Enhancing Operational Efficiency and Customer Satisfaction for Dream House NYC

Group 2: Jiarong Cui, Qi Xia, Xinyue Zhang, Xinyu Zhao

Problem statement

In today's real estate market, companies face intense competition and rapid changes. Effective data management is essential for maintaining a competitive edge. Dream House NYC (DHN), our client, manages extensive building, client, and transaction data across New York, New Jersey, and Connecticut. However, their current decentralized and manual systems result in frequent errors, inefficiencies, slow information processing, integration difficulties, low customer satisfaction, and a lack of data-driven decision-making.

To address these issues and enhance both internal efficiency and customer satisfaction, we propose developing an integrated, modernized database system. Our new database design will be divided into three main sections: building, customer, and company. Each section will include detailed subcategories, such as building owner information, customer transaction records, company employee, and operational cost data.

This unified database structure will ensure data consistency and integrity by correlating building, customer, and company information, thereby eliminating redundancies and inconsistencies. This improvement will enhance data accuracy and reliability while reducing errors and inaccuracies caused by manual inputs. Consequently, employees can focus more on customer service and other critical operations. Streamlined data management will enable quicker access to comprehensive customer and property information, allowing for more personalized and efficient customer service. Additionally, transaction statuses will be maintained in real-time, providing customers with the most current information on their transactions. Most importantly, insights generated through our database will help Dream House NYC to adjust their business strategy.

Proposal

Our database design phase began with a thorough market and technical study to accurately capture the specific needs of Dream House NYC (DHN) and its management team. This step was crucial for designing a database structure tailored to the company's business processes. Based on our analysis, we developed a comprehensive data model and an ER diagram, detailing the structure of tables, fields, data types, and their relationships.We implemented the database using

PostgreSQL, creating the necessary tables and establishing relationships. We populated the database with test data using Python to ensure functionality and accuracy. To facilitate a smooth transition, we plan to provide extensive training for DHN employees, ensuring they are proficient in using the new system.

These meticulously designed solutions, backed by thorough research, will significantly enhance DHN's internal operational efficiency and customer service, leading to increased customer satisfaction and a stronger competitive position in the market.

Team Structure and Timeline

Our team consists of four members: Cui, Xia, Zhang, and Zhao. We collaborated at every step, from understanding the company's and clients' needs, data collection, designing the data framework, to implementing the database.

Phase 1: Data Collection and Data Framework Design (July 1 - July 10, 2024)

- In this initial phase, we aimed to understand and define the database requirements and design its structure. Our tasks included gathering datasets, assessing their quality, identifying key data points and relationships to be included in the database, designing the data model and framework, and creating the ER diagram. We also specified the fields, data types, and relationships for each table.

Phase 2: SQL Queries and ER Diagramming (July 17 - July 23, 2024)

- The focus of this phase was on developing SQL queries and ER diagrams. We wrote SQL scripts to create the necessary table structures and relationships, and we drew ER diagrams to visually represent these relationships. Additionally, we wrote and tested basic SQL queries to ensure that data could be inserted, updated, and queried correctly.

Phase 3: Data Insertion (using Python) (July 23 - July 30, 2024)

- During this phase, we performed the data import process to ensure that all necessary data was accurately stored in the new database. We validated and tested the imported data to ensure its consistency and completeness.

Phase 4: Reporting and Presentation (July 30 - August 6, 2024)

- In the final phase, we prepared a detailed project report that included the data model, implementation process, test results, and performance evaluation. We also created presentations to demonstrate the features and benefits of the database system.

Data Schema

As mentioned, we categorized our schema into three parts: building-related, customer-related, and company-related. To improve the performance of our client, Dream Homes NYC, we structured fifteen tables in 3NF to reflect the data typically encountered by a real estate company. These tables are tightly correlated, ensuring a comprehensive and efficient data management system.

Building-related (9):

building_information:

➢building_id: INT PRIMARY KEY

➢building_name: VARCHAR(100), the name of the building

➢state: VARCHAR(20), the state where the building is located

➢address: VARCHAR(225) , the address of the building

➢zipcode: VARCHAR(10), the zipcode of the building

➢leasing office: VARCHAR(100), the email of the leasing office

This table provides the basic information related to the building. All nine tables in the building category are directly linked to the building_information table through the building_id.

owner_information:

