

GUIDE

erwin Data Modeler

RDF/TTL Semantic Export



Legal Notices

This Documentation, which includes embedded help systems and electronically distributed materials (hereinafter referred to as the "Documentation"), is for your informational purposes only and is subject to change or withdrawal by Quest Software, Inc and/or its affiliates at any time. This Documentation is proprietary information of Quest Software, Inc and/or its affiliates and may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of Quest Software, Inc and/or its affiliates

If you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all Quest Software, Inc and/or its affiliates copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to Quest Software, Inc and/or its affiliates that all copies and partial copies of the Documentation have been returned to Quest Software, Inc and/or its affiliates or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, QUEST SOFTWARE, INC. PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL QUEST SOFTWARE, INC. BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF QUEST SOFTWARE, INC. IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is Quest Software, Inc and/or its affiliates.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c) (1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Copyright © 2026 Quest Software, Inc and/or its affiliates All rights reserved. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Contact erwin

Understanding your Support

Review [support maintenance programs and offerings](#).

Registering for Support

Access the [erwin support](#) site and register for product support.

Accessing Technical Support

For your convenience, erwin provides easy access to "One Stop" support for all editions of [erwin Data Modeler](#), and includes the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- erwin Support policies and guidelines
- Other helpful resources appropriate for your product

For information about other erwin products, visit [erwin by Quest Products page](#).

Provide Feedback

If you have comments or questions, or feedback about erwin product documentation, you can send a message to techpubs@erwin.com.

News and Events

Visit [News and Events](#) to get up-to-date news, announcements, and events. View video demos and read up on customer success stories and articles by industry experts.

Contents

RDF/TTL Semantic Export	5
Why Use RDF/TTL Export	5
Export a Model as RDF/TTL	6
Scenario A: Enterprise Data Catalog Onboarding	8
Approach 1: AI Agent Enrichment	8
Approach 2: Direct Ingestion by Semantic Tooling	8
Scenario B: Pre-Deployment Contract Validation	9
Approach 1: AI Agent Generated Validation Tests	9
Approach 2: Direct Constraint Ingestion in CI	10

RDF/TTL Semantic Export

erwin Data Modeler (erwin DM) lets you export your models as RDF (Resource Description Framework) and Turtle (TTL). This capability adds business meaning to your existing data models so that knowledge graphs, semantic tools, governance platforms, and AI agents can use them directly.

Traditionally, erwin DM captures database structure: tables, columns, keys, data types, and relationships. It exports that structure as an .erwin model, SQL DDL, or XML. With RDF/TTL export, the same model becomes a semantic asset. Each foreign key becomes a named, business-meaningful relationship instead of a plain reference.

Why Use RDF/TTL Export

RDF/TTL export extends traditional data modeling by adding semantic context that supports governance, interoperability, and AI use cases.

Traditional Data Modeling	Data Modeling with RDF/TTL
Database design	Database design plus business knowledge
SQL generation	Knowledge graph generation
Used by database administrators	Used by database administrators, governance teams, and AI agents
Foreign keys	Semantic relationships
Human-readable models	Machine-understandable context

When you use RDF/TTL export, you get these benefits:

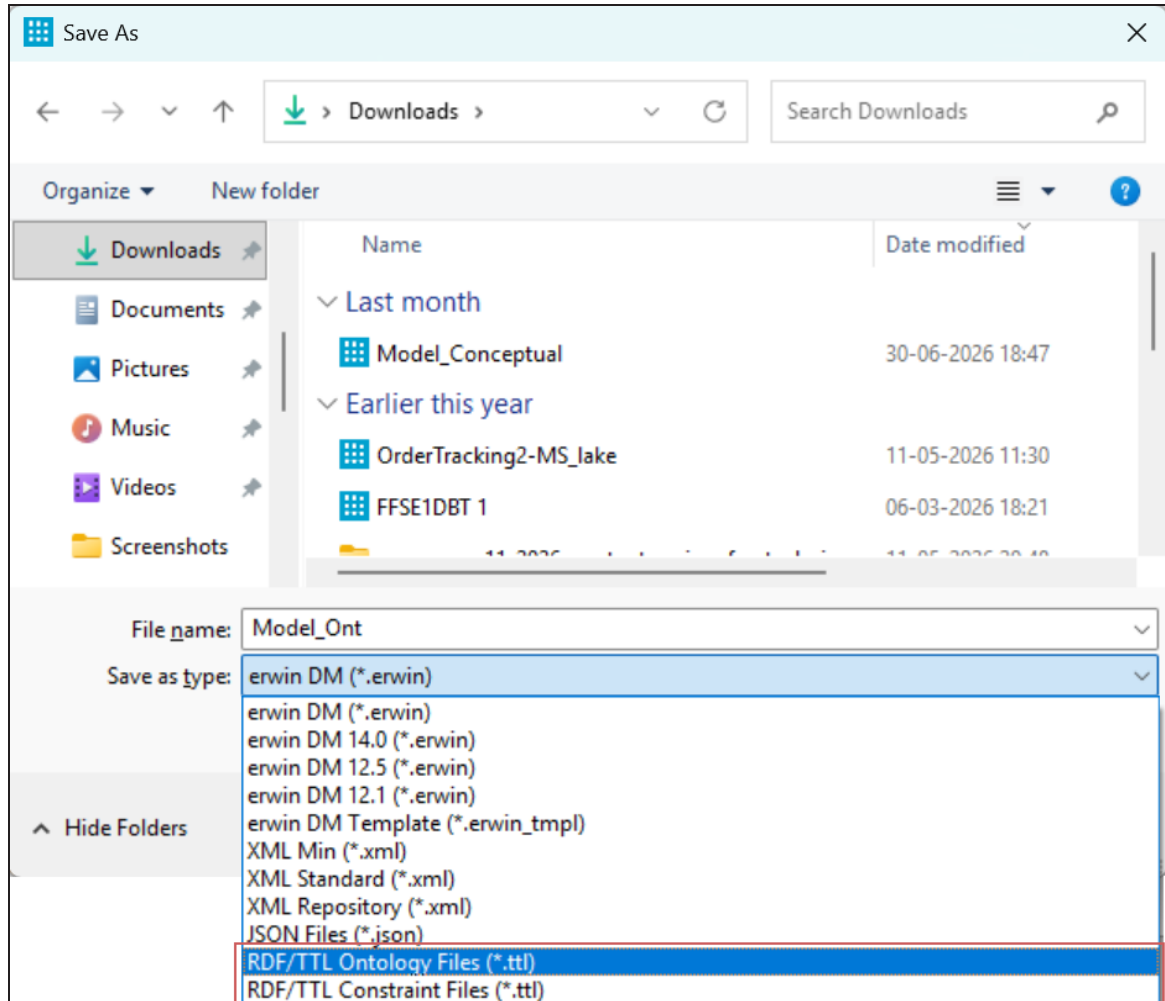
- Existing data models become a foundation for enterprise knowledge graphs.
- Business definitions and relationships are reused by semantic tools instead of being recreated manually.
- AI agents gain structured context about entities such as Customer, Product, Order, and Store, which improves the accuracy of generated answers and reduces incorrect answers.
- Integration with governance platforms and semantic technologies is easier because RDF/TTL is an open standard.

- Your organization can build graph-based applications while continuing to use erwin DM as the primary modeling environment.

Export a Model as RDF/TTL

To export a model as RDF/TTL:

1. Open a model and select **Save As** from the **File** menu.
The Save As window appears.
2. On the Save As window, enter a name for the file in the **File name** field and select one of the following options from the **Save as type** drop-down.
 - **RDF/TTL Ontology Files (*.ttl)**: Files that define the structure, vocabulary, and semantics of a domain. They contains classes along with properties and relationships derived from the model.
 - **RDF/TTL Constraint Files (*.ttl)**: Files that define validation rules that RDF data must comply. They represent sample or reference data.



3. Select **Save**.

The export generates a standards-based TTL file that any RDF-compliant tool can read.

To use the model, import the exported TTL file into your target semantic platform. Because the output follows standard RDF/TTL specifications, no proprietary connectors are required, making it easy to move and reuse the model across RDF-compliant tools and environments.

