Example authority

Data products: review document

Contents

1 Introduction . . . 3
1.1 Choose an output . . . 3
2 Data product . . . 4
3 Example query . . . 5
A Document provenance . . . 6
Diagram interpretation . . . 6

1 Introduction

Authored guidance from a Markdown file explains how to read this ontology.

The data product definition is generated from OWL. The example query is collected as source text without execution.

1.1 Choose an output

The same collected guidance can serve different readers without maintaining a second copy of the prose.[1]

Read on the web
Follow links between the Markdown introduction, the OWL class definition and the SPARQL example. The sources remain distinct, while the publication provides one reading path.

Review a PDF
Download the review edition for a meeting or an offline reading session. Both tabs are included as labelled sections in the PDF, so no guidance is hidden behind an interaction that is unavailable on paper.

> **Retain the original sources**
> The example query is documentation, not a build step. GraphArch retains its text and highlights the syntax without executing the query.

Source	What it contributes
Markdown	Authored explanations, tabs, callouts and this table
OWL	The data product class and its definition
SPARQL	A readable query example, never executed during publication

[1] One collected revision supplies both editions, including this footnote. The exported documentation graph can be rendered after the original files are removed.

2 Data product

A governed collection of data with an accountable owner.

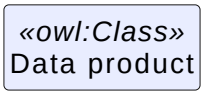


Diagram note

3 Example query

```
# grapharch:subject urn:mixed:query
PREFIX owl: <http://www.w3.org/2002/07/owl#>
SELECT ?product WHERE {
    ?product a owl:Class .
}
```

A Document provenance

This record identifies the generator and source revisions collected for this publication.

Generator	GraphArch 0.2.29 (build 412)
Generator commit	`8908a9610d77`

No Git repository revision was recorded for these sources.

Diagram interpretation

Class diagrams use UML shorthand for asserted OWL facts and declared SHACL constraints, without inference or a claim that validation passed. RDF properties remain independent resources even when drawn as class members or labelled associations. Domain/range declarations alone do not impose requiredness or cardinality. A property with a declared domain and range is a labelled association from the domain to the range; a datatype property, or one without a specific range, is an attribute of its domain. Multiple domains or ranges apply conjunctively, not as alternatives: each declared domain-range pair is drawn and holds for every use of the property. An explicitly declared union domain is drawn as one source box whose class alternatives are joined by or. A dotted rdfs:Resource source box is a presentation placeholder when no specific domain is declared in the selected inputs, not an asserted domain or a claim that none can be entailed. OWL class restrictions on the documented class and on neighbouring classes use UML notation instead of expression boxes, scoped to the restricted class and never declaring a global range or count. A datatype restriction is an attribute, name : type [multiplicity]; an object restriction is an association to its filler class with the multiplicity at the target end (a neighbouring class shows it as a typed attribute instead). An unqualified count uses the declared property range. someValuesFrom gives 1..* {qualified}; allValuesFrom gives * {all values}, narrowing the value type; a qualified count counts only values in the named filler and adds {qualified}; hasValue adds {= value}. Multiplicities follow one convention: equal bounds show one number such as 1, an unconstrained range shows *, and otherwise lower..upper such as 0..1 or 1..*. A maximum or exact count of zero, in OWL or SHACL, is a prohibition rather than a multiplicity: it appears as a class constraint row after the attributes, {no p}, or {no p in C} when only values in C are excluded. A zero minimum imposes no restriction and is omitted. {redefines} marks a restriction on a property whose declared domain is an asserted superclass, redefining the inherited property of the same name. An existential with a simple zero-minimum filler still requires a value; a positive count on the same class and property already guarantees that existence. Boolean class expressions, nested restrictions, inverse properties and property chains remain explicit boxes named after their form, such as Anonymous OWL intersection 1. Types and counts from simple property shapes enrich the UML notation. Identical qualifiers are shown once; where declaring shapes disagree, the disputed type or multiplicity is omitted from the drawing and remains in the dedicated shape documentation. No shape is chosen arbitrarily and conflicting counts are not silently intersected. An absent count imposes no lower or upper bound beyond zero or unbounded, respectively. Complex constraints require the linked full shape definition. Original OWL facts and shape provenance remain in the published source graph. Some diagrams show at most five complete OWL relationship groups within an eighteen-fact budget, selected in deterministic IRI order; omitted groups remain in the published source graph. These views are not exhaustive. Names from the same ontology as the documented class are shown by label; names from another ontology add their prefixed name in parentheses, and unlabelled names show their prefixed name.