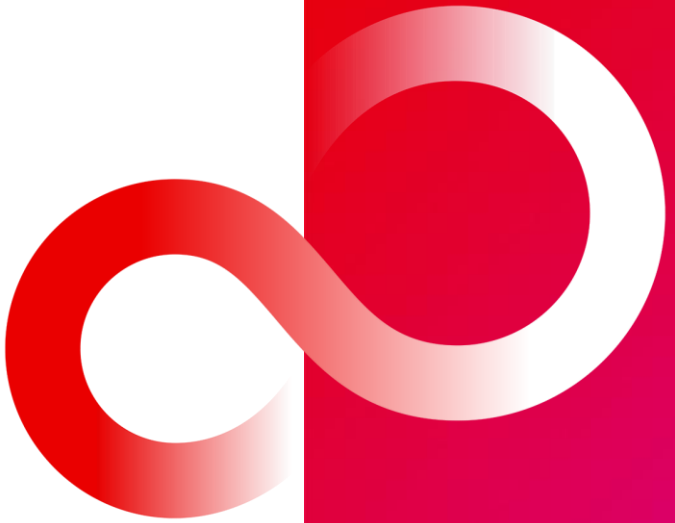
Oracle Database 23c 新機能検証

FUJITSU

JSON RELATIONAL DUALITY VIEW



2024年2月

富士通株式会社

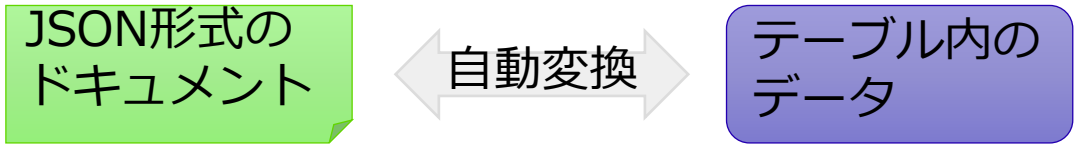
検証の目的

- JSON RELATIONAL DUALITY VIEW について以下を実測し、処理のオーバーヘッドを確認する
  - JSON RELATIONAL DUALITY VIEW アクセスのオーバーヘッド
  - テーブルサイズの大小に対するJSON RELATIONAL DUALITY VIEW アクセスのオーバーヘッド

JSON RELATIONAL DUALITY VIEWとは

FUJITSU

- 単一のデータベース内のリレーショナルおよびドキュメント世界のメリットを集約
- JSON形式のドキュメントと従来のテーブル内のデータをシームレスに変換する機能を提供



- 参考
  - （オラクルエンジニア通信 - 技術資料、マニュアル、セミナー）JSONとリレーショナルの二面性: ドキュメント、オブジェクトおよびリレーショナル・モデルの革新的な統合
    - https://blogs.oracle.com/oracle4engineer/post/ja-json-relational-duality-app-dev