➢building_id: INT, both the primary key and foreign key

➢owner_name: VARCHAR(100), the name of the owner of the house

➢owner_contact: VARCHAR(100), the email of the owner

This table describes the basic information related to the owner of the building. Since buildings can have different types of ownership, such as a private house owner, a management company, or multiple owners through the selling process, it is essential to maintain an up-to-date dataset of all owner information. This facilitates salespeople and brokers in contacting the house owner and closing sales. To clarify, the difference between the owner and the leasing office in the building_information table lies in their roles in property management and ownership. The leasing office is primarily responsible for the rental process and may not hold ownership of the property. A building can have both a leasing office and an owner, or only one of them.

listing_information:

➢listing_id: INT PRIMARY KEY

➢building_id: INT, foreign key which links to building_information table

➢room_number: INT, the room/apt number

➢bedroom: INT, the number of bedrooms in a house

➢bathroom: INT, the number of bathrooms in a house

➢availability: VARCHAR(50), the available date of the house, or 'off market'

➢price: DECIMAL(10,2), the rental or selling price of the house

➢sqft: INT, the size of the room

This table provides information on houses available in the market. Unlike the building information, this table details individual rooms, potentially listing multiple rooms within the same building. The data in this table will change frequently depending on market sales. The price column reflects the current market price of each house.

house_history:

➢listing_id: INT, primary key and foreign key

➢customer_id: INT, foreign key linked to customer

➢start_date: DATE, the date that the customer start to live

➢end_date: DATE, the date that the customer end to live

➢price: DECIMAL(10,2), the rental/selling price the customer paid

This table details the residential history of a house. Linked to the customer_information table, it aims to identify if a house has become an exclusive listing for our company. If a series of consecutive occupants of a house are clients of our client's company, then this house can be considered an exclusive listing. The more exclusive listings Dream Homes NYC have, the better the sales performance will be. Additionally, the price column records price fluctuations for each sale, providing insights into market trends.

open_house_event:

➢event_id: SERIAL, primary key

➢building_id: INT, foreign key which links to building_information table

➢event_type: VARCHAR(100), the type of the event

➢event_date: DATE, the date of the date

➢event_time: VARCHAR(10), the start time of the event

➢capacity: INT, the capacity of the event

This table provides information on open houses and marketing events held by the buildings. These events support the selling process by allowing brokers to bring their customers. Additionally, the data can help determine which types of events are more effective.

amenities:

➢building_id: INT, primary key and foreign key

➢amenity_fee: DECIMAL(10,2), the annual fee of the amenity

➢gym: BOOLEAN, whether the building has a gym

➢swimming_pool: BOOLEAN, whether the building has a swimming

➢basketball_court: BOOLEAN, whether the building has a swimming

➢parking: BOOLEAN, whether the building has a parking place

➢lounge: BOOLEAN, whether the building has a lounge

This table provides information on the amenities available in the buildings. This data assists brokers in identifying key selling points for each house.

neighborhood:

➢building_id: INT, primary key and foreign key

➢school: INT, the amount of schools around the building

➢grocery_store: INT, the amount of grocery stores around the building

➢pharmacy: INT, the amount of pharmacies around the building

➢park: INT, the amount of parks around the building

➢restaurant: INT, the amount of restaurants around the building

➢bar: INT, the amount of bars around the building

This table depicts the convenience level of living in the neighborhood. It helps brokers find specific matches for consumers' needs.

school:

➢ building_id: INT, primary key and foreign key

➢ school_id: INT, primary key

➢ school_name: INT, the name of the school

➢ school_rating: INT, the rating of the school

➢ school_type: INT, kindergarten, primary/middle/high school, university

Since school is an important consideration when choosing a place to live, we created a dedicated table to gather school data. Some families with children may want to live close to a school, while others may prefer to live farther away. This table contains information about schools around the building, helping brokers find specific matches for consumers' needs.

demographic:

➢ building_id: INT, primary key and foreign key

➢ state: VARCHAR(50), NY/NJ/CT

➢ area: VARCHAR(100), the community area of each state, like Queens/Manhattan/Bronx in NY

➢ income_level: DECIMAL(3,1), the income level of each community area

➢ criminal_rate_level: DECIMAL(3,1), the criminal rate level of each community area

Demographics are important considerations for consumers and significantly impact house pricing. Therefore, we created this table to monitor market trends and assist brokers in finding specific matches for consumers' needs.

