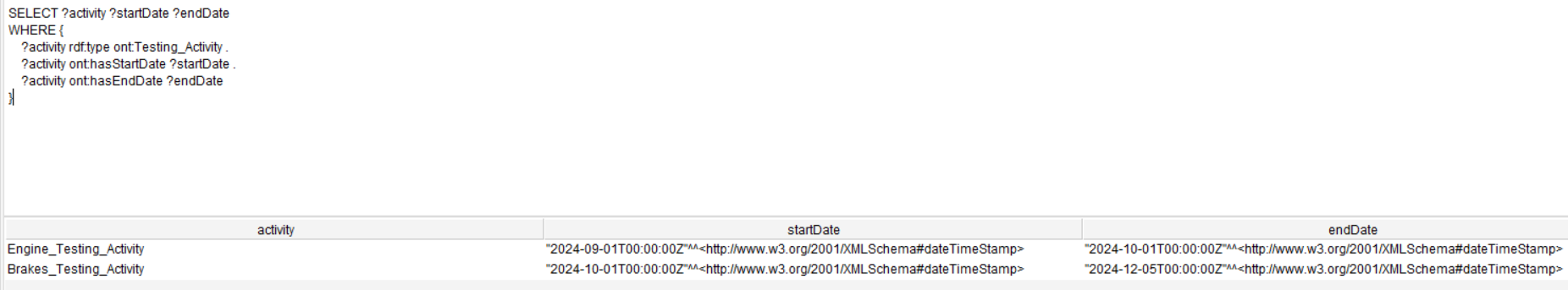
Domain: Testing Process and Activities with closer look at Unit Tesing

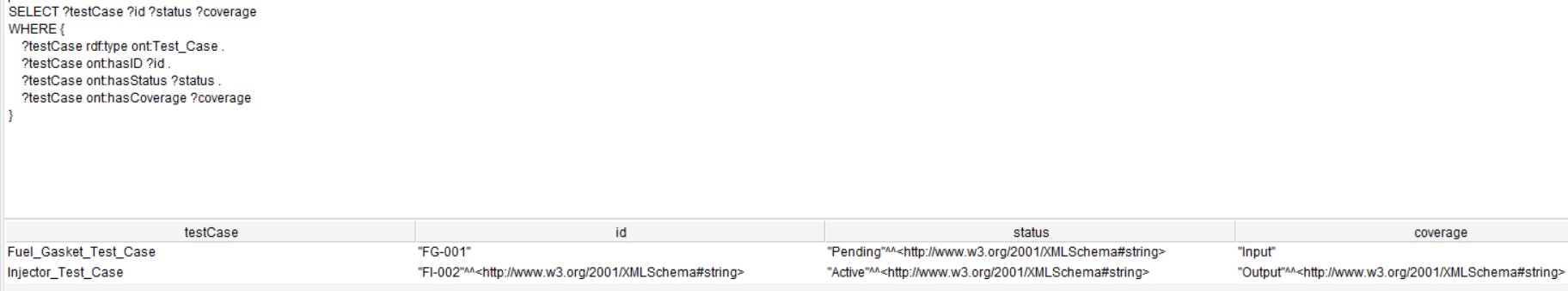
Questions:

1. Retrieves all testing activities with their start and end dates
2. Finds test cases and their associated metadata
3. Explores boundary testing and bound information
4. Lists testing activities within a specific testing process
5. Counts different types of testing that are subclasses of Unit Testing
6. Is there are test case with ID “FG-001”

