भारतीय मानक **IS 18003 (Part 3/Sec 1) : 2024**

Indian Standard

यूनिफाइड डाटा एक्सचेंज

भाग 3 एपीआई टेस्ट सूट

अनुभाग 1 संसाधन पहुँच सेवा

Unified Data Exchange

Part 3 API Test Suites

Section 1 Resource Access Service

ICS 33.020, 35.020

भारतीय मानक ब्यूरो
BUREAU OF INDIAN STANDARDS
मानक भवन, 9 बहादुर शाह ज़फर मार्ग, नई दिल्ली - 110002
MANAK BHAVAN, 9 BAHADUR SHAH ZAFAR MARG
NEW DELHI - 110002
www.bis.gov.in www.standardsbis.in

May 2024 **Price Group 14**

Smart Infrastructure Sectional Committee, LITD 28

FOREWORD

This Indian Standard (Part 3/Sec 1) was adopted by the Bureau of Indian Standards, after the draft is finalized by the Smart Infrastructure Sectional Committee, and approved by the Electronics and Information Technology Division Council.

This standard is one of the series of Indian Standards on unified data exchange. Other parts in this series are:

Part 1 Architecture
Part 2 API specifications

The composition of the Committee, responsible for the formulation of this standard is given in Annex D.

For the purpose of deciding whether a particular requirement of this standard is complied with, the final value, observed or calculated, expressing the result of a test or analysis, shall be rounded off in accordance with IS 2 : 2022 'Rules for rounding off numerical values (*second revision*)'. The number of significant places retained in the rounded off value should be the same as that of the specified value in this standard.

CONTENTS

LIST OF TABLES

0 INTRODUCTION

The smart cities are generating an enormous amount of data. If harnessed in the right way, this data can empower the stakeholders namely, the providers, the consumers and the governing agencies in solving the key challenges faced by the cities and add value by building innovative applications. One issue faced by the current smart cities is the inability to exchange data efficiently due to the proprietary and ad-hoc nature of the interfaces and their implementations.

To address the data exchange bottlenecks, a unified data exchange (DX) layer, which provides a standardised framework for accessing data in a unified format and allowing authorized data sharing amongst different entities, was defined in the Indian Standard IS 18003 (Part 1) Unified data exchange: Part 1 Architecture and the Indian Standard IS 18003 (Part 2) Unified data exchange: Part 2 API specifications.

The data exchange (DX) layer specifies three sets of services, namely the catalogue service, the authorization service and the resource access service. The detailed application programming interface (API) specifications for each of these services are provided in — IS 18003 (Part 2) : API specifications. In particular, the resource access service forms the data plane for the DX layer. It defines interfaces to allow data consumers to access data for a given resource as per the consent of the data provider. Further, it defines interfaces to allow data providers to publish data for a given resource.

This standard IS 18003 (Part 3) Compliance specifications: Part 1 (Resource access service) specifies the abstract test suite to define compliance to the resource access service as defined in IS 18003 (Part 2) API specifications. The test suite defines the minimum functionality required for a given functional profile for any compliant resource access service implementation.

The compliance specifications are divided into 5 clauses. Clause 1 gives the scope of this compliance specification document. Clause 2 lists the normative and informative references. Clause 3 gives the definitions on various terminologies and abbreviations used in this document. Clause 4 details the resource access service functional profiles and the details of the tests are presented in 5.

Indian Standard

UNIFIED DATA EXCHANGE

PART 3 API TEST SUITES

SECTION 1 RESOURCE ACCESS SERVICE

1 SCOPE

This standard (Part 3/Sec 1) defines the compliance test suites for any implementation of the data exchange (DX) resource server data access service as specified in IS 18003 (Part 2) of the standard.

Compliance specifications for catalogue service and authorization service will be specified in the future versions of this standard.

The target audience for this standard are the developers of DX data access services and the developers of testing and compliance suites belonging to independent testing and certification agencies. This standard will also be helpful for DX consumers to understand the implementation details of the APIs and the functional profiles.

2 REFERENCES

The standards given below contain provisions which, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of these standards.

2.1 Normative References

The following referenced documents are necessary for the application of the present document:

IS No./ Other Standards	*Title*
IS 18002 (Part 1) : 2021	Unified digital infrastructure — Data layer: Part 1 Reference architecture
IS 18003 (Part 1) : 2020	Unified data exchange: Part 1 Architecture
JSON-LD 1.1	A JSON-based serialization for linked data W3C recommendation July 2020
IETF RFC 2818	HTTP over TLS
IETF RFC 3986	Uniform resource identifier (URI): Generic syntax
IETF RFC 5246	The transport layer security (TLS) Protocol version 1.2
IETF RFC 7231	Hypertext transfer protocol (HTTP/1.1): Semantics and content
IETF RFC 7232	Hypertext transfer protocol (HTTP/1.1): Conditional requests
IETF RFC 7807	Problem details for HTTP APIs
IETF RFC 7946	The GeoJSON format
IETF RFC 8141	Uniform resource names (URNs);
IETF RFC 8259	The JavaScript Object notation (JSON) data interchange format
ISO 8601: 2004	Data elements and interchange formats — Information interchange — Representation of dates and times
Open geospatial consortium Inc. OGC 06-103r4	OpenGIS® implementation standard for geographic information — Simple feature access — Part 1: Common architecture

2.2 Informative References

The following referenced documents are necessary for the application of the present document:

Other Standards	*Title*
Advanced message queuing protocol (AMQP) : v0.9.1ETSI GS CIM 009 V1.4.1 (2021-02)	Context information management (CIM); NGSI-LD API;
ETSI GS CIM 009 V1.4.1 (2021-02)	Context Information Management (CIM); NGSI-LD API
IETF RFC 3987	Internationalized resource identifiers (IRIs)

Other Standards	*Title*
IETF RFC 6749	The OAuth 2.0 authorization framework
IETF RFC 6749	The OAuth 2.0 authorization framework: The bearer token usage
IETF RFC 7396	JSON merge patch
IETF RFC 7519	JSON web token (JWT)
IETF RFC 7522	Security assertion markup language (SAML) 2.0 profile for OAuth 2.0 client authentication and authorization grants
JSON schema	JavaScript Object Notation schema
MQTT 5.0,	OASIS Standard
Open ID connect core 1.0	OpenID connect core 1.0 incorporating errata set 2

3 TERMINOLOGY AND ABBREVIATIONS

3.1 Terminology — For the purpose of this standard, the definitions given in Table 1 shall apply.

Table 1 Definition of Key Terms Used in this Document

(*Clause* 3.1)

Sl No.	Term	Definition
(1)	(2)	(3)
i)	Provider	Legal Entity — Human (possibly delegated by an organisation), organisation or an organisational role that has responsibility to provide authorisation to use resources.
ii)	Consumer	Legal Entity — Human or organisation or an organisational role that consumes a resource via a web or mobile app.
iii)	Data exchange (DX)	Service — Hosts and manages meta-data about data resources, manages authorization for accessing the resources and provides data access for the available data resources.
iv)	DX catalogue service	Service — Provides services to manage meta-information about data resources and provides search functionalities to discover data resources hosted with the data exchange. A software entity providing this service will be referred to as catalogue server.
v)	DX resource access service	Service — Serves resources to authorised apps/consumers. A software entity providing this service will be referred to as resource server.
	DX authorization service	Service — Provides authorization to access data for data resources in accordance to the access policies set for the resources. A software entity providing this service will be referred to as authorization server.
vi)	Authorization token	A digital entity that is used to present the authorization credentials to the resource server.
vii)	Authentication token	A digital entity used to prove the identity of a user to the DX authorization service.
viii)	Catalogue item	An entry in the DX catalogue that describes the meta-information associated with DX entities, such as a data resource, a group of data resources, a DX provider etc. Information contained in a catalogue item depends upon the type of the item.
ix)	App	Application — Software (like a mobile app, web app, device app or server app), that uses resources to provide a service or experience to the consumer.
x)	Provider app	Application — An App that enables a Provider to manage the meta-data and access control in the data exchange, for the resources they are responsible for.

3.2 Abbreviations

List of abbreviations used in this document

Sl No.	*Abbreviation*	*Definition*
(1)	(2)	(3)
i)	DX	Data exchange
ii)	JSON	Java script object notation
iii)	API	Application programming interface
iv)	RS	Resource server
v)	CS	Catalogue server
vi)	AS	Authorization server
vii)	TLS	Transport level security
viii)	CRUD	Create, Read, Update, Delete API operations
ix)	JSON-LD	JavaScript Object Notation for Linked Data
x)	JWT	Json web token
xi)	URN	Uniform resource name
xii)	URL	Uniform resource locator
xiii)	IRI	Internationalized resource identifier
xiv)	AMQP	Advanced message queuing protocol
xv)	MQTT	Message queuing telemetry transport
xvi)	UUID	Universally unique identifier
xvii)	XML	extensible markup language

4 RESOURCE ACCESS SERVICE FUNCTIONAL PROFILES

Within the scope of this standard, a functional profile defines a logical grouping of functionalities provided by the resource access service. Different functional profiles are mutually exclusive in terms of functionality coverage and together all the profiles cover the full functionality defined for the resource access service as defined in IS 18003 (Part 2) of the standard. This specification provides abstract test suites to define compliance for a given functional profile. Thus, the functional profile defines the smallest resource access service functionality grouping for which compliance shall be provided.

The resource access service provides the temporal, spatial, attribute, complex and latest data search functionalities with which users can query a data resource in a DX system. The details of the resource access service functionality are discussed in **6.1.2** of the IS 18003 (Part 2) of the standard.

The query semantics and parameters for geo-spatial, temporal, attribute, complex searches, response filtering, query pagination and counting the number of results are described in detail in **6.1.4** of the of IS 18003 (Part 2) of the standard.

The common object definitions used within request/response bodies for the DX resource access service are described in Table 45 to Table 51 of **6.1.5** of the IS 18003 (Part 2) of the standard. Finally, the common response templates/schemas for all the DX Interface API response payloads are described in detail in the **8.1** IS 18003 (Part 2) of the standard.

4.1 Functional Profile

The functional profile provides the access possibilities for the resource access service also known as resource server for temporal search, spatial search, attribute search, subscription, complex search and response filtering.

Based on the requirements from a minimum viable product (MVP) point of view, we have classified the test cases into two groups. Group 0 defines the basic requirements and group 1 defines the advanced requirements. Table 2 and Table 3 give the functionality provided by Group 0, and Group 1 respectively.

Group 0 is the standard features and Group 1 is the advanced features that needs to be adhered to by the implementer. It is to be noted that the implementer can choose to implement either Group 0 or Group 1 or both.

Table 2 Functional Profile Group 0

(*Clauses* 4.1, 5.2, 5.3, 5.4, 5.5 *and* 5.6)

Sl No.	Group	Functionality
(1)	(2)	(3)
i)	G0 (Temporal)	Temporal searches using between, before and after time relations
ii)	G0 (Spatial)	Spatial search using BBox geometry
iii)	G0 (Attribute)	Attribute queries, using mathematical comparators

Table 3 Functional Profile Group 1

(*Clauses* 4.1, 5.1, 5.7, 5.8, 5.9, 5.10 *and* 5.14)

Sl No.	Group	Functionality
(1)	(2)	(3)
i)	G1 (Temporal)	Temporal latest data search
ii)	G1 (Spatial)	Spatial search using circle, line string and polygon geometry

Complex search supports searches using combinations of temporal, spatial and attribute queries from G0 and G1. Though the specification provides a sample data dump as mentioned in Annex A, response filtering and data response validation depends on the data used in the database. Hence, functional profile test cases for response filtering and data response are left out of scope of this standard, but shall be validated by the testing authorities. For each functional profile an abstract suite of test cases has been specified in **5**. To be conformant to a given functional profile a resource server implementation has to pass all the test cases specified for that particular functional profile.

Further, a given resource server implementation can comply with multiple functional profiles.

