



How to Search for Information in MidPoint MidPoint Query Language for Engineers

Martin Špánik, March 2024

Senior Identity Engineer

Agenda

- Overview
- midPoint Query Language
 - Language structure and elements
 - Querying references
 - Expressions
- Examples and tips
 - Basic search
 - Displaying object relations
 - Search in audit
- Real usage example
- Tools and documentation



Why MidPoint Query Language ?

- XML Query

```
<and>
  <substring>
    <path>emailAddress</path>
    <value>gmail.com</value>
    <anchorEnd>true</anchorEnd>
  </substring>
  <equal>
    <path>locality</path>
    <value>Stockholm</value>
  </equal>
</and>
```

- MidPoint Query Language

```
emailAddress endsWith 'gmail.com' and
locality = "Stockholm"
```

Overview of MidPoint Query Language

- Primary query language in midPoint
- Bounded to midPoint object model
 - Querying object relations
- Infix instead of prefix notation
- Available everywhere in midPoint
 - GUI – as advanced search
 - Configuration files
 - Code



Advanced Search – MidPoint Query Language in GUI

- Object type for query defined by view

✓ Advanced

<input type="checkbox"/>	^ Name	Personal Number	Full name	Email	Accounts	
<input type="checkbox"/>	 jane		Jane Smith	jane@testorg.com		
<input type="checkbox"/>	 john		John Brown	john.brown@testorg.com		

+ ↑ ↓ 🔄 ▶

Rows per page 20 1 to 2 of 2 << < 1 > >>

Name ⓘ john

Full name ⓘ John Brown

Given name ⓘ John

Family name ⓘ Brown

Email ⓘ john.brown@testorg.com

Locality ⓘ Stockholm

Show empty fields

MidPoint Query Language in Configuration

- XML - wrapped inside **<text>** element inside **<filter>** element within query
 - ```
<query>
 <filter>
 <text>name startsWith "J"</text>
 </filter>
</query>
```
  - Object type must be defined
- Query in Groovy code (API)
  - ```
import com.evolveum.midpoint.xml.ns._public.common.common_3.*
def query = midpoint.queryFor(UserType.class, "name startsWith 'J'")
def result = midpoint.searchObjects(query)
```

MidPoint Object Structure - User

```
<user xmlns...
  oid="346b732b-95f1-417d-b632-e5324d45dd00">
    <name>amkin</name>
    <extension xmlns:gen987="http://custom-schema/midpoint">
      <gen987:empStartDate>2021-09-01T00:00:00.000+02:00</gen987:empStartDate>
    </extension>
    <lifecycleState>active</lifecycleState>
    <assignment>
      <targetRef oid="a5f9fe1e-69a2-459b-ae65-f914bb0d40b1" relation="org:default" type="c:ArchetypeType"/>
    </assignment>
    <assignment>
      <targetRef oid="13b0c900-4849-4bb7-99cc-30a4998606e6" relation="org:default" type="c:RoleType"/>
    </assignment>
    <linkRef oid="b61df42d-ff8b-405d-bb6c-23b0f9675440" relation="org:default" type="c:ShadowType"/>
    <linkRef oid="f8888ae5-be2f-41c8-b048-b8d4f4b8dfa3" relation="org:default" type="c:ShadowType"/>
    <activation>
      <effectiveStatus>enabled</effectiveStatus>
      <enableTimestamp>2024-02-29T21:55:48.297+01:00</enableTimestamp>
    </activation>
    <locality>Leeds</locality>
    <emailAddress>amanda.king@testorg.com</emailAddress>
    <givenName>Amanda</givenName>
    <familyName>King</familyName>
    <organizationalUnit>D2</organizationalUnit>
    <personalNumber>50</personalNumber>
  </user>
```