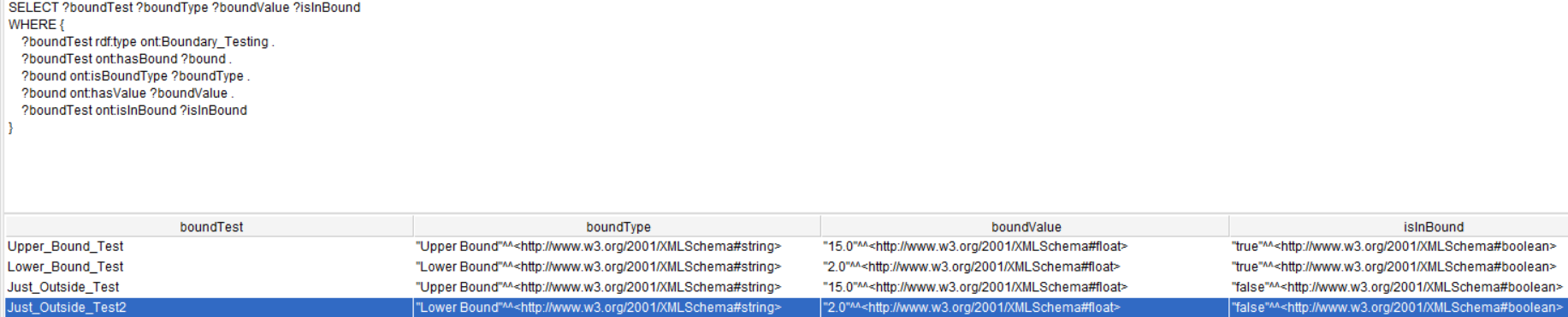
1.



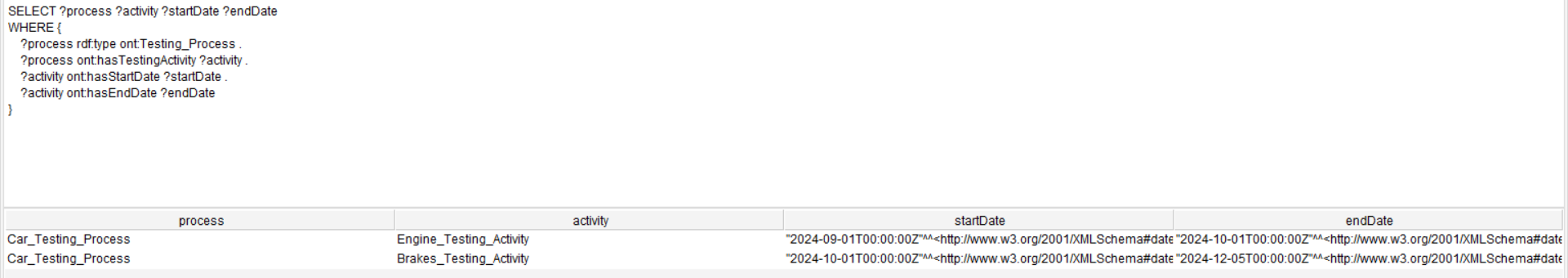
2.



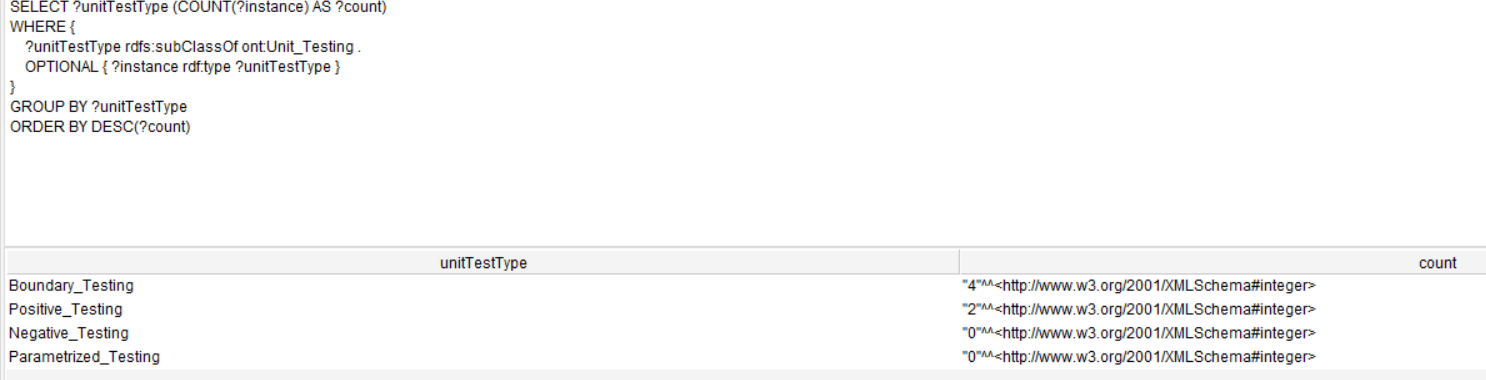
3.