It is to be noted that in a complex search, the parameters of complex search are derived using individual parameters from multiple test cases. For example, a complex search '043a&017a' (**5.16.1**) represents a single test case. The parameters of this test shall be derived from the parameters of test 043a (**5.9.1**) and 017a (**5.5.1**). An example is given in Annex B. Also note that this is a separate test case and even if an implementation passes 043a and 017a individually it may not necessarily support functionality required by this complex search test case.

5 TEST CASES FOR FUNCTIONAL PROFILES

Every test case shall be tested against DX resources (‘id’) with access policy ‘private’ and 'public’ to validate secure resource access and open resource access respectively. The test also verifies the interactions of the implementation with other DX services.

As mentioned earlier, in addition to the above scenarios, for every successful (2XX) and unsuccessful (4XX) test cases, tests shall be conducted to understand the data response and response URNs which shall be based on the dataset used in the database and the URNs defined in the implementation.

All the functional profile test numbers (001a, 002a…..0054c) represented in the following tables are associated with a specific test case as defined in the postman collection detailed in Annex C.

5.1 2XX Test Cases for G0 (Temporal) Queries

The tests under this section are associated with the G0 (Temporal) functional profile as defined in Table 3. These tests summaries the HTTP response code 2XX for the between/before/after temporal relations for ‘search’ and ‘count’ query implementation. The query semantics of the temporal search are described in **6.1.4.2** and parameters with status codes are described in Table 33, under **6.1.3.2** of (Part 2) respectively.

5.1.1 *Test* 001a, 003a, 005a, 007a, 009a, 011a

Sl No.	*Functional Profile*	G0 (*Temporal*)		
(1)	(2)	(3)		
i)	Purpose	To test the correctness of search/count query implementation for between/before/after temporal relations.		
ii)	Description	This test validates the search/count query implementation for between/before/after temporal relations. The ‘search’ query response will contain the result of a query within the temporal limits. The ‘count’ query will return the number of records found as a result of query execution for the temporal query.		
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response.		
iv)	Temporal relation	between	before	after
v)	Parameters for ‘search’ query	Parameter ‘timerel’ shall be set equal to ‘between’. Parameters ‘time’ and ‘end time’ have to be specified. The duration is expressed using the ISO 8601 standard.	Parameter ‘timerel’ shall be set equal to ‘before’. Parameter ‘time’ has to be specified. The duration is expressed using the ISO 8601 standard.	Parameter ‘timerel’ shall be set equal to ‘after’. Parameter ‘time’ has to be specified. The duration is expressed using the ISO 8601 standard.
vi)	Test no. for ‘search’ query	001a	003a	005a
vii)	Response code	200	200	200
viii)	Parameters for ‘count’ query	Parameter ‘options’ should be set to ‘count’. Rest of the parameters shall remain same as that for the ’search’ query	Parameter ‘options’ should be set to ‘count’. Rest of the parameters shall remain same as that for the ‘search’ query	Parameter ‘options’ should be set to ‘count’. Rest of the parameters shall remain same as that for the ‘search’ query
ix)	Test no. for ‘count’ query	007a	009a	011a
x)	Response code	200	200	200

5.2 4XX Test Cases for G0 (Temporal) Query

The tests under this section are associated with the G0 (Temporal) functional profile as defined in Table 2. These tests summarise the HTTP response code 4XX for the between/before/after temporal relations for 'search' and 'count' query implementation. The query semantics of the temporal search are described in **6.1.4.2** and parameters with status codes are described in Table 33, under **6.1.3.2** of IS 18003 (Part 2), respectively.

5.2.1 *Test* 002a, 004a, 006a, 008a, 010a, 012a

Sl No.	*Functional Profile*	G0 (*Temporal*)		
(1)	(2)	(3)		
i)	Purpose	To test the correctness of search/count query implementation for 'invalid parameter(s)' passed for between/before/after temporal relation.		
ii)	Description	This test validates the correctness of search/count query implementation for 'invalid parameter(s)' passed for between/before/after temporal relation.		
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response. Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes.		
iv)	Temporal relation	between	before	after
v)	Test no. for 'search' query	002a	004a	006a
vi)	Response code	400	400	400
vii)	Test no. for 'count' query	008a	010a	012a
viii)	Response code	400	400	400

5.2.2 *Test* 002b, 004b, 006b, 008b, 010b, 012b

Sl No.	*Functional Profile*	G0 (*Temporal*)		
(1)	(2)	(3)		
i)	Purpose	To test the correctness of search/count query implementation for 'nonexisting resource id' passed for between/before/after temporal relation.		
ii)	Description	This test validates the correctness of search/count functionality for 'nonexisting resource id' passed for between/before/after temporal relation.		
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response. Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes.		
iv)	Temporal relation	between	before	after
v)	Test no. for 'search' query	002b	004b	006b
vi)	Response code	404	404	404
vii)	Test no. for 'count' query	008b	010b	012b
viii)	Response code	404	404	404

5.2.3 *Test* 002c, 004c, 006c, 008c, 010c, 012c

Sl No.	*Functional Profile*	G0 (*Temporal*)		
(1)	(2)	(3)		
i)	Purpose	To test the correctness of search/count query implementation for 'invalid authorization token' for between/before/after temporal relation.		
ii)	Description	This test validates the correctness of search/count query implementation for 'invalid authorization token' for between/before/after temporal relation. It describes a test when the authentication has failed because the user has provided no-token, expired token or wrong token.		
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response. Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes		
iv)	Temporal relation	between	before	after
v)	Test no. for 'search' query	002c	004c	006c
vi)	Response code	401	401	401
vii)	Test no. for 'count' search query	008c	010c	012c
viii)	Response code	401	401	401

5.3 2XX Test Cases for G0 (Spatial) Query

The tests under this section are associated with the G0 (Spatial) functional profile as defined in Table 2. These tests summarise the HTTP response code 2XX for the BBox spatial geometry for 'search' and 'count' query implementation. The query semantics of the spatial search are described in under **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** in the IS 18003 (Part 2) : API specifications, respectively.

5.3.1 *Test* 013a, 015a

Sl No.	*Functional Profile*	G0 (*Spatial*)
(1)	(2)	(3)
i)	Purpose	To test the correctness of search/count query implementation for a geo-spatial query using BBox geometry
ii)	Description	This test validates the 'search' query implementation for a geo-spatial query using BBox geometry. The response will contain the results according to the spatial query.
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response.
iv)	Parameters	The parameter "georel" should be defined as 'within'. The parameter "geometry" should be defined as 'bbox'. For bbox, the two coordinate points represent the top-left and bottom-right vertices of the bounding box. The parameter "coordinates" should be defined as [[longitude-1,latitude-1], [longitude-2,latitude-2]]. Parameter 'options' should be set to 'count' for count query
v)	Spatial geometry	BBox
vi)	Test no. for 'search' query	013a
vii)	Response code	200
viii)	Test no. for 'count' query	015a
ix)	Response code	200

5.4 4XX Test Cases for G0 (Spatial) Query

The tests under this section are associated with the G0 (Spatial) functional profile as defined in Table 2. These tests summarise the HTTP response code 4XX for the BBox spatial geometry for 'search' and 'count' query implementation. The query semantics of the spatial search are described in under **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** in the IS 18003 (Part 2): API Specifications, respectively.

5.4.1 *Test* 014a, 016a

Sl No.	*Functional Profile*	G0 (*Spatial*)
(1)	(2)	(3)
i)	Purpose	To test the correctness of 'invalid parameter(s)' passed for a geo-spatial query using BBox geometry
ii)	Description	This test validates the correctness of 'invalid parameter(s)' passed for a geo-spatial query using BBox geometry
iii)	Methodology	Execute an API call that contains invalid parameter type for a spatial geo query (BBox). Verify the response code. Optionally, one can also verify that the response contains appropriate URN codes
iv)	Spatial geometry	BBox
v)	Test no for 'search' query	014a
vi)	Response code	400
vii)	Test no. for 'count' query	016a
viii)	Response code	400

5.4.2 *Test* 014b, 016b

Sl No.	*Functional Profile*	G0 (*Spatial*)
(1)	(2)	(3)
i)	Purpose	To test the correctness of an 'invalid authorisation token' for a geospatial query using BBox geometry
ii)	Description	This test validates the correctness for an invalid 'authorisation token'' for a geo-spatial query using BBox geometry. It describes a test when the authentication has failed because the user has provided no-token, expired token or wrong token.
iii)	Methodology	Execute an API call with parameters that contain an 'invalid authorisation token'. Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes
iv)	Spatial geometry	BBox
v)	Test no. for 'search' query	014b
vi)	Response code	401

Sl No.	*Functional Profile*	G0 (*Spatial*)
(1)	(2)	(3)
vii)	Test no. for 'count' query	016b
viii)	Response code	401
ix)	Functional profile	G0 (Spatial)
x)	Purpose	To test the correctness of search/count query implementation for attribute values as per the specified comparison operator

5.4.3 *Test* 014c, 016c

Sl No.	*Functional Profile*	G0 (*Spatial*)
(1)	(2)	(3)
i)	Purpose	To test the correctness of a query for a 'non-existing resource id' for a geo-spatial query using BBox geometry
ii)	Description"	This test validates the correctness of a 'non-existing id' for a geo-spatial query using BBox geometry. The requested resource could not be found because of a 'non-existing resource id'.
iii)	Methodology	Execute an API call with parameters with a 'non-existing resource id'.Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes
iv)	Spatial geometry	BBox
v)	Test no. for 'search' query	014c
vi)	Response code	404
vii)	Test no. for 'search' query	016c
viii)	Response code	404

5.5 2XX Test Cases for G0 (Attribute) Query

The tests under this section are associated with the G0 (Attribute) functional profile as defined in Table 2. These tests summarises the HTTP response code 2XX for resources using an operator which performs a specific mathematical, relational or logical operation for 'search' and 'count' query implementation. The query semantics of the attribute search are described in **6.1.4.3** and parameters with status codes are described in Table 31, under **6.1.3.1** of IS 18003 (Part 2) respectively.

5.5.1 *Test* 017a, 019a, 021a, 023a, 025a, 027a, 029a, 031a, 033a, 025a, 037a, 039a

Sl No.	*Description*	*This test Validates the Search/Count Query Implementation for Attribute Values as per the Specified Mathematical Operator*
(1)	(2)	(3)
i)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response. The appropriate operator should be used for the q parameter value. Parameter 'options' should be set to 'count' for count query

ii)	Operator	Lesser than (<)	Greater than (>)	Equal (==)	Not equal (!=)	Greater than equal (>=)	Lesser than equal (<=)
iii)	Test no. for ‘search’ query	017a	019a	021a	023a	025a	027a
iv)	Response code	200	200	200	200	200	200
v)	Test no. for ‘count’ query	029a	031a	033a	035a	037a	039a
vi)	Response code	200	200	200	200	200	200

5.6 4XX Test Cases for the G0 (Attribute) Query

The tests under this section are associated with the G0 (Attribute) functional profile as defined in Table 2. These tests summarises the HTTP response code 4XX for resources using an operator which performs a specific mathematical, relational or logical operation for ‘search’ and ‘count’ query implementation. The query semantics of the attribute search are described in **6.1.4.3** of Part 2 of the standard and parameters with status codes are described in Table 31, under **6.1.3.1** of Part 2 of the standard respectively.

5.6.1 *Test* 018a, 020a, 022a, 024a, 026a, 028a, 030a, 032a, 034a, 036a, 038a, 040a

Sl No.	*Functional Profile*	G0 (*Attribute*)					
(1)	(2)	(3)					
i)	Purpose	To test the correctness of search/count query implementation for an ‘invalid parameter(s)’ for attribute values as per the specified mathematical operator					
ii)	Description	This test validates the ‘invalid parameter(s)’ functionality for different mathematical operators					
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response. The appropriate comparison operator should be used for the q parameter value. Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes					
iv)	Operator	Lesser than (<)	Greater than (>)	Equal (==)	Not Equal (!=)	Greater than equal (>=)	Lesser than equal (<=)
v)	Test no. for ‘search’ query	018a	020a	022a	024a	026a	028a
vi)	Response code	400	400	400	400	400	400
vii)	Test no. for ‘count’ query	030a	032a	034a	036a	038a	040a
viii)	Response code	400	400	400	400	400	400

5.6.2 *Test* 018b, 020b, 022b, 024b, 026b, 028b, 030b, 032b, 034b, 036b, 038b, 040b

Sl No.	*Functional Profile*	G0 (*attribute*)
(1)	(2)	(3)
i)	Purpose	To test the correctness of search/count query implementation for a ‘non-existing resource id’ attribute value as per the specified mathematical operator.
ii)	Description	This test validates the ‘Empty response’ functionality for a ‘non-existing resource id’ attribute value as per the specified mathematical operator. The requested resource could to be found but may be available in future.
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response. The appropriate operator should be used for the q parameter value. Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes.

