ORMs and ERDs, OMG! Forward and Reverse Engineering Databases With Django's ORM and Migration Tools

Abe Kazemzadeh

SCALE23x, Pasadena, CA
abe.kazemzadeh AT stthomas DOT edu

2026-03-0x

Outline

Motivation

- I teach mainly proprietary databases (Oracle), but I like to use open source tools
- I saw parallels between forward and reverse engineering features in ER diagram tools and Django
  - E.g. Oracle Data Modeler, MySQL Workbench, DBeaver, Erwin
  - makemigrations/migrate and inspectdb in Django
- These features have different names and aren't usually presented together but I find it interesting to do so
- Forward and reverse engineering are features that are

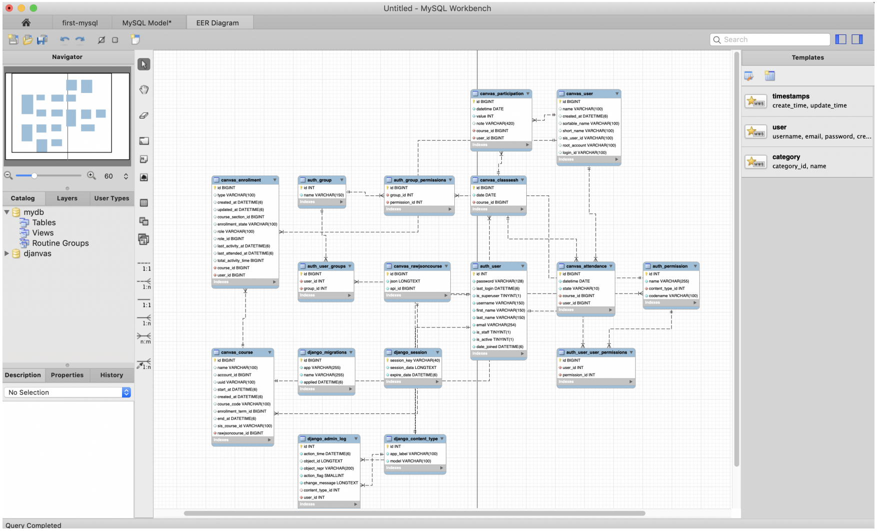
Motivation

- I teach mainly proprietary databases (Oracle), but I like to use open source tools
- I saw parallels between forward and reverse engineering features in ER diagram tools and Django
  - E.g. Oracle Data Modeler, MySQL Workbench, DBeaver, Erwin
  - makemigrations/migrate and inspectdb in Django
- These features have different names and aren't usually presented together but I find it interesting to do so
- Forward and reverse engineering are features that are

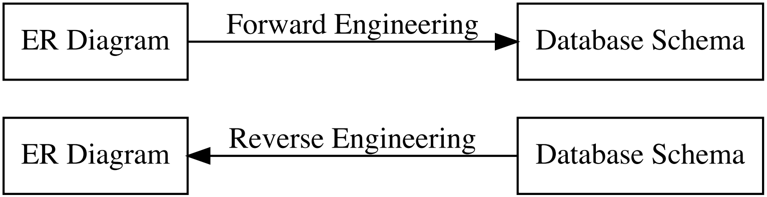
Motivation

- I teach mainly proprietary databases (Oracle), but I like to use open source tools
- I saw parallels between forward and reverse engineering features in ER diagram tools and Django
  - E.g. Oracle Data Modeler, MySQL Workbench, DBeaver, Erwin
  - makemigrations/migrate and inspectdb in Django
- These features have different names and aren't usually presented together but I find it interesting to do so
- Forward and reverse engineering are features that are

ER Diagram Example



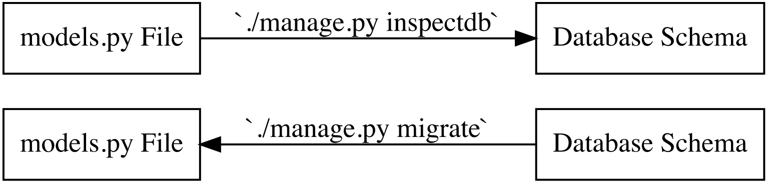
Forward and Reverse Engineering



Motivation

- I teach mainly proprietary databases (Oracle), but I like to use open source tools
- I saw parallels between forward and reverse engineering features in ER diagram tools and Django
    - E.g. Oracle Data Modeler, MySQL Workbench, DBeaver, Erwin
    - makemigrations/migrate and inspectdb in Django
- These features have different names and aren't usually presented together but I find it interesting to do so
- Forward and reverse engineering are features that are

