オラクル移行

準備、移行、最適化
Oracle スキーマと PL/SQL コードの準備と最適化
Visual Expertを使用します。

はじめに

Oracleデータベースの新バージョンへの移行は、複雑なプロジェクトです。単にデータを移行するだけでなく、大量のPL/SQLコードを適応させ、時間の経過とともに蓄積された時代遅れの要素を一掃し、リグレッションが発生しないようにする必要があります。そのため、オラクルの開発者とDBAは、プロジェクトの各段階で細心の注意を払わなければなりません。

幸いなことに、専用のツールを使用することで、作業をはるかに簡単に行うことができます。Visual Expertは、Oracle移行の各フェーズに自信を持って取り組むためのツールの組み合わせを提供します。Visual Expert は、静的PL/SQLコード解析ツールであり、既存コードの調査、理解、文書化に最適です。

この記事では、オラクル移行の典型的な段階を取り上げ、それぞれの段階で遭遇する困難と、このツールが特定の機能のおかげでどのようにそれを克服するのに役立つかを検討します。

主に内部PL/SQLコード（ストアドプロシージャ、トリガなど）を含むOracle 11gデータベースをOracle 19c（または21c）に移行した場合を例に考えてみましょう。

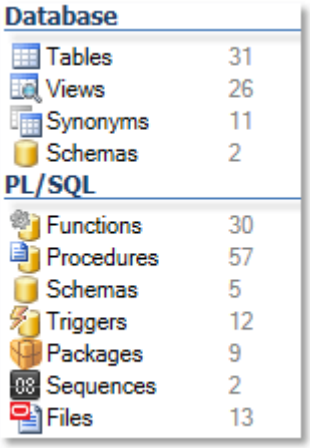
内容

ステップ1：コードとオブジェクトの量を評価する

目的：分析範囲を明確に設定し、移行作業量を見積もる。

課題移行プロジェクトの最初のステップは、作業の規模を評価することです。

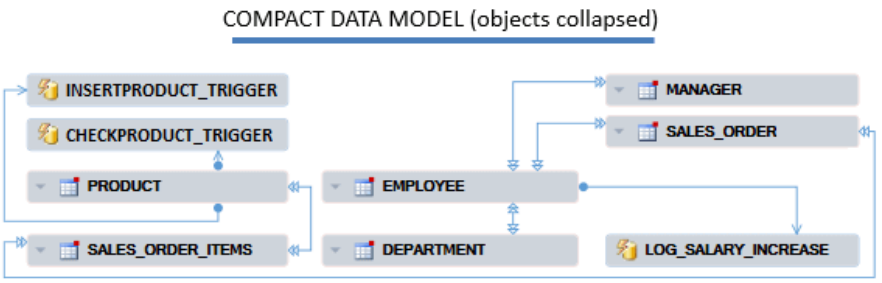
すべてのPL/SQLコード（パッケージ、プロシージャ、ファンクション、トリガ）と同様に、すべてのデータベースオブジェクト（テーブル、ビュー、インデックス...）をインベントリ化する必要があります。このボリュームの見積もりは、作業負荷と必要なリソースの計画に使用されます。

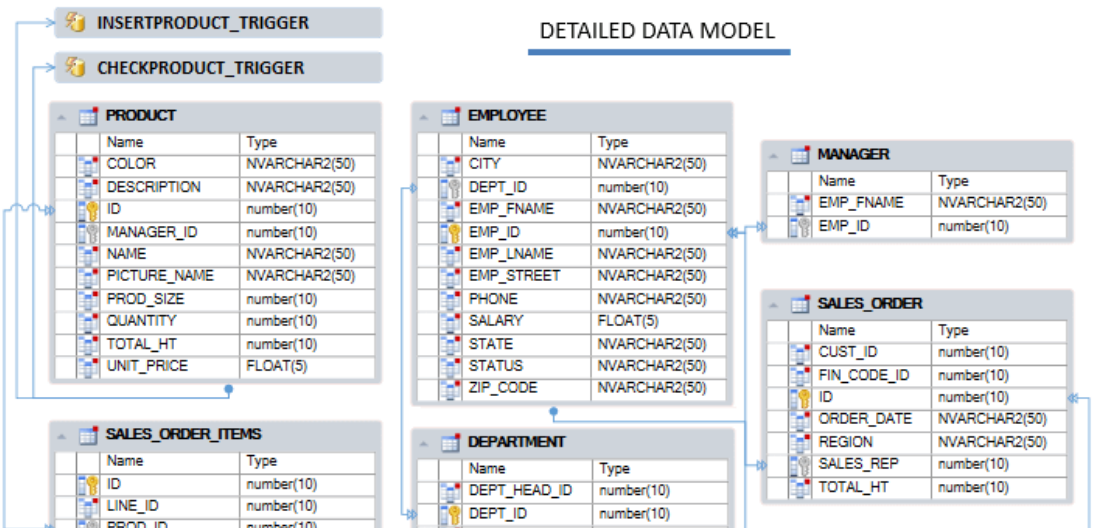


ツールがなければ、手作業でOracleカタログやPL/SQLコードを参照することになり、面倒でエラーが発生しやすい。Oracleデータベースには、何千ものオブジェクトと何百万行ものコードが含まれることがあります。最初から正確なグローバル・ビューを得ることは、アーキテクトやプロジェクト・マネージャーにとっての課題です。

- Visual　Expertは、コードとオブジェクトの自動インベントリを提供します。データベースが分析されると、このツールは詳細な**コード・メトリクスを計算**できます。PL/SQLコードの総行数（コメントと実際の命令を区別）、プロシージャと関数の数、パッケージの数、トリガーの数などです。また、タイプ別にオブジェクトをリストします。また、タイプ別に**オブジェクトをリスト**します。

