```
    hunger = db.Column(db.Integer, nullable=False, default=20)
```

- Have to specify the type of column
- Columns can contain ***NULL*** unless `nullable=False`
- Can specify ***default***, ***unique***, ***primary_key***, ***autoincrement***

Creating the Database

```
$ ipython3
In [1] %run app.py

In [2] db.create_all()
```

- Create all the tables using this database connection
- Only have to do once
  - No effect if tables already exist
- If you change table schema
  - drop table & re-run

> **Note: Do I always have to drop the table?**
>
> Dropping all of your tables may seem like an extreme move every time you make a change to your schema. There are tools that can help you update your schema more smoothly. *Database migrations* are a common way to do this, but this topic is beyond our scope.

Using our Model

```
>>> fluffy = Pet(name='Fluffy', species="Pet")
>>> fluffy.hunger
20

>>> db.session.add(fluffy)     # required to add to database!
>>> db.session.commit()        # commit the transaction
```

You only have to use `db.session.add()` to add a new object once – you don’t need to keep adding it to the session each time you change it.

> **Note: Transactions**
>
> Database management systems (Postgres included) support the concept of **transactions**. The idea here is that you may want to update multiple parts of the database simultaneously, and if any piece of the update fails, the entire transaction fails.