jql cheat sheet atlassian

JQL Cheat Sheet Atlassian: Mastering Jira Query Language for Efficient Issue Tracking

jql cheat sheet atlassian is an invaluable resource for anyone looking to harness the full power of Jira's search capabilities. Whether you're a project manager, developer, or Jira administrator, understanding how to craft precise queries can dramatically improve your workflow and make issue tracking more efficient. Atlassian's Jira Query Language (JQL) is a flexible and powerful way to filter and find issues based on complex criteria, but it can be intimidating at first glance. This guide will walk you through key aspects of JQL, providing tips, examples, and insights to help you become more confident in mastering this essential tool.

What Is JQL and Why Use It?

Jira Query Language (JQL) is Atlassian's proprietary query language designed specifically for Jira software. Unlike simple keyword searches, JQL allows you to build detailed queries using fields, operators, keywords, and functions. This means you can search not only by status or assignee but also by custom fields, dates, project types, and more.

Using JQL effectively can save you time by pinpointing exactly the issues you need to see. For example, you can find all unresolved bugs assigned to your team last week or track high-priority tickets that have been updated recently. This level of specificity is crucial for managing complex projects with many moving parts.

Understanding the Basics of JQL

Before diving into complex queries, it's essential to grasp the building blocks of JQL. At its core, a JQL query consists of:

- **Fields**: These are the attributes of issues, such as `status`, `assignee`, `priority`, `project`, or custom fields.
- **Operators**: Symbols or keywords that define the relationship between fields and values, like `=`, `!=`, `IN`, `NOT IN`, `>`, `<`, `~` (contains), and `!~` (does not contain).
- **Values**: The specific data points you want to filter on, such as `Open`, `John Doe`, or `High`.
- **Keywords**: Logical connectors like `AND`, `OR`, and `NOT` that combine multiple conditions.

For instance, a simple JQL query might look like this:

`status = "In Progress" AND assignee = currentUser()`

This query returns all issues that are currently in progress and assigned to the logged-in user.

Common Fields and Their Uses

Knowing which fields are available and how to use them is foundational. Some common Jira fields include:

- **project**: Filters issues by project key or name.
- **status**: Shows issues in a particular workflow state.
- **assignee**: Finds issues assigned to a specific user.
- **reporter**: Displays issues reported by a given user.
- **priority**: Filters by priority level like High, Medium, or Low.
- **created** / **updated**: Search based on dates.
- **labels**: Allows filtering by tags assigned to issues.

Essential JQL Operators and Their Functionality

Operators define how your query interprets the relationship between fields and values. Here's a quick rundown of the most useful operators you'll encounter in the Atlassian JQL cheat sheet:

- **=** (Equals) – Selects issues where a field matches a value exactly.
- **!=** (Not equals) – Finds issues where a field does not match a certain value.
- **IN** – Looks for issues where a field is within a list of values.
- **NOT IN** – Excludes issues with fields matching any value in a list.
- **~** (Contains) – Performs a text search for a substring in fields like summary or description.
- **!~** (Does not contain) – Finds issues where the field does not contain a substring.
- **>**, **<**, **>=**, **<=** – Comparison operators useful for dates and numeric fields.

For example, to find issues assigned to either John or Jane, you might use:

`assignee IN ("John", "Jane")`

These operators, combined with fields, allow you to create highly customized searches.

Advanced JQL Features to Boost Your Queries

Once you've mastered the basics, you can take advantage of some advanced JQL features that Atlassian provides to make your queries even smarter.

Functions in JQL

JQL supports functions that dynamically calculate values or return specific sets of issues. Common functions include:

- `currentUser()` – Returns the user currently logged in.
- `now()` – Refers to the current date/time.
- `startOfDay()`, `endOfDay()`, `startOfWeek()`, `endOfWeek()` – Useful for date-based queries.
- `membersOf("groupname")` – Finds issues assigned to members of a particular group.
- `issuekey` – Allows referencing specific issues.

For instance, to find issues updated in the last 7 days, you might write:

`updated >= -7d`

Or, to locate all issues assigned to the current user that are unresolved:

`assignee = currentUser() AND resolution = Unresolved`

Using Logical Operators Effectively

The logical operators `AND`, `OR`, and `NOT` allow you to combine conditions for more nuanced queries. Remember that AND has higher precedence than OR, so parentheses can help clarify your intentions.

