ORACLE

Manage JSON Data Without Limits in Oracle Database

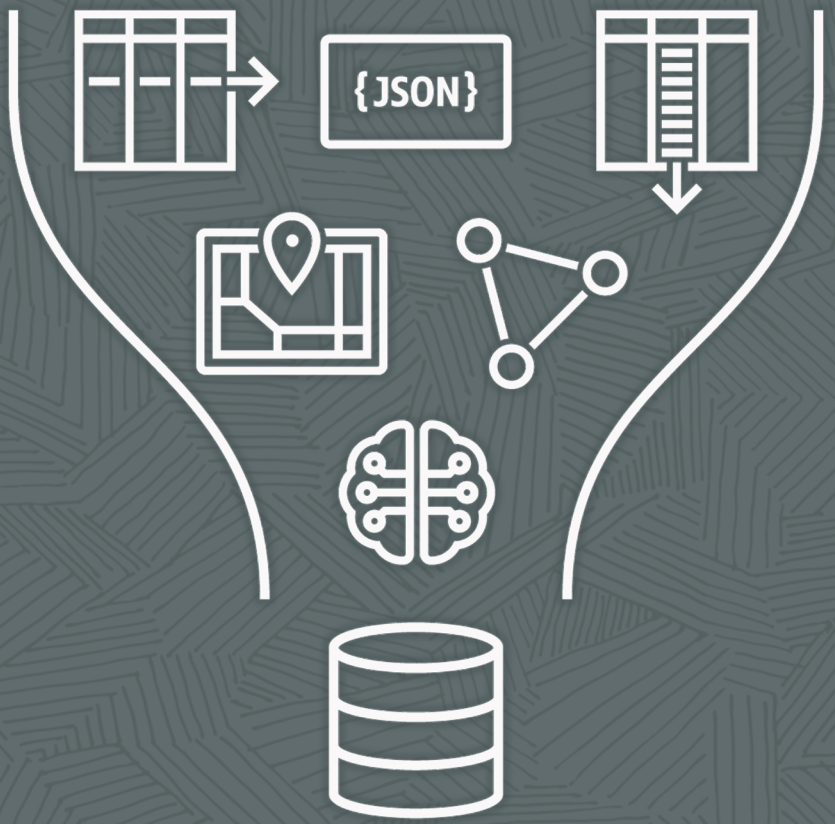
Simplify Development of JSON Applications with Oracle Converged Database

EMEA Data Management Experts Team

- Jesus Robles
- Stephane Duprat
- Rob Watson
- Witold Świerzy

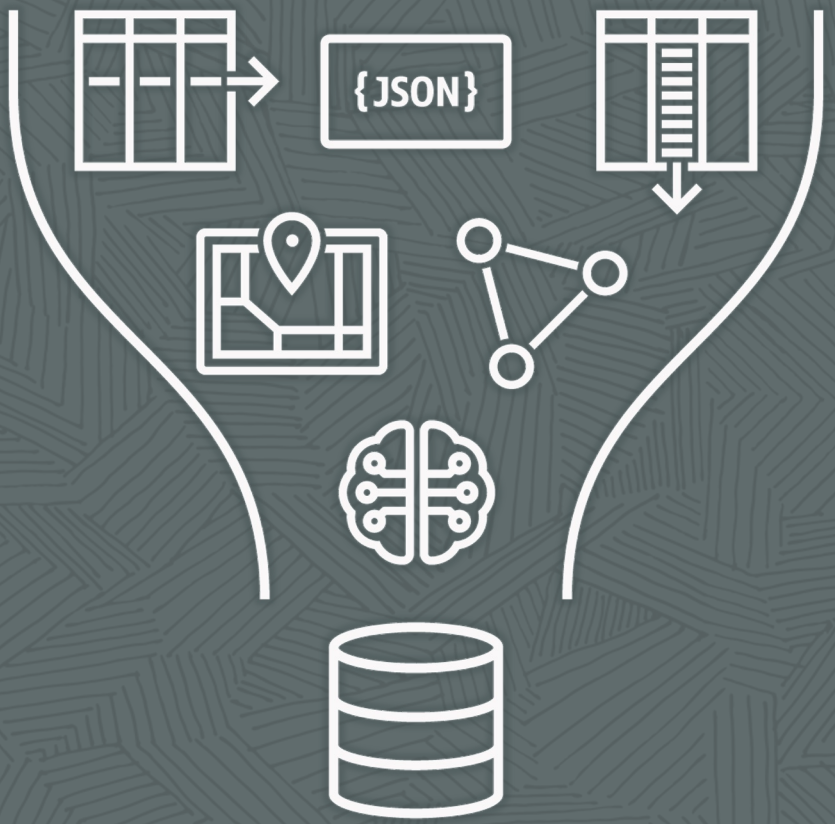
Agenda

- How to run the HOL
- HOL main concepts
- Q&A



Agenda

- **How to run the HOL**
- HOL main concepts
- Q&A



How to run the HOL

Overview of the HOL components

Documents

- JSON.PERF_v1.5_ENG.pdf: the user's guide, with the commands and explanations of the concepts
- SQL scripts: for easy copy/paste of the commands, a set of scripts is provided (avoid to copy/paste from the PDF).
  - 001.TEST.PERF.SOE.sql
  - 002.TEST.PERF.SOE-JSON_PART.sql
  - 003.JSON.MEMOPTIMIZE.FOR.WRITE.sql
  - 004.ORDS.sql

