

# STAR-Q: A Web-Based Platform for Annotating Qualified Knowledge Graph Statements in Texts and Tables

Thibault Ehrhart<sup>1</sup>, Yoan Chabot<sup>2</sup> and Raphael Troncy<sup>1</sup>

<sup>1</sup>EURECOM Sophia Antipolis, France

<sup>2</sup>Orange Research Belfort, France

## Abstract

The construction and evaluation of knowledge graph (KG) population systems depend on high-quality, entity-grounded annotated corpora. However, producing such corpora remains a laborious task: general-purpose annotation tools do not often support the grounding of annotations to KG identifiers, rarely handle tabular content, and provide little support for corpus-level quality control. In this paper, we present STAR-Q (Span and Table Annotation for RDF with *Qualifiers*), an open-source, web-based annotation platform designed to facilitate the annotation of both texts and tables with knowledge graph entities. STAR-Q lets users highlight mentions corresponding to subjects, predicates, objects, and ordered qualifier predicate-value pairs directly in documents, and then disambiguate them to entities and properties defined in a reference knowledge graph through a configurable lookup interface targeting any Wikibase instance. Wikidata is supported out of the box, with its large set of QIDs and PIDs. Private Wikibase deployments, such as enterprise knowledge graphs, can also be registered and selected per corpus. Annotations use a unified component schema covering both textual spans and character ranges within table cells; resolved values may be KG resources, corpus-local entities and relations, or typed literals. Beyond annotation, STAR-Q provides corpus management and import adapters for structured JSON/JSONL, archive-based corpus formats, Label Studio exports, and its own versioned full-corpus JSON. Annotations generated by the tool can be exported in RDF 1.2, either in a "truthy" statement mode or in a "full" mode preserving component-level text and table evidence using well-known ontologies (DCAT, PROV, NIF, CSVW, Web Annotations). Wikidata-grounded statements can additionally be exported as *QuickStatements* 3.0 commands for direct submission to Wikidata. We demonstrate STAR-Q through two scenarios: a didactic scenario based on Wookieepedia texts annotated with Wikidata entities curated by the WikiProject Star Wars community effort, and an industrial in-use scenario, where technical telecommunications documents are annotated against an enterprise knowledge graph. The tool is open source at <https://github.com/ecladatta/star-q>.

## Keywords

Text Annotation, Table Annotation, Entity Linking, Knowledge Graph Construction, Wikidata.

## 1. Introduction

Knowledge graphs are increasingly populated from heterogeneous document collections using automatic extraction systems, ranging from pipeline-based entity linking and relation classification approaches to LLM-based extractors [1, 2, 3, 4]. Regardless of the extraction paradigm, two needs always arise: (i) reference corpora annotated with KG-grounded triples, required to train, prompt, evaluate, and compare information extraction systems, and (ii) human curation interfaces, required to produce these reference annotations efficiently and consistently. Recent work on uncertainty management in KG construction further stresses the importance of traceable, human-validated ground truth throughout the KG lifecycle [5].

Producing entity-grounded annotated corpora is, however, considerably harder than producing classical span-level annotations. Annotators must not only delimit mentions in the source document, but also disambiguate them against a target knowledge graph such as Wikidata [6], select appropriate properties, represent qualifiers, handle literal values with their datatypes, and deal with entities that do not yet exist in the KG. The difficulty increases further when documents mix free text and tables: tables are pervasive in technical and scientific documents and carry dense factual content, but their

---

Wikidata'26: Wikidata Workshop at ISWC 2026

✉ [thibault.ehrhart@eurecom.fr](mailto:thibault.ehrhart@eurecom.fr) (T. Ehrhart); [yoan.chabot@orange.com](mailto:yoan.chabot@orange.com) (Y. Chabot); [raphael.troncy@eurecom.fr](mailto:raphael.troncy@eurecom.fr) (R. Troncy)

🆔 0000-0003-1377-8279 (T. Ehrhart); 0000-0001-5639-1504 (Y. Chabot); 0000-0003-0457-1436 (R. Troncy)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

annotation requires positional models (rows, cells, columns) that differ from character-offset-based text annotation, as shown by work on semantic table interpretation [7, 8].

Existing annotation tools only partially address these needs. General-purpose platforms such as BRAT [9], WebAnno [10], INCEpTION [11], doccano [12], and Label Studio<sup>1</sup> excel at configurable span, region, and relation annotation. INCEpTION additionally supports entity linking against external KGs, while Label Studio can display tabular data and annotate document layouts. However, none of them combines, in a single workflow, the annotation of both texts and tables with native statement-oriented annotation (subject/predicate/object with optional qualifiers) grounded in Wikidata identifiers. None manages corpus-local entities and relations absent from the target KG, or provide corpus-level analytics and consistency checking dedicated to triple completeness.

