『日常監視』と『障害対応』へのアプローチ
（対応）体系

障害や性能悪化に対しては、現象が発生していないか感知行為を実施し、発生していた場合には、解消のための対応を行う必要がある

また、リソース資源等に対しては、許容範囲基準を事前に決めておき、枯渇が起きる前に予防的に対処を行うようにする

障害や性能悪化問題に対しての対応の第一歩

問題が起こっていることを、まず感知する

次に、それが何なのかを把握する必要がある

そして、その原因が何なのかを追究する

そのためには、『どのように発見』するのか。そして、どの SQL 文が遅れているのかを特定する

【参考情報】

自動感知（アラート監視）が実施できる内容

- メトリック監視による異常値の自動監視
- 〃　異常終了などの障害発見
- しきい値を下回った時の修正処置の自動実行
- 不具合発見用トリガーの作成し、その結果の監視

感知のための考え方

1）.『発生しているのは何か』という現象（感知対象）の種類

- リソース資源の使用量が、限界に達している
- OLTP（オンライン要求）処理が想定レスポンス時間内で返答を出さない
- バッチ処理の単位時間処理件数が少ない
- バッチ処理が時間範囲内で終了しない
- データベース処理の異常終了が発生した
- Oracle システムに対する攻撃が行われている

2）. 監視対象項目の種類

- 活動状態監視（リソース使用量）

時間変動項目

- メモリ空容量
- CPU 使用率
- ディスク使用率
- バッファキャッシュ・ヒット率
- 待機イベントの発生割合と長時間実行 SQL 文

不可逆計数値項目

- ストレージ使用量
- 表領域使用量

レスポンス計測項目

- ターンアラウンド処理量
- レスポンス時間
- 非フリーズ状態確認

- 障害監視
- セキュリティ監視
- 活動記録保存の実施

3）.『どのように発見』するのかという方法

- ASH 動的ビューへの Select 文による長時間実行中の SQL 文の洗い出し
- 〃 異常現象の把握
- 確認用 Select 文（ロック状況、V$SESSION など）の発行による確認
- スクリプト作成による統計情報の履歴記録の調査
- 実行結果およびログ情報への定期的な内容確認
- アラート・ログの定期的な監視
- メトリック項目を使った自動感知（アラート監視）
- ~~統計情報（Statspack、AWR）などのレポートドキュメントの確認~~

アプローチ（対応）しなければならない現象、もしくは対象

A. メトリック項目への予防保全

リソース資源の空容量等に対し許容範囲基準を設定し、その値を下回る前に予防的な対応をとる

B. 異常終了への対応

以下のような処理の障害を対象とする

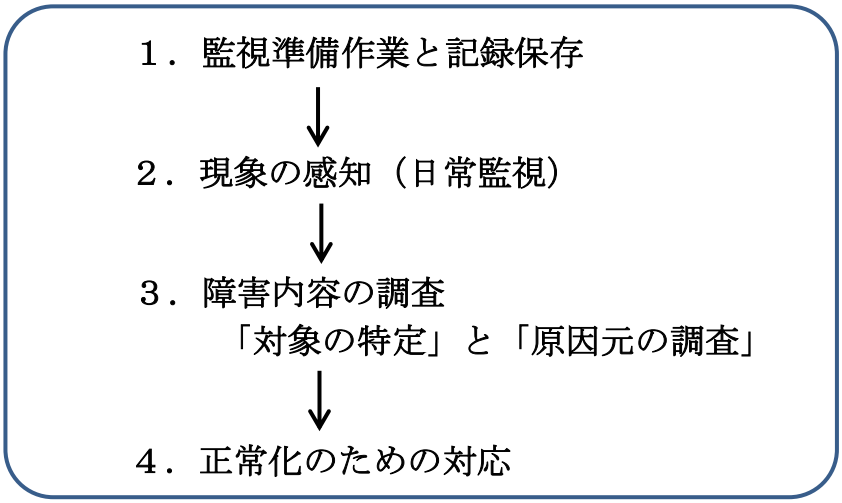
- バックグラウンドのプロセスが異常終了する
- サーバー（ユーザー要求処理）のプロセスが異常終了する
- 時間指定自動起動スクリプトの異常終了
- 手動スクリプト等でのバッチ処理の実行結果の異常
- バックアップ処理の異常終了
- Oracle システムに対する攻撃が疑われる異常終了メッセージ
- ~~SQL コマンドの実行において、エラーが起きる~~