Environment

- A VM with 4 cores is provided, with the Oracle Linux Server 8.4 operating system
- For the database, RDBMS 21.3 is used.
- A PDB named "ORCLPDB1" is used for the lab
- A schema "SOE" is used and has been populated with data.

You can easily setup and run this lab on-premises or in your Cloud tenancy ....

How to run the HOL

How to setup and run the lab on-premise/in your Cloud tenancy

Create a test database

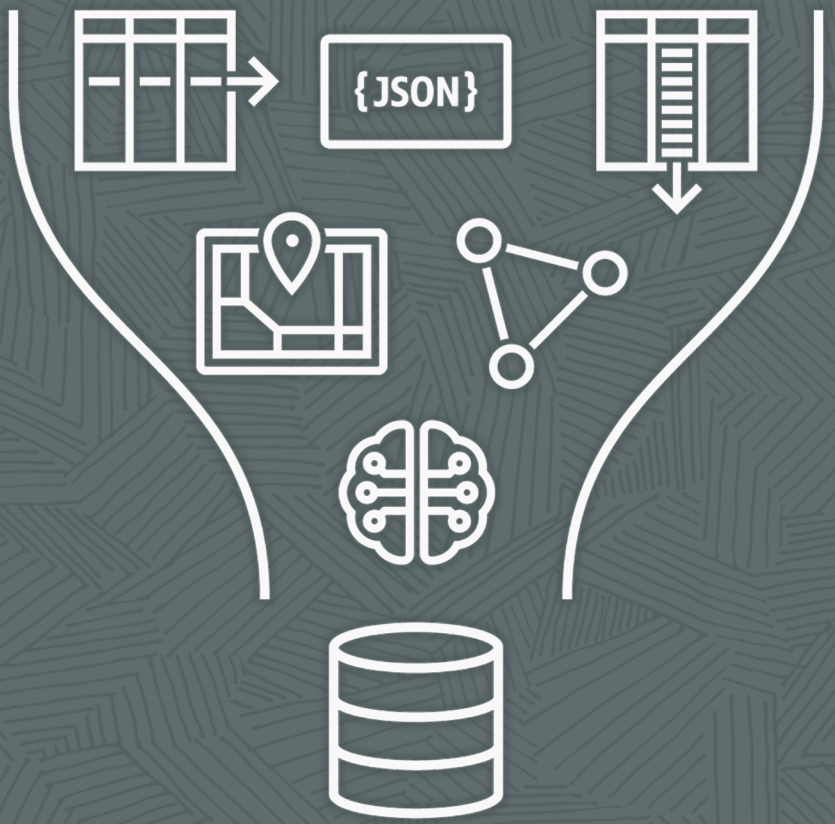
- Create a 21.3 (or superior) database
- Download the SOE schema datapump file, using the URL: https://objectstorage.eu-frankfurt-1.oraclecloud.com/p/T4LB-AJpkVCvBC15MQg9reMtKcsQVzvP0nUx1PpmnhjFS7F2aN5AVljzi_yo56uZ/n/oractdemeabdmautodb/b/TMP/o/expdp.soe.dmp
- Import the SOE schema in your database: connect to your PDB as system and run:

```
create user soe identified by "Oracle_4U" default tablespace USERS temporary tablespace TEMP;
grant connect, resource to SOE;
create directory DIR_DP as '/home/oracle/DATAPUMP'; -- Point to the directory where you saved the DMP file !!!
create tablespace TBS_OE datafile '/opt/oracle/oradata/MY19DB/MY19PDB1/tbs_oe.dbf' size 256M autoextend on maxsize 8G;
alter user SOE quota unlimited on USERS;
alter user SOE quota unlimited on TBS_OE;
exit
impdp system/passwd@yourPDB DIRECTORY=DIR_DP DUMPFILE=expdp.soe.dmp full=y logfile=impdp.soe.log
```

That's all you need ... You're ready to run the lab

Agenda

- How to run the HOL
- **HOL main concepts**
- Q&A



What you'll practice in this HOL

Overview of the main concepts

Managing JSON in the Oracle database

- Create a table to store JSON documents
- Practice cross-model queries: manipulate relational and JSON data in the same query

```
create table OI_JSON_ORDERS
(
ID number(12),
O_JSON VARCHAR2(4000),
CONSTRAINT O_JSON_insert_pk primary Key (id),
CONSTRAINT O_JSON_check CHECK (O_JSON IS JSON)
);
```

```
select W.WAREHOUSE_NAME, sum(to_number(json_value (OI.O_JSON, '$.ORDER_TOTAL'))) as TOTAL
from    OI_JSON_ORDERS OI,
        WAREHOUSES W
where   W.WAREHOUSE_ID = json_value (OI.O_JSON, '$.WAREHOUSE_ID')
and     W.warehouse_name in  ('McsRxsWjRxXMFDcobjhEIDdEsO','5eH6XK38SRmNEZCUg43EDIjDICDhbV','PLlypy')
group by W.WAREHOUSE_NAME
order by 1;
```

Quick demo ...