Here’s an example:

`(status = Open OR status = "In Progress") AND priority = High`

This query finds all high-priority issues that are either open or in progress.

Tips for Writing Efficient JQL Queries

Working with JQL can sometimes get tricky, especially when your projects grow in size and complexity. Here are some practical suggestions to optimize your JQL usage:

- **Start Simple:** Begin with basic queries and gradually add complexity to ensure accuracy.
- **Use Autocomplete:** Jira’s query editor offers autocomplete suggestions that reduce errors and speed up writing.
- **Leverage Saved Filters:** Save frequently used queries as filters for quick access and sharing with your team.
- **Test Queries:** Run your queries frequently to confirm they return the expected results.

- **Document Complex Queries:** Add comments or notes outside Jira to keep track of what intricate queries do.
- **Watch Out for Performance:** Avoid overly broad queries on large datasets, which can slow down Jira.

JQL Cheat Sheet Atlassian: A Handy Reference

Atlassian's official JQL cheat sheet is an excellent reference for anyone working with Jira. It summarizes key fields, operators, functions, and syntax rules in one place. Many teams keep a customized version of this cheat sheet handy, tailored to their specific Jira setup and workflows.

If you're looking for quick access during your work, consider bookmarking Atlassian's documentation or exporting cheat sheets to PDF. Additionally, various third-party plugins and browser extensions integrate JQL helpers to boost productivity.

Examples of Practical JQL Queries

To illustrate how versatile JQL can be, here are several example queries you might find useful:

1. **Find all unresolved bugs in a project:**

   `project = ABC AND issuetype = Bug AND resolution = Unresolved`
2. **Show issues assigned to you that are due this week:**

   `assignee = currentUser() AND due >= startOfWeek() AND due <= endOfWeek()`
3. **Search for issues with a specific label:**

   `labels IN (urgent, backend)`
4. **Identify recently updated high-priority tickets:**

   `priority = High AND updated >= -3d`
5. **Find issues reported by a group of users:**

   `reporter IN membersOf("qa-team")`

These queries demonstrate how combining fields, operators, and functions can help you tailor your issue searches precisely to your needs.

Integrating JQL into Jira Dashboards and Automation

Beyond manual searches, JQL plays a crucial role in Jira dashboards, boards, and automation rules. Filters created with JQL power gadgets on your dashboards, enabling real-time monitoring of issues matching specific criteria. Scrum and Kanban boards also use JQL to define which issues appear in each column or swimlane.

Moreover, automation rules often rely on JQL conditions to trigger actions, such as sending notifications or updating fields when certain criteria are met. Mastering the JQL cheat sheet Atlassian helps you configure these automations effectively, enhancing your team’s responsiveness and reducing manual effort.

Getting comfortable with JQL might seem daunting at first, but with practice and the right resources like the Atlassian JQL cheat sheet, you’ll soon find yourself crafting complex queries with ease. This mastery not only streamlines your issue tracking but also empowers your entire team to work smarter within Jira’s rich ecosystem.

Frequently Asked Questions

What is a JQL cheat sheet in Atlassian?

A JQL cheat sheet in Atlassian is a quick reference guide that summarizes common Jira Query Language (JQL) commands and syntax to help users efficiently search and filter issues in Jira.

Where can I find an official JQL cheat sheet for Atlassian Jira?

You can find the official JQL cheat sheet on Atlassian's documentation website, specifically in the Jira Software documentation under the JQL section.

What are some basic JQL operators included in a typical Atlassian JQL cheat sheet?

Basic JQL operators usually include '=', '!=', '~' (contains), '!~' (does not contain), 'IN', 'NOT IN', '>', '<', '>=' and '<=' to filter issues based on field values.

How can a JQL cheat sheet improve my productivity in Jira?

A JQL cheat sheet helps by providing quick syntax reminders and examples, enabling faster creation of complex queries to filter and find issues without needing to memorize all commands.

Does the Atlassian JQL cheat sheet cover advanced query

functions?

Yes, many Atlassian JQL cheat sheets include advanced functions like date comparisons, issue linking, subqueries, and aggregations to help users perform complex searches.

Can I customize my own JQL cheat sheet for Atlassian Jira?

Yes, users often create personalized JQL cheat sheets tailored to their project needs and frequently used queries, which can be maintained as documents or internal wiki pages.