MidPoint Query Language – Basic Info

- Item filterName value
 - `fullName = "John Doe"`
 - `givenName startsWith "J"`
 - `activation/effectiveStatus = "enabled"`
- Item
 - Item path to the attribute (name, not display name)
- Filters:
 - `=, <, >, !=, <=, >=`
 - `startsWith, endsWith, contains, fullText`
 - `exists`
 - Dates are compared as strings (ISO 8601 date format)
- Logical operators
 - `and, or, not`
 - `familyName="Doe" and not (givenName="John" or givenName="Bill")`
- String values enclosed by single (') or double quotes (")



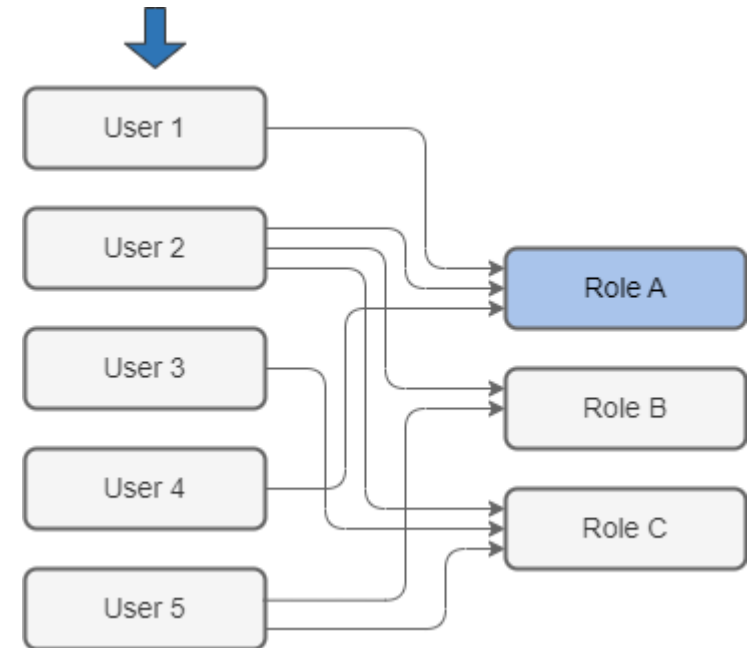
MidPoint Query Language – Query Matching Rules

- Case sensitivity specification of comparison filters (mostly)
 - `filter[matchingRuleName]`
- Matching rules are different for polyStrings and strings !
 - `stringIgnoreCase` – for string attributes
 - `origIgnoreCase` – for polystrings
- Examples
 - `emailAddress endsWith[stringIgnoreCase] "@testorg.com"`
 - `familyName contains[origIgnoreCase] "son"`
- Check attribute type at “Searchable items” page in docs



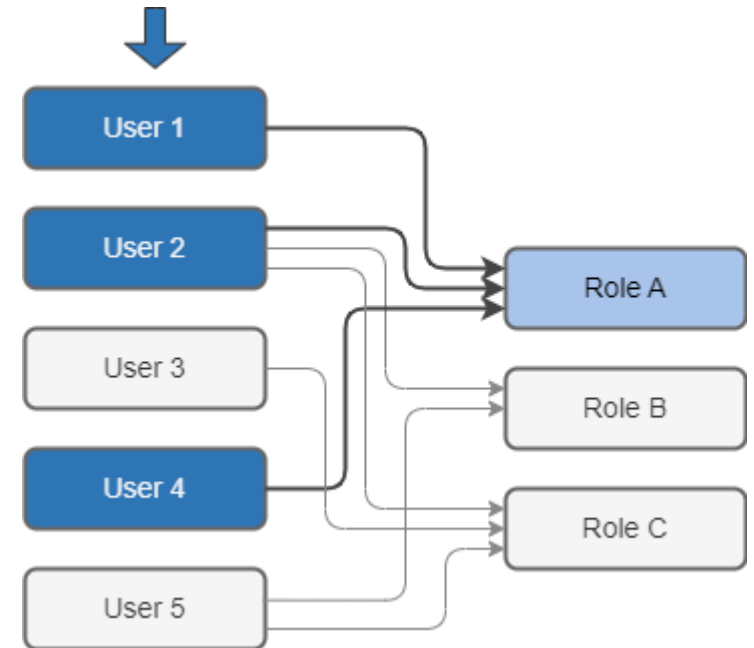
MidPoint Query Language - Querying References 1/2