In this paper, we present STAR-Q, an open-source, web-based platform designed to fill this gap. STAR-Q organizes annotation work around corpora of documents. Within a document, the user highlights subject, predicate, object, and qualifier components in text or in tables and associates them with KG entities suggested by a configurable lookup interface. The target Wikibase instance (Wikidata by default) and the corpus-local vocabulary are searched together. Corpus-local entities and relations can be created once and reused across documents when the target KG lacks the required resource. The platform tracks document completion and grounding coverage, distinguishes text-only, table-only, and joint statements, and provides statistics by property and entity. Corpora can be exported to and imported from a versioned JSON format. RDF export is available in two modes: one containing KG-ready statements and another one that also includes source evidence and provenance for every exported statement component (Figure 3).

The main contributions of this paper are the following:

1. STAR-Q, an open-source web platform for annotating KG statements, including ordered qualifier predicate-value pairs, across texts, tables, and joint text-table evidence;
2. a unified annotation schema whose components carry KG grounding (including corpus-local entities and relations and typed literals) and fine-grained provenance in text (element-relative character offsets) and tables (row/cell indices and within-cell character offsets);
3. corpus-level services supporting the annotation lifecycle: heterogeneous import, corpus and document management, analytics, consistency checking, re-importable versioned JSON, and truthy and provenance-rich RDF 1.2 export;
4. a demonstration of the system on two scenarios: Wookieepedia texts annotated with Wikidata entities from the WikiProject Star Wars, and technical telecommunications documents annotated against an enterprise knowledge graph.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 describes the architecture of STAR-Q and its main features. Section 4 presents two complementary demonstration scenarios. Finally, Section 5 concludes the paper detailing some limitations and outlines future works.

## 2. Related Work

We focus on tools supporting manual or semi-automatic annotation of documents for knowledge graph population, and position STAR-Q with respect to them.

**Span and relation annotation tools.** BRAT [9] popularized web-based span and relation annotation and remains widely used for information extraction datasets. WebAnno [10] and its successor INCEpTION [11] extended this paradigm with configurable annotation layers, machine-assisted pre-annotation through recommenders, and multi-annotator curation workflows including inter-annotator agreement computation. In addition, INCEpTION provides entity linking recommenders that can suggest identifiers from external KGs, including Wikidata. Label Studio also supports ML-assisted

---

<sup>1</sup><https://labelstud.io>

pre-annotation and review workflows. doccano [12] and Label Studio target lightweight labeling tasks (classification, NER, sequence-to-sequence) with modern web interfaces. Prodigy<sup>2</sup> focuses on scriptable, active-learning-driven annotation. These tools are mature and extensible, but they are text-centric: none of them provides native support for annotating tabular content with positional provenance, and triple-oriented annotation (explicitly distinguishing subject, predicate, and object components grounded in KG identifiers) must be emulated through generic relation layers.

**Table annotation and semantic table interpretation.** The SemTab challenge [7] has driven the development of systems matching tabular data to KGs through cell-entity annotation (CEA), column-type annotation (CTA), and column-property annotation (CPA) [13]. Systems such as DAGOBAB [8] and MantisTable [14] automate these tasks, and some provide interfaces for inspecting results [15]. However, these tools are geared toward automatic annotation of standalone tables rather than manual ground-truth construction over mixed text-and-table documents, and they do not provide corpus management, custom entity handling, or triple-completeness checking.

**Ground-truth construction for KG population.** Datasets aligning text with Wikidata triples, such as T-REx [16], are typically built through automatic distant supervision, which introduces noise that manual curation could avoid. On the curation side, platforms such as TripleFlow [17] support human-in-the-loop validation of automatically extracted, Wikidata-grounded triples. STAR-Q is complementary to this family of tools: rather than validating machine-extracted triples, it targets the upstream creation of manually annotated, entity-grounded reference corpora, covering both texts and tables, which can then be used to evaluate or train the extractors that validation platforms orchestrate.

Unlike the systems described above, STAR-Q combines, in a single platform, text and table annotation, native Wikidata-grounded statements with qualifiers, corpus-local entity and relation management, and corpus analytics with consistency checking. STAR-Q deliberately focuses on manual, high-quality ground-truth construction: it does not yet provide ML-assisted pre-annotation. Corpora can be shared within teams with role-based access, but explicit multi-annotator curation with conflict resolution is left for future work. Table 1 summarizes the comparison between STAR-Q and existing annotation tools.

**Table 1**  
Comparison of STAR-Q with existing annotation tools.

| System        | Text<br>annotation | Table<br>annotation | KG<br>grounding | ML pre-<br>annotation | Local<br>entities | Analytics<br>& checks | Collaborative<br>curation |
|---------------|--------------------|---------------------|-----------------|-----------------------|-------------------|-----------------------|---------------------------|
| BRAT          | ✓                  | ✗                   | ✗               | ✗                     | ✗                 | ✗                     | ✗                         |
| INCEpTION     | ✓                  | ✗                   | ✓               | ✓                     | ✗                 | ✗                     | ✓                         |
| doccano       | ✓                  | ✗                   | ✗               | ✗                     | ✗                 | ✗                     | ✗                         |
| Label Studio  | ✓                  | ✗                   | ✗               | ✓                     | ✗                 | ✗                     | ✓                         |
| MantisTable   | ✗                  | ✓                   | ✓               | ✓                     | ✗                 | ✗                     | ✗                         |
| TripleFlow    | ✗                  | ✗                   | ✓               | ✓                     | ✗                 | ✗                     | ✗                         |
| <b>STAR-Q</b> | ✓                  | ✓                   | ✓               | ✗                     | ✓                 | ✓                     | ✗                         |

