

Oracle DBA & Developer Days 2011

日本オラクル、今年最大の技術トレーニングイベント

2011年11月9日(水)～11月11日(金) シェラトン都ホテル東京



ORACLE®

**オラクルコンサルが語る
性能問題を未然防止するアプローチの真髓！！**

日本オラクル株式会社

コンサルティングサービス統括 テクノロジーソリューションコンサルティング統括本部
アソシエイトコンサルタント 栗田知代子

以下の事項は、弊社の一般的な製品の方向性に関する概要を説明するものです。また、情報提供を唯一の目的とするものであり、いかなる契約にも組み込むことはできません。以下の事項は、マテリアルやコード、機能を提供することをコミットメント(確約)するものではないため、購買決定を行う際の判断材料になさらないで下さい。オラクル製品に関して記載されている機能の開発、リリースおよび時期については、弊社の裁量により決定されます。

OracleとJavaは、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。文中の社名、商品名等は各社の商標または登録商標である場合があります。

Agenda

- Introduction
- 設計/開発におけるアプローチ
- テストにおけるアプローチ
- 運用におけるアプローチ
- アプリチームとDBチームの連携
- Summary

Introduction



目的とゴール

目的

性能問題を未然に防ぐアプローチを理解する

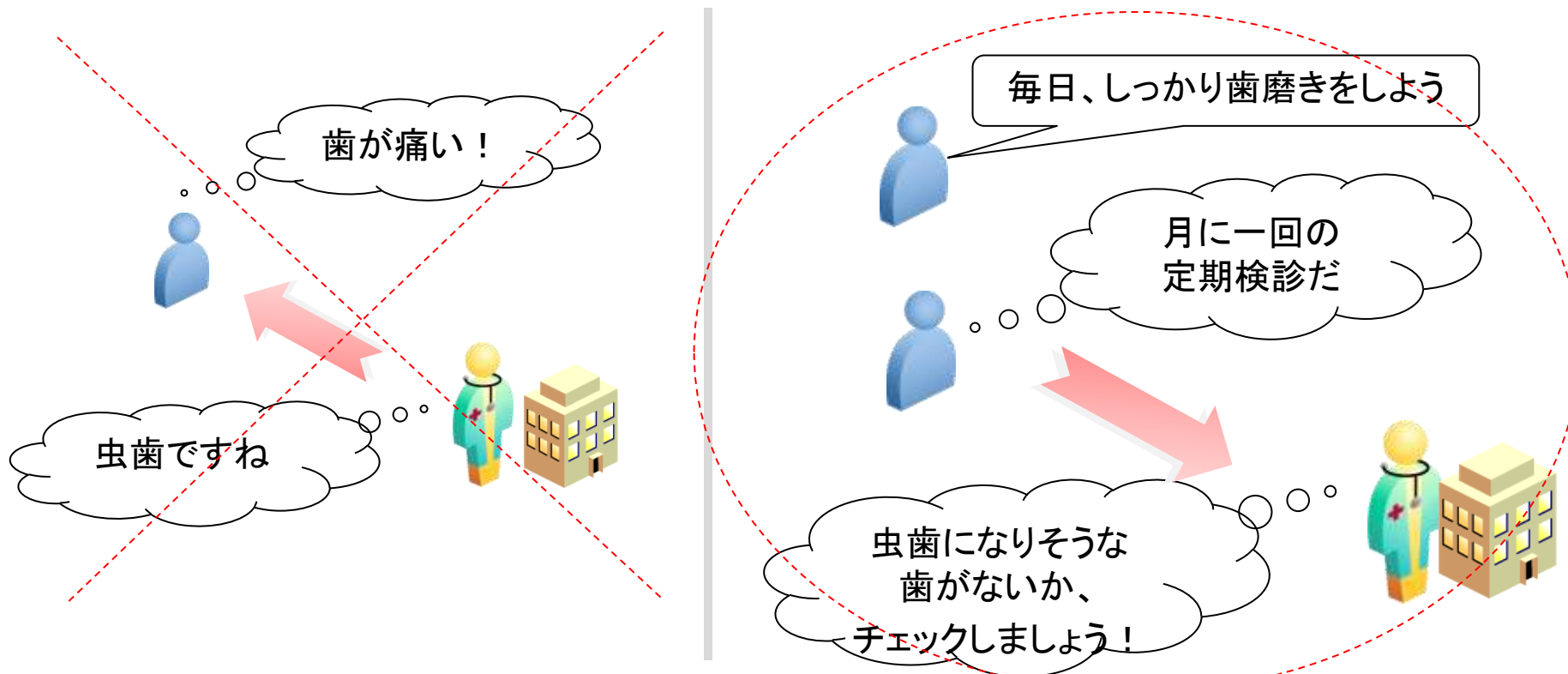
ゴール

- ・ 設計/開発におけるアプローチを身につける
- ・ テストにおけるアプローチを身につける
- ・ 運用におけるアプローチを身につける
- ・ アプリチームとDBチームの連携を身につける

