DEP API Lab

1. **Lab Overview**

Defining schemas and understanding how data is described by those schemas is one of the first things a customer will do on Adobe Experience Platform. This is the first step to ingesting data to the platform.

This lab will introduce you to the Adobe Experience Platform schema design user experience. You will learn how to create your own schemas, as well as browse existing components.

2. **Learning Objectives**

What should you walk away with after taking this Lab?

- Create various AEP/XDM components using API Calls. Examples are:
  1. Class
  2. Field Group
  3. Schema
  4. Descriptors
     a. Identities (including Primary)
     b. Relationship
     c. Reference

- Adding some Schema configurations through Schema UI interface.
- Browse AEP/XDM Components through API Calls
- Extending the Event Types via API Calls
- Validating the created Schemas through UI

3. **Pre Requisites**

Learners should have gone through the following concepts/courses/modules

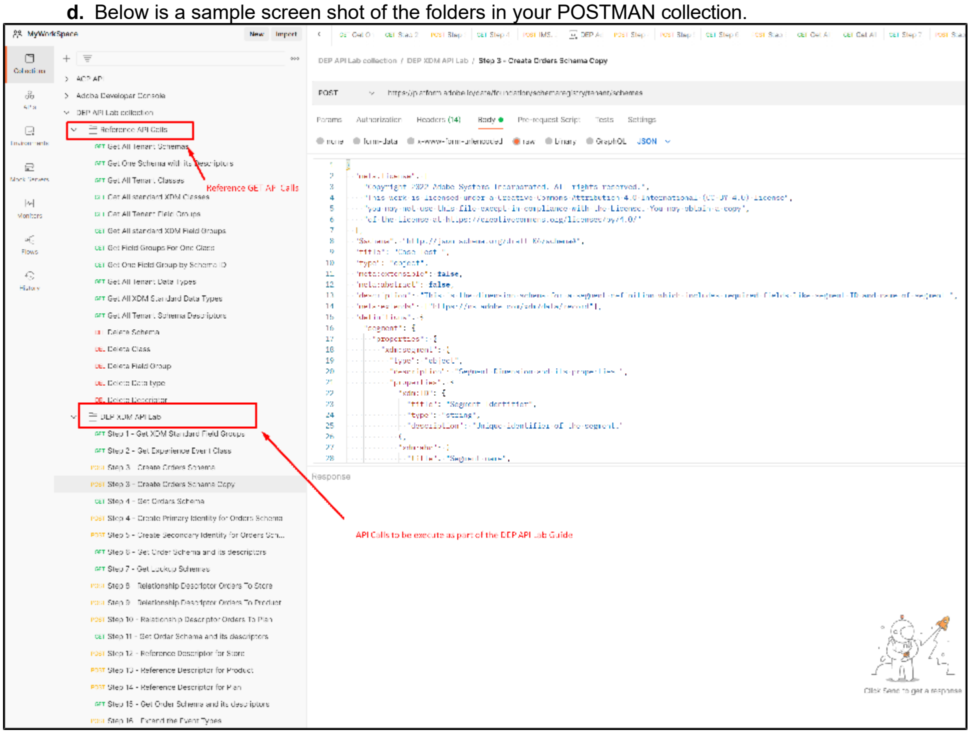
- Basics of XDM and Schema Composition
- Profile concepts and the important XDM components for Profile
- SID Methodology – Should have applied the SID methodology to the **Customer Source Data Model** and should have the expected **Customer Data - AEP XDM ERD**
- Should be aware of the Source to XDM Mapping Document (**Orders Schema - Customer to XDM Mapping.xlsx**) . This will be the source of all the Schemas created in this Lab.
- As part of your POST setup class, you would have POSTMAN setup and you should know how to set up your **Environment Configuration**, You should also know how to generate your access token and import a collection.

4. Lab Guide

This guide will walk you through step by step process to create Orders Schema and its corresponding relationships to other Lookup/Dimensional Schemas via API Calls.

4.1. Import the following collection into your POSTMAN. This collection has the following folders: DEP API Lab Postman Collection

1. **DEP XDM API Lab -**
    a. Contains all the API calls to create Order related XDM components like Schema, Field Groups, Primary Identities, Secondary Identities, Relationships and Reference Identities.
2. **Reference API Calls**
    a. Contains some reference "GET" API calls to help you navigate the Schema registry
    b. You can use these API calls to browse the XDM components you created and also to get identifiers of various XDM components which you would need as you go through the API Lab. It would be good to have at least have a glance at these.
    c. Please note that you will have to create/POST the XDM components first to expect results in the GET API calls.
    d. Below is a sample screen shot of the folders in your POSTMAN collection.
3.



- ACP API
- Adobe Developer Console
- DEP API Lab collection
    - Reference API Calls
        - GET Get All Tenant Schemas
        - GET Get One Schema with its Descriptors
        - GET Get All Tenant Classes
        - GET Get All standard XDM Classes
        - GET Get All Tenant Field Groups
        - GET Get All standard XDM Field Groups
        - GET Get Field Groups For One Class
        - GET Get One Field Group by Schema ID
        - GET Get All Tenant Data Types
        - GET Get All XDM Standard Data Types
        - GET Get All Tenant Schema Descriptors
        - DEL Delete Schema
        - DEL Delete Class
        - DEL Delete Field Group
        - DEL Delete Data type
        - DEL Delete Descriptor
    - DEP XDM API Lab
        - GET Step 1 - Get XDM Standard Field Groups
        - GET Step 2 - Get Experience Event Class
        - POST Step 3 - Create Orders Schema
        - POST Step 3 - Create Orders Schema Copy
        - GET Step 4 - Get Orders Schema
        - POST Step 4 - Create Primary Identity for Orders Schema
        - POST Step 5 - Create Secondary Identity for Orders Sch...
        - GET Step 6 - Get Order Schema and its descriptors
        - GET Step 7 - Get Lookup Schemas
        - POST Step 8 - Relationship Descriptor Orders To Store
        - POST Step 9 - Relationship Descriptor Orders To Product
        - POST Step 10 - Relationship Descriptor Orders To Plan
        - GET Step 11 - Get Order Schema and its descriptors
        - POST Step 12 - Reference Descriptor for Store
        - POST Step 13 - Reference Descriptor for Product
        - POST Step 14 - Reference Descriptor for Plan
        - GET Step 15 - Get Order Schema and its descriptors
        - POST Step 16 - Extend the Event Types

Reference GET API Calls

DEP API Lab collection / DEP XDM API Lab / **Step 3 - Create Orders Schema Copy**

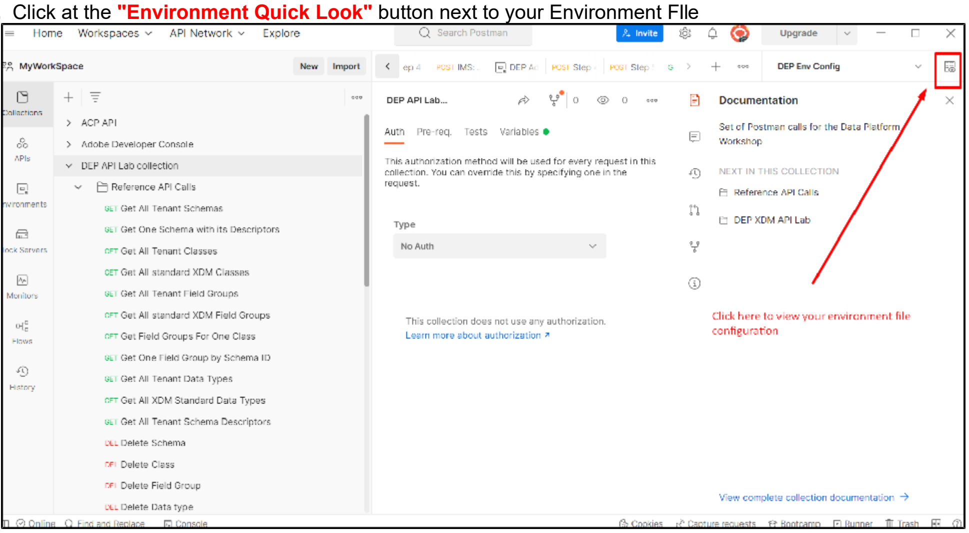
POST https://platform.adobe.io/data/foundation/schemaregistry/tenant/schemas

```
{
  "meta:license": [
    "Copyright 2022 Adobe Systems Incorporated. All rights reserved.",
    "This work is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) license",
    "you may not use this file except in compliance with the License. You may obtain a copy",
    "of the License at https://creativecommons.org/licenses/by/4.0/"
  ],
  "$schema": "http://json-schema.org/draft-06/schema#",
  "title": "Case Test ",
  "type": "object",
  "meta:extensible": false,
  "meta:abstract": false,
  "description": "This is the dimension schema for a segment definition which includes required fields like segment ID and name of segment.",
  "meta:extends": ["https://ns.adobe.com/xdm/data/record"],
  "definitions": {
    "segment": {
      "properties": {
        "xdm:segment": {
          "type": "object",
          "description": "Segment Dimension and its properties.",
          "properties": {
            "xdm:ID": {
              "title": "Segment Identifier",
              "type": "string",
              "description": "Unique identifier of the segment."
            },
            "xdm:name": {
              "title": "Segment name",
```

API Calls to be execute as part of the DEP API Lab Guide

4.2. Add the "SANDBOX_NAME" variable into your environment file.

1. Your Sandbox name should be provided to you when you start the API Lab.
2. Click at the **"Environment Quick Look"** button next to your Environment FIle
3.



- ACP API
- Adobe Developer Console
- DEP API Lab collection
    - Reference API Calls
        - GET Get All Tenant Schemas
        - GET Get One Schema with its Descriptors
        - GET Get All Tenant Classes
        - GET Get All standard XDM Classes
        - GET Get All Tenant Field Groups
        - GET Get All standard XDM Field Groups
        - GET Get Field Groups For One Class
        - GET Get One Field Group by Schema ID
        - GET Get All Tenant Data Types
        - GET Get All XDM Standard Data Types
        - GET Get All Tenant Schema Descriptors
        - DEL Delete Schema
        - DEL Delete Class
        - DEL Delete Field Group
        - DEL Delete Data type

This authorization method will be used for every request in this collection. You can override this by specifying one in the request.

Type: No Auth

This collection does not use any authorization.
Learn more about authorization ↗

Documentation

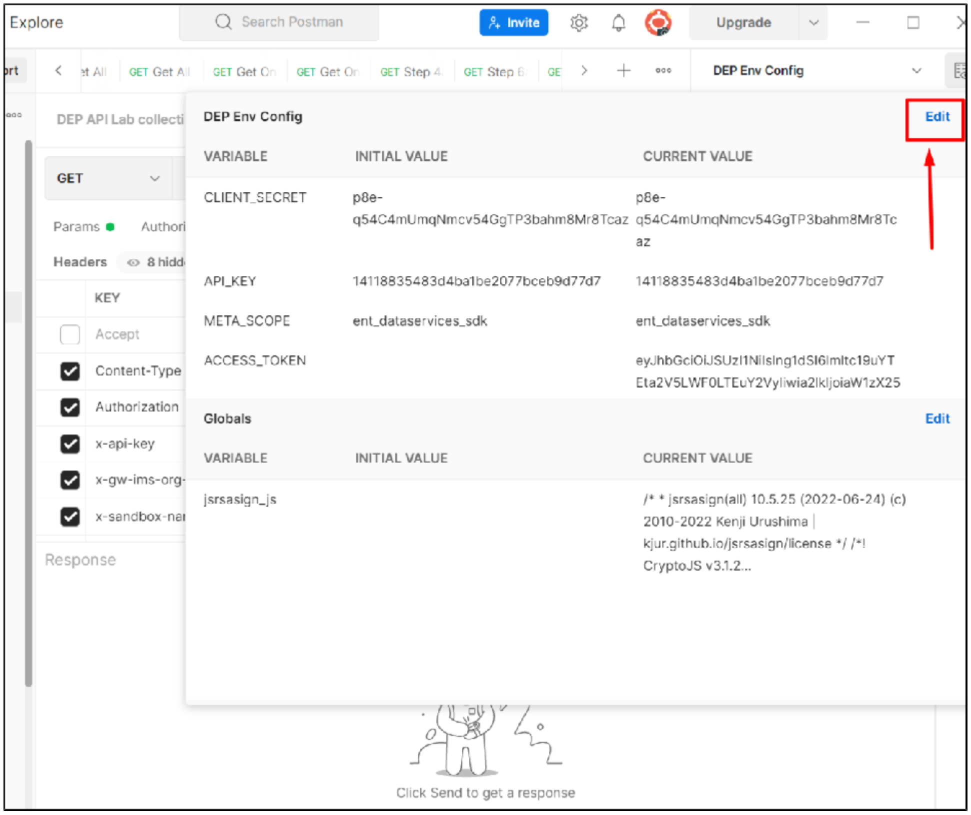
Set of Postman calls for the Data Platform Workshop

NEXT IN THIS COLLECTION
- Reference API Calls
- DEP XDM API Lab

View complete collection documentation →

Click here to view your environment file configuration

4. Click **"Edit"** button to add the variable for Sandbox as shown below:



DEP Env Config — Edit

VARIABLE	INITIAL VALUE	CURRENT VALUE
CLIENT_SECRET	p8e-q54C4mUmqNmcv54GgTP3bahm8Mr8Tcaz	p8e-q54C4mUmqNmcv54GgTP3bahm8Mr8Tcaz
API_KEY	14118835483d4ba1be2077bceb9d77d7	14118835483d4ba1be2077bceb9d77d7
META_SCOPE	ent_dataservices_sdk	ent_dataservices_sdk
ACCESS_TOKEN		eyJhbGciOiJSUzI1NiIsIng1dSI6Imltc19uYTEta2V5LWF0LTEuY2VyIiwia2lkIjoiaW1zX25

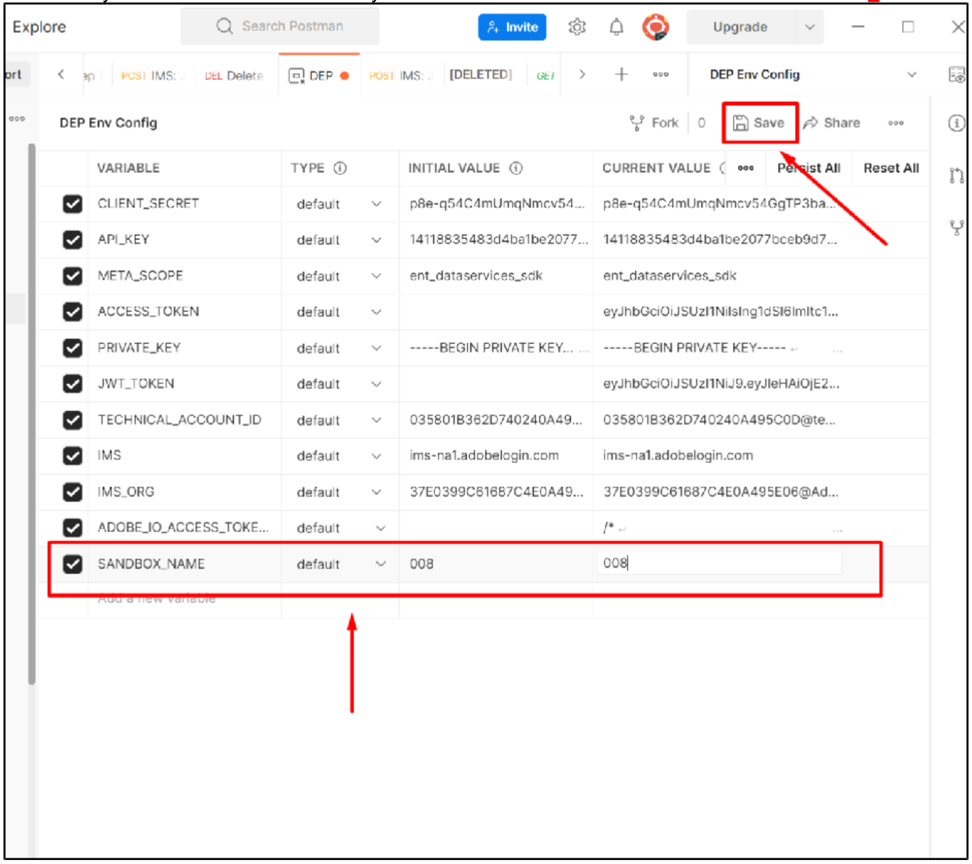
Globals — Edit

VARIABLE	INITIAL VALUE	CURRENT VALUE
jsrsasign_js		/* * jsrsasign(all) 10.5.25 (2022-06-24) (c) 2010-2022 Kenji Urushima | kjur.github.io/jsrsasign/license */ /*! CryptoJS v3.1.2...

