**[MySQL常用Json函数](https://www.cnblogs.com/waterystone/p/5626098.html)**

官方文档：[JSON Functions](http://dev.mysql.com/doc/refman/5.7/en/json-functions.html)

| **Name** | **Description** |
| --- | --- |
| [JSON\_APPEND()](http://dev.mysql.com/doc/refman/5.7/en/json-modification-functions.html#function_json-append) | Append data to JSON document |
| [JSON\_ARRAY()](http://dev.mysql.com/doc/refman/5.7/en/json-creation-functions.html#function_json-array) | Create JSON array |
| [JSON\_ARRAY\_APPEND()](http://dev.mysql.com/doc/refman/5.7/en/json-modification-functions.html#function_json-array-append) | Append data to JSON document |
| [JSON\_ARRAY\_INSERT()](http://dev.mysql.com/doc/refman/5.7/en/json-modification-functions.html#function_json-array-insert) | Insert into JSON array |
| [->](http://dev.mysql.com/doc/refman/5.7/en/json-search-functions.html#operator_json-column-path) | Return value from JSON column after evaluating path; equivalent to JSON\_EXTRACT(). |
| [JSON\_CONTAINS()](http://dev.mysql.com/doc/refman/5.7/en/json-search-functions.html#function_json-contains) | Whether JSON document contains specific object at path |
| [JSON\_CONTAINS\_PATH()](http://dev.mysql.com/doc/refman/5.7/en/json-search-functions.html#function_json-contains-path) | Whether JSON document contains any data at path |
| [JSON\_DEPTH()](http://dev.mysql.com/doc/refman/5.7/en/json-attribute-functions.html#function_json-depth) | Maximum depth of JSON document |
| [JSON\_EXTRACT()](http://dev.mysql.com/doc/refman/5.7/en/json-search-functions.html#function_json-extract) | Return data from JSON document |
| [->>](http://dev.mysql.com/doc/refman/5.7/en/json-search-functions.html#operator_json-inline-path) | Return value from JSON column after evaluating path and unquoting the result; equivalent to JSON\_UNQUOTE(JSON\_EXTRACT()). |
| [JSON\_INSERT()](http://dev.mysql.com/doc/refman/5.7/en/json-modification-functions.html#function_json-insert) | Insert data into JSON document |
| [JSON\_KEYS()](http://dev.mysql.com/doc/refman/5.7/en/json-search-functions.html#function_json-keys) | Array of keys from JSON document |
| [JSON\_LENGTH()](http://dev.mysql.com/doc/refman/5.7/en/json-attribute-functions.html#function_json-length) | Number of elements in JSON document |
| [JSON\_MERGE()](http://dev.mysql.com/doc/refman/5.7/en/json-modification-functions.html#function_json-merge) | Merge JSON documents |
| [JSON\_OBJECT()](http://dev.mysql.com/doc/refman/5.7/en/json-creation-functions.html#function_json-object) | Create JSON object |
| [JSON\_QUOTE()](http://dev.mysql.com/doc/refman/5.7/en/json-creation-functions.html#function_json-quote) | Quote JSON document |
| [JSON\_REMOVE()](http://dev.mysql.com/doc/refman/5.7/en/json-modification-functions.html#function_json-remove) | Remove data from JSON document |
| [JSON\_REPLACE()](http://dev.mysql.com/doc/refman/5.7/en/json-modification-functions.html#function_json-replace) | Replace values in JSON document |
| [JSON\_SEARCH()](http://dev.mysql.com/doc/refman/5.7/en/json-search-functions.html#function_json-search) | Path to value within JSON document |
| [JSON\_SET()](http://dev.mysql.com/doc/refman/5.7/en/json-modification-functions.html#function_json-set) | Insert data into JSON document |
| [JSON\_TYPE()](http://dev.mysql.com/doc/refman/5.7/en/json-attribute-functions.html#function_json-type) | Type of JSON value |
| [JSON\_UNQUOTE()](http://dev.mysql.com/doc/refman/5.7/en/json-modification-functions.html#function_json-unquote) | Unquote JSON value |
| [JSON\_VALID()](http://dev.mysql.com/doc/refman/5.7/en/json-attribute-functions.html#function_json-valid) | Whether JSON value is valid |

