

**IBM z/OS Connect (OpenAPI 2.0)**

# **Developing RESTful APIs for IMS DVM Services**



**IBM Z  
Wildfire Team –  
Washington System Center**

*Lab Version Date: April 23, 2022*

# Table of Contents

<b>Overview .....</b>	<b>3</b>
<b>Create Data Virtualization Manger services .....</b>	<b>4</b>
<i>Use the Data Virtualization Manager Studio to create a virtual table .....</i>	<i>4</i>
<b>Use the Data Virtualization Manager Studio to test SQL commands .....</b>	<b>24</b>
<b>Use the Data Virtualization Manager Studio to create a web service .....</b>	<b>30</b>
<i>Use the Data Virtualization Manager Studio to deploy the services .....</i>	<i>45</i>
<b>Create z/OS Connect APIs .....</b>	<b>50</b>
<i>Connect to a z/OS Connect Server.....</i>	<i>50</i>
<i>Create the IMS DVM API Project .....</i>	<i>53</i>
<i>Import the SAR files generated by the DVM Studio .....</i>	<i>55</i>
<i>Compose an API for the IMS DVM Rest Services.....</i>	<i>58</i>
<i>Deploy the API to a z/OS Connect Server.....</i>	<i>66</i>
<i>Test the IMS APIs using Swagger UI .....</i>	<i>68</i>
<i>Test the IMS APIs using Postman .....</i>	<i>84</i>

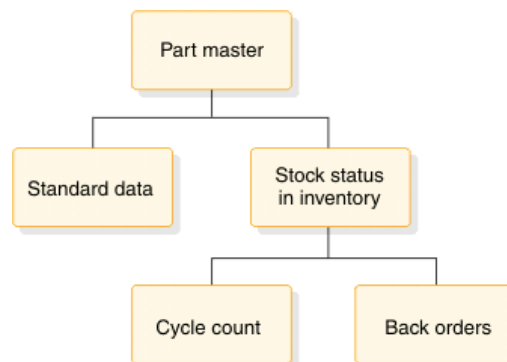
## Overview

The objective of these exercise is to gain experience using Data Virtualization Manager (DVM) Studio and the z/OS Connect API Toolkit to create RESTful API to IMS data bases.

For information about scheduling this workshop in your area contact your IBM representative.

## General Exercise Information and Guidelines

- ✓ This exercise requires using z/OS user identity *USER1*. The password for this user will be provided by the lab instructor.
- ✓ Do not hesitate to request assistance anytime you have any questions about the use of the Data Virtualization Manager Studio, IBM z/OS Explorer, z/OS Connect Toolkit features or other tools.
- ✓ The DL/I data base being used for this exercise is the sample part data base provided by the IMS sample application (see data base segment hierarchy below). For details of the data base and how to run the IMS sample, see URLs <https://www.ibm.com/docs/en/ims/15.1.0?topic=ivp-ims-sample-application> and <https://www.ibm.com/docs/en/ims/15.1.0?topic=application-running-ims-sample>.



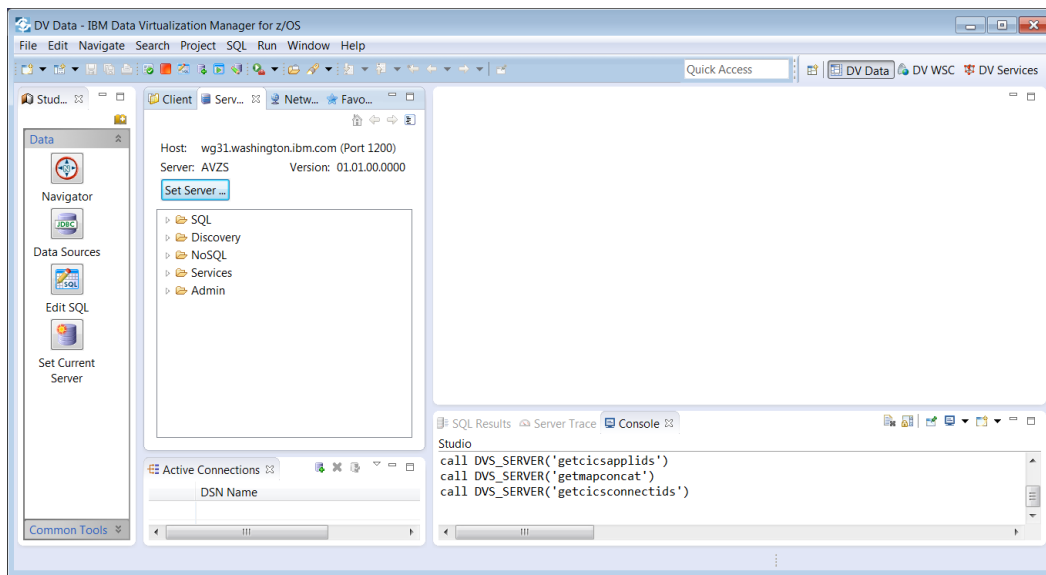
- ✓ Please note that there may be minor differences between the screen shots in this exercise versus what you see on your desktop. These differences should not impact the completion of this exercise.
- ✓ Text in **bold** and highlighted in **yellow** in this document should be available for copying and pasting in a file named *Development APIs for DVM CopyPaste* file on the desktop.

## Create Data Virtualization Manger services

### *Use the Data Virtualization Manager Studio to create a virtual table*

Access to an IMS data base using DVM SQL commands requires the creation of a DVM virtual table. The virtual table represents the layout or contents of the segments in the data base. In this section, two virtual tables will be created, one for the Part Master (PARTROOT) segment and one for the Stock Status inventory segment (STOKSTAT)/

1. On the workstation desktop, locate the Data Virtualization Manager Studio icon and double click on it to open the tool. You should automatically be connected to the DVM server running on z/OS, see below.



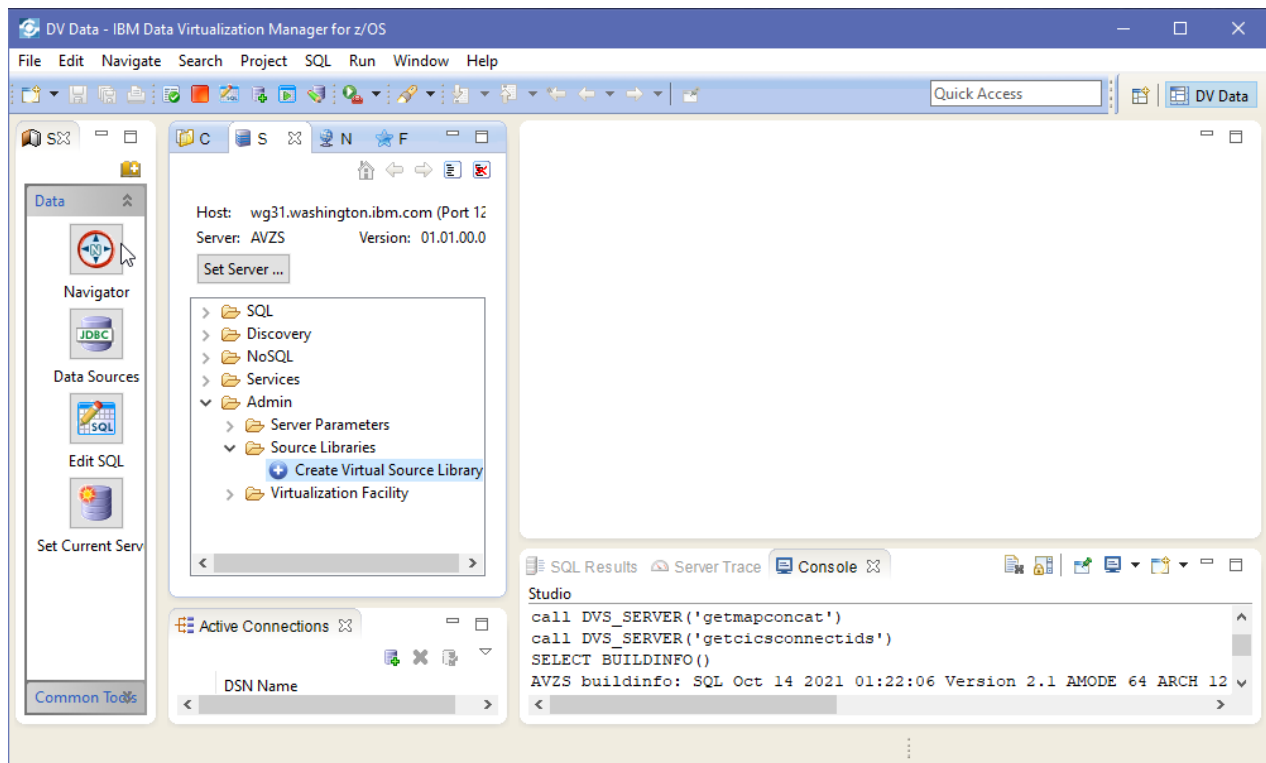
**Tech-Tip:** Eclipse based development tools like DVM Studio; provide a graphical interface consisting of multiple views within a single window.

A view is an area in the window dedicated to providing a specific tool or function. For example, in the window above, *Console*, *Studio Navigator* and *Server*, are views that use different areas of the window for displaying information. At bottom on the right there is a single area for displaying the contents of three views stacked together (commonly called a *stacked views*), *Console*, *SQL Results* and *Server Trace*. In a stacked view, the contents of each view can be displayed by clicking on the view tab (the name of the view).

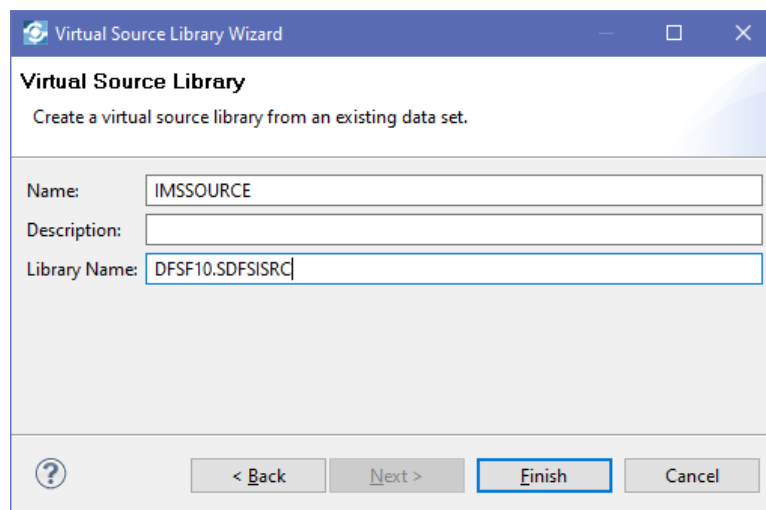
At any time, a specific view can be enlarged to fill the entire window by double clicking in the view's title bar. Double clicking in the view's title bar will be restored the original arrangement. If a DVM Studio view is closed or otherwise disappears, the original arrangement can be restored by selecting Windows → Reset Perspective in the window's tool bar.

Eclipse based tools also can display multiple views based on the current role of the user. In this context, a window is known as a perspective. The contents (or views) of a perspective are based on the role the user, i.e., developer or administrator.

2. In the *Server* view, expand *Admin* then expand *Source Libraries* to display the *Create Virtual Source Library* wizard, see below.

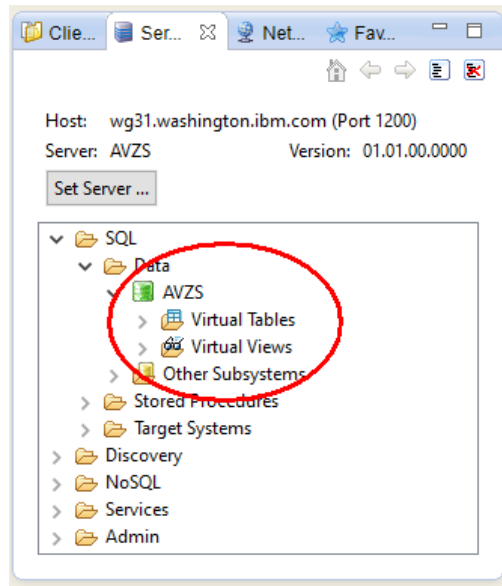


3. Double click the *Create Virtual Source Library* wizard and on the *Select a wizard* window select *Data Set* and click **Next** to continue.
4. On the *Virtual Source Library* enter **IMSSOURCE** as the *Name* of the virtual source library and **DFS10.SDFSISRC** as the *Library Name* and press **Finish** to continue.

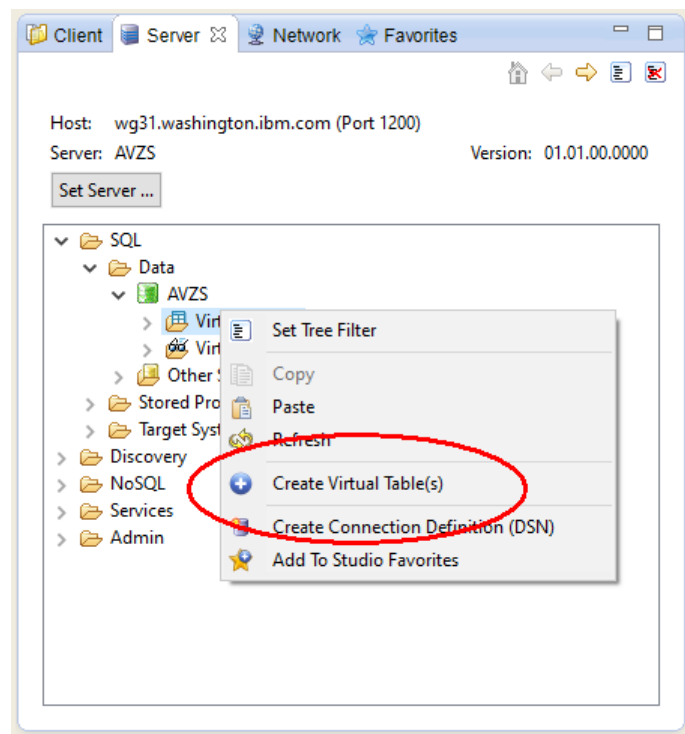


**Important:** The value used for names, e.g., IMSSOURCE, are somewhat arbitrary, but they do relate to later tasks. If you use the values and cases as supplied then in subsequent windows, results, etc. will be consistent with what is shown in this document.

- \_\_\_ 5. Next expand the *SQL* folder, then the *Data* folder and then the *AVZS* folder to display *Virtual Tables* and *Virtual Views*, see below.

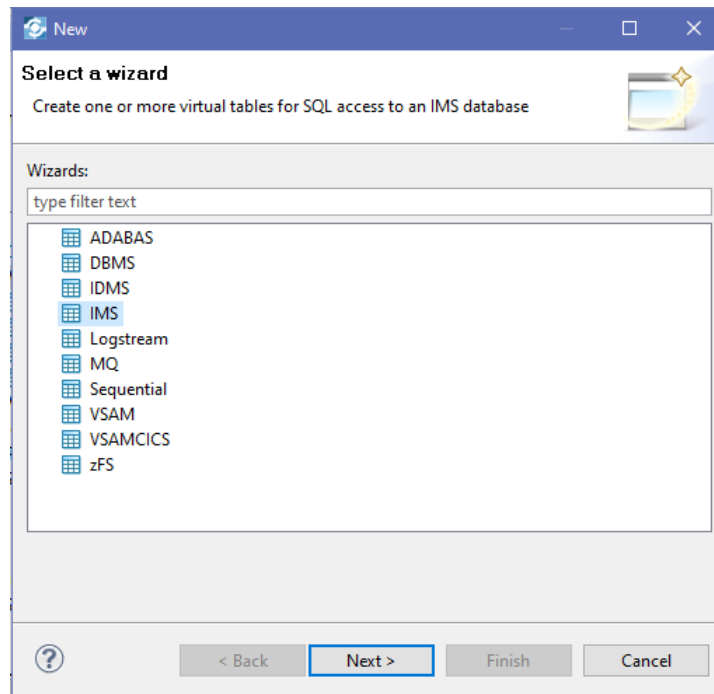


- \_\_\_ 6. Select the *Virtual Tables* folder and right mouse button click and then select the *Create Virtual Table(s)* option, see below.



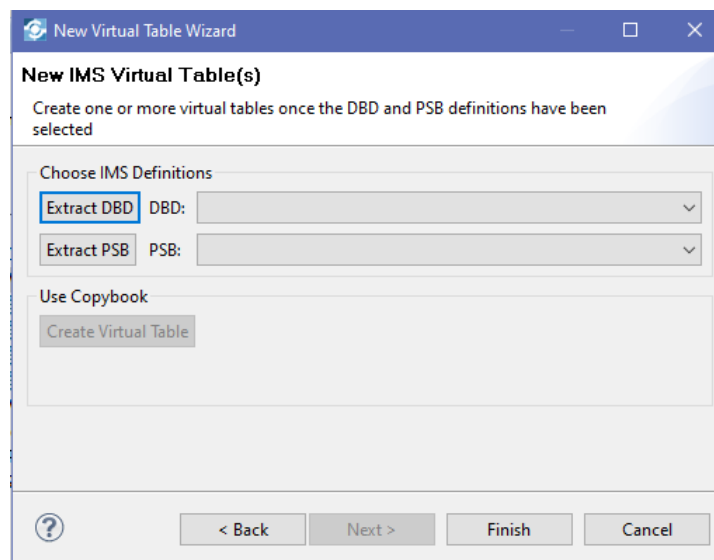
**Tech-Tip:** You may be presented with a *New Connection Definition (DSN)* pop-up. Just click the **OK** button to proceed.

7. On the *Select a wizard* window select *IMS* and press **Next** to continue.



N.B. Before IMS virtual tables can be created, the data base descriptor (DBD) and program specification block need to be imported into the studio.

8. On the *New IMS Virtual Table* window, click the **Extract DBD** button to obtain the data base descriptor (DBD) of the target data base.



9. On the *New IMS DBD Metadata* window, confirm the values for the *Host* (wg31.washington.ibm.com) and *Server* (AVZS(port 12300)). They should be correct and press the **Next** button to continue.

10. On the *Source Download* window use the pull-down arrow to select the *IMSSOURCE-DFSF10.SDFSISRC* source library (created earlier). This will download a list of the members in this partitioned data set. Select member *DI21PART* and use the **Download** button to have the source for this member downloaded to the workstation.

**New IMS DBD Metadata Wizard**

**Source Download**

Download and select the DBD file that defines the data layout. To download a Source Library Member select an Available Source Library from the list.

Download Folder:  
/Data Virtualization Manager/src/wg31.washington.ibm.com/AVZS/download

Available Source Libraries:  
IMSSOURCE - DFSF10.SDFSISRC

Filter patterns  
\* Apply

Source Library Members:		Downloaded Source Files:	
Name		Name	
DFSZLDSW		<input checked="" type="checkbox"/> DI21PART	
DI21PART			
DSPBPROC			
DSPUPJCL			
HWSCFG00			
ICJCL			
ICRCVJCL			
JOBCL			
LOGCLJCL			
MDFSYSN			

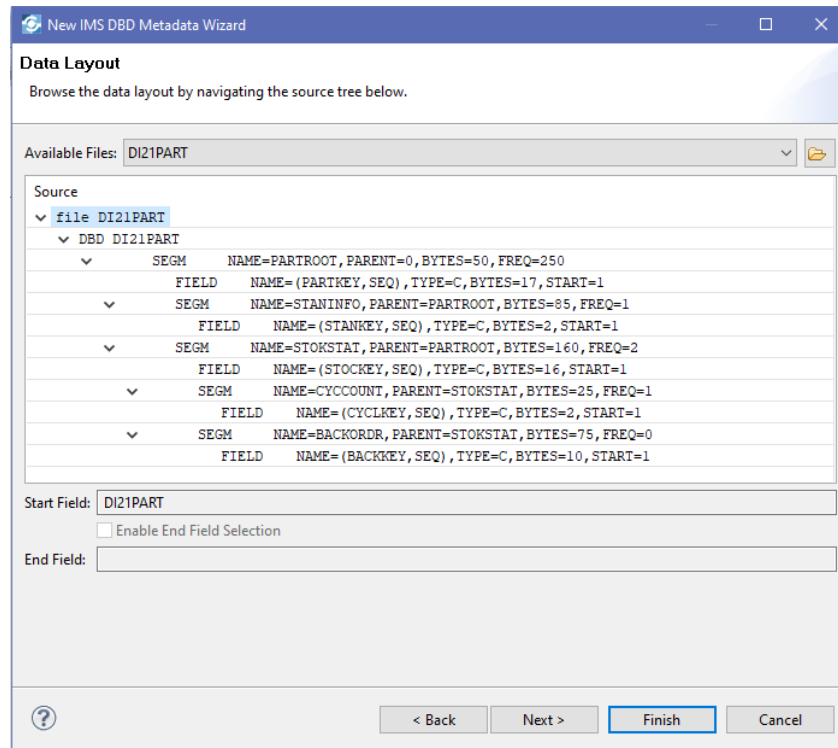
Download->

? < Back Next > Finish Cancel

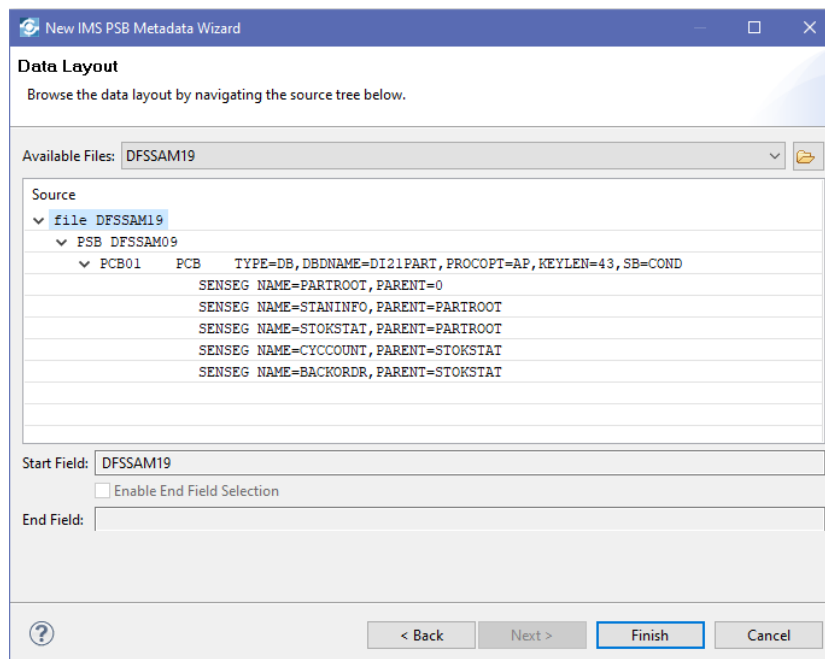
**Tech-Tip:** Enter a value of *DI\** in the *Filter patterns* area and press **Apply** to limit the list of members. Be sure to reset the filter pattern when listing other members.



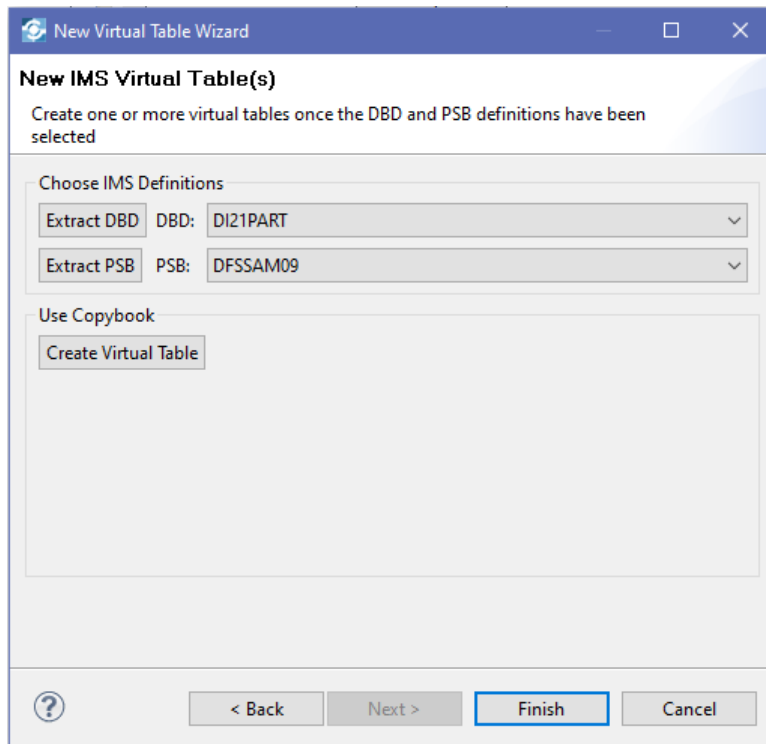
11. Select *DI21PART* and click **Next** to continue. This will display the *Data Layout* window. Scroll down and expand the structure (see below). This structure represents DBD of the database. This shows the hierarchy of the segments and the key names and key lengths. Click **Finish** to continue.



12. Repeat these steps to extract the program specification block (PSB) that will be used to access the data base. The PSB source is in member *DFSSAM19* in the *IMSSOURCE* virtual source library. Click **Finish** to continue.



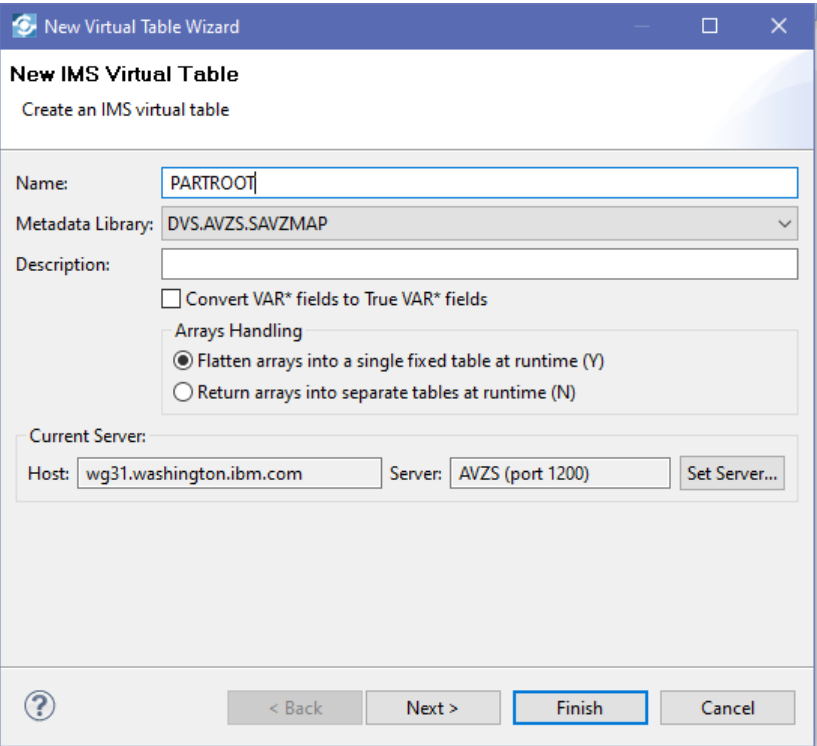
13. The *New Virtual Table Wizard – New IMS Virtual Table(s)* should now display the value *DI21PART* for the *DBD* and *DFSSAM19* for the *PSB*. Click the **Create Virtual Table** button to continue.



**Tech Tip:** Extracting or importing the source of the DBD and PSB only needs to be done once. Additional virtual tables for the same database can reuse the source of the DBD and PSB.

N.B. The name of the member in DFSF10.SDFSISRC containing the source of the PSB is DFSSAM19. In the source of the PSB, the name of the PSB is defined as DFHSAM09. In subsequent text and screen shots you will see references to member DFSSAM19 and references to PSB DFSSAM09.

14. Next create a virtual table for the root segment. On the *New Virtual Table Wizard – Create an IMS virtual table* window enter **PARTROOT** as the name of the table. Click **Next** to continue.

