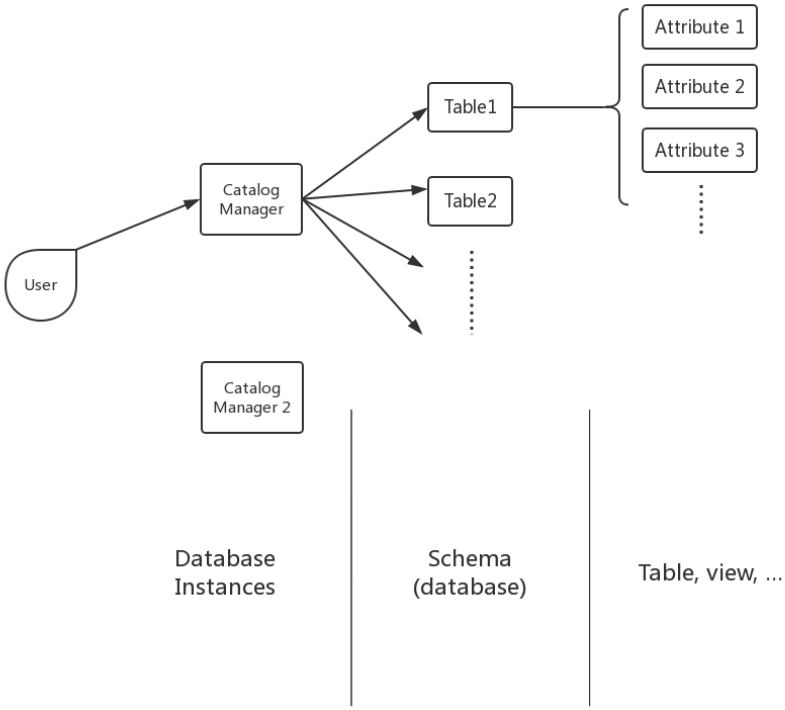
Catalog Manager 模块设计说明

Lyu Keyao

1. Catalog Manager 功能分析

一个数据库系统包含多个 Catalog，记录了多种数据库对象的信息，如表、属性、索引等。Catalog Manager 通过 API 接口的功能，将数据库的字典信息传递给 Record Manager 模块、Index Manager 模块和 Buffer Manager 模块。



2. 设计思路

Catalog Manager 管理数据库的所有字典信息，数据库中所有表的定义信息，包括：

- 表的名称，属性数，主键
- 属性类型，属性名，是否唯一，是否有索引

- 索引的数量，索引名字，索引对应的属性

以上字典信息需要通过 Catalog Manager 写入块中，以供数据库在后续的用户操作中对已经存在的信息进行管理和检查。Catalog 在块中的写入设计如下，使得 Catalog Manager 能够跟踪表的定义信息——Catalog 记录包含有关其行存储在该块中的表的元数据，读取其字符即可获得表的定义信息。

表的定义信息在块中的存储如下：

0000	Name	Attribute_num	Type_1 (-1/0/x)	name_1	Unique_1	...	...	#
起始字符	表名	属性的数量	第一个属性			第二个属性	...	换行

例如：

Book	**name**	**type**	**Unique**	**Primary key**	**Index**
Attribute1	bid	char	true	true	A
Attribute2	price	float	false	false	-
Attribute3	stock	int	false	false	-

在 catalog file 中储存为

```
0060 book 03 001 bid 1 000 price 0 -001 stock 0 00 ;01 01 A
#
```

3. 主要函数

3.1 创建表 creatTable

功能：根据 Interpreter 正则化语句后得到的信息，提取表名、属性信息、主键序号、索引信息，建立有对应格式的表。如果指定表名的 Table 已存在，抛出异常 table_exit()给 Interpreter 等待下一步处理。

```
void creatTable(string table_name, Attribute attribute, int primary, Index index);
```

3.2 删除表 dropTable

```
void dropTable(string table_name);
```

功能：根据 Interpreter 提取需要删除的表名，在文件中找到表所在的块，删除

异常：如果指定表名的 Table 不存在，抛出 table_not_exit()。

3.3 判断表是否已经存在 hasTable

```
bool hasTable(string table_name);
```

功能：在 interpreter 提取出 SQL 语句的具体操作后，对语句指明的表是否存在进行检验；或 Catalog Manager 的其他功能中需要首先确定是否存在某个表，则给出表名，在数据字典中遍历块，检查表名是否对应。若已经存在，返回 true；否则返回 false。比如：创建/删除表、插入/删除记录、创建/删除索引等命令前调用，检查是否存在相关的表以及是否要抛出 table_not_exit()。

3.4 判断指定表的属性是否已存在 has Attribute

```
bool hasAttribute(string table_name, string attr_name);
```

功能：在执行和属性相关的语句前被调用，检查属性在某个表中是否存在。若已经存在，返回 true；否则返回 false。如果不存在用户指明的表，抛出异常 table_not_exit()。

3.5 获取表的属性信息 getAttribute

```
Attribute getAttribute(string table_name);
```

功能：调用该函数获得指定表的属性信息。如果不存在用户指明的表，抛出异常 table_not_exit()。

3.6 创建索引语句 createIndex

```
void createIndex(string table_name, string arrt_name, string index_name);
```

功能：对指定属性建立索引，并且需要给索引命名，通过重新建表更新索引的信息。

异常：建立索引前有多个检查语句是否能执行的判断：

1. 如果不存在该名字的表，抛出 table_not_exit()
2. 如果不存在指定的属性，抛出 throw attribute_not_exit()
3. 如果索引数目超出上限，抛出 throw index_full()
4. 如果索引名重复，抛出 index_exist()
5. 如果指定的属性已有索引，抛出 index_exist()

3.7 找到索引对应的属性 IndextoAttr

```
string IndextoAttr(string table_name, string index_name);
```

功能：通过索引和属性之间的对应关系，找到索引对应的属性。

异常：调用 getAttribute 获得属性名对比字符串。如果不存在该名字的表，抛出 table_not_exit()；如果不存在该名字的索引，抛出 index_not_exist()。

3.8 删除索引 dropIndex

```
void dropIndex(string table_name, string index_name);
```

功能：将指定的索引从数据库中删除。实现方式是把索引的最后一个调到被删除的索引位置，然后删除最后一个索引记录。

异常：如果不存在该名字的表，抛出 table_not_exit()；如果不存在该名字的索引，抛出 index_not_exist()。

3.9 打印表

```
void showTable(string table_name);
```

功能：输出属性、主键、索引的信息

其他功能：

得到该行表的名字 getTableName

```
string getTableName(string buffer, int start, int &rear);
```

返回该表的位置

```
int getTablePlace(string name, int &suitable_block);
```

功能：参数中的引用传出该表的所在的块的位置（遍历所有的块，buffer manager 指出页的头地址，读取字符确定表 start 的位置），返回在块中的位置；如果未找到，则返回-1。

返回该表的 index

```
Index getIndex(string table_name);
```

功能：从 buffer manager 得到表的位置和对应的块，找到索引开始的位置，从记录中读取索引的信息。

返回文件大小，统计块数

```
int getBlockNum (string table_name);
```

4. 小结

Catalog Manager 负责管理数据库的字典信息，将表的定义写入文件，在后续操作中读取、更新定义信息，并传递给其他模块。通过这次实验，对数据库的实现和 SQL 语句的执行更加了解了，尤其是 Catalog 的实现和原理，如何管理数据库的字典信息

和其他功能，如何与其他模块共同工作。