**1. 概述**

MySQL里的json分为json array和json object。 $表示整个json对象，在索引数据时用**下标**(对于json array，从0开始)或**键值**(对于json object，含有特殊字符的key要用"括起来，比如$."my name")。

例如：[3, {"a": [5, 6], "b": 10}, [99, 100]]，那么：

$[0]：3

$[1]： {"a": [5, 6], "b": 10}

$[2] ：[99, 100]

$[3] ： NULL

$[1].a：[5, 6]

$[1].a[1]：6

$[1].b：10

$[2][0]：99

**2. 比较规则**

json中的数据可以用 =, <, <=, >, >=, <>, !=, and <=> 进行比较。但json里的数据类型可以是多样的，那么在不同类型之间进行比较时，就有优先级了，**高优先级的要大于低优先级的**（可以用JSON\_TYPE()函数查看类型）。优先级从高到低如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | BLOB  BIT  OPAQUE  DATETIME  TIME  DATE  BOOLEAN  ARRAY  OBJECT  STRING  INTEGER, DOUBLE  NULL |

**3. 常用函数**

**3.1 创建函数**

**3.1.1 JSON\_ARRAY**

JSON\_ARRAY(val1,val2,val3...)

生成一个包含指定元素的json数组。

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | mysql> SELECT JSON\_ARRAY(1, "abc", NULL, TRUE, CURTIME());  +---------------------------------------------+  | JSON\_ARRAY(1, "abc", NULL, TRUE, CURTIME()) |  +---------------------------------------------+  | [1, "abc", null, true, "11:30:24.000000"]   |  +---------------------------------------------+ |

**3.1.2 JSON\_OBJECT**

JSON\_OBJECT(key1,val1,key2,val2...)

生成一个包含指定K-V对的json object。如果有key为NULL或参数个数为奇数，则抛错。

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | mysql> SELECT JSON\_OBJECT('id', 87, 'name', 'carrot');  +-----------------------------------------+  | JSON\_OBJECT('id', 87, 'name', 'carrot') |  +-----------------------------------------+  | {"id": 87, "name": "carrot"}            |  +-----------------------------------------+ |

3.1.3 JSON\_QUOTE

JSON\_QUOTE(json\_val)

将json\_val用"号括起来。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | mysql> SELECT JSON\_QUOTE('null'), JSON\_QUOTE('"null"');  +--------------------+----------------------+  | JSON\_QUOTE('null') | JSON\_QUOTE('"null"') |  +--------------------+----------------------+  | "null"             | "\"null\""           |  +--------------------+----------------------+  mysql> SELECT JSON\_QUOTE('[1, 2, 3]');  +-------------------------+  | JSON\_QUOTE('[1, 2, 3]') |  +-------------------------+  | "[1, 2, 3]"             |  +-------------------------+ |

**3.1.4 CONVERT**

CONVERT(json\_string,JSON)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | mysql> select CONVERT('{"mail": "amy@gmail.com", "name": "Amy"}',JSON);  +----------------------------------------------------------+  | CONVERT('{"mail": "amy@gmail.com", "name": "Amy"}',JSON) |  +----------------------------------------------------------+  | {"mail": "amy@gmail.com", "name": "Amy"}                 |  +----------------------------------------------------------+ |

**3.2 查询函数**

**3.2.1 JSON\_CONTAINS**

JSON\_CONTAINS(json\_doc, val[, path])