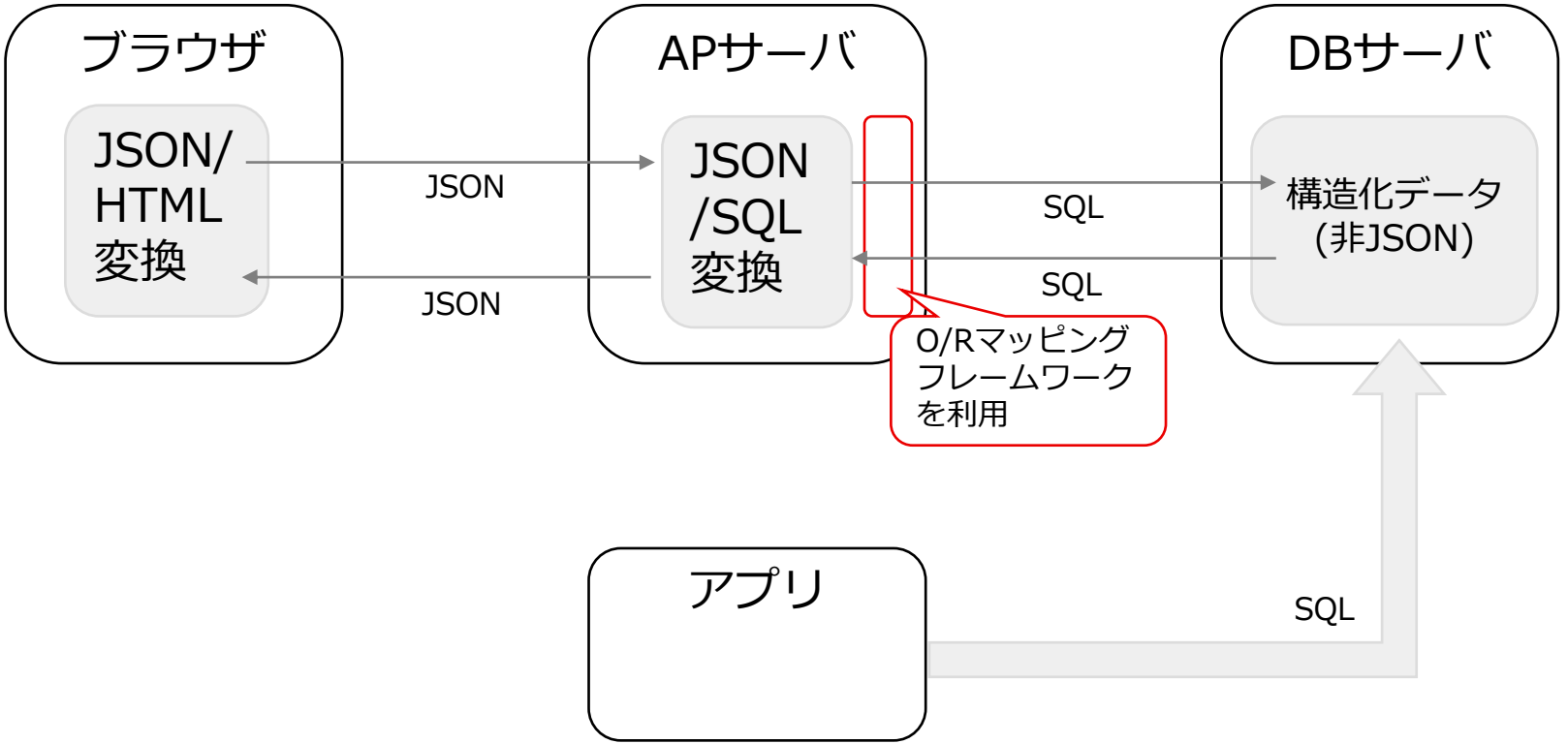
＜参考＞JSONとは

FUJITSU

- JSON =「JavaScript Object Notation」の略称です。
- JavaScriptはプログラミング言語の1つで、Object Notationは人間が容易に読み書きしながらデータを簡単に扱えるように構造化した記述方法を意味します。
- 例

