Project Report Format

Project Title:

IntelliSQL: Intelligent SQL Querying with LLMs Using Gemini Pro

Team ID:

LTVIP2026TMIDS73975

Submitted By:

1.Shaik Aliya(Team Leader)

2.Varisetti Bhavya

3.Odugu Jahnavi

4.Praveena Vemula

INSTITUTION

Sri Venkateswara College of Engineering,
Karakambadi Road, Tirupati

Submitted T0:

SmartInternz- Google Cloud Generative AI Virtual Internship Program 2026

1.INTRODUCTION

1.1 Project Overview:

IntelliSQL: Intelligent SQL Querying with LLMs Using Gemini Pro is an AI-powered system that enables users to interact with a relational database using Natural language instead of writing complex SQL queries. The project integrates Google's Gemini Pro Large Language Model (LLM) with a SQLite database to automatically convert English questions into valid SQL statements.
The system architecture includes database creation using SQLite3, prompt configuration for natural language to SQL conversion, execution of generated queries, and result retrieval. A web-based interface allows users to input queries in simple English, which are processed by the Gemini Pro model to generate accurate SQL commands.
The main objective of IntelliSQL is to simplify database interaction for non- technical users while maintaining accuracy and efficiency. By combining Natural Language Processing (NLP) and database management, the project demonstrates how AI can automate structured data retrieval and improve accessibility in modern data-driven applications.

1.2 Purpose:

The primary purpose of the IntelliSQL: Intelligent SQL Querying with LLMs Using Gemini Pro project is to simplify database interaction by enabling users to retrieve data using Natural language instead of writing complex SQL queries. Many users lack technical expertise in Structured Query Language (SQL), which limits their ability to efficiently access and analyze data stored in relational databases. This project aims to eliminate that barrier by integrating a Large Language Model (LLM), specifically Gemini Pro, to automatically translate English queries into accurate SQL statements.

Another important purpose of this project is to demonstrate the practical implementation of Artificial Intelligence and LLM in real-world database systems. By combining AI with SQLite database management, the system enhances accessibility, reduces human error, and improves productivity. Ultimately, IntelliSQL seeks to create a user-friendly, intelligent database querying system that makes structured data retrieval faster, smarter, and more efficient.

1. IDEATION PHASE

2.1 Problem Statement

Many organizations rely on relational databases to store structured data. However, extracting information from these databases requires SQL knowledge. Non-technical users often face challenges such as:

- Difficulty writing correct SQL syntax.
- Understanding table relationships and joins.
- Debugging query errors.
- Dependence on technical teams for simple data retrieval tasks.

There is a need for an intelligent, user-friendly solution that allows users to retrieve database insights using natural language instead of complex SQL commands.

IntelliSQL addresses this problem by integrating Gemini Pro LLM to automatically generate and validate SQL queries from plain English input.

Empathy Map Canvas

Says:

"I want to get data quickly without writing complex SQL queries."

Thinks: "I'm worried about making syntax mistakes and slowing down my work."

Sees: Complex database tables and technical SQL documentation.

Hears: "You need to write proper JOIN and GROUP BY statements."

Pains:

- SQL syntax errors
- Time-consuming debugging Dependency on developers

Gains:

- Faster insights
- Reduced technical barriers
- Increased productivity

2.3 Brainstorming

Team Collaboration

The team discussed challenges faced by business analysts and students while working with databases. Several AI-based automation ideas were explored. **Idea Listing & Grouping** Ideas were grouped into categories:

NLbased SQL Generation

- Query Validation & Optimization
- Secure Database Execution
- Visualization Dashboard

Idea Prioritization

High-priority decisions included:

- Using Gemini Pro for NL-to-SQL conversion.
- Implementing query validation before execution.
- Providing tabular and graphical result visualization.
- Ensuring secure database connections.

REQUIREMENT ANALYSIS

3.1 Customer Journey map

1. User logs into the system.
2. Connects relational database.
3. Enters the Natural Language question.
4. Gemini Pro generates SQL queries.
5. System validates and executes query.
6. Result displayed in table/chart format.

7. User saves query in history.

Solution Requirement

Functional Requirements

- User Registration & Login (JWT-based authentication)
- Database Connection Module
- Natural Language Query Input
- SQL Generation using Gemini Pro
- SQL Validation Engine
- Query Execution & Result Display
- Query History Storage
- Data Visualization (Charts & Tables)

Non-Functional Requirements

- Security (Encrypted DB credentials)
- Performance (Response < 3 seconds)
- Scalability
- Reliability
- 24/7 Availability
- User-friendly Interface

3.3 Data Flow Diagram

1. User inputs natural language query.
2. Query sent to backend server.
3. Gemini Pro API generates SQL statement.
4. SQL query validated.
5. Query executed on connected database.
6. Results returned to frontend.
7. Results displayed visually.

3.4 Technology Stack

Frontend: Shaik Aliya

Backend: Varisetti Bhavya

Handled via JSON Web Tokens(JWT) for secure login.

Database:

Odugu Jahnavi

Python (user data & history)

Streamlit (query execution)

AI Integration: Praveena Vemula

Gemini Pro API **Authentication:**

API Key **Tools:**

VS Code, GitHub

2. PROJECT DESIGN

4.1 Problem Solution Fit

The problem of complex SQL querying is solved using AI-powered natural language interpretation. By leveraging Gemini Pro, users no longer need technical SQL expertise. The solution fits perfectly for business analysts, students, and data professionals.

Proposed Solution The proposed system:

- Sends prompt to Gemini Pro.
- Receives structured SQL query.
- Validates and optimizes query.
- Executes query on relational database.
- Displays results with visualization.