C. 性能障害（パフォーマンス悪化）への対応（フリーズを含む）

以下のような処理が性能（パフォーマンス）についての現象を対象とする

- バッチ処理において、データの処理が予定時間を過ぎても終了しない
- オンライン要求による OLTP 処理が、レスポンスが悪く要求時間までに終了（返答）しない
- Oracle 全体に何らかの問題が発生して、SQL 処理が先に進んでいかない
- 特定の SQL 処理が Oracle システムの仕組みであるロックを行い、他の処理の進行の邪魔を行い、処理が先に進んでいかない

対処アプローチの順序項目



改善に向けての実作業項目

１．監視準備作業と記録保存

A. メトリック項目への予防保全

【実作業】

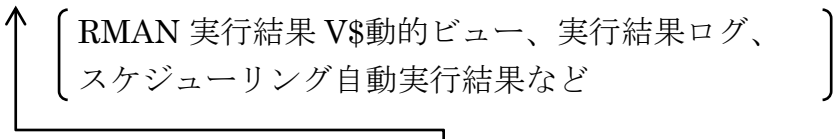
・リソース資源の空容量等に対し許容範囲基準を決めておく

・メトリック項目のしきい値に対して、アラート警告設定して自動監視を行う

B. 異常終了への対応

【実作業】

・障害が発生した時に、『障害が発生したこと』が感知できる基準や現象（の方法）を定めて、その資料がどれであるかを明確にしておく



・障害発生を明確するためのその資料を、確認する定期的なタイミングを決めておく

・アラート警告のためのメトリック設定

ORA-******の検知の方法
性能劣化の検知
ハングアップ（フリーズ）の検知方法

・アラート警告出力のメール発報へのトリガー設定

検知した内容の通知

C. パフォーマンス障害への対応

【実作業】

・遅延が発生した時に、『遅延が発生したこと』が感知できる基準や現象（の方法）を定めて、その資料がどれであるかを明確にしておく

・原因を調査するための元資料となるデータ（ASH 動的ビューなどの定期間隔実行での結果）を記録保存するためのスクリプトなどの作成

V$SESSION
処理トランザクション数

・作成したスクリプトを、スケジューリングにて定期的に実行させる

・Statspack や AWR が動作する環境を準備する

・AWR スナップショット採取、Statspack スナップショット採取を、スケジューリングにて定期的に実行させる

これにより、後になって原因を調査するための元資料となるデータが記録保存される

２．現象の感知（日常監視）

A. メトリック項目への予防保全

【実作業】

- アラート警告設定されたメトリック項目が出力されていないかアラート・ログを監視する
- アラート警告設定が出来ない項目に対しては、定期的に値のチェックを行い許容範囲内であることを確認する

B. 異常終了への対応

【実作業】

- アラート警告設定されたメトリック項目が出力されていないかアラート・ログを監視する
- その他の異常メッセージが出ていないかアラート・ログを確認する
- １．の B. で定めた基準や現象が発生していないかを、対象の資料で確認する

（RMAN 実行結果 V$動的ビュー
実行結果ログ、
スケジューリング自動実行結果）

C. パフォーマンス障害への対応

【実作業】

- ユーザーからの現象の申請を聞き取り、パフォーマンス障害が発生しているかの判断を行う
- １．の C. で定めた基準や現象が発生していないかを、対象の資料で確認する
- バッチ処理の処理時間、および負荷が高い時間帯の OLTP 処理件数やレスポンスについては、規則的に調査しておく
- 「リアルタイム監視」画面による Oracle インスタンスの SQL 実行状態監視を行う
- 「トップ・アクティビティ」画面による Oracle インスタンスの SQL 実行監視を行う

３．障害内容の調査

A. メトリック項目への予防保全

・現象の洗い出し

複数メトリック項目での並行発生が考えられるので、相互判断するために許容値の限界を超えた（アラート警告された）**すべての**項目の値を一覧提示する

・根本的な原因の推測 と その処理の特定

1）前日のメトリック項目の**値比較**や、発生前後の**時系列での変動**を調査する

2）発生時に、動作していた処理の SQL 文を洗い出す

3）根本的な原因が『何なのか』を推測し、それを引き起こした処理が『どれであったのか』を特定する

> **注）** 複数の処理を実行した結果であり、特定の SQL 処理だけが起因するものでない場合もある
> 例えば、長期に渡るディスク書込みの結果のストレージ不足などの場合では、根本的な原因は特定できるが、その SQL 処理については多数が関係していて１つに特定できない場合もある

