Design Document

1. 项目文件结构：

```
.
├── base.cpp
├── base.h
├── Database.cpp
├── Database.h
├── Lexer.cpp
├── Lexer.h
├── main.cpp
├── Makefile
├── outputGrammar.cpp
├── Parser.cpp
├── Parser.h
├── testArithmetic.cpp
├── testBoolean.cpp
└── testGrammar.cpp
```

base.cpp：包含一些基本的函数和数据结构，比如：计算算术表达式，计算逻辑表达式

Database.cpp：数据库的操作类，用单例模式实现，主要功能是进行数据库中表的创建，数据插入，数据查询，数据删除的操作

Lexer.cpp：词法分析器，进行词法分析

Parser.cpp：语法分析器，对词法分析器生成的 Token 列表进行处理

main.cpp：主函数

outputGrammar.cpp：输出语法分析器生成的 first 表，follow 表和预测分析表

testArithmetic.cpp：进行算术表达式测试的文件

testBoolean.cpp：进行布尔表达式测试的文件

testGrammar.cpp：进行语法测试的文件

Makefile：项目构建文件

2. 程序思路：

首先整个程序的结构主要分成三个部分：一个是词法分析器，一个是语法分析器，最后一个是数据库管理系统，词法分析器主要是把输入的字符串分割成一个个符号表中的 token（词法单元），其中把相应的行列信息包含进去，以提供错误时的信息；然后把生成的 token 列表传给语法分析器，语法分析器通过已经计算生成的预测分析表进行语法分析，一旦有语法错误，就输出出现错误的 Token，并把 Token 列表的指针指向下一个分号的下一位，如果正确，就把该正确的一条语句的 Token 传给数据库管理进行相应的数据库操作，然后返回一条操作信息，该信息包含操作的逻辑正确性和数据库查询的文字信息。语法分析器获得该信息后就把它显示到终端。

3. 主要数据结构和函数：（大写开头的皆为类，小写开头的均为函数）

base.cpp：

Table：数据库中的表

rows (vector<map<string,int>>)：数据库表中的行，每行对应一个键值对
primaryKey (map<string,bool>)：数据库表中的主键
defaultValues (map<string,int>)：数据库表中的默认值
fields (vector<string>)：数据库表中含有的所有字段

Symbol：生成预测分析表的符号
isTerminalSymbol (bool)：是否是终结符
canBeEmpty (bool)：当为非终结符时，是否存在用空推导出的产生式
isStartSymbol (bool)：是否是开始符号
expression (vector<vector<string>>)：当为非终结符时，所具有的所有产生式
firstSet (set<string>)：该符号的 first 表
followSet (set<string>)：该符号的 follow 表

Node：词法单元
token (string)：对应的 token
name (string)：该 token 的值
row (int)：在输入中的第几行（从 1 开始）
col (int)：在输入中的第几列（从 1 开始）

convertStringToInteger：string 转 int
convertIntegerToString：int 转 string
calculateArithmeticExpression：由一个正确的 token 列表计算算术表达式的值
calculateBooleanExpression：由一个正确的 token 列表和键值对计算逻辑表达式的值

Lexer.cpp：
work：主要的对外函数，从一个 string 生成 token 列表
print：输出词法分析器分析得到的 token 列表，例如：

```
create table student(sid int, age int default = 20);
(create, CREATE, row 1, col 1)
(table, TABLE, row 1, col 8)
(student, ID, row 1, col 14)
((, LP, row 1, col 21)
(sid, ID, row 1, col 22)
(int, INT, row 1, col 26)
(,, COMMA, row 1, col 29)
(age, ID, row 1, col 31)
(int, INT, row 1, col 35)
(default, DEFAULT, row 1, col 39)
(=, ASSIGN, row 1, col 47)
(20, NUM, row 1, col 49)
(), RP, row 1, col 51)
(;, SEM, row 1, col 52)
```

…

Parser.cpp：
work：由一个 token 列表处理相应的语句

initializeFirstSet：初始化 first 表
initializeFollowSet：初始化 follow 表
initializePredictSet：初始化预测分析表
errorRecovery：出现语法错误时进行错误恢复
doCreateStatement：语法正确的句子进行对应的 create 操作
doDeleteStatement：语法正确的句子进行对应的 delete 操作
doSelectStatement：语法正确的句子进行对应的 select 操作
doInsertStatement：语法正确的句子进行对应的 insert 操作
...

Database.cpp：
create：进行创建表的操作，返回操作结果
insert：进行插入值的操作，返回操作结果
query：进行查询的操作，输出查询结果并返回操作结果
deleteData：进行删除操作，返回操作结果
readFromFile（已弃用）：从文件中读入数据
writeToFile（已弃用）：把数据库的表写入文件
...

