

ORACLE®

Oracle Database 12c Release 2 CoreTech Seminar

12.2.0.1
Big Data

日本オラクル株式会社
クラウド・テクノロジー事業統括
Cloud/Big Data/DISプロダクト本部
立山 重幸
2016/10

リファレンス Oracle Database 12c Release 2 マニュアル・ドキュメント

- Oracle Database 12c Release 2(12.2) New Features
 - Big Data管理システム・インフラストラクチャ
http://docs.oracle.com/cd/E82638_01/NEWFT/GUID-6213395A-CFDD-4D42-8970-F8FA062CC4FA.htm#GUID-08A71091-F01B-4B6B-B64D-7C10D1846EB3

リファレンス Oracle Big Data SQL 3.1 マニュアル・ドキュメント

- Oracle Big Data SQL Online Documentation Library Release 3.1
 - Overview
 - <http://docs.oracle.com/bigdata/bds31/index.html>
- Big Data SQL Installation Guide
 - Changes in Oracle Big Data SQL 3.1
 - <http://docs.oracle.com/bigdata/bds31/BDSIG/preface.htm#GUID-012827C4-FE2A-4946-9588-D5D210C19661>

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Agenda

- 1 ▶ オラクルのビッグデータ戦略
- 2 ▶ レベルセッティング (Big Data SQLのデモ)
- 3 ▶ Oracle DB 12.2 & Big Data SQL 3.1 ビッグデータ関連ハイライト

1. オラクルのビッグデータ戦略

お客様における近年の状況



Graph



Python



node.js



R



Java



REST



開発環境及びコネクタの拡大

データマネジメントコンポーネントの拡大



Oracle顧客における近年の状況

オラクルは
どのように適合するのか



どのよう



データ統合における2つの考え方

JSONの例

データを統合

アプリケーション



SQL



分析

Oracle DBをJSON Store
として利用

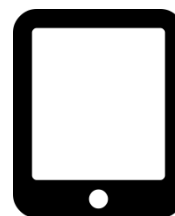
- ・柔軟な開発
 - 容易なフォーマット変更
 - 視認性の良さ

Oracle DBのJSON
データをSQLで利用

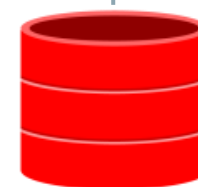
- ・柔軟な分析

データマネジメントを統合

アプリケーション



Big Data SQL



SQL



分析

NOSQLをJSON Store
として利用

- ・柔軟な開発

NOSQLのJSON データをSQLで利用

- ・柔軟な分析

データマネジメントを統合

One Fast, Secure SQL Queries over All your Data



Graph



Python



node.js



R



Java



REST



Java Database Connectivity

Oracle SQL



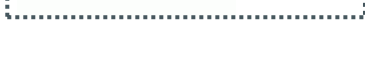
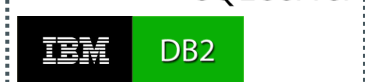
NoSQL

ORACLE[®]
NOSQL DATABASE



ORACLE

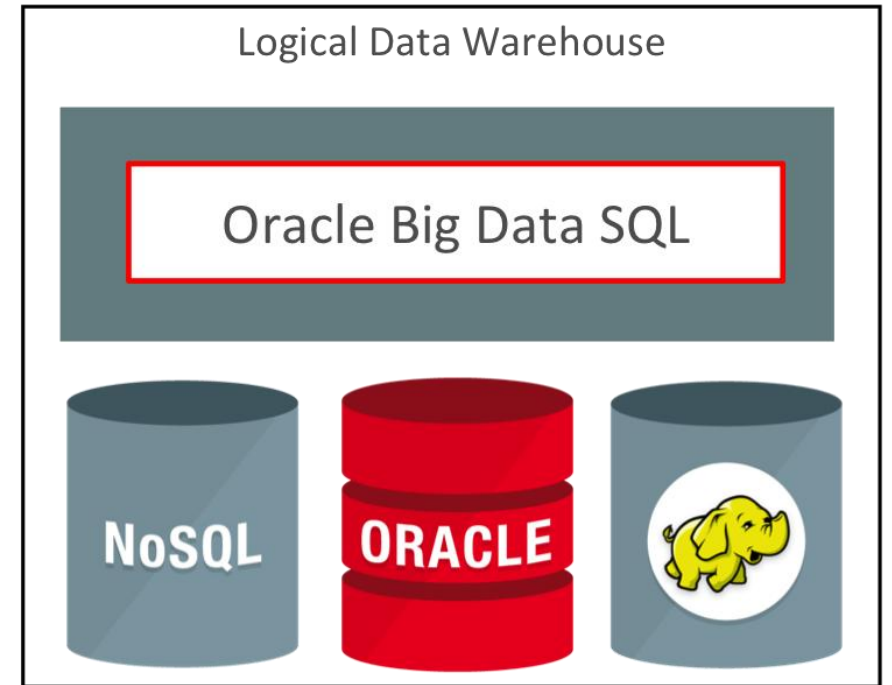
Future



Oracleは、ガートナが提唱するロジカル・データウェアハウスを実現

企業が現在必要としているのは、種別や形式が多様化した社内、社外のデータを従来型の社内データと併せて管理／処理することのできるデータ分析管理ソリューションです。

データには、インタラクション・データやIoT (Internet of Things) の各種センサー等からの観測データが含まれる可能性があります。こうしたデータも分析可能であることが当該市場のソフトウェアの新たな要件になっており、顧客は既存のエンタープライズ・データウェアハウス戦略を大きく超越する機能を求めています。



ロジカルデータウェアハウスの差別化要因

1. DWHのための最先端のリレーショナルデータベース

- DWHシェアNo1
- インメモリ技術

2. Big Data 製品との統合

- 幅広いポートフォリオ: HW, data management, tools, applications
- オンプレミス、クラウドを同じテクノロジーで実現

3. Oracle SQLでDWHやBig dataエコシステムに横断的にアクセス

- Hadoop, NoSQLやOracleデータベースなどの様々なデータにアクセス可能
- Exadata Storage Serverの技術を流用し高い性能を発揮

3.レベルセッティング (Big Data SQLのデモ)

新機能の前におさらい



SQL access to all your data using Big Data SQL

<http://www.oracle.com/technetwork/database/bigdata-appliance/oracle-bigdatalite-2104726.html>

Access

Secure

Analyze

Access Any Data



Personalized recommendations (NoSQL)

Clicks, ratings & comments (JSON on Hadoop)

Revenue from purchases (Oracle Relational Data Warehouse)

Example: [Click data](#) in its raw format.

Create Big Data SQL Enabled Tables

Query Complex Sources

このデモは、オンラインで映画を配信しているサイトを題材にしています。

売上や注文データはOracleDBにあります、
クリック履歴は、JSONフォーマットでHadoopに
レコメンド情報は、Key-ValueフォーマットでNoSQLに
格納されています。

File Browser

ACTIONS

View as binary

Download

View file location

Refresh

INFO

Last modified

Dec. 30, 2014 8:25 a.m.

User

oracle

Group

oracle

Size

31.0 MB

Mode

100644

Home

Page 1 to 1 of 7949



/ user / oracle / moviework / applog_json / movieapp_log_json.log

```

{"custid":1185972,"movieid":null,"genreid":null,"time":"2012-07-01:00:00:07","recommended":null,"activity":8}
{"custid":1354924,"movieid":1948,"genreid":9,"time":"2012-07-01:00:00:22","recommended":"N","activity":7}
{"custid":1083711,"movieid":null,"genreid":null,"time":"2012-07-01:00:00:26","recommended":null,"activity":9}
{"custid":1234182,"movieid":11547,"genreid":6,"time":"2012-07-01:00:00:32","recommended":"Y","activity":7}
{"custid":1010220,"movieid":11547,"genreid":6,"time":"2012-07-01:00:00:42","recommended":"Y","activity":6}
{"custid":1143971,"movieid":null,"genreid":null,"time":"2012-07-01:00:00:43","recommended":null,"activity":8}
{"custid":1253676,"movieid":null,"genreid":null,"time":"2012-07-01:00:00:50","recommended":null,"activity":9}
{"custid":1351777,"movieid":608,"genreid":6,"time":"2012-07-01:00:01:03","recommended":"N","activity":7}
{"custid":1143971,"movieid":null,"genreid":null,"time":"2012-07-01:00:01:07","recommended":null,"activity":9}
{"custid":1363545,"movieid":27205,"genreid":9,"time":"2012-07-01:00:01:18","recommended":"Y","activity":7}
{"custid":1067283,"movieid":1124,"genreid":9}
{"custid":1126174,"movieid":16309,"genreid":
{"custid":1234182,"movieid":11547,"genreid":
{"custid":1067283,"movieid":null,"genreid":n
{"custid":1377537,"movieid":null,"genreid":n
{"custid":1347836,"movieid":null,"genreid":n
{"custid":1137285,"movieid":null,"genreid":n
{"custid":1354924,"movieid":null,"genreid":n
{"custid":1036191,"movieid":null,"genreid":n
{"custid":1363545,"movieid":27205,"genreid":s
{"custid":1273464,"movieid":null,"genreid":null,
{"custid":1346299,"movieid":424,"genreid":18,"time":"2012-07-01:00:05:02","recommended":"Y","activity":4}
{"custid":1399170,"movieid":null,"genreid":null,"time":"2012-07-01:00:05:34","recommended":null,"activity":8}

```

Hadoopに入っている、クリック履歴のJSONファイルです。
 様々な顧客が様々な映画に対して、様々なアクションをしたログが記載されています。

Access

Secure

Analyze

▶ Access Any Data

▼ Create Big Data SQL Enabled Tables

クリック履歴のHDFSにアクセスするためのOracleDDL

```
CREATE TABLE movielog_t
(click VARCHAR2(4000))
ORGANIZATION EXTERNAL
(TYPE ORACLE_HDFS
DEFAULT DIRECTORY default_dir
LOCATION ('/user/oracle/moviework/appllog_json/'))
REJECT LIMIT UNLIMITED;
```

レコメンドのNoSQLにアクセスするためのOracleDDL

```
CREATE TABLE recommendation_t
custid NUMBER,
sno NUMBER,
genreid NUMBER,
movieid NUMBER )
ORGANIZATION EXTERNAL
(
TYPE ORACLE_HIVE
DEFAULT DIRECTORY default_dir
ACCESS PARAMETERS
( com.oracle.bigdata.tablename:moviework.recommendation )
)
REJECT LIMIT UNLIMITED;
```

▶ Query Complex Sources

先ほどのHadoop(HDFS)上のファイルに対して、ORACLE_HDFSアクセスドライバを使って、外部表を定義する事で、Oracle DBから簡単にアクセスする事が可能になります。

もちろん、NoSQLについても同様です。

これがBig Data SQLです。

Access

Secure

Analyze

▶ Access Any Data

▶ Create Big Data SQL Enabled Tables

▼ Query Complex Sources

Query Clicks (JSON)

```
SELECT
  m.click.custid,
  m.click.movieid,
  m.click.genreid,
  m.click.time
FROM movielog_t m
WHERE rownum < 20;
```

Custid	Movieid	Genreid	Time
1185972	-	-	2012-07-01:00:00:07
1354924	1948	9	2012-07-01:00:00:22
1083711	-	-	2012-07-01:00:00:00
1234182	11547	6	2012-07-01:00:00:00
1010220	11547	6	2012-07-01:00:00:00
1143971	-	-	2012-07-01:00:00:00
1253676	-	-	2012-07-01:00:00:00
1351777	608	6	2012-07-01:00:00:00
1143971	-	-	2012-07-01:00:00:00
1363545	27205	9	2012-07-01:00:00:00
1067283	1124	9	2012-07-01:00:01:20
1126174	16309	46	2012-07-01:00:01:35

Query Recommendations (NoSQL DB)

```
SELECT *
FROM recommendation_t
WHERE rownum <= 20;
```

Custid	Movieid	Genreid	Sno
1024438	121	11	1
1024438	196	11	1
1024438	435	11	1
1024438	10911	11	1

作成した外部表(movielog_t)に対して、クエリーを実行できます。

Oracle Database12cの機能により、JSONファイルをクエリ時にパースできます。

※BDSに限らずDBをJSONストアとして利用した場合でも同様のJSONクエリは可能です。

⇒詳細はDay2のDev Toolセッションご参照

Access

Secure

Analyze

▶ Oracle Data Redaction Over Customers Table

▼ Define Redaction Policy Over Data in HDFS and NoSQL DB





```
BEGIN
-- JSON file in HDFS
DBMS_REDACT.ADD_POLICY (
  object_schema => 'MOVIEDEMO',
  object_name => 'MOVIELOG_V',
  column_name => 'CUSTID',
  policy_name => 'movielog_v_redaction',
  function_type => DBMS_REDACT.PARTIAL,
  function_parameters => '9,1,7',
  expression => '1=1' );

-- Recommendations data from Oracle NoSQL Database
DBMS_REDACT.ADD_POLICY(
  object_schema => 'MOVIEDEMO',
  object_name => 'RECOMMENDATION',
  column_name => 'CUSTID',
  policy_name => 'recommendation_redaction',
  function_type => DBMS_REDACT.PARTIAL,
  function_parameters => '9,1,7',
  expression => '1=1' );
END;
/
```

▶ Query Redacted Data

リダクションの設定も可能です。

Access

Secure

Analyze

▶ Oracle Data Redaction Over Customers Table

▶ Define Redaction Policy Over Data in HDFS and NoSQL DB

▼ Query Redacted Data

Clicks

Custid	Movieid	Genreid	Time
9999999	-	-	2012-07-01:00:00:07
9999999	1948	9	2012-07-01:00:00:22
9999999	-	-	2012-07-01:00:00:26
9999999	11547	6	2012-07-01:00:00:32
9999999	11547	6	2012-07-01:00:00:42
9999999	-	-	2012-07-01:00:00:43
9999999	-	-	2012-07-01:00:00:50
9999999	608	6	2012-07-01:00:01:03
9999999	-	-	2012-07-01:00:01:07
9999999	27205	9	2012-07-01:00:01:18
9999999	1124	9	2012-07-01:00:01:26
9999999	16309	46	2012-07-01:00:01:32
9999999	11547	6	2012-07-01:00:01:38
9999999	-	-	2012-07-01:00:01:44
9999999	-	-	2012-07-01:00:01:50

Recommendations

Custid	Movieid	Genreid	Sno
9999999	121	11	1
9999999	196	11	1
9999999	435	11	1
9999999	488	11	1
9999999	857	11	1
9999999	956	11	1
9999999	1273	11	1
9999999	8367	11	1
9999999	9346	11	1
9999999	9836	11	1
9999999	10539	11	1

リダクションされたcustid

もちろん、他の表とJOINも可能です。

(補足) クリック履歴+売上 × SQLでどんな分析できる？

- 離反しそうな重要顧客を探す(RFM分析)

項目	ソース	適用する関数
Recency: 顧客が最後にサイトにアクセスした日時	クリック履歴	NTILE (5) over (order by max(time))
Frequency: サイト上での顧客のアクティビティ・レベル	クリック履歴	NTILE (5) over (order by count(1))
Monetary: 顧客が消費した金額	売上	NTILE (5) over (order by sum(sales))

抽出条件(R:F:M)
5:5:1

自社のサイトで下見をした後に他社のサイトで購入していると思われる顧客群

抽出条件(R:F:M)
2以下:x:4以上

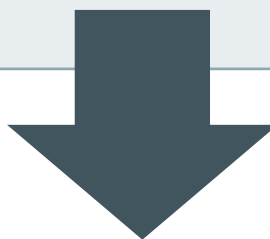
最近サイトにアクセスしていない重要顧客
(Recencyのスコアが低く、Monetaryのスコアが高い)

CU	RFM_MONETARY
11	4 5
12	4 5
11	4 5
11	5 5

(補足) クリック履歴+売上 × SQLでどんな分析できる？

- 離反しそうな重要顧客を探す(RFM分析)

項目	ソース	適用する関数
Recency: 顧客が最後にサイトにアクセスした日時	クリック履歴	NTILE (5) over (order by max(time))
Frequency: サイト上での顧客のアクティビティ・レベル	クリック履歴	NTILE (5) over (order by count(1))
Monetary: 顧客が消費した金額	売上	NTILE (5) over (order by sum(sales))



これは、ユーザー毎のレコード数をカウントして、5つのグループに分けている

↓
クリック回数が多いユーザーが上位に来てしまう(クリックは少ないけど、毎日来てくれる人が埋もれてしまう)

↓
あるユーザーが何回(セッション)アクセスしてくれたかを割り出す必要がある

(補足) クリック履歴+売上 × SQLでどんな分析できる？

- ユーザー毎のセッション数と滞在時間(パターンマッチング)

クリック履歴

ユーザ	アクション	時間
A	ログイン	7/1 10:00
B	ログイン	7/1 10:15
A	クリック	7/1 10:16
A	クリック	7/2 10:02
A	クリック	7/2 10:05
C	クリック	7/2 10:10
...		