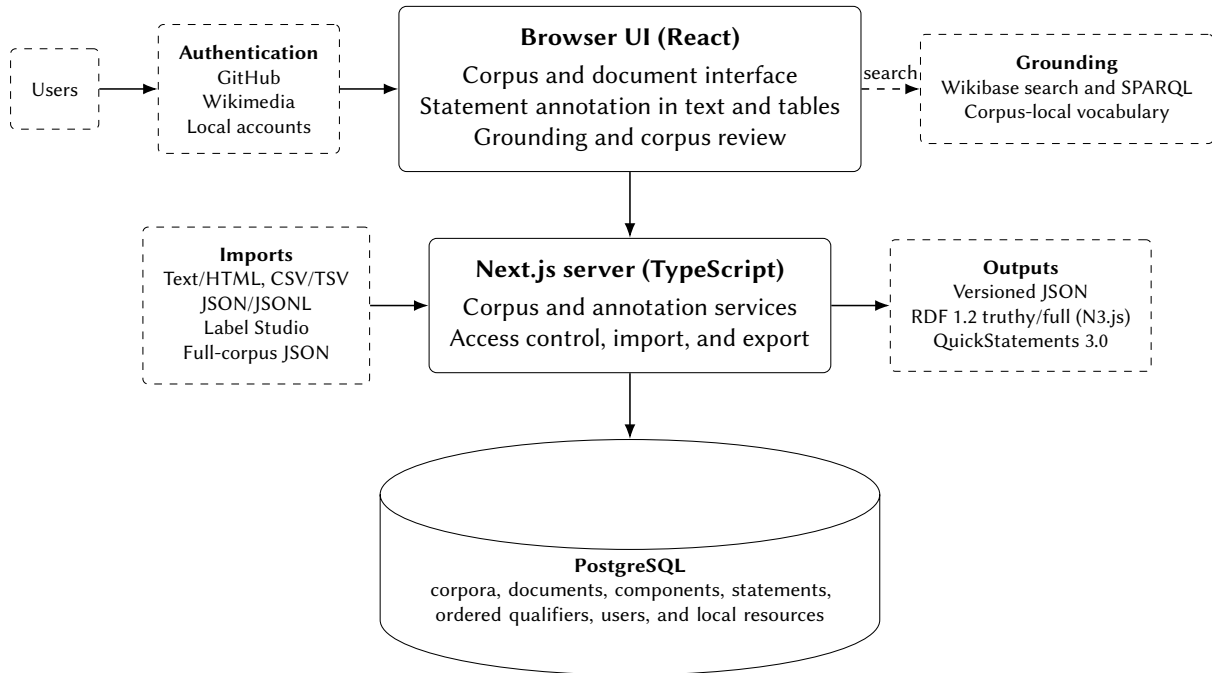
### 3. System Architecture

STAR-Q is a full-stack TypeScript application built with Next.js and React. PostgreSQL stores corpora, source documents, annotation components, statements, qualifiers, users, teams, and completion state. The Drizzle ORM manages the normalized relational schema and migrations. Persistent objects are identified by UUIDs. PostgreSQL’s fuzzy-string extensions are used for ranked lookup in the local vocabulary. N3.js serializes JSON exports into RDF 1.2 Turtle. The application is distributed as a standalone Node.js container and deployed with Docker Compose as three services: the non-root

<sup>2</sup><https://prodigy.ai>

web application, PostgreSQL with a persistent volume, and a one-shot database migration service. Authentication is required to annotate or manage corpora and uses Auth.js with GitHub, Wikimedia, and local username/password identity providers, each configurable during deployment. A setup flow creates the initial administrator at first launch. System administrators manage users, teams, API keys, and Wikibase instances from an admin dashboard. API keys created by administrators, supplied in the `x-api-key` header, grant read-only access to the corpus endpoints. Figure 1 summarizes the application boundaries and data flows.

**Figure 1:** Overall architecture of STAR-Q.



### 3.1. Corpus and Document Management

Annotation work in STAR-Q is organized around corpora identified by a title and creation/update timestamps. Each corpus is owned by a team (a user's personal team by default) and is private by default. Public corpora are readable without an account, and viewer and editor roles can be granted per corpus, either to individual users or to entire teams. Users can create, rename, duplicate, and delete corpora. Each corpus groups explicitly ordered documents managed through a dedicated document table that supports search, selection, deletion, and individual or bulk completion marking. Source metadata retains the original identifier, version date, hash, source URL, optional Wikidata identifier, and extraction method. Documents may contain free text, tables, or a mix of both. Import adapters normalize these inputs into a common document representation. Textual and tabular elements are interleaved according to their source offsets when available; otherwise, their stable import order is retained. Each resulting element receives a zero-based `elementIndex`. Text produced by a Wikitext extractor is converted to plain text before display and annotation.

