

# *Using JSON in Oracle 12c*





Presented by: John Jay King

Download this paper from: <http://www.kingtraining.com>



- Understand how Oracle 12c supports JSON
- Store JSON data in the database
- Become familiar with JSON conditionals and functions
- Query JSON data using JSON conditionals and functions
- Query JSON data using SQL via JSON\_TABLE



- John King – Partner, King Training Resources
- Oracle Ace Director 
- Member Oak Table Network 
- Providing training to Oracle and IT community for over 25 years – <http://www.kingtraining.com>
- “Techie” who knows Oracle, ADF, SQL, Java, and PL/SQL pretty well (along with many other topics)
- Member of AZORA, ODTUG, IOUG, and RMOUG



- JSON (JavaScript Object Notation) is a language-independent open data format ([www.json.org](http://www.json.org))
- Stores text in name-value pairs
- Originally in JavaScript, now also in: Java, R, .NET, PHP, Python, Node.js, and Oracle
- Most-often used for data interchange
- Frequently used to pass data to and from REST-style web services
- Becoming favored for big-data persistence



- 12c patch-set 2 (12.1.0.2) added JSON
- JSON documents are stored as VARCHAR2, CLOB, or BLOB data type
- JSON data works with all existing Oracle features including SQL and Analytics
- 12c supports path-based queries of JSON data stored in the database, JSON Path Language, and JSON Path Expressions
- JSON is used in SQL via SQL/JSON views
- JSON documents may be indexed



- JSON is text only, just like XML and thus is an excellent vehicle for data interchange
- JSON and XML are “human readable” and “self-describing” (sort of...)
- JSON and XML are hierarchical (data sets nested within data sets)
- JSON and XML offer validation capability; XML’s is more mature and capable today



- XML is verbose, JSON is shorter
- JSON has no end tags, required in XML
- JSON is quicker to read and write
- Reading XML documents requires “walking the DOM” – JSON does not
- JSON works more easily and is faster than XML when working with AJAX
- XML documents must be tested for “well-formed”-ness before processing



- Data is stored using comma-delimited name/value pairs
  - Field name/key (surrounded by double quotes)
  - Colon “:”,
  - Value (double quotes surround strings, numeric values unquoted, boolean true/false/null)
- Objects included inside curly braces “{“ & “}”

```
{ "lastName" : "King" }
```

- Arrays use brackets “[“ & “]” and commas

```
[ { "lastName" : "King" },  
  { "lastName" : "Manzo" } ]
```





```
<?xml version="1.0"?>
<myBooks>
  <book>
    <name>Learning XML</name>
    <author>Eric T. Ray</author>
    <publisher>O'Reilly</publisher>
  </book>
  <book>
    <name>XML Bible</name>
    <author>Elliotte Rusty Harold</author>
    <publisher>IDG Books</publisher>
  </book>
  <book>
    <name>XML by Example</name>
    <author>Sean McGrath</author>
  </book>
</myBooks>
```



```
{ "myBooks" :  
  [ { "book" : {  
      "name" : "Learning XML",  
      "author" : "Eric T. Ray",  
      "publisher" : "O'Reilly" }  
    },  
    { "book" : {  
      "name" : "XML Bible",  
      "author" : "Elliotte Rusty Harold",  
      "publisher" : "IDG Books" }  
    },  
    { "book" : {  
      "name" : "XML by Example",  
      "author" : "Sean McGrath",  
      "publisher" : "Prentice-Hall" }  
    }  
  ]  
}
```



- JSON documents are stored in the database using existing data types
  - VARCHAR2, CLOB and BLOB for character mode JSON
  - External JSON data sources (including those in HDFS file system) are accessible through external tables
- JSON data is available to SQL via relational views based upon JSON\_TABLE
- Oracle JSON documents may be indexed; JSON paths may use functional indexes



- JSON content is accessible from SQL and may be treated as JSON using:
  - JSON conditional operators
  - JSON functions
- JSON operators and functions use JSON Path language to navigate JSON objects



