

SYSTÈME DE GESTION DE BASES DE DONNÉES

ORDS – Oracle REST Data Services
Encodage BASE64

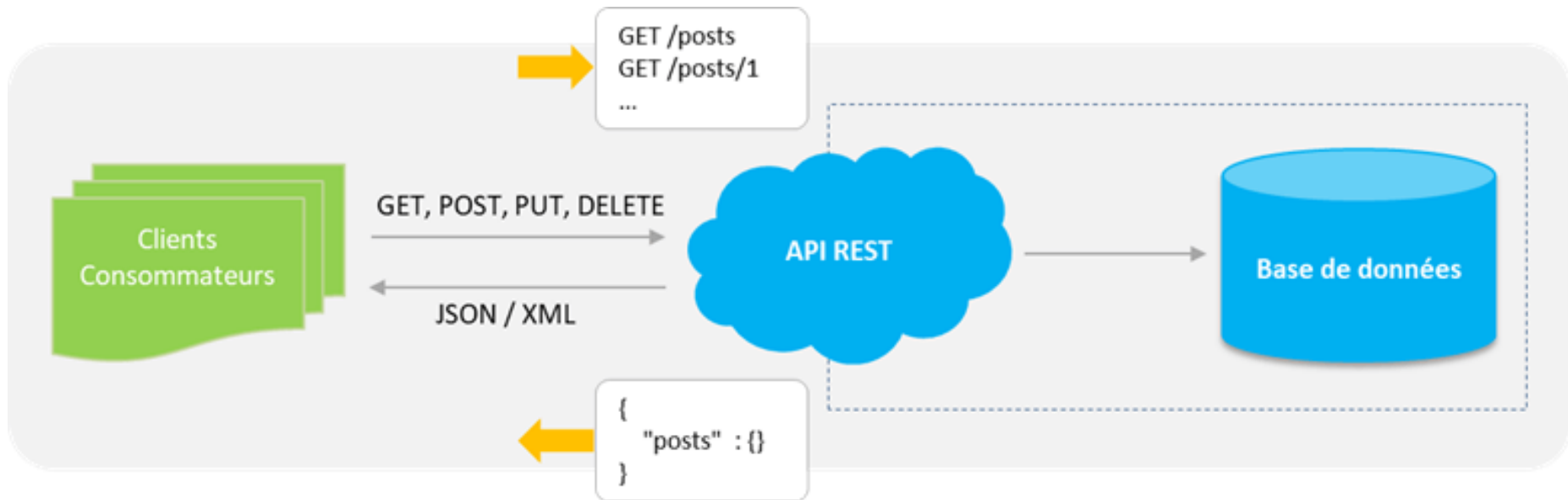
3^{ème} Informatique de gestion
2021-2022
Samuel Hiard

Agenda

- ORDS
 - REST
 - JSON
 - Auto-Rest
 - Handers REST manuels (GET et POST)
- PostMan
- Base 64

- ORDS
 - Oracle REST Data Services
- REST?
 - REST = **RE**presentational **S**tate **T**ransfer
- Plutôt une norme qu'un protocole
 - Définit les lignes directrices architecturales
- Utilise le protocole HTTP pour communiquer
 - GET, POST, PUT, HEAD, ...
- Lorsqu'un service propose une API REST, on dit que le service est RESTful.

REST (2)



img src : <https://blog.nicolashachet.com/wp-content/uploads/2012/06/api-rest-architecture.png>

Avantages/Inconvénients

■ Avantages

- Séparation claire entre le client et le serveur
 - Scalable : très facile d'ajouter un client
- Stateless
 - Pas besoin de contexte pour comprendre une requête
- Cacheable
 - En tout cas pour les méthodes GET → gain de temps et de réseau
- Interface uniforme
 - Protocole clair permettant la séparation facile des work units (front-end/back-end)
- Encapsulation
 - Pas besoin de savoir ce qu'il se trouve derrière (DB, serveur, NAS, ...)

Avantages/Inconvénients (2)

