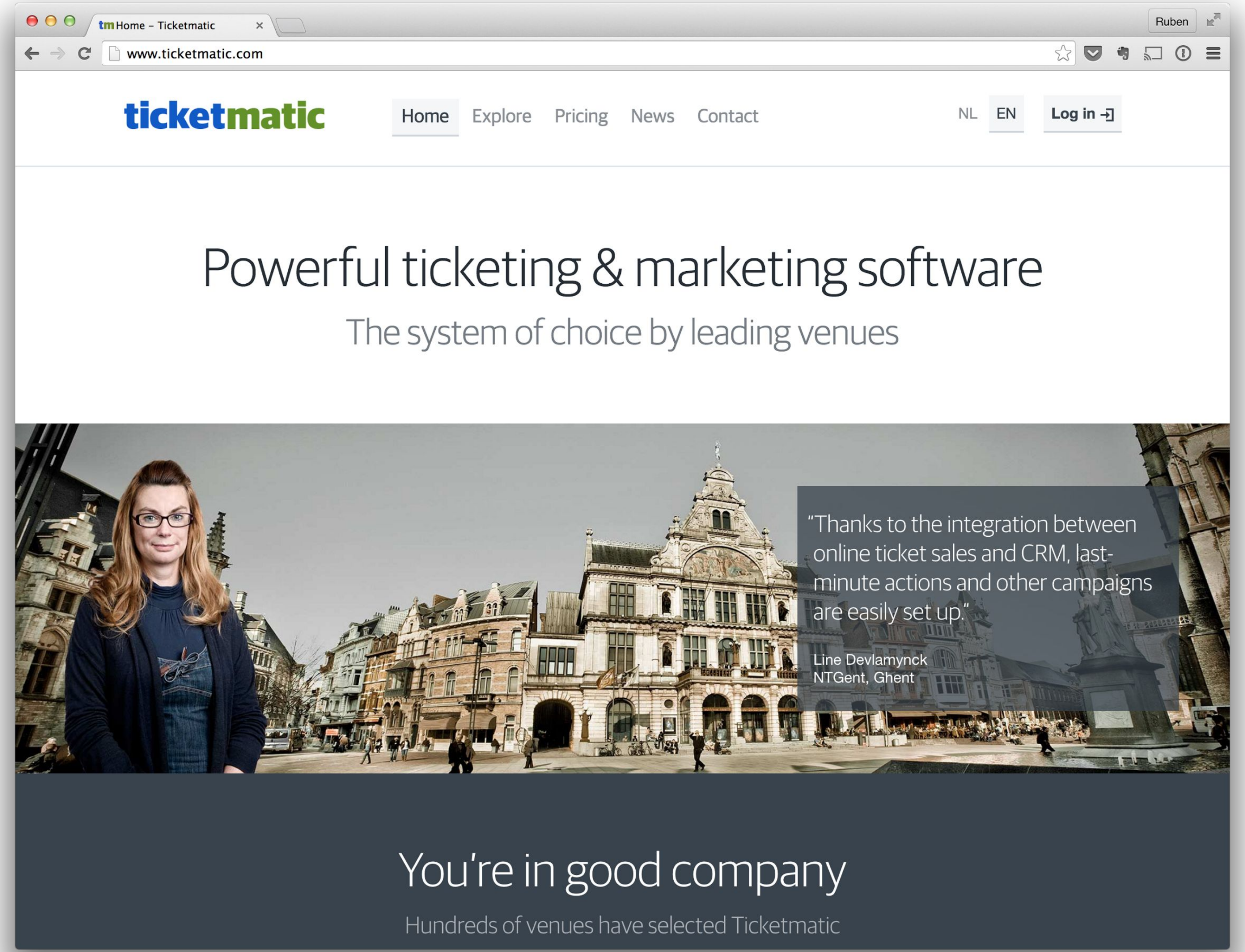
sql-migrate

@rubenv



Powerful ticketing & marketing software

The system of choice by leading venues

"Thanks to the integration between online ticket sales and CRM, last-minute actions and other campaigns are easily set up."

