🗄️ CareerPilot Database Schema

Database: PostgreSQL 15+
ORM: Sequelize v6
Migration Tool: Sequelize CLI

📋 Table of Contents

- Overview
- Database Diagram
- Tables
  - Users
  - Profiles
  - CVs
  - Skills
  - User Skills
  - Career Paths
  - Career Skills
  - Roadmaps
  - Roadmap Tasks
  - Progress
  - Chat Messages
  - Certifications
- Relationships
- Indexes
- Migration Strategy

🎯 Overview

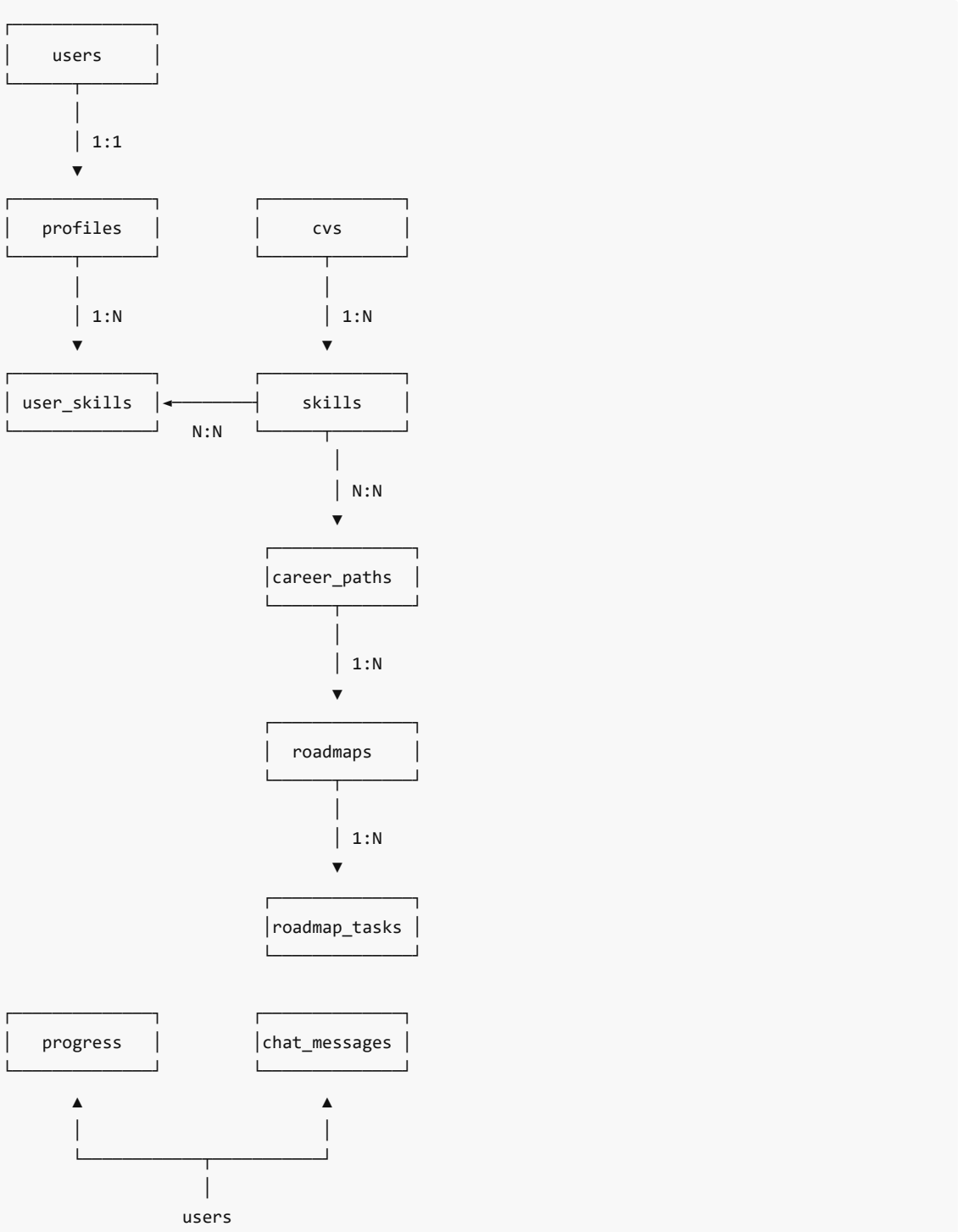
The CareerPilot database is designed to support:

- User authentication and profile management
- CV storage and analysis
- Skill tracking and management
- AI-powered career recommendations
- Personalized learning roadmaps
- Progress tracking and analytics
- AI chatbot conversations

Design Principles:

- Normalized schema (3NF)
- Clear foreign key relationships
- Optimized for read-heavy operations
- Scalable for future features
- ACID compliance

📐 Database Diagram



📊 Tables

1. users

Stores user authentication and basic account information.

```
CREATE TABLE users (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
```

```
    email VARCHAR(255) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL,
    phone VARCHAR(20),
    email_verified BOOLEAN DEFAULT FALSE,
    is_active BOOLEAN DEFAULT TRUE,
    role VARCHAR(20) DEFAULT 'user',
    last_login TIMESTAMP,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_users_email ON users(email);
CREATE INDEX idx_users_is_active ON users(is_active);
```

Columns:

Column	Type	Constraints	Description
`id`	UUID	PRIMARY KEY	Unique user identifier
`email`	VARCHAR(255)	UNIQUE, NOT NULL	User email (login)
`password_hash`	VARCHAR(255)	NOT NULL	Bcrypt hashed password
`first_name`	VARCHAR(100)	NOT NULL	User's first name
`last_name`	VARCHAR(100)	NOT NULL	User's last name
`phone`	VARCHAR(20)		Phone number
`email_verified`	BOOLEAN	DEFAULT FALSE	Email verification status
`is_active`	BOOLEAN	DEFAULT TRUE	Account active status
`role`	VARCHAR(20)	DEFAULT 'user'	User role (user/admin)
`last_login`	TIMESTAMP		Last login timestamp
`created_at`	TIMESTAMP		Account creation date
`updated_at`	TIMESTAMP		Last update timestamp

---

2. profiles

Extended user profile information for career guidance.

```
CREATE TABLE profiles (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    user_id UUID UNIQUE NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    bio TEXT,
    education VARCHAR(255),
    experience VARCHAR(100),
```

```
    current_role VARCHAR(150),
    career_goals TEXT,
    interests TEXT[], -- Array of interests
    linkedin_url VARCHAR(255),
    github_url VARCHAR(255),
    portfolio_url VARCHAR(255),
    location VARCHAR(150),
    preferred_work_type VARCHAR(50), -- remote, hybrid, onsite
    salary_expectation_min INTEGER,
    salary_expectation_max INTEGER,
    availability VARCHAR(50), -- immediate, 2weeks, 1month, etc.
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_profiles_user_id ON profiles(user_id);
```

Columns:

Column	Type	Description
`id`	UUID	Profile ID
`user_id`	UUID	Foreign key to users
`bio`	TEXT	User biography/summary
`education`	VARCHAR(255)	Education background
`experience`	VARCHAR(100)	Years of experience
`current_role`	VARCHAR(150)	Current job title
`career_goals`	TEXT	Career aspirations
`interests`	TEXT[]	Array of interests
`linkedin_url`	VARCHAR(255)	LinkedIn profile
`github_url`	VARCHAR(255)	GitHub profile
`portfolio_url`	VARCHAR(255)	Portfolio website
`location`	VARCHAR(150)	User location
`preferred_work_type`	VARCHAR(50)	Work preference
`salary_expectation_min`	INTEGER	Min salary expectation
`salary_expectation_max`	INTEGER	Max salary expectation
`availability`	VARCHAR(50)	Job availability

---

3. cvs

Stores uploaded CV files and parsed data.

```sql
CREATE TABLE cvs (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    file_name VARCHAR(255) NOT NULL,
    file_path VARCHAR(500) NOT NULL,
    file_size INTEGER NOT NULL,
    file_type VARCHAR(50) NOT NULL, -- pdf, docx
    status VARCHAR(50) DEFAULT 'uploaded', -- uploaded, parsing, parsed, error
    parsed_data JSONB, -- Structured CV data
    parsing_errors TEXT,
    uploaded_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    parsed_at TIMESTAMP,
    is_active BOOLEAN DEFAULT TRUE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_cvs_user_id ON cvs(user_id);
CREATE INDEX idx_cvs_status ON cvs(status);
CREATE INDEX idx_cvs_is_active ON cvs(is_active);
```