5.

6. Add the variable **"SANDBOX_NAME"** in the environment file as shown below:

7. Make sure you hit the "Save" button after you add the variable and the values in both the **"INITIAL_VALUE"** and **"CURRENT_VALUE"** Fields



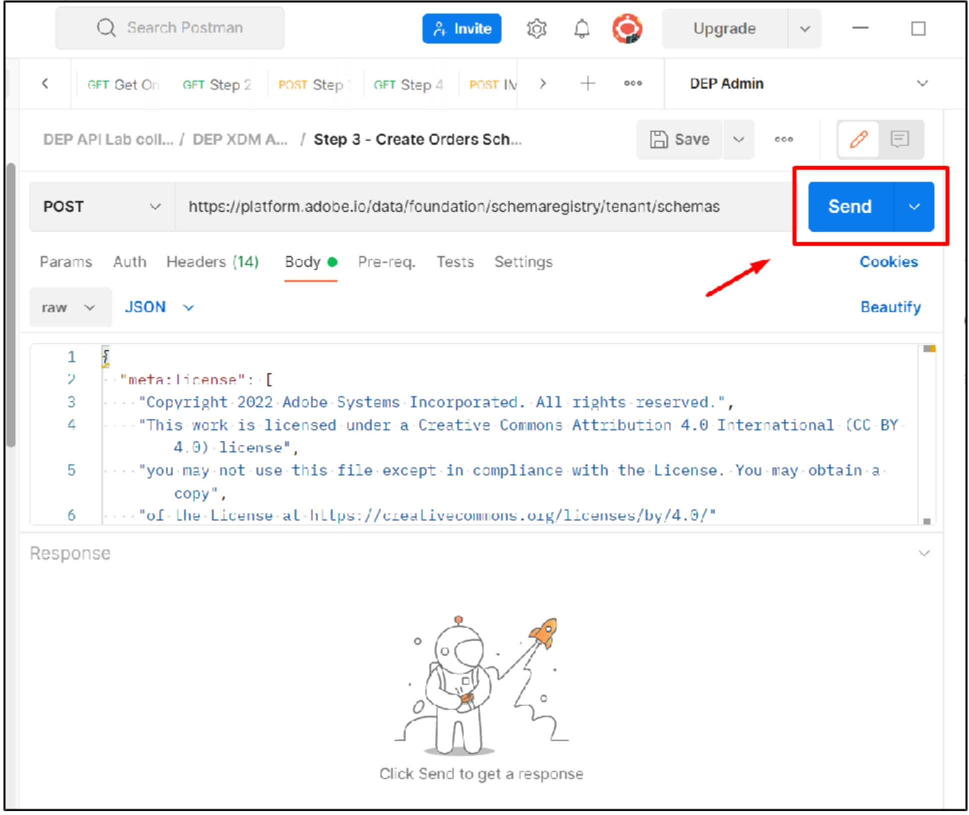
DEP Env Config

	VARIABLE	TYPE	INITIAL VALUE	CURRENT VALUE
☑	CLIENT_SECRET	default	p8e-q54C4mUmqNmcv54...	p8e-q54C4mUmqNmcv54GgTP3ba...
☑	API_KEY	default	14118835483d4ba1be2077...	14118835483d4ba1be2077bceb9d7...
☑	META_SCOPE	default	ent_dataservices_sdk	ent_dataservices_sdk
☑	ACCESS_TOKEN	default		eyJhbGciOiJSUzI1NiIsIng1dSI6Imltc1...
☑	PRIVATE_KEY	default	-----BEGIN PRIVATE KEY... ...	-----BEGIN PRIVATE KEY----- ↵ ...
☑	JWT_TOKEN	default		eyJhbGciOiJSUzI1NiJ9.eyJleHAiOjE2...
☑	TECHNICAL_ACCOUNT_ID	default	035801B362D740240A49...	035801B362D740240A495C0D@te...
☑	IMS	default	ims-na1.adobelogin.com	ims-na1.adobelogin.com
☑	IMS_ORG	default	37E0399C61687C4E0A49...	37E0399C61687C4E0A495E06@Ad...
☑	ADOBE_IO_ACCESS_TOKE...	default		/* ↵ ...
☑	SANDBOX_NAME	default	008	008

8.

4.3. **Get the Schema ID for the XDM standard "Order Details" Field Group**

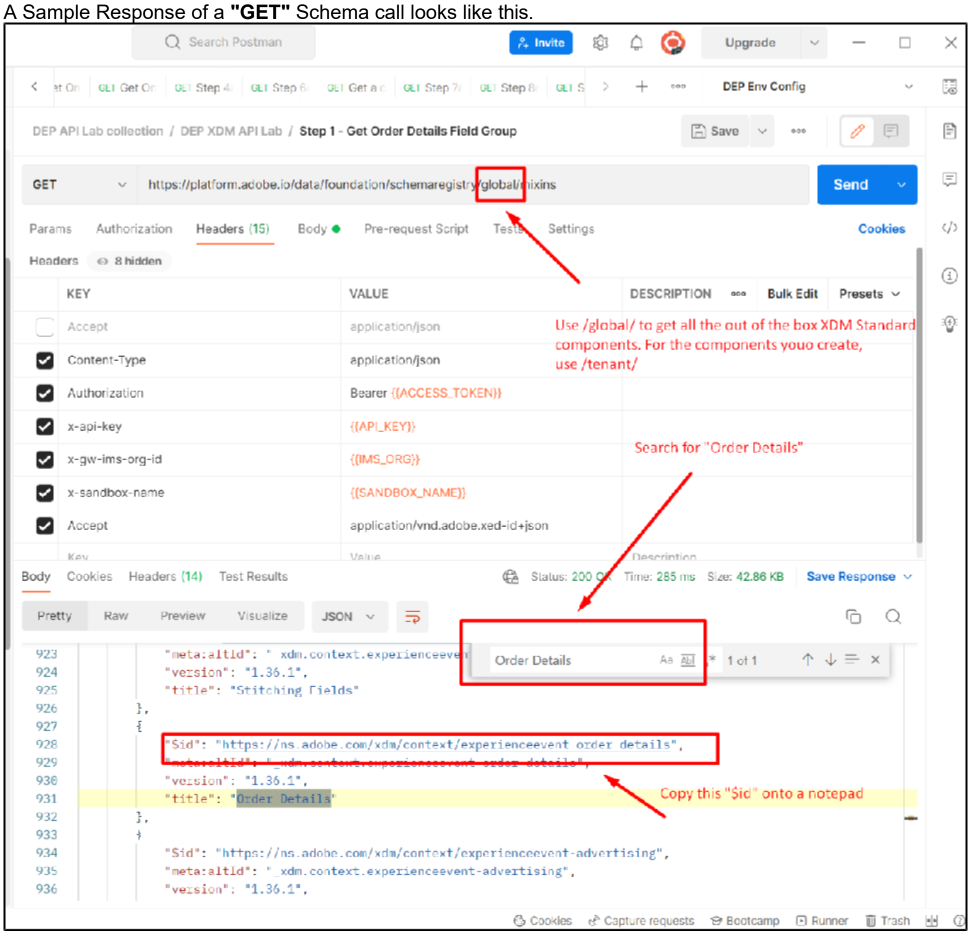
"Field Group" was referred as a "mixin" previously so these terms might be used interchangeably

1. **Click on "Step 1 - Get Order Details Field Group" API call in the "DEP XDM API Lab" Folder. Do not execute it yet.**
2. **Please note the following points:**
    a. Note the use of **"global" in https://platform.adobe.io/data/foundation/schemaregistry/global/mixins. "global"** is used to get the XDM provided out of the box standard XDM components (Field group/mixin in this case).
        i. Remember we have two types of owners in AEP (**Adobe/Standard and Tenant/custom**). The path to these objects is either through:
            1. Adobe/Standard objects will have global in the path e.g. https://platform.adobe.io/data/foundation/schemaregistry/**global**/mixins
            2. Tenant/Custom objects will have [tenant] in the path e.g. https://platform.adobe.io/data/foundation/schemaregistry/**tenant**/mixins
    b. This API call gives you all the standard XDM field Groups
3. **Execute "Step 1 - Get Order Details Field Group" by clicking the "Send" button**
    a. After you execute the GET API call, search for **"Order Details"** in the Response
    b. Copy the **"$id"** of the **"Order Details"** Field Group and have it handy with you
4. A Sample **"GET"** Schema call looks like below.
5. 

POST https://platform.adobe.io/data/foundation/schemaregistry/tenant/schemas

```
{
  "meta:license": [
    "Copyright 2022 Adobe Systems Incorporated. All rights reserved.",
    "This work is licensed under a Creative Commons Attribution 4.0 International (CC BY 4.0) license",
    "you may not use this file except in compliance with the License. You may obtain a copy",
    "of the License at https://creativecommons.org/licenses/by/4.0/"
```

6. A Sample Response of a **"GET"** Schema call looks like this.



GET https://platform.adobe.io/data/foundation/schemaregistry/global/mixins

	KEY	VALUE	DESCRIPTION
□	Accept	application/json	
☑	Content-Type	application/json	
☑	Authorization	Bearer {{ACCESS_TOKEN}}	
☑	x-api-key	{{API_KEY}}	
☑	x-gw-ims-org-id	{{IMS_ORG}}	
☑	x-sandbox-name	{{SANDBOX_NAME}}	
☑	Accept	application/vnd.adobe.xed-id+json	
	Key	Value	Description

Use /global/ to get all the out of the box XDM Standard components. For the components youo create, use /tenant/

Search for "Order Details"

Status: 200 OK Time: 285 ms Size: 42.86 KB

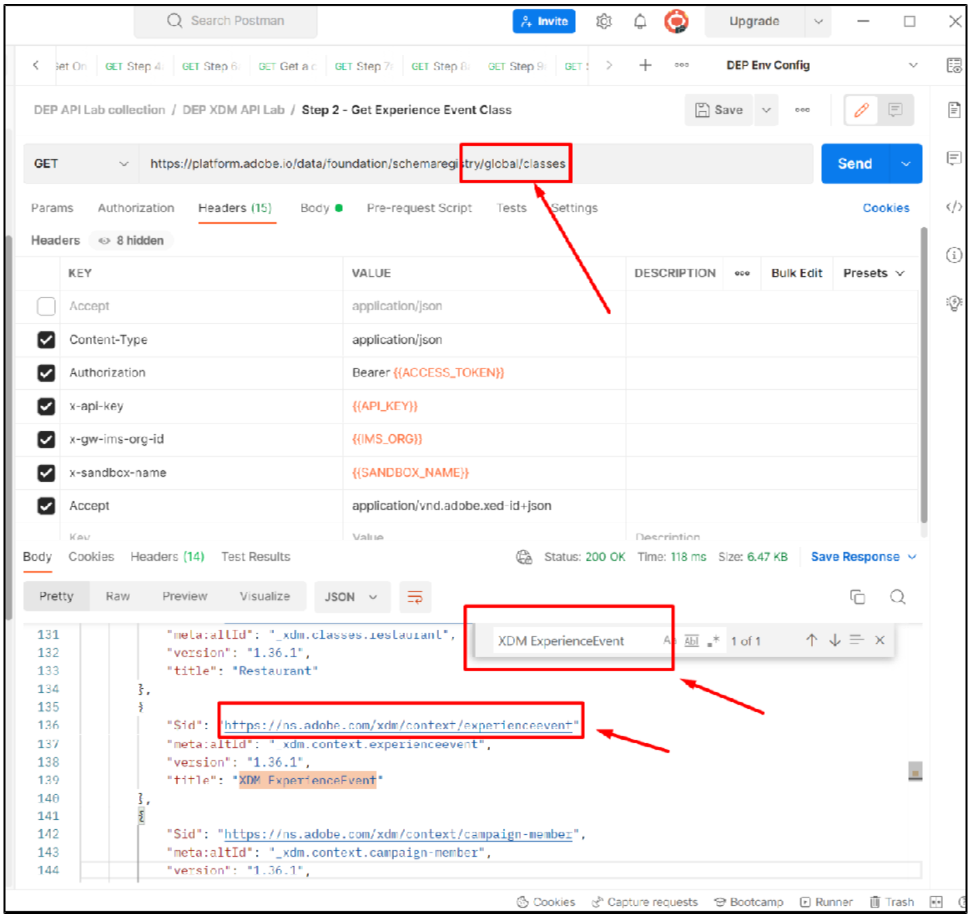
```
            "meta:altId": " xdm.context.experienceeven
            "version": "1.36.1",
            "title": "Stitching Fields"
        },
        {
            "$id": "https://ns.adobe.com/xdm/context/experienceevent order details",
            ~~"meta:altId": "_xdm.context.experienceevent-order-details",~~
            "version": "1.36.1",
            "title": "Order Details"
        },
        {
            "$id": "https://ns.adobe.com/xdm/context/experienceevent-advertising",
            "meta:altId": "_xdm.context.experienceevent-advertising",
            "version": "1.36.1",
```

Copy this "$id" onto a notepad

7.