ユーザの行動分析

ユーザ	セッションID	滞在時間
A	1	16分
A	2	3分
B	1	xxx
...		

ビジネスルール
クリック間が2時間以上空くものは、
別セッションとする
(映画見ている間はクリックされない)

(補足) クリック履歴+売上 × SQLでどんな分析できる？

- ユーザー毎のセッション数と滞在時間(パターンマッチング)

MATCH_RECOGNIZE関数

タスク	キーワード	説明
1.データを体系化する	PARTITION BY	行を論理的にグループに分割/パーティション化する
	ORDER BY	パーティション内で行を論理的に順序付ける
	PATTERN	照合する必要があるパターン変数、照合する必要があるシーケンス、および照合する必要がある行数を定義する
2.ビジネス・ルールを定義する	DEFINE	パターン変数を定義する条件を指定する
	AFTER MATCH	一致が見つかった後にマッチング・プロセスを再開する場所を決定する
3.出力メジャーの定義	MEASURES	行パターン・メジャー列を定義する
	MATCH_NUMBER	パターン変数を適用する行を見つける
	CLASSIFIER	特定の行に適用するパターンの要素を特定する
4.出力の制御	ONE ROW PER MATCH	各一致の出力のサマリー行を返す
	ALL ROWS PER MATCH	各一致の行ごとにディテール行を1つ返す

(補足) クリック履歴+売上 × SQLでどんな分析できる？

- ユーザー毎のセッション数と滞在時間(パターンマッチング)

MATCH_RECOGNIZE関数

1.データを体系化する

顧客ID単位で束にして、時系列にソート

3.出力メジャーの定義

セッション毎にナンバリング'MATCH_NUMBER()'して、カウントや時間差異を定義

4.出力の制御

セッションID毎にサマリーするか(ONE ROW)、全件出力するか(ALL ROWS)

2.ビジネス・ルールを定義する

次の行の時間が2時間以内なら同一セッションとする

```
SELECT
  hoge
FROM hoge hoge
MATCH_RECOGNIZE
  (PARTITION BY cust_id ORDER BY time_id
  MEASURES MATCH_NUMBER() AS session_id,
  COUNT(*) AS no_of_events,
  TO_CHAR(FIRST(bgn.time_id),'hh24:mi:ss') AS start_time,
  TO_CHAR(LAST(sess.time_id),'hh24:mi:ss') AS end_time,
  TO_CHAR(to_date('00:00:00','HH24:MI:SS') + (LAST(sess.time_id)-
  FIRST(bgn.time_id)),'hh24:mi:ss') AS mins_duration
  ONE ROW PER MATCH
  PATTERN (bgn sess+)
  DEFINE
    sess as time_id <= PREV(sess.time_id) + interval '2' hour
  )
```

(補足) クリック履歴+売上 × SQLでどんな分析できる？

- ユーザー毎のセッション数と滞在時間(パターンマッチング)

実行結果

CUST_ID	SESSION_ID	NO_OF_EVENTS	START_TIME	END_TIME	MINS_DURATION
1000050	1	2	08:10:57	08:20:01	00:09:04
1000050	2	4	13:40:07	13:50:12	00:10:05
1000083	1	14	08:49:22	13:23:30	04:34:08
1000186	1	9	22:38:43	23:28:06	00:49:23
1000299	1	2	20:14:49	20:22:55	00:08:06
1000299	2	2	17:07:25	17:16:05	00:08:40
1000386	1	3	19:06:04	19:20:04	00:14:00
1000498	1	6	22:58:09	23:18:01	00:19:52
1000672	1	7	23:46:19	00:04:35	00:18:16
1000672	2	4	05:31:01	05:46:52	00:15:51
1000672	3	17	16:28:38	17:58:12	01:29:34
1000679	1	14	00:34:03	01:37:39	01:03:36
1000679	2	4	09:47:49	10:08:31	00:20:42

クリック履歴を分析に使うための前準備が完了した状態

↓

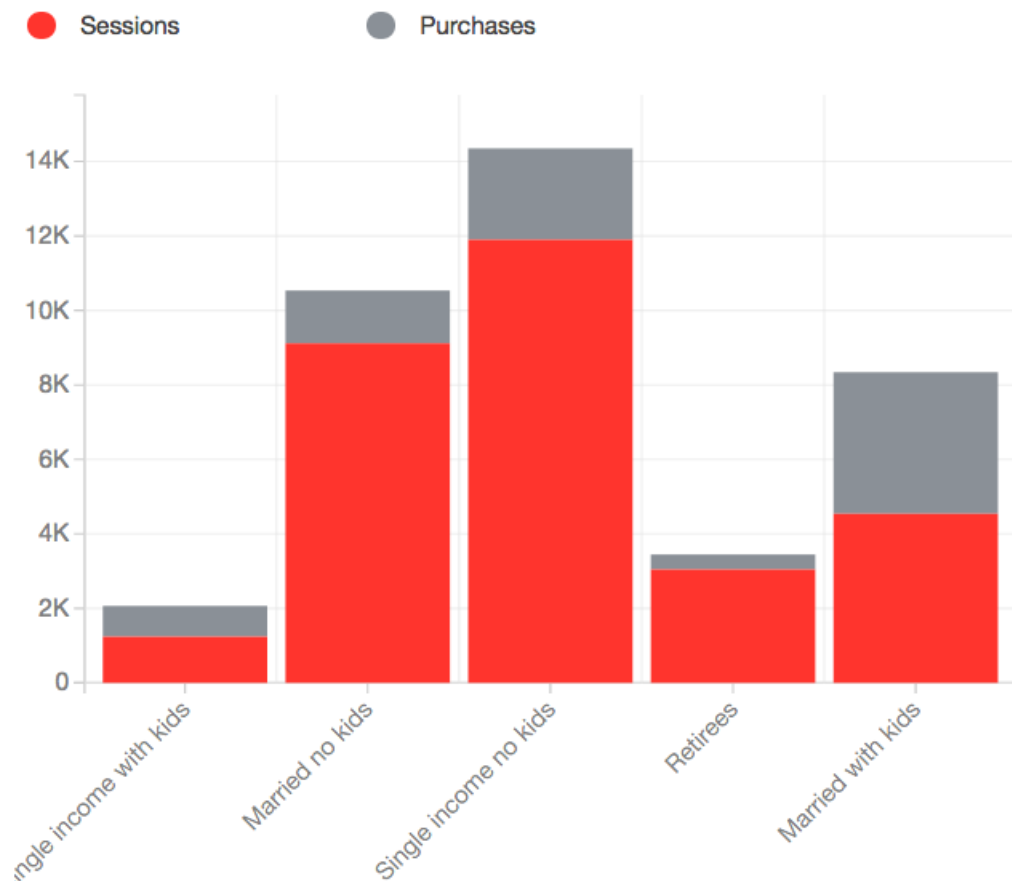
ここから、真にFrequencyが高いユーザーを評価できる

Access

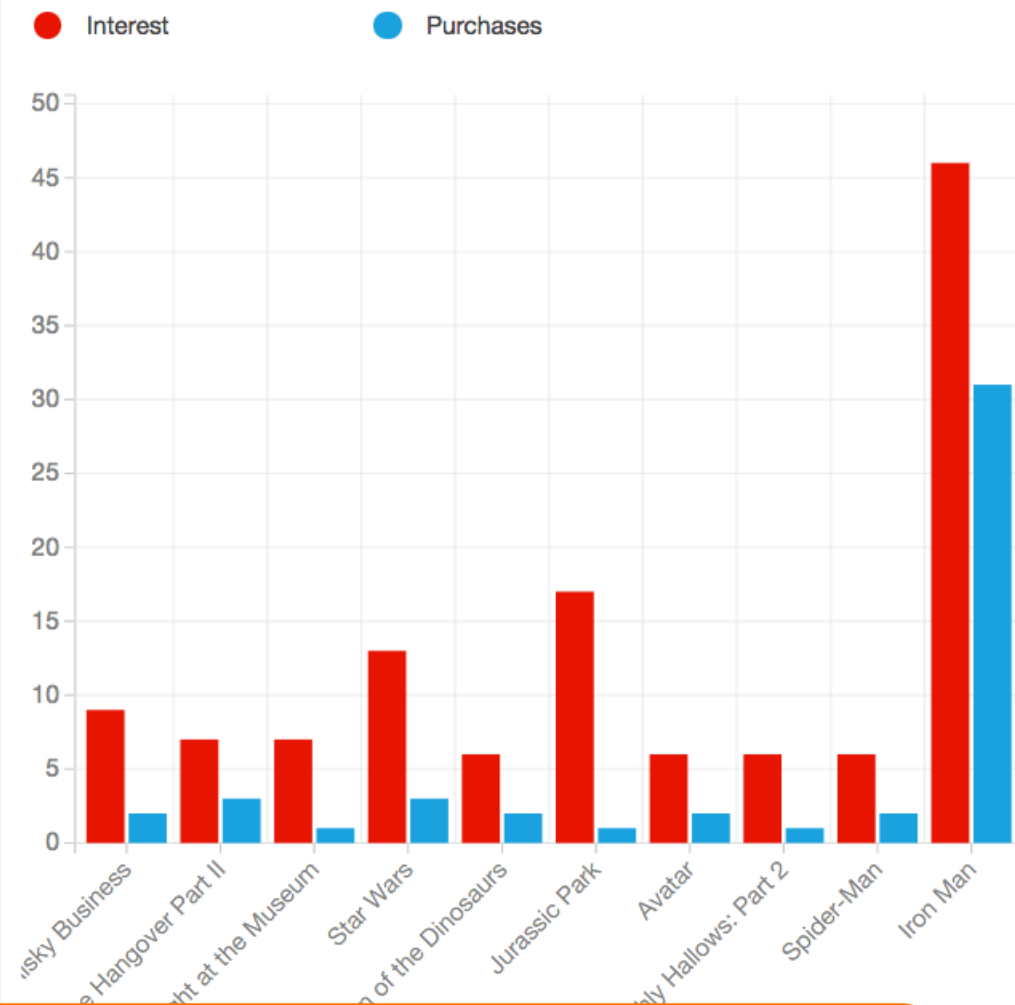
Secure

Analyze

Session Conversions



Recommendation Effectiveness



ユーザー毎のセッション数と売上が紐付けられるようになるため、顧客マスタと照合し、顧客属性別の行動分析などが可能になります。

(補足) その他、ビッグデータ関連で便利な12cの関数

- APPROX_COUNT_DISTINCT()
 - ビッグデータの大まかな把握(数%の精度誤差)
 - 通常のCount Distinctより5～50倍高速

- FETCH句のPERCENT
 - Top N%のデータを抽出
 - ...
ORDER BY sales DESC
FETCH FIRST 1 PERCENT ROWS ONLY;

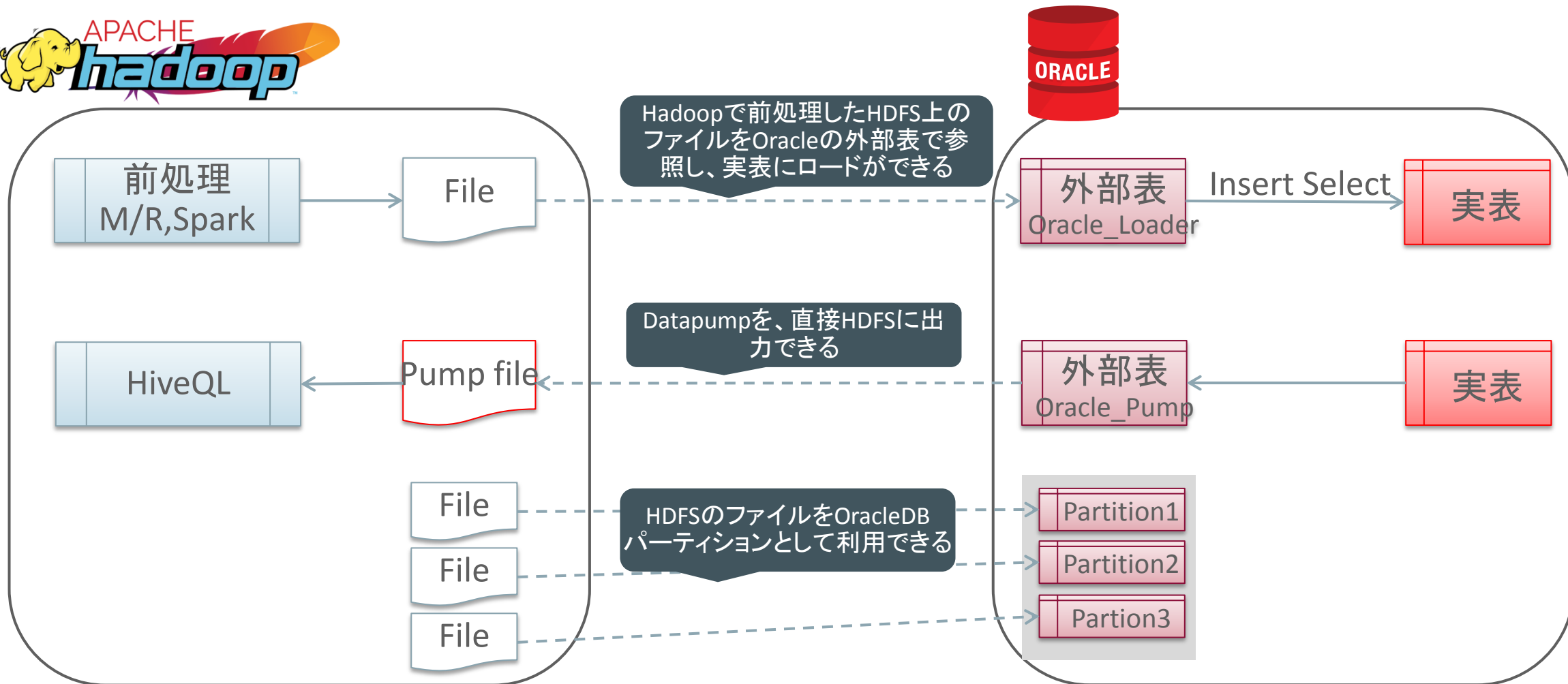
3.Oracle DB 12.2 & Big Data SQL 3.1 ビッグデータ関連ハイライト