Columns:

Column	Type	Description
`id`	UUID	CV record ID
`user_id`	UUID	Foreign key to users
`file_name`	VARCHAR(255)	Original filename
`file_path`	VARCHAR(500)	Storage path
`file_size`	INTEGER	File size in bytes
`file_type`	VARCHAR(50)	File format
`status`	VARCHAR(50)	Processing status
`parsed_data`	JSONB	Extracted CV data
`parsing_errors`	TEXT	Error messages if any
`uploaded_at`	TIMESTAMP	Upload timestamp
`parsed_at`	TIMESTAMP	Parsing completion time
`is_active`	BOOLEAN	Current CV flag

parsed_data JSONB structure:

```json
{
  "personalInfo": {
    "name": "John Doe",
```

```
    "email": "john@example.com",
    "phone": "+1234567890",
    "location": "New York, USA"
  },
  "skills": ["JavaScript", "React", "Node.js"],
  "experience": [
    {
      "title": "Junior Developer",
      "company": "Tech Corp",
      "duration": "2 years",
      "description": "..."
    }
  ],
  "education": [
    {
      "degree": "Bachelor of Computer Science",
      "institution": "State University",
      "year": "2023"
    }
  ],
  "certifications": ["AWS Cloud Practitioner"],
  "languages": ["English", "Spanish"]
}
```

---

4. skills

Master list of all available skills.

```sql
CREATE TABLE skills (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    name VARCHAR(150) UNIQUE NOT NULL,
    category VARCHAR(100) NOT NULL, -- Programming, Framework, Tool, Database, etc.
    description TEXT,
    popularity_score INTEGER DEFAULT 0, -- 0-100
    related_careers TEXT[], -- Array of related career IDs
    icon_url VARCHAR(255),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_skills_name ON skills(name);
CREATE INDEX idx_skills_category ON skills(category);
CREATE INDEX idx_skills_popularity ON skills(popularity_score DESC);
```

Columns:

Column	Type	Description
`id`	UUID	Skill ID
`name`	VARCHAR(150)	Skill name

`category`	VARCHAR(100)	Skill category
`description`	TEXT	Skill description
`popularity_score`	INTEGER	Market demand (0-100)
`related_careers`	TEXT[]	Related career paths
`icon_url`	VARCHAR(255)	Skill icon/logo

---

5. user_skills

Junction table linking users to skills with proficiency levels.

```sql
CREATE TABLE user_skills (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    skill_id UUID NOT NULL REFERENCES skills(id) ON DELETE CASCADE,
    level VARCHAR(50) DEFAULT 'beginner', -- beginner, intermediate, advanced, expert
    source VARCHAR(50), -- manual, cv_parsed, ai_detected
    verified BOOLEAN DEFAULT FALSE,
    years_of_experience DECIMAL(3,1),
    last_used DATE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    UNIQUE(user_id, skill_id)
);

CREATE INDEX idx_user_skills_user_id ON user_skills(user_id);
CREATE INDEX idx_user_skills_skill_id ON user_skills(skill_id);
CREATE INDEX idx_user_skills_level ON user_skills(level);
```

Columns:

Column	Type	Description
`user_id`	UUID	Foreign key to users
`skill_id`	UUID	Foreign key to skills
`level`	VARCHAR(50)	Proficiency level
`source`	VARCHAR(50)	How skill was added
`verified`	BOOLEAN	Skill verification status
`years_of_experience`	DECIMAL	Experience in years
`last_used`	DATE	Last time skill was used

---

6. career_paths

Available career paths and their requirements.

```sql
CREATE TABLE career_paths (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    title VARCHAR(200) NOT NULL,
    description TEXT NOT NULL,
    category VARCHAR(100), -- Development, Data, Design, etc.
    experience_level VARCHAR(50), -- entry, mid, senior
    responsibilities TEXT[],
    required_skills JSONB, -- Technical and soft skills
    optional_skills JSONB,
    education_requirements TEXT[],
    certifications TEXT[],
    salary_range_min INTEGER,
    salary_range_max INTEGER,
    market_demand VARCHAR(50), -- low, medium, high, very_high
    growth_potential VARCHAR(50),
    remote_friendly BOOLEAN DEFAULT FALSE,
    top_employers TEXT[],
    career_progression JSONB,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_career_paths_category ON career_paths(category);
CREATE INDEX idx_career_paths_experience_level ON career_paths(experience_level);
CREATE INDEX idx_career_paths_market_demand ON career_paths(market_demand);
```