4）何が根本的な原因（リソースの枯渇現象）を招いたかの発生理由を追究する

> 例えば、根本的な原因がディスク残容量不足とするならば、その理由は、
> 発生させた処理が該当する
> 「イレギュラー処理でデータ量が増大した」などとなる

・二次的現象の検証

1）原因元の現象と二次的に発生した異常現象を選別する

2）根本原因から推測して、二次的に発生した現象の発生との関連性を検討し、現象の根拠と発生の正当化の検証を行う

B. 異常終了への対応

・実行されていた処理の洗い出し

何を処理していたら異常終了が起こったかの『何を』を示す処理内容を提示する

複数で異常が発生している場合は、そのすべての処理について提示する

スクリプトの全文コード

SQL 処理文のコマンド内容

・根本的な原因の推測 と その処理の特定

1）異常終了したすべての処理からエラー・コードを洗い出し、その原因を調べる

2）すべての原因の中から根本原因となる原因を推測し、それを引き起こした処理（異常終了した処理ではなく、影響を与えた処理）を特定する

注） 複数の処理を実行した結果であり、特定の SQL 処理だけが起因するものでない場合もある

例えば、長期に渡るディスク書込みの結果のストレージ不足などの場合では、根本的な原因は特定できるが、その SQL 処理については多数が関係していて１つに特定できない場合もある

・二次的現象の検証

1）原因元の現象と二次的に発生した異常現象を選別する

2）根本原因から推測して、二次的に発生した現象の発生との関連性を検討し、現象の根拠と発生の正当化の検証を行う

C. パフォーマンス障害への対応

・現象の洗い出し

『**影響がどこに出ているのか**』を特定する

特定の SQL に限定 or システム全体できるのか
動的ビューへの Select 文の発行による長時間実行中の SQL 文の洗い出し
遅い SQL 群は、共通の規則グループでまとまった SQL なのかの確認
（待機クラス、特定アプリケーション、特定テーブルへのアクセスなど）

【実作業】

・Oracle 自体がフリーズ状態でないかを確認する
・アクティブ状態のタスクの数の把握する
・待機イベントの発生件数分布の状況を把握する
・V$SESSION にて、長時間実行中の SQL 文を洗い出す

・根本的な原因の推測 と その処理の特定

1）システム全体（OS 環境、Oracle インスタンス）を捉えて、悪い状態になっていないかの確認をとる

2）根本的な原因が『何なのか』を推測し、それを引き起こした処理が『どれにあったのか』を特定する
すなわち、二次的に発生した現象との選別を行い、根本的な原因元の**処理**がどれであったのかを特定し、原因を推定する。

注） 複数の処理を実行した結果であり、特定の SQL 処理だけが起因するものでない場合もある
例えば、大量に発生したディスク読込みによるディスク I/O へのアクセス待機イベントについては、レスポンス遅延の根本的な原因は特定できるが、その SQL 処理については多数が関係していて 1 つに特定できない場合もある

3）何が根本的な原因（リソースの枯渇現象）を招いたかの発生理由を追究する

例えば、根本的な原因がディスク残容量不足とするならば、その理由は、
発生させた処理が該当する
「イレギュラー処理でデータ量が増大した」などになる

【実作業】

－システム全体の把握－

・リソース（CPU、メモリ、ディスク I/O）の使用負荷の確認
・待機イベントの発生件数の推移を時間的に考察する
・待機タスクのイベントと待機元タスクとの関係把握
・待機イベントの種類による障害もしくはボトルネックの推測と検証
・（バッファ、キャッシュ）メモリの空き状態の調査
・メモリ・チューニング・アドバイスの状況調査
・ディスク残容量の調査
・表領域空容量の調査

－原因元処理の特定と原因追求－

・レコードの排他制御ロックによる待機時間発生による遅れの把握
・レコードの排他制御ロックが原因ならば、排他制御元の SQL 文の特定する
・待機イベントによる長時間処理なら、待機イベントの発生要因を調査する
・前日との値比較や、発生前後の時系列での変動を調査する
　AWR レポート、Statspack レポート
・どの SQL 文が原因元となっているのかを推測する
・排他制御ロックによって待機している SQL 文を洗い出す
・ディスク・アクセス時間の長時間化ならば、故障元ディスク装置の特定する
・データ量の増加に伴うディスク容量不足なら、データ量が増加した原因の追究

