Design Documentation

FINAL PROJECT COSC 2307 – Jojo 5149978

The Room Rental Management System (RRMS)

1. Introduction to the Database Design

The Room Rental Management System (RRMS) database is designed as a fully normalized relational schema supporting multi-landlord operations, rental contracts, automated recurring billing, and payment tracking. The system cleanly separates master data (landlords, properties, units, tenants, contracts), transactional data (rent charges, payments), and reference data (statuses, billing cycles, penalty rules, assets, utilities).
This design eliminates redundancy, prevents anomalies, and ensures strong data consistency. It also enables scalable reporting for monthly revenue, occupancy rate, overdue balances, and contract history.

2. Requirements & Scope Summary

At a high level, the RRMS database supports a multi-landlord SaaS scenario with the following scope:

- **Multi-Landlord & Property Management**
    - Store multiple landlords.
    - Each landlord can own multiple properties (houses, apartments, or buildings).
    - Each property can contain one or more rentable units (individual rooms, suites, or full-unit rentals).
- **Tenant & Contract Management (Billing-Centric)**
    - Store basic tenant profiles.
    - Store billing-centric rental contracts linking a tenant to a specific unit, with fields such as:
        - start and end date,
        - agreed rent amount,
        - billing cycle (e.g., monthly, yearly),

        - billing day / due date,
        - penalty rule (fixed or percentage).
    - Maintain contract status (e.g., Active, Ended) and preserve historical contracts.
- **Rent Charges, Penalties & Payments**
    - Generate periodic rent charges based on the contract's billing configuration.
    - Track each rent charge with:
        - billing period,
        - due date,
        - amount due and penalty amount,
        - status (Pending, Paid, PartiallyPaid, Overdue).
    - Record one or more payments per rent charge, supporting partial payments.
    - Automatically compute late penalties based on contract-defined rules.
- **Assets & Utilities**
    - Maintain a simple list of assets (furniture, appliances) per unit or property.
    - Maintain a simple list of utilities/services associated with a property or unit (e.g., hydro, water, internet).
- **Reporting & Analytics**
    - Enable queries and BI tools to report on:
        - monthly and yearly revenue,
        - occupancy rate by unit/property,
        - outstanding and overdue balances,
        - active vs historical contracts,
        - per-landlord portfolio performance.

Non-functional requirements include BCNF-level normalization, strong referential integrity, index support on key analytic fields (dates, landlord, contract, unit), and an analytics-friendly design for Power BI or similar tools.

3. Conceptual Design

The conceptual design of the Room Rental Management System (RRMS) defines the core business entities, their essential attributes, and the relationships that structure rental operations. This section outlines the entities, business rules, and cardinality constraints that guide the logical and physical design of the database. The model is designed to support multi-landlord operations, recurring billing, payment tracking, and analytics while ensuring strong data consistency and minimal redundancy.

3.1 Entities & Descriptions

1. Landlord

Represents a property owner or rental company operating within the system.
A landlord manages one or more properties and is the top-level entity for multi-tenant isolation.

Key attributes: landlord_id, first_name, last_name, business_name, email, phone.

2. Property

Represents a physical rental asset such as a house, townhouse, or apartment building. Each property is owned by a single landlord and contains one or more individually rentable units.

Key attributes: property_id, landlord_id, property_name, property_type, full_address, post_code.

3. Unit

Represents an individual rentable unit within a property (room, apartment, or full house). It is the primary entity used for determining occupancy and linking tenants to a rental space.

Key attributes: unit_id, property_id, unit_code, floor_no, capacity, unit_type, unit_status_id.

4. Tenant

Represents an individual renting a unit.
A tenant may have multiple contracts over time, but typically only one active rental contract at once.

Key attributes: tenant_id, first_name, last_name, occupation, email, phone, id_type, id_number.

5. PenaltyRule

Defines how late-payment penalties are calculated for contracts.
Rules may be reused across multiple contracts.

Key attributes: penalty_rule_id, penalty_type (fixed or percentage), penalty_amount, grace_period_days, rule_description.

6. BillingCycleType (Lookup)

Specifies the billing frequency of a contract (e.g., Monthly, Yearly, Custom).

Key attributes: billing_cycle_type_id, name, description.

7. ContractStatus (Lookup)

Represents the status of a rental contract (e.g., Active, PendingStart, Ended).

Key attributes: contract_status_id, status_code, description.

8. Contract

Represents a rental agreement between a tenant and a unit.
It contains all billing-relevant information used to generate monthly rent charges.

Key attributes:
contract_id, unit_id, tenant_id, penalty_rule_id, billing_cycle_type_id,
contract_status_id,
start_date, end_date, base_rent_amount, billing_day.

9. ChargeStatus (Lookup)

Represents the status of a rent charge (e.g., Pending, Paid, PartiallyPaid, Overdue).

Key attributes: charge_status_id, status_code, description.

10. RentCharge

Represents a single periodic rent charge generated for a contract (e.g., “January 2026”).
A rent charge anchors payments, penalties, and overdue tracking.

Key attributes:
rent_charge_id, contract_id, charge_period, period_start, period_end,
due_date, base_amount, penalty_amount, total_amount_due, charge_status_id.

