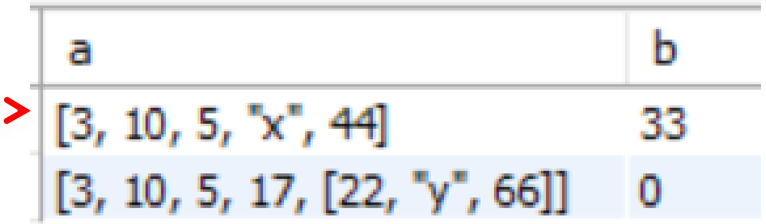
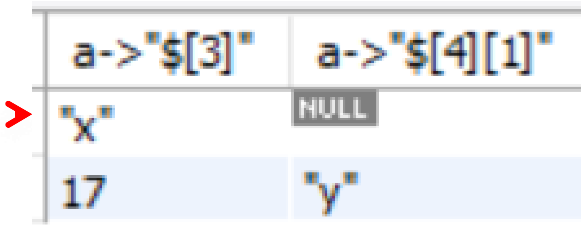
# MySQL中的JSON数据类型

```
create table  testJSON (a json, b int)
insert into  testJSON  values
          ( '[3, 10, 5, "x", 44]', 33 ),
          ( '[3, 10, 5, 17, [22, "y", 66]] ', 0 )
```



| a | b |
|---|---|
| [3, 10, 5, "x", 44] | 33 |
| [3, 10, 5, 17, [22, "y", 66]] | 0 |

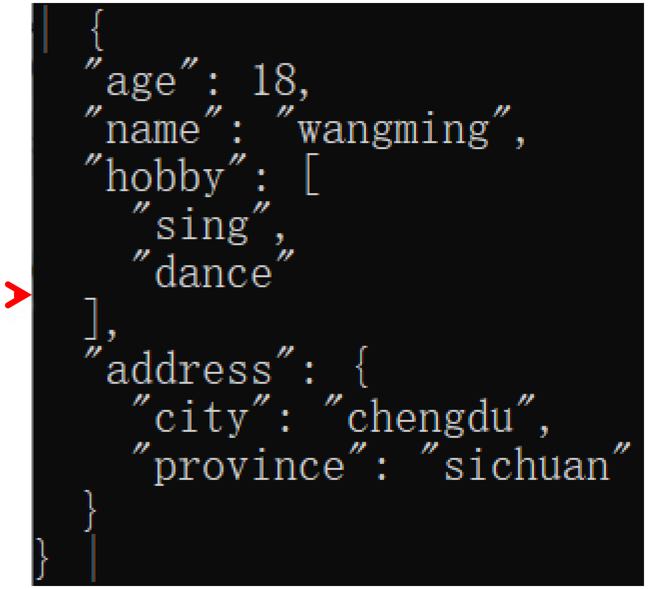
```
select  a->"$[3]", a->"$[4][1]"
from  testJSON;
```



| a->"$[3]" | a->"$[4][1]" |
|---|---|
| "x" | NULL |
| 17 | "y" |

# JSON的检索操作

```
create table  user (
        id  int not null primary key auto_increment,
        info  json );
insert into  user(info)  values (
        '{"name":"wangming",
        "age":18,
        "address":{"province":"sichuan","city":"chengdu"},
        "hobby" :["sing", "dance"]}' );
```

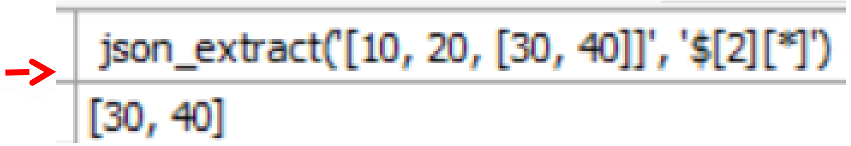


```
 {
  ″age″: 18,
  ″name″: ″wangming″,
  ″hobby″: [
    ″sing″,
    ″dance″
  ],
  ″address″: {
    ″city″: ″chengdu″,
    ″province″: ″sichuan″
  }
}
```