性能問題の未然防止の必要性

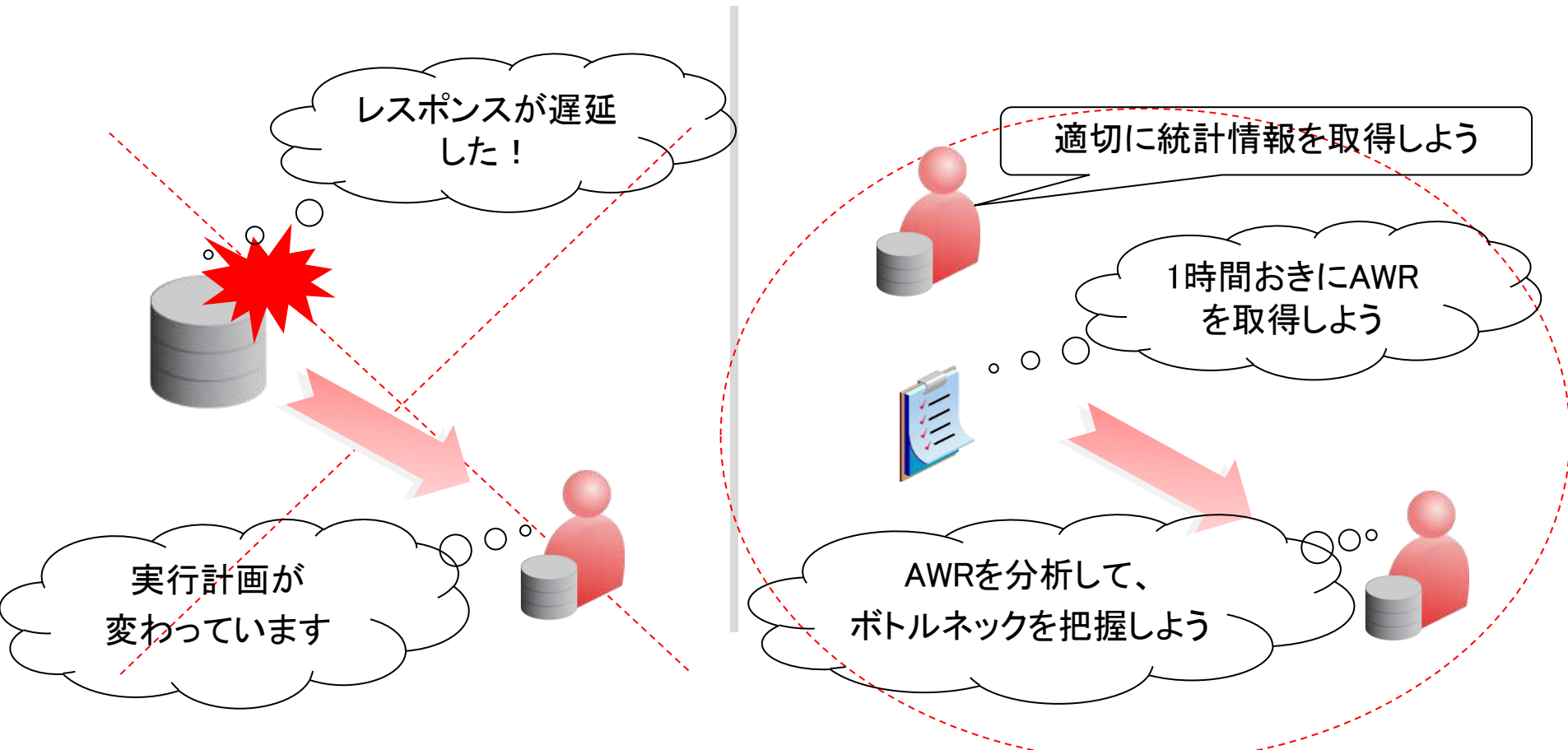
- 性能問題は発生してから対応するのではなく、未然防止することが重要

人の健康に例えると



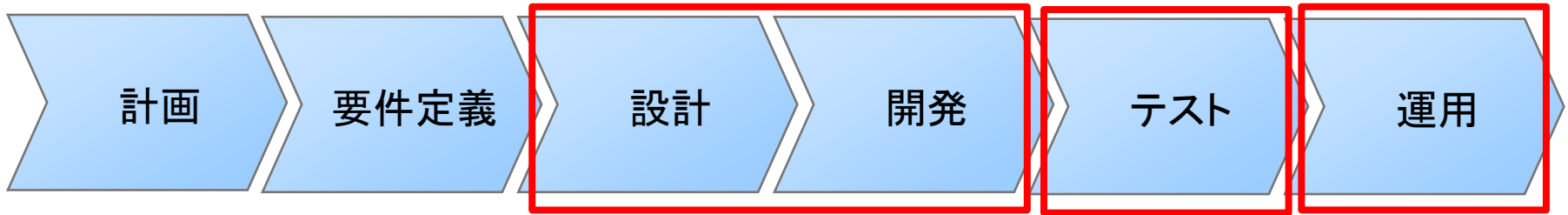
性能問題の未然防止の必要性

システムにおける未然防止の必要性



プロジェクトフェーズにおける未然防止

プロジェクトフェーズ



	計画	要件定義	設計	開発	テスト	運用
プロジェクト内容	システム全体要件、構成の概要レベルでの検討や、プロジェクトのスケジュール、体制の立案をするフェーズ	顧客と要件をすり合わせてシステム方式を検討するフェーズ	実際に設計を行うフェーズ。オブジェクト設計・アプリケーション設計・論理設計・物理設計など	実際に開発していくフェーズ	システムテストを行うフェーズ	実際に運用していくフェーズ
予防内容	プロジェクトの計画・体制を適切に立案する	顧客と検討した要件の実現性を確認する	パフォーマンスを意識して設計、品質チェックを行う	パフォーマンスを意識して開発、品質チェックを行う	質の高いシステムテストを行う	パフォーマンス問題を考慮した運用設計が行われているかが重要

設計/開発におけるアプローチ



シナリオ説明とポイント

- 設計・開発フェーズのCase Study

本番稼働中のA社では大量データを扱うバッチ処理の遅延が発生しております。遅延が発生している原因を調査したところ対象データが増え、アクセスブロック数が増加したことによるものでした。遅延対象のSQLの実行計画は適切なものが選択されており、SQLチューニングではこれ以上の性能改善は見込めません。

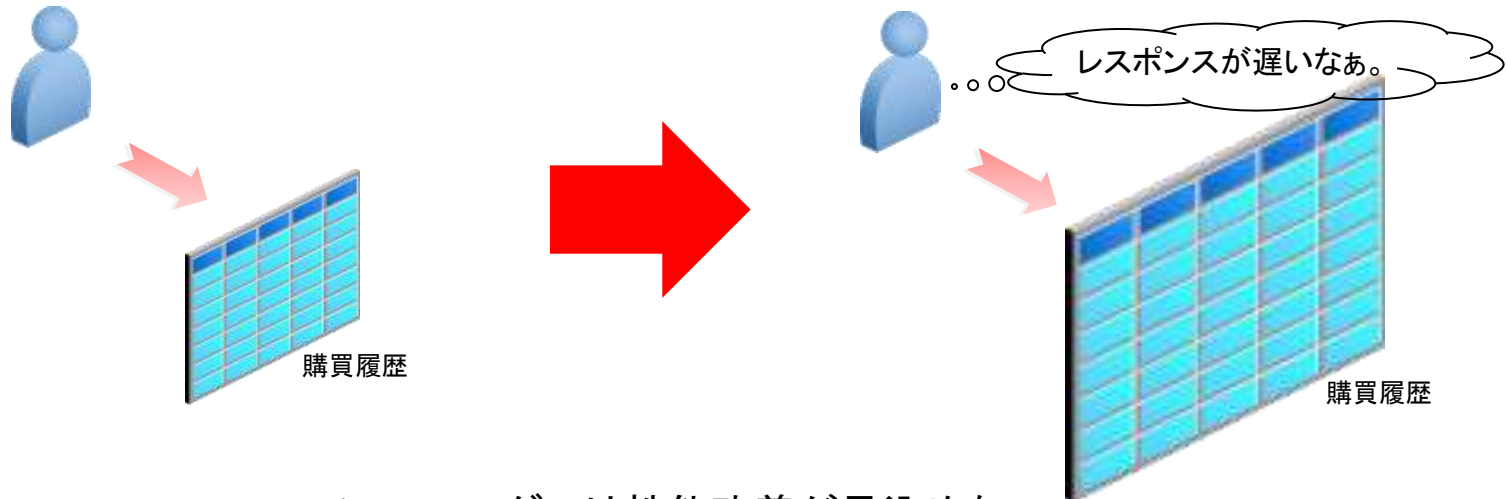
Case Studyのポイント

バッチ処理の遅延を未然に防ぐためには、
どのようなアプローチが必要だったのでしょうか？

原因分析

SQLはFull Scanをしており、データ量が増えたことによって、読み込みブロック数が増加した。

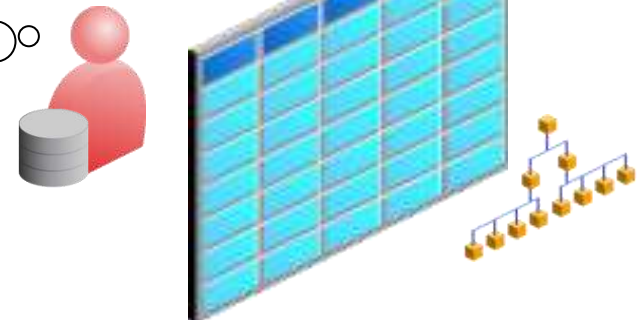
データは購買履歴が格納されており、今後も増加する傾向にある。



SQLチューニングでは性能改善が見込めない。

読み込んでいるデータは表の50%程度なので、索引は使えないなあ。

今後もデータが増えていくので、SQLがどんどん遅延していきそうだ。

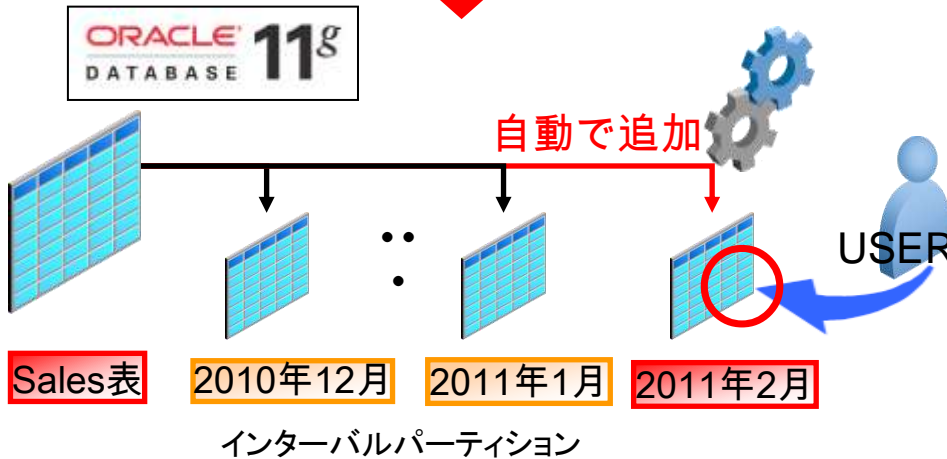
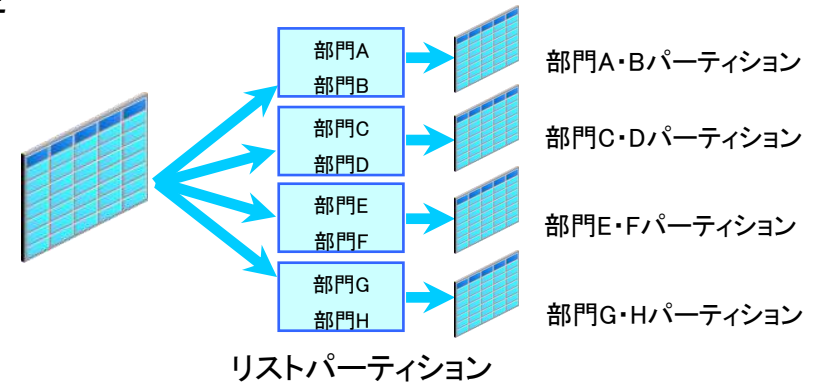
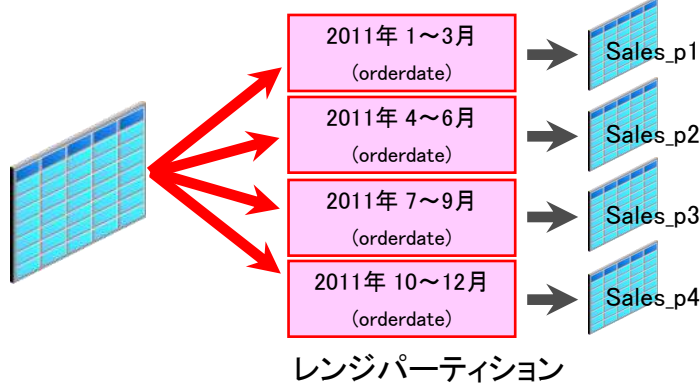