---

7. career_skills

Junction table for career path skill requirements.

```sql
CREATE TABLE career_skills (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    career_path_id UUID NOT NULL REFERENCES career_paths(id) ON DELETE CASCADE,
    skill_id UUID NOT NULL REFERENCES skills(id) ON DELETE CASCADE,
    importance VARCHAR(50), -- essential, important, nice_to_have
    minimum_level VARCHAR(50), -- beginner, intermediate, advanced
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    UNIQUE(career_path_id, skill_id)
);

CREATE INDEX idx_career_skills_career ON career_skills(career_path_id);
CREATE INDEX idx_career_skills_skill ON career_skills(skill_id);
```

---

8. roadmaps

Personalized learning roadmaps for users.

```sql
CREATE TABLE roadmaps (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
```

```
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    career_path_id UUID REFERENCES career_paths(id) ON DELETE SET NULL,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    duration_weeks INTEGER NOT NULL, -- Total duration
    intensity VARCHAR(50), -- light, moderate, intensive
    focus_areas TEXT[],
    start_date DATE,
    target_end_date DATE,
    actual_end_date DATE,
    status VARCHAR(50) DEFAULT 'active', -- active, completed, paused, abandoned
    progress_percentage DECIMAL(5,2) DEFAULT 0.00,
    phases JSONB, -- Roadmap structure
    milestones JSONB,
    resources JSONB,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_roadmaps_user_id ON roadmaps(user_id);
CREATE INDEX idx_roadmaps_status ON roadmaps(status);
CREATE INDEX idx_roadmaps_career_path ON roadmaps(career_path_id);
```

phases JSONB structure:

```
[
  {
    "phase": 1,
    "title": "Foundation Month",
    "duration": "4 weeks",
    "weeks": [
      {
        "week": 1,
        "focus": "Node.js Basics",
        "tasks": [...],
        "learningGoals": [...]
      }
    ]
  }
]
```

---

9. roadmap_tasks

Individual tasks within roadmaps.

```
CREATE TABLE roadmap_tasks (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    roadmap_id UUID NOT NULL REFERENCES roadmaps(id) ON DELETE CASCADE,
    phase_number INTEGER NOT NULL,
    week_number INTEGER NOT NULL,
```

```
    title VARCHAR(255) NOT NULL,
    description TEXT,
    type VARCHAR(50), -- course, project, reading, practice
    link VARCHAR(500),
    estimated_hours DECIMAL(5,2),
    actual_hours DECIMAL(5,2),
    priority VARCHAR(50), -- high, medium, low
    status VARCHAR(50) DEFAULT 'pending', -- pending, in_progress, completed, skipped
    completed_at TIMESTAMP,
    notes TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_roadmap_tasks_roadmap_id ON roadmap_tasks(roadmap_id);
CREATE INDEX idx_roadmap_tasks_status ON roadmap_tasks(status);
CREATE INDEX idx_roadmap_tasks_week ON roadmap_tasks(week_number);
```

---

10. progress

Tracks user learning progress and activities.

```
CREATE TABLE progress (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    roadmap_id UUID REFERENCES roadmaps(id) ON DELETE SET NULL,
    activity_type VARCHAR(100) NOT NULL, -- course_completed, project_built,
certification_earned
    activity_details JSONB NOT NULL,
    hours_spent DECIMAL(5,2),
    skills_learned TEXT[],
    date_logged DATE DEFAULT CURRENT_DATE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_progress_user_id ON progress(user_id);
CREATE INDEX idx_progress_roadmap_id ON progress(roadmap_id);
CREATE INDEX idx_progress_date ON progress(date_logged DESC);
CREATE INDEX idx_progress_activity_type ON progress(activity_type);
```

---

11. chat_messages

AI chatbot conversation history.

```
CREATE TABLE chat_messages (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    user_message TEXT NOT NULL,
    ai_response TEXT NOT NULL,
    context_used JSONB, -- What data was used for the response
```

```
    model_used VARCHAR(50), -- gpt-4, gpt-4o-mini, etc.
    tokens_used INTEGER,
    response_time_ms INTEGER,
    user_feedback VARCHAR(20), -- helpful, not_helpful, neutral
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_chat_messages_user_id ON chat_messages(user_id);
CREATE INDEX idx_chat_messages_created_at ON chat_messages(created_at DESC);
```

