Database Commands Documentation

Dimitri Chrysafis

April 28, 2024

1 Introduction

This document provides documentation for the commands available in the Database class.

2 Commands

2.1 createTable

Description: Creates a new table in the database.
Parameters:

- `tableName`: The name of the table to create.
- `columns`: The columns of the table.

Usage:

```
db.createTable("students", {"id", "name", "age"});
```

2.2 insertInto

Description: Inserts data into a table in the database.
Parameters:

- `tableName`: The name of the table to insert data into.
- `values`: The values to insert into the table.

Usage:

```
db.insertInto("students", {"1", "John", "25"});
```

2.3 selectFrom

Description: Selects data from a table in the database.
Parameters:

- `tableName`: The name of the table to select data from.

Usage:

```
db.selectFrom("students");
```

2.4 deleteFrom

Description: Deletes data from a table in the database based on a condition.
Parameters:

- `tableName`: The name of the table to delete data from.
- `condition`: The condition to apply for deleting rows.

Usage:

```
// Example: Delete rows where age is less than 18
db.deleteFrom("students", [](const std::string& age) { return std::stoi(age) <
```

2.5 update

Description: Updates data in a table in the database based on a condition.
Parameters:

- `tableName`: The name of the table to update data in.
- `columnToUpdate`: The column to update.
- `newValue`: The new value to set for the specified column.
- `condition`: The condition to apply for updating rows.

Usage:

```
// Example: Update age to 30 for rows where name is "John"
db.update("students", "age", "30", [](const std::string& name) { return name =
```

2.6 alterTableAddColumn

Description: Adds a new column to a table in the database.
Parameters:

- `tableName`: The name of the table to add a column to.
- `newColumnName`: The name of the new column to add.

Usage:

```
db.alterTableAddColumn("students", "favorite_color");
```

2.7 dropTable

Description: Drops a table from the database.
Parameters:

- `tableName`: The name of the table to drop.

Usage:

```
db.dropTable("students");
```

2.8 createIndex

Description: Creates an index on a column of a table in the database.
Parameters:

- `indexName`: The name of the index to create.
- `tableName`: The name of the table to create the index on.
- `columnName`: The name of the column to create the index on.

Usage:

```
db.createIndex("age_index", "students", "age");
```

2.9 dropIndex

Description: Drops an index from the database.
Parameters:

- `indexName`: The name of the index to drop.

Usage:

```
db.dropIndex("age_index");
```

2.10 createView

Description: Creates a view in the database.
Parameters:

- `viewName`: The name of the view to create.
- `query`: The SQL query defining the view.

Usage:

```
db.createView("student_view", "SELECT * FROM students WHERE age > 25");
```

2.11 dropView

Description: Drops a view from the database.
Parameters:

- `viewName`: The name of the view to drop.

Usage:

```
db.dropView("student_view");
```

2.12 insertIntoSelect

Description: Inserts data into a table from the result of a SELECT query.
Parameters:

- `tableName`: The name of the table to insert data into.
- `columns`: The columns of the table.
- `selectQuery`: The SQL SELECT query to execute.

Usage:

```
// Example: Insert data into students table from another table using a select
db.insertIntoSelect("students", {"id", "name", "age"}, "SELECT * FROM other_t
```

2.13 truncateTable

Description: Truncates a table in the database (removes all data).
Parameters:

- `tableName`: The name of the table to truncate.

Usage:

```
db.truncateTable("students");
```

2.14 printTable

Description: Prints the contents of a table in the database.
Parameters:

- `tableName`: The name of the table to print.

Usage:

```
db.printTable("students");
```