Line Devlamynck
NTGent, Ghent

You're in good company

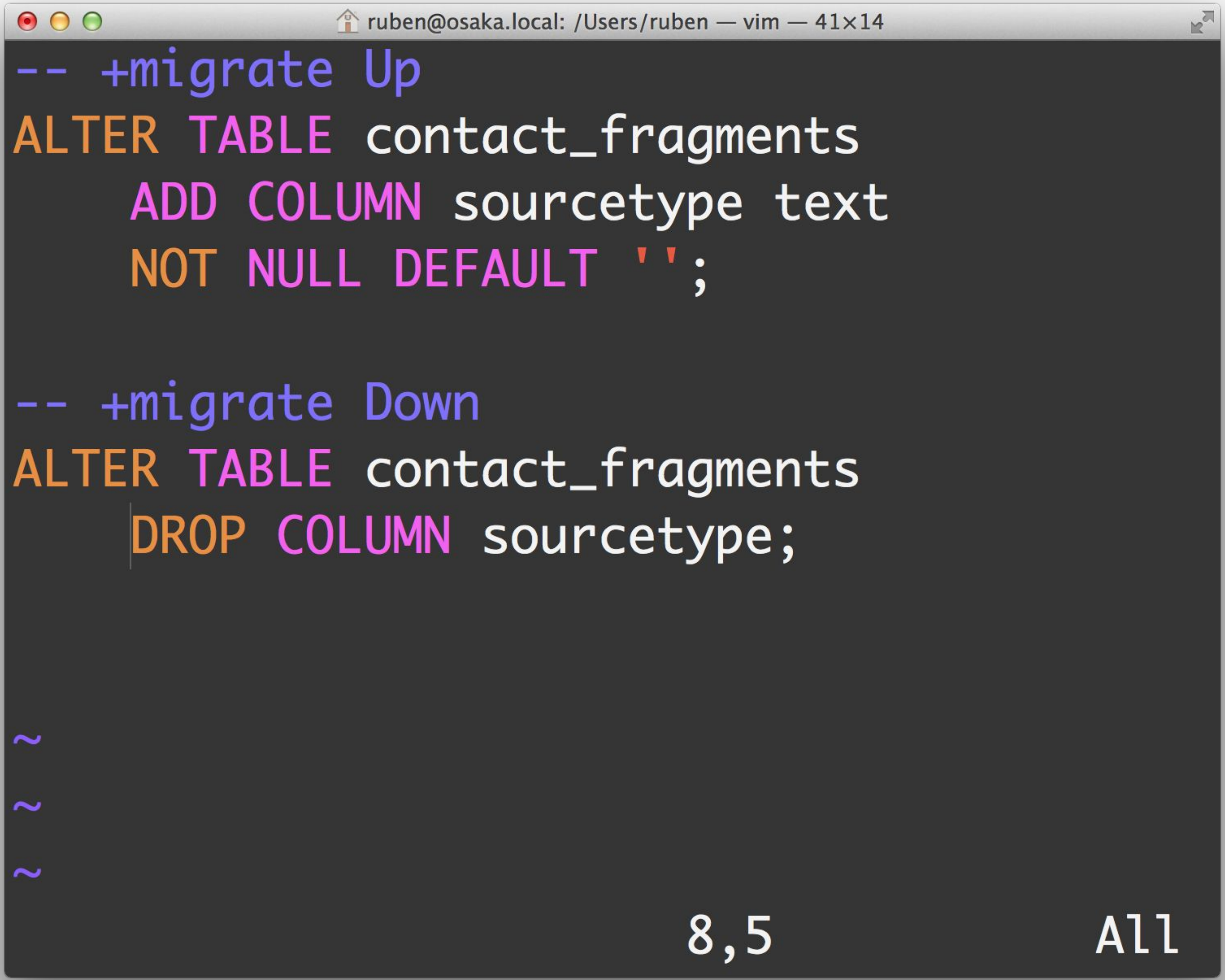
Hundreds of venues have selected Ticketmatic