4.4. Get the Schema ID for the XDM standard "Experience Event" Class.

1. **Click on "Step 2 - Get Experience Event Class"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
   a. In this step, we are looking for **"classes"** and not **Field Groups (Mixins)**
   b. This API call gives you all the standard XDM Classes
   c. **Execute "Step 2 - Get Experience Event Class" by clicking the "Send" button**
   d. After you execute the GET API call, search for **"XDM ExperienceEvent"** in the Response
   e. Copy the **"$id"** of the **"XDM ExperienceEvent"** Field Group and have it handy with you.

3. 

DEP API Lab collection / DEP XDM API Lab / Step 2 - Get Experience Event Class

GET https://platform.adobe.io/data/foundation/schemaregistry/global/classes

Headers (15)

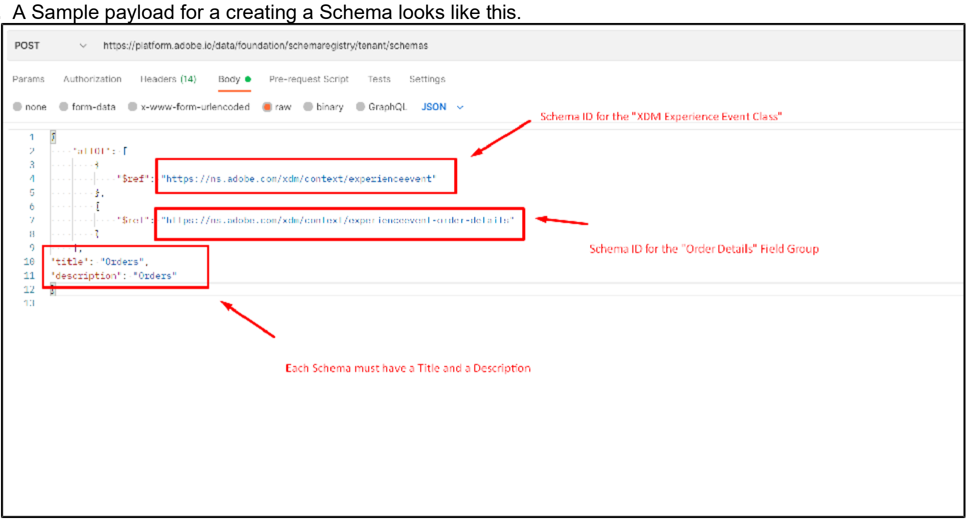
	KEY	VALUE	DESCRIPTION
□	Accept	application/json	
☑	Content-Type	application/json	
☑	Authorization	Bearer {{ACCESS_TOKEN}}	
☑	x-api-key	{{API_KEY}}	
☑	x-gw-ims-org-id	{{IMS_ORG}}	
☑	x-sandbox-name	{{SANDBOX_NAME}}	
☑	Accept	application/vnd.adobe.xed-id+json	

Status: 200 OK Time: 118 ms Size: 6.47 KB

```
            "meta:altId": "_xdm.classes.restaurant",
            "version": "1.36.1",
            "title": "Restaurant"
        },
        {
            "$id": "https://ns.adobe.com/xdm/context/experienceevent",
            "meta:altId": "_xdm.context.experienceevent",
            "version": "1.36.1",
            "title": "XDM ExperienceEvent"
        },
        {
            "$id": "https://ns.adobe.com/xdm/context/campaign-member",
            "meta:altId": "_xdm.context.campaign-member",
            "version": "1.36.1",
```

4.5. Create the "Orders" Schema.

1. **Click on "Step 3 - Create Orders Schema"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
    a. A Schema must include a **"Class"** and a possible **"Field Group".** This schema would be created by combining
        i. **"XDM Experience Event"** Class
        ii. **"Order Details"** Field Group
    b. Each Schema must have a **"title"** and a **"description".**
3. A Sample payload for a creating a Schema looks like this.

4. 

POST https://platform.adobe.io/data/foundation/schemaregistry/tenant/schemas

```
{
    "allOf": [
        {
            "$ref": "https://ns.adobe.com/xdm/context/experienceevent"
        },
        {
            "$ref": "https://ns.adobe.com/xdm/context/experienceevent-order-details"
        }
    ],
'title': "Orders",
'description': "Orders"
}
```

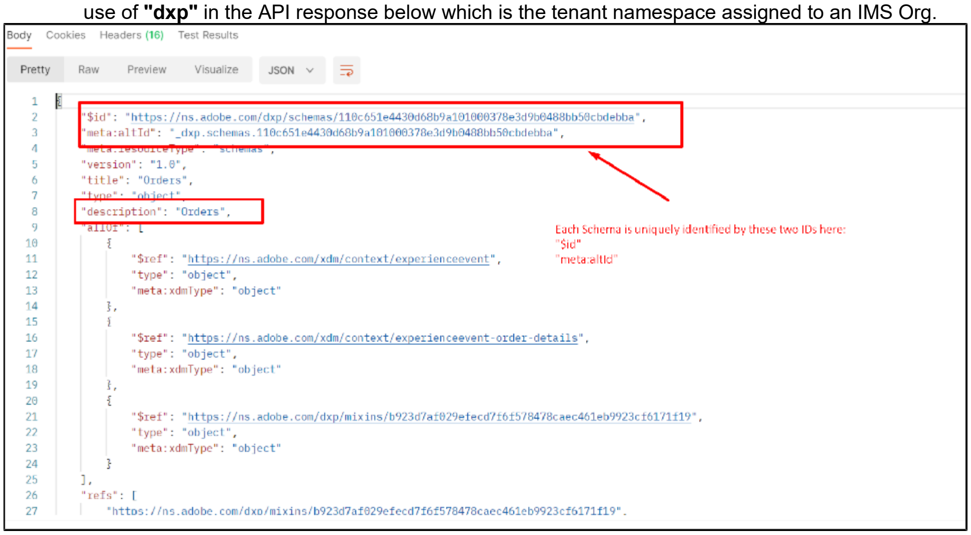
Schema ID for the "XDM Experience Event Class"

Schema ID for the "Order Details" Field Group

Each Schema must have a Title and a Description

5. **Execute "Step 3 - Create Orders Schema" by clicking the "Send" button**
    a. Copy the **"$id"** and **"meta:altId"** to use for future references to **"Orders"** Schema. Keep this copied IDs handy in a notepad as they will be required for future reference in further steps.

6. A Sample Response of a **"POST"** Schema call looks like this.
    a. Note how the Schema URL are structured for Adobe standard XDM components as compared to the Customer created ones. See the use of **"dxp"** in the API response below which is the tenant namespace assigned to an IMS Org.



```
{
    "$id": "https://ns.adobe.com/dxp/schemas/110c651e4430d68b9a101000378e3d9b0488bb50cbdebba",
    "meta:altId": "_dxp.schemas.110c651e4430d68b9a101000378e3d9b0488bb50cbdebba",
    "meta:resourceType": "schemas",
    "version": "1.0",
    "title": "Orders",
    "type": "object",
    "description": "Orders",
    "allOf": [
        {
            "$ref": "https://ns.adobe.com/xdm/context/experienceevent",
            "type": "object",
            "meta:xdmType": "object"
        },
        {
            "$ref": "https://ns.adobe.com/xdm/context/experienceevent-order-details",
            "type": "object",
            "meta:xdmType": "object"
        },
        {
            "$ref": "https://ns.adobe.com/dxp/mixins/b923d7af029efecd7f6f578478caec461eb9923cf6171f19",
            "type": "object",
            "meta:xdmType": "object"
        }
    ],
    "refs": [
        "https://ns.adobe.com/dxp/mixins/b923d7af029efecd7f6f578478caec461eb9923cf6171f19".
```

Each Schema is uniquely identified by these two IDs here:
"$id"
"meta:altId"

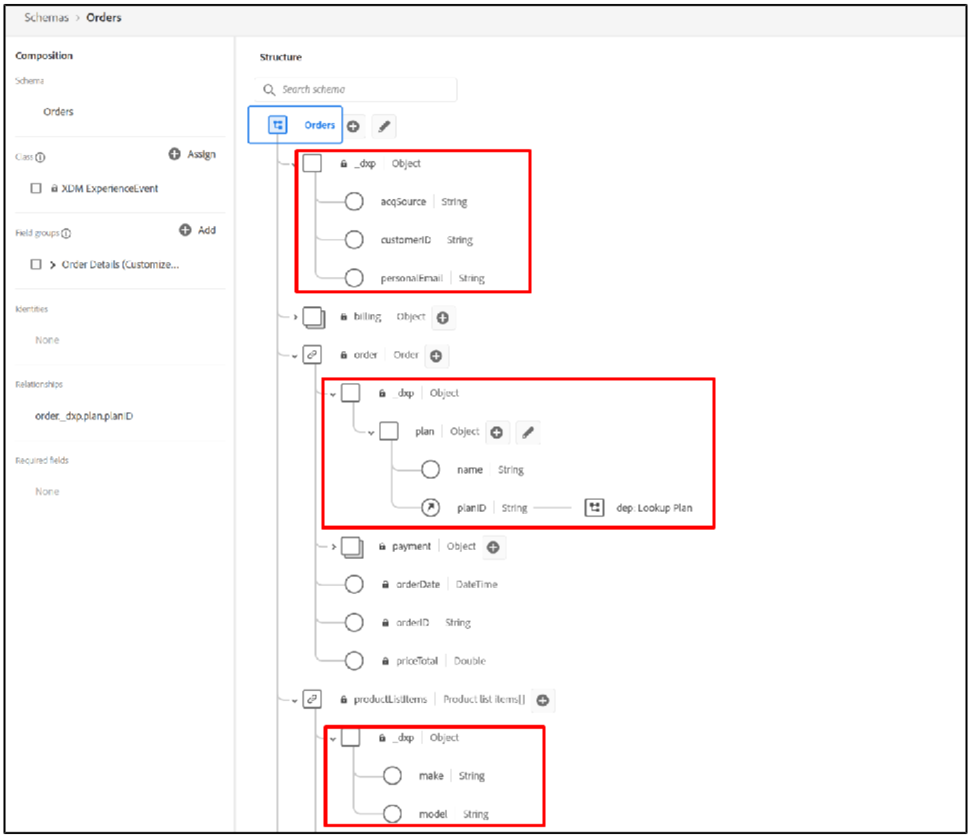
7.

4.6. Add Custom Properties to Orders Schema.

1. Go to the Schema UI editor and add the following properties to the **"Orders"** Schema
2. These additional properties are also in the Source to XDM Mapping document here : **Orders Schema - Customer to XDM Mapping.xlsx**

Customer Identity Descriptor

```
_dxp {
  acqSource
  customerID
  personalEmail
},
order {
  _dxp {
    plan {
      planID
      name
    }
  }
},
productListItems[
  _dxp {
    make
    model
  }
]
```

3. 

Schemas > Orders

Composition

Schema

Orders

Class ⓘ ⊕ Assign

□ XDM ExperienceEvent

Field groups ⓘ ⊕ Add

□ > Order Details (Customize...

Identities

None

Relationships

order._dxp.plan.planID

Required fields

None

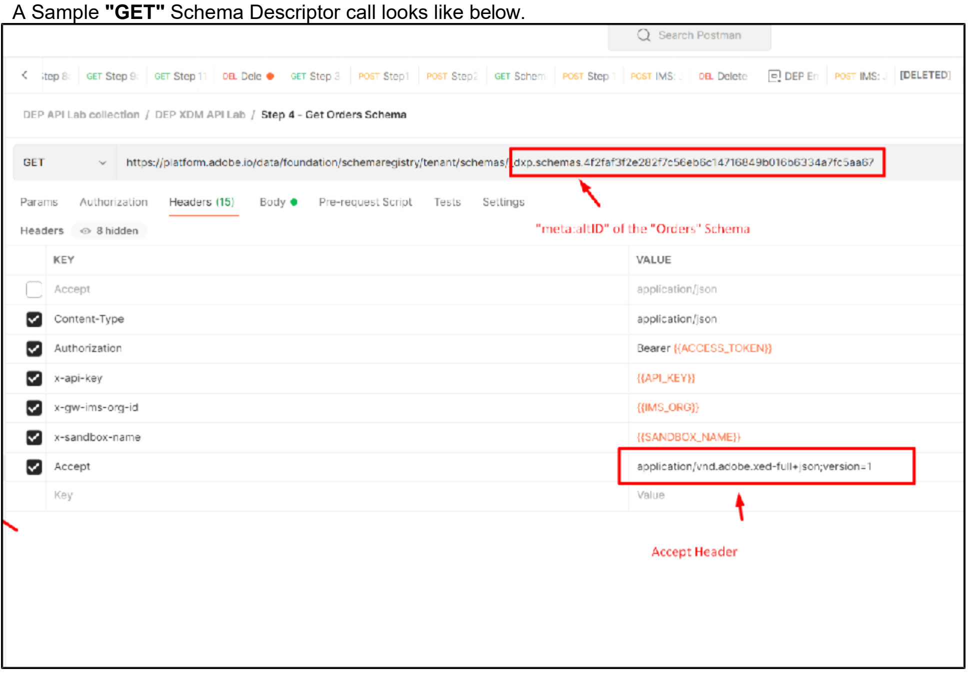
Structure

Search schema

Orders
- _dxp | Object
  - acqSource | String
  - customerID | String
  - personalEmail | String
- billing | Object
- order | Order
  - _dxp | Object
    - plan | Object
      - name | String
      - planID | String — dep: Lookup Plan
  - payment | Object
  - orderDate | DateTime
  - orderID | String
  - priceTotal | Double
- productListItems | Product list items[]
  - _dxp | Object
    - make | String
    - model | String

4.7. Browse the "Orders" Schema you created above

1. **Click on "Step 4 - Get Orders Schema"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
   a. Note the use of **"tenant" in https://platform.adobe.io/data/foundation/schemaregistry/tenant/mixins.**
   b. Observe the use of **"meta:altId"** to fetch the Schema from the registry. Replace the meta:altID in your API from the **step 4.5 above.**
   c. Also observe the **"Accept"** header which tells the schema registry what to return back in the API Response. You can see other "Accept" header options here Schema Registry APIs
   d. **Execute "Step 4 - Get Orders Schema" by clicking the "Send" button**
   e. With the pre-loaded Accept Header in the postman collection, you will see the fully exploded hierarchical view of the Orders Schema where every single property belonging to a class or a field group will be exposed.
   f. All the object properties will be exploded all the way to their leaf nodes.
3. A Sample **"GET"** Schema Descriptor call looks like below.

4. 