Customer-related (2):

customer_information:

➢cutomer_id: INT PRIMARY KEY

➢first_name: VARCHAR(50), the first name of customer

➢last_name: VARCHAR(50), the last name of customer

➢contact: VARCHAR(15) , the phone number of the customer

➢email: VARCHAR(100), the email of the customer

➢employee_id: INT, the broker who take charge of the customer

This table plays an important role in customer relationship management. It assists in connecting customers with broker activities.

transaction:

➢transaction_id: INT PRIMARY KEY

➢customer_id: INT, foreign key which links to customer_information table

➢building_id: INT, foreign key which links to building_information table

➢transaction_date: DATE , the date of the transaction

➢transaction_type: VARCHAR(50), the type of the transaction, rental or selling

➢transaction_amount: DECIMAL (10,2), the broker who take charge of the customer

This table records transactions from the customer side, meaning the money customers paid to the company (company revenue). It helps determine whether a customer is a repeat customer, supports calculations for the financial team, and tracks the performance of brokers. If the customer_id appear multiple times, then he/she can be considered a repeat customer.

Company-related (4):

employees:

➢employee_id: INT PRIMARY KEY

➢first_name: VARCHAR(50), the first name of employee

➢last_name: VARCHAR(50), the last name of employee

➢department: VARCHAR(50), the department of employee

➢position: VARCHAR(100), the position of employee

➢contact_info: VARCHAR(15), the phone number of employee

This table records all employee information and is crucial for linking with other employee-related tables.

agent_customer:

➢ employee_id: INT PRIMARY KEY

➢ customer_id: INT, foreign key which links to customer_information table

➢ interested_listing_id: INT, foreign key which links to building_information table

This table demonstrates the relationship between employees and customers. It records which customers each employee is responsible for and the listings that interest the customers. This information helps sales managers supervise the performance of their brokers, thereby enhancing overall team performance.

offices:

➢ office_id: INT PRIMARY KEY

➢ state: VARCHAR(50), NY/NJ/CT

➢ address: VARCHAR(255), the address of the offices

This table aims to provide detailed information about the three offices. It helps analyze and compare the performance of each office, thereby identifying market trends.

operational_cost:

➢ cost_id: INT PRIMARY KEY

➢ office_id: INT, foreign key which links to offices table

➢ cost_type: VARCHAR(50), the type of the cost, such as office supplies/rent

➢cost_amount: DECIMAL(10,2), the amount of each cost

➢transaction_date: DATE, date of the purchase

➢employee_id: INT, foreign key which links to employees table

This table records transactions related to company operations, representing the company's costs. It details who made the purchase, what was purchased, the date, and the purpose. This information helps manage costs and calculate financial metrics, playing a crucial role in building sustainable and healthy financial performance for the company.

Database Design Process:

In the Dream House project, our team prioritized establishing a comprehensive database structure before generating or collecting data. This approach ensured efficiency and effectiveness in organizing and retrieving relevant information.

Once the database structure was defined, we focused on establishing relationships between the tables to ensure efficient data access and usability for both customers and employees.

Building Relationships:

- The "Building Information" table serves as the primary table, linked with other building-related tables through "building_id." This includes tables for "Demographic," "Amenities," "Neighborhood," "Owner Information," "School," and "Open House Event."
- The "Listing Information" table connects to "Building Information" through "building_id," providing details about each unit's availability, size, and pricing.

Customer Relationships:

- The "Customer Information" table is linked to the "Agent-Customer" table via "customer_id," recording which agents are assigned to each customer and their interested listings.
- The "House History" table, connected through "listing_id" and "customer_id," logs the rental history of each customer.
- The "Transactions" table connects to both "Customer Information" and "Building Information" through "customer_id" and "building_id," detailing each transaction's type, amount, and status.

Company Relationships:

- The "Employee Information" table is linked to the "Agent-Customer" table via "employee_id," detailing which agents manage which customers.
- The "Operational Cost" table connects to "Office Information" and "Employee Information" through "office_id" and "employee_id," tracking expenses associated with specific employees and offices.

Benefits for Clients:

- **Employees**: The relational database design allows employees to efficiently access and manage detailed building and customer information. They can identify popular properties, strategize marketing efforts, and track transaction statuses. The integrated data helps employees optimize resource allocation, enhance customer service, and improve operational efficiency.
- **Executives:** This information helps senior managers and executives assess the company's performance. By comparing the statistics of the three offices, they can make data-driven strategic decisions. The employee data aids in revising the reward system, while the housing and customer data helps in identifying market trends.

