Elastic Search

- Elastic Search is built on top of a search software known as Lucene,
- The underlying data structure used in that software is known as an inverted index.
- And in computer science, an inverted index basically maps words or numbers to the actual document locations of where those words or term

Converting the document into an inverted index

Let us take an example of two sentence

1. The thin lifeguard was swimming in the lake
2. Swimmer's race with the skinny lifeguard in the lake

Token	Exist In
Swimmers	2
The	1
lake	1,2
lifeguard	1,2
race	2
skinny	2
swimming	1
The	1,2
was	1
with	2

To convert the above table into an inverted index there is a process called text analyzer. In the text analyzer, there will be two steps to perform the inverted index

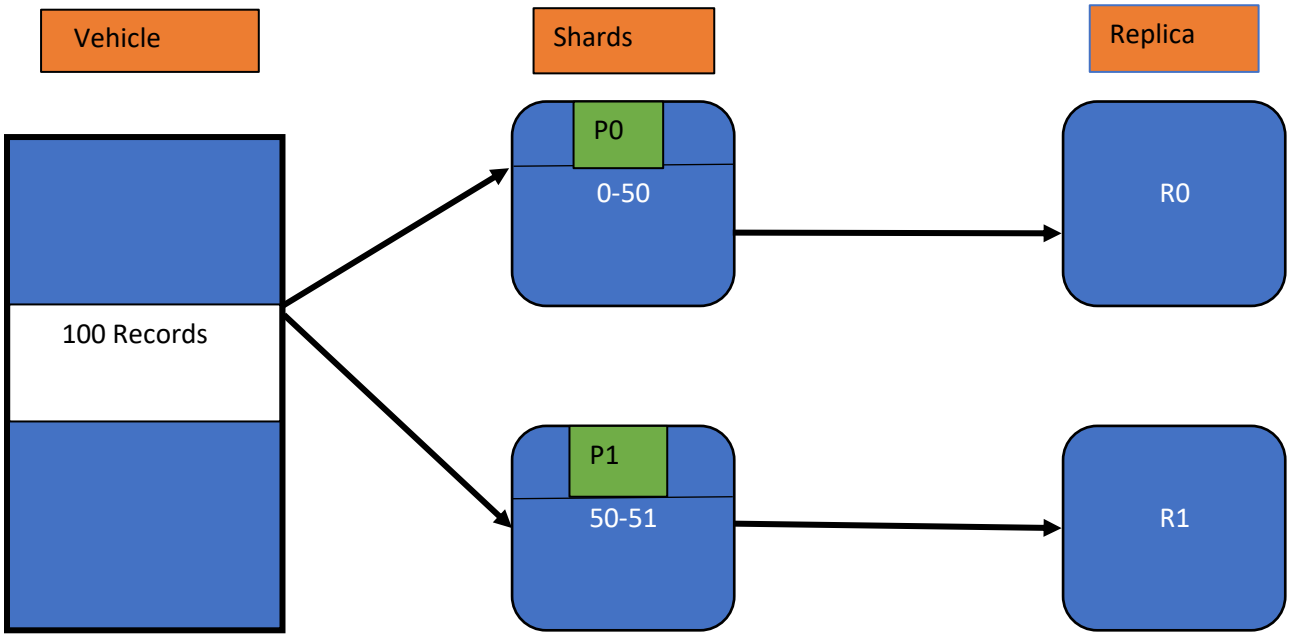
Tokenizer: A *tokenizer* receives a stream of characters, breaks it up into individual *tokens* (usually individual words), and outputs a stream of *tokens*

Filter:

- Remove stop words: remove stop words such as “the”, “and”
- Lower case: remove all lower case of all the data
- Stemming: removing the root word
- Synonyms: index the any one of the synonyms words

Inverted Index of above Table

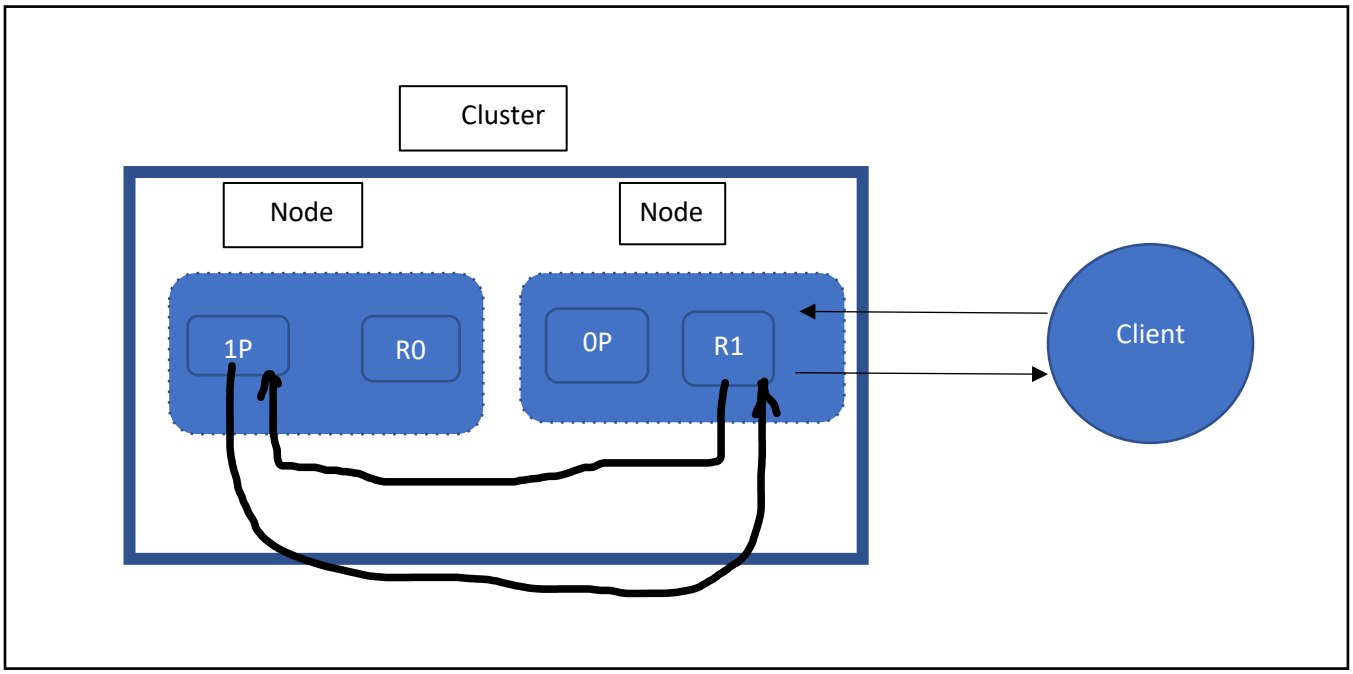
Token	Exist In
in	1,2
lake	1,2
lifeguard	1,2
race	2
swim	1,2
thin	1,2
was	1
with	2



Shard: The shard is the unit at which Elasticsearch distributes data around the cluster

Replica:
A replica shard is a copy of a primary shard. Replicas provide redundant copies of your data to protect against hardware failure and increase capacity to serve read requests like searching or retrieving a document.