・二次的現象の追究

1）原因元の現象と二次的に発生した異常現象を選別する
2）根本原因から推測して、二次的に発生した現象の発生との関連性を検討し、現象の根拠と発生の正当化の検証を行う

> **注）** バックグラウンド・プロセス処理の負荷やハードディスクの故障などの場合もあるので、原因元は存在するが、原因元処理については、特定の SQL 処理に対応できない場合もある

【実作業】

・V$SESSION と V$LOCK を使ってレコードの排他制御ロック状況の調査し、ロック元とロック先のセッション関係を把握する
・長時間処理が終わっていない SQL 文に対して、CPU 使用時間と待機時間を把握する
・長時間処理が終わっていない SQL 文に対して、待機イベントの時間状況を把握する
・システム全体での待機イベントの発生状況を把握する

４．正常化のための対応

３．で調べた原因元への改善方法を調べ、対応することにある

A. メトリック項目への予防保全

根本原因となっている現象に対しての対応を行う

【実作業】

- 初期化パラメータのチューニング
- メモリ使用量のチューニング
- ハードウェア・リソース（CPU 追加、ストレージ追加、メモリ追加など）の拡充
- スケールアップ、スケールアウトの立案

B. 異常終了への対応

調査結果に基づいた根本原因の取り除きを行う

【実作業】

- データベース・ブロックのエラーの修復
- SGA、PGA メモリの拡張
- フラッシュ・リカバリ領域の拡張

C. パフォーマンス障害への対応

現象の根本となっている原因や処理に対しての改善対応を行う

【実作業】

- SQL チューニング・アドバイザ―の実行
- SQL アクセス・アドバイザ―の実行
- SQL 実行計画の変更
- SQL 文へのヒント句指定
- 実行計画の固定化
- 個別インデックスの作成
- 初期化パラメータのチューニング
- メモリ使用量のチューニング
- オプティマイザ統計情報の取得
- テーブルの列に対してヒストグラムを作成
- オブジェクト（テーブル、インデックス）の断片化の解消
- オブジェクト（テーブル、インデックス）のメモリ常駐化
- オブジェクトのディスク分散
- テーブルのパーティション化
- マテリアライズドビューの作成
- 処理のパラレル（並行多重）化
- スケールアップ、スケールアウトの立案