Documents and existing annotations can be ingested from structured JSON/JSONL files, archive-based corpus packages, or Label Studio JSON outputs. The format is selected explicitly at import time, and the import layer validates required metadata and annotation fields, reporting fatal errors separately from recoverable warnings while normalizing all accepted inputs to the common representation. A versioned full-corpus JSON export can also be imported into an existing or newly created corpus. Grouping documents into corpora therefore defines both the scope of the reusable local vocabulary (Section 3.3) and the granularity of analytics (Section 3.4).

### 3.2. Triple-Oriented Annotation of Texts and Tables

The core of STAR-Q is its annotation interface. The user highlights a mention in the document and assigns it a role: subject, predicate, object, qualifier predicate, or qualifier value. The three main components form a statement, to which an ordered list of qualifier predicate–value pairs can be attached. Both texts and tables can contribute to the same statement: a subject may, for instance, be identified in a paragraph while its object value lies in a table cell, and a date used as a qualifier value may occur in another element.

Every component is encoded as a `DocumentAnnotationComponent`. In addition to its UUID and semantic role, it stores the selected surface form, source element, annotation modality, resolved label and value, optional corpus-local resource reference, and optional literal datatype. Text components use zero-based, end-exclusive character offsets relative to their containing text element. Table components combine a zero-based table element, row, and cell index with an end-exclusive character range inside the selected cell. Consequently, the system represents both whole-cell annotations and finer spans within a cell. The selected surface form remains independent of its resolved KG value, so grounding can be changed without altering the evidence target.

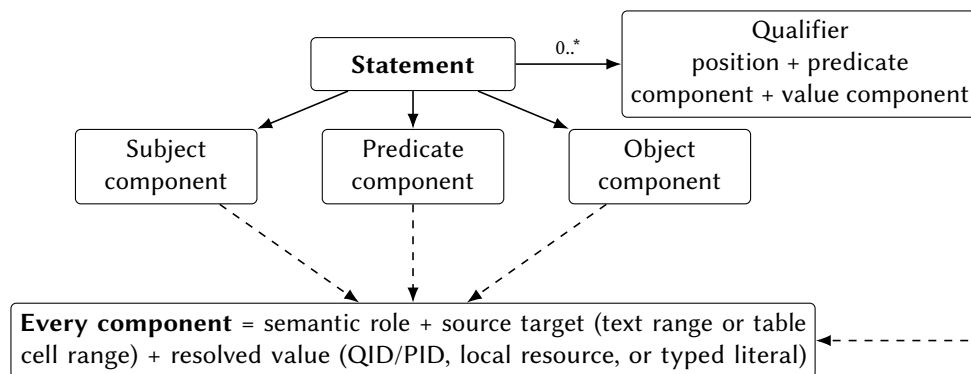
A statement row references three required components. Each qualifier is stored separately and references two further components: one with role `qualifier-predicate` and one with role `qualifier-value`, both with a zero-based position. This representation preserves qualifier order and allows the two sides of a qualifier to originate from different text or table elements. Empty or half-complete qualifier pairs cannot be saved.

The annotation workflow consists of selecting a span, assigning its role, resolving it through the target KG, the corpus vocabulary, or a literal datatype, and assembling the components into a statement. Whitespace around selections is removed while the stored boundaries are adjusted accordingly. Existing highlights may overlap or be shared by multiple statements. When several statements use the same source segment, a selection popover lets the annotator choose which one to inspect. Statements can be edited, cloned together with their qualifiers, filtered by modality or grounding status, sorted by source position, label, or creation order, and deleted individually or in batches. Keyboard shortcuts cover the frequent role-assignment, edit, clone, save, and delete actions.

When one component varies along a column or across a set of cells, batch table annotation avoids repeating the same selections. The annotator selects cells in a column, a row, or individually, then labels the selection as subject, predicate, or object.

Figure 2 summarizes the conceptual model. Every statement component combines three concerns: its semantic role, its source target, and its resolved value. A source target is either a character range within a text element, or a row and cell coordinate with a within-cell range in a table element. A resolved value is a target-KG identifier, a corpus-local resource, or a typed literal. The same component model is used for qualifier predicates and values, so their evidence is preserved independently of the three main components.

**Figure 2:** Conceptual model of STAR-Q.



3.3. Entity Grounding and Custom Entities

Once a mention is highlighted, STAR-Q initializes a lookup from its surface form and returns ranked labels, identifiers, and descriptions. Role-aware search requests items for subjects, objects, and qualifier values, and properties for statement and qualifier predicates. Lookup targets a Wikibase instance. Wikidata is the default, and its English search service returns QIDs or PIDs. Administrators can register additional instances by providing their URL, SPARQL endpoint, and concept base URI, and each corpus then selects the instance it targets. This is how private Wikibase deployments, such as enterprise knowledge graphs, replace or supplement the default Wikidata public endpoint. Remote lookup and corpus-local search execute in parallel, so the annotator sees target-KG and local candidates in one interaction. Objects and qualifier values may instead be typed literals or corpus-local resources, and literals can carry any RDF-compatible XSD datatype (e.g., string, integer, date, or gYear).

