调研报告

李乐程 李奕杉 闵安娜

分工

李乐程：调研HSQLDB的存储文件格式和元数据文件格式，以及在数据读写过程中使用的多版本并发控制（MVCC）机制。
闵安娜：调研MySQL的存储文件格式和元数据文件格式，以及在数据读写过程中使用的多版本并发控制（MVCC）机制。
李奕杉：给出压缩在两种数据库中的作用。完善MySQL存储文件格式，校对报告正确性，给出ThssDB的技术选型。

HSQLDB

HSQLDB是一款轻量级的Java关系型数据库管理系统，支持多种操作系统平台，包括内存模式、磁盘模式等多种模式。表可分为`MEMORY table`和`CACHED table`，前者将数据全部暂存至内存，后者只有部分在内存而其他在硬盘。

元数据文件格式

HSQLDB的元数据信息以SQL语句的形式存储在`.script`文件中，创建数据库时打开或创建。包含数据库的table、constraint等定义的SQL语句。每个检查点(`checkpoint`)会更新一次。

持久化的数据也会以`INSERT`的形式存储在`.script`文件中，构建数据库时一同加入。

存储文件格式

`.data`文件，存放CACHED table的余下数据，采用二进制格式存储。

`.log`文件，记录数据库的未提交修改。

`.backup`文件是压缩的`.data`文件，可以是纪录修改部分或整个文件，和`.log`文件一同用于备份还原。

`.lobs`文件是大二进制数据(LOB)，包括所有LOB对象。

检查点如正常关闭后会进行持久化写入`.script`文件，然后保留放置或清空。

HSQLDB可以使用NIO加快访问，使用操作系统的缓存机制加快读写速度。
在每个`CHECKPOINT`，文件会进行复制，然后在复制的文件上进行修改，修改完成后，将原文件删除，将复制的文件重命名为原文件名。

MVCC机制

HSQLDB有四种通用的隔离级别：`READ UNCOMMITTED`，`READ COMMITTED`，`REPEATABLE READ`，`SERIALIZABLE`。 MVCC的隔离级别有**读一致性**和**快照隔离**，其中`READ UNCOMMITTED`提升至`READ COMMITTED`被归类至读一致性，`REPEATABLE READ`和`SERIALIZABLE`被归类至快照隔离。

每当一个事务开始，HSQLDB会拷贝一份数据库的快照(即当前状态)到内存，在结束前的读操作都是在快照上而不是在原数据库上。 MVCC不使用锁来限制事务间的读写冲突，事务可以同时进行读写操作。

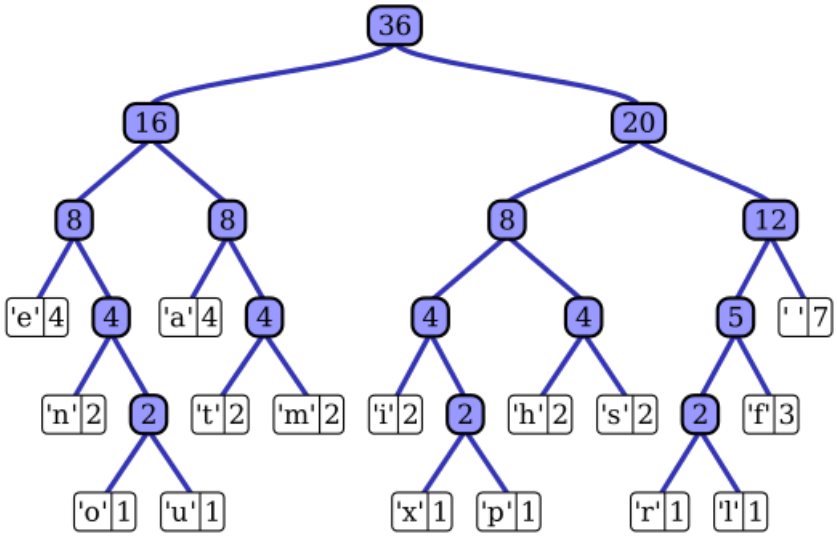
- 读一致性：当两个事务作用时间重叠，如果事务一修改了某行但未提交，而事务二要对该行进行修改，那么事务二必须等待事务一提交后才能修改。
- 快照隔离：当两个事务作用时间重叠，如果事务一修改了某行而事务二要对该行进行修改，不论事务一是否已提交，事务二的修改将会失效并回退或报错。
- 防死锁：发现冲突时，事务可以回退所有操作以允许其他事务继续操作，或者报错但不回退。

特别地，如果一个事务只有读操作，那它会不管其他事务的操作直接执行。

压缩机制

元数据文件默认不进行压缩，但是可以进行压缩（`gzip`，`tar`）。
`BLOB`一般进行压缩，而`CLOB`默认不进行压缩（但可以提供设置选项进行压缩）。

`Gzip`压缩算法：基于`DEFLATE`算法。将文件看做一个连续的字节流，按照其低位到高位的顺序进行编码。
`DEFLATE`算法将文件分为若干个块，每个块包含一个头部和一个数据部分。头部包含了3比特的信息，提供了编码方法（无压缩、静态霍夫曼树、动态霍夫曼树）



