OASIS OPEN PROJECTS

OSLC Query Version 3.0

OASIS Standard
26 August 2021

This stage:

https://docs.oasis-open-projects.org/oslc-op/query/v3.0/os/oslc-query.html (Authoritative)
https://docs.oasis-open-projects.org/oslc-op/query/v3.0/os/oslc-query.pdf

Previous stage:

https://docs.oasis-open-projects.org/oslc-op/query/v3.0/ps01/oslc-query.html (Authoritative)
https://docs.oasis-open-projects.org/oslc-op/query/v3.0/ps01/oslc-query.pdf

Latest stage:

https://docs.oasis-open-projects.org/oslc-op/query/v3.0/oslc-query.html (Authoritative)
https://docs.oasis-open-projects.org/oslc-op/query/v3.0/oslc-query.pdf

Latest version:

https://open-services.net/spec/query/latest

Latest editor's draft:

https://oslc-op.github.io/oslc-specs/specs/query/oslc-query.html

Open Project:

OASIS Open Services for Lifecycle Collaboration (OSLC) OP

Project Chairs:

Jim Amsden (jamsden@us.ibm.com), IBM
Andrii Berezovskyi (andriib@kth.se), KTH

Editors:

Jim Amsden (jamsden@us.ibm.com), IBM

David Honey (david_honey@persistent.com), Persistent Systems Ltd

RDF Namespaces:

http://open-services.net/ns/core#

Abstract:

OSLC Query provides a mechanism for a client to query or search for RDF resources that match a given criteria. The response to a successful query includes the RDF of a *query result container* that references the member resources found by the query, and optionally includes selected properties of each member resource.

Status:

This document was last revised or approved by the membership of OASIS on the above date. The level of approval is also listed above. Check the "Latest stage" location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Open Project are listed at https://open-services.net/about/.

Comments on this work can be provided by opening issues in the project repository or by sending email to the project's public comment list oslc-core-comment.

The English version of this specification is the only normative version. Non-normative translations may also be available. Note that any machine-readable content (Computer Language Definitions) declared Normative for this Work Product is provided in separate plain text files. In the event of a discrepancy between any such plain text file and display content in the Work Product's prose narrative document(s), the content in the separate plain text file prevails.

Citation format:

When referencing this specification the following citation format should be used:
[OSLC-Query-3.0]
OSLC Query Version 3.0. Edited by Jim Amsden and David Honey. 26 August 2021. OASIS Standard. https://docs.oasis-open-projects.org/oslc-op/query/v3.0/os/oslc-query.html. Latest stage: https://docs.oasis-open-projects.org/oslc-op/query/v3.0/oslc-query.html.

Notices

with the IPR Mode of the OASIS Open Project that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Open Project Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see https://www.oasis-open.org/policies-guidelines/trademark for above guidance.

Table of Contents

1. Introduction

This section is non-normative.

OSLC Query provides a mechanism for a client to query or search for RDF resources by performing a `GET` or `POST` on a `oslc:queryBase` URI. Clients may discover such query base URIs from a `oslc:queryCapability` declared in an OSLC service. The response to a successful query is a response body that includes the RDF of a *query result container* that references the member resources matching the query, and optionally contains selected properties of each member resource.

Two separate capabilities may be provided:

- A query based on an optional query expression consisting of at least one property, comparison operator, and value. The query result container only references resources whose properties match the specified query expression. This is analogous to queries specified by SQL or SPARQL.
- A full-text search based on one more search terms [FULLTEXT-SEARCH]. The query result container only references resources that have any text-based properties that contain one or more of the search terms.

Servers may support a combined query and full-text search. Clients may also specify which member properties should be included in the query response.

1.1 Terminology

Terminology is based on OSLC Core Overview [OSLC-Core-30-Part1], W3C Linked Data Platform [LDP], W3C's Architecture of the World Wide Web [WEBARCH], and Hyper-text Transfer Protocol [HTTP11].

Query capability

Describes an HTTP endpoint that may be used to query for or search for artifacts that match the given criteria.

Query base URI

The URI associated with a query capability that a GET and/or POST on will perform a query or search for artifacts.

Query expression

A string that defines one or more query terms, where each term defines a property name, operator, and value.

Query result container

> An RDF container in the response to a successful query or search that references the resources found.

Search term

> A text string or keyword that is to be searched for in any text property of artifacts of a type specified by a query capability.

Sort key

> A property to be used to sort the results of the query based on the value of the property for each member of the query result container.

1.2 References

1.2.1 Normative references

[RFC2119]

> S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels*. IETF, March 1997. Best Current Practice. URL: https://www.rfc-editor.org/rfc/rfc2119

[RFC8174]

> B. Leiba. *Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words*. IETF, May 2017. Best Current Practice. URL: https://www.rfc-editor.org/rfc/rfc8174

1.2.2 Informative references

[BNF]

> *Backus–Naur form*. https://en.wikipedia.org. URL: https://en.wikipedia.org/wiki/Backus%E2%80%93Naur_form

[FULLTEXT-SEARCH]

> *Full-text search*. https://en.wikipedia.org. URL: http://en.wikipedia.org/wiki/Full_text_search

[HTTP11]

> R. Fielding, Ed.; J. Reschke, Ed.. *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing*. IETF, June 2014. Proposed Standard. URL: https://httpwg.org/specs/rfc7230.html

[LDP]

Steve Speicher; John Arwe; Ashok Malhotra. *Linked Data Platform 1.0*. W3C, 26 February 2015. W3C Recommendation. URL: https://www.w3.org/TR/ldp/

[OSLC-Core-30-Part1]

Steve Speicher; Jim Amsden. *OSLC Core Version 3.0. Part 1: Overview*. OASIS. URL: https://oslc-op.github.io/oslc-specs/specs/core/oslc-core.html

[OSLC-Discovery-30]

Jim Amsden; Martin Sarabura. *OSLC Core Version 3.0. Part 2: Discovery*. OASIS. Working Draft. URL: https://oslc-op.github.io/oslc-specs/specs/core/discovery.html

[OSLCCM]

Steve Speicher. *Open Services for Lifecycle Collaboration Change Management Specification Version 2.0*. http://open-services.net. Final. URL: http://open-services.net/bin/view/Main/CmSpecificationV2

[OSLCCore2]

S. Speicher; D. Johnson. *OSLC Core 2.0*. http://open-services.net. Finalized. URL: http://open-services.net/bin/view/Main/OslcCoreSpecification

[OSLCQuery2]

Arthur Ryman. *Open Services for Lifecycle Collaboration Core Specification Version 2.0 Query Syntax*. http://open-services.net. Finalized. URL: http://open-services.net/bin/view/Main/OSLCCoreSpecQuery

[OSLCShape]

Arthur Ryman; Jim Amsden. *OSLC Resource Shape 3.0*. OASIS. URL: http://docs.oasis-open.org/oslc-core/oslc-core/v3.0/oslc-core-v3.0-part6-resource-shape.html

[RDF-SPARQL-QUERY]

Eric Prud'hommeaux; Andy Seaborne. *SPARQL Query Language for RDF*. W3C, 15 January 2008. W3C Recommendation. URL: https://www.w3.org/TR/rdf-sparql-query/

[WEBARCH]

Ian Jacobs; Norman Walsh. *Architecture of the World Wide Web, Volume One*. W3C, 15 December 2004. W3C Recommendation. URL: https://www.w3.org/TR/webarch/

[XMLLiteral]

Resource Description Framework (RDF): Concepts and Abstract Syntax. https://www.w3.org. URL: https://www.w3.org/TR/rdf-concepts/#dfn-rdf-XMLLiteral

[xsdDatatypes]

RDF 1.1 Concepts and Abstract Syntax - The XML Schema Built-in Datatypes. https://www.w3.org. URL: https://www.w3.org/TR/rdf11-concepts/#xsd-datatypes

[xsdDateTime]

XML Schema Part 2: Datatypes Second Edition - dateTime. https://www.w3.org. URL: https://www.w3.org/TR/2004/REC-xmlschema-2-20041028/#dt-dateTime

1.3 Namespaces

In addition to the namespace URIs and namespace prefixes `oslc`, `rdf`, `rdfs`, `dcterms`, `ldp` and `foaf` defined in the [OSLC-Core-30-Part1], this specification also uses the following namespace prefix definitions:

- oslc_cm : `http://open-services.net/ns/cm#` [OSLCCM]

1.4 Typographical Conventions and Use of RFC Terms

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Motivation

This section is non-normative.

OSLC servers will often manage large amounts of potentially complex resources. Practical use of this information will require some query capability that minimally supports selection of matching elements, filtering of desired properties and ordering. Ideally OSLC Core would rely on existing standard query services such as [RDF-SPARQL-QUERY]. However, the reality is that many OSLC servers are adapters built on existing data management capabilities that are not built on an RDF foundation and do not provide SPARQL endpoints for query. In order to address the need for standard query capability to enable integration, [OSLC-Core-30-Part1] defines a Query Capability and a Query Syntax that are intended to be simple enough that they can be implemented on a wide range of existing server architectures and persistence technologies. This specification improves on the [OSLCCore2] query capability by:

- Describing how its capabilities can be implemented partially and incrementally to make it easier for server implementations to get started and evolve as needs arise.
- Defining more relaxed semantics for query comparisons and being more agnostic of persistence and query technologies that make it easier for servers to implement and be compliant.
- Providing more complete information and guidance for server implementations.
- Better describing how clients may discover and use query capabilities.
- Including examples of query requests and responses.

3. Discovering a Query Capability

This section is non-normative.