查询json文档是否在指定path包含指定的数据，包含则返回1，否则返回0。如果有参数为NULL或path不存在，则返回NULL。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | mysql> SET @j = '{"a": 1, "b": 2, "c": {"d": 4}}';  mysql> SET @j2 = '1';  mysql> SELECT JSON\_CONTAINS(@j, @j2, '$.a');  +-------------------------------+  | JSON\_CONTAINS(@j, @j2, '$.a') |  +-------------------------------+  |                             1 |  +-------------------------------+  mysql> SELECT JSON\_CONTAINS(@j, @j2, '$.b');  +-------------------------------+  | JSON\_CONTAINS(@j, @j2, '$.b') |  +-------------------------------+  |                             0 |  +-------------------------------+    mysql> SET @j2 = '{"d": 4}';  mysql> SELECT JSON\_CONTAINS(@j, @j2, '$.a');  +-------------------------------+  | JSON\_CONTAINS(@j, @j2, '$.a') |  +-------------------------------+  |                             0 |  +-------------------------------+  mysql> SELECT JSON\_CONTAINS(@j, @j2, '$.c');  +-------------------------------+  | JSON\_CONTAINS(@j, @j2, '$.c') |  +-------------------------------+  |                             1 |  +-------------------------------+ |

**3.2.2 JSON\_CONTAINS\_PATH**

JSON\_CONTAINS\_PATH(json\_doc, one\_or\_all, path[, path] ...)

查询是否存在指定路径，存在则返回1，否则返回0。如果有参数为NULL，则返回NULL。

one\_or\_all只能取值"one"或"all"，one表示只要有一个存在即可；all表示所有的都存在才行。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | mysql> SET @j = '{"a": 1, "b": 2, "c": {"d": 4}}';  mysql> SELECT JSON\_CONTAINS\_PATH(@j, 'one', '$.a', '$.e');  +---------------------------------------------+  | JSON\_CONTAINS\_PATH(@j, 'one', '$.a', '$.e') |  +---------------------------------------------+  |                                           1 |  +---------------------------------------------+  mysql> SELECT JSON\_CONTAINS\_PATH(@j, 'all', '$.a', '$.e');  +---------------------------------------------+  | JSON\_CONTAINS\_PATH(@j, 'all', '$.a', '$.e') |  +---------------------------------------------+  |                                           0 |  +---------------------------------------------+  mysql> SELECT JSON\_CONTAINS\_PATH(@j, 'one', '$.c.d');  +----------------------------------------+  | JSON\_CONTAINS\_PATH(@j, 'one', '$.c.d') |  +----------------------------------------+  |                                      1 |  +----------------------------------------+  mysql> SELECT JSON\_CONTAINS\_PATH(@j, 'one', '$.a.d');  +----------------------------------------+  | JSON\_CONTAINS\_PATH(@j, 'one', '$.a.d') |  +----------------------------------------+  |                                      0 |  +----------------------------------------+ |

**3.2.3 JSON\_EXTRACT**

JSON\_EXTRACT(json\_doc, path[, path] ...)

从json文档里抽取数据。如果有参数有NULL或path不存在，则返回NULL。如果抽取出多个path，则返回的数据封闭在一个json array里。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | mysql> SELECT JSON\_EXTRACT('[10, 20, [30, 40]]', '$[1]');  +--------------------------------------------+  | JSON\_EXTRACT('[10, 20, [30, 40]]', '$[1]') |  +--------------------------------------------+  | 20                                         |  +--------------------------------------------+  mysql> SELECT JSON\_EXTRACT('[10, 20, [30, 40]]', '$[1]', '$[0]');  +----------------------------------------------------+  | JSON\_EXTRACT('[10, 20, [30, 40]]', '$[1]', '$[0]') |  +----------------------------------------------------+  | [20, 10]                                           |  +----------------------------------------------------+  mysql> SELECT JSON\_EXTRACT('[10, 20, [30, 40]]', '$[2][\*]');  +-----------------------------------------------+  | JSON\_EXTRACT('[10, 20, [30, 40]]', '$[2][\*]') |  +-----------------------------------------------+  | [30, 40]                                      |  +-----------------------------------------------+ |

