Проект содержит реализацию модельного SQL-интерпретатора.

Вариант задания:

1.Один клиент. Клиент и сервер на одной ЭВМ.

2.Клиент получает от пользователя запрос на SQL, анализирует его и в случае ошибки сообщает об этом пользователю, иначе передает серверу запрос в некотором внутреннем представлении. Сервер обращается к БД, определяет ответ на запрос и передает его клиенту. Клиент выдает пользователю ответ сервера.

1) В файлах Client.cpp Client.hpp Server.cpp Server.hpp содержится реализация архитектуры «Клиент — Сервер»:

Команды пользователя -> Клиент: обработка команд, формирование запроса, если успешно, то установление связи с сервером, иначе печатается сообщение об ошибке -> Сервер: выполнение полученных команд (работа с БД), передача клиенту ответа, сообщения об успешном выполнении или сообщения об ошибке -> Клиент: получение результата -> Пользователь

2) В файлах RDBMSTable.cpp RDBMSTable.hpp table.c table.h содержится реализация базы данных.

table.c table.h - реализация БД, приложенная к заданию.

RDBMSTable.cpp RDBMSTable.hpp - некая "оболочка" для БД, позволяющая работать с классом string, а также генерирующая исключения в случае ошибкок в работе БД.

3) SQLInterface.cpp SQLInterface.hpp содержит функцию DBMS(), которая запускает процессы клиента и сервера и таким образом позволяет пользователю работать с БД.

4) Для разбора строки используются классы Lexeme, Scanner, Parser.

Lexeme содержит информацию о лексеме (тип и строковое представление).

Scanner разбивает строку символов на лексемы. Метод gl() позволяет вызвать Scanner и получить очередную лексему.

Parser анализирует строку лексем методом рекурсивного спуска по грамматике, представленной в gram.txt.

5) Класс Query:

Запрос формируется в виде объекта класса Query:

- cl_type - информация о типе запроса SELECT / INSERT / UPDATE / DELETE / CREATE / DROP или о завершении работы клиента (Q).
- table_name - имя таблицы, к которой относится запрос
- cond_type - тип условия в WHERE-clause like / in / expression / all
- neg - информация о негативном условии для NOT LIKE и NOT IN

Запрос содержит:

- data - вектор данных
- constants - вектор констант (для WHERE [NOT] IN и WHERE [NOT] LIKE)
- expr_poliz - вектор элементов полиза, являющийся полизом выражения, значение которого будет присвоено изменяемому полю (для UPDATE)
- cond_poliz - вектор элементов полиза, являющийся полизом выражения, стоящего после WHERE-clause.

Структура для запроса каждого типа:

SELECT

data	Информация о том, что должны быть выбраны все поля или Имена полей, которые должны быть выбраны
expr_poliz	пусто
cond_poliz	Логическое выражение, стоящее после WHERE или Информация о том, что все строки удовлетворяют условию или Имя поля, проверяемого в случае WHERE [NOT] LIKE или Выражение, проверяемое в случае WHERE [NOT] IN
constants	Строка, с которой происходит сравнение в случае WHERE [NOT] LIKE или Список констант, с которыми происходит сравнение в случае WHERE [NOT] IN

INSERT

data	Значения полей строки, которая должна быть добавлена в таблицу
expr_poliz	пусто
cond_poliz	пусто
constants	пусто

UPDATE

data	Имя изменяемого поля
expr_poliz	Выражение, значение которого присваивается полю типа Long Или Новое значение поля типа Text
cond_poliz	см. SELECT
constants	см. SELECT

DELETE

data	пусто

expr_poliz	пусто
cond_poliz	см. SELECT
constants	см. SELECT

CREATE

data	Чередуются имена полей и их размеры
expr_poliz	пусто
cond_poliz	пусто
constants	пусто

DROP

data	пусто
expr_poliz	пусто
cond_poliz	пусто
constants	пусто

6) Файл Query.cpp содержит методы, используемые анализатором для построения запроса. Используется клиентом.

7) Файл Query_execute.cpp содержит методы, необходимые для выполнения запроса. Здесь происходит анализ и выявление запросов, содержащих синтаксические ошибки, не выявленные на этапе синтаксического анализа, и семантический анализ. Используется сервером.

8) Файл Query_messages.cpp содержит текст сообщений об ошибках. Используется и клиентом, и сервером.

9) Файлы Query_calc_like.cpp Query_calc_like.hpp содержат реализацию проверки соответствия строки правилу при альтернативе [NOT] LIKE в WHERE-clause.

Алгоритм проверки:

Пусть проверяемая строка str.

1) Строка с правилом разбивается на подправила, не содержащие %.

Например: %ab%[^mno]_%aaa% будет разбита на правила ab, [^mno]_, aaa

2) Если строка с правилом начинается не с %, то проверяется соответствие первых n символов str первому правилу, где n - длина первого правила. Текущим назначается второе правило.

Если строка с правилом начинается с %, то текущим назначается первое правило.

3) Для текущего правила ищется первая подстрока str, удовлетворяющая ему.

Если такая строка не находится, то считается, что str не удовлетворяет правилу и алгоритм останавливается.

Иначе от str “отрезается” найденная подстрока и всё, что встретилось до неё.

Текущим правилом становится следующее и алгоритм повторяется для обрезанной строки.

4) Если после последнего правила встретился % и алгоритм не остановился, то str удовлетворяет правилу.

Если после последнего правила не встретился %, а в строке str ещё остались символы, то str не удовлетворяет правилу.

Проверка соответствия строки substr правилу без %:

- Если текущее правило ‘_’, то происходит переход к следующему символу substr, следующий символ строки правила определяет следующее правило.

- Если текущее правило ‘[’, то считываются символы строки правила до ‘]’ и формируется строка - список возможных символов.

  Если после ‘[’ встречается ‘^’, то аналогично формируется список невозможных символов. ‘^’ не после ‘[’ считается обычным символом.

  Если список возможных символов состоит из трех символов, второй из которых ‘-’, то считается, что список задан диапазоном от первого символа до последнего. Иначе ‘-’ считается обычным символом.

  Если ‘]’ так и не встретилось, то пользователю выдаётся сообщение о синтаксической ошибке.

Проверяется [не]принадлежность текущего символа substr и происходит переход к следующему символу. Следующий символ строки правила определяет следующее правило.

- Если текущее правило - любой другой символ, то происходит сравнение текущего символа substr с символом-правилом, происходит переход к следующему символу substr и к следующему правилу.