- By taking the above diagram as a reference, we take an example of an index of vehicles that consists of 100 records now these 100 records are split into 50-50 for each shard I.e., P0, P1, and further these shards have a replica I.e., R0, R1

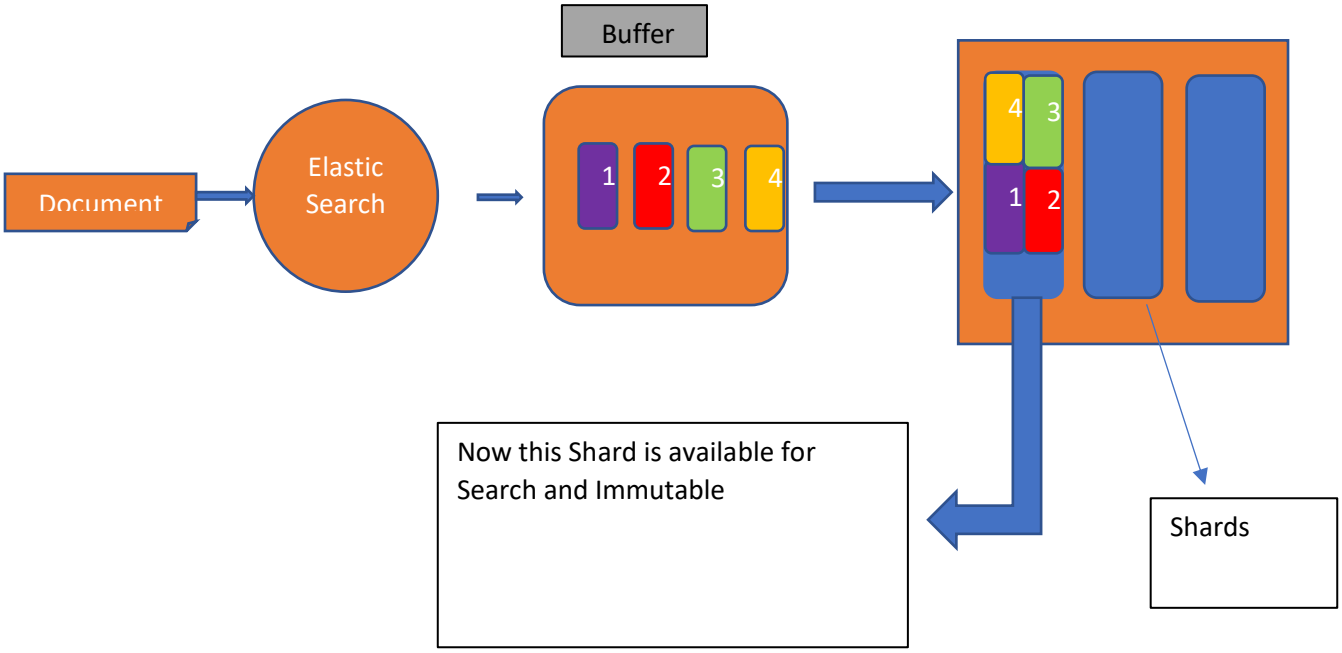


Cluster: cluster is a group of nodes that have the same cluster.name attribute.
Node: A node is usually a physical computer system with a distinct host IP address that is running one or more application servers.

- In the above example, the cluster has two nodes, each node has respective sherds and replica whenever the client request data and the data presented in node –2 at the replica and(R1) also in node –1 at shard(1P)

- According to the load and availability of node, the data may come from the Replica(R1) or Sherd (1P)

How Exactly the search Works:



- The process of inserting data into Elasticsearch has a special terminology, and that's called indexing, So, index a document, that means to insert the document into Elasticsearch.

How Elastic search relates to a Data base

Relational DB	Elastic Search
Table	Index
Row	Document
Column	Field

Data base Table

Employees

userId	jobTitleName	firstName	lastName	preferredFullName	employeeCode	region	phoneNumber	emailAddress
rirani	Developer	Romin	Irani	Romin Irani	E1	CA	408-1234567	romin.k.irani@gmail.com
nirani	Developer	Neil	Irani	Neil Irani	E2	CA	408-1111111	neilrirani@gmail.com

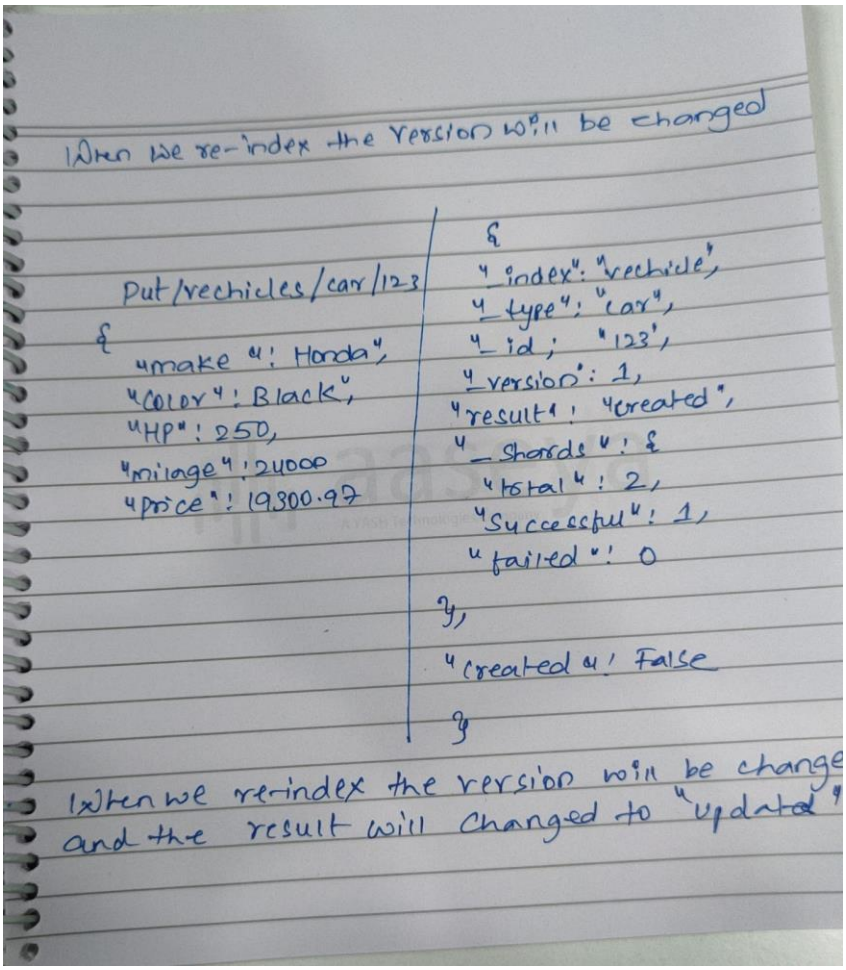
The Data base table is converted into below JSON document by the elastic Search

Elastic search is a document Oriented

```
{
```

```
"Employees" : [
{
"userId":"rirani",
"jobTitleName":"Developer",
"firstName":"Romin",
"lastName":"Irani",
"preferredFullName":"Romin Irani",
"employeeCode":"E1",
"region":"CA",
"phoneNumber":"408-1234567",
"emailAddress":"romin.k.irani@gmail.com"
},
{
"userId":"nirani",
"jobTitleName":"Developer",
"firstName":"Neil",
"lastName":"Irani",
"preferredFullName":"Neil Irani",
"employeeCode":"E2",
"region":"CA",
"phoneNumber":"408-1111111",
"emailAddress":"neilrirani@gmail.com"
}
]
}
```


- When we re-index the version will be changed, and the results will change to update



Reference link: - https://docs-previous.pega.com/system-administration/87/elasticsearch?