Additional Resources

Mastering JQL: The Ultimate JQL Cheat Sheet Atlassian Users Need

jql cheat sheet atlassian serves as an indispensable resource for Jira users aiming to harness the full power of Jira Query Language (JQL) to streamline project management and issue tracking. As Atlassian's proprietary query language, JQL offers a flexible and precise way to filter, sort, and manipulate data within Jira, enabling users—from developers to project managers—to retrieve exactly the information they need from complex issue databases. This article delves into the essentials of the JQL cheat sheet Atlassian provides, unpacking its syntax, key functions, and practical applications that elevate Jira's usability.

Understanding the Importance of JQL in Jira

Jira's core strength lies in its ability to track issues, tasks, bugs, and workflows across diverse teams. However, without a robust querying mechanism, users might find themselves overwhelmed by the sheer volume of data. JQL addresses this by offering a structured yet intuitive language to precisely specify search criteria. The JQL cheat sheet Atlassian offers acts as a quick reference guide to help users craft effective queries without needing extensive database expertise.

Unlike basic search filters, JQL supports complex expressions, logical operators, and dynamic functions, making it a powerful tool for extracting actionable insights. Whether you want to identify overdue tasks, monitor sprint progress, or generate custom reports, mastering JQL can significantly enhance data accessibility and decision-making.

Core Components of the JQL Cheat Sheet Atlassian Provides

At its foundation, the JQL cheat sheet breaks down the language into several crucial components that users need to grasp:

- **Fields:** These are the attributes or columns within Jira issues, such as status, assignee, reporter, priority, creation date, and more.
- **Operators:** Symbols or keywords that define relationships or comparisons, including '=', '!=',

'>', '<', 'IN', 'NOT IN', 'CONTAINS (~)', and logical connectors like AND, OR, NOT.

- **Keywords:** Reserved words to build queries, such as ORDER BY, IS, WAS, CHANGED.
- **Functions:** Dynamic expressions that return values, for example, currentUser(), now(), startOfDay(), membersOf(), enabling real-time and user-specific query conditions.

This modular structure provides a clear framework for users to combine these elements into queries that can be as simple or as intricate as necessary.

Practical Applications and Examples from the JQL Cheat Sheet Atlassian

Having access to a JQL cheat sheet benefits users by offering ready-made query patterns and syntax reminders that reduce trial-and-error. Let’s explore some practical use cases that highlight the versatility and efficiency JQL brings to Jira workflows.

Filtering Issues by Status and Assignee

One of the most common scenarios is filtering issues assigned to a particular user and currently in a specific status:

```
assignee = currentUser() AND status = "In Progress"
```

This query swiftly identifies all tasks the logged-in user is working on, focusing on active work items. The cheat sheet underscores how combining fields and operators optimizes daily task management.

Sorting and Ordering Results

JQL allows results to be ordered based on fields:

```
project = "Website Redesign" ORDER BY priority DESC, created ASC
```

This query lists issues in the "Website Redesign" project, prioritizing high-priority issues first and then sorting older issues earlier. The cheat sheet’s explanations of ORDER BY syntax enable users to customize their views effectively.

Utilizing Date Functions to Track Recent Activity

Date-based queries are vital for sprint reviews or monitoring recent updates:

```
updated >= startOfWeek()
```

This retrieves issues updated since the beginning of the week. The cheat sheet Atlassian provides enumerates various date functions such as startOfDay(), endOfMonth(), and relative date modifiers, empowering teams to track changes over specific periods.

Advanced Queries Using Membership and Custom Fields

JQL's capacity extends to querying group memberships or custom fields, enhancing flexibility:

```
assignee IN membersOf("developers") AND cf[12345] = "High"
```

Here, cf[12345] represents a custom field ID for priority or severity. The cheat sheet's guidance on referencing custom fields and user groups simplifies advanced query construction.

Comparing JQL to Other Query Tools in Jira

While Jira also offers basic filters and dashboard gadgets, JQL stands out due to its precision and adaptability. Unlike static filters, JQL queries can be saved, shared, and integrated into Jira dashboards, reports, and automation rules. The cheat sheet Atlassian provides not only functions as a syntax guide but also as a bridge for users transitioning from point-and-click filters to more dynamic querying.

However, JQL's learning curve may be steeper for new users, especially those unfamiliar with query languages. The cheat sheet serves as an essential onboarding tool, providing examples and explanations that flatten this curve. Compared to SQL, JQL is more domain-specific and user-friendly for non-technical users, albeit less powerful in terms of relational database operations.