Clients may discover query capabilities as described in [OSLC-Discovery-30] using the `oslc:queryCapability` property of an `oslc:Service`. The relationship between these resources is shown in Figure 1.
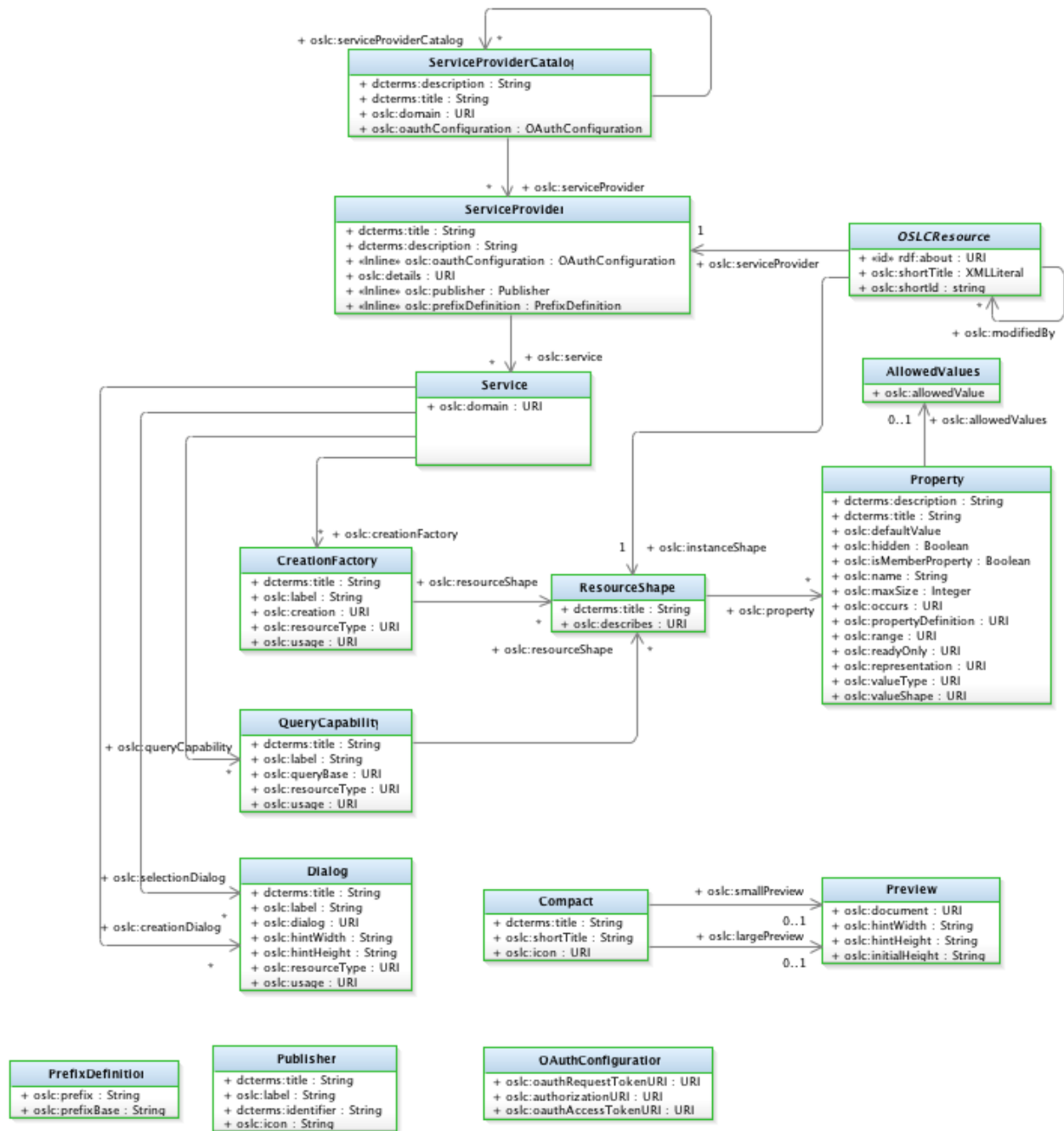
Figure 1 - OSLC discovery resources

A query capability **SHOULD** specify exactly one `oslc:resourceShape` and that shape **SHOULD** describe the query result container. [query-1]

The resource shape referenced by `oslc:resourceShape` **MUST** declare at least one OSLC property with `oslc:isMemberProperty "true"^^xsd:boolean`, and that property **SHOULD** specify exactly one `oslc:valueShape` referencing a resource shape that describes the referenced members of that container. [query-2]

<div style="border-left: 4px solid #c9a227; background:#faf8ee; padding:1em;">

EXAMPLE 1: Query capability resource shape

```
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix oslc: <http://open-services.net/ns/core#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .


<https://example.org/jazz/oslc/context/_Q2fMII8EEd2Q-OW8dr3S5w/trs/shapes/workitem
s/query> a oslc:ResourceShape ;
    oslc:describes <https://example.org/jazz/oslc/contexts/_Q2fMII8EEd2Q-OW8dr3S5w
/workitems> ;
    oslc:property <https://example.org/jazz/oslc/context/_Q2fMII8EEd2Q-OW8dr3S5w/t
rs/shapes/workitems/query/property/member> ;
    dcterms:title "Shape of Work Item query resource"^^rdf:XMLLiteral .


<https://example.org/jazz/oslc/context/_Q2fMII8EEd2Q-OW8dr3S5w/trs/shapes/workitem
s/query/property/member>
    a oslc:Property ;
    oslc:isMemberProperty "true"^^xsd:boolean ;
    oslc:name "member" ;
    oslc:occurs oslc:Zero-or-many ;
    oslc:propertyDefinition rdfs:member ;
    oslc:range <http://open-services.net/ns/cm#ChangeRequest> ;
    oslc:readOnly "true"^^xsd:boolean ;
    oslc:representation oslc:Inline ;
    oslc:valueShape <https://example.org/jazz/oslc/context/_Q2fMII8EEd2Q-OW8dr3S5w
/shapes/workitems/defect> ;
    oslc:valueType oslc:AnyResource ;
    dcterms:title "Work Item Member"^^rdf:XMLLiteral .
```

</div>

The resource shape referenced by `oslc:valueShape` in the query result container's resource shape **SHOULD** declare the properties that can be used in `oslc.select` and optionally `oslc.where`. [query-3]

OSLC properties that have a `oslc:queryable "false"^^xsd:boolean` value cannot be specified in `oslc.where`.

Where a member resource supports nested properties, for example, the `foaf:name` of a `dcterms:creator`, the member resource shape **MAY** provide a `oslc:valueShape` for that property. [query-4]

This allows clients to discover both immediate and nested properties that might be used.

4. Using a Query Capability

Executing a query or search is achieved by performing either of the following:

- a **GET** on the `oslc:queryBase` of a query capability, with zero, one or more query parameters shown in Table 1.
- a **POST** on the `oslc:queryBase` of a query capability with a `application/x-www-form-urlencoded` body containing zero, one or more query parameters shown in Table 1.

Servers **MUST** support a GET on the `oslc:queryBase` of a query capability. [query-5]

Servers **SHOULD** support POST on the `oslc:queryBase` of a query capability to avoid clients having to construct URIs that are too long for a GET. [query-6]

A **GET** or **POST** request on the `oslc:queryBase` of a query capability **MAY** use zero, one or more of the OSLC query parameters listed in Table 1. [query-7]

Clients should use the POST mechanism described above if the URI of a GET would otherwise be too long.

For a successful operation, the response status code **MUST** be `200 OK` for a non-paged result, or for an OSLC paged result. [query-8]

For a successful operation, the response body **MUST** be RDF that includes a query result container whose subject is the `oslc:queryBase`. [query-9]

Table 1 - OSLC Query Parameters

Parameter	Description
`oslc.where`	Limits the query results to members that satisfy the specified query expression.
`oslc.searchTerms`	Score each member resource using a full-text search on its text-valued properties, and sort them in descending order of score.
`oslc.orderBy`	Sort the members using the specified sort keys. If supported, servers **SHOULD** include the `oslc:order` pseudo-property, and for a paged response, order the pages so that the first page contains the lowest ordered members, and the last page contains the highest ordered members. See oslc.orderBy for details.
`oslc.select`	Include the specified immediate and nested properties of the member resources. See oslc.select for details.

`oslc.prefix`	Specify a namespace prefix that may be used in `oslc.where`, `oslc.orderBy`, and `oslc.select`. See `oslc.prefix` in [OSLC-Core-30-Part1] for details.
`oslc.paging`	Specifies whether a paged result is requested. See *Resource paging* in [OSLC-Core-30-Part1] and oslc.paging for details.
`oslc.pageSize`	Specifies a suggested maximum page size for a paged result. See *Resource paging* in [OSLC-Core-30-Part1] and oslc.pageSize for details..

