



Wishlist / Subscription

We assume that every wishlist is associated with exactly one subscription. Conceptually, a wishlist represents a concrete request formulated by the client at some point in time (for instance, "properties with 3 bedrooms, balcony, and parking"). As soon as the corresponding request has been processed and completed, the wishlist is considered closed and immutable in that it is kept for traceability and analytic purposes, but will no longer be modified.

Conversely, one subscription can be associated with more than one wishlist over time. This models the fact that a client will declare several distinct sets of criteria as their needs evolve, e.g., first looking for a rental and then later for a purchase. The model thus represents a 1-to-many relationship between Subscription and Wishlist, which is consistent with the business logic whereby the subscription is the long-term contract, while the wishlists are episodic, time-bounded requests.

Property Feature / Feature

We take this to mean that the attribute Value_Type in the Feature table represents the data type used to express the value of that feature. More concretely, Value_Type can take one of three conceptual categories:

Boolean: features which are either present or absent, such as "has a pool", "has a garage",

Numeric: features that are numeric in nature - "number of bedrooms", "square footage"

Enumeration: for features that belong to a predefined list of values, such as "heating type = electric / gas / oil".

This assumption enables the system to treat heterogeneous feature semantics in a uniform manner while maintaining consistent storage and validation rules. It also details how the various Value_Boolean, Value_Number, and Value_Enum attributes in Property Feature are to be utilized based on the Value_Type defined in Feature.

Clients / Subscription

We will also assume that a client may only be active in one subscription at any given time, and that one subscription is classified as either "free" or "paid". If a customer cancels their subscription and at some point later reinstates it, for our purposes here, we will still consider this to be the same logical subscription as long as the ClientID has not changed. In other words, the lifecycle of a subscription may include states like active, suspended, and possibly even reactivated, but is tracked under a single SubscriptionID.

An exception is made in "structural change" scenarios such as a change of address that leads to the creation of a new ClientID, per our assumption. In that case, the new client record will be associated with a new subscription. This assumption therefore enforces a 1-to-1 relationship between Client and Subscription, which simplifies billing, tracking of usage, and business rules, for example, eligibility to certain services depending on subscription type.

Clients/Wishlist

We have not specified a direct relationship between Clients and Wishlist. Instead, we take the link to be implicitly mediated by the Subscription table; given that each SubscriptionID corresponds to exactly one ClientID and every Wishlist is assigned a SubscriptionID, we can always trace through this chain to determine which client owns a given wishlist.

Since one client can have one and only one subscription, and each subscription can have multiple wishlists, this design yields an indirect 1-to-many relationship between Client and Wishlist. This modeling keeps the schema normalized, avoids redundant foreign keys, and still enables us to retrieve all the wishlists that belong to one certain client.

Wishlist

We will assume here that the Weight attribute of the Wishlist-or, rather, of the associated detail-is limited to two values: "must have" and "nice-to-have". This is a deliberately simple weighting system: it provides the system with a way of ranking the criteria when matching properties against client expectations, without overly complicated scoring rules.

Limiting Weight to the above two values ensures that an importance order in a descending manner can be straightforwardly established during evaluation, as the

identification process will have to meet all "must have" criteria before the refinement of results or ranking can be performed using "nice-to-have" criteria. This assumption thus directly supports use cases like recommendation algorithms or search filters where being able to tell essential vs. optional criteria is key.

Customers / Billing Account

One client might have more than one billing account; this would occur, for example, when a client moves and thus gets a new billing address, or when they deal with different accounts in different contexts - such as personal residence versus investment properties.

Under this assumption, the relationship between Clients and Billing Account is 1-to-many. This allows us to preserve billing history and tax information tied to specific addresses while still recognizing that they all belong to the same client entity. The latter also supports such scenarios where historic billing accounts must be retained for legal or auditing purposes even after the client's main address changes.

Address / Property

We assume that one address can be associated with multiple properties, which models the situations when, for example, apartment buildings, condominium complexes, or multi-unit dwellings have several distinct properties sharing the same civic address regarding street, city, and postal code but represent different units.

In this design, the relationship is 1-to-many: Address → Property. This especially holds true in situations where individual units within an address are not differentiated, such as when unit numbers are handled elsewhere or not required. This allows the system to maintain various property records with their own features, listings, and pricing while reutilizing the same base address data, maintaining normalization without duplication. Means of Payment We assume that each billing account can be associated with several payment methods. For instance, a client can register both a personal PayPal account and a corporate credit card under one and the same billing account. The actual selection of the preferred payment method is done on a per-payment-transaction basis; however, all qualifying payment methods are persisted and administered at the billing-account level. Thus, the relationship between Billing Account and Payment Method is 1-to-many. Every payment record refers to precisely one payment method, allowing easy traceability of how a certain invoice was settled, eg which card or PayPal account was used. Simultaneously, the billing account serves as a container for all of the payment options available to the client for that account; thereby supporting flexible payment behavior without compromising data integrity or auditability.