未然防止策検討

- 未然防止をするためには、データ量が増加することを考慮したうえで、オブジェクト設計を行う。
 - パーティション設計
 - 圧縮設計
- オブジェクト設計だけをガイドするのではなく、処理の特徴を理解して下記もガイドしていく。
 - アプリケーションの処理方式
 - SQLコーディング方法
 - PL/SQLコーディング方法

パーティション設計

- パーティション表の選定
- パーティションの種類・パーティションキーの選定



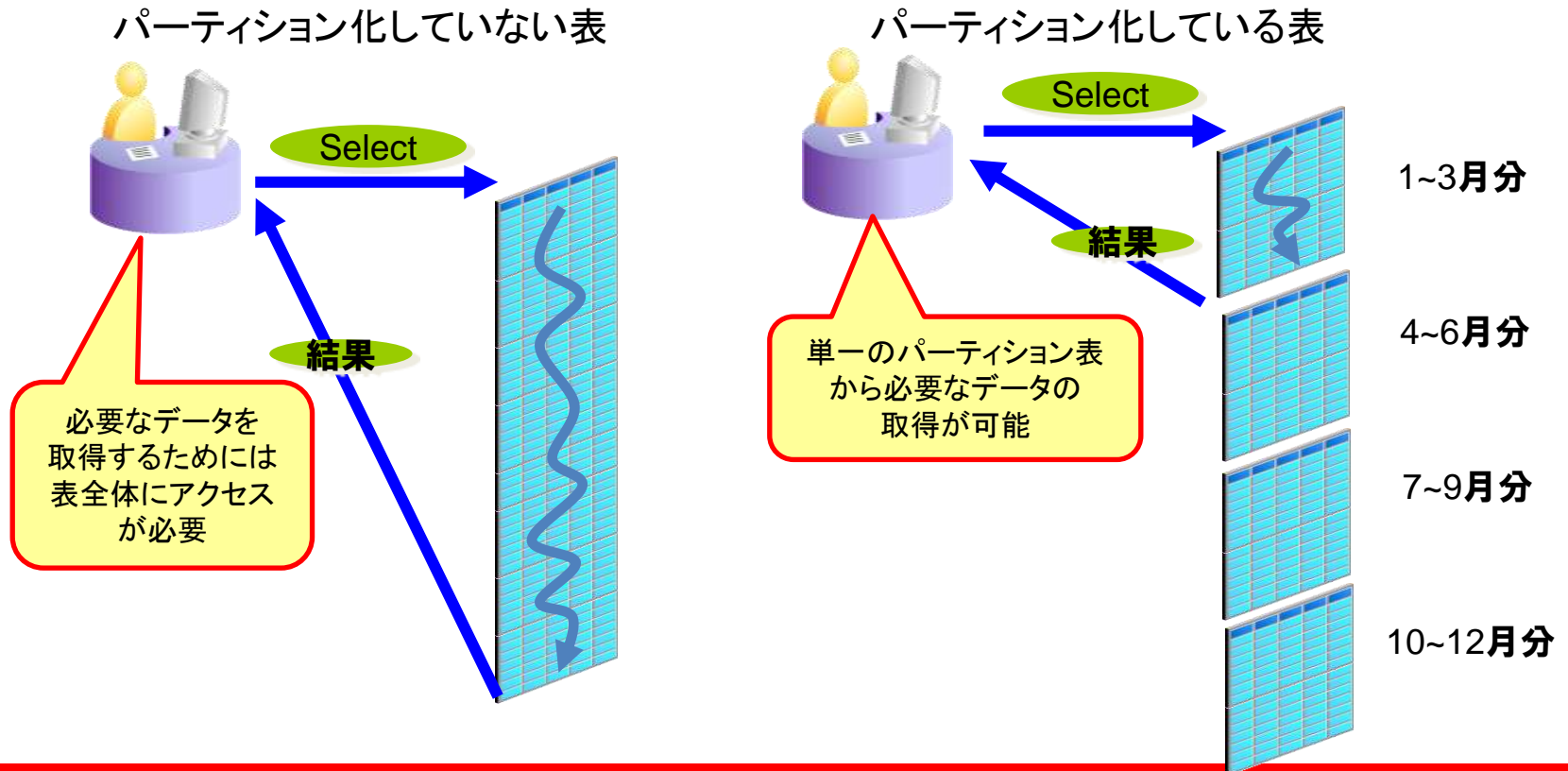
パーティション作成方法

```
SQL> create table <テーブル名> (<列名> <データ型>)
partition by range(<パーティション・キー>)
(partition <パーティション名> values less than(<データ範囲>),
...
partition <パーティション名> values less than(<データ範囲>));
```

アクセスブロック数を減らすために

パーティション・プルーニングの機能を理解したうえで、
オブジェクト設計・開発 (SQL作成) を行う

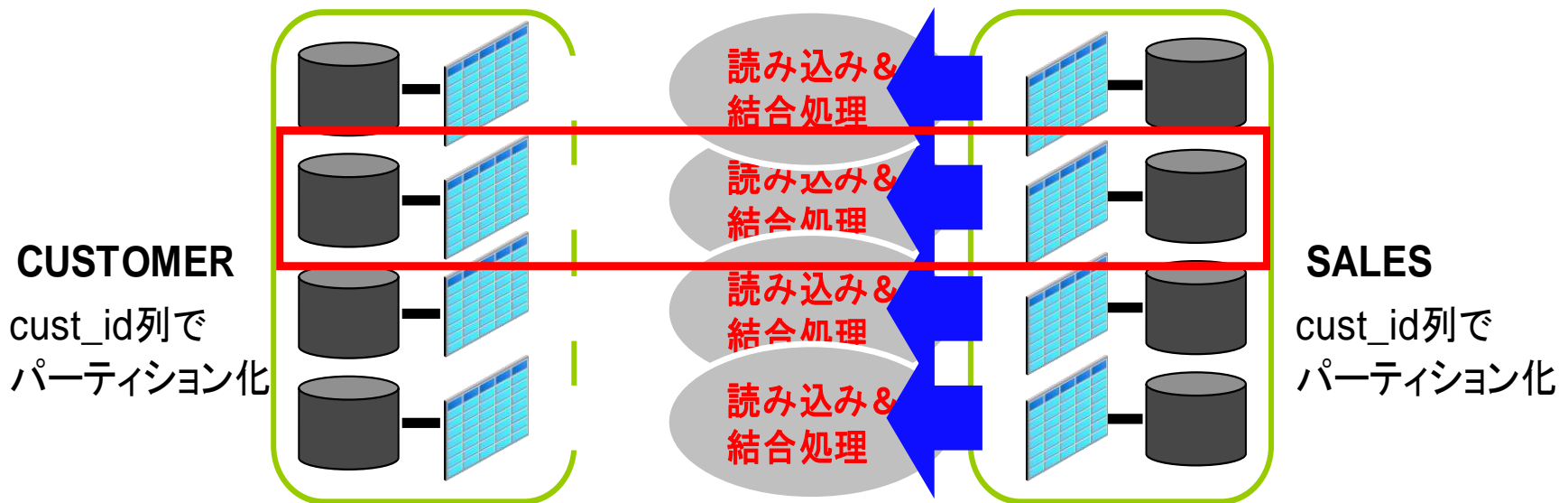
パーティション・プルーニングを使用するためにはwhere句にパーティション・キーを入れる。



効率的な結合方法

パーティション・ワイズジョインの機能を理解したうえで、
オブジェクト設計・開発(SQL作成)を行う

パーティション・ワイズジョインを使用するために双方の表を同じパーティション・キーで
パーティション化する。
表にパラレル設定を行い、パーティション・ワイズジョインを行う。



実行計画の確認方法

•パーティション・プルーニング

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time	Pstart	Pstop
0	SELECT STATEMENT		10000	234K	133 (12)	00:00:02		
1	PARTITION LIST SINGLE		10000	234K	133 (12)	00:00:02	KEY	KEY
2	TABLE ACCESS STORAGE FULL	SALES_LIST	10000	234K	133 (12)	00:00:02	2	2

•パーティション・ワイズジョイン (Rows,Bytes,Cost,Time列は削除しています)

Id	Operation	Name	Pstart	Pstop	TQ	IN-OUT	PQ Distrib
0	SELECT STATEMENT						
1	PX COORDINATOR						
2	PX SEND QC (RANDOM)	:TQ10000			Q1, 00	P->S	QC (RAND)
3	PX PARTITION LIST ALL		1	4	Q1, 00	PCWC	
* 4	HASH JOIN				Q1, 00	PCWP	
5	TABLE ACCESS STORAGE FULL	SALES_LIST	1	4	Q1, 00	PCWP	
6	TABLE ACCESS STORAGE FULL	SALES_LIST_2	1	4	Q1, 00	PCWP	