Elastic Search in PEGA: -

- The search engine in Pega Platform retrieves results by performing a search query that you specify in the search gadget of your application.
- Pega Platform supports embedded Elasticsearch or Search and Reporting Service (SRS) as the search engine, depending on your deployment.
- **Elastic Search: -**
  Elasticsearch is a third-party search engine that efficiently analyzes enormous volumes of data to ensure a rapid approach to find pertinent information within applications.

- **Search and reporting service: -**
  In order to provide efficient search capabilities, the Search and Reporting Service externalizes full-text search features onto a separate microservice. Because you can easily implement new service functionalities.

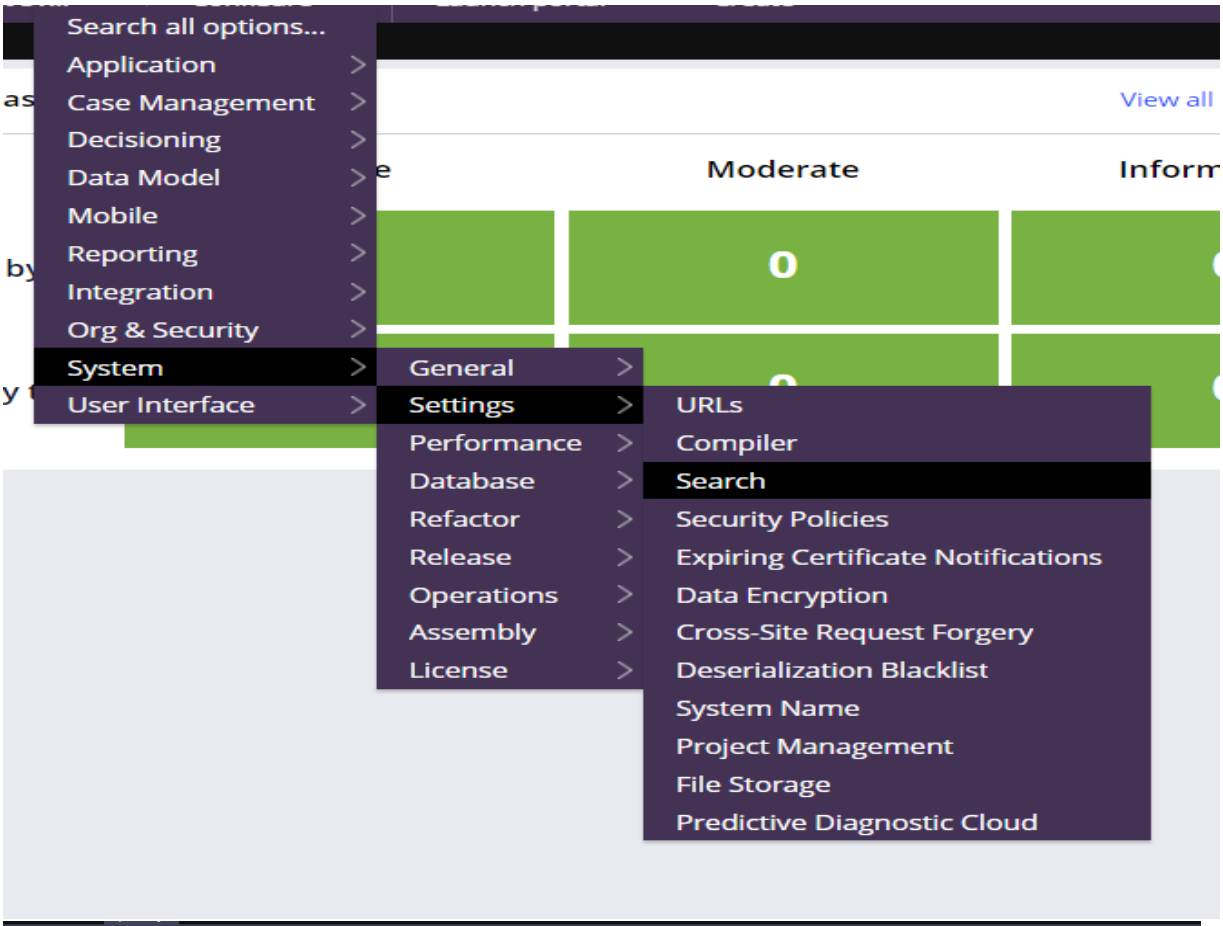
Search Landing Page Overview: -

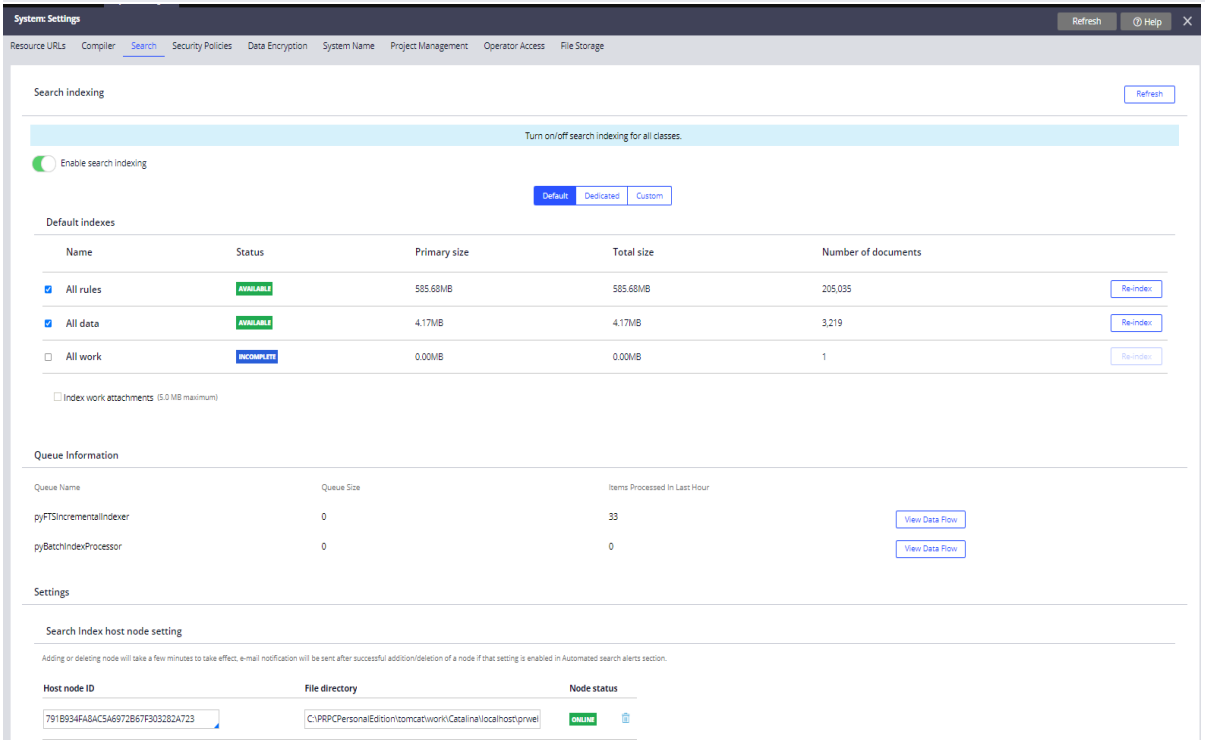
- The search landing page makes it easier for you to manage the data that is made available to the Search and Reporting Service (SRS).