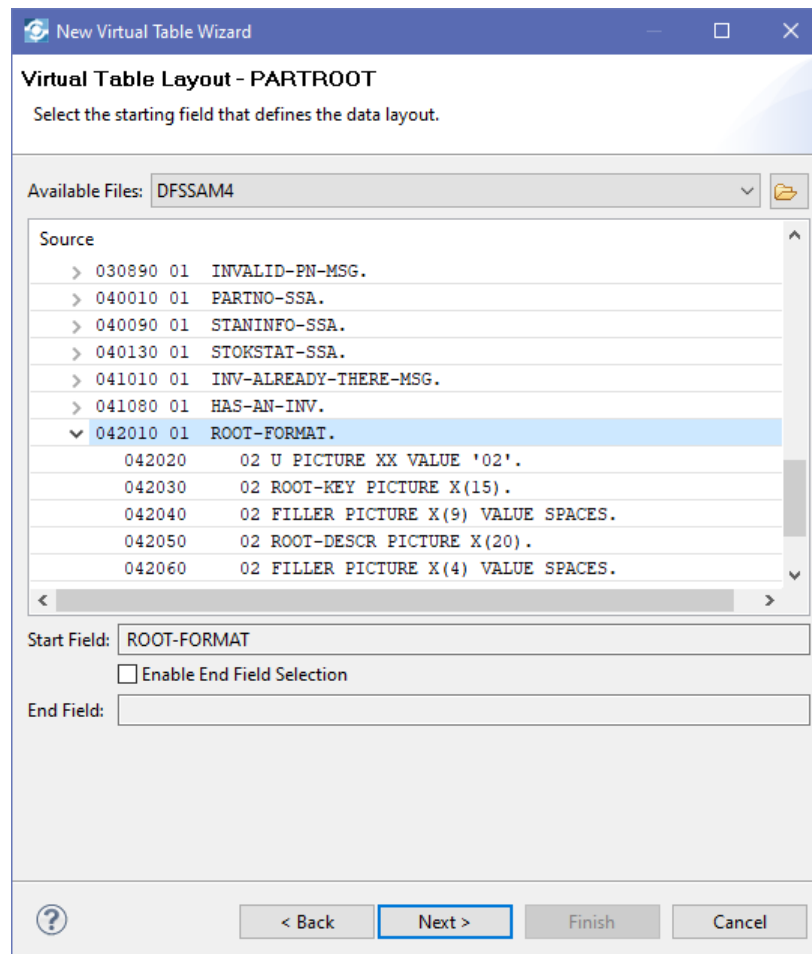


The image shows a screenshot of the 'New Virtual Table Wizard' window, specifically the 'New IMS Virtual Table' step. The window has a title bar with a gear icon and the text 'New Virtual Table Wizard'. Below the title bar, the main heading is 'New IMS Virtual Table' with the subtitle 'Create an IMS virtual table'. The form contains several fields and options: 'Name:' with a text box containing 'PARTROOT'; 'Metadata Library:' with a dropdown menu showing 'DVS.AVZS.SAVZMAP'; 'Description:' with an empty text box; a checkbox for 'Convert VAR\* fields to True VAR\* fields' which is unchecked; an 'Arrays Handling' section with two radio buttons: 'Flatten arrays into a single fixed table at runtime (Y)' (which is selected) and 'Return arrays into separate tables at runtime (N)'; and a 'Current Server:' section with 'Host:' (text box with 'wg31.washington.ibm.com'), 'Server:' (text box with 'AVZS (port 1200)'), and a 'Set Server...' button. At the bottom, there is a help icon (question mark), and four buttons: '< Back', 'Next >', 'Finish' (which is highlighted with a blue border), and 'Cancel'.

15. On the *Source Download – PARTROOT* window, use the pull-down arrow to select *IMSSOURCE – DFSF10.SDFSISRC* and then select member *DFSSAM4*. Click **Download** to download the source to the studio. Click **Next** to continue.

The screenshot shows the 'New Virtual Table Wizard' dialog box, specifically the 'Source Download - PARTROOT' step. The dialog has a title bar with a question mark icon and standard window controls. The main content area includes a 'Download Folder' text box with the path '/Data Virtualization Manager/src/wg31.washington.ibm.com/AVZS/download'. Below this is a dropdown menu for 'Available Source Libraries' with 'IMSSOURCE - DFSF10.SDFSISRC' selected. A 'Filter patterns' text box contains an asterisk, and an 'Apply' button is to its right. In the center, there is a 'Download->' button. To the left of this button is a list box titled 'Source Library Members' containing the following members: DFS003JC, DFS3DCBL, DFSACALO, DFSACCB, DFSASCBL, DFSAUT11, DFSAUTAL, DFSAUTDB, DFSAUTEL, and DFSDRD1. To the right of the 'Download->' button is another list box titled 'Downloaded Source Files' with three entries: DFSSAM19 (unchecked), DFSSAM4 (checked), and DI21PART (unchecked). At the bottom of the dialog are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. A help icon (?) is located at the bottom left.

16. This will display the *Virtual Table Layout – PARTROOT* window. In the case, this member is a COBOL program where the layout of the root segment is defined in a 01 COBOL level structure, *ROOT-FORMAT*. Scroll down the source of the program and locate and select this structure and expand it to show its contents (see below). Click **Next** to continue.



17. On the *IMS Information - PARTROOT* use the pull-down arrow to select the *PARTROOT* segment. Select the radio button beside *Use IMS/DBCTL (read/write, transaction integrity)*. Click the **Finish** button to continue. Click **Cancel** on the *New IMS Virtual Table(s)* window.

New Virtual Table Wizard

**IMS Information - PARTROOT**

Specify the IMS information

Server Configuration

Default IMS ID: IVP1    IMS-Direct Protocol: Enabled

DBD Name: DI21PART

Segment Name: PARTROOT

PSB Name: DFSSAM09

☐ Use IMS-Direct (read-only, high performance bulk data access)

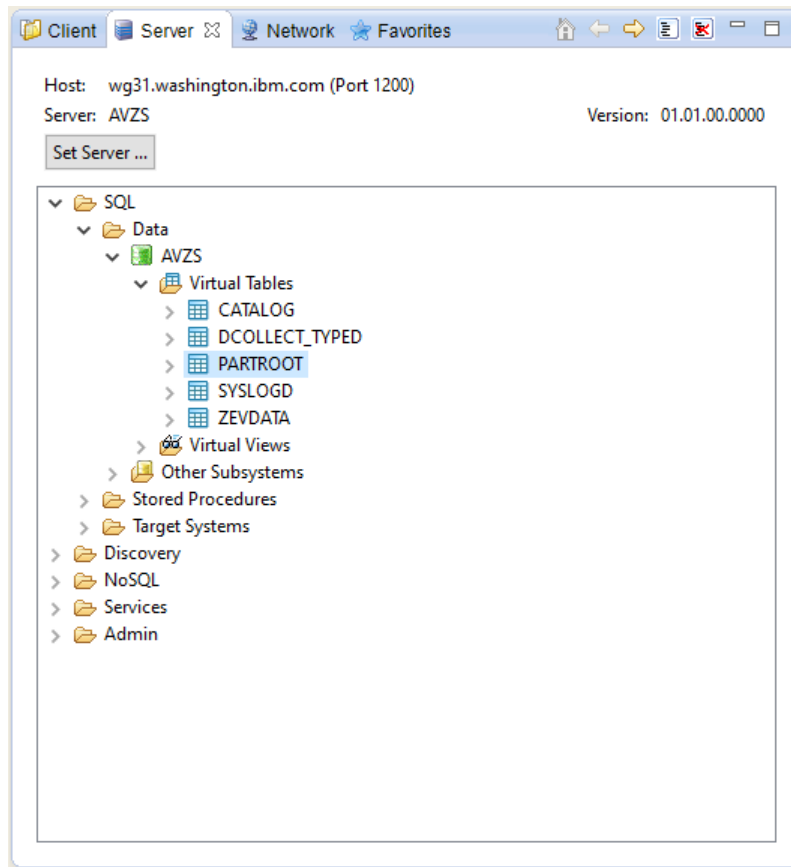
☒ Use IMS/DBCTL (read/write, transactional integrity)

IMS ID Override (used with IMS-Direct only): IVP1

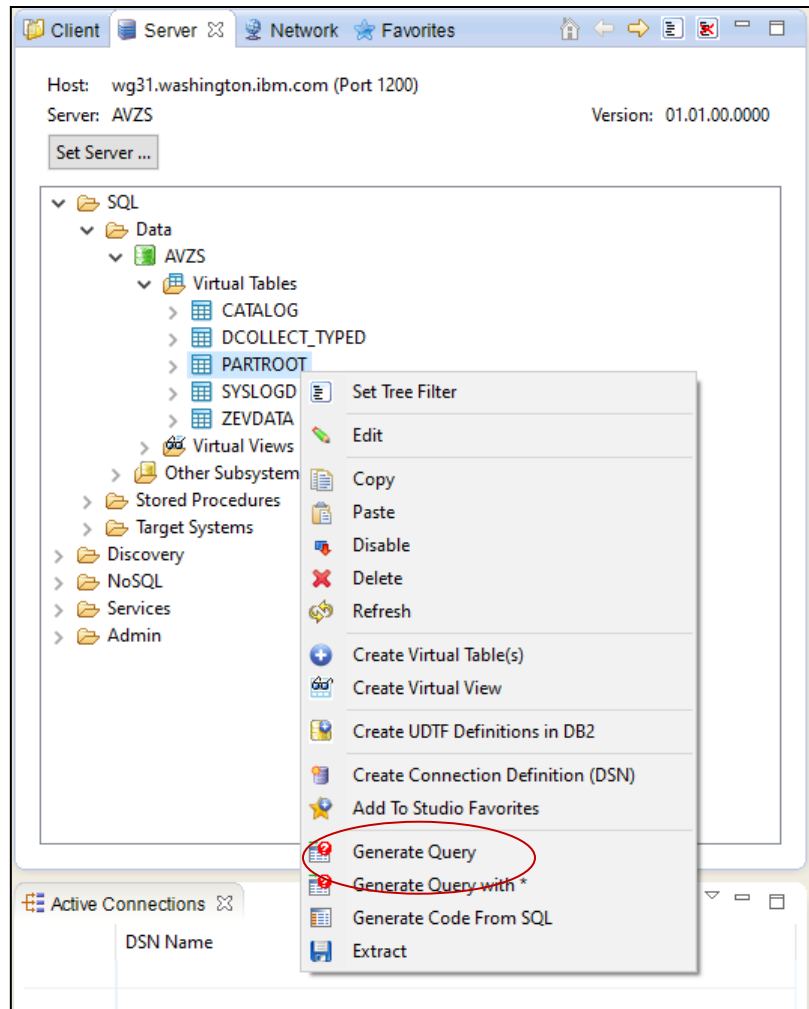
Advanced >>

?    < Back    Next >    **Finish**    Cancel

7. In the list of *Virtual Tables* an entry for *PARTROOT* should now appear. Select *PARTROOT* and right mouse button click.



8. Select option *Generate Query* and click **Yes** on the *Execute Query?* pop up window. If a *New Connection Definition(DSN)* pop up window appears, click **OK** to continue.





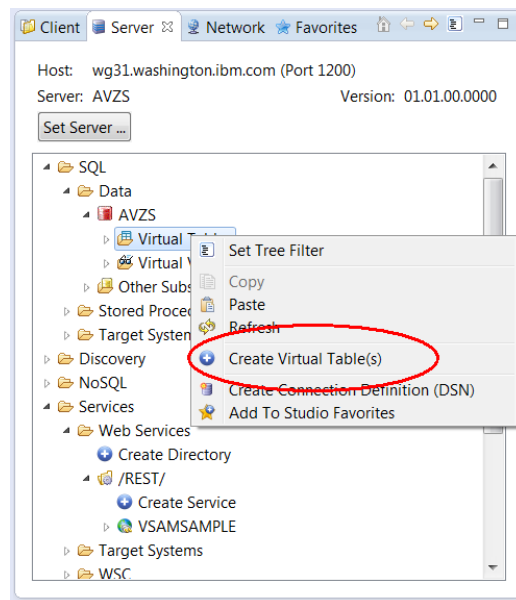
9. This will access the DL/I data base and display the root segments in the view on the lower right-hand side

U	ROOT_KEY	ROOT_DESCR	RECORD_ID	CHILD_ID
0	02	AN960C10	WASHER	02AN960C10
1	02	CK05CW181K	CAPACITOR	02CK05CW181K
2	02	CSR13G104KL	KRIJ50KS	02CSR13G104KL
3	02	JAN1N976B	DIODE CODE-A	02JAN1N976B
4	02	MS16995-28	SCREW	02MS16995-28
5	02	N51P3003F000	SCREW	02N51P3003F000
6	02	RC07GF273J	RESISTOR	02RC07GF273J
7	02	106B1293P009	RESISTOR	02106B1293P009
8	02	250236-001	CAPACITOR	02250236-001
9	02	250239	TRANSISTOR	02250239
10	02	250241-001	CONNECTOR	02250241-001
11	02	250794	RESISTOR	02250794
12	02	250796	SWITCH	02250796
13	02	250891	SERVO VALVE	02250891
14	02	252252-003	COUPLING	02252252-003
15	02	3003802	CHASSIS	023003802
16	02	3003806	SWITCH	023003806
17	02	3007228	HOUSING	023007228
18	02	3008027	CARD FRONT	023008027
19	02	3009228	CAPACITOR	023009228
20	02	3009270	HOUSING	023009270
21	02	3009280	HOUSING CONV	023009280
22	02	3013405-002	MOUNTING	023013405-002
23	02	3013412	COVER	023013412
24	02	3013429-001	COVER ASSY	023013429-001
25	02	3013460-001	CAPACITOR	023013460-001

Verification of the virtual table completes the creation of the virtual table for the root segment.

Now, create a virtual table for the Stock Status segment.

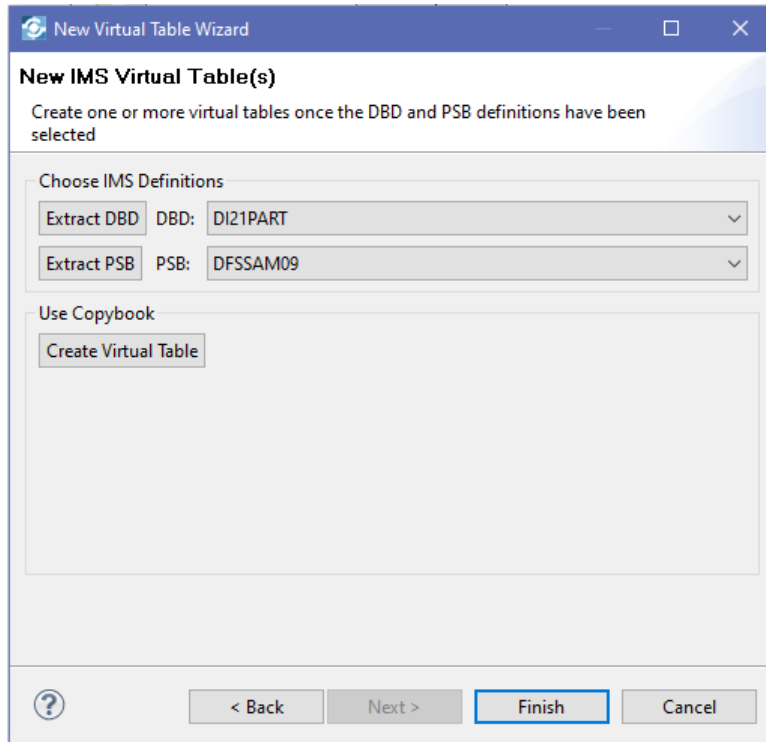
14. Select the *Virtual Tables* folder and right mouse button click and then select the *Create Virtual Table(s)* property, see below.



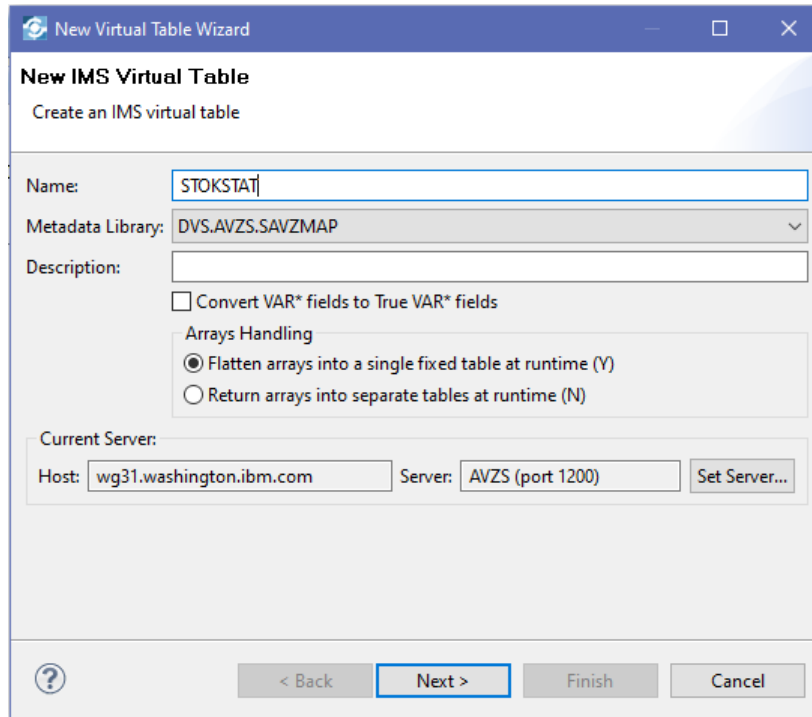
18. On the *Select a wizard* window select *IMS* and press **Next** to continue.

N.B. The data base descriptor (DBD) and program specification block were imported into the studio in an earlier step.

19. On the *New IMS Virtual Table* window, use the pull-down arrows to select *DI21PART* DBD and *DFSSAM09* PSB. Click the **Create Virtual Table** button to continue.



20. Create a virtual table for the stock status segment as before. On the *New Virtual Table Wizard – Create an IMS virtual table* window enter **STOKSTAT** as the name of the table. Click **Next** to continue.

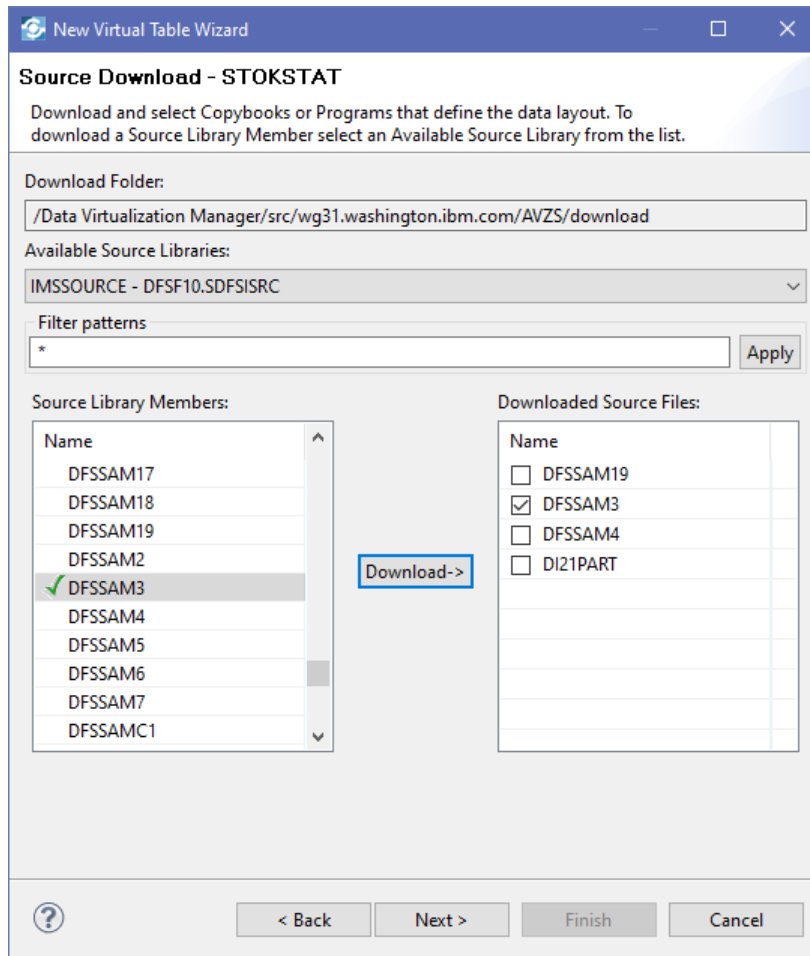


The image shows a screenshot of the 'New Virtual Table Wizard' window, specifically the 'Create an IMS virtual table' step. The window has a blue title bar with the text 'New Virtual Table Wizard'. Below the title bar, the main heading is 'New IMS Virtual Table' and the subtitle is 'Create an IMS virtual table'. The form contains the following fields and options:

- Name:** A text box containing 'STOKSTAT'.
- Metadata Library:** A dropdown menu showing 'DVS.AVZS.SAVZMAP'.
- Description:** An empty text box.
- ☐ Convert VAR\* fields to True VAR\* fields
- Arrays Handling:**
  - ☒ Flatten arrays into a single fixed table at runtime (Y)
  - ☐ Return arrays into separate tables at runtime (N)
- Current Server:**
  - Host:** A text box containing 'wg31.washington.ibm.com'.
  - Server:** A text box containing 'AVZS (port 1200)'.
  - Set Server...** A button.

At the bottom of the window, there is a navigation bar with a help icon (question mark), a '< Back' button, a 'Next >' button (which is highlighted with a blue border), a 'Finish' button, and a 'Cancel' button.

21. On the *Source Download – STOKSTAT* window, use the pull-down arrow to select *IMSSOURCE – DFSF10.SDFSISRC* and select member *DFSSAM3* to download to the studio. Click **Next** to continue.



22. This will display the *Virtual Table Layout – STOKSTAT* window. In the case, this member is a COBOL COPY book where the layout of the root segment is defined in a 01 COBOL level structure, *STOCK-STATUS-RET*. Scroll down the source of the program and locate and select this structure and expand it to show its contents (see below). Click **Next** to continue.

**New Virtual Table Wizard**

**Virtual Table Layout - STOKSTAT**

Select the starting field that defines the data layout.

Available Files: DFSSAM3

Source

- > 1 STAN-INFO-RET REDEFINES SEG-RET-AREA
- ▼ 1 STOCK-STATUS-RET REDEFINES STAN-INFO-RET
  - 001180 02 FILLER1 PICTURE XX.
  - 001200 02 SS-AREA PICTURE X.
  - 001220 02 SS-DEPT PICTURE XX.
  - 001240 02 SS-PROJ PICTURE XXX.
  - 001260 02 SS-DIV PICTURE XX.
  - 001280 02 FILLER2 PICTURE X(10).
  - 001300 02 SS-UNIT-PRICE PICTURE 9(6)V999.
  - 001320 02 FILLER3 PICTURE X(05).
  - 001340 02 SS-UNIT-OF-MEAS PICTURE X(04).
  - 001360 02 FILLER4 PICTURE X(33).
  - 001380 02 SS-STOCK-DATE PICTURE X(03).
  - 001400 02 FILLER5 PICTURE X(15).
  - 001420 02 SS-CUR-DECTS PICTURE 9(12)V99.

Start Field: STOCK-STATUS-RET

☐ Enable End Field Selection

End Field:

< Back Next > Finish Cancel

23. On the *New Virtual Table Wizard – IMS Information – STOKSTAT* window, use the pull-down arrow to select the *STOKSTAT* segment. Select the radio button beside *Use IMS/DBCTL (read/write, transactional integrity)*. Click the **Finish** button to continue. Click **Cancel** to close the *New IMS Virtual Table(s)* window.

New Virtual Table Wizard

IMS Information - STOKSTAT

Specify the IMS information

Server Configuration

Default IMS ID: IVP1 IMS-Direct Protocol: Enabled

DBD Name: DI21PART

Segment Name: STOKSTAT

PSB Name: DFSSAM09

☐ Use IMS-Direct (read-only, high performance bulk data access)

☒ Use IMS/DBCTL (read/write, transactional integrity)

IMS ID Override (used with IMS-Direct only): IVP1

Advanced >>

< Back Next > Finish Cancel

21. In the list of *Virtual Tables* an entry for *STOKSTAT* should now appear. Select *STOKSTAT* and right mouse button click.
22. Select option *Generate Query* and click **Yes** on the *Execute Query?* pop up window.
23. This will access the DL/I data base and display the stock status segments in the view on the lower right-hand side.

	FILLER1	SS_AREA	SS_DEPT	SS_PROJ	SS_DIV	FILLER2	SS_UNIT_PRICE	FILLER3	SS_UNIT_OF_MEAS	FILLER4	SS_STOCK_DATE	FILLER5	SS_CUR_REQMTS	SS_UNPL_REQMTS	SS
0	00		AA	165	11		0.000		EACH	00...		51...	131.0	15.0	20
1	00		AK	287	7F	...	0.000		EACH	00...		36...	88.0	0.0	0
2	00	2	80	091	26		0.000		EACH	...	513	51...	630.0	0.0	14
3	00		VF	529	06		1.000		EACH	00...		24...	0.0	0.0	0
4	00	2	59	003	26		0.340			...		51...	1320.0	0.0	60
5	00	2	59	109	26		0.340			...		51...	8.0	0.0	0
6	00		DB	745	5R	...	2.710		EACH	00...		48...	14.0	0.0	0
7	00		SK	217	13	...	2.710		EACH	00...		26...	4.0	0.0	0
8	00	2	55	025	26		0.000			...		47...	14.0	0.0	0
9	00	2	55	091	26		0.000			...		51...	17.0	0.0	20
10	00		AA	165	11		0.152		EACH	00...		48...	26.0	0.0	0
11	00		BA	165	15		0.069		EACH	00...		45...	6.0	0.0	0
12	00		FF	554	6D	...	0.069		EACH	00...		44...	44.0	0.0	0
13	00	2	59	109	26		6.980			...		49...	95.0	0.0	0
14	00	2	59	060	26		0.000	0000	0000	...		40...	313.0	0.0	0
15	00		AK	245	27		0.240		EACH	00...		21...	33.0	0.0	0
16	00	2	80	091	26		0.000	0000	0000	...	516	51...	17.0	0.0	0
17	00	2	80	111	26		0.000			...		45...	26.0	0.0	0
18	00	2	59	003	26		0.000	0000	0000	...	393	49...	1055.0	200.0	0
19	00	2	59	060	26		0.000	0000	0000	...		29...	0.0	0.0	0
20	00	2	59	109	26		1.820			...		48...	320.0	0.0	20
21	00	2	59	003	26		0.000	0000	0000	...	393	48...	410.0	200.0	60
22	00	2	59	060	26		0.000	0000	0000	...		44...	0.0	0.0	0
23	00	2	59	109	26		3.670			...		51...	72.0	0.0	0
24	00	2	59	109	26		6.500			...		51...	68.0	0.0	10

Verification of the virtual table completes the creation of the virtual tables for the stock status segment.

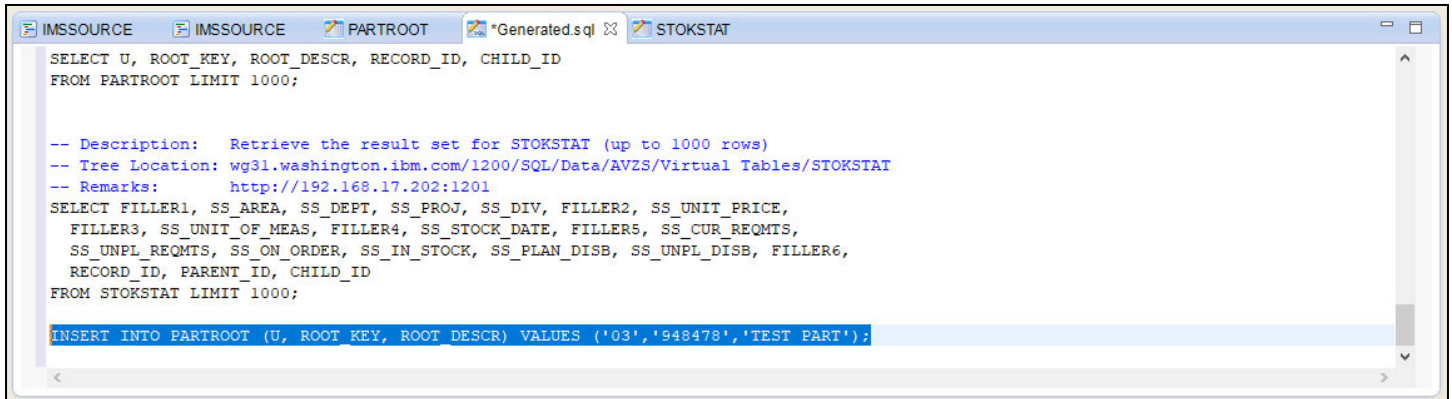
24. (Optional) Create virtual tables for the other segments, STANINFO, CYCOUNT and BACKORDR. The COBOL layout for STANINFO can be found member DFHSAM2 in structure STANINFO-FORMAT. The COBOL layout for segment CYCOUNT can be found member DFHSAM3 in structure CYCLE-COUNT-RET and the COBOL layout for segment BACKORDR can also be found in member DFSSAM3 in structure BACK-ORDER-RET.