# JSON的检索操作

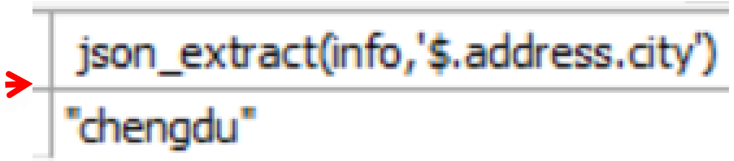
**json_extract** ( json_doc, path[, path] ... )

```
select   json_extract ( '[10, 20, [30, 40]]', '$[2][*]' );
```



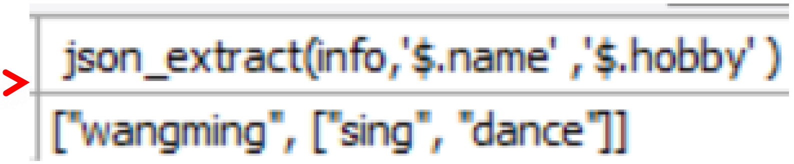
```
json_extract('[10, 20, [30, 40]]', '$[2][*]')
[30, 40]
```

```
select   json_extract ( info, '$.address.city' )
from     user;
```



```
json_extract(info,'$.address.city')
"chengdu"
```

```
select   json_extract ( info, '$.name’ , '$.hobby' )
from     user;
```

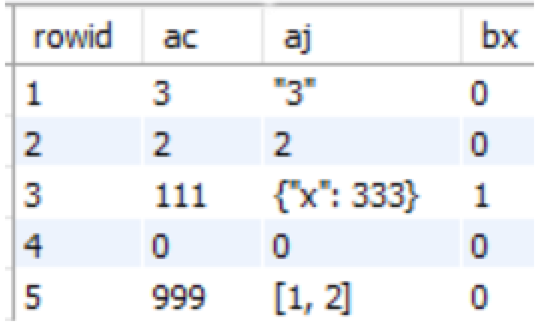


```
json_extract(info,'$.name' ,'$.hobby' )
["wangming", ["sing", "dance"]]
```

# 将JSON展开为平面表

**json_table**( expr, path **columns** (column_list) )

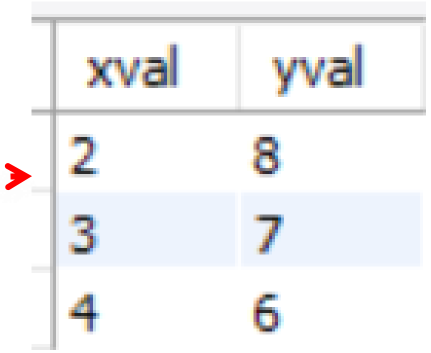
```
select  *
from    json_table( '[{"a":"3"},{"a":2},{"b":1},{"a":0},{"a":[1,2]}]’,
        "$[*]"   columns ( rowid  for ordinality,
                     ac  varchar(100)  path "$.a"  default  '111’ on empty  default  '999’  on error,
                      aj  json path  "$.a"  default  '{"x": 333}’  on empty,
                      bx  int  exists path  "$.b") )  as  tt;
```



| rowid | ac | aj | bx |
|---|---|---|---|
| 1 | 3 | "3" | 0 |
| 2 | 2 | 2 | 0 |
| 3 | 111 | {"x": 333} | 1 |
| 4 | 0 | 0 | 0 |
| 5 | 999 | [1, 2] | 0 |

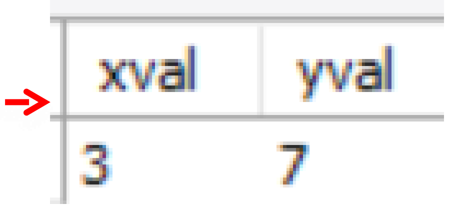
# 将JSON展开为平面表

```
select  *
from    json_table( '[{"x":2,"y":"8"},{"x":"3","y":"7"},{"x":"4","y":6}]',
        "$[*]"  columns ( rowid  for ordinality,
                    xval  varchar(100)  path "$.x“,
                    yval  varchar(100)  path "$.y) );
```



