Global Unity Realty Services: Database Requirements Document

Lloyd Thomas

Objective:

Global Unity Realty Services (GURS) aims to provide comprehensive property management solutions on a global scale, fostering unity and collaboration in the real estate sector.

Problem Description:

The absence of centralized and comprehensive management systems inhibits investors from engaging in diverse property ventures, especially in third-world countries.

Challenges such as navigating complex regulatory environments, overcoming communication barriers with tenants, tracking maintenance tasks and expenses, generating detailed financial reports, mitigating compliance risks, and managing documents heighten these concerns.

Database Rules:

Each Property Portfolio is managed by a client user. The client user table has a unique identifier (user_id) and contains attributes such as the client's first and last name and last login to ensure trackability.

Property Portfolio has a unique identifier (portfolio_id). Each portfolio has a name, total value, and status. The portfolio is made up of and displays properties, leases, financial reports, and transactions made by the client user. A property portfolio cannot exist without a client user, and a client user can have multiple portfolios, but a portfolio can only pertain to one user.

Transactions have a unique identifier (transaction_id). Each transaction has a type, date, and amount. A client user can have multiple transactions, but each transaction pertains only to one user. Furthermore, if a property portfolio record is destroyed, so is the record of transactions.

Each Property has a unique identifier (property_id). Each property has a value, address, acquisition_date, square footage, and type. A property portfolio can have 0 to many properties; however, a property can only pertain to one portfolio. If an instance of a portfolio is deleted, so are the instances of properties beneath that portfolio.

Each financial report has its own unique identifier (report_id). Each report has its own unique type, start date, end date, income, expenses, and summary. Also, every report is attached to a unique property, which can be seen as the financial report stores its property_id. Furthermore, a financial report can exist independently from its property, but their relationships are severely correlated. A property can contain multiple financial reports, but a financial report can only pertain to one property.

Each lease has its own unique identifier (lease_id). Each lease is made up of its start date, end date, rent amount, renewal date, and summary. A property can have multiple leases; however, a lease can only pertain to one property. Furthermore, if the property that the lease pertains to is destroyed, so is the instance of the lease. Also, the lease hosts information on the tenant on the lease, its tenant_id. A lease can have 0 to many tenants.

Each tenant has its own unique identifier (tenant_id). Each tenant table includes the tenant's first and last name alongside their email. The Tenants are also connected to the Lease table by having a unique lease id. Furthermore, a tenant can have one to many leases.

Possible Nouns and Verbs

Nouns:
Client User
Property Portfolio
Lease
Tenant
Property
Property Type
Financial Report
Report Type
Transaction
Transaction Type

Verbs:
Manage (e.g., manage property portfolios)
Owns (e.g., owns a property)
Initiates (e.g., initiates a lease agreement)
Create (e.g., create a new client user)
Update (e.g., update property information)
Delete (e.g., delete a financial report)
View (e.g., view property portfolio details)
Assign (e.g., assign a tenant to a lease)
Generate (e.g., generate a financial report)
Calculate (e.g., calculate total income)
Filter (e.g., filter transactions by type)
Search (e.g., search for properties by address)
Analyze (e.g., analyze lease renewal dates)