113 DBMS_JSON

The `DBMS_JSON` package provides an interface for data-guide operations.

This chapter contains the following topics:

DBMS_JSON Overview

Package `DBMS_JSON` provides subprograms for manipulating JavaScript Object Notation (JSON) data that is stored in Oracle Database.

DBMS_JSON Security Model

`PUBLIC` is granted the `EXECUTE` privilege on package `DBMS_JSON`. Its subprograms execute with invoker's rights privileges.

DBMS_JSON Constants

The DBMS_JSON package uses these constants to define the JSON schema types and data-guide formatting options.

Table 113-1 DBMS_JSON Constants Defined for JSON Data-Guide Formatting

Name	Value	Description
FORMAT_FLAT	2	Display flat format

Table 113-1 (Cont.) DBMS_JSON Constants Defined for JSON Data-Guide Formatting

Name	Value	Description
FORMAT_HIERARCHICAL	1	Display hierarchical format. **Note:** If a field X in the JSON data has different data types in different documents (number/ boolean/string), FORMAT_HIERARCHICAL will propagate the type to the common denominator (usually string)
FORMAT_SCHEMA	3	Display JSON schema format **Note:** If a field X in the JSON data has different data types in different documents (number/ boolean/string), FORMAT_SCHEMA will list all encountered types in a JSON Array. Oracle recommends the use of FORMAT_SCHEMA if you want to use the generated schema for validation.
PRETTY	1	Use appropriate indention to improve readability

Table 113-2 DBMS_JSON Constants for JSON Schema Types

Name	Type	Value	Description
`TYPE_ARRAY`	NUMBER(2)	6	A JSON array
`TYPE_BOOLEAN`	NUMBER(2)	2	A JSON boolean
`TYPE_GEOJSON`	NUMBER(2)	7	Geographic JSON data
`TYPE_NULL`	NUMBER(2)	1	The `JSON NULL` value
`TYPE_NUMBER`	NUMBER(2)	3	A JSON number
`TYPE_OBJECT`	NUMBER(2)	5	A JSON object
`TYPE_STRING`	NUMBER(2)	4	A JSON string
`TYPE_BINARY`	NUMBER(2)	17	Oracle extended JSON type binary
`TYPE_DATE`	NUMBER(2)	13	Oracle extended JSON type date
`TYPE_DOUBLE`	NUMBER(2)	12	Oracle extended JSON type double
`TYPE_DSINTERVAL`	NUMBER(2)	16	Oracle extended JSON type day-second interval
`TYPE_FLOAT`	NUMBER(2)	11	Oracle extended JSON type float
`TYPE_TIMESTAMP`	NUMBER(2)	14	Oracle extended JSON type timestamp
`TYPE_YMINTERVAL`	NUMBER(2)	15	Oracle extended JSON type year-month interval

Table 113-3 DBMS_JSON Constants for mvrefreshmode Parameter

Name	Type	Value	Description
MV_REFRESH_ON_STATEMENT	NUMBER(2)	1	Creates the materialized view with refresh on statement.
MV_REFRESH_ON_COMMIT	NUMBER(2)	2	Creates the materialized view with refresh on commit.
MV_REFRESH_ON_DEMAND	NUMBER(2)	3	Creates the materialized view with refresh on demand.

See Also:

JSON Developer's Guide

Summary of DBMS_JSON Subprograms

This table lists the `DBMS_JSON` subprograms and briefly describes them.

DBMS_JSON Package Subprograms

Subprogram	Description
ADD_VIRTUAL_COLUMNS Procedure	Add virtual columns based on data-guide information. This has no effect when running on the shard catalog server — no virtual column is added.
CREATE_VIEW Procedure	Create a view with relational columns and scalar JSON fields as specified in a data guide.
CREATE_VIEW_ON_PATH Procedure	Create a view based on data-guide information, with relational columns, top-level scalar types, and fully expanded sub-tree under a given path. When running on the shard catalog server this raises an error stating that the data guide is empty.
DROP_VIRTUAL_COLUMNS Procedure	Drop virtual columns created by procedure `add_virtual_columns`. This has no effect when running on the shard catalog server.
GET_INDEX_DATAGUIDE Function	Get JSON data guide from a data guide-enabled JSON search index. When running on the shard catalog server this returns a single empty row as result.
GET_VIEW_SQL Function	Get the data definition language (DDL) statement for creating a view without actually creating the view.
JSON_TYPE_CONVERTIBLE_CHECK Procedure	Check whether existing data stored as JSON text can be migrated to the JSON data type.
RENAME_COLUMN Procedure	Set the preferred name for a view column or a virtual column creating using a data guide. This has no effect when running on the shard catalog server.