■ Inconvénients

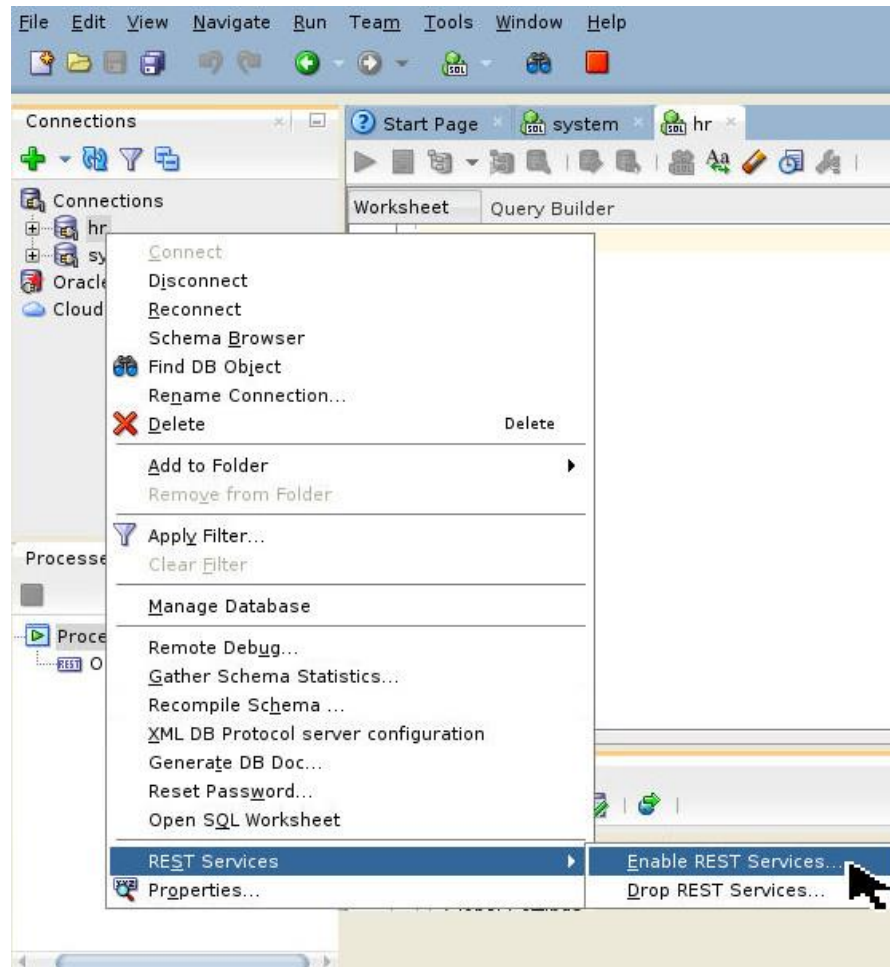
- Stateless
 - Il faut préciser le contexte à chaque fois, donc plus verbeux
- Génère un peu plus d'overhead que la connexion directe
 - Donc, un peu plus lent

Installation ORDS

- Télécharger un fichier ords.war
 - Lancer l'installation en ligne de commande
 - Java -jar ords.war
- Vidéo explicative :
 - <https://www.youtube.com/watch?v=1LBmLTe8sIc>
 - Procédure d'installation à 9 min 20
 - Pour le sid, utilisez « xepdb1 » si vous êtes sur Oracle Express