4. 运行测试指令：
make：编译链接并生成 ssql 可执行文件
make run：运行 ssql 可执行文件
make ari：编译链接生成 test 文件并运行 test 进行算术表达式的测试
make bool：编译链接生成 test 文件并运行 test 进行逻辑表达式的测试
make output：输出 first 表，follow 表和预测分析表
make grammar：编译链接生成 test 文件并运行 test 进行语法测试

5. 文法生成的结构：
消除左递归后的文法：

```
**********************************************************
***                       Grammar                        ***
**********************************************************
ssql_stmt -> create_stmt | insert_stmt | delete_stmt | query_stmt
create_stmt -> CREATE TABLE ID LP decl_list RP SEM
insert_stmt -> INSERT INTO ID LP column_list RP VALUES LP value_list RP SEM
delete_stmt -> DELETE FROM ID where_clause SEM
query_stmt -> SELECT select_list FROM ID where_clause SEM
decl_list -> decl decl_list_extra
decl_list_extra -> COMMA decl decl_list_extra | EPSILON
decl -> ID INT default_spec | PRIMARY KEY LP column_list RP
default_spec -> DEFAULT ASSIGN simple_expr | EPSILON
column_list -> ID column_list_extra
column_list_extra -> COMMA ID column_list_extra | EPSILON
value_list -> simple_expr value_list_extra
```

value_list_extra -> COMMA simple_expr value_list_extra | EPSILON
where_clause -> WHERE disjunct | EPSILON
boolean -> LP disjunct RP | NOT boolean | comp
rop -> NOTEQUAL | EQUAL | GT | LT | GTE | LTE
select_list -> column_list | MUL
simple_expr -> simple_term simple_expr_extra
simple_expr_extra -> ADD simple_term simple_expr_extra | SUB simple_term simple_expr_extra | EPSILON
simple_term -> simple_unary simple_term_extra
simple_term_extra -> MUL simple_unary simple_term_extra | DIV simple_unary simple_term_extra | EPSILON
simple_unary -> LP simple_expr RP | SUB simple_unary | ADD simple_unary | NUM
disjunct -> conjunct disjunct_extra
disjunct_extra -> OR conjunct disjunct_extra | EPSILON
conjunct -> boolean conjunct_extra
conjunct_extra -> AND boolean conjunct_extra | EPSILON
comp -> expr rop expr
expr -> term expr_extra
expr_extra -> ADD term expr_extra | SUB term expr_extra | EPSILON
term -> unary term_extra
term_extra -> MUL unary term_extra | DIV unary term_extra | EPSILON
unary -> SUB unary | ADD unary | ID | NUM
****

First 表：
```
**********************************************************
***                    First Table                    ***
**********************************************************
ssql_stmt {CREATE, DELETE, INSERT, SELECT}
create_stmt {CREATE}
insert_stmt {INSERT}
delete_stmt {DELETE}
query_stmt {SELECT}
decl_list {ID, PRIMARY}
decl_list_extra {COMMA, EPSILON}
decl {ID, PRIMARY}
default_spec {DEFAULT, EPSILON}
column_list {ID}
column_list_extra {COMMA, EPSILON}
value_list {ADD, LP, NUM, SUB}
value_list_extra {COMMA, EPSILON}
where_clause {EPSILON, WHERE}
boolean {ADD, ID, LP, NOT, NUM, SUB}
```

```
rop {EQUAL, GT, GTE, LT, LTE, NOTEQUAL}
select_list {ID, MUL}
simple_expr {ADD, LP, NUM, SUB}
simple_expr_extra {ADD, EPSILON, SUB}
simple_term {ADD, LP, NUM, SUB}
simple_term_extra {DIV, EPSILON, MUL}
simple_unary {ADD, LP, NUM, SUB}
disjunct {ADD, ID, LP, NOT, NUM, SUB}
disjunct_extra {EPSILON, OR}
conjunct {ADD, ID, LP, NOT, NUM, SUB}
conjunct_extra {AND, EPSILON}
comp {ADD, ID, NUM, SUB}
expr {ADD, ID, NUM, SUB}
expr_extra {ADD, EPSILON, SUB}
term {ADD, ID, NUM, SUB}
term_extra {DIV, EPSILON, MUL}
unary {ADD, ID, NUM, SUB}
CREATE {CREATE}
INSERT {INSERT}
TABLE {TABLE}
DELETE {DELETE}
SELECT {SELECT}
ID {ID}
INT {INT}
PRIMARY {PRIMARY}
KEY {KEY}
ASSIGN {ASSIGN}
DEFAULT {DEFAULT}
NUM {NUM}
EPSILON {EPSILON}
INTO {INTO}
VALUES {VALUES}
WHERE {WHERE}
AND {AND}
NOTEQUAL {NOTEQUAL}
EQUAL {EQUAL}
LT {LT}
GT {GT}
LTE {LTE}
GTE {GTE}
FROM {FROM}
LP {LP}
RP {RP}
ADD {ADD}
```

```
SUB {SUB}
MUL {MUL}
DIV {DIV}
OR {OR}
NOT {NOT}
SEM {SEM}
COMMA {COMMA}
**********************************************************
```