Pros and Cons of Using JQL

- **Pros:**
    - Highly flexible search capabilities tailored to Jira's data model.
    - Ability to combine multiple criteria and logical operators.
    - Supports dynamic functions for real-time data querying.

- Integrates seamlessly with Jira dashboards and automation.

- **Cons:**
  - Initial learning curve for non-technical users.
  - Limited to Jira's internal data structure—less suitable for cross-tool queries.
  - Complex queries can become difficult to read without proper formatting.

The Atlassian JQL cheat sheet mitigates many cons by providing a structured learning aid, examples, and quick syntax reminders.

Enhancing Productivity With JQL Cheat Sheet Atlassian

For teams seeking to optimize their Jira usage, the JQL cheat sheet is more than just a reference—it's a productivity enhancer. By enabling users to formulate precise queries, it reduces the time spent sifting through irrelevant issues and focuses attention on priority tasks. Project managers can generate targeted reports effortlessly, while developers can monitor their workloads with granularity.

Moreover, the cheat sheet supports continuous learning. As Atlassian updates Jira's capabilities, the cheat sheet evolves, reflecting new operators, functions, or best practices. This dynamic aspect ensures that teams remain aligned with the latest Jira features, maintaining efficiency and accuracy in issue tracking.

Incorporating the cheat sheet into training sessions, documentation, or internal wikis can foster a culture of JQL literacy, empowering all Jira users regardless of their technical background.

With the comprehensive guidance contained in the JQL cheat sheet Atlassian provides, users unlock the latent potential of Jira's querying system. This foundational tool transforms how teams interact with their data, making project tracking smarter, faster, and more tailored to individual and organizational needs.

Jql Cheat Sheet Atlassian

jql cheat sheet atlassian: *Architecting Automation: A DevOps Guide to Atlassian Tools and Cloud Infrastructure 2025* Author:1 MANOGNA SAMMETA, Author:2. PROF. SANDEEP KUMAR,

PREFACE In the rapidly evolving landscape of software delivery and IT operations, organizations face unprecedented pressure to deliver value faster, while maintaining stability, security, and scalability. The principles of DevOpscollaboration, automation, and continuous improvementhave emerged as the antidote to siloed teams, manual handoffs, and brittle release processes. Yet knowing DevOps principles and actually implementing them at scale across a diverse toolchain and cloud infrastructure are two quite different challenges. This book, Architecting Automation: A DevOps Guide to Atlassian Tools and Cloud Infrastructure, is designed for practitioners, engineers, and architects who are committed to transforming their delivery pipelines into robust, resilient, and fully automated systems. Whether you're an Atlassian administrator tasked with scaling Jira and Confluence across distributed teams, a DevOps engineer building self-service CI/CD pipelines in Bitbucket and Bamboo, or a cloud architect integrating cloud-native services with Opsgenie alerts and Status page communications, you'll find practical guidance, reference architectures, and real-world patterns to accelerate your automation journey. We begin by grounding ourselves in the core tenets of DevOps and how they manifest in the Atlassian ecosystem. You'll learn how to align processes around value streams, model workflows in Jira Software, and structure Confluence spaces as living documentation portals. From there, we dive into version control, branching strategies, and pull-request–driven development in Bitbucket, demonstrating how to enforce code quality gates, best practices, and compliance rules. The heart of this guide explores continuous integration and continuous delivery (CI/CD) pipelines. We'll build end-to-end pipelines using Atlassian's Bamboo and Bitbucket Pipelines, woven together with Infrastructure as Code in Terraform and cloud-native services on AWS, Azure, and Google Cloud Platform. You'll discover how to parameterize builds, manage credentials with Vault, automate release orchestration across environments, and implement blue/green and canary deployments that minimize risk. No DevOps transformation is complete without robust monitoring, incident management, and post-mortem processes. We'll cover how to configure Opsgenie for multi-channel alerting, integrate PagerDuty for on-call rotations, and publish dynamic status updates via Status page. You'll see how to embed observability into your architecturecapturing metrics, logs, and tracesand drive continuous feedback loops that feed back into your Jira backlog as actionable tasks. Security and governance often feel like speed bumps in a DevOps pipeline. This book shows you how to codify security policies as code, using tools like Checkov and Jira's Policy as Code plugins, and integrate automated compliance checks into your CI pipelines. We'll explore how to manage role-based access controls across Atlassian tools and cloud accounts, ensuring that your automation is secure by design. Throughout the chapters, you'll find concrete reference architectures, sample scripts, and configuration snippets that you can adapt to your own environment. Real-world case studies illustrate how organizationsfrom nimble startups to large enterpriseshave leveraged Atlassian's platform and cloud infrastructure to reduce lead times, improve release stability, and foster a culture of ownership and continuous improvement. This is more than a cookbook; it's a blueprint for architecting automation that scales with your organization's needs. By the end of this journey, you'll have the knowledge to design and operate an integrated DevOps ecosystemone where Atlassian tools, cloud services, and custom integrations work in concert to deliver software and infrastructure changes with the speed, quality, and reliability your customers expect. Welcome to the future of DevOps automation. Let's get started. Authors MANOGNA SAMMETA PROF. SANDEEP KUMAR