Oracle DB 12.2

ビッグデータに関連するDB12.2の新機能



- **External Tables Can Access Data Stored in Hadoop Data Sources Including HDFS and Hive**
 - Oracle DBの外部表からHDFSとHive表にアクセス可能
 - ORACLE_LOADER , ORACLE_DATAPUMP Access Driverを利用
- **ORACLE_DATAPUMP and ORACLE_LOADER Access Driver Supports New File Format**
 - ORACLE_DATAPUMPアクセスドライバで直接HDFSにdata pump出力が可能
 - 従来のpumpは、出力最後にヘッダーのブロックを更新する使用であったためHDFS出力不可だった
- **Partitioned External Tables**
 - HiveやHDFSをOracleDBのパーティション外部表として利用可能

<図解> 何が出来るようになったのか？



<補足>

- Oracle LoaderのHDFSアクセスご利用のために、Big Data Connectorsのライセンスが必要になります。
- Oracle PumpのHDFSアクセスご利用のためには、Big Data SQLのライセンスが必要になります
 - 詳細は、Big Data SQL Users's Guide「4 Copying Oracle Tables to Hadoop」をご参照ください
- 前バージョンからの進化ポイント
 - Pumpを直接HDFSに書き出せるようになった

version	アンロードするまでの手順
DB 12.1 BDS 3.0	
DB 12.2 BDS 3.1	

(ご参考)ORACLE_DATAPUMP_ACCESS Driverを用いたHDFSへのアンロード例

```
CREATE TABLE inventories_xt  
ORGANIZATION EXTERNAL  
(  
  TYPE ORACLE_DATAPUMP  
  DEFAULT DIRECTORY def_dir1  
  LOCATION ('inv_xt.dmp')  
  ACCESS PARAMETERS (HADOOP_TRAILERS ENABLED)  
)  
AS SELECT * FROM inventories;
```

Create Directoryで、HDFSのパスを指定しておく

従来のpumpは、データ出力後に先頭のブロックを更新するという使用であったが、追記しかできないHDFSに対応するために、当オプションが用意された

件数確認

```
SQL> SELECT COUNT(*) FROM inventories_xt;
```

```
COUNT (*)
```

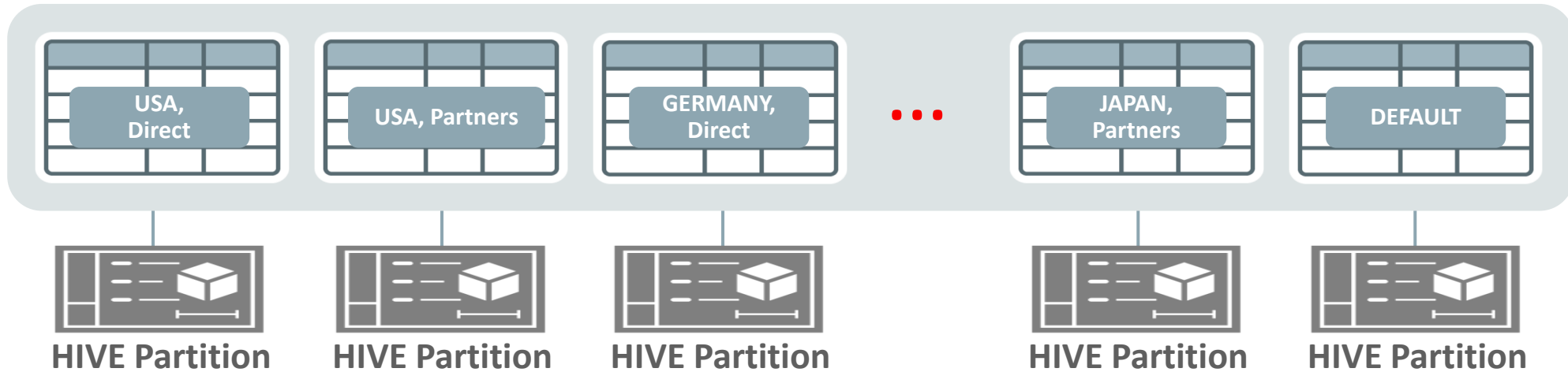
```
-----  
1112
```

元表と外部表の差異確認

```
SQL> SELECT * FROM inventories MINUS SELECT * FROM inventories_xt;
```

```
no rows selected
```

Oracle Partitioning: 外部表のパーティション化



- 全てのパーティション手法を利用して、外部表をパーティション化可能
 - 12.2で大幅強化されたパーティション機能の全貌はDay3のDB Coreセッションご参照
- パーティション・プルーニングとメンテナンス
 - パーティションの追加、削除、交換

Big Data SQL 3.1

Big Data SQL 3.1 ハイライト①

Oracle Database Tablespaces in HDFS

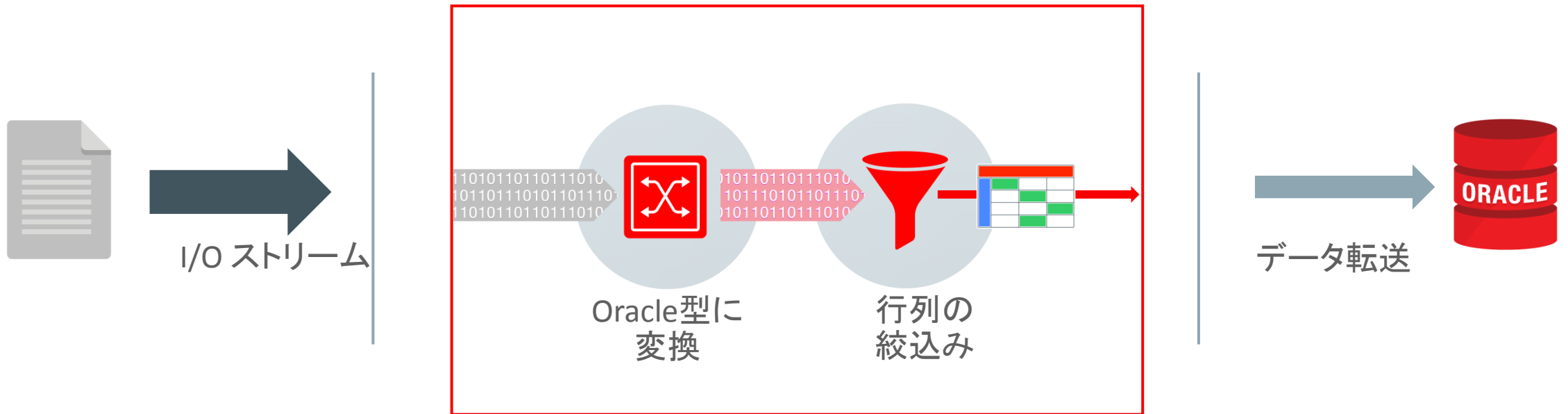
- OracleDBのテーブルスペースをHDFSに配置できます
- 従来の外部表ではなく、Read-Onlyの実表と同等に扱うことが可能です
- メリット
 - 圧倒的なクエリ性能の向上（パース不要、Index、In-Memory）
- デメリット
 - Oracleバイナリファイルのため、Hadoopの他のアプリケーションからは利用不可

比較: 従来の外部表方式 vs 実表形式

	外部表	実表
様々なフォーマット対応	Yes (text , parquet , ORC ,etc)	No (dbf)
データ活用	Oracle DB + Hadoopアプリ (BDSだけではなく、HiveやSparkなど からファイルを利用可能)	Oracle DB (BDS only)
索引、インメモリ分析など	No	Yes
データ管理	View	Table Partition
Smart Scan	Yes	Yes
Storage Indexes	Yes	Yes
Bloom Filters	Yes	Yes
Partition Pruning	Hive	Oracle
Predicate Pushdown	Yes	Yes

Big Data SQLの内部動作

Smart Scan

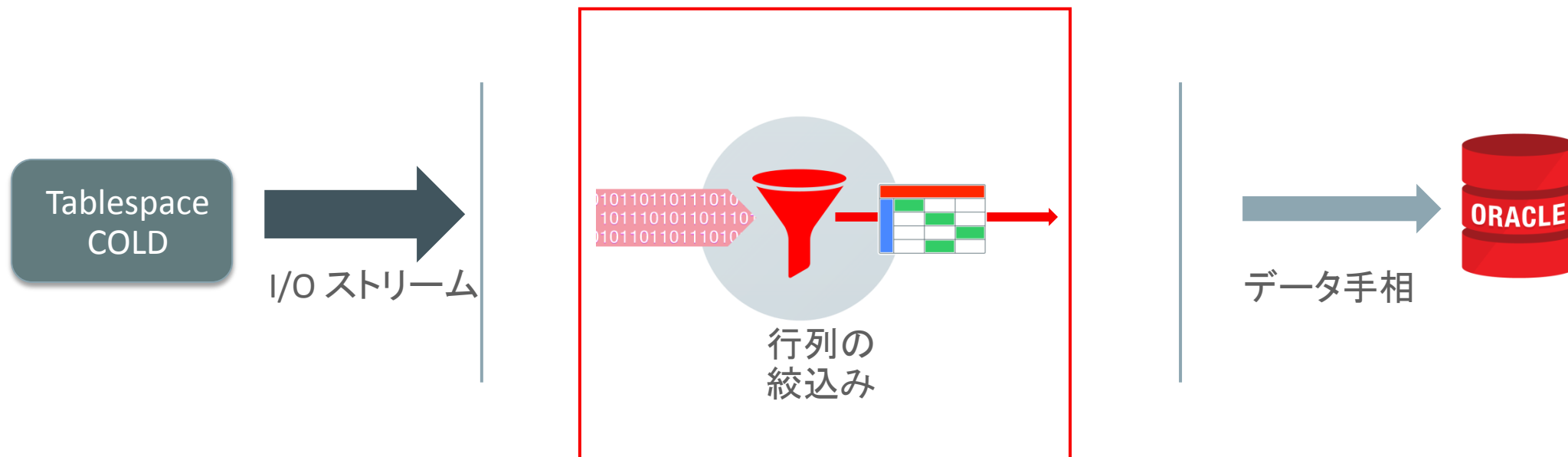


Big Data SQLの内部動作

Smart Scan on Oracle Tablespace in HDFS

NEW IN
3.1

Oracle型へのパースが発生しないため
パフォーマンスが向上



新機能を利用したソリューション

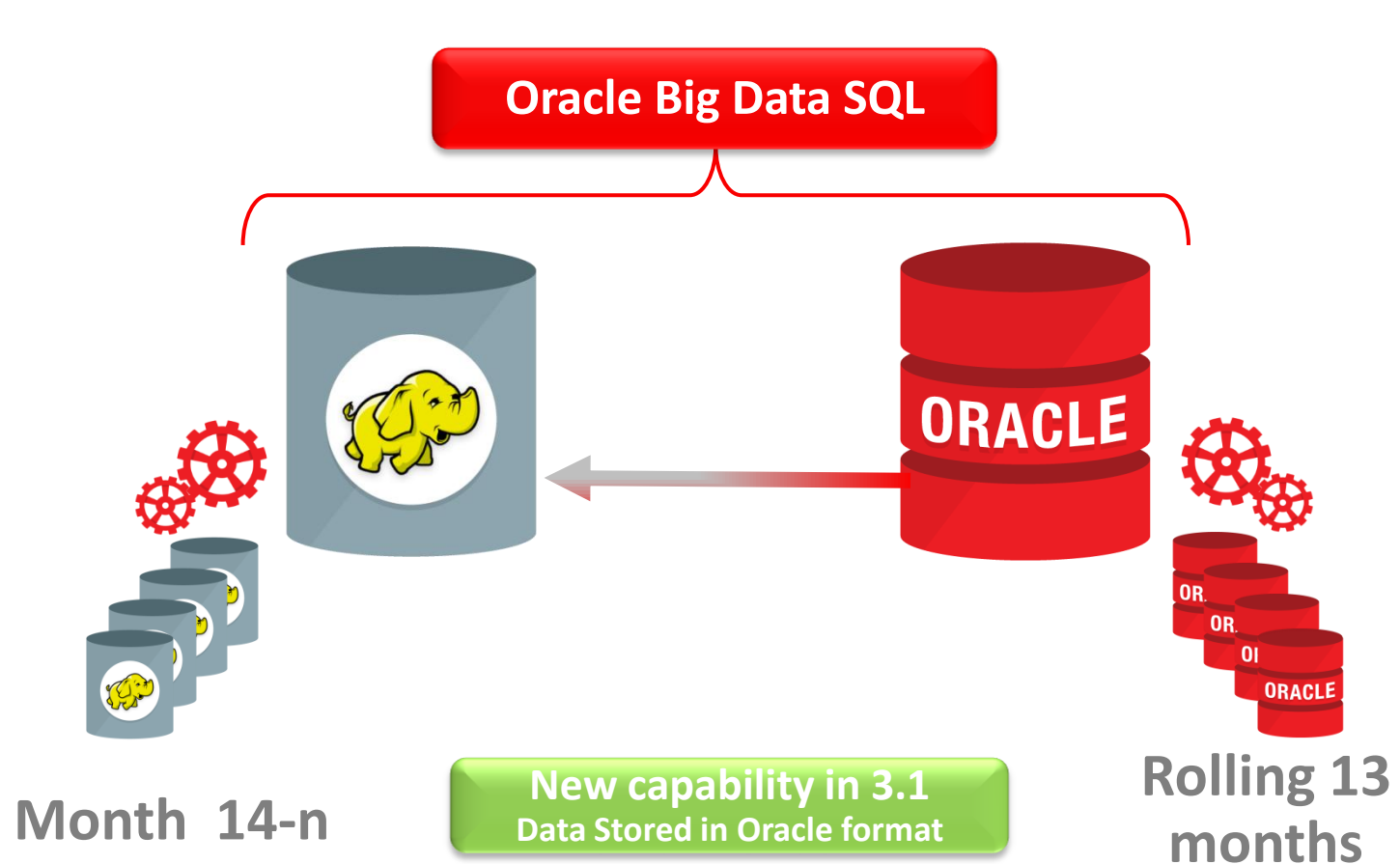
Active Archive for Long-Term Storage Savings

Put your cold data in Hadoop, retain the power of Oracle with Big Data SQL



Active Archive for Long-Term Storage Savings

より多くのデータを安価にSQL分析可能に



Big Data SQL 3.0 w/ DB12.1

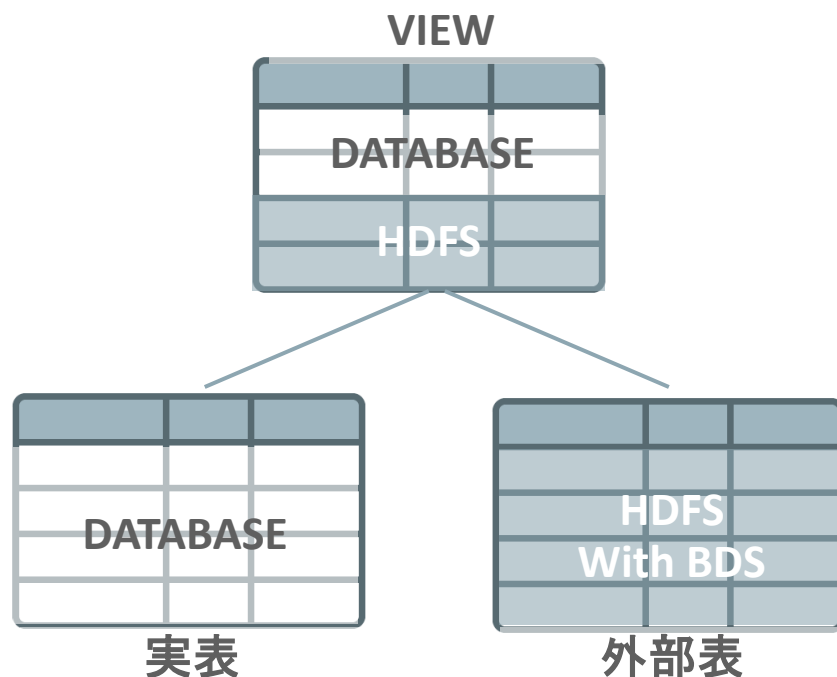
- HadoopフォーマットでHDFSにデータ保持
- Big Data SQLによりスキャンや絞込み処理をHadoop側にオフロード可能