- JSON content is accessible from SQL via new condition operators
  - IS JSON                      Validate JSON, often in CHECK constraint and WHERE (IS / IS NOT)
  - JSON\_EXISTS                True if JSON path exists in document
  - JSON\_TEXTCONTAINS            True if text string is found in JSON property values (uses Oracle Text)



- JSON content is also accessed via new functions
  - JSON\_VALUE Used to query a scalar value from a JSON document
  - JSON\_QUERY Used to query all or part of a JSON document
  - JSON\_TABLE Used to query JSON document and create relational-style columns



- IS JSON returns TRUE if specified expression is a valid JSON document

```
**expr**      IS JSON | IS NOT JSON  
              FORMAT JSON  
              STRICT | LAX  
              WITH | WITHOUT UNIQUE KEYS
```

- IS true if valid JSON, IS NOT true if not JSON
- FORMAT JSON (only required for BLOB data)
- STRICT / LAX strictness level required for true
- WITH / WITHOUT if unique keys are required



```
create table deptj
(id raw(16) not null,
 dept_info clob constraint deptjson
           check (dept_info is json)
);
```

```
create table deptj
(id raw(16) not null,
 dept_info clob constraint deptjson
           check (dept_info is json strict)
);
```





```
select id,dept_info
from deptj
where dept_info is json
```

```
select id,dept_info
from deptj
where dept_info is json strict;
```

```
select id,dept_info
from deptj
where dept_info is json format json strict;
```



```

insert into deptj values
(sys_guid(),
'{"departments":{
  "DEPTNO": 10, "DNAME": "ACCOUNTING", "LOC": "NEW YORK",
  "deptemps": [
    { "EMPNO": 7782,
      "ENAME": "CLARK",
      "JOB": "MANAGER",
      "MGR": 7839,
      "HIREDATE": "09-JUN-81",
      "pay": {
        "SAL": 2450,
        "COMM": null},
      "DEPTNO": "10"
    },
    /* more */
  ]
}

```



```
select dept_info  
from deptj;
```

DEPT\_INFO

-----

```
{"departments": {  
  "DEPTNO": 10,  
  "DNAME": "ACCOUNTING",  
  "LOC": "NEW YORK",  
  "deptemps": [  
    {  
      "EMPNO": 7782,  
      "ENAME": "CLARK",  
      **** more ****
```



- \$.jsonpath Identifies path in JSON document
  - \$. (required)
  - Jsonpath (describes part of JSON document to be searched, if omitted entire document returned)
- JSON path used by JSON\_QUERY, JSON\_VALUE, JSON\_TABLE, JSON\_EXISTS, JSON\_TEXTCONTAINS



- JSON\_VALUE finds a value in a JSON document and returns it as VARCHAR2 or NUMBER to SQL

```
JSON_VALUE (expr | expr FORMAT JSON,  
            '$.jsonpath'  
            RETURNING VARCHAR2 (n) | NUMBER (n,n)  
            ERROR | NULL | DEFAULT xxx ON ERROR  
            )
```



```
select json_value(dept_info, '$.departments.DNAME')  
from deptj;
```

**DNAME**

-----

**ACCOUNTING**

**RESEARCH**

**SALES**

**OPERATIONS**



- JSON\_VALUE returns scalar values from JSON data

```
select json_value(dept_info
                 , '$.departments.DNAME')
from deptj;
```

```
select json_value(dept_info
                 , '$.departments.deptemps[0].ENAME')
from deptj;
```

```
select dept_info
       from deptj
       where json_value(dept_info,
                       '$[0].departments.DEPTNO') = '10'
```



