DATABASE MANAGEMENT STUDIO

A MINI-PROJECT REPORT

Submitted by

MUKESH V	**241901059**
DHANESHWARAN S	**241901024**

in partial fulfillment of the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING
(CYBER SECURITY)

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project **"DATABASE MANAGEMENT STUDIO"** is the bonafide work of **"MUKESH V (241901059), DHANESHWARAN S (241901024)"** who carried out the project work under my supervision.

SIGNATURE

Ms. R. Rupmala

ASSISTANT PROFESSOR SG

Dept. of Computer Science and Engineering (Cyber Security),
Rajalakshmi Engineering College
Chennai

This mini project report is submitted for the viva voce examination to be held on

_____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We hereby declare that the mini project report **Database Management Studio**, submitted as part of the curriculum requirements for the Bachelor of Engineering (B.E) degree affiliated to Anna University, is a bonafide work carried out by us under the supervision of Ms. R. Rupmala, Assistant Professor, Department of Computer Science Engineering and Cyber Security, Rajalakshmi Engineering College, Chennai.

This submission represents our ideas in our own words, and where ideas or words of others have been included, we have adequately and accurately cited and referenced
the original sources.

We also declare that we have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data, idea, fact, or source in our submission. We understand that any violation of the above will be grounds for disciplinary action by the institute and/or the University and may also evoke penal action from the sources which have not been properly cited or from whom proper permission has not been obtained. This report has not previously formed the basis for the award of any degree, diploma, or similar title of any other University.

ABSTRACT

Database Management Studio is a modern, cross-platform database management studio built with Java 21 and JavaFX. It features a custom Object-Relational Mapping (ORM) implementation and AI-powered database assistance, supporting PostgreSQL, MySQL, and Oracle databases. The project showcases advanced software engineering practices, including custom ORM design patterns, asynchronous API integration, multi-threaded UI architecture, and cross-platform deployment strategies. The application offers a professional JavaFX interface with dark/light theme support, advanced import/export capabilities (CSV, Excel, PDF), and robust security measures to prevent SQL injection attacks. Key achievements include a custom lightweight ORM framework with full CRUD operations, multi-database support, Google Gemini API integration for intelligent database assistance, a modern and intuitive user interface, comprehensive data processing and export functionality, and a security-first design.

ACKNOWLEDGEMENT

We like to convey our sincere appreciation to all who have supported and mentored us during the successful completion of this project work.

We express our profound gratitude to **Mr. Benedict J.N**., Associate Professor (SG) and Head of the Department of Computer Science Engineering and Cyber Security at Rajalakshmi Engineering College, for furnishing us with essential resources, support, and an enabling environment to execute this project.

We express our profound gratitude to **Ms. R. Rupmala**, Assistant Professor in the Department of Computer Science Engineering and Cyber Security at Rajalakshmi Engineering College, for her unwavering support, invaluable insights, and collaboration throughout our endeavor.

We would like to convey our gratitude to our faculty members and colleagues for their valuable feedback and encouragement.

We express our gratitude to our families and friends for their steadfast support, patience, and encouragement, which were instrumental in the effective execution of this seminar. .

MUKESH V

DHANESHWARAN S

TABLES OF CONTENT

LIST OF ABBREVIATION

Abbreviation	Expansion
ORM	Object-Relation Mapping
CRUD	Create, Read, Update, Delete
AI	Artificial Intelligence
JDBC	Java Database Connectivity
API	Application Programming Interface
UI	User Interface
UX	User Experience
CSV	Comma-Separated Values
PDF	Portable Document Format
SQL	Structured Query Language
JAR	Java Archive
ER	Entity-Relationship
RDS	Relational Database Service

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In the contemporary data-driven software landscape, database management tools serve as critical infrastructure for developers, data analysts, and database administrators. Traditional database management tools often suffer from several limitations including vendor lock-in, complexity, high costs, limited intelligence, and poor user experience.Database Management Studio addresses these challenges by providing an accessible, powerful, and intelligent database management solution that democratizes database operations while maintaining professional-grade capabilities.

1.2 SCOPE OF THE WORK

- Multi-database support (PostgreSQL, MySQL, Oracle)
- Custom ORM implementation with full CRUD operations
- AI-powered database assistance using Google Gemini API
- Data import/export functionality (CSV, Excel, PDF)
- Modern JavaFX user interface with dark/light themes
- Cross-platform compatibility (Windows, macOS, Linux)
- Comprehensive error handling and security measures

1.3 PROBLEM STATEMENT

Traditional database management tools present several challenges:

1. **Vendor Lock-in**: Most commercial tools are tied to specific database vendors, limiting flexibility
2. **Complexity**: Enterprise tools require extensive training and have steep learning curves

3. **Cost**: Professional-grade tools are prohibitively expensive for individual developers and small teams
4. **Limited Intelligence**: Few tools offer AI-powered assistance for database operations
5. **Poor User Experience**: Many tools lack modern, intuitive interfaces
6. **Security Concerns**: Some tools may not implement proper security measures against SQL injection

1.4 AIM AND OBJECTIVES OF THE PROJECT

Primary Objectives:

1. **Democratize Database Management**: Create an accessible, powerful tool for database operations
2. **Educational Value**: Demonstrate advanced software engineering concepts through practical implementation
3. **Innovation**: Integrate cutting-edge AI technologies with traditional database operations
4. **Cross-Platform Compatibility**: Ensure seamless operation across Windows, macOS, and Linux
5. **Security-First Design**: Implement robust security measures from the ground up