What you'll practice in this HOL

Overview of the main concepts

Boost the performance of cross-model queries

- Create an index on a JSON field
- Partition on a relational column or on a JSON field to leverage partition pruning
- Create a SEARCH index

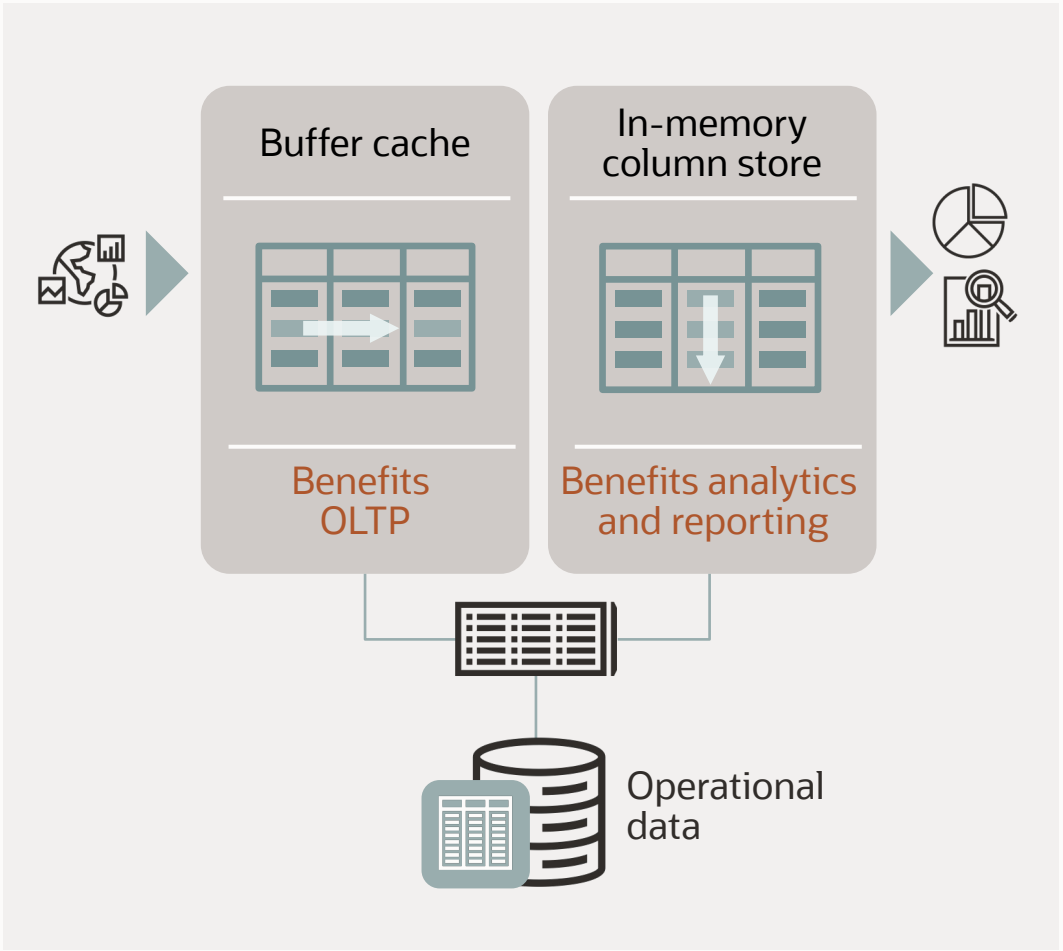
```
create index I_CUST_ID  on
OI_JSON_ORDER_ITEMS
(
    json_value (OI_JSON, '$.CUSTOMER_ID'  returning NUMBER(12) error on error null on empty)
) LOCAL;

Index created.
```

```
create search index I_JSON_SEARCH on OI_JSON_ORDER_ITEMS(OI_JSON) for JSON;


Index created.
```

```
CREATE TABLE OI_JSON_ORDER_ITEMS_PART
(id NUMBER(12) NOT NULL PRIMARY KEY,
   OI_JSON VARCHAR2(4000),
   ORDER_DATE DATE GENERATED ALWAYS AS
     (json_value (OI_JSON, '$.ORDER_DATE' RETURNING DATE))
)
PARTITION BY RANGE (ORDER_DATE) INTERVAL(NUMTOYMINTERVAL(1, 'MONTH'))
  (
    PARTITION OIJSONPART_P0 VALUES LESS THAN (TO_DATE('2007-02-01', 'YYYY-MM-DD'))
  );
```

Quick demo ...