Big Data 3.1^{NEW} w/ DB12.2

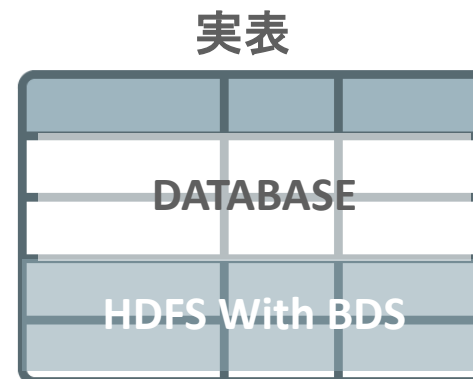
- **Oracleフォーマット**でHDFSにデータ保持
- dbfをネイティブに取り扱うため、パフォーマンスがさらに向上

Archive Data: Big Data SQL は2つの方式で実現できます

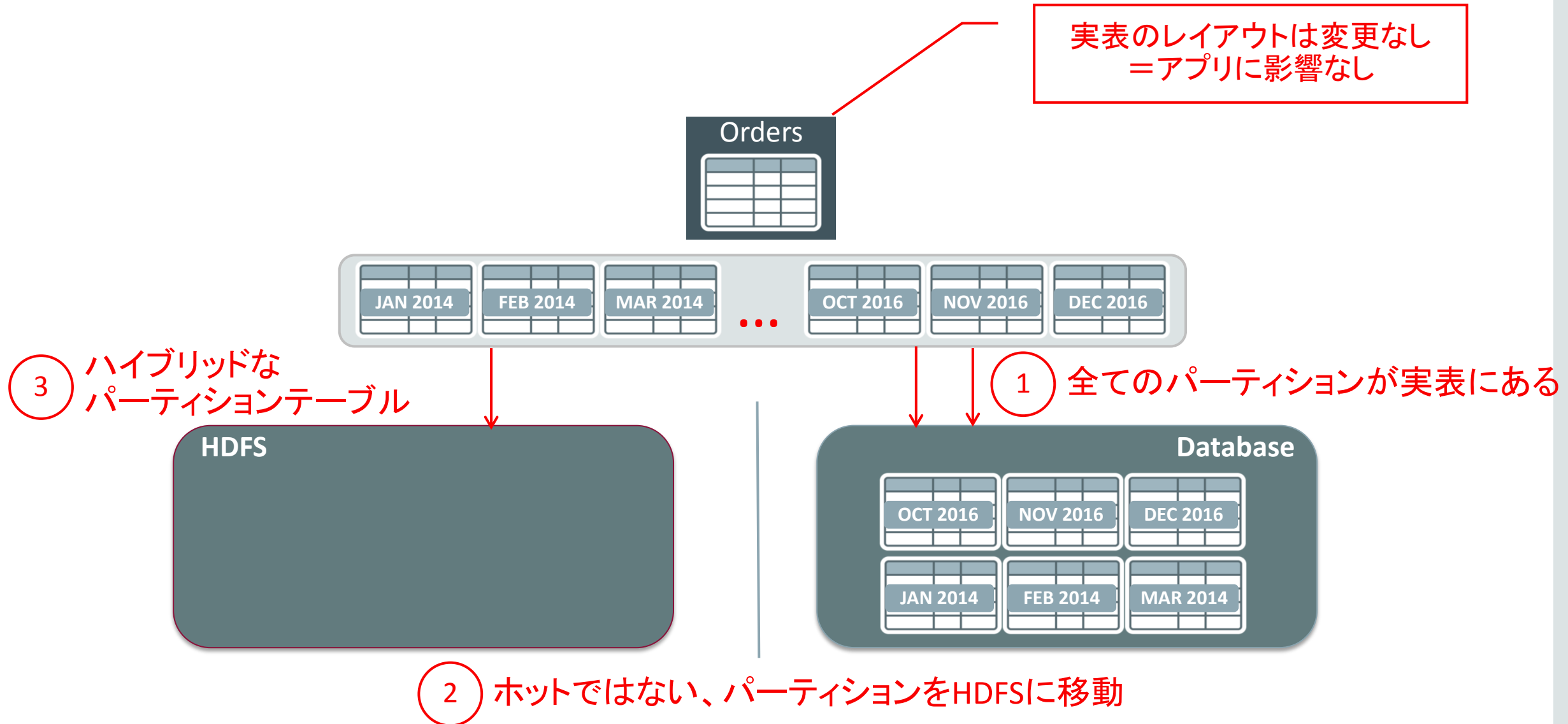
1. Viewによる統合(従来の方式)



2. ハイブリッドテーブル



ハイブリッドテーブルのイメージ



ハイブリッドテーブルのコード例

-- コールドテーブルスペースの作成

```
CREATE TABLESPACE movie_cold_hdfs DATAFILE '/movie_cold_hdfs1.dbf' SIZE 100M reuse AUTOEXTEND ON nologging;
```

-- 作成したテーブルスペースにテーブル配置

```
ALTER TABLE movie_fact MOVE PARTITION 2014_MAR TABLESPACE movie_cold_hdfs ONLINE UPDATE INDEXES;
```

-- テーブルスペースをリードオンリーに変更 (HDFSに格納する場合は、読み取り専用になります)

```
ALTER TABLESPACE movie_cold_hdfs READ ONLY;
```

-- テーブルスペースをオフラインに変更

```
ALTER TABLESPACE movie_cold_hdfs OFFLINE;
```

-- データファイルをHDFSにコピー

```
hadoop fs -put movie_cold_hdfs1.dbf /tablespaces/
```

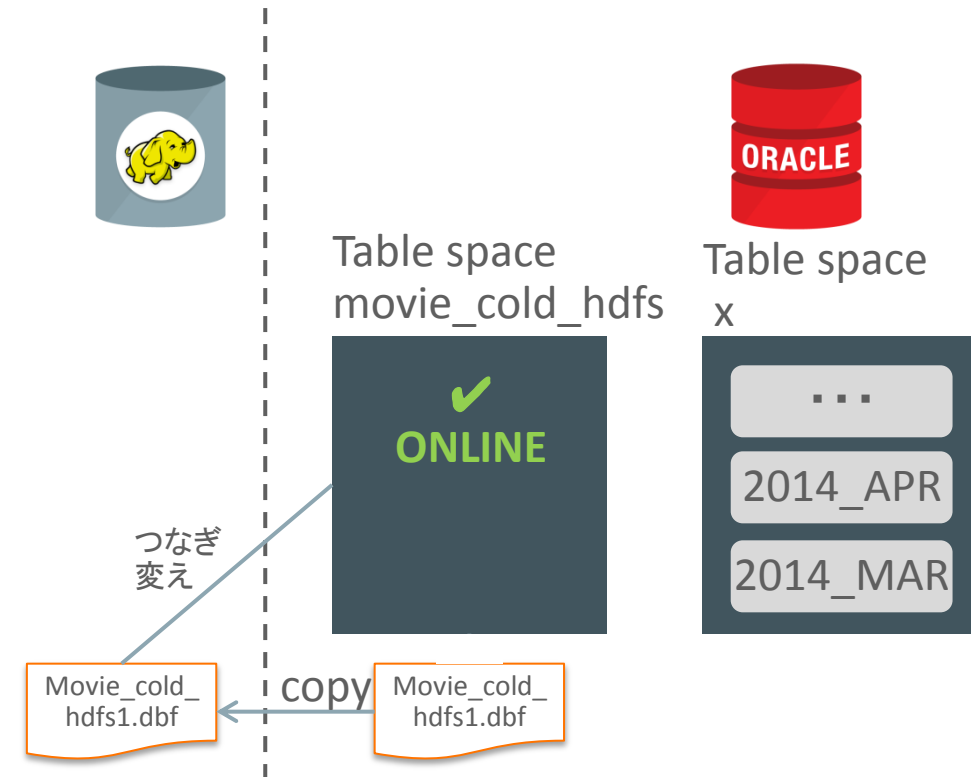
-- データファイルをつなぎ変える(DB->HDFSに変更) ※要NFSマウント

```
ALTER TABLESPACE movie_cold_hdfs RENAME DATAFILE 'movie_cold_hdfs1.dbf' TO '/mnt/hdfs:myclust/tablespaces/movie_cold_hdfs1.dbf';
```

-- テーブルスペースをオンラインに戻す

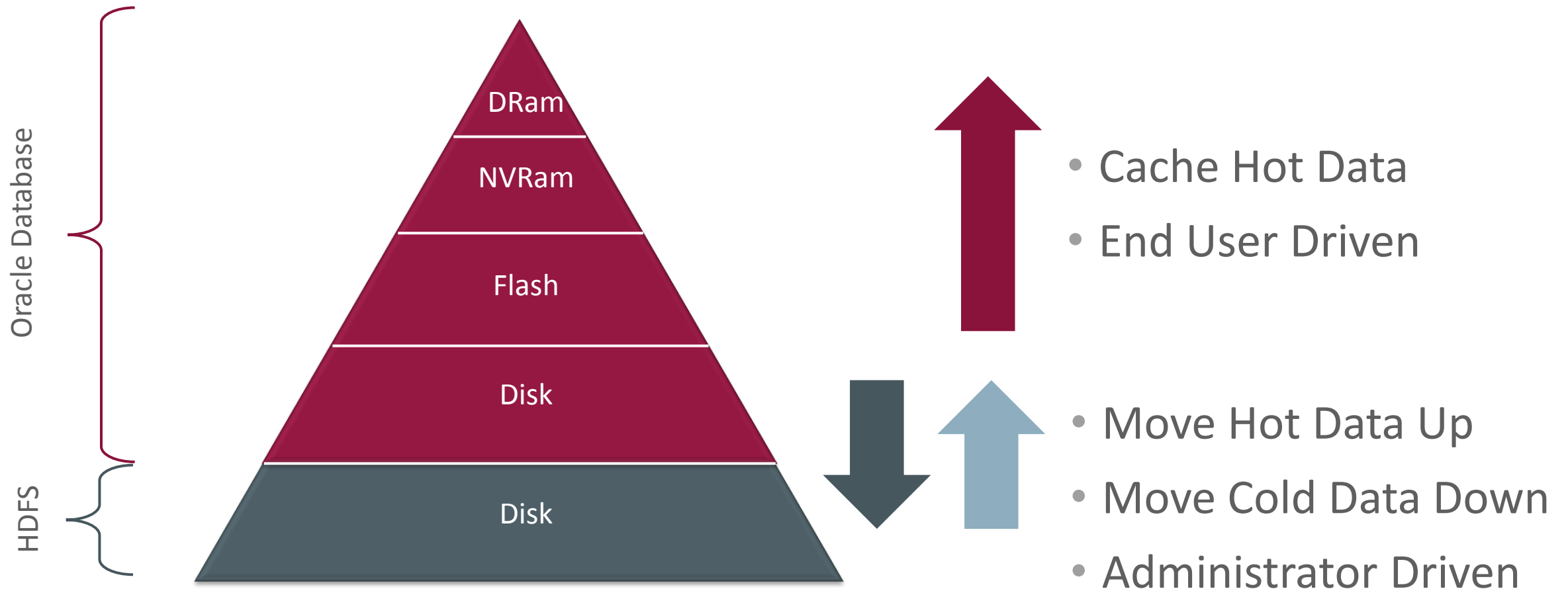
```
ALTER TABLESPACE movie_cold_hdfs ONLINE;
```

```
select avg(rating) from movie_fact where ... -- 元々と同じSQLが実行できます
```