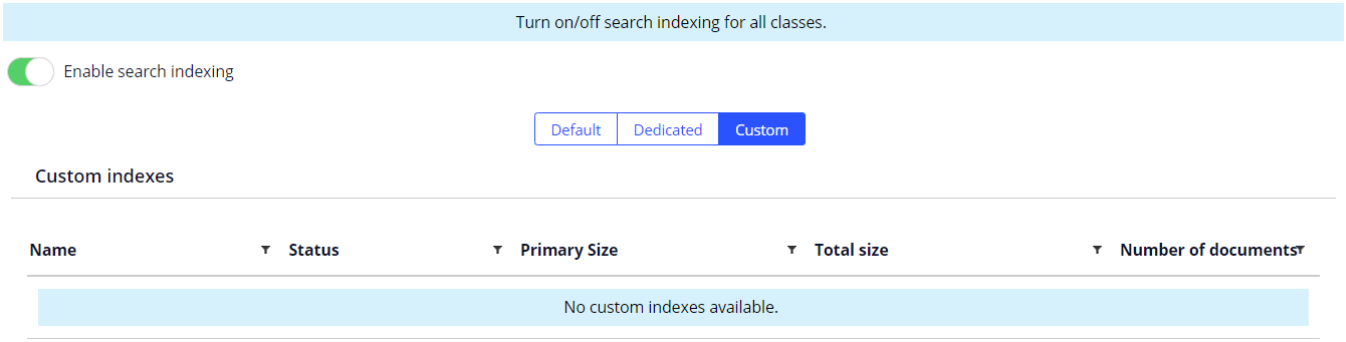
- You can choose the classes of certain applications you wish to index to make them searchable. For instance, to make bug resolution data searchable, you can index a class that holds information about resolved bugs.

You can perform the following actions on search landing page:

- Add properties or classes that you want to index by adding custom search properties.
- Synchronize specific data to make the applications up to date.
- For example, you can manually synchronize after resolving issues with broken items.
- Verify the status of data that you index and analyze potential errors.
- Index the data of specific classes into the SRS.
- Manage **fuzzy** search parameters to enable the service to search for similar words that you enter.
- For example, if you search for **tst**, the system returns test and tests. The results depend on the fuzzy search parameters that you set.


Search all options...
Application
Case Management
Decisioning
Data Model
Mobile
Reporting
Integration
Org & Security
System
User Interface
General
Settings
Performance
Database
Refactor
Release
Operations
Assembly
License
URLs
Compiler
Search
Security Policies
Expiring Certificate Notifications
Data Encryption
Cross-Site Request Forgery
Deserialization Blacklist
System Name
Project Management
File Storage
Predictive Diagnostic Cloud
View all
Moderate
Inform
0


System: Settings
Refresh
Help
Resource URLs
Compiler
Search
Security Policies
Data Encryption
System Name
Project Management
Operator Access
File Storage
Search indexing
Refresh
Turn on/off search indexing for all classes.
Enable search indexing
Default
Dedicated
Custom
Default indexes
Name
Status
Primary size
Total size
Number of documents
All rules
AVAILABLE
585.68MB
585.68MB
205,035
Re-index
All data
AVAILABLE
4.17MB
4.17MB
3,219
Re-index
All work
INCOMPLETE
0.00MB
0.00MB
1
Re-index
Index work attachments (5.0 MB maximum)
Queue Information
Queue Name
Queue Size
Items Processed In Last Hour
pyFTSIncrementalIndexer
0
33
View Data Flow
pyBatchIndexProcessor
0
0
View Data Flow
Settings
Search Index host node setting
Adding or deleting node will take a few minutes to take effect. e-mail notification will be sent after successful addition/deletion of a node if that setting is enabled in Automated search alerts section.
Host node ID
File directory
Node status
791B934FA8AC5A6972B67F303282A723
C:\PRPCPersonalEdition\tomcat\work\Catalina\localhost\prwel
ONLINE



Default indexes

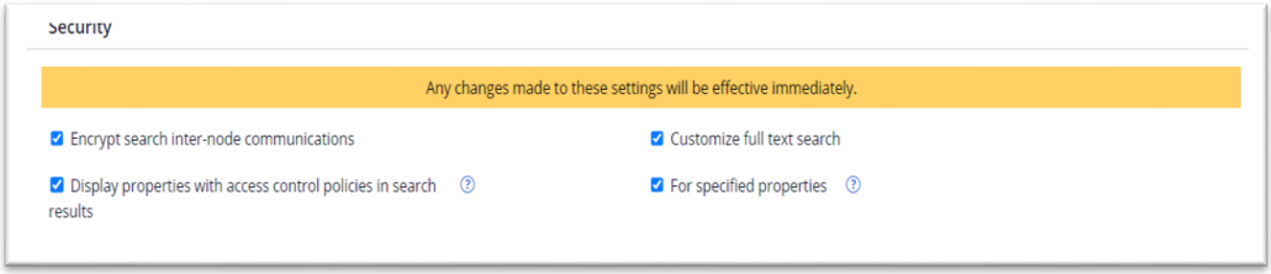
Type less indexes for *Rules-*, *Data-*, and *Work-* classes.

Dedicated indexes

Indexes for specific classes that you can independently configure.

Custom indexes

Special purpose indexes that you create and manage outside of Dev Studio.