- JSON\_TABLE maps JSON data into relational rows and columns

```
JSON_TABLE ( expr | expr FORMAT JSON, '$.jsonpath'
            ERROR | NULL | DEFAULT xxx ON ERROR
            COLUMNS (
                colname1 datatype
                    EXISTS PATH '$.jsonpath'
                    ERROR|NULL|DEFAULT xxx ON ERROR,
                colname2 datatype FORMAT JSON
                    jsonquerywrapper PATH '$.jsonpath'
                    ERROR|NULL|DEFAULT xxx ON ERROR,
                colname3 datatype PATH '$.jsonpath'
                    ERROR|NULL|DEFAULT xxx ON ERROR,
                NESTED PATH '$.jsonpath' COLUMNS (...),
                colname4 FOR ORDINALITY
            )
```





```

select dname,ename,job,sal
from deptj, json_table(dept_info,'$.departments'
    columns (dname varchar2(15) path '$.DNAME'
    ,nested path '$.deptemps[*]'
        columns (ename varchar2(20) path '$.ENAME'
        ,job varchar2(20) path '$.JOB'
        ,nested path '$.pay'
            columns (sal number path '$.SAL')
        )
    )
);

```

DNAME	ENAME	JOB	SAL
-----	-----	-----	-----
ACCOUNTING	CLARK	MANAGER	2450
ACCOUNTING	KING	PRESIDENT	5000
**** more ****			



- Oracle 12c support for JSON is timely and useful
- JSON is coming to an application near you soon
- JSON is:
  - Most-common mechanism for interacting with AJAX
  - Becoming most-common mechanism for web service data (especially REST/HTTP API based services)
  - Cornerstone of several "big data" data stores
- Take the time to learn JSON, JSON syntax, and Oracle's implementation

# RMOUG Training Days 2016

February 9-11, 2016

(Tuesday-Thursday)

Denver Convention Center



## Tracks

- Application Development
- Business Intelligence
- Database Administration
- DBA Deep Dive
- Database Tools of the Trade
- Hyperion
- Middleware
- Professional Empowerment

PHOTO CREDIT: Mike Landrum, SQL Developer and the "Data Tsunami" from i-Behavior

[www.rmoug.org](http://www.rmoug.org)



# COLLABORATE 16 – IOUG Forum

April 10 – 14, 2016

**Mandalay Bay  
Las Vegas, NV**



# ODTUG Kscope16

SAVE THE DATE  
CHICAGO, ILLINOIS  
JUNE 26-30  
[www.kscope16.com](http://www.kscope16.com)





## *Using JSON in Oracle 12c*

To contact the author:

**John King**

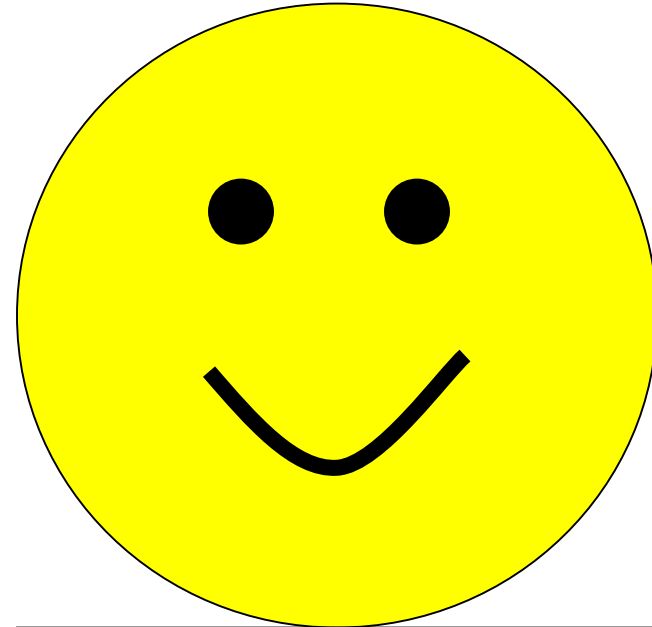
**King Training Resources**

P. O. Box 1780

Scottsdale, AZ 85252 USA

1.800.252.0652 - 1.303.798.5727

Email: [john@kingtraining.com](mailto:john@kingtraining.com)



**Thanks for your attention!**

Today's slides and examples are on the web:  
<http://www.kingtraining.com>



- End