Вахрушев Данил P33312

Низкоуровневое Программирование. Лабораторная работа 3

Installation and start

Requirements:

- cmake (3.16)
- make
- g++

```
cmake .
```

- Server

  ```
  make lab3-server
  ./lab3-server <port>
  ```

- Client:

  ```
  make lab3-client
  ./lab3-client <host> <port> <database>
  ```

Цель задания

На базе данного транспортного формата описать схему протокола обмена информацией и воспользоваться существующей библиотекой по выбору для реализации модуля, обеспечивающего его функционирование. Протокол должен включать представление информации о командах создания, выборки, модификации и удаления данных в соответствии с данной формой, и результатах их выполнения.

Задачи

1. Изучить выбранную библиотеку
2. На основе существующей библиотеки реализовать модуль, обеспечивающий взаимодействие
3. Реализовать серверную часть в виде консольного приложения
4. Реализовать клиентскую часть в виде консольного приложения

Аспекты реализации

Структура проекта:

- `client` - клиенсткое приложение
- `server` - серверное приложение
- `parser_lib` - получено в результате выполнения ЛР 2
- `dbms` - получено в результате выполнения ЛР 1
- `*.xsd` - описанные схемы сообщений
- `common` - сгенерированные по схемам файлы, общие для клиента и сервера

Для линковки модулей друг с другом используется CMake.

Описание работы и реализации

- Xml-библиотека поддерживающая кодогенерацию на основе схемы - `XSD Code Synthesis`
- Сетевое взаимодействие реализовано посредством сокетов API ОС;
- Результаты запроса отсылаются клиенту пакетами, в каждом пакете максимум 10 записей из запроса.
- Клиент отправляет серверу сообщения, определенного формата, есть 2 типа сообщений
  - `connect` - запрос на подключение к определенной базе
  - `query` - запрос на исполнение запроса
- Ответ от сервера содержит в себе:
  - статус запроса
  - сообщение от ошибке
  - флажок, последний ли это ответ от сервера
  - возможный результат запроса:
    - Заголовок
    - Тело ответа

Порадок обработки запроса следующий:

1. Клиент отправляет запрос и ожидает ответа
2. Сервер начинает исполнять запрос и отправляет клиенту результат:
   - в случае с `select`, первое сообщение - это header
   - при остальных запросах в ответе есть флажок finished

    - сообщение об ошибке, если что-то пошло не так
3. Если запрос - select, то клиент запрашивает следующий пакет записей, и так пока сервер не отправит finished
4. В конце всех запросов от сервера должен приходить ответ finished.

Результаты

```
 > FOR X IN STUDS RETURN X;
Schema table doesn't exists
> FOR X IN GROUPS RETURN X;
Schema table doesn't exists

> CREATE TABLE STUDS { "stud_name": string, "stud_group_id": int };
> CREATE TABLE GROUPS { "group_id": int, "group_name": string };

> FOR X IN STUDS RETURN X;
stud_group_id       stud_name
> FOR X IN GROUPS RETURN X;
group_name          group_id

> INSERT { "stud_name": "Vasya", "stud_group_id": 1 } INTO STUDS;
> INSERT { "stud_name": "Misha", "stud_group_id": 1 } INTO STUDS;
> INSERT { "stud_name": "Stepa", "stud_group_id": 2 } INTO STUDS;
> INSERT { "group_id": 1, "group_name": "P314" } INTO GROUPS;
> INSERT { "group_id": 2, "group_name": "N521" } INTO GROUPS;

> FOR X IN STUDS RETURN X;
stud_group_id       stud_name
1                   Vasya
1                   Misha
2                   Stepa

> FOR X IN GROUPS RETURN X;
group_name          group_id
P314                1
N521                2

> FOR X IN STUDS
        FOR Y IN GROUPS
                RETURN ALL;
stud_group_id       stud_name           group_name          group_id
1                   Vasya               P314                1
1                   Vasya               N521                2
1                   Misha               P314                1
1                   Misha               N521                2
2                   Stepa               P314                1
2                   Stepa               N521                2

> FOR X IN STUDS
        FOR Y IN GROUPS
                FILTER group_id == stud_group_id
                RETURN ALL;
stud_group_id       stud_name           group_name          group_id
1                   Vasya               P314                1
1                   Misha               P314                1
2                   Stepa               N521                2

> FOR X IN STUDS
        RETURN X;
stud_group_id       stud_name
1                   Vasya
1                   Misha
2                   Stepa
> FOR X IN STUDS
        FILTER stud_name == "Misha"
        UPDATE X WITH { "stud_group_id": 2 } IN STUDS;
```

```
> FOR X IN STUDS RETURN X;
stud_group_id       stud_name
1                   Vasya
2                   Misha
2                   Stepa

> FOR X IN STUDS
        FILTER stud_group_id == 1
        REMOVE X IN STUDS;
> FOR X IN STUDS RETURN X;
stud_group_id       stud_name
2                   Misha
2                   Stepa

> DROP TABLE GROUPS;
> FOR X IN GROUPS RETURN X;
Schema table doesn't exists
```

XML-схема запроса от клиента приведена здесь

XML-схема ответа от сервера

Вывод

В процессе выполнения лабораторной работы я познакомился с понятием xml-схемы и научился применять ее для генерации парсеров. Также я поближе познакомился с сетевым взаимодействием