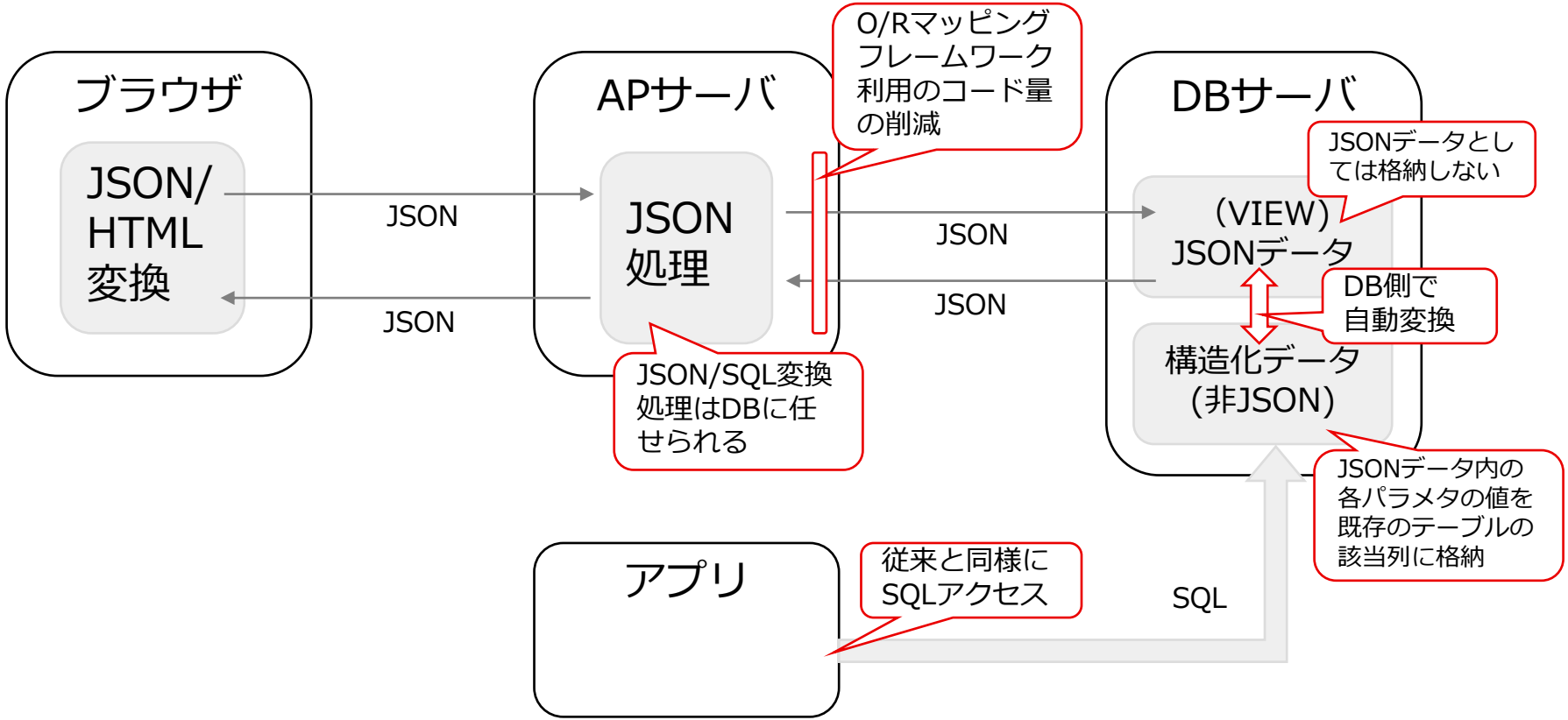
```
{
  "total": 3,
  "users": [
    { "name": "佐藤太郎", "age": 22 },
    { "name": "高橋次郎", "age": 18, },
    { "name": "田中三郎", "age": 21, }
  ]
}
```

従来



RELATIONAL DUALITY VIEW 機能利用時

FUJITSU



テーブル・ビュー

FUJITSU

- テーブルとJSON RELATIONAL DUALITY VIEW定義

```
# 既存のテーブル定義
SQL> desc EXTABLE1
 名前          NULL?    型
 ------------ -------- ----------------------------
 ID                     NOT NULL VARCHAR2(20)
 PARAM1                          VARCHAR2(40)
 PARAM2                          DATE
 PARAM3                          VARCHAR2(200)
 PARAM4                          NUMBER(38)
 PARAM5                          VARCHAR2(200)
```

```
# 作成した JSON RELATIONAL DUALITY VIEW 定義に
# 対する定義
SQL> desc EXTABLE1_DV
 名前          NULL?    型
 ------------ -------- ----------------------------
 DATA                   JSON
```

列名 DATA 固定の
単一の列を作成

```
# 作成する JSON RELATIONAL DUALITY VIEW 定義
CREATE OR REPLACE JSON RELATIONAL DUALITY VIEW
ExTable1_DV AS
  SELECT JSON {
               'exId'      : ex1.id,
               'exParam1'  : ex1.param1,
               'exParam2'  : ex1.param2,
               'exParam3'  : ex1.param3,
               'exParam4'  : ex1.param4,
               'exParam5'  : ex1.param5 }
    FROM ExTable1 ex1 WITH INSERT UPDATE DELETE;
```

アプリ実装の違い

	従来	RELATIONAL DUALITY VIEW 利用時
アプリ側	• テーブルの列構造を考慮した実装が必要 • 特定列のデータのみを操作したい場合、それに合わせた実装が必要	• 単一の列(DATA)をもつテーブルに見え、バックのテーブルの構造の考慮は不要 • 特定列のデータのみを操作したい場合、実装の差異はほとんどないが、それぞれの場合に合わせたJSON RELATIONAL DUALITY VIEW定義が必要になる
DB側		• JSONデータとテーブル列の対応関係の定義（JSON RELATIONAL DUALITY VIEW）が必要