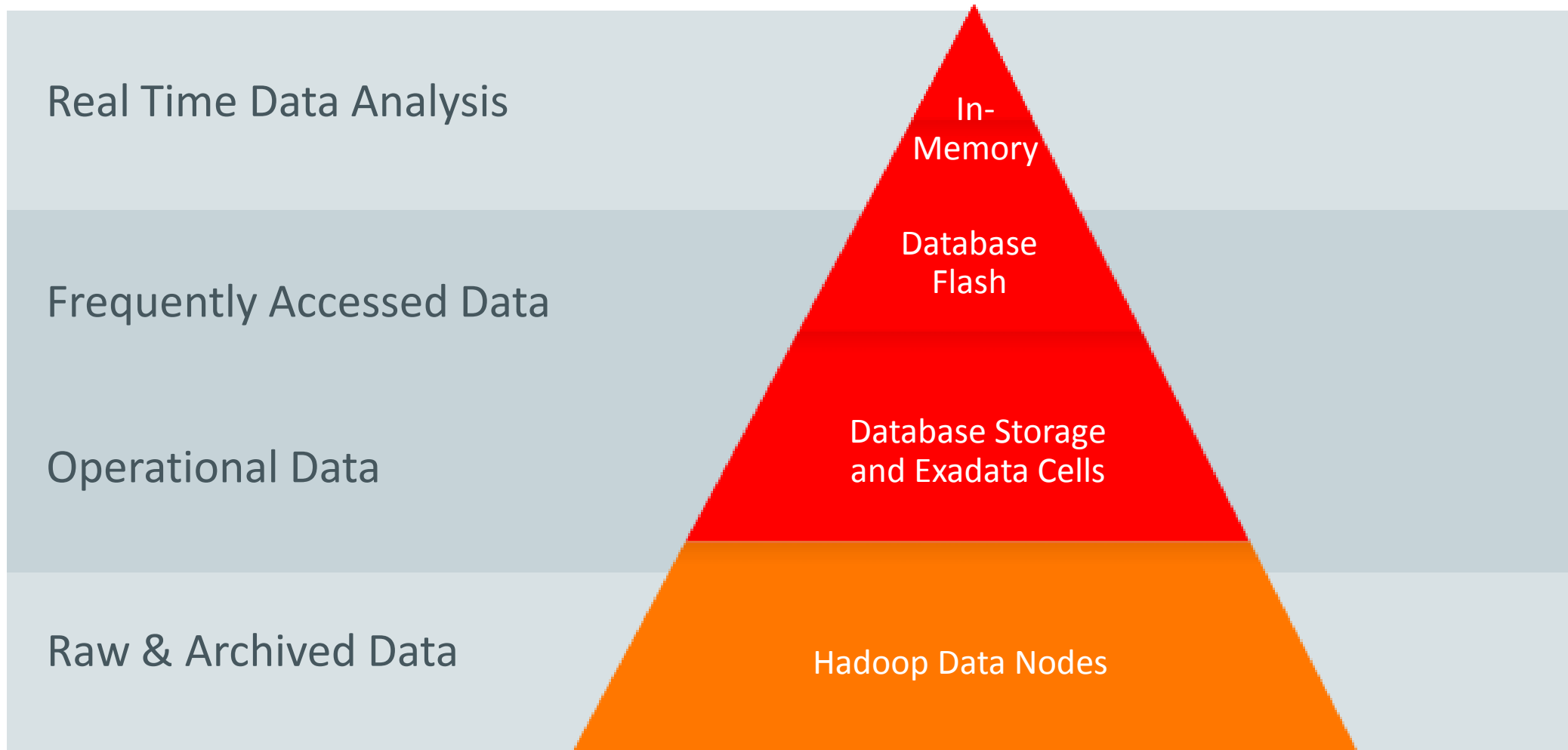


我々のゴール

– 真に統合されたライフサイクルマネジメントの実現



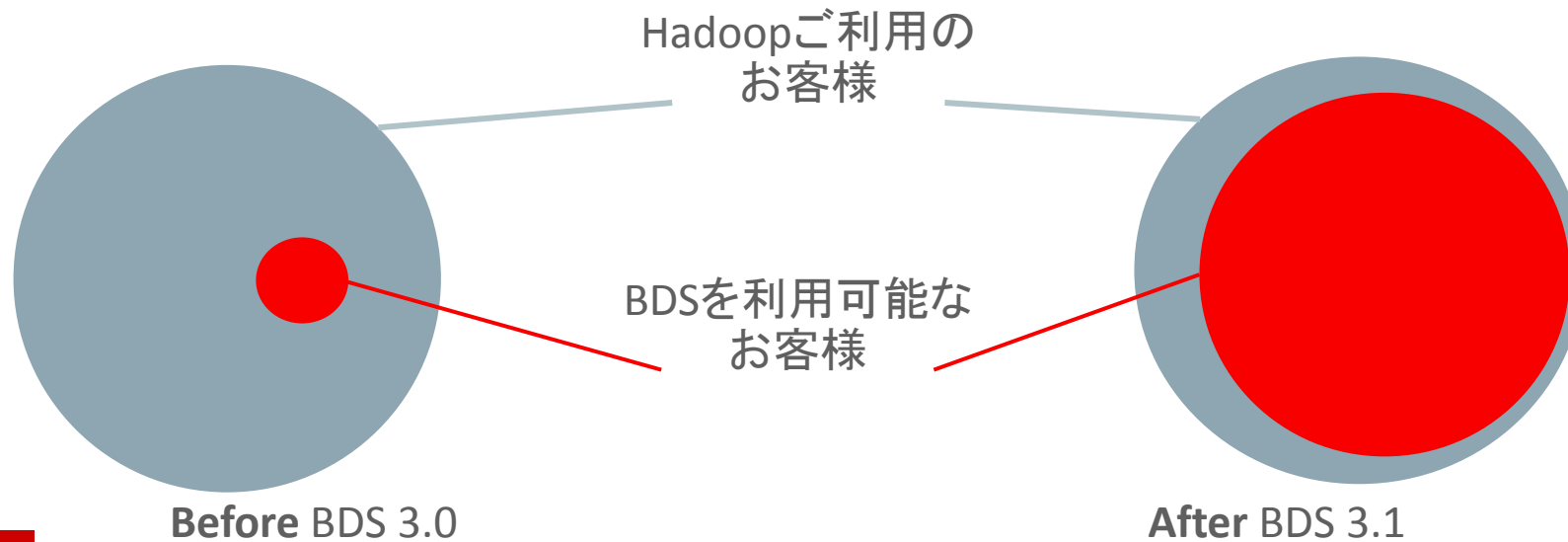
ユースケース



Big Data SQL 3.1 ハイライト②

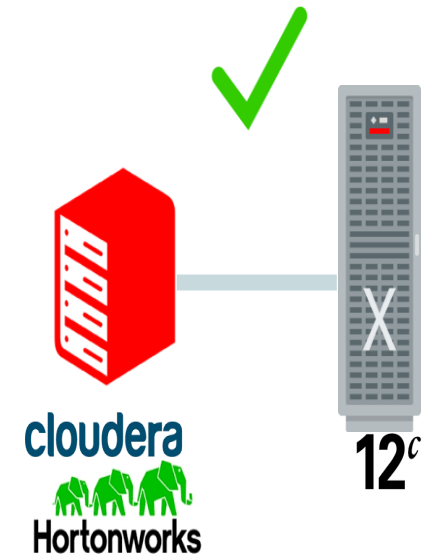
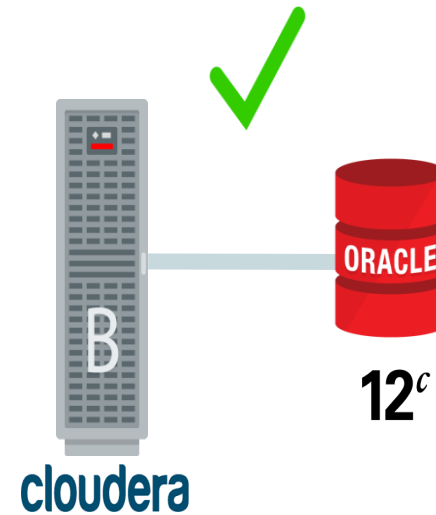
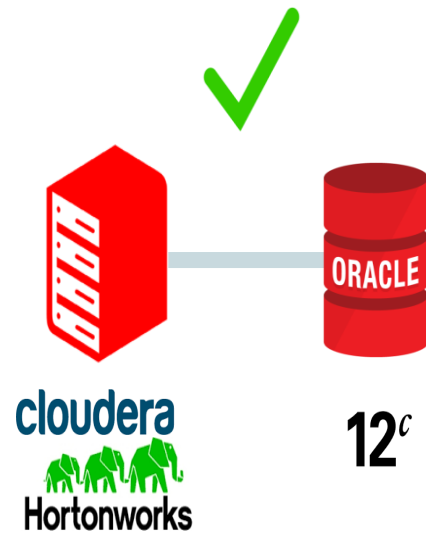
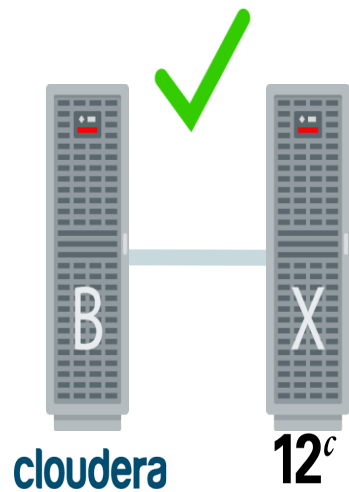
より多くのお客様へご提供可能に

- エンジニアドシステム以外のお客様もご利用可能になりました。
 - ◆一般的なH/Wでの Hadoop 及び、Oracleデータベース
 - ◆ClouderaとHortonworksをサポート
 - ◆もちろん、SmartScan等の各種固有機能はそのまま動作します。

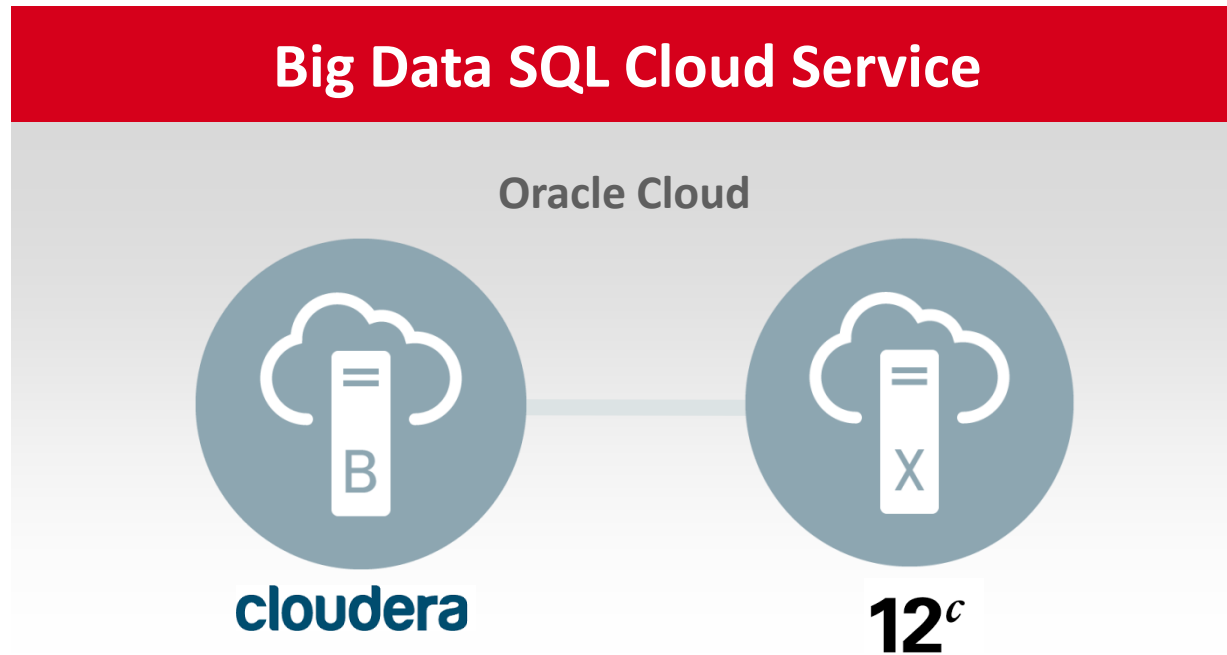


Big Data SQLを導入可能な組み合わせ

Hadoop	Oracle Database (12c only) on:	Availability
Oracle Big Data Appliance w/Cloudera	Oracle Exadata	v1.0 -
Commodity Cloudera or Hortonworks Clusters	Intel Commodity	v3.0-
Oracle Big Data Appliance w/Cloudera	Intel Commodity	v3.1-
Commodity Cloudera or Hortonworks Clusters	Oracle Exadata	v3.1-



予告 Big Data SQL Cloud Service



Big Data SQLのパフォーマンス向上のための様々な機能

- **(Hive) Partition Pruning**

- クエリ実行前にHiveのカatalogを参照し、必要なパーティションのみにアクセス
- Oracleデータベースのメタデータによりpartition pruningを最適化(外部表のパーティショニング)
- Goal: IO削減

- **Storage Index**

- IOをスキップするためのブロックに関するマーキングをメタデータで管理
- Goal: IO削減

- **Smart Scan**

- 必要なデータのみをOracleデータベースに返します
- Goal: データ移動の最小化

- **Bloom Filtering**

- ブルーム・フィルタリングを使ってHadoopノードにプッシュダウン
- Goal: Joinの最適化及びデータもとでの処理

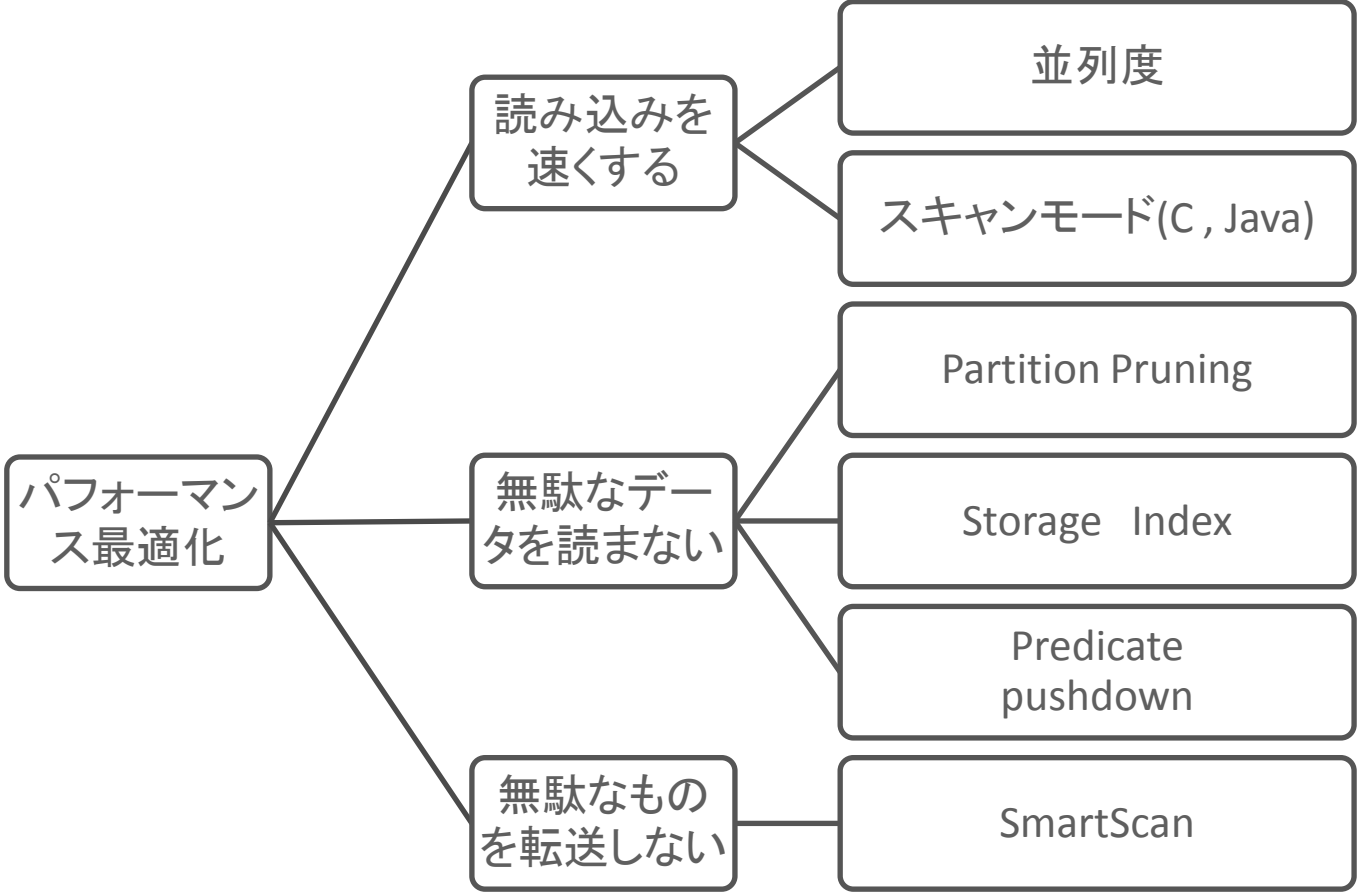
- **Predicate and Column Projection Pushdown**