Specific Goals:

- Develop a custom ORM framework with full CRUD capabilities
- Integrate AI assistance for intelligent query generation and optimization
- Create a modern, responsive user interface
- Support multiple database systems with unified interface
- Implement comprehensive data import/export functionality
- Ensure high performance and security standards

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Minimum Requirements:

- **Processor**: Intel Core i3 or equivalent (2 GHz)
- **Memory**: 4 GB RAM
- **Storage**: 500 MB available disk space
- **Display**: 1280 x 720 resolution
- **Network**: Internet connection for AI features

Recommended Requirements:

- **Processor**: Intel Core i5 or equivalent (3 GHz or higher)
- **Memory**: 8 GB RAM or higher
- **Storage**: 1 GB available disk space
- **Display**: 1920 x 1080 resolution or higher
- **Network**: Stable internet connection

2.2 SOFTWARE SPECIFICATIONS

Operating System:

- Windows 10/11
- macOS 10.15 or later
- Linux (Ubuntu 18.04+, CentOS 7+, etc.)

Runtime Environment:

- **Java**: OpenJDK 21 or Oracle JDK 21
- **JavaFX**: Version 21 (included in distribution)

Supported Databases:

- **PostgreSQL**: Version 12.0 or later
- **MySQL**: Version 8.0 or later
- **Oracle Database**: Version 19c or later

Development Tools:

- **Build Tool**: Gradle 7.0+
- **IDE**: IntelliJ IDEA, Eclipse, or VS Code with Java extensions
- **Version Control**: Git

Dependencies:

- PostgreSQL JDBC Driver (42.7.3)
- MySQL Connector/J (8.0.33)
- Oracle JDBC Driver (23.3.0.23.09)
- Apache POI (5.2.5) for Excel operations
- OpenPDF (1.3.30) for PDF generation
- BootstrapFX (0.4.0) for UI styling

CHAPTER 3

MODULE DESCRIPTION

3.1 Core ORM Module:

- **Purpose**: Provides the custom Object-Relational Mapping functionality
- **Key Classes**:
  SimpleORM
- **Functions**:
  - createTable()
    : Creates database tables dynamically
  - dropTable()
    : Removes database tables
  - select()
    : Performs SELECT queries with optional WHERE clauses
  - insert()
    : Inserts new records using prepared statements
  - update()
    : Updates existing records with prepared statements
  - delete()
    : Deletes records with optional WHERE clauses

3.2 User Interface Module:

- **Purpose**: Manages the graphical user interface and user interactions
- **Key Components**:
  - HelloApplication
    : Main JavaFX application class
  - HelloController
  - : Handles UI events and business logic

    - Hello-view.fxml
    - : Defines the UI layout and structure
- **Features**:
    - Tabbed interface for different functionalities
    - Connection management (URL and parameter-based)
    - Query execution and result display
    - Table browsing and data manipulation
    - AI assistance integration
    - Theme switching (dark/light mode)

3.3 Data Processing Module:

- **Purpose**: Handles import/export operations
- **Supported Formats**:
    - CSV (Comma-Separated Values)
    - Excel (.xlsx)
    - PDF reports
- **Libraries Used**:
    - Apache POI for Excel operations
    - OpenPDF for PDF generation

3.4 AI Integration Module:

- **Purpose**: Provides intelligent database assistance
- **Integration**: Google Gemini API
- **Features**:
    - Query optimization suggestions
    - Natural language to SQL conversion
    - Database schema analysis
    - Performance recommendations

3.5 Build and Distribution Module:

- **Purpose**: Manages project dependencies and build process
- **Features**:
    - Gradle build system configuration
    - Shadow JAR creation for distribution
    - Runtime image generation
    - Cross-platform packaging

CHAPTER 4

SAMPLE CODING

4.1 Core Architecture

SimpleORM.java - Custom ORM Implementation

```
package com.Database Management Studio.orm; import
java.sql.*;
import java.util.ArrayList; import
java.util.List;
import java.util.Map;

public class SimpleORM {
    private Connection connection;

    public SimpleORM(Connection connection) {
        this.connection = connection;
    }

    public void createTable(String tableName, Map<String, String>
columns) throws SQLException {
        StringBuilder sql = new StringBuilder("CREATE TABLE " +
tableName + " (");
        List<String> colDefs = new ArrayList<>(); for
        (Map.Entry<String, String> entry :
columns.entrySet()) {
            colDefs.add(entry.getKey() + " " + entry.getValue());
        }
        sql.append(String.join(", ", colDefs)).append(");");
        executeUpdate(sql.toString());
    }
```

```
    public ResultSet select(String tableName, String whereClause)
throws SQLException {
            String sql = "SELECT * FROM " + tableName;
        if (whereClause != null && !whereClause.isEmpty()) { sql +=
            " WHERE " + whereClause;
        }
        return executeQuery(sql);
    }

    public int insert(String tableName, Map<String, Object> values)
throws SQLException {
        StringBuilder sql = new StringBuilder("INSERT INTO "
+ tableName + " (");
        StringBuilder vals = new StringBuilder("VALUES ("); List<String>
        cols = new ArrayList<>();
        List<String> placeholders = new ArrayList<>(); for
        (Map.Entry<String, Object> entry :
values.entrySet()) {
            cols.add(entry.getKey());
            placeholders.add("?");
        }
        sql.append(String.join(", ", cols)).append(") ");
        vals.append(String.join(", ",
placeholders)).append(")");
        sql.append(vals);
        try (PreparedStatement stmt =
connection.prepareStatement(sql.toString())) { int i
            = 1;
            for (Object value : values.values()) {
                stmt.setObject(i++, value);
            }
            return stmt.executeUpdate();
        }
    }

    // Additional CRUD methods...
}
HelloApplication.java - Main Application Class package
com.Database Management Studio.orm;
```

```
import javafx.application.Application; import
javafx.fxml.FXMLLoader;
import javafx.scene.Scene; import
javafx.stage.Stage;

import java.io.IOException;

public class HelloApplication extends Application { @Override
    public void start(Stage stage) throws IOException { FXMLLoader
        fxmlLoader = new
FXMLLoader(HelloApplication.class.getResource("hello-view.fx ml"));
                                    Scene scene = new
800);                               Scene(fxmlLoader.load(), 1200,

                                    stage.setTitle("SQL ORM Studio by
                                    MULTi"); stage.setScene(scene);

scene.getStylesheets().add("org/kordamp/bootstrapfx/bootstra
pfx.css");

scene.getStylesheets().add(getClass().getResource("dark.css"
).toExternalForm());
        stage.show();
    }

    public static void main(String[] args) { launch();
    }
}
```

