

# ISO/IEC 19075-6:2021-08 (E)

## Information technology - Guidance for the use of database language SQL - Part 6: Support for JSON

---

Contents	Page
Foreword.....	ix
Introduction.....	xi
<b>1 Scope.....</b>	<b>1</b>
<b>2 Normative references.....</b>	<b>2</b>
<b>3 Terms and definitions.....</b>	<b>3</b>
<b>4 JavaScript Object Notation (JSON).....</b>	<b>6</b>
4.1 Context for JSON.....	6
4.2 What is JSON?.....	6
4.3 Representations of JSON data.....	7
4.3.1 Introduction to representations of JSON data.....	7
4.3.2 Avro.....	7
4.3.3 BSON.....	8
4.4 Schemas.....	8
4.4.1 JSON schemata and validity.....	8
4.4.2 Avro schemata.....	9
4.4.3 BSON schemata.....	9
4.5 Why does JSON matter in the context of SQL? What is JSON's relationship to NoSQL?.....	10
4.6 Use cases for JSON support in SQL.....	11
4.6.1 Introduction to the use cases.....	11
4.6.2 JSON data ingestion and storage.....	11
4.6.3 JSON data generation from relational data.....	11
4.6.4 Querying JSON as persistent semi-structured data model instances.....	12
4.7 What features address those use cases?.....	12
4.7.1 Addressing the use cases.....	12
4.7.2 Storing JSON data in an SQL table.....	12
4.7.3 Generating JSON in an SQL query.....	12
4.7.4 Querying JSON data in SQL tables using SQL.....	13
<b>5 The SQL/JSON data model.....</b>	<b>14</b>
5.1 Introduction to the SQL/JSON data model.....	14
5.2 SQL/JSON items.....	15
5.2.1 Definition of SQL/JSON items.....	15
5.2.2 Atomic values.....	16
5.2.3 SQL/JSON arrays.....	17
5.2.4 SQL/JSON objects.....	17
5.3 SQL/JSON sequences.....	17
5.4 Parsing JSON.....	18
5.5 Serializing JSON.....	18
<b>6 SQL/JSON functions.....</b>	<b>19</b>

6.1	Introduction to SQL/JSON functions. . . . .	19
6.2	Handle JSON using built-in functions. . . . .	19
6.3	JSON API common syntax. . . . .	19
6.3.1	Introduction to JSON API common syntax. . . . .	19
6.3.2	JSON value expression. . . . .	20
6.3.3	Path expression. . . . .	20
6.3.4	PASSING clause. . . . .	20
6.3.5	JSON output clause. . . . .	21
6.3.6	ON ERROR and ON EMPTY syntax. . . . .	21
6.4	Query functions. . . . .	22
6.4.1	The four query functions. . . . .	22
6.4.2	JSON_EXISTS. . . . .	22
6.4.3	JSON_VALUE. . . . .	25
6.4.4	JSON_QUERY. . . . .	29
6.4.5	JSON_TABLE. . . . .	33
6.4.5.1	Introduction to JSON_TABLE. . . . .	33
6.4.5.2	COLUMNS clause that is not nested. . . . .	34
6.4.5.3	Nested COLUMNS clause. . . . .	36
6.4.5.4	PLAN clause. . . . .	37
6.4.6	Conformance features for query operators. . . . .	41
6.5	Constructor functions and IS JSON predicate. . . . .	43
6.5.1	Tables used to illustrate constructor functions and the IS JSON predicate. . . . .	43
6.5.2	JSON_OBJECT. . . . .	44
6.5.3	JSON_OBJECTAGG. . . . .	45
6.5.4	JSON_ARRAY. . . . .	46
6.5.5	JSON_ARRAYAGG. . . . .	47
6.5.6	IS JSON predicate. . . . .	49
6.5.7	Handling of JSON nulls and SQL nulls. . . . .	49
6.5.8	Conformance features for constructor functions. . . . .	50
<b>7</b>	<b>SQL/JSON path language. . . . .</b>	<b>51</b>
7.1	Overview of SQL/JSON path language. . . . .	51
7.2	Objectives for the SQL/JSON path language. . . . .	52
7.3	Modes. . . . .	53
7.3.1	Introduction to modes. . . . .	53
7.3.2	Example of strict vs lax. . . . .	54
7.4	Lexical issues. . . . .	56
7.5	Syntax summary. . . . .	58
7.6	Formal semantics and notational conventions. . . . .	58
7.7	Primitive operations in formal semantics. . . . .	59
7.7.1	Concatenation. . . . .	59
7.7.2	unwrap(). . . . .	59
7.7.3	wrap(). . . . .	60
7.8	Mode declaration. . . . .	60
7.9	<JSON path primary>. . . . .	61
7.9.1	Introduction to JSON path primaries. . . . .	61
7.9.2	Literals. . . . .	61
7.9.3	Variables. . . . .	62