If a GET or POST on a `oslc:queryBase` neither specifies `oslc.where` nor `oslc.searchTerms`, the server **MUST** return a query response describing all the resources that have one or more of the RDF types specified by the `oslc:resourceType` of the `oslc:QueryCapability` that defines the `oslc:queryBase`. [query-10]

5. Query Result Containers

The response body for a successful **GET** or **POST** on a **oslc:queryBase MUST** include a query result container whose subject is the **oslc:queryBase** URI. [query-11]

The container **SHOULD** be a *Linked Data Platform Container* (LDPC) referencing each member resource found by the query. The response **SHOULD** include a **Link** header that describes the type of LDPC returned. This provides compatibility with OSLC Core 3.0 and its wider use of LDPCs. [LDP] [query-12]

If the query capability that declared the base URI does not declare a **oslc:resourceShape** then the container **MUST** include an **rdfs:member** reference to each of the result members. [query-13]

If the query capability that declared the base URI declares a **oslc:resourceShape** and that resource shape defines a container property with **oslc:isMemberProperty "true"^^xsd:boolean** then the query result container **MUST** include the specified member property referencing each of query result member. [query-14]

EXAMPLE 2: Query result container using rdfs:member as the membership property

```
HTTP/1.1 200 OK
Content-Type: text/turtle
ETag: "_87e52ce291112"
Link: <http://www.w3.org/ns/ldp#DirectContainer>; rel="type",
      <http://www.w3.org/ns/ldp#Resource>; rel="type"


@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix ldp: <http://www.w3.org/ns/ldp#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix ex: <http://example.org/>.

<http://example.org/queryBase/>
    a ldp:DirectContainer;
    ldp:membershipResource <http://example.org/queryBase/>;
    ldp:hasMemberRelation rdfs:member;
    ldp:contains ex:member1, ex:member2;
    rdfs:member ex:member1, ex:member2.
```

If the query capability that declared the base URI declares a **oslc:resourceShape** and that resource shape defines a container property with **oslc:isMemberProperty "true"^^xsd:boolean** that is **ldp:contains**, the query result container may use any form of LDPC, such as **ldp:BasicContainer**. This provides the most compact representation of query result members that is compatible with OSLC Query 2.0 and conformant with LDPC [LDP]. All of the examples showing a query response following this section assume that the query capability resource shape defines that **ldp:contains** is

the membership property.

EXAMPLE 3: Query result container using ldp:contains as the membership property

```
HTTP/1.1 200 OK
Content-Type: text/turtle
ETag: "_87e52ce291112"
Link: <http://www.w3.org/ns/ldp#BasicContainer>; rel="type",
      <http://www.w3.org/ns/ldp#Resource>; rel="type"


@prefix dcterms: <http://purl.org/dc/terms/>.
@prefix ldp: <http://www.w3.org/ns/ldp#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix ex: <http://example.org/>.

<http://example.org/queryBase/>
    a ldp:BasicContainer;
    ldp:contains ex:member1, ex:member2.
```

6. Query Result Paging

A query capability **MAY** support *Resource paging* as described in [OSLC-Core-30-Part1]. [query-15]

A paged response **MAY** occur if paging is supported and the request includes a `oslc.paging=true` query parameter, or automatically if the server determines that the total query result set exceeds some implementation dependent threshold. [query-16]

7. Query Parameters

7.1 Introduction

The query parameters defined here are intended to satisfy simple query requirements, and be easy to both use and implement. More complex query requirements should be satisfied using a full query language, e.g. SQL, SPARQL or whatever query language the server and/or its underlying data sources support.

This specification formally defines the syntax of the `oslc.where`, `oslc.searchTerms`, and `oslc.orderBy` parameters using Backus-Naur Form [BNF] and informally illustrates their semantics with simple examples. See oslc.where syntax, oslc.searchTerms syntax, and oslc.orderBy syntax for the formal syntax.

The query parameters **MAY** contain characters that **MUST** be URL encoded when transmitted in an HTTP request. [query-17]

The examples below show both unencoded and encoded query parameters for clarity.

All query parameter names are prefixed with the characters "`oslc.`" The following sections define these query parameters.

An OSLC domain specification **MAY** use some or all of these query parameters, and **SHOULD** use these rather than defining new query parameters that have the same or very similar meanings. Each of these query parameters **SHOULD** appear at most once in a query. The behavior is undefined when a query parameter appears more than once. [query-18]

In the following sections, syntax is formally defined using a common extended form of BNF. Informal definitions and comments are delimited by `/*` and `*/`.

7.2 oslc.where

7.2.1 Introduction

This section is non-normative.

The `oslc.where` query parameter defines a *query expression* that is used to query for resources matching the query expression. It is serves a similar purpose to the `WHERE` clause of a SQL statement.

For example, suppose that a change management system provides a query capability with a `oslc:queryBase` of `https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems`, and a client

wants to query for all change requests created by the user `https://example.org/jts/users/deb`. This can be performed using a `GET` on `https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems` using the query parameters shown in [Table 2](#):

Table 2 - oslc.where parameter

Parameter	Unencoded value
`oslc.where`	`dcterms:creator=<https://example.org/jts/users/deb>`

EXAMPLE 4: Query for all change requests created by Deb

```
GET https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems
    ?oslc.where=dcterms%3Acreator%3D%3Chttps%3A%2F%2Fexample.org%2Fjts%2Fusers%2Fd
eb%3E

OSLC-Core-Version: 2.0
Accept: text/turtle

-- response --
200 OK
Etag: "5f836103-c64e-3c3f-aa5e-dd0baec92511"
OSLC-Core-Version:  2.0
Vary:  Accept, OSLC-Core-Version
Content-Type:  text/turtle;charset=UTF-8
Content-Language:  en-US
Link: <http://www.w3.org/ns/ldp#BasicContainer>; rel="type", <http://www.w3.org/ns
/ldp#Resource>; rel="type"

@prefix dcterms:  <http://purl.org/dc/terms/> .
@prefix ldp:      <http://www.w3.org/ns/ldp#> .
@prefix oslc:     <http://open-services.net/ns/core#> .
@prefix rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems>
    a ldp:BasicContainer ;
    ldp:contains
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
9> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
22> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
11> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
20> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
1> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
27> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
28> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
17> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
5> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
23> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
7> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
12> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
8> .
```

Note that the query result in the above example does not contain any properties of the members found by the query because `oslc.select` was not specified. Clients should specify `oslc.select` if they require a specific set of member properties to be included in the response.

Conditions may use the usual binary comparison operators and be combined using the boolean conjunction operator, `and`.

Suppose a client now wants to query for all change requests created by Deb that are not fixed. This can be performed using a `GET` on `https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems` using the query parameters shown in Table 3:

Table 3 - oslc.where parameter

Parameter	Unencoded value
`oslc.where`	`dcterms:creator=<https://example.org/jts/users/deb> and oslc_cm:fixed=false`

EXAMPLE 5: Query for all change requests created by Deb that are not fixed

```
GET https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems
    ?oslc.where=dcterms%3Acreator%3D%3Chttps%3A%2F%2Fexample.org%2Fjts%2Fusers%2Fd
eb%3E%20and%20oslc_cm%3Afixed%3Dfalse

OSLC-Core-Version: 2.0
Accept: text/turtle

-- response --
200 OK
Etag: "5f836103-c64e-3c3f-aa5e-dd0baec92511"
OSLC-Core-Version:  2.0
Vary:  Accept, OSLC-Core-Version
Content-Type:  text/turtle;charset=UTF-8
Content-Language:  en-US
Link: <http://www.w3.org/ns/ldp#BasicContainer>; rel="type", <http://www.w3.org/ns
/ldp#Resource>; rel="type"

@prefix dcterms:  <http://purl.org/dc/terms/> .
@prefix ldp:      <http://www.w3.org/ns/ldp#> .
@prefix oslc:     <http://open-services.net/ns/core#> .
@prefix rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems>
    a ldp:BasicContainer ;
    ldp:contains
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
22> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
20> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
1> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
27> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
28> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
5> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
23> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
7> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
8> .
```

An `oslc.where` query expression may query for nested properties. The following example show a query for change requests created by any user named "Deb". This can be performed using a `GET` on `https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems` using the query parameters shown in Table 4:

Table 4 - oslc.where parameter

Parameter	Unencoded value
`oslc.where`	`dcterms:creator {foaf:name="Deb"}`

EXAMPLE 6: Query for all change requests created by any user named 'Deb'

```
GET https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems
    ?oslc.where=dcterms%3Acreator%20%7Bfoaf%3Aname%3D%22Deb%22%7D


OSLC-Core-Version: 2.0
Accept: text/turtle


-- response --
200 OK
Etag: "5f836103-c64e-3c3f-aa5e-dd0baec92511"
OSLC-Core-Version:  2.0
Vary:  Accept, OSLC-Core-Version
Content-Type:  text/turtle;charset=UTF-8
Content-Language:  en-US
Link: <http://www.w3.org/ns/ldp#BasicContainer>; rel="type", <http://www.w3.org/ns
/ldp#Resource>; rel="type"


@prefix dcterms:  <http://purl.org/dc/terms/> .
@prefix ldp:      <http://www.w3.org/ns/ldp#> .
@prefix oslc:     <http://open-services.net/ns/core#> .
@prefix rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .


<https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems>
    a ldp:BasicContainer ;
    ldp:contains
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
9> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
22> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
11> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
20> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
1> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
27> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
28> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
17> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
5> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
23> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
7> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
12> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
8> .
```

7.2.2 oslc.where Syntax

```
oslc_where     ::= "oslc.where=" compound_term
compound_term ::= simple_term (space? boolean_op space? simple_term)*
simple_term    ::= term | scoped_term
space          ::= " " /* a space character */
boolean_op     ::= "and"
term           ::= identifier_wc comparison_op value | identifier_wc space in_op s
pace? in_val
scoped_term    ::= identifier_wc "{" compound_term "}"
identifier_wc ::= identifier | wildcard
identifier     ::= PrefixedName
PrefixedName  ::= /* see "SPARQL Query Language for RDF", http://www.w3.org/TR/rd
f-sparql-query/#rPrefixedName */
wildcard       ::= "*"
comparison_op ::= "=" | "!=" | "<" | ">" | "<=" | ">="
in_op          ::= "in"
in_val         ::= "[" value ("," value)* "]"
value          ::= uri_ref_esc | PrefixedName | literal_value
uri_ref_esc    ::= /* an angle bracket-delimited URI reference in which > and \ ar
e \-escaped. */
literal_value ::= boolean | decimal | string_esc (LANGTAG | ("^^" PrefixedName))?
boolean        ::= "true" | "false"
decimal        ::= /* see "XML Schema Part 2: Datatypes Second Edition", http://ww
w.w3.org/TR/xmlschema-2/ */
string_esc     ::= /* a string enclosed in double quotes, with certain characters
escaped. See below. */
LANGTAG        ::= /* see "SPARQL Query Language for RDF", http://www.w3.org/TR/rd
f-sparql-query/#rLANGTAG */
```

wildcard

The "*" `wildcard` matches any property.