- カラムストア型のファイルフォーマット(Parquet、ORC)対応
- Goal: IO削減

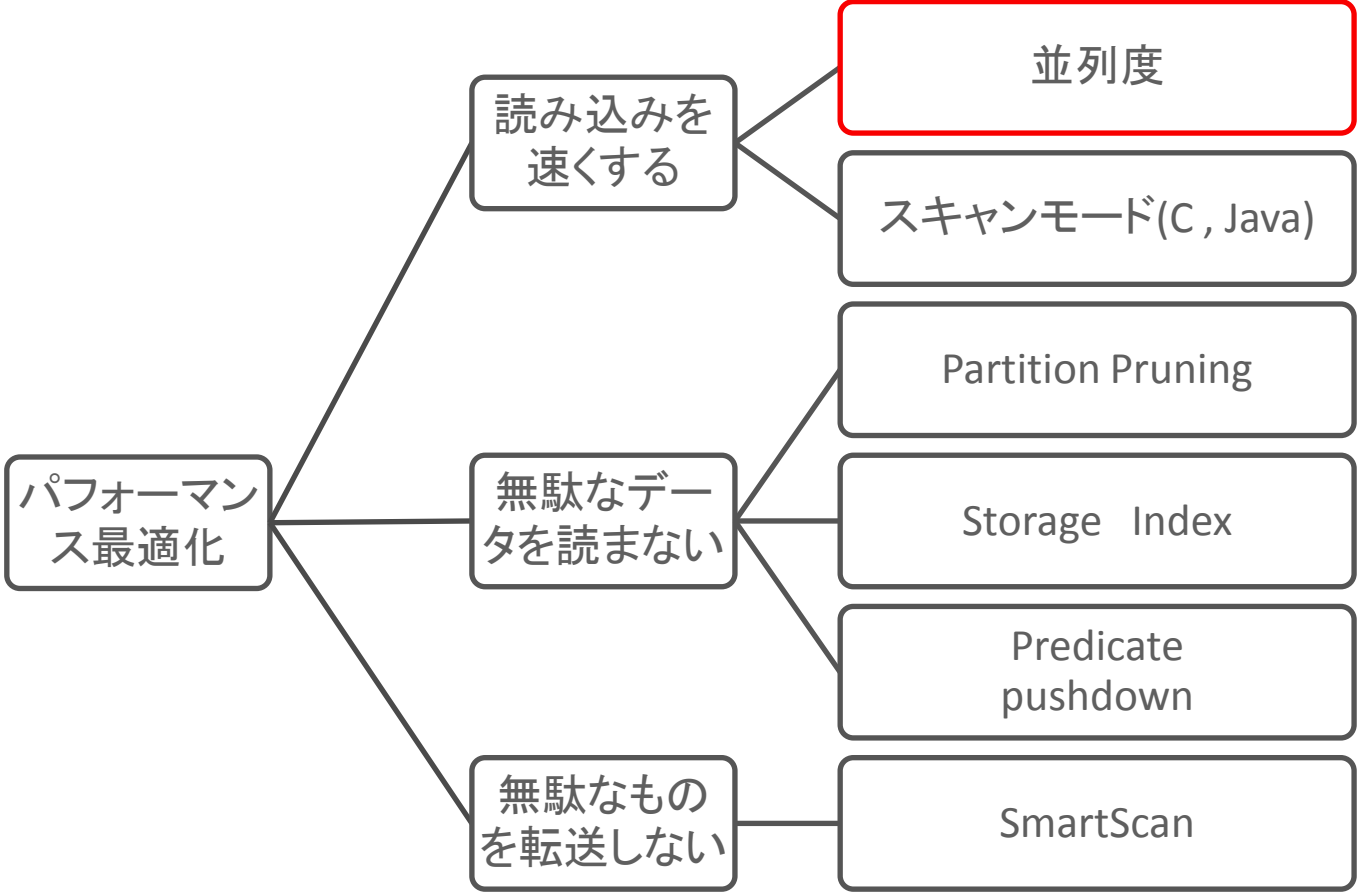
- **Security**

- OracleデータベースのセキュリティポリシーをOracle以外のデータベースに対して適用
- Goal: 高度なセキュリティ機能の有効化

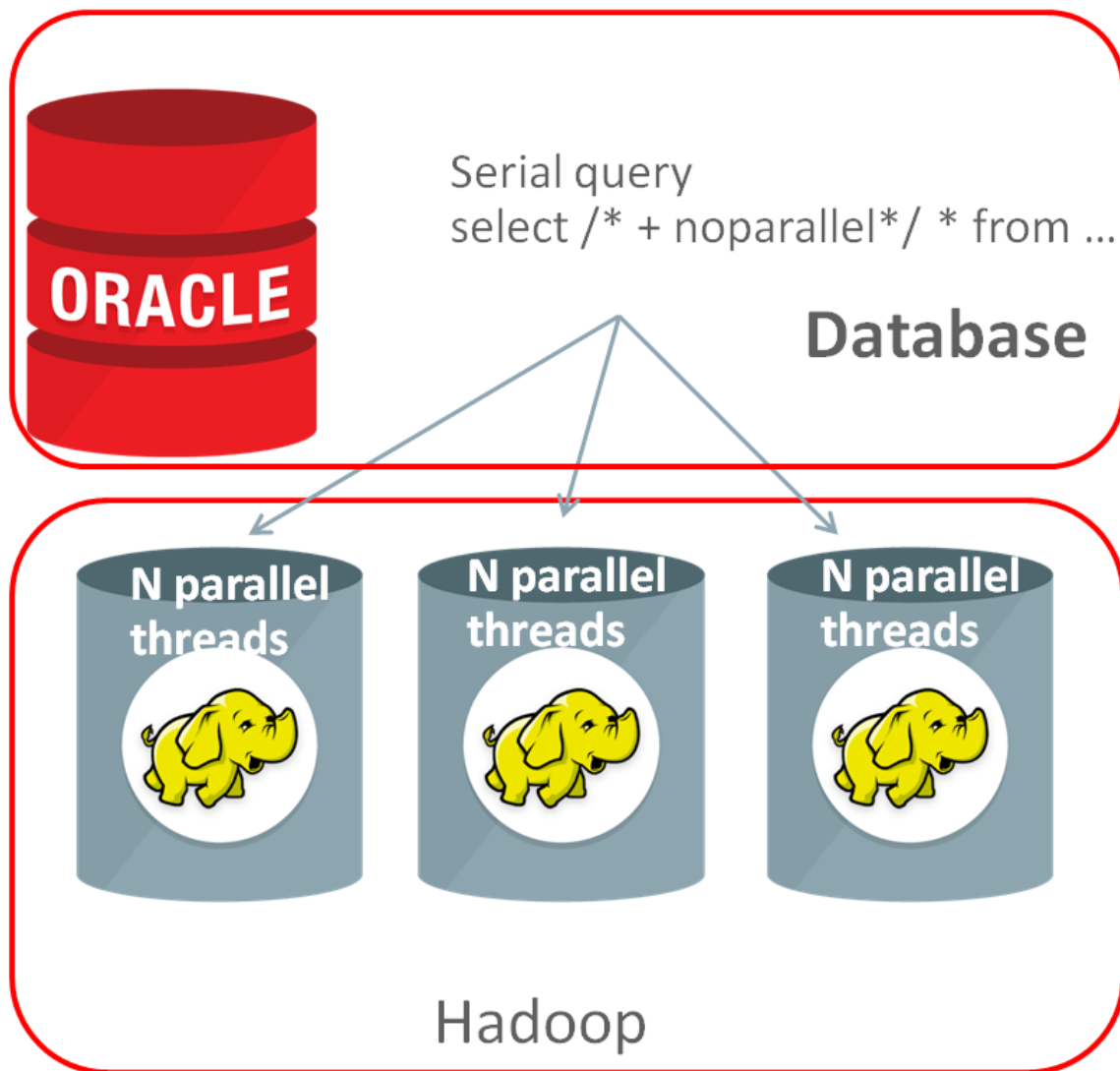
Big Data SQLにおけるパフォーマンス最適化



Big Data SQLにおけるパフォーマンス最適化



Oracle DBの並列度とHadoop側の並列度の関係



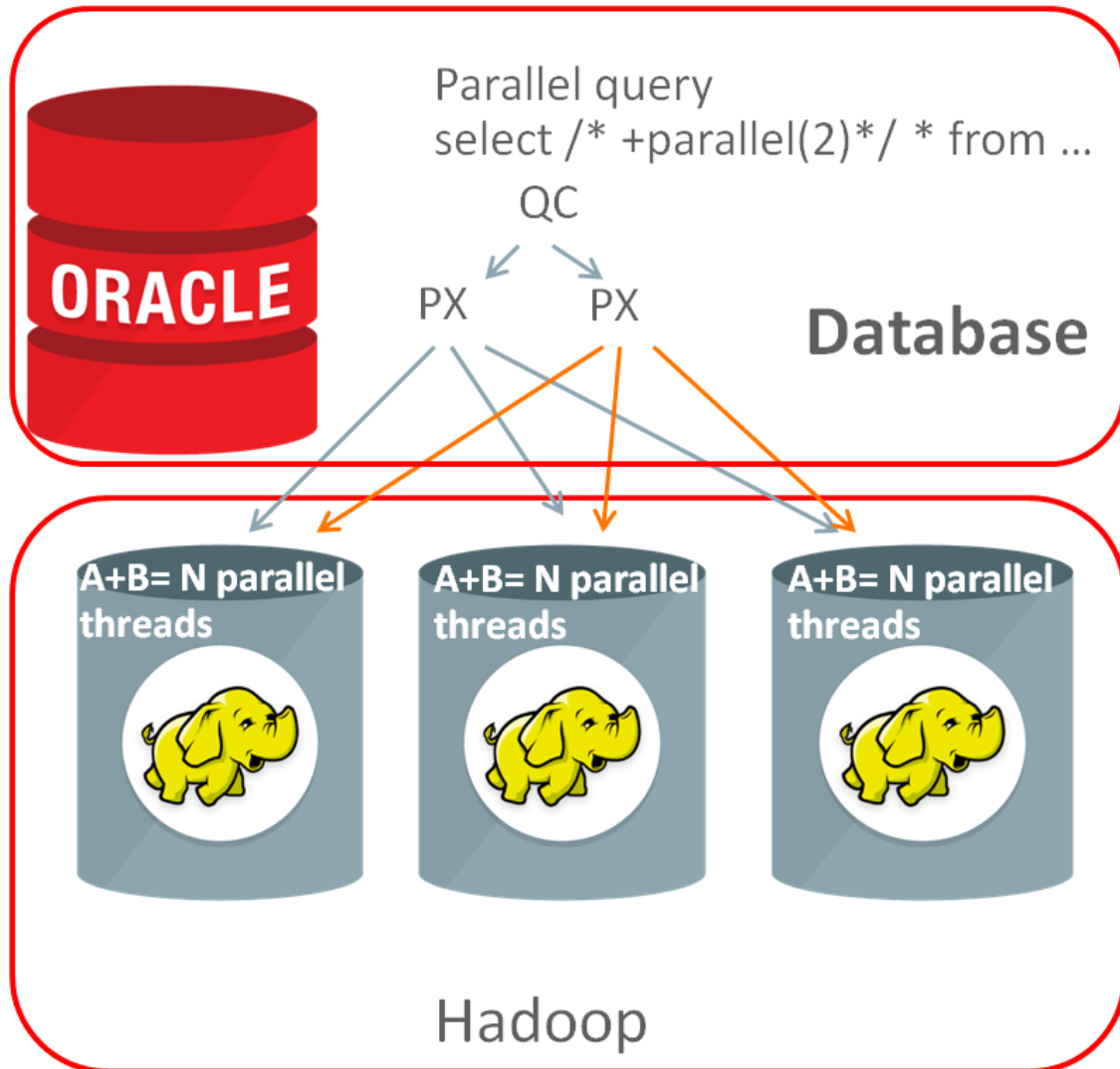
- Hadoop側の並列度は、最適化されたDOPが自動選択される

-DB側のDOP指定との関連はなく、DB側のリソースを抑えるためにシングルスレッドで実行してもHadoopレイヤでの処理は分散される

Hadoop側で実行できる処理(後述)については分散処理なので高速化できる

ただし、DBサーバ側での処理(集計等)はシングルのためボトルネックになる可能性がある

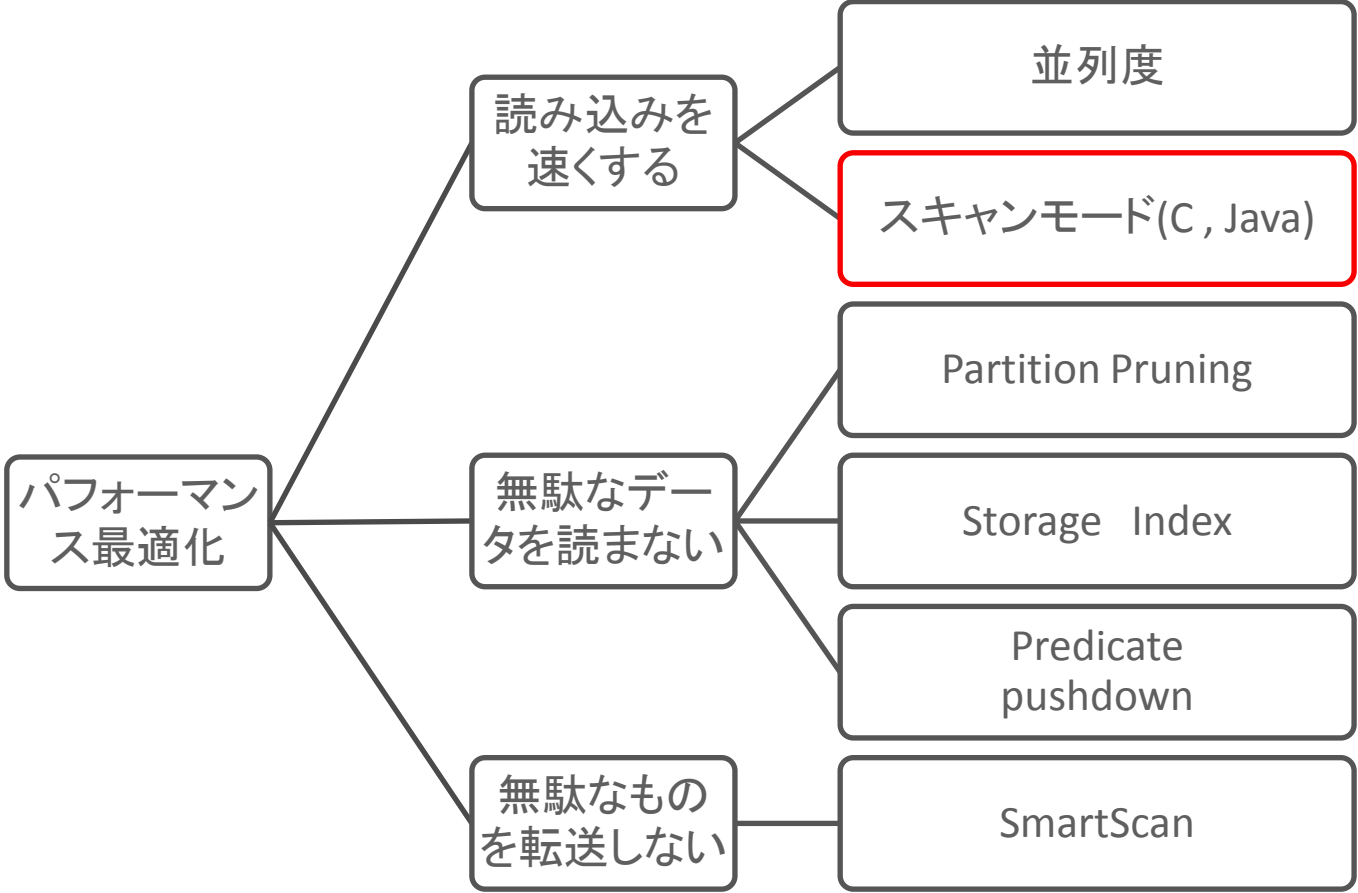
Oracle DBの並列度とHadoop側の並列度の関係



- DB側で並列度を上げた場合、PXサーバ毎にHadoopのリソースが割り振られる
⇒DB側のスレッドを増減してもHadoop側の並列度には影響を与えない

Hadoop側の並列度に関しては、チューニング不要

Big Data SQLにおけるパフォーマンス最適化



スキャンモード

- パフォーマンスに影響を与える2種類のスキャンモードがあります。

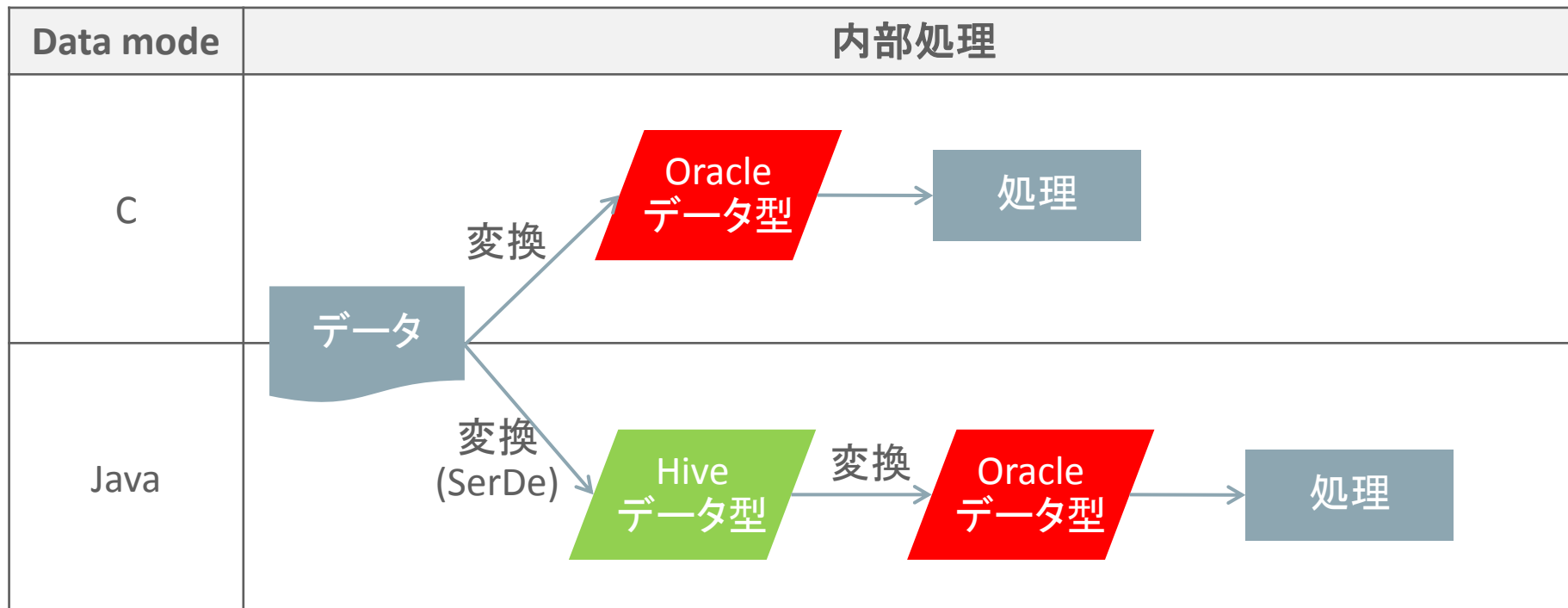
C : 独自実装の高速処理モード 現バージョンではデリミタ区切りのテキストのみに対応
Java : 内部的にHiveのStorageHandlerを利用したデータの読取りを行う

- Data Modeは、Create Table時にcom.oracle.bigdata.datamodeで指定します。
無指定の場合は、auto(デフォルト)となり自動適用されます(C優先)

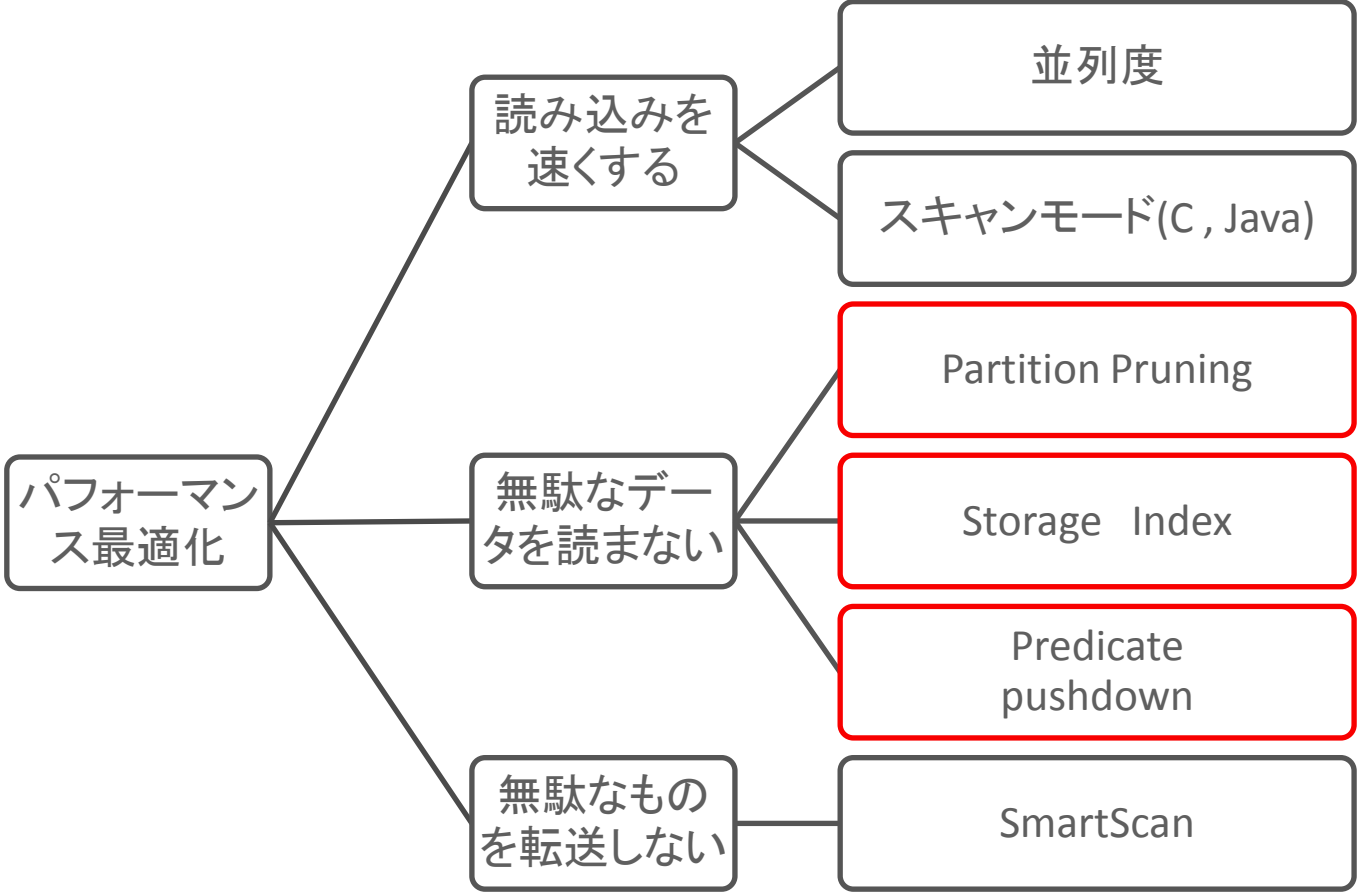
```