SQL is awesome

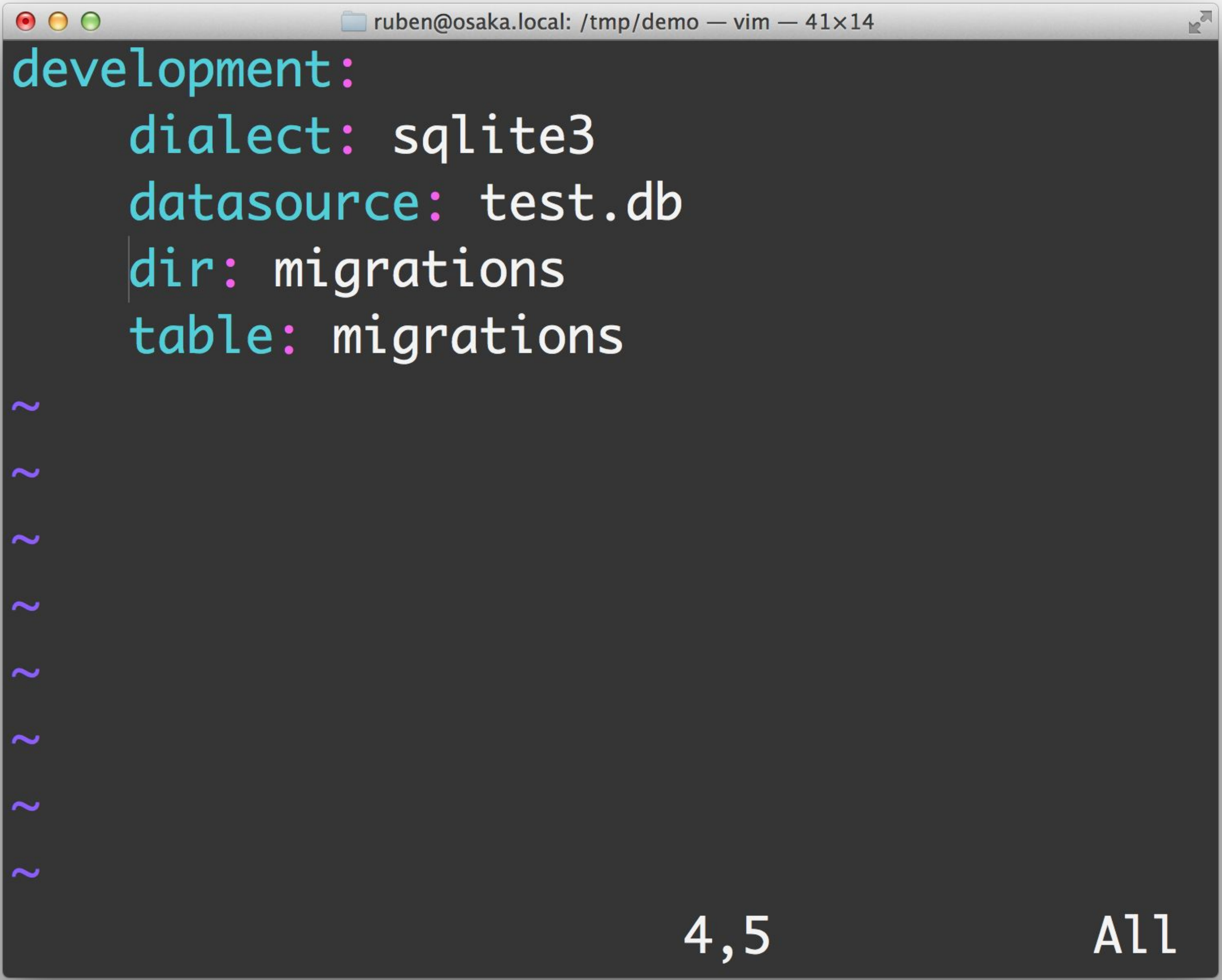
SQL ~~is~~ awesome

can be

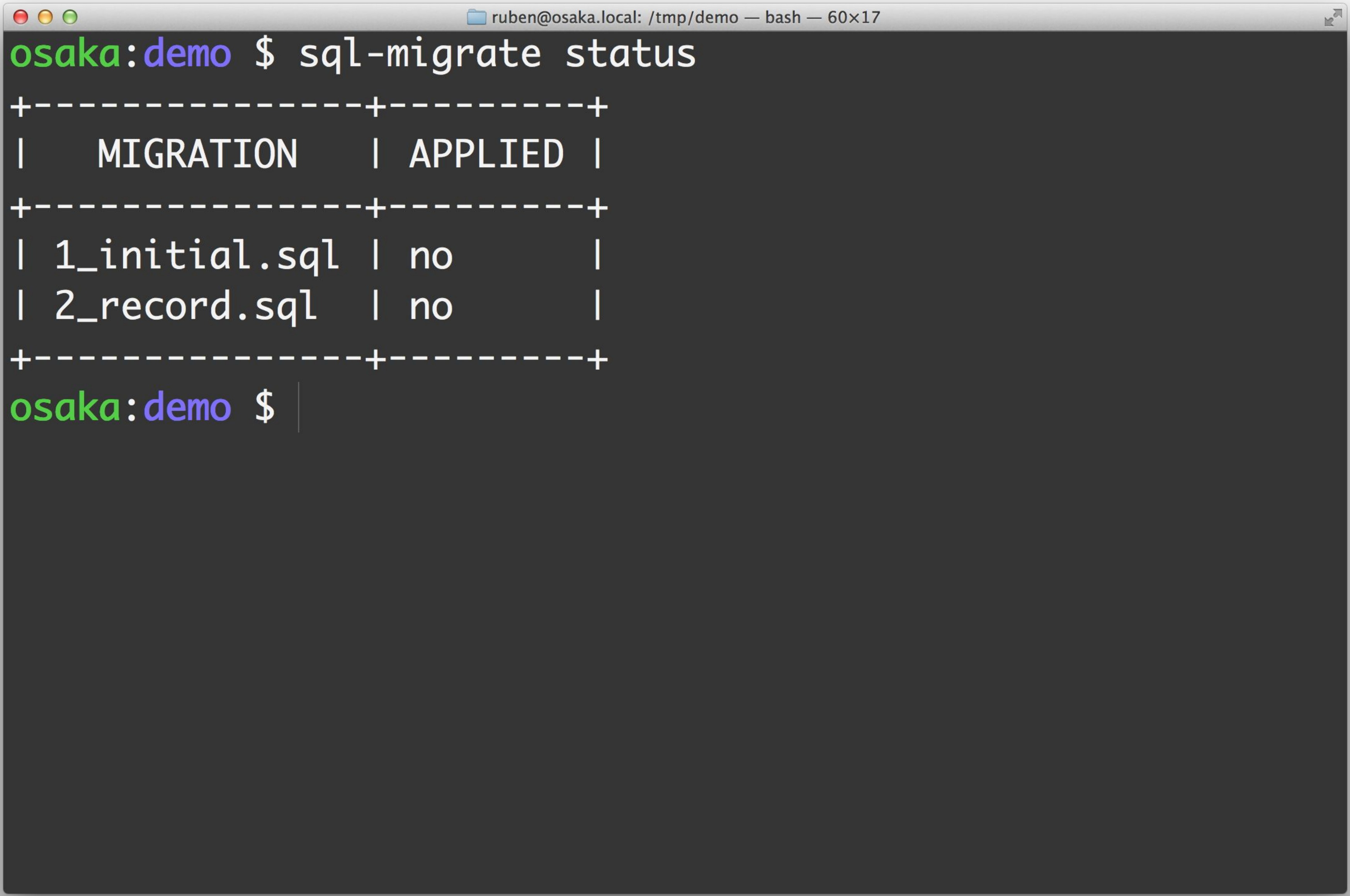
Schema migrations



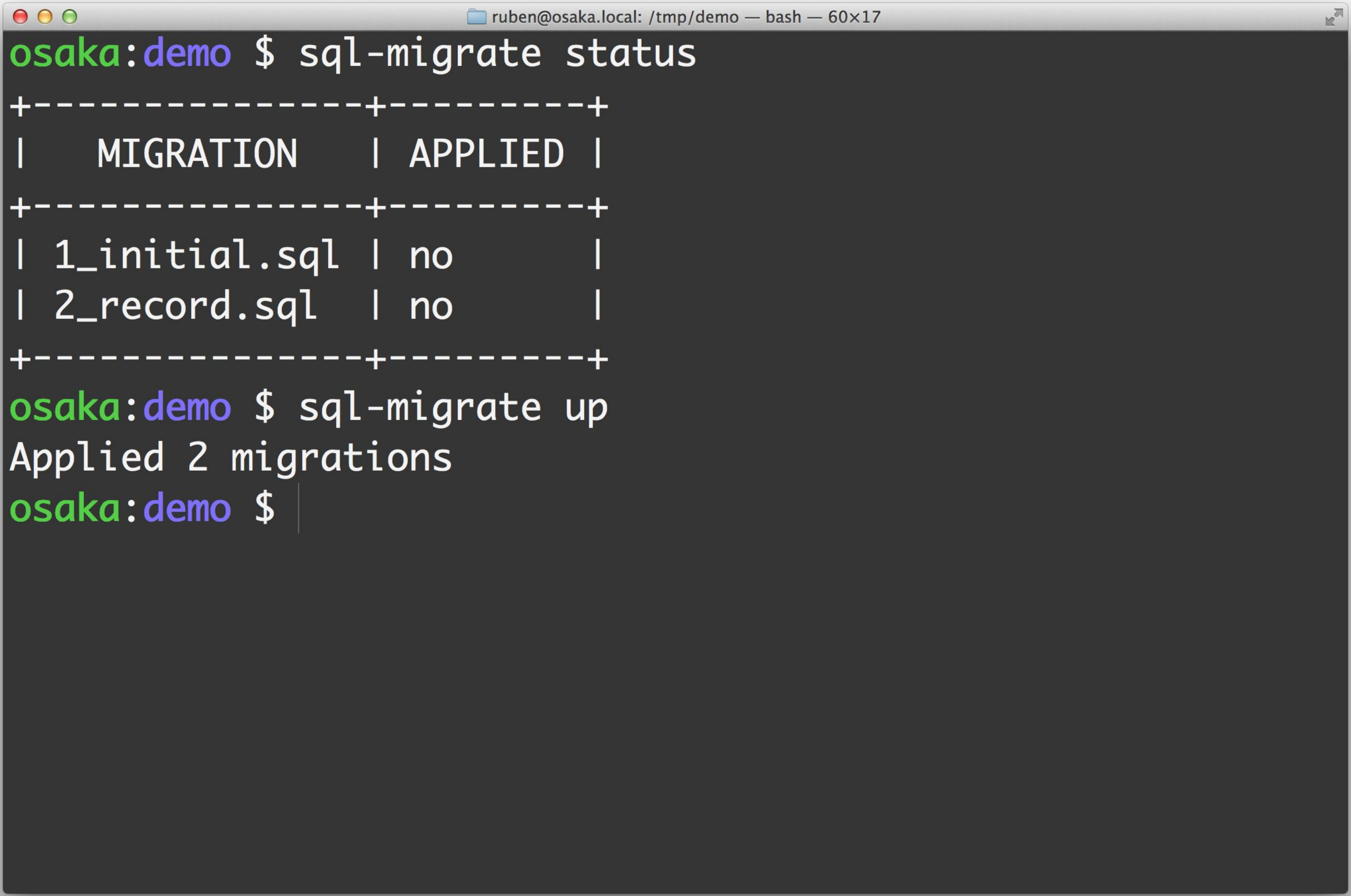
```
-- +migrate Up
ALTER TABLE contact_fragments
    ADD COLUMN sourcetype text
    NOT NULL DEFAULT '';

-- +migrate Down
ALTER TABLE contact_fragments
    DROP COLUMN sourcetype;



~
~
~
                            8,5          All
```