## Use the Data Virtualization Manager Studio to test SQL commands

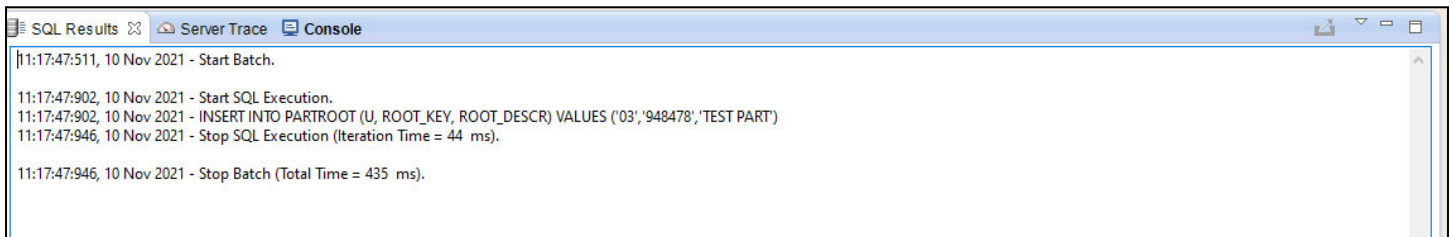
Now that the virtual tables have been created, they can be used to explore and test SQL commands that will be used in DVM services using the *Generated.sql* pane of the DVM studio.

1. In the DVM studio, focus on the *Generated.sql* pane, see below. Enter the SQL INSERT command as shown below:

***INSERT INTO PARTROOT (U, ROOT\_KEY, ROOT\_DESCR) VALUES ('03','948478','TEST PART')***



2. Select the entire *INSERT* command and right mouse button click and then select the *Execute SQL* option. You should see results like the one below in the SQL Results pane.





3. Use the SQL SELECT command to display the inserted root segment using the same process.

***SELECT \* FROM PARTROOT WHERE ROOT\_KEY='948478'***

U	ROOT_KEY	ROOT_DESCR	RECORD_ID	CHILD_ID
0	948478	TEST PART	03948478	F0F3F9F4F8F4F7F8404040404...

**Tech Tip:** Notice that two new columns have been retrieved, RECORD\_ID and CHILD\_ID. Not shown here is a PARENT\_ID column

The **RECORD\_ID** column is the key feedback area from the DLI call for a target segment and contains the sequence field information for all segments from the root segment to the target segment accessed (in this case, the root segment itself).

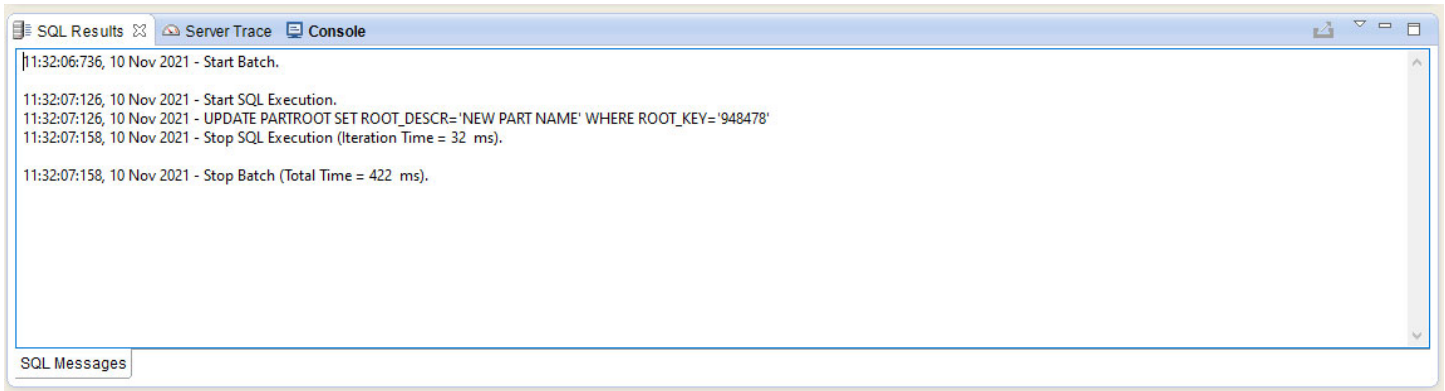
The **CHILD\_ID** column contains the hex key to the child record if it has a child record.

The **PARENT\_ID** column is the hex form of the RECORD\_ID for a parent segment, and does not include the sequence field of the target segment.

Segments can be inserted, updated, or deleted using the RECORD\_ID and/or the PARENT\_ID parameters. These columns will be used later when accessing a root segment's child segment, e.g.,  
**SELECT \* FROM PARTROOT WHERE CHILD\_ID='F0F3F9F4F8F4F7F840404040404040'**

4. Use the SQL UPDATE command to update the new root segment's description.

**UPDATE PARTROOT SET ROOT\_DESCR='NEW PART NAME' WHERE ROOT\_KEY='948478'**



5. Use the SQL SELECT command to display the root updated description.

**SELECT \* FROM PARTROOT WHERE CHILD\_ID='F0F3F9F4F8F4F7F84040404040404040'**

The screenshot shows the 'SQL Results' tab of the IBM z/OS Connect interface. It displays a table with the following columns: U, ROOT\_KEY, ROOT\_DESCR, RECORD\_ID, and CHILD\_ID. The first row contains the following data:

U	ROOT_KEY	ROOT_DESCR	RECORD_ID	CHILD_ID
03	948478	NEW PART NAME	03948478	F0F3F9F4F8F4F7F8404040404040...

Below the table, there are controls for 'Columns Group' (set to 1 of 1) and 'Columns per group' (set to 25). At the bottom, it indicates '1 rows' and has a tab labeled 'SQL Messages'.

**Tech Tip:** The value of the CHILD\_ID in the WHERE clause is determined by the value of the CHILD\_ID as displayed in Step 3 above.

SQL Results

18:26:15:212, 11 Nov 2021 - Start Batch.

18:26:15:587, 11 Nov 2021 - Start SQL Execution.

18:26:15:587, 11 Nov 2021 - INSERT INTO STOKSTAT (FILLER1,SS\_AREA,SS\_DEPT,SS\_PROJ,SS\_UNIT\_OF\_MEAS,SS\_UNIT\_PRICE,SS\_CUR\_REQMTS,SS\_UNPL\_REQMTS,SS\_ON\_ORDER,SS\_IN\_STOCK,SS\_PLAN\_DISB,SS\_UNPL\_DISB,PARENT\_ID) VALUES('00','2','23','54','EACH',0.000,1,2,3,4,5,6,'F0F3F9F4F8F4F7F84040404040404040')

18:26:15:622, 11 Nov 2021 - Stop SQL Execution (Iteration Time = 35 ms).

18:26:15:622, 11 Nov 2021 - Stop Batch (Total Time = 410 ms).

SQL Messages

Below are the results of using the IMS test program (DLT0) to dump the new segments.

Use the Data Virtualization Manager Studio to test SQL commands  
© Copyright IBM 2020, 2022 All rights reserved  
Mitch Johnson (mitchj@us.ibm.com)

7. Use a SELECT SQL command using the PARENT\_ID in a WHERE clause to display the new segment.

```
SELECT * FROM STOKSTAT WHERE PARENT_ID='F0F3F9F4F8F4F7F84040404040404040'
```

SS_AREA	SS_DEPT	SS_PROJ	SS_DIV	RECORD_ID	PARENT_ID	CHILD_ID
0	12	23	54	03948478	F0F3F9F4F8F4F7F...	F0F3F9F4F8F4F7F...

Columns Group: 1 of 1 Columns per group: 25

1 rows SQL Messages

8. Use a SELECT SQL command using the SS\_AREA column in the WHERE clause display all segments where SS\_DEPT is either '55' or '23'

```
SELECT FILLER1, SS_AREA, SS_DEPT, SS_PROJ, SS_DIV, RECORD_ID, PARENT_ID, CHILD_ID  
FROM STOKSTAT WHERE SS_DEPT='55' OR SS_DEPT='23'
```

FILLER1	SS_AREA	SS_DEPT	SS_PROJ	SS_DIV	RECORD_ID	PARENT_ID	CHILD_ID
0	00	2	54		02AN960C10 ...	F0F2C1D5F9F6...	F0F2C1D5F9F6...
1	00	2	025	26	02CSR13G104K...	F0F2C3E2D9F1...	F0F2C3E2D9F1...
2	00	2	091	26	02JAN1N976B ...	F0F2D1C1D5F1...	F0F2D1C1D5F1...

Columns Group: 1 of 1 Columns per group: 25

3 rows SQL Messages

9. Use a SQL DELETE command to delete the child segment using its RECORD\_ID.

```
DELETE FROM STOKSTAT WHERE PARENT_ID=' F0F3F9F4F8F4F7F84040404040404040'
```

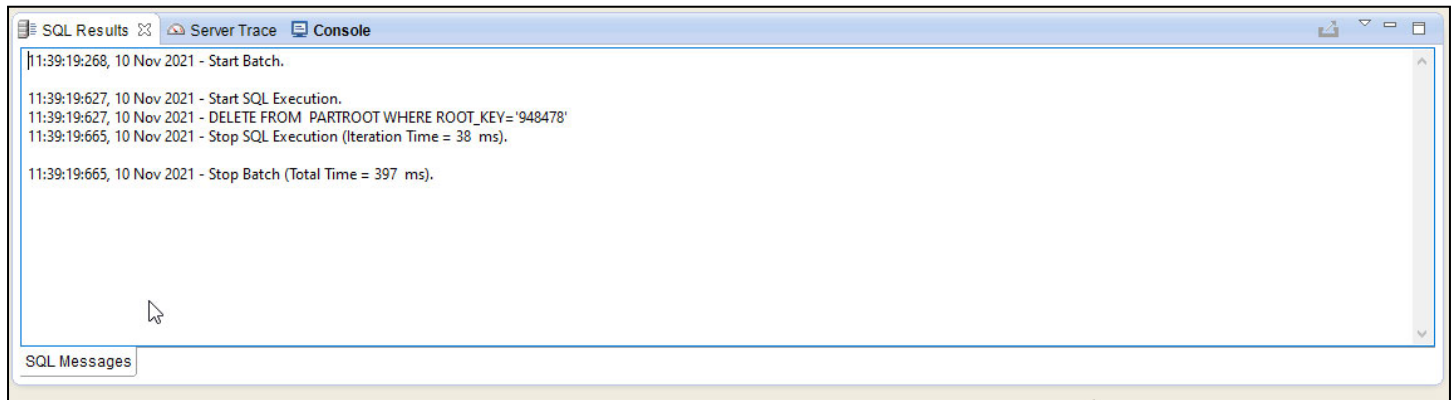
```

18:16:55:030, 11 Nov 2021 - Start Batch.
18:16:55:405, 11 Nov 2021 - Start SQL Execution.
18:16:55:405, 11 Nov 2021 - DELETE FROM STOKSTAT WHERE PARENT_ID='F0F2C1D5F9F6F0C3F1F040404040404040'
18:16:55:442, 11 Nov 2021 - Stop SQL Execution (Iteration Time = 37 ms).
18:16:55:442, 11 Nov 2021 - Stop Batch (Total Time = 412 ms).
  
```

SQL Messages

10. Finally use a SQL DELETE command to delete the new root segment.

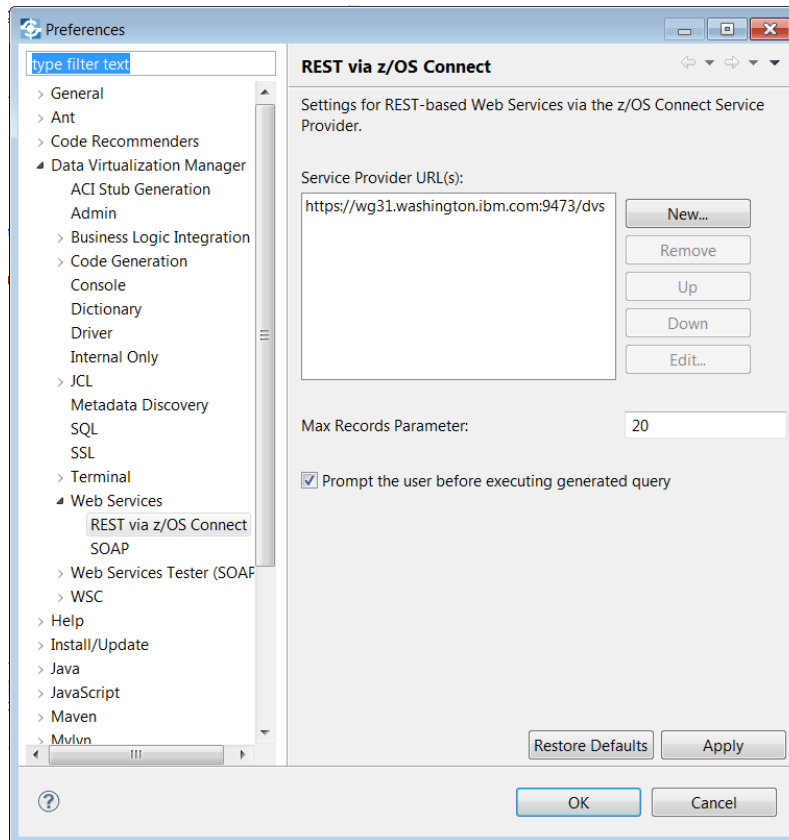
**DELETE FROM PARTROOT WHERE ROOT\_KEY='948478'**



The purpose of exploring these SQL command using the studio was to demonstrate how the virtual tables can be accessed to insert, update, retrieve and delete segments in a hierarchical data base. What we learned can be now be used to create DVM services.

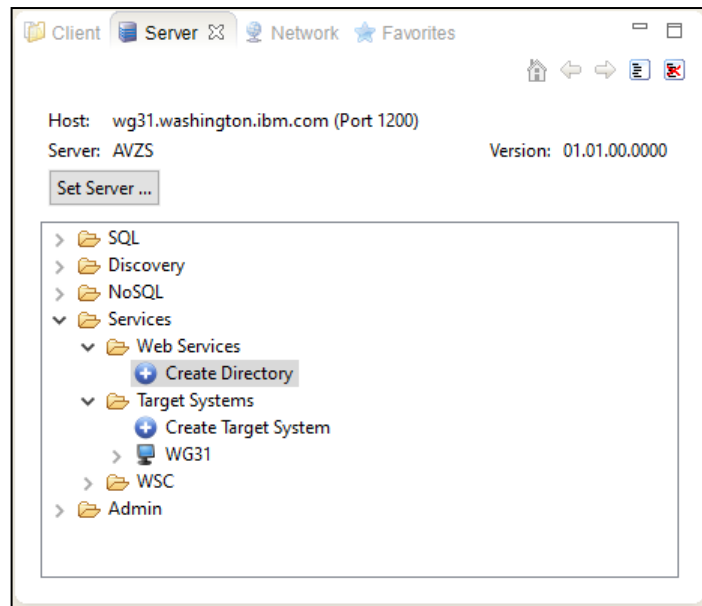
## Use the Data Virtualization Manager Studio to create a web service

1. Confirm that the DVM studio is properly configured to communicate with the z/OS Connect server which has the DVM service provider installed. On the DVM Studio toolbar click on *Windows* then *Preferences* to display the Eclipse *Preferences* window. Expand *Data Virtualization Manager* and then *Web Services* to select *REST via z/OS Connect*, see below:

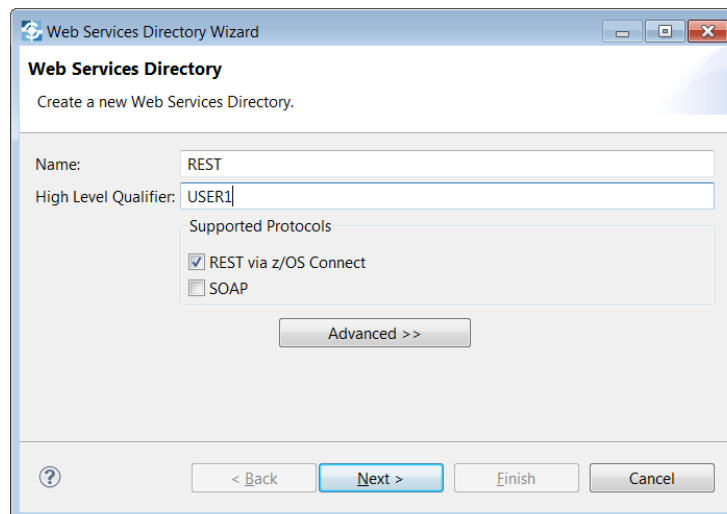


2. Ensure the *Service Provider URL(s)* is set to **<https://wg31.washington.ibm.com:9473/dvs>**. If not, select the provider and use the **Edit** button to set it correctly or if a *Service Provide URL* is not present, add one.

3. Back in the *Server* view, expand the *Services* and then the *Web Services* folders.

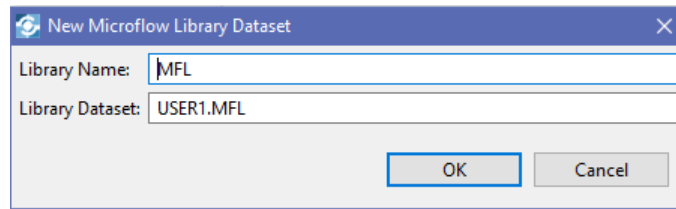


4. If the *REST* Web Services directory does not exist, double click on *Create Directory* to open the *Web Services Directory Wizard*. Enter **REST** and **USER1** as shown below. Click **Next** to continue. Otherwise continue with Step 7.



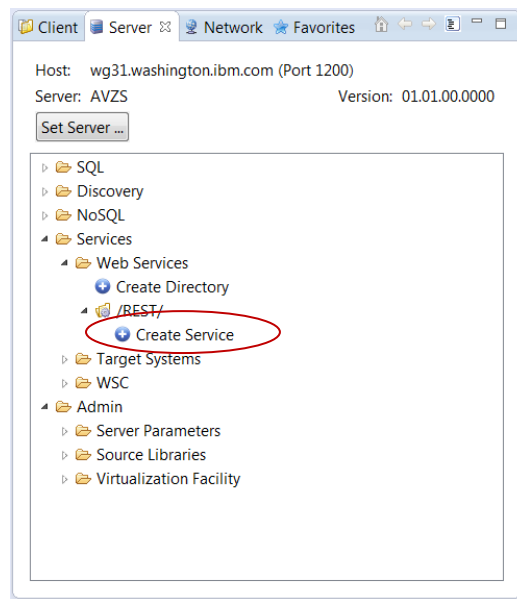
**Tech-Tip:** The *High-Level Qualifier* will be used to create a micro flow partitioned data set with a data set name of USER1.MFL.

- \_\_\_ 5. On the *Web Service Directory Wizard – Microflow Library* window, if there is a current Microflow liberty (e.g., USER1.MFL), select it. Otherwise click the **Create New Microflow Library** button and accept the defaults on the *New Microflow Library Dataset* pop-up window. Click **OK** to continue.



- \_\_\_ 6. On the *Microflow Library* window, select *MFL* under *Current Microflow Libraries* and click **Finish** to continue.

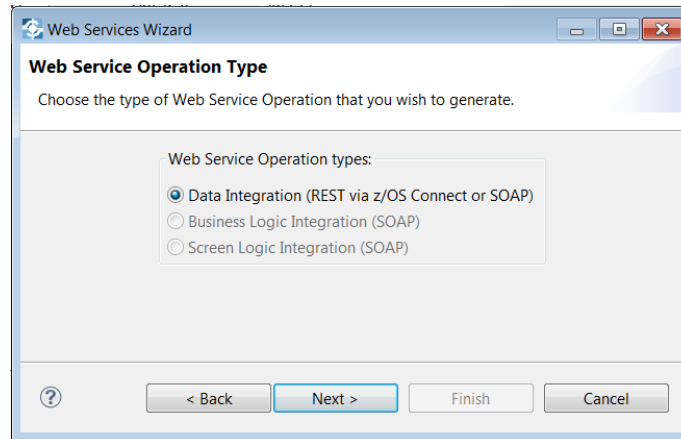
- \_\_\_ 7. Expand */REST/* and use the *Create Service* wizard to create a new web service.



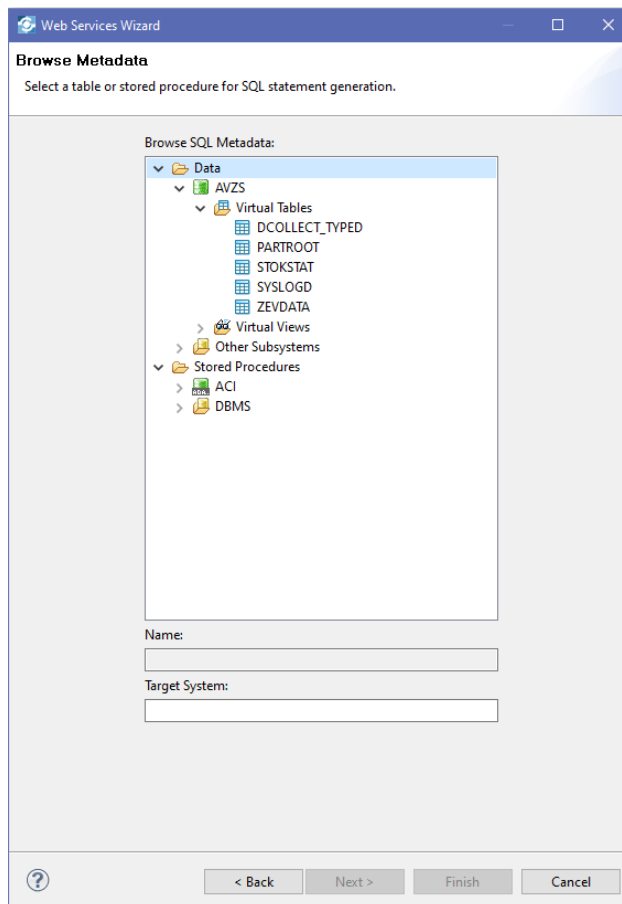
- \_\_\_ 8. On the *Web Service – Create a new Web Service* window, enter **PARTROOT** as the *Name* and press **Next** to continue.



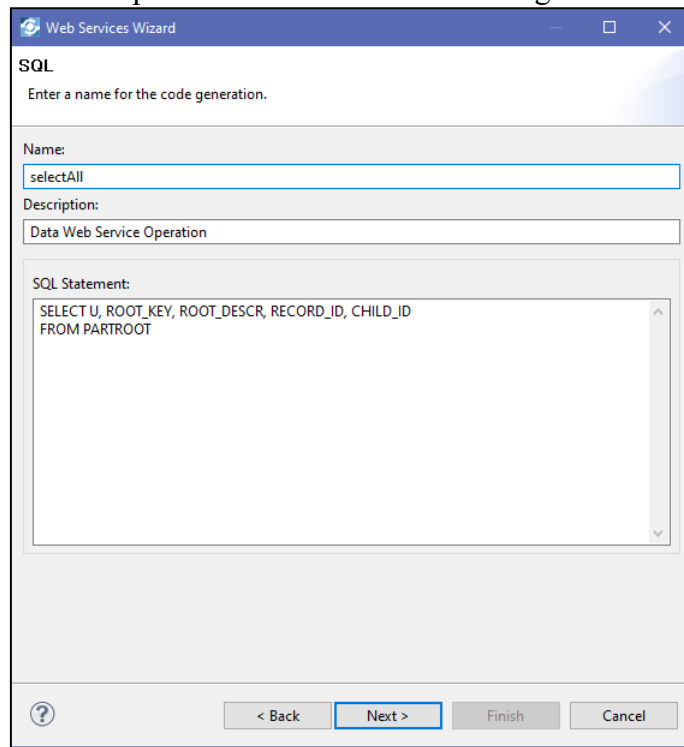
9. On the *Web Service Operation Type* window, ensure the radio button beside *Data Integration (REST via z/OS Connect or SOAP)* is selected and click **Next** to continue.



10. On the *Browse Metadata* window, expand the *Data* folder then the *AVZS* folder and then the *Virtual Tables* folder to display the virtual table created in the previous section. Select virtual table *PARTROOT* and press **Next** to continue.



11. The default SQL statement will be displayed on the *SQL* window. Change the name of the operation to ***selectAll*** to indicate that this operation will retrieve all root segments from the data base. Click **Next** to continue.

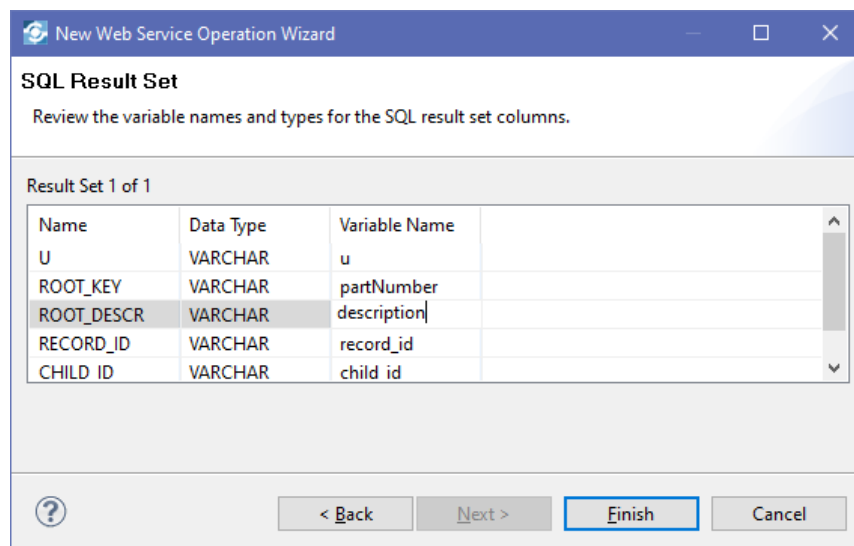


The screenshot shows the 'Web Services Wizard' window with the 'SQL' tab selected. The 'Name' field contains 'selectAll'. The 'Description' field contains 'Data Web Service Operation'. The 'SQL Statement' field contains the following SQL code:

```
SELECT U, ROOT_KEY, ROOT_DESCR, RECORD_ID, CHILD_ID
FROM PARTROOT
```

At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted.

12. The next window to be displayed will show the results that will be returned when the operation is executed. Change the *Variable names* for *ROOT\_KEY* and *ROOT\_DESCR* to **partNumber** and **description** as shown below (all of them could have been renamed but we chose not to).



The screenshot shows the 'New Web Service Operation Wizard' window with the 'SQL Result Set' tab selected. The 'Result Set 1 of 1' table displays the following columns: Name, Data Type, and Variable Name.

Name	Data Type	Variable Name
U	VARCHAR	u
ROOT_KEY	VARCHAR	partNumber
ROOT_DESCR	VARCHAR	description
RECORD_ID	VARCHAR	record_id
CHILD ID	VARCHAR	child id

At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Finish' button is highlighted.

**Tech-Tip:** This is useful because we could have removed some of the columns on the previous windows. This display simply confirms which columns will be returned.

13. Click **Finish** to continue.

14. Use the *Create Operation* wizard under *PARTROOT* to create a new operation. Select the *PARTROOT* virtual table and click **Next** and proceed as before. This operation should be named *insertPart* and the *SQL Statement* should be changed to

**INSERT INTO PARTROOT (U, ROOT\_KEY, ROOT\_DESCR) VALUES (?, ?, ?)**

The screenshot shows a 'New Web Service Operation Wizard' window. It has a title bar with a question mark icon and standard window controls. The main area is titled 'SQL' and contains the instruction 'Enter a name for the code generation.' Below this are three input fields: 'Name:' with the value 'insertPart', 'Description:' with the value 'Data Web Service Operation', and 'SQL Statement:' with the value 'INSERT INTO PARTROOT (U, ROOT\_KEY, ROOT\_DESCR) VALUES (?, ?, ?)'. At the bottom, there are four buttons: a help button (question mark icon), '< Back', 'Next >' (which is highlighted with a blue border), and 'Cancel'.