Note:

In the context of sharding, each individual shard maintains its own data-guide information, which is obtained from the JSON documents stored in that shard. When running on individual shard, procedures in this package that use data-guide information use only the information that is maintained for that shard.

ADD_VIRTUAL_COLUMNS Procedure

This procedure adds virtual columns based on the data guide.

The virtual column name is the value of `o:preferred_vc_name` in the data guide. The procedure ignores JSON objects, arrays, and fields under arrays in the data guide. Before it adds virtual columns, procedure `ADD_VIRTUAL_COLUMNS` first drops any existing virtual columns that were projected from fields in the same JSON column by a previous invocation of `ADD_VIRTUAL_COLUMNS` or by data-guide change-trigger procedure `add_vc` (in effect, it does what procedure `DBMS_JSON.DROP_VIRTUAL_COLUMNS` does).

See Also:

- DROP_VIRTUAL_COLUMNS Procedure
- *Oracle Database JSON Developer's Guide*

Syntax

```
DBMS_JSON.ADD_VIRTUAL_COLUMNS (
    tablename                IN  VARCHAR2,
    jcolname                 IN  VARCHAR2,
    dataguide                IN  CLOB,
    resolvenameconflicts     IN  BOOLEAN  DEFAULT TRUE,
    colnameprefix            IN VARCHAR2  DEFAULT NULL,
    mixedcasecolumns         IN BOOLEAN   DEFAULT FALSE);
```

For the following signature you must have a data guide-enabled search index on the JSON column. This is not needed for the previous signature.

```
DBMS_JSON.ADD_VIRTUAL_COLUMNS (
    tablename   IN  VARCHAR2,
    jcolname    IN  VARCHAR2,
    frequency       NUMBER    DEFAULT 0,
    hidden          BOOLEAN   DEFAULT FALSE);
```

Parameters

Table 113-4   ADD_VIRTUAL_COLUMNS Procedure Parameters

Parameter	Description
`tablename`	Name of the table containing JSON column `jcolname`.
`jcolname`	Name of the JSON column in table `tablename` that contains the data from which to create the virtual column.
`dataguide`	The data guide. When `o:hidden` in the data guide for a particular JSON field is set to `TRUE`, the corresponding virtual column is added as a hidden column. The default value of `o:hidden` is `FALSE`.
`resolvenameconflicts`	By default, this parameter is set to `TRUE`. The procedure automatically resolves the virtual column name conflicts by appending a sequence number. If it is set to `FALSE`, then in the event of any conflicts among `o:preferred_column_name`, an error is raised.
`colnameprefix`	By default, the virtual column name is the same as the JSON field name. This parameter allows you to add a prefix to the virtual column names.
`mixedcasecolumns`	By default, the virtual column names are case sensitive. If this parameter value is set to `FALSE`, the virtual column names become non-case-sensitive.

Table 113-4 (Cont.) ADD_VIRTUAL_COLUMNS Procedure Parameters

Parameter	Description
`frequency`	Sets the minimum frequency threshold to display JSON columns. A frequency of 0 means display all JSON columns. Also, all JSON columns are displayed if statistics have not been collected, effectively overriding any value set by this parameter.
`hidden`	`TRUE` means the added virtual column is hidden; `FALSE` means it is not. The default is `FALSE`.

Usage Notes

Procedure `DBMS_STATS.GATHER_STATS` collects statistics in the data guide. If the frequency statistic has not been collected, frequency is `NULL`. Setting the frequency to a value greater than zero means do not include columns for which there are no frequency statistics collected (statistic is `NULL`), unless `DBMS_STATS.GATHER_STATS` has never been executed. In that case, the frequency parameter is ignored and all columns are displayed in the view.

