Hotel Reservation Management System

Database Final Project

Hotel Reservation System

A Comprehensive Database Project with additional Web Application

- Developed using PostgreSQL for the backend database
- Streamlit for the frontend user interface
- Focus on efficient reservation management, data integrity, and reporting

This project demonstrates a full-stack solution for hotel operations, meeting all requirements for database design, normalization, and application integration.

Project Overview

Project Description

- Implements a hotel reservation database in PostgreSQL
- Manages guests, rooms, reservations, payments, and audits
- Supports key operations: booking, cancellation, reporting
- Normalized to 3NF to ensure data integrity and efficiency
- Includes a Streamlit web app for interactive management

Key Components:

- SQL scripts for schema, data, queries, transactions, indexes, functions, and triggers
- Documentation including backup/restore guide and ER diagram
- Frontend app for real-time interactions

This system simulates a real-world hotel management tool, ensuring scalability and reliability.

Objectives and Requirements Met

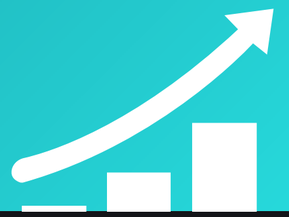
Main Objectives:

- Design a normalized database schema for hotel reservations
- Implement business logic via functions and triggers (e.g., prevent double bookings, auto-update room status)
- Provide basic and advanced SQL queries for operations and analytics
- Develop a user-friendly frontend for end-users
- Ensure data security through transactions and audits

How Requirements Are Met:

- 3NF Normalization: Eliminated redundancies in tables like guests, rooms, and reservations
- ACID Compliance: Demonstrated via transaction examples
- Performance Optimization: Indexes on frequently queried fields
- Reporting: Views and advanced queries for occupancy, revenue, and guest history

This addresses all project guidelines, including real-time updates and error handling