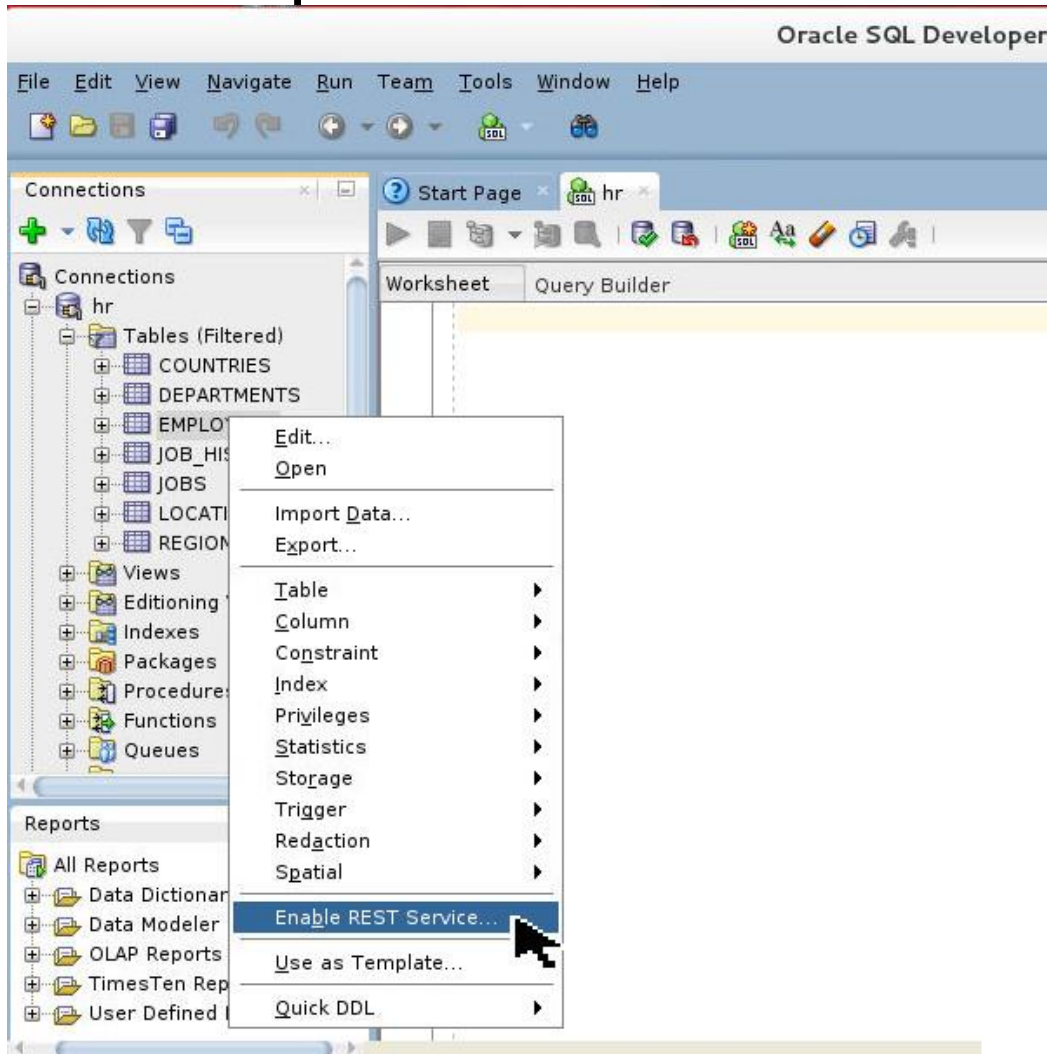
Activer les services REST

- Chaque utilisateur doit activer son service
 - Après installation



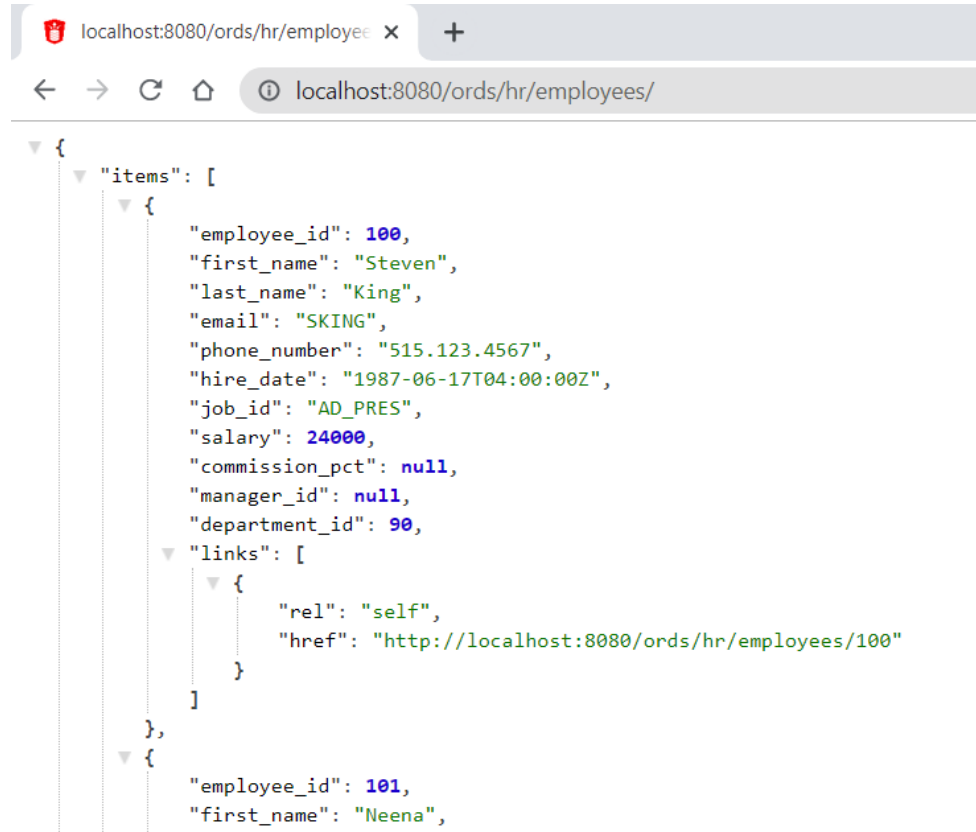
ORDS Automatique

■ Le plus simple



Testons le service

■ Ca marche!



The screenshot shows a web browser window with the address bar displaying `localhost:8080/ords/hr/employees/`. The main content area displays a JSON response from a REST API. The JSON is a list of employee objects. The first object is for Steven King, and the second object is for Neena. The JSON structure is as follows:

```
{
  "items": [
    {
      "employee_id": 100,
      "first_name": "Steven",
      "last_name": "King",
      "email": "SKING",
      "phone_number": "515.123.4567",
      "hire_date": "1987-06-17T04:00:00Z",
      "job_id": "AD_PRES",
      "salary": 24000,
      "commission_pct": null,
      "manager_id": null,
      "department_id": 90,
      "links": [
        {
          "rel": "self",
          "href": "http://localhost:8080/ords/hr/employees/100"
        }
      ]
    },
    {
      "employee_id": 101,
      "first_name": "Neena",

```

■ Note : Le résultat est sous format JSON

JSON (en 1 transparent)

- JSON = **J**ava**S**cript **O**bject **N**otation
 - Représentation d'un objet (javascript)

