

JSON Files

Scala

Java

Python

R

SQL

```
CREATE TEMPORARY VIEW jsonTable
USING org.apache.spark.sql.json
OPTIONS (
  path "examples/src/main/resources/people.json"
)

SELECT * FROM jsonTable
```

Data Source Option

Data source options of JSON can be set via:

- the .option/.options methods of
 - DataFrameReader
 - DataFrameWriter
 - DataStreamReader
 - DataStreamWriter
- the built-in functions below
 - from_json
 - to_json
 - schema_of_json
- OPTIONS clause at [CREATE TABLE USING DATA SOURCE](#)

Property Name	Default	Meaning	Scope
timeZone	(value of spark.sql.session.timeZone configuration)	Sets the string that indicates a time zone ID to be used to format timestamps in the JSON datasources or partition values. The following formats of timeZone are supported: <ul style="list-style-type: none">Region-based zone ID: It should have the form 'area/city', such as 'America/Los_Angeles'.Zone offset: It should be in the format '(+ -)HH:mm', for example '-08:00' or '+01:00'. Also 'UTC' and 'Z' are supported as aliases of '+00:00'. Other short names like 'CST' are not recommended to use because they can be ambiguous.	read/write
primitivesAsString	false	Infers all primitive values as a string type.	read
prefersDecimal	false	Infers all floating-point values as a decimal type. If the values do not fit in decimal, then it infers them as doubles.	read
allowComments	false	Ignores Java/C++ style comment in JSON records.	read
allowUnquotedFieldNames	false	Allows unquoted JSON field names.	read
allowSingleQuotes	true	Allows single quotes in addition to double quotes.	read

»

allowNumericLeadingZero	false	Allows leading zeros in numbers (e.g. 00012).	read
allowBackslashEscapingAnyCharacter	false	Allows accepting quoting of all character using backslash quoting mechanism.	read
mode	PERMISSIVE	<p>Allows a mode for dealing with corrupt records during parsing.</p> <ul style="list-style-type: none">PERMISSIVE: when it meets a corrupted record, puts the malformed string into a field configured by <code>columnNameOfCorruptRecord</code>, and sets malformed fields to <code>null</code>. To keep corrupt records, an user can set a string type field named <code>columnNameOfCorruptRecord</code> in an user-defined schema. If a schema does not have the field, it drops corrupt records during parsing. When inferring a schema, it implicitly adds a <code>columnNameOfCorruptRecord</code> field in an output schema.DROPMALFORMED: ignores the whole corrupted records. This mode is unsupported in the JSON built-in functions.FAILFAST: throws an exception when it meets corrupted records.	read
columnNameOfCorruptRecord	(value of <code>spark.sql.columnNameOfCorruptRecord</code> configuration)	Allows renaming the new field having malformed string created by PERMISSIVE mode. This overrides <code>spark.sql.columnNameOfCorruptRecord</code> .	read
dateFormat	yyyy-MM-dd	Sets the string that indicates a date format. Custom date formats follow the formats at datetime pattern . This applies to date type.	read/write
timestampFormat	yyyy-MM-dd'T'HH:mm:ss[.SSS][XXX]	Sets the string that indicates a timestamp format. Custom date formats follow the formats at datetime pattern . This applies to timestamp type.	read/write
timestampNTZFormat	yyyy-MM-dd'T'HH:mm:ss[.SSS]	Sets the string that indicates a timestamp without timezone format. Custom date formats follow the formats at Datetime Patterns . This applies to timestamp without timezone type, note that zone-offset and time-zone components are not supported when writing or reading this data type.	read/write
multiLine	false	Parse one record, which may span multiple lines, per file. JSON built-in functions ignore this option.	read
allowUnquotedControlChars	false	Allows JSON Strings to contain unquoted control characters (ASCII characters with value less than 32, including tab and line feed characters) or not.	read

»

encoding	Detected automatically when multiLine is set to true (for reading), UTF-8 (for writing)	For reading, allows to forcibly set one of standard basic or extended encoding for the JSON files. For example UTF-16BE, UTF-32LE. For writing, Specifies encoding (charset) of saved json files. JSON built-in functions ignore this option.	read/write
lineSep	\r, \r\n, \n (for reading), \n (for writing)	Defines the line separator that should be used for parsing. JSON built-in functions ignore this option.	read/write
samplingRatio	1.0	Defines fraction of input JSON objects used for schema inferring.	read
dropFieldIfAllNull	false	Whether to ignore column of all null values or empty array/struct during schema inference.	read
locale	en-US	Sets a locale as language tag in IETF BCP 47 format. For instance, locale is used while parsing dates and timestamps.	read
allowNonNumericNumbers	true	Allows JSON parser to recognize set of “Not-a-Number” (NaN) tokens as legal floating number values. <ul style="list-style-type: none">• +INF: for positive infinity, as well as alias of +Infinity and Infinity.• -INF: for negative infinity, alias - Infinity.• NaN: for other not-a-numbers, like result of division by zero.	read
compression	(none)	Compression codec to use when saving to file. This can be one of the known case-insensitive shorten names (none, bzip2, gzip, lz4, snappy and deflate). JSON built-in functions ignore this option.	write
ignoreNullFields	(value of spark.sql.jsonGenerator.ignoreNullFields configuration)	Whether to ignore null fields when generating JSON objects.	write

Other generic options can be found in [Generic File Source Options](#).