ruben@osaka.local: /tmp/demo — vim — 41×14

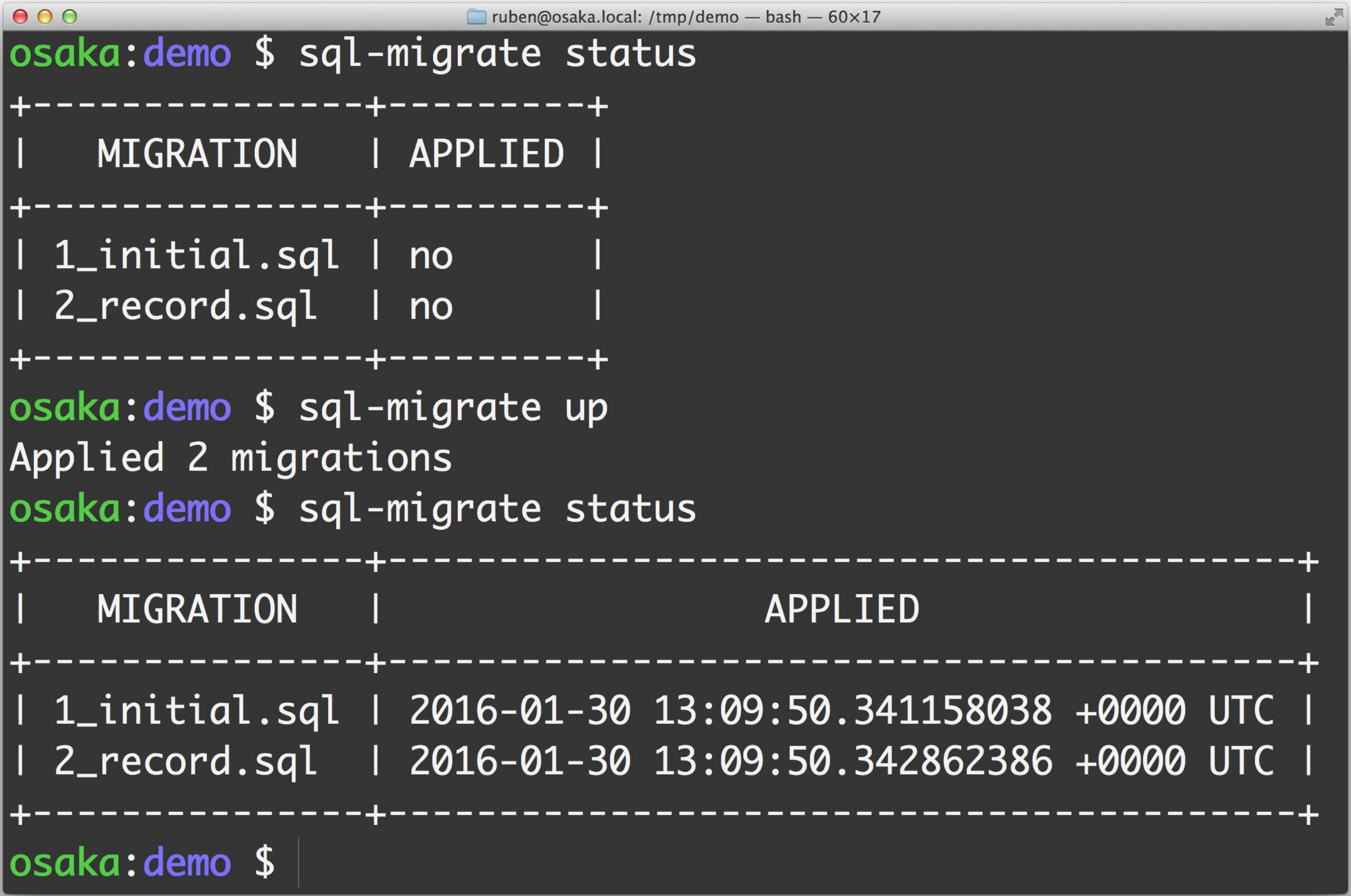
```
development:
    dialect: sqlite3
    datasource: test.db
    dir: migrations
    table: migrations
~
~
~
~
~
~
~
~
                        4,5           All
```



