UNIVERSIDAD CATÓLICA DE TEMUCO

Programación III

# TDA con ORM (SQLAlchemy) TAREA 01

MARIO LEANDRO CASTILLO SANHUEZA

UNIVERSIDAD CATÓLICA DE TEMUCO

# TDA Cola en Juegos RPGs

UNIVERSIDAD CATÓLICA DE TEMUCO

- Contexto
  - En videojuegos RPGs , las misiones secundarias se gestionan como colas (FIFO).
- Ejemplo típico:
  - "Derrota a 5 slimes" (primera misión aceptada).
  - "Recoge 3 hierbas medicinales"
- Por qué usar Cola:
  - Justicia en el orden de completado.
  - Evita que el jugador postergue misiones difíciles

# SQLAlchemy ORM en 1 minuto

UNIVERSIDAD CATÓLICA DE TEMUCO

¿Qué hace?

- Mapea clases de Python ↔ Tablas de BD.

Ventajas para TDA:

- Persistencia automática de datos.
- Operaciones abstractas (ej: `add(), query()`)

Ejemplo de Clase:

test.py > ...

```python
from sqlalchemy import Column, Integer, String
from sqlalchemy.ext.declarative import declarative_base

Base = declarative_base()

class Cola(Base):
    __tablename__ = 'colas'
    id = Column(Integer, primary_key=True)
    elementos = Column(String)  # Almacenamiento como JSON
```

# Uso de la librería

UNIVERSIDAD CATÓLICA DE TEMUCO

1. Tipos de Columnas:

```
from sqlalchemy import Column, Integer, String, Text, Enum, DateTime, ForeignKey
```

- Column: Define columnas en tablas de BD.
- Integer, String, Text: Tipos de datos para atributos.
- Enum: Para campos con valores predefinidos (ej: estados).
- DateTime: Manejo de fechas automático…
- ForeignKey: Claves foráneas para relaciones entre tablas.

2. Configuración Base:

```
from sqlalchemy.ext.declarative import declarative_base
```

- declarative_base(): Crea la clase base para modelos SQLAlchemy.

3. Relaciones entre Tablas

```
from sqlalchemy.orm import relationship
```

- relationship: Establece conexiones entre modelos (ej: 1-N, N-M)

# Uso SQLAlchemy

UNIVERSIDAD CATÓLICA DE TEMUCO

```python
# Se define la base del modelo ORM
Base = declarative_base()

class Mision(Base):
    """
    Representa una misión dentro del juego.
    """
    __tablename__ = 'misiones'

    id = Column(Integer, primary_key=True)  # Identificador único de la misión
    nombre = Column(String(50), nullable=False)  # Nombre de la misión (obligatorio)
    descripcion = Column(Text, nullable=True)  # Descripción opcional de la misión
    experiencia = Column(Integer, default=0)  # XP de recompensa, por defecto 0
    estado = Column(Enum('pendiente', 'completada', name='estados'), nullable=False)  # Estado de la misión
    fecha_creacion = Column(DateTime, default=datetime)  # Fecha de creación con valor por defecto
```

# TDA Cola en Juegos RPGs

UNIVERSIDAD CATÓLICA DE TEMUCO

```python
    # Relación con MisionPersonaje
    personajes = relationship("MisionPersonaje", back_populates="mision")

class Personaje(Base):
    """
    Representa un personaje dentro del juego.
    """
    __tablename__ = 'personajes'

    id = Column(Integer, primary_key=True)  # Identificador único del personaje
    nombre = Column(String(30), nullable=False)  # Nombre del personaje (obligatorio)
```

```python
class MisionPersonaje(Base):
    """
    Tabla intermedia para la relación muchos a muchos entre Personaje y Mision.
    También permite manejar el orden FIFO de las misiones.
    """
    __tablename__ = 'misiones_personaje'

    personaje_id = Column(Integer, ForeignKey('personajes.id'), primary_key=True)  # FK a personaje
    mision_id = Column(Integer, ForeignKey('misiones.id'), primary_key=True)  # FK a misión
    orden = Column(Integer)  # Para mantener el orden FIFO de las misiones

    # Relaciones inversas
    personaje = relationship("Personaje", back_populates="misiones")
    mision = relationship("Mision", back_populates="personajes")
```