Follow 表：

```
**********************************************************
***                        Follow Table                        ***
**********************************************************
ssql_stmt {$}
create_stmt {$}
insert_stmt {$}
delete_stmt {$}
query_stmt {$}
decl_list {RP}
decl_list_extra {RP}
decl {COMMA, RP}
default_spec {COMMA, RP}
column_list {FROM, RP}
column_list_extra {FROM, RP}
value_list {RP}
value_list_extra {RP}
where_clause {SEM}
boolean {AND, OR, RP, SEM}
rop {ADD, ID, NUM, SUB}
select_list {FROM}
simple_expr {COMMA, RP}
simple_expr_extra {COMMA, RP}
simple_term {ADD, COMMA, RP, SUB}
simple_term_extra {ADD, COMMA, RP, SUB}
simple_unary {ADD, COMMA, DIV, MUL, RP, SUB}
disjunct {RP, SEM}
disjunct_extra {RP, SEM}
conjunct {OR, RP, SEM}
conjunct_extra {OR, RP, SEM}
comp {AND, OR, RP, SEM}
expr {AND, EQUAL, GT, GTE, LT, LTE, NOTEQUAL, OR, RP, SEM}
expr_extra {AND, EQUAL, GT, GTE, LT, LTE, NOTEQUAL, OR, RP, SEM}
term {ADD, AND, EQUAL, GT, GTE, LT, LTE, NOTEQUAL, OR, RP, SEM, SUB}
term_extra {ADD, AND, EQUAL, GT, GTE, LT, LTE, NOTEQUAL, OR, RP, SEM,
```

SUB}

unary {ADD, AND, DIV, EQUAL, GT, GTE, LT, LTE, MUL, NOTEQUAL, OR, RP, SEM, SUB}

```
**********************************************************
```