15. Click **Next** to continue. Since a WHERE clause has been added with variables, providing values for these variables will be required. The next window to be displayed, *SQL Inputs*, will give us an opportunity to give meaningful names to these variables. On this window click on the value of variable name in the *Name* column and change the names of the inputs **keyPrefix**, **partNumber** and **description** as shown below. Click **Next** to continue.

Name	Data Type	Default Value
keyPrefix	VARCHAR	
partNumber	VARCHAR	
description	VARCHAR	

**Tech-Tip:** The key to the root segment is the concatenation of the *keyPrefix* and *partNumber*.

16. No results are returned on an insert, so click **Finish** on the *SQL Result Set* window.

17. Use the *Create Operation* wizard under *PARTROOT* create a new operation. Select the *PARTROOT* virtual table and click **Next** and proceed as before. This operation should be named *selectByPart* and the *SQL Statement* should be changed to:

***SELECT \* FROM PARTROOT WHERE ROOT\_KEY=?***

The screenshot shows a 'New Web Service Operation Wizard' window. The title bar says 'New Web Service Operation Wizard'. The main area is titled 'SQL' and has a subtitle 'Enter a name for the code generation.' Below this are three input fields: 'Name:' with the value 'selectByPart', 'Description:' with the value 'Select by part number', and 'SQL Statement:' with the value 'SELECT \* FROM PARTROOT WHERE ROOT\_KEY=?;'. At the bottom, there are four buttons: a help icon (?), '< Back', 'Next >' (which is highlighted with a blue border), 'Finish', and 'Cancel'.

18. Click **Next** to continue. Since a *WHERE* clause has been added with variable, providing a meaningful name for this variable will be required. The next window to be displayed, *SQL Inputs*, will give us an opportunity to give meaningful name to the variable. On this window click on the values of variable name in the *Name* column and change the contents to *partNumber*. Click **Next** to continue.

Name	Data Type	Default Value
partNumber	VARCHAR	

19. The columns that will be returned are displayed on the *SQL Result Set* window. Change the Variable Names as shown below and then click **Finish** to continue.

Change:

- *ROOT\_KEY* to *partNumber*
- *ROOT\_DESCR* to *description*

Name	Data Type	Variable Name
U	VARCHAR	u
ROOT_KEY	VARCHAR	partNumber
ROOT_DESCR	VARCHAR	description
RECORD_ID	VARCHAR	record_id
CHILD_ID	VARCHAR	child_id

20. Use the *Create Operation* wizard under *PARTROOT* create a new operation. Select the *PARTROOT* virtual table and click **Next** and proceed as before. This operation should be named ***updatePart*** and the *SQL Statement* should be changed to:

**UPDATE PARTROOT SET ROOT\_DESCR= ? WHERE ROOT\_KEY=?**

The screenshot shows the 'New Web Service Operation Wizard' dialog box with the 'SQL' tab selected. The dialog has a title bar with a question mark icon, a close button, and a maximize button. The main content area is divided into sections: 'SQL' (with a sub-instruction 'Enter a name for the code generation.'), 'Name:', 'Description:', and 'SQL Statement:'. The 'Name' field contains 'updatePart'. The 'Description' field contains 'Update the part segment'. The 'SQL Statement' field contains the text 'UPDATE PARTROOT SET ROOT\_DESCR= ? WHERE ROOT\_KEY=?;'. At the bottom, there are four buttons: a help button (question mark icon), '< Back', 'Next >', and 'Finish'. The 'Next >' button is highlighted with a blue border.

**SQL**  
Enter a name for the code generation.

Name:  
updatePart

Description:  
Update the part segment

SQL Statement:  
UPDATE PARTROOT SET ROOT\_DESCR= ? WHERE ROOT\_KEY=?;

? < Back Next > Finish Cancel

21. Click **Next** to continue. Since a *WHERE* clause has been added with variable, providing a meaningful name for this variable will be required. The next window to be displayed, *SQL Inputs*, will give us an opportunity to give meaningful name to the variable. On this window click on the values of variable name in the *Name* column and change the contents to *description* and *partNumber*. Click **Next** to continue.

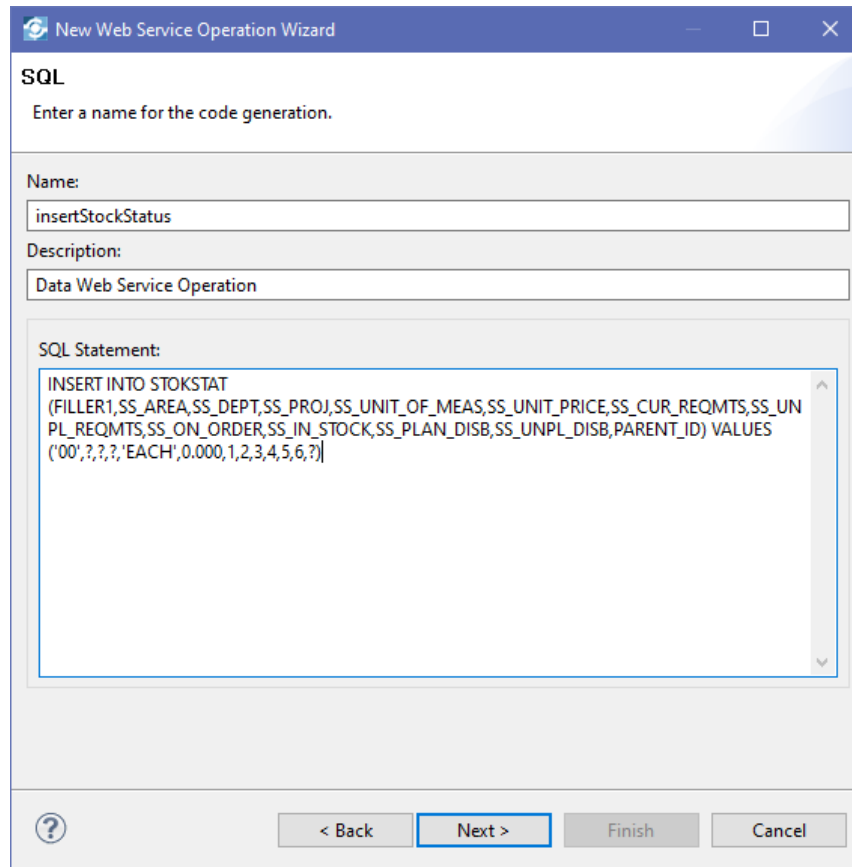
Name	Data Type	Default Value
description	VARCHAR	
partNumber	VARCHAR	

22. No result set for this operation so click Finish on the *SQL Result Set* window.



23. Select Use the *Create Operation* wizard under *PARTROOT* create a new operation. Select the *STOKSTAT* virtual table and click **Next**. This operation should be named *insertStockStatus* and the *SQL Statement* should be changed to:

```
INSERT INTO STOKSTAT  
(FILLER1,SS_AREA,SS_DEPT,SS_PROJ,SS_UNIT_OF_MEAS,SS_UNIT_PRICE,SS_CUR_REQM  
TS,SS_UNPL_REQMTS,SS_ON_ORDER,SS_IN_STOCK,SS_PLAN_DISB,SS_UNPL_DISB,PARE  
NT_ID) VALUES('00',?,?,?, 'EACH',0.000,1,2,3,4,5,6,?)
```



**New Web Service Operation Wizard**

**SQL**  
Enter a name for the code generation.

Name:  
insertStockStatus

Description:  
Data Web Service Operation

SQL Statement:  

```
INSERT INTO STOKSTAT  
(FILLER1,SS_AREA,SS_DEPT,SS_PROJ,SS_UNIT_OF_MEAS,SS_UNIT_PRICE,SS_CUR_REQMTS,SS_UN  
PL_REQMTS,SS_ON_ORDER,SS_IN_STOCK,SS_PLAN_DISB,SS_UNPL_DISB,PARENT_ID) VALUES  
('00',?,?,?, 'EACH',0.000,1,2,3,4,5,6,?)
```

Buttons: < Back, Next >, Finish, Cancel

24. Click **Next** to continue. Since a *WHERE* clause has been added with variable, providing a meaningful name for this variable will be required. The next window to be displayed, *SQL Inputs*, will give us an opportunity to give meaningful name to the variable. On this window click on the values of variable name in the *Name* column and change the contents to *area*, *dept*, *project* and *parentID*. Click **Next** to continue.

Name	Data Type	Default Value
area	VARCHAR	
dept	VARCHAR	
project	VARCHAR	
parentID	VARCHAR	

25. No result set is returned so click **Finish** on the *SQL Result Set* window.

26. Use the *Create Operation* wizard under *PARTROOT* create a new operation. Select the *STOKSTAT* virtual table. This operation should be named *selectStockStatus* and the *SQL Statement* should be changed to:

```
SELECT  
(FILLER1,SS_AREA,SS_DEPT,SS_PROJ,SS_UNIT_OF_MEAS,SS_UNIT_PRICE,SS_CUR_REQMTS,  
SS_UNPL_REQMTS,SS_ON_ORDER,SS_IN_STOCK,SS_PLAN_DISB,SS_UNPL_DISB,PARENT_ID)  
FROM STOKSTAT WHERE PARENT_ID=?
```

**New Web Service Operation Wizard**

**SQL**  
Enter a name for the code generation.

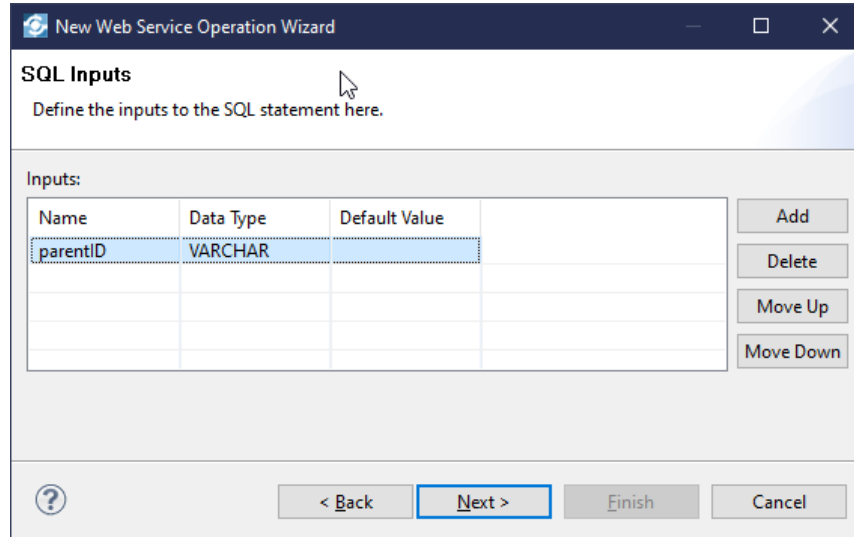
Name:  
selectStockStatus

Description:  
Select Stock Status segments for a specific root segment

SQL Statement:  
SELECT  
(FILLER1,SS\_AREA,SS\_DEPT,SS\_PROJ,SS\_UNIT\_OF\_MEAS,SS\_UNIT\_PRICE,SS\_CUR\_REQMTS,SS\_UNPL\_REQMTS,SS\_ON\_ORDER,SS\_IN\_STOCK,SS\_PLAN\_DISB,SS\_UNPL\_DISB,PARENT\_ID) FROM  
STOKSTAT WHERE PARENT\_ID=?

? < Back Next > Finish Cancel

27. Click **Next** to continue. Since a *WHERE* clause has been added with variable, providing a meaningful name for this variable will be required. The next window to be displayed, *SQL Inputs*, will give us an opportunity to give meaningful name to the variable. On this window click on the values of variable name in the *Name* column and change the contents to *parentID*. Click **Next** to continue.

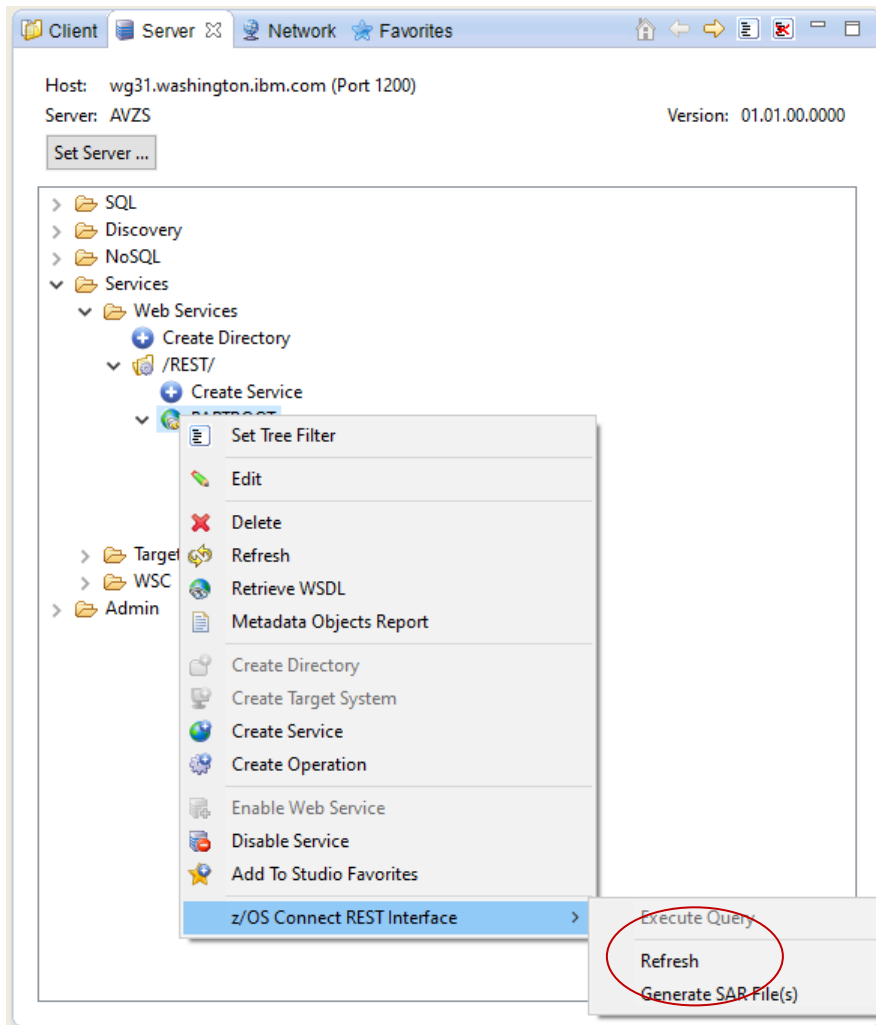


Name	Data Type	Default Value
parentID	VARCHAR	

28. Click **Finish** after optionally changing the names of the fields in the result set.

## Use the Data Virtualization Manager Studio to deploy the services

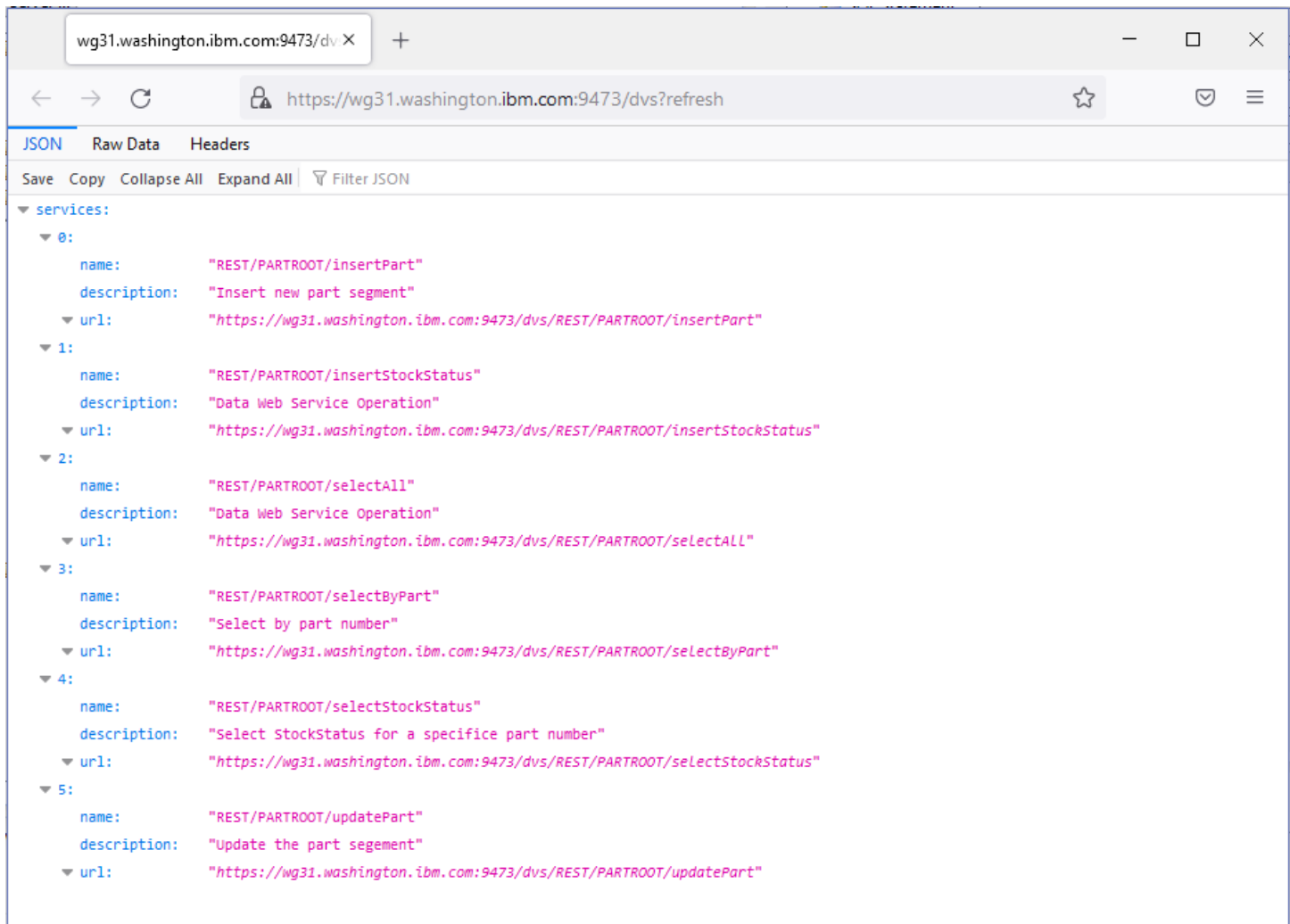
1. These operations need to be deployed to z/OS Connect server. Select *PARTROOT* folder under */REST/* and right mouse button click. Select the *z/OS Connect REST Interface* option then the *Refresh* option. This will install the operations in *PARTROOT* into the z/OS Connect server as services.



**Tech Tip:** Operations can be deployed individually by selecting the specific operation and right mouse button clicking and selecting **Refresh**.

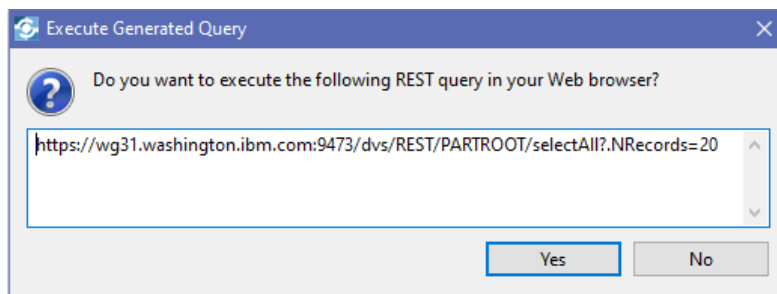
**Tech Tip:** You may be challenged by Firefox because the digital certificate used by the Liberty z/OS server is self-signed Click the **Advanced** button to continue. Scroll down and then click on the **Accept the Risk and Continue** button. Next you may see a prompt you for a userid and password. If you do see the prompt, enter the username **USER1** and password **USER1** and click **OK**.

2. When finished all the operations should be displayed as services in the z/OS Connect server by entering URL **<https://wg31.washington.ibm.com:9473/dvs>**.

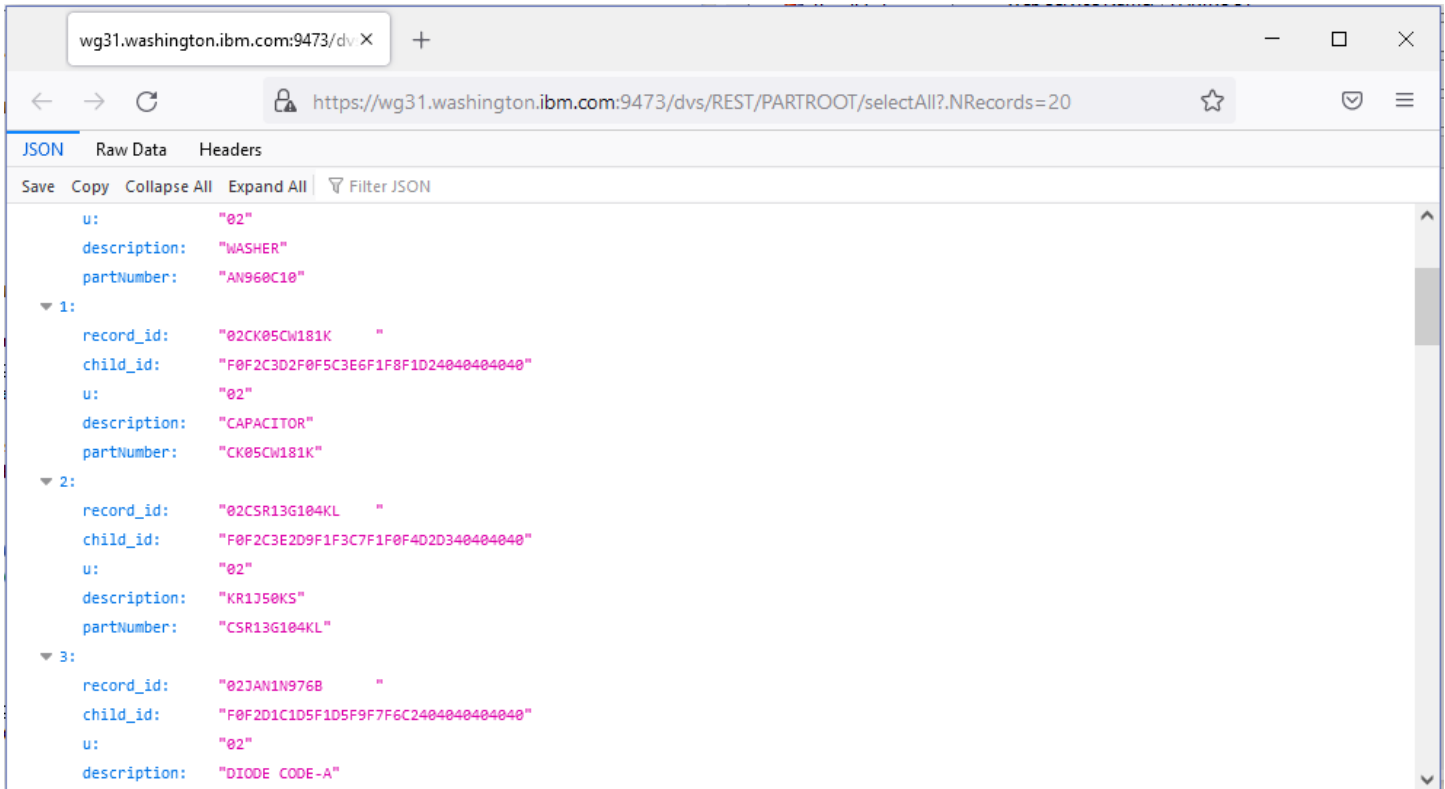


3. A subset of these z/OS Connect services can now be tested using the DVM Data Manager Studio (only the services that do selects) Select the *selectAll* operation and right mouse button click. Select the *z/OS Connect REST Interface* option then the *Execute Query* option.

4. This pop-up window should be displayed. Click **OK** to continue.

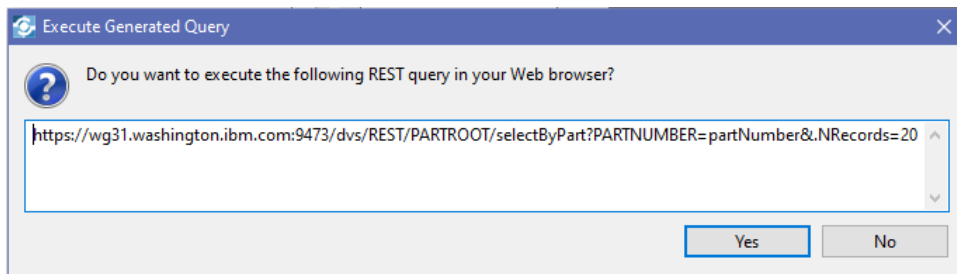


5. A web browser session should open with results like the one below.

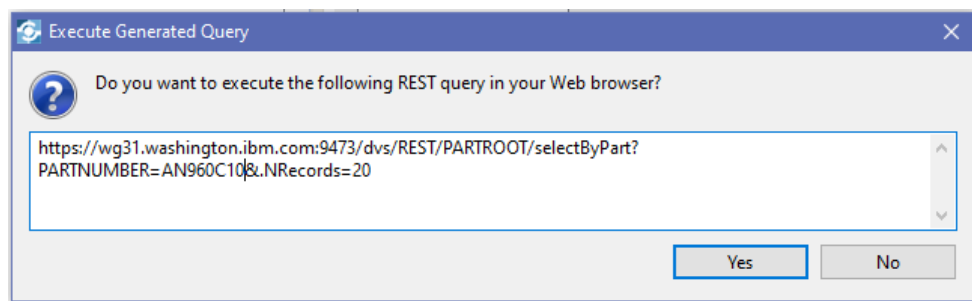


6. Select the *selectByPart* operation and right mouse button click. Select the *z/OS Connect REST Interface* option then the *Execute Query* option

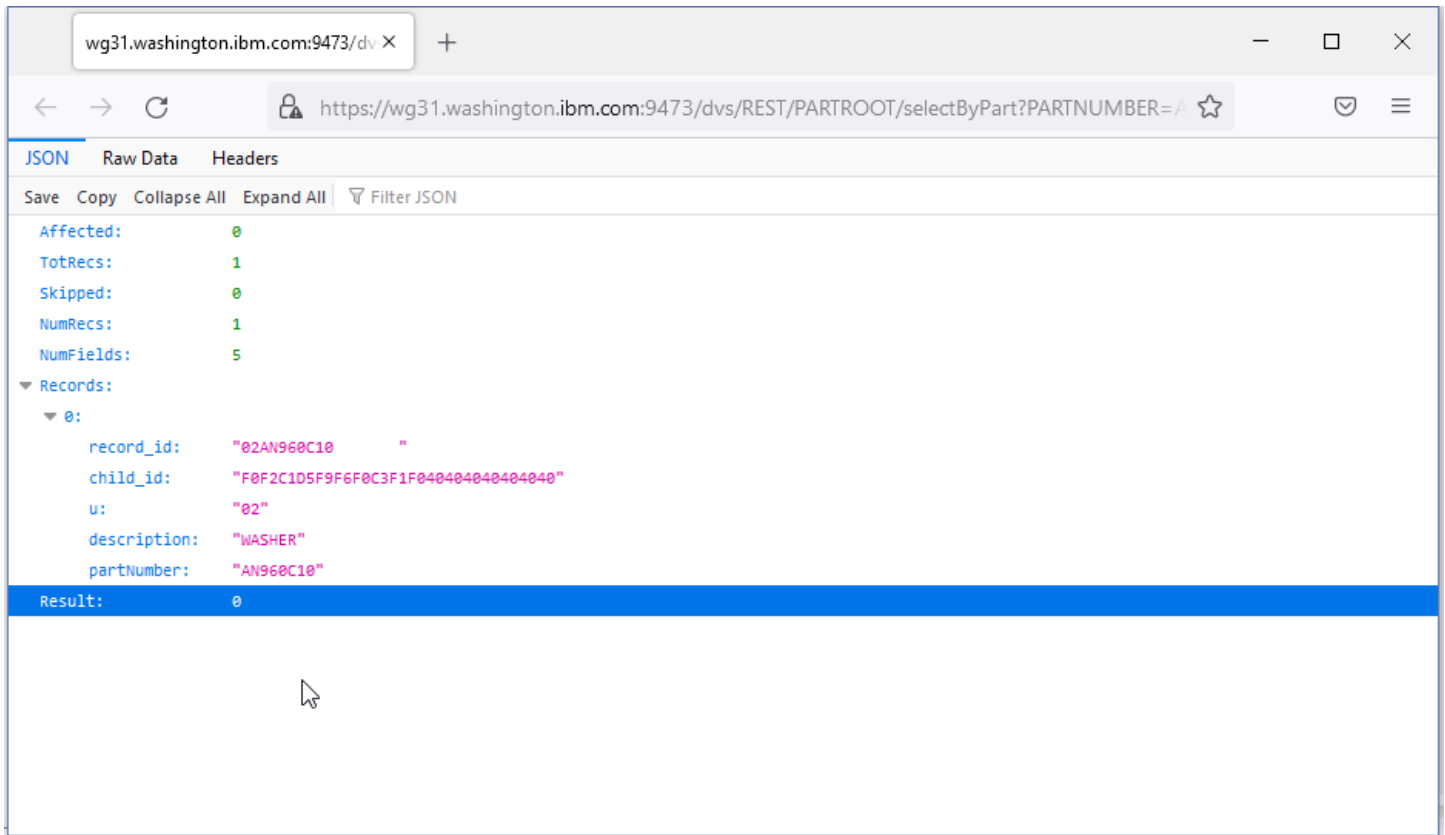
7. This pop-up window should be displayed.