CREATE_VIEW Procedure

This procedure creates a view with relational columns, using scalar JSON fields as specified in the data guide. A data guide-enabled JSON search index is not required for this procedure; the data guide is passed to the procedure.

See Also:

Oracle Database JSON Developer's Guide

Syntax

```
PROCEDURE CREATE_VIEW (
   viewname                VARCHAR2,
   tablename               VARCHAR2,
   jcolname                VARCHAR2,
   dataguide               CLOB,
   resourcepath            VARCHAR2 DEFAULT NULL,
   materialize             BOOLEAN  DEFAULT FALSE,
   mvrefreshmode           NUMBER   DEFAULT MV_REFRESH_ON_STATEMENT,
   path                    VARCHAR2 DEFAULT '$',
   resolvenameconflicts    BOOLEAN  DEFAULT TRUE,
   colnameprefix           VARCHAR2 DEFAULT NULL,
   mixedcasecolumns        BOOLEAN DEFAULT FALSE);
```

Parameters

Table 113-5 DBMS_JSON.CREATE_VIEW Procedure Parameters

Parameter	Description
`viewname`	Name of the view.
`tablename`	Name of the table containing JSON column `jcolname`.

Table 113-5 (Cont.) DBMS_JSON.CREATE_VIEW Procedure Parameters

Parameter	Description
`jcolname`	Name of the JSON column in table `tablename` that is used to create the view.
`dataguide`	The data guide.
`resourcepath`	This parameter is for internal use. Value of this parameter is always `NULL`.
`materialize`	The value of this parameter is boolean and indicates if the view is materialized or not.
`mvrefreshmode`	When materialize is true, this parameter specifies the materialized view refresh mode. For more information on materialized view refresh mode options, see DBMS_JSON Constants.
`path`	The path of the `JSON` field to be expanded. It uses `JSON` path-expression syntax. It expands the descendants under the specified path, and creates view columns for each scalar value in the resulting sub-tree. The path $ creates a view starting from the `JSON` document root.
`resolvenameconflicts`	By default, this parameter is set to `TRUE`. The procedure automatically resolves the virtual column name conflicts by appending a sequence number. If it is set to `FALSE`, then in the event of any conflicts among `o:preferred_column_name`, an error is raised.
`colnameprefix`	By default, the view column name is the same as the `JSON` field name. This parameter allows users to provide a prefix to prepend to the view column names.
`mixedcasecolumns`	By default, the view column names are case sensitive. You can use this parameter to change the case sensitivity behavior of the view column names.

CREATE_VIEW_ON_PATH Procedure

This procedure creates a view with relational columns, using top-level scalar values and the scalar values in the expanded sub-tree under a given path. The JSON column must have a data guide-enabled search index.

See Also:

Oracle Database JSON Developer's Guide

Syntax

```
PROCEDURE CREATE_VIEW_ON_PATH(
   viewname VARCHAR2,
   tablename VARCHAR2,
   jcolname VARCHAR2,
   path VARCHAR2,
   frequency NUMBER DEFAULT 0);
```

Parameters

Table 113-6 CREATE_VIEW_ON_PATH Procedure Parameters

Parameter	Description
`viewname`	Name of the view.
`tablename`	Name of the table containing JSON column `jcolname`.
`jcolname`	Name of the JSON column in table `tablename` that is used to create the view. The column must have a data guide-enabled JSON search index, or else an error is raised.
`path`	The path of the JSON field to be expanded. It uses JSON path-expression syntax. It expands the descendants under the specified path, and creates view columns for each scalar value in the resulting sub-tree. The path `$` creates a view starting from the JSON document root.
`frequency`	The minimum frequency threshold for displaying the JSON columns. A frequency of 0 means display all JSON columns. All JSON columns are also displayed if statistics have not been collected, effectively overriding any value set by this parameter. The view only displays JSON fields with frequency greater than the given `frequency`. It does not display JSON fields added after collecting statistics if the given frequency is greater than 0, if their statistic columns are `NULL`.

DROP_VIRTUAL_COLUMNS Procedure

Drop all virtual columns that were added using PL/SQL procedure `DBMS_JSON.add_virtual_columns` or using data-guide change-trigger procedure `add_vc`.