在MySQL 5.7.9+里可以用"**->**"替代。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | mysql> SELECT c, JSON\_EXTRACT(c, "$.id"), g       > FROM jemp       > WHERE JSON\_EXTRACT(c, "$.id") > 1       > ORDER BY JSON\_EXTRACT(c, "$.name");  +-------------------------------+-----------+------+  | c                             | c->"$.id" | g    |  +-------------------------------+-----------+------+  | {"id": "3", "name": "Barney"} | "3"       |    3 |  | {"id": "4", "name": "Betty"}  | "4"       |    4 |  | {"id": "2", "name": "Wilma"}  | "2"       |    2 |  +-------------------------------+-----------+------+  3 rows in set (0.00 sec)    mysql> SELECT c, c->"$.id", g       > FROM jemp       > WHERE c->"$.id" > 1       > ORDER BY c->"$.name";  +-------------------------------+-----------+------+  | c                             | c->"$.id" | g    |  +-------------------------------+-----------+------+  | {"id": "3", "name": "Barney"} | "3"       |    3 |  | {"id": "4", "name": "Betty"}  | "4"       |    4 |  | {"id": "2", "name": "Wilma"}  | "2"       |    2 |  +-------------------------------+-----------+------+  3 rows in set (0.00 sec) |

在MySQL 5.7.13+，还可以用"**->>**"表示去掉抽取结果的"号，下面三种效果是一样的：

* JSON\_UNQUOTE( JSON\_EXTRACT(column, path) )
* JSON\_UNQUOTE(column -> path)
* column->>path

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38 | mysql> SELECT \* FROM jemp WHERE g > 2;  +-------------------------------+------+  | c                             | g    |  +-------------------------------+------+  | {"id": "3", "name": "Barney"} |    3 |  | {"id": "4", "name": "Betty"}  |    4 |  +-------------------------------+------+  2 rows in set (0.01 sec)    mysql> SELECT c->'$.name' AS name      ->     FROM jemp WHERE g > 2;  +----------+  | name     |  +----------+  | "Barney" |  | "Betty"  |  +----------+  2 rows in set (0.00 sec)    mysql> SELECT JSON\_UNQUOTE(c->'$.name') AS name      ->     FROM jemp WHERE g > 2;  +--------+  | name   |  +--------+  | Barney |  | Betty  |  +--------+  2 rows in set (0.00 sec)    mysql> SELECT c->>'$.name' AS name      ->     FROM jemp WHERE g > 2;  +--------+  | name   |  +--------+  | Barney |  | Betty  |  +--------+  2 rows in set (0.00 sec) |

**3.2.4 JSON\_KEYS**

JSON\_KEYS(json\_doc[, path])

获取json文档在指定路径下的所有键值，返回一个json array。如果有参数为NULL或path不存在，则返回NULL。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | mysql> SELECT JSON\_KEYS('{"a": 1, "b": {"c": 30}}');  +---------------------------------------+  | JSON\_KEYS('{"a": 1, "b": {"c": 30}}') |  +---------------------------------------+  | ["a", "b"]                            |  +---------------------------------------+  mysql> SELECT JSON\_KEYS('{"a": 1, "b": {"c": 30}}', '$.b');  +----------------------------------------------+  | JSON\_KEYS('{"a": 1, "b": {"c": 30}}', '$.b') |  +----------------------------------------------+  | ["c"]                                        |  +----------------------------------------------+ |

**3.2.5 JSON\_SEARCH**

JSON\_SEARCH(json\_doc, one\_or\_all, search\_str[, escape\_char[, path] ...])

