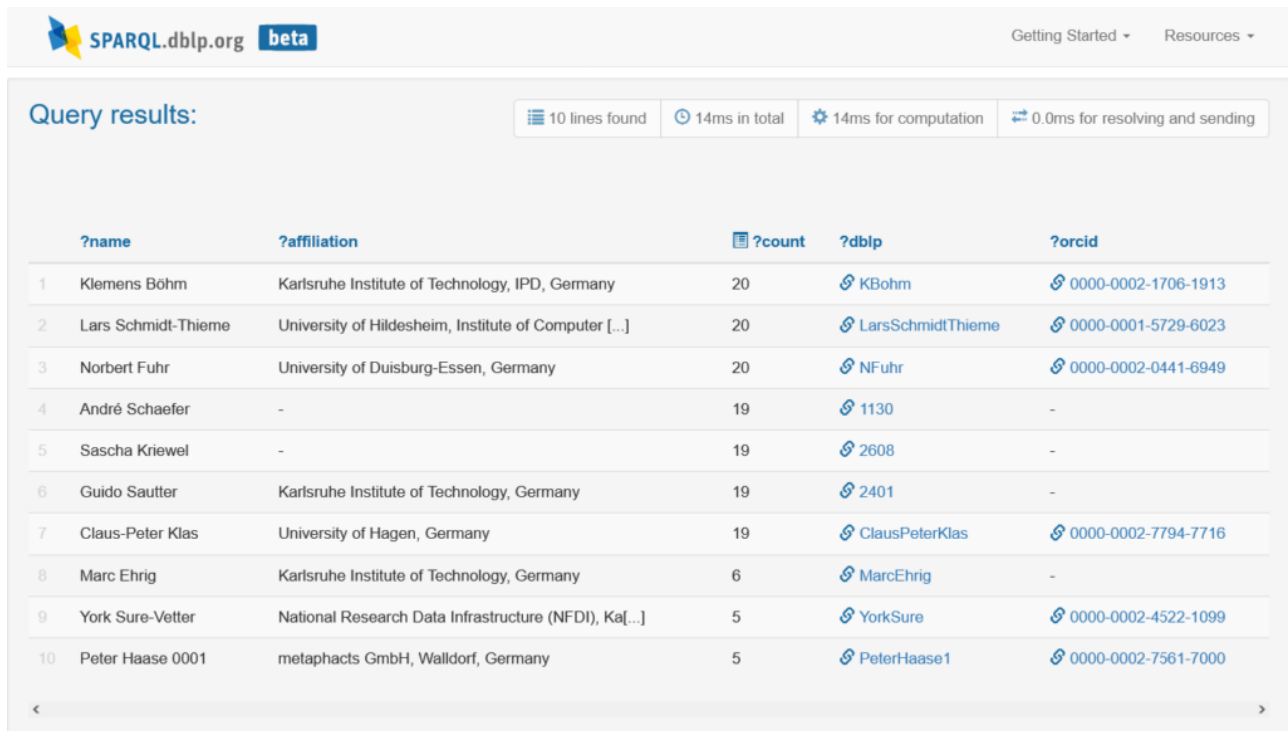


The **dblp Knowledge Graph (dblp KG)** is a fully semantic view on all the data and relationships that you can find in the [dblp computer science bibliography](#). In the recent years, the dblp team has been actively working on building the dblp KG, as we already discussed in [several recent blog posts](#). It has already proven to be quite useful, as the dblp KG makes sharing dblp's curated data and combining it with other semantic data sources easy and straightforward. But it also enables us to launch a new tool that will allow you to generate new insights well beyond the current capabilities of our prepared web pages and our simple text-based search: our brand new **dblp SPARQL query service**.



The screenshot shows the SPARQL.dblp.org beta interface. At the top, there's a header with the logo and navigation links. Below the header, the query results are displayed. The results are shown in a table with 5 columns: ?name, ?affiliation, ?count, ?dblp, and ?orcid. The table lists 10 results, each with a rank number, name, affiliation, count, and links to the dblp profile and ORCID iD.

