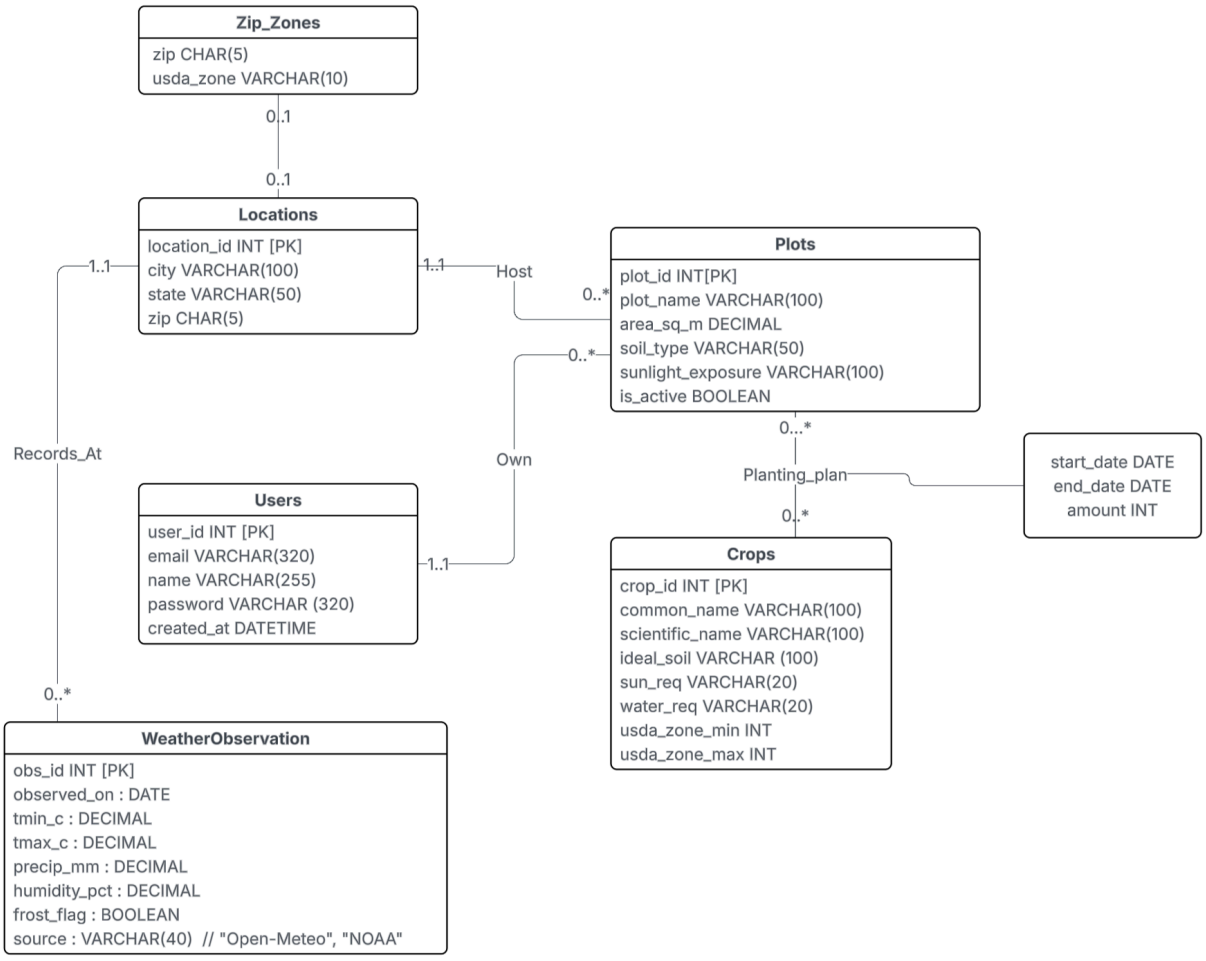
Conceptual and Logical Database Design

UML Diagram



Entities: Assumptions and Explanations

User: One account manages gardens. We expect one owner per plot and possible future roles/permissions. Keep as an entity so one user can own many plots.

Location: A physical site (address, lat/long, USDA zone). Multiple plots can share one site and all weather attaches to the site. Keep as an entity to keep geo/zone data in one place.

GardenPlot: A distinct bed/area with stable traits (area, soil, sun, irrigation, active flag). It belongs to exactly one owner and one location.

Crop: Catalog of plant types with specific crop facts (days to maturity, spacing, sun/water needs). Kept as an entity so these facts are defined once and reused across many plantings.

WeatherObservation: Daily weather measured at a location (observed_on date, tmin/tmax, precip, humidity, wind, optional soil temp, frost flag, source). Stored as its own entity for recommendations and trend analysis across time.

Relationships: Cardinality and Explanations

User ← owns → GardenPlot:
1 ↔ many.
Each plot has exactly one accountable owner, while a user may own zero or many plots.