Sl No.	*Functional Profile*	G0 (*attribute*)					
(1)	(2)	(3)					
iv)	Operator	Lesser than (<)	Greater than (>)	Equal (==)	Not equal (!=)	Greater than equal (>=)	Lesser than equal (<=)
v)	Test no. for 'search' query	018b	020b	022b	024b	026b	028b
vi)	Response code	404	404	404	404	404	404
vii)	Test no. for 'count' query	030b	032b	034b	036b	038b	040b
(viii)	Response code	404	404	404	404	404	404

5.6.3 *Test* 018c, 020c, 022c, 024c, 026c, 028c, 030c, 032c, 034c, 036c, 038c, 040c

Sl No.	*Functional Profile*	G0 (*Attribute*)					
(1)	(2)	(3)					
i)	Purpose	To test the correctness of search/count for an 'invalid authorization token' for Attribute values as per the specified mathematical operator.					
ii)	Description	This test validates the correctness of search/count for an 'invalid authorization token' for Attribute values as per the specified mathematical operator. It describes a test when the authentication has failed because the user has provided no-token, expired token or wrong token.					
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response. The appropriate operator should be used for the q parameter value. Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes					
iv)	Operator	Lesser than (<)	Greater than (>)	Equal (==)	Not Equal (!=)	Greater than equal (>=)	Lesser than equal (<=)
v)	Test no for 'search' query	018c	020c	022c	024c	026c	028c
vi)	Response code	401	401	401	401	401	401
vii)	Test no. for 'count' query	030c	032c	034c	036c	038c	040c
viii)	Response code	401	401	401	401	401	401

5.7 2XX Test Cases for G1 (Temporal) Profile

The tests under this section are associated with the G1 (Temporal) functional profile as defined in Table 3. These tests summarise the HTTP response code 2XX for resources for the latest data for a 'search' query implementation. The parameters with status codes are described in Table 29, under **6.1.3.1** of Part 2 of the standard respectively.

5.7.1 *Test* 041a

Sl No.	*Functional Profile*	G1 (*Temporal*)
(1)	(2)	(3)
i)	Purpose	To test the correctness of search query implementation for latest data
ii)	Description	This test validates the correctness of search query implementation for latest data
iii)	Methodology	Execute an API call that shall return a successful and verifiable response. Verify that output data is as per the test design.
iv)	Test no.	041a
v)	Response code	200

5.8 4XX Test Cases for G1 (Temporal) Profile

The tests under this section are associated with the G1 (Temporal) functional profile as defined in Table 3. These tests summarise the HTTP response code 4XX for resources for the latest data for a 'search' query implementation. The parameters with status codes are described in Table 29, under **6.1.3.1** of Part 2 of the standard respectively.

5.8.1 *Test* 042a

Sl No.	*Functional Profile*	G1 (*Temporal*)
(1)	(2)	(3)
i)	Purpose	To test the correctness of a 'non-existing resource id' for a latest data search
ii)	Description	This test validates the correctness of a 'non-existing resource id' for a latest data search
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response. Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes.
iv)	Test no.	042a
v)	Response code	404

5.8.2 *Test* 042b

Sl No.	*Functional Profile*	G1 (*Temporal*)
(1)	(2)	(3)
i)	Purpose	To test the correctness of an 'invalid authorization token' for a latest data search
ii)	Description	The test validates the correctness of an 'invalid authorization token' for a latest data search. It describes a test when the authentication has failed because the user has provided no-token, expired token or wrong token.
iii)	Methodology	Execute an API call that shall return a successful and verifiable response. Verify that output data is as per the test design. Optionally, one can also verify that the response contains appropriate URN codes.
iv)	Test no.	042b
v)	Response code	401

5.9 2XX Test Cases for G1 (Spatial) Query

The tests under this section are associated with the G1 (Spatial) functional profile as defined in Table 3. These tests summarise the HTTP response code 2XX for the circle/polygon/linestring spatial geometry for 'search' and 'count' query implementation. The query semantics of the spatial search are described in **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively.

5.9.1 *Test* 043a, 045a, 047a, 049a, 051a, 053a

Sl No.	*Functional Profile*	G1 (*Spatial*)		
(1)	(2)	(3)		
i)	Purpose	To test the correctness of search/count query implementation for a geospatial query using for circle/polygon/linestring geometry.		
ii)	Description	This test validates the search/count query implementation for a geo-spatial query using circle/polygon/linestring geometry. The 'search' query response will contain the result of a query within the spatial limits. The 'count' query will return the number of records found as a result of query execution for the spatial query.		
iii)	Methodology	Execute an API call with parameters that shall return a successful and verifiable response. Verify that output data is as per the test design.		
iv)	Geometry	Circle	Polygon	Linestring
v)	Parameters for 'search'	The parameter 'georel' shall be set equal to 'near;maxdistance= Radius'. The data needs to be searched within this radius. The geometry should be set to 'Point' and coordinates of the centre of the circle should be specified.	The 'georel' parameter should be set to 'within'. The coordinates of the Polygon need to be set like [longitude1, latitude1], [longitude-2,latitude-2],...., [longitude-n, latituden]. Note that the first coordinate should be the same as the last coordinate.	The parameter georel should be sed to 'intersects'. The coordinates of the polygon need to be set like [longitude1, latitude-1], [longitude-2, latitude2],...., [longituden, latitude-n]] for the linestring.
vi)	Test no. for 'search'	043a	045a	047a
vii)	Response code	200	200	200
viii)	Geometry	Circle	Polygon	Linestring
ix)	Parameters for 'count' search	The parameter 'options' should be set to count. Rest of the parameters shall remain same as that for the 'search' query	The parameter options' should be set to count. Rest of the parameters shall remain same as that for the search' query	The parameter 'options' should be set to count. Rest of the parameters shall remain same as that for the 'search' query
x)	Test no for 'count' search	049a	051a	053a
xi)	Response code	200	200	200

5.10 4XX Test Cases for G1 (Spatial) Query

The tests under this section are associated with the G1 (Spatial) functional profile as defined in Table 3. These tests summarise the HTTP response code 4XX for the circle/polygon/linestring spatial geometry for 'search' and 'count' query implementation. The query semantics of the spatial search are described in **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively.

5.10.1 *Test* 044a, 046a, 048a, 050a, 052a, 054a

Sl No.	*Functional Profile*	G1 (*Spatial*)		
(1)	(2)	(3)		
i)	Purpose	To test the correctness of search/count functionality for a 'invalid parameter(s)' for circle/polygon/linestring spatial query.		
ii)	Description	This test validates the search/count functionality for 'invalid parameter(s)' for circle/polygon/linestring temporal query.		
iii)	Methodology	Execute an API call that contains an 'invalid parameter(s)' key for the spatial geo query.		
iv)	Geometry	Circle	Polygon	Linestring
v)	Test no. for 'search'	044a	046a	048a
vi)	Response code	400	400	400
vii)	Test no. for 'count' search	050a	052a	054a
viii)	Response code	400	400	400

5.10.2 *Test* 044b, 046b, 048b, Test 050b, Test 052b, Test 054b

Sl No.	*Functional Profile*	G1 (*Spatial*)		
(1)	(2)	(3)		
i)	Purpose	To test the correctness of search/count functionality for 'invalid authorization token' for a geo-spatial query using circle/polygon/linestring geometry.		
ii)	Description	This test validates the search/count functionality for 'invalid authorization token' for a geo-spatial query using circle/polygon/linestring geometry. It describes a test when the authentication has failed because the user has provided no-token, expired token or wrong token.		
iii)	Methodology	Execute an API call that contains an 'invalid authorization token' for the spatial geo query.		
iv)	Geometry	Circle	Polygon	Linestring
v)	Test no. for 'search'	044b	046b	048b
vi)	Response code	401	401	401
vii)	Test no. for 'count' search	050b	052b	054b
viii)	Response code	401	401	401

5.10.3 *Test* 044c, 046c, 048c, 050c, 052c, 054c

Sl No.	*Functional Profile*	G1 (*Spatial*)		
(1)	(2)	(3)		
i)	Purpose	To test the correctness of search/count functionality for 'non-existing resource id' for a geo-spatial query using for circle/polygon/linestring geometry.		
ii)	Description	This test validates the search/count functionality for 'non-existing resource id' for a geo-spatial query using for circle/polygon/linestring geometry.		
iii)	Methodology	Execute an API call that contains a 'non-existing resource id' for the spatial geo query.		
iv)	Geometry	Circle	Polygon	Linestring
v)	Test no. for 'search'	044c	046c	048c
vi)	Response code	404	404	404
vii)	Test no. for 'count' search	050c	052c	054c
viii)	Response code	404	404	404

5.11 Tests for G2 Complex : G0 (Temporal) and G0 (Spatial - Box)

The combination of tests under this section are associated with the G2 complex functional profile. These tests summarise the HTTP response codes 2XX and 4XX for the G0 (Temporal) query along with G0 (Spatial) query. The query semantics of the temporal search are described in **6.1.4.2** and parameters with status codes are described in Table 33, under **6.1.3.2** in the IS 18003 (Part 2): API specifications, respectively. The query semantics of the spatial search are described in **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively.

5.11.1 G0 *Temporal* (*Between*) *with* G0 *Spatial* (*BBox*)

	Temporal	Between
	Spatial	BBox
Response code	Search type	Test
200	Search	'001a&013a'
	Count	'007a&015a'
	Search	'002a&014a'
400	Count	'008a&016a'
404	Search	'002b&014c'
	Count	'008b&016c'
401	Search	'002c&014b'
	Count	'008c&016b'

5.11.2 G0 *Temporal* (*Before*) with G0 Spatial (BBox)

	Temporal	Before
	Spatial	BBox
Response code	Search type	Test
200	Search	'003a&013a'
	Count	'009a&015a'
400	Search	'004a&014a'
	Count	'010a&016a'
404	Search	'004b&014c'
	Count	'010b&016c'
401	Search	'004c&014b
	Count	'010c&016b'

5.11.3 G0 *Temporal* (*After*) *with* G0 *Spatial*

	Temporal	After
	Spatial	BBox
Response code	Search type	Test
200	Search	'005a&013a'
	Count	'011a&015a'
400	Search	'006a&014a'
	Count	'012a&016a'
404	Search	'006b&014c'
	Count	'012b&016c'
401	Search	'006c&014b'
	Count	'012c&016b'

5.12 Tests for G2 Complex: G0 (Temporal) and G0 (Attribute)

The combination of tests under this section are associated with the G2 Complex functional profile. These tests summarise the HTTP response code 2XX and 4XX for the G0 (Temporal) query along with G0 (Attribute) query. The query semantics of the temporal search are described in **6.1.4.2** and parameters with status codes are described in Table 33, under **6.1.3.2** of (Part 2) respectively. The query semantics of the attribute search are described in **6.1.4.3** and parameters with status codes are described in Table 31, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively.