圧縮設計

• 圧縮方式の選択

6, BBB, AAA
5, AAA, BBB
4, AAA, AAA
3, CCC, AAA
2, BBB, CCC
1, AAA, BBB



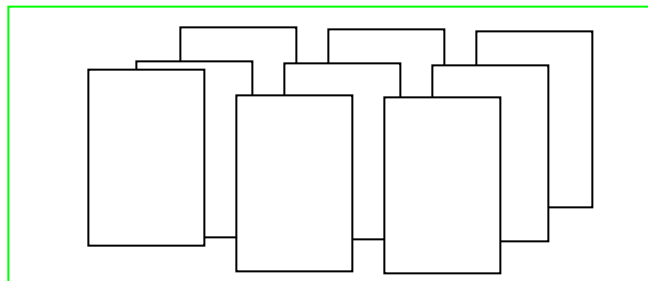
AAA ●	BBB ■
CCC ▲	
6, ■, ●	5, ●, ■
4, ●, ●	3, ▲, ●
2, ■, ▲	1, ●, ■



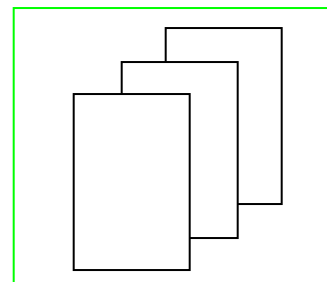
OLTP表圧縮・基本圧縮

※ブロックのイメージ図です

- 圧縮表の選定
- 圧縮率の見積もり



未圧縮データ



圧縮データ

AAAAAA: ●	
6, FFFFFFFF	
5, EEEEEEE	
3, ●	4, ●
1, ●	2, ●

圧縮効果低

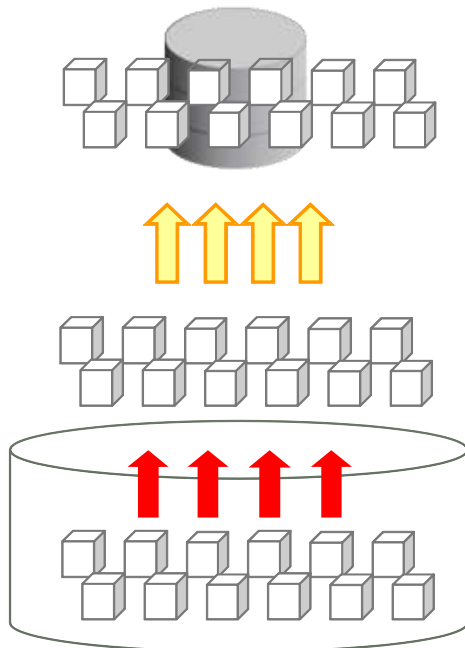
AAAAAA: ●	
5, ●	6, ●
3, ●	4, ●
1, ●	2, ●

圧縮効果高

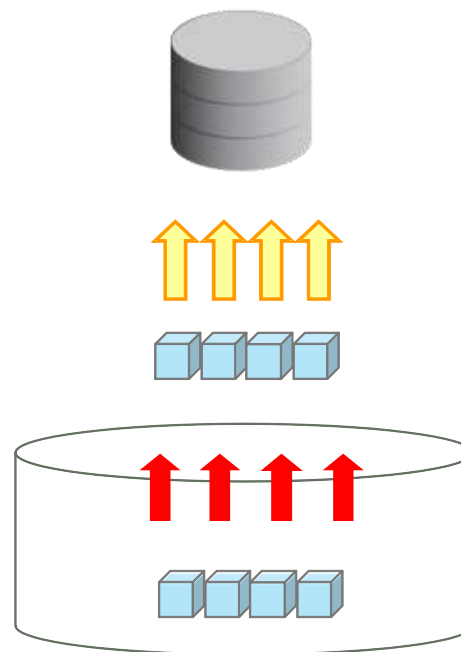
圧縮時の読み込みブロック数の減少

圧縮することによって、読み込みブロック数を減少させる。
圧縮ブロックの展開には、CPUのオーバーヘッドを考慮。

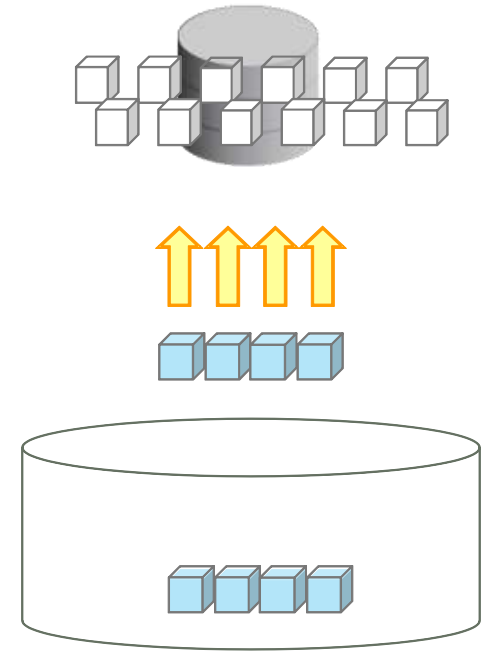
非圧縮



圧縮



読み込みブロック数の減少



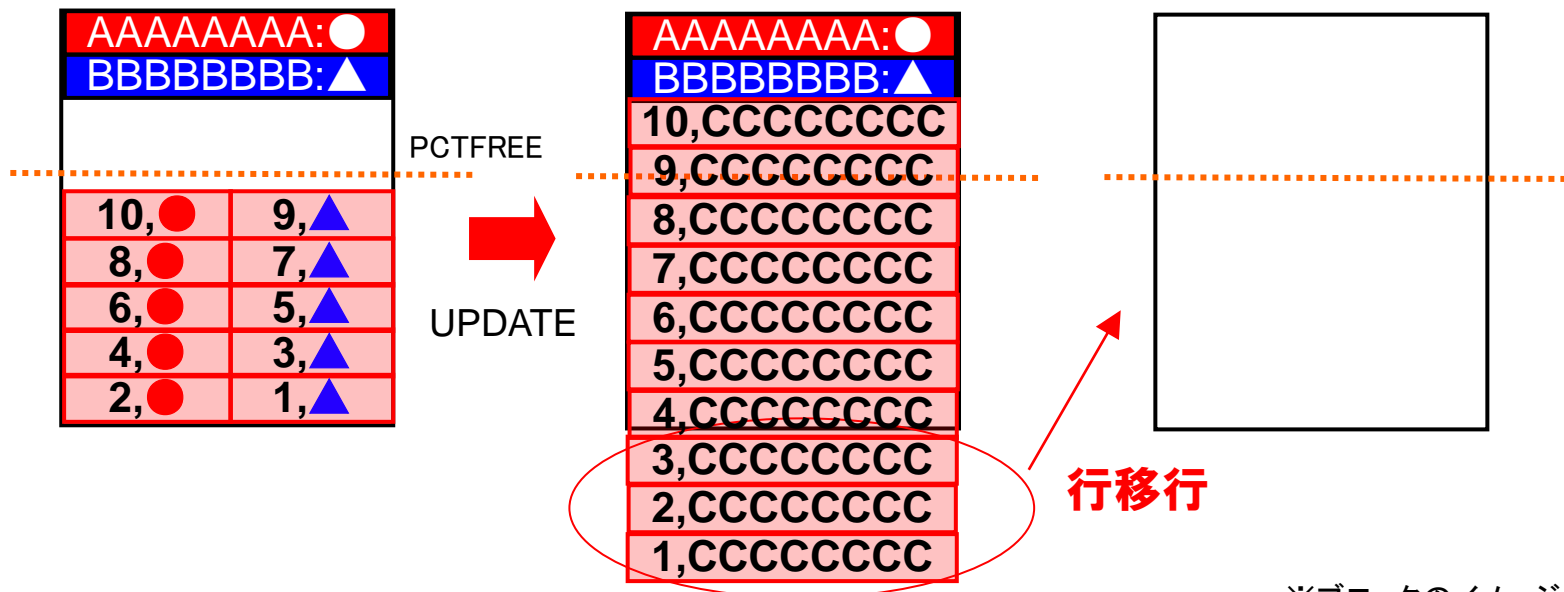
展開時のCPUオーバーヘッド

圧縮時の注意事項

仕組みを理解したうえで、オブジェクト設計を行う

大量UPDATEが行われる表は、圧縮に不向き

- 行移行が発生して表のサイズが大きくなる
- UPDATEの性能が劣化する
- UPDATEの時に非圧縮表と比べてUNDO・REDOを多く生成する



※ブロックのイメージ図です

設計・開発フェーズの考慮点

- オブジェクト設計
 - 索引設計
 - パーティション設計
 - 圧縮設計
 - ローディング設計
 - フラッシュ・キャッシュ設計
- アプリケーションの処理方式
- SQLコーディング方法
- PL/SQLコーディング方法

アプリケーションの処理や、H/Wスペックなどを考慮して、
向いている機能・ガイドを提供する必要がある

設計・開発フェーズのアプローチ

業務要件を満たすアプリを設計・開発するだけではなく、
DBの性能を考慮した設計・開発を行うことが重要！

アプリチーム

- 処理内容を考慮したオブジェクト設計を行う
- 効率の良いSQLを作成する
- 使用可能なDBの機能を理解して、使用する
- 性能問題を発見しやすいようなアプリを作成する