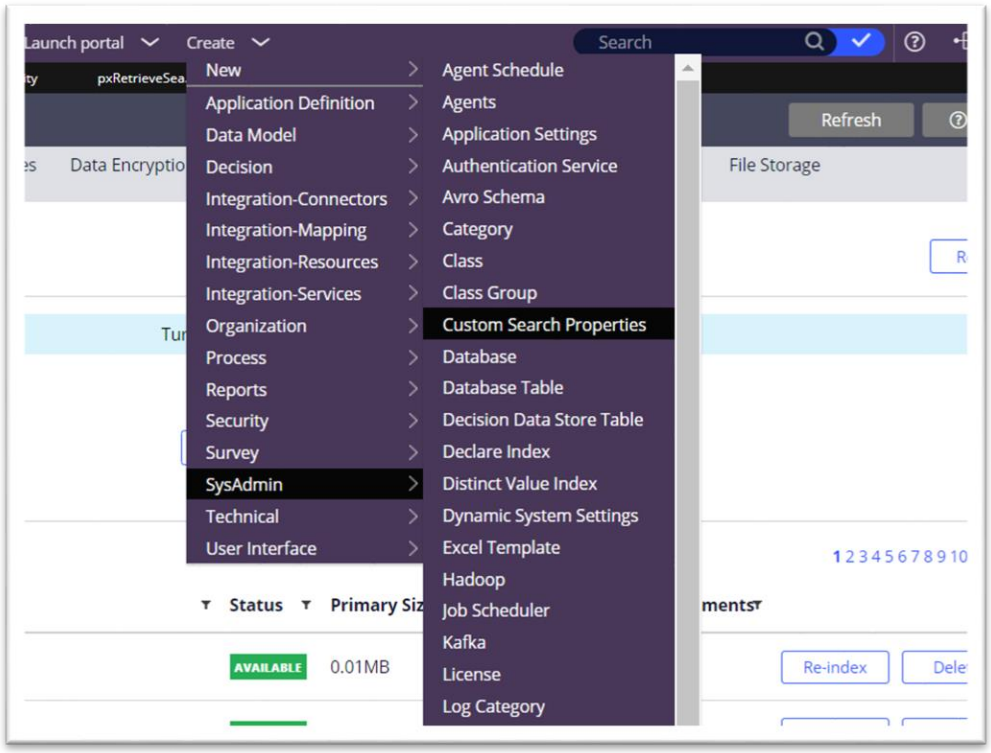


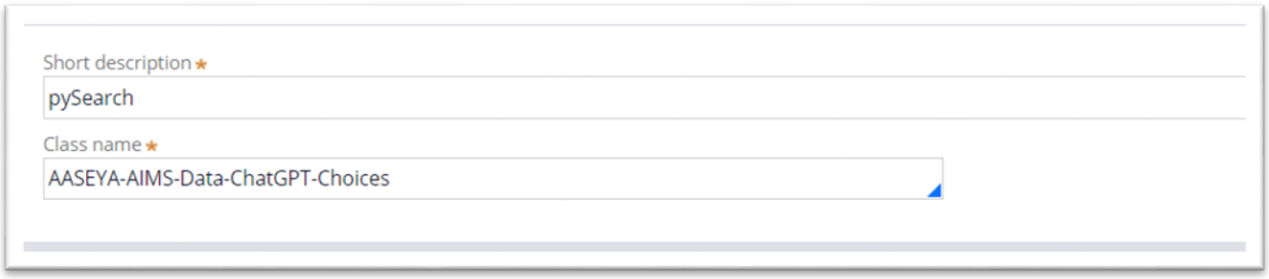
- ❑ **Display properties with access control polices in search results**

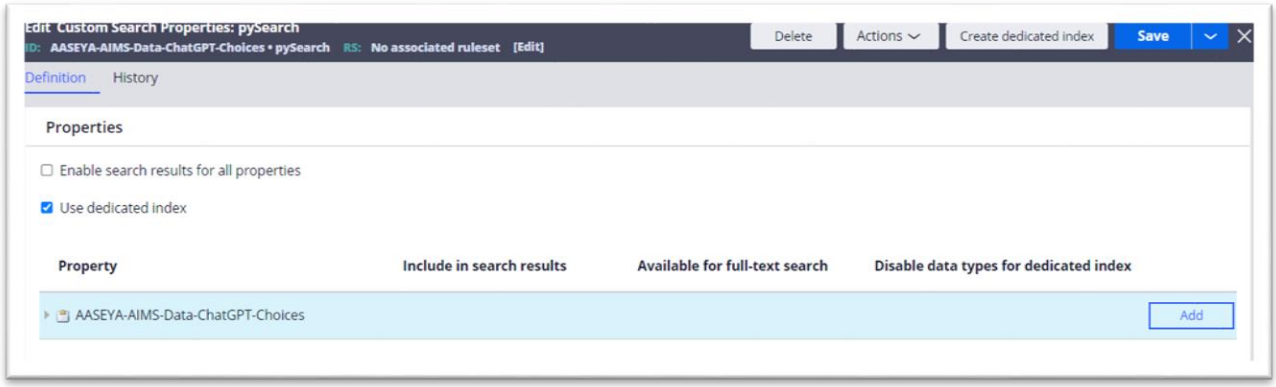
You can choose whether to include or exclude properties with access control rules from search results, allowing you to manage who has access to private information.

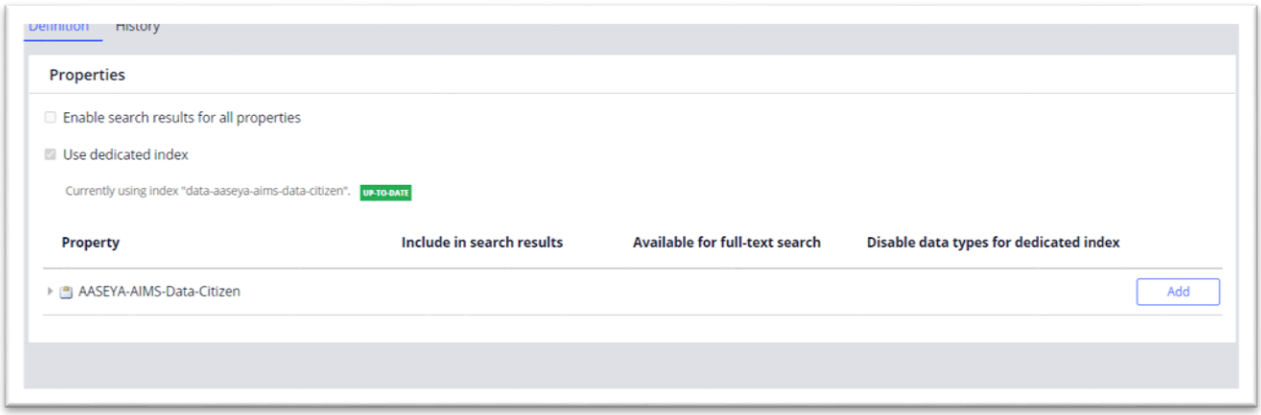
Access control policies are disregarded when the option to display properties with access control policies is chosen. Whether or not the access control policies are satisfied, properties with access control policies are not included in search results when you choose not to display properties with access control policies.