Search Postman
< Step 8: | GET Step 9: | GET Step 11 | DEL Dele | GET Step 3 | POST Step1 | POST Step2 | GET Schem | POST Step | POST IMS: | DEL Delete | DEP En | POST IMS: | [DELETED]

DEP API Lab collection / DEP XDM API Lab / Step 4 - Get Orders Schema

GET https://platform.adobe.io/data/foundation/schemaregistry/tenant/schemas/_dxp.schemas.4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67

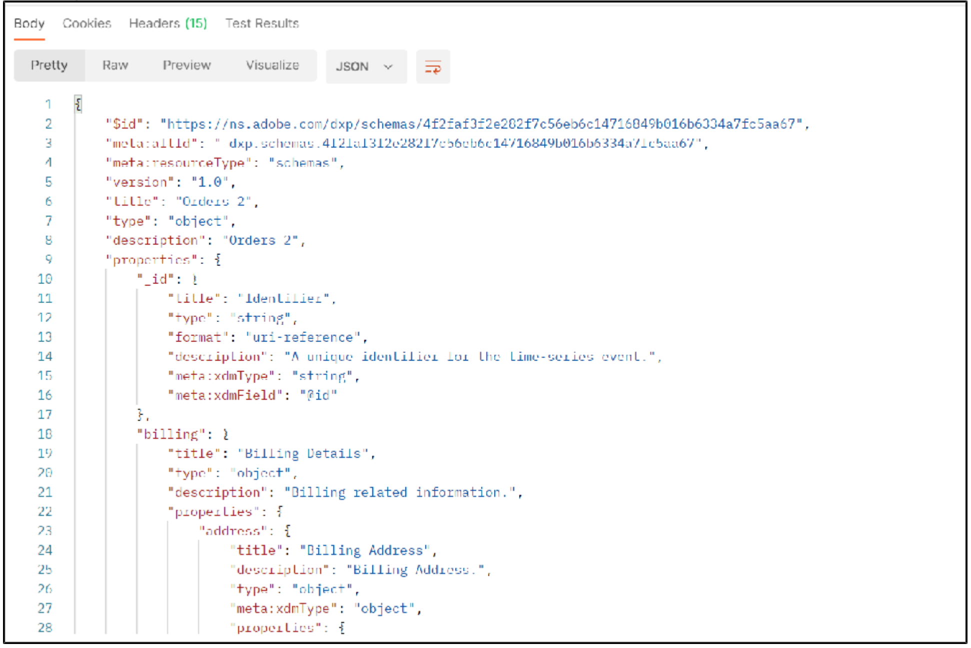
Params | Authorization | Headers (15) | Body | Pre-request Script | Tests | Settings
Headers 8 hidden

"meta:altID" of the "Orders" Schema

	KEY	VALUE
□	Accept	application/json
☑	Content-Type	application/json
☑	Authorization	Bearer {{ACCESS_TOKEN}}
☑	x-api-key	{{API_KEY}}
☑	x-gw-ims-org-id	{{IMS_ORG}}
☑	x-sandbox-name	{{SANDBOX_NAME}}
☑	Accept	application/vnd.adobe.xed-full+json;version=1
	Key	Value

Accept Header

5. A Sample API **"RESPONSE"** looks like below:

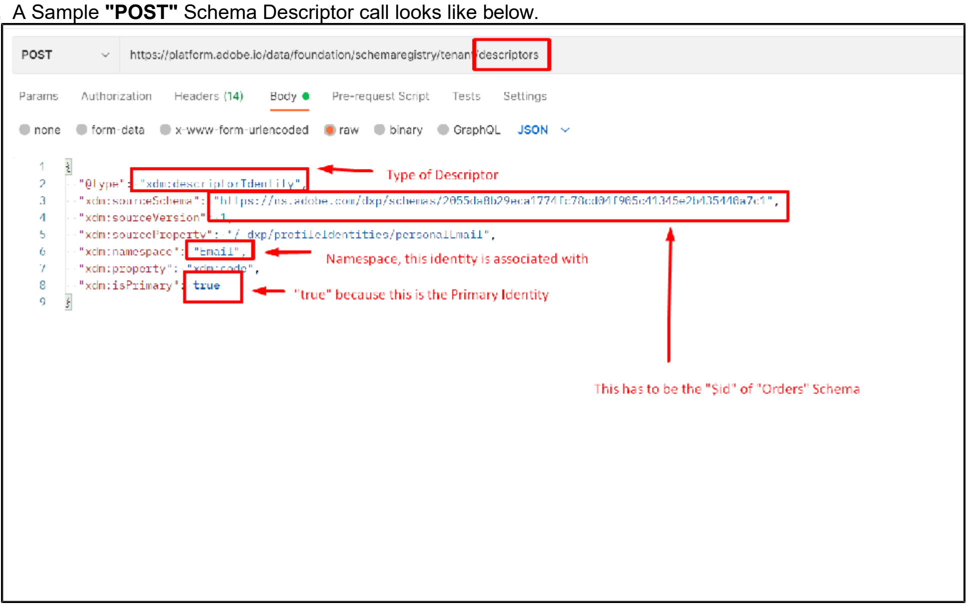


```
{
    "$id": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
    "meta:altId": "_dxp.schemas.4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
    "meta:resourceType": "schemas",
    "version": "1.0",
    "title": "Orders 2",
    "type": "object",
    "description": "Orders 2",
    "properties": {
        "_id": {
            "title": "Identifier",
            "type": "string",
            "format": "uri-reference",
            "description": "A unique identifier for the time-series event.",
            "meta:xdmType": "string",
            "meta:xdmField": "@id"
        },
        "billing": {
            "title": "Billing Details",
            "type": "object",
            "description": "Billing related information.",
            "properties": {
                "address": {
                    "title": "Billing Address",
                    "description": "Billing Address.",
                    "type": "object",
                    "meta:xdmType": "object",
                    "properties": {
```

6.

4.8. **Create "Primary Identity" Descriptor for "Orders" Schema**

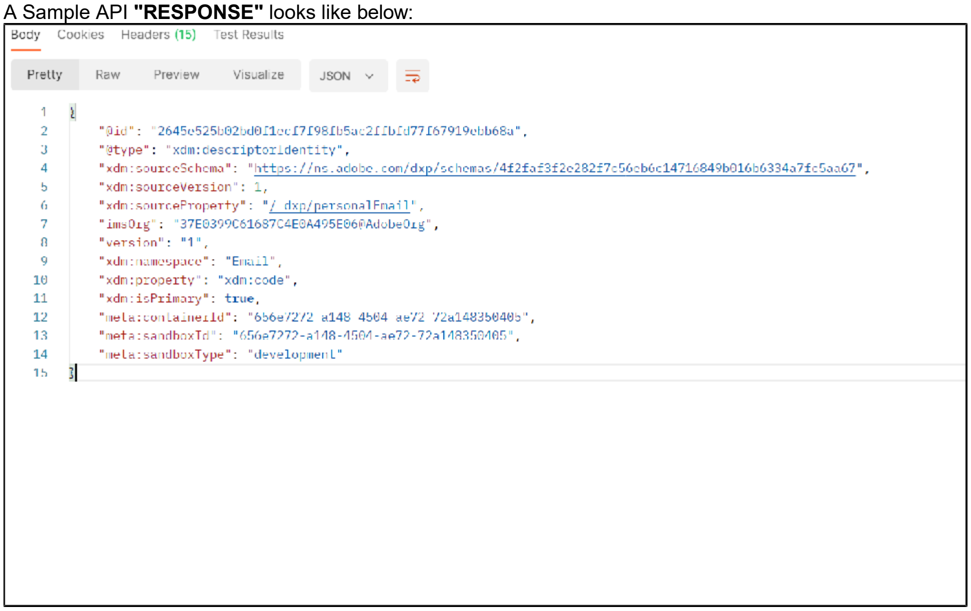
1. **Click on "Step 5.1 - Create Primary Identity for Orders Schema"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
    a. You would have identified the **"Primary"** Identities for **"Orders"** Schema in the SID lab.
    b. Note the use of **"descriptors" in https://platform.adobe.io/data/foundation/schemaregistry/tenant/descriptors.**
    c. Schema Properties:
        i. Type - Descriptor type. Its **"xdm:descriptorIdentity"** in this case. You would have gone through the **Schema Descriptors** in the **SID class**.
        ii. Source Schema - The **"$id"** of the **"Orders"** Schema since we are defining an identity for the "Orders" Schema.
        iii. Source Property - Full **"path"** for the Schema property which needs to be marked as an Identity (**"/_dxp/personalEmail"** in this case).
        iv. Name Space - The **"namespace"** to which this identity is associated with. (**"Email"** in this case).
        v. Is Primary - This marked as **"true"** when the identity is Primary. For all other identities, this must be "false" as a schema can only have 1 Primary Identity.
3. A Sample **"POST"** Schema Descriptor call looks like below.


POST
https://platform.adobe.io/data/foundation/schemaregistry/tenant/descriptors
Params
Authorization
Headers (14)
Body
Pre-request Script
Tests
Settings
○ none
○ form-data
○ x-www-form-urlencoded
● raw
○ binary
○ GraphQL
JSON
Type of Descriptor
Namespace, this identity is associated with
"true" because this is the Primary Identity
This has to be the "$id" of "Orders" Schema

```
{
    "@type": "xdm:descriptorIdentity",
    "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/2055da8b29eca1774fc78cd04f905c41345e2b435448a7c1",
    "xdm:sourceVersion": 1,
    "xdm:sourceProperty": "/_dxp/profileIdentities/personalEmail",
    "xdm:namespace": "Email",
    "xdm:property": "xdm:code",
    "xdm:isPrimary": true
}
```

4.

5. A Sample API **"RESPONSE"** looks like below:



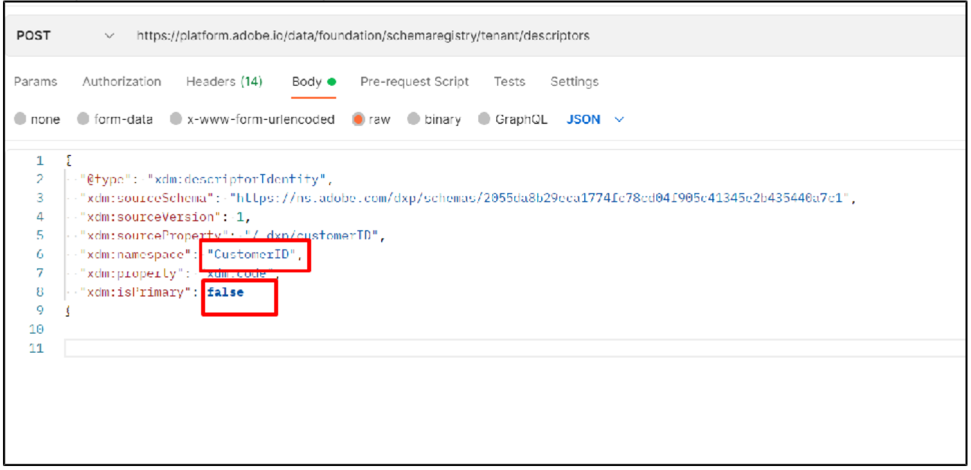
```
{
    "@id": "2645e525b02bd0f1ecf7f98fb5ac2ffbfd77f67919ebb68a",
    "@type": "xdm:descriptorIdentity",
    "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
    "xdm:sourceVersion": 1,
    "xdm:sourceProperty": "/_dxp/personalEmail",
    "imsOrg": "37E0399C61687C4E0A495E06@AdobeOrg",
    "version": "1",
    "xdm:namespace": "Email",
    "xdm:property": "xdm:code",
    "xdm:isPrimary": true,
    "meta:containerId": "656e7272-a148-4504-ae72-72a148350405",
    "meta:sandboxId": "656e7272-a148-4504-ae72-72a148350405",
    "meta:sandboxType": "development"
}
```

6.

7. **Execute "Step 5.1 - Create Primary Identity for Orders Schema" by clicking the "Send" button**

4.9. **Create Remaining Identity Descriptor(s) for "Orders" Schema**

1. **Click on "Step 5.2 - Create CustomerID Identity for Orders Schema"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
   a. You would have identified the **"Secondary"** Identities for **"Orders"** Schema in the SID lab.
   b. Descriptor Properties:
      i. Type - Descriptor type. Its **"xdm:descriptorIdentity"** in this case
      ii. Source Schema - The **"$id"** of the **"Orders"** Schema since we are defining an identity for the "Orders" Schema.
      iii. Source Property - Full **"path"** for the Schema property which needs to be marked as an Identity (**"/_dxp/customerID"** in this case).
      iv. Source Version - Version number of the Source schema. Always defined as **1**
      v. Name Space - The **"namespace"** to which this identity is associated with (**"customerID"** in this case).
      vi. Is Primary - This marked as **"false"** when the identity is non-Primary.
      vii. You can have more than 1 secondary identities for a Schema as long as they are valid and do not create collisions across different individuals.
3. A Sample **"POST"** Schema Descriptor call looks like below.

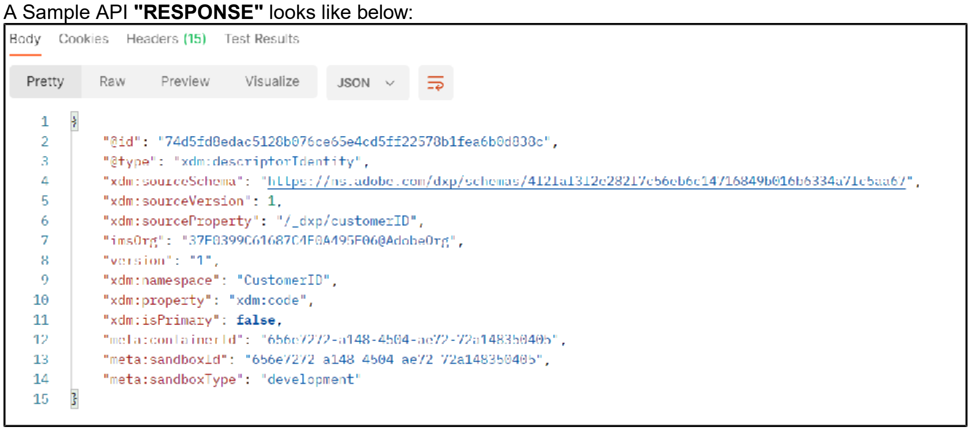


https://platform.adobe.io/data/foundation/schemaregistry/tenant/descriptors

```
{
  "@type": "xdm:descriptorIdentity",
  "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/2055da8b29eca1774fc78cd04f905c41345e2b435440a7c1",
  "xdm:sourceVersion": 1,
  "xdm:sourceProperty": "/_dxp/customerID",
  "xdm:namespace": "CustomerID",
  "xdm:property": "xdm:code",
  "xdm:isPrimary": false
}
```

4.

