**Record模块**

**总体功能说明：**

Record模块负责管理记录表中数据的数据文件。主要功能为实现数据文件的创建与删除、记录的插入、删除与查找操作，并对外提供相应的接口。其中记录的查找操作要求能够支持不带条件的查找和带一个条件的查找（包括等值查找、不等值查找和区间查找）。

数据文件由一个或多个数据块组成，块大小应与缓冲区块大小相同。一个块中包含一条至多条记录，为简单起见，只要求支持定长记录的存储，且不要求支持记录的跨块存储。

**具体接口：**

**外部接口：**

def create\_table(tablename)

创建表操作，在目标路径建立起以tablename为名的表。

def delete\_table(tablename)

删除表操作，删除以tablename为名的文件。

def insert(tablename,values)

插入操作，将数据values（列表形式）转化为字符串，把需要插入的值直接合并，并在前面加一位有效位用于之后的校验，最后将数据传输到buffer，通过buffer中的函数传递到以tablename为名的表中。

def select\_record(tablename, columnname, clauses, where, length)

select输出指定子句操作，首先判断where是否为空，如果是则直接调用buffer模块从表中映射需要select的列数据并输出。若否则根据where提供的数据区域调用buffer 模块，找到需要select的列数据并输出。

def delete\_record(tablename, clauses,length)

删除数据操作与select操作在搜寻所需数据时的操作相似，在找到需要删除的数据后会标记其有效位由1置为0，并将其列进列表中，返回值为要删除的所有数据。

def create\_index(tablename, id, type,length)

创建索引操作，输入为表名，索引名称，类型和数据长度，返回值为定位到的所有数据的位置。

def truncate(tablename, where)

作为间接接口调用Buffer中的truncate函数。

**内部接口：**

def decrypt(code)

用于分割存入buffer中的code，返回结果为有效位和对应的数据。

**心得体会：**

在完成这次的大作业的过程中，我主要负责编写record这一模块。这是我第一次用python完成比较大型的项目，在编写这一模块的过程中，我对python语言的使用取得了较为明显的进步。因为record模块与buffer模块的联系十分紧密，因此我们在实现模块时对于接口的对接，具体函数的实现方法等都有较为详细的讨论，在模块对接，沟通交流等方面也有了一些进步。总的来说，这次的大作业不仅让我更深入的了解了数据库的相关知识，也锻炼了我的代码能力和项目模块对接能力，使我受益匪浅。