JSON RELATIONAL DUALITY VIEW／テーブル変換

FUJITSU

- アプリから見えるJSON RELATIONAL DUALITY VIEW

列名	型
DATA	JSON

```
# アプリが使用するJSONデータ
{
  “exId": "O20231012002",
  “exParam1": “param1 data.",
  "exParam2": "2023-10-12",
  "exParam3": “param3 data.",
  "exParam4": 999,
  "exParam5": “param5 data.”
}
```

JSONデータ内のパラメタを各テーブルの列に自動的に関連付け

- テーブル定義

列名	型
ID	VARCHAR2 (20)
PARAM1	VARCHAR2 (40)
PARAM2	DATE
PARAM3	VARCHAR2(200)
PARAM4	NUMBER(38)
PARAM5	VARCHAR2(200)

測定モデル

FUJITSU

- JSON RELATIONAL DUALITY VIEW について以下を実測し、処理のオーバーヘッドの確認
  - JSON RELATIONAL DUALITY VIEW アクセスのオーバーヘッド
  - テーブルサイズの大小に対するJSON RELATIONAL DUALITY VIEW アクセスのオーバーヘッド

テーブル定義 1

FUJITSU

- テーブル定義 (ExTable1)

名前	NULL?	型
ID	NOT NULL	VARCHAR2(20)
PARAM1		VARCHAR2(40)
PARAM2		DATE
PARAM3		VARCHAR2(200)
PARAM4		INT
PARAM5		VARCHAR2(200)

- JSON RELATIONAL DUALITY VIEW 定義

```
CREATE OR REPLACE JSON RELATIONAL DUALITY
VIEW ExTable1_DV AS
  SELECT JSON {
      'exId'     : ex1.id,
      'exParam1' : ex1.param1,
      'exParam2' : ex1.param2,
      'exParam3' : ex1.param3,
      'exParam4' : ex1.param4,
      'exParam5' : ex1.param5 }
    FROM ExTable1 ex1 WITH INSERT UPDATE
DELETE;
```

ID列以外の列数：5
検証での更新データサイズ合計：320バイト

テーブル定義２

FUJITSU

- テーブル定義 (ExTable3)

名前	NULL?	型
ID	NOT NULL	VARCHAR2(20)
PARAM1		VARCHAR2(40)
PARAM2		DATE
PARAM3		VARCHAR2(200)
PARAM4		INT
PARAM5		VARCHAR2(200)
PARAM6		VARCHAR2(200)
・・・		（同一）
PARAM10		VARCHAR2(200)

ID列以外の列数：10
検証での更新データサイズ合計：820バイト

- JSON RELATIONAL DUALITY VIEW 定義

```
CREATE OR REPLACE JSON RELATIONAL DUALITY
VIEW ExTable2_DV AS
  SELECT JSON {
      'exId'     : ex2.id,
      'exParam1' : ex2.param1,
      'exParam2' : ex2.param2,
      'exParam3' : ex2.param3,
      'exParam4' : ex2.param4,
      'exParam5' : ex2.param5,
      'exParam6' : ex2.param6,
       ・・・
      'exParam10' : ex2.param10 }
    FROM ExTable2 ex2 WITH INSERT UPDATE
DELETE;
```

テーブル定義３

FUJITSU

- テーブル定義 (ExTable3)

名前	NULL?	型
ID	NOT NULL	VARCHAR2(20)
PARAM1		VARCHAR2(40)
PARAM2		DATE
PARAM3		VARCHAR2(200)
PARAM4		INT
PARAM5		VARCHAR2(200)
PARAM6		VARCHAR2(200)
・・・		（同一）
PARAM20		VARCHAR2(200)

ID列以外の列数：20
検証での更新データサイズ合計：1820バイト

- JSON RELATIONAL DUALITY VIEW 定義

```
CREATE OR REPLACE JSON RELATIONAL DUALITY
VIEW ExTable3_DV AS
  SELECT JSON {
      'exId'     : ex2.id,
      'exParam1' : ex3.param1,
      'exParam2' : ex3.param2,
      'exParam3' : ex3.param3,
      'exParam4' : ex3.param4,
      'exParam5' : ex3.param5,
      'exParam6' : ex3.param6,
       ・・・
      'exParam20' : ex3.param20 }
    FROM ExTable3 ex3 WITH INSERT UPDATE
DELETE;
```