5.12.1 G0 *Temporal* (*Between*) *with* G0 (*Attribute*)

Temporal		Between					
Attribute		<	>	==	!=	>=	<=
Code	Search Type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	‘001a&017a’	‘001a&019a’	‘001a&021a’	‘001a&023a’	‘001a&025a’	‘001a&027a’
	Count	‘007a&029a’	‘007a&031a’	‘007a&033a’	‘007a&035a’	‘007a&037a’	‘007a&039a’
400	Search	‘002a&018a’	‘002a&020a’	002a&022a’	‘002a&024a’	‘002a&026a’	‘002a&028a’
	Count	‘008a&030a’	‘008a&032a’	‘008a&034a’	‘008a&036a’	‘008a&038’	‘008a&040a’
404	Search	‘002b&018b’	002b&020b’	‘002b&022b’	‘002b&024b’	‘002b&026b’	‘002b&028b’
	Count	‘008b&030b’	008b&032b’	‘008b&034b’	‘008b&036b’	‘008b&038b’	‘008b&040b’
401	Search	‘002c&’018c’	002c&020c’	‘002c&022c’	‘002c&024c’	‘002c&026c’	‘002c&028c’
	Count	‘008c&030c’	008c&032c’	008c&034c’	‘008c&036c’	‘008c&038c’	‘008c&040c’

5.12.2 G0 *Temporal* (*Before*) *with* G0 (*Attribute*)

Temporal		Before					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	‘003a&017a	‘003a&019a’	‘003a&021a’	‘003a&023a’	‘003a&025a’	‘003a&027a’
	Count	‘009a&02’9a	‘009a&031a’	‘009a&033a’	‘009a&035a’	‘009a&037a’	‘009a&039a’
400	Search	‘004a&018a	‘004a&020a’	‘004a&022a’	‘004a&024a’	‘004a&026a’	‘004a&028a’
	Count	‘010a&030a	‘010a&032a’	‘010a&034a’	‘010a&036a’	‘010a&038a’	‘010a&040a’
404	Search	‘004b&018b	‘004b&020b’	‘004b&022b’	‘004b&024b’	‘004b&026b’	‘004b&028b’
	Count	‘010b&030b	‘010b&032b’	‘010b&034b’	‘010b&036b’	‘010b&038b’	‘010b&040b’
401	Search	‘004c&018c	‘004c&020c’	‘004c&’022c’	‘004c&024c’	‘004c&026c’	‘004c&028c’
	Count	‘010c&030c	‘010c&032c’	‘010c&034c’	‘010c&036c’	‘010c&038c’	‘010c&040c’

5.12.3 G0 *Temporal* (*After*) *with* G0 (*Attribute*)

Temporal		Before					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	‘003a&017a’	‘003a&019a’	‘003a&021a’	‘003a&023a’	‘003a&025a’	‘003a&027a’
	Count	‘009a&029a’	‘009a&031a’	‘009a&033a’	‘009a&035a’	‘009a&037a’	‘009a&039a’
400	Search	‘004a&018a’	‘004a&020a’	‘004a&022a’	‘004a&024a’	‘004a&026a’	‘004a&028a’
	Count	‘010a&030a’	‘010a&032a’	‘010a&034a’	‘010a&036a’	‘010a&038a’	‘010a&040a’
404	Search	‘004b&018b’	‘004b&020b’	‘004b&022b’	‘004b&024b’	‘004b&026b’	‘004b&028b’
	Count	‘010b&030b’	‘010b&032b’	‘010b&034b’	‘010b&036b’	‘010b&038b’	‘010b&040b’
401	Search	‘004c&018c’	‘004c&020c’	‘004c&’022c’	‘004c&024c’	‘004c&026c’	‘004c&028c’
	Count	‘010c&030c’	‘010c&032c’	‘010c&034c’	‘010c&036c’	‘010c&038c’	‘010c&040c’

5.13 Tests for G2 Complex on G0 (Spatial) and G0 (Attribute)

The combination of tests under this section are associated with the G2 Complex functional profile. These tests summarise the HTTP response code 2XX and 4XX for the G0 (Spatial) query on G0 (Attribute) query. The query semantics of the spatial search are described in **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively. The query semantics of the attribute search are described in **6.1.4.3** and parameters with status codes are described in Table 31, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively.

5.13.1 G0 (*Spatial- BBox*) *and* G0 (*Attribute*)

Spatial		BBox					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	‘013a&017a’	‘013a&019a’	‘013a&021a’	‘013a&023a’	‘013a&025a’	‘013a&027a’
	Count	‘015a&029a’	‘015a&031a’	‘015a&033a’	‘015a&035a’	‘015a&037a’	‘015a&039a’
400	Search	‘014a&018a’	‘014a&020a’	‘014a&022a’	‘014a&024a’	‘014a&026a’	‘014a&028a’
	Count	‘016a&030a’	‘016a&032a’	‘016a&034a’	‘016a&036a’	‘016a&038a’	‘016a&040a’
404	Search	‘014c&018b’	‘014c&020b’	‘014c&022b’	‘014c&024b’	‘014c&026b’	‘014c&028b’
	Count	‘016c&030b’	‘016c&032b’	‘016c&034b’	‘016c&036b’	‘016c&038b’	‘016c&040b’
401	Search	‘014b&018c’	‘014b&020c’	‘014b&022c’	‘014b&024c’	‘014b&026c’	‘014b&028c’
	Count	‘016b&030c’	‘016b&032c’	‘016b&034c’	‘016b&036c'	‘016b&038c’	‘016b&040c’

5.14 G2 (Complex) : Using G0 (Temporal), G0 (Spatial: BBOX) and G0 (Attribute)

The combination of tests under this section are associated with the G2 Complex functional profile as defined in Table 3. These tests summarise the HTTP response code 2XX and 4XX for the temporal between/before/after queries with a Spatial BBox geometry query with an attribute query for ‘search’ and ‘count’ query implementation. The query semantics of the temporal search are described in **6.1.4.2** and parameters with status codes are described in Table 33, under **6.1.3.2** of IS 18003 (Part 2) of the standard respectively. The query semantics of the spatial search are described in **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively. The query semantics of the attribute search are described in **6.1.4.3** and parameters with status codes are described in Table 31, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively.

5.14.1 G0 (*Temporal: Between*), G0 (*Spatial: BBOX*) *and* G0 (*Attribute*)

Temporal		Between					
Spatial		BBox					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	‘001a&01 3a&017a’	‘001a&01 3a&019a’	‘001a&01 3a&021a’	‘001a&01 3a&023a’	‘001a&01 3a&025a’	‘001a&01 3a&027a’
	Count	‘007a&01 5a&029a’	‘007a&01 5a&031a’	‘007a&01 5a&033a’	‘007a&01 5a&035a’	‘007a&01 5a&037a’	‘007a&01 5a&039a’
400	Search	‘002a&01 4a&018a’	‘002a&01 4a&020a’	‘002a&01 4a&022a’	‘002a&01 4a&024a’	‘002a&01 4a&026a’	‘002a&01 4a&028a’
	Count	‘008a&01 6a&030a’	‘008a&01 6a&032a’	‘008a&01 6a&034a’	‘008a&01 6a&036a’	‘008a&01 6a&038a’	‘008a&01 6a&040a’
404	Search	‘002b&01 4c&018b’	‘002b&01 4c&020b’	‘002b&01 4c&022b’	‘002b&01 4c&024b’	‘002b&01 4c&026b’	‘002b&01 4c&028b’
	Count	‘008b&01 6c&030b’	‘008b&01 6c&032b’	‘008b&01 6c&034b’	‘008b&01 6c&036b’	‘008b&01 6c&038b’	‘008b&01 6c&040b’
401	Search	‘002c&01 4b&018c’	‘002c & 01 4b & 020c’	‘002c&01 4b&022c’	‘002c&01 4b&024c’	‘002c&01 4b&026c’	‘002c&01 4b&028c’
	Count	‘008c&01 6b&030c’	‘008c&01 6b&032c’	‘008c&01 6b&034c’	‘008c&01 6b&036c’	‘008c&01 6b&038c’	‘008c&01 6b&040c’

5.14.2 G0 (*Temporal: Before*), G0 (*Spatial: BBOX*) *and* G0 (*Attribute*)

Temporal		Before					
Spatial		BBox					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	‘003a&01 3a&017a’	‘003a&01 3a&019a’	‘003a&01 3a&021a’	‘003a&01 3a&023a’	‘003a&01 3a&025a’	‘003a&01 3a&027a’
	Count	‘009a&01 5a&029a’	‘009a&01 5a&031a’	‘009a&01 5a&033a’	‘009a&01 5a&035a’	‘009a&01 5a&037a’	‘009a&01 5a&039a’
400	Search	‘004a&01 4a&018a’	‘004a&01 4a&020a’	‘004a&01 4a&022a’	‘004a&01 4a&024a’	‘004a&01 4a&026a’	‘004a&01 4a&028a’
	Count	‘010a&01 6a&030a’	‘010a&01 6a&032a’	‘010a&01 6a&034a’	‘010a&01 6a&036a’	‘010a&01 6a&038a’	‘010a&01 6a&040a’
404	Search	‘004b&01 4c&018b’	‘004b&01 4c&020b’	‘004b&01 4c&022b’	‘004b&01 4c&024b’	‘004b&01 4c&026b’	‘004b&01 4c&028b’
	Count	‘010b&01 6c&030b’	‘010b&01 6c&032b’	‘010b&01 6c&034b’	‘010b&01 6c&036b’	‘010b&01 6c&038b’	‘010b&01 6c&040b’
401	Search	‘004c&01 4b&018c’	‘004c&01 4b&020c’	‘004c&01 4b&022c’	‘004c&01 4b&024c’	‘004c&01 4b&026c’	‘004c&01 4b&028c’
	Count	‘010c&01 6b&030c’	‘010c&01 6b&032c’	‘010c&01 6b&034c’	‘010c&01 6b&036c’	‘010c&01 6b&038c’	‘010c&01 6b&040c’

5.14.3 G0 (*Temporal:After*), G0 (*Spatial:BBOX*) *and* G0 (*Attribute*)

Temporal		After					
Spatial		BBox					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	‘005a&01 3a&017a’	‘005a&01 3a&019a’	‘005a&01 3a&021a’	‘005a&01 3a&023a’	‘005a&01 3a&025a’	‘005a&01 3a&027a’
	Count	‘011a&01 5a&029a’	‘011a&01 5a&031a’	‘011a&01 5a&033a’	‘011a&01 5a&035a’	‘011a&01 5a&037a’	‘011a&01 5a&039a’
400	Search	‘006a&01 4a&018a’	‘006a&01 4a&020a’	‘006a&01 4a&022a’	‘006a&01 4a&024a’	‘006a&01 4a&026a’	‘006a&01 4a&028a’
	Count	‘012a&01 6a&030a’	‘012a&01 6a&032a’	‘012a&01 6a&034a’	‘012a&01 6a&036a’	‘012a&01 6a&038a’	‘012a&01 6a&040a’
404	Search	‘006b&01 4c&018b’	‘006b&01 4c&020b’	‘006b&01 4c&022b’	‘006b&01 4c&024b’	‘006b&01 4c&026b’	‘006b&01 4c&028b’
	Count	‘012b&01 6c&030b’	‘012b&01 6c&032b’	‘012b&01 6c&034b’	‘012b&01 6c&036b’	‘012b&01 6c&038b’	‘012b&01 6c&040b’
401	Search	‘006c&01 4b&018c’	‘006c&01 4b&020c’	‘006c&01 4b&022c’	‘006c&01 4b&024c’	‘006c&01 4b&026c’	‘006c&01 4b&028c’
	Count	‘012c&01 6b&030c’	‘012c&01 6b&032c’	‘012c&01 6b&034c’	‘012c&01 6b&036c’	‘012c&01 6b&038c’	‘012c&01 6b&040c’

5.15. G2 (Complex) : Using G0 (Temporal) and G1 (Spatial)

The combination of tests under this section are associated with the G2 complex functional profiles. These tests summarise the HTTP response code 2XX and 4XX for the temporal between/before/after queries with a spatial circle/polygon/linestring geometry query with an attribute query for ‘search’ and ‘count’ query implementation. The query semantics of the temporal search are described in **6.1.4.2** and parameters with status codes are described in Table 33, under **6.1.3.2** of IS 18003 (Part 2) of the standard respectively. The query semantics of the spatial search are described in **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively.

