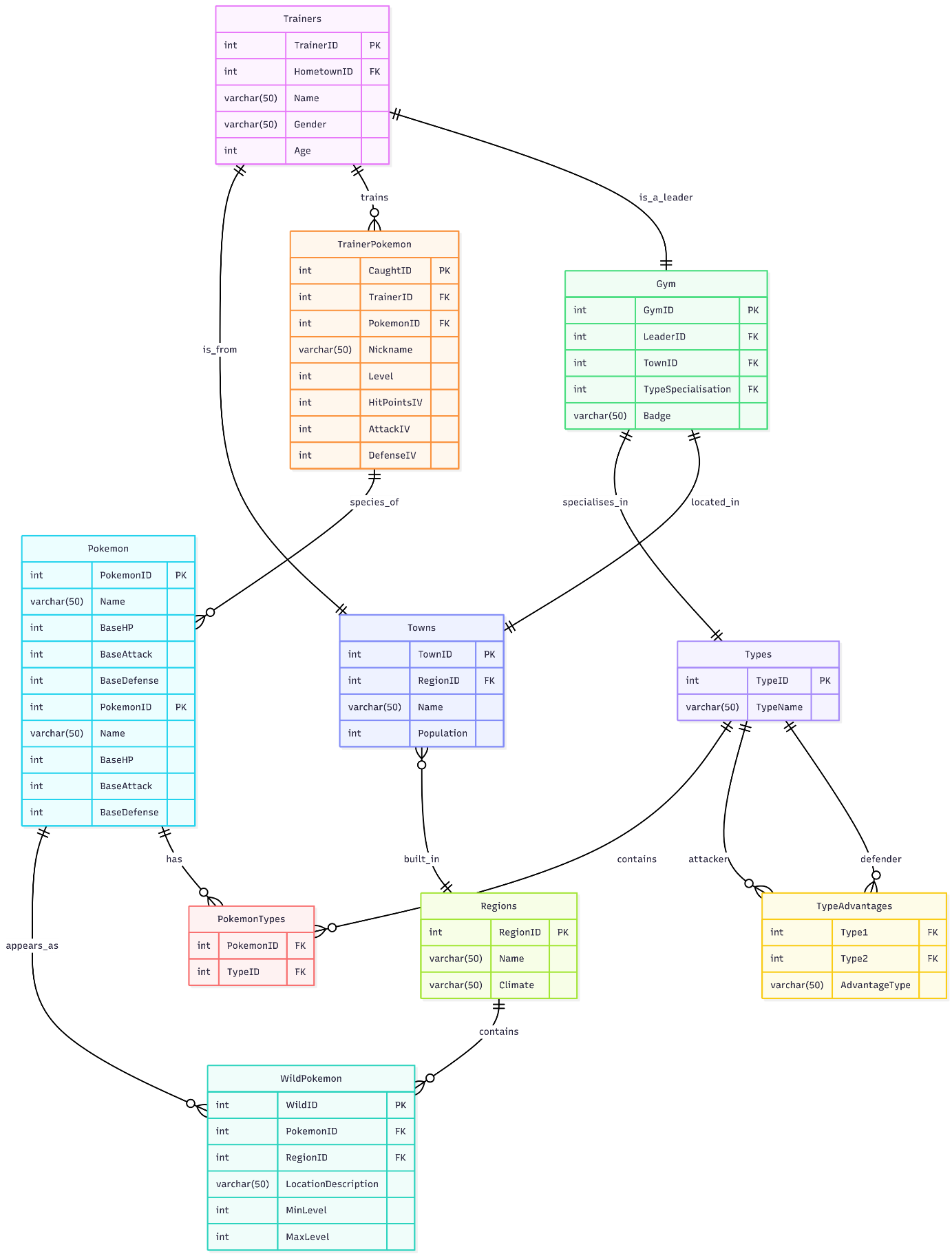
# ER Diagrams

Diagrams presenting first, second and third normal forms and what changed between them

## 1NF Diagram



### What changed

Defined primary keys, made values atomic by separating them into tables instead of having them all in one table