11. PaymentMethod (Lookup)

Represents allowed payment methods (e.g., Cash, e-Transfer, Bank Transfer).

Key attributes: payment_method_id, method_code, description.

12. Payment

Represents a payment made toward a specific rent charge.
The system supports partial payments, multiple payments per charge, and payment history.

Key attributes:
payment_id, rent_charge_id, payment_date, amount_paid,
payment_method_id, reference_number.

13. UnitAsset

Represents furniture or appliances associated with a rental unit.
Supports move-in/move-out tracking.

Key attributes: unit_asset_id, unit_id, asset_name, quantity, condition.

14. UnitUtility

Represents utilities or services associated with a unit (e.g., hydro, water, internet).

Key attributes: unit_utility_id, unit_id, utility_type, billing_method.

3.2 Business Rules & Cardinality

Landlords, Properties, and Units

1. **One landlord may own many properties.**
   Cardinality: Landlord (1) — (N) Property
2. **One property may contain multiple units.**
   Cardinality: Property (1) — (N) Unit
3. **Each unit belongs to exactly one property.**
   Cardinality: Unit (N) — (1) Property

Tenants and Contracts

4. **One tenant may have multiple contracts over time.**
   Cardinality: Tenant (1) — (N) Contract
5. **One unit may have multiple historical contracts but only one active contract at a time.**
   Business constraint: No overlapping active rental periods.

6. **Each contract links exactly one tenant and one unit.**
   Cardinality: Contract — Tenant (N — 1), Contract — Unit (N — 1)

Contracts, Rent Charges, and Payments

7. **A contract can generate many rent charges (typically monthly).**
   Cardinality: Contract (1) — (N) RentCharge
8. **Each rent charge is tied to a single contract.**
   Cardinality: RentCharge (N) — (1) Contract
9. **A rent charge may have zero, one, or multiple payments.**
   Cardinality: RentCharge (1) — (N) Payment
10. **Each payment must reference exactly one rent charge.**
    Cardinality: Payment (N) — (1) RentCharge
11. **A penalty rule may apply to many contracts.**
    Cardinality: PenaltyRule (1) — (N) Contract

Assets and Utilities

12. **A unit may have multiple assets.**
    Cardinality: Unit (1) — (N) UnitAsset
13. **A unit may have multiple utilities.**
    Cardinality: Unit (1) — (N) UnitUtility

Multi-Landlord Isolation

14. **All operational data must be scoped to a landlord through the hierarchy:**
    Landlord → Property → Unit → Contract → RentCharge → Payment.
    This ensures strict data isolation across landlords within the SaaS environment.

3.3 Conceptual ER Diagram

The conceptual ERD for the RRMS consists of the following major entity groups:

Core Entities:

Landlord, Property, Unit, Tenant, Contract.

Transactional Entities:

RentCharge, Payment.

Reference Entities (Lookups):

BillingCycleType, ContractStatus, ChargeStatus, PaymentMethod, UnitStatus (optional).

Supportive Entities:

PenaltyRule, UnitAsset, UnitUtility.

Key Relationships:

Landlord → Property → Unit

Tenant → Contract → RentCharge → Payment

PenaltyRule → Contract

Unit → UnitAsset / UnitUtility

This structure ensures a modular, scalable, and analytics-friendly foundation for the RRMS.

4. Logical Design

The logical design provides a detailed relational representation of the system based on the conceptual model. It defines the structure of each table, primary keys, foreign keys, and the logical relationships between entities. This layer ensures that the database design satisfies normalization principles (up to BCNF) and accurately reflects the business rules of the Room Rental Management System (RRMS).

A complete logical Entity–Relationship Diagram (ERD), including all entities and their PK/FK relationships, is **attached at the end of this document**.

4.1 Relational Structure Overview

The RRMS logical model is centered around the hierarchy:

Landlord → Property → Unit → Contract → RentCharge → Payment

Additional supporting entities include:

- Tenants
- Penalty rules
- Billing and status lookup tables
- Unit-level assets and utilities

This structure maintains strict logical separation between landlords, supports contract-driven billing, enables partial payments, and retains full historical data.

4.2 Logical Entities, Attributes, and Keys

4.2.1 LANDLORD

Attribute	Description
landlord_id (PK)	Unique identifier
first_name	Landlord first name
last_name	Landlord last name
business_name	Company/rental business name
email	Primary contact
phone	Phone number
interac_contact	Payment contact

4.2.2 PROPERTY

Attribute	Description
property_id (PK)	Unique identifier
landlord_id (FK)	Links property to landlord
property_name	Label of property
property_type	House, apartment, building
full_address	Address
post_code	Postal code
interac_contact	Payment contact

4.2.3 UNIT

Attribute	Description
unit_id (PK)	Unique identifier
property_id (FK)	Unit belongs to a property
unit_code	Room/Unit code
floor_no	Floor level
capacity	Maximum occupants
unit_type	Room, suite, full rental
unit_status_id (FK)	Status lookup (e.g., available, occupied)

