Mysql binary log 基本文件格式

Mysql对应的版本为：5.5.19,低于这个版本的日志文件格式没有进行兼容。

Bin-log文件以4个字节的魔数开始， 0xfe 0x62 0x69 0x6e = 0xfe 'b' 'i' 'n'，标识该文件为bin-log文件。很多文件中均包含类似的魔数，bmp文件中包含‘bmp’，jpeg中包含‘jpeg’等类似的魔数标志位。4个字节的魔数之后即为一系列的event事件。

Bin-log文件以event为基本的数据管理单元，每个event都包含一个完整sql操作所影响的数据信息，例如insert 所插入的数据，delete 删除的数据，update替换和更新的数据。

Event struct分为两个部分：event header和event data。Event header主要是event的数据描述信息，data则为event 中描述的操作所影响的详细数据。

|  |
| --- |
| Event header |
| Event data |

|  |  |  |  |
| --- | --- | --- | --- |
| Part | Item | Length | Desc |
| Event header | Timestamp | 4 |  |
| Type\_code | 1 |  |
| Server\_id | 4 |  |
| Event\_length | 4 |  |
| Next\_position | 4 |  |
| Flags | 2 |  |
| Extra\_headers |  | 在v4版本没有采用该字段，为此length为0 |
| Event data | Fixed part |  | 每类event对应该部分是固定的 |
| Variable part |  | 即使是同一类event的长度也是不固定的，需要根据表的数据信息来确定 |

Event data部分中的数据，需要根据对应 event来进行逐步解析。

Event主要分为两种类型：fortmat description event和 data event。

Format description event用于描述binlog文件信息和后续data event中的header length信息；data event，主要为sql操作所影响的数据信息；

Format description event的主要数据格式为：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Event  header | Item name | Start | size | Desc |
| Timestamp | 0 | 4 |  |
| Type\_code | 4 | 1 | =15(for 5.5.19 |
| Server\_id | 5 | 4 |  |
| Event\_length | 9 | 4 |  |
| Next\_position | 13 | 4 |  |
| flags | 17 | 2 |  |
| Event  data | Binlog\_version | 19 | 2 | =4 |
| Server\_version | 21 | 50 |  |
| Create\_timestamp | 71 | 4 |  |
| Header\_length | 75 | 1 |  |
| Post-header lengths for all event types | 76 | N=(event\_length-76) |  |

event\_length >= 91 = 76 + post-header lengths for all event types.(目前该值为27)

**Event header**

Timestamp，4 bytes，描述event开始执行时的时间；

Type\_code，1byte，描述事件类型，例如ROTATE\_EVENT**、**TABLE\_MAP\_EVENT、WRITE\_ROWS\_EVENT、UPDATE\_ROWS\_EVENT、DELETE\_ROWS\_EVENT，format description event对应的值为15；

Server\_id，4 bytes，描述产生事件的server id；

Event\_length，4 bytes，描述event 结构体的数据长度，包括event header 和event data的长度，可以根据这个量值来检查event的数据完整性；

Next\_position，4 bytes，描述下一个event数据开始的位置；

Flags，4 bytes，描述event的标志位，例如文件是否正常关闭等标识信息；

**Event data**

Binlog\_version，2 bytes，对于5.5.19来讲，值为4；

Server\_version，50 bytes，server version,例如5.5.9-log；

Create\_timestamp，4 bytes，描述bin-log文件创建的时间；

Header\_length，1 byte，描述event header的长度，data event中存在extra\_headers变长变量，header\_length – 19为计算出的extra\_headers的长度。

**Data event**为数据描述型event，也是binlog最主要的event，除了文件首部的format description event，其他均为data event。

主要数据格式为：

|  |  |  |  |
| --- | --- | --- | --- |
| Event  header | Item | Start | length |
| Timestamp | 0 | 4 |
| Type\_code | 4 | 1 |
| Server\_id | 5 | 4 |
| Event\_length | 9 | 4 |
| Next\_position | 13 | 4 |
| Flags | 17 | 2 |
| Extra\_headers | 19 | x-19 |
| Event  data | Fixed part | x | y |
| Variable part |  |  |