	?name	?affiliation	?count	?dblp	?orcid
1	Klemens Böhm	Karlsruhe Institute of Technology, IPD, Germany	20	<a href="#">KBohm</a>	<a href="#">0000-0002-1706-1913</a>
2	Lars Schmidt-Thieme	University of Hildesheim, Institute of Computer [...]	20	<a href="#">LarsSchmidtThieme</a>	<a href="#">0000-0001-5729-6023</a>
3	Norbert Fuhr	University of Duisburg-Essen, Germany	20	<a href="#">NFuhr</a>	<a href="#">0000-0002-0441-6949</a>
4	André Schaefer	-	19	<a href="#">1130</a>	-
5	Sascha Kriewel	-	19	<a href="#">2608</a>	-
6	Guido Sautter	Karlsruhe Institute of Technology, Germany	19	<a href="#">2401</a>	-
7	Claus-Peter Klas	University of Hagen, Germany	19	<a href="#">ClausPeterKlas</a>	<a href="#">0000-0002-7794-7716</a>
8	Marc Ehrig	Karlsruhe Institute of Technology, Germany	6	<a href="#">MarcEhrig</a>	-
9	York Sure-Vetter	National Research Data Infrastructure (NFDI), Ka[...]	5	<a href="#">YorkSure</a>	<a href="#">0000-0002-4522-1099</a>
10	Peter Haase 0001	metaphacts GmbH, Walldorf, Germany	5	<a href="#">PeterHaase1</a>	<a href="#">0000-0002-7561-7000</a>

Example query result: non-coauthors of Michael Ley with many coauthors in common.

The dblp SPARQL query service allows you to explore the entire **semantic content of dblp** (enhanced with open citation data) in a convenient user interface. In contrast to previous approaches to semantic search on dblp, the data stock here is not merely based on a one-time snapshot of dblp data, but is **synchronized daily** with the current dblp data. Just like the regular dblp.org website, the service therefore benefits from the continuous curation and disambiguation work of the dblp team.

# Getting started

You can access our new SPARQL query service at <https://sparql.dblp.org>. Please note that we are currently still in the **public beta phase** of the service. So be prepared for disruptions and outages while we work on getting the new website ready for the task.

If you are an experienced SPARQL user, you may just want to have a look at the [dblp RDF schema](#) and jump straight into the fray. For beginners, however, we prepared a [guided tour of the dblp KG](#) in order show you some examples of what data you can expect to find and how you can run your first meaningful queries.

But we want to go a step further and enable even people with no experience (or no interest) in SPARQL at all to easily run queries. To this end, we have prepared a number of context-sensitive queries and embedded them directly in the user interface of the dblp.org website, such that they can be executed with just one click. These queries include statistics about an author's coauthors (and non-coauthors) or finding related conferences with a big community overlap. The embedded queries also serve as examples for the capabilities of the query service, as well as a starting point for writing your own queries.



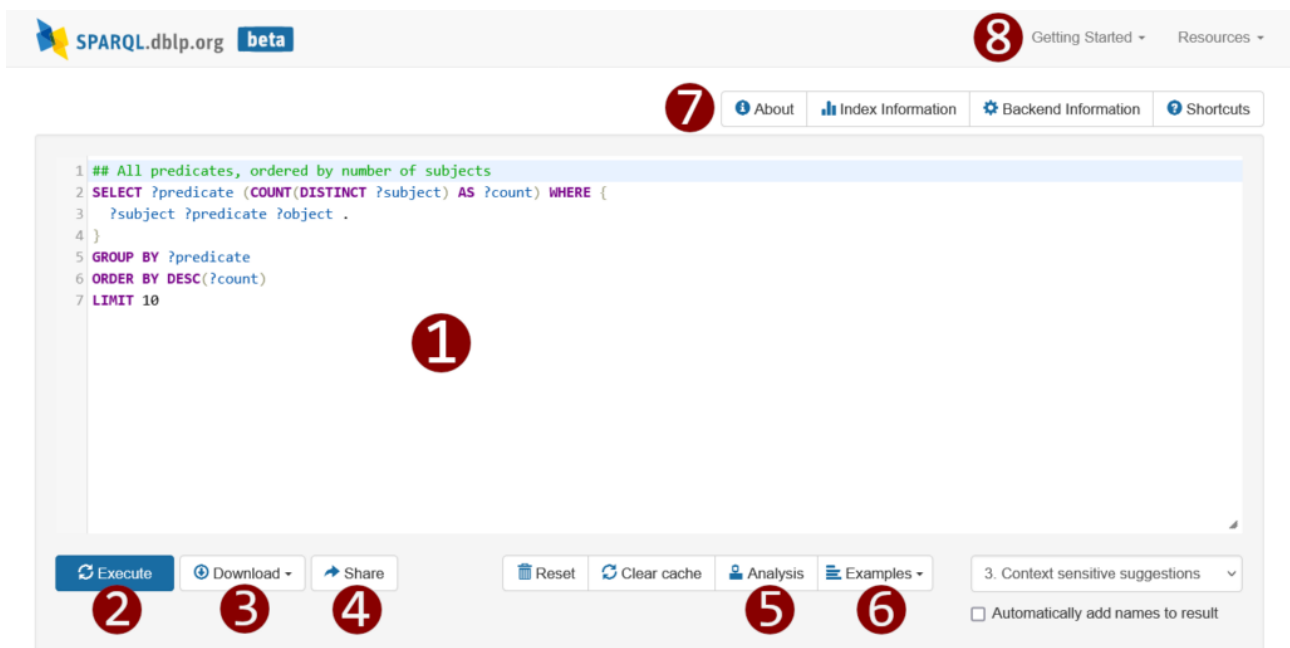

Embedded SPARQL queries on the dblp.org website.

