Semantic Data Integration Assignment

Assignment: Semantic Integration Solution for Heterogeneous Data Sources

Assignment Requirements Checklist

Requirement 1: Identify Three Heterogeneous Data Sources

Status: COMPLETE

Created three distinct heterogeneous data sources:

1. **SQLite Relational Database** (`data_sources/students.db`)
   - 2 tables: Students (10 records), Enrollments (15 records)
   - Normalized relational schema with foreign keys
   - Page Reference: PROJECT_REPORT.md Section 2.1
2. **XML Semi-Structured File** (`data_sources/courses.xml`)
   - 10 courses with nested instructor elements
   - Hierarchical structure with attributes
   - Page Reference: PROJECT_REPORT.md Section 2.2
3. **CSV Tabular File** (`data_sources/departments.csv`)
   - 6 department records
   - Flat comma-separated format
   - Page Reference: PROJECT_REPORT.md Section 2.3

Evidence Files: - `generate_data_sources.py` - Data generation script - `data_sources/schemas.json` - Schema documentation

---

Requirement 2: Design a Domain Ontology

Status: COMPLETE

Created comprehensive OWL ontology for University Academic System:

Ontology Specifications: - **Namespace**: `http://university.edu/ontology#` - **Classes**: 6 (Person, Student, Instructor, Course, Department, Enrollment) - **Object Properties**: 7 (enrolledIn, teaches, offeredBy, hasEnrollment, enrollmentFor, majorIn, headedBy) - **Datatype Properties**: 20+ (firstName, lastName, email, studentID, courseCode, etc.) - **Total Ontology Triples**: 135

Evidence Files: - `ontology/university_ontology.owl` - Complete OWL ontology - Page Reference: PROJECT_REPORT.md Section 3

Key Design Features: - Class hierarchy (Person → Student/Instructor) - Domain/Range constraints on properties - XSD datatype specifications - Semantic relationships modeling real-world connections

---

Requirement 3: Create Semantic Mappings

Status: COMPLETE

Developed four comprehensive mappings transforming heterogeneous sources to RDF:

1. **SQLite Students → uni:Student**
   - Maps 10 students with 8 properties each
   - Creates majorIn relationships to departments
   - 90 total triples
2. **SQLite Enrollments → uni:Enrollment**
   - Maps 15 enrollments with relationship chains
   - Creates Student → Enrollment → Course links
   - 90 total triples
3. **XML Courses/Instructors → uni:Course and uni:Instructor**
   - Maps 10 courses and 8 instructors
   - Handles nested XML structure and deduplication
   - 84 total triples
4. **CSV Departments → uni:Department**
   - Maps 6 departments with fuzzy matching for heads
   - Links department heads to existing instructors
   - 36 total triples

Total RDF Instance Data: 452 triples

Evidence Files: - `integration/semantic_integration.py` - Complete mapping implementation - `mappings/mapping_documentation.json` - Detailed mapping specifications - Page Reference: PROJECT_REPORT.md Section 4

Requirement 4: Develop Integration Pipeline/Framework

Status: COMPLETE

Built Python-based semantic integration framework:

Framework Components: 1. Namespace Manager (UNI, DATA, OWL, RDFS, XSD) 2. Ontology Loader (loads base 135 triples) 3. Data Mappers (4 specialized mappers per source) 4. Relationship Resolver (cross-source entity references) 5. RDF Serializer (4 output formats) 6. Documentation Generator (mapping metadata)

Technologies Used: - Python 3.11 - rdflib 7.0+ (RDF/SPARQL processing) - pandas 2.3.3 (CSV processing) - sqlite3 (database access) - xml.etree.ElementTree (XML parsing)

Performance Metrics: - Integration Time: <1 second - Total Graph Size: 587 triples (452 instance + 135 ontology) - Success Rate: 100%

Evidence Files: - `integration/semantic_integration.py` - Main integration script - `output/integrated_data.ttl` - Turtle format output - `output/integrated_data.rdf` - RDF/XML format output - `output/integrated_data.n3` - Notation3 format output - `output/integrated_data.nt` - N-Triples format output - Page Reference: PROJECT_REPORT.md Section 5

Requirement 5: Use SPARQL to Query and Validate

Status: COMPLETE

Implemented comprehensive SPARQL query engine with 6 demonstration queries and 4 validation tests:

Demonstration Queries:

1. **Query 1: Students with Majors and GPAs**
   - Cross-source join (SQLite + CSV)
   - Ordered by GPA descending
   - Result: 10 students
2. **Query 2: Courses by Instructor**
   - One-to-many relationships
   - Result: 10 course assignments, 8 instructors
3. **Query 3: Student Enrollments with Grades**
   - Three-source join (SQLite Students + Enrollments + XML Courses)
   - Complex relationship traversal
   - Result: 15 enrollments
4. **Query 4: Department Statistics**
   - Aggregation with COUNT and GROUP BY
   - Result: 6 departments with course counts
5. **Query 5: Computer Science Students**
   - Filtered query by major
   - Result: CS student enrollments
6. **Query 6: 4-Credit Courses**
   - Literal value filtering
   - Multi-source integration
   - Result: 4 high-credit courses

Validation Tests:

1. **Property Completeness**: 100% (10/10 students have all required properties)
2. **Referential Integrity**: 100% (15/15 enrollments link to valid students/courses)
3. **Instructor Assignment**: 100% (10/10 courses have instructors)
4. **Datatype Consistency**: 100% (10/10 students have valid GPAs 0.0-4.0)

Cross-Source Validation: - All course codes in enrollments exist in course catalog (10/10) - All department references are valid (6/6) - All student-course relationships preserved (15/15)

Overall Result: **SEMANTIC CONSISTENCY VALID (100%)**

Evidence Files: - `queries/sparql_queries.py` - Complete query engine - `queries/query_results.json` - All query results - `queries/query_1_results.json` through `query_6_results.json` - Individual results - Page Reference: PROJECT_REPORT.md Section 6

Summary of Deliverables

Source Code Files

File	Purpose	Lines of Code
`generate_data_sources.py`	Generate heterogeneous data sources	120
`integration/semantic_integration.py`	Semantic integration framework	177
`queries/sparql_queries.py`	SPARQL query & validation engine	515
Total		**812 lines**

Data Files

File	Description	Size
`data_sources/students.db`	SQLite database	24 KB
`data_sources/courses.xml`	XML course catalog	2.1 KB
`data_sources/departments.csv`	CSV departments	512 bytes
`ontology/university_ontology.owl`	OWL ontology	9.4 KB
`output/integrated_data.ttl`	RDF Turtle format	5.7 KB
`output/integrated_data.rdf`	RDF/XML format	23 KB
`output/integrated_data.n3`	RDF N3 format	5.8 KB
`output/integrated_data.nt`	RDF N-Triples format	9 KB

Documentation Files

File	Description	Pages
`README.md`	Project overview & quick start	8 pages
`PROJECT_REPORT.md`	Comprehensive technical report	35 pages

File	Description	Pages
`ASSIGNMENT_SUBMISSION.md`	This document	7 pages
Total Documentation		**50 pages**

Key Results and Achievements

Quantitative Results

- **3** heterogeneous data sources unified
- **49** total entities integrated (10 students, 8 instructors, 10 courses, 6 departments, 15 enrollments)
- **587** total RDF triples (452 instance + 135 ontology)
- **6** SPARQL demonstration queries
- **4** validation tests with 100% success
- **4** RDF serialization formats
- **100%** semantic consistency validation

Quality Metrics

- **Data Completeness**: 100% (no data loss)
- **Relationship Accuracy**: 100% (all foreign keys preserved)
- **Semantic Consistency**: 100% (all constraints satisfied)
- **Query Correctness**: 100% (all queries return expected results)
- **Integration Performance**: <1 second total execution time

Technical Achievements

1. Successfully unified relational, semi-structured, and tabular data
2. Designed comprehensive domain ontology with proper OWL semantics
3. Implemented robust cross-source entity resolution
4. Achieved seamless SPARQL querying across all sources
5. Validated semantic consistency through automated tests

How to Verify This Work

Step 1: Install Dependencies

```
cd semantic_integration
pip install -r requirements.txt
```

Step 2: Generate Data Sources

```
python generate_data_sources.py
```

Expected Output: Creates students.db, courses.xml, departments.csv

Step 3: Run Semantic Integration

```
python integration/semantic_integration.py
```

Expected Output: 452 RDF triples, 4 serialization formats

Step 4: Execute SPARQL Queries

```
python queries/sparql_queries.py
```

Expected Output: 6 queries executed, 100% validation success

Project Structure for Review

```
semantic_integration/
  README.md                          # Project overview
  PROJECT_REPORT.md                  # 35-page comprehensive report
  ASSIGNMENT_SUBMISSION.md           # This submission document
  requirements.txt                   # Python dependencies
  generate_data_sources.py           # Data source generator

  data_sources/                      # Heterogeneous data sources
      students.db                   # SQLite database
      courses.xml                   # XML file
      departments.csv               # CSV file
      schemas.json                  # Schema documentation

  ontology/                          # Domain ontology
      university_ontology.owl       # OWL ontology (135 triples)

  mappings/                          # Mapping specifications
      mapping_documentation.json    # Complete mapping docs

  integration/                       # Integration framework
      semantic_integration.py       # Main integration script

  queries/                           # SPARQL queries
      sparql_queries.py             # Query engine
      query_results.json            # All results
      query_*_results.json          # Individual query results

  output/                            # RDF exports
       integrated_data.ttl           # Turtle format
       integrated_data.rdf           # RDF/XML format
```

```
integrated_data.n3              # N3 format
integrated_data.nt              # N-Triples format
```

Documentation

Primary Document: `PROJECT_REPORT.md`

This 35-page comprehensive report includes: - Executive Summary - Complete technical documentation of all 5 requirements - Detailed analysis of heterogeneous data sources - Ontology design rationale - Semantic mapping specifications - Integration framework architecture - SPARQL query examples and results - Validation methodology and results - Challenges and solutions - Lessons learned - Applications and use cases - Conclusions

---

Conclusion

This assignment successfully demonstrates a complete semantic data integration solution that: - Unifies 3 heterogeneous data sources into a single queryable knowledge base - Uses W3C standard technologies (RDF, OWL, SPARQL) - Achieves 100% semantic consistency - Enables seamless cross-source querying - Provides comprehensive documentation

All assignment requirements have been met and validated with quantitative evidence.

Github Repository

The complete project is available at semantic_integration directory with all source code, data files, and documentation.