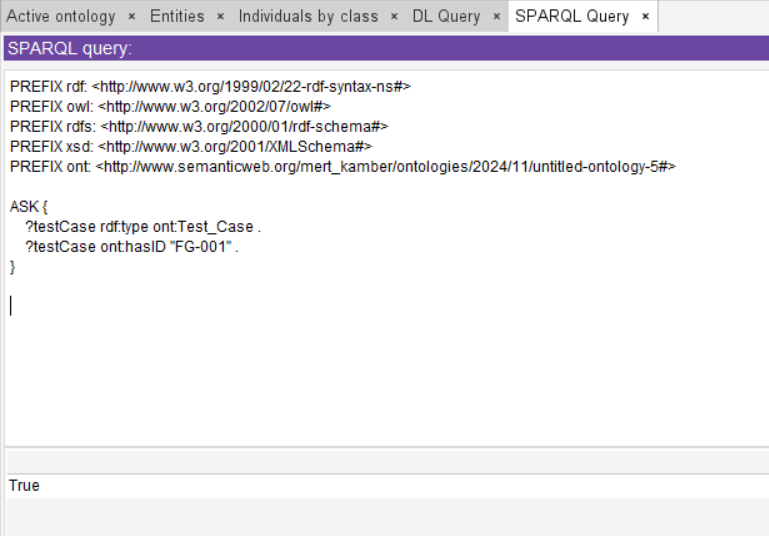
4.



5.



6.



# Intro

 **Software Verification and Validation (V&V):** Ensures conformity with specifications and satisfaction of user needs.

 **Software Testing:** A dynamic V&V process comparing actual versus expected software behavior.

 **Knowledge Management (KM):** Critical for organizing, sharing, and reusing knowledge in software testing.

 **Ontologies:** Provide structured, shared vocabularies for effective KM and support decision-making.

 **ROoST Ontology:** A proposed reference ontology addressing gaps in existing software testing ontologies for better KM and knowledge integration.

# Background

## Software testing

 **Verification and Validation (V&V):** Activities to ensure software quality, recognized by standards like CMMI, ISO 12207, and IEEE 29119.

 **Dynamic and Static V&V:** Testing is dynamic V&V requiring program execution, unlike static methods (e.g., inspections).

 **Testing Process:** Includes test planning, test case design, test execution, and result analysis, forming a structured approach.

 **Testing Techniques:**

* **Black-box:** Focus on input/output without code insight.
* **White-box:** Uses internal code structure.
* **Defect-based:** Targets likely faults.
* **Model-based:** Uses formal specifications like Finite State Machines.

 **Testing Levels:**

* **Unit Testing:** Individual components.
* **Integration Testing:** Combined components.
* **System Testing:** Entire system behavior.

 **Testing Artifacts:** Test plans, cases, results, and reports guide and document the process.

 **Test Environment:** Includes hardware, software, and human resources, supporting robust testing practices.

## Testing ontologies

 **Knowledge Reuse in Testing:**

* Enhances efficiency by reusing past experiences or test cases.
* Facilitates knowledge sharing through reference ontologies.

 **Domain Ontology for Software Testing:**

* Provides a shared vocabulary for knowledge management.
* Supports communication, integration, and representation.

 **Systematic Literature Review (SLR):**

* Identified 12 existing testing ontologies.
* Evaluated their coverage, adherence to standards, and structure.

 **Characteristics of Good Ontologies:**

* Comprehensive coverage, international standards compliance, and robust structure.
* Incorporates concepts, relationships, axioms, and evaluation methodologies.

 **Future Needs:**

* Development of high-quality, reference software testing ontologies.
* Creation of operational versions based on these references.