5.15.1 G0 (*Temporal: Between*) *and* G1 (*Spatial: Circle*)

Temporal		Between					
Spatial		Circle					
Test	Search type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	‘001a&043a’	‘001a&043a’	‘001a&043a’	‘001a&043a’	‘001a&043a’	‘001a&043a’
	Count	‘007a&049a’	‘007a&49a’	‘007a&049a’	‘007a&049a’	‘007a&049a’	‘007a&049a’
400	Search	‘002a&044a’	‘002a&044a’	‘002a&044a’	‘002a&44a’	‘002a&044a’	‘002a&44a’
	Count	‘008a&050a’	‘008a&050a’	‘008a&050a’	‘008a&050a’	‘008a&050a’	‘008a&050a’
404	Search	‘002b&044c’	‘002b&044c’	‘002b&044c’	‘002b&044c’	‘002b&044c’	‘002b&044c’
	Count	‘008b&050c’	‘008b&050c’	‘008b&050c’	‘008c&050c’	‘008b&050c’	‘008b&050c’
401	Search	‘002c&044b’	‘002c&044b’	‘002c&044b’	‘002c&044b’	‘002d&044b’	‘002c&044b’
	Count	‘008c&050b’	‘008c&050b’	‘008c&050b’	‘008c&050b’	‘008c&050b’	‘008c&050b’

5.15.2 G0 (*Temporal: Between*) *and* G1 (*Spatial: Polygon*)

Temporal		Between					
Spatial		Polygon					
Search type		Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘001a&045a’	‘001a&045a’	‘001a&045a’	‘001a&045a’	‘001a&045a’	‘001a&045a’
	Count	‘007a&51a’	‘007a&51a’	‘007a&051a’	‘007a&051a’	‘007a&51a’	‘007a&51a’
400	Search	‘002a&46a’	‘002a&046a’	‘002a&'046a’	‘002a&46a’	‘002a&46a’	‘002a&46a’
	Count	‘008a&052a’	‘008a&052a’	‘008a&052a’	‘008a&052a’	‘008a&052a’	‘008a&052a’
404	Search	‘002b&046c’	‘002b&046c’	‘002b&046c’	‘002b&046c’	‘002b&046c’	‘002b&046c’
	Count	‘008b&052c’	‘008b&052c’	‘008b&052c’	‘008b&052c’	‘008b&052c’	‘008b&052c’
401	Search	‘002c&046b’	‘002c&046b’	‘002c&046b’	‘002c&046b’	‘002c&046b’	‘002c&046b’
	Count	‘008c&052b’	‘008c&052b’	‘008c&052b’	‘008c&052b’	‘008c&052b’	‘008c&052b’

5.15.3 G0 (*Temporal: Between*) *and* G1 (*Spatial: Linestring*)

Temporal		Between					
Spatial		Linestring					
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘001a&047a’	‘001a&047a’	‘001a&047a’	‘001a&047a’	‘001a&47a’	‘001a&047a’
	Count	‘007a&053a’	‘007a&053a’	‘007a&053a’	‘007a&053a’	‘007a&053a’	‘007a&053a’
400	Search	‘002a&048a’	‘002a&048a’	‘002a&048a’	‘002a&048a’	‘002a&048a’	‘002a&048a’
	Count	‘008a&054a’	‘008a&054a’	‘008a&054a’	‘008a&054a’	‘008a&054a’	‘008a&054a’
404	Search	‘002b&048c’	‘002b&048c’	‘002b&048c’	‘002b&048c’	‘002b&048c’	‘002b&048c’
	Count	‘008b&054c’	‘008b&054c’	‘008b&054c’	‘008b&054c’	‘008b&054c’	‘008b&054c’
401	Search	‘002c&048b’	‘002c&048b’	‘002c&048b’	‘002c&048b’	‘002c&048b’	‘002c&048b’
	Count	‘008c&054b’	‘008c&054b’	‘008c&054b’	‘008c&054b’	‘008c&054b’	‘008c&054b’

5.15.4 G0 (*Temporal: Before*) *and* G1 (*Spatial: Circle*)

Temporal		Before					
Spatial		Circle					
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘003a&043a’	‘003a&043a’	‘003a&043a’	‘003a&043a’	‘003a&043a’	‘003a&043a’
	Count	‘009a&049a’	‘009a&049a’	‘009a&049a’	‘009a&049a’	‘009a&049a’	‘009a&049a’
400	Search	‘004a&044a’	‘004a&044a’	‘004a&044a’	‘004a&044a’	‘004a&044a’	‘004a&044a’
	Count	‘010a&050a’	‘010a&050a’	‘010a&050a’	‘010a&050a’	‘010a&050a’	‘010a&050a’
404	Search	‘004b&044c’	‘004b&044c’	‘004b&044c’	‘004b&044c’	‘004b&044c’	‘004b&044c’
	Count	‘010b&050c’	‘010b&050c’	‘010b&050c’	‘010b&050c’	‘010b&050c’	‘010b&050c’
401	Search	‘004c&044b’	‘004c&044b’	‘004c&044b’	‘004c&044b’	‘004c&044b’	‘004c&044b’
	Count	‘010c&050b’	‘010c&050b’	‘010c&050b’	‘010c&050b’	‘010c&050b’	‘010c&050b’

5.15.5 G0 (*Temporal: Before*) *and* G1 (*Spatial: Polygon*)

Temporal		Before					
Spatial		Polygon					
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘003a&045a’	‘003a&045a’	‘003a&045a’	‘003a&045a’	‘003a&045a’	‘003a&045a’
	Count	‘009a&051a’	‘009a&051a’	‘009a&051a’	‘009a&051a’	‘009a&051a’	‘009a&051a’
400	Search	‘004a&046a’	‘004a&046a’	‘004a&046a’	‘004a&046a’	‘004a&046a’	‘004a&046a’
	Count	‘010a&052a’	‘010a&052a’	‘010a&052a’	‘010a&052a’	‘010a&052a’	‘010a&052a’
404	Search	‘004b&046c’	‘004b&046c’	‘004b&046c’	‘004b&046c’	‘004b&046c’	‘004b&046c’
	Count	‘010b&052c’	‘010b&052c’	‘010b&052c’	‘010b&052c’	‘010b&052c’	‘010b&052c’
401	Search	‘004c&046b’	‘004c&046b’	‘004c&046b’	‘004c&046b’	‘004c&046b’	‘004c&046b’
	Count	‘010c&052b’	‘010c&052b’	‘010c&052b’	‘010c&052b’	‘010c&052b’	‘010c&052b’

5.15.6 G0 (*Temporal: Before*) *and* G1 (*Spatial: Linestring*)

Temporal		Before					
Spatial		Linestring					
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘003a&047a’	‘003a&047a’	‘003a&047a’	‘003a&047a’	‘003a&047a’	‘003a&047a’
	Count	‘009a&053a’	‘009a&053a’	‘009a&053a’	‘009a&053a’	‘009a&053a’	‘009a&053a’
400	Search	‘004a&048a’	‘004a&048a’	‘004a&048a’	‘004a&048a’	‘004a&048a’	‘004a&048a’
	Count	‘010a&054a’	‘010a&054a’	‘010a&054a’	‘010a&054a’	‘010a&054a’	‘010a&054a’
404	Search	‘004b&048c’	‘004b&048c’	‘004b&048c’	‘004b&048c’	‘004b&048c’	‘004b&048c’
	Count	‘010b&054c’	‘010b&054c’	‘010b&054c’	‘010b&054c’	‘010b&054c’	‘010b&054c’
401	Search	‘004c&048b’	‘004c&048b’	‘004c&048b’	‘004c&048b’	‘004c&048b’	‘004c&048b’
	Count	‘010c&054b’	‘010c&054b’	‘010c&054b’	‘010c&054b’	‘010c&054b’	‘010c&054b’

5.15.7 G0 (*Temporal: After*) *and* G1 (*Spatial: Circle*)

Temporal		After					
Spatial		Circle					
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘005a&043a’	‘005a&043a’	‘005a&043a’	‘005a&043a’	‘005a&043a’	‘005a&043a’
	Count	‘011a&049a’	‘011a&049a’	‘011a&049a’	‘011a&049a’	‘011a&049a’	‘011a&049a’
400	Search	‘006a&044a’	‘006a&044a’	‘006a&044a’	‘006a&044a’	‘006a&044a’	‘006a&044a’
	Count	‘012a&050a’	‘012a&050a’	‘012a&050a’	‘012a&050a’	‘012a&050a’	‘012a&050a’
404	Search	‘006b&044c’	‘006b&044c’	‘006b&044c’	‘006b&044c’	‘006b&044c’	‘006b&044c’
	Count	‘012b&050c’	‘012b&050c’	‘012b&050c’	‘012b&050c’	‘012b&050c’	‘012b&050c’
401	Search	‘006c&044b’	‘006c&044b’	‘006c&044b’	‘006c&044b’	‘006c&044b’	‘006c&044b’
	Count	‘012c&050c’	‘012c&050c’	‘012c&050c’	‘012c&050c’	‘012c&050c’	‘012c&050c’

5.15.8 G0 (*Temporal: After*) *and* G1 (*Spatial: Polygon*)

Temporal		After					
Spatial		Polygon					
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘005a&045a’	‘005a&045a’	‘005a&045a’	‘005a&045a’	‘005a&045a’	‘005a&045a’
	Count	‘011a&051a’	‘011a&051a’	‘011a&051a’	‘011a&051a’	‘011a&051a’	‘011a&051a’
400	Search	‘006a&046a’	‘006a&046a’	‘006a&046a’	‘006a&046a’	‘006a&046a’	‘006a&046a’
	Count	‘012a&052a’	‘012a&052a’	‘012a&052a’	‘012a&052a’	‘012a&052a’	‘012a&052a’
404	Search	‘006b&046c’	‘006b&046c’	‘006b&046c’	‘006b&046c’	‘006b&046c’	‘006b&046c’
	Count	‘012b&052c’	‘012b&052c’	‘012b&052c’	‘012b&052c’	‘012b&052c’	‘012b&052c’
401	Search	‘006c&046b’	‘006c&046b’	‘006c&046b’	‘006c&046b’	‘006c&046b’	‘006c&046b’
	Count	‘012c&052b’	‘012c&052b’	‘012c&052b’	‘012c&052b’	‘012c&052b’	‘012c&052b’

5.15.9 G0 (*Temporal: After*) *and* G1 (*Spatial: Linestring*)

Temporal		After					
Spatial		Linestring					
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘005a&047a’	‘005a&047a’	‘005a&047a’	‘005a&047a’	‘005a&047a’	‘005a&047a’
	Count	‘011a&053a’	‘011a&053a’	‘011a&053a’	‘011a&053a’	‘011a&053a’	‘011a&053a’
400	Search	‘006a&048a’	‘006a&048a’	‘006a&048a’	‘006a&048a’	‘006a&048a’	‘006a&048a’
	Count	‘012a&054a’	‘012a&054a’	‘012a&054a’	‘012a&054a’	‘012a&054a’	‘012a&054a’
404	Search	‘006b&048c’	‘006b&048c’	‘006b&048c’	‘006b&048c’	‘006b&048c’	‘006b&048c’
	Count	‘012b&054c’	‘012b&054c’	‘012b&054c’	‘012b&054c’	‘012b&054c’	‘012b&054c’
401	Search	‘006c&048b’	‘006c&048b’	‘006c&048b’	‘006c&048b’	‘006c&048b’	‘006c&048b’
	Count	‘012c&054b’	‘012c&054b’	‘012c&054b’	‘012c&054b’	‘012c&054b’	‘012c&054b’

5.16 G2 (Complex): Using G0 (Attribute) and G1 (Spatial)

The combination of tests under this section are associated with the complex profiles. These tests summarise the HTTP response code 2XX and 4XX for the temporal between/before/after queries with a spatial circle/polygon/linestring geometry query with an attribute query for ‘search’ and ‘count’ query implementation. The query semantics of the spatial search are described in **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** of IS 18003 (Part 2) of the standard respectively. The query semantics of the attribute search are described in **6.1.4.3** and parameters with status codes are described in Table 31, under **6.1.3.1** of IS 18003 (Part 2) of the standard, respectively.

