# Task 2

## A. Deriving 4 new triples

Below are 4 nontrivial RDFS entailments—i.e., new triples that weren't explicitly in the graph but can be inferred by the RDFS rules.

---

1. **ex:MilesDavis rdf:type ex:Person**

- **Proof**
  - The graph states: ex:MilesDavis rdf:type ex:Artist.
  - Also: ex:Artist rdfs:subClassOf ex:Person.
  - By **RDFS Subclass Rule** (if C1 rdfs:subClassOf C2 and x rdf:type C1, then x rdf:type C2), it can be inferred ex:MilesDavis rdf:type ex:Person.

---

2. **ex:BlueNoteRecords rdf:type ex:Label**

- **Proof**
  - The graph states: ex:BlueNoteRecords rdf:type ex:JazzLabel.
  - Also: ex:JazzLabel rdfs:subClassOf ex:Label.
  - **Subclass inference**: an instance of ex:JazzLabel is also an instance of ex:Label.
  - Therefore: ex:BlueNoteRecords rdf:type ex:Label.

---

3. **ex:CoolJazz rdf:type ex:Genre**

- **Proof**
  - The triple: ex:BlueNoteAlbum ex:belongsToGenre ex:CoolJazz.
  - ex:belongsToGenre rdfs:range ex:Genre.
  - **RDFS Range Rule**: if P rdfs:range C and x P y, then y rdf:type C.
  - Hence we infer ex:CoolJazz rdf:type ex:Genre.

---

4. **ex:CoolJazz rdf:type rdfs:Class**

- **Proof**
  - The triple: ex:CoolJazz rdfs:subClassOf ex:Jazz.
  - The assignment includes a special rule "**Subclasses are Classes**": if X rdfs:subClassOf Y, then X rdf:type rdfs:Class.
  - So it can be concluded ex:CoolJazz rdf:type rdfs:Class.

These four new statements are **not** in the original data but **follow** from the RDFS schema rules.

---

## B. Python Program to Compute All RDFS Entailments (5 marks)

The Python script (task_2.py) does the following

1. Loads the original G from a Turtle file.
2. Applies RDFS reasoning (using owlrl) to produce the closure.
3. Saves the entailed graph to T2Entailed.ttl.
4. (In the code for 2.C) verifies that the 4 new inferred triples are present.

**Usage**:

- pip install rdflib and pip install owlrl.
- Update the variable G_FILE inside the code to point to your actual .ttl input.
- Run python task_2.py.

***Solution:***

```
Loaded '29' triples from 'task2_data.ttl'.
After RDFS reasoning, we have '214' triples.
Added 50 new statements for 'subclass => class' rule.
After extension, graph has 214 triples.

Entailed graph saved to 'T2Entailed.ttl' with 214 triples.
```

---

## C. Verifying the 4 Derived Triples (8 marks)

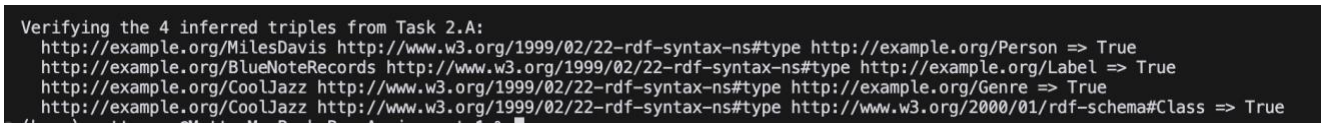
In the code, ***verify_inferred_triples*** function has been implemented that checks whether each of the four new inferred triples appears in the final entailed graph. Specifically:

1. The RDFS closure in entailed_graph has already been computed.

2. Each triple in the "derived" list, its tested if (subject, predicate, object) is in entailed_graph.

3. If it is present, print True; otherwise, False.

***Solution:***




This confirms that the entailed graph contains the new statements from Task 2.A. Since each triple is found to be True, we verify that the original graph entails them.