Prisma Database ORM toolkit

START HERE

`npx prisma init`
Scaffold schema and config

`npx prisma migrate dev --name` *`<name>`*
Create and apply a migration

`npx prisma generate`
Build the client

`npx prisma db push`
Sync schema, no migration

`npx prisma migrate deploy`
Apply migrations in prod

`npx prisma studio`
Browse data in a browser

`npx prisma format`
Tidy the schema file

`npx prisma migrate status`
Is the database current?

CONCEPTS

Since Prisma 7 the CLI no longer runs generate or seed for you after migrate dev, migrate reset or db push. Run them explicitly.

Edit schema.prisma models, relations
migrate dev SQL file + apply
↓ changes
generate typed client
↓ then
migrate deploy CI and production
↓ commit, ship
Typical development loop

The database URL lives in prisma.config.ts (usually read from DATABASE_URL), not in the schema. Commands that touch the database need that file.

SETUP

`npx prisma init`
Create schema, config and .env

`npx prisma init --datasource-provider` *`<provider>`*
Pick the database provider

`npx prisma init --url` *`<url>`*
Set a custom datasource URL

`npx prisma init --with-model`
Add a sample User model

`npx prisma init --output` *`<path>`*
Where the client is generated

`npx prisma version`
CLI, client and engine versions

`npx prisma version --json`
Same, machine readable

`npx prisma debug`
Diagnostics for a bug report

GENERATE

`npx prisma generate`
Generate the client from the schema

`npx prisma generate --watch`
Regenerate on schema changes

`npx prisma generate --generator` *`<name>`*
Run only one generator

GENERATE (cont.)

`npx prisma generate --no-engine`
Client without an engine (Accelerate, edge)

`npx prisma generate --no-hints`
Suppress hints and banners

`npx prisma generate --allow-no-models`
Generate even with no models

SCHEMA TOOLING

`npx prisma validate`
Check the schema for errors

`npx prisma format`
Format schema.prisma

`npx prisma format --check`
Fail if the schema is unformatted

`npx prisma` *`<command>`* **`--schema`** *`<path>`*
Use a schema in another location

`npx prisma` *`<command>`* **`--config`** *`<path>`*
Use another prisma.config.ts
Documented for db execute, migrate diff and studio.

MIGRATE

`npx prisma migrate dev`
Apply schema changes in development
Uses a shadow database, detects drift, and may offer a reset.

`npx prisma migrate dev --name` *`<name>`*
Name the new migration

`npx prisma migrate dev --create-only`
Write the SQL, do not apply it

`npx prisma migrate deploy`
Apply pending migrations (prod, CI)

`npx prisma migrate status`
Compare migrations with the database
Exits with 1 on unapplied, failed or diverged migrations, handy in CI.

`npx prisma migrate reset`
Drop data and reapply all migrations
Development only. Asks for confirmation unless --force.

MIGRATE (cont.)

`npx prisma migrate reset --force`
Reset without the prompt

`npx prisma migrate resolve --applied` *`<migration>`*
Mark a migration as applied

`npx prisma migrate resolve --rolled-back` *`<migration>`*
Mark a failed migration as rolled back

`npx prisma migrate diff --from-`*`<src>`* **`--to-`***`<src>`*
Compare two schema states
Sources: -empty, -schema, -migrations, -config-datasource.

`npx prisma migrate diff --from-config-datasource --to-schema` *`<file>`* **`--script`**
SQL to move DB to schema

`npx prisma migrate diff ... --exit-code`
Exit 2 when there are changes

`npx prisma migrate diff ... -o` *`<file>`*
Write the diff to a file

DATABASE

`npx prisma db push`
Sync schema to DB without migrations

`npx prisma db push --accept-data-loss`
Proceed despite data loss

`npx prisma db push --force-reset`
Reset the database first

`npx prisma db pull`
Introspect DB into the schema
Overwrites manual edits to the schema file.

`npx prisma db pull --print`
Print the schema, do not write

`npx prisma db pull --force`
Discard manual schema changes

`npx prisma db seed`
Run the configured seed script

`npx prisma db seed --` *`<args>`*
Pass arguments to the seed file

`npx prisma db execute --file` *`<file.sql>`*
Run a SQL file on the database

`npx prisma db execute --stdin`
Run SQL from standard input

STUDIO AND DEV DB

`npx prisma studio`
Open the data browser

`npx prisma studio --port` *`<port>`*
Change the port (default 5555)

`npx prisma studio --browser none`
Do not open a browser

`npx prisma studio --url` *`<url>`*
Override the config database URL

`npx prisma dev`
Start a local Prisma Postgres

`npx prisma dev --detach`
Run it in the background

`npx prisma dev ls`
List local instances

`npx prisma dev stop` *`<glob>`*
Stop matching instances

`npx prisma dev rm` *`<glob>`*
Delete instance data

ENVIRONMENT

`DATABASE_URL=`*`<url>`*
Connection string, read by prisma.config.ts

`DEBUG="prisma:*"`
Verbose logging

`NO_COLOR=1`
Disable coloured output

RECIPES

New project on Postgres

```
npm install -D prisma
npx prisma init --datasource-provider postgresql
# set DATABASE_URL in .env, add models
npx prisma migrate dev --name init
npx prisma generate
```

Adopt an existing database

```
npx prisma init --url <url>
npx prisma db pull
npx prisma generate
```

Deploy step in CI

```
npx prisma migrate deploy
npx prisma generate
# fail the build on drift:
npx prisma migrate status
```

Review SQL before applying

```
npx prisma migrate dev --create-only --name <name>
# edit prisma/migrations/<ts>_<name>/migration.sql
npx prisma migrate dev
```