# **Cold Storage Marketplace ERD**

ER (Entity-Relationship) diagrams graphically model database schemas, showing entities (tables), their attributes (columns), and relationships. In our design, each entity has a **primary key (PK)** that uniquely identifies its records, and **foreign keys (FK)** to reference related entities, enforcing referential integrity. Unique constraints (e.g. on Users.email) ensure no duplicate values in important fields.

* **Users** – user\_id (PK, INTEGER), email (VARCHAR(255), UNIQUE), name (VARCHAR), created\_at (TIMESTAMP), updated\_at (TIMESTAMP). *(Customers or storage owners; uniquely identified by user\_id.)*
* **Cold\_Rooms** – cold\_room\_id (PK, INTEGER), owner\_id (INTEGER, FK→Users.user\_id), name (VARCHAR), location (GEOGRAPHY), capacity (INTEGER), created\_at (TIMESTAMP), updated\_at (TIMESTAMP). *(Refrigerated storage units; each belongs to an owner user.)*
* **Bookings** – booking\_id (PK, INTEGER), user\_id (INTEGER, FK→Users.user\_id), cold\_room\_id (INTEGER, FK→Cold\_Rooms.cold\_room\_id), start\_date (TIMESTAMP), end\_date (TIMESTAMP), status (VARCHAR), created\_at (TIMESTAMP), updated\_at (TIMESTAMP). *(Reservations linking a user to a cold room for a time range.)*
* **Payments** – payment\_id (PK, INTEGER), booking\_id (INTEGER, FK→Bookings.booking\_id), amount (DECIMAL(10,2)), payment\_date (TIMESTAMP), method (VARCHAR), status (VARCHAR), created\_at (TIMESTAMP), updated\_at (TIMESTAMP). *(Payment records for bookings.)*
* **Notifications** – notification\_id (PK, INTEGER), user\_id (INTEGER, FK→Users.user\_id), message (TEXT), type (VARCHAR), is\_read (BOOLEAN), created\_at (TIMESTAMP). *(Alerts sent to users about booking or system events.)*
* **Cold\_Room\_Verifications** – verification\_id (PK, INTEGER), cold\_room\_id (INTEGER, FK→Cold\_Rooms.cold\_room\_id), status (VARCHAR), verified\_by (INTEGER, FK→Users.user\_id), verified\_at (TIMESTAMP), notes (TEXT). *(Tracks administrative verification of each cold room’s data.)*
* **Admin\_Actions** – action\_id (PK, INTEGER), admin\_id (INTEGER, FK→Users.user\_id), action\_type (VARCHAR), target\_entity (VARCHAR), target\_id (INTEGER), action\_at (TIMESTAMP). *(Logs actions taken by administrators on various entities.)*

**Relationships (Cardinalities):** Each **User** can own many **Cold\_Rooms** and make many **Bookings** (1:N), while each Cold\_Room belongs to one User. Each **Cold\_Room** can have many **Bookings** and many **Cold\_Room\_Verifications** (1:N). Each **Booking** is linked to one User and one Cold\_Room (N:1 on each side). Each **Booking** may have one or more **Payments** (1:N). Each **User** can receive multiple **Notifications** (1:N). Each **User** (as an admin) can log many **Admin\_Actions** (1:N). These one-to-many relationships are indicated on the ERD with “1” and “N” labels at the ends of the connecting lines.

**Diagram Conventions:** In the final ERD image, each table is shown as a box listing its fields and data types, with the PK underlined or marked (PK) and FK marked (FK). Cardinalities (“1” or “N”) are labeled on the relationship lines to show one-to-many links. This clear, professional layout of entities and relationships (as depicted above) ensures backend engineers and architects can easily understand the schema and how data flows through the system.

**Sources:** ER diagrams are a standard tool for designing relational databases. In ER modeling, a *primary key* uniquely identifies each entity record, and a *foreign key* points to a primary key in another table to define relationships. Cardinality annotations (1:N, etc.) explicitly describe how many instances of one entity relate to another, aiding in accurate database design.