DBチーム

- APチーム向けのガイドを作成する
- APチームのレビューを開発・設計フェーズで行う

テストにおけるアプローチ



シナリオ説明とポイント

- テストフェーズのCase Study

A社では本番運用を始めたところ、SQLの遅延が発生しました。
昨日までは、SQL遅延は発生していませんでしたが、
カットオーバー後3日目になって急に遅延が発生しました。
本日は、休日なので、昨日よりもユーザのアクセス数が増えているようです。

Case Studyのポイント

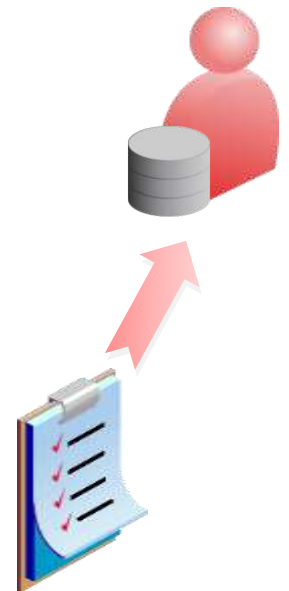
SQLの遅延を未然に防ぐためには、
どのようなアプローチが必要だったのでしょうか？

原因分析

- AWRの分析
 - SQLが遅延している時間帯のTop5待機イベントに db file sequential read待ちが発生している。
- ASHの分析
 - db file sequential read待ちが発生しているSQLは今回遅延しているSQLであることがわかった。

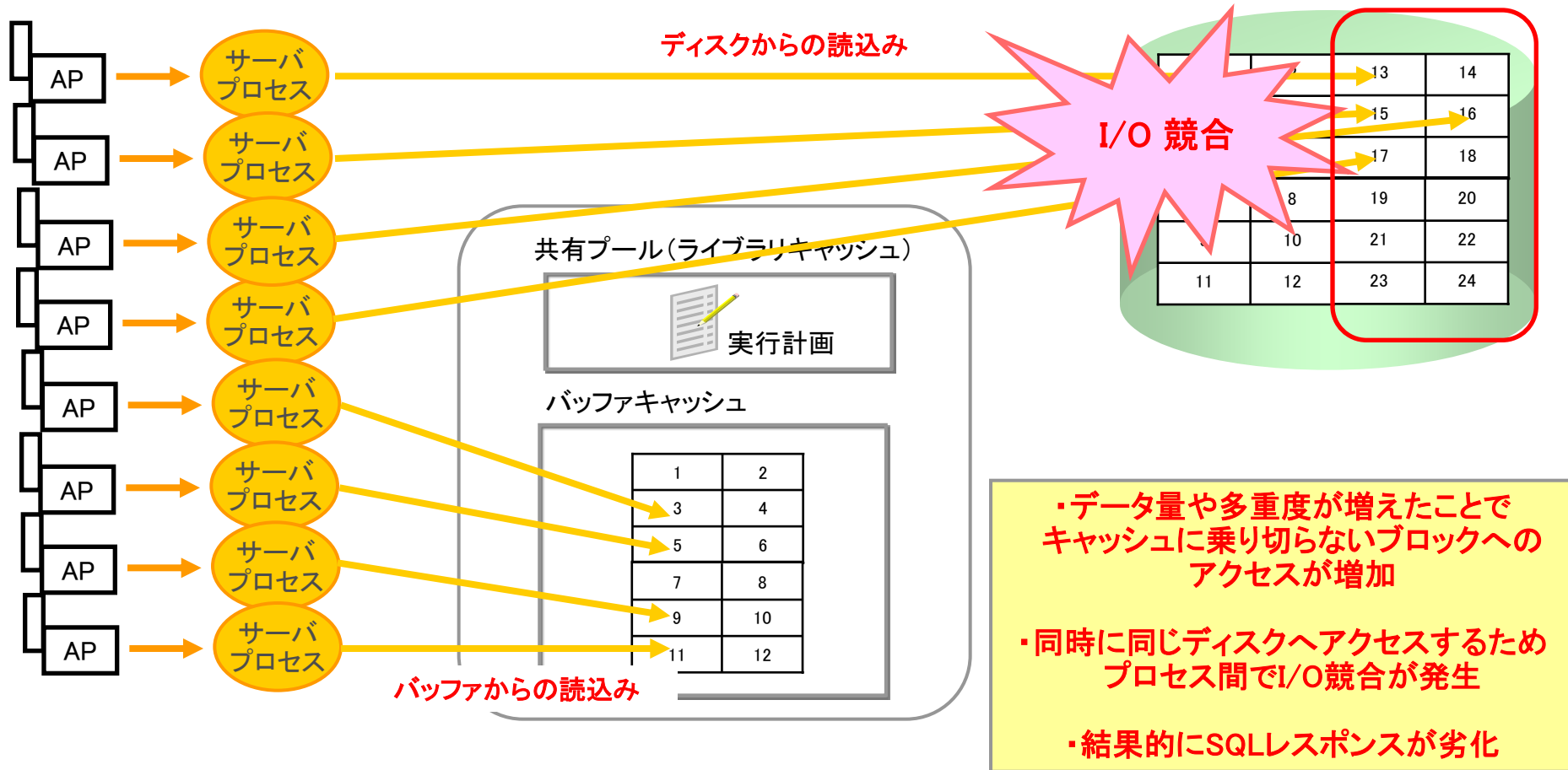
db file sequential read
待ちが発生している。
db file sequential readで待機して
いるSQLは遅延しているSQLだ！

なぜ、本番運用が始まるまで気付けなかったのでしょうか？
テストも実施していたのに、なぜ気付けなかったのでしょうか？



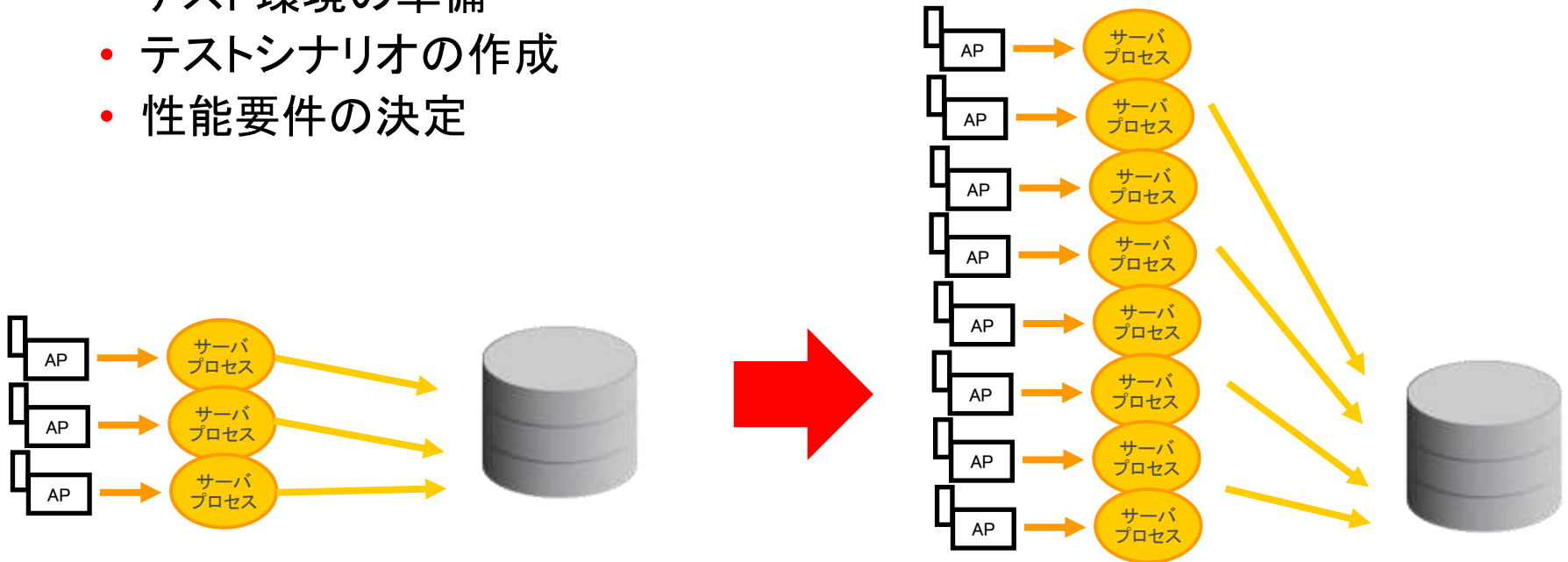
SQLが遅延した原因

セッション数が増えたことによって、プロセス間でI/O競合が発生した。



未然防止策検討

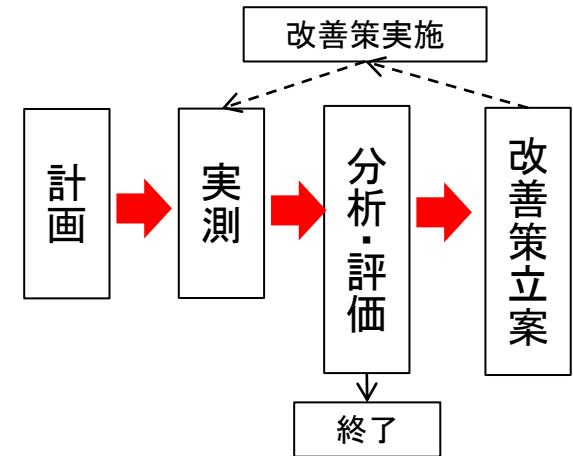
- テスト計画の進め方を見直すべきだった。
 - テスト環境の準備
 - テストシナリオの作成
 - 性能要件の決定



