535-4294901760/5-00000005.page
Read page #6.. saving it to pages-1296031385/2-28442624/6-00000006.page
Read page #7.. saving it to pages-1296031385/10-8/7-00000007.page
Read page #8.. saving it to pages-1296031385/0-1/8-00000008.page
Read page #9.. saving it to pages-1296031385/0-5/9-00000009.page
```

- Our workdir is 'pages-1296031385

Extract basic SYS files

- InnoDB store indexes and not tables
- That's what you can see in the pages-XXX directory
- Let's try to locate where is our `store` table …

Get systables and sysindexes

- This is the way to find where the belonging pages stored to the `store` table
- There are two files in include/
  - table_defs.h.SYS_INDEXES
  - table_defs.h.SYS_TABLES
- sys_tables are always in pages_xxx/0-1
- sys_indexes are always in pages_xxx/0-3

(things will be clear soon...)

Get sys_tables list

- Make a symlink and recompile
  `cd include/; rm table_defs.h \
  ln -s table_defs.h.SYS_TABLES table_defs.h`
- Recompile …
- Run contraint parser to get the table list with PK Ids (file format is -4 (redundant) for SYS tables/indexes)
  `contraints_parser -4 -f ../pages-1296027482/0-1/* |grep store`

```
debianvm:~/recovery/percona-innodb-recovery-tool# ./constraints_parser -4 -f ../pages-1296027482/0-1/* |grep store
SYS_TABLES      "sakila/store"  43      2147483652      1       0       0       ""      0
```

- 3rd column, PK ID > 43 (remember this)

PERCONA
Performance Consulting Experts

Get sys_indexes list

- Make a symlink and recompile

  `cd include/; rm table_defs.h \

  ln -s table_defs.h.SYS_INDEXES table_defs.h`
- Recompile …
- Run contraint parser to get the table list with PK Ids (file format is -4 (redundant) for SYS tables/indexes)

  `contraints_parser -4 -f ../pages-1296027482/0-3/* |awk '/PRIMARY/ && /43/'`
- Find the 3rd column's value where the second column = 43

```
debianvm:~/recovery/percona-innodb-recovery-tool# ./constraints_parser -4 -f ../pages-1296031385/0-3/* | aw
SYS_INDEXES     25      43      "PRIMARY"       1       3       0       4294967295
SYS_INDEXES     43      91      "PRIMARY"       1       3       0       315
```

- It's 91!

PERCONA
Performance Consulting Experts

Create table_defs again

- Well, data recovery is not easy
- Create the same structured `store` table in mysql
- Recovery tool needs a table definition on its own `language`
- Use create_defs.pl to get definition file for `store`
  - `./create_defs.pl –user root –db sakila –table store > include/table_defs.h`
  - (don't forget to delete the symlink first)
- Recompile

Let's see what we have

- No we can run the constraint parser against real data
  - File format here is compact
- Remember the magic number, 91!

```
debianvm:~/recovery/percona-innodb-recovery-tool# ./constraints_parser -5 -f ../pages-1296031385/0-91/*
store   11      0       6657    0
store   195     152     0       419420418
store   0       242     42648   0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
store   0       0       0       0
```

- Not exactly what we were expected :)

PERCONA
Performance Consulting Experts

Fine tune table_defs.h

- table_defs.pl is not perfect, we can surely tune something on the table_defs.h file
- Try to be a bit more specific
- Store table's content:

```
debianvm:~/recovery# mysql sakila -e 'select * from store';
+----------+-----------------+------------+---------------------+
| store_id | manager_staff_id | address_id | last_update         |
+----------+-----------------+------------+---------------------+
|        1 |               1 |          1 | 2006-02-15 04:57:12 |
|        2 |               2 |          2 | 2006-02-15 04:57:12 |
+----------+-----------------+------------+---------------------+
```

- Notice:
  - First 3 column got only 1 int, last field is timestmap

table_defs.h

- Read the readme for details about file formats
- Parameters for the first column by default and after tuning:

```
name: "store",
{
        { /* tinyint(3) unsigned */
                name: "store_id",
                type: FT_UINT,
                fixed_length: 1,

                has_limits: TRUE,
                limits: {
                        can_be_null: FALSE,
                        uint_min_val: 0,
                        uint_max_val: 255
                },

                can_be_null: FALSE
        },
```

```
has_limits: TRUE,
limits: {
        can_be_null: FALSE,
        uint_min_val: 1,
        uint_max_val: 10_
},
```

- Done on the other 2 int column too
- Recompile

Let's see what's up now

- Running contraints_parser again...

```
debianvm:~/recovery/percona-innodb-recovery-tool# ./constraints_parser -5 -f ../pages-1296031385/0-91/*
store   1       1       1       1139975832
store   2       2       2       1139975832
debianvm:~/recovery/percona-innodb-recovery-tool# mysql sakila -e 'select FROM_UNIXTIME(1139975832);'
+---------------------------+
| FROM_UNIXTIME(1139975832) |
+---------------------------+
| 2006-02-15 04:57:12       |
+---------------------------+
```

- Remember, original data:

```
debianvm:~/recovery# mysql sakila -e 'select * from store';
+----------+-----------------+------------+---------------------+
| store_id | manager_staff_id | address_id | last_update         |
+----------+-----------------+------------+---------------------+
|        1 |               1 |          1 | 2006-02-15 04:57:12 |
|        2 |               2 |          2 | 2006-02-15 04:57:12 |
+----------+-----------------+------------+---------------------+
```

Filepertable

- How to handle that?!

Questions/Comments ?

- It's not easy
- However, you should try!
- Download the slides, try to repeat
- Email me if you have (easy) questions
  - istvan.podor at percona .com
    // www.percona.com
    // https://launchpad.net/percona-innodb-recovery-tool
- We are Hiring!
  - Consultants and Support Engineers

PERCONA
Performance Consulting Experts