8. Change the string *PARTNUMBER=partNumber* to ***PARTNUMBER=AN960C10***



9. Click **Yes** and a web browser tab should be opened with results like these.

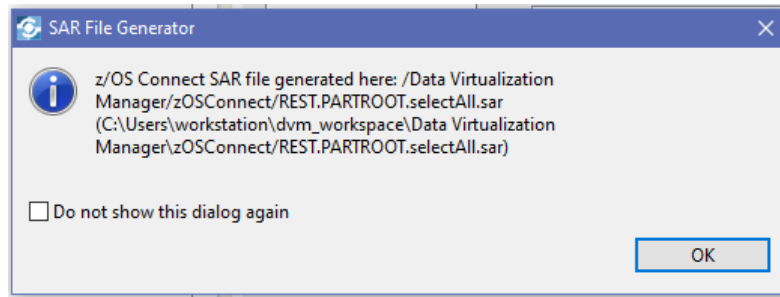


**Tech-Tip:** The above results show some new fields in the response messages. Their meanings are provided below:

Affected:	The number of records deleted, updated or inserted by this request.
TotRecs:	The number of records found.
Skipped:	The number of records skipped
NumRecs:	The number of records returned.
NumFields:	The number of fields returned for each record.



10. Finally, a z/OS Connect service archive (SAR) files need to be exported from the DVM Studio for use in the z/OS Connect API Editor. Select *PARTROOT* and right mouse button click. Select the *z/OS Connect REST Interface* option then the *Generate SAR File(s)* option. A pop-up window will appear for each SAR file to be exported, click **OK** on each.



This exports the SAR files to a subdirectory in the DVM Toolkit's workspace directory, e.g., *C:\Users\workstation\dvm\_workspace\Data Virtualization Manager\zOSConnect*. This pop-up will be

**Tech-Tip:** The directory where the SAR file is exported may be different on your system. Make a note of this directory name so you will know from where to import the SAR file later. On some images, this directory will be *C:\Users\administrator\dvm\_workspace\Data Virtualization Manager\zOSConnect*.

repeated for each operation. This directory will be referenced in a latter section of this exercise.

## Create z/OS Connect APIs

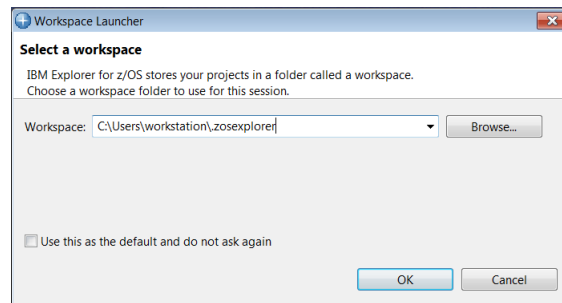
### Connect to a z/OS Connect Server

Begin by establishing a connection to DVM z/OS Connect server from IBM z/OS Explorer.

1. On the workstation desktop, locate the *z/OS Explorer* icon and double click on it to open the Explorer.

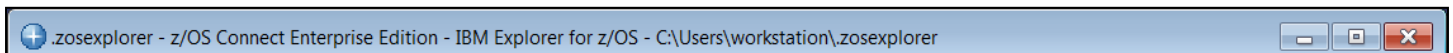
**Tech-Tip:** Windows desktop tools can be opened either by double clicking the icon or by selecting the icon and right mouse button clicking and then selecting the *Open* option.

2. You will be prompted for a workspace:



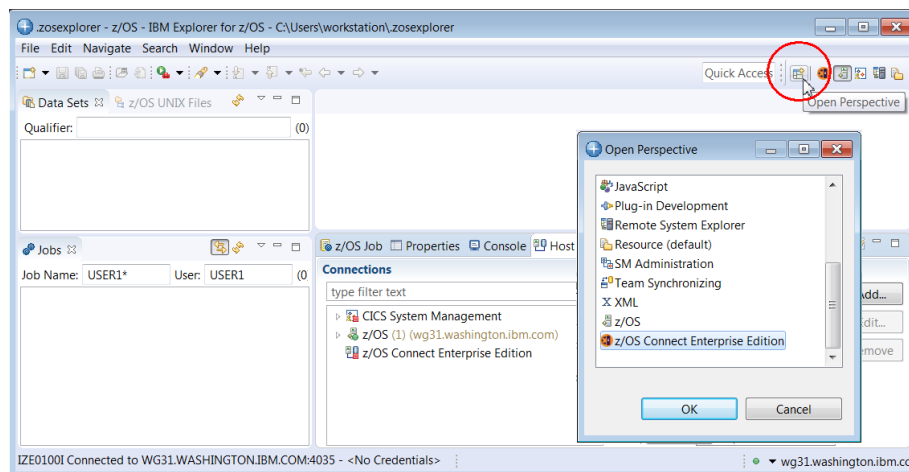
Take the default value by clicking **OK**.

3. The Explorer should open in the *z/OS Connect Enterprise Edition* perspective. Verify this by looking in the upper left corner. You should see:

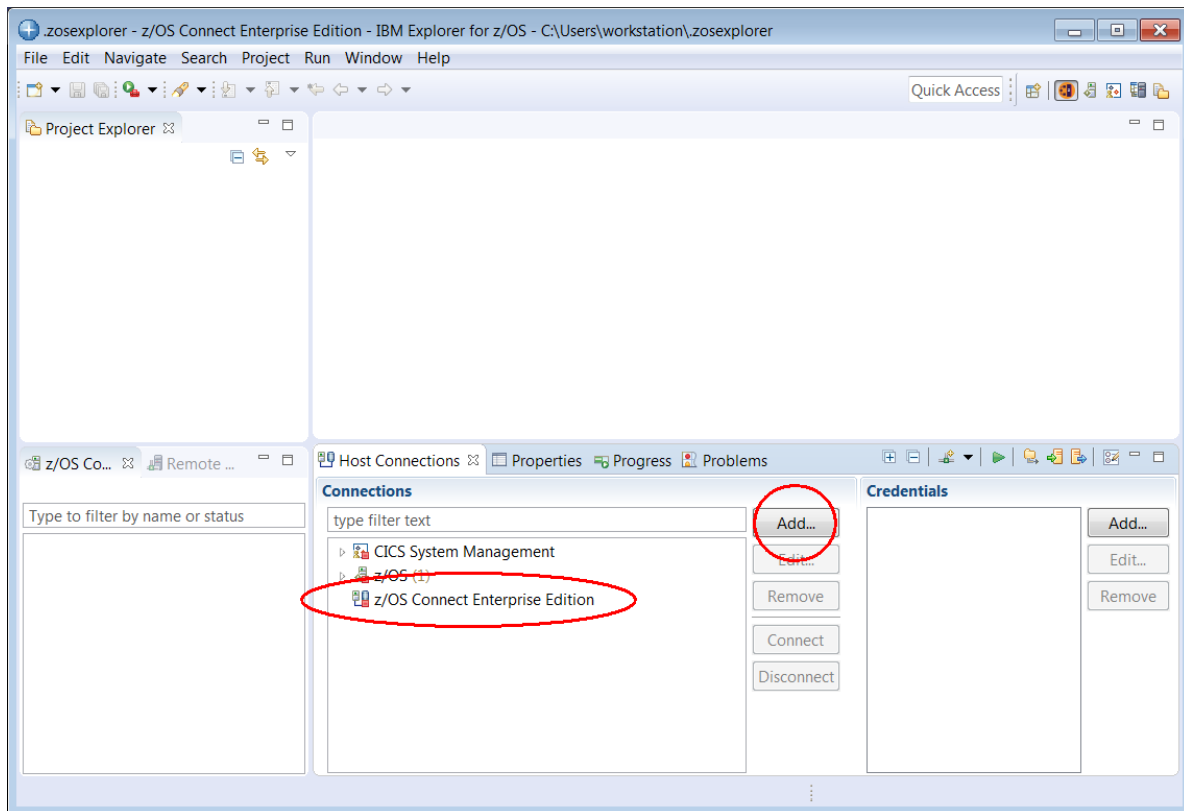


N.B. If a *Welcome* screen is displayed then click the white X beside *Welcome* to close this view.

4. If the current perspective is not *z/OS Connect Enterprise Edition*, select the *Open Perspective* icon on the top right side to display the list of available perspectives, see below. Select **z/OS Connect Enterprise Edition** and click the **OK** button to switch to this perspective.



5. To add a connection to the z/OS Connect server, select *z/OS Connect Enterprise Edition* connection in the *Host connections* tab in the lower view and then click the **Add** button.



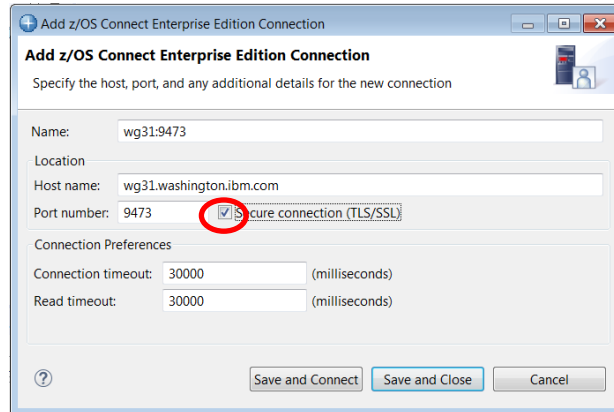
**Tech-Tip:** Eclipse based development tools like z/OS Explorer; provide a graphical interface consisting of multiple views within a single window.

A view is an area in the window dedicated to providing a specific tool or function. For example, in the window above, *Host Connections* and *Project Explorer* are views that use different areas of the window for displaying information. At bottom on the right there is a single area for displaying the contents of four views stacked together (commonly called a *stacked views*), *z/OS Host Connections*, *Properties*, *Progress* and *Problems*. In a stacked view, the contents of each view can be displayed by clicking on the view tab (the name of the view).

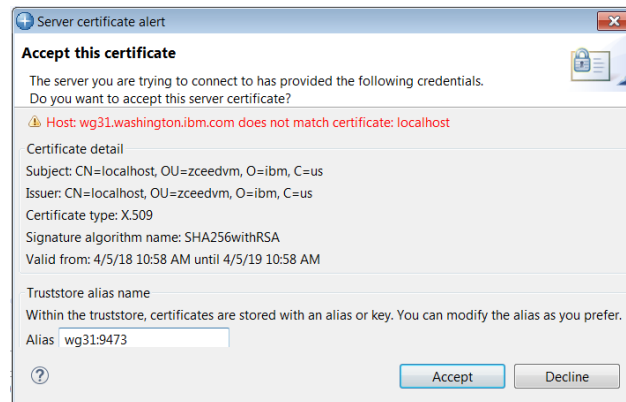
At any time, a specific view can be enlarged to fill the entire window by double clicking in the view's title bar. Double clicking in the view's title bar will be restored the original arrangement. If a z/OS Explorer view is closed or otherwise disappears, the original arrangement can be restored by selecting Windows → Reset Perspective in the window's tool bar.

Eclipse based tools also can display multiple views based on the current role of the user. In this context, a window is known as a perspective. The contents (or views) of a perspective are based on the role the user, i.e., developer or administrator.

6. In the pop-up list displayed, select *z/OS Connect Enterprise Edition* and on the *Add z/OS Connect Enterprise Edition Connection* window enter **wg31.washington.ibm.com** for the *Host name*, 9473 for the *Port Number*, check the box for *Secure connection (TLS/SSL)* and then click the **Save and Connect** button.



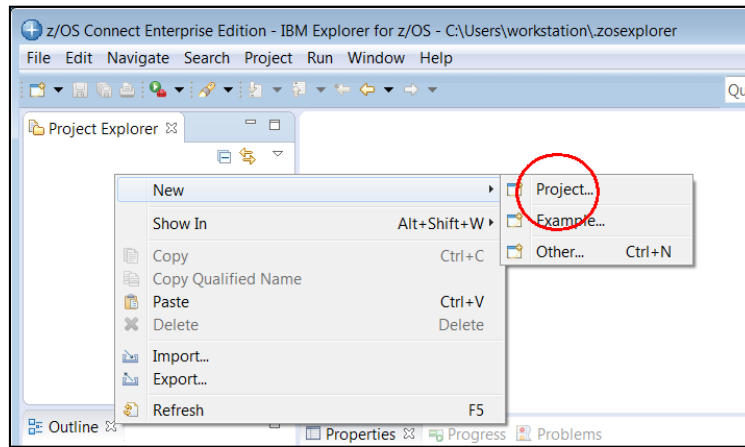
7. On the *z/OS Connect Enterprise Edition – User ID* required screen, create new credentials for a *User ID* of **USER1** and for *Password or Passphrase* enter **USER1**'s password. Click **OK** to continue.
8. Click the **Accept** button on the *Server certificate alert – Accept this certificate* screen. You may be presented with another prompt for a *userid* and *password*, enter **USER1** and **USER1**'s password again.



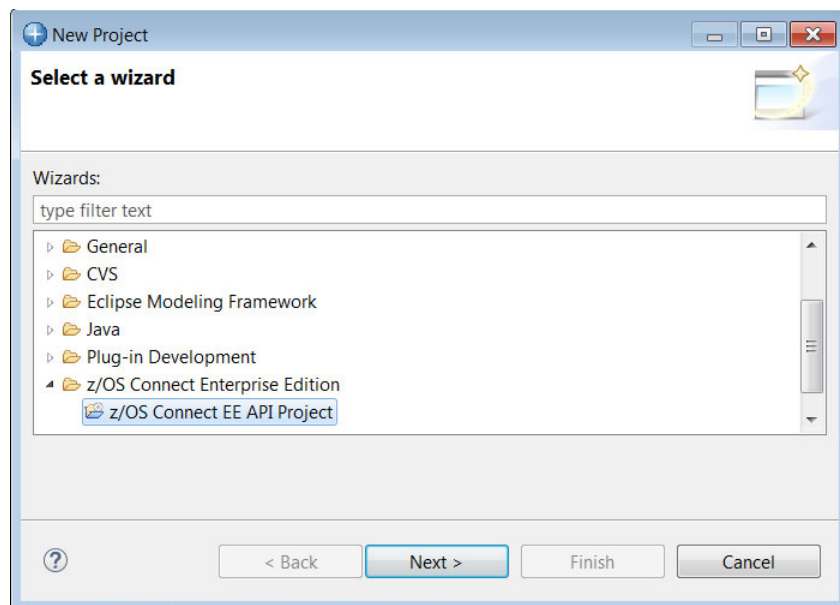
9. The status icon beside **wg31:9473** should now be a green circle with a lock. This shows that a secure connection has been established between the *z/OS Explorer* and the *z/OS Connect* server. A red box indicates that no connection exists.

## Create the IMS DVM API Project

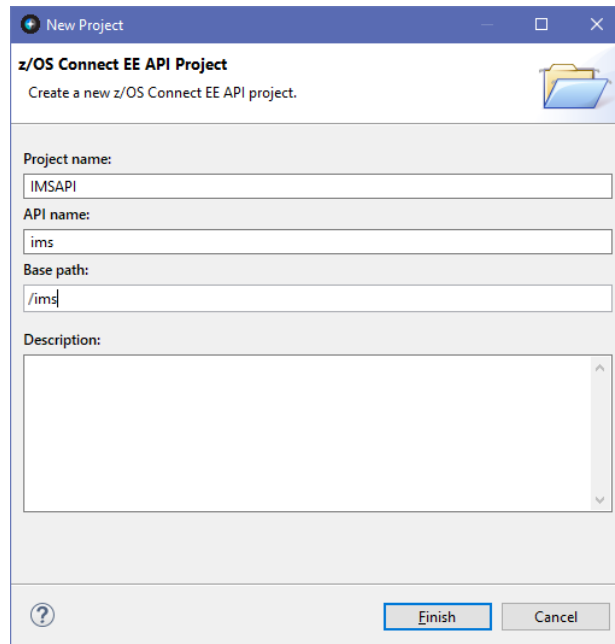
1. In the *z/OS Connect Enterprise Edition* perspective of the z/OS Explorer, create a new API project by clicking the right mouse button and selecting *New* → *Project*:



2. In the *New Project* window, scroll down and open the *z/OS Connect Enterprise Edition* folder and select *z/OS Connect API Project* and then click the **Next** button.



3. Enter **IMSAPI** for the *Project name*. Be sure the *API name* is set to **ims** and the *Base path* is set to **/ims**



**New Project**

**z/OS Connect EE API Project**  
Create a new z/OS Connect EE API project.

Project name:

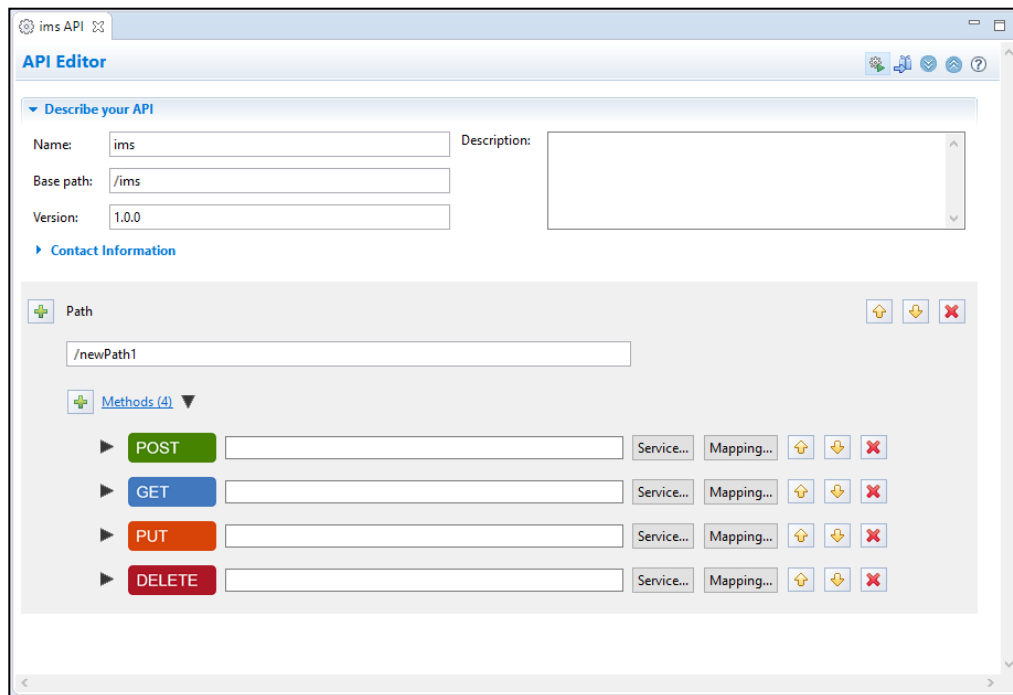
API name:

Base path:

Description:

**Note:** The Base path name of **/ims** is used to distinguish a request for this API from other APIs in the same server. It can be any value as long as the value is unique within the server. The same is true of any sub path names added to the base path. Sub path names are used to distinguish one service from another within an API.

4. You should now see something like the view below. The view may need to be adjusted by dragging the view boundary lines.



**ims API**

**API Editor**

**Describe your API**

Name:  Description:

Base path:

Version:

**Contact Information**

**Path**

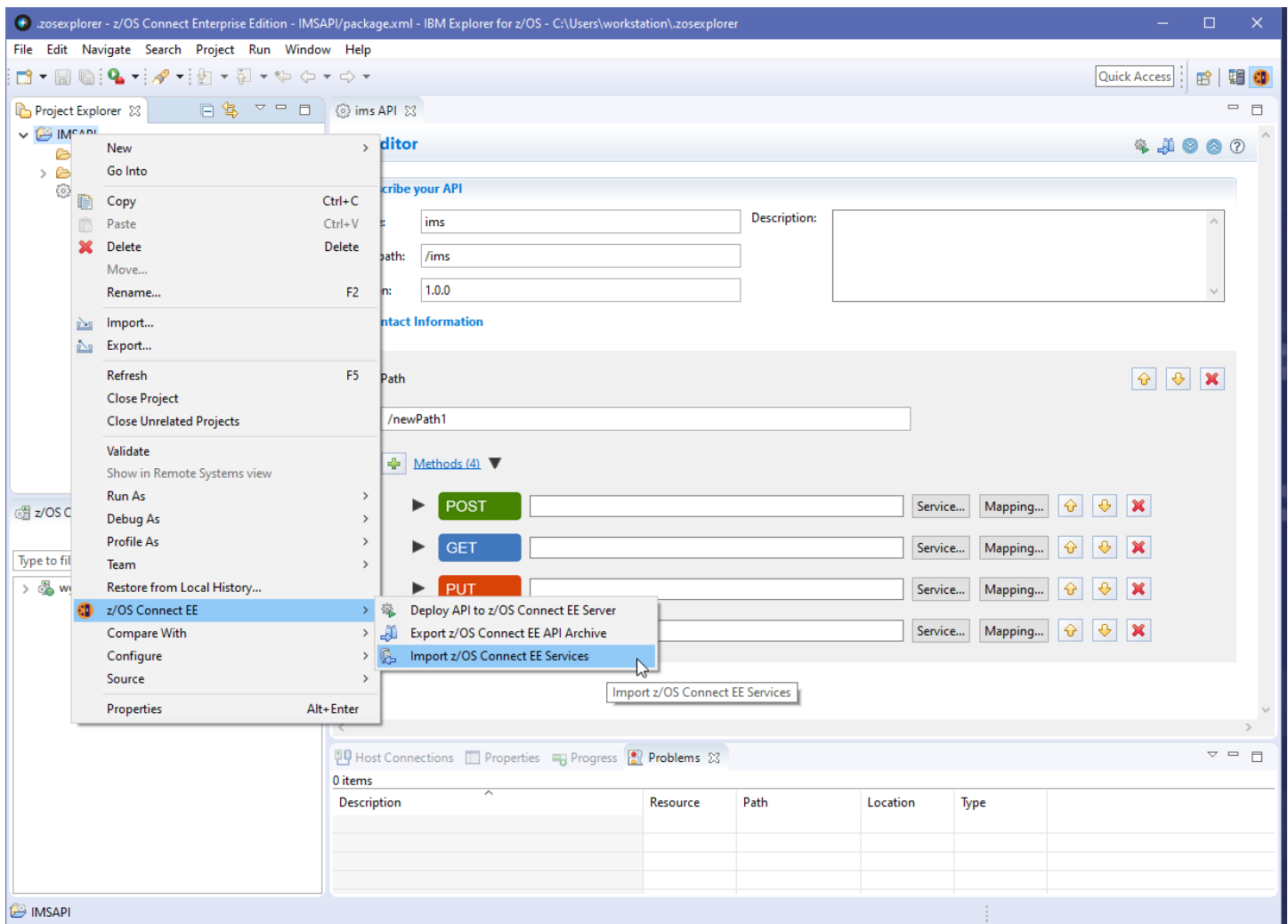
**Methods (4)**

Method	Path	Service...	Mapping...	Up	Down	Delete
POST	<input type="text"/>	Service...	Mapping...	<input type="button" value="Up"/>	<input type="button" value="Down"/>	<input type="button" value="X"/>
GET	<input type="text"/>	Service...	Mapping...	<input type="button" value="Up"/>	<input type="button" value="Down"/>	<input type="button" value="X"/>
PUT	<input type="text"/>	Service...	Mapping...	<input type="button" value="Up"/>	<input type="button" value="Down"/>	<input type="button" value="X"/>
DELETE	<input type="text"/>	Service...	Mapping...	<input type="button" value="Up"/>	<input type="button" value="Down"/>	<input type="button" value="X"/>

**Tech-Tip:** If the API Editor view is closed, it can be reopened by double clicking the *package.xml* file in the API project.

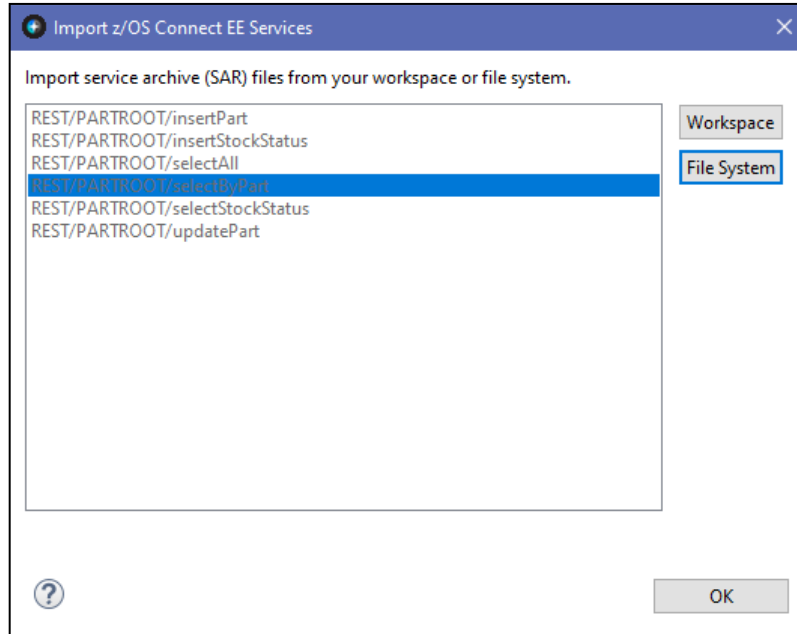
## Import the SAR files generated by the DVM Studio

1. In the z/OS Explorer in the z/OS Connect Enterprise Edition perspective in the the *Project Explorer* view (upper left), right-click on the *IMSAPI* project, then select *z/OS Connect* and then *Import z/OS Connect Services* (see below):



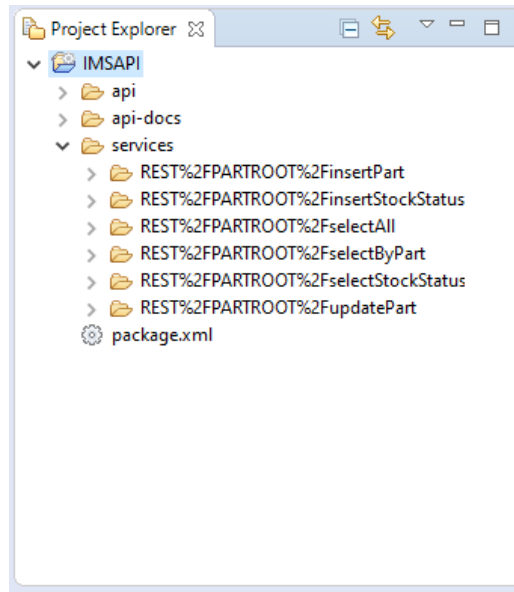
**Tech-Tip:** Remember from step 10 on page 49, the directory where the SAR file is to be imported from may be different on your system. On some images, this directory will be *C:\Users\administrator\dvm\_workspace\Data Virtualization Manager\zOSConnect*.