Real-world corpora inevitably mention entities or relations missing from the target KG. To avoid breaking the annotation flow, STAR-Q lets the user create corpus-local resources. A resource is defined once by a label, value, datatype, and type (entity or relation) and becomes immediately reusable in all documents of the corpus. Annotation components store the resource UUID rather than duplicating its label and value. Edits therefore propagate consistently to every use. Local search performs case-insensitive label/value matching and ranks candidates with Levenshtein distance. This normalized vocabulary avoids inconsistent spellings and constitutes a curated set of candidates that can be exported for later KG enrichment. The corresponding resources receive stable, corpus-scoped IRIs in RDF exports.

Figure 3: Example of a qualified annotated triple made with the STAR-Q annotation tool.

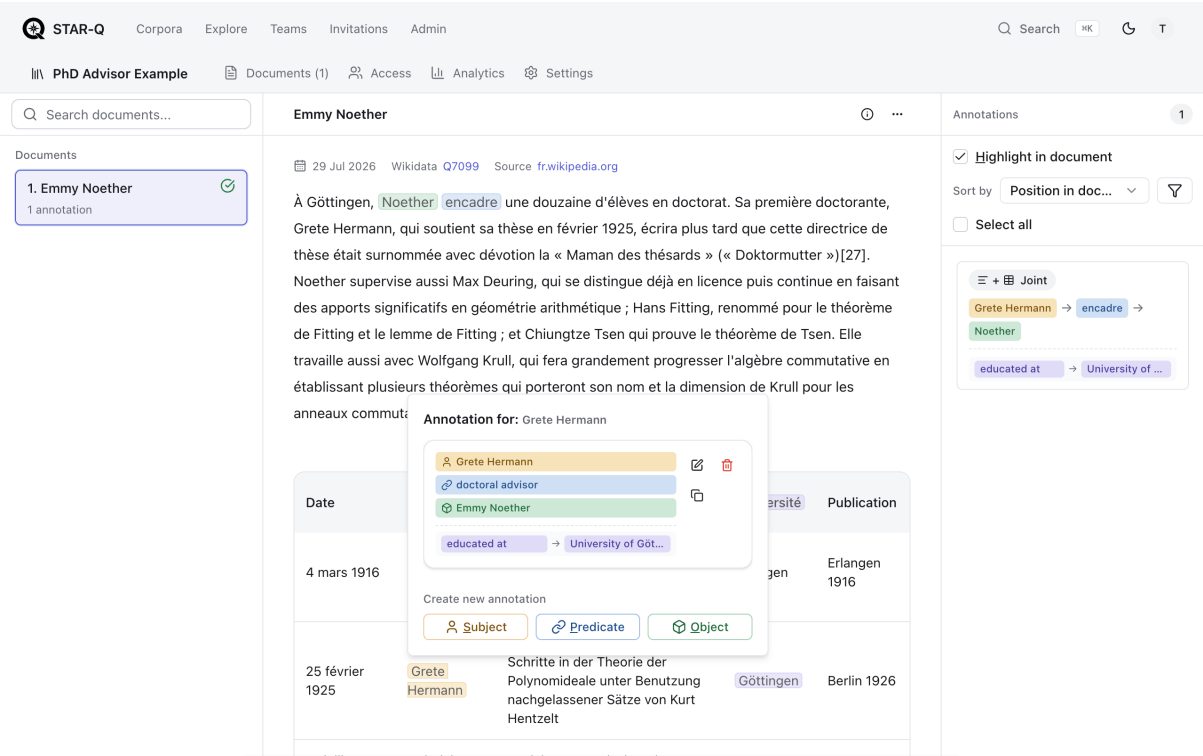


Figure 3 depicts an annotation of a textual passage and a table, both taken from the French Wikipedia page of “Emmy Noether” at [https://fr.wikipedia.org/wiki/Emmy\\_Noether](https://fr.wikipedia.org/wiki/Emmy_Noether). The truthy triples are:

```
wd:Q68550 wdt:P184 wd:Q7099 .
<< wd:Q68550 wdt:P184 wd:Q7099 >> pq:P69 wd:Q152838 .
```

The mentions supporting these triples are spans of text and table cell values.

### 3.4. Analytics and Consistency Checking

To help annotators and corpus managers monitor the annotation effort, STAR-Q provides a corpus-level analytics dashboard. It reports document and statement counts, completed-document counts, unique properties and entities, their occurrence counts, and lists of the documents in which each resource occurs. It also categorizes statements as text-only, table-only, or joint according to the provenance of their three main components. The joint category makes the use of evidence across modalities directly measurable.

STAR-Q requires valid textual or tabular mentions for all three main components before a statement can be saved. Grounding may nevertheless remain unresolved. The dashboard therefore identifies statements whose subject, predicate, or object has no Wikidata identifier or corpus-local resource and links these cases back to their documents. Ungrounded objects are reported for review but do not reduce the score because objects may legitimately be literals. A heuristic completion score combines explicitly completed documents (60%), grounded predicates (20%), and grounded subjects (20%). The score is a workflow indicator rather than a semantic correctness measure.