- Search for all objects that “**have reference**” of the object
- Assignments / Inducements
 - `assignment/targetRef`
 - `inducement/targetRef`
- Linked accounts on resources
 - `linkRef`
- Archetypes
 - `archetypeRef`
- Indirect assignments
 - `roleMembershipRef`



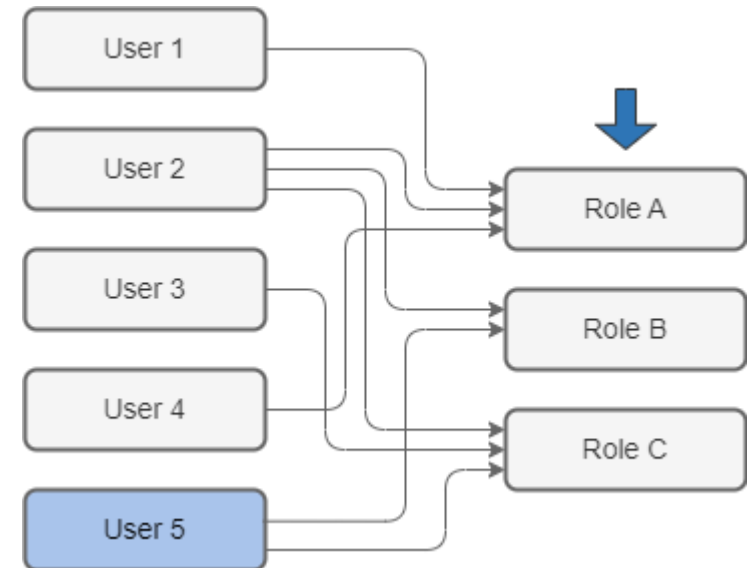
MidPoint Query Language - Querying References 1/2

- Filter: **matches**
 - `assignment/targetRef matches (oid = efaf89f4-77e9-460b-abc2-0fbfd60d9167)`
 - In All users view, list all users that have role with following OID directly assigned
- Dereferencing: **@**
 - `assignment/targetRef/@/name = "Superuser"`
 - In All users view, list all users that have role "Superuser" directly assigned
- Type operator
 - `roleMembershipRef/@ matches (. type ServiceType)`
 - All services directly or indirectly assigned.



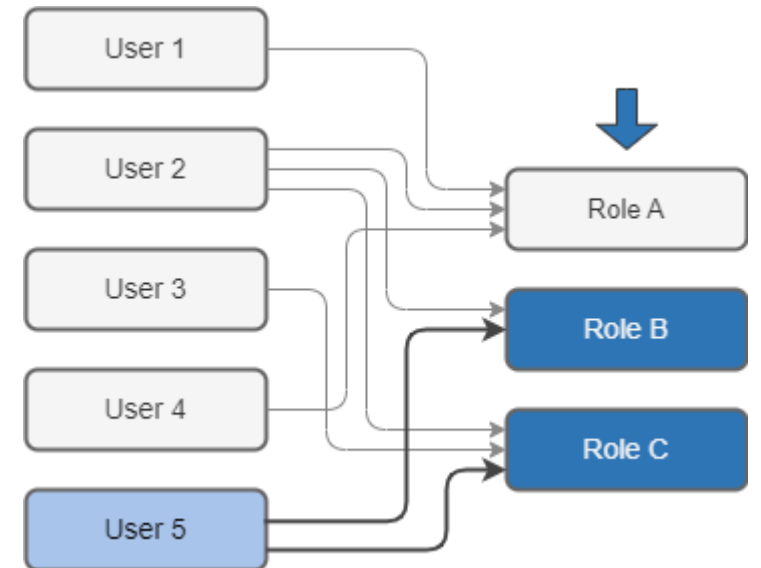
MidPoint Query Language - Querying References 2/2