### 1NF Diagram Mermaid.js Code

erDiagram

    Trainers {

        int TrainerID PK

        int HometownID FK

        varchar(50) Name

        varchar(50) Gender

        int Age

    }

    WildPokemon {

        int WildID PK

        int PokemonID FK

        int RegionID FK

        varchar(50) LocationDescription

        int MinLevel

        int MaxLevel

    }

    TrainerPokemon {

        int CaughtID PK

        int TrainerID FK

        int PokemonID FK

        varchar(50) Nickname

        int Level

        int HitPointsIV

        int AttackIV

        int DefenseIV

    }

    Pokemon {

        int PokemonID PK

        varchar(50) Name

        int BaseHP

        int BaseAttack

        int BaseDefense

    }

    Gym {

        int GymID PK

        int LeaderID FK

        int TownID FK

        int TypeSpecialisation FK

        varchar(50) Badge

    }

    Types {

        int TypeID PK

        varchar(50) TypeName

    }

    Pokemon {

        int PokemonID PK

        varchar(50) Name

        int BaseHP

        int BaseAttack

        int BaseDefense

    }

    PokemonTypes {

        int PokemonID FK

        int TypeID FK

    }

    TypeAdvantages {

        int Type1 FK

        int Type2 FK

        varchar(50) AdvantageType

    }

    Towns {

        int TownID PK

        int RegionID FK

        varchar(50) Name

        int Population

    }

    Regions {

        int RegionID PK

        varchar(50) Name

        varchar(50) Climate

    }

    %% Relationships

    Pokemon ||--o{ WildPokemon : appears\_as

    Regions ||--o{ WildPokemon : contains

    Towns }o--|| Regions : built\_in

    Trainers ||--|| Towns : is\_from

    Trainers ||--o{ TrainerPokemon : trains

    TrainerPokemon ||--o{ Pokemon : species\_of

    Pokemon ||--o{ PokemonTypes : has

    Types ||--o{ PokemonTypes : contains

    Types ||--o{ TypeAdvantages : attacker

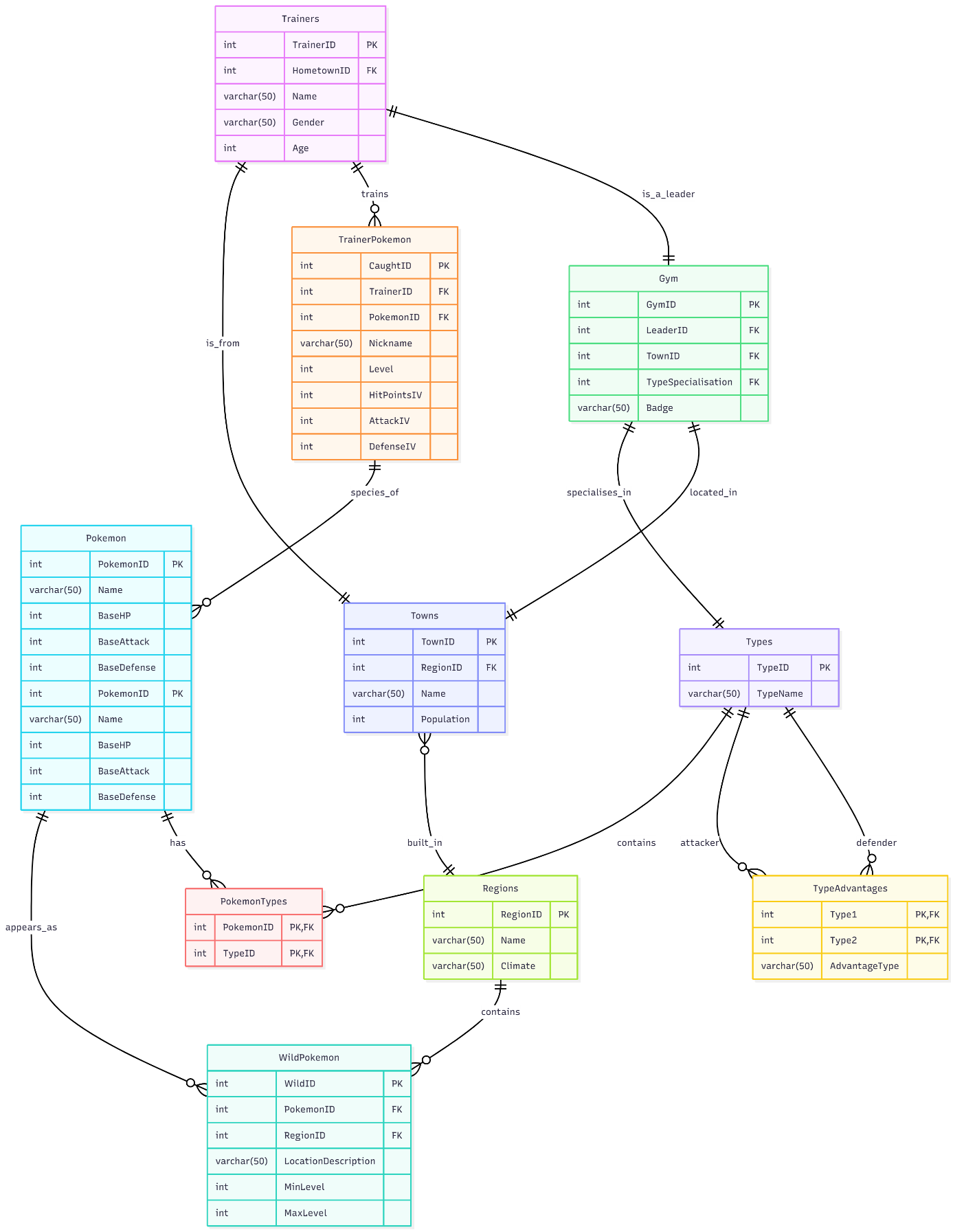
    Types ||--o{ TypeAdvantages : defender