See Also:

- ADD_VIRTUAL_COLUMNS Procedure
- *Oracle Database JSON Developer's Guide*

Syntax

```
PROCEDURE DROP_VIRTUAL_COLUMNS(
   tablename VARCHAR2,
   jcolname VARCHAR2);
```

Parameters

Table 113-7 DBMS_JSON.DROP_VIRTUAL_COLUMNS Procedure Parameters

Parameter	Description
`tablename`	Name of the table containing JSON column `jcolname`.
`jcolname`	Name of the JSON column in table `tablename`.

GET_INDEX_DATAGUIDE Function

`GET_INDEX_DATAGUIDE` gets JSON data guide from data guide-enabled JSON search index.

See Also:

Oracle Database JSON Developer's Guide

Syntax

```
FUNCTION GET_INDEX_DATAGUIDE(
   tablename VARCHAR2,
   jcolname VARCHAR2,
   format NUMBER,
   pretty NUMBER DEFAULT 0)
   RETURN CLOB;
```

Parameters

Table 113-8 DBMS_JSON.GET_INDEX_DATAGUIDE Procedure Parameters

Parameter	Description
`tablename`	Name of the table containing JSON column `jcolname`.
`jcolname`	Name of the JSON column in table `tablename` that has a data guide-enabled JSON search index.
`format`	The data-guide format: • `FORMAT_HIERARCHICAL` — hierarchical format • `FORMAT_FLAT` — flat format
`pretty`	A value of `DBMS_JSON.PRETTY` means pretty-print the data guide, using indention to improve readability.

Example 113-1 Example Get Data Guide in Hierarchical Pretty Format

This example returns the data guide in hierarchical format.

```
SELECT DBMS_JSON.GET_INDEX_DATAGUIDE('T1', 'PO',
DBMS_JSON.FORMAT_HIERARCHICAL, DBMS_JSON.PRETTY)
FROM DUAL;
```

GET_VIEW_SQL Function

This function returns the creating view DDL without actually creating the view. A data guide-enabled JSON search index is not required for this function; the data guide is passed to the function.

See Also:

Oracle Database JSON Developer's Guide

Syntax

```
FUNCTION GET_VIEW_SQL (
   viewname                VARCHAR2,
   tablename               VARCHAR2,
   jcolname                VARCHAR2,
   dataguide               CLOB,
   materialize             BOOLEAN  DEFAULT FALSE,
   mvrefreshmode           NUMBER   DEFAULT MV_REFRESH_ON_STATEMENT,
   path                    VARCHAR2 DEFAULT '$',
   resolvenameconflicts    BOOLEAN  DEFAULT TRUE,
   colnameprefix           VARCHAR2 DEFAULT NULL,
   mixedcasecolumns        BOOLEAN DEFAULT TRUE)
   RETURN CLOB;
```

Parameters

Table 113-9 DBMS_JSON.GET_VIEW_SQL Function Parameters

Parameter	Description
`viewname`	Name of the view.
`tablename`	Name of the table containing JSON column `jcolname`.
`jcolname`	Name of the JSON column in table `tablename` that is used to create the view.
`dataguide`	The data guide.
`materialize`	The value of this parameter is Boolean and indicates if the view is materialized or not.
`mvrefreshmode`	When materialize is true, this parameter specifies the materialized view refresh mode. For more information on materialized view refresh mode options, see DBMS_JSON Constants.
`path`	The path of the `JSON` field to be expanded. It uses `JSON` path-expression syntax. It expands the descendants under the specified path, and creates view columns for each scalar value in the resulting sub-tree. The path $ creates a view starting from the `JSON` document root.
`resolvenameconflicts`	By default, this parameter is set to `TRUE`. The function automatically resolves the virtual column name conflicts by appending a sequence number. If it is set to `FALSE`, then in the event of any conflicts among `o:preferred_column_name`, an error is raised.

Table 113-9 (Cont.) DBMS_JSON.GET_VIEW_SQL Function Parameters

Parameter	Description
`colnameprefix`	By default, the view column name is the same as the `JSON` field name. This parameter allows users to provide a prefix to prepend to the view column names.
`mixedcasecolumns`	By default, the view column names are case sensitive. You can use this parameter to change the case sensitivity behavior of the view column names.

