To transition into React, Python, TypeScript, and a modern database while leveraging your existing knowledge in JavaScript, ASP.NET, and SQL Server, here’s a structured learning path:

**1. Prerequisite: Strengthen JavaScript Fundamentals**

* **Why?** JavaScript is the foundation for React and TypeScript.
* **Topics to Master:**
  + Variables (var, let, const)
  + Data Types (strings, numbers, arrays, objects)
  + Functions (declaration, arrow functions, callbacks)
  + ES6+ Features: Destructuring, Template Literals, Spread/Rest Operators
  + Promises and Async/Await
  + DOM Manipulation and Event Handling
  + Modules and Imports/Exports
* **Resources:**
  + [JavaScript.info](https://javascript.info/)
  + [freeCodeCamp JavaScript](https://www.freecodecamp.org/)

**2. Learn Python (Beginner to Advanced)**

* **Why?** Python is versatile for backend development, data processing, and scripting.
* **Topics to Cover:**
  + Python Basics:
    - Variables, Data Types, Operators
    - Control Flow (if-else, loops)
    - Functions, Modules, and Packages
  + Advanced Python:
    - Object-Oriented Programming (Classes and Objects)
    - Error Handling (try-except)
    - File Handling (read/write files)
    - Python Libraries: os, sys, collections, etc.
  + Web Development:
    - Learn Flask (lightweight framework) or Django (full-stack framework)
    - Building REST APIs
    - Template Rendering (Jinja2 in Flask)
  + Database Integration:
    - Use **SQLite** or **PostgreSQL** with Python’s SQLAlchemy or Django ORM.
* **Resources:**
  + [Real Python](https://realpython.com/)
  + [Automate the Boring Stuff with Python](https://automatetheboringstuff.com/)

**3. Learn React.js (Beginner to Advanced)**

* **Why?** React is widely used for building modern, scalable frontend applications.
* **Topics to Cover:**
  + React Basics:
    - Components, JSX
    - Props and State
    - Event Handling
  + React Advanced:
    - Lifecycle Methods
    - React Hooks (useState, useEffect, useContext)
    - Context API for State Management
  + Routing:
    - React Router (navigation and URL parameters)
  + Advanced Topics:
    - Redux or Zustand for State Management
    - Error Boundaries
    - Code Splitting and Lazy Loading
  + Integration with Backend:
    - Fetching Data with fetch or axios
    - Handling Authentication
* **Resources:**
  + [React Docs](https://reactjs.org/)
  + [Scrimba React Course](https://scrimba.com/learn/learnreact)

**4. Learn TypeScript (Beginner to Advanced)**

* **Why?** TypeScript adds static typing to JavaScript, improving code maintainability and scalability.
* **Topics to Cover:**
  + Basics of TypeScript:
    - Type Annotations (variables, functions)
    - Interfaces and Types
    - Union and Intersection Types
  + Advanced TypeScript:
    - Generics
    - Utility Types (Partial, Pick, Omit)
    - Type Narrowing and Guards
  + TypeScript with React:
    - Typing Props and State
    - Using useRef, useReducer with TypeScript
* **Resources:**
  + [TypeScript Docs](https://www.typescriptlang.org/docs/)
  + [Academind TypeScript Tutorial](https://academind.com/)

**5. Learn a Modern Database (PostgreSQL or MongoDB)**

* **Why?** Modern databases like **PostgreSQL** (SQL-based) and **MongoDB** (NoSQL) are widely used and complement Python and React.
* **PostgreSQL**:
  + Installation and Basic Commands
  + Writing SQL Queries
  + Transactions and Indexing
  + Advanced Topics:
    - Functions and Triggers
    - Using psycopg2 or SQLAlchemy with Python
* **MongoDB**:
  + Basics: Collections, Documents, and CRUD Operations
  + Schema Design
  + Indexing and Aggregations
  + Using mongoose or Python's pymongo for integration
* **Resources:**
  + [PostgreSQL Tutorial](https://www.postgresqltutorial.com/)
  + [MongoDB University](https://university.mongodb.com/)

**6. Build Projects (Hands-On Learning)**

* Start small and gradually increase complexity:
  1. **Frontend with React**:
     + To-Do List App
     + Weather App (fetch API)
  2. **Backend with Python**:
     + REST API for a Blog
     + User Authentication System
  3. **Full-Stack**:
     + Task Manager App (React + Flask/Django + PostgreSQL)
     + E-commerce App (React + Django + MongoDB/PostgreSQL)
* Use GitHub for version control and showcase your projects.

**7. Cloud and Deployment**

* Learn to deploy your applications:
  + **Frontend**: Use **Netlify** or **Vercel**.
  + **Backend**: Deploy APIs with **Heroku** or **AWS Lambda**.
  + **Database**: Host databases on **AWS RDS** or **MongoDB Atlas**.

**Timeline**

* **Month 1-2**: Strengthen JavaScript and Start Python Basics.
* **Month 3-4**: Learn React Basics and Flask/Django.
* **Month 5-6**: Advance in TypeScript and React, Integrate Backend.
* **Month 7**: Learn PostgreSQL/MongoDB.
* **Month 8-9**: Build and Deploy Full-Stack Projects.

Let me know if you need specific tutorials, project ideas, or further guidance on any step!