- Search for all objects that “**are referenced**” by the object



MidPoint Query Language - Querying References 2/2

- Filter: **referencedBy**
 - `. referencedBy (@type=UserType and @path=assignment/targetRef and name="administrator")`
 - In All roles view, list all roles directly assigned to user administrator
- Referenced object identifier: .
- **@type** operator



MidPoint Query Language - Expressions

- Supported in configuration only (not allowed in GUI)
- Script
 - `metadata/createTimestamp > `basic.fromNow("-P30D")``
 - Expression is identified by ` (backtick character)
 - Multiline by ``` (triple backtick, followed by newline)
 - Can use midPoint functions
- Evaluation of search expressions is limited. Fully works in:
 - Dashboards
 - Reports
 - Object collections



Examples – Expressions in Queries

- Object collection – all users created within last 48 hours:

```
<objectCollection ...  
  
  <name>Users created in the last 48 hours</name>  
  <type>UserType</type>  
  <filter>  
    <q:text>  
      metadata/createTimestamp greater ``  
        calendar = basic.addDuration(basic.currentDateTime(), "-P2D")  
        return calendar  
      ``  
    </q:text>  
  </filter>  
</objectCollection>
```

<https://docs.evolveum.com/midpoint/reference/concepts/query/midpoint-query-language/expressions/>



Examples – Queries in Groovy Code

- Standard definition of the query in groovy code

```
import com.evolveum.midpoint.xml.ns._public.common.common_3.*;

def query = midpoint.queryFor(UserType.class, "activation matches
(effectiveStatus = 'enabled' and enableTimestamp >= '2024-03-01')")

def result = midpoint.searchObjects(query)
```

<https://docs.evolveum.com/midpoint/reference/concepts/query/midpoint-query-language/query-language-in-groovy/>



Query in AUDIT

- What has happened
- Searching in
 - Audit events – AuditEventRecordType
 - Deltas – ObjectDeltaOperationType
- Check “Searchable Items” for attribute names
- Limit each search in large audits by timestamps
 - `timestamp >= "2024-03-01"`
- Limitation of querying elements in database columns
 - can't go with query to delta objects



Query Playground and Query Converter

Query playground

Query playground

Query playground

Query converter

MidPoint query

Object type: User

☐ Distinct

☐ Translate from Query API script

1

emailAddress endsWith 'gmail.com'

Or use an example: Undefined

Note: midPoint query contains a filter along with paging instruction, wrapped together within the <query> element. In contrast, when used in "Advanced filter" in GUI, only the <filter> sub-element is applicable. Paging is managed by the GUI itself.

Translate and execute

Translate to SQL query

Use in object list

SQL query

1

select u.oid, u.fullObject

2

from m_user u

3

where u.emailAddress like ?

4

limit ?

Query parameters

1

1 = %gmail.com

2

2 = 10000

Result: retrieved 2 object(s)

1

POV:user:716701f8-38fe-4138-b2f6-a5469be21fcf(cyril)


2


POV:user:f4d4cea8-cc15-4117-93a8-e0b97e7bc8a4(boris)

Query Playground and Query Converter

Query playground

Query playground





Query playground

Query converter

Object type: User

XML

JSON

YAML

XML Query

```
1 <query>
2   <filter>
3     <substring>
4       <path>emailAddress</path>
5       <value>gmail.com</value>
6       <anchorEnd>true</anchorEnd>
7     </substring>
8   </filter>
9 </query>
```