5.16.1 G0 (*Attribute*) *and* G1 (*Spatial:Circle*)

Spatial		Circle					
Attribute		<	>	==	!=	>=	<=
Test	Search type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	‘043a&017a’	‘043a&019a’	‘043a&021a’	‘043a&023a’	‘043a&025a’	‘043a&027a’
	Count	‘049a&029a’	‘049a&031a’	‘049a&033a’	‘049a&035a’	‘049a&037a’	‘049a&039a’
400	Search	‘044a&018a’	‘044a&020a’	‘044a&022a’	‘044a&024a’	‘044a&026a’	‘044a&028a’
	Count	‘050a&030a’	‘050a&032a’	‘050a&034a’	‘050a&036a’	‘050a&038a’	‘050a&040a’
404	Search	‘044c&018b’	‘044c&020b’	‘044c&022b’	‘044c&024b’	‘044c&026b’	‘044c&028b’
	Count	‘050c&030b’	‘050c&032b’	‘050c&034b’	‘050c&036b’	‘050c&038b’	‘050c&040b’
401	Search	‘044b&018c’	‘044b&020c’	‘044b&022c’	‘044b&024c’	‘044b&026c’	‘044b&028c’
	Count	‘050b&030c’	‘050b&032c’	‘050b&034c’	‘050b&036c’	‘050b&038c’	‘050b&040c’

5.16.2 G0 (*Attribute*) *and* G1 (*Spatial: Polygon*)

Spatial		Polygon					
Attribute		<	>	==	!=	>=	<=
Search type		Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘045a&017a’	‘045a&019a’	‘045a&021a’	‘045a&023a’	‘045a&025a’	‘045a&027a’
	Count	‘051a&029a’	‘051a&031a’	‘051a&033a’	‘051a&035a’	‘051a&037a’	‘051a&039a’
400	Search	‘046a&018a’	‘046a&020a’	‘046a&022a’	‘046a&024a’	‘046a&026a’	‘046a&028a’
	Count	‘052a&030a’	‘052a&032a’	‘052a&034a’	‘052a&036a’	‘052a&038a’	‘052a&040a’
404	Search	‘046c&018b’	‘046c&020b’	‘046c&022b’	‘046c&024b’	‘046c&026b’	‘046c&028b’
	Count	‘052c&030b’	‘052c&032b’	‘052c&034b’	‘052c&036b’	‘052c&038b’	‘052c&040b’
401	Search	‘046b&018c’	‘046b&020c’	‘046b&022c’	‘046b&024c’	‘002d&026c’	‘046b&028c’
	Count	‘052b&030c’	‘052b&032c’	‘052b&034c’	‘052b&036c’	‘052b&038c’	‘052b&040c’

5.16.3 G0 (*Attribute*) *and* G1 (*Spatial: Linestring*)

Spatial		Linestring					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘047a&017a’	‘047a&019a’	‘047a&021a’	‘047a&023a’	‘047a&025a’	‘047a&027a’
	Count	‘053a&029a’	‘053a&031a’	‘053a&033a’	‘053a&035a’	‘053a&037a’	‘053a&039a’
400	Search	‘048a&018a’	‘048a&020a’	‘048a&022a’	‘048a&024a’	‘048a&026a’	‘048a&028a’
	Count	‘054a&030a’	‘054a&032a’	‘054a&034a’	‘054a&036a’	‘054a&038a’	‘054a&040a’
404	Search	‘048c&018b’	‘048c&020b’	‘048c&022b’	‘048c&024b’	‘048c&026b’	‘048c&028b’
	Count	‘054c&030b’	‘054c&032b’	‘054c&034b’	‘054c&036b’	‘054c&038b’	‘054c&040b’
401	Search	‘048b&018c’	‘048b&020c’	‘048b&022c’	‘048b&024c’	‘048b&026c’	‘048b&028c’
	Count	‘054b&030c’	‘054b&032c’	‘054b&034c’	‘054b&036c’	‘054b&038c’	‘054b&040c’

5.17 G2 (Complex) : Using G0 (Temporal), G1 (Spatial) and G0 (Attribute)

The combination of tests under this section are associated with the complex functional profile. These tests summarise the HTTP response code 2XX and 4XX for the temporal between/before/after queries with a spatial circle/polygon/linestring geometry query with an attribute query for 'search' and 'count' query implementation. The query semantics of the temporal search are described in **6.1.4.2** and parameters with status codes are described in Table 33, under **6.1.3.2** of IS 18003 (Part 2) of the standard, respectively. The query semantics of the spatial search are described in **6.1.2.3** and parameters with status codes are described in Table 30, under **6.1.3.1** of IS 18003 (Part 2) of the standard, respectively. The query semantics of the attribute search are described in **6.1.4.3** and parameters with status codes are described in Table 31, under **6.1.3.1** of IS 18003 (Part 2) of the standard, respectively.

5.17.1 G0 (*Temporal: Between*), G1 (*Spatial: Circle*) *and* G0 (*Attribute*)

Temporal				Between			
Spatial				Circle			
Attribute		<	>	==	!=	>=	<=
Test	Search type	Tests	Tests	Tests	Tests	Tests	Tests
200	Search	'001a&043a&017a'	'001a&043a&019a'	'001a&043a&021a'	'001a&043a&023a'	'001a&043a&025a'	'001a&043a&027a'
	Count	'007a&049a&029a'	'007a&49a&031a'	'007a&049a&033a'	'007a&049a&035a'	'007a&049a&037a'	'007a&049a&039a'
400	Search	'002a&044a&018a'	'002a&044a&020a'	'002a&044a&022a'	'002a&44a&024a'	'002a&044a&026a'	'002a&44a&028a'
	Count	'008a&050a&030a'	'008a&050a&032a'	'008a&050a&034a'	'008a&050a&036a'	'008a&050a&038a'	'008a&050a&040a'
404	Search	'002b&044c&018b'	'002b&044c&020b'	'002b&044c&022b'	'002b&044c&024b'	'002b&044c&026b'	'002b&044c&028b'
	Count	'008b&050c&030b'	'008b&050c&032b'	'008b&050c&034b'	'008b&050c&036b'	'008b&050c&038b'	'008b&050c&040b'
401	Search	'002c&044b&018c'	'002c&044b&020c'	'002c&044b&022c'	'002c&044b&024c'	'002c&044b&026c'	'002c&044b&028c'
	Count	'008c&050b&030c'	'008c&050b&032c'	'008c&050b&034c'	'008c&050b&036c'	'008c&050b&038c'	'008c&050b&040c'

5.17.2 G0 (*Temporal: Between*), G1 (*Spatial: Polygon*) *and* G0 (*Attribute*)

Temporal		Between					
Spatial		Polygon					
Attribute		<	>	==	!=	>=	<=
Search type		Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘001a&45a&017a’	‘001a&045a&019a’	‘001a&045a&021a’	‘001a&045a&023a’	‘001a&045a&025a’	‘001a&045a&027a’
	Count	‘007a&051a&029a’	‘007a&051a&031a’	‘007a&051a&033a’	‘007a&051a&035a’	‘007a&51a&037a’	‘007a&51a&039a’
400	Search	‘002a&46a&018a’	‘002a&046a&020a’	‘002a&046a&022a’	‘002a&046a&024a’	‘002a&046a&026a’	‘002a&046a&028a’
	Count	‘008a&052a&030a’	‘008a&052a&032a’	‘008a&052a&034a’	‘008a&052a&036a’	‘008a&052a&038a’	‘008a&052a&040a’
404	Search	‘002b&046c&018b’	‘002b&046c&020b’	‘002b&046c&022b’	‘002b&046c&024b’	‘002b&046c&026b’	‘002b&046c&028b’
	Count	‘008b&052c&030b’	‘008b&052c&032b’	‘008b&052c&034b’	‘008b&052c&036b’	‘008b&052c&038b’	‘008b&052c&040b’
401	Search	‘002c&046b&018c’	‘002c&046b&020c’	‘002c&046b&022c’	‘002c&046b&024c’	‘002c&046b&026c’	‘002c&046b&028c’
	Count	‘008c&052b&030c’	‘008c&052b&032c’	‘008c&052b&034c’	‘008c&052b&036c’	‘008c&052b&038c’	‘008c&052b&040c’

5.17.3 G0 (*Temporal: Between*), G1 (*Spatial: Linestring*) *and* G0 (*Attribute*)

Temporal		Between					
Spatial		Linestring					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘001a&047a&017a’	‘001a&047a&019a’	‘001a&047a&021a’	‘001a&047a&023a’	‘001a&47a&025a’	‘001a&047a&027a’
	Count	‘007a&053a&029a’	‘007a&053a&031a’	‘007a&053a&033a’	‘007a&053a&035a’	‘007a&053a&037a’	‘007a&053a&039a’
400	Search	‘002a&048a&018a’	‘002a&048a&020a’	‘002a&048a&022a’	‘002a&048a&024a’	‘002a&048a&026a’	‘002a&048a&028a’
	Count	‘008a&054a&030a’	‘008a&054a&032a’	‘008a&054a&034a’	‘008a&054a&036a’	‘008a&054a&038a’	‘008a&054a&040a’
404	Search	‘002b&048c&018b’	‘002b&048c&020b’	‘002b&048c022b’	‘002b&048c&024b’	‘002b&048c&026b’	‘002b&048c&028b’
	Count	‘008b&054c&030b’	‘008b&054c&032b’	‘008b&054c&034b’	‘008b&054c&036b’	‘008b&054c&038b’	‘008b&054c&040b’
401	Search	‘002c&048b&018c’	‘002c&048b&020c’	‘002c&048b&022c’	‘002c&048b&024c’	‘002c&048b&026c’	‘002c&048b&028c’
	Count	‘008c&054b&030c’	‘008c&054b&032c’	‘008c&054b&034c’	‘008c&054b&036c’	‘008c&054b&038c’	‘008c&054b&040c’

5.17.4 G0 (*Temporal: Before*), G1 (*Spatial: Circle*) *and* G0 (*Attribute*)

Temporal		Before					
Spatial		Circle					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘003a&043a&017a’	‘003a&043a&019a’	‘003a&043a&021a’	‘003a&043a&023a’	‘003a&043a&025a’	‘003a&043a&027a’
	Count	‘009a&049a&029a’	‘009a&049a&031a’	‘009a&049a&033a’	‘009a&049a&035a’	‘009a&049a&037a’	‘009a&049a&039a’
400	Search	‘004a&044a&018a’	‘004a&044a&020a’	‘004a&044a&022a’	‘004a&044a&024a’	‘004a&044a&026a’	‘004a&044a&028a’
	Count	‘010a&050a&030a’	‘010a&050a&032a’	‘010a&050a&034a’	‘010a&050a&036a’	‘010a&050a&038a’	‘010a&050a&040a’
404	Search	‘004b&044c&018b’	‘004b&044c&020b’	‘004b&044c022b’	‘004b&044c&024b’	‘004b&044c&026b’	‘004b&044c&028b’
	Count	‘010b&050c&030b’	‘010b&050c&032b’	‘010b&050c&034b’	‘010b&050c&036b’	‘010b&050c&038b’	‘010b&050c&040b’
401	Search	‘004c&044b&018c’	‘004c&044b&020c’	‘004c&044b&022c’	‘004c&044b&024c’	‘004c&044b&026c’	‘004c&044b&028c’
	Count	‘010c&050b&030c’	‘010c&050b&032c’	‘010c&050b&034c’	‘010c&050b&036c’	‘010c&050b&038c’	‘010c&050b&040c’

5.17.5 G0 (*Temporal: Before*), G1 (*Spatial: Polygon*) *and* G0 (*Attribute*)

Temporal		Before					
Spatial		Polygon					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘003a&045a&017a’	‘003a&045a&019a’	‘003a&045a&021a’	‘003a&045a&023a’	‘003a&045a&025a’	‘003a&045a&027a’
	Count	‘009a&051a&029a’	‘009a&051a&031a’	‘009a&051a&033a’	‘009a&051a&035a’	‘009a&051a&037a’	‘009a&051a&039a’
400	Search	‘004a&046a&018a’	‘004a&046a&020a’	‘004a&046a&022a’	‘004a&046a&024a’	‘004a&046a&026a’	‘004a&046a&028a’
	Count	‘010a&052a&030a’	‘010a&052a&032a’	‘010a&052a&034a’	‘010a&052a&036a’	‘010a&052a&038a’	‘010a&052a&040a’
404	Search	‘004b&046c&018b’	‘004b&046c&020b’	‘004b&046c&022b’	‘004b&046c&024b’	‘004b&046c&026b’	‘004b&046c&028b’
	Count	‘010b&052c&030b’	‘010b&052c&032b’	‘010b&052c&034b’	‘010b&052c&036b’	‘010b&052c&038b’	‘010b&052c&040b’
401	Search	‘004c&046b&018c’	‘004c&046b&020c’	‘004c&046b&022c’	‘004c&046b&024c’	‘004c&046b&026c’	‘004c&046b&028c’
	Count	‘010c&052b&030c’	‘010c&052c&032c’	‘010c&052c&034c’	‘010c&052c&036c’	‘010c&052c&038c’	‘010c&052c&040c’