The dashboard additionally reports coherence warnings grounded in the Wikidata constraint model. For each statement whose predicate is a Wikidata property, STAR-Q retrieves the property’s subject-type and value-type constraints (P2302) and verifies, through the SPARQL endpoint of the targeted instance, that the annotated subject and object belong to the expected classes. Violations are listed with the document and property involved, and each warning links back to the affected annotation. Items without type information are reported as unverifiable rather than violated, and qualifier values are checked against the range constraints of their qualifier property. These checks are optional per corpus, non-blocking, and tolerant of missing constraint definitions, and the lookup can downrank candidates that do not match the expected domain or range.

### 3.5. Export and Programmatic Access

A corpus can be exported at any time in three complementary formats. The versioned JSON format, currently version 1.4, is a self-contained, re-importable representation. Its `exportMeta` object identifies the format type and version. It contains corpus metadata; explicitly ordered documents with their original normalized source data and completion timestamps; statement and component UUIDs; ordered qualifier predicate–value pairs; and corpus-local resources. On import, references to documents, components, qualifiers, and local resources are remapped to the destination corpus.

The RDF 1.2 Turtle export is serialized with N3.js and provides two modes.

**Truthy mode** emits grounded statements directly. Wikidata items use `wd:`, main properties use `wdt:`, and qualifier properties use `pq:` as namespaces. These namespaces are derived from the concept base URI of the targeted instance, so a custom Wikibase receives its own namespaces. Existing absolute HTTP(S) or URN identifiers are preserved, while corpus-local entities and relations receive stable, corpus-scoped IRI under a deployment-configurable base IRI. When qualifiers are present, a blank-node reifier identifies the statement’s RDF triple term through `rdf:reifies` and carries its qualifier triples. Hence, the example in Figure 3 can also be serialized as:

```
wd:Q68550 wdt:P184 wd:Q7099 .
_:statement
  rdf:reifies <<(wd:Q68550 wdt:P184 wd:Q7099)>> ;
  pq:P69 wd:Q152838 .
```

**Full mode** makes use of additional well-known vocabularies to represent provenance and the source evidence for each statement. More precisely, the corpus is a `dcat:Dataset` [18], while documents are `foaf:Document` resources that contain parts. Textual content becomes a NIF context [19, 20] from which offsets can be determined. Tables, columns, and cells are represented with CSVW [21]. Header-row targets are represented as CSVW columns while other table targets are represented as cells located through row and column indices. Each exported component is a Web Annotation whose target is the corresponding source range, column, or cell and whose body is its disambiguated value [22]. Components remain distinct when they share the same target. A stable statement IRI reifies the main

triple and links it to its three component annotations using PROV-O [23]. Qualifier triples are attached to that statement IRI and blank-node reifiers associate them with their own predicate and value evidence.

Literal values are serialized according to their declared XSD datatype, e.g., integers and decimals as `xsd:integer` and `xsd:decimal`, partial dates as `xsd:gYear`, `xsd:gMonth`, and `xsd:gDay`, and URLs as `xsd:anyURI`. Values such as months and days are normalized to valid XSD lexical forms.

A third export emits QuickStatements 3.0 commands, one TSV line per statement with optional qualifier columns, for statements grounded in Wikidata or Wikibase identifiers. Corpus-local resources, which lack such identifiers, are skipped and reported in the export response. These commands can be submitted to Wikidata or a Wikibase instance to enrich the target KG with the expert-validated statements.

Read-only HTTP endpoints expose the corpus list (`GET /api/corpus`) and, for a given corpus, its metadata, analytics, local vocabulary, and exports. JSON is the default representation. RDF is selected with `format=rdf&mode=truthy`, `format=rdf&mode=full`, or the `Accept: text/turtle` header. These endpoints may be protected by an authenticated session or an administrator-created API key.

## 4. Demonstration

The STAR-Q annotation tool has already been used to generate the ReTaT corpus [4] as part of the TRIPLET 2026 challenge.<sup>3</sup> In this section, we further demonstrate STAR-Q through two complementary scenarios, one didactic and one industrial, illustrating the complete workflow from corpus import to annotation, quality review, and export.

### 4.1. A Didactic Scenario: Wookieepedia and the Wikidata WikiProject Star Wars

The first scenario is deliberately playful and self-contained, so that readers can grasp the annotation workflow on universally recognizable content. We build a corpus of encyclopedic texts from Wookieepedia<sup>4</sup>, the Star Wars fan wiki, covering characters, starships, and planets. These documents mix descriptive paragraphs with infobox-style tables, making them an ideal showcase for mixed text-and-table annotation. Mentions are grounded against Wikidata entities curated by the WikiProject Star Wars<sup>5</sup>, a community effort to model the Star Wars universe in Wikidata. A user can, for instance, annotate a sentence stating that a planet is located in a given star system, linking the subject and object to their Wikidata QIDs and the predicate to the appropriate PID, then annotate a numeric characteristic of the planet (e.g. its diameter or rotation period) from an infobox-style table, assigning it the integer datatype. Fictional entities that are not yet covered by the WikiProject provide a natural occasion to demonstrate custom entity creation and reuse across the corpus, and the analytics dashboard summarizes which properties and entities dominate the resulting annotations.

