



Zurück zur Übersichtsseite

How to invoke a GraphQL API with Oracle APEX

Nina Brötje APEX / Oracle, Datenbank 16.05.2024 0 Kommentare

Technical information

- Oracle APEX Version 23.2.4
- Pokemon API on <https://stepzen.com/blog/free-public-graphql-apis>

Introduction

GraphQL is a query language and runtime environment for APIs. Unlike REST, GraphQL allows clients to request only the specific data they need, avoiding overfetching and underfetching, thus improving data transfer efficiency. With its type system and query structures, GraphQL provides a clear way to exchange data between client and server.

Many people already know that a consumption of a GraphQL API is possible via the **apex_web_service** package in PL/SQL. I will show you how to do this declaratively in Oracle APEX.

Easy way with PL/SQL

With PL/SQL it is very easy to consume a GraphQL API using the function **apex_web_service.make_rest_request**. You can call this function with the GraphQL endpoint, specifying the GraphQL query or mutation in the request body. You then handle the response returned by the API, within your APEX application, parsing and processing the data as needed. This method allows you to interact with GraphQL APIs from within your APEX application's PL/SQL code.

```

1  declare
2      l_query      clob;
3      l_url        varchar2(50)    := 'https://beta.pokeapi.co/graphql/v1beta';
4      l_body        clob          := '{ "query": "{ pokémon_v2_pokémon(limit: 10) { height id name order pokémon_sp" } }';
5      l_response   clob;
6      l_status_code number;
7  begin
8
9      apex_web_service.g_request_headers(1).name :=  'Content-Type';
10     apex_web_service.g_request_headers(1).value :=  'application/json';
11
12     l_response := apex_web_service.make_rest_request(
13         p_url          => l_url,
14         p_http_method  => 'POST',
15         p_body          => l_body
16     );
17
18     l_status_code := apex_web_service.g_status_code;
19
20     if l_status_code = 200 then
21         apex_collection.create_or_truncate_collection(
22             p_collection_name => 'POKEMON_COLLECTION'
23         );
24
25         apex_collection.add_member(
26             p_collection_name => 'POKEMON_COLLECTION',
27             p_clob001           => l_response
28         );
29     else
30         -- e.g. error handling
31         null;
32     end if;
33
34 end;