Django InspectDB and Migration



Motivation

- I teach mainly proprietary databases (Oracle), but I like to use open source tools
- I saw parallels between forward and reverse engineering features in ER diagram tools and Django
  - E.g. Oracle Data Modeler, MySQL Workbench, DBeaver, Erwin
  - makemigrations/migrate and inspectdb in Django
- These features have different names and aren't usually presented together but I find it interesting to do so
- Forward and reverse engineering are features that are

Motivation

- I teach mainly proprietary databases (Oracle), but I like to use open source tools
- I saw parallels between forward and reverse engineering features in ER diagram tools and Django
    - E.g. Oracle Data Modeler, MySQL Workbench, DBeaver, Erwin
    - makemigrations/migrate and inspectdb in Django
- These features have different names and aren't usually presented together but I find it interesting to do so
- Forward and reverse engineering are features that are

Goals

- Describe and compare/contrast forward and reverse engineering to Django's migrations and inspectdb commands
- Give a tutorial example with MySQL workbench and Django
- Compare and contrast ER diagramming tools
- Show an alternative to ER diagramming using Django and Python's UML generation tool, `pyconvert` (from pylint package)
- Two-way communication is ideal: please ask questions and share your experiences

Goals

- Describe and compare/contrast forward and reverse engineering to Django's migrations and inspectdb commands
- Give a tutorial example with MySQL workbench and Django
- Compare and contrast ER diagramming tools
- Show an alternative to ER diagramming using Django and Python's UML generation tool, `pyconvert` (from pylint package)
- Two-way communication is ideal: please ask questions and share your experiences

Goals

- Describe and compare/contrast forward and reverse engineering to Django's migrations and inspectdb commands
- Give a tutorial example with MySQL workbench and Django
- Compare and contrast ER diagramming tools
- Show an alternative to ER diagramming using Django and Python's UML generation tool, `pyconvert` (from pylint package)
- Two-way communication is ideal: please ask questions and share your experiences

Goals

- Describe and compare/contrast forward and reverse engineering to Django's migrations and inspectdb commands
- Give a tutorial example with MySQL workbench and Django
- Compare and contrast ER diagramming tools
- Show an alternative to ER diagramming using Django and Python's UML generation tool, `pyconvert` (from pylint package)
- Two-way communication is ideal: please ask questions and share your experiences

Goals

- Describe and compare/contrast forward and reverse engineering to Django's migrations and inspectdb commands
- Give a tutorial example with MySQL workbench and Django
- Compare and contrast ER diagramming tools
- Show an alternative to ER diagramming using Django and Python's UML generation tool, `pyconvert` (from pylint package)
- Two-way communication is ideal: please ask questions and share your experiences

Overview

Outline

Terminology

- SQL: Structured Query Language, how to talk to databases
  - DDL: data definition (creating table structure)
  - DML: data manpulation (selecting/updating/deleting)
- ER diagram: visualize databases, node ≡ table, edge≡foreign key relationship between tables
- Logical design vs relational design: is a many-to-many relationship shown as an edge or as an intersection/association table
- Forward engineering: convert/export an ER diagram into a database schema (CREATE TABLE statements)
- Reverse engineering: convert/import a database schema (existing database) to an ER diagram

Terminology

- SQL: Structured Query Language, how to talk to databases
  - DDL: data definition (creating table structure)
  - DML: data manpulation (selecting/updating/deleting)
- ER diagram: visualize databases, node ≡ table, edge≡foreign key relationship between tables
- Logical design vs relational design: is a many-to-many relationship shown as an edge or as an intersection/association table
- Forward engineering: convert/export an ER diagram into a database schema (CREATE TABLE statements)
- Reverse engineering: convert/import a database schema (existing database) to an ER diagram

Terminology

- SQL: Structured Query Language, how to talk to databases
  - DDL: data definition (creating table structure)
  - DML: data manpulation (selecting/updating/deleting)
- ER diagram: visualize databases, node ≡ table, edge≡foreign key relationship between tables
- Logical design vs relational design: is a many-to-many relationship shown as an edge or as an intersection/association table
- Forward engineering: convert/export an ER diagram into a database schema (CREATE TABLE statements)
- Reverse engineering: convert/import a database schema (existing database) to an ER diagram

