# Core Schema: The Universal Development Framework

When stripped to its essential foundation, your system represents a powerful, domain-agnostic development framework. Here's the core schema that could serve as a foundation for any vertical:

## 1. Foundational Tables

### The Progression Framework

```sql

-- Domain-agnostic development pillars (was "arc\_pillar")

CREATE TABLE framework\_pillar (

id UUID PRIMARY KEY,

name TEXT NOT NULL,

description TEXT

);

-- Progress phases within pillars (was "arc\_phase")

CREATE TABLE framework\_phase (

id UUID PRIMARY KEY,

pillar\_id UUID REFERENCES framework\_pillar(id),

name TEXT NOT NULL,

intent TEXT,

key\_metric TEXT,

description TEXT

);

-- Desired outcomes within phases (was "arc\_outcome")

CREATE TABLE framework\_outcome (

id UUID PRIMARY KEY,

phase\_id UUID REFERENCES framework\_phase(id),

description TEXT NOT NULL

);

```

### The Entity Structure

```sql

-- Universal entity table (person → entity)

CREATE TABLE entity (

id UUID PRIMARY KEY,

name TEXT NOT NULL,

type TEXT NOT NULL, -- 'individual', 'team', 'department', etc.

current\_phase\_id UUID REFERENCES framework\_phase(id),

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Groups/units (was "team"/"pod")

CREATE TABLE unit (

id UUID PRIMARY KEY,

name TEXT NOT NULL,

unit\_type TEXT NOT NULL, -- 'department', 'squad', 'pod', etc.

current\_phase\_id UUID REFERENCES framework\_phase(id),

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Entity membership in units

CREATE TABLE entity\_unit\_membership (

entity\_id UUID REFERENCES entity(id),

unit\_id UUID REFERENCES unit(id),

role TEXT,

PRIMARY KEY (entity\_id, unit\_id)

);

```

### The Skill/Knowledge Taxonomy

```sql

-- Domain-specific taxonomy (was "skill\_tag")

CREATE TABLE taxonomy\_item (

id UUID PRIMARY KEY,

name TEXT NOT NULL,

description TEXT,

category TEXT,

subcategory TEXT,

parent\_item\_id UUID REFERENCES taxonomy\_item(id),

is\_active BOOLEAN DEFAULT true,

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

```

### Development Plans & Activities

```sql

-- Activity templates (was "drill\_template")

CREATE TABLE activity\_template (

id UUID PRIMARY KEY,

name TEXT NOT NULL,

intent TEXT NOT NULL,

description TEXT,

difficulty\_level INTEGER, -- OCP related

supported\_formats JSONB, -- e.g., group sizes, settings

default\_duration\_minutes INTEGER,

setup\_instructions TEXT,

created\_by\_entity\_id UUID REFERENCES entity(id),

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Activity constraints (was "constraint\_definition")

CREATE TABLE modifier (

id UUID PRIMARY KEY,

name TEXT NOT NULL,

description TEXT,

category TEXT,

effect TEXT, -- 'increase\_challenge', 'decrease\_challenge', 'change\_focus'

is\_active BOOLEAN DEFAULT true,

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Activity template taxonomy mapping

CREATE TABLE activity\_template\_taxonomy (

activity\_template\_id UUID REFERENCES activity\_template(id),

taxonomy\_item\_id UUID REFERENCES taxonomy\_item(id),

PRIMARY KEY (activity\_template\_id, taxonomy\_item\_id)

);

```

### Sessions & Execution

```sql

-- Development session (was "session")

CREATE TABLE development\_session (

id UUID PRIMARY KEY,

unit\_id UUID REFERENCES unit(id),

session\_date DATE NOT NULL,

focus TEXT,

status TEXT DEFAULT 'draft',

created\_by\_entity\_id UUID REFERENCES entity(id),

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Attendance tracking

CREATE TABLE attendance (

id UUID PRIMARY KEY,

development\_session\_id UUID REFERENCES development\_session(id),

entity\_id UUID REFERENCES entity(id),

status TEXT NOT NULL, -- 'present', 'absent', etc.

UNIQUE(development\_session\_id, entity\_id)

);

-- Session blocks (was "session\_block")

CREATE TABLE session\_block (

id UUID PRIMARY KEY,

development\_session\_id UUID REFERENCES development\_session(id),

activity\_template\_id UUID REFERENCES activity\_template(id),

block\_order INTEGER NOT NULL,

duration\_minutes INTEGER NOT NULL,

applied\_modifiers JSONB, -- Was "applied\_constraints"

entity\_adaptations JSONB, -- Was "pdp\_overlays"

challenge\_assessment TEXT, -- OCP assessment

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