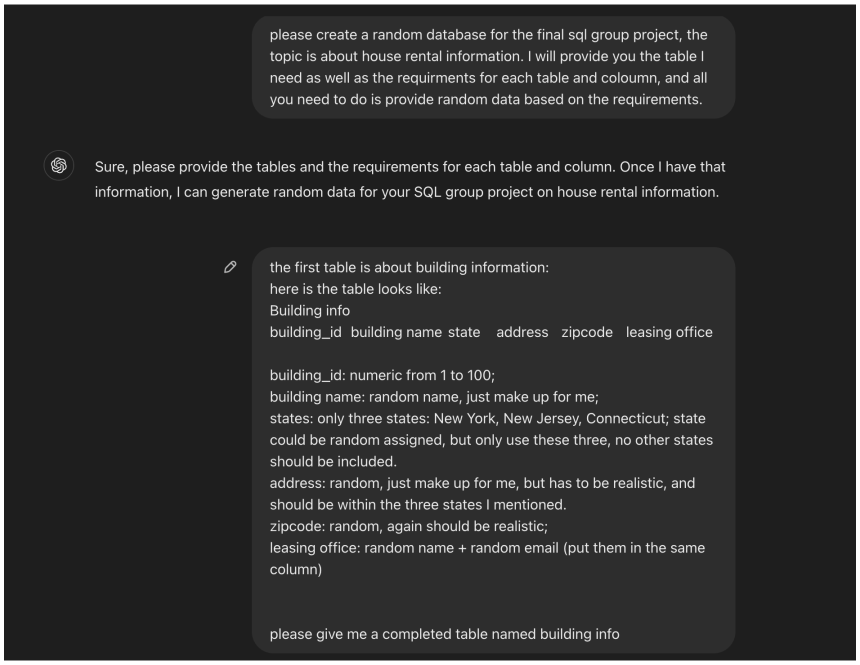
By establishing these relationships, our database design ensures that users can navigate and retrieve relevant information easily, enhancing overall user experience and decision-making processes.

ETL Process

Data Extraction

To populate our database, we needed to extract relevant data. Ideally, this data would be sourced from real-world records available on official websites, such as state government and building websites. However, given the project's scope and the difficulty in obtaining comprehensive real-world data, we opted to use OpenAI to generate the required data.

Due to OpenAI's limitations in generating perfectly structured data without human intervention, we decided to generate data for each table individually. This method, though time-consuming, ensured the highest data quality and consistency across all tables. By carefully crafting prompts to define how each column should be populated (e.g., building_id as serial numbers from 1 to 100, customer_id as 5-digit random numbers, transaction dates between 2020 and 2024, and room prices based on room attributes), we guided OpenAI to produce the desired datasets. (pic 1) We generated 15 interrelated tables and rigorously reviewed the data for accuracy and relevance. If discrepancies were found, we refined our prompts and regenerated the data. Special care was taken to ensure relational integrity; for example, all employee_id entries in the agent_customer table were restricted to sales department employees, and all IDs across tables were cross-verified for consistency.



please create a random database for the final sql group project, the topic is about house rental information. I will provide you the table I need as well as the requirments for each table and coloumn, and all you need to do is provide random data based on the requirements.

Sure, please provide the tables and the requirements for each table and column. Once I have that information, I can generate random data for your SQL group project on house rental information.

the first table is about building information:
here is the table looks like:
Building info
building_id building name state address zipcode leasing office

building_id: numeric from 1 to 100;
building name: random name, just make up for me;
states: only three states: New York, New Jersey, Connecticut; state could be random assigned, but only use these three, no other states should be included.
address: random, just make up for me, but has to be realistic, and should be within the three states I mentioned.
zipcode: random, again should be realistic;
leasing office: random name + random email (put them in the same column)

please give me a completed table named building info

Pic 1. Screenshot of OpenAI Prompt

Data Transformation

The next phase involved transforming the data to prepare it for analysis and integration into PostgreSQL. We utilized Python for this task due to its powerful data manipulation libraries and ease of use.

First, we installed and imported the necessary packages: pandas for data manipulation and psycopg2 for PostgreSQL interaction. The pandas library provided robust tools for data cleaning, transformation, and analysis through DataFrames, while psycopg2 allowed seamless connection and operations with the PostgreSQL database. Our data transformation process mirrored the methodology we practiced in Homework 6. The steps included:

- Installing and Importing Packages:
- Loading Data:
    - We iteratively loaded the 15 tables into pandas DataFrames.
- Data Cleaning and Transformation:
    - Before loading the data into PostgreSQL, we ensured it was clean and formatted correctly. This involved checking for missing values, ensuring data types were correct, and making any necessary transformations.
- Connecting to PostgreSQL:

Data Loading

To load the data into PostgreSQL, we executed the following steps:

- Creating Tables in PostgreSQL:
    - Using the provided SQL script, we created the necessary tables in the PostgreSQL database. Here's an example of creating the building_information table:
- Inserting Data into PostgreSQL:
    - We iterated through each DataFrame and inserted the data into the corresponding PostgreSQL table. Here's an example for the building_information table:
    - We also add duplicate handling code to prevent errors caused by duplications.
- Ensuring Data Integrity:
    - Throughout the data loading process, we monitored for errors and inconsistencies. This included verifying foreign key relationships and ensuring all references between tables were correctly maintained.