`Tar`算法：将多个文件打包成一个文件，不进行压缩。

MySQL

`MySQL`是一种关系型数据库管理系统，用于存储和管理数据。有两种常用存储引擎，`InnoDB`和`MyISAM`。

- `InnoDB`：`InnoDB`是MySQL的默认存储引擎。它采用了MVCC机制实现多版本并发控制，可以提供高并发和高可靠性的数据存储服务。
- `MyISAM`：`MyISAM`是MySQL的另一个常用的存储引擎，不支持事务和行级锁，但支持全文索引。

元数据文件格式

- `InnoDB`：`InnoDB`的元数据信息存储在系统表空间中，包括数据库名、表名、字段名、索引等信息。其中，每个表都有一个对应的`.frm`文件存储表的元数据信息。包含表的模式、定义。可以通过页面压缩来减少存储空间。（将原数据写入空白块，而非实际的空间）。

-`MyISAM`：使用MyISAM引擎的系统数据库使用的是`.myi`、`.myd`、`.frm`文件，`.myd`存储`MyISAM`表的数据，`.myi`存储`MyISAM`表的索引。`.frm`文件存储表的模式、定义。

MySQL中，可以通过查询数据库`INFORMATION_SCHEMA`来获取元数据信息，其中包含表格`TABLES`、`COLUMNS`等。

存储文件格式

包含表和索引等数据。表的行格式决定了其行的物理存储方式，这会影响性能。当单个磁盘页中容纳更多行时，查询可以更快地工作，缓冲命中更高，

- `InnoDB`：将数据存储在`.ibd`文件中，每个表都有一个对应的`.ibd`文件。也可以在一个文件中包含多个表。每个表中的数据分为多个页，排列在B-树中。太长而不能放在B-树页中的数据在额外的页中存储，被称作溢出页。分页存储可以提高缓冲命中率，减少I/O操作。一般地每个页大小为16KB，可以通过配置文件进行修改。
  `InnoDB`支持异步I/O，可以进行预读操作，将数据预先读入内存中，减少I/O操作。
  `InnoDB`支持双重写入缓冲区，将数据写入的重写入缓冲区，然后再写入磁盘中，可以提高安全性。
- `MyISAM`:将数据存储在`.frm`、`.MYD`、`.MYI`三个文件中，其中`.frm`文件存储表的元数据信息，`.MYD`文件存储表的数据，`.MYI`文件存储表的索引。

除了以上几种存储文件格式之外，MySQL还支持其他的存储引擎，比如CSV、Blackhole等，MySQL也支持将数据放入内存中进行存储。

MVCC机制

`InnoDB`使用MVCC机制来实现并发控制，它与HSQLDB类似，都是在每个事务开始时拷贝一份数据库的快照到内存中。在快照中，每行数据都有一个版本号，表示最近的一次修改时间戳，同时MySQL为每个事务分配了一个唯一的事务ID，称为`transaction ID`。

在MySQL的实现中，事务的隔离级别可以设置为`READ UNCOMMITTED`，`READ COMMITTED`，`REPEATABLE READ`，`SERIALIZABLE`。其中，`READ COMMITTED`和`REPEATABLE READ`是使用MVCC实现的，`READ UNCOMMITTED`是不使用MVCC的，而`SERIALIZABLE`则是在MVCC的基础上再增加了锁来保证串行执行。

- `READ COMMITTED`隔离级别下，读取到的数据是已提交的最新版本，如果事务A在事务B提交之前读取了一行数据，而事务B在提交时修改了这行数据，则事务A再次读取这行数据时会得到修改后的结果。
- `REPEATABLE READ`隔离级别下，读取到的数据与第一次读取时的数据一致。在事务中，第一次读取数据时，所有读取的行都会被标记上该事务的`transaction ID`。如果在事务中更新了一行数据，则MySQL会将该行数据插入一份新的版本，并将新版本的版本号设置为该事务的`transaction ID`。在该事务中后续的查询中，MySQL会只返回版本号小于等于该事务的`transaction ID`的数据行。

MySQL使用了两个隐式锁来实现MVCC机制：`共享锁`和`排他锁`。共享锁用于读操作，而排他锁用于写操作。在MVCC中，读操作不会对其他事务的读操作产生影响，而写操作则会对其他事务的读写操作产生影响。

在具体实现中，MySQL使用了`undo log`和`read view`来支持MVCC机制。`undo log`是一种用于回滚操作的日志，保存了每个事务所做的修改。`read view`则是一个快照，保存了当前事务开始时所有已提交的事务的`transaction ID`。当一个事务要读取数据时，MySQL会将当前时间的`transaction ID`与所有已提交的事务的`transaction ID`进行比较，以判断当前数据是否可见。

