

List of Examples

3-1	Creating a JavaScript Module in the Database	3-4
3-2	Create a Call Specification for a Public Function	3-6
3-3	Public and Private Functions in a JavaScript Module	3-7
3-4	Providing JavaScript Source Code Using a BFILE	3-8
3-5	Providing JavaScript Source Code Using a CLOB	3-8
3-6	Specification of a VERSION string in CREATE MLE MODULE	3-9
3-7	Addition of JSON Metadata to the MLE Module	3-10
3-8	Drop an MLE Module	3-10
3-9	Drop an MLE Module Using IF EXISTS	3-10
3-10	Alter an MLE Module	3-10
3-11	Externalize JavaScript Module Source Code	3-12
3-12	Find MLE Modules Defined in a Schema	3-13
3-13	Map Identifier to JavaScript Module	3-16
3-14	Import Module Functionality	3-17
3-15	List Available MLE Environments Using USER_MLE_ENVS	3-21
3-16	List Module Import Information Using USER_MLE_ENV_IMPORTS	3-21
4-1	Using the Q-Quote Operator to Provide JavaScript Code Inline with PL/SQL	4-2
4-2	Loading JavaScript code from a BFILE with DBMS_LOB.LOADCLOBFROMFILE()	4-3
4-3	Loading JavaScript Code from a BFILE by Referencing an MLE Module from DBMS_MLE	4-5
4-4	Returning the Result of the Last Execution	4-6
5-1	Use an MLE Environment to Map an Import Name to a Module	5-2
5-2	Function Export using Named Exports	5-3
5-3	Function Export Using Export Keyword Inline	5-4
5-4	Export a Class Using a Default Export	5-4
5-5	Named Export of Single Function	5-5
5-6	Module Object Definition	5-6
5-7	Named Imports Using Specified Identifiers	5-6
5-8	Named Imports with Aliases	5-7
5-9	Default Import	5-7
5-10	Default Import with Built-in Module	5-7
6-1	Creating MLE Call Specifications	6-2
6-2	Simple Inline MLE Call Specification	6-8
6-3	Inline MLE Call Specification Returning JSON	6-9
6-4	Execution Context Dependencies	6-13
6-5	Show JavaScript Call Specification Metadata	6-15
6-6	OUT and IN OUT Parameters with JavaScript	6-16

7-1	Getting Started with the MLE JavaScript SQL Driver	7-3
7-2	Use Global Variables to Simplify SQL Execution	7-5
7-3	Selecting Data Using Direct Fetch: Arrays	7-7
7-4	Selecting Data Using Direct Fetch: Objects	7-8
7-5	Fetching Rows Using a ResultSet	7-9
7-6	Using the Iterable Protocol with ResultSets	7-10
7-7	Updating a Row Using the MLE JavaScript SQL Driver	7-11
7-8	Using Named Bind Variables	7-13
7-9	Using Positional Bind Variables	7-14
7-10	Using the RETURNING INTO Clause	7-15
7-11	Performing a Batch Operation	7-17
7-12	Calling PL/SQL from JavaScript	7-18
7-13	SQL Error Handling Inside a JavaScript Function	7-21
7-14	Error Handling Using JavaScript throw() Command	7-23
7-15	Inserting JSON Data into a Database Table	7-25
7-16	Use JavaScript to Manipulate JSON Data	7-28
7-17	Inserting a CLOB into a Table	7-30
7-18	Read an LOB	7-31
7-19	Using JavaScript Native Data Types vs Using Wrapper Types	7-35
7-20	Overriding the Global oracledb.fetchAsPlsqlWrapper Property	7-36
8-1	SODA with MLE JavaScript General Workflow	8-6
8-2	Opening an Existing Document Collection	8-9
8-3	Fetching All Existing Collection Names	8-10
8-4	Filtering the List of Returned Collections	8-10
8-5	Dropping a Collection	8-11
8-6	Creating SODA Documents	8-13
8-7	Inserting a SODA Document into a Collection	8-14
8-8	Inserting an Array of Documents into a Collection	8-15
8-9	Finding a Document by Key	8-17
8-10	Looking up Documents Using Multiple Keys	8-18
8-11	Using a QBE to Filter Documents in a Collection	8-19
8-12	Using skip() and limit() in a Pagination Query	8-20
8-13	Specifying Document Versions	8-21
8-14	Counting the Number of Documents Found	8-21
8-15	Replacing a Document in a Collection and Returning the Result Document	8-23
8-16	Removing a Document from a Collection Using a Document Key	8-24
8-17	Removing JSON Documents from a Collection Using a Filter	8-24

8-18	Creating a B-Tree Index for a JSON Field with SODA for In-Database JavaScript	8-25
8-19	Creating a JSON Search Index with SODA for In-Database JavaScript	8-26
8-20	Dropping an Index with SODA for In-Database JavaScript	8-27
8-21	Generating a Data Guide for a Collection	8-27
8-22	Use SODA for In-Database JavaScript	8-29
9-1	JSON Template for Specifying Debugpoints	9-2
9-2	JSON Template for Specifying Watch Action	9-3
9-3	JSON Template for Specifying Snapshot Action	9-3
9-4	Watching a Variable in an MLE Module	9-4
9-5	Enabling Debugging of an MLE Module	9-5
9-6	Obtain Textual Representation of Debug Output	9-7
9-7	Throwing ORA-04161 Error and Querying the Stack Trace	9-11
9-8	Redirect stdout to CLOB and DBMS_OUTPUT for MLE Module	9-14
9-9	Redirect stdout to CLOB and DBMS_OUTPUT for Dynamic MLE	9-15
10-1	Runtime State Isolation Scenario	10-6
10-2	Using Bind Variables Rather than String Concatenation	10-11
10-3	Use DBMS_ASSERT to Verify Valid Input	10-11
10-4	Using Bind Variables Rather than String Concatenation	10-15
10-5	Use DBMS_ASSERT to Verify Valid Input	10-15
10-6	Business Logic Stored in MLE Modules	10-17
10-7	Generic Data Processing Libraries	10-18
10-8	Use Generic Libraries in Business Logic	10-19
A-1	Use VECTOR Data Type with MLE	A-7