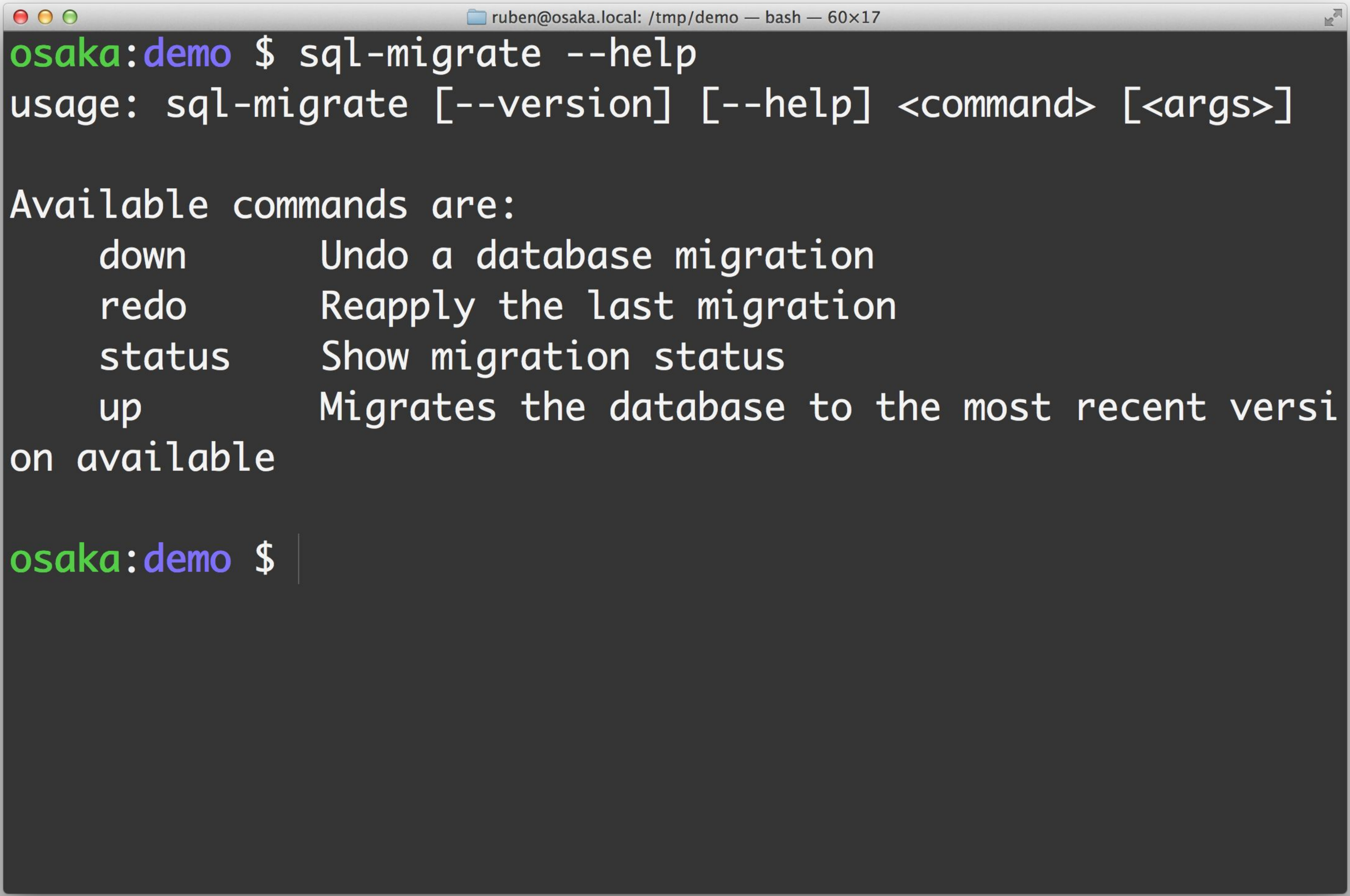
```
osaka:demo $ sql-migrate status
+----------------+---------+
|   MIGRATION    | APPLIED |
+----------------+---------+
| 1_initial.sql  | no      |
| 2_record.sql   | no      |
+----------------+---------+
osaka:demo $
```