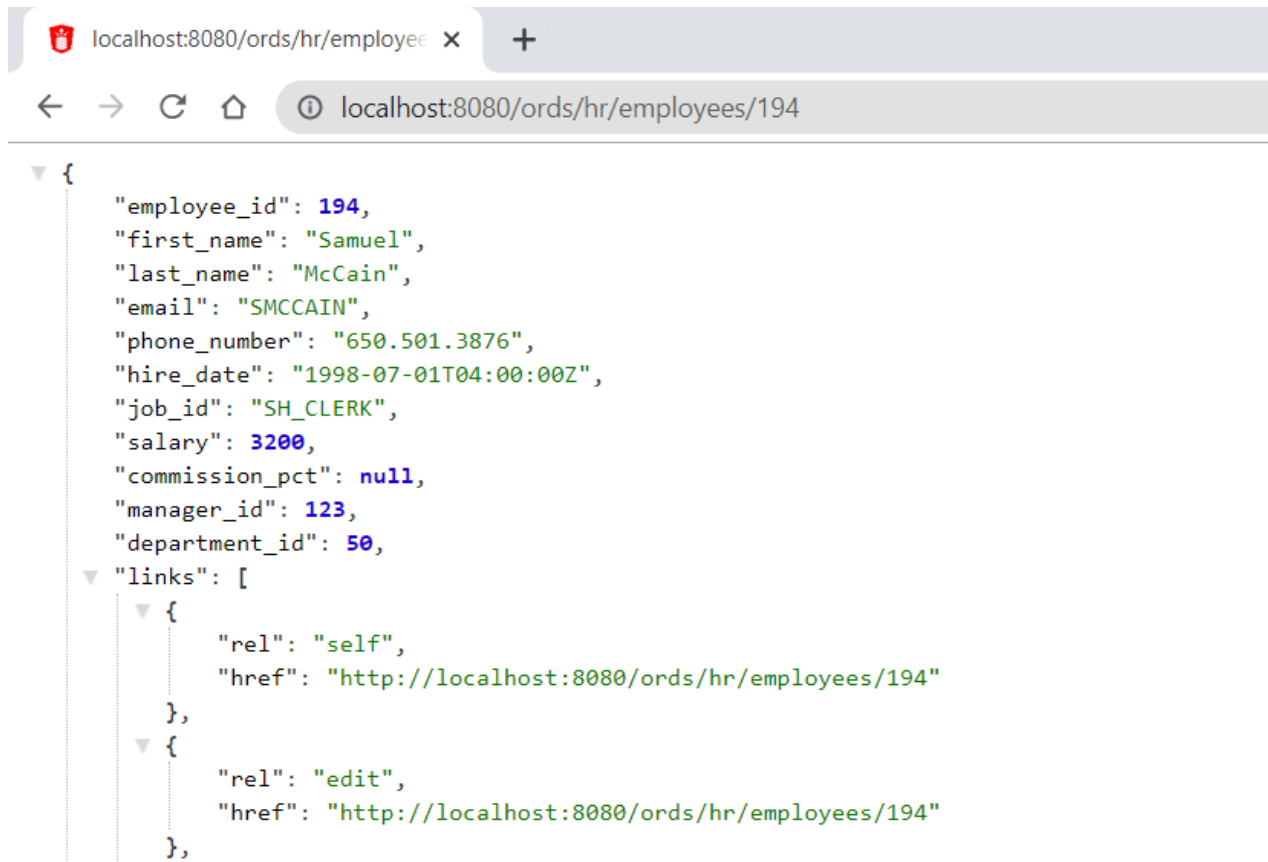
- Syntaxe:
 - Données en paires nom/valeurs
 - « nom »: « valeur »
 - Une virgule pour séparer des données
 - Les accolades contiennent des objets
 - Les crochets contiennent des tableaux

Exemple de JSON

```
{  
  "name": "John",  
  "age": 30,  
  "parent": {  
    "name": "Carl",  
    "age": 56  
  },  
  "cars": ["Ford", "BMW", "Fiat"]  
}
```