查询包含指定字符串的paths，并作为一个json array返回。如果有参数为NUL或path不存在，则返回NULL。

one\_or\_all："one"表示查询到一个即返回；"all"表示查询所有。

search\_str：要查询的字符串。 可以用LIKE里的'%'或‘\_’匹配。

path：在指定path下查。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127 | mysql> SET @j = '["abc", [{"k": "10"}, "def"], {"x":"abc"}, {"y":"bcd"}]';    mysql> SELECT JSON\_SEARCH(@j, 'one', 'abc');  +-------------------------------+  | JSON\_SEARCH(@j, 'one', 'abc') |  +-------------------------------+  | "$[0]"                        |  +-------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', 'abc');  +-------------------------------+  | JSON\_SEARCH(@j, 'all', 'abc') |  +-------------------------------+  | ["$[0]", "$[2].x"]            |  +-------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', 'ghi');  +-------------------------------+  | JSON\_SEARCH(@j, 'all', 'ghi') |  +-------------------------------+  | NULL                          |  +-------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '10');  +------------------------------+  | JSON\_SEARCH(@j, 'all', '10') |  +------------------------------+  | "$[1][0].k"                  |  +------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '10', NULL, '$');  +-----------------------------------------+  | JSON\_SEARCH(@j, 'all', '10', NULL, '$') |  +-----------------------------------------+  | "$[1][0].k"                             |  +-----------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '10', NULL, '$[\*]');  +--------------------------------------------+  | JSON\_SEARCH(@j, 'all', '10', NULL, '$[\*]') |  +--------------------------------------------+  | "$[1][0].k"                                |  +--------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '10', NULL, '$\*\*.k');  +---------------------------------------------+  | JSON\_SEARCH(@j, 'all', '10', NULL, '$\*\*.k') |  +---------------------------------------------+  | "$[1][0].k"                                 |  +---------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '10', NULL, '$[\*][0].k');  +-------------------------------------------------+  | JSON\_SEARCH(@j, 'all', '10', NULL, '$[\*][0].k') |  +-------------------------------------------------+  | "$[1][0].k"                                     |  +-------------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '10', NULL, '$[1]');  +--------------------------------------------+  | JSON\_SEARCH(@j, 'all', '10', NULL, '$[1]') |  +--------------------------------------------+  | "$[1][0].k"                                |  +--------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '10', NULL, '$[1][0]');  +-----------------------------------------------+  | JSON\_SEARCH(@j, 'all', '10', NULL, '$[1][0]') |  +-----------------------------------------------+  | "$[1][0].k"                                   |  +-----------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', 'abc', NULL, '$[2]');  +---------------------------------------------+  | JSON\_SEARCH(@j, 'all', 'abc', NULL, '$[2]') |  +---------------------------------------------+  | "$[2].x"                                    |  +---------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '%a%');  +-------------------------------+  | JSON\_SEARCH(@j, 'all', '%a%') |  +-------------------------------+  | ["$[0]", "$[2].x"]            |  +-------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '%b%');  +-------------------------------+  | JSON\_SEARCH(@j, 'all', '%b%') |  +-------------------------------+  | ["$[0]", "$[2].x", "$[3].y"]  |  +-------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '%b%', NULL, '$[0]');  +---------------------------------------------+  | JSON\_SEARCH(@j, 'all', '%b%', NULL, '$[0]') |  +---------------------------------------------+  | "$[0]"                                      |  +---------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '%b%', NULL, '$[2]');  +---------------------------------------------+  | JSON\_SEARCH(@j, 'all', '%b%', NULL, '$[2]') |  +---------------------------------------------+  | "$[2].x"                                    |  +---------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '%b%', NULL, '$[1]');  +---------------------------------------------+  | JSON\_SEARCH(@j, 'all', '%b%', NULL, '$[1]') |  +---------------------------------------------+  | NULL                                        |  +---------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '%b%', '', '$[1]');  +-------------------------------------------+  | JSON\_SEARCH(@j, 'all', '%b%', '', '$[1]') |  +-------------------------------------------+  | NULL                                      |  +-------------------------------------------+    mysql> SELECT JSON\_SEARCH(@j, 'all', '%b%', '', '$[3]');  +-------------------------------------------+  | JSON\_SEARCH(@j, 'all', '%b%', '', '$[3]') |  +-------------------------------------------+  | "$[3].y"                                  |  +-------------------------------------------+ |

**3.3 修改函数**

**3.3.1 JSON\_APPEND/JSON\_ARRAY\_APPEND**

JSON\_ARRAY\_APPEND(json\_doc, path, val[, path, val] ...)