Related to jql cheat sheet atlassian

JQL cheat sheet - Atlassian JQL is a powerful search language specifically designed for Jira that allows users to create complex queries for issue tracking, advanced filtering, and reporting

Visor's JQL Cheat Sheet Essentially, JQL is what's called a query language: a specific way of writing asks to get the most flexible search possible from a database. Here, we go over the basics of how to use this

JQL Cheat Sheet: Master JQL Query Language & Syntax JQL is a query language used to filter issues in Jira. It helps users to create structured and complex searches to extract unique information

or data from the JIRA tool

Introduction to JIRA Query Language (JQL) - The Requirements Discover the power of JIRA Query Language (JQL) with practical examples, syntax tips, and visualization ideas

Use advanced search with Jira Query Language (JQL) | Jira Cloud You can use the Jira Query Language (JQL) to specify criteria that cannot be defined in the quick or basic searches. For example, you can use the ORDER BY clause in a JQL query to search

JQL operators - Jira Cloud | Atlassian Support An operator in JQL is one or more symbols or words, which compares the value of a field on its left with one or more values (or functions) on its right, such that only true results are retrieved

Unlock Advanced Search with Jira Query Language (JQL) - Atlassian Advanced search allows you to build structured queries using Jira Query Language (JQL) to search for work items within and across projects. Query results can be saved and used as

The Ultimate JQL Cheat Sheet: How To Get Started, from an Expert Essentially, JQL is what's called a query language: a specific way of writing asks to get the most flexible search possible from a database. It can be very powerful, but it also can

The Jira JQL Advanced Guide: How to Search Jira Issues Like a Pro JQL, or Jira Query Language, is a flexible tool that allows you to search for issues in Jira and pinpoint exactly what you are looking for. Knowing how to search your Jira instance

JQL Basics with Syntax - Atlassian Community Jira Query Language (or JQL) is one of the most powerful tools available in Jira. The system uses the following data to filter issues. And, in turn, helps you find what you're

JQL cheat sheet - Atlassian JQL is a powerful search language specifically designed for Jira that allows users to create complex queries for issue tracking, advanced filtering, and reporting

Visor's JQL Cheat Sheet Essentially, JQL is what's called a query language: a specific way of writing asks to get the most flexible search possible from a database. Here, we go over the basics of how to use this

JQL Cheat Sheet: Master JQL Query Language & Syntax JQL is a query language used to filter issues in Jira. It helps users to create structured and complex searches to extract unique information or data from the JIRA tool

Introduction to JIRA Query Language (JQL) - The Requirements Discover the power of JIRA Query Language (JQL) with practical examples, syntax tips, and visualization ideas

Use advanced search with Jira Query Language (JQL) | Jira Cloud You can use the Jira Query Language (JQL) to specify criteria that cannot be defined in the quick or basic searches. For example, you can use the ORDER BY clause in a JQL query to search

JQL operators - Jira Cloud | Atlassian Support An operator in JQL is one or more symbols or words, which compares the value of a field on its left with one or more values (or functions) on its right, such that only true results are retrieved

Unlock Advanced Search with Jira Query Language (JQL) - Atlassian Advanced search allows you to build structured queries using Jira Query Language (JQL) to search for work items within and across projects. Query results can be saved and used as

The Ultimate JQL Cheat Sheet: How To Get Started, from an Expert Essentially, JQL is what's called a query language: a specific way of writing asks to get the most flexible search possible from a database. It can be very powerful, but it also can