If you happen to have any questions on using the dblp SPARQL query service, or if you want to get in touch with other users, we prepared a [GitHub Discussions forum](#) for our dblp community. The dblp team will keep an eye on the forum and try to help with this and that problem, although our time and resources are limited. But we hope that thanks to the forum, users can help each other solving any particular problems and have the opportunity to discuss their experiences with the query service.

## The QLever engine and UI

The dblp SPARQL query service is powered by [the QLever SPARQL engine](#), developed by [Hannah Bast](#), [Johannes Kalmbach](#), and others at the University of Freiburg. Hannah is a long-time supporter of dblp (e.g. by providing the CompleteSearch engine that powers dblp.org), and Hannah and Johannes' contribution was crucial to the creation of our query service.

The QLever engine is free and open-source software, provided under Apache License 2.0. The engine comes with its convenient **QLever UI** that makes writing SPARQL queries easy:



The QLever UI.

1. The interactive editor supports syntax-highlighting, context-sensitive auto-completion, and a number of keyboard shortcuts to allow for fast writing of queries.
2. Clicking on “Execute” (or hitting ctrl + enter) sends the query to the backend server. QLever achieves a high execution speed on the dblp KG and often calculates results within fractions of a second, with complex queries taking a maximum of only a few seconds.
3. You can download result tables in several formats.
4. Sharing of queries is supported via persistent short URL handles.
5. The QLever UI provides a graphical analysis of the query execution, listing the size requirements and execution times of individual steps. This helps you to identify bottlenecks in complex queries and to improve query performance.
6. You can select and run some illustrative examples directly from the UI.
7. Additional technical information and statistics about the index can be found right above the editor window.
8. The top-right menu provides links to further useful resources, and even links advise and data about how to set up your own instance of a dblp SPARQL query service.

The QLever Engine also provides a **SPARQL endpoint API** at <https://sparql.dblp.org/sparql>, following the SPARQL standard. QLever supports the media types *application/csv* (CSV), *application/tab-separated-values* (TSV), *application/sparql-results+json* (JSON), as well as its own custom JSON format of MIME type *application/glever-results+json*. For example, the following *curl* call will print ten arbitrary statements as TSV in your command line.

```
curl -s https://sparql.dblp.org/sparql -H "Accept: text/tab-separated-values" -  
H "Content-type: application/sparql-query" --data "SELECT * WHERE { ?subject  
?predicate ?object } LIMIT 10"
```

As with all the API services dblp provides, we ask you to please not overwhelm our live APIs with an excessive number of request. To ensure that our service is available to everyone, there is some rate limiting in place to protect us from aggressive scripting. If

you know that you will need a higher rate of queries, we encourage you to [set up your own instance of a dblp SPARQL query service](#) and to do all of your request locally. It really is surprisingly easy.

## Open citation references

The probably biggest benefit of open knowledge graphs is that multiple data sources can be easily combined. We demonstrate this in our SPARQL query service by not only building it upon the bibliographic data from the dblp KG, but also combining it with the citation data from [OpenCitations](#). OpenCitations is an awesome infrastructure that collects and aggregates citation metadata from openly available sources like Crossref or Datacite and provides the citation links under CC0 license. We have already [integrated OpenCitations data in the dblp website](#) in the past. Now, all open citation links to and from dblp publications are also available in our query service.