Filtrer par clé primaire

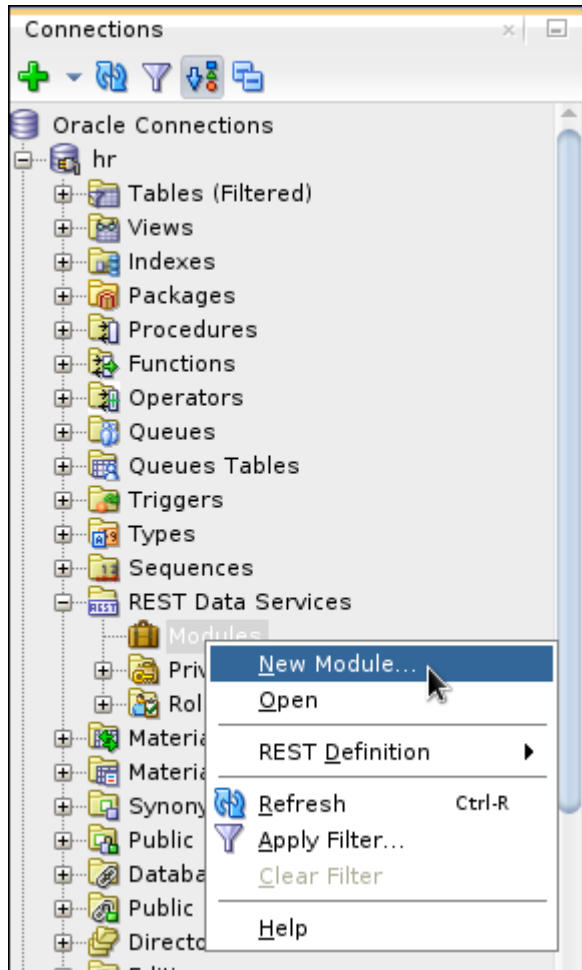
- Il suffit de rajouter la valeur de la clé dans l'adresse



```
{
  "employee_id": 194,
  "first_name": "Samuel",
  "last_name": "McCain",
  "email": "SMCCAIN",
  "phone_number": "650.501.3876",
  "hire_date": "1998-07-01T04:00:00Z",
  "job_id": "SH_CLERK",
  "salary": 3200,
  "commission_pct": null,
  "manager_id": 123,
  "department_id": 50,
  "links": [
    {
      "rel": "self",
      "href": "http://localhost:8080/ords/hr/employees/194"
    },
    {
      "rel": "edit",
      "href": "http://localhost:8080/ords/hr/employees/194"
    }
  ]
}
```

Une approche plus manuelle

■ Etape 1 : Créer le service REST



A fournir :

- Nom de module
- URI du module (souvent = nom)

Il y aura également création d'un template.

Il faut fournir :

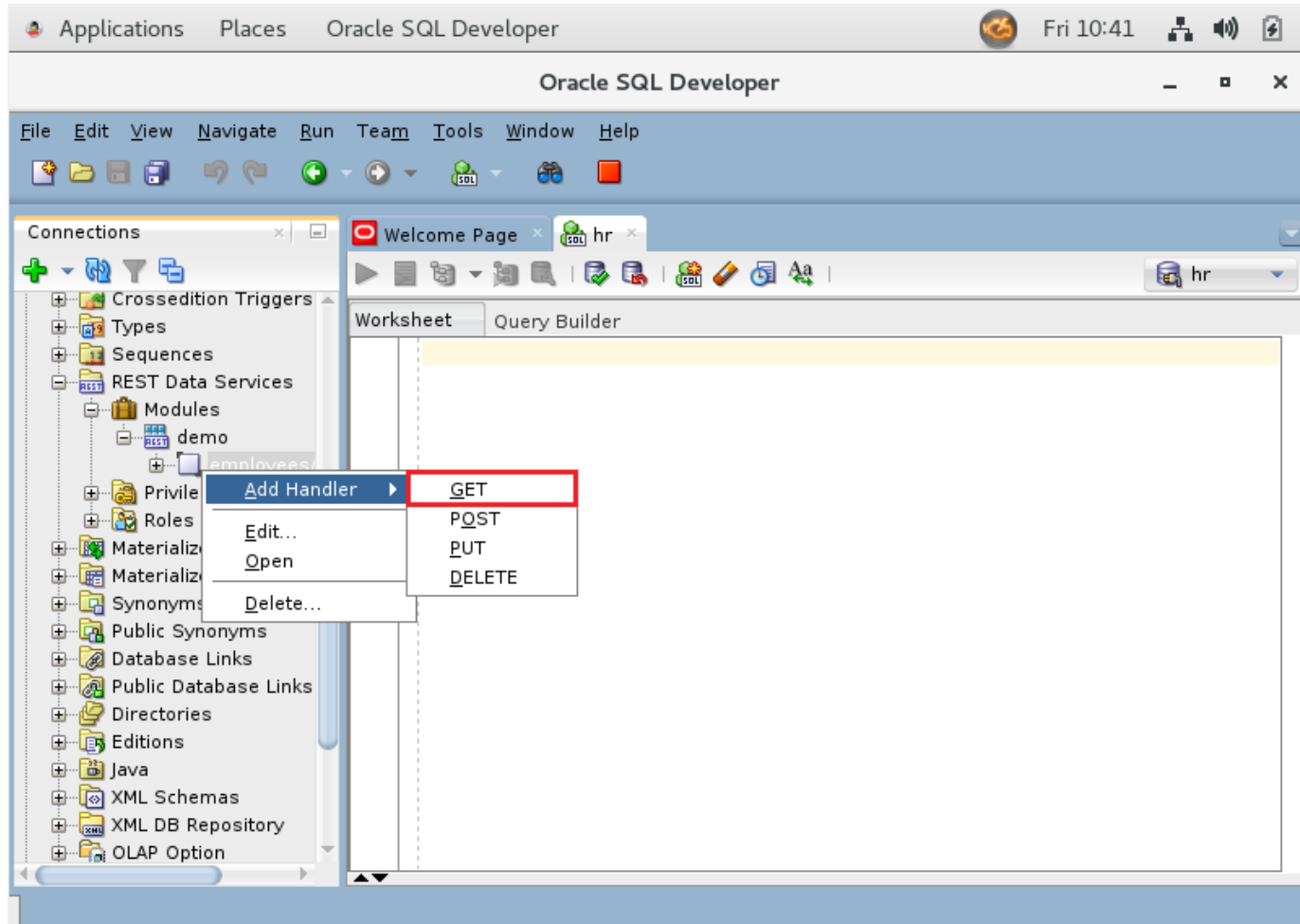
- Son nom
- Son URI

Note : L'exemple vous donnera une indication à quelle URL est publiée votre service