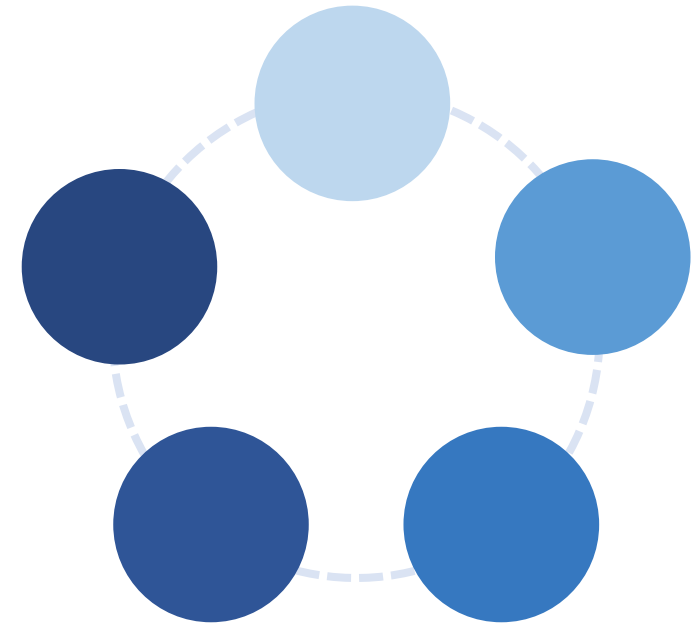
Converted midPoint Query

```
1 emailAddress endsWith 'gmail.com'
```

Convert query

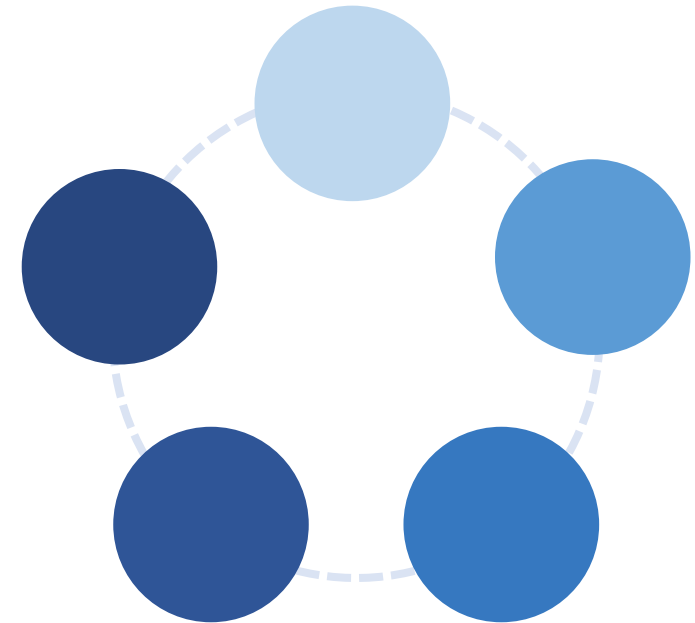
Real Example

- Using midPoint query in GUI
 - Actual state information
 - Users, Roles, Assignments, Accounts
 - What happened – audit search
- Using midPoint query in midPoint Studio
- Converting queries with Query converter



Real Example: Environment

- midPoint 4.8.3-SNAPSHOT
- 2 resources
 - Users – inbound – 50 users – HR import
 - AppAccess – outbound
- Application roles
 - Defining access to applications
 - Assignment creates account and assigns entitlement in AppAccess resource
 - Some roles with riskLevel defined
- Business role
- Assignments and inducements



Documentation

- <https://docs.evolveum.com>
- Search for “query” or “midPoint query”
- <https://docs.evolveum.com/midpoint/reference/concepts/query/midpoint-query-language>



Future

- Semantic autocomplete
 - in 4.9 – easier queries preparation
- Webinar for advanced usage of MQL
 - deeper explanation of language structure
 - more complex examples
- Continual updates of documentation and examples



Conclusion

- Midpoint Query Language
 - More natural and user friendly
 - Same queries available everywhere in midPoint
 - Possible to use by advanced users
 - Query playground and Query converter
- <https://docs.evolveum.com>



Thank you for your attention

Do you have any **questions**? Feel free to contact us at info@evolveum.com

Follow us on social media or **join us** at GitHub or Gitter!



Evolveum

© 2024 Evolveum s.r.o. All rights reserved.