測定モデル

FUJITSU

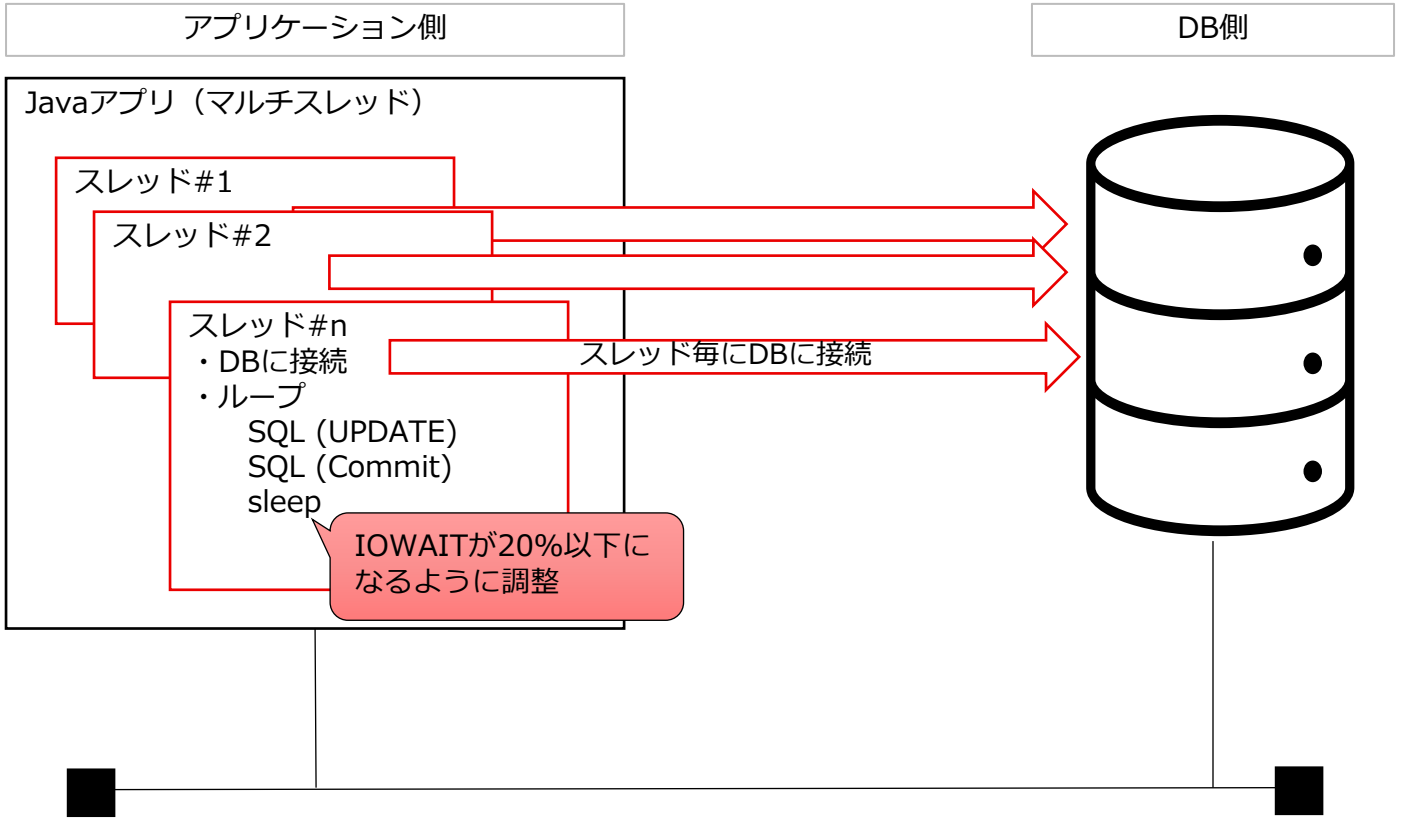
- JSON RELATIONAL DUALITY VIEW についてベンチマークを実施
  - JSON RELATIONAL DUALITY VIEW アクセスのオーバーヘッド
  - テーブルモデル