- 限界性能試験を行い、未然に気がついていれば対応策を検討できた

テストフェーズの考慮点

- 計画
 - テスト環境の準備
 - テストシナリオの作成
 - 性能要件の決定
- 実測
 - AWRやASHを取得(取得間隔について)
 - OS情報の取得(Oracle以外の情報の取得)
- 分析・評価
 - レスポンス要件を満たしているか
 - テスト後の分析を行っているか
- 改善策立案

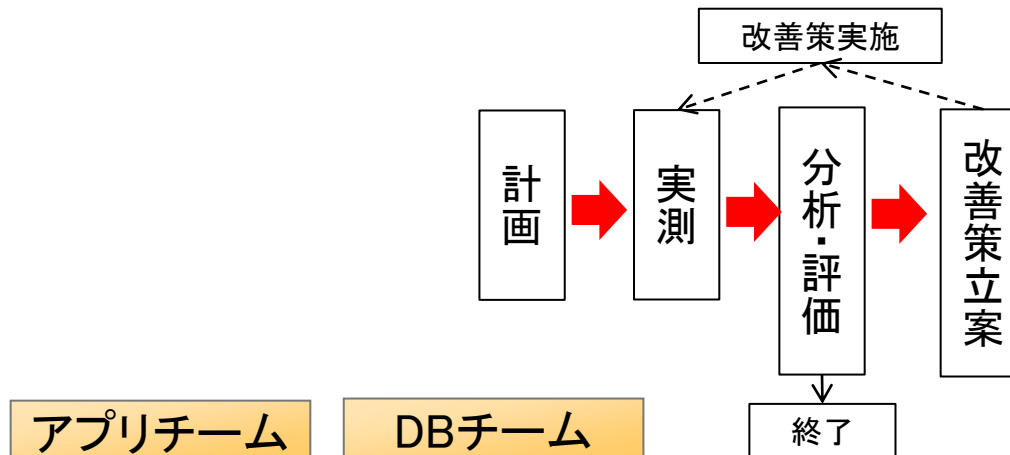


テストフェーズに十分な時間を取り、本番を想定したテストを実施する

テストフェーズのアプローチ

テスト計画が重要！

PDCAサイクルを回しながら、本番処理を想定したテストを実施する



- ・ テストのサイクルをしっかりと回せているか
- ・ 本番を想定したテストができているか
- ・ テストの性能要件をクリアするだけでなく、
長期走行や負荷増大時の性能要件まで考慮できているか

運用におけるアプローチ



シナリオ説明とポイント

- 運用フェーズのCase Study

A社ではサービスイン後、徐々にレスポンスが遅くなっているようです。
現状ではレスポンス要件を満たしているため、大きな問題になっておりませんが、
このままのペースで遅くなってしまうと、大きな問題になる可能性があります。

Case Studyのポイント

レスポンス要件を満たさなくなる前に、
未然に防ぐためにどのようなアプローチが必要でしょうか？

原因分析

A社では定期的にAWRを取得していたので、原因追求のために分析を行いました。

Load Profile

-----	Per Second	Per Transaction
-----	-----	-----
Redo size:	1,033,356.56	2,179,978.69
Logical reads:	40,550.44	85,545.58
Block changes:	7,648.16	16,134.63
Physical reads:	1.56	3.28
Physical writes:	122.98	259.44
User calls:	21.45	45.25
Parses:	2.50	5.47
Hard parses:	0.05	0.05
Sorts:	1.89	1.89
Logons:	0.00	0.00
Executes:	14.42	30.42
Transactions:	0.47	

SQLの実行回数

秒間14.42回→秒間15.13回(5%増加)

(略)

Top 5 Timed Events

-----	-----	-----	Avg %Total		-----
			wait	Call	
Event	Waits	Time (s)	(ms)	Time	Wait Class
-----	-----	-----	-----	-----	-----
CPU time		155		34.2	
log file sync	334	121	363	26.8	Commit
log file parallel write	205	54	263	11.9	System I/O
log file switch (checkpoint in	22	13	595	2.9	Configurat
control file parallel write	102	13	127	2.9	System I/O

Load Profile

-----	Per Second	Per Transaction
-----	-----	-----
Redo size:	1,006,177.50	2,116,770.26
Logical reads:	49,668.70	104,491.73
Block changes:	7,413.87	15,597.11
Physical reads:	17.21	36.21
Physical writes:	85.72	180.34
User calls:	22.69	47.74
Parses:	2.70	5.68
Hard parses:	0.05	0.10
Sorts:	0.90	1.90
Logons:	0.00	0.00
Executes:	15.13	31.84
Transactions:	0.48	

論理読み込み量

秒間40,550回→秒間49,668回(22%増加)

(略)

Top 5 Timed Events

-----	-----	-----	Avg %Total		-----
			wait	Call	
Event	Waits	Time (s)	(ms)	Time	Wait Class
-----	-----	-----	-----	-----	-----
CPU time		167		38.2	
log file sync	319	95	299	21.9	Commit
log file parallel write	203	52	256	11.9	System I/O
control file parallel write	101	11	108	2.9	System I/O
latch: cache buffers chains	285	10	35	2.9	System I/O

CPU時間

155秒→167秒(8%増加)

原因分析

Segments by Logical Reads DB/Inst: ORA/ora1020 Snaps: 1145-1146
-> Total Logical Reads: 5,474,917
-> Captured Segments account for 98.0% of Total

Owner	Tablespace Name	Object Name	Subobject Name	Obj. Type	Logical Reads	%Total
SCOTT	USERS	TABLE07	TABLE		4,238,464	77.42
SCOTT	USERS	INDEX04	INDEX		329,344	6.02
SCOTT	USERS	INDEX03	INDEX		287,168	5.25
SCOTT	USERS	INDEX02	INDEX		256,656	4.69
SCOTT	USERS	INDEX01	INDEX		248,912	4.55

Segments by Logical Reads DB/Inst: ORA/ora1020 Snaps: 1153-1154
-> Total Logical Reads: 6,478,487
-> Captured Segments account for 96.7% of Total

Owner	Tablespace Name	Object Name	Subobject Name	Obj. Type	Logical Reads	%Total
SCOTT	USERS	TABLE07	TABLE		5,190,544	80.12
SCOTT	USERS	INDEX04	INDEX		308,384	4.76
SCOTT	USERS	INDEX03	INDEX		268,880	4.15
SCOTT	USERS	INDEX02	INDEX			
SCOTT	USERS	INDEX01	INDEX			

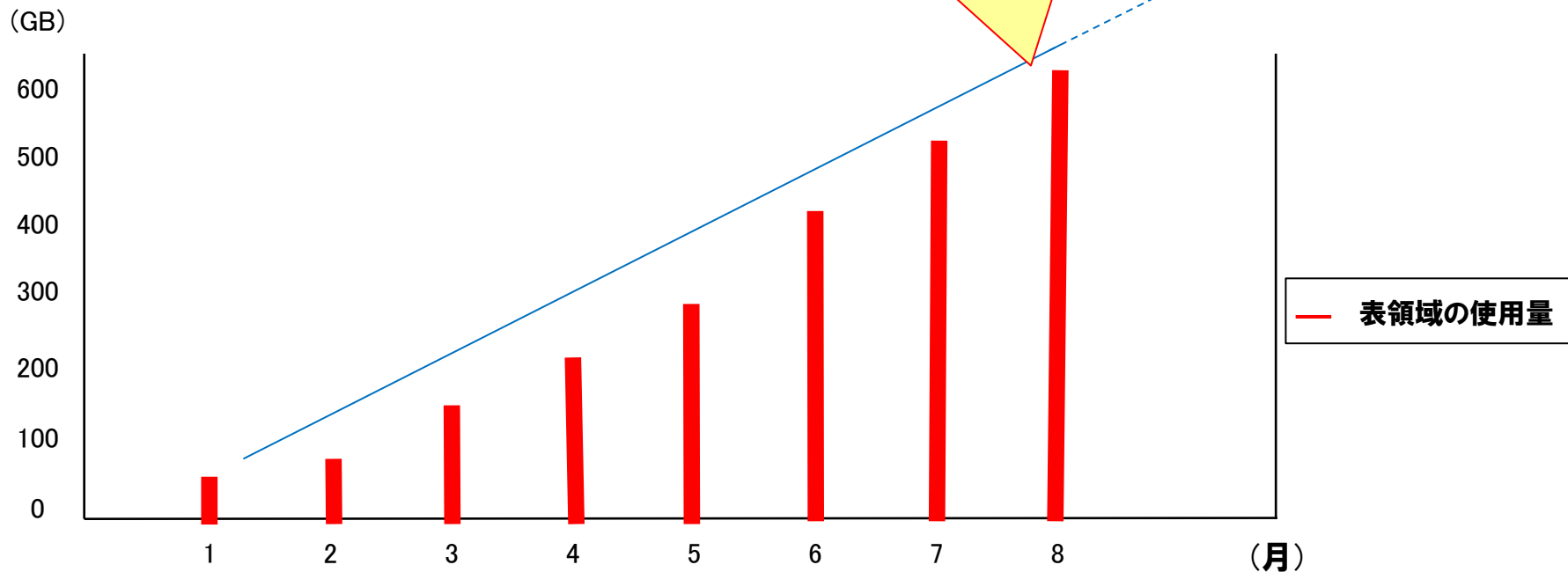
Segment統計 論理読み込み量
4,238,464 回 → 5,190,544 回

