**SQLAlchemy常用数据类型：**

**1.** Integer：整形，映射到数据库中是int类型。

**2.** Float：浮点类型，映射到数据库中是float类型。他占据的32位。

**3.** Double：双精度浮点类型，映射到数据库中是double类型，占据64位 (SQLALCHEMY中没有)。

**4.** String：可变字符类型，映射到数据库中是varchar类型.

**5.** Boolean：布尔类型，映射到数据库中的是tinyint类型。

**6.** DECIMAL：定点类型。是专门为了解决浮点类型精度丢失的问题的。在存储钱相关的字段的时候建议大家都使用这个数据类型。并且这个类型使用的时候需要传递两个参数，第一个参数是用来标记这个字段总能能存储多少个数字，第二个参数表示小数点后有多少位。

**7.** Enum：枚举类型。指定某个字段只能是枚举中指定的几个值，不能为其他值。在ORM模型中，使用Enum来作为枚举，示例代码如下：

class News(Base):

\_\_tablename\_\_ = 'news'

tag = Column(Enum("python",'flask','django'))

在Python3中，已经内置了enum这个枚举的模块，我们也可以使用这个模块去定义相关的字段。示例代码如下：

class TagEnum(enum.Enum):

python = "python"

flask = "flask"

django = "django"

class News(Base):

\_\_tablename\_\_ = 'news'

id = Column(Integer,primary\_key=True,autoincrement=True)

tag = Column(Enum(TagEnum))

news = News(tag=TagEnum.flask)

**8.** Date：存储时间，只能存储年月日。映射到数据库中是date类型。在Python代码中，可以使用`datetime.date`来指定。

**9.** DateTime：存储时间，可以存储年月日时分秒毫秒等。映射到数据库中也是datetime类型。在Python代码中，可以使用`datetime.datetime`来指定。

**10.** Time：存储时间，可以存储时分秒。映射到数据库中也是time类型。在Python代码中，可以使用`datetime.time`来至此那个。示例代码如下：

class News(Base):

\_\_tablename\_\_ = 'news'

create\_time = Column(Time)

news = News(create\_time=time(hour=11,minute=11,second=11))

**11.** Text：存储长字符串。一般可以存储6W多个字符。如果超出了这个范围，可以使用LONGTEXT类型。映射到数据库中就是text类型。

**12.** LONGTEXT：长文本类型，映射到数据库中是longtext类型。

**演示整体代码如下：**

|  |
| --- |
| **from** sqlalchemy **import** create\_engine,Column,Integer,String,Float,Enum,Boolean,DECIMAL,Text,Date,DateTime,Time  **from** sqlalchemy.ext.declarative **import** declarative\_base  **from** sqlalchemy.dialects.mysql **import** LONGTEXT  **from** sqlalchemy.orm **import** sessionmaker  **import** enum  **from** datetime **import** date  **from** datetime **import** datetime  **from** datetime **import** time  #准备数据库的一堆信息 ip port user pwd 数据库的名称 按要求组织格式  HOSTNAME = **'127.0.0.1'**  PORT = **'3306'**  DATABASE = **'first\_sqlalchemy'**  USERNAME = **'root'**  PASSWORD = **'root'**  #dialect+driver://username:password@host:port/database?charset=utf8  #按照上述的格式来 组织数据库信息  DB\_URI =**"mysql+pymysql://{username}:{password}@{host}:{port}/{db}?charset=utf8"**.\  format(username=USERNAME,password=PASSWORD,host=HOSTNAME,port=PORT,db=DATABASE)  #创建数据库引擎  engine = create\_engine(DB\_URI)  #创建会话对象  session = sessionmaker(engine)()  #定义一个枚举类  **class** TagEnum(enum.Enum):  python=**"PYHTON2"**  flask=**"FLASK2"**  django =**"DJANGO"**  #创建一个ORM模型 说明基于sqlalchemy 映射到mysql数据库的常用字段类型有哪些？  Base = declarative\_base(engine)  **class** News(Base):  \_\_tablename\_\_=**'news'**  id = Column(Integer,primary\_key=**True**,autoincrement=**True**)  price1 = Column(Float) #存储数据时存在精度丢失问题  price2 = Column(DECIMAL(10,4))  title = Column(String(50))  is\_delete =Column(Boolean)  tag1 =Column(Enum(**'PYTHON'**,**'FLASK'**,**'DJANGO'**)) #枚举常规写法  tag2 =Column(Enum(TagEnum)) #枚举另一种写法  create\_time1=Column(Date)  create\_time2=Column(DateTime)  create\_time3=Column(Time)  content1 =Column(Text)  content2 =Column(LONGTEXT)  # Base.metadata.drop\_all()  # Base.metadata.create\_all()  #新增数据到表news中  # a1 = News(price1=1000.0078,price2=1000.0078,title='测试数据',is\_delete=True,tag1="PYTHON",tag2=TagEnum.flask,  # create\_time1=date(2018,12,12),create\_time2=datetime(2019,2,20,12,12,30),create\_time3=time(hour=11,minute=12,second=13),  # content1="hello",content2 ="hello hi nihao")  a1 = News(price1=1000.0078,price2=1000.0078,title=**'测试数据'**,is\_delete=**False**,tag1=**"PYTHON"**,tag2=TagEnum.python,  create\_time1=date(2018,12,12),create\_time2=datetime(2019,2,20,12,12,30),create\_time3=time(hour=11,minute=12,second=13),  content1=**"hello"**,content2 =**"hello hi nihao"**)  session.add(a1)  session.commit() |