Ejercicio 1:

**Descripción**: Esta ontología modela operadores de un videojuego táctico de disparos en equipo llamado Rainbow Six Siege, representando sus atributos, roles y equipamiento. Se estructura en torno a la clasificación de operadores en función de su género, país de origen, equipo especial al que pertenecen, rol en el juego, nivel de armadura y velocidad. Además, detalla el equipamiento disponible para cada operador, incluyendo armas principales, secundarias y gadgets especializados.

**Tipos de conceptos y entidades modelados**

* **Operadores** (ex:Operator): Representan a los personajes jugables en el videojuego.
* **Género** (ex:Male, ex:Female): Clasifica a los operadores según su identidad de género.
* **Roles** (ex:Attacker, ex:Defender): Diferencia entre operadores ofensivos y defensivos.
* **Equipos especiales** (ex:SAS, ex:GIGN, ex:FBI\_SWAT, etc.): Indican la unidad táctica a la que pertenece cada operador.
* **Atributos físicos y de jugabilidad**:
  + ex:Armor: Nivel de armadura.
  + ex:Speed: Velocidad de movimiento.
* **Equipamiento**:
  + ex:PrimaryWeapon: Armas principales que pueden utilizar los operadores.
  + ex:SecondaryWeapon: Armas secundarias disponibles.
  + ex:Gadget: Habilidad o herramienta especial de cada operador.

**Elementos de otras ontologías o grafos de conocimiento utilizados**

* **Wikidata** (wd:, wdt:): Se utiliza para vincular entidades con conceptos existentes en la base de conocimiento de Wikidata, como los géneros (wd:Q6581097, wd:Q6581072) y la colección de operadores (wd:Q17183996).
* **OWL** (owl:sameAs): Permite establecer equivalencias entre las entidades definidas en la ontología y sus correspondientes en Wikidata.
* **RDFS** (rdfs:label): Se emplea para asignar etiquetas en inglés a los operadores y otros elementos del modelo.

Ejercicio 2:

URL Jena: <http://156.35.95.58:3030/#/dataset/DatasetR6/query>

Fotos de apoyo para ver la subida y existencia de datos en documento: Imágenes Apoyo

URL Virtuoso: <http://156.35.95.58:8890/conductor>

Fotos de apoyo para ver la subida y existencia de datos en documento: Imágenes Apoyo

Consulta 1:

Descripción: Te devuelve todos los agentes

Consulta:

**PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>**

**PREFIX ex: <http://example.org/characters/>**

**SELECT ?character ?role WHERE {**

**?character ex:Role ?role .**

**}**

Salida:

**Resultados de la consulta SPARQL:**

**http://example.org/characters/Castle**

**http://example.org/characters/Pulse**

**http://example.org/characters/Doc**

**http://example.org/characters/Rook**

**http://example.org/characters/Jager**

**http://example.org/characters/Frost**

**http://example.org/characters/Valkyrie**

**http://example.org/characters/Echo**

**http://example.org/characters/Mira**

**http://example.org/characters/Sledge**

**http://example.org/characters/Thatcher**

**http://example.org/characters/Ash**

**http://example.org/characters/Montagne**

**http://example.org/characters/Twitch**

**http://example.org/characters/Blitz**

**http://example.org/characters/IQ**

**http://example.org/characters/Fuze**

**http://example.org/characters/Glaz**

**http://example.org/characters/Buck**

**http://example.org/characters/Blackbeard**

**http://example.org/characters/Capitao**

**http://example.org/characters/Jackal**

Consulta 2:

Descripción: Te devuelve todos los agentes cuyo país es Francia

Consulta:

**PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>**

**PREFIX ex: <http://example.org/characters/>**

**SELECT ?character WHERE {**

**?character ex:Country ex:France .**

**}**

Salida:

**Resultados de la consulta SPARQL:**

**http://example.org/characters/Montagne**

**http://example.org/characters/Twitch**

**http://example.org/characters/Doc**

**http://example.org/characters/Rook**

Consulta 3:

Descripción: Te devuelve todos los agentes cuya armadura es igual a 3

Consulta:

**PREFIX ex: <http://example.org/characters/>**

**PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>**

**SELECT ?character WHERE {**

**?character ex:Armor "3"^^xsd:integer .**

**}**

Salida:

**Resultados de la consulta SPARQL:**

**http://example.org/characters/Montagne**

**http://example.org/characters/Blitz**

**http://example.org/characters/Fuze**

**http://example.org/characters/Doc**

**http://example.org/characters/Rook**

**http://example.org/characters/Echo**

**http://example.org/characters/Mira**

Consulta 4:

Descripción: Te devuelve todos los agentes cuya velocidad es igual a 3

Consulta:

**PREFIX ex: <http://example.org/characters/>**

**PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>**

**SELECT ?character WHERE {**

**?character ex:Speed "3"^^xsd:integer .**

**}**

Salida:

PS C:\Users\ivans\Desktop\EntregaWebSemantica> python ./Python.py

**Resultados de la consulta SPARQL:**

**http://example.org/characters/Ash**

**http://example.org/characters/IQ**

**http://example.org/characters/Pulse**

Consulta 5:

Descripción: Te devuelve el gadget de el operador que le pasas: Twitch

Consulta:

**PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>**

**PREFIX ex: <http://example.org/characters/>**

**SELECT ?gadget WHERE {**

**ex:Twitch ex:Gadget ?gadget .**

**}**

Salida:

**Resultados de la consulta SPARQL:**

**http://example.org/characters/ShockDrone**

Consulta 6:

Descripción: Te devuelve todos los integrantes del grupo swat: GIGN

Consulta:

**PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>**

**PREFIX ex: <http://example.org/characters/>**

**SELECT ?character WHERE {**

**?character ex:Team ex:GIGN .**

**}**

Salida:

**Resultados de la consulta SPARQL:**

**http://example.org/characters/Montagne**

**http://example.org/characters/Twitch**

**http://example.org/characters/Doc**

**http://example.org/characters/Rook**

Consulta 7 FEDERADA:

Descripción: Te devuelve todos los agentes junto a la fecha de publicación del videojuego en wikidata

Consulta:

**PREFIX wd: <http://www.wikidata.org/entity/>**

**PREFIX wdt: <http://www.wikidata.org/prop/direct/>**

**PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>**

**PREFIX ex: <http://example.org/characters/>**

**SELECT ?operatorName ?releaseDate WHERE {**

**SERVICE <https://query.wikidata.org/sparql> {**

**wd:Q17183996 wdt:P577 ?releaseDate.**

**}**

**?operator rdfs:label ?operatorName .**

**FILTER(LANG(?operatorName) = "en")**

**FILTER(?operatorName != "Male"@en && ?operatorName != "Female"@en) # Excluir géneros**

**}**

Salida:

**Resultados de la consulta SPARQL:**

**Operator: Sledge, Release Date: 2015-12-01T00:00:00Z**

**Operator: Thatcher, Release Date: 2015-12-01T00:00:00Z**

**Operator: Ash, Release Date: 2015-12-01T00:00:00Z**

**Operator: Montagne, Release Date: 2015-12-01T00:00:00Z**

**Operator: Twitch, Release Date: 2015-12-01T00:00:00Z**

**Operator: Blitz, Release Date: 2015-12-01T00:00:00Z**

**Operator: IQ, Release Date: 2015-12-01T00:00:00Z**

**Operator: Fuze, Release Date: 2015-12-01T00:00:00Z**

**Operator: Glaz, Release Date: 2015-12-01T00:00:00Z**

**Operator: Buck, Release Date: 2015-12-01T00:00:00Z**

**Operator: Blackbeard, Release Date: 2015-12-01T00:00:00Z**

**Operator: Capitão, Release Date: 2015-12-01T00:00:00Z**

**Operator: Jackal, Release Date: 2015-12-01T00:00:00Z**

**Operator: Castle, Release Date: 2015-12-01T00:00:00Z**

**Operator: Pulse, Release Date: 2015-12-01T00:00:00Z**

**Operator: Doc, Release Date: 2015-12-01T00:00:00Z**

**Operator: Rook, Release Date: 2015-12-01T00:00:00Z**

**Operator: Jäger, Release Date: 2015-12-01T00:00:00Z**

**Operator: Frost, Release Date: 2015-12-01T00:00:00Z**

**Operator: Valkyrie, Release Date: 2015-12-01T00:00:00Z**

**Operator: Echo, Release Date: 2015-12-01T00:00:00Z**

**Operator: Mira, Release Date: 2015-12-01T00:00:00Z**

Consulta 8 FEDERADA:

Descripción: Te devuelve todos los agentes con su país y asociado el país en Wikidata

Consulta:

**PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>**

**PREFIX ex: <http://example.org/characters/>**

**PREFIX wd: <http://www.wikidata.org/entity/>**

**PREFIX wdt: <http://www.wikidata.org/prop/direct/>**

**SELECT ?operator ?name ?countryLabel WHERE {**

**?operator rdfs:label ?name ;**

**ex:Country ?country .**

**# Mapeo manual de los identificadores locales a Wikidata**

**VALUES (?country ?wikidataCountry) {**

**(ex:UnitedKingdom wd:Q145)**

**(ex:UnitedStates wd:Q30)**

**(ex:France wd:Q142)**

**(ex:Germany wd:Q183)**

**(ex:Russia wd:Q159)**

**(ex:Canada wd:Q16)**

**(ex:Brazil wd:Q155)**

**(ex:Spain wd:Q29)**

**}**

**# Consulta federada a Wikidata**

**SERVICE <https://query.wikidata.org/sparql> {**

**?wikidataCountry rdfs:label ?countryLabel .**

**FILTER (LANG(?countryLabel) = "en")**

**}**

**}**

Salida:

**Resultados de la consulta SPARQL:**

**Operator: Sledge (http://example.org/characters/Sledge), Country: United Kingdom**

**Operator: Thatcher (http://example.org/characters/Thatcher), Country: United Kingdom**

**Operator: Ash (http://example.org/characters/Ash), Country: United States**

**Operator: Blackbeard (http://example.org/characters/Blackbeard), Country: United States**

**Operator: Castle (http://example.org/characters/Castle), Country: United States**

**Operator: Pulse (http://example.org/characters/Pulse), Country: United States**

**Operator: Valkyrie (http://example.org/characters/Valkyrie), Country: United States**

**Operator: Montagne (http://example.org/characters/Montagne), Country: France**

**Operator: Twitch (http://example.org/characters/Twitch), Country: France**

**Operator: Doc (http://example.org/characters/Doc), Country: France**

**Operator: Rook (http://example.org/characters/Rook), Country: France**

**Operator: Blitz (http://example.org/characters/Blitz), Country: Germany**

**Operator: IQ (http://example.org/characters/IQ), Country: Germany**

**Operator: Jäger (http://example.org/characters/Jager), Country: Germany**

**Operator: Fuze (http://example.org/characters/Fuze), Country: Russia**

**Operator: Glaz (http://example.org/characters/Glaz), Country: Russia**

**Operator: Buck (http://example.org/characters/Buck), Country: Canada**

**Operator: Frost (http://example.org/characters/Frost), Country: Canada**

**Operator: Capitão (http://example.org/characters/Capitao), Country: Brazil**

**Operator: Jackal (http://example.org/characters/Jackal), Country: Spain**

**Operator: Mira (http://example.org/characters/Mira), Country: Spain**

Ejercicio 4:

Archivo shex: schema.shexe

Archivo shape\_map: shape\_map.sm

Explicacion archivos:

El primer archivo define las **shapes en ShEx** para validar operadores de un juego, estableciendo sus propiedades como Gender, Role, Team, Weapons, y Gadgets. También incluye shapes para armas, gadgets, género, equipos y roles.

El segundo archivo (shape-map.sm) **mapea operadores específicos** (Sledge, Thatcher, Ash, etc.) a la shape <OperatorShape>, indicando que deben cumplir con sus restricciones.

Juntos, estos archivos sirven para validar que los datos RDF de los operadores sigan una estructura coherente y completa.

Ejercicio 5:

Código Python de extracción de Shapes:

**from shexer.shaper import Shaper  
from shexer.consts import TURTLE, SHEXC  
  
# Archivo de entrada con el grafo RDF  
archivo\_rdf = "datos.ttl" # Asegúrate de que este archivo existe en el mismo directorio  
  
# Configurar el extractor  
shaper = Shaper(  
 graph\_file\_input=archivo\_rdf, # Se usa 'graph\_file\_input' en vez de 'target\_graph'  
 input\_format=TURTLE, # Formato del archivo (TURTLE, NT, JSON\_LD, etc.)  
 namespaces\_dict={"ex": "http://example.org/"}, # Agregar prefijos si es necesario  
 all\_classes\_mode=True # Extraer shapes para todas las clases encontradas  
)  
  