TABLE07表のサイズが増加していることが要因。
データ量の増加の妥当性について調査したところ、ユーザ発行のSQLによって、余分なデータが追加されていることが判明。

未然防止策検討

- 定常的な性能分析の実施
 - ベースライン活用
 - キャパシティ・プランニング

データ量の増加傾向。
妥当なものであるかのチェック。
リソースの枯渇時期の予測が可能。



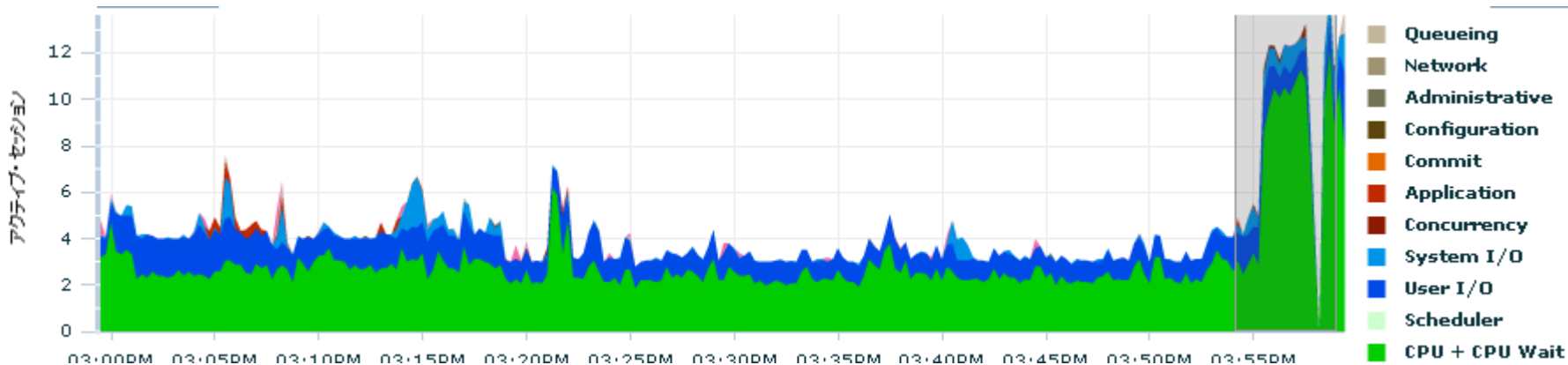
未然防止策検討

Enterprise Managerを使用して、視覚的に見ることが可能

表領域の使用率

表領域名	割当済サイズ(MB)	使用されている領域(MB)	使用されている割当済領域(%)	自動拡張	割当済空き領域(MB)	ステータス	データファイルタイプ	エクステンツ管理	セグメント管理
EXAMPLE	100.0	0.1	0.1	YES	99.9	✓	1 PERMANENT	LOCAL	AUTO
SYSAUX	1,118.6	997.0	89.1	YES	121.6	✓	1 PERMANENT	LOCAL	AUTO
SYSTEM	720.0	719.1	99.9	YES	0.9	✓	1 PERMANENT	LOCAL	MANUAL
TEMP	247.0	5.0	2.0	YES	242.0	✓	1 TEMPORARY	LOCAL	MANUAL

DBの処理傾向



運用で困らないために

- 性能問題を未然防止するために運用設計でやること
 - 性能監視
 - AWRの保存期間・保存場所
 - OSの性能情報
 - ジョブの開始終了時間
 - 領域監視
 - オブジェクトサイズの増加傾向
 - メンテナンス
 - 統計情報
 - 表・索引の再編成
 - 再圧縮

性能問題が発生しないように抜け漏れなく運用設計を行うことが重要

運用フェーズのアプローチ

運用設計をしっかりと行うことが重要！

運用フェーズだからこそ実際のデータ量や処理の変化に気づき、
性能遅延の未然防止を行うことができる

アプリチーム

DBチーム

- ・ 性能問題を未然防止するために運用設計で設計することを理解して、適切な運用設計を行う
- ・ 実際のデータ量や処理変動に気がつくことができるフェーズなので、日ごろからキャパシティプランニングを行う
- ・ データ量や処理変動に気づき、対応を行う

アプリチーム・DBチームの連携



シナリオ説明とポイント

- DBチーム・アプリチームの連携Case Study

A社では統計情報の運用方針を検討しています。

DBチーム、アプリチームどちらがどのように運用を行うか相談しています。

DBチームとしては、適切な統計情報の取得タイミングはアプリチームにしかわからないので、アプリチームに管理して欲しいと思っています。

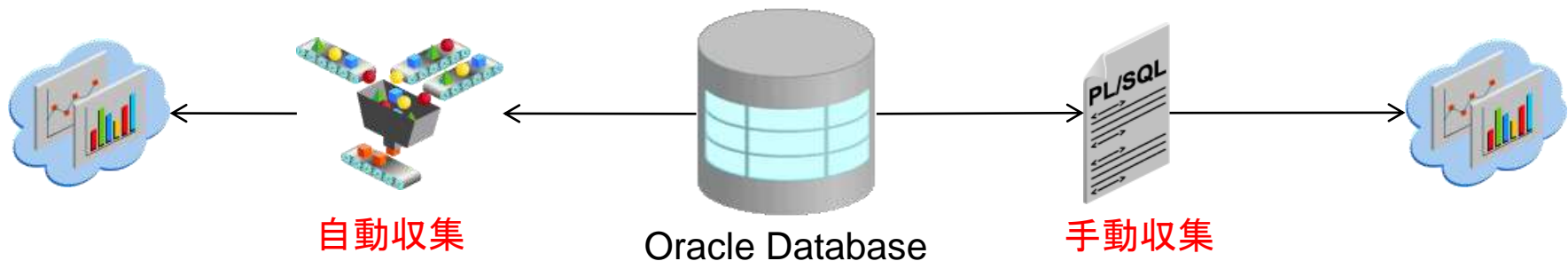
一方、アプリチームとしては、取得方法やツールをDBチームに作成して欲しいと思っています。