4.2 User Interface Implementation

FXML Layout Structure

```
<?xml version="1.0" encoding="UTF-8"?>
<VBox xmlns:fx="http://javafx.com/fxml" fx:controller="com.Database
Management
```

```
Studio.orm.HelloController">
    <MenuBar>
        <Menu text="File">
            <MenuItem text="Exit" onAction="#exitApp"/>
        </Menu>
        <Menu text="View">
            <CheckMenuItem fx:id="darkModeMenuItem" text="Dark
Mode" onAction="#toggleTheme"/>
            <MenuItem text="API Settings"
onAction="#showApiSettings"/>
        </Menu>
    </MenuBar>
    <TabPane tabClosingPolicy="UNAVAILABLE"
prefHeight="800.0" prefWidth="1200.0" VBox.vgrow="ALWAYS">
        <Tab text="Connection">
            </Tab>
        <Tab text="Query">
            </Tab>
        <Tab text="Tables">
            </Tab>
        <Tab text="AI Assistant">
            </Tab>
    </TabPane>
</VBox>
```

4.3 Build Configuration

build.gradle

```
plugins {
    id 'java'
    id 'application'
    id 'org.openjfx.javafxplugin' version '0.1.0'
    id 'com.github.johnrengelman.shadow' version '8.1.1' id
    'org.beryx.runtime' version '1.13.0'
}

group 'com.Database Management Studio'
```

```
version '1.0-SNAPSHOT'

sourceCompatibility = '21'
targetCompatibility = '21'

javafx {
    version = '21'
    modules = ['javafx.controls', 'javafx.fxml']
}

dependencies {
    implementation 'org.openjfx:javafx-controls:21:win'
    implementation 'org.openjfx:javafx-fxml:21:win'
    implementation
'org.kordamp.bootstrapfx:bootstrapfx-core:0.4.0'
    implementation 'org.postgresql:postgresql:42.7.3'
    implementation 'mysql:mysql-connector-java:8.0.33'
    implementation
'com.oracle.database.jdbc:ojdbc11:23.3.0.23.09'
    implementation 'org.apache.poi:poi:5.2.5'
    implementation 'org.apache.poi:poi-ooxml:5.2.5'
    implementation 'com.github.librepdf:openpdf:1.3.30'
}
```