# Extraer en formato ShEx  
shex\_shapes = shaper.shex\_graph(string\_output=True)  
with open("shapes.shex", "w", encoding="utf-8") as f:  
 f.write(shex\_shapes)  
  
print("Shapes generadas y guardadas en 'shapes.shex'")**

Archivo .shex en la carpeta Ejercicio 5

Reflexión personal de la comparación:

**Comparación entre las Shapes creadas manualmente y las generadas por la herramienta**

1. **Estructura y modularidad**
   * La shape creada manualmente es más organizada y modular, con cada concepto clave (Operador, Arma, Gadget, etc.) definido como una shape separada.
   * La shape generada por la herramienta es más compacta y menos modular, ya que agrupa las relaciones dentro de una única definición sin desglosarlas en entidades individuales.
2. **Uso de IRIs y Literales**
   * En la shape manual, algunos atributos como ex:Country están definidos como literales (xsd:string).
   * En la shape extraída, ex:Country y otros atributos son tratados como IRIs, lo que indica que en los datos originales estos valores referencian entidades en lugar de ser simples cadenas de texto.
3. **Restricciones de cardinalidad**
   * La shape manual usa restricciones claras como + (al menos uno) y \* (cero o más).
   * La shape generada incluye estadísticas sobre la distribución real de los datos (por ejemplo, 50.0 % obj: IRI. Cardinality: {1}), proporcionando información sobre la frecuencia de aparición de cada propiedad.
4. **Elementos adicionales**
   * La shape extraída incluye rdfs:label xsd:string, algo que no está presente en la manual. Esto sugiere que en los datos originales cada operador tenía un nombre o etiqueta.

**Reflexión sobre el uso de Shapes creadas manualmente vs. extraídas**

* **Shapes creadas desde cero**  
  Son más útiles cuando se diseñan modelos de datos desde el principio, ya que permiten definir estructuras semánticas claras, bien organizadas y modulares. También son preferibles en contextos donde se necesita control total sobre la ontología.
* **Shapes extraídas**  
  Son ideales para analizar datos existentes y extraer patrones de uso. Suelen reflejar la realidad de los datos en lugar de una estructura idealizada. Son útiles en proyectos de integración de datos o validación de conjuntos de datos heterogéneos.

Ejercicio 7:

Enlace a la iteración con un LLM: <https://chatgpt.com/share/67af47a0-9f5c-800f-a654-fa076f9bb158>

Reflexión:

La comparación entre grafos de conocimiento (RDF/SPARQL) y modelos de lenguaje (LLMs) resalta diferencias clave en precisión, interpretabilidad y aplicabilidad. En este ejercicio, observamos que mientras SPARQL proporciona datos estructurados y verificables, el LLM cometió errores en la recuperación de información, como la omisión de un agente en la pregunta 4.

**Ventajas de los Grafos de Conocimiento (RDF/SPARQL)**

1. **Alta precisión y confiabilidad**: La información proviene de fuentes verificadas, reduciendo la posibilidad de errores y malinterpretaciones.
2. **Estructura clara y organizada**: La recuperación de datos sigue un formato bien definido, lo que evita inconsistencias.
3. **Trazabilidad de la información**: Se puede verificar el origen de cada dato, lo que facilita la auditoría y validación.
4. **Actualización colaborativa**: Plataformas como Wikidata permiten la mejora y actualización constante de los datos.

**Ventajas de los Modelos de Lenguaje (LLMs)**

1. **Interacción en lenguaje natural**: No requiere conocimientos técnicos en consultas estructuradas, lo que facilita su uso.
2. **Capacidad de inferencia y contextualización**: Puede generar explicaciones detalladas y adaptar respuestas a diferentes contextos.
3. **Integración flexible de datos**: Puede combinar información estructurada con conocimiento implícito adquirido en su entrenamiento.
4. **Síntesis y generación de contenido**: Facilita la comprensión de datos complejos al traducirlos en explicaciones claras y accesibles.

**Limitaciones de los LLMs**

1. **Errores en la recuperación de datos**: En la pregunta 4, el modelo omitió un agente clave, evidenciando fallas en la precisión de la información.
2. **Falta de trazabilidad**: No siempre es posible verificar la fuente de los datos generados por el modelo.
3. **Inconsistencia en las respuestas**: La formulación de la pregunta puede influir en la variabilidad de los resultados.
4. **Riesgo de "alucinaciones"**: Los LLMs pueden generar información incorrecta o inexistente, especialmente si carecen de fuentes confiables.

Ejercicio 8:

Mi presentación fue sobre: SPARQL vs SQL