```

```

37  select
38      jt.id,
39      jt.name,
40      jt.height,
41      jt."order",
42      jt.species_id
43  from
44      apex_collections c
45      cross join json_table(
46          c.clob001,
47          '$.data.pokémon_v2_pokémon[*]'
48          columns(
49              id      number      path "id",
50              name    varchar2(500)  path "name",
51              height   number      path height,
52              "order"  number      path "order",
53              species_id number      path pokémon_species_id
54          )
55      ) as jt
56  where
57      collection_name = 'POKEMON_COLLECTION';

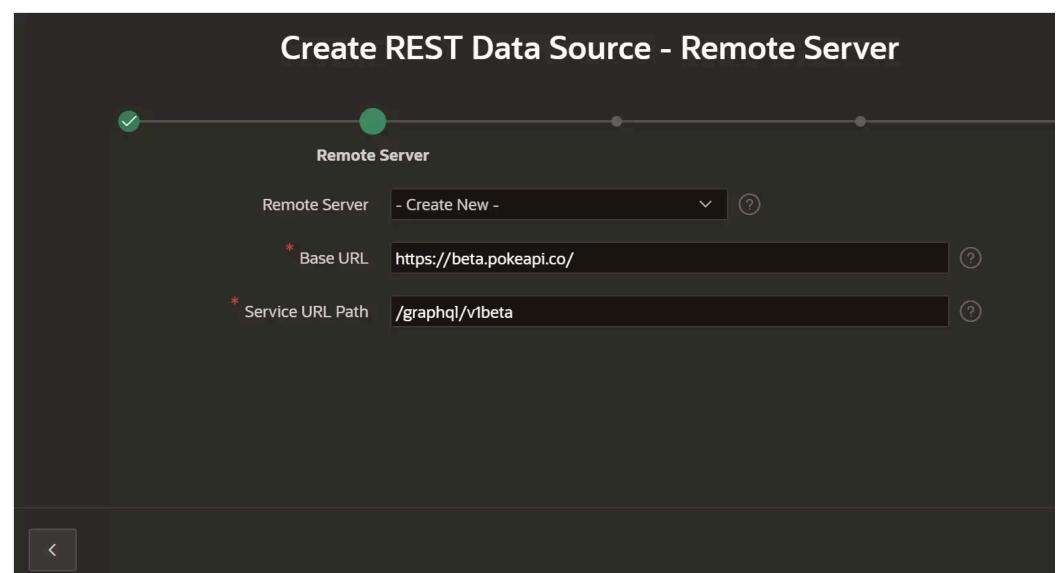
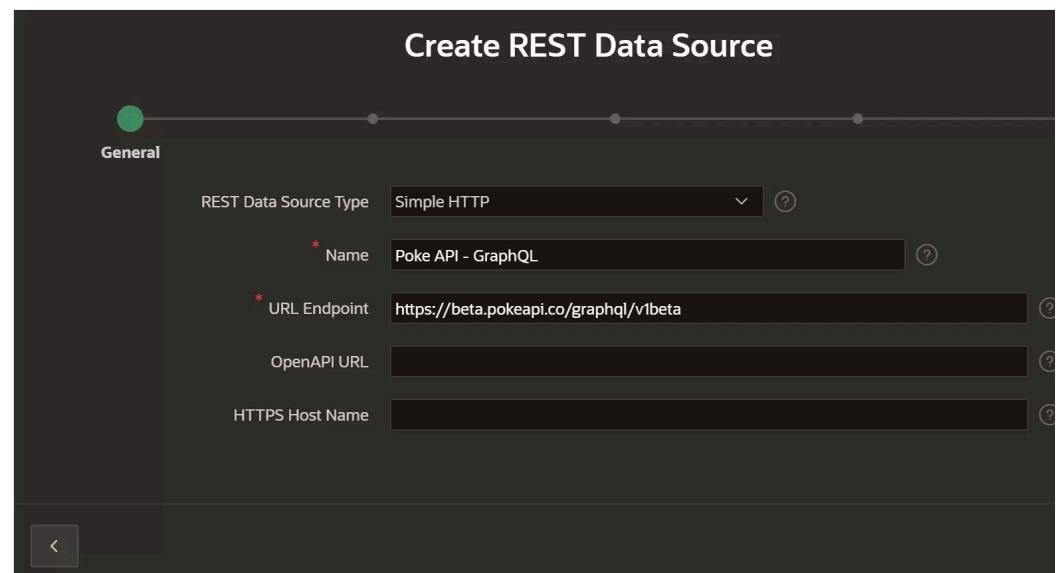
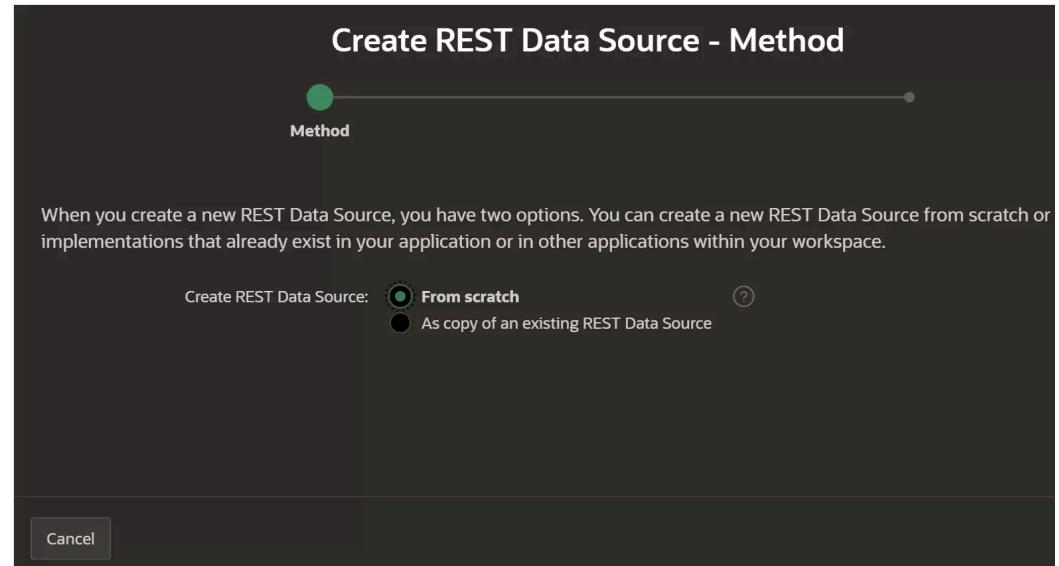
```