5. A Sample API **"RESPONSE"** looks like below:

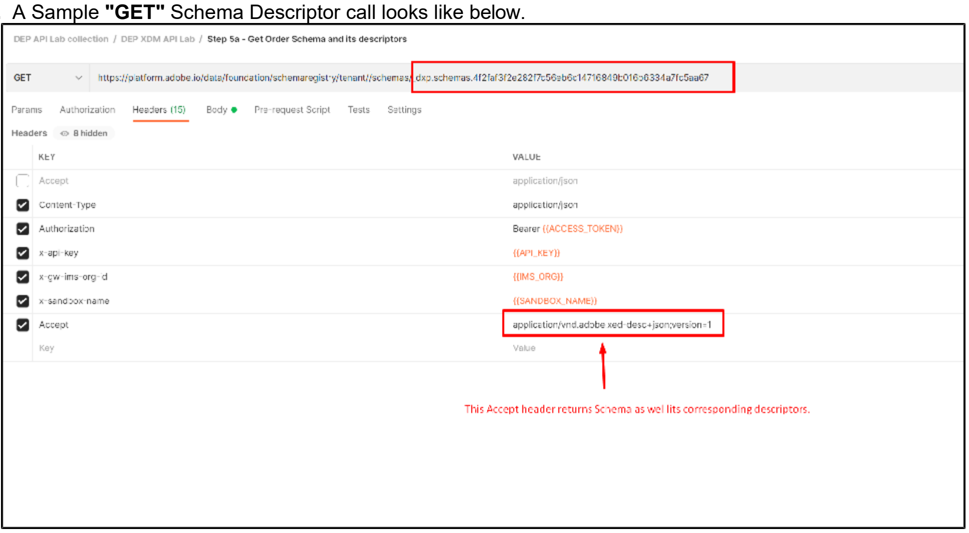


```
{
    "@id": "74d5fd8edac5128b076ce65e4cd5ff22578b1fea6b0d838c",
    "@type": "xdm:descriptorIdentity",
    "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4121a1312e28217c56eb6c14716849b016b6334a71c5aa67",
    "xdm:sourceVersion": 1,
    "xdm:sourceProperty": "/_dxp/customerID",
    "imsOrg": "37F0399C61687C4F0A495F06@AdobeOrg",
    "version": "1",
    "xdm:namespace": "CustomerID",
    "xdm:property": "xdm:code",
    "xdm:isPrimary": false,
    "meta:containerId": "656e7272-a148-4504-ae72-72a148350405",
    "meta:sandboxId": "656e7272 a148 4504 ae72 72a148350405",
    "meta:sandboxType": "development"
}
```

6.
7. **Execute "Step 5.2 - Create CustomerID Identity for Orders Schema" by clicking the "Send" button**

4.10. Browse the "Orders" Schema to see Identities.

1. **Click on "Step 6 - Get Order Schema and its descriptors"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
    a. A Sample **"GET "** Schema call looks like below.
    b. Observe the use of **"meta:altId"** to fetch the Schema from the registry.
    c. Also observe the changed **"Accept"** header in this case. This accept header will return the Schemas along with its associated descriptors in the API response. This tells the schema registry what to return back in the API Response. You can see other "Accept" header options here Schema Registry APIs
    d. With the pre-loaded Accept Header in the postman collection, you will see the fully exploded hierarchical view of the Orders Schema where every single property belonging to a class or a field group will be exposed.
    e. All the object properties will be exploded all the way to their leaf nodes.
3. A Sample **"GET"** Schema Descriptor call looks like below.



DEP API Lab collection / DEP XDM API Lab / Step 5a - Get Order Schema and its descriptors
Params
Authorization
Headers (15)
Body ●
Pre-request Script
Tests
Settings
Headers
8 hidden

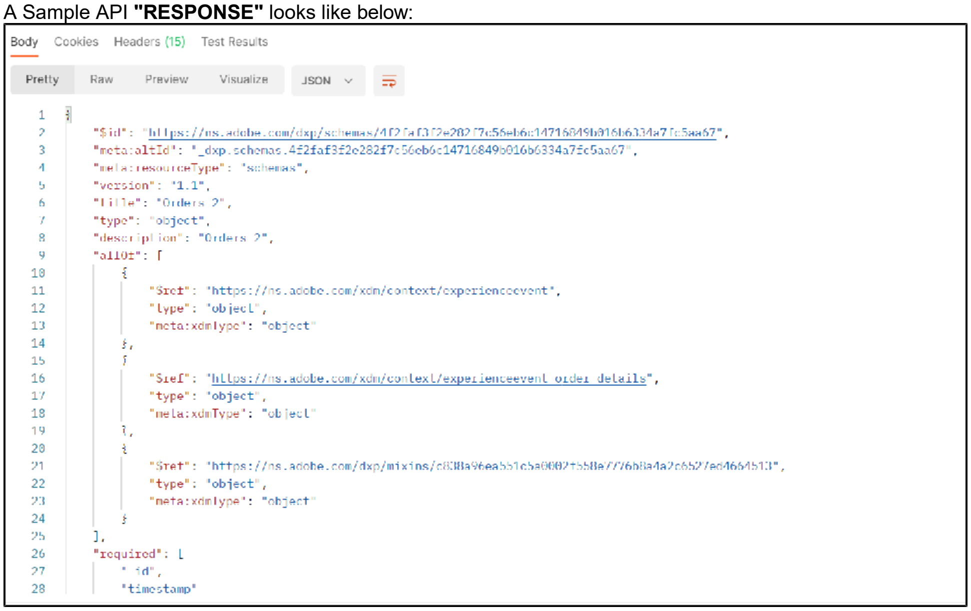
GET https://platform.adobe.io/data/foundation/schemaregistry/tenant//schemas/dxp.schemas.4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67

	KEY	VALUE
□	Accept	application/json
☑	Content-Type	application/json
☑	Authorization	Bearer {{ACCESS_TOKEN}}
☑	x-api-key	{{API_KEY}}
☑	x-gw-ims-org-id	{{IMS_ORG}}
☑	x-sandbox-name	{{SANDBOX_NAME}}
☑	Accept	application/vnd.adobe.xed-desc+json;version=1
	Key	Value

This Accept header returns Schema as wel lits corresponding descriptors.

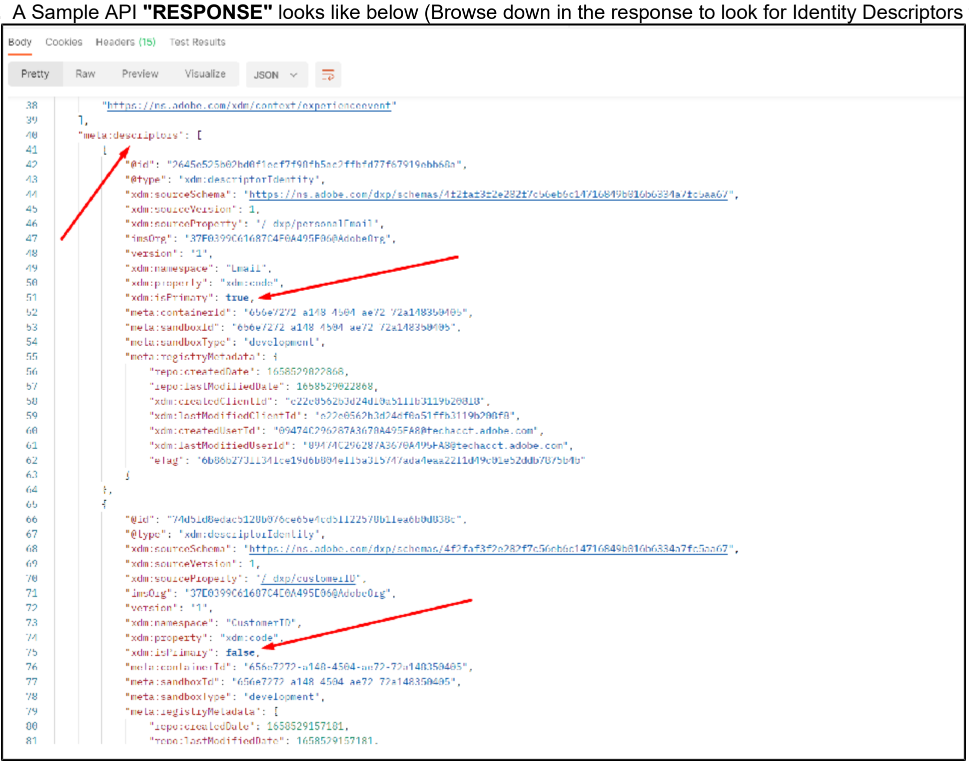
4.

5. A Sample API **"RESPONSE"** looks like below:



```
{
    "$id": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
    "meta:altId": "_dxp.schemas.4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
    "meta:resourceType": "schemas",
    "version": "1.1",
    "title": "Orders 2",
    "type": "object",
    "description": "Orders 2",
    "allOf": [
        {
            "$ref": "https://ns.adobe.com/xdm/context/experienceevent",
            "type": "object",
            "meta:xdmType": "object"
        },
        {
            "$ref": "https://ns.adobe.com/xdm/context/experienceevent_order_details",
            "type": "object",
            "meta:xdmType": "object"
        },
        {
            "$ref": "https://ns.adobe.com/dxp/mixins/c838a96eab51cba0002fb58e7776b8a4a2c6527ed4664513",
            "type": "object",
            "meta:xdmType": "object"
        }
    ],
    "required": [
        "_id",
        "timestamp"
```

6.
7. A Sample API **"RESPONSE"** looks like below (Browse down in the response to look for Identity Descriptors for this Schema):



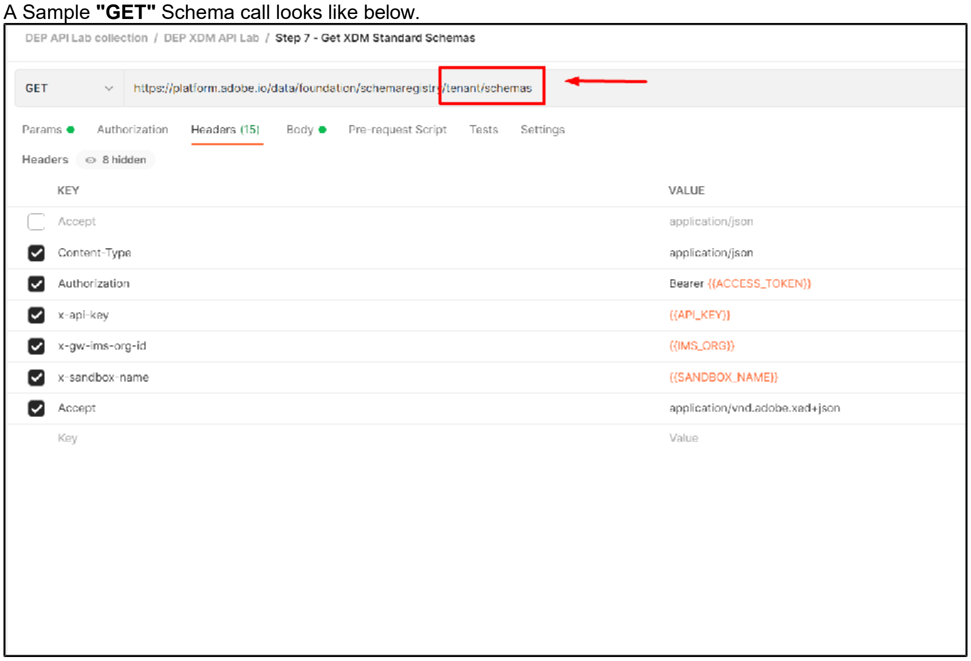
```
            "https://ns.adobe.com/xdm/context/experienceevent"
        ],
        "meta:descriptors": [
            {
                "@id": "2645e525b02bd0f1ecf7f90fb5ac2ffbfd77f67919ebb68a",
                "@type": "xdm:descriptorIdentity",
                "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
                "xdm:sourceVersion": 1,
                "xdm:sourceProperty": "/_dxp/personalEmail",
                "imsOrg": "37F0399C61687C4F0A495F06@AdobeOrg",
                "version": "1",
                "xdm:namespace": "Email",
                "xdm:property": "xdm:code",
                "xdm:isPrimary": true,
                "meta:containerId": "656e7272-a148-4504-ae72-72a148350405",
                "meta:sandboxId": "656e7272-a148-4504-ae72-72a148350405",
                "meta:sandboxType": "development",
                "meta:registryMetadata": {
                    "repo:createdDate": 1658529022868,
                    "repo:lastModifiedDate": 1658529022868,
                    "xdm:createdClientId": "e22e0562b3d24d10a511fb3119b20818",
                    "xdm:lastModifiedClientId": "e22e0562b3d24df0a51ffb3119b200f8",
                    "xdm:createdUserId": "09474C296287A3670A495FA8@techacct.adobe.com",
                    "xdm:lastModifiedUserId": "09474C296287A3670A495FA8@techacct.adobe.com",
                    "eTag": "6b86b27311341ce19d6b804e115a315747ada4eaa2211d49c01e52ddb7875b4b"
                }
            },
            {
                "@id": "74d51d8edac5120b076ce65e4cd51122570b11ea6b0d838c",
                "@type": "xdm:descriptorIdentity",
                "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
                "xdm:sourceVersion": 1,
                "xdm:sourceProperty": "/_dxp/customerID",
                "imsOrg": "37E0399C61687C4E0A495E06@AdobeOrg",
                "version": "1",
                "xdm:namespace": "CustomerID",
                "xdm:property": "xdm:code",
                "xdm:isPrimary": false,
                "meta:containerId": "656e7272-a148-4504-ae72-72a148350405",
                "meta:sandboxId": "656e7272-a148-4504-ae72-72a148350405",
                "meta:sandboxType": "development",
                "meta:registryMetadata": {
                    "repo:createdDate": 1658529157181,
                    "repo:lastModifiedDate": 1658529157181,
```

8.
9. **Execute "Step 6 - Get Order Schema and its descriptors" by clicking the "Send" button**

4.11. Get the Schema ID for the Lookup Schemas

1. **Execute the "Step 7 - Get Lookup Schemas"** API call in the **"DEP XDM API Lab"** Folder.
2. Within your response from **Step 7** above
   a. Look for the Product Schema (Search keyword "**dep: Lookup Product**")
   b. Look for the Store Schema (Search keyword "**dep: Lookup Store**")
   c. Look for the Plan Schema (Search keyword "**dep: Lookup Plan**")
   a. **Please note the following points:**
      i. These three schemas should be pre-deployed on your Sandbox.
      ii. Relationship Descriptors will be created from Orders schema to each of these three lookup schemas.
      iii. Copy the **"$id"** of these three schemas and have them handy with you. They will be required in CREATE/POST Relationship descriptor API calls.