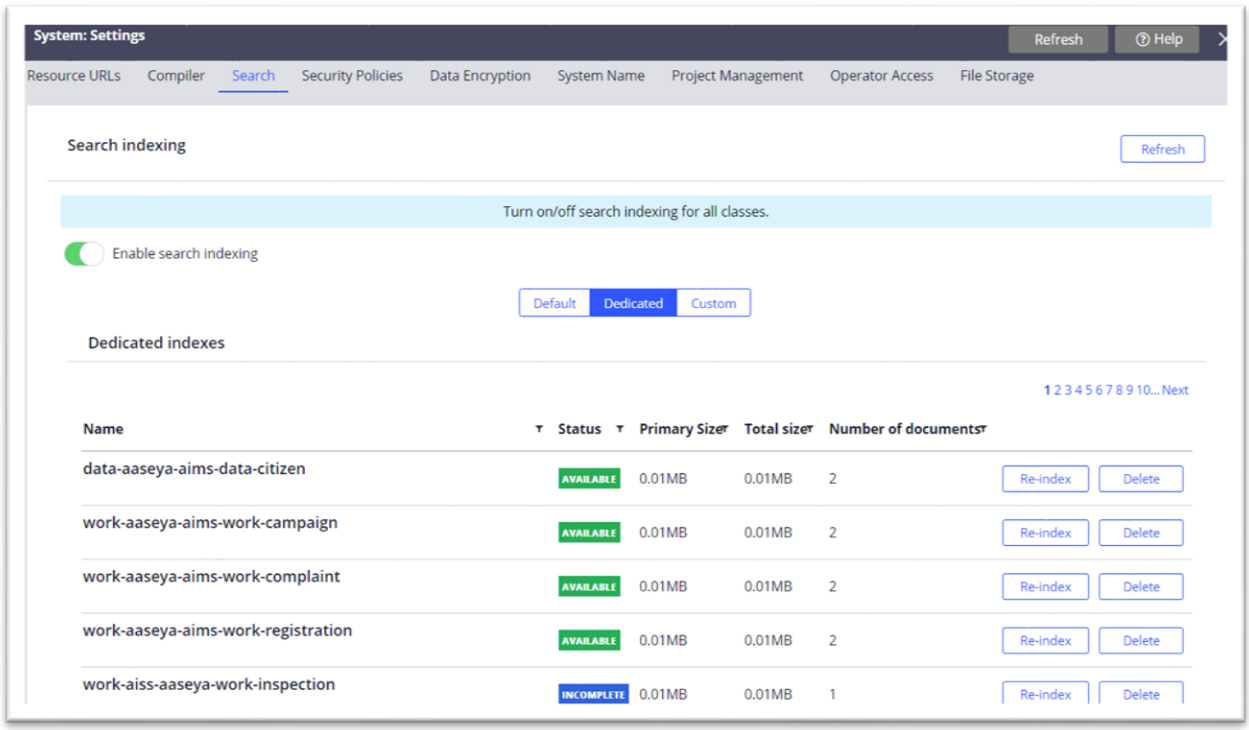
Specifying custom search properties






Edit Custom Search Properties: pySearch
ID: AASEYA-AIMS-Data-ChatGPT-Choices • pySearch RS: No associated ruleset [Edit]
Delete
Actions
Create dedicated index
Save
Definition
History
Properties
Enable search results for all properties
Use dedicated index
Property
Include in search results
Available for full-text search
Disable data types for dedicated index
AASEYA-AIMS-Data-ChatGPT-Choices
Add


Definition
History
Properties
Enable search results for all properties
Use dedicated index
Currently using index "data-aaseya-aims-data-citizen".
UP-TO-DATE
Property
Include in search results
Available for full-text search
Disable data types for dedicated index
AASEYA-AIMS-Data-Citizen
Add


System: Settings
Refresh
Help
Resource URLs
Compiler
Search
Security Policies
Data Encryption
System Name
Project Management
Operator Access
File Storage
Search indexing
Refresh
Turn on/off search indexing for all classes.
Enable search indexing
Default
Dedicated
Custom
Dedicated indexes
1 2 3 4 5 6 7 8 9 10 ... Next
Name
Status
Primary Size
Total size
Number of documents
data-aaseya-aims-data-citizen
AVAILABLE
0.01MB
0.01MB
2
Re-index
Delete
work-aaseya-aims-work-campaign
AVAILABLE
0.01MB
0.01MB
2
Re-index
Delete
work-aaseya-aims-work-complaint
AVAILABLE
0.01MB
0.01MB
2
Re-index
Delete
work-aaseya-aims-work-registration
AVAILABLE
0.01MB
0.01MB
2
Re-index
Delete
work-aiss-aaseya-work-inspection
INCOMPLETE
0.01MB
0.01MB
1
Re-index
Delete

Dedicated Search Indexes: -

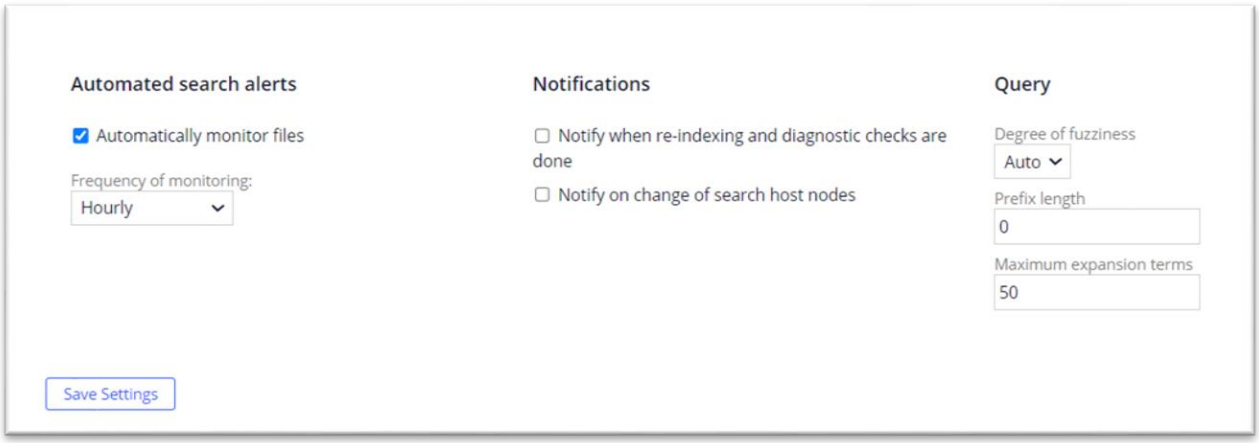
Dedicated search indexes are property type-aware, per-class indexes that return the data type of each returned property. Default indexes are type less, that is, all properties are of type string. Dedicated indexes let you execute queries with filters, date calculations, and aggregations directly in the query.

How it works

You can create a dedicated index for any class, even if the class contains descendant classes. You can configure individual properties in a class to be type less, that is, they will be returned as strings.

A data type conflict occurs when an index contains two fields with the same name that are of different data types. If a conflict occurs, indexing is handled as follows:

- If there is a conflict, the class is indexed in the *Work-* or *Data-* type less index.
- If there is no conflict, the class is indexed in the dedicated index.
- If a property is added in production that results in a data type conflict, the class's index status is either changed to "Conflicts Found" until the conflicts are resolved, or to "Recreate index" if the instances of the class have already been indexed. The status is displayed on the Custom Search Properties page for the class and on the **Search** landing page.



Automatic Search Alerts:

- **Automatically Monitor Files** – Select to automatically review the index files at the interval set in the **Frequency of Monitoring** field

Notifications:

- **Notify when re-indexing and diagnostic checks are done** – Select to send notifications after reindexing and performing diagnostic checks. When checked, enter a list of email addresses to send notifications, and select the email account to use for sending the messages.

- **Notify on change of search host nodes** – Select to send notifications when a search host node has been modified. When checked, enter a list of email addresses to send notifications, and select the email account to use for sending the messages.

- Click **Save Settings**.