在指定path的json array尾部追加val。如果指定path是一个json object，则将其封装成一个json array再追加。如果有参数为NULL，则返回NULL。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41 | mysql> SET @j = '["a", ["b", "c"], "d"]';  mysql> SELECT JSON\_ARRAY\_APPEND(@j, '$[1]', 1);  +----------------------------------+  | JSON\_ARRAY\_APPEND(@j, '$[1]', 1) |  +----------------------------------+  | ["a", ["b", "c", 1], "d"]        |  +----------------------------------+  mysql> SELECT JSON\_ARRAY\_APPEND(@j, '$[0]', 2);  +----------------------------------+  | JSON\_ARRAY\_APPEND(@j, '$[0]', 2) |  +----------------------------------+  | [["a", 2], ["b", "c"], "d"]      |  +----------------------------------+  mysql> SELECT JSON\_ARRAY\_APPEND(@j, '$[1][0]', 3);  +-------------------------------------+  | JSON\_ARRAY\_APPEND(@j, '$[1][0]', 3) |  +-------------------------------------+  | ["a", [["b", 3], "c"], "d"]         |  +-------------------------------------+    mysql> SET @j = '{"a": 1, "b": [2, 3], "c": 4}';  mysql> SELECT JSON\_ARRAY\_APPEND(@j, '$.b', 'x');  +------------------------------------+  | JSON\_ARRAY\_APPEND(@j, '$.b', 'x')  |  +------------------------------------+  | {"a": 1, "b": [2, 3, "x"], "c": 4} |  +------------------------------------+  mysql> SELECT JSON\_ARRAY\_APPEND(@j, '$.c', 'y');  +--------------------------------------+  | JSON\_ARRAY\_APPEND(@j, '$.c', 'y')    |  +--------------------------------------+  | {"a": 1, "b": [2, 3], "c": [4, "y"]} |  +--------------------------------------+    mysql> SET @j = '{"a": 1}';  mysql> SELECT JSON\_ARRAY\_APPEND(@j, '$', 'z');  +---------------------------------+  | JSON\_ARRAY\_APPEND(@j, '$', 'z') |  +---------------------------------+  | [{"a": 1}, "z"]                 |  +---------------------------------+ |

**3.3.2 JSON\_ARRAY\_INSERT**

JSON\_ARRAY\_INSERT(json\_doc, path, val[, path, val] ...)

 在path指定的json array元素插入val，原位置及以右的元素顺次右移。如果path指定的数据非json array元素，则略过此val；如果指定的元素下标超过json array的长度，则插入尾部。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | mysql> SET @j = '["a", {"b": [1, 2]}, [3, 4]]';  mysql> SELECT JSON\_ARRAY\_INSERT(@j, '$[1]', 'x');  +------------------------------------+  | JSON\_ARRAY\_INSERT(@j, '$[1]', 'x') |  +------------------------------------+  | ["a", "x", {"b": [1, 2]}, [3, 4]]  |  +------------------------------------+  mysql> SELECT JSON\_ARRAY\_INSERT(@j, '$[100]', 'x');  +--------------------------------------+  | JSON\_ARRAY\_INSERT(@j, '$[100]', 'x') |  +--------------------------------------+  | ["a", {"b": [1, 2]}, [3, 4], "x"]    |  +--------------------------------------+  mysql> SELECT JSON\_ARRAY\_INSERT(@j, '$[1].b[0]', 'x');  +-----------------------------------------+  | JSON\_ARRAY\_INSERT(@j, '$[1].b[0]', 'x') |  +-----------------------------------------+  | ["a", {"b": ["x", 1, 2]}, [3, 4]]       |  +-----------------------------------------+  mysql> SELECT JSON\_ARRAY\_INSERT(@j, '$[2][1]', 'y');  +---------------------------------------+  | JSON\_ARRAY\_INSERT(@j, '$[2][1]', 'y') |  +---------------------------------------+  | ["a", {"b": [1, 2]}, [3, "y", 4]]     |  +---------------------------------------+  mysql> SELECT JSON\_ARRAY\_INSERT(@j, '$[0]', 'x', '$[2][1]', 'y');  +----------------------------------------------------+  | JSON\_ARRAY\_INSERT(@j, '$[0]', 'x', '$[2][1]', 'y') |  +----------------------------------------------------+  | ["x", "a", {"b": [1, 2]}, [3, 4]]                  |  +----------------------------------------------------+ |

**3.3.3 JSON\_INSERT/JSON\_REPLACE/JSON\_SET**

JSON\_INSERT(json\_doc, path, val[, path, val] ...)