Usage Notes

- If Wide Tables are enabled for the database (`MAX_COLUMNS=EXTENDED`):
  - When `viewname` is `NULL`, the function returns only the select statement of the view DDL and it can select more than 4096 columns.
  - When `viewname` is not `NULL`, the function returns create view DDL and it selects at most 4096 columns.
  - As one `json_table` can only produce at most 4096 columns, the function will split paths into joins among multiple `json_tables` if the paths are more than 4096, when `viewname` is `NULL`.
- If Wide Tables are not enabled for the database (`MAX_COLUMNS=STANDARD`):
  - When `viewname` is `NULL`, the function returns only the select statement of the view DDL and it can select more than 1000 columns.
  - When `viewname` is not `NULL`, the function returns create view DDL and it selects at most 1000 columns.
  - As one `json_table` can only produce at most 1000 columns, the function will split paths into joins among multiple `json_tables` if the paths are more than 1000, when `viewname` is `NULL`.

See Also:

Oracle Database Reference for more information on the `MAX_COLUMNS` initialization parameter

JSON_TYPE_CONVERTIBLE_CHECK Procedure

This procedure checks whether existing data stored as JSON text can be migrated to the JSON data type.

See Also:

Oracle Database JSON Developer's Guide

Syntax

```
PROCEDURE JSON_TYPE_CONVERTIBLE_CHECK (
   owner                    VARCHAR2,
   tableName                VARCHAR2,
   columnName               VARCHAR2,
   statusTableName          VARCHAR2,
   fastCheck                BOOLEAN DEFAULT FALSE,
   appendStatus             BOOLEAN DEFAULT FALSE);
```

Parameters

Table 113-10 DBMS_JSON.JSON_TYPE_CONVERTIBLE_CHECK Procedure Parameters

Parameter	Description
`owner`	Name of the owner of the table.
`tableName`	Name of the table.
`columnName`	Name of the column in the table `tableName` that contains data to convert to the JSON type.
`statusTableName`	Name of the table to use to add the tracking status of the operation. This table might already exist or it might need to be created. If it already exists, the procedure verifies that it complies with the expected shape.
`fastCheck`	The value of this optional parameter is Boolean. If this parameter is set to `TRUE`, the `is_json` check constraint is run to verify that the input data is convertible. If the parameter is `FALSE`, then the `oson` constructor is run to verify that the input data is convertible. The default value is `FALSE`.
`appendStatus`	The value of this optional parameter is Boolean. If this parameter is set to `TRUE`, the status table will not be truncated. If the parameter is `FALSE`, then if the status table already exists, it will be truncated. In either case, if the status table does not already exist, it is created and will then contain only new data from running the procedure. The default value is `FALSE`.

RENAME_COLUMN Procedure

This procedure sets the preferred name for a JSON column, to be used by the create view, or add virtual columns procedure.

See Also:

Oracle Database JSON Developer's Guide

Syntax

```
PROCEDURE RENAME_COLUMN(
   tablename VARCHAR2,
   jcolname VARCHAR2,
   path VARCHAR2,
   type NUMBER,
   preferred_name VARCHAR2);
```

Parameters

Table 113-11   RENAME_COLUMN Procedure Parameters

Parameter	Description
`tablename`	Name of the table containing JSON column `jcolname`.
`jcolname`	Name of the JSON column in table `tablename`. It must have a data guide-enabled JSON search index, or else an error is raised.
`path`	Path to the JSON field on which to set the preferred column name.
`type`	The type of the JSON field targeted by `path`. Two JSON fields can have the same path if they are of different types. Possible values: • `TYPE_NULL` • `TYPE_STRING` • `TYPE_NUMBER` • `TYPE_BOOLEAN` • `TYPE_OBJECT` • `TYPE_ARRAY`
`preferred_name`	Preferred name for the JSON field specified by `path`. If there is a name conflict, a system generated name is used instead.

Example 113-2   Example Renaming a Column

This example renames a field to item_name.

```
EXEC DBMS_JSON.RENAME_COLUMN('T1', 'PO', '$.purchaseOrder.items.name',
DBMS_JSON.TYPE_STRING, 'item_name');
```