	ExTable1	ExTable2	ExTable3
更新データ量	320	820	1820
更新列数	6	11	21

  - 測定環境

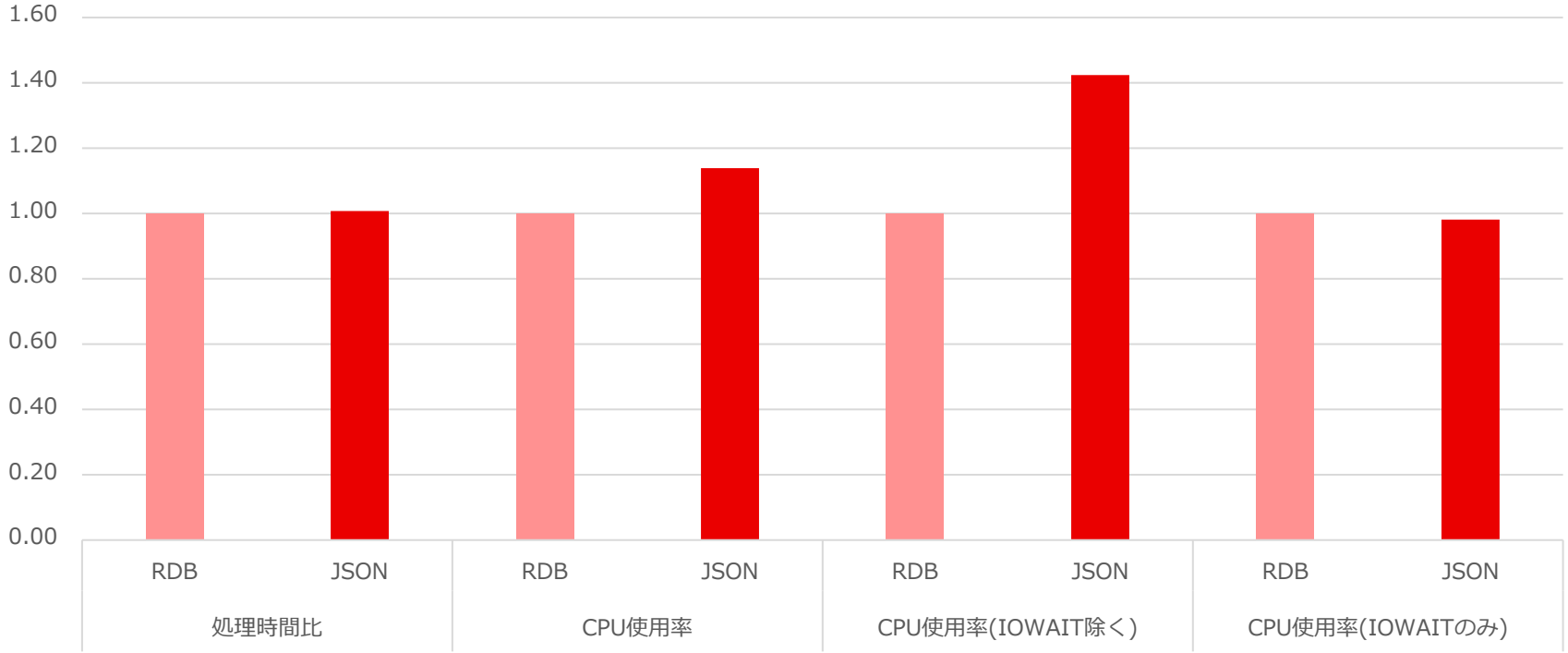
	VM環境	ソフトウェア
アプリケーション側	CPU: 2 MEM: 8GB DISK: 100GB	OS: RHEL 8.4 Oracle Java SE 17.0.1 Oracle JDBC Driver 23.2.0.0.0 (ojdbc11.jar)
DB側	CPU: 2 MEM: 4GB DISK: 50GB	OS: RHEL 8.6 Oracle Database 23c Enterprise Edition