The following scenarios show how AI agents and semantic tools use exported RDF/TTL to solve real enterprise problems. Each scenario includes two approaches:

- An **AI-agent-driven approach**, where an AI agent parses the exported RDF/TTL and performs additional reasoning or automation.

- A **direct ingestion approach**, where semantic tooling or CI pipelines consume the exported RDF/TTL without an intermediate AI agent.

Scenario A: Enterprise Data Catalog Onboarding

Consider a scenario when your organization needs to onboard model semantics into an enterprise data catalog or knowledge graph.

Problem: Data governance teams struggle to ingest ER model semantics into RDF-based catalog and knowledge graph tools because model metadata is trapped in proprietary structures that aren't directly consumable by semantic tooling.

Solution: You export the model as an RDF/TTL Ontology File. The exported file includes classes and datatype properties that represent your model's entities and attributes, so the semantics can be ingested directly by semantic tooling or parsed by an AI agent for further enrichment.

Approach 1: AI Agent Enrichment

In this approach, an AI agent parses the exported RDF/TTL Ontology File and enriches your enterprise catalog automatically.

1. The architect uses erwin DM to export the model as .ttl and provides it to the AI agent.
2. The AI agent parses the exported classes and properties.
3. The AI agent links business glossary terms to the parsed classes and properties, and flags any gaps it finds.
4. The catalog is automatically enriched with the linked terms and flagged gaps.

Example output:

```
erm:Customer a owl:Class ;  
             rdfs:label "Customer" .
```

Approach 2: Direct Ingestion by Semantic Tooling

In this approach, semantic tooling ingests the exported RDF/TTL Ontology File directly, without an AI agent as an intermediary.

To get the an RDF/TTL Ontology File, follow these steps:

1. Select a model in erwin DM.
2. Save the model as an RDF/TTL Ontology File.

3. Review the output. It includes an **owl:Class** entry for each entity and an **owl:DatatypeProperty** entry for each attribute.

Then, your catalog pipeline ingests the TTL file and automatically links terms to your enterprise ontology. Your governance team queries lineage and definitions across systems using RDF and SPARQL workflows.

Example output:

```
erm:customer_address a owl:DatatypeProperty ;  
    rdfs:domain erm:Customer ;  
    rdfs:range xsd:string.
```

Scenario B: Pre-Deployment Contract Validation

Consider a scenario when your team needs to validate data contracts, such as required fields and cardinality, before deploying a model-backed schema.

Problem: Teams deploying JSON, OpenAPI, or BigQuery-backed models often miss requiredness and cardinality constraints, which causes runtime validation failures after deployment.

Solution: You export the model using constraint mode and generate the RDF/TTL Constraint File. Constraint mode converts attribute metadata, such as required fields, minimum and maximum item counts, and nullability, into shape-style TTL constraints so that constraints can be validated before release.

Approach 1: AI Agent Generated Validation Tests

In this approach, an AI agent reads the semantic constraints in the exported RDF/TTL Constraint File and generates validation tests for your CI pipeline.

1. Export the model as an RDF/TTL Constraint File.
2. The AI agent extracts required fields and data types from the exported constraints.
3. The AI agent generates positive and negative test payloads based on the extracted constraints.
4. Your CI pipeline validates the contracts using the generated payloads before release.

Example output:

```
erm:OrderShape a sh:NodeShape ;  
    sh:targetClass erm:Order.
```

Approach 2: Direct Constraint Ingestion in CI

In this approach, your CI pipeline ingests the generated shape constraints directly, without an AI agent generating the test payloads.

1. Model a BigQuery, OpenAPI, or similar entity that includes array fields and required fields.
2. Save the model as an RDF/TTL Constraint File.
3. Review the generated TTL. It encodes cardinality and datatype constraints as **sh:minCount**, **sh:maxCount**, and **datatype** statements.
4. Your CI step validates sample payloads against the generated shapes before deployment.

Constraint mode is intended for pre-deployment validation. Review generated shapes against your actual schema requirements before you rely on them in a production CI pipeline.