Exemple : <http://localhost:8080/ords/hr/demo/employees>

Création d'un handler

- Pont entre la requête HTTP et le SGBD



Choix d'un type de requête

The image shows a software dialog box titled "Modifier Gestionnaire de ressources". It contains several input fields and a dropdown menu. The "Méthode" field is set to "GET". The "Type de source" dropdown menu is open, showing a list of options. The "Résultats" and "Format de données" fields are empty. The "Taille de pagination" field is empty. The "Commentaires" field is empty. At the bottom, there are three buttons: "Aide", "Appliquer", and "Annuler".

Modifier Gestionnaire de ressources

Gestionnaire de méthode

Méthode : GET

Type de source : Requête de collecte

- Requête de collecte
- Élément de requête de collecte
- PL/SQL
- Ressource de support
- Requête (en phase d'abanbon, utiliser Requête de collecte)
- Requête sur une ligne (en phase d'abanbon, utiliser Élément de requête de collecte)
- Flux (en phase d'abandon)

Résultats

Format de données

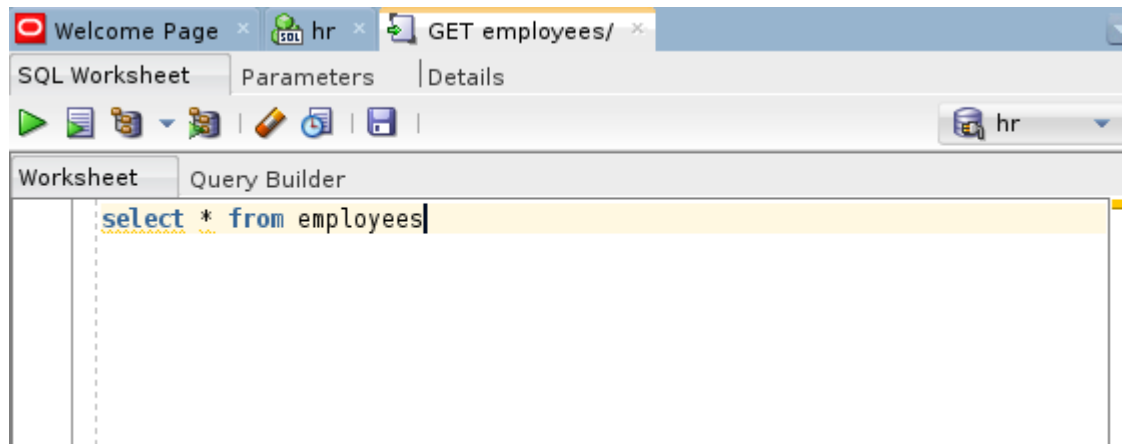
Taille de pagination

Commentaires

Aide Appliquer Annuler

Spécifier le code du Handler

- Requête de collecte :
 - Select « simple ». Le résultat du select sera retourné via REST



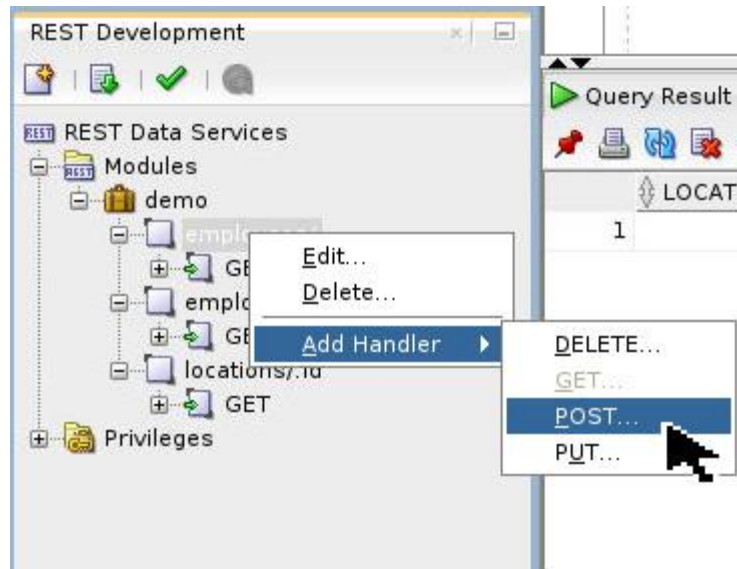
- PL/SQL :
 - Utiliser le package http (voir plus tard)
 - Autres techniques, par ex. XML