3. A Sample **"GET"** Schema call looks like below.



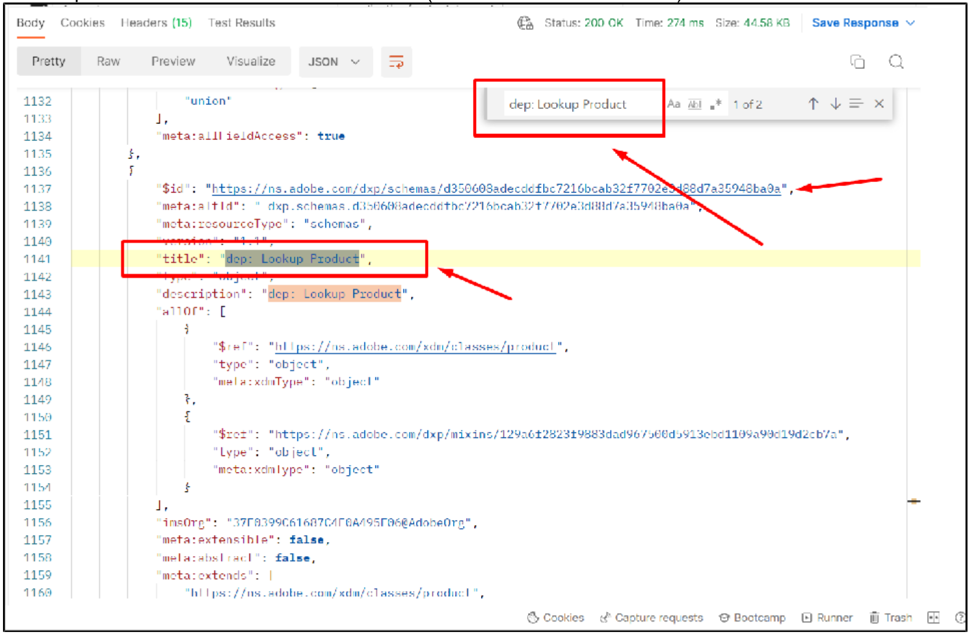
DEP API Lab collection / DEP XDM API Lab / Step 7 - Get XDM Standard Schemas

GET https://platform.adobe.io/data/foundation/schemaregistry/tenant/schemas

	KEY	VALUE
□	Accept	application/json
☑	Content-Type	application/json
☑	Authorization	Bearer {{ACCESS_TOKEN}}
☑	x-api-key	{{API_KEY}}
☑	x-gw-ims-org-id	{{IMS_ORG}}
☑	x-sandbox-name	{{SANDBOX_NAME}}
☑	Accept	application/vnd.adobe.xed+json
	Key	Value

4.

5. A Sample API **"RESPONSE"** looks like below (Search for Product Schema):



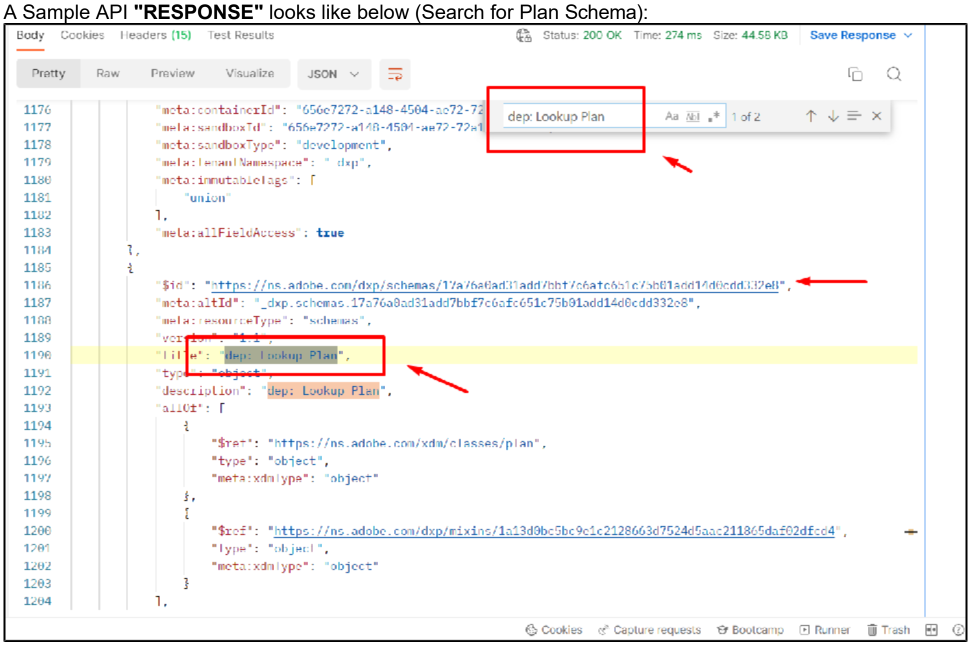
Status: 200 OK Time: 274 ms Size: 44.58 KB

dep: Lookup Product

```
                "union"
            ],
            "meta:allFieldAccess": true
        },
        {
            "$id": "https://ns.adobe.com/dxp/schemas/d350608adecddfbc7216bcab32f7702e3d88d7a35948ba0a",
            "meta:altId": " dxp.schemas.d350608adecddfbc7216bcab32f7702e3d88d7a35948ba0a",
            "meta:resourceType": "schemas",
            "version": "1.1",
            "title": "dep: Lookup Product",
            "type": "object",
            "description": "dep: Lookup Product",
            "allOf": [
                {
                    "$ref": "https://ns.adobe.com/xdm/classes/product",
                    "type": "object",
                    "meta:xdmType": "object"
                },
                {
                    "$ref": "https://ns.adobe.com/dxp/mixins/129a6f2823f9883dad967500d5913ebd1109a90d19d2cb7a",
                    "type": "object",
                    "meta:xdmType": "object"
                }
            ],
            "imsOrg": "37F0399C61687C4F0A495F06@AdobeOrg",
            "meta:extensible": false,
            "meta:abstract": false,
            "meta:extends": [
                "https://ns.adobe.com/xdm/classes/product",
```

6.

7. A Sample API **"RESPONSE"** looks like below (Search for Plan Schema):

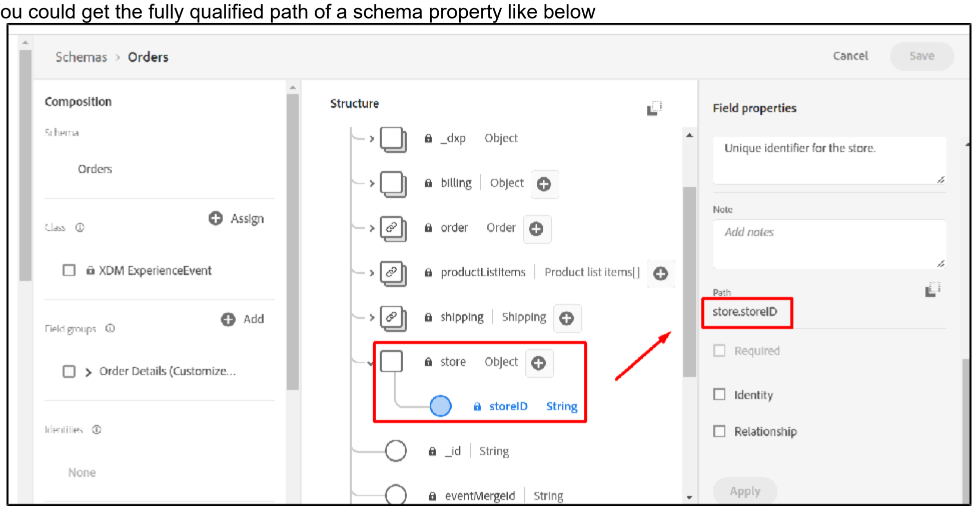


```
                "meta:containerId": "656e7272-a148-4504-ae72-72
                "meta:sandboxId": "656e7272-a148-4504-ae72-72e1
                "meta:sandboxType": "development",
                "meta:tenantNamespace": "_dxp",
                "meta:immutableTags": [
                    "union"
                ],
                "meta:allFieldAccess": true
            },
            {
                "$id": "https://ns.adobe.com/dxp/schemas/17a76a0ad31add7bbf7c6afc651c75b01add14d0cdd332e8",
                "meta:altId": "_dxp.schemas.17a76a0ad31add7bbf7c6afc651c75b01add14d0cdd332e8",
                "meta:resourceType": "schemas",
                "version": "1.1",
                "title": "dep: Lookup Plan",
                "type": "object",
                "description": "dep: Lookup Plan",
                "allOf": [
                    {
                        "$ref": "https://ns.adobe.com/xdm/classes/plan",
                        "type": "object",
                        "meta:xdmType": "object"
                    },
                    {
                        "$ref": "https://ns.adobe.com/dxp/mixins/1a13d0bc5bc9c1c2128663d7524d5aac211865daf02dfcd4",
                        "type": "object",
                        "meta:xdmType": "object"
                    }
                ],
```

8.

4.12. Create "Orders to Store" Relationship Descriptor for "Orders" Schema

1. **Click on "Step 8 - Relationship Descriptor Orders To Store"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
   a. You would have identified the **"relationships"** from **"Orders"** Schema in the **SID lab**.
   b. **"Orders : Store"** relationship cardinality is **"M:1"** .
   c. A relationship descriptor is always defined from either the **"Profile"** or the **"Experience Event"** class based schemas (Experience Event class in this case).
   d. It's worth noting that AEP only supports **1 hop join** from Profile or EE schemas i.e. we can only create **1 level lookup** relationships.
   e. You should have the **"$id"** of the Store Schema.
   f. Descriptor Properties:
      i. Type - Descriptor type. Its **"xdm:descriptorOneToOne"** in this case
         1. Even though the descriptor type refers to **OneToOne** , we use this to define relationships for **M:1** cardinality as well.
      ii. Source Schema - The **"$id"** of the schema, the relationship is originating from (**"Orders"** Schema in this case). Please note that you can create a descriptor from either side of the relationship but Profile services will only look for relationships originating from either the "profile" class based schemas or "Experience Event" Class based schemas.
      iii. Source Property - Full **"path"** for the Schema property which needs to be marked as a Relationship (**"/store/storeID"** in this case). You could get the fully qualified path of a schema property like below
         1.

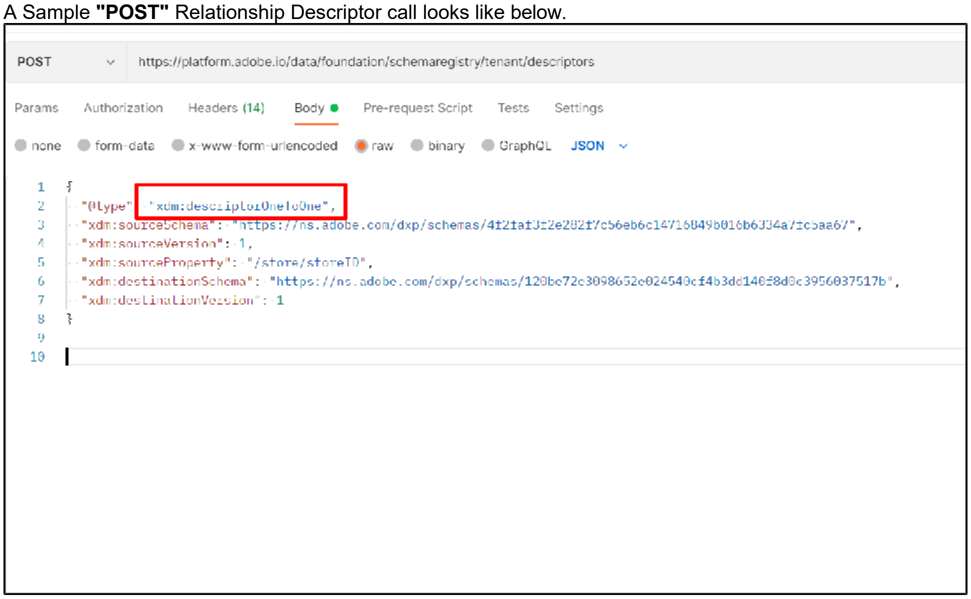


Unique identifier for the store.

Path: store.storeID

      iv. Source Version - Version number of the Source schema. Always defined as **1**
      v. Destination Schema - The **"$id"** of the schema, the relationship is referring to (**"Store"** Schema in this case).
      vi. Destination Property - This is an optional parameter. A relationship descriptor assumes that the source property , the relation is defined on is marked as a primary identity in the Destination Schema.
      vii. Destination Version - Version number of the destination schema. Always defined as **1**

3. A Sample **"POST"** Relationship Descriptor call looks like below.

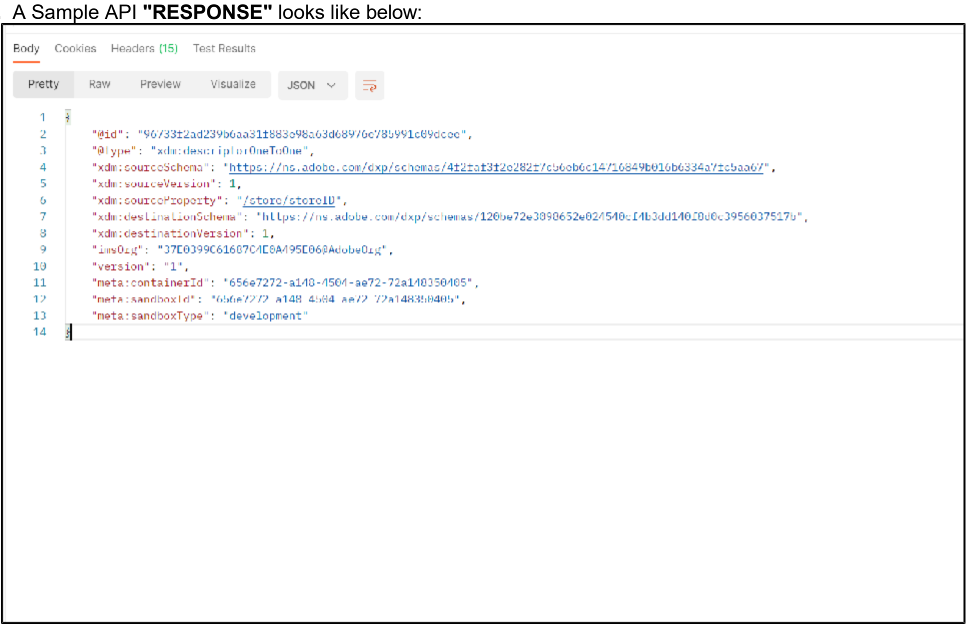

Params
Authorization
Headers (14)
Body
Pre-request Script
Tests
Settings
○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

POST https://platform.adobe.io/data/foundation/schemaregistry/tenant/descriptors

```
{
  "@type": "xdm:descriptorOneToOne",
  "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
  "xdm:sourceVersion": 1,
  "xdm:sourceProperty": "/store/storeID",
  "xdm:destinationSchema": "https://ns.adobe.com/dxp/schemas/120be72e3098652e024540cf4b3dd140f8d0c3956037517b",
  "xdm:destinationVersion": 1
}
```

4.

