Drizzle ORM

- Prerequisites
- 1. Install Drizzle
- 2. Configure Drizzle
  - 2.1. Define a schema
  - 2.2. Connect Drizzle ORM to the database with Hyperdrive
  - 2.3. Configure Drizzle-Kit for migrations (optional)
- 3. Deploy your Worker

import { Render, Steps } from "~/components";

Drizzle ORM is a lightweight TypeScript ORM with a focus on type safety. This example demonstrates how to use Drizzle ORM with PostgreSQL via Cloudflare Hyperdrive in a Workers application.

Prerequisites

- A Cloudflare account with Workers access
- A PostgreSQL database
- A Hyperdrive configuration to your PostgreSQL database

1. Install Drizzle

Install the Drizzle ORM and its dependencies such as the node-postgres (`pg`) driver:

```
npm i drizzle-orm pg dotenv
npm i -D drizzle-kit tsx @types/pg @types/node
```

Add the required Node.js compatibility flags and Hyperdrive binding to your `wrangler.jsonc` file:

2. Configure Drizzle

2.1. Define a schema

With Drizzle ORM, we define the schema in TypeScript rather than writing raw SQL.

1. Create a folder `/db/` in `/src/`. 2. Create a `schema.ts` file. 3. In `schema.ts`, define a `users` table as shown below.

```ts title="src/db/schema.ts"
// src/db/schema.ts
import { pgTable, serial, varchar, timestamp } from "drizzle-orm/pg-core";

export const users = pgTable("users", {
id: serial("id").primaryKey(),
name: varchar("name", { length: 255 }).notNull(),
email: varchar("email", { length: 255 }).notNull().unique(),
createdAt: timestamp("created_at").defaultNow(),
});
```

2.2. Connect Drizzle ORM to the database with Hyperdrive

Use your Hyperdrive configuration for your database when using the Drizzle ORM.

Populate your `index.ts` file as shown below.

```
// src/index.ts
import { Client } from "pg";
import { drizzle } from "drizzle-orm/node-postgres";
import { users } from "./db/schema";

export interface Env {
HYPERDRIVE: Hyperdrive;
}

export default {
```

```
	async fetch(request, env, ctx): Promise<Response> {
		// Create a new client instance for each request.
		const client = new Client({
			connectionString: env.HYPERDRIVE.connectionString,
		});

		// Connect to the database
		await client.connect();

		// Create the Drizzle client with the node-postgres connection
		const db = drizzle(client);

		// Sample query to get all users
		const allUsers = await db.select().from(users);

		return Response.json(allUsers);
	},
} satisfies ExportedHandler<Env>;
```

:::note Use node-postgres with Drizzle ORM when connecting through Hyperdrive. :::

:::caution[Known issue with Postgres.js] Pairing Drizzle ORM with the Postgres.js driver over Hyperdrive is not currently supported. Switch your driver to node-postgres (`pg`), the recommended driver for Drizzle ORM with Hyperdrive. :::

2.3. Configure Drizzle-Kit for migrations (optional)

:::note You need to set up the tables in your database so that Drizzle ORM can make queries that work.

If you have already set it up (for example, if another user has applied the schema to your database), or if you are starting to use Drizzle ORM and the schema matches what already exists in your database, then you do not need to run the migration. :::

You can generate and run SQL migrations on your database based on your schema using Drizzle Kit CLI. Refer to Drizzle ORM docs for additional guidance.

1. Create a `.env` file the root folder of your project, and add your database connection string. The Drizzle Kit CLI will use this connection string to create and apply the migrations.

```
# .env
# Replace with your direct database connection string
DATABASE_URL='postgres://user:password@db-host.cloud/database-name'
```

2. Create a `drizzle.config.ts` file in the root folder of your project to configure Drizzle Kit and add the following content:

```
// drizzle.config.ts
import "dotenv/config";
import { defineConfig } from "drizzle-kit";
export default defineConfig({
  out: "./drizzle",
  schema: "./src/db/schema.ts",
  dialect: "postgresql",
  dbCredentials: {
    url: process.env.DATABASE_URL!,
  },
});
```

3. Generate the migration file for your database according to your schema files and apply the migrations to your database.
   Run the following two commands:

```
npx drizzle-kit generate
```

```
No config path provided, using default 'drizzle.config.ts'
Reading config file 'drizzle.config.ts'
1 tables
users 4 columns 0 indexes 0 fks

[✓] Your SQL migration file ➜ drizzle/0000_mysterious_queen_noir.sql
```

```
npx drizzle-kit migrate
```

```
No config path provided, using default 'drizzle.config.ts'
Reading config file 'drizzle.config.ts'
Using 'postgres' driver for database querying
```

3. Deploy your Worker

Deploy your Worker.

```
npx wrangler deploy
```

□□□ #2
□ □□□ □□□ 2026-11-01 02:30:03 UTC
□ □□□ □□□ 2026-11-01 02:30:04 UTC