By meticulously following these steps, we successfully transformed and loaded our generated data into the PostgreSQL database, ensuring a robust and reliable foundation for subsequent analysis.

Analytical Procedures and Insights

In this section, our group outlined twelve key analytical procedures that provided valuable insights for our client. Each procedure is supported by its corresponding codes and highlights the importance and benefits for database users.

- Average Price per Square Foot by State
    - This query calculates the average price per square foot for buildings in each state. By analyzing this data, users can identify which states have higher or lower property costs, enabling strategic decisions regarding where to focus marketing efforts, investment, and development. For instance, a state with a lower average price per square foot might be targeted for budget-friendly housing projects, whereas states with higher averages could be promoted as premium locations.
    - Customers can use this information to understand the cost landscape of different states, aiding in decision-making for property investments. Data analysts can integrate this data into reports to advise clients on profitable investment regions.
- Popular Buildings for Each State
    - This query ranks the top five buildings with the highest sales in each state. Understanding which buildings generate the most revenue can help prioritize marketing and sales efforts. For example, resources can be allocated to advertise these top-performing buildings more aggressively or replicate their success strategies in other properties. Additionally, it provides insights into customer preferences, guiding future development projects.
    - Customers can gain insights into popular and high-performing properties, which may indicate higher resale value and demand. Sales teams can focus on these properties for targeted marketing and promotions.
- Ranking of Annual Amenity Fee
    - This query ranks buildings based on their annual amenity fees. By knowing which buildings have the highest amenity fees, users can understand the added value perceived by tenants. This can guide decisions on what amenities to invest in and promote, balancing the cost of providing amenities with the potential increase in rental income and tenant satisfaction.
    - Customers can gain insights into popular and high-performing properties, which may indicate higher resale value and demand. Sales teams can focus on these properties for targeted marketing and promotions.
- Ranking of Average School Rating Around Each Building
    - This query compares the average crime rates among states. Lower crime rates are a significant factor for potential renters or buyers, as safety is a top priority. Highlighting properties in states with lower crime rates can attract more tenants and justify higher rental prices. Conversely, properties in higher crime areas may need additional security measures or discounted pricing to attract tenants.

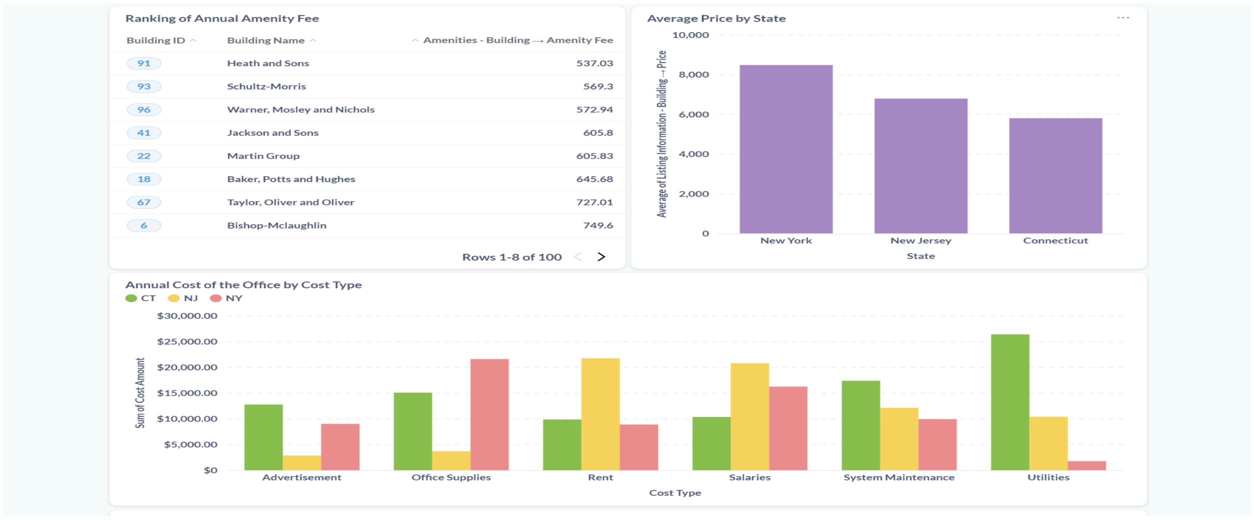
    - Customers can make informed decisions based on safety and crime statistics. Property managers can adjust security measures and pricing strategies accordingly.