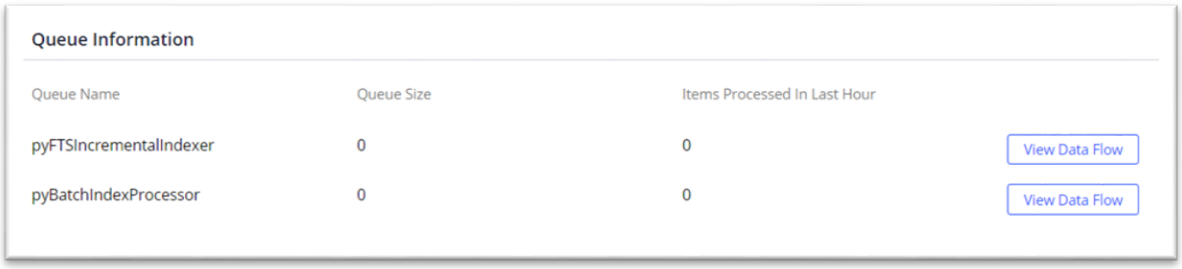
fuzzy search:

1. In the **Settings** section, in the **Degree of fuzziness** field, select the edit distance, that is, the search string length within which the approximate matching of characters occurs. If you select **Auto**, the maximum edit distance is:
- 0 for strings of one or two characters.
- 1 for strings of three, four, or five characters.
- 2 for strings of more than five characters.

- For example, performing a fuzzy search query for the term "catchr" with a degree of fuzziness of 1, finds matches like "catch" (by deleting 1 character) and "catcher" (by adding 1 character), but does not find matches like "catches" (by adding 1 character and replacing 1, which adds up to an edit distance of 2).

2. In the **Prefix length** field, enter the number of initial characters in the entered string to which fuzzy matching does not apply, given that the initial characters match exactly. Increasing this parameter value results in faster search queries.
3. For example, performing a fuzzy search query for the term "windwo" with a prefix length of 3, does not apply fuzzy search to the first 3 characters "win," and applies fuzzy search to the rest of the characters to find matches like "window," "winter," "winner," and so on.
4. In the **Maximum expansion terms** field, enter the number of alternative spellings for the search string that you want to allow while searching. Decreasing this parameter value results in faster search queries, but might not return as many potential matches.
5. For example, performing a fuzzy search query for the term "codngi" with 2 maximum expansion terms, finds only 2 matches like "code" and "coding".
6. Click **Save Settings**.

Queue Processors in Search Index: -

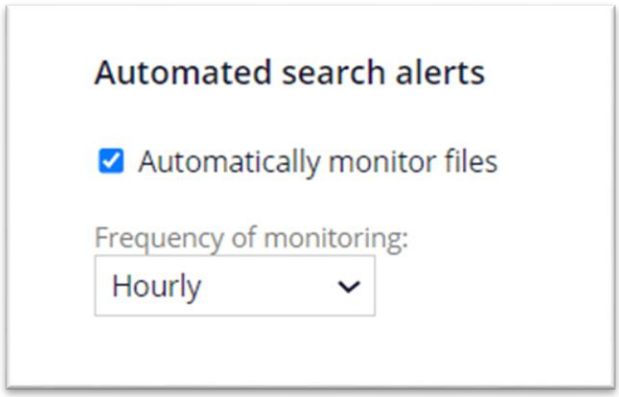


Batch Index Queue Processor: -

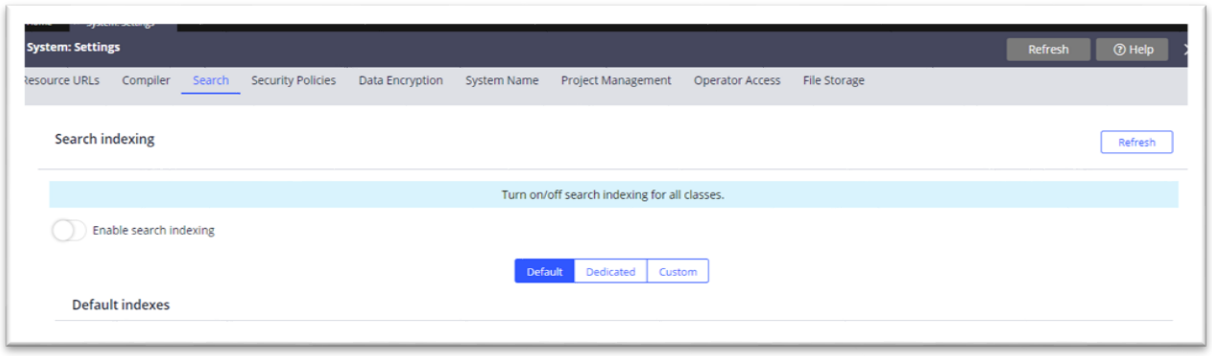
This queue processor is responsible for creating index. When we click on the Re-Index background this queue processor will run and creates an index of bulk rules in pega.

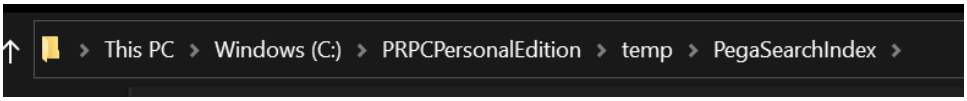
Incremental Indexer Queue Processor: -

This queue processor is responsible for inserting new data or record to index. When any rule got created in pega this processor helps in inserting new record or Data to index table. Also based on Automated search alerts frequency of monitoring also it will update to the table

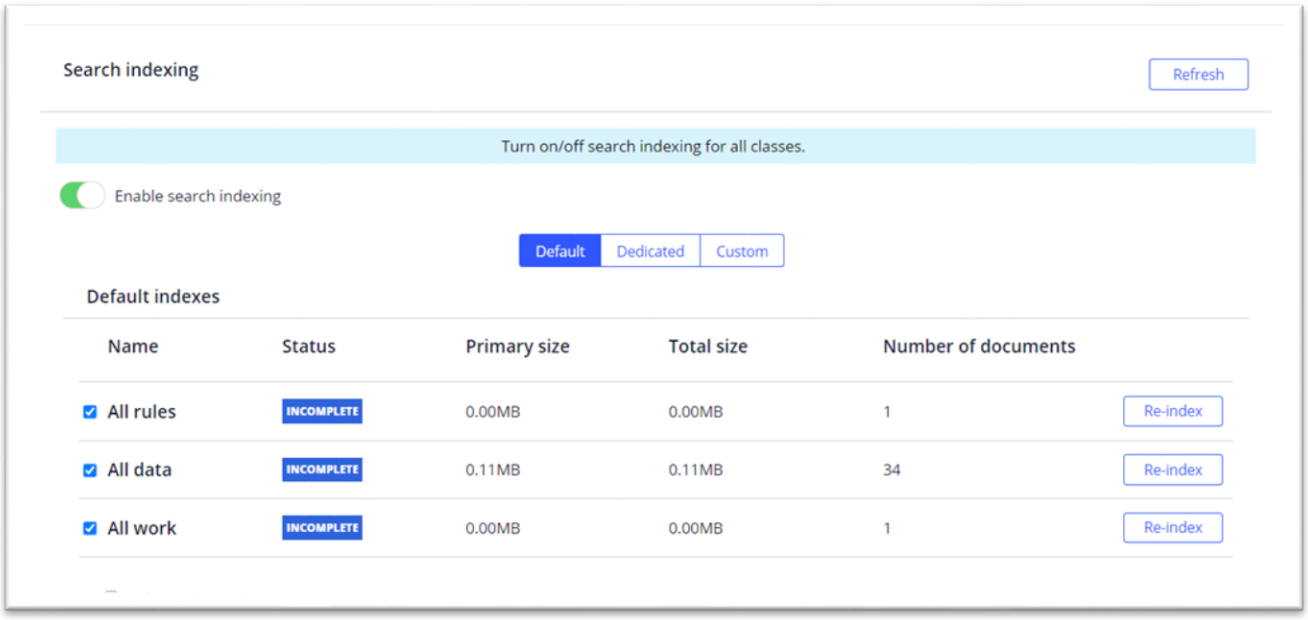


To test how re-index and batch queue processor works.