    Gym ||--|| Types : specialises\_in

    Trainers ||--|| Gym : is\_a\_leader

    Gym ||--|| Towns : located\_in

## 2NF Diagram



### What changed

Added composite keys to PokemonTypes and TypeAdvantages tables to ensure they each entry in those depends on complete key (PK, FK in the code states both values form a composite key)

### 2NF Diagram Mermaid.js Code

erDiagram

    Trainers {

        int TrainerID PK

        int HometownID FK

        varchar(50) Name

        varchar(50) Gender

        int Age

    }

    WildPokemon {

        int WildID PK

        int PokemonID FK

        int RegionID FK

        varchar(50) LocationDescription

        int MinLevel

        int MaxLevel

    }

    TrainerPokemon {

        int CaughtID PK

        int TrainerID FK

        int PokemonID FK

        varchar(50) Nickname

        int Level

        int HitPointsIV

        int AttackIV

        int DefenseIV

    }

    Pokemon {

        int PokemonID PK

        varchar(50) Name

        int BaseHP

        int BaseAttack

        int BaseDefense

    }

    Gym {

        int GymID PK

        int LeaderID FK

        int TownID FK

        int TypeSpecialisation FK

        varchar(50) Badge

    }

    Types {

        int TypeID PK

        varchar(50) TypeName

    }

    Pokemon {

        int PokemonID PK

        varchar(50) Name

        int BaseHP

        int BaseAttack

        int BaseDefense

    }

    %% PK, FK states both values form a composite key

    PokemonTypes {

        int PokemonID PK, FK

        int TypeID PK, FK

    }

    %% PK, FK states both values form a composite key

    TypeAdvantages {

        int Type1 PK, FK

        int Type2 PK, FK

        varchar(50) AdvantageType

    }

    Towns {

        int TownID PK

        int RegionID FK

        varchar(50) Name

        int Population

    }

    Regions {

        int RegionID PK

        varchar(50) Name

        varchar(50) Climate

    }

    %% Relationships

    Pokemon ||--o{ WildPokemon : appears\_as

    Regions ||--o{ WildPokemon : contains

    Towns }o--|| Regions : built\_in

    Trainers ||--|| Towns : is\_from

    Trainers ||--o{ TrainerPokemon : trains

    TrainerPokemon ||--o{ Pokemon : species\_of

    Pokemon ||--o{ PokemonTypes : has

    Types ||--o{ PokemonTypes : contains

    Types ||--o{ TypeAdvantages : attacker