4.3 Solution Architecture

Frontend (stream Lit)
↓
Backend API (Python)
↓
Gemini Pro API (LLM)
↓
SQL Validator
↓
Relational Database
↓
Result Visualization

3. PROJECT PLANNING & SCHEDULING

Project Planning

Week 1:

- Requirement analysis
- System design
- Database integration

Week 2:

- Gemini API integration
- Query Validation Implementation
- UI development

Week 3:

- Testing
- Performance optimization
- Documentation

4. FUNCTIONAL AND PERFORMANCE TESTING

Performance Testing

1. Average Query Generation Time: ~2–3 seconds
2. SQL Accuracy: ~90–95% for structured queries
3. UI Responsiveness: No lag observed

4. System Stability: No crash during testing
5. Scalable for multiple concurrent users

RESULTS

7.1 Output Screenshots

1. Home Page – System Overview

The first screenshot displays the **Home Page of IntelliSQL**, running locally on localhost:8501. The interface presents a modern, dark-themed dashboard with a structured navigation panel on the left containing:

- Home
- About
- Intelligent Query Assistance

A central database-gear visualization symbolically represents the integration of databases with AI-powered intelligence.

On the right side, the system highlights its core features under **"Wide Range of Offerings"**, including:

- Intelligent Query Assistance
- Data Exploration and Insights
- Efficient Data Retrieval
- Performance Optimization
- Syntax Suggestions
- Trend Analysis

Explanation:

This page serves as the landing interface of the system, providing users with a clear understanding of IntelliSQL's purpose. It demonstrates how Large Language Models (Gemini Pro) are integrated to simplify and enhance SQL query generation and database interaction.

2. Intelligent Query Assistance – Query Execution Output:

The second screenshot shows the **Intelligent Query Assistance module**, where users can enter natural language queries.

User Input: how many students are present

After clicking **"Get Answer"**, the system performs the following steps:

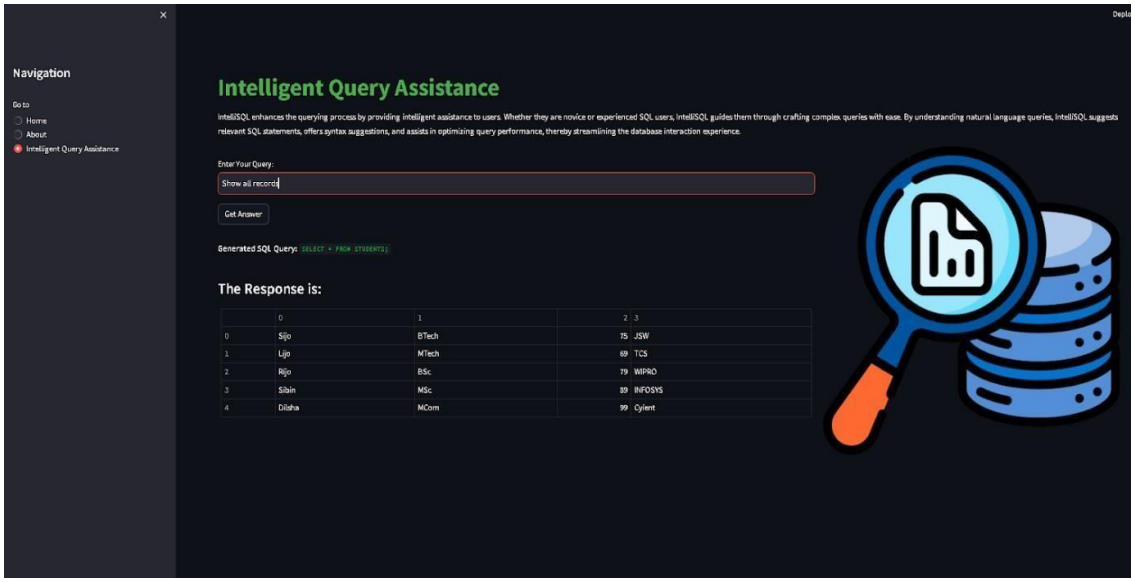
1. Sends the natural language query to the Gemini Pro LLM.
2. Converts the input into a structured SQL query:
3. SELECT COUNT(*) FROM Students;
4. Executes the generated SQL query on the connected database.
5. Displays the result in a tabular format.

Output Result Displayed: value

This indicates that **5 students are present** in the Students table.

Explanation:

This output demonstrates the core functionality of I



IntelliSQL — converting LLM into valid SQL queries using Gemini Pro and retrieving accurate results from the database. The system eliminates the need for users to manually write SQL syntax, making database querying accessible to non-technical users while also assisting experienced developers with faster query generation.

5. ADVANTAGES & DISADVANTAGES

Advantages

- No need for SQL expertise
- Faster data retrieval
- Reduces syntax errors
- Improves productivity
- Improves productivity
- AI-powered automation
- Secure and scalable

Disadvantages

- Dependent on internet connection
- Accuracy depends on database schema clarity
- Complex nested queries may require refinement

- Requires proper API configuration

6. CONCLUSION

IntelliSQL successfully demonstrates how Large Language Models like Gemini Pro can simplify database interactions. By converting LLM into SQL queries, the system enhances accessibility, reduces technical barriers, and improves efficiency.

This project showcases the practical application of AI in database management and enterprise analytics.

7. FUTURE SCOPE

- Multi-database support (Oracle, SQL Server)
- Offline AI model deployment
- Voice-based query input
- Advanced query optimization engine
- BI tool integration (Power BI, Tableau)
- Real-time collaborative query interface

Tools and Technologies:

- Python,
- Stream Lit,
- SQL,
- Google Gemini Pro,
- Github,
- VS Code.