2. In the *Import z/OS Connect Services* window click on the **File System** button and navigate to directory *C:\Users\workstation\dvm\_workspace\Data Virtualization Manager\zOSConnect*. Select all the SAR files and click on the **Open** button. (Hint: use the *Ctrl-A* key sequence to select all the files).
3. The service archive files should appear in the *Import Services* window. Click the **OK** button twice to import them into the workspace.





4. In the *Project Explorer* view (upper left), expand the *services* folder to see the imported service:



## Compose an API for the IMS DVM Rest Services

- \_\_\_ 1. Start by entering a *Path* of */parts* in the *z/OS Connect API Editor* view as shown below:

The screenshot displays the 'API Editor' window for an API named 'ims'. The 'Describe your API' section is expanded, showing the following fields:

- Name:
- Base path:
- Version:
- Description:

Below these fields is the 'Contact Information' section. The 'Path' field is set to 

Below the path field is the 'Methods (4)' section, which lists four methods:

- POST:  Service... Mapping...
- GET:  Service... Mapping...
- PUT:  Service... Mapping...
- DELETE:  Service... Mapping...

2. The initial API to be added will be when no path or query parameter will be required, the supported HTTP methods will only be the **GET** method. Remove the **POST**, **PUT** and **DELETE** methods by clicking the red **X** icon to the right of each method.

The screenshot shows the API editor for 'ims API'. The Name is 'ims', Base path is '/ims', and Version is '1.0.0'. Under 'Contact Information', the Path is '/parts'. The 'Methods (4)' section lists GET, POST, DELETE, and PUT. Each method has a 'Service...' button, a 'Mapping...' button, and three action icons (up, down, and a red 'X'). The red 'X' icons for POST, DELETE, and PUT are circled in red, indicating they should be removed.

3. That should leave you with just the **GET** method.

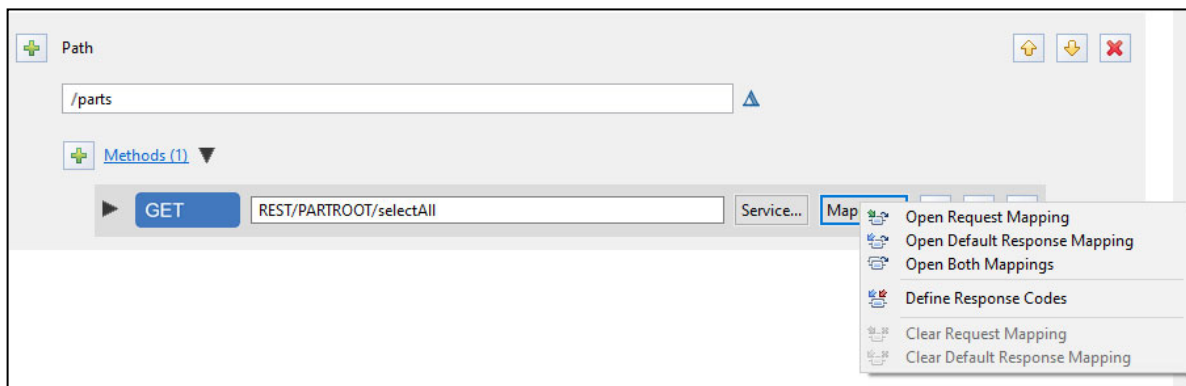
4. Click on the **Service** button to the right of the **GET** method. Then select the *REST/PARTROOT/selectAll* service from the list of services and click **OK**. This will populate the field to the right of the method.

The screenshot shows the API editor after removing the POST, DELETE, and PUT methods. Only the GET method is listed. The 'Service...' button has been clicked, and the field to the right of the method is populated with 'REST/PARTROOT/selectAll'.

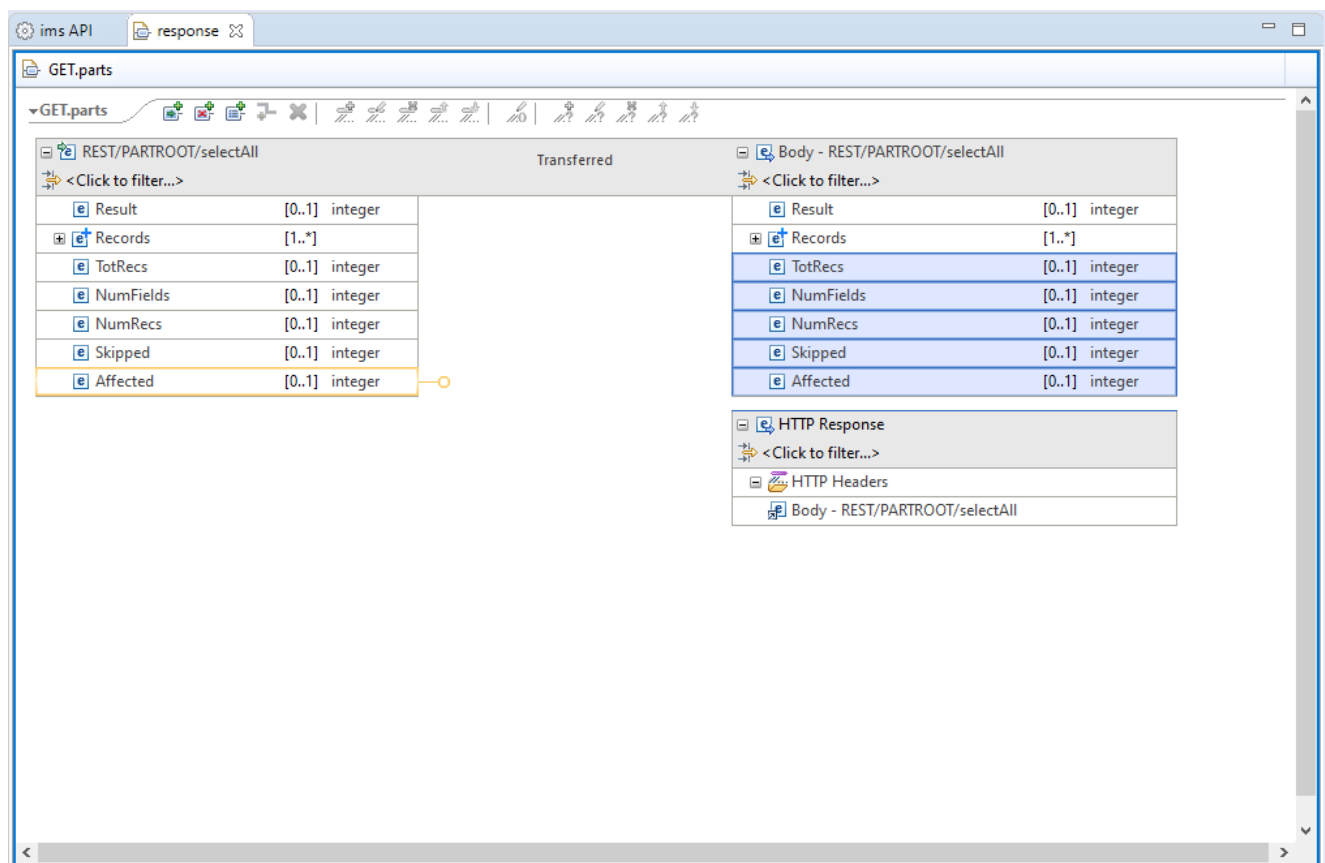
5. Save the changes so far by using the key sequence **Ctrl-S**.

**Tech-Tip:** If any change is made in any edit view an asterisk (\*) will appear before the name of the artifact in the view tab, e.g., *\*package.xml*. Changes can be saved at any time by using the **Ctrl-S** key sequence.

6. Next, click on the **Mapping** button beside the **GET** method and then select *Open Default Response Mapping*:

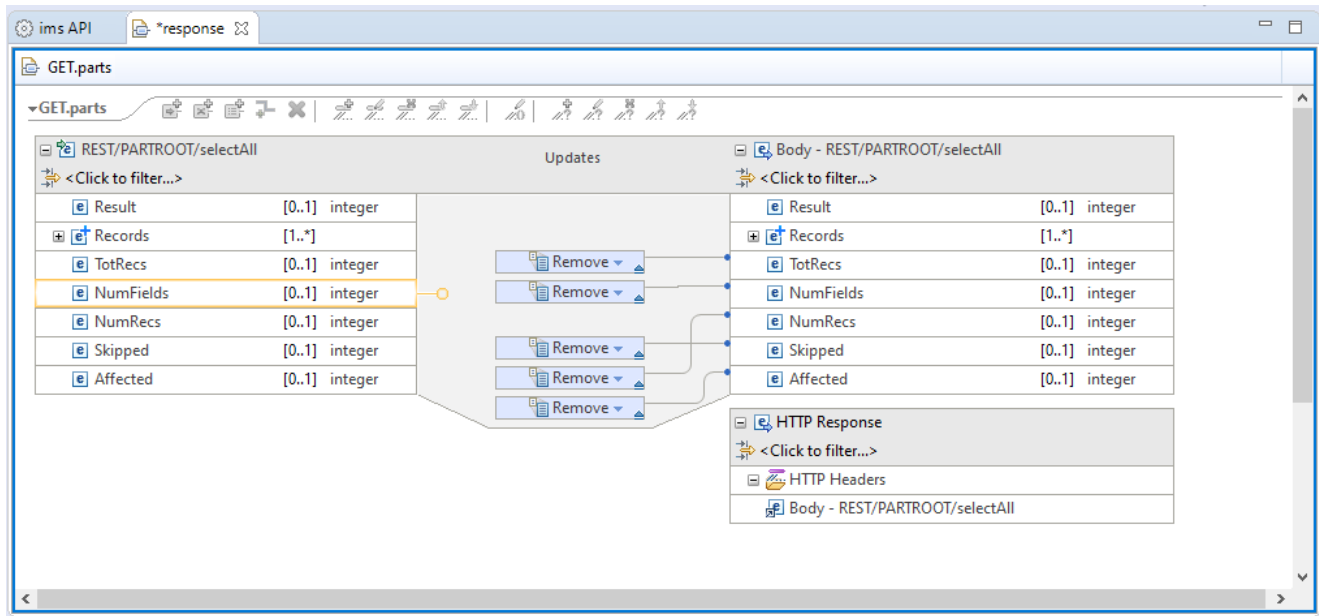


7. Use the left mouse button and draw a dotted line box that **fully** includes the *TotRecs*, *NumFields*, *NumRecs*, *Skipped* and *Affected* fields. When you release the button, these fields should be selected (the background should be blue).

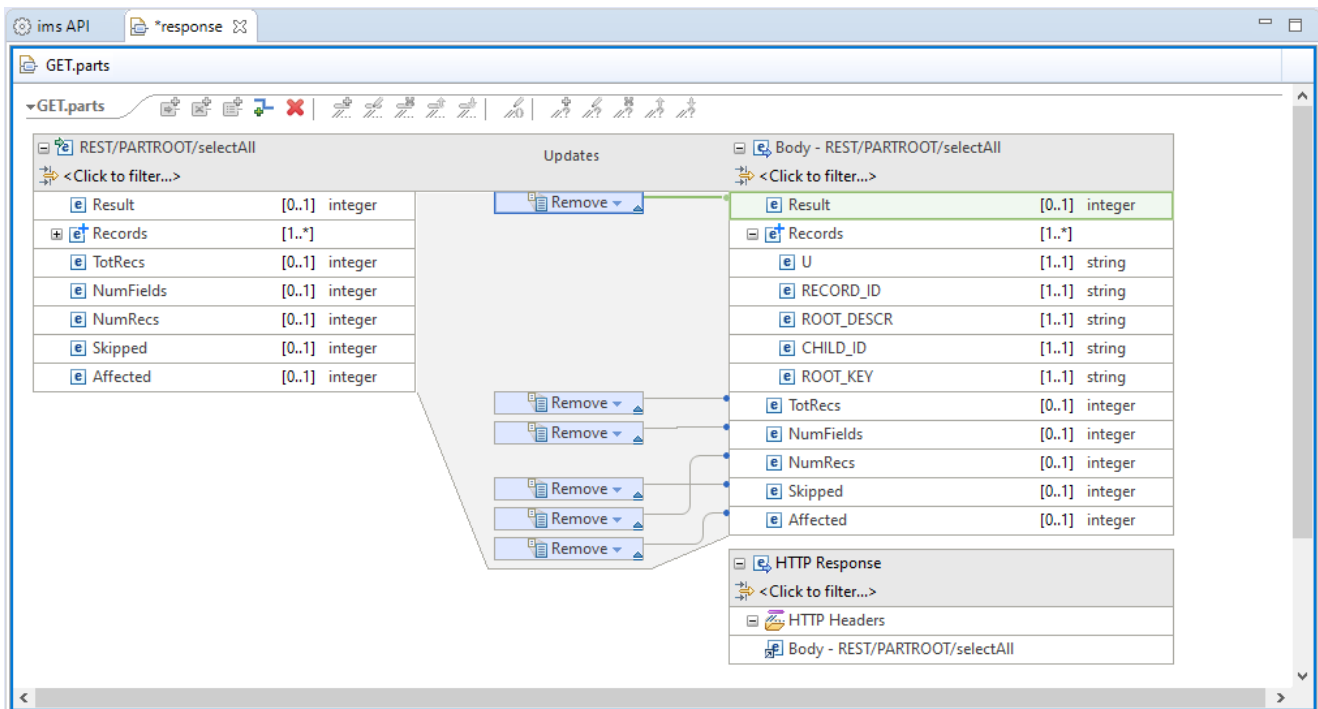


**Tech-Tip:** This step is being done just to show how fields can be removed from the service interface response message. Response fields like *TotRecs*, *Affected*, *NumRecs*, etc. can be checked like this to set appropriate HTTP response codes. For example, if you were doing a GET (a SELECT) and the value of *NumRecs* was zero, the HTTP response code could be set to 404 – *Not Found*.

8. Right mouse button click on any of the selected fields and select the *Add Remove transform* from the list of options.
9. This action generates multiple “Remove” requests (see below) for the selected fields. These fields are not required to be display so they will be removed from the response.



10. Select the *Result* field and remove it from the response. If not expanded already, expand the *Records* structure and you should see the ‘columns’ that will be displayed in the response.



- \_\_\_ 11. Use the **Ctrl-S** key sequence to save all changes and close the *GET.item* response view.
- \_\_\_ 12. Next, click on the **Mapping** button beside the **GET** method and then select *Open Request Mapping* for this method. Note there are no fields in a request message. Close the request view.
- \_\_\_ 13. Next, we want to add a *Path* for a **POST** method for the *insertPart* service. Click the plus icon beside *Path* on the z/OS Connect API Editor view to add another path to the API.

**Tech-Tip:** Additional *Paths* can be added by clicking the + icon beside *Path* and additional *Methods* can be added by clicking the + icon beside *Methods*.

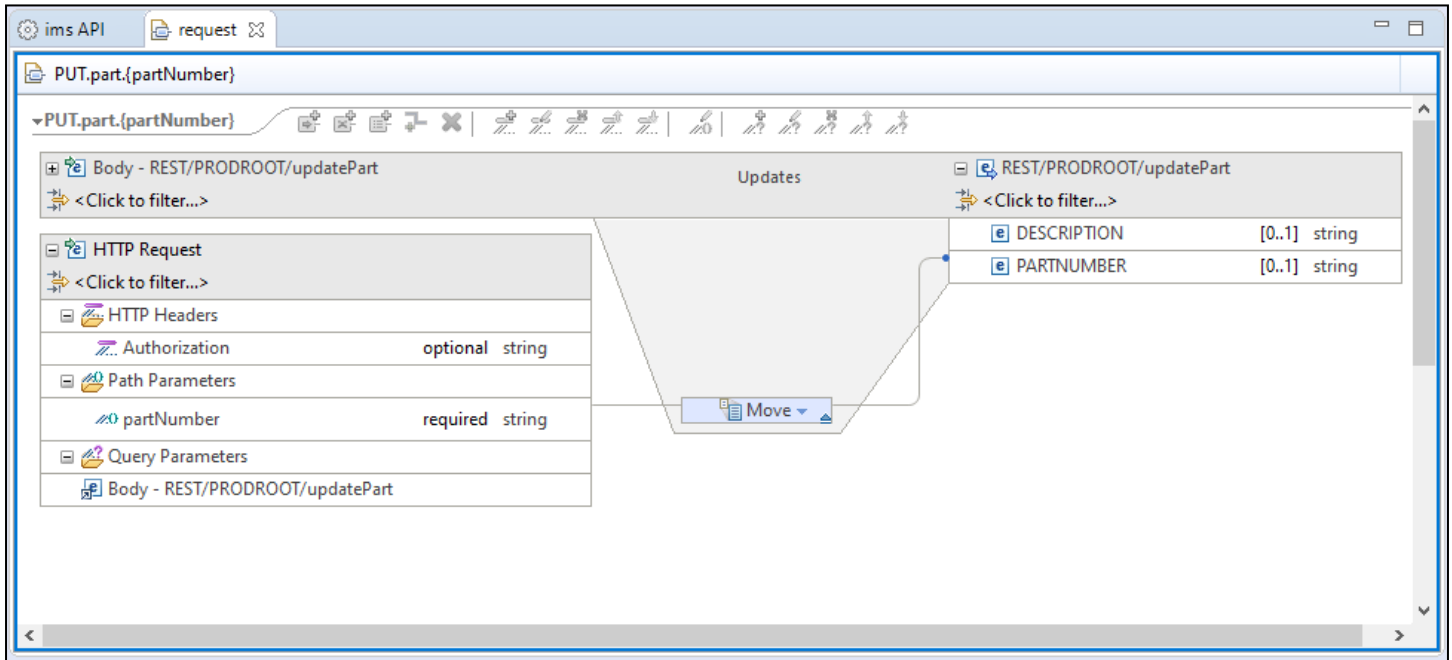
- \_\_\_ 14. Enter a path value of */part* and remove the **PUT**, **GET** and **DELETE** methods. Associate service *REST/PRODROOT/insertPart* with the **POST** method and path.

**Note:** The */part* path again is somewhat arbitrary, but again it is used to distinguish this request from other requests that may be configured in the same API.

The full URL to invoke the methods for this path of the API will be  
<https://hostname:port/ims/part>

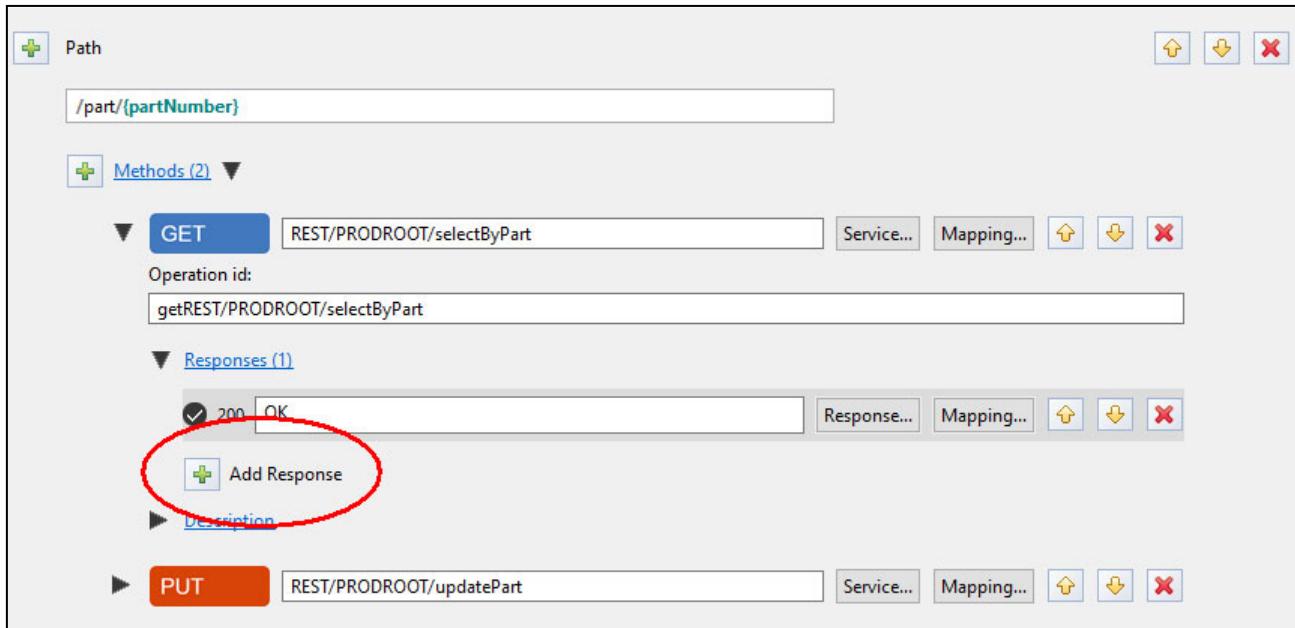
- \_\_\_ 15. Save the changes by using the key sequence **Ctrl-S**.
- \_\_\_ 16. Click on the **Mapping** button beside the **POST** method and then select *Open Request Mapping* for this method. Note the fields in a request message. Close the request view.
- \_\_\_ 17. Save the changes using the key sequence **Ctrl-S** and close the *POST.part* request view.
- \_\_\_ 18. Next, click on the **Mapping** button beside the **POST** method and then select *Open Default Response Mapping*: Remove the *Result*, *TotRecs*, *NumFields*, *NumRecs*, *Skipped*, and *Affected* fields from the response.
- \_\_\_ 19. Save the changes using the key sequence **Ctrl-S** and close the *POST.part* response view.
- \_\_\_ 20. Next, we want to add a *Path* for **GET** and **PUT** method for retrieve and updating of a specific part root segment. Click the plus icon beside *Path* on the z/OS Connect API Editor view to add path */part/{partNumber}* to the API.

21. Remove the **POST** and **DELETE** methods. Associate the **GET** method with service *REST/PRODROOT/selectByPart* and the **PUT** method with service *REST/PRODROOT/updatePart*.
22. For the **GET** methods, use the **Mapping** button to open the request message and map the path parameter *partNumber* to the *partNumber* field in the request message. This is done by selecting *partNumber* on the left-hand side and dragging it over to *partNumber* on the right-hand side to make a *Move* connection so the value or contents of the *partNumber* path parameter are moved into *partNumber* field of the request.

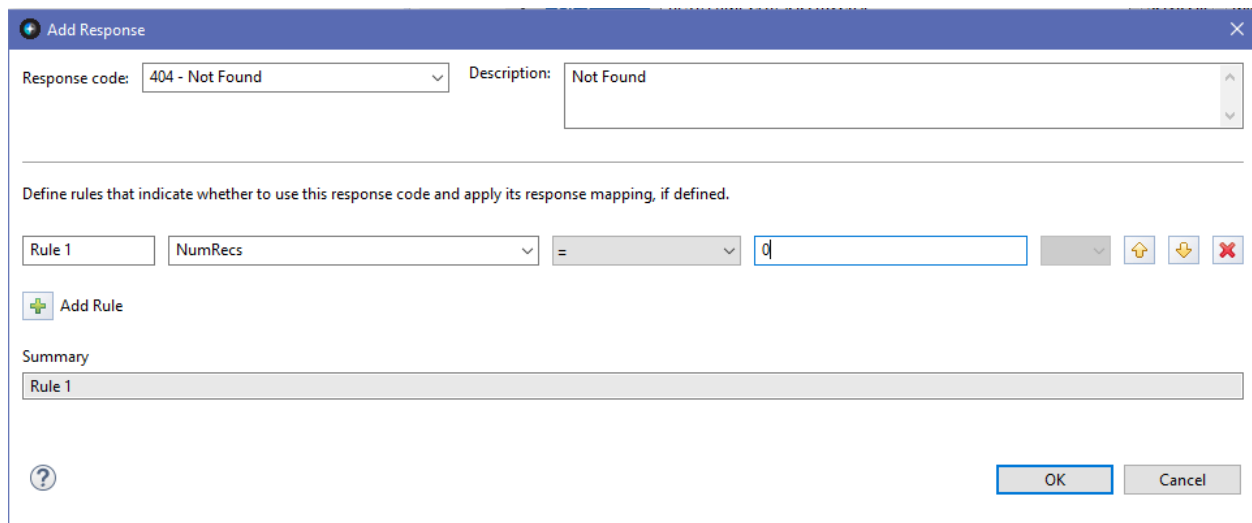


23. Repeat this mapping for the **PUT** method.
24. Define a *Response Code* for **PUT** request that if the number of records returned equal zero that will set the HTTP response code to 404. Click on the **Mapping** button and select the *Define Response Codes* option.

25. Click the plus sign beside **Add Response** to open the *Add Response* window.



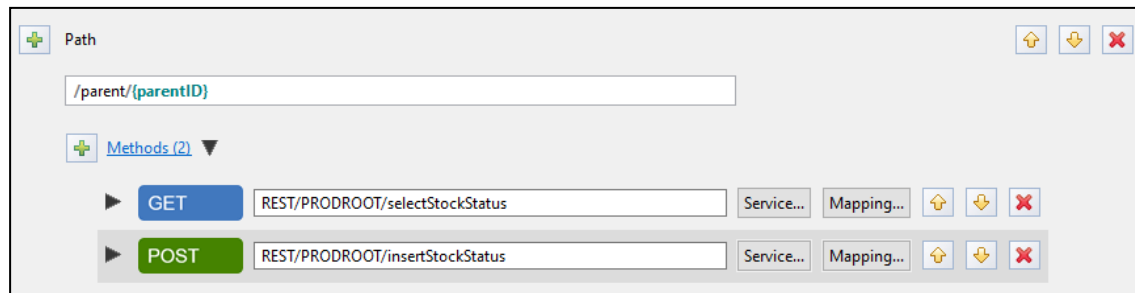
26. Use the pull-down arrow to select *404 – Not Found* for the *Response Code*. Use the pull-down arrows to select field *NumRecs* and the equal sign for *Rule 1*. Enter **0** in the open area for *Rule 1*. When finished your windows should look like the one below. Click **OK** to continue.



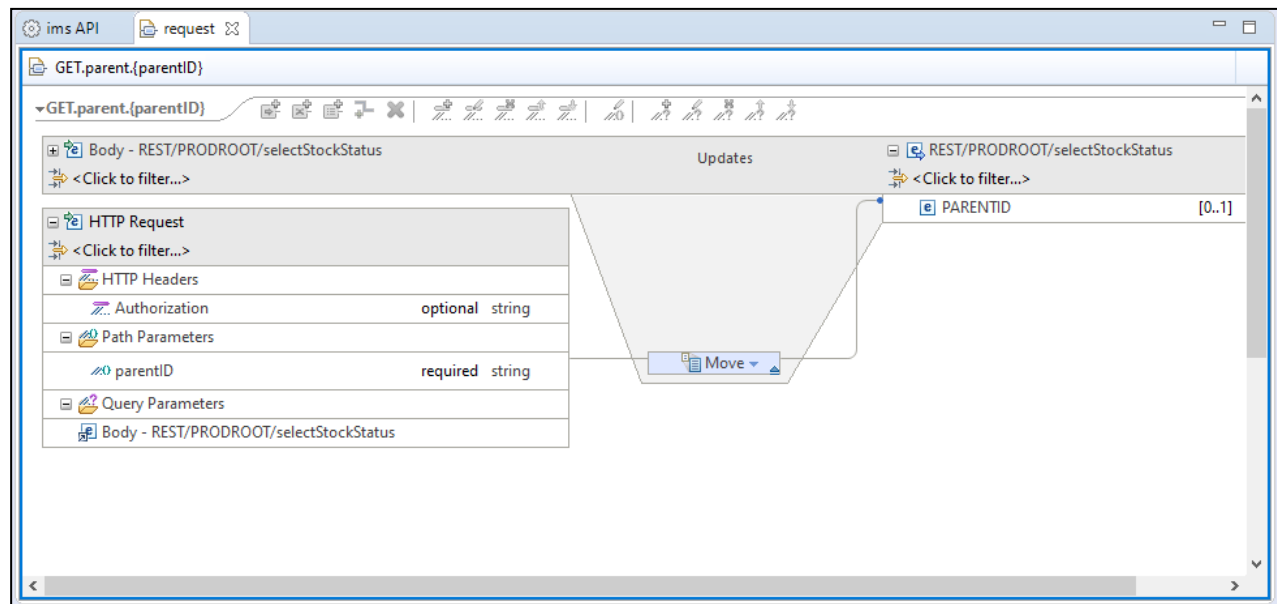
27. Next, we want to add a *Path* for **GET** and **POST** method for insert and retrieving a stock status segment of a specific part root segment. Click the plus icon beside *Path* on the z/OS Connect API Editor view to add path */parent/{parentID}* to the API.