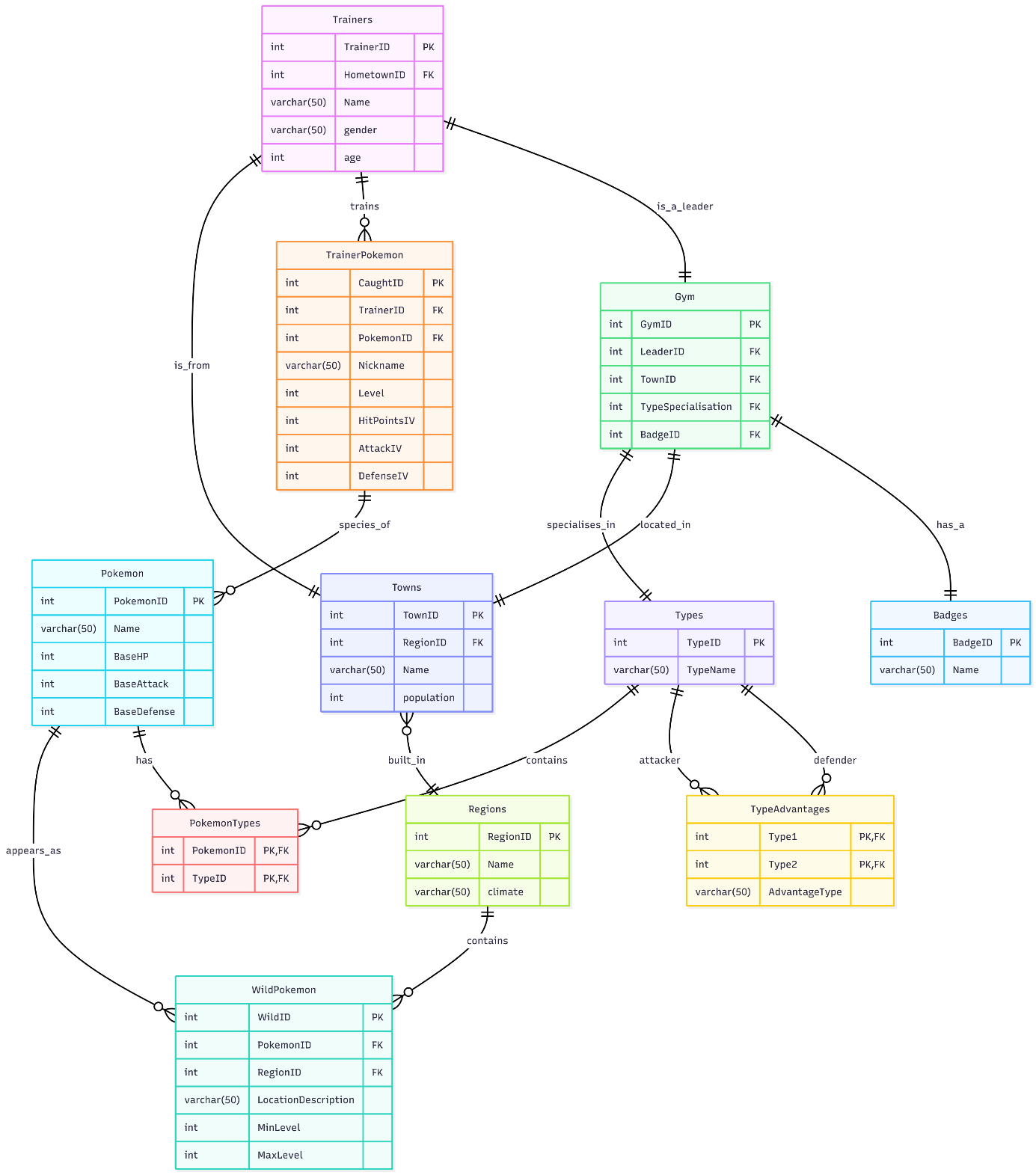
    Types ||--o{ TypeAdvantages : defender

    Gym ||--|| Types : specialises\_in

    Trainers ||--|| Gym : is\_a\_leader

    Gym ||--|| Towns : located\_in

## 3NF Diagram



#### What changed

Removed Transitive dependency between badge and type specialisation in Gym table

Moved badges into separate table

### 3NF Diagram Mermaid.js Code

erDiagram

    Trainers {

        int TrainerID PK

        int HometownID FK

        varchar(50) Name

        varchar(50) gender

        int age

    }

    WildPokemon {

        int WildID PK

        int PokemonID FK

        int RegionID FK

        varchar(50) LocationDescription

        int MinLevel

        int MaxLevel

    }

    TrainerPokemon {

        int CaughtID PK

        int TrainerID FK

        int PokemonID FK

        varchar(50) Nickname

        int Level

        int HitPointsIV

        int AttackIV

        int DefenseIV

    }

    Pokemon {

        int PokemonID PK

        varchar(50) Name

        int BaseHP

        int BaseAttack

        int BaseDefense

    }

    Gym {

        int GymID PK

        int LeaderID FK

        int TownID FK

        int TypeSpecialisation FK

        int BadgeID FK

    }

    Types {

        int TypeID PK

        varchar(50) TypeName

    }

    PokemonTypes {

        int PokemonID PK, FK

        int TypeID PK, FK

    }

    TypeAdvantages {

        int Type1 PK, FK

        int Type2 PK, FK

        varchar(50) AdvantageType

    }

    Towns {

        int TownID PK

        int RegionID FK

        varchar(50) Name

        int population

    }

    Regions {

        int RegionID PK

        varchar(50) Name

        varchar(50) climate

    }

    Badges {

        int BadgeID PK

        varchar(50) Name

    }

    %% Relationships

    Pokemon ||--o{ WildPokemon : appears\_as

    Regions ||--o{ WildPokemon : contains

    Towns }o--|| Regions : built\_in

    Trainers ||--|| Towns : is\_from

    Trainers ||--o{ TrainerPokemon : trains

    TrainerPokemon ||--o{ Pokemon : species\_of

    Pokemon ||--o{ PokemonTypes : has

    Types ||--o{ PokemonTypes : contains

    Types ||--o{ TypeAdvantages : attacker

    Types ||--o{ TypeAdvantages : defender

    Gym ||--|| Types : specialises\_in

    Trainers ||--|| Gym : is\_a\_leader

    Gym ||--|| Towns : located\_in

    Gym ||--|| Badges : has\_a