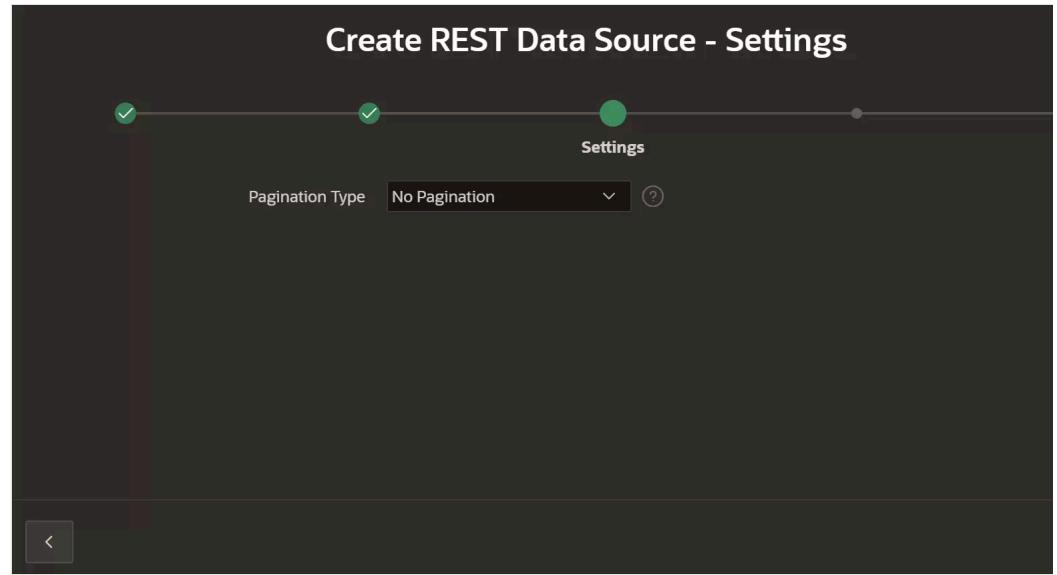
But can we not also do this declaratively in APEX? This has not yet been documented so far, but there is a very simple trick th.

Do it declaratively in the APEX backend

To be able to use a GraphQL API in Oracle APEX as a REST data source, we have to trick it a little bit.

The first steps are analogue to a simple REST API.