- Average Income Level by State
    - This query calculates the average income level in each state. Higher income levels may indicate a market for premium housing, while lower levels could suggest a need for affordable housing options. This insight helps in tailoring property offerings and marketing messages to different economic segments, ensuring alignment with the financial capabilities of potential tenants.
    - Customers can evaluate the cost of amenities relative to the rental price, aiding in budgeting decisions. Property managers can use this data to adjust amenity offerings and fees to maximize tenant satisfaction and profitability.
- Crime Rate Comparison Among States
    - This query compares the average crime rates among states. Lower crime rates are a significant factor for potential renters or buyers, as safety is a top priority. Highlighting properties in states with lower crime rates can attract more tenants and justify higher rental prices. Conversely, properties in higher crime areas may need additional security measures or discounted pricing to attract tenants.
    - Customers can make informed decisions based on safety and crime statistics. Property managers can adjust security measures and pricing strategies accordingly.
- Ranking of Customer Retention Rate
    - This query ranks customers based on their rental frequency, identifying those who have rented properties multiple times. Customers with higher rental frequencies are likely to be more loyal and may be potential candidates for future leases. This insight helps in identifying valuable customers and tailoring retention strategies to encourage repeat business.
    - Customers who are identified as high-potential for future leases can be offered special promotions or loyalty rewards to retain their business. Sales and customer service teams can focus their efforts on nurturing relationships with these valuable customers.
- Employee Performance Metrics
    - This query evaluates employee performance by the number of customers served and total sales generated. Recognizing top performers allows for better resource allocation, targeted training, and incentive programs. It also helps identify employees who may need additional support or training to improve their performance.

    - Customers can ensure they are working with top-performing agents for better service. Management can reward high performers and provide support where needed, optimizing workforce efficiency.
- Total Cost of Each Office
    - This query calculates the total operational cost for each office. Monitoring operational expenses is crucial for financial management, allowing users to identify high-cost offices and implement cost-saving measures. Understanding where money is spent can help streamline operations and improve overall profitability.
    - Customers benefit indirectly through more efficiently managed offices, potentially leading to better service and lower costs. Financial teams can focus on cost control measures and enhance overall financial health.
- Profit of Each Office
    - This query calculates the profit of each office by subtracting operational costs from total revenue. Understanding profitability at the office level is essential for strategic planning, allowing users to identify high-performing offices and replicate their success. It also highlights underperforming offices that may need operational changes or support to improve their financial outcomes.
    - Customers benefit indirectly through more efficiently managed offices, potentially leading to better service and lower costs. Strategic planners can use this data to make informed decisions about resource allocation and performance improvements.

Metabase Dashboard

To visualize the database and help users to gain insights, we generated a metabase dashboard.


Ranking of Annual Amenity Fee

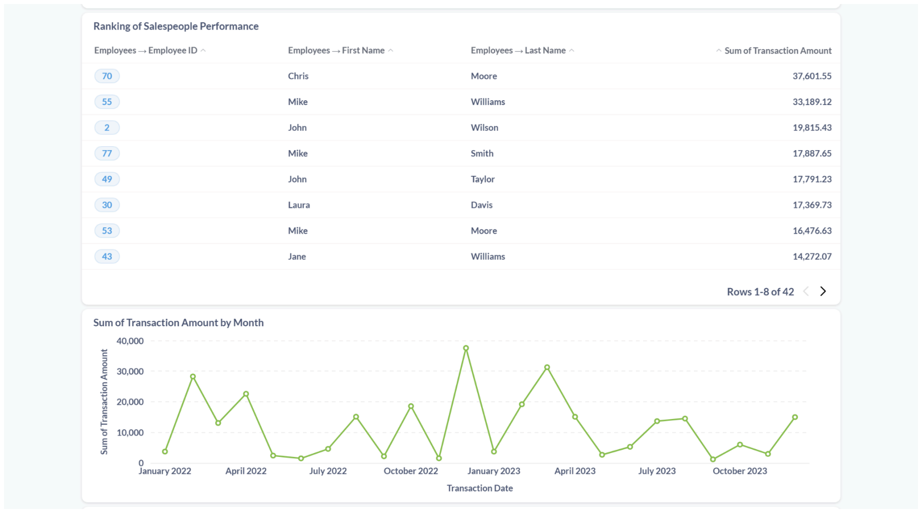
Building ID	Building Name	Amenities - Building → Amenity Fee
91	Heath and Sons	537.03
93	Schultz-Morris	569.3
96	Warner, Mosley and Nichols	572.94
41	Jackson and Sons	605.8
22	Martin Group	605.83
18	Baker, Potts and Hughes	645.68
67	Taylor, Oliver and Oliver	727.01
6	Bishop-Mclaughlin	749.6

