УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Низкоуровневое программирование

Лабораторная работа №3

Преподаватель: Кореньков Юрий Дмитриевич
Выполнил: Ле Чонг Дат
Группа: P33302

2023 г.

1 Introduction

This report discusses the implementation and testing of a simple database server. The server communicates with clients using the Protocol Buffers (protobuf) format and provides basic functionalities of a database system.

2 Implementation

2.1 Protocol Buffers Messages

The protobuf messages used for client-server communication are defined as follows:

```
syntax = "proto3";

package database;

message Request {
    string query = 1;
}

message Response {
    string message = 1;
    bool success = 2;
}
```

Листинг 1: Protobuf Messages

2.2 Seed Script

The seed script used for testing the database server is shown below:

```
CREATE TABLE students (id INT, name VARCHAR(255), age INT);

INSERT INTO students(id, name, age) VALUES (1, alex, 15);
INSERT INTO students(id, name, age) VALUES (2, misa, 16);
INSERT INTO students(id, name, age) VALUES (3, anton, 17);
SELECT id, name, age FROM students;

INSERT INTO students(id, name, age) VALUES (4, dale, 20);
INSERT INTO students(id, name, age) VALUES (5, kate, 30);
INSERT INTO students(id, name, age) VALUES (6, john, 25);

SELECT id, name, age FROM students;

SELECT id, name, age FROM students WHERE age > 18;
SELECT name, age FROM students WHERE name = misa;
SELECT name FROM students WHERE age > 16 AND age < 18;

UPDATE students SET age = 18 WHERE age > 20;
SELECT id, name, age FROM students;

UPDATE students SET name = sasha WHERE id != 3;
SELECT id, name, age FROM students;

UPDATE students SET name = john WHERE id = 4;

SELECT id, name, age FROM students;

DELETE FROM students WHERE id = 4;
```

```
SELECT id, name, age FROM students;

DELETE FROM students WHERE id != 3 AND age = 16 OR age = 17;
SELECT id, name, age FROM students;

exit
```

Листинг 2: Seed Script

3 Testing Results

The server was tested using the above seed script. The commands were executed successfully, and the server responded as expected for each query. The responses from the server were verified to ensure that the database operations were carried out correctly.

```
> Executing command: CREATE TABLE students (id INT, name VARCHAR
    (255), age INT);
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
  ]
}
> Executing command: INSERT INTO students(id, name, age) VALUES (1,
     alex, 15);
> Response from server: Insert successfully
> Executing command: INSERT INTO students(id, name, age) VALUES (2,
     misa, 16);
> Response from server: Insert successfully
> Executing command: INSERT INTO students(id, name, age) VALUES (3,
     anton, 17);
> Response from server: Insert successfully
> Executing command: SELECT id, name, age FROM students;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"id": 3, "name": "anton", "age": 17},
    {"id": 2, "name": "misa", "age": 16},
    {"id": 1, "name": "alex", "age": 15}
  ]
}
> Executing command: INSERT INTO students(id, name, age) VALUES (4,
     dale, 20);
> Response from server: Insert successfully
> Executing command: INSERT INTO students(id, name, age) VALUES (5,
     kate, 30);
> Response from server: Insert successfully
> Executing command: INSERT INTO students(id, name, age) VALUES (6,
     john, 25);
> Response from server: Insert successfully
> Executing command: SELECT id, name, age FROM students;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"id": 6, "name": "john", "age": 25},
    {"id": 5, "name": "kate", "age": 30},
    {"id": 4, "name": "dale", "age": 20},
    {"id": 3, "name": "anton", "age": 17},
    {"id": 2, "name": "misa", "age": 16},
    {"id": 1, "name": "alex", "age": 15}
  ]
}
> Executing command: SELECT id, name, age FROM students WHERE age >
    18;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"id": 6, "name": "john", "age": 25},
    {"id": 5, "name": "kate", "age": 30},
    {"id": 4, "name": "dale", "age": 20}
  ]
}
> Executing command: SELECT name, age FROM students WHERE name =
   misa;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"name": "misa", "age": 16}
  ]
}
> Executing command: SELECT name FROM students WHERE age > 16 AND
   age < 18;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"name": "anton"}
  ]
}
> Executing command: UPDATE students SET age = 18 WHERE age > 20;
> Response from server: Update successfully
> Executing command: SELECT id, name, age FROM students;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"id": 6, "name": "john", "age": 18},
    {"id": 5, "name": "kate", "age": 18},
    {"id": 4, "name": "dale", "age": 20},
    {"id": 3, "name": "anton", "age": 17},
    {"id": 2, "name": "misa", "age": 16},
    {"id": 1, "name": "alex", "age": 15}
  ]
}
> Executing command: UPDATE students SET name = sasha WHERE id !=
  3;
> Response from server: Update successfully
> Executing command: SELECT id, name, age FROM students;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"id": 6, "name": "sasha", "age": 18},
    {"id": 5, "name": "sasha", "age": 18},
    {"id": 4, "name": "sasha", "age": 20},
    {"id": 3, "name": "anton", "age": 17},
    {"id": 2, "name": "sasha", "age": 16},
    {"id": 1, "name": "sasha", "age": 15}
  ]
}
> Executing command: UPDATE students SET name = john WHERE id = 4;
> Response from server: Update successfully
> Executing command: SELECT id, name, age FROM students;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"id": 6, "name": "sasha", "age": 18},
    {"id": 5, "name": "sasha", "age": 18},
    {"id": 4, "name": "john", "age": 20},
    {"id": 3, "name": "anton", "age": 17},
    {"id": 2, "name": "sasha", "age": 16},
    {"id": 1, "name": "sasha", "age": 15}
  ]
}
> Executing command: DELETE FROM students WHERE id = 4;
> Response from server: Delete successfully
> Executing command: SELECT id, name, age FROM students;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"id": 6, "name": "sasha", "age": 18},
    {"id": 5, "name": "sasha", "age": 18},
    {"id": 3, "name": "anton", "age": 17},
    {"id": 2, "name": "sasha", "age": 16},
    {"id": 1, "name": "sasha", "age": 15}
  ]
}
> Executing command: DELETE FROM students WHERE id != 3 AND age =
    16 OR age = 17;
> Response from server: Delete successfully
> Executing command: SELECT id, name, age FROM students;
> Response from server: {
  "tableName": "students",
  "columns": [
    {"columnName": "age", "dataType": "INTEGER"},
    {"columnName": "name", "dataType": "STRING"},
    {"columnName": "id", "dataType": "INTEGER"}
  ],
  "rows": [
    {"id": 6, "name": "sasha", "age": 18},
    {"id": 5, "name": "sasha", "age": 18},
    {"id": 3, "name": "anton", "age": 17},
    {"id": 1, "name": "sasha", "age": 15}
  ]
}
```

Листинг 3: Terminal Output

4 Conclusion

The implementation of the simple database server demonstrates the basic functionalities of data storage and retrieval using protocol buffers. The server successfully handles various types of SQL commands, and the testing results show that it operates as expected. This project serves as a foundational step towards building more complex database systems.