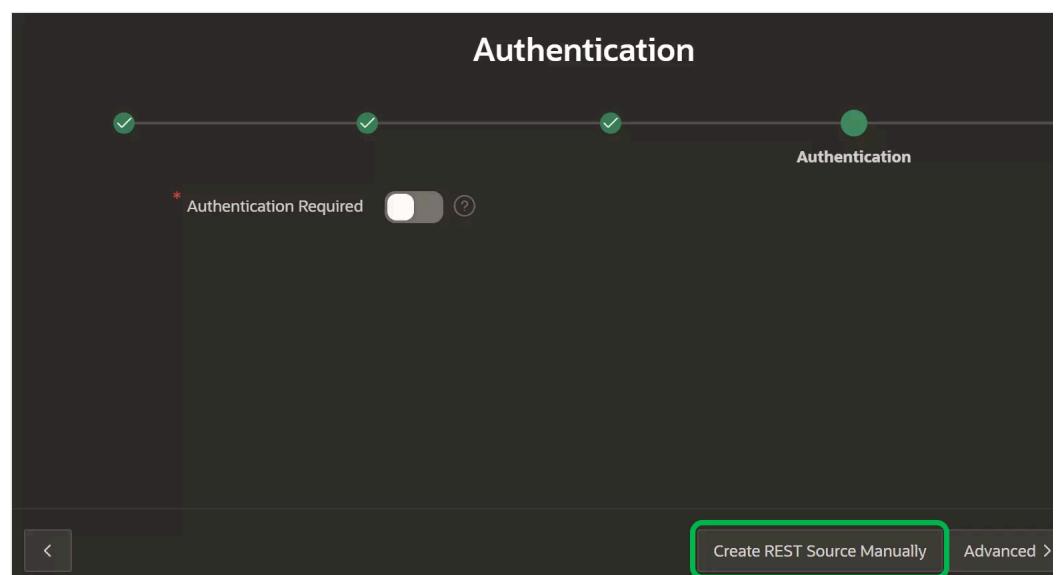


On the *Authentication* page of the wizard, it is now important **not** to click on *Discover*.

In this case, our new "REST Data Source" would be tested before it is created, which can lead to various errors ("GET query mi Unknown", error of the API itself, etc.).

There are several reasons for this: A GraphQL API is always addressed with a POST method, whereby the request body contains query. This is also used to fetch rows. Such a POST method can be created under *Advanced*. However, this is defined by default "fetch rows". APEX creates a GET for the "fetch rows" by default, but this is not supported by the GraphQL API.

We click on *Create REST Source Manually* instead to prevent this.



Now our GraphQL Data Source has been created. But we need to make a few more changes.

REST Data Sources					Synchronizations	Utilization	History
Actions		REST Source Name	Synchronized	Operations	Endpoint URL	Authentication	
Poke API - GraphQL			No	4	https://beta.pokeapi.co/graphql/v1beta	No	

As we have created our GraphQL Data Source manually, all four HTTP methods are created automatically. We do not need DE to consume a GraphQL API. These are dropped. However, our POST method must be modified.

Edit	Name	Operation	Database Action	URL Pattern	Parameters	Test
	-	DELETE	Delete row	/:id	-	
	-	GET	Fetch rows	.	-	
	-	POST	Insert row	.	-	
	-	PUT	Update row	/:id	-	

To do this, it is important to know what our GraphQL query and its JSON response look like (see the following illustration).

```
1 * {
2 * pokemon_v2_pokemon(limit: 10) {
3   height
4   id
5   name
6   order
7   pokemon_species_id
8 }
9 }
```

```
{
  "data": {
    "pokemon_v2_pokemon": [
      {
        "height": 7,
        "id": 1,
        "name": "bulbasaur",
        "order": 1,
        "pokemon_species_id": 1
      },
      {
        "height": 10,
```

In our case, we want to use the POST method to fetch data. So, we set the database operation to "fetch rows".

In the request body template, we insert our JSON body that is to be sent to the API. Attention: Paragraphs can lead to errors here.

REST Source Operation

Show All Operation Operation Parameters Caching Advanced

URL Pattern .

REST Source Base URL <https://beta.pokeapi.co/graphql/v1beta>

* HTTP Method POST

Database Operation Fetch rows

Request Body Template {"query": "{ pokemon_v2_pokemon(limit: 10) { height id name order pokemon_spes..."}

Next, we modify the data profile.

REST Data Source

Show All REST Data Source Settings Authentication **Data Profile** Operations Parameters Sub...

Data Profile

JSON	Table	5	4
Response Format	Returns	Columns	Visible

The row selector must be adapted according to the expected JSON response. The columns created by APEX by default are the columns are added.

Data Profile

Data Profiles describe how Data Source format are being parsed and converted to rows and columns. For XML or JSON data format, the **Row Selector** attribute stores an XML or JSON path expression pointing to the node containing Profile Columns determine how one row is parsed and converted to multiple columns.