4.2.4 TENANT

Attribute	Description
tenant_id (PK)	Unique identifier
first_name	Tenant first name
last_name	Tenant last name
occupation	Job/role
email	Contact email
phone	Contact number
id_type	ID type (e.g., passport)
id_number	ID number

4.2.5 PENALTYRULE

Attribute	Description
penalty_rule_id (PK)	Unique identifier
penalty_type	FIXED or PERCENTAGE
penalty_amount	Amount or rate
grace_period_days	Late grace period
rule_description	Explanation

4.2.6 BILLINGCYCLETYPE (Lookup)

Attribute	Description
billing_cycle_type_id (PK)	Unique identifier
name	Monthly, Yearly, Custom
description	Notes

4.2.7 CONTRACTSTATUS (Lookup)

Attribute	Description
contract_status_id (PK)	Active, Pending, Ended
status_code	Status text
description	Notes

4.2.8 CONTRACT

Attribute	Description
contract_id (PK)	Unique identifier
unit_id (FK)	Unit being rented
tenant_id (FK)	Tenant renting the unit
penalty_rule_id (FK)	Applied penalty rule
billing_cycle_type_id (FK)	Billing frequency
contract_status_id (FK)	Contract status
start_date	Start of tenancy
end_date	End of tenancy
base_rent_amount	Rent amount
billing_day	Due day of every cycle

4.2.9 CHARGESTATUS (Lookup)

Attribute	Description
charge_status_id (PK)	Unique identifier
status_code	Pending, Paid, Partially Paid, Overdue
description	Notes

4.2.10 RENTCHARGE

Attribute	Description
rent_charge_id (PK)	Unique identifier
contract_id (FK)	Which contract it belongs to
charge_period	Billing label (e.g., "2025-01")
period_start	Start of billing period
period_end	End of billing period
due_date	Payment deadline

base_amount	Rent amount
penalty_amount	Late penalty
total_amount_due	Sum due
charge_status_id (FK)	Charge status

4.2.11 PAYMENTMETHOD (Lookup)

Attribute	Description
payment_method_id (PK)	Cash, e-Transfer, Bank transfer
method_code	Method name
description	Notes

4.2.12 PAYMENT

Attribute	Description
payment_id (PK)	Unique identifier
rent_charge_id (FK)	Charge being paid
payment_date	Date received
amount_paid	Payment amount
payment_method_id (FK)	How it was paid
reference_number	Receipt or transfer note

4.2.13 UNITASSET

Attribute	Description
unit_asset_id (PK)	Unique identifier
unit_id (FK)	Linked unit
asset_name	Furniture/equipment
quantity	Count
condition	Asset condition

4.2.14 UNITUTILITY

Attribute	Description
unit_utility_id (PK)	Unique identifier
unit_id (FK)	Linked unit
utility_type	Hydro, Water, Internet
billing_method	Included or separate

5. Physical Design

The physical design specifies how the logical model is implemented in PostgreSQL using SQL data types, constraints, indexes, and optimization techniques. This layer focuses on storage, performance, and enforcement of data integrity.

5.1 Table Definitions (PostgreSQL Physical Schema)

5.1.1 Physical Characteristics

Logical Element	Physical Implementation
Identifiers (PK)	SERIAL / BIGSERIAL or GENERATED ALWAYS AS IDENTITY
Text attributes	VARCHAR(n) or TEXT
Monetary values	NUMERIC(12,2)
Boolean values	BOOLEAN
Dates	DATE
Status domains	Stored via lookup tables

5.2 Key Physical Constraints

Primary Keys

All main entities use:

PRIMARY KEY (entity_id)

ensuring entity integrity.

Foreign Keys

Referential integrity enforced with:

REFERENCES parent_table(id)

ON UPDATE CASCADE

ON DELETE RESTRICT

Check Constraints

Examples:

billing_day BETWEEN 1 AND 31

penalty_type IN ('FIXED','PERCENTAGE')

Unique Constraints

- email uniqueness for landlords & tenants
- (property_id, unit_code) to prevent duplicate units

5.3 Indexing Strategy

Indexes improve query performance for analytical reports and join-heavy queries:

Recommended Indexes

Table	Index
CONTRACT	idx_contract_unit (unit_id)
RENTCHARGE	idx_rentcharge_contract (contract_id)
RENTCHARGE	idx_rentcharge_status (charge_status_id)
PAYMENT	idx_payment_rentcharge (rent_charge_id)
UNIT	idx_unit_property (property_id)
PROPERTY	idx_property_landlord (landlord_id)

5.4 Storage, Performance, and Optimization Notes

- PostgreSQL automatically clusters sequential ID fields, improving scan performance.
- Analytical queries benefit from indexes on foreign keys.
- Rent history aggregation is optimized via indexing on charge_period.
- Lookup tables ensure low storage cost and fast joins.
- Tables are normalized to BCNF, reducing redundancy and minimizing storage overhead.

5.5 Physical Design Summary

The physical design provides:

- PostgreSQL-ready data types
- Primary/foreign key enforcement
- Performance-oriented indexes
- Data integrity via constraints
- Storage-efficient normalized structures

This ensures the RRMS database performs reliably for operational tasks and scalable analytical reporting.