goose Database migrations

CONCEPTS

Set GOOSE_DRIVER, GOOSE_DBSTRING and GOOSE_MIGRATION_DIR (or a .env file) and drop the driver and dbstring from every command.

create new .sql file
↓ edit
Up / Down SQL annotations
↓ goose up
Database goose_db_version
↓ goose status
Verified applied list

Each file needs exactly one '-- +goose Up' marker; Down is optional but must come after Up.

CONNECT

goose *<driver>* *<dbstring>* *<command>*
Full invocation form

goose -dir *<path>* *<command>*
Migration directory (default .)

export GOOSE_DRIVER=*<driver>*
Driver from environment

export GOOSE_DBSTRING=*<dbstring>*
Connection string from environment

export GOOSE_MIGRATION_DIR=*<path>*
Directory from environment

goose -env *<file>* *<command>*
Load variables from another env file

goose -table *<name>* *<command>*
Custom version table (schema.table ok)

goose -timeout *<duration>* *<command>*
Cap query duration, e.g. 1h13m

goose -v *<command>*
Verbose output

goose -version
Print goose version

goose -h
Help and driver list

DBSTRING EXAMPLES

Driver and connection string

driver	dbstring
sqlite3	./app.db
postgres	"user=postgres dbname=app sslmode=disable"
mysql	"user:pass@/app?parseTime=true"
mssql	"sqlserver://user:pass@host:1433?database=app"
clickhouse	"tcp://127.0.0.1:9000"
redshift	"postgres://user:pass@host:5439/db"

Other drivers: tidb, starrocks, spanner, azuresql, ydb, turso. MySQL needs parseTime=true for status, and multiStatements=true to run several statements per file.

CREATE

goose create *<name>* **sql**
New SQL migration

goose create *<name>* **go**
New Go migration

goose -s create *<name>* **sql**
Sequential number, not timestamp

goose -dir *<path>* **create** *<name>* **sql**
Create in a given directory

goose fix
Renumber timestamps to sequential

goose validate
Check files without running them

SQL ANNOTATIONS

-- +goose Up
Start of forward statements

-- +goose Down
Start of rollback statements

-- +goose NO TRANSACTION
Run outside a transaction (file top)

-- +goose StatementBegin
Open a block with inner semicolons

-- +goose StatementEnd
Close that block

-- +goose ENVSUB ON
Expand ${VAR} in the SQL below

SQL ANNOTATIONS (cont.)

Every statement must end with a semicolon. Functions and triggers need a StatementBegin/End pair.

MIGRATE

goose up
Apply all pending migrations

goose up-by-one
Apply just the next one

goose up-to *<version>*
Migrate up to a version

goose down
Roll back the latest one

goose down-to *<version>*
Roll back to a version

goose redo
Re-run the latest migration

goose reset
Roll back all migrations

goose -allow-missing up
Apply out-of-order migrations

INSPECT

goose status
List migrations and applied time

goose version
Print current DB version

goose env
Print goose environment

GO MIGRATIONS

goose.AddMigration(Up, Down)
Register in init()

func Up(tx *sql.Tx) error
Migration function signature

Go migrations need your own goose binary that imports the migrations package; the stock CLI only runs SQL files.

INSTALL

go install github.com/pressly/goose/v3/cmd/goose@latest
Install with Go