Data Profile

- Name: Poke API - GraphQL
- Format: JSON
- Row Selector: `data.pokemon_v2_pokemon`
- Contains Single Row:

Columns

	Sequence ↑↓	Name	Column Type	Data Type	Selector	Primary Key	Visible
1	1	ID	Data	NUMBER	id	Yes	No
2	2	name	Data	VARCHAR2	name	No	Yes
3	3	height	Data	NUMBER	height	No	Yes
4	4	order	Data	NUMBER	order	No	Yes
5	5	species_ID	Data	NUMBER	pokemon_species_id	No	Yes

Now let's test it.

Test Results

Parsed Data Response Headers Response Body

Id	Name	Height	Order
1	bulbasaur	7	1
2	ivysaur	10	2
3	venusaur	20	3
4	charmander	6	5
5	charmeleon	11	6
6	charizard	17	7
7	squirtle	5	10
8	wartortle	10	11
9	blastoise	16	12
10	caterpie	3	14

Yay! Successful!

Now we can use this new GraphQL Data Source using its POST method for our APEX components.

Utilise the dynamic magic of GraphQL

We are using GraphQL, which actually stands for dynamism. So, the example shown above is really for showing that the connection is declarative. But can we also make the whole thing dynamic? Of course we can!

To do this, we change our POST method once again

Set the *Request Body Template* to a substitution string, for example `#body#` and let it map to an operation parameter.

REST Source Operation

Show All Operation Operation Parameters Caching Advanced

REST Source Base URL: <https://beta.pokeapi.co/graphql/v1beta>

* HTTP Method: POST

Database Operation: Fetch rows

Request Body Template: #body#

6 of 32760

Operation Parameters

Name	Type	Direction	Default Value	Required	Status
body	Request or Response Body	In	{"query": "{ pokemon_v2_pokemon(limit: 10) { height id name order pokemon_species_id } }"}	No	No

The corresponding parameter can then be used in the Page Designer and be set dynamically. It is important that the corresponding JSON object that contains the correct GraphQL query.

The screenshot shows the Oracle APEX component configuration interface. On the left, the component tree is visible with nodes like 'Components', 'Breadcrumb Bar', 'Body', 'Pokémon GraphQL', 'Columns', 'Pokémon GraphQL - Dynamic Query', 'Columns', and 'Parameters'. The 'Parameters' node is expanded, showing a parameter named 'body' with a value type of 'P99_JSON'. On the right, the 'Identification' and 'Operation' sections are displayed. The 'Identification' section includes 'Name' (body) and 'Operation' (HTTP Method: POST, Database Operation: Fetch Rows). The 'Parameters' section shows 'Direction' (In) and 'Data Type' (String). A context menu is open over the 'Value' field in the 'Parameters' section, listing options such as 'REST Source Default', 'Static Value', 'Audit Information', 'Changed By', 'Changed On', 'SQL Query (return single v... Expression)', 'Function Body', 'Preference', and 'Null'. The 'SQL Query (return single v... Expression)' option is highlighted.

And now we can use our GraphQL data source within our components.

The screenshot shows the 'TRIOLOGY Sample REST Services' application. The main page title is 'Pokémon GraphQL'. Below it is a table displaying data from a GraphQL query. The table has columns: 'Pokémon' (sorted by name), 'Height', 'Order', and 'Species'. The data rows are:

Pokémon	Height	Order	Species
blastoise	16	12	
bulbasaur	7	1	
caterpie	3	14	
charizard	17	7	
charmander	6	5	
charmeleon	11	6	
ivysaur	10	2	
squirtle	5	10	
venusaur	20	3	
wartortle	10	11	

At the bottom of the page, there is a footer note: '23.2.0 Built with ❤️ using Oracle APEX by TRIOLOGY GmbH'.