The Jira JQL Advanced Guide: How to Search Jira Issues Like a Pro JQL, or Jira Query Language, is a flexible tool that allows you to search for issues in Jira and pinpoint exactly what you are looking for. Knowing how to search your Jira instance

JQL Basics with Syntax - Atlassian Community Jira Query Language (or JQL) is one of the most powerful tools available in Jira. The system uses the following data to filter issues. And, in turn, helps you find what you're

JQL cheat sheet - Atlassian JQL is a powerful search language specifically designed for Jira that allows users to create complex queries for issue tracking, advanced filtering, and reporting

Visor’s JQL Cheat Sheet Essentially, JQL is what’s called a query language: a specific way of writing asks to get the most flexible search possible from a database. Here, we go over the basics of how to use this

JQL Cheat Sheet: Master JQL Query Language & Syntax JQL is a query language used to filter issues in Jira. It helps users to create structured and complex searches to extract unique information or data from the JIRA tool

Introduction to JIRA Query Language (JQL) - The Requirements Discover the power of JIRA Query Language (JQL) with practical examples, syntax tips, and visualization ideas

Use advanced search with Jira Query Language (JQL) | Jira Cloud You can use the Jira Query Language (JQL) to specify criteria that cannot be defined in the quick or basic searches. For example, you can use the ORDER BY clause in a JQL query to search

JQL operators - Jira Cloud | Atlassian Support An operator in JQL is one or more symbols or words, which compares the value of a field on its left with one or more values (or functions) on its right, such that only true results are retrieved

Unlock Advanced Search with Jira Query Language (JQL) - Atlassian Advanced search allows you to build structured queries using Jira Query Language (JQL) to search for work items within and across projects. Query results can be saved and used as

The Ultimate JQL Cheat Sheet: How To Get Started, from an Expert Essentially, JQL is what’s called a query language: a specific way of writing asks to get the most flexible search possible from a database. It can be very powerful, but it also can

The Jira JQL Advanced Guide: How to Search Jira Issues Like a JQL, or Jira Query Language, is a flexible tool that allows you to search for issues in Jira and pinpoint exactly what you are looking for. Knowing how to search your Jira instance

JQL Basics with Syntax - Atlassian Community Jira Query Language (or JQL) is one of the most powerful tools available in Jira. The system uses the following data to filter issues. And, in turn, helps you find what you’re

JQL cheat sheet - Atlassian JQL is a powerful search language specifically designed for Jira that allows users to create complex queries for issue tracking, advanced filtering, and reporting

Visor’s JQL Cheat Sheet Essentially, JQL is what’s called a query language: a specific way of writing asks to get the most flexible search possible from a database. Here, we go over the basics of how to use this

JQL Cheat Sheet: Master JQL Query Language & Syntax JQL is a query language used to filter issues in Jira. It helps users to create structured and complex searches to extract unique information or data from the JIRA tool

Introduction to JIRA Query Language (JQL) - The Requirements Discover the power of JIRA Query Language (JQL) with practical examples, syntax tips, and visualization ideas

Use advanced search with Jira Query Language (JQL) | Jira Cloud You can use the Jira Query Language (JQL) to specify criteria that cannot be defined in the quick or basic searches. For example, you can use the ORDER BY clause in a JQL query to search

JQL operators - Jira Cloud | Atlassian Support An operator in JQL is one or more symbols or words, which compares the value of a field on its left with one or more values (or functions) on its right, such that only true results are retrieved

Unlock Advanced Search with Jira Query Language (JQL) - Atlassian Advanced search allows you to build structured queries using Jira Query Language (JQL) to search for work items within and across projects. Query results can be saved and used as

The Ultimate JQL Cheat Sheet: How To Get Started, from an Expert Essentially, JQL is what’s called a query language: a specific way of writing asks to get the most flexible search possible from a database. It can be very powerful, but it also can

The Jira JQL Advanced Guide: How to Search Jira Issues Like a JQL, or Jira Query Language, is a flexible tool that allows you to search for issues in Jira and pinpoint exactly what you are looking for. Knowing how to search your Jira instance

JQL Basics with Syntax - Atlassian Community Jira Query Language (or JQL) is one of the most powerful tools available in Jira. The system uses the following data to filter issues. And, in turn, helps you find what you're

Related Articles

- gmat official guide 2022 free download
- godfrey pontoon wiring diagram
- mastery problem 2 m

Back to Home