### 4.2. An Industrial In-Use Scenario: Annotating Telco Documents with an Enterprise Knowledge Graph

The second scenario reflects the industrial motivation behind STAR-Q. At a major telecommunications operator, an enterprise knowledge graph structures telecommunications knowledge extracted from technical documentation. Building and evaluating the extraction systems that populate such an enterprise KG requires reference annotations produced by domain experts on real technical documents. In this scenario, STAR-Q targets the enterprise KG, a private Wikibase instance, instead of the public Wikidata endpoint. Domain experts annotate technical telecommunications texts, grounding mentions of network equipment, technologies, and their relations in the entities and properties of the enterprise graph, and creating custom entities for domain concepts not yet present in it. The consistency checker and the analytics dashboard are used to steer the annotation campaign, and the JSON export feeds both

---

<sup>3</sup><https://ecladatta.github.io/triplet2026/>

<sup>4</sup>[https://starwars.fandom.com/wiki/Main\\_Page](https://starwars.fandom.com/wiki/Main_Page)

<sup>5</sup>[https://www.wikidata.org/wiki/Wikidata:WikiProject\\_Star\\_Wars](https://www.wikidata.org/wiki/Wikidata:WikiProject_Star_Wars)

the evaluation of automatic extractors, while the QuickStatements export feeds the enrichment of the graph with expert-validated statements. This in-use case demonstrates that the same platform supports both open, community-driven annotation against Wikidata and closed, enterprise-grade annotation against a private KG.

## 5. Conclusion and Future Work

High-quality, entity-grounded annotated reference corpora are a prerequisite for building, evaluating, and comparing knowledge graph population systems, yet producing them remains costly and poorly supported by existing tools, especially when documents mix texts and tables. STAR-Q was designed with this gap in mind. It offers a unified web platform in which annotators delimit subject, predicate, object, and ordered qualifier mentions in both textual and tabular content; ground them through configurable Wikidata, Wikibase, or enterprise KG providers; and manage corpus-local entities and relations that can later enrich the target KG. The component model retains surface evidence independently of resolved values and supports statements whose evidence crosses modality boundaries. Corpus management, analytics, consistency checking, and re-importable versioned JSON support the full lifecycle of an annotation campaign. Truthy and full RDF 1.2 export modes make the resulting statements available either directly or with component-level source evidence and provenance.

While the tool is mature, we observe a number of limitations. First, unit handling is currently implicit and error-prone, especially for numeric values found in tables. We plan to add explicit unit support at the object level (e.g. a Wikidata unit entity or a corpus-level custom unit), with a clear UI for selecting and assigning the unit and for exporting it. This support should integrate cleanly with existing annotations and remain optional. Second, constraint checking currently covers the subject-type and value-type constraints of Wikidata properties. We plan to extend validation to a broader part of the Wikidata constraint model rather than limiting it to domain and range. Finally, we plan to run a user evaluation involving the volunteering wikipedians in order to assess the usability of STAR-Q including the annotation speed, the resulting annotation quality, and broader feedback from annotators.

Our demonstrations, spanning a didactic Star Wars scenario grounded in the Wikidata WikiProject Star Wars and an industrial scenario grounded in the enterprise KG of a major telecommunications operator, illustrate how a single annotation workflow can serve both open community knowledge graphs and private enterprise ones. This tool can be used to detect inconsistencies across text, table and knowledge graphs as evidenced in [24]. The tool is open source at <https://github.com/ecladatta/star-q> while a public instance is deployed at <https://starq.ecladatta.eurecom.fr/>.

## Acknowledgments

This work was supported by the French National Research Agency (Agence Nationale de la Recherche - ANR) under the ECLADATTA project (ExtraCtion of LATent knowledge in Documents by conjointly Analyzing Texts and TAbles, <https://ecladatta.github.io/>), grant number ANR-22-CE23-0020 (2023–2026).

## Declaration on Generative AI

During the preparation of this work, the authors used GLM-5.3-Flash in order to: Grammar and spelling check, Paraphrase and reword. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication’s content.

