P3 Logical ERD Model

Topic: EV Charging Station Management System

Team Project Submission
Course: Data mgt and Database Design
Northeastern University

Group Members:

- Devanshu Chicholikar
- Saurabh Kashyap
- Joseph Alex Chakola
- XingXing Xiao

OU GitHub URL is below (public repo):
https://github.com/starsbro/DAMG6210_Group02

Professor’s comment for P2:
Some elements of **the logical model are present** (e.g., UserID as a foreign key in other entities).
However, your entities and relationships are well organized and clearly structured.

The initial ERD has some logical models present. We modified it.
A fine-tuned ERD as below:
https://drive.google.com/file/d/1LZu0PHrNYvao-j1FphIokI9uf1-vx8Hl/view?usp=sharing

We didn’t update our design, Our P3 Logical ERD Model as below:

https://drive.google.com/file/d/1g4_e1tQ1uhCachDW3DLGr6DmrVHi3zCe/view?usp=sharing

1) Scope

This document summarizes the logical data model for the EV Charging Station system. It refines the prior conceptual design by specifying attributes, primary keys (PK), and foreign keys (FK). The lifecycle and business rules remain the same.

2) What Changed from P-2

- Added concrete attributes for each entity (IDs, names, timestamps, amounts, statuses).
- Declared primary keys (surrogate integer IDs) for all tables.
- Introduced foreign keys to realize relationships and optionalities from the conceptual model.
- Preserved original cardinalities and business rules (no double-booking, optional session for a reservation, 1:1 invoice per session, multi-attempt payments, subscriptions).

3) Entities & Key Foreign Keys

- USER (PK: UserID) — parent for VEHICLE, RESERVATION, PAYMENT_METHOD, NOTIFICATION.
- VEHICLE (PK: VehicleID; FK: UserID→USER) — vehicle belongs to one user.
- STATION (PK: StationID).
- CHARGE_POINT (PK: ChargePointID; FK: StationID→STATION).
- RESERVATION (PK: ReservationID; FKs: UserID→USER, ChargePointID→CHARGE_POINT, VehicleID→VEHICLE) — time window & status; overlapping windows disallowed.
- CHARGING_SESSION (PK: ChargingSessionID; FKs: UserID→USER, ChargePointID→CHARGE_POINT, BillingPlanID→SUBSCRIPTION_PLAN) — may exist without a reservation (walk-ins).
- INVOICE (PK: InvoiceID; FKs: UserID→USER, ChargingSessionID→CHARGING_SESSION (1:1), UserSubscriptionID→USER_SUBSCRIPTION for monthly fees).
- PAYMENT (PK: TransactionID; FKs: InvoiceID→INVOICE, PaymentMethodID→PAYMENT_METHOD) — multiple attempts per invoice.
- PAYMENT_METHOD (PK: PaymentMethodID; FK: UserID→USER).
- SUBSCRIPTION_PLAN (PK: BillingPlanID) — rate attributes.
- USER_SUBSCRIPTION (PK: UserSubscriptionID; FKs: UserID→USER, BillingPlanID→SUBSCRIPTION_PLAN) — history of plan enrollment and source of recurring invoices.
- NOTIFICATION (PK: NotificationID; FKs: UserID→USER, optional links to ChargingSessionID/ReservationID/TransactionID).
- MAINTENANCE_RECORD (PK: MaintenanceRecordID; FK: ChargePointID→CHARGE_POINT) — service history.
- OPERATOR (PK: OperatorID; FK: StationID→STATION) — station accountability.

4) Relationship Confirmations

- USER 1–M VEHICLE; USER 1–M RESERVATION; STATION 1–M CHARGE_POINT.
- CHARGE_POINT 1–M RESERVATION with no overlapping active windows (enforced in DB/app logic).

- RESERVATION 1–0..1 CHARGING_SESSION (supports no-show and walk-in policies).
- CHARGING_SESSION 1–1 INVOICE; INVOICE 1–M PAYMENT (retry attempts).
- USER 1–M PAYMENT_METHOD; USER_SUBSCRIPTION 1–M INVOICE for recurring fees.
- OPERATOR 1–M MAINTENANCE_RECORD (via station/point oversight).

5) Integrity & Normalization

- Primary keys are single-column surrogate IDs; foreign keys maintain referential integrity.
- Core parent/lookup tables (USER, STATION, SUBSCRIPTION_PLAN) are in 3NF; no transitive dependencies on non-keys.
- Business rules (no double-booking, invoice generation, payment retries) are supported through the FK paths and time-window checks.

6) Assumptions

- Walk-ins allowed → a CHARGING_SESSION may have no RESERVATION.
- Exactly one INVOICE per CHARGING_SESSION; multiple PAYMENT attempts per INVOICE.
- Payment methods are tokenized; no sensitive card data stored.

7) Conclusion

The logical ERD formalizes the original design with explicit columns and foreign keys, preserving the end-to-end lifecycle: reserve → charge → invoice → pay. This model is ready for DDL implementation.