【調査資料】

性能障害（パフォーマンス問題）に係る５大調査事項

性能障害（パフォーマンス問題）に関しては、まず Oracle システムの稼働状態を調査し、全体の処理状況を把握して、問題がどこにあるのかを特定する

その上で、問題部分に対して適切な対応を実施する

［問題部分および原因の特定］

OS リソースの状況把握
（CPU 使用状況、ディスク・アクセス状況、メモリ使用状況）

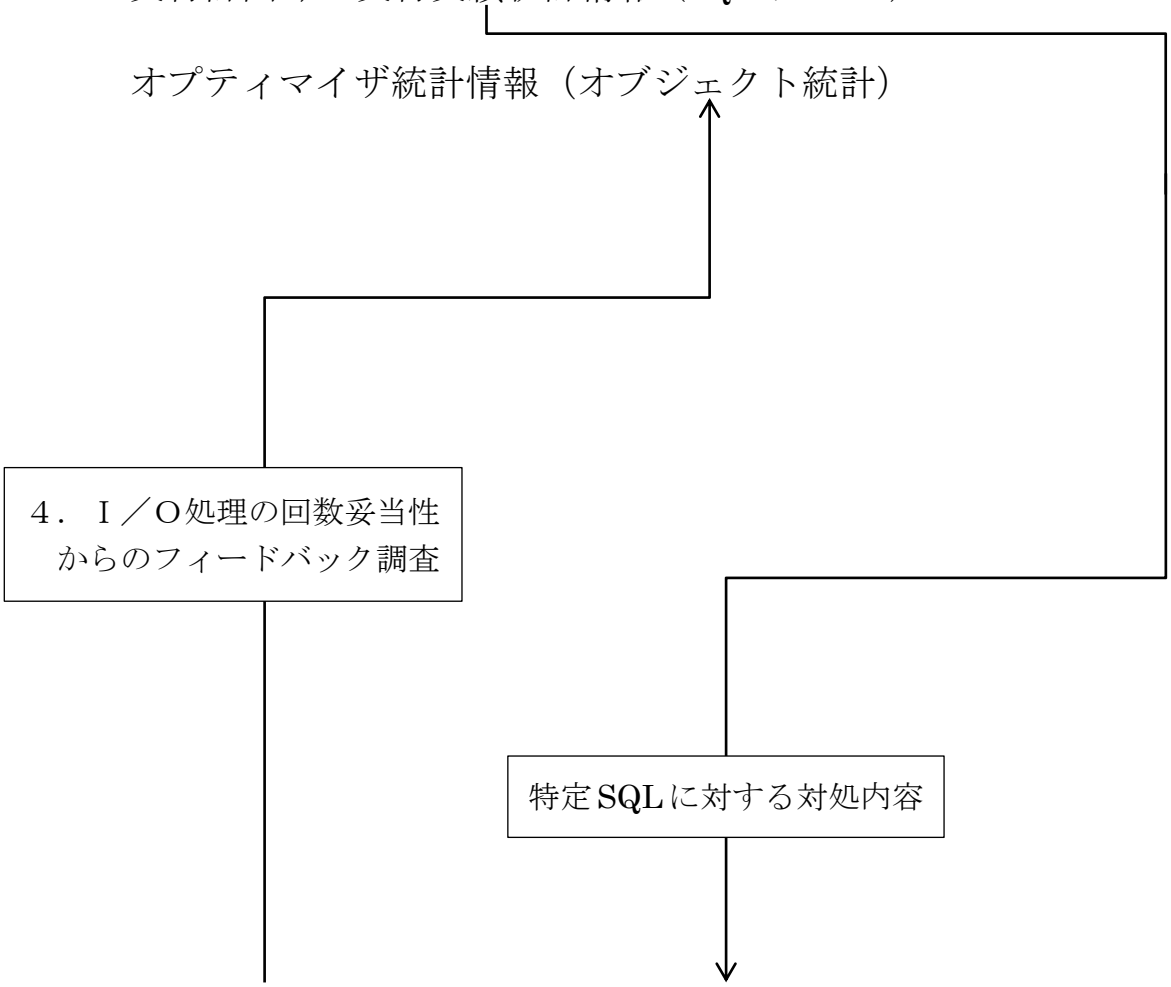
パフォーマンス統計
（AWR レポート、Statspack レポートによる SQL 実行実績統計）

待機イベント

［問題部分の把握後の詳細調査］

実行計画 ／ 実行実績統計情報（SQL トレース）

オプティマイザ統計情報（オブジェクト統計）



【判断材料】

特定 SQL に対する性能障害（パフォーマンス問題）への対処方法の指針

遅延原因の対象 SQL 文を見つけた後のパフォーマンス改善内容

（ SQL 文に対する実行時間短縮のための対処方法 ）

SQL 処理が長時間になっているのを短縮させるためには、**以下の内容を調査し**てその問題の原因を特定し、ボトルネックとなっているポイントへの対処を行う。
その結果、パフォーマンスが向上する

パフォーマンス対応のための検討内容

1．SQL の処理手順（SQL 実行計画でのデータへのアクセス方法）

・索引を使ったアクセスを実行しているか
・テーブルへのフルアクセスの方が、効率が良くないか
・テーブルへのアクセス順序や結合方法は、適切か

2．ボトルネック部分の発見

・ボトルネックとなっているアクセスパス（実行計画部分）は、何処なのか

3．ＣＰＵ負荷 or 待機イベント時間 の判断

・CPU 負荷が高い
→ CPU 負荷を下げる SQL 文への変更やオブジェクト（インデックス、パーティション・テーブル、マテリアライズドビュー）の作成
・待機イベントの割合が高い
→ 待機イベントの待機時間を短縮するような方策を実施

4．Ｉ／Ｏ処理の回数妥当性

・I/O リクエスト数や読取りバイト数は、妥当か
表や索引が断片化していると、I/O リクエスト数や読取りバイト数は大きくなる（回数からの判断は無理、各オブジェクトの統計情報を取得すること）

5．メモリ／一時表領域の使用における処理時間への影響

・ハッシュ結合やソート処理で使用するデータのワーク・エリアは、PGA メモリを使用しているのか、一時表領域を使用しているのか
一時表領域を使用している場合で、「direct path read temp」、「direct path write temp」が大半を占めている場合、PGA メモリの拡大やハッシュ結合の結合手順の変更を検討する