Case Studyのポイント

統計情報の運用方針を検討する際に、性能問題を未然に防ぐためにどのようなアプローチが必要でしょうか？

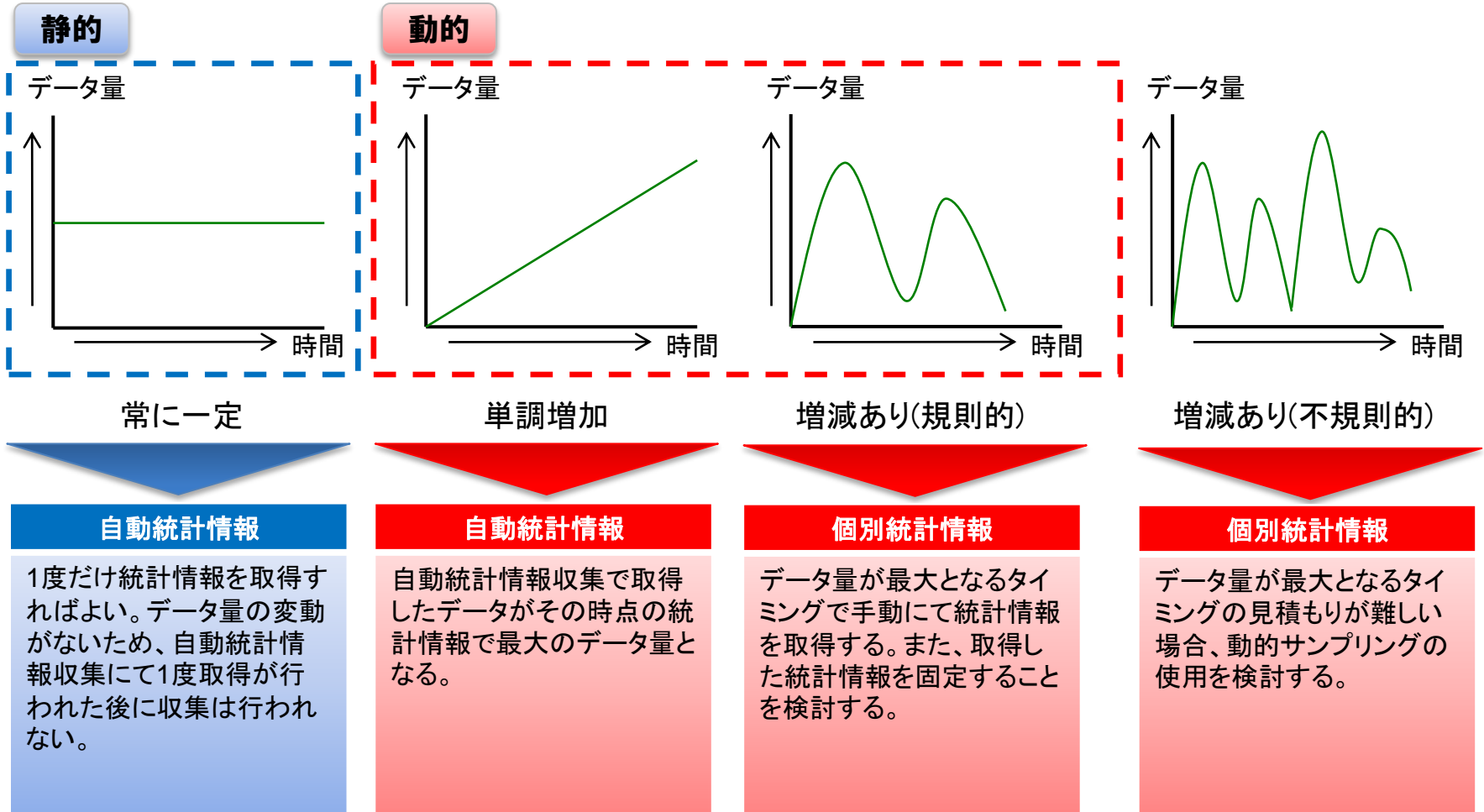
未然防止策検討

- アプリチーム
 - 適切なタイミングで統計情報を取得する
- DBチーム
 - 統計情報の取得手法・取得方法を提示する
 - ツールを作成する場合には、アプリチームが意識するポイントを少なくし、ツールの仕様を共有する



統計情報取得手法

- 表のデータ変動特性別に個別に統計情報取得手法を判断する



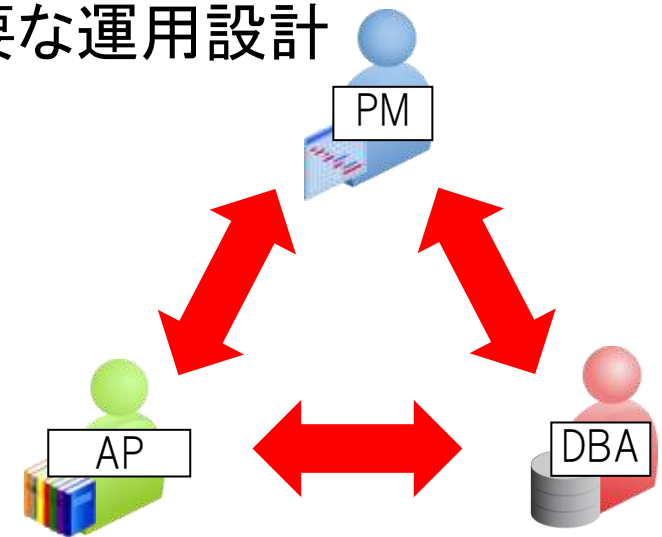
役割分担の明確化

- アプリチーム・DBチームの連携が必要な運用設計

- 統計情報管理
- 表・索引のメンテナンス etc

- 役割分担例

- アプリチーム
 - ジョブを組み込む
 - 表・索引のリビルドタイミングの決定
- DBチーム
 - ガイド(リビルド対象の考え方)・ツールの作成



作るべきガイド・ツールなどが抜けて落ちてしまうと性能品質が下がってしまうので、DBチームとアプリチームの役割分担・連携を行い、運用に抜け漏れがないように気をつけましょう。

役割分担時に、PMが関わることも重要です。

Summary



Summary

目的

性能問題を未然に防ぐアプローチを理解する

ゴール

- ・ 設計/開発におけるアプローチを身につける
 - ・ 業務要件を満たすアプリを設計・開発するだけでなく、DBの性能を考慮した設計・開発を行うことが重要！
- ・ テストにおけるアプローチを身につける
 - ・ テスト計画が重要！PDCAサイクルを回しながら、本番処理を想定したテストを実施する。
- ・ 運用におけるアプローチを身につける
 - ・ 運用設計をしっかりと行うことが重要！運用フェーズだからこそ、実際のデータ量や処理の変化に気づき、性能遅延の未然防止を行うことができる。
- ・ アプリチームとDBチームの連携を身につける
 - ・ 作るべきガイド・ツールなどが抜けて落ちてしまうと性能品質が下がってしまうので、DBチームとアプリチームの役割分担・連携を行い、運用に抜け漏れがないように気をつけましょう。役割分担時に、PMが関わることも重要です。

OTNセミナーオンデマンド

コンテンツに対する
ご意見・ご感想を是非お寄せください。

OTNオンデマンド 感想



http://blogs.oracle.com/oracle4engineer/entry/otn_ondemand_questionnaire

上記に簡単なアンケート入力フォームをご用意しております。

セミナー講師/資料作成者にフィードバックし、
コンテンツのより一層の改善に役立てさせていただきます。

是非ご協力をよろしくお願いいたします。

OTNセミナーオンデマンド

日本オラクルのエンジニアが作成したセミナー資料・動画ダウンロードサイト

掲載コンテンツカテゴリ(一部抜粋)

Database 基礎

Database 現場テクニック

Database スペシャリストが語る

Java

WebLogic Server/アプリケーション・グリッド

EPM/BI 技術情報

サーバー

ストレージ



超入門! Oracle データベースって何
再生時間: 60分

100以上のコンテンツをログイン不要でダウンロードし放題

データベースからハードウェアまで充実のラインナップ

毎月、旬なトピックの新作コンテンツが続々登場

例えばこんな使い方

- 製品概要を効率的につかむ
- 基礎を体系的に学ぶ/学ばせる
- 時間や場所を選ばず(オンデマンド)に受講
- スマートフォンで通勤中にも受講可能



毎月チェック!



コンテンツ一覧 はこちら

<http://www.oracle.com/technetwork/jp/ondemand/index.html>

新作&おすすめコンテンツ情報 はこちら

<http://oracletech.jp/seminar/recommended/000073.html>

OTNオンデマンド



オラクルエンジニア通信

オラクル製品に関わるエンジニアの方のための技術情報サイト

オラクルエンジニア通信 - 技術資料、マニュアル、セミナー

Oracleエンジニアのための技術情報サイト by Oracle Japan

新着情報を知りたい

技術資料を探したい

セミナーを受けたい

About

Oracleエンジニアの方がスキルアップしていただくために、厳選した情報をお届けしています

技術資料	<p>インストールガイド・設定チュートリアルetc. 欲しい資料への最短ルート</p>	アクセスランキング	<p>他のエンジニアは何を見ているのか？人気資料のランキングは毎月更新</p>
特集テーマ Pick UP	<p>性能管理やチューニングなど月間テーマを掘り下げて詳細にご説明</p>	技術コラム	<p>SQLスクリプト、索引メンテナンスetc. 当たり前運用/機能が見違える!?</p>

<http://blogs.oracle.com/oracle4engineer/>

オラクルエンジニア通信



The screenshot shows the top navigation bar of the oracletech.jp website. It features the 'oracletech.jp' logo in red and black, with the tagline '好奇心が、エンジニア人生を豊かにする。' below it. To the right is the 'ORACLE' logo, a search bar, and social media icons for Twitter, Facebook, LinkedIn, YouTube, and RSS. Below these are five red navigation buttons: '製品/技術情報', 'スキルアップ', 'セミナー', 'キャンペーン', and 'ちょっと一息'.

製品/技術
情報



Oracle Databaseっていくら？オプション機能も見積れる簡単ツールが大活躍

セミナー



基礎から最新技術までお勧めセミナーで自分にあった学習方法が見つかる

スキルアップ



ORACLE MASTER ! 試験頻出分野の模擬問題と解説を好評連載中

Viva!
Developer



全国で活躍しているエンジニアにスポットライト。きらりと輝くスキルと視点を盗もう

<http://oracletech.jp/>

oracletech



あなたにいちばん近いオラクル



Oracle Direct

まずはお問合せください

Oracle Direct



システムの検討・構築から運用まで、ITプロジェクト全般の相談窓口としてご支援いたします。
システム構成やライセンス/購入方法などお気軽にお問い合わせ下さい。

Web問い合わせフォーム

専用お問い合わせフォームにてご相談内容を承ります。
http://www.oracle.co.jp/inq_pl/INQUIRY/quest?rid=28

※フォームの入力にはログインが必要となります。
※こちらから詳細確認のお電話を差し上げる場合がありますので
ご登録の連絡先が最新のものになっているかご確認下さい。

フリーダイヤル

0120-155-096

※月曜～金曜
9:00～12:00、13:00～18:00
(祝日および年末年始除く)

ORACLE®

Hardware and Software **Engineered to Work Together**

ORACLE®