5.17.6 G0 (*Temporal: Before*), G1 (*Spatial: Linestring*) *and* G0 (*Attribute*)

Temporal		Before					
Spatial		Linestring					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘003a&047a&017a’	‘003a&047a&019a’	‘003a&047a&021a’	‘003a&047a&023a’	‘003a&047a&025a’	‘003a&047a&027a’
	Count	‘009a&053a&029a’	‘009a&053a&031a’	‘009a&053a&033a’	‘009a&053a&035a’	‘009a&053a&037a’	‘009a&053a&039a’
400	Search	‘004a&048a&018a’	‘004a&048a&020a’	‘004a&048a&022a’	‘004a&048a&024a’	‘004a&048a&026a’	‘004a&048a&028a’
	Count	‘010a&054a&030a’	‘010a&054a&032a’	‘010a&054a&034a’	‘010a&054a&036a’	‘010a&054a&038a’	‘010a&054a&040a’
404	Search	‘004b&048c&018b’	‘004b&048c&020b’	‘004b&048c&022b’	‘004b&048c&024b’	‘004b&048c&026b’	‘004b&048c&028b’
	Count	‘010b&054c&030b’	‘010b&054c&032b’	‘010b&054c&034b’	‘010b&054c&036b’	‘010b&054c&038b’	‘010b&054c&040b’
401	Search	‘004c&048b&018c’	‘004c&048b&020c’	‘004c&048b&022c’	‘004c&048b&024c’	‘004c&048b&026c’	‘004c&048b&028c’
	Count	‘010c&054b&030c’	‘010c&054c&032c’	‘010c&054c&034c’	‘010c&054c&036c’	‘010c&054c&038c’	‘010c&054c&040c’

5.17.7 G0 (*Temporal: After*), G1 (*Spatial: Circle*) *and* G0 (*Attribute*)

Temporal		After					
Spatial		Circle					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘005a&043a&017a’	‘005a&043a&019a’	‘005a&043a&021a’	‘005a&043a&023a’	‘005a&043a&025a’	‘005a&043a&027a’
	Count	‘011a&049a&029a’	‘011a&049a&031a’	‘011a&049a&033a’	‘011a&049a&035a’	‘011a&049a&037a’	‘011a&049a&039a’
400	Search	‘006a&044a&018a’	‘006a&044a&020a’	‘006a&044a&022a’	‘006a&044a&024a’	‘006a&044a&026a’	‘006a&044a&028a’
	Count	‘012a&050a&030a’	‘012a&050a&032a’	‘012a&050a&034a’	‘012a&050a&036a’	‘012a&050a&038a’	‘012a&050a&040a’
404	Search	‘006b&044c&018b’	‘006b&044c&020b’	‘006b&044c&022b’	‘006b&044c&024b’	‘006b&044c&026b’	‘006b&044c&028b’
	Count	‘012b&050c&030b’	‘012b&050c&032b’	‘012b&050c&034b’	‘012b&050c&036b’	‘012b&050c&038b’	‘012b&050c&040b’
401	Search	'006c&044b&018c’	‘006c&044b&020c’	‘006c&044b&022c’	‘006c&044b&024c’	‘006c&044b&026c’	‘006c&044b&028c’
	Count	‘012c&050b&030c’	‘012c&050b&032c’	‘012c&050b&034c’	‘012c&050b&036c’	‘012c&050b&038c’	‘012c&050b&040c’

5.17.8 G0 (*Temporal: After*), G1 (*Spatial: Polygon*) *and* G0 (*Attribute*)

Temporal		After					
Spatial		Polygon					
Attribute		<	>	==	!=	>=	<=
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	‘005a&045a&017a’	‘005a&045a&019a’	‘005a&045a&021a’	‘005a&045a&023a’	‘005a&045a&025a’	‘005a&045a&027a’
	Count	‘011a&051a&029a’	‘011a&051a&031a’	‘011a&051a&033a’	‘011a&051a&035a’	‘011a&051a&037a’	‘011a&051a&039a’
400	Search	‘006a&046a&018a’	‘006a&046a&020a’	‘006a&046a&022a’	‘006a&046a&024a’	‘006a&046a&026a’	‘006a&046a&028a’
	Count	‘012a&052a&030a’	‘012a&052a&032a’	‘012a&052a&034a’	012a&05’ 2a&036a’	012a&05’ 2a&038a’	‘012a&052a&040a’
404	Search	‘006b&046c&018b’	‘006b&046c&020b’	‘006b&046c&022b’	‘006b&046c&024b’	‘006b&046c&026b’	‘006b&046c&028b’
	Count	‘012b&052c&030b’	‘012b&052c&032b’	‘012b&052c&034b’	‘012b&052c&036b’	‘012b&052c&038b’	‘012b&052c&040b’
401	Search	‘006c&046b&018c’	‘006c&046b&020c’	‘006c&046b&022c’	‘006c&046b&024c’	‘006c&046b&026c’	‘006c&046b&028c’
	Count	‘012c&052b&030c’	‘012c&052b&032c’	‘012c&052b&034c’	‘012c&052b&036c’	‘012c&052b&038c’	‘012c&052b&040c’

5.17.9 G0 (*Temporal: After*), G1 (*Spatial:Linestring*) *and* G0 (*Attribute*)

Temporal				After			
Spatial				Linestring			
Attribute		<	>	==	!=	>=	<=
Code	Search type	Test	Tests	Tests	Tests	Tests	Tests
200	Search	005a&04 7a&017a	005a&04 7a&019a	005a&04 7a&021a	005a&04 7a&023a	005a&04 7a&025a	005a&04 7a&027a
	Count	‘011a&05 3a&029a’	‘011a&05 3a&031a’	‘011a&05 3a&033a’	‘011a&05 3a&035a’	‘011a&05 3a&037a’	‘011a&05 3a&039a’
400	Search	‘006a&04 8a&018a’	‘006a&04 8a&020a’	‘006a&04 8a&022a’	‘006a&04 8a&024a’	‘006a&04 8a&026a’	‘006a&04 8a&028a’
	Count	‘012a&05 4a&030a’	‘012a&05 4a&032a’	‘012a&05 4a&034a’	‘012a&05 4a&036a’	‘012a&05 4a&038a’	‘012a&05 4a&040a’
404	Search	‘006b&04 8c&018b’	‘006b&04 8c&020b’	‘006b&04 8c&022b’	‘006b&04 8c&024b’	‘006b&04 8c&026b’	‘006b&04 8c&028b’
	Count	‘012b&05 4c&030b’	‘012b&05 4c&032b’	‘012b&05 4c&034b’	‘012b&05 4c&036b’	‘012b&05 4c&038b’	‘012b&05 4c&040b’
401	Search	‘006c&04 8b&018c’	‘006c&04 8b&020c’	‘006c&04 8b&022c’	‘006c&04 8b&024c’	‘006c&04 8b&026c’	‘006c&04 8b&028c’
	Count	‘012c&05 4b&030c’	‘012c&05 4b&032c’	‘012c&05 4b&034c’	‘012c&05 4b&036c’	‘012c&05 4b&038c’	‘012c&05 4b&040c’

5.18 POST API

The specifications of the HTTP protocol do not specify any maximum length of a URL, however, there are practical limits imposed by web browsers. There is a constraint on the URL length in GET requests and if it exceeds the limit used by the web browser, it returns an error. To avoid this situation, the user may use POST API requests. POST API requests do not pose any such restrictions and give the user the liberty to make complex queries. This section shows a few examples of such POST queries.

5.18.1 *Sample Query for a POST API* (*Temporal*) *Query*

The following section gives a sample for a POST API for a tempoal search. The details of the parameters are available in the tests mentioned under **5**. For G0 (Temporal) profile please refer to **5.1** and **5.2** and for G1(Temporal) profile, please refer to **5.7** and **5.8**.

```
{  "temporalQ": { "timerel": "between",
      "time": "2020-06-01T14:20:00Z",
      "endtime": "2020-06-03T15:00:00Z",
      "timeProperty": "observationDateTime" } }
```

5.18.2 *Sample Query for a POST API* (*Spatial*) *Query*

The following section gives a sample for a POST API for a spatial search. The details of the parameters are available in the tests mentioned under **5**. For G0 (Spatial) profile please refer to **5.3** and **5.4** and for G1 (Spatial) profile, please refer to **5.9** and **5.10**.

```
{ "type": "Query", "entities":
 [ { "id":"iisc.ac.in/89a36273d77dac4cf38114fca1bbe64392547f86/rs.iudx.io/surat-itmsrealtime-
information/surat-itms-live-eta" } ],
   "geoQ": { "geometry": "Point",
      "coordinates": [21.178,72.834],
      "georel": "near;maxDistance=1000",
      "geoproperty": "location" },
       "options":"count" }
```

5.18.3 *Sample Query for a POST API* (*Attribute*) *Query*

The following section gives a sample for a POST API for a tempoal search. The details of the parameters are available in the tests mentioned under **5**. For G0 (Attribute) profile please refer to **5.5** and **5.6**.

```
{ "type": "Query", "entities":
[{ id": "iisc.ac.in/89a36273d77dac4cf38114fca1bbe64392547f86/rs.iudx.io/surat-itmsrealtime-
information/surat-itms-live-eta" }
```

5.18.4 *Sample Query for a POST API Complex Query*

The following section gives a sample for a POST API for a Complex search. A complex search is a combination of any of the spatial, temporal or attribute search/count query implementations. The details of the parameters are available in the tests mentioned under **5**.

```
{
 "type": "Query",
 "entities": [
  {
   "id":
"suratmunicipal.org/6db486cb4f720e8585ba1f45a931c63c25dbbbda/rs.iudx.org.in/surat-itmsrealtime-info/surat-itms-live-eta"
  }
 ],
 "geoQ": {
  "geometry": "Point",
  "coordinates": [
   25.319768,
   82.987988
  ],
  "georel": "near;maxDistance=1000",
  "geoproperty": "geoJsonLocation"
 },
 "temporalQ": {
  "timerel": "between",
  "time": "2020-06-01T14:20:00Z",
  "endtime": "2020-06-03T15:00:00Z",
  "timeProperty": "observationDateTime"
 },
 "q": "speed>=50",
 "attrs": "id,location" }
```


5.19 G3 (Subscription)

Subscription queries allow users to access resources as a stream using streaming protocols such as MQTT. Using this type of API, a consumer can obtain streaming information with which a streaming client can be developed that can be directly connected to the streaming server. By registering a subscription with the resource access service, users shall be provided with a dedicated channel with which data will be made available. The subscriptions API allows DX consumers to register (POST), modify (PATCH), list (GET) and delete (DELETE) a subscription for one or more data resources through a streaming service. The tests in this section verifies the correctness of subscription registration. The parameters and status codes for registration, updation and append (PUT) are available in Table 39 whereas the listing and deleting is available in Table 40 under **6.1.3.5** of IS 18003 (Part 2) of the standard.

5.19.1 *POST Queries*

Purpose	To test the correctness of ‘creating a subscription’	To test the correctness of ‘invalid parameters’ while creating a subscription’	To test the correctness of ‘invalid credentials’ while creating a subscription’
Description	This test validates the correctness of ‘creating a subscription’	This test validates the correctness of ‘invalid parameters’ for subscription	This test validates the correctness for an ‘invalid authorization token’ for a subscription query. It describes a test when the authentication has failed because the user has provided notoken, expired token or wrong token.