在指定path下插入数据，如果path已存在，则忽略此val（**不存在才插入**）。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | mysql> SET @j = '{ "a": 1, "b": [2, 3]}';  mysql> SELECT JSON\_INSERT(@j, '$.a', 10, '$.c', '[true, false]');  +----------------------------------------------------+  | JSON\_INSERT(@j, '$.a', 10, '$.c', '[true, false]') |  +----------------------------------------------------+  | {"a": 1, "b": [2, 3], "c": "[true, false]"}        |  +----------------------------------------------------+ |

JSON\_REPLACE(json\_doc, path, val[, path, val] ...)

替换指定路径的数据，如果某个路径不存在则略过（**存在才替换**）。如果有参数为NULL，则返回NULL。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | mysql> SET @j = '{ "a": 1, "b": [2, 3]}';  mysql> SELECT JSON\_REPLACE(@j, '$.a', 10, '$.c', '[true, false]');  +-----------------------------------------------------+  | JSON\_REPLACE(@j, '$.a', 10, '$.c', '[true, false]') |  +-----------------------------------------------------+  | {"a": 10, "b": [2, 3]}                              |  +-----------------------------------------------------+ |

JSON\_SET(json\_doc, path, val[, path, val] ...)

设置指定路径的数据（**不管是否存在**）。如果有参数为NULL，则返回NULL。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | mysql> SET @j = '{ "a": 1, "b": [2, 3]}';  mysql> SELECT JSON\_SET(@j, '$.a', 10, '$.c', '[true, false]');  +-------------------------------------------------+  | JSON\_SET(@j, '$.a', 10, '$.c', '[true, false]') |  +-------------------------------------------------+  | {"a": 10, "b": [2, 3], "c": "[true, false]"}    |  +-------------------------------------------------+  mysql> SELECT JSON\_INSERT(@j, '$.a', 10, '$.c', '[true, false]');  +----------------------------------------------------+  | JSON\_INSERT(@j, '$.a', 10, '$.c', '[true, false]') |  +----------------------------------------------------+  | {"a": 1, "b": [2, 3], "c": "[true, false]"}        |  +----------------------------------------------------+  mysql> SELECT JSON\_REPLACE(@j, '$.a', 10, '$.c', '[true, false]');  +-----------------------------------------------------+  | JSON\_REPLACE(@j, '$.a', 10, '$.c', '[true, false]') |  +-----------------------------------------------------+  | {"a": 10, "b": [2, 3]}                              |  +-----------------------------------------------------+ |

**3.3.4 JSON\_MERGE**

JSON\_MERGE(json\_doc, json\_doc[, json\_doc] ...)

merge多个json文档。规则如下：

* 如果都是json array，则结果自动merge为一个json array；
* 如果都是json object，则结果自动merge为一个json object；
* 如果有多种类型，则将非json array的元素封装成json array再按照规则一进行mege。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | mysql> SELECT JSON\_MERGE('[1, 2]', '[true, false]');  +---------------------------------------+  | JSON\_MERGE('[1, 2]', '[true, false]') |  +---------------------------------------+  | [1, 2, true, false]                   |  +---------------------------------------+  mysql> SELECT JSON\_MERGE('{"name": "x"}', '{"id": 47}');  +-------------------------------------------+  | JSON\_MERGE('{"name": "x"}', '{"id": 47}') |  +-------------------------------------------+  | {"id": 47, "name": "x"}                   |  +-------------------------------------------+  mysql> SELECT JSON\_MERGE('1', 'true');  +-------------------------+  | JSON\_MERGE('1', 'true') |  +-------------------------+  | [1, true]               |  +-------------------------+  mysql> SELECT JSON\_MERGE('[1, 2]', '{"id": 47}');  +------------------------------------+  | JSON\_MERGE('[1, 2]', '{"id": 47}') |  +------------------------------------+  | [1, 2, {"id": 47}]                 |  +------------------------------------+ |

**3.3.5 JSON\_REMOVE**

JSON\_REMOVE(json\_doc, path[, path] ...)  
移除指定路径的数据，如果某个路径不存在则略过此路径。如果有参数为NULL，则返回NULL。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | mysql> SET @j = '["a", ["b", "c"], "d"]';  mysql> SELECT JSON\_REMOVE(@j, '$[1]');  +-------------------------+  | JSON\_REMOVE(@j, '$[1]') |  +-------------------------+  | ["a", "d"]              |  +-------------------------+ |