Implementations **MAY** support the use of the wildcard in property names and nested properties. [query-19]

boolean_op

The `boolean_op` term represents a boolean operation that lets you combine simple boolean expressions to form a compound boolean expression. The only boolean operation allowed is " and " which represents conjunction. The boolean operator " or " for disjunction is not allowed in the interests of keeping the syntax simple. The effect of " or " in the simple case of testing a property for equality with one of several values can be achieved through the use of the " in " operator. For example, the following query expression might find change requests with severity "high" or "medium":

Table 5 - oslc.where parameter

Parameter	Unencoded value
oslc.where	oslc_cm:severity in ["high","medium"]

EXAMPLE 7

```
GET http://example.org/queryBase?oslc.where=oslc_cm%3Aseverity%20in%20%5B%22hi
gh%22%2C%22medium%22%5D
```

space
The **space** term represents a single space character.

A space character **MAY** be used to delimit the **binary_op** term in the **compound_term** term to improve readability. [query-20]

comparison_op
The **comparison_op** term represents one of the following binary comparison operators:

Table 6 - Binary Comparison Operators

=	test for equality
!=	test for inequality
<	test less-than
>	test greater-than
<=	test less-than or equal
>=	test greater-than or equal

Semantics of datatypes and operations on these datatypes are described in oslc.where Semantics.

in_op
The **in_op** term represents the operator " in " which is a test for equality to any of the values in a list. The list is a comma-separated sequence of values, enclosed in square brackets, whose syntax is defined by the term **in_val**.

value
The **value** term represents either a URI reference (**uri_ref_esc**), a **PrefixedName** or a literal value (**literal_value**). The use of values and comparison operators is described in oslc.where semantics

literal_value
The **literal_value** term represents either a plain literal or a typed literal. A plain literal is a quoted string (**string_esc**), optionally followed by a language tag (**LANGTAG**). For example, **"Bonjour"@fr** is a plain literal with a language tag for French.

A typed literal is formed by suffixing a quoted string (**string_esc**) with "^^" followed by the

prefixed name (`PrefixedName`) of a datatype URI.

If the range of a property includes literal values from more than one datatype, then a typed literal **MUST** be used in order to avoid ambiguity. Otherwise a plain literal **MAY** be used and the service **SHOULD** infer the datatype. [query-21]

The terms `boolean` and `decimal` are short forms for typed literals. For example, `true` is a short form for `"true"^^xsd:boolean`, `42` is a short form for `"42"^^xsd:integer` and `3.14159` is a short form for `"3.14159"^^xsd:decimal`.

`decimal`

The `decimal` term represents a decimal number as defined in XML Schema Part 2: Datatypes Second Edition. As mentioned above, this term is a short form for typed literals whose datatype URIs are either xsd:integer or xsd:decimal. An integer literal value is a special case of a decimal literal value, namely one in which the decimal point is omitted from the lexical representation. For example, 42 is a valid decimal number which happens also to be a valid integer and so it is a short form for the typed literal "42"^^xsd:integer.

`string_esc`

The `string_esc` term represents an arbitrary sequence of characters. The sequence of characters is enclosed in double quote (") characters. Therefore, the double quote character itself **MUST** be escaped. More generally, all occurrences of the double quote character in the string **MUST** be replaced by the sequence \" and all occurrences of the backslash character \ in the actual value **MUST** be replaced by the sequence \\ in the escaped value. This escaping process **MUST** be reversed to give the actual value of the string. [query-22]

`scoped_term`

The `scoped_term` represents the ability to query on nested properties. Servers **MAY** choose not to support such nested property queries, and **SHOULD** respond with `501 Not Implemented` if they are not supported. [query-23]

7.2.3 oslc.where Semantics

The data types supported in OSLC resource shapes are described in *Literal Value Types* in [OSLCShape]. The table below shows the forms of `value` that might be used with a comparison operator, and which comparison operators must or may be supported, and their semantics.

Table 7 - oslc.where Semantics

Value type	Supported value forms	Comparison operators

`rdf:XMLLiteral`	An [XMLLiteral], such as `"abc"^^rdf:XMLLiteral`, **MUST** be supported. A server **MAY** treat the literal in the same way as a plain string literal with the data type removed such as `"abc"`. A plain string literal, such as `"abc"` **MUST** be supported. A server **MAY** treat the literal in the same way as an XML literal string such as `"abc"^^rdf:XMLLiteral`. An xsd:string typed literal [xsdDatatypes], such as `"abc"^^xsd:string`, **MUST** be supported. A server **MAY** treat the literal in the same way as an XML literal string such as `"abc"^^rdf:XMLLiteral`. Servers **MAY** validate the string for compliance to an [XMLLiteral]. [query-24]	The operators `=`, `!=`, and `in` **MUST** be supported. Servers **MUST** implement, and **SHOULD** document, one of the following semantics: • case sensitive string compare • case insensitive string compare • case insensitive string pattern match where `%` means match zero or more characters, and `_` means match any single character. A server **MAY** choose which of the above to apply based on whether the string contains a `%` or _ character, and/or the property being compared. A server **SHOULD** document if this is the case. Other operators **MAY** be supported and **MAY** have implementation-dependent semantics. A server **MAY** report such usage as unsupported. [query-25]

`xsd:boolean`	The values `true` and `false` **MUST** be supported. The `xsd:boolean` [xsdDatatypes] values `"true"^^xsd:boolean` and `"false"^^xsd:boolean` **MUST** be supported and **MUST** have the same meaning as `true` and `false` respectively. The values `"true"` and `"false"` **MAY** be supported, and if so, **MUST** have the same meaning as `true` and `false` respectively. A language tagged literal, such as `"true"@en` **MAY** be supported and **MAY** have implementation and/or locale dependent behavior. [query-26]	The operators `=`, `!=`, and `in` **MUST** be supported. Other operators **MAY** be supported and **MAY** have implementation-dependent semantics. [query-27]
`xsd:dateTime`	Values that are conformant with an `xsd:dateTime` [xsdDatatypes] [xsdDateTime], such as `"2018-01-30T12:25:00"^^xsd:dateTime`, **MUST** be supported. Plain string literals whose string value conforms to an `xsd:dateTime`, such as `"2018-01-30T12:25:00"`, **MAY** be supported, and if so, **MUST** have the same meaning as an `xsd:dateTime` typed literal such as `"2018-01-30T12:25:00"^^xsd:dateTime″`. [query-28]	The operators `=`, `!=`, `<`, `>`, `<=`, `>=`, and `in` **MUST** be supported, and **MUST** have the semantics of a `xsd:dateTime` comparison. [query-29]

`xsd:decimal`	Decimal literals with or without a decimal point, such as `42` and `42.0`, **MUST** be supported if the server supports `xsd:decimal` property values. An `xsd:decimal` typed literal [xsdDatatypes], such as `"42"^^xsd:decimal` or `"42.0"^^xsd:decimal`, **MUST** be supported if the server supports `xsd:decimal` property values. A plain string literal with a decimal value, such as `"42"` or `"42.0"`, **MAY** be supported, and if so, **MUST** have the same meaning as the `xsd:decimal` typed literal. [query-30]	The operators =, !=, <, >, <=, >=, and `in` **MUST** be supported if the server supports `xsd:decimal` property values, and if so, **MUST** have the semantics of a `xsd:decimal` comparison. [query-31]
`xsd:double`	Decimal literals with or without a decimal point, such as `42` and `42.0`, **MUST** be supported if the server supports `xsd:double` property values. An `xsd:double` typed literal [xsdDatatypes], such as `"42"^^xsd:double` or `"42.0"^^xsd:double`, **MUST** be supported if the server supports `xsd:double` property values. A plain string literal with a double value, such as `"42"` or `"42.0"`, **MAY** be supported, and if so, **MUST** have the same meaning as the `xsd:double` typed literal. [query-32]	The operators =, !=, <, >, <=, >=, and `in` **MUST** be supported if the server supports `xsd:double` property values, and if so, **MUST** have the semantics of an `xsd:double` comparison. [query-33]

`xsd:float`	Decimal literals with or without a decimal point, such as `42` and `42.0`, **MUST** be supported if the server supports `xsd:float` property values. An `xsd:float` typed literal [xsdDatatypes], such as `"42"^^xsd:float` or `"42.0"^^xsd:float`, **MUST** be supported if the server supports `xsd:float` property values. A plain string literal with a float value, such as `"42"` or `"42.0"`, **MAY** be supported, and if so, **MUST** have the same meaning as the `xsd:float` typed literal. [query-34]	The operators `=`, `!=`, `<`, `>`, `<=`, `>=`, and `in` **MUST** be supported if the server supports `xsd:float` property values, and if so, **MUST** have the semantics of an `xsd:float` comparison. [query-35]
`xsd:integer`	Decimal literals without a decimal point, such as `42`, **MUST** be supported. An `xsd:integer` typed literal [xsdDatatypes], such as `"42"^^xsd:integer`, **MUST** be supported. An `xsd:decimal` typed literal [xsdDatatypes] without a decimal point, such as `"42"^^xsd:decimal`, **MAY** be supported, and if so, **MUST** have the same meaning as an `xsd:integer` typed literal. A plain string literal with a decimal value, such as `"42"`, or a language tag string with a decimal value, such as `"42"@en`, **MAY** be supported, and if so, **MUST** have the same meaning as a decimal value. [query-36]	The operators `=`, `!=`, `<`, `>`, `<=`, `>=`, and `in` **MUST** be supported and **MUST** have the semantics of an `xsd:integer` comparison. [query-37]

| `xsd:string` | A plain string literal, such as `"abc"` **MUST** be supported.