```

### Development Plans

```sql

-- Individual development plan (was "pdp")

CREATE TABLE development\_plan (

id UUID PRIMARY KEY,

entity\_id UUID REFERENCES entity(id),

unit\_context\_id UUID REFERENCES unit(id),

start\_date DATE NOT NULL,

end\_date DATE,

overall\_focus TEXT,

status TEXT DEFAULT 'active',

created\_by\_entity\_id UUID REFERENCES entity(id),

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Development plan items

CREATE TABLE development\_plan\_item (

id UUID PRIMARY KEY,

development\_plan\_id UUID REFERENCES development\_plan(id),

item\_type TEXT NOT NULL,

taxonomy\_item\_id UUID REFERENCES taxonomy\_item(id),

modifier\_id UUID REFERENCES modifier(id),

description TEXT NOT NULL,

priority INTEGER,

target\_metric TEXT,

current\_progress TEXT,

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

```

### Reflection & Feedback

```sql

-- Reflection entries

CREATE TABLE reflection (

id UUID PRIMARY KEY,

entity\_id UUID REFERENCES entity(id),

development\_session\_id UUID REFERENCES development\_session(id),

session\_block\_id UUID REFERENCES session\_block(id),

reflection\_text TEXT NOT NULL,

rating JSON, -- Flexible ratings structure

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Optimal Challenge Point log

CREATE TABLE challenge\_point\_log (

id UUID PRIMARY KEY,

entity\_id UUID REFERENCES entity(id),

unit\_id UUID REFERENCES unit(id),

session\_block\_id UUID REFERENCES session\_block(id),

assessed\_by\_entity\_id UUID REFERENCES entity(id),

challenge\_level TEXT NOT NULL, -- 'below\_optimal', 'optimal', 'above\_optimal'

notes TEXT,

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

```

## 2. Core Workflows

The foundational workflows that would work across any vertical:

1. \*\*Growth Framework Management\*\*

- Define development pillars, phases, outcomes

- Map progression pathways

2. \*\*Taxonomy Management\*\*

- Build domain-specific skill/competency hierarchies

- Tag activities with taxonomy items

3. \*\*Development Planning\*\*

- Create individual/unit development plans

- Map taxonomy items to entities

4. \*\*Session Planning & Generation\*\*

- Select appropriate activities based on:

- Current development phase

- Optimal challenge points

- Unit composition

- Apply modifiers to adjust difficulty

5. \*\*Execution & Feedback\*\*

- Track attendance/participation

- Log challenge point assessments

- Capture reflections

- Record progress against plans

## 3. Application Across Verticals

This core schema supports various use cases:

\*\*Sports Coaching (any sport)\*\*

- Entities = Athletes/Teams

- Taxonomy = Sport-specific skills

- Activities = Practice drills

- Modifiers = Game constraints

- Units = Teams/Positional groups

\*\*Business Consulting\*\*

- Entities = Employees/Departments

- Taxonomy = Business competencies

- Activities = Training exercises/Workshops

- Modifiers = Complexity factors

- Units = Departments/Cross-functional teams

\*\*Education\*\*

- Entities = Students/Classes

- Taxonomy = Learning objectives

- Activities = Lesson plans

- Modifiers = Difficulty adjustments

- Units = Classes/Study groups

\*\*Leadership Development\*\*

- Entities = Leaders/Teams

- Taxonomy = Leadership competencies

- Activities = Development exercises

- Modifiers = Context factors

- Units = Management teams

The beauty of this architecture is that it maintains the Development ARC philosophy and Optimal Challenge Point concept while being completely domain-agnostic. The implementation for any specific vertical would involve:

1. Populating the taxonomy with domain-specific skills/competencies

2. Creating activity templates relevant to that domain

3. Defining appropriate modifiers for that context

4. Customizing the UI terminology and workflows

This core schema is essentially a "development operating system" - a flexible framework for optimizing human growth and performance that can be applied to virtually any domain where deliberate practice and progressive development matter.