7.9.4	Parentheses. . . . .	63
7.10	Accessors. . . . .	63
7.10.1	Introduction to accessors. . . . .	63
7.10.2	Member accessor. . . . .	64
7.10.3	Member wildcard accessor. . . . .	67
7.10.4	Element accessor. . . . .	68
7.10.5	Element wildcard accessor. . . . .	70
7.10.6	Sequence semantics of the accessors. . . . .	71
7.11	Item methods. . . . .	71
7.11.1	Introduction to item methods. . . . .	71
7.11.2	type(). . . . .	72
7.11.3	size(). . . . .	72
7.11.4	Numeric item methods (double, ceiling, floor, abs). . . . .	73
7.11.5	datetime(). . . . .	73
7.11.6	keyvalue(). . . . .	73
7.12	Arithmetic expressions. . . . .	75
7.12.1	Introduction to arithmetic expressions. . . . .	75
7.12.2	Unary plus and minus. . . . .	75
7.12.3	Binary operations. . . . .	76
7.13	Filter expression. . . . .	76
7.13.1	Introduction to filter expressions. . . . .	76
7.13.2	true/false and <i>True/False</i> . . . . .	77
7.13.3	null and <i>Unknown</i> . . . . .	78
7.13.4	Error handling in filters. . . . .	78
7.13.5	Truth tables. . . . .	81
7.13.6	Comparison predicates. . . . .	82
7.13.7	like_regex predicate. . . . .	83
7.13.8	starts with predicate. . . . .	84
7.13.9	exists predicate. . . . .	84
7.13.10	is unknown predicate. . . . .	86
7.14	Conformance features for SQL/JSON path language. . . . .	87
	<b>Bibliography. . . . .</b>	<b>88</b>
	<b>Index. . . . .</b>	<b>89</b>

## Table

Page

1	JSON, SQL/JSON, and SQL (other than SQL/JSON values and counterparts) . . . . .	14
2	Parallels between JSON text and SQL/JSON data model. . . . .	18
3	JSON_EXISTS sample data. . . . .	22
4	Result of the sample query. . . . .	23
5	Accessor example. . . . .	24
6	Result 1. . . . .	26
7	Result 2. . . . .	26
8	Result 3. . . . .	27
9	Result 4. . . . .	27
10	Result 5. . . . .	28
11	Result 6. . . . .	29
12	JSON_EXISTS results. . . . .	30
13	ON EMPTY results. . . . .	31
14	ARRAY WRAPPER results. . . . .	31
15	Illustrating differences. . . . .	32
16	Comparison of wrapper options. . . . .	32
17	JSON_TABLE sample data in a book recommendation table. . . . .	33
18	Unnested query result. . . . .	35
19	Nested query result. . . . .	36
20	PLAN query result. . . . .	39
21	Second PLAN query result. . . . .	39
22	Third PLAN query result. . . . .	40
23	DEPTS table. . . . .	43
24	JOBS table. . . . .	43
25	EMPLOYEES table. . . . .	44
26	The JSON object returned. . . . .	45
27	Returned JSON object with the corresponding job sequence number. . . . .	46
28	ARRAYAGG query result. . . . .	48
29	Second ARRAYAGG query result. . . . .	48
30	Third ARRAYAGG query result. . . . .	49
31	Three aspects of path evaluation governed by modes. . . . .	54
32	Example of strict vs lax. . . . .	55
33	Features of the SQL/JSON path language. . . . .	58
34	Data used by unwrap() example. . . . .	59
35	Data used by wrap() example. . . . .	60
36	Examples of atomic values in the SQL/JSON path language . . . . .	61
37	Examples of the escaping rules. . . . .	61
38	Evaluation of '\$.phones.type' in lax mode. . . . .	65
39	Intermediate step. . . . .	65
40	Evaluation of '\$.phones[*].type' . . . . .	66
41	Evaluation of '\$.phones[*] ? (exists (@.type)).type' . . . . .	66
42	Evaluation of '\$.phones.*' in lax mode. . . . .	67
43	Evaluation of '\$.phones[*].*' . . . . .	67
44	Evaluation of 'lax \$.sensors.*[0, last, 2]' . . . . .	69
45	The step in the evaluation . . . . .	71
46	Result of the query with the sample data . . . . .	74
47	Evaluation of 'lax -\$.readings.floor()' . . . . .	75

48	Evaluation of 'lax (-\$.readings).floor()'	76
49	Table T with two rows	78
50	Computation on row K=102	79
51	Modified table T	79
52	Computation on row K=102 in modified table T	79
53	Computation of 'lax \$ ? (@.hours > 9)' on row K=102	80
54	Computation of 'strict \$ ? (@.hours > 9)' in strict mode	80
55	Result of &&	81
56	Result of	81
57	Result of !	81
58	Supported comparisons	82
59	Final result	83
60	A table with JSON column	85
61	Evaluation of 'strict \$ ? (exists (@.name)).name' on row K=201	85
62	Evaluation of 'strict \$ ? (exists (@.name)).name' on row K=202	85

## Figure

Page

1	Relationships between “JSON” and “SQL/JSON”	14
2	The SQL/JSON path language architecture	51