Breakthrough: Dual Format Database



- BOTH row and column formats for same table
- Simultaneously active and transactionally consistent
- Analytics & reporting use new in-memory Column format
- OLTP uses proven row format
- No application changes required

Oracle In-Memory: Simple to Implement

1. Configure Memory Capacity

   ```
   inmemory_size = XXX GB
   ```

2. Configure tables or partitions to be in memory

   ```
   alter table | partition  …  inmemory;
   ```

3. Later drop analytic indexes to speed up OLTP

Database In-Memory Technology

Scanning and filtering data more efficiently

Columnar Format

Access only the columns you need

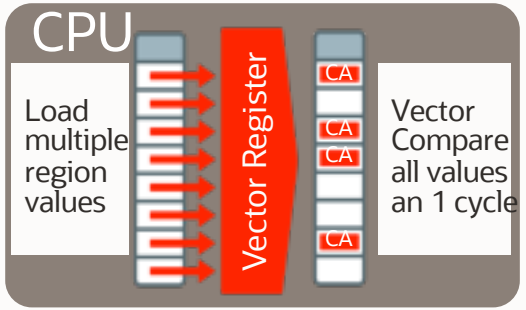
Compression

Scan & filter data in compressed format

Storage Indexes

Prune out any unnecessary data from the column

SIMD Vector Processing



Process multiple column values in a single CPU instruction

What you'll practice in this HOL

Overview of the main concepts

Boost the performance of JSON analytical queries

- Use In-memory column store with JSON data

```
select PARTITION_NAME, HIGH_VALUE from user_tab_partitions where table_name = 'OI_JSON_ORDER_ITEMS';

SYS_P2582
TO_DATE(' 2009-03-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA'

alter table OI_JSON_ORDER_ITEMS modify partition SYS_P2582 inmemory priority critical;

Table altered.
```

```
select sum(ARR.UNIT_PRICE*ARR.QUANTITY) as "GrantTotal"
from OI_JSON_ORDER_ITEMS OIJ,
     json_table(OIJ.OI_JSON,
                '$' COLUMNS (ORDER_DATE DATE path '$.ORDER_DATE',
                             NESTED PATH '$.ITEMS[*]'
                             COLUMNS (UNIT_PRICE NUMBER path '$.UNIT_PRICE',
                                      QUANTITY NUMBER path '$.QUANTITY'))
                ) as ARR
where OIJ.ORDER_DATE between to_date('01-FEB-09 00:00:00','DD-MON-RR HH24:MI:SS') and to_date('28-FEB-09 23:59:59','DD-MON-RR HH24:MI:SS');
```

Quick demo ...