CREATE TABLE BDS_SAMPLE
  Col1 VARCHAR2(30) , Col2 NUMBER(30)
ORGANIZATION EXTERNAL (
  TYPE ORACLE_HDFS
  DEFAULT DIRECTORY DEFAULT_DIR
  ACCESS PARAMETERS(
    com.oracle.bigdata.cluster=cluster1
    com.oracle.bigdata.rowformat=Delimited fields terminated by '|'
    com.oracle.bigdata.datamode=c)
  LOCATION ('hdfs://cluster1-ns/user/hive/warehouse/lineitems'));
```

スキャンモード

- Java data modeはSerDeを利用するため、利便性に優れるものの、内部的には一度、Hiveデータ型に変換された後に、Oracleデータ型への変換が行われるため、C data modeより実行速度が劣ります。

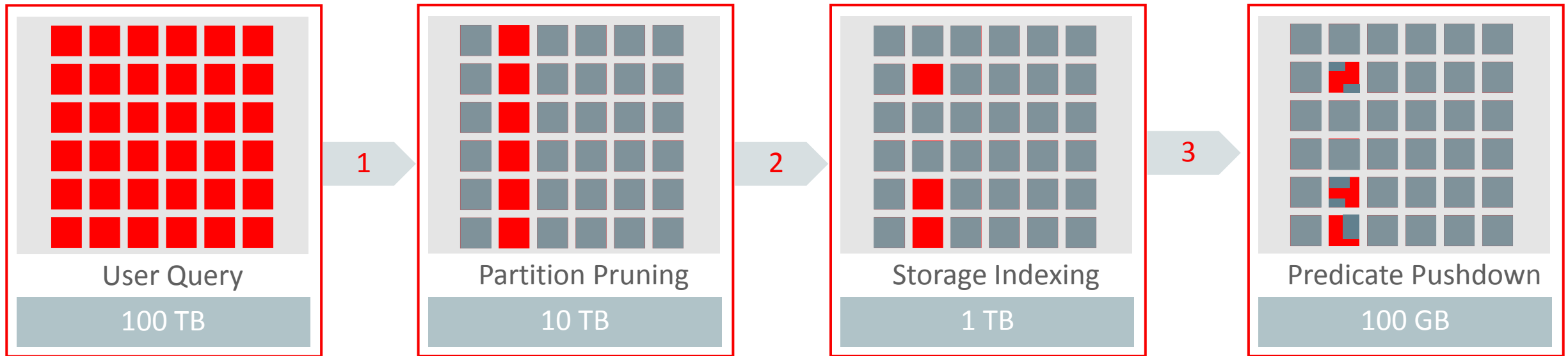


Big Data SQLにおけるパフォーマンス最適化



Big Data SQLパフォーマンス機能

SCAN性能向上のための仕組み



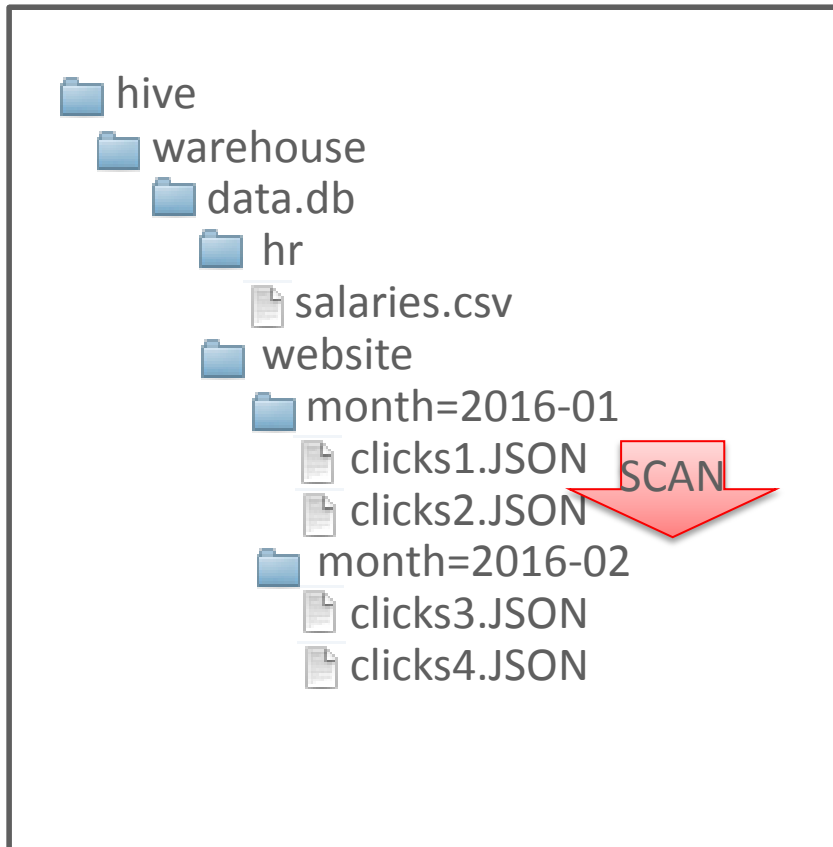
Hiveパーティション情報を利用して必要なディレクトリだけSCAN

自動作成されたIndexにより不要なブロックを読み飛ばす

Parquet/ORCフォーマットの場合は、ブロック内のスキャンを高速化



Hive Partition Pruning



BDS



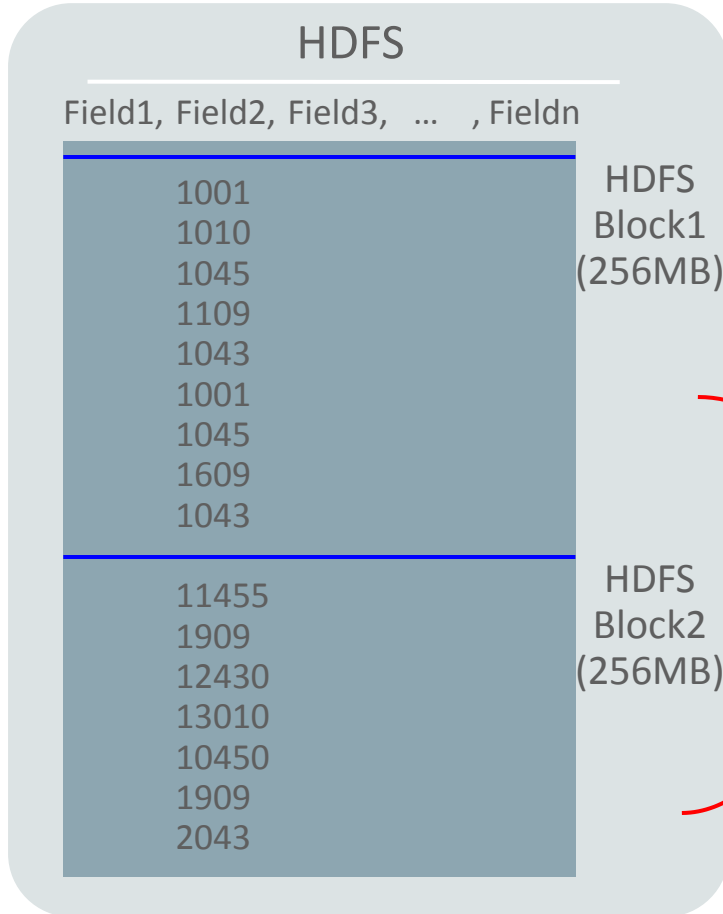
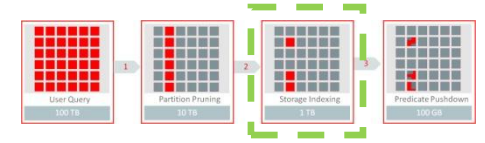
- BDSでもHiveパーティションが有効

Select * from website where month = 2016-01

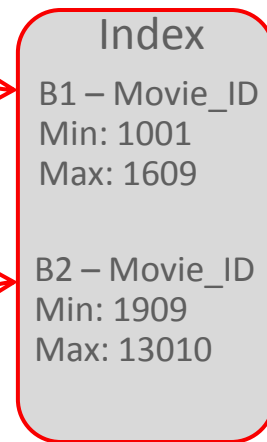
パーティションディレクトリの配下にあるファイルのみスキャンする(無駄読みを減らす)

BDSのDDL/SQL記述は特に意識不要
(Hive DDLでパーティション構造を記述)