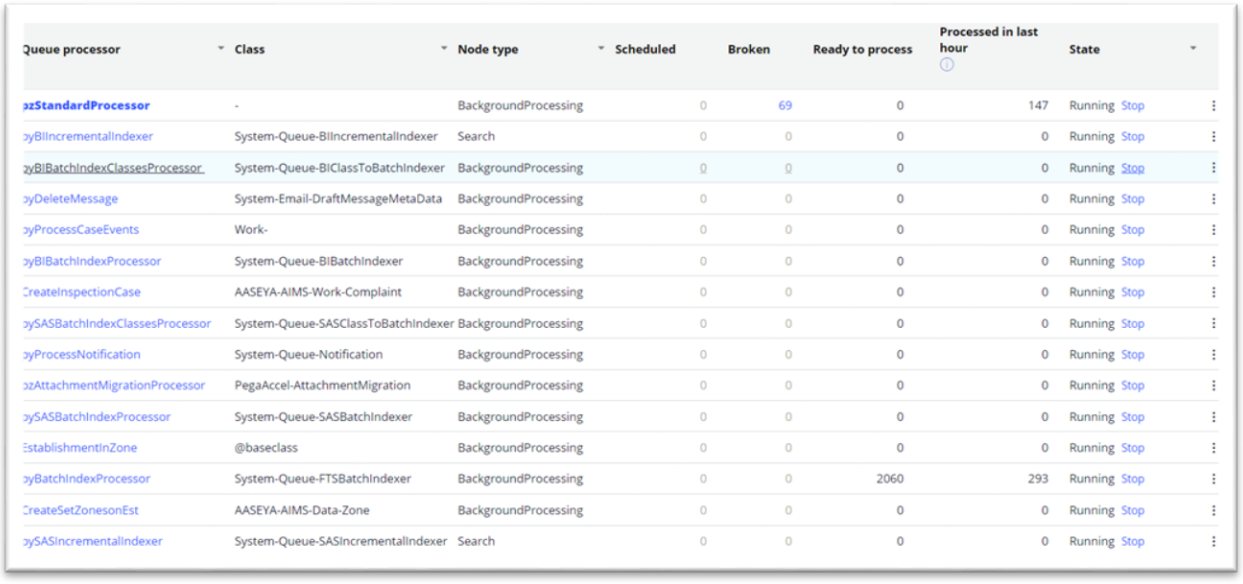


1. Disable the search indexing.
2. 

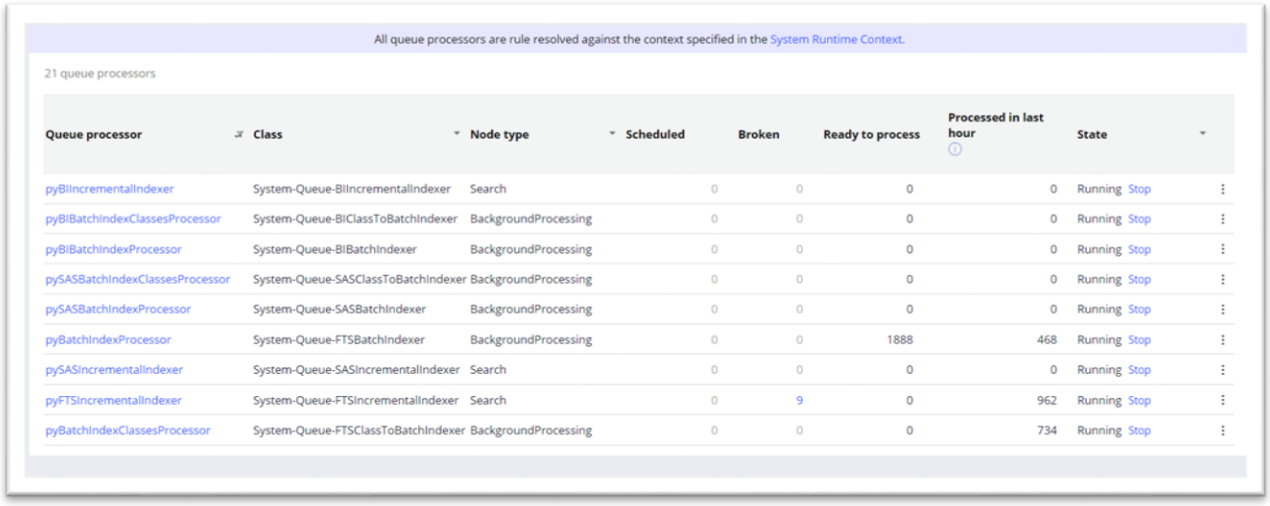
3. To test we can delete search index folder.
4. In dev studio disable the search indexing.
5. Then shutdown the server and then delete folders inside the search index folder.
6. And again, start the server you can find incomplete status in search index .



6. You can find some node files created while running the server.
7. Click Re-index to find batch queue processor runs in admin studio.



in this you can find pyBatchIndexProcessor running. That is ready to process.



All queue processors are rule resolved against the context specified in the System Runtime Context.

21 queue processors

Queue processor	Class	Node type	Scheduled	Broken	Ready to process	Processed in last hour	State
pyBIIncrementalIndexer	System-Queue-BIIncrementalIndexer	Search	0	0	0	0	Running Stop
pyBIBatchIndexClassesProcessor	System-Queue-BIClassToBatchIndexer	BackgroundProcessing	0	0	0	0	Running Stop
pyBIBatchIndexProcessor	System-Queue-BIBatchIndexer	BackgroundProcessing	0	0	0	0	Running Stop
pySASBatchIndexClassesProcessor	System-Queue-SASClassToBatchIndexer	BackgroundProcessing	0	0	0	0	Running Stop
pySASBatchIndexProcessor	System-Queue-SASBatchIndexer	BackgroundProcessing	0	0	0	0	Running Stop
pyBatchIndexProcessor	System-Queue-FTSBatchIndexer	BackgroundProcessing	0	0	1888	468	Running Stop
pySASIncrementalIndexer	System-Queue-SASIncrementalIndexer	Search	0	0	0	0	Running Stop
pyFTSIncrementalIndexer	System-Queue-FTSIncrementalIndexer	Search	0	9	0	962	Running Stop
pyBatchIndexClassesProcessor	System-Queue-FTSClassToBatchIndexer	BackgroundProcessing	0	0	0	734	Running Stop

After re-indexing the status of the rules becomes too available.