An xsd:string typed literal [xsdDatatypes], such as `"abc"^^xsd:string`, **MUST** be supported. A server **MUST** treat the literal in the same way as an plain string literal string such as `"abc"`.

An [XMLLiteral], such as `"abc"^^rdf:XMLLiteral`, **MUST** be supported. A server **MAY** treat the literal in the same way as a plain literal string such as `"abc"`.

A string with a language tag, such as `"abc"@en`, **MAY** be supported. The string **MAY** be treated as a plain string literal, or it **MAY** be treated as a language specific string that only matches a string with an identical value and language tag.

[query-38] | The operators `=`, `!=`, and `in` **MUST** be supported.

Servers **MUST** implement, and **SHOULD** document, one of the following semantics:

• case sensitive string compare
• case insensitive string compare
• case insensitive string pattern match where `%` means match zero or more characters, and `_` means match any single character.

A server **MAY** choose which of the above to apply based on whether the string contains a `%` or `_` character, and/or the property being compared. A server **SHOULD** document if this is the case.

Other operators **MAY** be supported and **MAY** have implementation-dependent semantics. A server **MAY** report such usage as unsupported.

[query-39] |

| `oslc:Resource` | A uri_ref_esc such as `<http://example.org>` **MUST** be supported.

A PrefixedName such as `oslc:Zero-or-many`, **MAY** be supported. If supported, the value **MUST** be treated as equivalent to its corresponding uri_ref_esc.

[query-40] | The operators `=`, `!=`, and `in` **MUST** be supported. A server **MUST** implement this as a case-sensitive URI string comparison.

Other operators **MAY** be supported and **MAY** have implementation-dependent semantics, or a server **MAY** report such usage as unsupported.

[query-41] |

If `oslc.where` specifies a property that has a defined namespace prefix but the property is not known, an implementation **MAY** either:

1. Respond with a `400 Bad Request` as described in [Error Handling](#).
2. Execute the query and return a normal query response.

[query-42]

An unknown property is one that is not defined in the resource shape for that type of resource. Responding with an error code is appropriate for implementations that have finite and enumerable properties of resources that cannot determine how to query for undefined properties. Responding with a normal query response is appropriate for implementations that have open sets of properties, not all of which might be in a resource shape.

7.3 oslc.searchTerms

Resource properties often contain text so it is useful to search for resources that contain specified terms. The `oslc.searchTerms` query parameter lets you perform a full-text search [FULLTEXT-SEARCH] on a set of resources. In a full-text search, each resource is matched against the list of search terms and assigned a numeric score. A high score indicates a good match. The matching resources are returned in the response, sorted by the score in descending order. Each resource that is returned in the response is annotated with a special property, `oslc:score`, that gives its match score.

An OSLC domain specification that supports full-text search **SHOULD** specify which resource properties may be supported in a full-text search. [query-43]

When `oslc.searchTerms` is used in the request, each matching resource (hit) in the response **MAY** contain an `oslc:score` property. Note that `oslc:score` is not purely a property of the resource since it also depends on the search terms. It is therefore a pseudo-property whose validity is limited to the

HTTP response, and for a paged response, its subsequent pages. [query-44]

The `oslc:score` property **MUST** be a non-negative number and **SHOULD** be in the range from 0-100. Results **MUST** be ordered with the entry with the largest `oslc:score` occurring first. [query-45]

The `oslc.orderBy` query parameter **MAY** be used with `oslc.searchTerms`. When `oslc.orderBy` is used with `oslc.searchTerms` the result **MUST** be first sorted in descending order by the `oslc:score` pseudo-property, and then by the other sort keys specified in `oslc.orderBy`. This behavior is like prepending the list of sort keys specified in `oslc.orderBy` with the key `-oslc:score`. However, the pseudo-property `oslc:score` **MUST NOT** appear explicitly in `oslc.orderBy`. [query-46]

The `oslc.where` query parameter **MAY** be used with `oslc.searchTerms`. When `oslc.where` is used with `oslc.searchTerms` and is supported, then the set of resources searched for matches **MUST** be restricted to only those resources that satisfy the conditions in `oslc.where`. [query-47]

For example, the following query returns the high severity change requests that deal with database performance:

Table 8 - query parameters

Parameter	Unencoded value
`oslc.where`	`oslc_cm:severity="high"`
`oslc.searchTerms`	`"database","performance"`

EXAMPLE 8

```
GET http://example.org/queryBase?oslc.where=oslc_cm%3Aseverity%3D%22high%22
    &oslc.searchTerms=%22database%22%2C%22performance%22
```

Syntax

```
oslc_searchTerms ::= "oslc.searchTerms=" search_terms
search_terms     ::= string_esc ("," string_esc)*
```

7.4 oslc.orderBy

The `oslc.orderBy` query parameter lets you sort the result set. It is like the `ORDER BY` clause of a SQL statement.

If a server supports the `oslc.orderBy` parameter and this is specified in the request:

- A paged response **SHOULD** reflect the ordering of members specified by `oslc.orderBy`. The first page **SHOULD** contain the members that are ordered first, and any subsequent pages **SHOULD** contain members ordered after the members in the previous page. [query-48]

- The query result **MAY** include a `oslc:order` property whose positive integer value reflects the

<mark>ordering of the the members specified by `oslc.orderBy`. [query-49]</mark>

Note that `oslc:order` is not purely a property of the resource since it also depends on the `oslc.orderBy` value. It is therefore a pseudo-property whose validity is limited to the HTTP response to the query and, for a paged response, its subsequent pages.

<mark>For a paged response, the smallest `oslc:order` value in a page **MUST** be greater than largest `oslc:order` value in any previous page. [query-50]</mark>

<mark>If `oslc.orderBy` is not specified, the ordering of the results is undefined and **MAY** be implementation-dependent. [query-51]</mark>

You can specify a list of one or more immediate or nested properties of the resources in the member list, and a sort direction for each where " + " means ascending order and " -" means descending order. The following example sorts the high severity change requests by the name of the creator, with most recently created first:

Table 9 - query parameters

Parameter	Unencoded value
`oslc.where`	`oslc_cm:severity="high"`
`oslc.orderBy`	`dcterms:creator{+foaf:name},-dcterms:created`

EXAMPLE 9: Sort high severity change requests created by creator name, and created date-time

```
GET http://example.org/queryBase
    ?oslc.orderBy=dcterms%3Acreator%7B%2Bfoaf%3Aname%7D%2C-dcterms%3Acreated
    &oslc.where=oslc_cm%3Aseverity%3D%22high%22
```

Syntax

```
oslc_orderBy        ::= "oslc.orderBy=" sort_terms
sort_terms          ::= sort_term ("," sort_term)*
sort_term           ::= scoped_sort_terms | ("+" | "-") identifier
scoped_sort_terms   ::= identifier "{" sort_terms "}"
```

7.5 oslc.select

The `oslc.select` query parameter lets you specify which immediate and nested properties of the resources in the member list to be included in the response. It is like the **SELECT** clause of a SQL statement.

<mark>If `oslc.select` is not specified, the response **MAY** include some or none of the properties of members in the query result container. [query-52]</mark>

Servers **SHOULD** avoid returning a large number of properties of members by default. [query-53]

If a server chooses to include a multi-valued property by default, and the query response is not paged, it **MUST** include all the values of that property. [query-54]

Clients should specify `oslc.select` if they require particular properties to be returned in the query response.

If `oslc.where` specifies a property that has a defined namespace prefix but the property is not known, an implementation **MAY** either:

1. Respond with a `400 Bad Request` as described in Error Handling.
2. Execute the query and return a normal query response.

[query-55]

Servers **SHOULD** accept the value `rdf:nil`. If specified as the only property, servers **SHOULD** omit all properties of the member resources. If `oslc.select` specifies `rdf:nil` and other properties, servers **MAY** choose to respond with `400 Bad Request`, or ignore the `rdf:nil`, or have other implementation dependent behavior. [query-56]

The syntax of the `oslc.select` query parameter is the same as that of the `oslc.properties` query parameter. See [OSLC-Core-30-Part1] for the syntax.

The difference between `oslc.select` and `oslc.properties` is that:

- `oslc.select` specifies properties against the subject URI of each resource found by the query (the members of the query result container).
- `oslc.properties` specifies properties of the resource whose subject is the request URI. For a query result, this is the query result container, not the resources returned by the query. This query parameter is therefore of limited use for query capabilities.

As in the case of `oslc.properties`, the wildcard is equivalent to the list of all properties of the member resources.

Implementations **MAY** support the use of the wildcard for immediate and/or nested properties of member resources. Servers **SHOULD** document any restriction on the use of the wildcard. [query-57]

A server **MAY** include additional member properties in the response to those specified in the `oslc.select` query parameter [query-58]. Clients should not assume that only the specified properties will be included.

For example, the following query finds the change requests created by any user named "Deb" and includes the `dcterms:title` and `dcterms:creator` for each change request found by the query and the `foaf:name` of the person that last modified the change request.