---

12. certifications

User certifications and achievements.

```
CREATE TABLE certifications (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    user_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    name VARCHAR(255) NOT NULL,
    issuing_organization VARCHAR(255) NOT NULL,
    issue_date DATE,
    expiry_date DATE,
    credential_id VARCHAR(255),
    credential_url VARCHAR(500),
    verified BOOLEAN DEFAULT FALSE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE INDEX idx_certifications_user_id ON certifications(user_id);
CREATE INDEX idx_certifications_expiry ON certifications(expiry_date);
```

🔗 Relationships

One-to-One

- `users` ↔ `profiles` (One user has one profile)

One-to-Many

- `users` → `cvs` (One user has many CVs)
- `users` → `roadmaps` (One user has many roadmaps)
- `users` → `progress` (One user has many progress entries)
- `users` → `chat_messages` (One user has many messages)
- `users` → `certifications` (One user has many certifications)
- `roadmaps` → `roadmap_tasks` (One roadmap has many tasks)
- `career_paths` → `roadmaps` (One career path has many roadmaps)

Many-to-Many

- `users` ↔ `skills` (via `user_skills`)
- `career_paths` ↔ `skills` (via `career_skills`)

🔍 Indexes

Optimized indexes for common queries:

```
-- Authentication queries
CREATE INDEX idx_users_email_password ON users(email, password_hash);

-- Profile lookups
CREATE INDEX idx_profiles_location ON profiles(location);

-- CV searches
CREATE INDEX idx_cvs_user_active ON cvs(user_id, is_active);

-- Skill filtering
CREATE INDEX idx_user_skills_level_verified ON user_skills(level, verified);

-- Roadmap queries
CREATE INDEX idx_roadmaps_user_status ON roadmaps(user_id, status);
CREATE INDEX idx_roadmap_tasks_roadmap_status ON roadmap_tasks(roadmap_id, status);

-- Progress analytics
CREATE INDEX idx_progress_user_date ON progress(user_id, date_logged DESC);

-- Chat history
CREATE INDEX idx_chat_user_date ON chat_messages(user_id, created_at DESC);
```

🚀 Migration Strategy

Initial Setup

```
# Create database
createdb careerpilot_db

# Run migrations
npx sequelize-cli db:migrate

# Seed initial data
npx sequelize-cli db:seed:all
```

Migration Files Structure

```
migrations/
├── 20251118000001-create-users.js
├── 20251118000002-create-profiles.js
├── 20251118000003-create-cvs.js
├── 20251118000004-create-skills.js
├── 20251118000005-create-user-skills.js
├── 20251118000006-create-career-paths.js
```

```
├── 20251118000007-create-career-skills.js
├── 20251118000008-create-roadmaps.js
├── 20251118000009-create-roadmap-tasks.js
├── 20251118000010-create-progress.js
├── 20251118000011-create-chat-messages.js
└── 20251118000012-create-certifications.js
```

Seed Data

```
seeders/
├── 20251118000001-demo-users.js
├── 20251118000002-popular-skills.js
├── 20251118000003-career-paths.js
└── 20251118000004-sample-roadmaps.js
```

🛠️ Database Maintenance

Backup Strategy

```bash
# Daily backup
pg_dump careerpilot_db > backup_$(date +%Y%m%d).sql

# Restore
psql careerpilot_db < backup_20251118.sql
```

Performance Monitoring

```sql
-- Find slow queries
SELECT query, mean_exec_time, calls
FROM pg_stat_statements
ORDER BY mean_exec_time DESC
LIMIT 10;

-- Check table sizes
SELECT relname, pg_size_pretty(pg_total_relation_size(relid))
FROM pg_catalog.pg_statio_user_tables
ORDER BY pg_total_relation_size(relid) DESC;
```

📈 Scaling Considerations

Future Optimizations

- Partition `progress` table by date (monthly partitions)
- Implement read replicas for analytics
- Add caching layer (Redis) for frequently accessed data
- Archive old chat messages (>6 months)
- Implement full-text search (PostgreSQL FTS or Elasticsearch)

Last Updated: November 18, 2025