ruben@osaka.local: /tmp/demo — bash — 60×17

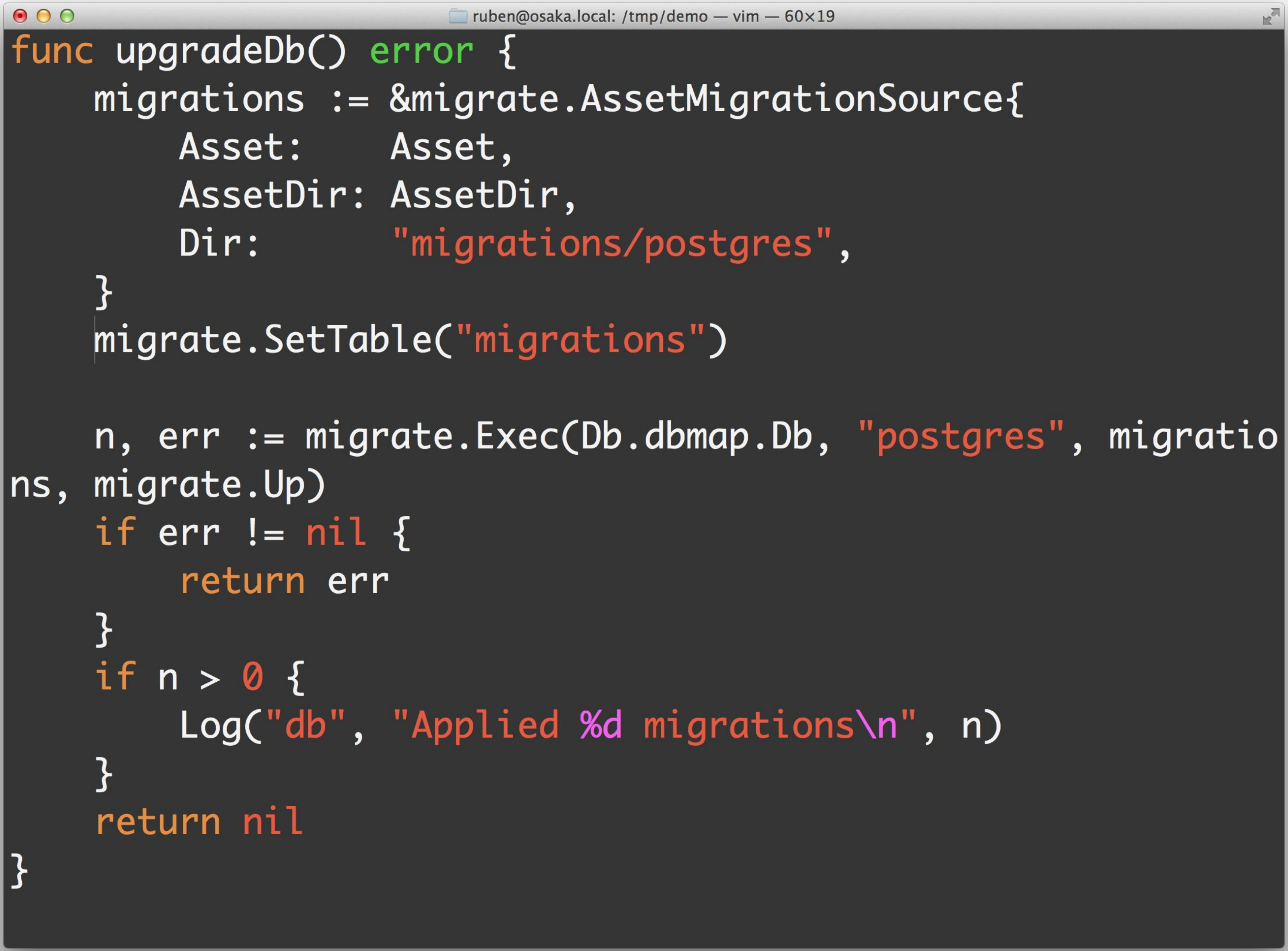
```
osaka:demo $ sql-migrate status
+---------------+---------+
|   MIGRATION   | APPLIED |
+---------------+---------+
| 1_initial.sql | no      |
| 2_record.sql  | no      |
+---------------+---------+
osaka:demo $ sql-migrate up
Applied 2 migrations
osaka:demo $
```



```
osaka:demo $ sql-migrate status
+---------------+---------+
|   MIGRATION   | APPLIED |
+---------------+---------+
| 1_initial.sql | no      |
| 2_record.sql  | no      |
+---------------+---------+
osaka:demo $ sql-migrate up
Applied 2 migrations
osaka:demo $ sql-migrate status
+---------------+-----------------------------------------+
|   MIGRATION   |                 APPLIED                 |
+---------------+-----------------------------------------+
| 1_initial.sql | 2016-01-30 13:09:50.341158038 +0000 UTC |
| 2_record.sql  | 2016-01-30 13:09:50.342862386 +0000 UTC |
+---------------+-----------------------------------------+
osaka:demo $
```