Table 10 - query parameters

Parameter	Unencoded value
oslc.where	dcterms:creator {foaf:name="Deb"}
oslc.select	dcterms:title,dcterms:creator,oslc:modifiedBy{foaf:name}

EXAMPLE 10: Query for all change requests created by any user named 'Deb'

```
GET https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems
    ?oslc.where=dcterms%3Acreator%20%7Bfoaf%3Aname%3D%22Deb%22%7D
    &oslc.select=dcterms%3Atitle%2Cdcterms%3Acreator%2Coslc%3AmodifiedBy%3Amodifie
r%7Bfoaf%3Aname%7D

OSLC-Core-Version: 2.0
Accept: text/turtle

 -- response --
200 OK
Etag:  "5f836103-c64e-3c3f-aa5e-dd0baec92511"
OSLC-Core-Version:  2.0
Vary:  Accept, OSLC-Core-Version
Content-Type:  text/turtle;charset=UTF-8
Content-Language:  en-US
Link: <http://www.w3.org/ns/ldp#BasicContainer>; rel="type", <http://www.w3.org/ns
/ldp#Resource>; rel="type"

@prefix dcterms:  <http://purl.org/dc/terms/> .
@prefix ldp:      <http://www.w3.org/ns/ldp#> .
@prefix oslc:     <http://open-services.net/ns/core#> .
@prefix rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<https://example.org/ccm/oslc/contexts/_by884MNWEeekg_dNxwf1pg/workitems>
    a ldp:BasicContainer ;
    ldp:contains
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
9> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
22> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
11> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
20> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
1> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
27> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
28> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
17> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
5> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
23> ,
```

```
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
7> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
12> ,
        <https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/
8> .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/9>
        a       oslc_cm:ChangeRequest ;
        dcterms:creator <https://example.org/jts/users/deb> ;
        oslc:modifiedBy <https://example.org/jts/users/deb> ;
        dcterms:title "To many messages logged in the console"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/22>
        a       oslc_cm:ChangeRequest ;
        dcterms:creator <https://example.org/jts/users/deb> ;
        oslc:modifiedBy <https://example.org/jts/users/bob> ;
        dcterms:title "Calculation error"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/11>
        a       oslc_cm:ChangeRequest ;
        dcterms:creator <https://example.org/jts/users/deb> ;
        oslc:modifiedBy <https://example.org/jts/users/deb> ;
        dcterms:title "Some accessibility issues"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/20>
        a       oslc_cm:ChangeRequest ;
        dcterms:creator <https://example.org/jts/users/deb> ;
        oslc:modifiedBy <https://example.org/jts/users/bob> ;
        dcterms:title "Browser Exception"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/1>
        a       oslc_cm:ChangeRequest ;
        dcterms:creator <https://example.org/jts/users/deb> ;
        oslc:modifiedBy <https://example.org/jts/users/deb> ;
        dcterms:title "Not possible to change a user password"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/27>
        a       oslc_cm:ChangeRequest ;
        dcterms:creator <https://example.org/jts/users/deb> ;
        oslc:modifiedBy <https://example.org/jts/users/deb> ;
        dcterms:title "Improve link colors"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/28>
        a       oslc_cm:ChangeRequest ;
        dcterms:creator <https://example.org/jts/users/deb> ;
        oslc:modifiedBy <https://example.org/jts/users/deb> ;
        dcterms:title "Login not working anymore"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/17>
        a       oslc_cm:ChangeRequest ;
        dcterms:creator <https://example.org/jts/users/deb> ;
        oslc:modifiedBy <https://example.org/jts/users/deb> ;
        dcterms:title "Increase size of overall application window"^^rdf:XMLLiteral
.
```

```
<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/5>
      a       oslc_cm:ChangeRequest ;
      dcterms:creator <https://example.org/jts/users/deb> ;
      dcterms:title "Improve loan calculation algorithm"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/23>
      a       oslc_cm:ChangeRequest ;
      dcterms:creator <https://example.org/jts/users/deb> ;
      oslc:modifiedBy <https://example.org/jts/users/deb> ;
      dcterms:title "Search is not finding this term"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/12>
      a       oslc_cm:ChangeRequest ;
      dcterms:creator <https://example.org/jts/users/deb> ;
      dcterms:title "Button sizes are too small"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/7>
      a       oslc_cm:ChangeRequest ;
      dcterms:creator <https://example.org/jts/users/deb> ;
      oslc:modifiedBy <https://example.org/jts/users/deb> ;
      dcterms:title "Offer more services related to loans"^^rdf:XMLLiteral .

<https://example.org/ccm/resource/itemName/com.ibm.team.workitem.WorkItem/8>
      a       oslc_cm:ChangeRequest ;
      dcterms:creator <https://example.org/jts/users/deb> ;
      oslc:modifiedBy <https://example.org/jts/users/bob> ;
      dcterms:title "Add context sensitive help support everywhere"^^rdf:XMLLitera
l .

<https://example.org/jts/users/deb>
      foaf:name "Deb" .

<https://example.org/jts/users/bob>
      foaf:name "Bob" .
```

7.6 oslc.paging

Servers **MAY** support a `oslc.paging` query parameter. Implementations **MAY** choose to ignore the parameter. [query-59]

Servers that support paging of query results **SHOULD** use a paged response when `oslc.paging=true` is specified. [query-60]

Servers **MAY** also provide a paged response if the `oslc.paging` parameter is not specified and the query result includes a large number of members or a large number of RDF statements. [query-61]

EXAMPLE 11: Query for all change requests with paging

```
GET https://example.org/jazz/oslc/contexts/_Q2fMII8EEd2Q-OW8dr3S5w/workitems&oslc.
paging=true
Accept: text/turtle
```

```
OSLC-Core-Version: 2.0

 -- response --
200 OK
Date:  Fri, 11 May 2018 09:49:35 GMT
X-Powered-By:  Servlet/3.0
Etag:  "ca3acc7e-36b4-3c35-a51c-08d29266c24c"
Expires:  Fri, 11 May 2018 09:49:35 GMT
OSLC-Core-Version:  2.0
Vary:  Accept,OSLC-Core-Version
Content-Type:  text/turtle;charset=UTF-8
Content-Language:  en-US
Cache-Control:  private, must-revalidate, max-age=0, no-cache=set-cookie
X-Cluster-SA:  Node-Jazz
Set-Cookie:  JSA_AUTH_COMPLETE=true; Path=/; Secure
X-jazzweb3:  D=466712 t=1526032175078031
Transfer-Encoding:  chunked
Strict-Transport-Security:  max-age=31536000; includeSubDomains

@prefix dcterms:  <http://purl.org/dc/terms/> .
@prefix rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix oslc:     <http://open-services.net/ns/core#> .

<https://example.org/jazz/oslc/contexts/_Q2fMII8EEd2Q-OW8dr3S5w/workitems>
    a ldp:BasicContainer ;
ldp:contains
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/126
544> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/163
518> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/993
98> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/157
559> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/128
024> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/126
992> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/906
41> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/163
178> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/276
059> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/954
69> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/217
716> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/161
823> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/423
17> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/705
84> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/126
```

```
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/120
434> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/104
751> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/139
381> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/263
005> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/105
938> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/133
616> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/204
463> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/250
934> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/661
80> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/529
81> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/697
89> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/112
342> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/210
019> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/785
77> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/100
271> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/109
336> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/720
37> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/128
115> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/716
59> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/104
241> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/751
03> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/182
157> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/156
808> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/108
460> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/472
83> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/285
162> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/135
240> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/206
800> ,
```

```
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/117
595> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/593
62> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/135
620> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/222
107> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/133
136> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/801
40> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/251
947> ,
    <https://example.org/jazz/resource/itemName/com.ibm.team.workitem.WorkItem/744
96> .


<https://example.org/jazz/oslc/contexts/_Q2fMII8EEd2Q-OW8dr3S5w/workitems?_resultT
oken=_nIokIFUAEeibptZOC98gng&oslc.pageSize=50&_startIndex=0>
    a       oslc:ResponseInfo ;
    oslc:nextPage <https://example.org/jazz/oslc/contexts/_Q2fMII8EEd2Q-OW8dr3S5w/
workitems?_resultToken=_nIokIFUAEeibptZOC98gng&oslc.pageSize=50&_startIndex=50> ;
    oslc:totalCount "82991" ;
    dcterms:title "Work Items" .
```

7.7 oslc.pageSize

Servers **MAY** support a `oslc.pageSize` query parameter. Implementations **MAY** choose to ignore the parameter, or **MAY** use it as a hint for how a response should be paged and use page boundaries that do not exactly match the specified page size. [query-62]

The page size is expressed in terms of the number of RDF statements that are included in the response body.

8. Error Handling

If a `GET` or `POST` of a `oslc:queryBase` cannot be executed, the response **MUST** include a response body whose RDF contains at least one `oslc:Error` describing the cause of the error. [query-63]

If the error is due to use of `oslc.where`, `oslc.searchTerms`, `oslc.orderBy`, or `oslc.select` and that query parameter is not supported, the response status **SHOULD** be `501 Not Implemented`. Implementations that do not support a specific query parameter need not parse or validate its value. [query-64]

If the error is due to a supported but malformed query parameter value, such as one that does not match the supported syntax, then the response status code **SHOULD** be `400 Bad Request`. [query-65]

If the error is due to a supported `oslc.where`, `oslc.select`, or `oslc.orderBy` parameter, and its value references a namespace that is not defined by default or with `oslc.prefix`, the response status code **SHOULD** be `400 Bad Request`. [query-66]

If `oslc.where` specifies a property that has a defined namespace prefix, and the property is defined in the resource shape for the query capability with `oslc:queryable` of `"false"^^xsd:boolean`, the server **MUST** Respond with a `400 Bad Request`. [query-67]

If a query uses a supported query parameter, and the parameter value is syntactically valid, but the implementation does not support a specific use of such a query parameter, the response status **SHOULD** be `501 Not Implemented`. For example, this might apply if a specific use of the wildcard in `oslc.select` is not supported. [query-68]

If the query is valid, but the server failed to execute the request in some unexpected way, then the response status **SHOULD** be `500 Internal Server Error`. [query-69]

9. Version Compatibility

This specification follows the specification version guidelines given in [OSLC-Core-30-Part1].