预测分析表：（箭头坐标为一个非终结符和终结符，右边为相应的产生式）

```
**********************************************************
***                    Predict Table                    ***
**********************************************************
ssql_stmt, CREATE ---------> create_stmt
ssql_stmt, DELETE ---------> delete_stmt
ssql_stmt, INSERT ---------> insert_stmt
ssql_stmt, SELECT ---------> query_stmt
create_stmt, CREATE ---------> CREATE TABLE ID LP decl_list RP SEM
insert_stmt, INSERT ---------> INSERT INTO ID LP column_list RP VALUES LP value_list RP SEM
delete_stmt, DELETE ---------> DELETE FROM ID where_clause SEM
query_stmt, SELECT ---------> SELECT select_list FROM ID where_clause SEM
decl_list, ID ---------> decl decl_list_extra
decl_list, PRIMARY ---------> decl decl_list_extra
decl_list_extra, COMMA ---------> COMMA decl decl_list_extra
decl_list_extra, RP ---------> EPSILON
decl, ID ---------> ID INT default_spec
decl, PRIMARY ---------> PRIMARY KEY LP column_list RP
default_spec, COMMA ---------> EPSILON
default_spec, DEFAULT ---------> DEFAULT ASSIGN simple_expr
default_spec, RP ---------> EPSILON
column_list, ID ---------> ID column_list_extra
column_list_extra, COMMA ---------> COMMA ID column_list_extra
column_list_extra, FROM ---------> EPSILON
column_list_extra, RP ---------> EPSILON
value_list, ADD ---------> simple_expr value_list_extra
value_list, LP ---------> simple_expr value_list_extra
value_list, NUM ---------> simple_expr value_list_extra
value_list, SUB ---------> simple_expr value_list_extra
value_list_extra, COMMA ---------> COMMA simple_expr value_list_extra
value_list_extra, RP ---------> EPSILON
where_clause, SEM ---------> EPSILON
where_clause, WHERE ---------> WHERE disjunct
boolean, ADD ---------> comp
boolean, ID ---------> comp
boolean, LP ---------> LP disjunct RP
boolean, NOT ---------> NOT boolean
boolean, NUM ---------> comp
```

```
boolean, SUB --------> comp
rop, EQUAL --------> EQUAL
rop, GT --------> GT
rop, GTE --------> GTE
rop, LT --------> LT
rop, LTE --------> LTE
rop, NOTEQUAL --------> NOTEQUAL
select_list, ID --------> column_list
select_list, MUL --------> MUL
simple_expr, ADD --------> simple_term simple_expr_extra
simple_expr, LP --------> simple_term simple_expr_extra
simple_expr, NUM --------> simple_term simple_expr_extra
simple_expr, SUB --------> simple_term simple_expr_extra
simple_expr_extra, ADD --------> ADD simple_term simple_expr_extra
simple_expr_extra, COMMA --------> EPSILON
simple_expr_extra, RP --------> EPSILON
simple_expr_extra, SUB --------> SUB simple_term simple_expr_extra
simple_term, ADD --------> simple_unary simple_term_extra
simple_term, LP --------> simple_unary simple_term_extra
simple_term, NUM --------> simple_unary simple_term_extra
simple_term, SUB --------> simple_unary simple_term_extra
simple_term_extra, ADD --------> EPSILON
simple_term_extra, COMMA --------> EPSILON
simple_term_extra, DIV --------> DIV simple_unary simple_term_extra
simple_term_extra, MUL --------> MUL simple_unary simple_term_extra
simple_term_extra, RP --------> EPSILON
simple_term_extra, SUB --------> EPSILON
simple_unary, ADD --------> ADD simple_unary
simple_unary, LP --------> LP simple_expr RP
simple_unary, NUM --------> NUM
simple_unary, SUB --------> SUB simple_unary
disjunct, ADD --------> conjunct disjunct_extra
disjunct, ID --------> conjunct disjunct_extra
disjunct, LP --------> conjunct disjunct_extra
disjunct, NOT --------> conjunct disjunct_extra
disjunct, NUM --------> conjunct disjunct_extra
disjunct, SUB --------> conjunct disjunct_extra
disjunct_extra, OR --------> OR conjunct disjunct_extra
disjunct_extra, RP --------> EPSILON
disjunct_extra, SEM --------> EPSILON
conjunct, ADD --------> boolean conjunct_extra
conjunct, ID --------> boolean conjunct_extra
conjunct, LP --------> boolean conjunct_extra
conjunct, NOT --------> boolean conjunct_extra
```

```
conjunct, NUM ---------> boolean conjunct_extra
conjunct, SUB ---------> boolean conjunct_extra
conjunct_extra, AND ---------> AND boolean conjunct_extra
conjunct_extra, OR ---------> EPSILON
conjunct_extra, RP ---------> EPSILON
conjunct_extra, SEM ---------> EPSILON
comp, ADD ---------> expr rop expr
comp, ID ---------> expr rop expr
comp, NUM ---------> expr rop expr
comp, SUB ---------> expr rop expr
expr, ADD ---------> term expr_extra
expr, ID ---------> term expr_extra
expr, NUM ---------> term expr_extra
expr, SUB ---------> term expr_extra
expr_extra, ADD ---------> ADD term expr_extra
expr_extra, AND ---------> EPSILON
expr_extra, EQUAL ---------> EPSILON
expr_extra, GT ---------> EPSILON
expr_extra, GTE ---------> EPSILON
expr_extra, LT ---------> EPSILON
expr_extra, LTE ---------> EPSILON
expr_extra, NOTEQUAL ---------> EPSILON
expr_extra, OR ---------> EPSILON
expr_extra, RP ---------> EPSILON
expr_extra, SEM ---------> EPSILON
expr_extra, SUB ---------> SUB term expr_extra
term, ADD ---------> unary term_extra
term, ID ---------> unary term_extra
term, NUM ---------> unary term_extra
term, SUB ---------> unary term_extra
term_extra, ADD ---------> EPSILON
term_extra, AND ---------> EPSILON
term_extra, DIV ---------> DIV unary term_extra
term_extra, EQUAL ---------> EPSILON
term_extra, GT ---------> EPSILON
term_extra, GTE ---------> EPSILON
term_extra, LT ---------> EPSILON
term_extra, LTE ---------> EPSILON
term_extra, MUL ---------> MUL unary term_extra
term_extra, NOTEQUAL ---------> EPSILON
term_extra, OR ---------> EPSILON
term_extra, RP ---------> EPSILON
term_extra, SEM ---------> EPSILON
term_extra, SUB ---------> EPSILON
```

```
unary, ADD ---------> ADD unary
unary, ID ---------> ID
unary, NUM ---------> NUM
unary, SUB ---------> SUB unary
**********************************************************
```

6. 小组分工

邓宇恒：数据库操作类，语法分析器，设计文档

龚科：词法分析器，select 语句的输出样式，测试

付洋：测试

陈元仿：测试，测试文档

7. 组员信息

邓宇恒 12330071

龚科 12330088

付洋 12330085

陈元仿 12330052