CHAPTER 5

SCREENSHOTS

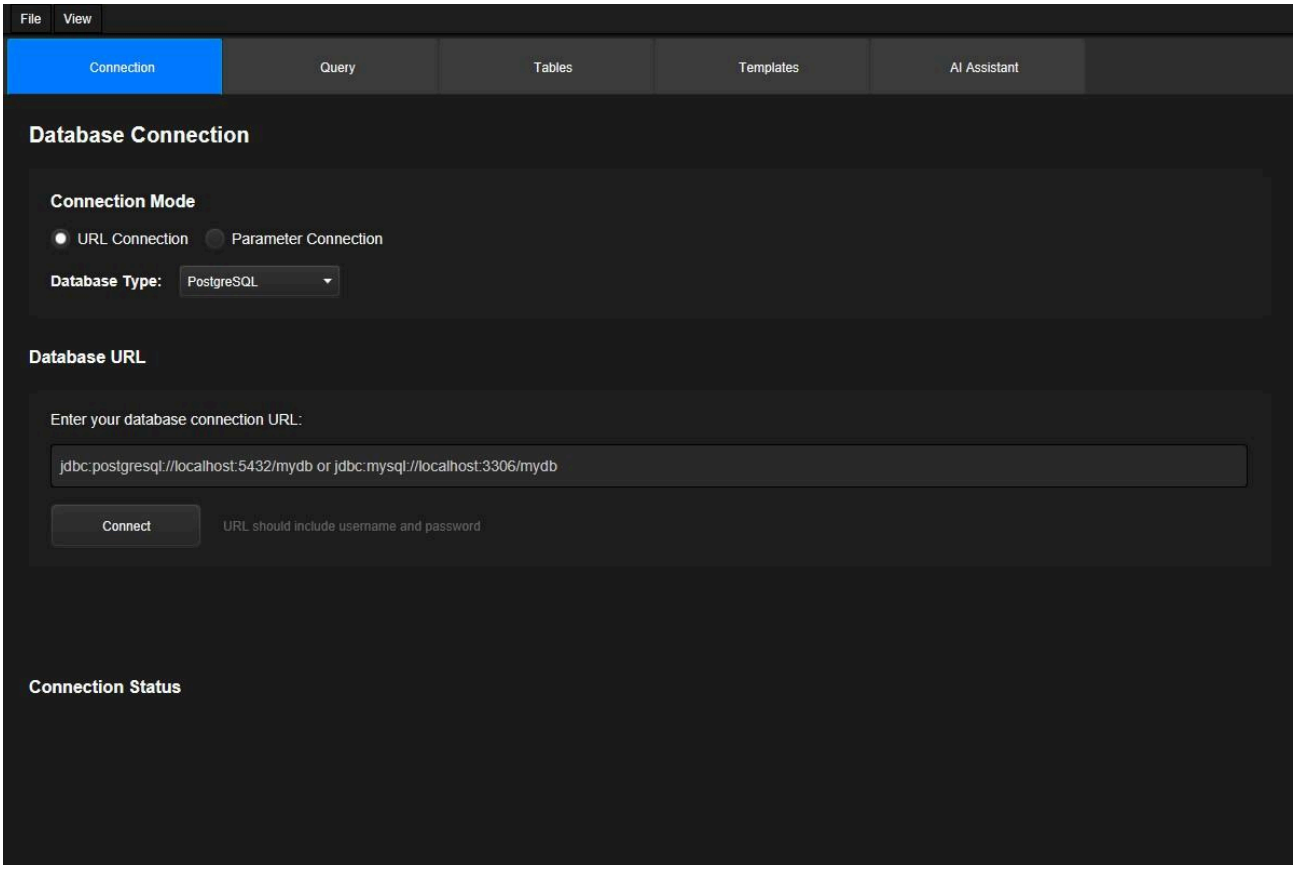

Fig 5.1 URL Connection page

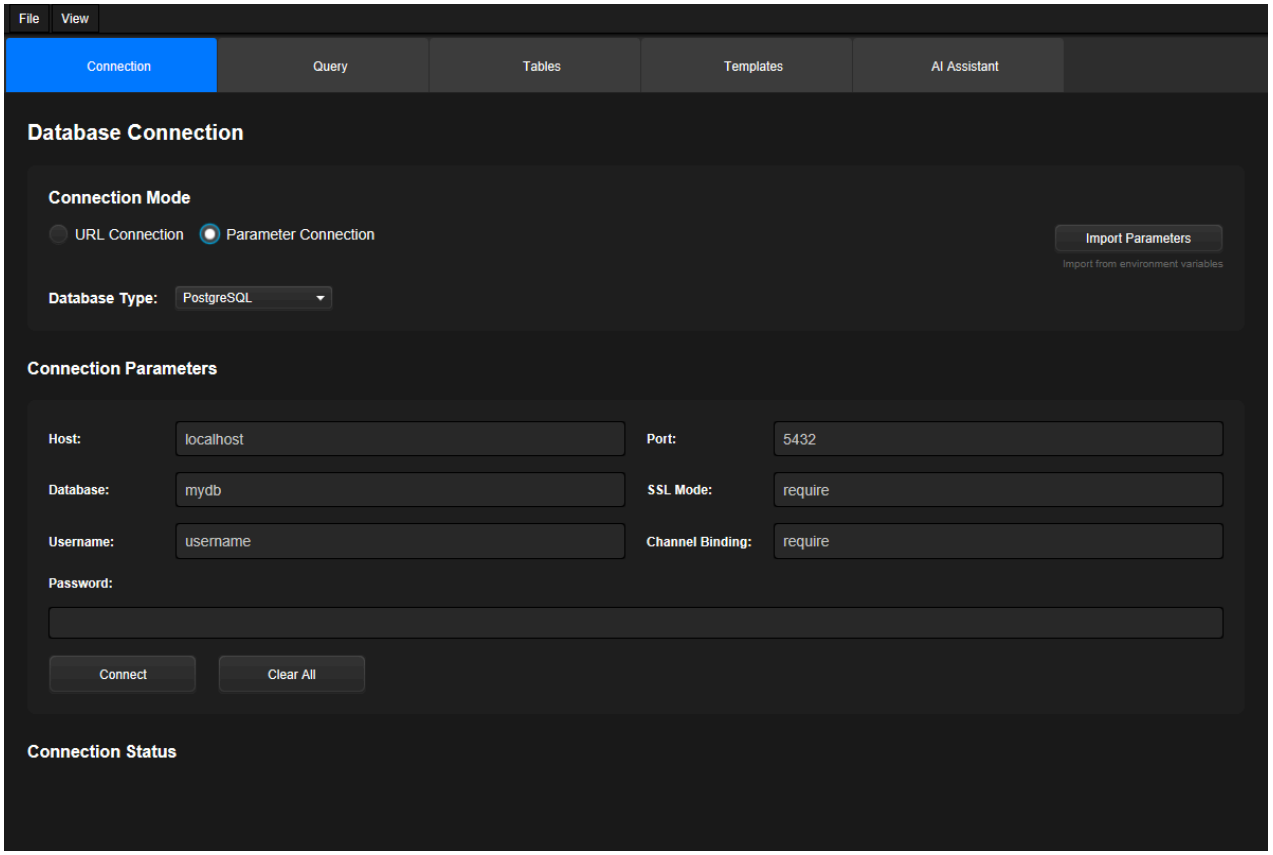

Fig 5.1.1 Parameter Connection page

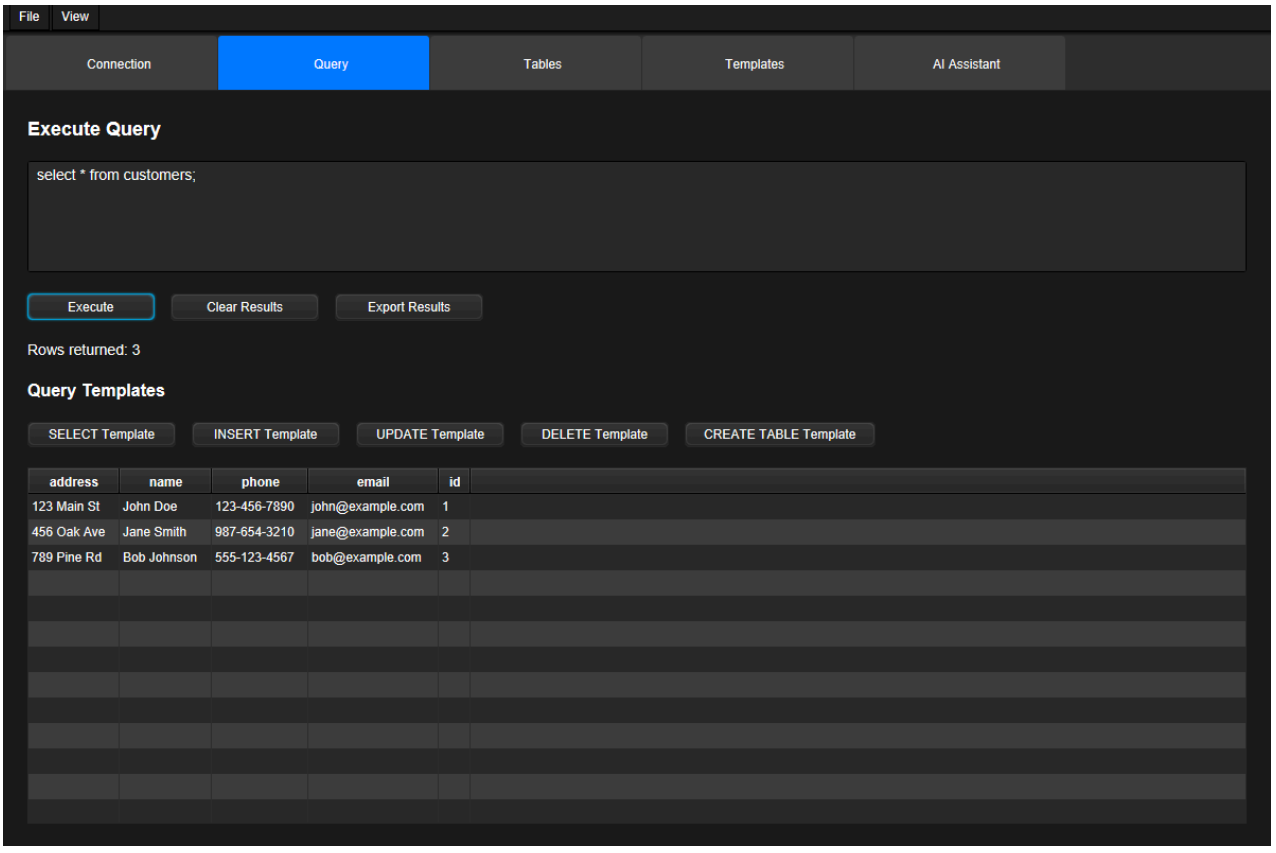

Fig 5.2 Query Section

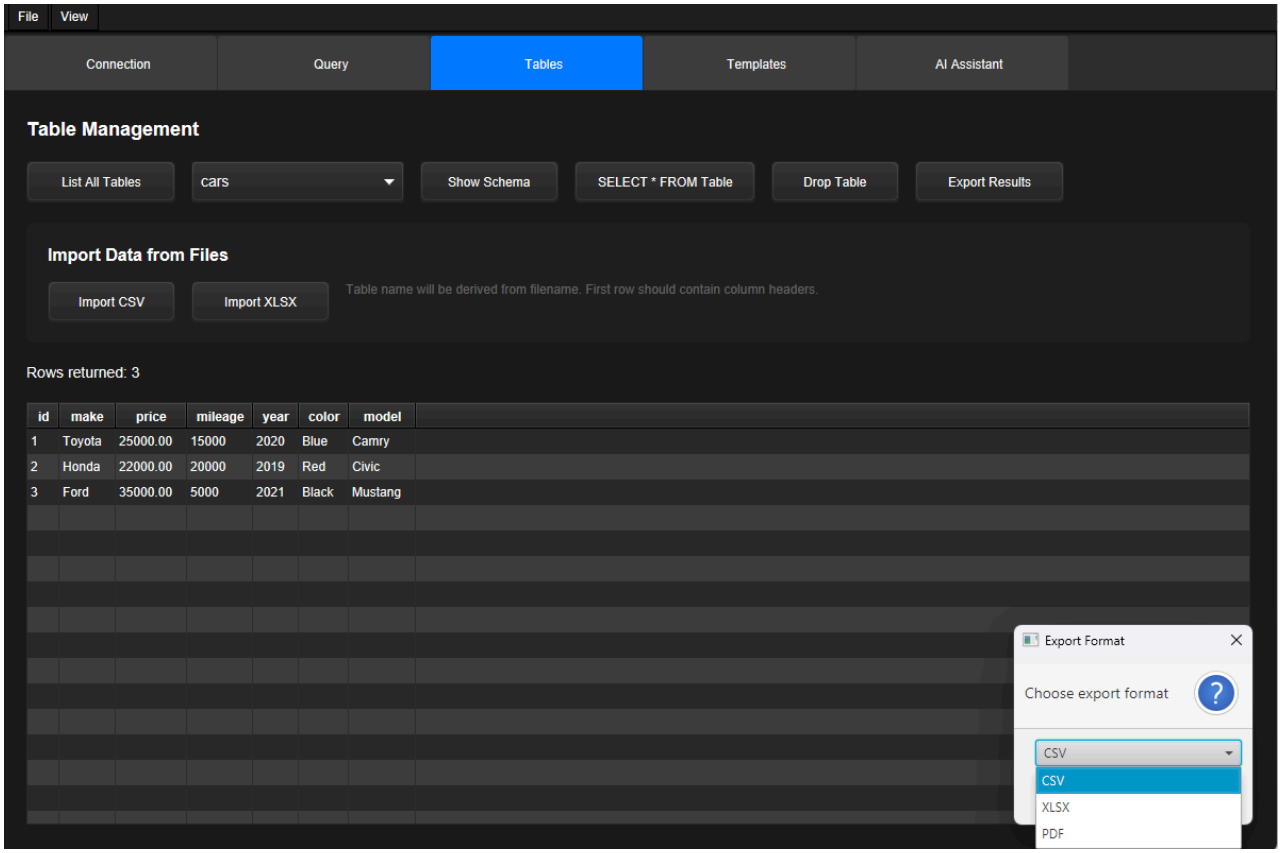

Fig 5.3 Tables Section

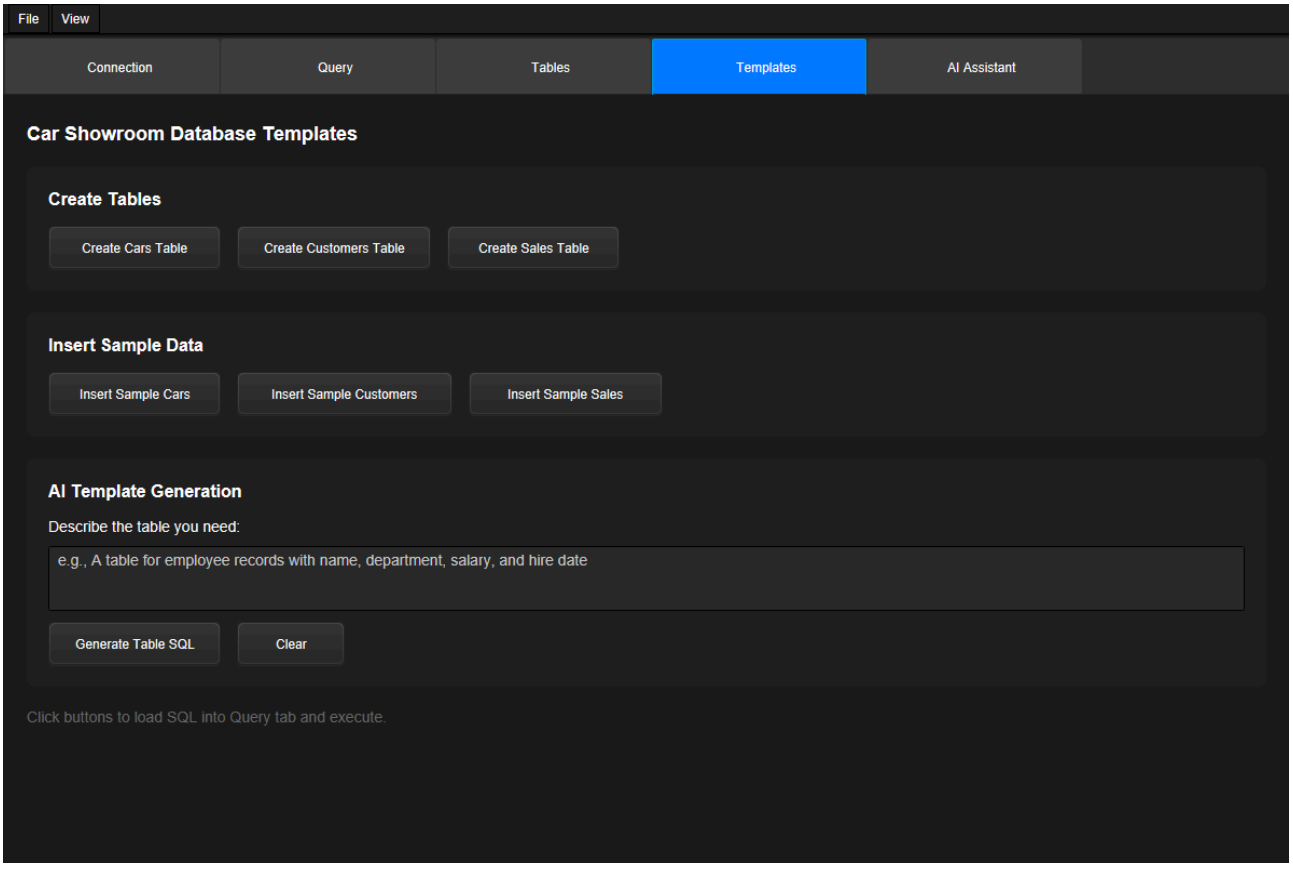

Fig 5.4 Templates Section

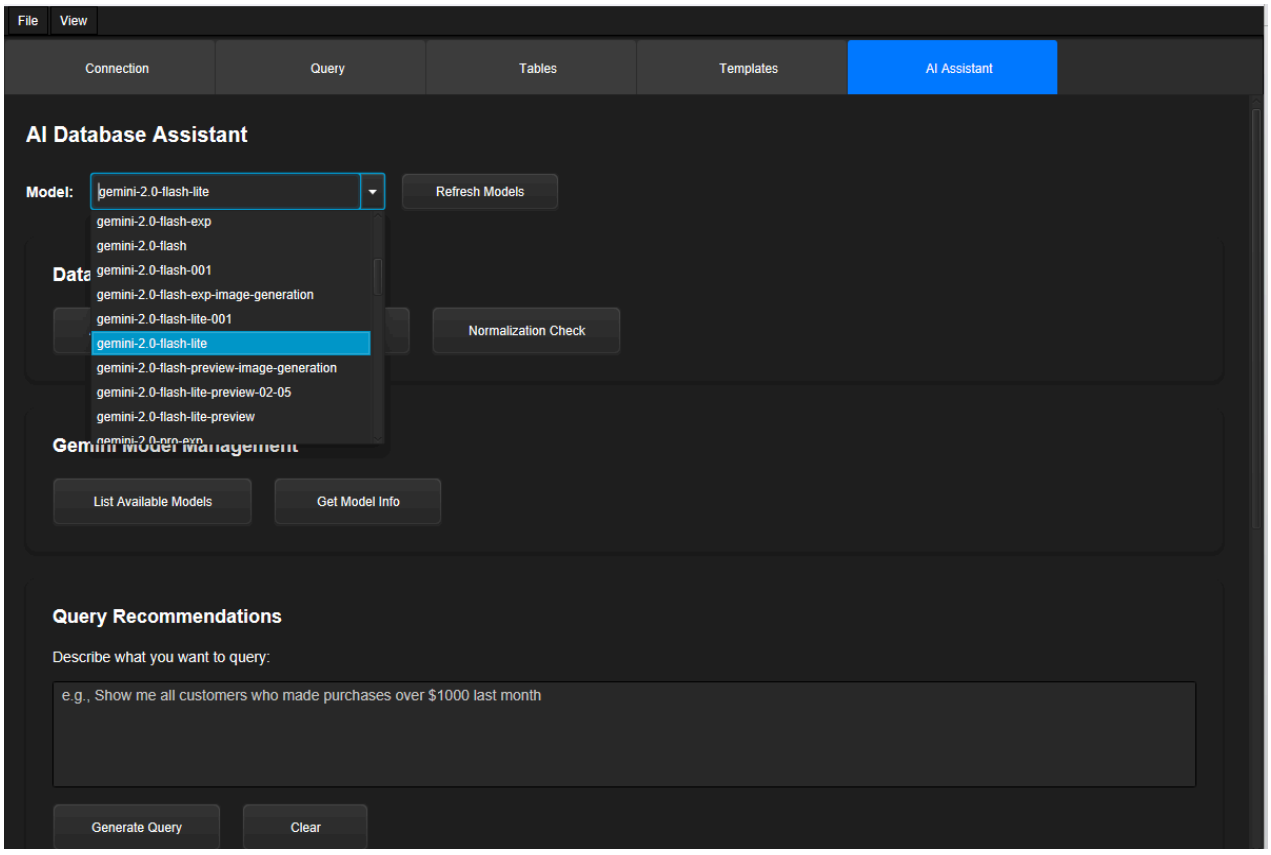

Fig 5.5 AI Assistant Section

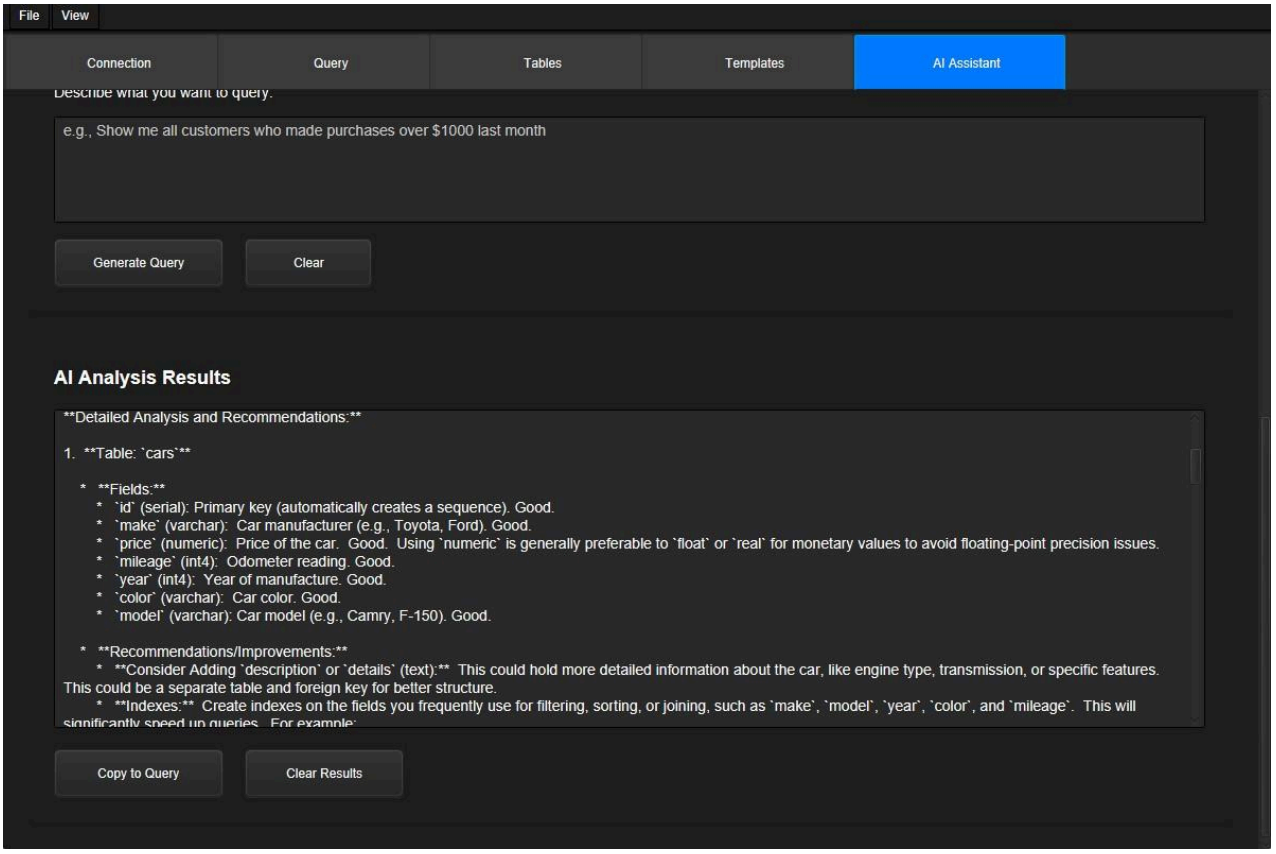

Fig 5.5.1 AI Assistant Section [Result for Schema Analysis]

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

The Database Management Studio project successfully demonstrates the implementation of a modern, feature-rich database management studio that addresses many of the shortcomings found in