OpenCitations uses the [CiTo Ontology](#) and persistent IDs to model citation links between two documents. The documents are identified by the [OpenCitations Meta ID \(OMID\)](#), which serves as an abstraction for all possible persistent identifiers (like, say, DOI and ISBN) of the same publication. Links between dblp publications and OMIDs are provided in our knowledge graph by means of the *dblp:omid* property whenever possible. Using these links, you can easily obtain citation statistics, for instance, finding the most cited publications indexed in dblp.

```
## Most cited publications in dblp
PREFIX dblp: <https://dblp.org/rdf/schema#>
PREFIX cito: <http://purl.org/spar/cito/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX schema: <https://schema.org/>
SELECT ?url (?publ as ?dblp) ?title (COUNT(?citation) as ?cites) WHERE {
  ?citation cito:hasCitedEntity ?omid .
  ?publ dblp:omid ?omid .
  ?publ rdfs:label ?title .
  ?omid schema:url ?url
}
GROUP BY ?url ?publ ?title
ORDER BY DESC(?cites)
```

LIMIT 10

Please be aware that providing **open citation data** is only possible if the citation references are openly shared by the publishers. Furthermore, although it is no longer a technical requirement of OpenCitations, in practice, collecting citation data still crucially depends on DOIs. This means that open citation data in the absence of DOIs is still quite rare. At time of writing, we find that still only about 64% of all publications in dblp do have at least one reference associated (i.e., it is an object of the *cito:hasCitingEntity* predicate), and about 58% of all publications are themselves cited by another publication (i.e., are an object of *cito:hasCitedEntity*). If citation data about your publications is not openly available yet, then please consider asking your publisher to release your citation data to the public. For more information please see the [Initiative for Open Citations \(I4OC\)](#). Please also note that there is no way of submitting missing references or citation data directly to dblp.

## The bigger picture: querying beyond dblp

SPARQL is designed to make use of a **federation of SPARQL endpoints**. That is, a query can be distributed among multiple SPARQL servers on the Internet, having their partial results computed remotely, combined, and then its result listed in one common local result table. This allows for the ad-hoc combination of completely independent data sources, provided that the partial results can be joined by reusing globally shared unique identifiers. In the context of the dblp KG, such identifiers that are provided are — among others — DOI or OMID for publications, ORCID for people, or Wikidata Q identifiers for essentially any entity.

As an example, you can use the Wikidata Q identifiers provided for dblp authors to combine dblp KG information with biographical data available at Wikidata. For instance, the query below allows you get an impression about the gender distribution among the authors of a given journal. As dblp is not collecting any biographical information, so this

is a result that you could not have achieved using dblp data alone.

```
## Gender distribution among Semantic Web journal authors
PREFIX dblp: <https://dblp.org/rdf/schema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
SELECT ?gender ?label (COUNT(DISTINCT ?author) as ?freq) WHERE {
  ?publ dblp:publishedInStream <https://dblp.org/streams/journals/semweb> .
  ?publ dblp:authoredBy ?author .
  ?author dblp:wikidata ?wikiq .
  SERVICE <https://query.wikidata.org/sparql> {
    ?wikiq wdt:P2456 [] .
    ?wikiq wdt:P21 ?gender .
    ?gender rdfs:label ?label .
    FILTER (lang(?label) IN ("en"))
  }
}
GROUP BY ?label ?gender
ORDER BY DESC(?freq)
```

One severe limitation of federated queries is that they can be pretty slow. Depending on the computational strain of the remote server and the size of the remote query's result tables, the likelihood of slow responses or even timeouts is still rather high. Overall response times are usually dominated by the slowest individual query sent. Therefore, such practical limitations must be carefully considered when designing a federated query.

## Outlook and feedback

Our goal is to maintain and expand the dblp SPARQL query service as a cornerstone of the dblp computer science bibliography, just like our dblp website, our open APIs and our dump download releases. During the current beta phase, our priority is to test its stability and performance and to adapt the service to the needs of the community. At the same time, we are also actively working on expanding the content of the dblp KG by further utilizing more of the currently only weakly structured semantic information listed on the dblp website, like research institution entities, affiliation links of authors, as well as metadata about the date and location of conference events.

If you have any ideas or criticisms, or if you experience any trouble working with the query service, we will be grateful to hear about it. For any suggestions, questions or discussions about using the service, you are welcome to use our [GitHub Discussions forum](#). If you have technical problems or questions about the QLever engine, it might be best to directly contact the developers via the [QLever GitHub page](#). In case of general question about dblp, or if you find data errors or inconsistencies that need correction, you may of course always contact us via [dblp\(at\)dagstuhl.de](mailto:dblp(at)dagstuhl.de). We are looking forward to hearing from you!

## Acknowledgement

The dblp SPARQL query service is launching as a service of the [National Research Data Infrastructure \(NFDI\)](#), in particular of the consortia [NDFI4DataScience](#) and [NFDIxCS](#). Parts of creating the service has been funded by grants [GEPRIS 460234259](#) and [GEPRIS 501930651](#), as well as as part of the [DFG-LIS project “Unknown Data”](#) ([GEPRIS 460676019](#)).