28. Remove the **PUT** and **DELETE** methods. Associate the **GET** method with service *REST/PRODROOT/selectyStockStatus* and the **POST** method with service *REST/PRODROOT/insertStockStatus*.



29. For both the **GET** and **POST** methods, use the **Mapping** button to open the request message and map the path parameter *parentID* to the *parentID* field in the request message, as shown earlier.

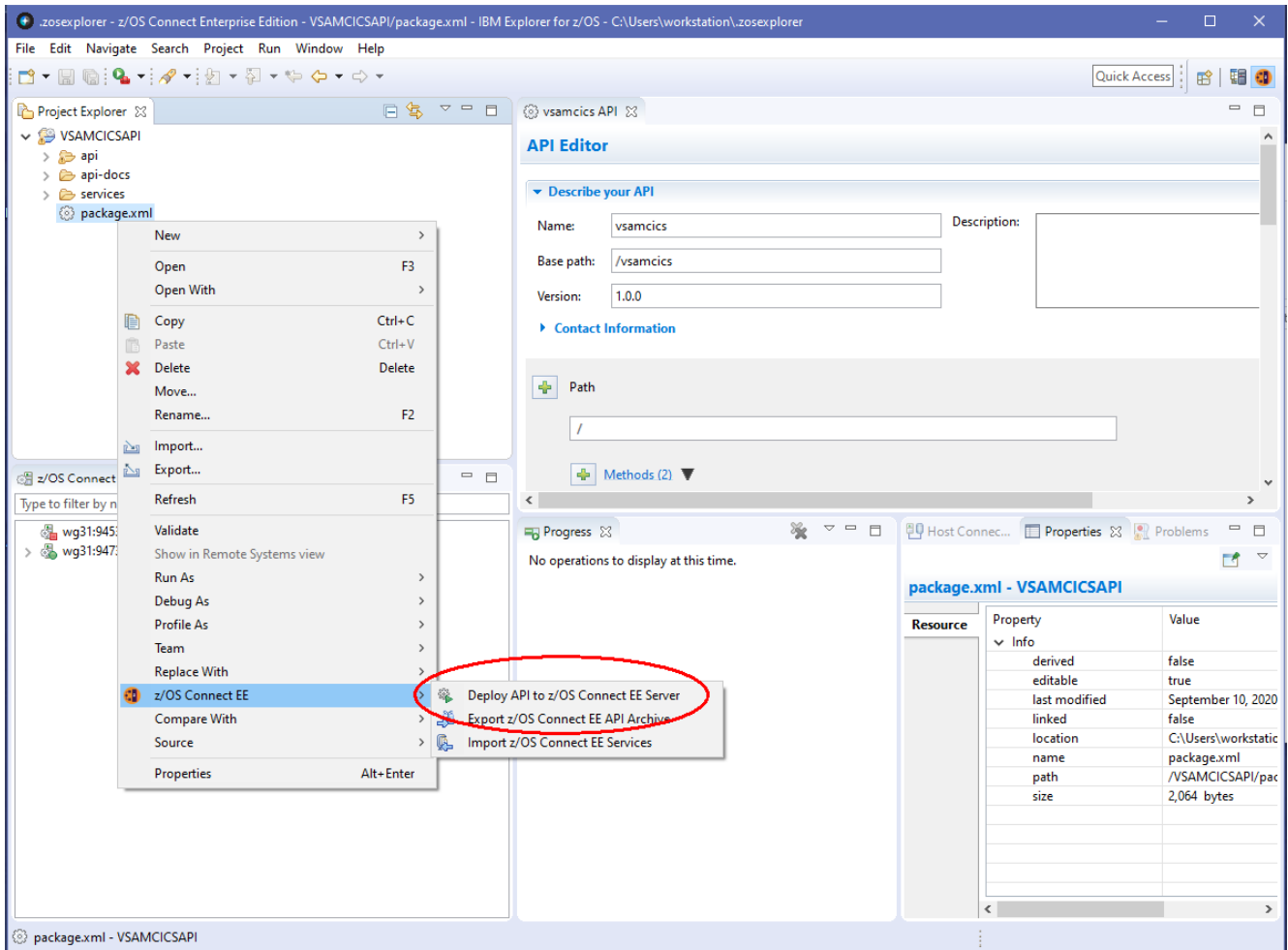


## Summary

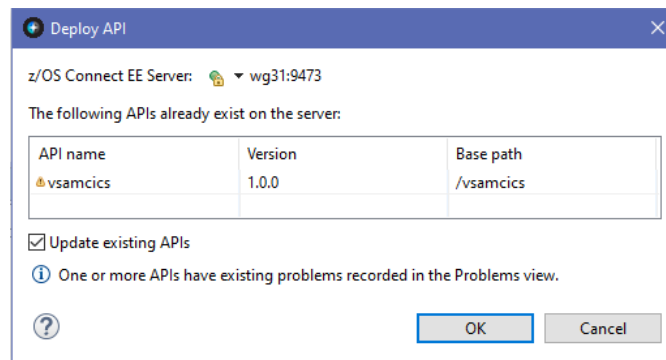
You created the API, which consists of multiple paths and the request and response mapping associated with each. That API will now be deployed into a z/OS Connect server.

## Deploy the API to a z/OS Connect Server

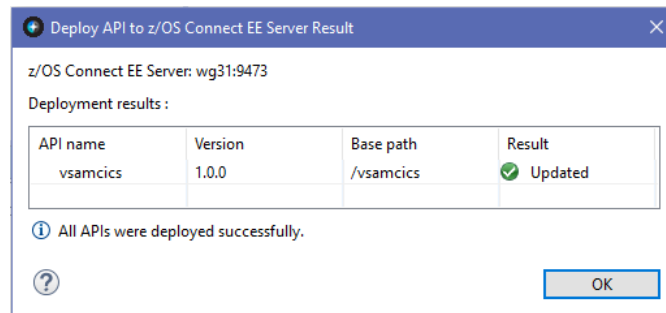
1. In the *Project Explorer* view (upper left), right-mouse click on the *IMSAPI* folder, then select *z/OS Connect* → *Deploy API to z/OS Connect Server*.



2. If the z/OS Explorer is connected to only one z/OS Connect server, there will be only one choice (*wg31:9473*). If z/OS Explorer had multiple connections to z/OS Connect servers then the pull-down arrow would allow a selection to which server to deploy, select *wg31:9473* from the list. Click **OK** on this window to continue.

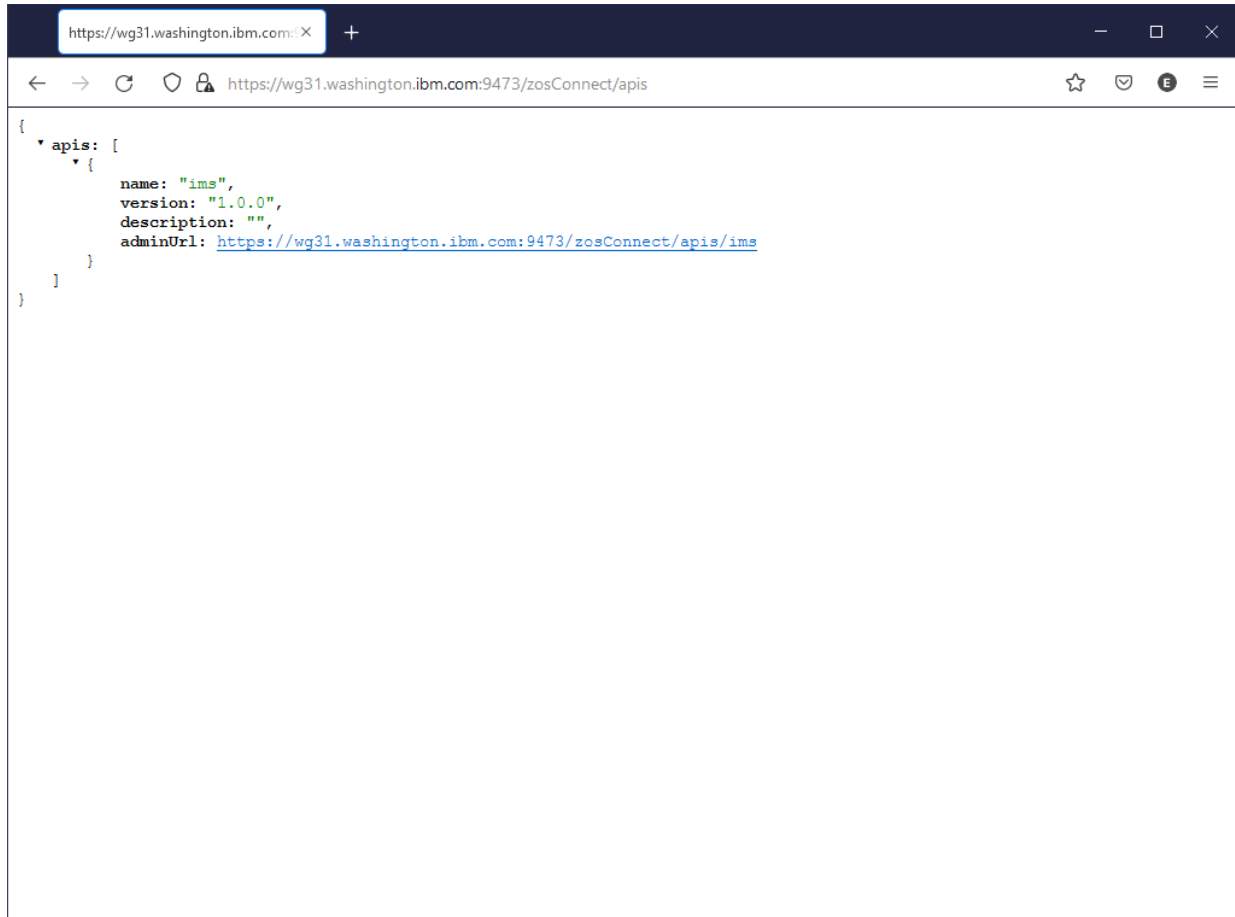


3. The API artifacts will be transferred to z/OS in an API archive (AAR) file and copied into the */var/ats/zosconnect/servers/zceedvm/resources/zosconnect/apis* directory.



## Test the IMS APIs using Swagger UI

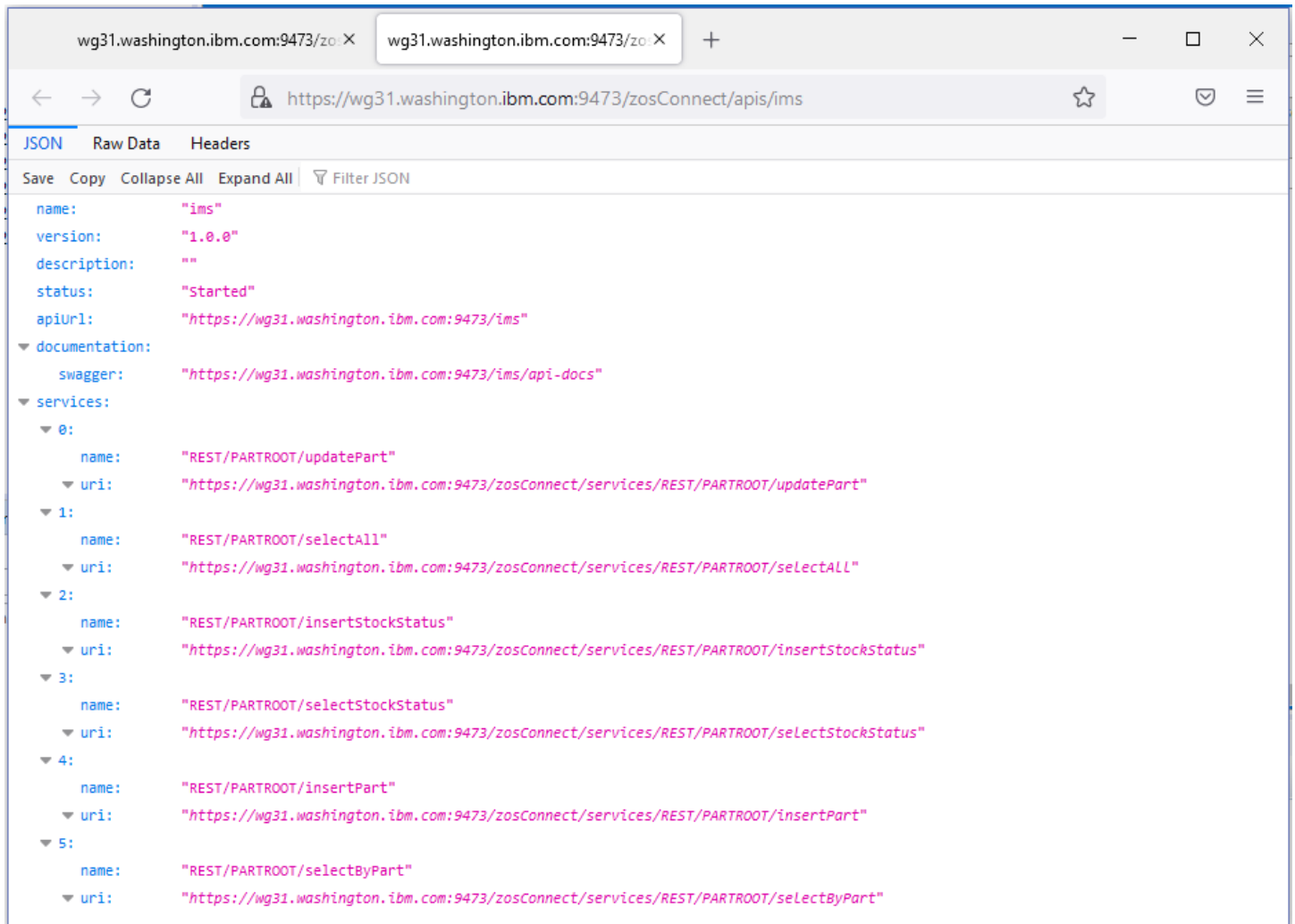
1. Next enter URL <https://wg31.washington.ibm.com:9473/zosConnect/apis> in the Firefox browser and you should see the window below.



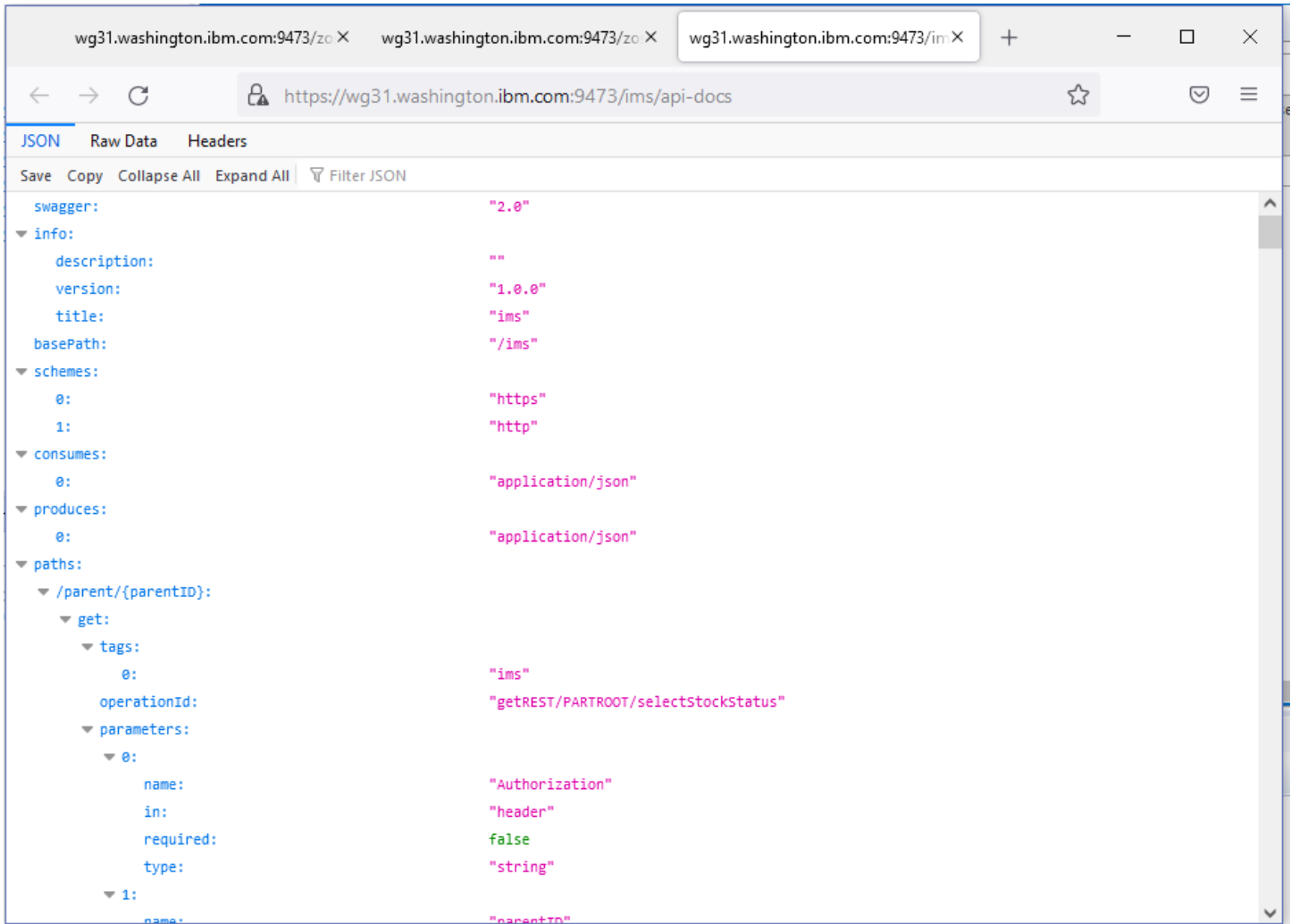
**Tech Tip:** You may be challenged by Firefox because the digital certificate used by the Liberty z/OS server is self-signed. Click the **Advanced** button to continue. Scroll down and then click on the **Accept the Risk and Continue** button. Next you may see a prompt for a userid and password. If you do see the prompt, enter the username **USER1** and password **user1** and click **OK**.

**Tech Tip:** It is very important to access the z/OS Connect server from a browser prior to any testing using the Swagger UI. Accessing a z/OS Connect URL from a browser starts an SSL handshake between the browser and the server. If this handshake has not performed prior to performing any test the test will fail with no message in the browser and no explanation. Ensuring this handshake has been performed is why you may be directed to access a z/OS Connect URL prior to using the Swagger UI.

2. If you click on *adminUrl* URL the window below should be displayed:

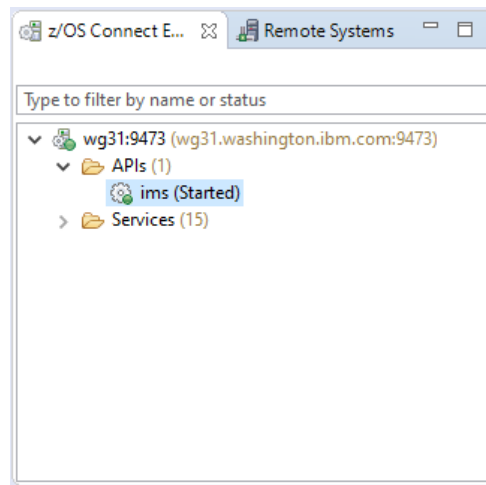


3. Finally click on the *swagger* URL for and you should see the Swagger document associated with this API.

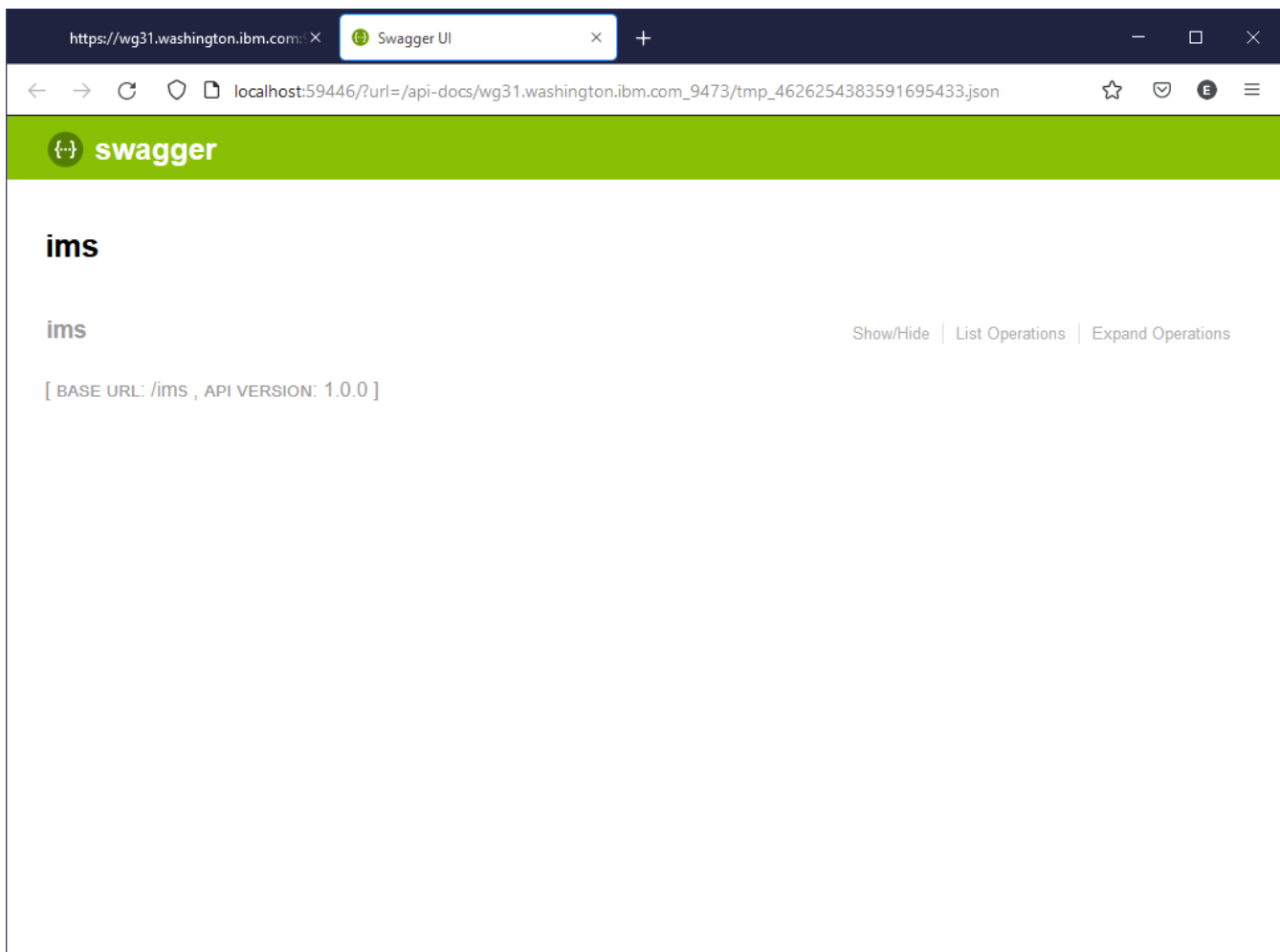


Explore this Swagger document and you will see the results of the request and response mapping performed earlier. This Swagger document can be used by a developer or other tooling to develop REST clients for this specific API.

4. In the lower left-hand side of the *z/OS Connect Explorer* perspective there is view entitled *z/OS Connect Servers*. Expand *wg31:9473* and then expand the *APIs* folder. You should see a list of the APIs installed in the server.



5. Right mouse button click on *ims* and select *Open in Swagger UI*. Click **OK** if an informational prompt appears. This will open a new view showing a *Swagger* test client (see below).



6. Click on *List Operations* option in this view and this will display a list of available HTTP methods in this API.

The screenshot shows a web browser window with the Swagger UI. The browser's address bar displays the URL: `localhost:49957/?url=/api-docs/wg31.washington.ibm.com_9473/tmp_635742273778`. The Swagger UI header is green with the 'swagger' logo. The main content area is titled 'ims' and features a 'List Operations' tab selected. Below the title, a list of API endpoints is displayed, each with its HTTP method and path:

Method	Path
GET	/parent/{parentID}
POST	/parent/{parentID}
POST	/part
GET	/part/{partNumber}
PUT	/part/{partNumber}
GET	/parts

At the bottom of the interface, the text '[ BASE URL: /ims , API VERSION: 1.0.0 ]' is visible.



7. Select the *GET* method for selecting all records from the IMS data base by clicking on the */parts* URI string. Remember this was the *Path* specified for the *GET* method for the *selectAllItems* service when the API was defined. This action will expand this method in this view and provides a Swagger UI test client (you may have to use the slider bar and adjust the perspective to see the entire client).

GET /parts

Response Class (Status 200)  
OK

Model | Example Value

```
{
  "Records": [
    {
      "U": "string",
      "RECORD_ID": "string",
      "ROOT_DESCR": "string",
      "CHILD_ID": "string",
      "ROOT_KEY": "string"
    }
  ]
}
```

Response Content Type

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	<input type="text"/>		header	string

8. Enter **Basic VVNFUjE6VVNFUjE=** in the box beside *Authorization* and press the **Try it out!** button. You may see a Security Alert pop-up warning about the self-signed certificate being used by the z/OS Connect server. Click **Yes** on this pop-up.

**Tech Tip:** The string *VVNFUjE6VVNFUjE=* is the string *USER1:USER1* encoded in base 64. See URL <https://www.base64encode.org/> for information on how this string was generated.

9. Scroll down the view and you should see the *Response Body* which contains the results of the GET method (see below). Note that the columns removed from the interface in an earlier step are not present.

Response Body

```
{
  "Records": [
    {
      "record_id": "02AN960C10",
      "child_id": "F0F2C1D5F9F6F0C3F1F0404040404040",
      "u": "02",
      "description": "WASHER",
      "partNumber": "AN960C10"
    },
    {
      "record_id": "02CK05CW181K",
      "child_id": "F0F2C3D2F0F5C3E6F1F8F1D24040404040",
      "u": "02",
      "description": "CAPACITOR",
      "partNumber": "CK05CW181K"
    },
    {
      "record_id": "02CSR13G104KL",
      "child_id": "F0F2C3E2D9F1F3C7F1F0F4D2D340404040",
      "u": "02",
      "description": "RESISTOR",
      "partNumber": "CSR13G104KL"
    }
  ]
}
```

10. Select the *GET* method for selecting a single record from the IMS data base by clicking on the `/part/{partNumber}` URI string. Remember this was the *Path* specified for the *GET* method for the *selectByPart* service when the API was defined. This action will expand this method in this view and provides a Swagger UI test client (you may have to use the slider bar and adjust the perspective to see the entire client).

11. Enter **Basic VVNFUjE6VVNFUjE=** in the box beside *Authorization* and **AN960C10** in the area beside *partNumber* and press the **Try it out!** button.

Response Content Type application/json ▾

### Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	<input "="" type="text" value="Basic VVNFUjE6VVNFUjE="/>		header	string
partNumber	<input type="text" value="AN960C10"/>		path	string

### Response Messages

HTTP Status Code	Reason	Response Model	Headers
404	Not Found	Model <b>Example Value</b>	

```
{
  "Result": 0,
  "Records": [
    {
      "U": "string",
      "RECORD_ID": "string",
      "ROOT_DESCR": "string",
      "CHILD_ID": "string",
      "ROOT_KEY": "string"
    }
  ],
}
```

Try it out! [Hide Response](#)

12. Scroll down the view and you should see the *Response Body* which contains the results of the GET method (see below). Note that the columns removed from the interface in an earlier step are not present.