10. Conformance

Implementations of this specification need to satisfy the following conformance clauses.

Clause Number	Requirement
query-1	A query capability **SHOULD** specify exactly one `oslc:resourceShape` and that shape **SHOULD** describe the query result container.
query-2	The resource shape referenced by `oslc:resourceShape` **MUST** declare at least one OSLC property with `oslc:isMemberProperty "true"^^xsd:boolean`, and that property **SHOULD** specify exactly one `oslc:valueShape` referencing a resource shape that describes the referenced members of that container.
query-3	The resource shape referenced by `oslc:valueShape` in the query result container's resource shape **SHOULD** declare the properties that can be used in `oslc.select` and optionally `oslc.where`.
query-4	Where a member resource supports nested properties, for example, the `foaf:name` of a `dcterms:creator`, the member resource shape **MAY** provide a `oslc:valueShape` for that property.
query-5	Servers **MUST** support a GET on the `oslc:queryBase` of a query capability.
query-6	Servers **SHOULD** support POST on the `oslc:queryBase` of a query capability to avoid clients having to construct URIs that are too long for a GET.
query-7	A `GET` or `POST` request on the `oslc:queryBase` of a query capability **MAY** use zero, one or more of the OSLC query parameters listed in Table 1.
query-8	For a successful operation, the response status code **MUST** be `200 OK` for a non-paged result, or for an OSLC paged result.
query-9	For a successful operation, the response body **MUST** be RDF that includes a query result container whose subject is the `oslc:queryBase`.
query-10	If a GET or POST on a `oslc:queryBase` neither specifies `oslc.where` nor `oslc.searchTerms`, the server **MUST** return a query response describing all the resources that have one or more of the RDF types specified by the `oslc:resourceType` of the `oslc:QueryCapability` that defines the `oslc:queryBase`.
query-11	The response body for a successful `GET` or `POST` on a `oslc:queryBase` **MUST** include a query result container whose subject is the `oslc:queryBase` URI.
query-12	The container **SHOULD** be a *Linked Data Platform Container* (LDPC) referencing each member resource found by the query. The response **SHOULD** include a `Link` header that describes the type of LDPC returned. This provides compatibility with OSLC Core 3.0 and its wider use of LDPCs. [LDP]

Clause Number	Requirement
query-13	If the query capability that declared the base URI does not declare a `oslc:resourceShape` then the container **MUST** include an `rdfs:member` reference to each of the result members.
query-14	If the query capability that declared the base URI declares a `oslc:resourceShape` and that resource shape defines a container property with `oslc:isMemberProperty "true"^^xsd:boolean` then the query result container **MUST** include the specified member property referencing each of query result member.
query-15	A query capability **MAY** support *Resource paging* as described in [OSLC-Core-30-Part1].
query-16	A paged response **MAY** occur if paging is supported and the request includes a `oslc.paging=true` query parameter, or automatically if the server determines that the total query result set exceeds some implementation dependent threshold.
query-17	The query parameters **MAY** contain characters that **MUST** be URL encoded when transmitted in an HTTP request.
query-18	An OSLC domain specification **MAY** use some or all of these query parameters, and **SHOULD** use these rather than defining new query parameters that have the same or very similar meanings. Each of these query parameters **SHOULD** appear at most once in a query. The behavior is undefined when a query parameter appears more than once.
query-19	Implementations **MAY** support the use of the wildcard in property names and nested properties.
query-20	A space character **MAY** be used to delimit the `binary_op` term in the `compound_term` term to improve readability.
query-21	If the range of a property includes literal values from more than one datatype, then a typed literal **MUST** be used in order to avoid ambiguity. Otherwise a plain literal **MAY** be used and the service **SHOULD** infer the datatype.
query-22	The `string_esc` term represents an arbitrary sequence of characters. The sequence of characters is enclosed in double quote (") characters. Therefore, the double quote character itself **MUST** be escaped. More generally, all occurrences of the double quote character in the string **MUST** be replaced by the sequence \" and all occurrences of the backslash character \ in the actual value **MUST** be replaced by the sequence \\ in the escaped value. This escaping process **MUST** be reversed to give the actual value of the string.
query-23	The `scoped_term` represents the ability to query on nested properties. Servers **MAY** choose not to support such nested property queries, and **SHOULD** respond with `501 Not Implemented` if they are not supported.

Clause Number	Requirement
query-24	An [XMLLiteral], such as `"abc"^^rdf:XMLLiteral`, **MUST** be supported. A server **MAY** treat the literal in the same way as a plain string literal with the data type removed such as `"abc"`. A plain string literal, such as `"abc"` **MUST** be supported. A server **MAY** treat the literal in the same way as an XML literal string such as `"abc"^^rdf:XMLLiteral`. An xsd:string typed literal [xsdDatatypes], such as `"abc"^^xsd:string`, **MUST** be supported. A server **MAY** treat the literal in the same way as an XML literal string such as `"abc"^^rdf:XMLLiteral`. Servers **MAY** validate the string for compliance to an [XMLLiteral].
query-26	The values `true` and `false` **MUST** be supported. The `xsd:boolean` [xsdDatatypes] values `"true"^^xsd:boolean` and `"false"^^xsd:boolean` **MUST** be supported and **MUST** have the same meaning as `true` and `false` respectively. The values `"true"` and `"false"` **MAY** be supported, and if so, **MUST** have the same meaning as `true` and `false` respectively. A language tagged literal, such as `"true"@en` **MAY** be supported and **MAY** have implementation and/or locale dependent behavior.
query-27	The operators `=`, `!=`, and `in` **MUST** be supported. Other operators **MAY** be supported and **MAY** have implementation-dependent semantics.
query-28	Values that are conformant with an `xsd:dateTime` [xsdDatatypes] [xsdDateTime], such as `"2018-01-30T12:25:00"^^xsd:dateTime`, **MUST** be supported. Plain string literals whose string value conforms to an `xsd:dateTime`, such as `"2018-01-30T12:25:00"`, **MAY** be supported, and if so, **MUST** have the same meaning as an `xsd:dateTime` typed literal such as `"2018-01-30T12:25:00"^^xsd:dateTime"`.

Clause Number	Requirement
query-29	The operators `=`, `!=`, `<`, `>`, `<=`, `>=`, and `in` **MUST** be supported, and **MUST** have the semantics of a `xsd:dateTime` comparison.
query-30	Decimal literals with or without a decimal point, such as `42` and `42.0`, **MUST** be supported if the server supports `xsd:decimal` property values. An `xsd:decimal` typed literal [xsdDatatypes], such as `"42"^^xsd:decimal` or `"42.0"^^xsd:decimal`, **MUST** be supported if the server supports `xsd:decimal` property values. A plain string literal with a decimal value, such as `"42"` or `"42.0"`, **MAY** be supported, and if so, **MUST** have the same meaning as the `xsd:decimal` typed literal.
query-31	The operators `=`, `!=`, `<`, `>`, `<=`, `>=`, and `in` **MUST** be supported if the server supports `xsd:decimal` property values, and if so, **MUST** have the semantics of a `xsd:decimal` comparison.
query-32	Decimal literals with or without a decimal point, such as `42` and `42.0`, **MUST** be supported if the server supports `xsd:double` property values. An `xsd:double` typed literal [xsdDatatypes], such as `"42"^^xsd:double` or `"42.0"^^xsd:double`, **MUST** be supported if the server supports `xsd:double` property values. A plain string literal with a double value, such as `"42"` or `"42.0"`, **MAY** be supported, and if so, **MUST** have the same meaning as the `xsd:double` typed literal.
query-33	The operators `=`, `!=`, `<`, `>`, `<=`, `>=`, and `in` **MUST** be supported if the server supports `xsd:double` property values, and if so, **MUST** have the semantics of an `xsd:double` comparison.

Clause Number	Requirement
query-34	Decimal literals with or without a decimal point, such as `42` and `42.0`, **MUST** be supported if the server supports `xsd:float` property values. An `xsd:float` typed literal [xsdDatatypes], such as `"42"^^xsd:float` or `"42.0"^^xsd:float`, **MUST** be supported if the server supports `xsd:float` property values. A plain string literal with a float value, such as `"42"` or `"42.0"`, **MAY** be supported, and if so, **MUST** have the same meaning as the `xsd:float` typed literal.
query-35	The operators `=`, `!=`, `<`, `>`, `<=`, `>=`, and `in` **MUST** be supported if the server supports `xsd:float` property values, and if so, **MUST** have the semantics of an `xsd:float` comparison.
query-36	Decimal literals without a decimal point, such as `42`, **MUST** be supported. An `xsd:integer` typed literal [xsdDatatypes], such as `"42"^^xsd:integer`, **MUST** be supported. An `xsd:decimal` typed literal [xsdDatatypes] without a decimal point, such as `"42"^^xsd:decimal`, **MAY** be supported, and if so, **MUST** have the same meaning as an `xsd:integer` typed literal. A plain string literal with a decimal value, such as `"42"`, or a language tag string with a decimal value, such as `"42"@en`, **MAY** be supported, and if so, **MUST** have the same meaning as a decimal value.
query-37	The operators `=`, `!=`, `<`, `>`, `<=`, `>=`, and `in` **MUST** be supported and **MUST** have the semantics of an `xsd:integer` comparison.