Rows 1-8 of 100

- Annual Amenity Fee Ranking

    - This table displays the building ID, name, and annual amenity fee, sorted in ascending order from $537 to $2998.

- Average Price by State
    - This bar chart illustrates the average price of buildings within different states. In New York, the average building price is above $8000, in New Jersey, it exceeds $6000, and in Connecticut, it ranges between $5000 and $6000.
- Annual Office Costs by Type
    - This bar chart compares the different costs associated with offices in various states. In New York, office supplies are particularly expensive. In New Jersey, the main expense is rent, while in Connecticut, the primary cost is utilities.



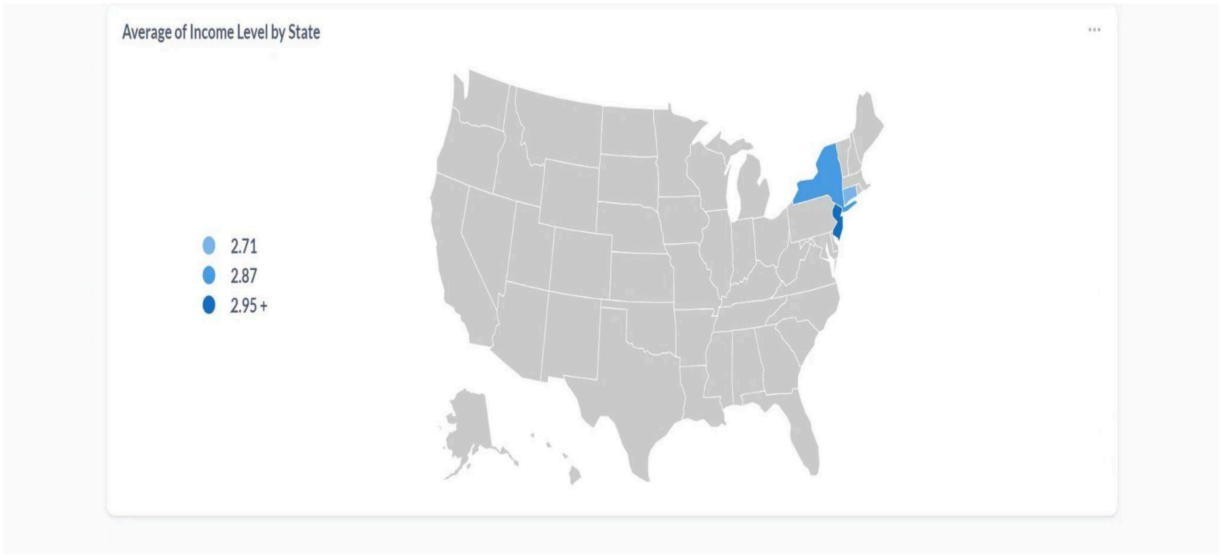
Ranking of Salespeople Performance

Employees → Employee ID	Employees → First Name	Employees → Last Name	Sum of Transaction Amount
70	Chris	Moore	37,601.55
55	Mike	Williams	33,189.12
2	John	Wilson	19,815.43
77	Mike	Smith	17,887.65
49	John	Taylor	17,791.23
30	Laura	Davis	17,369.73
53	Mike	Moore	16,476.63
43	Jane	Williams	14,272.07

Rows 1-8 of 42

Sum of Transaction Amount by Month

- Ranking of Salespeople Performance
    - This table ranks employees based on their sales performance, sorting their customers' transaction amounts in descending order. The results show that Chris Moore has the best performance.
- Monthly Transaction Amount
    - This chart visualizes the total transaction amounts by month. While no specific trend is observed, it is evident that transaction amounts are consistently low in January, increasing afterward. There is also a noticeable dip in July, likely due to the summer break.



- Average Income Level by State
  - This map clearly shows the average income levels in three different states. The colors, ranging from dark to light, represent average income levels from highest to lowest: $2.95 in New Jersey, $2.87 in New York, and $2.75 in Connecticut. (Here is sample data, which may conflict with the fact.)