traditional database tools. The system provides a comprehensive solution that combines custom ORM capabilities, AI-powered assistance, and a modern user interface. Key achievements include:

- Successful implementation of a custom ORM framework with full CRUD operations
- Integration of AI assistance for intelligent database operations
- Support for multiple database systems through a unified interface
- Professional-grade user interface with modern design principles
- Robust security implementation using prepared statements
- Cross-platform compatibility and distribution

  The project serves as both a practical tool and an educational resource, demonstrating advanced software engineering concepts including ORM design patterns, asynchronous programming, UI/UX design, and API integration.

6.1 FUTURE ENHANCEMENT

Planned Enhancements:

1. **Advanced Query Builder**: Visual query construction interface
2. **Database Schema Visualization**: ER diagram generation and editing

3. **Performance Monitoring**: Real-time query performance analysis
4. **Backup and Recovery**: Automated database backup solutions
5. **Multi-language Support**: Internationalization and localization
6. **Plugin Architecture**: Extensible plugin system for custom functionality
7. **Cloud Integration**: Support for cloud database services (AWS RDS, Google Cloud SQL, Azure Database)
8. **Advanced Analytics**: Built-in data analysis and visualization tools
9. **Team Collaboration**: Multi-user editing and version control for database schemas
10. **Mobile Companion**: Mobile application for remote database monitoring

