Luxury Property Management System

Samy BOUAISSA Amine M'ZALI

EFREI Paris

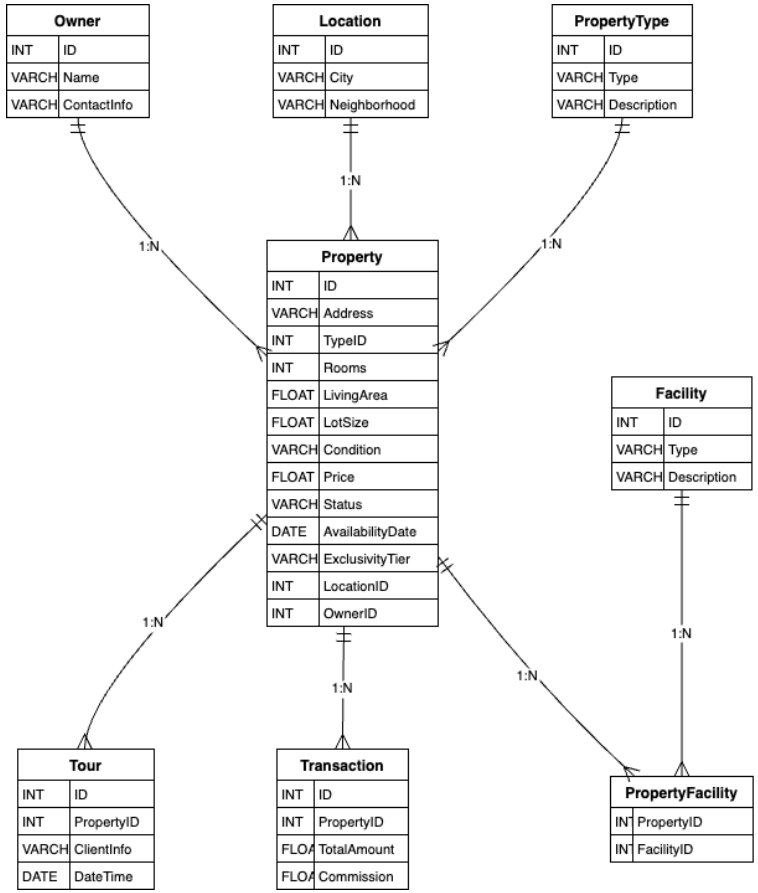
16 janvier 2025

Introduction

- Development of a database system for luxury property management.
- Combines relational and NoSQL database technologies.
- Focus areas :
  - Entity-Relationship modeling.
  - Normalization.
  - PL/SQL procedures.
  - Object-relational features.
  - NoSQL databases (MongoDB, Neo4J).

Entity-Relationship Diagram

- Represents main entities :
  - `Property`, `Owner`, `Transaction`, `Tour`, `Facility`.
- Captures relationships and attributes.



Logical Model

- Translation of E/R diagram into relational schema.
- Key tables :
  - `Property`, `Owner`, `Transaction`, `Tour`, `Facility`.
- Relationships represented using primary and foreign keys.

Normalization

- **1NF** : Atomic values ensured (e.g., `Property` table has no multi-valued attributes).
- **2NF** : No partial dependencies (e.g., non-key attributes fully dependent on primary keys).
- **3NF** : No transitive dependencies (e.g., separated `PropertyCondition` table).
- **BCNF** : All functional dependencies resolve to candidate keys.
- **4NF** : Multi-valued dependencies managed (e.g., `PropertyFacility` table).
- **5NF** : Join dependencies resolved (e.g., `Tour` links properties and clients).

PL/SQL Procedures

- Automation with PL/SQL procedures :
  - `validate_property_price` : Ensures price €10,000.
  - `check_tour_conflict` : Prevents scheduling conflicts.
  - `update_property_status` : Marks properties as sold.
  - `calculate_commission` : Computes dynamic commissions.
  - `validate_exclusive_tour` : Validates VIP tours for premium properties.
- Packaged in `LuxuryPropertyUtils`.

Why Procedures Instead of Triggers ?

Why Procedures ?

- **Explicit Invocation :** Procedures run only when called, giving full control over execution.
- **Easier Debugging :** They can be tested independently with defined parameters.
- **Better Performance :** No overhead from automatically firing triggers on every table event.
- **Reusability :** Code can be invoked from multiple parts of the application.
- **Predictable Flow :** Logic is clearer than event-driven triggers, which may conflict or cascade.

Object-Relational Database (ORD)

- Enhanced data modeling with custom types :
  - `AddressType`, `PropertyType`.
- Hierarchical relationships using inheritance :
  - `Mansion`, `Apartment` extend `BaseProperty`.
- Improved flexibility and maintainability.

NoSQL Databases

- **MongoDB :** Manages document-based data.
  - Example : Multimedia property descriptions, client feedback.
- **Neo4J :** Analyzes graph-based relationships.
  - Example : Client-property interactions, referral networks.

Conclusion

- Combined relational and NoSQL databases for a robust solution.
- Ensured data integrity with normalization and constraints.
- Automated critical operations with PL/SQL.
- Enhanced data representation with object-relational features.
- Flexible and scalable handling of unstructured data using NoSQL.