Methodology	Execute an API call that shall return a successful and verifiable response. Verify that output data is as per the test design.		
Test no.	085a	086a	086b
Response code	200	400	401

5.19.2 *GET Queries*

Purpose	To test the correctness of 'retrieving a subscription'	To test the correctness of 'invalid parameters' while retrieving a subscription'	To test the correctness of 'invalid credentials' while retrieving a subscription'
Description	This test validates the correctness of 'retrieving a subscription'	This test validates the correctness of 'invalid parameters' for subscription	This test validates the correctness for an 'invalid authorization token' for subscription. It describes a test when the authentication has failed because the user has provided no-token, expired token or wrong token.
Methodology	Execute an API call that shall return a successful and verifiable response. Verify that output data is as per the test design.		
Test no.	087a	088a	088b
Response code	200	400	401

5.19.3 *PUT Queries*

Purpose	To test the correctness of 'updating a subscription'	To test the correctness of 'invalid parameters' while updating a subscription'	To test the correctness of 'invalid credentials' while updating a subscription'
Description	This test validates the correctness of 'retrieving a subscription'	This test validates the correctness of 'invalid parameters' for subscription	This test validates the correctness for an 'invalid authorization token' for subscription It describes a test when the authentication has failed because the user has provided no-token, expired token or wrong token.
Methodology	Execute an API call that shall return a successful and verifiable response. Verify that output data is as per the test design.		
Test no.	089a	090a	090b
Response code	200	400	401

5.19.4 *PATCH Queries*

Purpose	To test the correctness of ‘appending a subscription’	To test the correctness of ‘invalid parameters’ while appending a subscription’	To test the correctness of ‘invalid credentials’ while appending a subscription’
Description	This test validates the correctness of ‘retrieving a subscription’	This test validates the correctness of ‘invalid parameters’ for subscription	This test validates the correctness for an ‘invalid authorization token’ for subscription
			It describes a test when the authentication has failed because the user has provided no-token, expired token or wrong token.
Methodology	Execute an API call that shall return a successful and verifiable response. Verify that output data is as per the test design.		
Test no.	091a	092a	092b
Response code	200	400	401

5.19.5 *DELETE Queries*

Purpose	To test the correctness of ‘deleting a subscription’	To test the correctness of ‘invalid parameters’ while deleting a subscription’	To test the correctness of ‘invalid credentials’ while deleting a subscription’
Description	This test validates the correctness of ‘deleting a subscription’	This test validates the correctness of ‘invalid parameters’ for subscription	This test validates the correctness for an ‘invalid authorization token’ for subscription It describes a test when the authentication has failed because the user has provided no-token, expired token or wrong token.
Methodology	Execute an API call that shall return a successful and verifiable response. Verify that output data is as per the test design.		
Test no.	093a	094a	094b
Response code	200	400	401

ANNEX A

(*Clause* 4.1)

POSTMAN COLLECTION FOR EXECUTING THE TESTS

The data dump for the postman collection for executing the tests are available here. (https://github.com/datakaveri/udx-compliance-specification/tree/main/resource-access-service/data-dump)

ANNEX B

(*Clause* 4.1)

COMPLEX SEARCH TEST CASE

This annexure presents an example to conduct a complex test based on the combination of tests in Group 0 and Group 1. Note that this is a separate test case and even if an implementation passes the tests individually it may not necessarily support functionality required by this complex search test case.

A complex test using G0 (Temporal: Between), G1 (Spatial: Circle) and G0 (Attribute) will have the parameters for a 'between' temporal search on a 'circle' spatial search along with an attribute search with '<' mathematical operator. This test is a combination of three test 001a as in **5.1** and 043a as in **5.9** and test 017a as in **5.5**. The parameters for this complex test are shown below:

An example of a complex test.	
Parameter	Parameter value
id	iisc.ac.in/89a36273d77dac4cf38114fca1bbe64392547f86/rs.iudx.io/pune-env-flood/FWR055
timerel	between
time	2020-10-18T14:20:00Z
endtime	2020-10-19T14:20:00Z
geoproperty	location
georel	near;maxdistance=10
geometry	Point
coordinates	[21.178,72.834]
offset	0
limit	10
q	reference level<15.0

ANNEX C

(*Clause* 5)

POSTMAN COLLECTION CONSISTING OF TEST CASES

The postman collection consisting of test cases are available here. (https://github.com/datakaveri/udx-compliance-specification/tree/main/resource-access-service/postman-collection)

ANNEX D

(*Foreword*)

COMMITTEE COMPOSITION

Smart Infrastructure Sectional Committee, LITD 28

Organization	*Representative(s)*
Indian Institute of Science, Bengaluru	SHRI INDER S. GOPAL **(*Chairperson*)**
Aveva Software Private Limited, Bengaluru	SHRI MSNR HARISH
Centre for Development of Telematics, New Delhi	SHRI AURINDAM BHATTACHARYA MS ANUPAMA CHOPRA (*Alternate*)
CyanConnode Private Limited, Bengaluru	SHRI MANISH WIDHANI SHRI DEEPAK NIMARE (*Alternate*)
eGovernments Foundation, Bengaluru	SHRI KRISHNAKUMAR THIAGARAJAN
ERNET India, New Delhi	DR A. PAVENTHAN SHRI HARI KRISHNA ATLURI (*Alternate*)
Esri India Technologies Private Limited, Noida	SHRI VIJAY KUMAR SHRI RUPESH KUMAR (*Alternate* I) MS SEEMA JOSHI (*Alternate* II)
IEEE India, Bengaluru	SHRI MUNIR MOHAMMED
India Smart Grid Forum, New Delhi	SHRI REJI KUMAR PILLAI
Ministry of Housing and Urban Affairs, New Delhi	SHRI KUNAL KUMAR SHRI PADAM VIJAY (*Alternate*)
National Smart Grid Mission, Ministry of Power, Gurugram	SHRI ARUN MISRA SHRIMATI KUMUD WADHWA (*Alternate* I) SHRI GYAN PRAKASH (*Alternate* II)
PHYTEC Embedded Private Limited, Bengaluru	SHRI B. VALLAB RAO (VASU)
Qualcomm India Private Limited, Bengaluru	DR PUNIT RATHOD DR VINOSH BABU JAMES (*Alternate*)
Renesas Electronics, Bengaluru	SHRI RAVINDRA CHATURVEDI SHRI SAURABH GOSWAMI (*Alternate*)
Seconded European Standardization Expert for India (SESEI), New Delhi	SHRI DINESH CHAND SHARMA
Secure Meters Limited, Gurugram	SHRI MADHUR KUMAR SRIVASTAVA SHRI PUNEET KHURANA (*Alternate* I) SHRI KAUSTUBH PATIL (*Alternate* II) SHRI UTTAM KOTDIYA (*Alternate* III) SHRI ANIL MEHTA (*Alternate* IV)

Organization	*Representative(s)*
Senra Tech Private Limited, New Delhi	SHRI DHIRAJ KUMAR SHRI ANKUSH KOCHHAR (*Alternate*)
Shrama Technologies Private Limited, Bangalore	SHRI AMARJEET KUMAR
Siemens Limited, Mumbai	SHRI RAVI MADIPADGA SHRI MANOJ BELGAONKAR (*Alternate* I) SHRI PRADEEP KAPOOR (*Alternate* II) SHRI VIKRAM GANDOTRA (*Alternate* III)
Standardization Testing and Quality Certification (STQC), Pune	MS LIPIKA KAUSHIK
Tata Consultancy Services Limited, Mumbai	SHRI RAMESH BALAJI SHRI DEBASHIS MITRA (*Alternate*)
Tata Consulting Engineers Limited, Navi Mumbai	SHRI JAGDISH SHIVRAJ SHIGE SHRI MANOJ KUMAR (*Alternate*)
Tejas Networks Limited, Bengaluru	DR KANWAR JIT SINGH
Telecommunication Engineering Center, New Delhi	MS ASHIMA SHRI SUSHIL KUMAR (*Alternate* I) SHRI UTTAM CHAND (*Alternate* II)
Telecommunications Standards Development Society India, New Delhi	MS BINDOO SRIVASTAVA
In Personal Capacity (*India*)	MS LAVANYA NUPUR
In Personal Capacity [(*IUDX, IISc*) *CV Raman Road, Bengaluru - 560012*]	SHRI VASANTH RAJARAMAN
BIS Director General	SHRIMATI REENA GARG, SCIENTIST 'G' AND HEAD (ELECTRONICS AND INFORMATION TECHNOLOGY) [REPRESENTING DIRECTOR GENERAL (*Ex-officio*)]

Member Secretary
SHRI DEVANSH DEOLEKAR
SCIENTIST 'D'/JOINT DIRECTOR
(ELECTRONICS AND INFORMATION TECHNOLOGY), BIS

Panel involved in the Finalization - LITD 28/P12 Data Exchange Architecture

Organization	*Representative(s)*
Indian Institute of Science, Bengaluru	SHRI ABHAY SHARMA **(*Convenor*)**
Indian Institute of Science, Bengaluru	SHRI VASANTH RAJARAMAN SHRI RAKSHIT RAMESH (*Alternate* I) SHRI MAHIDHAR CHELLAMANI (*Alternate* II)
International Institute of Information Technology, Bangalore	DR SRINATH SRINIVASA
International Institute of Information Technology, Hyderabad	SHRI ANURADHA VATTEM
Microsoft India, Noida	SHRI RAJESH KUMAR
NEC Technologies India Private Limited, New Delhi	SHRI ANAND SAHU
Siemens Limited, Mumbai	SHRI SABI SHAW
Tata Consultancy Services Limited, Mumbai	SHRI SANDEEP SAXENA

Bureau of Indian Standards

BIS is a statutory institution established under the *Bureau of Indian Standards Act*, 2016 to promote harmonious development of the activities of standardization, marking and quality certification of goods and attending to connected matters in the country.

Copyright

Review of Indian Standards

Amendments are issued to standards as the need arises on the basis of comments. Standards are also reviewed periodically; a standard along with amendments is reaffirmed when such review indicates that no changes are needed; if the review indicates that changes are needed, it is taken up for revision. Users of Indian Standards should ascertain that they are in possession of the latest amendments or edition by referring to the website-www.bis.gov.in or www.standardsbis.in.

This Indian Standard has been developed from Doc No.: LITD 28 (21132).

Amendments Issued Since Publication

Amend No.	Date of Issue	Text Affected

BUREAU OF INDIAN STANDARDS

Headquarters:

Manak Bhavan, 9 Bahadur Shah Zafar Marg, New Delhi 110002
Telephones: 2323 0131, 2323 3375, 2323 9402 *Website*: www.bis.gov.in

Regional Offices: *Telephones*

Central : 601/A, Konnectus Tower -1, 6th Floor, DMRC Building, Bhavbhuti Marg, New Delhi 110002 { 2323 7617

Eastern : 8th Floor, Plot No 7/7 & 7/8, CP Block, Sector V, Salt Lake, Kolkata, West Bengal 700091 { 2367 0012 2320 9474

Northern : Plot No. 4-A, Sector 27-B, Madhya Marg, Chandigarh 160019 { 265 9930

Southern : C.I.T. Campus, IV Cross Road, Taramani, Chennai 600113 { 2254 1442 2254 1216

Western : Manakalya, 4th Floor, NTH Complex (W Sector), F-10, MIDC, Andheri (East), Mumbai 400093 { 283 25838

Branches : AHMEDABAD, BENGALURU, BHOPAL, BHUBANESHWAR, CHANDIGARH, CHENNAI, COIMBATORE, DEHRADUN, DELHI, FARIDABAD, GHAZIABAD, GUWAHATI, HARYNA, HUBLI, HYDERABAD, JAIPUR, JAMMU & KASHMIR, JAMSHEDPUR, KOCHI, KOLKATA, LUCKNOW, MADURAI, MUMBAI, NAGPUR, NOIDA, PARWANOO, PATNA, PUNE, RAIPUR, RAJKOT, SURAT, VIJAYAWADA.

Published by BIS, New Delhi