# Métodos más usados en SQLAlchemy

UNIVERSIDAD CATÓLICA DE TEMUCO

```python
# 1. Consultas (Queries)
session.query(Mision).all()           # Obtener todos los registros
session.query(Mision).first()         # Primer registro

# 2. Creación y Adición de Registros
nuevo_obj = Mision(campo1=valor1, campo2=valor2)
session.add(nuevo_obj)                # Añadir a la sesión
session.add_all([obj1, obj2])         # Añadir múltiples objetos
session.commit()                      # Confirmar cambios en BD

# 3. Actualización de Registros
obj = session.get(Mision, id)         # Obtener objeto por ID
obj.campo = nuevo_valor               # Modificar atributos
session.commit()                      # Guardar cambios

# 4. Eliminación de Registros
obj = session.get(Mision, id)
session.delete(obj)                   # Eliminar objeto
session.commit()                      # Confirmar eliminación

# 5. Relaciones
# Relación 1-N
parent.hijos.append(child)            # Añadir hijo a padre

# Relación N-M
obj1.relacion.append(obj2)            # Añadir a tabla intermedia
session.commit()
```

# ACTIVIDAD

UNIVERSIDAD CATÓLICA DE TEMUCO

Tarea: Sistema de Misiones RPG con Colas (Estructuras de Datos)

Objetivos de Aprendizaje:

Implementar y aplicar el TDA Cola (FIFO) en un contexto real

Integrar estructuras de datos con persistencia (base de datos)

Manejar relaciones muchos-a-muchos con priorización

Desarrollar una API REST con operaciones básicas CRUD

Requisitos Técnicos

1. Estructura de Datos Obligatoria (TDA Cola)

Implementar una cola para gestionar el orden FIFO de misiones por personaje

Debe incluir como mínimo:

enqueue(mission): Añadir misión al final

dequeue(): Eliminar/retornar la primera misión

first(): Ver la primera misión sin remover

is_empty(): Verificar si está vacía

size(): Obtener cantidad de misiones

2. Endpoints Requeridos (FastAPI)

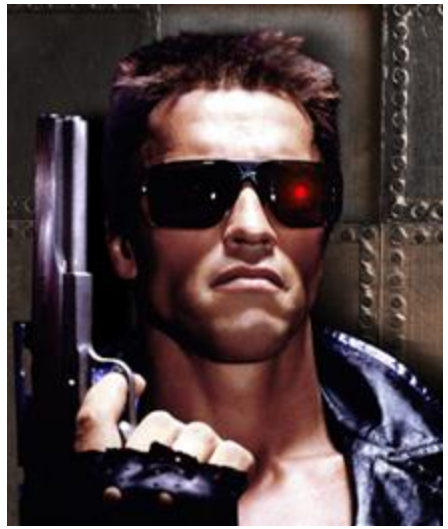
| Método | Ruta | Descripción |
|---|---|---|
| POST | /personajes | Crear nuevo personaje |
| POST | /misiones | Crear nueva misión |
| POST | /personajes/{id}/misiones/{id} | Aceptar misión (encolar) |
| POST | /personajes/{id}/completar | Completar misión (desencolar + sumar XP) |
| GET | /personajes/{id}/misiones | Listar misiones en orden FIFO |

# ACTIVIDAD

Documentacion: https://docs.sqlalchemy.org/en/20/orm/quickstart.html

Que debe entregar:

- Video explicativo del trabajo realizado, compartido mediante un enlace de Youtube. **(Duración máximo 10 min).**
  - **Explicar donde se utiliza la TDA-Cola**
  - **Explicar donde se utiliza el ORM.**
  - **Explicar donde se utiliza la API.**
- Código subido a un repositorio en Github.
- Trabajo individual .
- Plazo de entrega: Miercoles 09/04



UNIVERSIDAD CATÓLICA DE TEMUCO

# ¿CONSULTAS?

MARIO LEANDRO CASTILLO SANHUEZA

UNIVERSIDAD CATÓLICA DE TEMUCO