Terminology

- SQL: Structured Query Language, how to talk to databases
  - DDL: data definition (creating table structure)
  - DML: data manpulation (selecting/updating/deleting)
- ER diagram: visualize databases, node $\equiv$ table, edge$\equiv$foreign key relationship between tables
- Logical design vs relational design: is a many-to-many relationship shown as an edge or as an intersection/association table
- Forward engineering: convert/export an ER diagram into a database schema (CREATE TABLE statements)
- Reverse engineering: convert/import a database schema (existing database) to an ER diagram

Terminology

- SQL: Structured Query Language, how to talk to databases
  - DDL: data definition (creating table structure)
  - DML: data manpulation (selecting/updating/deleting)
- ER diagram: visualize databases, node $\equiv$ table, edge$\equiv$foreign key relationship between tables
- Logical design vs relational design: is a many-to-many relationship shown as an edge or as an intersection/association table
- Forward engineering: convert/export an ER diagram into a database schema (CREATE TABLE statements)
- Reverse engineering: convert/import a database schema (existing database) to an ER diagram

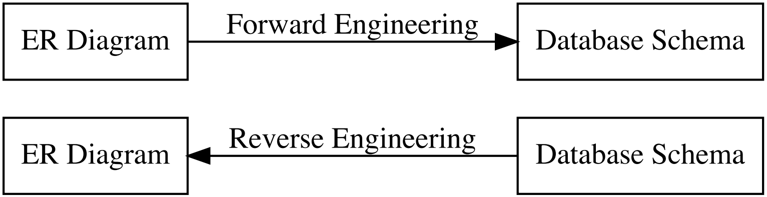
Terminology

- SQL: Structured Query Language, how to talk to databases
  - DDL: data definition (creating table structure)
  - DML: data manpulation (selecting/updating/deleting)
- ER diagram: visualize databases, node $\equiv$ table, edge$\equiv$foreign key relationship between tables
- Logical design vs relational design: is a many-to-many relationship shown as an edge or as an intersection/association table
- Forward engineering: convert/export an ER diagram into a database schema (CREATE TABLE statements)
- Reverse engineering: convert/import a database schema (existing database) to an ER diagram

Terminology

- SQL: Structured Query Language, how to talk to databases
  - DDL: data definition (creating table structure)
  - DML: data manpulation (selecting/updating/deleting)
- ER diagram: visualize databases, node ≡ table, edge≡foreign key relationship between tables
- Logical design vs relational design: is a many-to-many relationship shown as an edge or as an intersection/association table
- Forward engineering: convert/export an ER diagram into a database schema (CREATE TABLE statements)
- Reverse engineering: convert/import a database schema (existing database) to an ER diagram

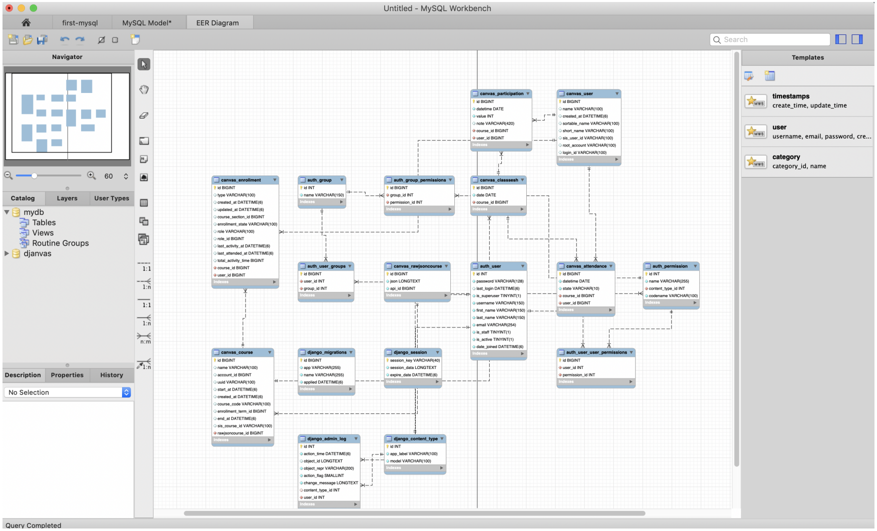
Forward and Reverse Engineering



Examples

- DBeaver
- MySQL Workbench
- Oracle Data Modeler (free but not open source)
- Erwin (not free or open source)