Storage Index



SQL:
Select * from hoge
where
MOVIE_ID = 1109



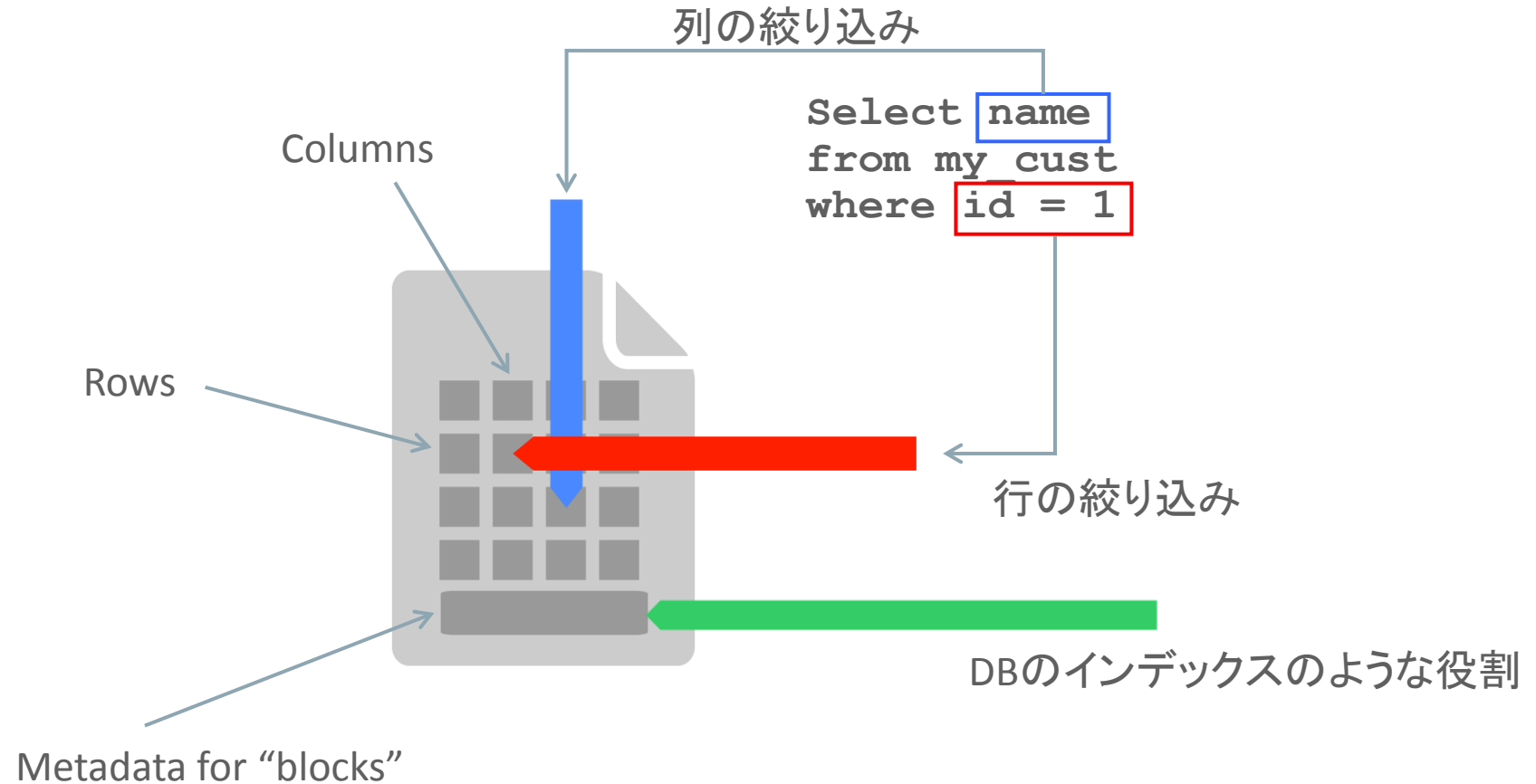
- 初回アクセス時に自動でブロック毎の各カラムのmin/max値のインデックス情報を作成（各ノードのメモリ上）
- 2回目以降のアクセス時には、条件に合致するデータがないブロックは読み飛ばす
- BDSのDDL/SQL記述は特に意識不要

Predicate Push Down

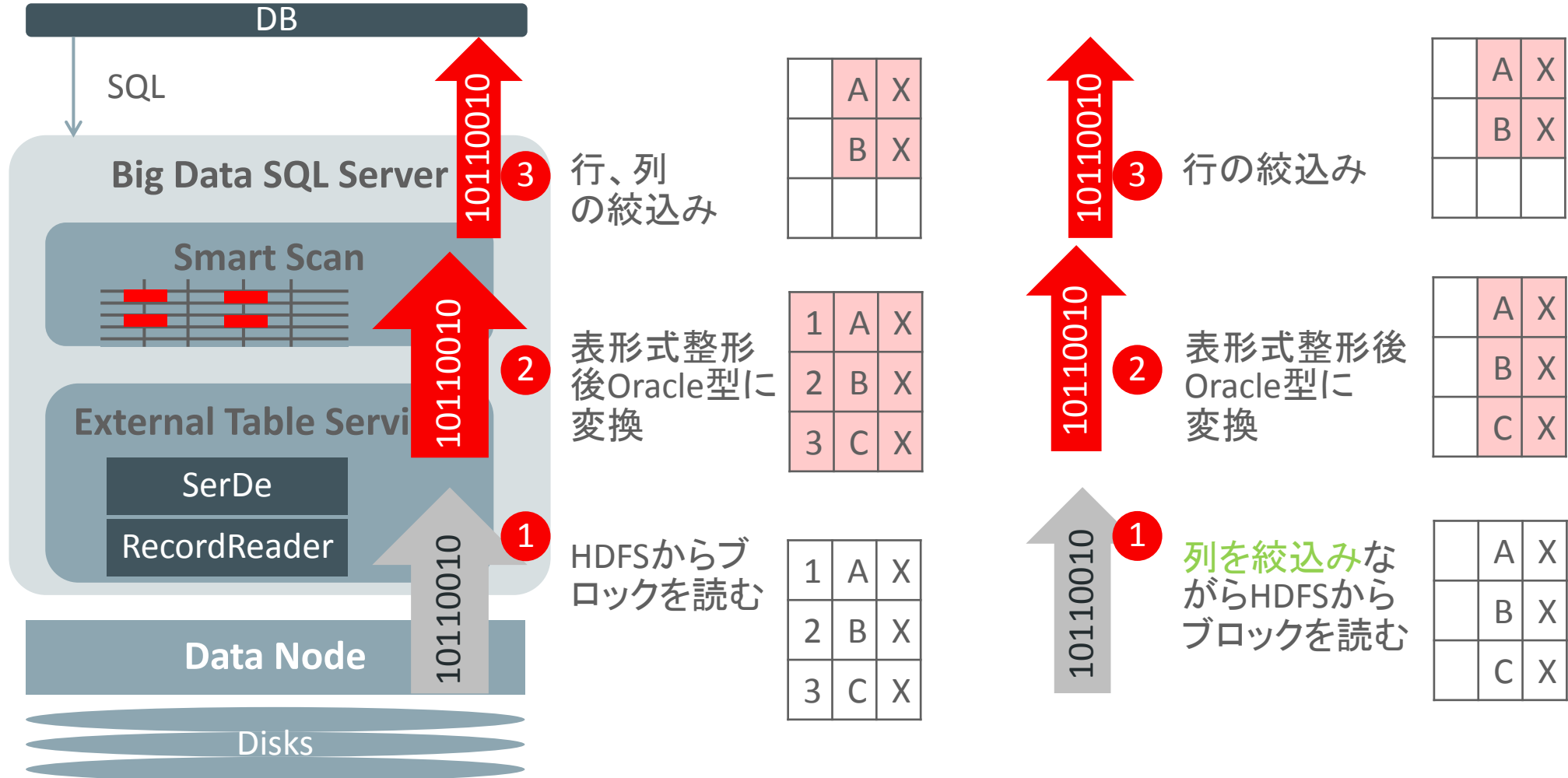
Parquet/ORC Fileに対する処理をネイティブに実行



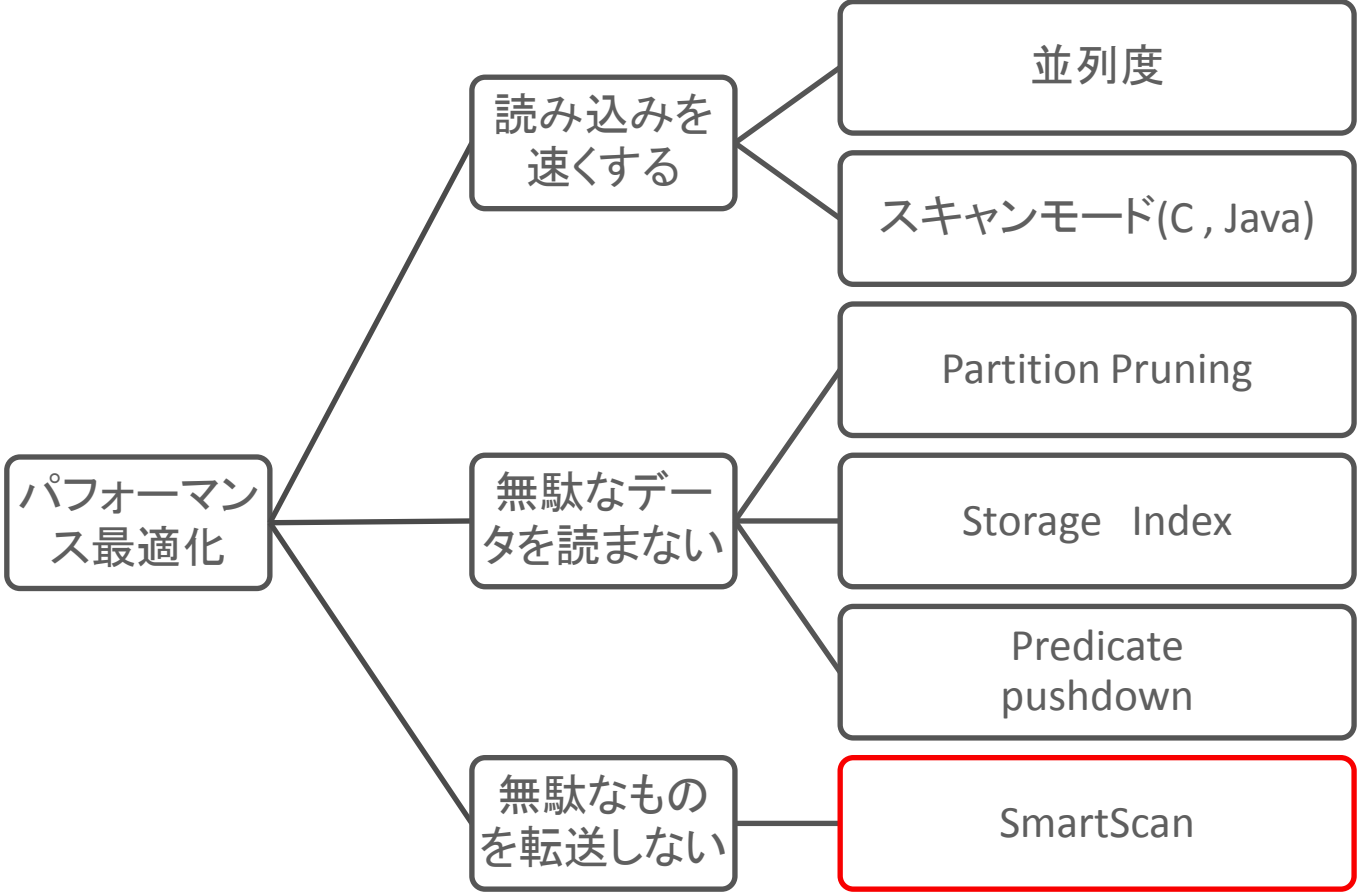
例) Parquet files



通常のSCANとPredicate Pushdownの違い

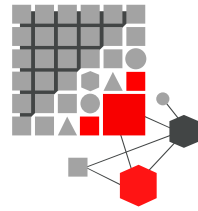


Big Data SQLにおけるパフォーマンス最適化

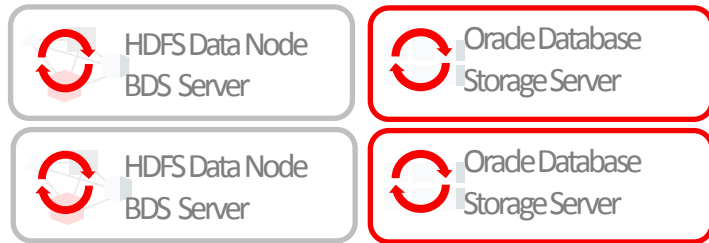


Exadataでお馴染みのSmart Scan機能をHadoopに実装

RDBMSとHadoop
に対するクエリ



Oracle SQL



 **Fast**

大規模並列分散処理

ローカルでの絞り込み

転送データの極小化

Big Data SQLがHadoop側で処理できるSQL

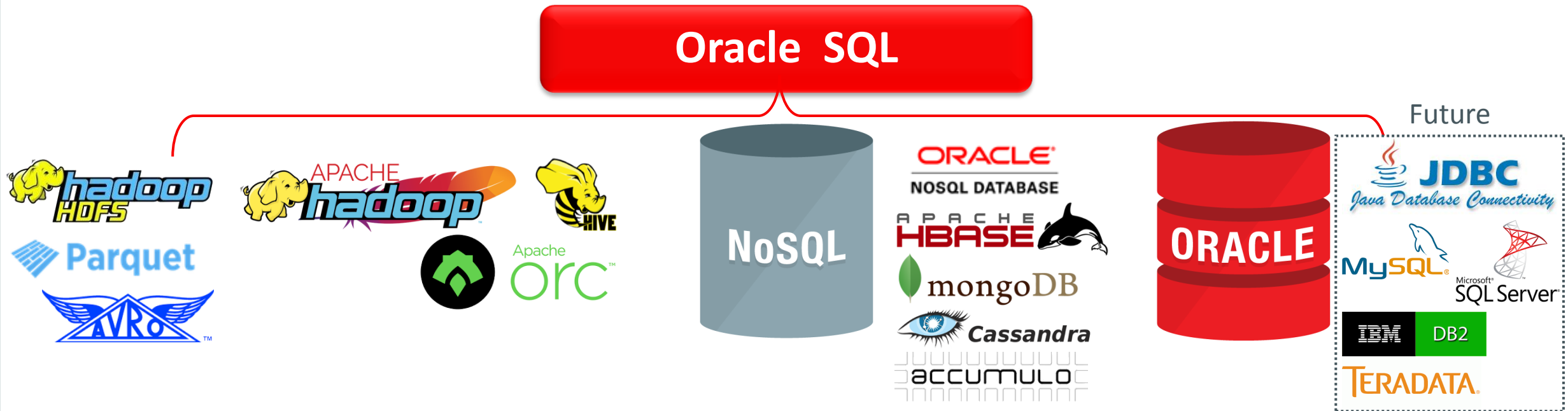
- Hadoop scans (InputFormat, SerDe)
 - 様々な形式のデータを読み取れます
- JSON parsing
 - Oracle Database12cのJSONパースの機能をHadoop側で実行
- WHERE clause evaluation
- Column projection
- Bloom filters for faster join
- 1件処理できる関数(文字列操作やOracle Data MiningのScoringなど)

本日のまとめ

One Fast, Secure SQL Queries over All your Data

オラクルは、Big Data SQLを中核に、RDBMSとHadoopをシームレスに統合できる世界を実現します。

- RDBは、より安価に大量のデータを保有できるようになります。
- Hadoopは、より簡単にパワフルなSQLを使ってセキュアにDBユーザーにデータを展開できます。



Integrated Cloud

Applications & Platform Services

ORACLE®