Event header length的长度为：x，主要被extra\_headers的长度影响，但是在版本为4的binlog中，extra\_header的长度为0，所以event header length 为19；

Event data length的长度为： event\_length – x；

Fixed data length 的长度为： y，实际长度要根据对应的事件类型来具体确定；

Variable part的长度为：event\_length – (x+y)；

**关键事件类型的event data数据格式解析**

关注的事件类型有FORMAT\_DESCRIPTION\_EVENT、TABLE\_MAP\_EVENT、WRITE\_ROWS\_EVENT、UPDATE\_ROWS\_EVENT、DELETE\_ROWS\_EVENT、ROTATE\_EVENT。

(只是说明event data部分，event header不再额外介绍)

**FORMAT\_DESCRIPTION\_EVENT，参照上面的format description event部分。**

**1、TABLE\_MAP\_EVENT，描述表的主要结构信息，例如数据库名、表名、列数据类型等信息。**

|  |  |  |  |
| --- | --- | --- | --- |
| Part | No | Item | Length |
| Fixed data  part | 1 | Tableid | 6 |
| 2 | 保留字段 | 2 |
| Variable data  part | 3 | Db name length | 1 |
| 4 | Db name string | 根据上一字段获取 |
| 5 | Table name length | 1 |
| 6 | Table name string | 根据上一字段获取 |
| 7 | 列数 | Packed integer |
| 8 | 列数据类型 | 一列一个字节 |
| 9 | metadata 长度 | Packed integer |
| 10 | Metadata | 根据上一字段指定 |
| 11 | 列是否允许为空 | INT((列数+7)/8)bytes |

7、9号字段长度均是指定为packed integer。

Packed integer为mysql内部的一种存储变长长度的方法。具体实现方法可以参见：

mysql-5.5.9 /sql-common/pack.c/net\_field\_length。

9号字段metadata数据的内容要根据具体的数据类型来进行解析，在string类型中可以决定数据的长度，在MYSQL\_TYPE\_DATE、MYSQL\_TYPE\_ENUM、MYSQL\_TYPE\_SET的数据类型需要根据meta对应值来来进行数据类型的二次转换

**2、ROTATE\_EVENT，bin-log文件跳转信息，描述event在下一个文件中的起始位置、下一个文件名**

|  |  |  |  |
| --- | --- | --- | --- |
| Part | No | Item | Length |
| Fixed data part | 1 | Event在下一个文件的起始位置 | 8 |
| Variable data part | 2 | 下一个bin-log文件名称 | 需要计算 |

1号event在下一个文件的起始位置为4，文件开始的前4个为bin-log文件标识；

2号下一个bin-log文件名称的长度需要进行计算，计算方式为：

(Event\_length – header\_length -8)，event\_length在本事件的event header part中已经指定，header\_length在format description中已经指定；

**3、WRITE\_ROWS\_EVENT、UPDATE\_ROWS\_EVENT、DELETE\_ROWS\_EVENT，这三个事件都很多共同部分在fixed data part中**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Part | No | Item | Length | Desc |
| Fixed data part | 1 | Table id | 6 |  |
| 2 | 保留字段 | 2 |  |
| Variable data part | 3 | 列数 | Packed integer | 又是packed integer,具体数据又要进行解析 |
| 4 | Bit标志位 | 变长的，计算公式  INT((列数+7)/8)bytes | 解析中没有用到该值 |
| 5 | Bit标志位(跟4一样) | INT((列数+7)/8)bytes | 仅限于update |
| 6 | Bit标志位，只是该列的值是否为空 | INT((列数+7)/8)bytes | 标识该列的值是否为空 |
| 7 | Row\_image | 具体值就自己算吧 | Event操作所影响的列数据 |
| 8 | Bit标志位（同6） |  | 仅限于update |
| 9 | Row\_image |  | 仅限于update |

4、5字段中的值，在解析过程中没有用到，具体意义不明；

6号字段bit标志位，只是在改行数据中，对应列的值是否为空；如果列值为空，则不用具体解析row\_image，（此时row\_image一定为空）

7号字段row\_image数据， event所描述的操作事件所影响的具体数据，例如delete删除的数据，insert 插入的数据，update操作之前的数据、操作之后的数据。