Memoptimized Rowstore : Fast Ingest Support

NEW IN 19c

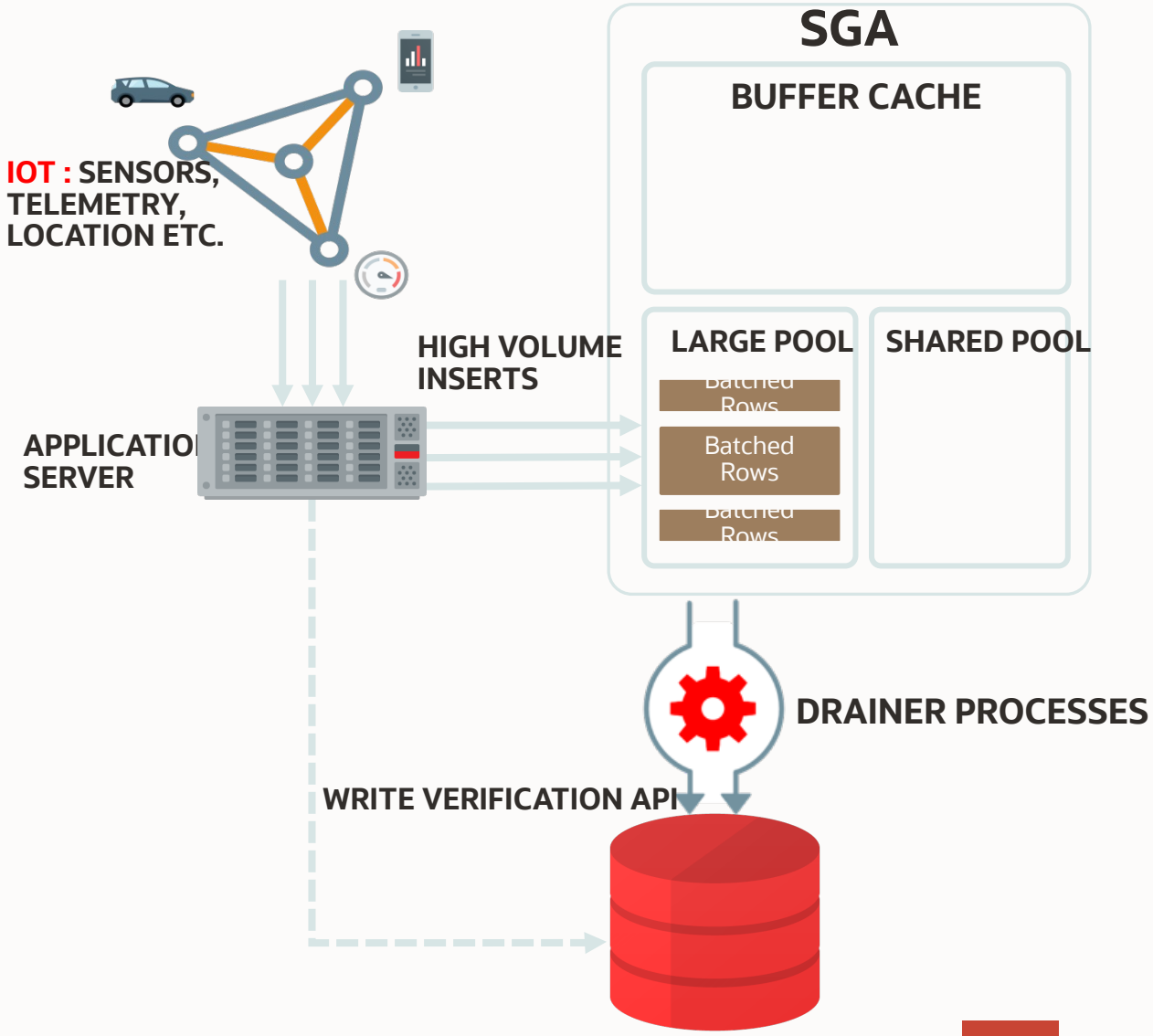
A memory optimised mechanism for inserting data into the database

Ideal for light weight IoT transactions

Rows are cached in memory and asynchronously drained to disk

An API allows developers to check on the durability of their inserts

- Declare table `MEMOPTIMIZE FOR WRITE`
- Use new hint `MEMOPTIMIZE_WRITE`



What you'll practice in this HOL

Overview of the main concepts

Boost JSON document ingestion

- Use MEMOPTIMIZE FOR WRITE tables

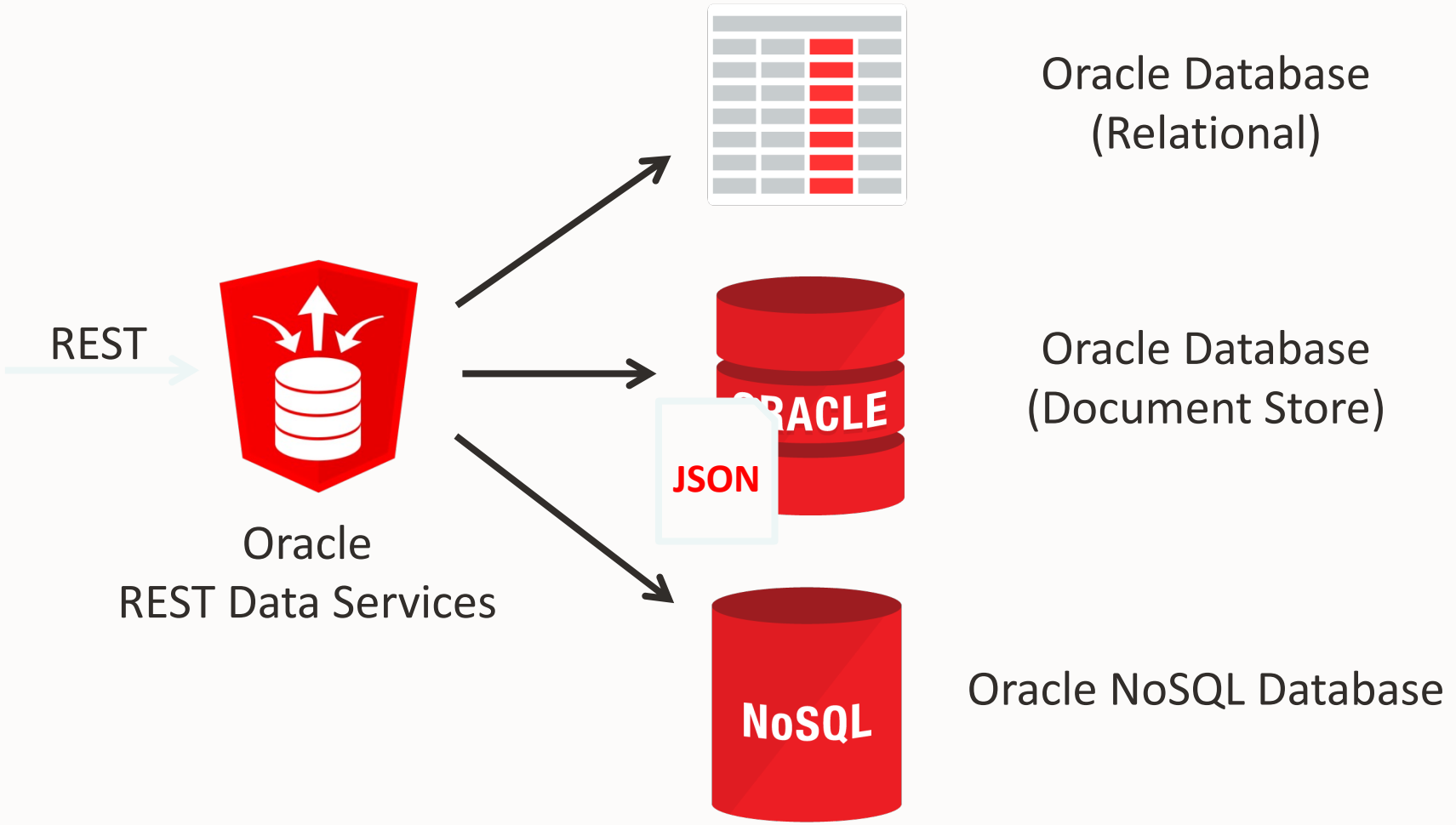
```
create table OI_JSON_MEMOPT4WRITE
(
    ID number(12),
    OI_JSON varchar2(4000),
    CONSTRAINT oi_json_MEMOPT_pk primary Key (id),
    CONSTRAINT OI_json_MEMOPT_check CHECK (OI_json IS JSON)
) segment creation immediate memoptimize for write;
```

```
create or replace procedure PC_INS_MEMOPT4WRITE (p_num_rows IN PLS_INTEGER)
IS
    CURSOR c_oi (p_num IN PLS_INTEGER)
    IS
        select id, OI_json
        from OI_JSON_ORDER_ITEMS
        where rownum <= p_num;
begin
    FOR cur in c_oi (p_num_rows)
    LOOP
        insert /*+ memoptimize_write */ into OI_JSON_MEMOPT4WRITE (id,oi_json) values (cur.id,cur.oi_json);
        commit;
    END LOOP;
END;
/
```