Technology Roadmap:

- **Short-term (6 months)**: Query builder, schema visualization, performance monitoring
- **Medium-term (12 months)**: Cloud integration, advanced analytics, team collaboration
- **Long-term (24 months)**: Plugin ecosystem, mobile application, enterprise features

  The project establishes a solid foundation for continued development and enhancement, positioning Database Management Studio as a competitive alternative to commercial database management tools.

REFERENCES

1. Oracle Corporation. (2023). JavaFX Documentation. Retrieved from https://openjfx.io/
2. PostgreSQL Global Development Group. (2023). PostgreSQL Documentation. Retrieved from https://www.postgresql.org/docs/
3. Oracle Corporation. (2023). MySQL Documentation. Retrieved from https://dev.mysql.com/doc/
4. Oracle Corporation. (2023). Oracle Database Documentation. Retrieved from https://docs.oracle.com/en/database/
5. Google LLC. (2023). Google Gemini API Documentation. Retrieved from https://ai.google.dev/docs
6. Apache Software Foundation. (2023). Apache POI Documentation. Retrieved from https://poi.apache.org/
7. OpenPDF. (2023). OpenPDF Documentation. Retrieved from https://github.com/LibrePDF/OpenPDF
8. Gradle Inc. (2023). Gradle Build Tool Documentation. Retrieved from https://gradle.org/documentation/
9. Kordamp. (2023). BootstrapFX Documentation. Retrieved from https://github.com/kordamp/bootstrapfx
10. Fielding, R. T., & Taylor, R. N. (2002). Principled design of the modern Web architecture. ACM Transactions on Internet Technology, 2(2), 115-150.