Clause Number	Requirement
query-38	A plain string literal, such as **"abc"** **MUST** be supported. An xsd:string typed literal [xsdDatatypes], such as **"abc"^^xsd:string**, **MUST** be supported. A server **MUST** treat the literal in the same way as an plain string literal string such as **"abc"**. An [XMLLiteral], such as **"abc"^^rdf:XMLLiteral**, **MUST** be supported. A server **MAY** treat the literal in the same way as a plain literal string such as **"abc"**. A string with a language tag, such as **"abc"@en**, **MAY** be supported. The string **MAY** be treated as a plain string literal, or it **MAY** be treated as a language specific string that only matches a string with an identical value and language tag.
query-39	The operators **=**, **!=**, and **in** **MUST** be supported. Servers **MUST** implement, and **SHOULD** document, one of the following semantics: • case sensitive string compare • case insensitive string compare • case insensitive string pattern match where **%** means match zero or more characters, and _ means match any single character. A server **MAY** choose which of the above to apply based on whether the string contains a **%** or _ character, and/or the property being compared. A server **SHOULD** document if this is the case. Other operators **MAY** be supported and **MAY** have implementation-dependent semantics. A server **MAY** report such usage as unsupported.
query-40	A uri_ref_esc such as **<http://example.org>** **MUST** be supported. A PrefixedName such as **oslc:Zero-or-many**, **MAY** be supported. If supported, the value **MUST** be treated as equivalent to its corresponding uri_ref_esc.

Clause Number	Requirement
query-41	The operators `=`, `!=`, and `in` **MUST** be supported. A server **MUST** implement this as a case-sensitive URI string comparison. Other operators **MAY** be supported and **MAY** have implementation-dependent semantics, or a server **MAY** report such usage as unsupported.
query-43	An OSLC domain specification that supports full-text search **SHOULD** specify which resource properties may be supported in a full-text search.
query-44	When `oslc.searchTerms` is used in the request, each matching resource (hit) in the response **MAY** contain an `oslc:score` property. Note that `oslc:score` is not purely a property of the resource since it also depends on the search terms. It is therefore a pseudo-property whose validity is limited to the HTTP response, and for a paged response, its subsequent pages.
query-45	The `oslc:score` property **MUST** be a non-negative number and **SHOULD** be in the range from 0-100. Results **MUST** be ordered with the entry with the largest `oslc:score` occurring first.
query-46	The `oslc.orderBy` query parameter **MAY** be used with `oslc.searchTerms`. When `oslc.orderBy` is used with `oslc.searchTerms` the result **MUST** be first sorted in descending order by the `oslc:score` pseudo-property, and then by the other sort keys specified in `oslc.orderBy`. This behavior is like prepending the list of sort keys specified in `oslc.orderBy` with the key `-oslc:score`. However, the pseudo-property `oslc:score` **MUST NOT** appear explicitly in `oslc.orderBy`.
query-47	The `oslc.where` query parameter **MAY** be used with `oslc.searchTerms`. When `oslc.where` is used with `oslc.searchTerms` and is supported, then the set of resources searched for matches **MUST** be restricted to only those resources that satisfy the conditions in `oslc.where`.
query-48	A paged response **SHOULD** reflect the ordering of members specified by `oslc.orderBy`. The first page **SHOULD** contain the members that are ordered first, and any subsequent pages **SHOULD** contain members ordered after the members in the previous page.
query-49	The query result **MAY** include a `oslc:order` property whose positive integer value reflects the ordering of the the members specified by `oslc.orderBy`.
query-50	For a paged response, the smallest `oslc:order` value in a page **MUST** be greater than largest `oslc:order` value in any previous page.
query-51	If `oslc.orderBy` is not specified, the ordering of the results is undefined and **MAY** be implementation-dependent.
query-52	If `oslc.select` is not specified, the response **MAY** include some or none of the properties of members in the query result container.

Clause Number	Requirement
query-53	Servers **SHOULD** avoid returning a large number of properties of members by default.
query-54	If a server chooses to include a multi-valued property by default, and the query response is not paged, it **MUST** include all the values of that property.
query-55	If `oslc.where` specifies a property that has a defined namespace prefix but the property is not known, an implementation **MAY** either: 1. Respond with a `400 Bad Request` as described in Error Handling. 2. Execute the query and return a normal query response.
query-56	Servers **SHOULD** accept the value `rdf:nil`. If specified as the only property, servers **SHOULD** omit all properties of the member resources. If `oslc.select` specifies `rdf:nil` and other properties, servers **MAY** choose to respond with `400 Bad Request`, or ignore the `rdf:nil`, or have other implementation dependent behavior.
query-57	Implementations **MAY** support the use of the wildcard for immediate and/or nested properties of member resources. Servers **SHOULD** document any restriction on the use of the wildcard.
query-58	A server **MAY** include additional member properties in the response to those specified in the `oslc.select` query parameter
query-59	Servers **MAY** support a `oslc.paging` query parameter. Implementations **MAY** choose to ignore the parameter.
query-60	Servers that support paging of query results **SHOULD** use a paged response when `oslc.paging=true` is specified.
query-61	Servers **MAY** also provide a paged response if the `oslc.paging` parameter is not specified and the query result includes a large number of members or a large number of RDF statements.
query-62	Servers **MAY** support a `oslc.pageSize` query parameter. Implementations **MAY** choose to ignore the parameter, or **MAY** use it as a hint for how a response should be paged and use page boundaries that do not exactly match the specified page size.
query-63	If a `GET` or `POST` of a `oslc:queryBase` cannot be executed, the response **MUST** include a response body whose RDF contains at least one `oslc:Error` describing the cause of the error.
query-64	If the error is due to use of `oslc.where`, `oslc.searchTerms`, `oslc.orderBy`, or `oslc.select` and that query parameter is not supported, the response status **SHOULD** be `501 Not Implemented`. Implementations that do not support a specific query parameter need not parse or validate its value.

Clause Number	Requirement
query-65	If the error is due to a supported but malformed query parameter value, such as one that does not match the supported syntax, then the response status code **SHOULD** be `400 Bad Request`.
query-66	If the error is due to a supported `oslc.where`, `oslc.select`, or `oslc.orderBy` parameter, and its value references a namespace that is not defined by default or with `oslc.prefix`, the response status code **SHOULD** be `400 Bad Request`.
query-67	If `oslc.where` specifies a property that has a defined namespace prefix, and the property is defined in the resource shape for the query capability with `oslc:queryable` of `"false"^^xsd:boolean`, the server **MUST** Respond with a `400 Bad Request`.
query-68	If a query uses a supported query parameter, and the parameter value is syntactically valid, but the implementation does not support a specific use of such a query parameter, the response status **SHOULD** be `501 Not Implemented`. For example, this might apply if a specific use of the wildcard in `oslc.select` is not supported.
query-69	If the query is valid, but the server failed to execute the request in some unexpected way, then the response status **SHOULD** be `500 Internal Server Error`.

Appendix A. Relationship with OSLC Query 2.0

This section is non-normative.

This appendix summarizes OSLC Query 3.0 changes from, and compatibility with OSLC Query 2.0, [OSLCQuery2].

Topic	Details
Query result container	Query result containers are compatible with OSLC 2.0 clients. OSLC 2.0 clients should ignore the additional LDPC triples and response headers. The default `rdfs:member` predicate when no resource shape is specified ensures that OSLC 2.0 clients can consume the query response from a Query 3.0 server.
Query comparison semantics	The OSLC Query 2.0 specification required strict adherence to SPARQL semantics for relational operators and property comparisons. Most implementations based on a standard relational database could only satisfy that requirement by maintaining RDF datasets and using SPARQL on them. Few existing implementations did this because it was overly burdensome, resulting in most implementations being non-compliant. In practice, few clients could assume complete compliance. While it is possible that an OSLC 2.0 client might get results from a OSLC Query 3.0 server that are are not compliant with Query 2.0, it is expected that in the majority of cases this will be inconsequential.
`oslc.where` syntax	The only change of `oslc.where` syntax between OSLC Query 3.0 and OSLC Query 2.0 is the addition of `PrefixedName` in the `value` element. The ability to use a prefixed name in a value is useful, especially for enumerated values, and its omission was an oversight in the OSLC Query 2.0 specification.
wildcard in `oslc.select` syntax	The use of a wildcard in `oslc.select` has been made optional.
`oslc.orderBy`	The semantics of `oslc.orderBy` has been clarified. A new optional `oslc:order` pseudo-property has been defined to allow clients to determining the ordering of result members within a non-paged or paged response.
Error handling	The error handling in Query 2.0 was not explicitly described. Query 3.0 clarifies expected and required error handling for consistency and to make it easier for clients to consume.

Discovery of selectable or queryable properties	OSLC Query 2.0 described that a client could discover the resource shape of the returned members but did not define whether those properties were queryable or just selectable with `oslc.select`. OSLC Core 3.0 introduced a new `oslc:queryable` property for an OSLC property in an OSLC resource shape. This allows OSLC 3.0 clients to differentiate which properties are supported in a query expression specified by `oslc.where` or are supported in a `oslc.select` value.

Appendix B. Acknowledgements

This section is non-normative.

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Participants:

> James Amsden, IBM (Editor)
> Geoff Clemm, IBM
> Nick Crossley, IBM (Chair)
> Ian Green, IBM
> Peter Hack, IBM
> David Honey, IBM
> Sam Padget, IBM
> Martin Pain, IBM
> Martin Sarabura, PTC
> Brian Steele, IBM

Appendix C. Change History

This section is non-normative.

Revision	Date	Editor	Changes Made
01	2018-12-14	Jim Amsden	Committee Specification Public Review Draft revision 01 published.