Implement a GraphQL API with ORDS 23.3

As a small hint: Since ORDS 23.3 it is now also possible to implement GraphQL APIs with Oracle Database. Jeff Smith has written this: <https://www.thatjeffsmith.com/archive/2023/10/ords-23-3-is-now-available-or-hello-graphql/>



Über den Autor

Nina Brötje

Kommentare

Keine Kommentare

Kommentar schreiben

Name *

E-Mail *

Kommentar *

UeBZGXs

Captcha*

Prüfen

* Diese Felder sind erforderlich

Absenden

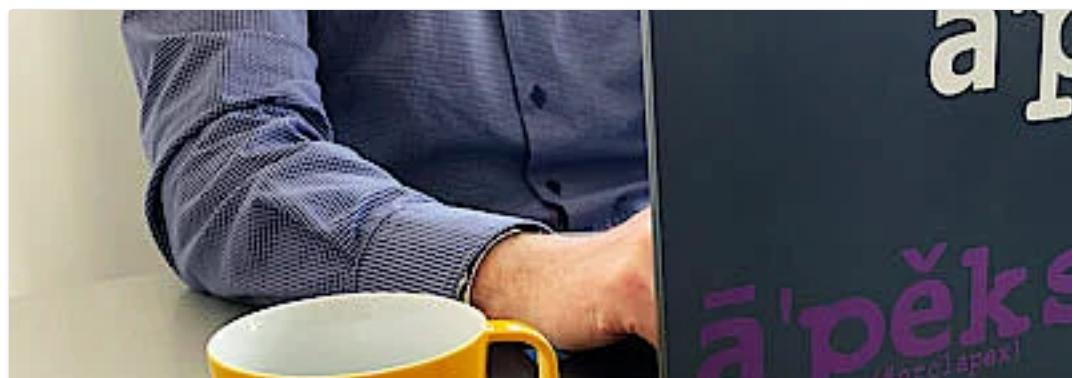
Ähnliche Beiträge



How to import an Oracle APEX Application export file from a higher lower version

Maik Becker, Swathi Mary Ambati 08.03.2019

Recently we developed Oracle APEX application in our APEX 18.1 environment. Trying to share the tool with one of our customers they are still using 5.1.4 and won't be able to update within short time.



Mehrsprachige Datenbankanwendungen mit Oracle APEX

Swathi Mary Ambati 06.08.2018

Hallo Zusammen! Wenn du ein DBA/PLSQL-Guru bist, aber nicht viel Erfahrung in der Webentwicklung hast, dann ist Oracle APEX die beste Option. Egal ob Anfänger oder Fortgeschritten, dieser Artikel gibt dir einen Einblick, wie eine mehrsprachige Anwendung erstellt werden kann. Hier zeigen wir dir, wie der in Oracle APEX integrierte Übersetzungsmechanismus funktioniert und wie es geht.



Namenskonventionen in der Oracle Datenbank

Jan Niemann 31.07.2017

Ein wichtiger Teil von Code Conventions sind die Namenskonventionen. In den Namenskonventionen wird vereinbart, wie im Falle einer Datenbank handelt es sich bei den zu benennenden „Dingen“ um Schema-Objekte wie Tabellen, Sequenzen



Eine Chance für Azubis und Studis – die DOAG NextGen

Laura Fleischer 05.04.2023

Azubis! Studis! 2022 habe ich an der APEX Connect im Phantasialand teilgenommen und es war eine großartige Erfahrung. Programm der Doag könnt ihr das auch!



Faceted Search with Kanban Board - #JoelKallmanDay

Maik Becker [11.10.2022](#)

I decided to write this blogpost to give a better insight and explain, how easy it is to implement Faceted Search with different example with the Kanban Board.

Kategorien

APEX / Oracle (9)

Code Review (2)

Datenbank (7)

Java (12)

Open Source (5)

Projektmanagement (7)

Quality (14)

Security (6)

Unternehmen & Personal (18)

Web (8)

Letzte Blogbeiträge

[How to invoke a GraphQL API with Oracle APEX](#)

[OWASP Juice Shop](#)

[Eine Chance für Azubis und Studis – die DOAG NextGen](#)

[Faceted Search with Kanban Board - #JoelKallmanDay](#)



TRIOLOGY GmbH



Brabandtstraße 9-10
38100 Braunschweig
Deutschland



+49 531 23528-0



+49 531 23528-19



info@triology.de

[Leistungen](#)

[Referenzen](#)

[Karriere](#)

[Unternehmen](#)

[Blog](#)

[Kontakt](#)



[Impressum](#)

[Datenschutz](#)

[Hinweisgebersystem](#)