Quick demo ...

Oracle REST Data Services

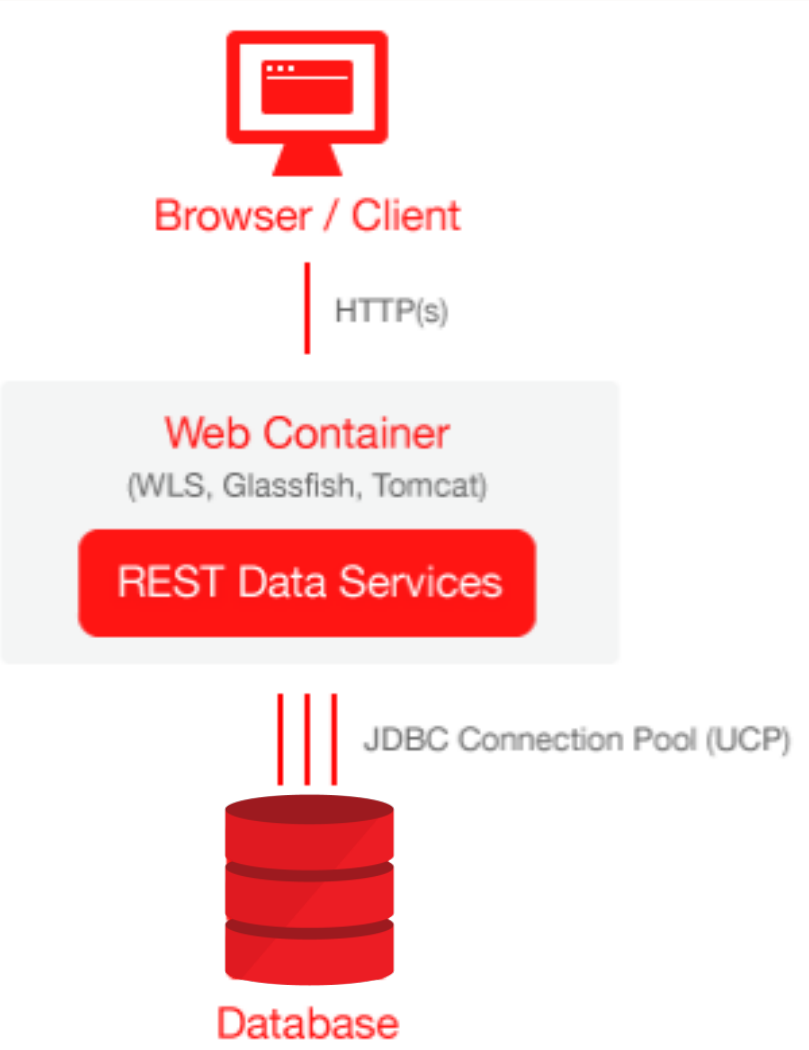
REST-enable your data



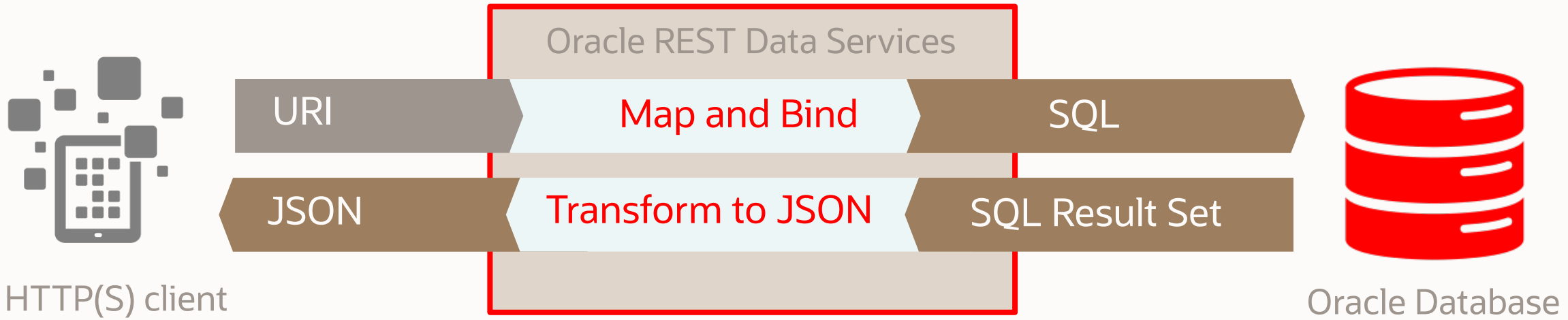
Oracle Database Cloud Service/on-premise

Oracle REST Data Services

- Available Standalone (Jetty), Weblogic, Tomcat & Glassfish
- Turns Database Service into an RESTFul API service
- Fully provisioned and functional in all cloud editions
- Available in 11g, 12c and 18c, no extra cost
- Allows publishing of URI based access to Oracle database over REST
- Results in JSON or CSV
- Mapping of URI to SQL or PL/SQL
- All HTML methods GET, PUT, POST, DELETE, PATCH
- Oauth2 integration
- Highly scalable, can use all feature of database



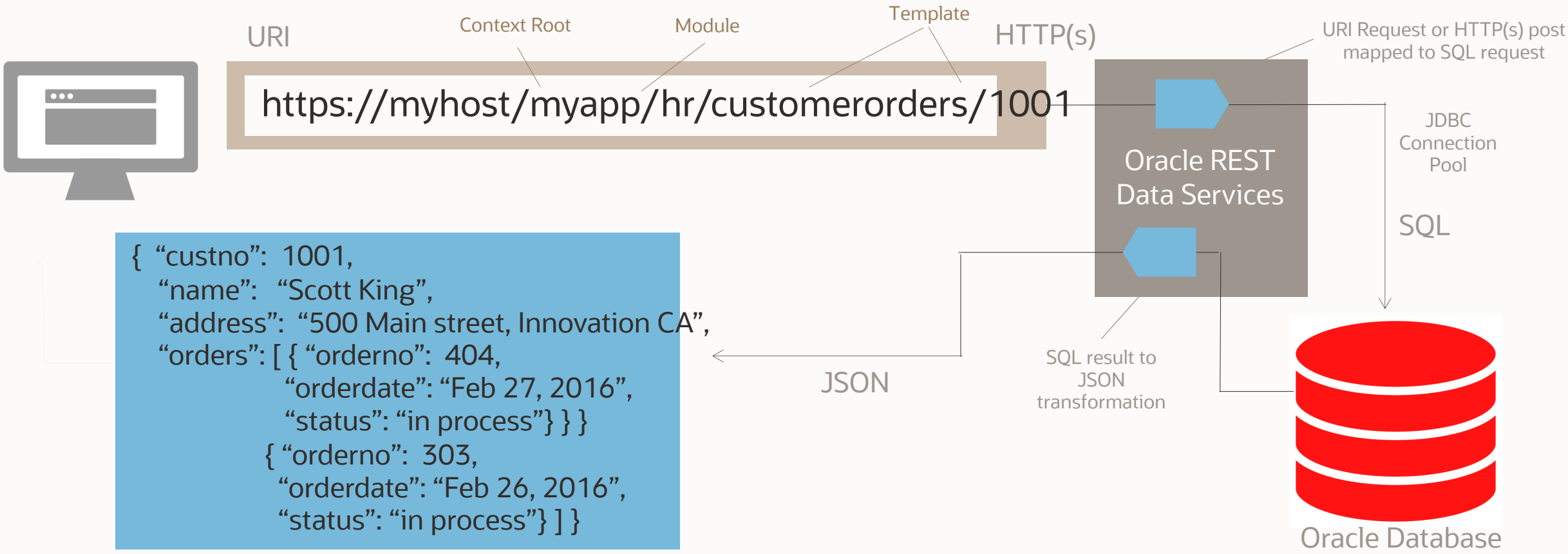
Oracle REST Data Services
Serving JSON results from relational data



