Model

```
from django.db import models
from django.core.validators import (
    MaxValueValidator,
    MinValueValidator,
)

class Author(models.Model):
    # A very simple model example
    name = models.CharField(max_length=100)

class Book(models.Model):
    # Use a "ForeignKey" for a ManyToOne relationship
    author = models.ForeignKey(
        Author,
        on_delete=models.CASCADE,
    )

    title = models.CharField(max_length=100)
    release_date = models.DateField()

    # Very useful fields: "created" stores when it
    # originally was created, "last_updated" stores
    # whenever edited
    created = models.DateTimeField(
        auto_now_add=True)
    last_updated = models.DateTimeField(
        auto_now=True)

    # For "multiple-choice" fields, use this pattern
    CATEGORIES = {
        "fict": "Fiction",
        "nonfict": "Non-fiction",
    }
    category = models.CharField(
        max_length=10,
        default="fict",
        choices=CATEGORIES,
    )

    # More complicated example with custom validators
    num_stars = models.IntegerField(
        validators=[
            MaxValueValidator(5),
            MinValueValidator(1),
        ],
    )


# ManyToMany relationships
class ReadingList(models.Model):
    books = models.ManyToManyField(Book)
```

Migration work-flow

```
# Generate migrations for some recent DB changes
python manage.py makemigrations

# Double check where we're at
python manage.py showmigrations

# Apply the generated migrations
python manage.py migrate
```

DB Relationships

One-to-many An instance can be associated with an arbitrary number of other instances. *Example:* `Artist` can have released multiple `Albums`.

Many-to-many Freely associate any number to any number. *Example:* `Album` can have many `Tags`, and each Tag can have many albums. For Twitter, `User` can follow an arbitrary number of other `Users`.

One-to-one For every insteance of one model, there exists exactly one instance of another, effectively "splitting a model into two". *Example:* `Album` with `AlbumArtwork`.