Lab 11: Name Entity Linking

Named Entity Linking (NEL) is a natural language processing (NLP) task that involves linking named entities mentioned in text to corresponding entries or entities in a knowledge base or database. The goal of NEL is to disambiguate named entities by determining their identity or meaning based on context and linking them to unique identifiers in a knowledge base.

Question: Extend the NER Experimet to include Named Entity Linking associated recognised entities with their corresponding database.

In [2]:
```
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.tag import pos_tag
from nltk.chunk import conlltags2tree, tree2conlltags

nltk.download('punkt')
nltk.download('maxent_ne_chunker')
nltk.download('words')
nltk.download('averaged_perceptron_tagger')

# A dictionary to store entity information
entity_database = {
    "Geeta": {
        "name": "Geeta",
        "gender": "Female",
        "nationality": "Indian"
    },
    "India": {
        "name": "India",
        "type": "Country"
    },
    "Sitar": {
        "name": "Sitar",
        "type": "Instrument"
    }
}

def preprocess(text):
    tokens = word_tokenize(text)
    stop_words = set(stopwords.words('english'))
    filtered_tokens = [token for token in tokens if token.lower() not in stop_words and token.isalpha()]
    return filtered_tokens

def extract_entities(text):
    tokens = preprocess(text)
    tagged_tokens = pos_tag(tokens)
    ne_tree = nltk.ne_chunk(tagged_tokens)
    iob_tags = tree2conlltags(ne_tree)
    return iob_tags

def link_entities(entities):
    linked_entities = []
    for word, pos_tag, entity_tag in entities:
        if entity_tag != 'O':
            entity_info = entity_database.get(word, {})
            linked_entities.append((word, pos_tag, entity_tag, entity_info))
        else:
            linked_entities.append((word, pos_tag, entity_tag, {}))
    return linked_entities

text = 'Geeta is a girl from India. She likes playing Sitar.'
entities = extract_entities(text)
linked_entities = link_entities(entities)

for entity in linked_entities:
    word, pos_tag, entity_tag, entity_info = entity
    if entity_info:
        print(f"Word: {word}, POS Tag: {pos_tag}, Entity Tag: {entity_tag}")
        print(f"Entity Information: {entity_info}")
    else:
        print(f"Word: {word}, POS Tag: {pos_tag}, Entity Tag: {entity_tag}")
```

```
Word: Geeta, POS Tag: NNP, Entity Tag: B-GPE
Entity Information: {'name': 'Geeta', 'gender': 'Female', 'nationality': 'Indian'}
Word: girl, POS Tag: NN, Entity Tag: O
Word: India, POS Tag: NNP, Entity Tag: B-GPE
Entity Information: {'name': 'India', 'type': 'Country'}
Word: likes, POS Tag: VBZ, Entity Tag: O
Word: playing, POS Tag: VBG, Entity Tag: O
Word: Sitar, POS Tag: NNP, Entity Tag: B-PERSON
Entity Information: {'name': 'Sitar', 'type': 'Instrument'}

[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\ahana\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package maxent_ne_chunker to
[nltk_data]     C:\Users\ahana\AppData\Roaming\nltk_data...
[nltk_data]   Package maxent_ne_chunker is already up-to-date!
[nltk_data] Downloading package words to
[nltk_data]     C:\Users\ahana\AppData\Roaming\nltk_data...
[nltk_data]   Package words is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\ahana\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```