Location ← hosts → GardenPlot:
1 ↔ many.
A site can have many plots, each plot sits at exactly one site.

GardenPlot ← grows → Crop (relationship with attributes – Planting):
many ↔ many, with relationship attributes

Location ← records_at → WeatherObservation:
1 ↔ many.
Many daily observations over time per site. Each weather row belongs to one location.

Functional Dependencies

Location Table
- location_id → city, state, zip, usda_zone
- zip → usda_zone

User Table
- user_id → email, name, password, created_at
- email → user_id, name, password, created_at

GardenPlot Table
- plot_id → plot_name, location_id, user_id, area_sq_m, soil_type, sunlight_exposure, is_active

Crop Table
- crop_id → common_name, scientific_name, ideal_soil, sun_req, water_req, usda_zone_min, usda_zone_max

Planting_plan Table
- plot_id, crop_id → start_date, end_date, amount

Weather Observation Table

- obs_id → location_id, observed_on, tmin_c, tmax_c, precip_mm, humidity_pct, wind_kph, frost_flag, source

BCNF Normalization

Location Table

- R (location_id, city, state, zip, usda_zone)
- Dependencies
    - location_id → city, state, zip, usda_zone
    - zip → usda_zone

1) Check if R is in BCNF
    a) zip → usda_zone violates BCNF because it is not a superkey
    b) zip += (zip, usda_zone)
    c) R1 = (zip, usda_zone), R2 = (location_id, zip, city, state)
2) We see that both are in BCNF because all keys are now superkeys

User Table

- R(user_id, email, name, password, created_at)
- Dependencies
    - user_id → email, name, password, created_at
    - email → user_id, name, password, created_at

1) Since the only keys are superkeys, this is already in BCNF

GardenPlot Table

- R(plot_id, plot_name, location_id, user_id, area_sq_m, soil_type, sunlight_exposure, is_active)
- Dependencies
    - plot_id → plot_name, location_id, user_id, area_sq_m, soil_type, sunlight_exposure, is_active

1) Since the only key is a superkey, this is already in BCNF

Crop Table

- R(crop_id, common_name, scientific_name, ideal_soil, sun_req, water_req, usda_zone_min, usda_zone_max)
- Dependencies
    - crop_id → common_name, scientific_name, ideal_soil, sun_req, water_req, usda_zone_min, usda_zone_max

1) Since the only key is a superkey, this is already in BCNF

Planting_plan Table

- R(plot_id, crop_id, start_date, end_date, amount)
- Dependencies

    - plot_id, crop_id → start_date, end_date, amount
1) Since the only key is a superkey, this is already in BCNF

Weather Observation Table

- R(obs_id, location_id, observed_on, tmin_c, tmax_c, precip_mm, humidity_pct, wind_kph, frost_flag, source)
- Dependencies
    - obs_id → location_id, observed_on, tmin_c, tmax_c, precip_mm, humidity_pct, wind_kph, frost_flag, source

1) Since the only key is a superkey, this is already in BCNF

Relational Schema

```
Location (
        location_id: INT [PK],
        city: VARCHAR(100),
        state: VARCHAR(50),
        zip: CHAR(5) [FK to ZipZone.zip]
)

ZipZone(
        zip: CHAR(5) [PK],
        usda_zone: VARCHAR(10)
}

User (
        user_id: INT PRIMARY KEY,
        email: VARCHAR(320) UNIQUE,
        name: VARCHAR(255),
        password: VARCHAR(320),
        created_at: DATETIME
)

GardenPlot (
        plot_id: INT [PK],
        user_id: INT [FK to User.user_id],
        location_id: INT [FK to Location.location_id],
        plot_name: VARCHAR(100),
        area_sq_m: DECIMAL,
        soil_type: VARCHAR(50),
        sunlight_exposure: VARCHAR(100),
        is_active: BOOLEAN
)
```

```
Crop (
        crop_id: INT [PK],
        common_name: VARCHAR(100),
        scientific_name: VARCHAR(100),
        ideal_soil VARCHAR(100),
        sun_req: VARCHAR(20),
        water_req: VARCHAR(20),
        usda_zone_min INT,
        usda_zone_max INT
)

Planting_Plan (
        plot_id: INT NOT NULL [FK to GardenPlot.plot_id],
        crop_id: INT NOT NULL [FK to Crop.crop_id],
        start_date: DATE,
        end_date: DATE,
        amount INT,
        PRIMARY KEY (plot_id, crop_id)
)

WeatherObservation (
        obs_id: INT [PK],
        location_id: INT [FK to Location.location_id],
        observed_on: DATE,
        tmin_c: DECIMAL,
        tmax_c: DECIMAL,
        precip_mm: DECIMAL,
        humidity_pct: DECIMAL,
        frost_flag: BOOLEAN,
        source: VARCHAR(40)
)
```