Plan on How Customers Will Interact with the Database System

To ensure efficient and effective interaction with the database system we designed, our plan considers the needs of both analysts and C-level officers. Here's a detailed explanation:

Interaction for Analysts and C-level Officers:

For analysts, the system will be designed to support direct querying. Analysts will have access to a robust SQL interface and tools like Jupyter Notebooks, where they can run Python scripts. This setup will allow them to execute the analytical procedures we designed directly, providing the flexibility to explore and analyze data in real time. The implementation will include detailed documentation and pre-written SQL queries and Python scripts to facilitate ease of use.

For C-level officers, we will provide comprehensive, user-friendly reports. These reports will be generated using tools like Tableau or Metabase, which can connect to our database and present data visually through dashboards and charts. These tools will allow executives to review high-level insights and trends without needing to understand the underlying technical details.

Execution of Analytical Procedures:

Analysts will run the code for the analytical procedures through SQL clients or Python environments. They will be provided with scripts and queries that have been thoroughly tested and documented. The SQL queries will be executed directly against the database, and Python scripts will use libraries such as Pandas and Psycopg2 to interact with the PostgreSQL database, enabling complex data manipulations and transformations.

Reviewing Results for Executives:

Executives will review the results through dashboards and automated reports generated by business intelligence tools. These tools will pull data from the database and present it in a way that highlights key metrics and trends. This approach ensures that executives have access to up-to-date and actionable insights without needing to dive into the raw data.

Tools and Programming Languages Implemented:

We implemented several tools and programming languages, including:

- SQL for database querying and management.
- Python for data manipulation, transformation, and analysis.
- Tableau or Metabase for creating interactive dashboards and reports.
- PostgreSQL as the database management system.

Using programming languages like Python and SQL for database actions offers several benefits. It allows for complex data manipulations and analyses, automation of repetitive tasks, and integration with various data sources. Programming languages also provide powerful libraries and frameworks that enhance data processing capabilities and ensure accuracy and efficiency in handling large datasets.

Interaction for Non-Technical Personnel:

Non-technical personnel will not interact with the database. Since the database contains personal information, such as addresses, phone numbers, email addresses, and transaction details, we aim to maintain a high level of security. Therefore, only technical employees will have access to the database, while the IT department manager will have the authority to modify the information. Customers, employees from other departments, and senior executives will not have access to or be able to view the database. Instead, they will receive insights concluded by data analysts.

Planning for Redundancy and Performance:

We planned for redundancy and performance by implementing best practices in database design and management. This includes using indexing, partitioning, and optimizing queries to ensure fast retrieval times. For redundancy, we considered using replication and backup strategies to protect data integrity and availability. Depending on the client's needs, hosting on the cloud can offer additional benefits such as scalability, cost-efficiency, and built-in redundancy. However, if the client prefers greater control over their data, an on-premises solution might be more suitable.

Overall, our plan ensures that both analysts and executives can effectively interact with the database system, leveraging the tools and technologies that best suit their needs while maintaining high performance and reliability.

Conclusion

In conclusion, our project aimed to address the operational inefficiencies and data management challenges faced by Dream House NYC (DHN) through the development of an integrated and modernized database system. By conducting a thorough market and technical study, we designed a robust database structure divided into three main sections: building, customer, and company. This structure ensures data consistency, integrity, and reliability, which are crucial for accurate and efficient operations.

Throughout the project, our team collaborated closely, from initial data collection and framework design to the implementation and testing phases. We utilized PostgreSQL for database creation and Python for data insertion, ensuring the system's functionality and accuracy. Extensive training plans for DHN employees were also developed to facilitate smooth adoption and proficient use of the new system.

Our comprehensive and well-researched solutions will significantly enhance DHN's internal operational efficiency and customer service. This improvement will lead to increased customer satisfaction and a stronger competitive position in the real estate market.

By streamlining data management and providing real-time access to comprehensive information, our database system empowers DHN to make data-driven decisions, optimize resource allocation, and deliver personalized and efficient services to their clients. The successful implementation of this project demonstrates the potential for technological advancements to drive business growth and operational excellence.

Appendix

Links：

Github：

https://github.com/azhangxy2000/apan5310_final_group2.git

ER-Diagram：
https://lucid.app/lucidchart/6e1e2d70-8ede-49cf-9381-a67c7805e5dc/edit?page=0_0&invitationId=inv_cdacf5f3-1423-40b9-ae02-157072b81552#

Metabase Dashboard：

http://localhost:3000/public/dashboard/791c27c0-daa8-4354-8437-cb4e88da214e