**3.3.6 JSON\_UNQUOTE**

JSON\_UNQUOTE(val)

去掉val的引号。如果val为NULL，则返回NULL。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | mysql> SET @j = '"abc"';  mysql> SELECT @j, JSON\_UNQUOTE(@j);  +-------+------------------+  | @j    | JSON\_UNQUOTE(@j) |  +-------+------------------+  | "abc" | abc              |  +-------+------------------+  mysql> SET @j = '[1, 2, 3]';  mysql> SELECT @j, JSON\_UNQUOTE(@j);  +-----------+------------------+  | @j        | JSON\_UNQUOTE(@j) |  +-----------+------------------+  | [1, 2, 3] | [1, 2, 3]        |  +-----------+------------------+ |

**3.4 JSON特性查询**

**3.4.1 JSON\_DEEPTH**

 JSON\_DEPTH(json\_doc)

获取json文档的深度。如果参数为NULL，则返回NULL。

空的json array、json object或标量的深度为1。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | mysql> SELECT JSON\_DEPTH('{}'), JSON\_DEPTH('[]'), JSON\_DEPTH('true');  +------------------+------------------+--------------------+  | JSON\_DEPTH('{}') | JSON\_DEPTH('[]') | JSON\_DEPTH('true') |  +------------------+------------------+--------------------+  |                1 |                1 |                  1 |  +------------------+------------------+--------------------+  mysql> SELECT JSON\_DEPTH('[10, 20]'), JSON\_DEPTH('[[], {}]');  +------------------------+------------------------+  | JSON\_DEPTH('[10, 20]') | JSON\_DEPTH('[[], {}]') |  +------------------------+------------------------+  |                      2 |                      2 |  +------------------------+------------------------+  mysql> SELECT JSON\_DEPTH('[10, {"a": 20}]');  +-------------------------------+  | JSON\_DEPTH('[10, {"a": 20}]') |  +-------------------------------+  |                             3 |  +-------------------------------+ |

**3.4.2 JSON\_LENGTH**

JSON\_LENGTH(json\_doc[, path])

获取指定路径下的长度。如果参数为NULL，则返回NULL。

长度的计算规则：

* 标量的长度为1；
* json array的长度为元素的个数；
* json object的长度为key的个数。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | mysql> SELECT JSON\_LENGTH('[1, 2, {"a": 3}]');  +---------------------------------+  | JSON\_LENGTH('[1, 2, {"a": 3}]') |  +---------------------------------+  |                               3 |  +---------------------------------+  mysql> SELECT JSON\_LENGTH('{"a": 1, "b": {"c": 30}}');  +-----------------------------------------+  | JSON\_LENGTH('{"a": 1, "b": {"c": 30}}') |  +-----------------------------------------+  |                                       2 |  +-----------------------------------------+  mysql> SELECT JSON\_LENGTH('{"a": 1, "b": {"c": 30}}', '$.b');  +------------------------------------------------+  | JSON\_LENGTH('{"a": 1, "b": {"c": 30}}', '$.b') |  +------------------------------------------------+  |                                              1 |  +------------------------------------------------+ |

**3.4.3 JSON\_TYPE**

JSON\_TYPE(json\_val)

获取json文档的具体类型。如果参数为NULL，则返回NULL。

**3.4.4 JSON\_VALID**

JSON\_VALID(val)

判断val是否为有效的json格式，是为1，不是为0。如果参数为NUL，则返回NULL。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | mysql> SELECT JSON\_VALID('{"a": 1}');  +------------------------+  | JSON\_VALID('{"a": 1}') |  +------------------------+  |                      1 |  +------------------------+  mysql> SELECT JSON\_VALID('hello'), JSON\_VALID('"hello"');  +---------------------+-----------------------+  | JSON\_VALID('hello') | JSON\_VALID('"hello"') |  +---------------------+-----------------------+  |                   0 |                     1 |  +---------------------+-----------------------+ |