ER Diagram Example



Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
- Fwd/rev engineering features not always available
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
- Fwd/rev engineering features not always available
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
- Fwd/rev engineering features not always available
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
  - E.g. MySQL workbench
- Fwd/rev engineering features not always available
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
  - E.g. MySQL workbench
- Fwd/rev engineering features not always available
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
  - E.g. MySQL workbench
- Fwd/rev engineering features not always available
  - sometimes only available on "enterprise" editions
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
  - E.g. MySQL workbench
- Fwd/rev engineering features not always available
  - sometimes only available on "enterprise" editions
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
  - E.g. MySQL workbench
- Fwd/rev engineering features not always available
  - sometimes only available on "enterprise" editions
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
  - E.g. MySQL workbench
- Fwd/rev engineering features not always available
  - sometimes only available on “enterprise” editions
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
  - E.g. MySQL workbench
- Fwd/rev engineering features not always available
  - sometimes only available on "enterprise" editions
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
  - E.g. MySQL workbench
- Fwd/rev engineering features not always available
  - sometimes only available on "enterprise" editions
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Pros and Cons of ER Diagram Fwd/Rev Engineering

Pros

- Visual representation
- GUI
- Management of Detail
- Ease for beginners
- May or may not be integrated in a DB IDE

Cons

- Tools often specfic to DBs
  - E.g. MySQL workbench
- Fwd/rev engineering features not always available
  - sometimes only available on "enterprise" editions
- Not designed to be integrated into applications (tools, not libraries)
- Often linked to a specific datasource
- May or may not be integrated in a DB IDE

Outline

Django Overview

- Django is a Python server-side web framework
- We will be considering a part of Django, the ORM (object relational mapper)
- This component deals with mapping objects, defined in Python code, to a relational database

Django Overview

- Django is a Python server-side web framework
- We will be considering a part of Django, the ORM (object relational mapper)
- This component deals with mapping objects, defined in Python code, to a relational database

Django Overview

- Django is a Python server-side web framework
- We will be considering a part of Django, the ORM (object relational mapper)
- This component deals with mapping objects, defined in Python code, to a relational database

Django Migrations

- Roughly equivalent to forward engineering
  - But includes version info, data/DML in addition to schema/structure/DDL

Django Inspect DB

- Roughly equivalent to reverse engineering

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL
    - Tracks schema changes/versions
      Can migrate data, not only schema
      Stores migrations in the DB

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL
  - Tracks schema changes/versions
  - Can migrate data, not only schema
  - Stores migrations in the DB

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL
  - Tracks schema changes/versions
  - Can migrate data, not only schema
  - Stores migrations in the DB

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL
  - Tracks schema changes/versions
  - Can migrate data, not only schema
  - Stores migrations in the DB

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL
  - Tracks schema changes/versions
  - Can migrate data, not only schema
  - Stores migrations in the DB

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL
    - Tracks schema changes/versions
    - Can migrate data, not only schema
    - Stores migrations in the DB

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL
    - Tracks schema changes/versions
    - Can migrate data, not only schema
    - Stores migrations in the DB

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL
  - Tracks schema changes/versions
  - Can migrate data, not only schema
  - Stores migrations in the DB

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Pros and Cons of Django ORM Migrations/Inspectdb

Pros

- Cross-database support
- Django is designed to be used as a library
- Command-line scripting support
- Does more than just fwd/rev engineering schemas/DDL
  - Tracks schema changes/versions
  - Can migrate data, not only schema
  - Stores migrations in the DB

Cons

- Not as beginner friendly as ER diagrams
- No GUI or visualizations
- Doesn't have IDE support

Outline

Forward Engineering

- ER use-case: database architecture to SQL
- Django use-case: normal Django usage
    - Django manages SQL

Reverse Engineering

- ER use-case: visualize database structure
- Django use-case: convert existing DB structure to python code (model.py classes)
  - Django does not manage SQL (managed=False in the model)

Outline

Summary

- We discussed the similarities and differences between Django's migration features and ER diagrams forward and reverse engineering features
- We saw how we can use Django to convert between SQL and Python code
- We discussed different ERD software

Summary

- We discussed the similarities and differences between Django's migration features and ER diagrams forward and reverse engineering features
- We saw how we can use Django to convert between SQL and Python code
- We discussed different ERD software

Summary

- We discussed the similarities and differences between Django's migration features and ER diagrams forward and reverse engineering features
- We saw how we can use Django to convert between SQL and Python code
- We discussed different ERD software

Questions

- Any questions or comments?