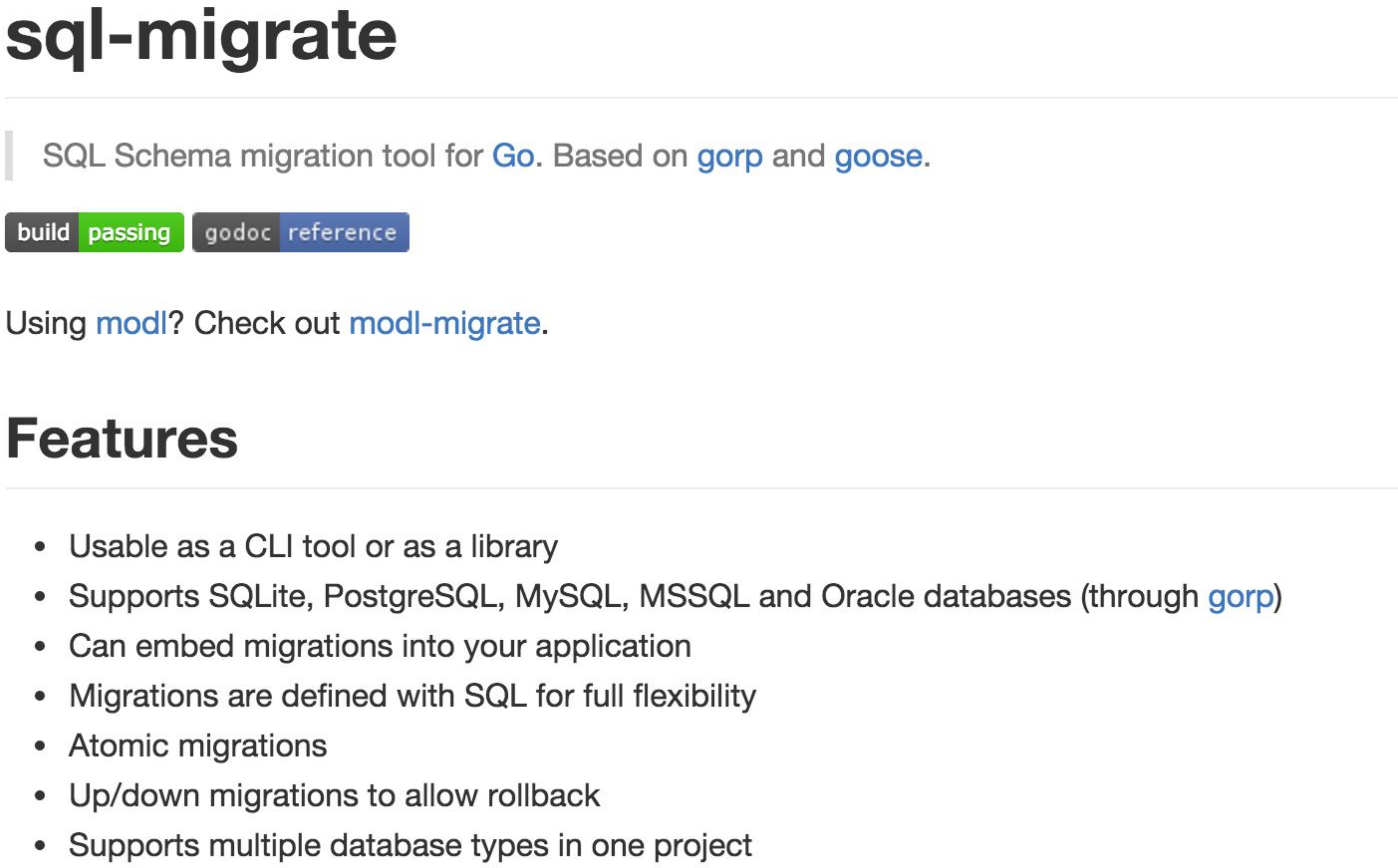


```
osaka:demo $ sql-migrate --help
usage: sql-migrate [--version] [--help] <command> [<args>]

Available commands are:
    down       Undo a database migration
    redo       Reapply the last migration
    status     Show migration status
    up         Migrates the database to the most recent versi
on available

osaka:demo $
```



```
func upgradeDb() error {
    migrations := &migrate.AssetMigrationSource{
        Asset:    Asset,
        AssetDir: AssetDir,
        Dir:      "migrations/postgres",
    }
    migrate.SetTable("migrations")

    n, err := migrate.Exec(Db.dbmap.Db, "postgres", migratio
ns, migrate.Up)
    if err != nil {
        return err
    }
    if n > 0 {
        Log("db", "Applied %d migrations\n", n)
    }
    return nil
}
```

https://github.com/rubenv/sql-migrate



sql-migrate

> SQL Schema migration tool for Go. Based on gorp and goose.

Using modl? Check out modl-migrate.

Features

- Usable as a CLI tool or as a library
- Supports SQLite, PostgreSQL, MySQL, MSSQL and Oracle databases (through gorp)
- Can embed migrations into your application
- Migrations are defined with SQL for full flexibility
- Atomic migrations
- Up/down migrations to allow rollback
- Supports multiple database types in one project

drone.io

CoreOS dex

CloudFoundry notifications

Thanks!

https://github.com/rubenv/sql-migrate

Ruben Vermeersch
https://rocketeer.be
@rubenv