5. A Sample API **"RESPONSE"** looks like below:



```
{
    "@id": "96733f2ad239b6aa31f883e98a63d68976e785991c09dcee",
    "@type": "xdm:descriptorOneToOne",
    "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
    "xdm:sourceVersion": 1,
    "xdm:sourceProperty": "/store/storeID",
    "xdm:destinationSchema": "https://ns.adobe.com/dxp/schemas/120be72e3098652e024540cf4b3dd140f8d0c3956037517b",
    "xdm:destinationVersion": 1,
    "imsOrg": "37E0399C61687C4E0A495E06@AdobeOrg",
    "version": "1",
    "meta:containerId": "656e7272-a148-4504-ae72-72a148350405",
    "meta:sandboxId": "656e7272-a148-4504-ae72-72a148350405",
    "meta:sandboxType": "development"
}
```

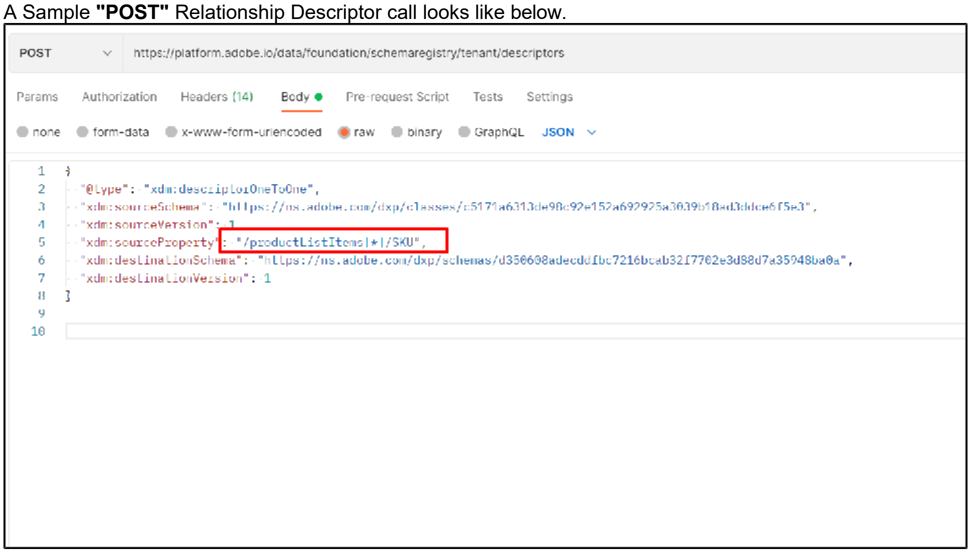
6.

7. **Execute "Step 8 - Relationship Descriptor Orders To Store" by clicking the "Send" button**

4.13. Create "Orders to Product" Relationship Descriptor for "Orders" Schema

1. **Click on "Step 9 - Relationship Descriptor Orders To Product"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
    a. Descriptor Properties:
        i. Type - Descriptor type. Its **"xdm:descriptorOneToOne"** in this case
        ii. Source Schema - The **"$id"** of the schema, the relationship is originating from (**"Orders"** Schema in this case).
        iii. Source Property - Full **"path"** for the Schema property which needs to be marked as a Relationship (**"/productListItems[*]/SKU"** in this case).
            1. Observe the property path here when the property belongs to an array of objects. The array is referenced as **ArrayName[*].**
        iv. Source Version - Version number of the Source schema. Always defined as **1**
        v. Destination Schema - The **"$id"** of the schema, the relationship is referring to (**"Product"** Schema in this case).
        vi. Destination Property - This is an optional parameter. A relationship descriptor assumes that the source property , the relation is defined on is marked as a primary identity in the Destination Schema.
        vii. Destination Version - Version number of the destination schema. Always defined as **1**

3. A Sample **"POST"** Relationship Descriptor call looks like below.



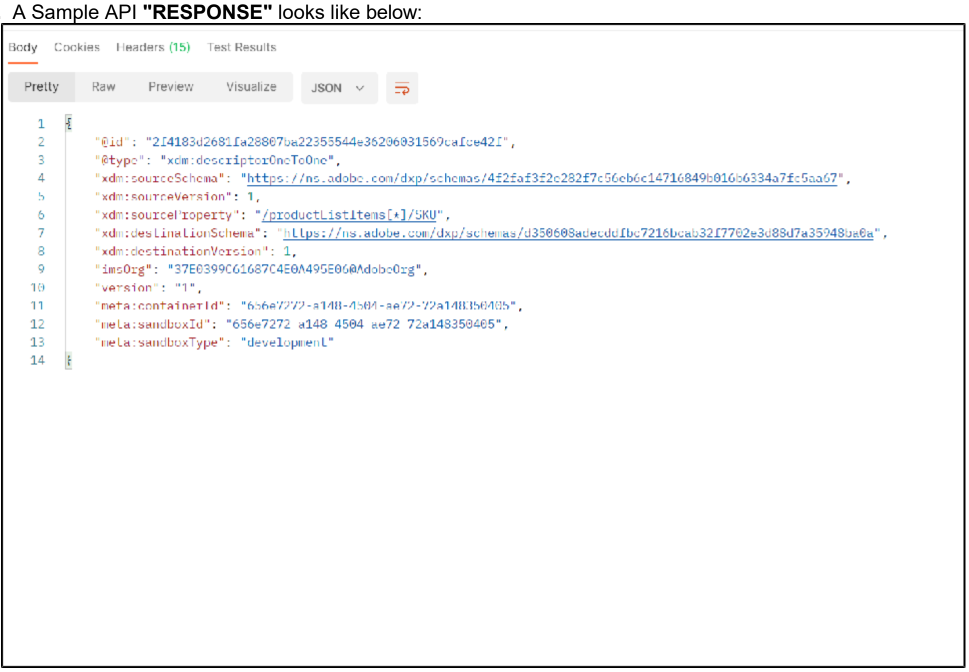
POST https://platform.adobe.io/data/foundation/schemaregistry/tenant/descriptors

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
  "@type": "xdm:descriptorOneToOne",
  "xdm:sourceSchema": "https://ns.adobe.com/dxp/classes/c5171a6313de98c92e152a692925a3039b18ad3ddce6f5e3",
  "xdm:sourceVersion": 1,
  "xdm:sourceProperty": "/productListItems[*]/SKU",
  "xdm:destinationSchema": "https://ns.adobe.com/dxp/schemas/d350608adecddfbc7216bcab32f7702e3d88d7a35948ba0a",
  "xdm:destinationVersion": 1
}
```

4.

5. A Sample API **"RESPONSE"** looks like below:



```
{
    "@id": "2f4183d2681fa28807ba22355544e36206031569cafce42f",
    "@type": "xdm:descriptorOneToOne",
    "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2c282f7c56cb6c14716849b016b6334a7fc5aa67",
    "xdm:sourceVersion": 1,
    "xdm:sourceProperty": "/productListItems[*]/SKU",
    "xdm:destinationSchema": "https://ns.adobe.com/dxp/schemas/d350608adecddfbc7216bcab32f7702e3d88d7a35948ba0a",
    "xdm:destinationVersion": 1,
    "imsOrg": "37E0399C61687C4E0A495E06@AdobeOrg",
    "version": "1",
    "meta:containerId": "656e7272-a148-4504-ae72-72a148350405",
    "meta:sandboxId": "656e7272-a148-4504-ae72-72a148350405",
    "meta:sandboxType": "development"
}
```

6.

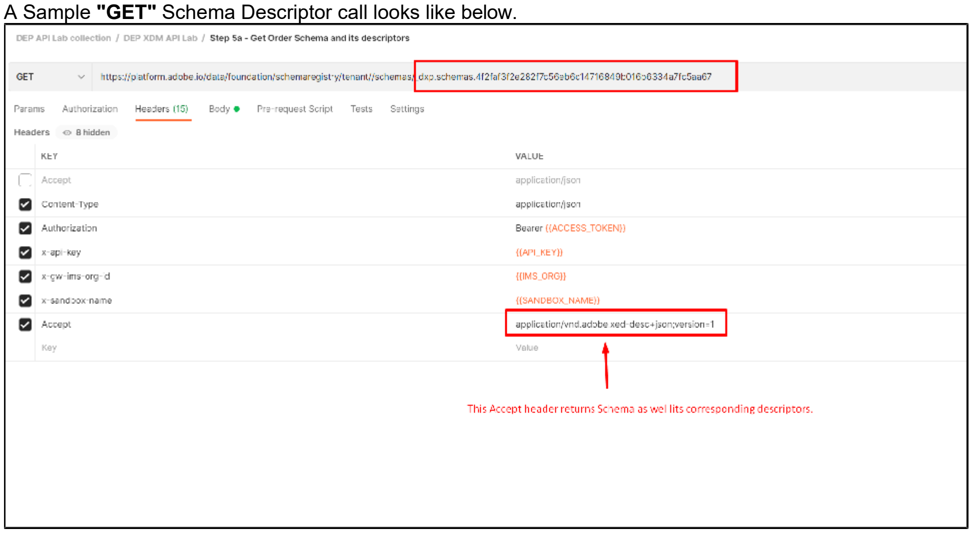
7. **Execute "Step 9 - Relationship Descriptor Orders To Product" by clicking the "Send" button**

4.14. **Create "Orders to Plan" Relationship Descriptor for "Orders" Schema**

1. **Execute "Step 10 - Relationship Descriptor Orders To Plan" by clicking the "Send" button**

4.15. **Browse the "Orders" Schema to see your Relationship Descriptors.**

1. **Execute the "Step 11 - Get Order Schema and its descriptors" by clicking the "Send" button**
2. A Sample **"GET"** Schema Descriptor call looks like below.



DEP API Lab collection / DEP XDM API Lab / Step 5a - Get Order Schema and its descriptors

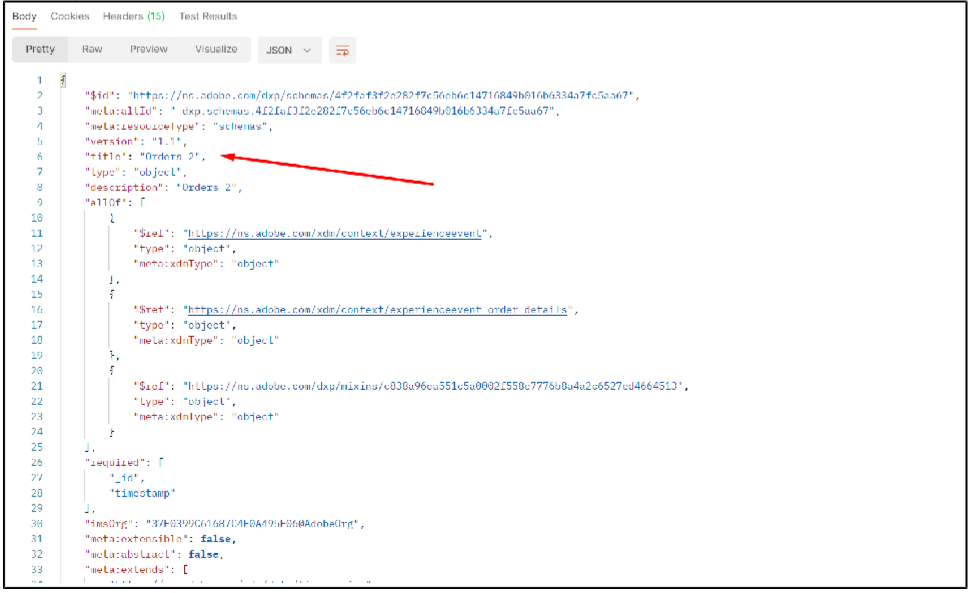
```
GET  https://platform.adobe.io/data/foundation/schemaregistry/tenant//schemas/_dxp.schemas.4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67
```

	KEY	VALUE
□	Accept	application/json
☑	Content-Type	application/json
☑	Authorization	Bearer {{ACCESS_TOKEN}}
☑	x-api-key	{{API_KEY}}
☑	x-gw-ims-org-id	{{IMS_ORG}}
☑	x-sandbox-name	{{SANDBOX_NAME}}
☑	Accept	application/vnd.adobe.xed-desc+json;version=1
	Key	Value

This Accept header returns Schema as wel lits corresponding descriptors.

3.

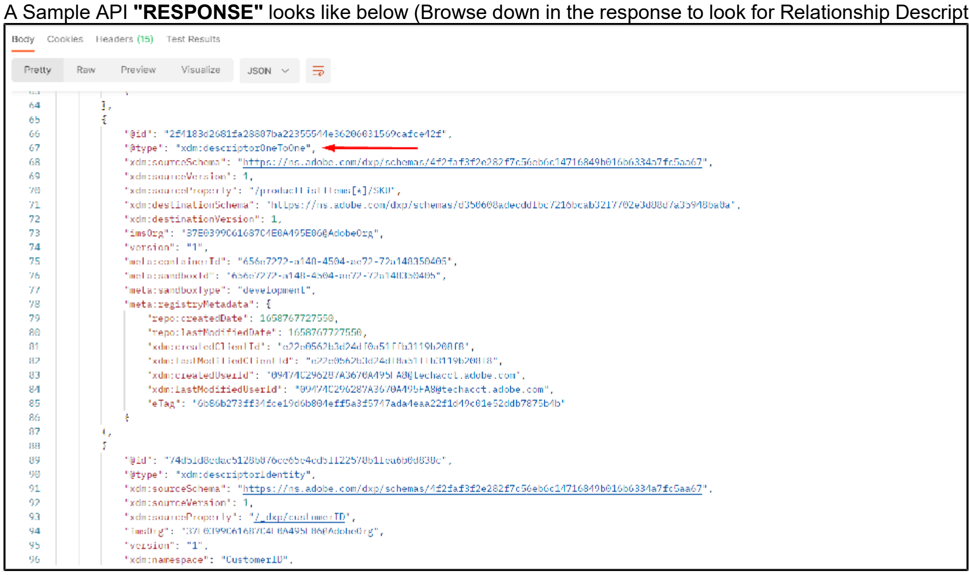
4. A Sample API **"RESPONSE"** looks like below:



```
{
    "$id": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
    "meta:altId": "_dxp.schemas.4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
    "meta:resourceType": "schemas",
    "version": "1.1",
    "title": "Orders 2",
    "type": "object",
    "description": "Orders 2",
    "allOf": [
        {
            "$ref": "https://ns.adobe.com/xdm/context/experienceevent",
            "type": "object",
            "meta:xdmType": "object"
        },
        {
            "$ref": "https://ns.adobe.com/xdm/context/experienceevent_order_details",
            "type": "object",
            "meta:xdmType": "object"
        },
        {
            "$ref": "https://ns.adobe.com/dxp/mixins/c030a96ea551c5a0002f550e7776b0a4a2c6527cd4664513",
            "type": "object",
            "meta:xdmType": "object"
        }
    ],
    "required": [
        "_id",
        "timestamp"
    ],
    "imsOrg": "37E0399C61687C4E0A495E86@AdobeOrg",
    "meta:extensible": false,
    "meta:abstract": false,
    "meta:extends": [
```

5.

6. A Sample API **"RESPONSE"** looks like below (Browse down in the response to look for Relationship Descriptors for this Schema):

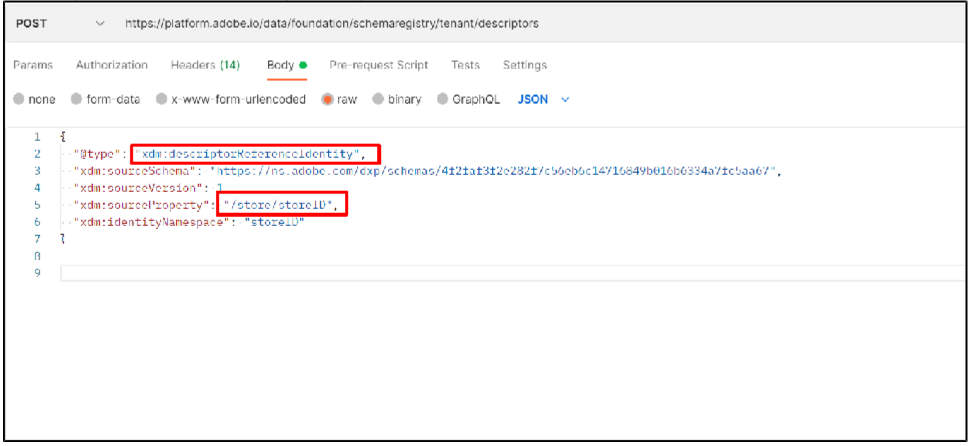