## References

- [1] C. M. Songuon, Y. Chabot, L. Jarnac, V.-P. Huynh, A guided and flexible LLM-based approach for knowledge extraction from text, in: 13th Knowledge Capture Conference (K-CAP), ACM, Dayton, OH, USA, 2025. doi:10.1145/3731443.3771373.
- [2] J. L. Martinez-Rodriguez, A. Hogan, I. Lopez-Arevalo, Information extraction meets the semantic web: A survey, *Semantic Web* 11 (2020) 255–335. doi:10.3233/SW-180333.
- [3] Y. Rebboud, P. Lisena, R. Troncy, CausalSense: Leveraging Common Sense Knowledge and LLMs for Joint Event Extraction and Relation Classification, in: 15th Language Resources and Evaluation Conference (LREC), 2026, pp. 7619–7630. doi:10.63317/4uiusu5nfowx.
- [4] M. Ettaleb, T. Ehrhart, N. Aussenac-Gilles, Y. Chabot, M. Kamel, V. Moriceau, R. Troncy, F. Wei, ReTaT: A Unified Benchmark for Relation Extraction across Text and Table, in: 15th Language Resources and Evaluation Conference (LREC), 2026, pp. 1341–1351. doi:10.63317/4k8jqsbkjuir.
- [5] L. Jarnac, Y. Chabot, M. Couceiro, Uncertainty management in the construction of knowledge graphs: A survey, *Transactions on Graph Data and Knowledge* 3 (2025) 1–48. doi:10.4230/TGDK.3.1.3.
- [6] D. Vrandečić, M. Krötzsch, Wikidata: A free collaborative knowledgebase, *Communications of the ACM* 57 (2014) 78–85. doi:10.1145/2629489.
- [7] E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas, SemTab 2019: Resources to benchmark tabular data to knowledge graph matching systems, in: *The Semantic Web – ESWC 2020*, Springer, 2020, pp. 514–530. doi:10.1007/978-3-030-49461-2\_30.
- [8] Y. Chabot, T. Labbé, J. Liu, R. Troncy, DAGOBAB: An end-to-end context-free tabular data semantic annotation system, in: *Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab@ISWC)*, volume 2553 of *CEUR Workshop Proceedings*, 2019, pp. 41–48.
- [9] P. Stenetorp, S. Pyysalo, G. Topić, T. Ohta, S. Ananiadou, J. Tsujii, brat: a web-based tool for NLP-assisted text annotation, in: 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Association for Computational Linguistics, 2012, pp. 102–107.
- [10] S. M. Yimam, I. Gurevych, R. Eckart de Castilho, C. Biemann, WebAnno: A flexible, web-based and visually supported system for distributed annotations, in: 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Association for Computational Linguistics, 2013, pp. 1–6.
- [11] J.-C. Klie, M. Bugert, B. Boullosa, R. Eckart de Castilho, I. Gurevych, The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation, in: 27th International Conference on Computational Linguistics: System Demonstrations, Association for Computational Linguistics, 2018, pp. 5–9.
- [12] H. Nakayama, T. Kubo, J. Kamura, Y. Taniguchi, X. Liang, doccano: Text annotation tool for human, <https://github.com/doccano/doccano>, 2018.
- [13] J. Liu, Y. Chabot, R. Troncy, V.-P. Huynh, T. Labbé, P. Monnin, From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods, *Journal of Web Semantics* 76 (2023) 100761.
- [14] M. Cremaschi, R. Avogadro, D. Chierigato, MantisTable: An automatic approach for the semantic table interpretation, in: *Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab@ISWC)*, volume 2553 of *CEUR Workshop Proceedings*, 2019, pp. 15–24.
- [15] C. Sarthou-Camy, G. Jourdain, Y. Chabot, P. Monnin, F. Deuzé, V.-P. Huynh, J. Liu, T. Labbé, R. Troncy, DAGOBAB UI: A New Hope for Semantic Table Interpretation, in: *The Semantic Web: ESWC 2022 Satellite Events*, Springer International Publishing, 2022, pp. 107–111.
- [16] H. Elsahar, P. Vougiouklis, A. Remaci, C. Gravier, J. Hare, F. Laforest, E. Simperl, T-REx: A large scale alignment of natural language with knowledge base triples, in: 11th International Conference on Language Resources and Evaluation (LREC), European Language Resources Association (ELRA), 2018.
- [17] S. Hadjab, A. Py, C. M. Songuon, Y. Chabot, A Human-in-the-Loop Text-to-Triples Pipeline for Multi-Extractor Knowledge Graph Construction and Wikidata Alignment, in: 6th Wikidata

Workshop, CEUR Workshop Proceedings, 2026.

- [18] R. Albertoni, D. Browning, S. Cox, A. Gonzalez Beltran, A. Perego, P. Winstanley, Data Catalog Vocabulary (DCAT) – Version 3, W3C Recommendation, World Wide Web Consortium, 2024. URL: <https://www.w3.org/TR/vocab-dcat-3/>.
- [19] S. Hellmann, J. Lehmann, S. Auer, M. Brümmer, Integrating NLP using linked data, in: International Semantic Web Conference (ISWC), Springer, 2013, pp. 98–113. doi:10.1007/978-3-642-41338-4\_7.
- [20] E. Wilde, URI Fragment Identifiers for the text/plain Media Type, RFC 5147, Internet Engineering Task Force, 2008. doi:10.17487/RFC5147.
- [21] J. Tennison, G. Kellogg, I. Herman, Model for Tabular Data and Metadata on the Web, W3C Recommendation, World Wide Web Consortium, 2015. URL: <https://www.w3.org/TR/tabular-data-model/>.
- [22] R. Sanderson, P. Ciccarese, B. Young, Web Annotation Vocabulary, W3C Recommendation, World Wide Web Consortium, 2017. URL: <https://www.w3.org/TR/annotation-vocab/>.
- [23] T. Lebo, S. Sahoo, D. McGuinness, PROV-O: The PROV Ontology, W3C Recommendation, World Wide Web Consortium, 2013. URL: <https://www.w3.org/TR/prov-o/>.
- [24] F. Wei, T. Ehrhart, R. Troncy, Detecting Knowledge Inconsistencies Across Text, Tables, and Knowledge Graphs, in: 25th International Conference on Knowledge Engineering and Knowledge Management (EKAW), 2026.