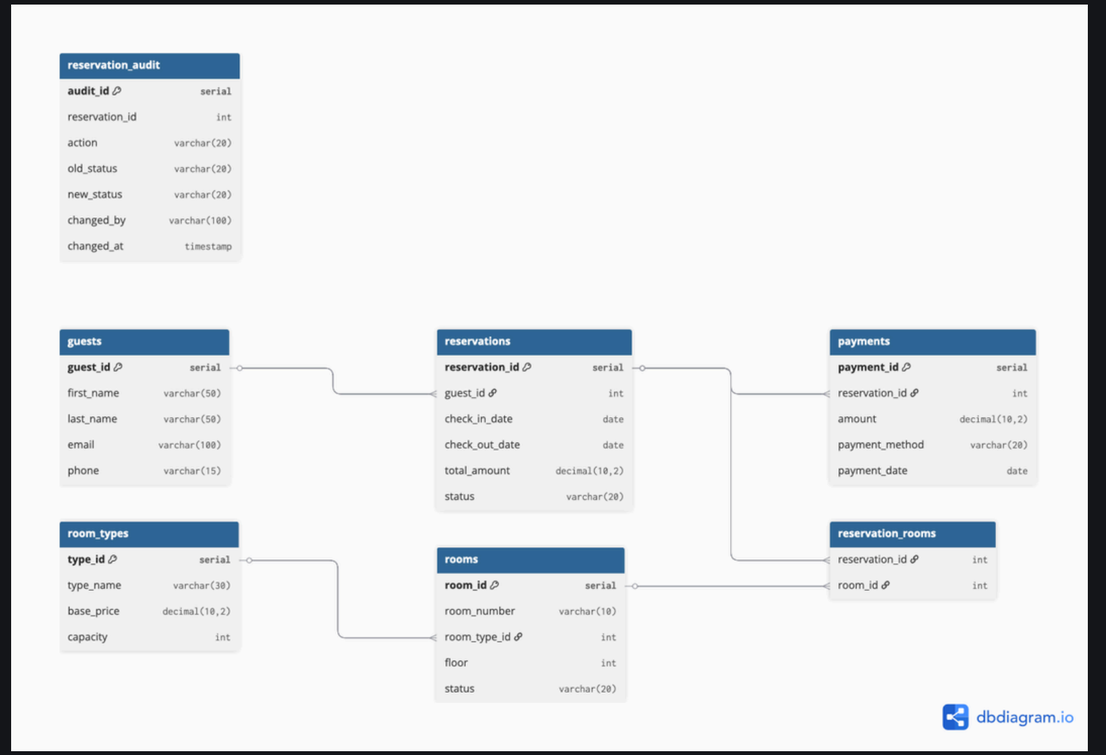


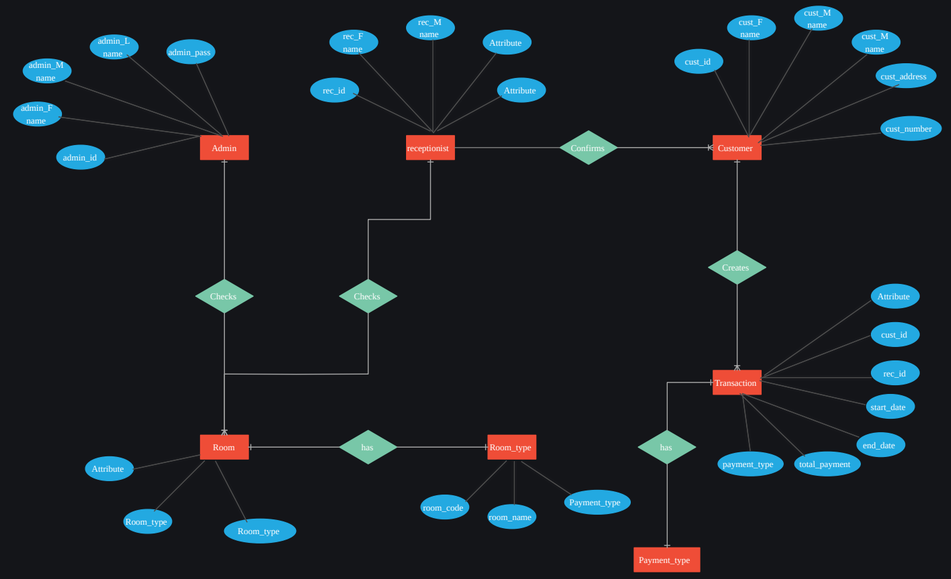
Database Design

ER Diagram and Schema

- Core Entities: Guests, Room Types, Rooms, Reservations, Reservation Rooms (junction), Payments, Reservation Audit
- Relationships:
  - One-to-Many: Guest to Reservations, Room Type to Rooms
  - Many-to-Many: Reservations to Rooms (via junction table)
- Constraints: Primary/Foreign Keys, Checks (e.g., check_out > check_in), Uniques (e.g., email, room_number)

The schema is defined in 01-create-tables.sql, ensuring referential integrity.





OVERVIEW OF IMPLEMENTATION

01-create-tables.sql: Defines tables, constraints, relationships 02-insert-data.sql: Populates with realistic sample data (15 guests, room types, rooms, reservations)

03-basic-queries.sql: CRUD operations and simple joins 04-advanced-queries.sql: Complex analytics (CTEs, window functions) 05-transactions.sql: Multi-step transactions with COMMIT/ROLLBACK 06-indexes.sql: Performance indexes + views + EXPLAIN ANALYZE

07-functions-triggers.sql: Business logic (availability check, prevent double-booking, audit logging)

Additional: app.py (Streamlit frontend) for user-friendly interface

TRANSACTIONS & INDEXING

Transactions:

- Complete booking: Insert reservation, link rooms, update status, record payment (atomic)
- Cancellation with rollback and savepoint examples
- Demonstrates ACID properties for reliability

Indexing:

- Indexes on frequently searched columns (e.g., email, dates, status)
- EXPLAIN ANALYZE shows significant performance gains
- Views for common reports (available rooms, current reservations, guest history)

Triggers/Functions: Prevent double bookings, auto-update room status, log changes

BACKUP AND RECOVERY STRATEGY

- Documented in docs/08-backup-restore.md
- Uses pg_dump for logical backups:

```
pg_dump hotel_reservation_db > backup.sql
```

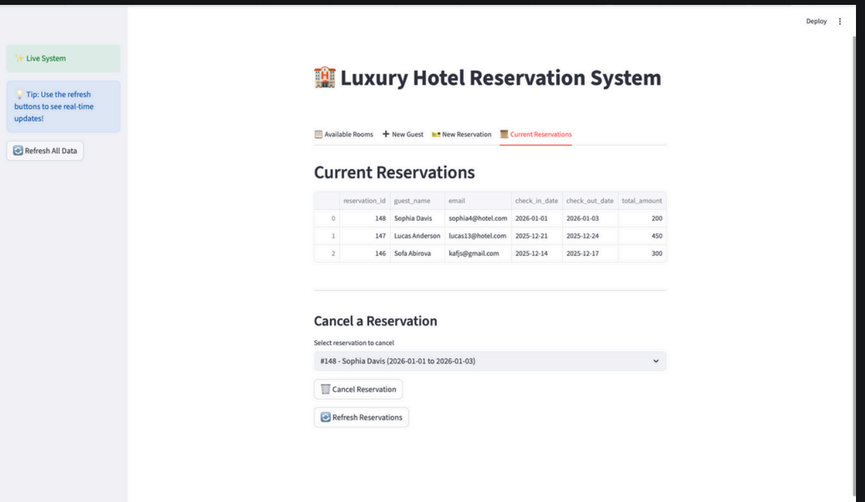
- Restore with pg_restore or psql
- Supports full database, schema-only, or data-only backups
- Recommendations: Scheduled dumps, compression, off-site storage

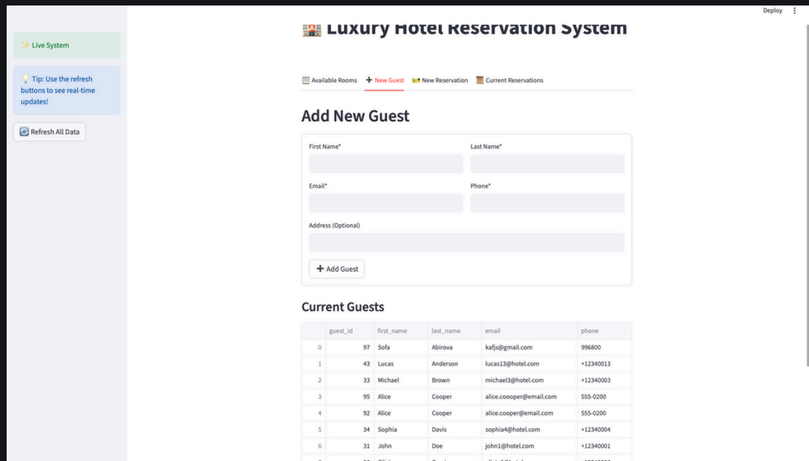
Ensures data recovery in case of failure.

WEB APPLICATION

- Built with Streamlit (app.py)
- Tabs: Available Rooms, Add Guest, New Reservation (with date-based search), Current Reservations (with cancel)
- Real-time updates, form validation, balloons for success
- Connects directly to PostgreSQL for live data

Provides a practical, user-friendly interface beyond pure SQL.





thank you!