| xval | yval |
|---|---|
| 2 | 8 |
| 3 | 7 |
| 4 | 6 |

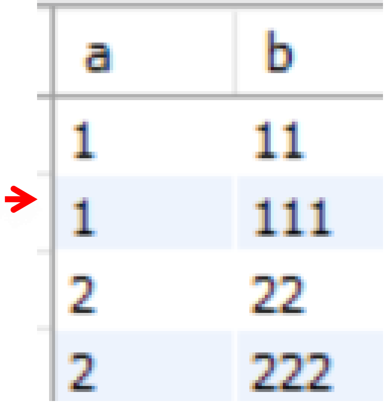
```
select  *
from    json_table( '[{"x":2,"y":"8"},{"x":"3","y":"7"},{"x":"4","y":6}]',
        "$[1]"  columns ( rowid  for ordinality,
                    xval  varchar(100)  path "$.x“,
                    yval  varchar(100)  path "$.y) );
```



| xval | yval |
|---|---|
| 3 | 7 |

# 将JSON展开为平面表：数组的unwind操作

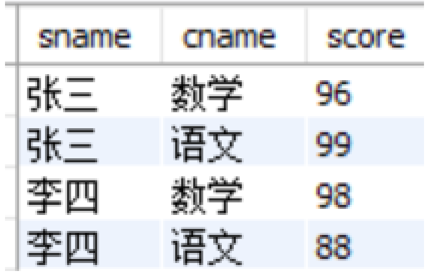
```
select  *
from   json_table('[ {"a": 1, "b": [11,111]}, {"a": 2, "b": [22,222]}, {"a":3}]',
          "$[*]"   columns(
                   a  int path  '$.a’,
                   nested path  '$.b[*]’  columns (b int path '$')
                   ) ) as  jt
where b  is not null;
```



| a | b |
|---|---|
| 1 | 11 |
| 1 | 111 |
| 2 | 22 |
| 2 | 222 |

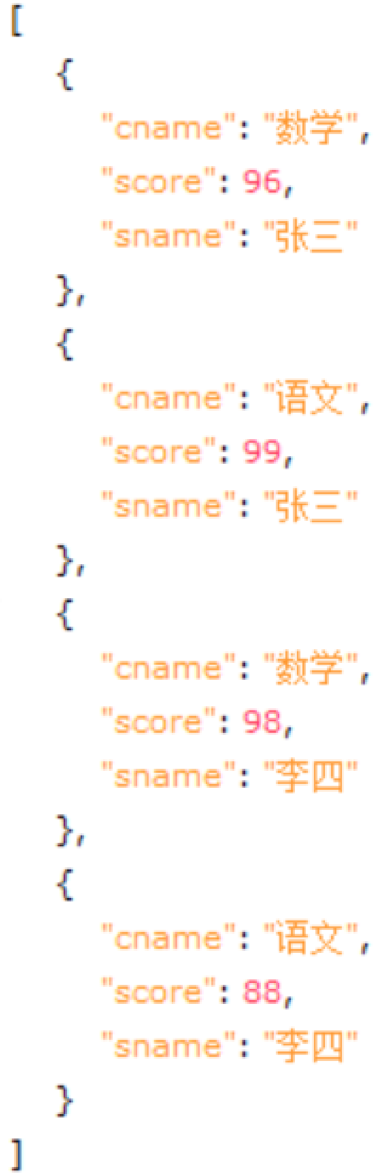
# 将平面表转为JSON

```
create table score(sname char(10), cname char(10), score int);
insert into score values ( '张三', '数学', '96' ),( '张三', '语文', '99’ ),
                         ( '李四', '数学', '98' ),( '李四', '语文',  '88’);
```



| sname | cname | score |
|---|---|---|
| 张三 | 数学 | 96 |
| 张三 | 语文 | 99 |
| 李四 | 数学 | 98 |
| 李四 | 语文 | 88 |

```
select concat ( '[',
        group_concat ( json_object (  'sname',sname,
                                      'cname',cname,
                                      'score', score) ),
        ']' ) as scores  from  score );
```



```
[
    {
        "cname": "数学",
        "score": 96,
        "sname": "张三"
    },
    {
        "cname": "语文",
        "score": 99,
        "sname": "张三"
    },
    {
        "cname": "数学",
        "score": 98,
        "sname": "李四"
    },
    {
        "cname": "语文",
        "score": 88,
        "sname": "李四"
    }
]
```