Distributed System - Project Report

Une image contenant Bleu électrique, bleu, logo, Graphique

Description générée automatiquement

Title: Distributed System Project Report

Course: Distributed System CO3072

Professor: Dr. Thoai Nam

Authors:

Julien Holzer, 2360059

Lê Tiến Phát, 2252600

Group: No12

Institution: Ho Chi Minh City University of Technology

Faculty: Computer Science and Engineering

Date: 05 January 2025

Project subject: Data stream and storage with analysis system for a restaurant recommendations application

Table of contents

[Introduction 3](#_Toc187164489)

[System Architecture 3](#_Toc187164490)

[Database Layer: TimescaleDB 3](#_Toc187164491)

[Application Layer 4](#_Toc187164492)

[Distributed System Aspects 5](#_Toc187164493)

[Data Distribution 5](#_Toc187164494)

[System Characteristics 6](#_Toc187164495)

[Implementation Details 7](#_Toc187164496)

[Data Model 7](#_Toc187164497)

[Key Features 7](#_Toc187164498)

[Technical Considerations 8](#_Toc187164499)

[Benefits of TimescaleDB 8](#_Toc187164500)

[System Extensibility 8](#_Toc187164501)

[Performance Considerations 9](#_Toc187164502)

[Query Optimization 9](#_Toc187164503)

[Scalability Testing 9](#_Toc187164504)

[Future Recommendations 10](#_Toc187164505)

[Conclusion 10](#_Toc187164506)

[Demo of the lab 10](#_Toc187164507)

# Introduction

The Restaurant Recommendation System is a distributed system designed to offer personalized restaurant recommendations for users in various cities in the United States. By employing TimescaleDB as the core of its data management layer, the system optimizes time-series data storage and querying while maintaining high performance and scalability. The application incorporates distributed system principles such as horizontal scalability, fault tolerance, and concurrency, ensuring a reliable and location-agnostic experience for users. This report provides a detailed overview of the system’s architecture, features, technical considerations, and recommendations for future enhancements.

# System Architecture

## Database Layer: TimescaleDB

TimescaleDB forms the core of the system's database layer and provides several advanced capabilities:

1. **Time-Series Optimization**  
   TimescaleDB automatically partitions data by time, allowing for efficient handling of temporal queries. It also optimizes data retention and archival, ensuring that both recent and historical data are managed effectively.
2. **Scalability Features**  
   The database supports horizontal scaling through a multi-node architecture, enabling parallel query processing and automated data partitioning. This design ensures the system can handle increasing volumes of data without compromising performance.
3. **Performance Benefits**  
   TimescaleDB enhances query performance with specialized indexing, data compression, and efficient handling of large datasets. These features allow the system to process user queries rapidly, even under high loads.

## Application Layer

The system is built on a three-tier architecture that separates concerns for better maintainability and scalability:

1. **Data Access Layer**  
   The data access layer interacts directly with TimescaleDB to handle data cleaning, validation, and transaction management. It ensures that only clean and accurate data is stored in the database.
2. **Business Logic Layer**  
   This layer implements restaurant ranking algorithms, sort orders, and review management. It acts as the brain of the system, processing data and applying logic to provide meaningful recommendations to users.
3. **Presentation Layer**  
   The presentation layer provides an interactive, console-based user interface. It handles input validation, formats results for display, and ensures a user-friendly experience.

# Distributed System Aspects

## Data Distribution

The system adopts distributed computing principles to ensure high performance and reliability:

1. **Horizontal Scalability**  
   TimescaleDB distributes data across multiple nodes, enabling each node to handle queries independently. Automatic load balancing ensures even distribution of queries across the cluster.
2. **Concurrent Access**  
   The system supports simultaneous querying and updating by multiple users. Transaction management ensures data consistency, while connection pooling optimizes resource utilization.
3. **Location Independence**  
   Users can access data without knowing its physical location. The system handles query routing transparently, retrieving data from the appropriate database nodes.

## System Characteristics

The system embodies several key properties of distributed systems:

1. **Transparency**
   * **Location Transparency**: Users are not aware of the physical storage location of data.
   * **Replication Transparency**: Data replication is automatic and invisible to users.
   * **Concurrency Transparency**: Multiple users can access the system simultaneously without conflicts.
2. **Reliability**  
   The system ensures fault tolerance through data replication and implements transaction management to maintain consistency. Error handling and recovery mechanisms further enhance reliability.
3. **Scalability**  
   The architecture supports horizontal scaling, allowing the system to handle growing data volumes and user demands efficiently. Distributed processing optimizes performance under increasing loads.

# Implementation Details

## Data Model

The database schema is designed to store and manage comprehensive restaurant information, including:

* Restaurant name.
* Location (street adress).
* Rating and review metrics.
* Price range.
* Contact information.
* Trip Advisor webpage URL.
* Menu Link (Optional).

## Key Features

1. **Search Capabilities**  
   Users can search for restaurants by city, filter results by food type, and sort them based on various criteria such as ratings and price. Integration with a URL shortening service ensures efficient link sharing.
2. **Data Management**  
   The system supports adding and updating reviews, validating and cleaning input data, and automating tasks like ID generation and price range standardization.
3. **User Interface**  
   An interactive console-based interface allows users to query the database, validate inputs, and receive formatted, easy-to-read results. Error handling and feedback mechanisms improve usability.

# Technical Considerations

## Benefits of TimescaleDB

The choice of TimescaleDB offers significant advantages:

* **Time-Series Functionality**: Efficient storage and querying of temporal data, along with automated partitioning.
* **PostgreSQL Compatibility**: Access to PostgreSQL’s full feature set and ecosystem of tools.
* **Scalability**: Native support for distributed architectures and performance optimization features.

## System Extensibility

The system is designed for future enhancements, including:

* Restaurants from more countries around the World
* Advanced search and filtering options.
* Geographic search capabilities and analytics functions.
* Extending the interface to include web and mobile platforms.

# Performance Considerations

## Query Optimization

The system employs several strategies to optimize query performance:

* **Database Level**: Specialized indexing, query optimization, and connection pooling.
* **Application Level**: Prepared statements, batch processing, and efficient result set management.

## Scalability Testing

Performance testing evaluates the system’s ability to handle:

* Concurrent user access.
* Increasing data volumes.
* Query response times.
* Efficient resource utilization.

# Future Recommendations

To further improve the system’s capabilities:

1. **Infrastructure Enhancements**
   * Implement a caching layer for faster query responses.
   * Introduce a load balancer to manage traffic.
   * Develop a monitoring and logging system to track performance.
2. **Feature Expansion**
   * Enable real-time review updates.
   * Add advanced analytics and reporting capabilities.
3. **Interface Development**
   * Create a web-based interface for broader accessibility.
   * Develop mobile applications for on-the-go usage.
   * Provide API documentation for third-party integration.

# Conclusion

The Restaurant Recommendation System demonstrates the successful integration of distributed system principles with the advanced capabilities of TimescaleDB. By leveraging time-series optimization, scalability, and fault tolerance, the system delivers a robust and efficient platform for restaurant data management and recommendations. With a solid foundation in place, future enhancements can further extend its features and scalability, ensuring the system remains adaptable to evolving requirements.

# Demo of the Project

The demo of the project is available on YouTube at the following link:  
  
https://youtu.be/ikh8LREbzPs