System Requirements

We require Confluence 5.10 and Java 8.

Advanced search is only available when using PostgreSQL

Feature	All databases	PostgreSQL	MySQL
Create, edit, search requirements	✓	✓	✓
JIRA connector	✓	✓	✓
Reports	✓	✓	✓
Search by key, title, space or status	✓	✓	✓
Search by page, JIRA issue or property	?*	✓	✓

* We use Atlassian's official library for our storage needs, which name is ActiveObjects. They test thoroughly with all vendors, but we've noticed that we sometimes meet bugs with MS SQL and Oracle. Please report any error to us and we'll improve it.

Out of Memory Errors when a matrix is displayed

Building the coverage, dependency and traceability matrixes may use a lot of resources.

- Before 2.0, we had hard limits for the size of the coverage, dependency and traceability matrixes. We relied on the EXPORT permission of the space to allow users to access those matrixes.
- Since 2.0, See the Global limit for information about size limits.
- As a consequence of the global limit in 2.0, we'll soon **remove** the EXPORT permission to access those tabs, and we'll allow anyone with VIEW permission to access them. If that causes a problem, please notify us. This is mentioned in Release Notes 2.0.

See Performance if you are interested.

Compliance considerations

Some customers use Requirement Yogi for legal or compliance purposes. We don't assess whether Requirement Yogi is suitable or unsuitable for this purpose, but here is some interesting information:

- The coverage features, the dependency matrix, the link with JIRA are very useful tools in achieving traceability.
- Requirement Yogi creates a copy of all requirements when you create baselines, including their text. The text is frozen, it can't be updated again. Links to dependencies in the same baseline are saved.
- The baseline and the saved version of the requirements may be deleted if the page of the baseline is deleted.
- Confluence allows deleting a version in the page history. It means a dedicated person could delete the version of the requirements that was agreed upon and saved in the baseline, so that the text of the requirement in the baseline doesn't match the text in that version of the page.
- Confluence allows add-ons to change its data, notably the page history. A malevolent add-on could in theory modify an older version of a page to make it seam like a specific requirement was written.

We depend on the core behavior of Confluence and some features are intended for Agile-oriented companies with assumed positive intentions. If you regularly work with malevolent people, you may want to ask us which defensive customizations you could deploy (Notably preventing pages and versions from being deleted, configuring Applinks in a certain way for authentication purpose, setting up version-level signatures).

Maximum number of requirements

There is no hard-defined number of requirement, either by space or globally. However Requirement Yogi relies on a library provided by Atlassian for storage, named ActiveObjects, which has a limit of 2,147,483,647 creations per table.

The table which will contain the biggest number of inserts is certainly AO_32F7CE_AOPROPERTY, which contains the properties of requirements. Each time a page is saved, Requirement Yogi drops the requirements and their associated objects, and recreates the ones that still exist in the new version of the page. Since ActiveObjects doesn't reuse the IDs of deleted items, the IDs keep growing. Here is the condition which must always be satisfied:

$$n_{pages} \times n_{edits} \times n_{req} \times n_{prop} < L$$

n_{pages} Number of pages with requirements
n_{edits} Number of edits of the page
n_{req} Number of requirements on the page
n_{prop} Number of properties per requirement
The sum must be made for all pages with requirements.

L: The limit, 2,147,483,647 (~2 billion)

For example, this allows for 200 pages with 1000 requirements on each page, each having 10 properties, editing those pages 2000 times and still not reaching the limit.

How to solve it, if you reach the limit?

You would be able to resolve the issue by "compacting" the IDs. The operation consists of renumbering primary keys in tables from 0, leaving no gaps, then resetting the database sequences that provide the next available IDs.