# Test Case for Knowledge Graph Relationship Validation

## Test Case ID: KG-REL-001

### Test Case Title:

Verify Organization Relationship Security Rules between Sponsor and Supplier Organizations

### Test Objective:

To validate that the correct security constraints are applied when establishing relationships between Sponsor and Supplier organizations in the Knowledge Graph.

### Pre-conditions:

- Knowledge Graph design documentation is available  
- Relationship rules between organizations are defined  
- Security role definitions are documented  
- [Zeigo Hub Flow: Lucidchart](https://lucid.app/lucidchart/9314b1ae-fc3d-47c3-993b-037221f7f05e/edit?page=~qw_OeySQ3cP)

### Test Data Requirements:

- Sample Sponsor Organization: "SponsrCorp"  
- Sample Supplier Organization: "SupplyCo"  
- Security Role: "ZeigoWriterRole"  
- Required Relationship Type: "supplier\_to\_sponsor"

### Test Steps:

1. Model Review - Relationship Definition

- Review the Knowledge Graph design to confirm that "Organization Relationships" entity connects Sponsor and Supplier groups

- Expected Result: Design shows "Organization Relationships" as the joining entity with appropriate connections

- Actual Result: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

- Pass/Fail: \_\_\_\_\_\_\_

2. Model Review - Security Rules

- Review security role definitions to determine if "ZeigoWriterRole" has appropriate permissions for cross-organization relationships

- Expected Result: ZeigoWriterRole permissions should include ability to view but not modify relationship data

- Actual Result: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

- Pass/Fail: \_\_\_\_\_\_\_

3. Walkthrough Validation - Creation Process

- Perform a walkthrough of the relationship creation process between sponsor and supplier

- Expected Result: Process requires appropriate approvals and validation steps

- Actual Result: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

- Pass/Fail: \_\_\_\_\_\_\_

4. Design Analysis - Access Constraints

- Analyze access pattern designs to validate that supplier data is properly segmented from sponsor data

- Expected Result: Design shows clear boundaries with explicit sharing mechanisms

- Actual Result: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

- Pass/Fail: \_\_\_\_\_\_\_

5. Security Model Analysis - Permission Inheritance

- Review how permissions propagate through relationships

- Expected Result: Permission model shows clear rules for inheritance that prevent unintended access

- Actual Result: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

- Pass/Fail: \_\_\_\_\_\_\_

6. Prototype Verification - Mock Security Model (if prototype available)

- Use any available prototype or model to verify relationship behavior

- Expected Result: Relationships enforce appropriate constraints

- Actual Result: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

- Pass/Fail: \_\_\_\_\_\_\_

### Post-conditions:

- All identified issues in the relationship model are documented  
- Recommendations for improvements to the security model are provided  
- Design requirements are updated based on findings

### Test Artifacts:

- Marked-up diagram showing potential security vulnerabilities  
- Documentation of permission model inconsistencies or gaps  
- List of recommended design changes

### Risk Assessment:

- HIGH: Improper relationship security could lead to unauthorized data access between organizations  
- MEDIUM: Unclear permission inheritance could create maintenance challenges  
- LOW: Relationship naming conventions may cause confusion during implementation

### Test Environment:

- No environment needed; this is a document/design review test

### Execution Method:

- Manual review and analysis  
- Cross-reference with requirements documentation  
- Security model review by domain experts

### Estimated Duration:

- 2-3 hours for complete review and documentation