**Curl**

```
curl -X GET --header 'Accept: application/json' --header 'Authorization: Basic VVNFUjE6VVNFUjE=' 'https://wg31.washington.ibm.com:94'
```

**Request URL**

```
https://wg31.washington.ibm.com:9473/ims/part/AN960C10
```

**Request Headers**

```
{
  "Accept": "application/json",
  "Authorization": "Basic VVNFUjE6VVNFUjE="
}
```

**Response Body**

```
{
  "Affected": 0,
  "TotRecs": 1,
  "Skipped": 0,
  "NumRecs": 1,
  "NumFields": 5,
  "Records": [
    {
      "record_id": "02AN960C10",
      "child_id": "F0F2C1D5F9F6F0C3F1F0404040404040",
      "u": "02",
      "description": "WASHER",
      "partNumber": "AN960C10"
    }
  ],
  "Result": 0
}
```

**Response Code**

```
200
```

13. Select the *POST* method for inserting a root segment in the DLI data base clicking on the *POST /part* URI string. Remember this was the *Path* specified for the *POST* method for the *insertPart* service when the API was defined. Enter **Basic VVNFUjE6VVNFUjE=** in the box beside *Authorization*, a value of **948478** in the area under *PARTNUMBER* a random value in the area beside *Description* and a value of **09** in the area under *KEYPREFIX*. Then press the **Try it out!** button.

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Basic VVNFUjE6VVNFUjE=		header	string
postREST/PARTROOT /insertPart_request	-	request body	body	Model Example Value
<div> <div>DESCRIPTION</div> <div>new part</div> <div><small>{name: DESCRIPTION, columnType: 1, dataFormat: CHARACTER, type: VARCHAR}</small></div> </div> <div> <div>PARTNUMBER</div> <div>948478</div> <div><small>{name: PARTNUMBER, columnType: 1, dataFormat: CHARACTER, type: VARCHAR}</small></div> </div> <div> <div>KEYPREFIX</div> <div>09</div> <div><small>{name: KEYPREFIX, columnType: 1, dataFormat: CHARACTER, type: VARCHAR}</small></div> </div>				<pre>{   "DESCRIPTION": "string",   "PARTNUMBER": "string",   "KEYPREFIX": "string" }</pre>
Parameter content type: application/json ▼				
Try it out!				

14. The results should show a **Response Code** of 200.

Curl
<pre>curl -X POST --header 'Content-Type: application/json' --header 'Accept: application/json' --header 'Authorization: Basic VVNFUjE6VVNFUjE='&lt;div&gt;&gt;</pre>
Request URL
<pre>https://wg31.washington.ibm.com:9473/ims/part</pre>
Request Headers
<pre>{   "Accept": "application/json",   "Authorization": "Basic VVNFUjE6VVNFUjE=" }</pre>
Response Body
<pre>{   "Records": [] }</pre>
Response Code
<pre>200</pre>

15. Select the *PUT* method for updating a root segment in the DLI data base clicking on the *Post /part* URI string. Remember this was the *Path* specified for the *PUT* method for the *upsRWPart* service when the API was defined. Enter **Basic VVNFUjE6VVNFUjE=** in the box beside *Authorization*, a value of **948478** in the area under *PARTNUMBER* a value of **Updated** in the area under *Description*. Then press the **Try it out!** button.

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Basic VVNFUjE6VVNFUjE=		header	string
partNumber	948478		path	string
putREST/PARTROOT /updatePart_request	-	request body	body	Model Example Value

DESCRIPTION

Updated

{name: DESCRIPTION, columnType: 1, dataFormat: CHARACTER, type: VARCHAR}

Parameter content type: application/json

{

"DESCRIPTION": "string"

}

Try it out!

16. The results should show a **Response Code** of 200.

Response Body

```
{
  "Affected": 0,
  "TotRecs": 3,
  "Skipped": 0,
  "NumRecs": 3,
  "NumFields": 13,
  "Records": [
    {
      "ss_proj": "165",
      "ss_unit_price": 0,
      "filler1": "00",
      "ss_on_order": 20,
      "ss_unpl_reqmts": 15,
      "ss_in_stock": 126,
      "parent_id": "F0F2C1D5F9F6F0C3F1F0404040404040",
      "ss_dept": "AA",
      "ss_cur_reqmts": 131,
      "ss_area": " ",
      "ss_unit_of_meas": "EACH",
      "ss_qty": 100
    }
  ]
}
```

Response Code

200

17. Select the *GET* method for retrieving a stock status segment in the DLI data base clicking on the *GET* */parent/{parentID}* URI string. Remember this was the *Path* specified for the *PUT* method for the *selectStockStatus* service when the API was defined. Enter **Basic VVNFUjE6VVNFUjE=** in the box beside *Authorization*, a value of **F0F2C1D5F9F6F0C3F1F0404040404040** in the area beside **parentID**. Then press the **Try it out!** button.

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization	Basic VVNFUjE6VVNFUjE=		header	string
parentID	F0F2C1D5F9F6F0C3F1F0404040404040		path	string

Try it out! [Hide Response](#)



18. The results should show a **Response Code** of 200 and a list of stock status segments for this root segment.

**Response Body**

```
{
  "Affected": 0,
  "TotRecs": 3,
  "Skipped": 0,
  "NumRecs": 3,
  "NumFields": 13,
  "Records": [
    {
      "ss_proj": "165",
      "ss_unit_price": 0,
      "filler1": "00",
      "ss_on_order": 20,
      "ss_unpl_reqmts": 15,
      "ss_in_stock": 126,
      "parent_id": "F0F2C1D5F9F6F0C3F1F0404040404040",
      "ss_dept": "AA",
      "ss_cur_reqmts": 131,
      "ss_area": " ",
      "ss_unit_of_meas": "EACH",
      "ss_in_stock": 126
    }
  ]
}
```

**Response Code**

200

19. Select the *POST* method for inserting a stock status segment in the DLI data base clicking on the *POST* */parent/{parentID}* URI string. Remember this was the *Path* specified for the *PUT* method for the *insertStockStatus* service when the API was defined. Enter **Basic VVNFUjE6VVNFUjE=** in the box beside *Authorization*, a value of **F0F9F9F4F8F4F7F84040404040404040** in the area beside **parentID**. Enter **1** under *AREA*, **22** under *PROJECT* and **333** under *DEPT*. Then press the **Try it out!** button.

Parameters				
Parameter	Value	Description	Parameter Type	Data Type
Authorization	Basic VVNFUjE6VVNFUjE=		header	string
parentID	F0F9F9F4F8F4F7F84040404040404040		path	string
postREST/PARTROOT /insertStockStatus_request	-	request body	body	Model Example Value
<div> <div>AREA</div> <div>1</div> <div>{name: AREA, columnType: 1, dataFormat: CHARACTER, type: VARCHAR}</div> </div> <div> <div>PROJECT</div> <div>22</div> <div>{name: PROJECT, columnType: 1, dataFormat: CHARACTER, type: VARCHAR}</div> </div> <div> <div>DEPT</div> <div>333</div> <div>{name: DEPT, columnType: 1, dataFormat: CHARACTER, type: VARCHAR}</div> </div>			<pre>{   "AREA": "string",   "PROJECT": "string",   "DEPT": "string" }</pre>	
Parameter content type: application/json				
<div>Try it out!</div> <div>Hide Response</div>				

The results should show a **Response Code** of 200.

**Response Body**

```
{
  "Affected": 1,
  "TotRecs": 0,
  "Skipped": 0,
  "NumRecs": 0,
  "NumFields": 0,
  "Records": [],
  "Result": 0
}
```

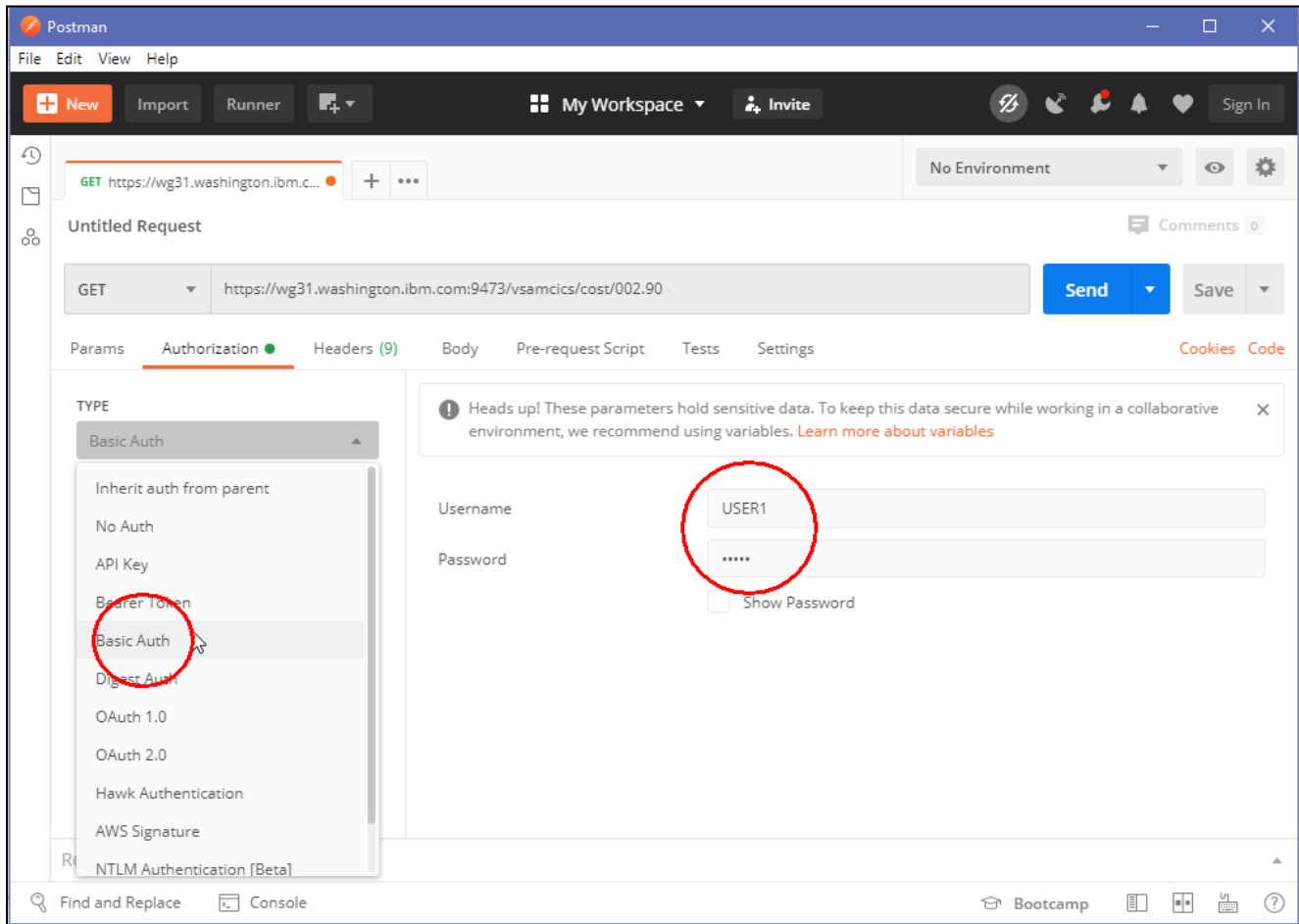
**Response Code**

200

## Test the IMS APIs using Postman

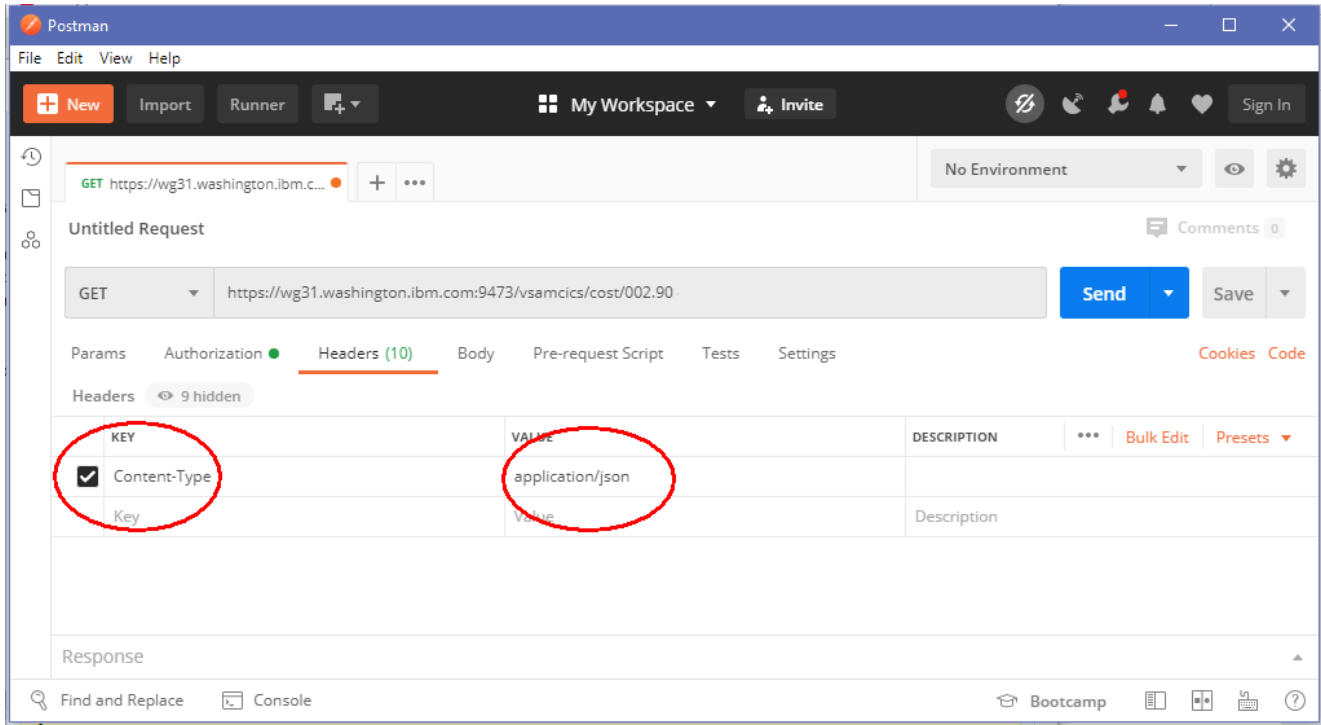
The other API services will be tested using Postman.

1. Open the *Postman* tool icon on the desktop. If necessary reply to any prompts and close any welcome messages.
2. Next, select the *Authorization* tab to enter an authorization identity and password. Use the pull down arrow to select *Basic Auth* and enter *USER1* as the *Username* and *USER1* as the *Password*.



**Tech-Tip:** If the above Postman view is not displayed select *File* on the toolbar and then choose *New Tab* on the pull down. Alternatively, if the *Launchpad* view is displayed, click on the *Create a request* option.

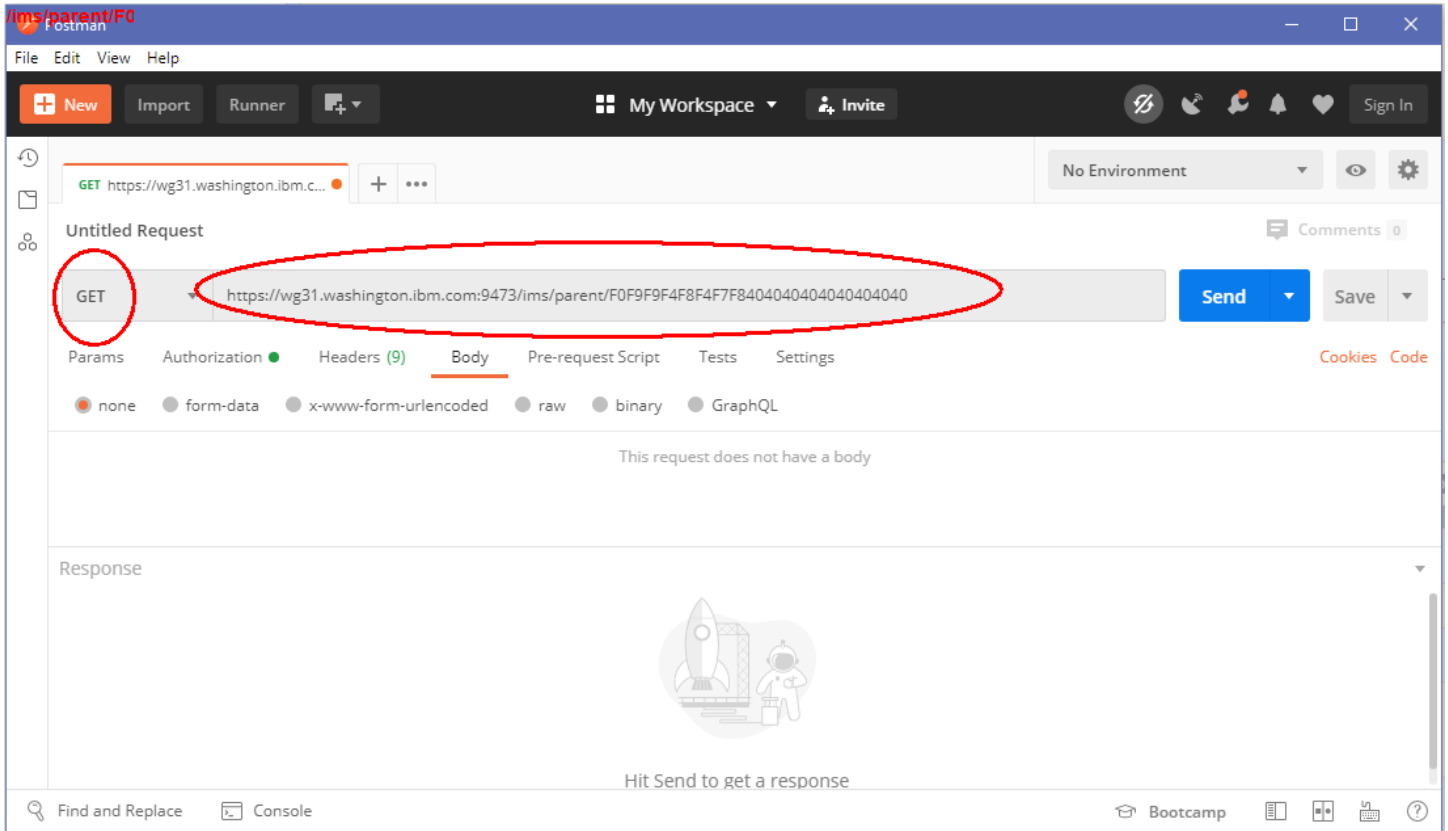
3. Next, select the *Headers* tab. Under *KEY* use the code assist feature to enter ***Content-Type***, and under *VALUE*, use the code assist feature to enter ***application/json***.



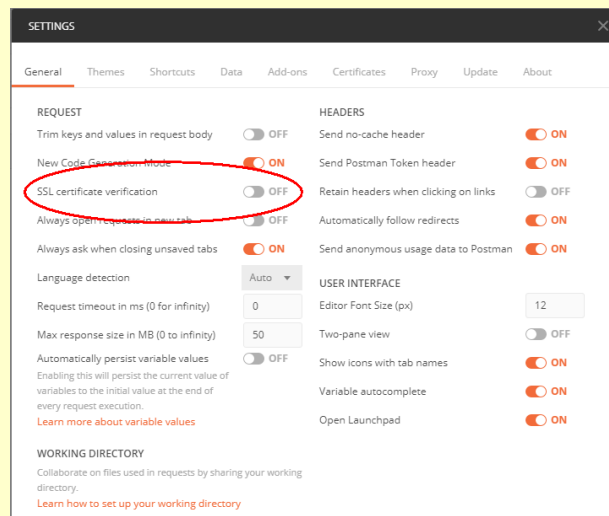
**Tech-Tip:** Code assist simply means that when text is entered in field, all the valid values for that field that match the typed text will be displayed. You can select the desired value for the field from the list displayed and that value will populate that field.

4. To test the *selectByCost* service use the down arrow to select **GET** and enter

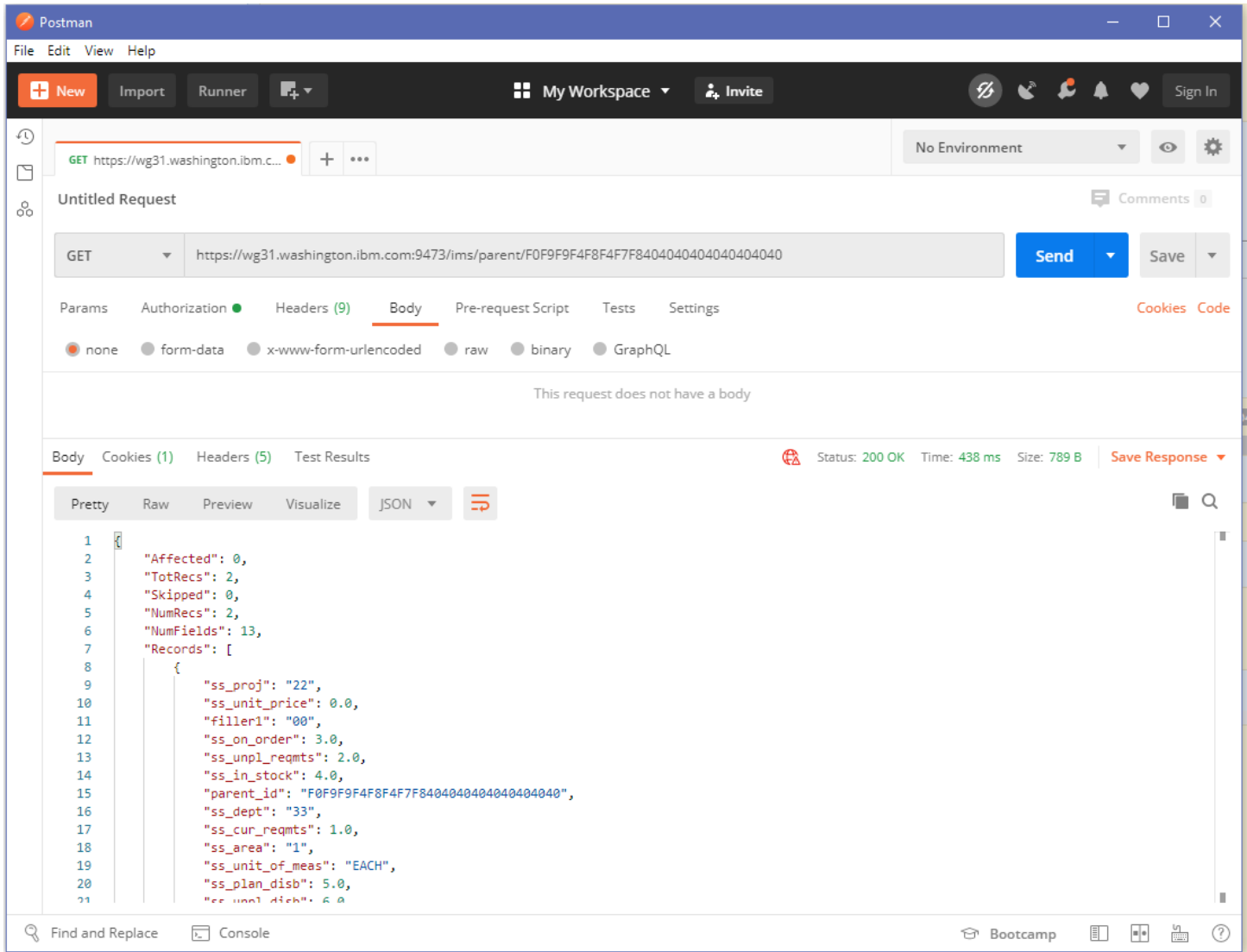
**<https://wg31.washington.ibm.com:9473/ims/parent/F0F9F9F4F8F4F7F84040404040404040>** in the URL area (see below) to select all stock status segments for this root segment.



**Tech-Tip:** The Postman settings have been changed to disable *SSL certificate verification*, a settings option.



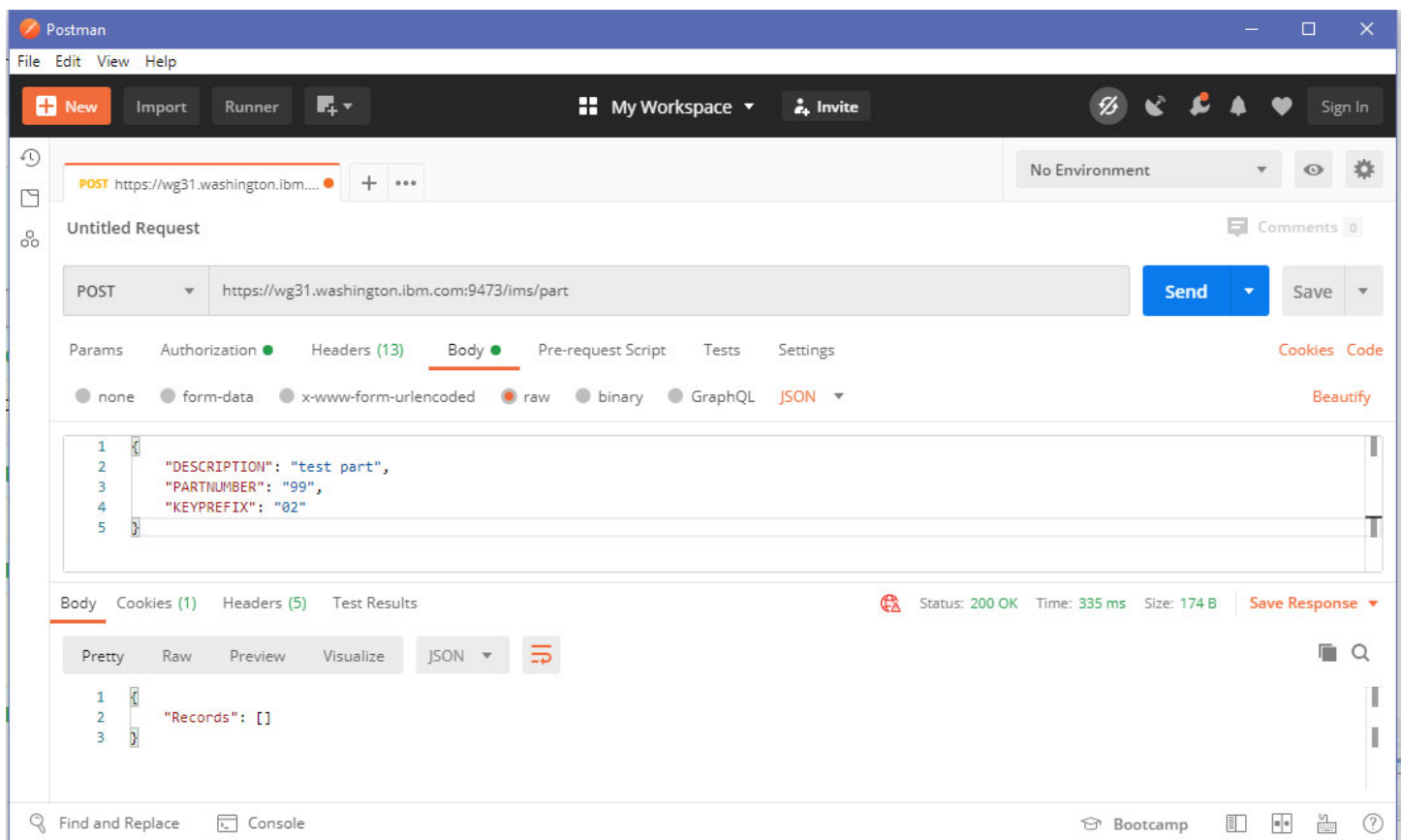
5. Next select the *Body* tab and select the *raw* radio button. Then press the **Send** button. A response message should come back indicating the service has been started and other details about the service. You may have to ‘drag’ the response body area up to display the response message.



6. To test the *insertPart* service use the down arrow to select **POST** and enter <https://wg31.washington.ibm.com:9473/ims/part> in the URL area (see below). Enter the JSON request message below.

```
{
  "DESCRIPTION": "test part",
  "PARTNUMBER": "99",
  "KEYPREFIX": "02"
}
```

Press the Send button to insert a new root segment this informaton



**Tech-Tip:** Use a GET to URL <https://wg31.washington.ibm.com:9473/ims/part/99> to display the item and confirm the insert has taken place.



## *Summary*

You use DVM to develop 6 services. The SAR files for the services were imported in the API Editor of z/OS Connect. The API Editor was used to develop a RESTful API. You have verified the API. The API layer provided a further level of abstraction and allows a more flexible use of HTTP verbs, and better mapping of data via the API editor function.