除了使用MVCC机制外，MySQL还使用了`间隙锁`来解决幻读问题。

压缩机制

通过压缩，可以使得数据库大小降低，减少I/O次数，提高CPU利用率，提高缓存命中率，降低内存成本。

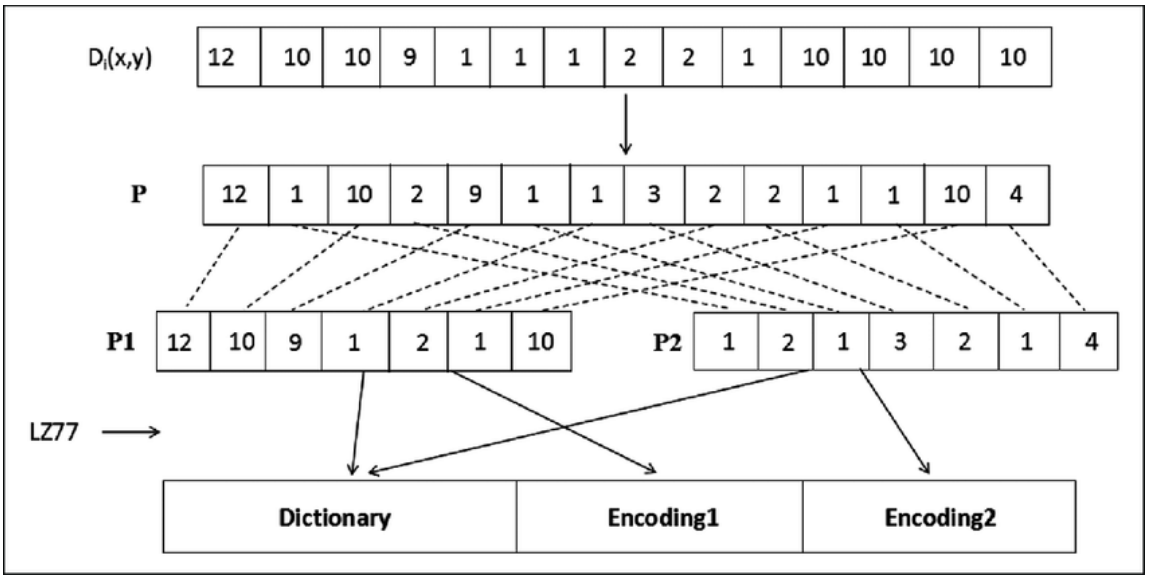
- `InnoDB`：
  - 支持行格式压缩，控制存储在索引页中的列数量，将额外数据存储在溢出页中。
  - 支持表压缩，将所有列和索引进行压缩。将大字段存储在单独的页面中，将索引页和大字段进行压缩。如果表较大，可能会有多个压缩页。对于重要B树节点所在的页面，会尽量减少压缩、解压次数
  - 使用`zlib`库，使用`LZ77`压缩算法，工作原理是使用滑动窗口，将编码器/解码器中已经出现过的匹配数据替换当前数据而实现压缩。这个匹配信息使用称为长度-距离对的一对数据进行编码，它等同于“每个给定长度个字符都等于后面特定距离字符位置上的未压缩数据流。”
    伪代码如下：

```
    while input is not empty do
    match := longest repeated occurrence of input that begins in window

    if match exists then
        d := distance to start of match
        l := length of match
        c := char following match in input
    else
        d := 0
        l := 0
        c := first char of input
    end if

    output (d, l, c)

    discard l + 1 chars from front of window
    s := pop l + 1 chars from front of input
    append s to back of window
repeat
```



- `MyISAM`：支持行格式压缩，但压缩格式的表是只读的。可以使用`myisampack`工具进行表压缩。

总结

HSQLDB和MySQL的元数据文件格式、存储文件格式、MVCC机制、压缩的异同如下：

- 元数据文件格式：HSQLDB使用.script文件存储元数据信息，MySQL使用系统表空间（如INFORMATION_SCHEMA）来存储元数据信息。
- 存储文件格式：MySQL使用多个文件来存储数据库，如数据文件、索引文件、日志文件等。通过将数据分成多页进行存储。HSQLDB使用单个文件来存储数据库。
- MVCC机制：HSQLDB和MySQL都支持MVCC机制，都使用快照机制，但HSQLDB的实现没有锁，而MySQL则使用了锁。
- 压缩方面：HSQLDB使用Gzip压缩算法来压缩数据，而MySQL则支持多种压缩算法，如LZ77等。HSQLDB只支持表压缩，MySQL还支持行格式压缩，压缩输入的数据。相同点是，它们都可以压缩数据以减小存储空间和提高数据访问速度。

ThssDB技术选型

经过调研，我们选择使用`InnoDB`类似的存储技术。
元文件格式：类似于HSQLDB，元数据信息以SQL语句形式保存在文件中，文件名为`数据库名.script`。
数据文件格式：类似于`InnoDB`，每个表都有一个对应的文件。我们使用B+树来存储数据，考虑使用溢出页列表来存储大字段。可能会加入分页机制。
MVCC机制：类似于HSQLDB，不使用锁。使用快照与事务ID来实现`READ COMMITTED`和`REPEATABLE READ`隔离级别。使用`undo log`来实现回滚操作。
压缩：将整表进行压缩，使用GZip算法，考虑使用Java的Apache Commons Compress库。

参考

1. HSQLDB
2. Deflate
3. MySQL
4. LZ77