Habyt Rental System

Introduction:

Habyt Rental System is designed to facilitate the efficient management of property rental. This system enables property managers to streamline the rental process, from listing available units to managing applicants and handling financial transactions. The system is built on a normalized schema, using Docker for containers and Python for data processing and manipulation.

Schema:

Property:

At the core of this schema lies the concept of a property, representing the physical location where rental units are situated. Each property is uniquely identified by a `propertyId` and is associated with an `address`, providing essential location information. Properties serve as the foundation upon which units are built, forming the backbone of Habyt Rental System.

Unit:

Units are the individual rentable spaces within a property, encompassing both co-living rooms and traditional apartments. Each unit is assigned a `unitId` for identification purposes and is characterized by its `occupancyType`, which can either be shared (for co-living) or private (for traditional apartments). Units are linked to their respective properties through the propertyId attribute, ensuring a clear association between physical space and its location. Additionally, units may possess attributes such as `address` and `roomNumber`, offering further details about their specific characteristics within a property.

Applicant and Co-Applicant:

An essential aspect of the rental process is the presence of applicants, individuals seeking to rent a unit within a property. Applicants are identified by a unique `applicantId` and are directly linked to the unit they are applying for through the `unitId` attribute. Additionally, applicants may have co-applicants associated with them, reflecting situations where multiple individuals apply for the same unit together.

Co-applicants are connected to their corresponding applicants through the applicantId attribute, establishing a hierarchical relationship within the rental application process.

Price, Concession, and Fee:

To facilitate transparent pricing and financial transactions, this schema incorporates entities for price, concession, and fee management.

Prices represent the cost of renting a unit for a specified duration and are tied to the unit through the `unitId` attribute. Concessions, on the other hand, denote any discounts or special offers applicable to a unit's rental price, providing flexibility in pricing structures. Fees encompass additional charges associated with renting a unit, covering expenses such as screening fees. Both concessions and fees are linked to their respective units via the `unitId` attribute, allowing for seamless integration into the rental transaction process.

Schema Objective:

By carefully structuring these entities and their relationships, this schema establishes a robust foundation for managing rental units effectively. From properties and units to applicants and financial considerations, each component plays a vital role in facilitating the rental process, ensuring clarity, transparency, and efficiency for both landlords and tenants alike.

For more information about the relational schema, see relational_schema.png.

What about scalability? What if Habyt starts renting to businesses?

Business Entity:

We'll add a new entity to handle businesses seeking rental spaces, likely including `businessId`, name, industry type, and contact info. Businesses interact differently from individuals, so we'll tailor data fields to capture their unique requirements and preferences.

Unit Classification:

To accommodate business rentals, we may need to classify units further based on suitability for commercial purposes. Some units may be tagged as "commercial spaces" or "office suites," each with specific attributes and rental considerations for business tenants.

Lease Agreements:

Business rentals often involve complex lease agreements. Our schema should handle the creation and management of commercial lease agreements, covering lease duration, rental rates, renewal options, and commercial-specific terms and conditions.

Financial Considerations:

With business rentals, pricing, concession, and fee structures may need adjustments. Businesses have different budgeting constraints and negotiation tactics, requiring flexibility in pricing models and payment terms.

Tenant Screening:

While tenant screening remains relevant, evaluating business tenants differs significantly. Our schema may need additional fields for assessing business viability, such as financial statements, business plans, and references.

Reporting and Analytics:

Expanding to commercial properties may require advanced reporting and analytics capabilities. Our schema may need enhancements to generate custom reports, financial analyses, and occupancy forecasts tailored to commercial leasing activities.

In summary, integrating business rentals introduces complexities, requiring modifications to accommodate commercial leasing. With careful planning, our schema can support diverse rental scenarios, ensuring scalability and flexibility for future growth.