Paramètres du service REST

- Possibilité de passer un paramètre à la commande REST
- Lors de la création du modèle, spécifier un nom de variable
 - Ex : employees/:id
- Utiliser cette variable dans le gestionnaire
 - Ex : `SELECT * FROM employees WHERE employee_id = :id;`
- Possibilité de cumuler les paramètres
 - Ex : employees/:nom/:prenom

Verbe POST

- Également possible de définir un handler POST



Changements par rapport à GET

- Il faut spécifier le type d'input
 - Ici : « application/json »
- Comme il s'agit d'un code PL/SQL, cela ne renverra rien par défaut
 - Utilisation de HTP.PRN

Changement par rapport à GET (2)

■ Exemple :

```
DECLARE
```

```
    id EMPLOYEES.EMPLOYEE_ID%type;
```

```
BEGIN
```

```
    SELECT employee_id into id from employees where  
UPPER>Last_Name) LIKE UPPER('%' || :lastname || '%') AND  
UPPER(First_Name) LIKE UPPER('%' || :firstname || '%') FETCH  
first 1 row only;
```

```
    http.prn(id) ;
```

```
END;
```

■ Mais comment appeler le handler?

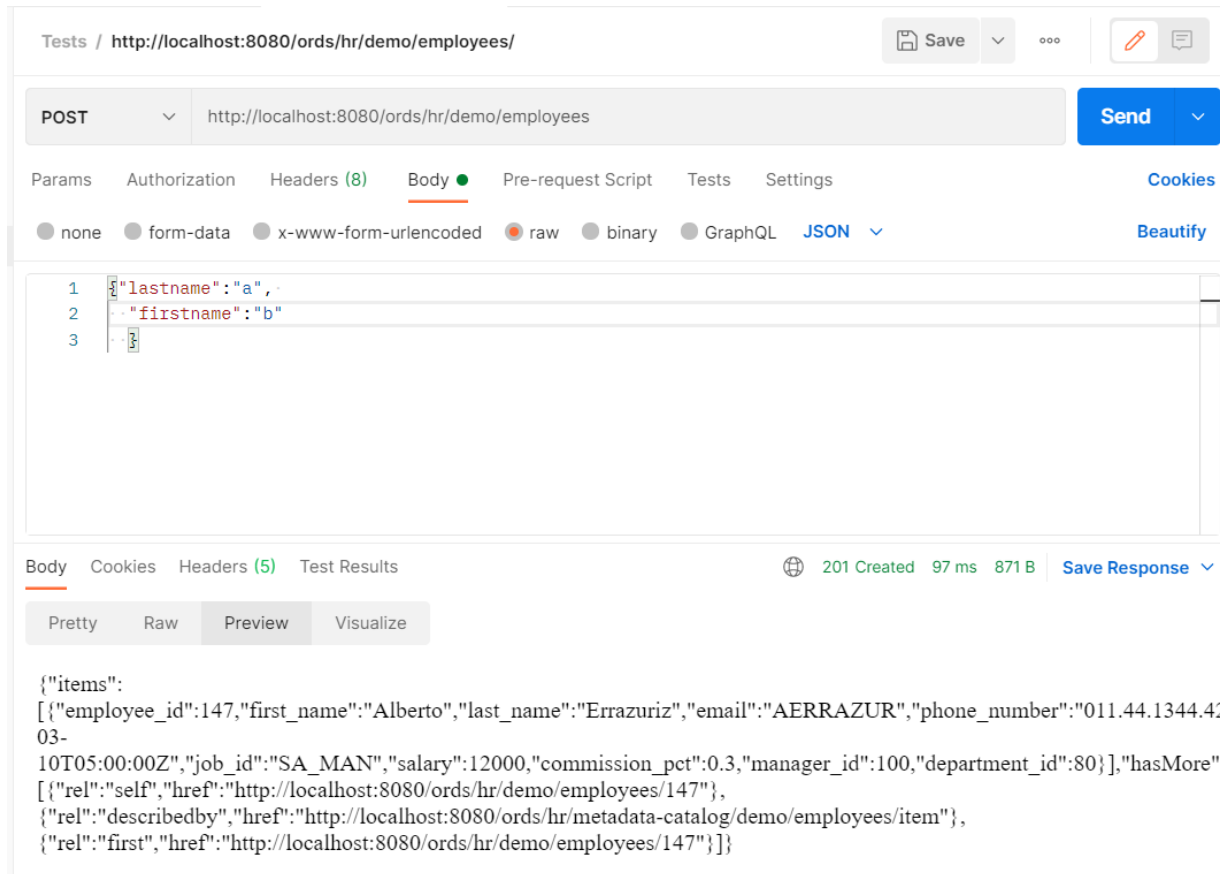
- Par défaut, le navigateur utilise le verbe GET

Agenda

- ORDS
 - REST
 - JSON
 - Auto-Rest
 - Handers REST manuels (GET et POST)
- PostMan
- Base 64

Postman

- Logiciel gratuit permettant d'envoyer et de recevoir des requêtes HTTP



Rappel : Logique des verbes

■ GET

- Récupération de résultats

■ POST

- Ajout
- Suppression
- Modification

Agenda

- ORDS
 - REST
 - JSON
 - Auto-Rest
 - Handers REST manuels (GET et POST)
- PostMan
- Base 64