```
        },
        {
            "@id": "2f4183d2681fa28807ba2235554e36206031569cafce42f",
            "@type": "xdm:descriptorOneToOne",
            "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
            "xdm:sourceVersion": 1,
            "xdm:sourceProperty": "/productListItems[*]/SKU",
            "xdm:destinationSchema": "https://ns.adobe.com/dxp/schemas/d350608adecdd1bc7216bcab3217702e3d88d7a35948ba0a",
            "xdm:destinationVersion": 1,
            "imsOrg": "37E0399C61687C4E0A495E86@AdobeOrg",
            "version": "1",
            "meta:containerId": "656e7272-a140-4504-ae72-72a140350405",
            "meta:sandboxId": "656e7272-a140-4504-ae72-72a140350405",
            "meta:sandboxType": "development",
            "meta:registryMetadata": {
                "repo:createdDate": 1658767727550,
                "repo:lastModifiedDate": 1658767727550,
                "xdm:createdClientId": "e22e0562b3d24df0a51ffb3119b200f8",
                "xdm:lastModifiedClientId": "e22e0562b3d24df0a51ffb3119b200f8",
                "xdm:createdUserId": "09474C296287A3670A495FA8@techacct.adobe.com",
                "xdm:lastModifiedUserId": "09474C296287A3670A495FA8@techacct.adobe.com",
                "eTag": "6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b"
            }
        },
        {
            "@id": "74d51d8edac5128b876ce65e4cdb1122578b11ea6b0d830c",
            "@type": "xdm:descriptorIdentity",
            "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
            "xdm:sourceVersion": 1,
            "xdm:sourceProperty": "/_dxp/customerID",
            "imsOrg": "37E0399C61687C4E0A495E86@AdobeOrg",
            "version": "1",
            "xdm:namespace": "CustomerID",
```

7.

4.16. Create "Store" Reference Identity Descriptor for "Orders" Schema

1. **Click on "Step 12 - Reference Descriptor for Store"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
    a. You would have identified the **"relationships"** from **"Orders"** Schema in the **SID lab.**
    b. Reference Identity descriptors are created automatically from the backend when you create relationships from Schema UI. You have to create them explicitly when you create schema using API calls.
    c. A reference descriptor is required to be defined for the **"Source Property"** in each **"Relationship descriptor"**.
    d. Descriptor Properties:
        i. Type - Descriptor type. Its **"xdm:descriptorReferenceIdentity "** in this case
        ii. Source Schema - The **"$id"** of the schema, the relationship is originating from (**"Orders"** Schema in this case).
        iii. Source Property - Full **"path"** for the Schema property which needs to be marked as a reference identity(**"/store/storeID"** in this case).
        iv. Source Version - Version number of the Source schema. Always defined as **1**
        v. Identity Namespace- The namespace for the Source Schema Property (**"storeID"** in this case)
3. A Sample **"POST"** Relationship Descriptor call looks like below.

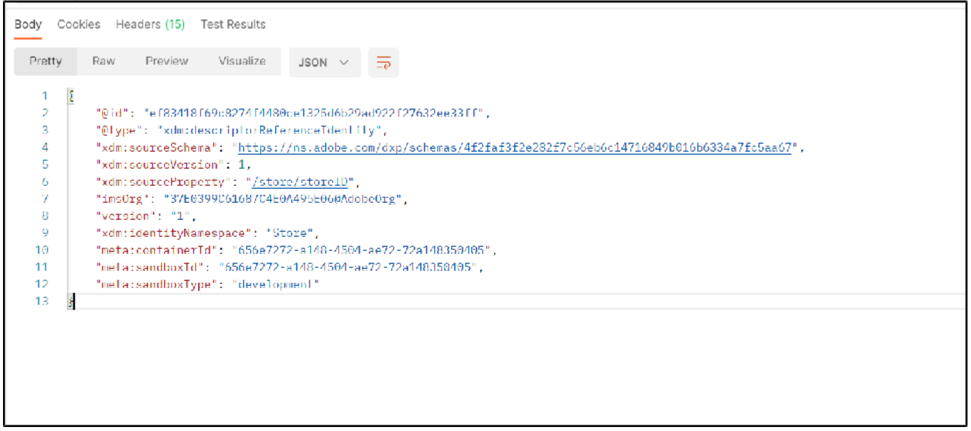


POST https://platform.adobe.io/data/foundation/schemaregistry/tenant/descriptors

Params
Authorization
Headers (14)
Body
Pre-request Script
Tests
Settings
○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
  "@type": "xdm:descriptorReferenceIdentity",
  "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
  "xdm:sourceVersion": 1
  "xdm:sourceProperty": "/store/storeID",
  "xdm:identityNamespace": "storeID"
}
```

4.
5. A Sample API **"RESPONSE"** looks like below:



```
{
    "@id": "ef83418f69c8274f4480ce1325d6b29ad922f27632ee33ff",
    "@type": "xdm:descriptorReferenceIdentity",
    "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/4f2faf3f2e282f7c56eb6c14716849b016b6334a7fc5aa67",
    "xdm:sourceVersion": 1,
    "xdm:sourceProperty": "/store/storeID",
    "imsOrg": "37E0399C61687C4E0A495E060@AdobeOrg",
    "version": "1",
    "xdm:identityNamespace": "Store",
    "meta:containerId": "656e7272-a148-4504-ae72-72a148350405",
    "meta:sandboxId": "656e7272-a148-4504-ae72-72a148350405",
    "meta:sandboxType": "development"
}
```

6.
7. **Execute the "Step 12 - Reference Descriptor for Store" by clicking the "Send" button**

4.17. Create "Product" Reference Identity Descriptor for "Orders" Schema

1. **Please note the following points:**
    a. Descriptor Properties:
        i. Type - Descriptor type. Its **"xdm:descriptorReferenceIdentity "** in this case
        ii. Source Schema - The **"$id"** of the schema, the relationship is originating from (**"Orders"** Schema in this case).
        iii. Source Property - Full **"path"** for the Schema property which needs to be marked as a reference identity(**"/productListItems[*]/SKU"** in this case).
        iv. Source Version - Version number of the Source schema. Always defined as **1**
        v. Identity Namespace- The namespace for the Source Schema Property (**"productID"** in this case)
2. **Execute the "Step 13 - Reference Descriptor for Product" by clicking the "Send" button**

4.18. Create "Plan" Reference Identity Descriptor for "Orders" Schema

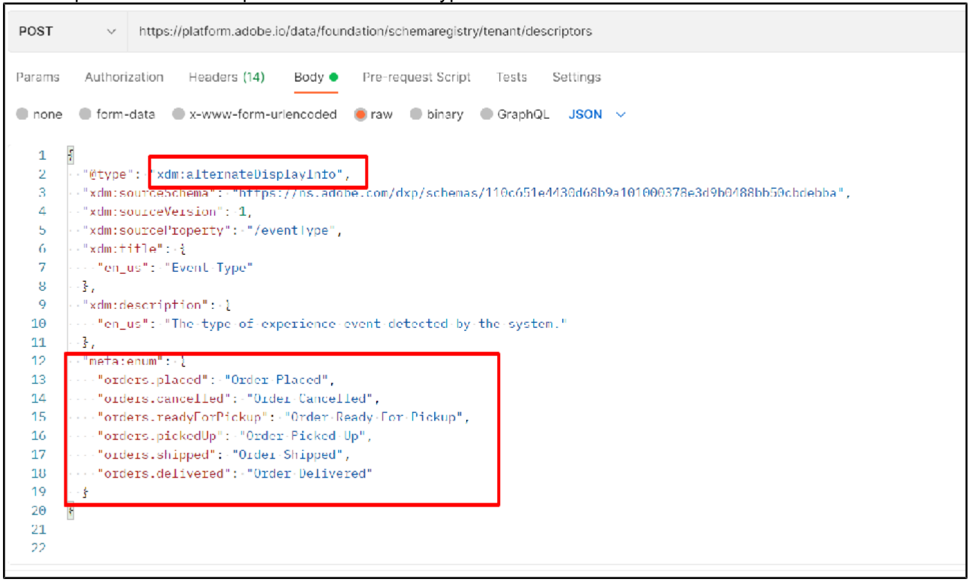
1. **Please note the following points:**
    a. Descriptor Properties:
        i. Type - Descriptor type. Its **"xdm:descriptorReferenceIdentity "** in this case
        ii. Source Schema - The **"$id"** of the schema, the relationship is originating from (**"Orders"** Schema in this case).
        iii. Source Property - Full **"path"** for the Schema property which needs to be marked as a reference identity(**"/order/_dxp/plan/planID"** in this case).
        iv. Source Version - Version number of the Source schema. Always defined as **1**
        v. Identity Namespace- The namespace for the Source Schema Property (**"planID"** in this case)
2. **Execute the "Step 14 - Reference Descriptor for Plan" by clicking the "Send" button**

4.19. **Browse the "Orders" Schema to see your Reference Descriptors.**

1. **Execute the "Step 15 - Get Order Schema and its descriptors" by clicking the "Send" button**
2. **You should be able to browse the following Schema descriptors:**
   - **a.** Identities (Primary/Secondary)
   - **b.** Relationships
   - **c.** Reference Identities

4.20. **Extend Event Types for Order Schema**

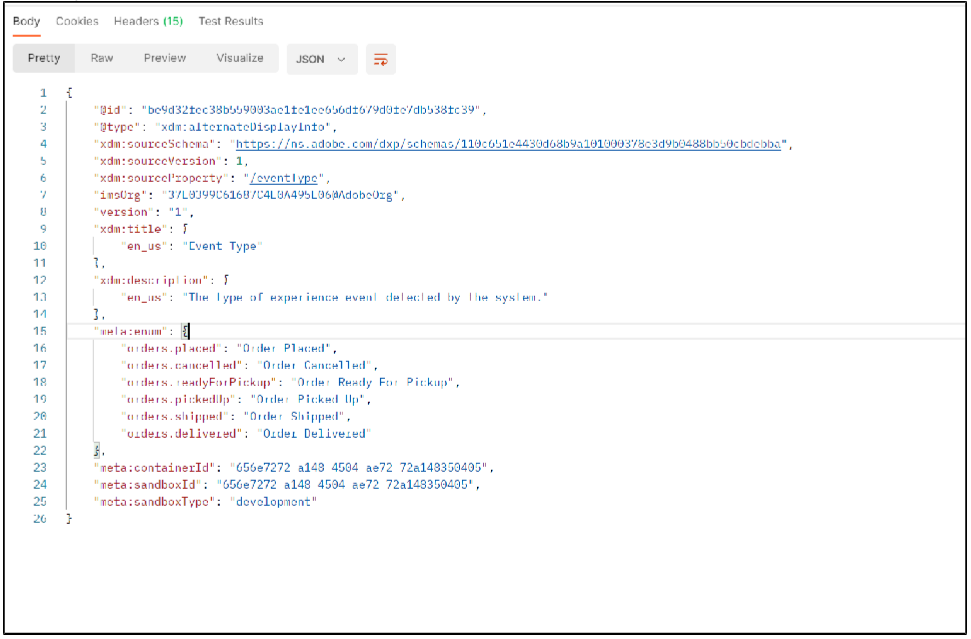
1. **Click on "Step 16 - Extend the Event Types"** API call in the **"DEP XDM API Lab"** Folder. **Do not execute it yet.**
2. **Please note the following points:**
   - **a.** Standard XDM Event Types are provided by the Experience Event Class
   - **b.** If the customers need to add more event types as per their use cases, they can do that using the API calls (UI enablement is in progress to do the same).
   - **c.** **"Orders"** data will be sending the following event types to the platform:
     - **i.** Order Placed
     - **ii.** Order Cancelled
     - **iii.** Order Ready For Pickup
     - **iv.** Order Picked Up
     - **v.** Order Shipped
     - **vi.** Order Delivered
3. A Sample **"POST"** Descriptor call to add event types looks like below.



POST https://platform.adobe.io/data/foundation/schemaregistry/tenant/descriptors

```
{
  "@type": "xdm:alternateDisplayInfo",
  "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/110c651e4430d68b9a101000378e3d9b0488bb50cbdebba",
  "xdm:sourceVersion": 1,
  "xdm:sourceProperty": "/eventType",
  "xdm:title": {
    "en_us": "Event Type"
  },
  "xdm:description": {
    "en_us": "The type of experience event detected by the system."
  },
  "meta:enum": {
    "orders.placed": "Order Placed",
    "orders.cancelled": "Order Cancelled",
    "orders.readyForPickup": "Order Ready For Pickup",
    "orders.pickedUp": "Order Picked Up",
    "orders.shipped": "Order Shipped",
    "orders.delivered": "Order Delivered"
  }
}
```

4.
5. A Sample API **"RESPONSE"** looks like below:



```
{
    "@id": "be9d32fec38b559003ae1fe1ee656df679d0fe7db538fc39",
    "@type": "xdm:alternateDisplayInfo",
    "xdm:sourceSchema": "https://ns.adobe.com/dxp/schemas/110c651e4430d68b9a101000378e3d9b0488bb50cbdebba",
    "xdm:sourceVersion": 1,
    "xdm:sourceProperty": "/eventType",
    "imsOrg": "37L0399C61687C4L0A495L06@AdobeOrg",
    "version": "1",
    "xdm:title": {
        "en_us": "Event Type"
    },
    "xdm:description": {
        "en_us": "The type of experience event detected by the system."
    },
    "meta:enum": {
        "orders.placed": "Order Placed",
        "orders.cancelled": "Order Cancelled",
        "orders.readyForPickup": "Order Ready For Pickup",
        "orders.pickedUp": "Order Picked Up",
        "orders.shipped": "Order Shipped",
        "orders.delivered": "Order Delivered"
    },
    "meta:containerId": "656e7272 a148 4504 ae72 72a148350405",
    "meta:sandboxId": "656e7272 a148 4504 ae72 72a148350405",
    "meta:sandboxType": "development"
}
```

6.

7. **Execute the "Step 16 - Extend the Event Types" by clicking the "Send" button**