- Data stored in standard relational tables and columns
- Oracle REST Data Services (ORDS) Developer defines URI<>SQL mapping
- App Developer calls named URI over HTTP(S) gets and posts

Oracle REST Data Services

HTTP(s) API App-Dev with Relational Tables in Oracle Database



ORDS maps standard URI requests to corresponding relational SQL (not schemaless): e.g. SQL SELECT from customers and orders table.
ORDS also transforms the SQL results into JavaScript Object Notation (JSON), other formats include HTML, binary and CSV.
Fully committed to supporting any and all standards required by Fusion / SaaS / FMW; we are actively engaged in the ongoing dialog.

What you'll practice in this HOL

Overview of the main concepts

APIfy your data

- Use Oracle Rest Data Services (ORDS)

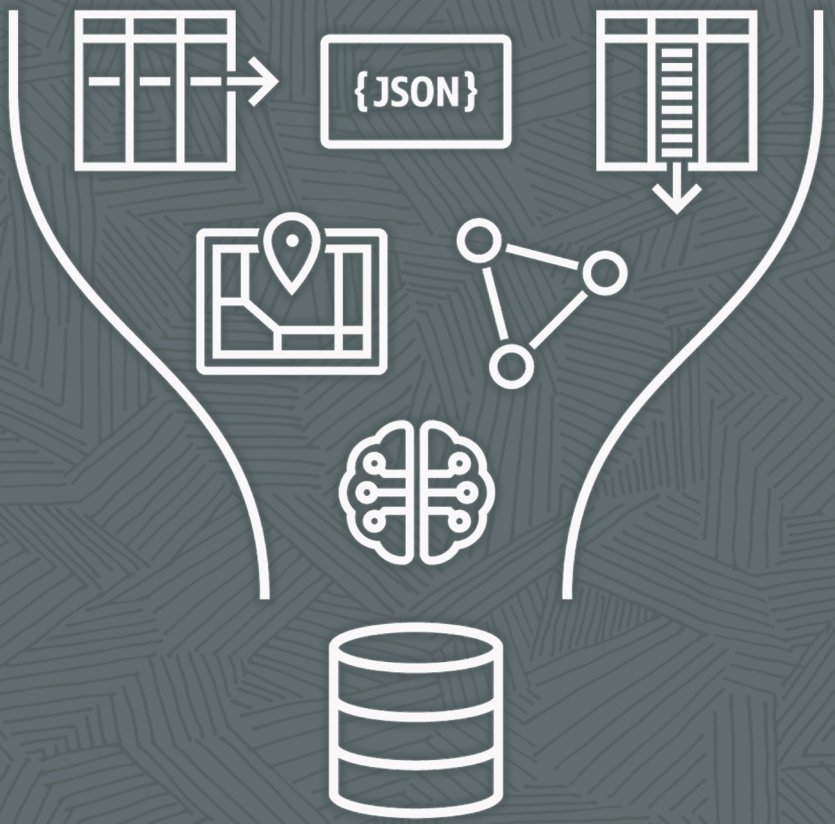
```
BEGIN
    ords_admin.enable_schema (
        p_enabled               => TRUE,
        p_schema                => 'SOE',
        p_url_mapping_type      => 'BASE_PATH',
        p_url_mapping_pattern   => 'soe',
        p_auto_rest_auth        => TRUE    -- this flag says, don't expose my REST APIs
    );
    COMMIT;
END;
/
```

```
DECLARE
  PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN

    ORDS.ENABLE_OBJECT(p_enabled => TRUE,
                       p_schema => 'SOE',
                       p_object => 'CUSTOMERS',
                       p_object_type => 'TABLE',
                       p_object_alias => 'customers',
                       p_auto_rest_auth => FALSE);

    commit;

END;
/
```

```
BEGIN
  ORDS.define_service(
    p_module_name    => 'salesreporting',
    p_base_path      => 'salesrep/',
    p_pattern        => 'sales/:custid',
    p_method         => 'GET',
    p_source_type    => ORDS.source_type_collection_feed,
    p_source         => 'SELECT to_char(O.order_date,''YYYYMM'') as MONTH, sum(OI.unit_price) as TOTAL
    FROM orders O, order_items OI where O.order_id = OI.order_id and O.customer_id = :custid group by to_char(O.order_date,''YYYYMM'')',
    p_items_per_page => 0);

  COMMIT;
END;
/
```

Quick demo ...

Q&A




ORACLE