Le problème des blobs

- Les Blobs sont des objets binaires
- Ils ne peuvent pas transiter tels quels sur le réseau
- Le gestionnaire ORDS va automatiquement transformer l'objet BLOB en chaîne de caractères grâce à l'encodage Base64

Encodage en base 64

- Transformer les objets binaires en chaînes de caractère
- Plusieurs méthodes
 - Quoted-printable encoding
 - Replace les caractères « bizarres » par un code précédé de =
 - Ex : caractère → `caract=E8re`
 - Plutôt pour les chaînes en grande partie valides
 - Base 64 encoding
 - Considère les données comme une chaîne de bits
 - Encode individuellement chaque groupe de 6 bits
 - 64 valeurs possibles (d'où le nom)
 - Chaque valeur a une correspondance vers un caractère dans une table.

Implémentation

- Pas besoin de réinventer la roue :

```
create or replace FUNCTION base64encode(p_blob IN BLOB) RETURN CLOB
-- -----
-- File Name      : https://oracle-base.com/dba/miscellaneous/base64encode.sql
-- Author         : Tim Hall
-- Description    : Encodes a BLOB into a Base64 CLOB.
-- Last Modified: 10/11/2021 by Samuel HIARD to remove the newlines
-- -----
IS
  l_clob CLOB;
  l_step PLS_INTEGER := 12000; -- make sure you set a multiple of 3 not higher than 24573
BEGIN
  IF(p_blob is not null) THEN
    FOR i IN 0 .. TRUNC((DBMS_LOB.getlength(p_blob) - 1)/l_step) LOOP
      l_clob := l_clob ||

      replace(
        replace(
          utl_raw.cast_to_varchar2(utl_encode.base64_encode(DBMS_LOB.substr(p_blob, l_step, i * l_step + 1))),
          chr(10)),
          chr(13)
        );
    END LOOP;
  END IF;
  RETURN l_clob;
END base64encode;
```

Utilisation de la fonction

■ Dans le handler

```
OPEN c
FOR SELECT base64encode(img)
FROM BlobTest
FETCH NEXT 1 ROW ONLY ;
```

■ Résultat

Body Cookies Headers (4) Test Results

Status: 200 OK Time: 805 ms Size: 560.03 KB Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2
3   "BASE64ENCODE(IMG)": "iVBORw0KGgoAAAANSUhuEugAAApUAAAGjCAYAAAB9mu6KAAAGAE1EQVR4nNS9948kSZbn9zFz91CpZWwwbD3d092jbva0x7s1QfK4dxQgCIIE
+ C8SBAECPOLI5exid+8wezM7ukVVV3dVl66szEoZGRnK3Yw/mJm7uYqIFNUza91ZEeFu4pnZs/e+75kS41hpKoLWlY+vNNSVobXwf9m/QhAKtABE+V1F3kII
+ 7yUUS5uXTqfnjLdJo6U+bhaq1ye510U0tWVXRK9ExPn9Vdp7+llAghvD8otrNpt/9XzMv155fll6mUQmuBUoo4ju1vXXhvytcaS4sgCAJAKySK0ThhNI4Zj7P05g
+ UUitiJVGufyU6eu0TWxcWwxKq3K/aGFo80rv/vJlFtoJAJl7LoTI4lawsN+Prg2DQ0baz/VLGAWEYUCrEdFsNmg0IsIgSLmnjn/
+ 9MrJ2JfcsI0zkiBQCwLASBoIwDGg0IgIp0FoRj8q0udePAFJKU1Wli0PEXLNjWSMQgJSGlQVMf2RjWIPQ+X5NXN+BSGWczvhV2LxwfZ3Rk7ajNHUxbStzveb4w/Svq37V
+ PI6TyiEkAiCtL2VUqUUVePALzdrf18ulfsu41GRY6FpsqKKD7Q2bZc2ewW9AFIISGUcoHSBffPy0SsqH88vV2ssBwjy4y7LMethTKcJMqbVKs+qNmL6yNHsXmqN
+ U9MaCuN0B6dNp0rUx0Yd1AIoQ1JQoJ241QzVjHPX7zm7/7mc7767Dk/+vR9/rP//FM2tucRgVeSVl6/1
+ m9at3iZEGWT33fCwGBFEhpZLKUApUoEhWD1lbuSwTCfEoQQpFowXgk+Ku/+jkfffQ+17ZwcvxiWkfk27+CftK49TrVjfU6XwbUZnHc
+ Kk1SMdD7mFRlwpEIw11u2m0MHUrvnbywecR07vIiwDD3CnfFHU0tjoed6Zhki6vDSLncEL0uc8q+V/I0MZz/
```

Décodage

- Objectif : pouvoir récupérer le contenu binaire
- Traitement
 - Ex : Transformation de ce contenu dans une image et association à un JLabel
- En java, classe Base64 pour décoder
 - Pas demandé dans le projet

Côté Java