ベンチマーク構成



結果

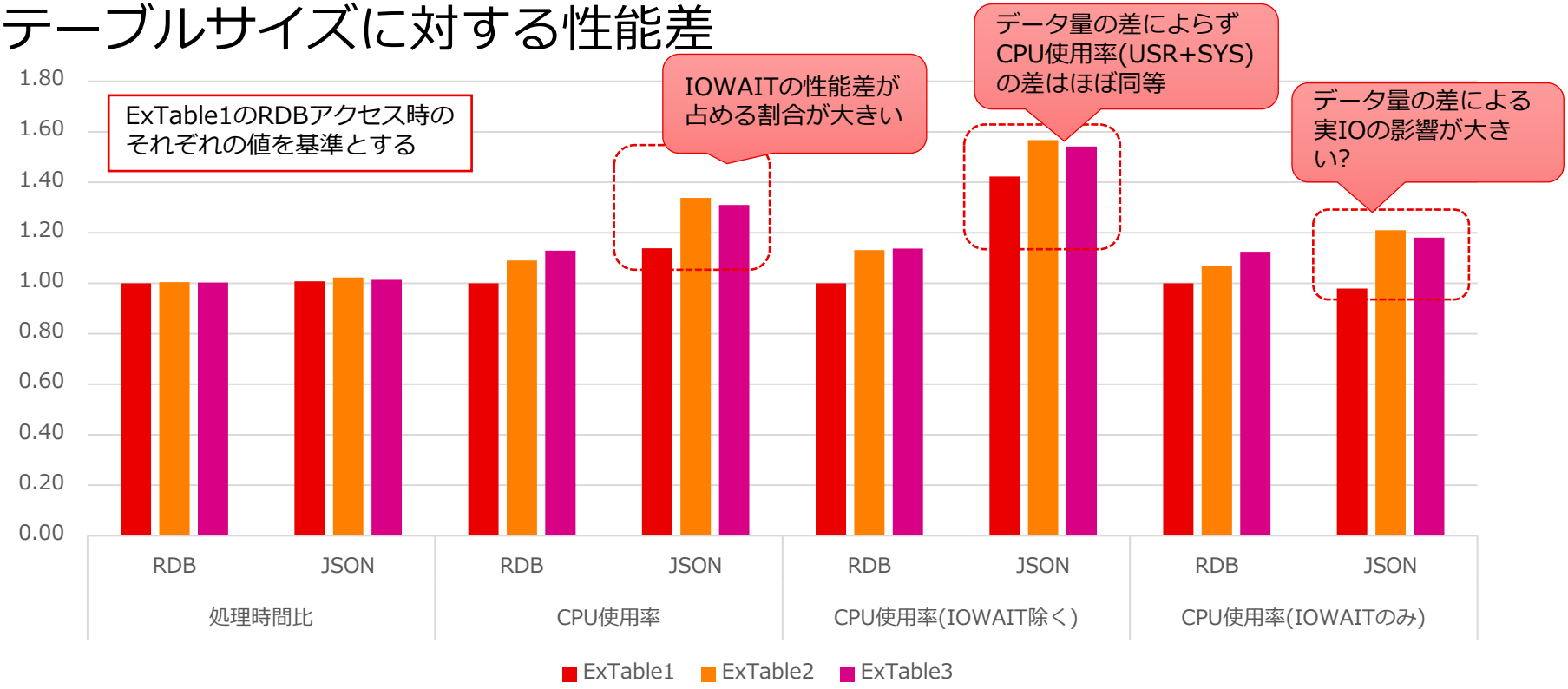
- UPDATEに対する性能差 (ExTable1)



結果

FUJITSU

- テーブルサイズに対する性能差



まとめ

- JSON RELATIONAL DUALITY VIEW利用のメリット
  - マイクロサービスなどでJSONデータを利用する場合、JSONデータを加工せずにDBにリクエストを送ることが可能となる。（アプリ実装量の削減が可能）
  - テーブル構造を直接参照しないことから、DB側の仕様変更の影響を受けづらくできると考えられる。
- JSON RELATIONAL DUALITY VIEW利用のデメリット
  - テーブルサイズの大小にかかわらず、CPU使用率（USR+SYS）は平均して5%前後高くなる。ただし、テーブルサイズの増大によるIO負荷より影響は微小と考える。
  - DB側でJSON RELATIONAL DUALITY VIEW定義の追加が必要になる。

参考

FUJITSU

- Database / Oracle / Oracle Database / Release 23 JSON-Relational Duality Developer's Guide
  - https://docs.oracle.com/en/database/oracle/oracle-database/23/jsnvu/introduction-car-racing-duality-views-example.html

- （オラクルエンジニア通信 - 技術資料、マニュアル、セミナー）JSONとリレーショナルの二面性: ドキュメント、オブジェクトおよびリレーショナル・モデルの革新的な統合
  - https://blogs.oracle.com/oracle4engineer/post/ja-json-relational-duality-app-dev

Thank you