- Visual Expert は、**既存のデータベースのモデリングにも**使用できます。Visual Expertは、**データモデルの複雑さ**（テーブルとリレーションシップの数、機能サブドメインへの分解など）を示す**エンティティ-リレーションシップ図を**自動的に生成します。アーキテクトはこのようにして、モデルのある部分が非常に密集しているのか（相互に接続された多数のテーブル）、それとも孤立しているのかを確認することができます。こうしてアーキテクトは、モデルのある部分が非常に密集している（相互接続されたテーブルが多い）のか、逆に孤立しているのかを見ることができる。このようにして、モデルのどの部分を移行するのが最も重要かを評価することができます（密な部分は、多く





ステップ2：コードのクリーンアップ

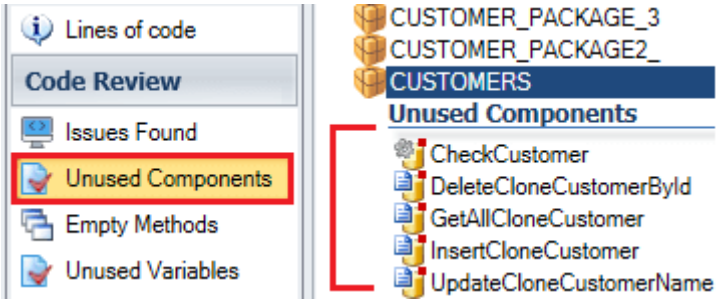
目的：技術的負債とリスクを軽減するために、未使用の項目を削除する。

困難：移行する前に、コードとオブジェクトを整理するのが賢明である。もう使われていない要素（呼び出されなくなった古いプロシージャ、忘れ去られたテンポラリテーブルやテストテーブル、時代遅れのビューやシノニムなど）を、なぜ新しいバージョンに移行するのか？

それらを移行することは、無駄に時間を消費し、リスクを増大させるだろう（例えば、時代遅れのパッケージには19cと互換性のないコードが含まれているかもしれない）。

しかし、手作業で未使用のオブジェクトを検出することは、大規模なデータベースでは非常に困難である。開発者は、隠れた機能が壊れることを恐れて、コードを削除したがらない。予防措置として、不必要な労力と重い技術的負債を犠牲にして、すべてを移行することが多い。

Visual Expert は、Oracle データベース内の「*デッドコード*」をリストアップすることで、このタスクを支援します。この情報により、チームは Oracle 19c に**移行しない**オブジェクトを決定し、不要な要素を排除することができます。



- 依存関係の完全な静的解析により、**潜在的に使用されていないPL/SQLオブジェクト**（他のプロシージャ、トリガ、または既知の外部アプリケーションによって参照/呼び出されることのないオブジェクト）**を特定することが**できます。
- また、**コードで使用されたことのないテーブル**（SELECT/INSERT/UPDATE/DELETEクエリで参照されたことのない**テーブル**）を発見することもできます。
- 最後に、空のプロシージャ、統合すべき重複コードなど、クリーンアップすべき他の要素にもフラグを立てる。

ステップ3：交換すべき旧式コンポーネントの特定

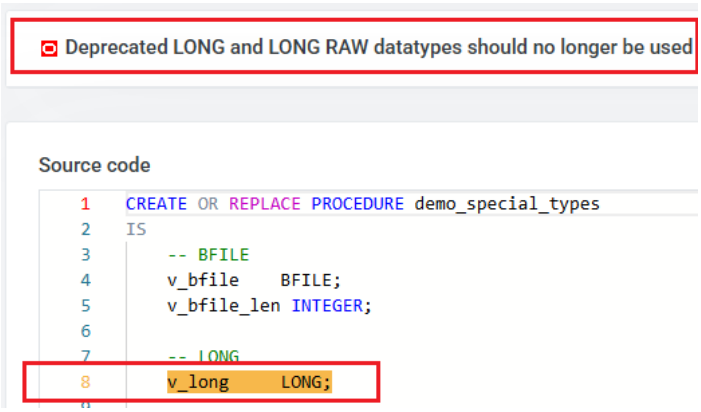
目的：移行前に交換すべき減価償却技術を特定する。

課題：未使用の要素に加えて、チームは**必要だが時代遅れの要素を**処理する必要がある。

時間の経過とともに、Oracle は特定の機能を廃止し、新しい機能を導入します。例えば、**DBMS_JOB 関数は** DBMS_SCHEDULER に取って代わられ、昇格しなくなりました。LONG のような特定の**データ型は** CLOB/BLOB に取って代わられ、廃止されました。したがって、レガシーでサポートされていない技術を維持することを避けるために、これらの要素を置き換えることを推奨します。

難しいのは、データベーススキーマやコードの中で、これらの旧式の要素が使用されてい るすべての場所を体系的に検出することである。見落としは容易に気づかれない可能性がある（例えば、技術的なテーブルに隠された LONG 型の列）。したがって、開発者とDBAは、必要な置換を計画するために、徹底的に検出しなければなりません。手作業で何千行ものコードをスキャンしたり、Oracle辞書を閲覧したりすることは、現実的な解決策ではありません。そのため、ツールが必要となります。

Visual Expertは、その綿密な依存関係分析のおかげで、このタスクを得意としている。



- Oracle 関数、パッケージ、サブプログラムなど、廃止されたオブジェクトを参照するすべてのプロシージャ、トリガ、関数、パッケージ、サブプログラム、および型をリストできます。
- LONGやDBMS_XMLGENパッケージなど、非推奨の型に基づいてテーブルやカラムをリストアップすることもできます。
- また、Visual Expertには自動コード検査エンジンも搭載されており、廃止された要素の使用を警告するルールがあります。
- このように、Visual Expertは**互換性スキャナとして**機能し、ベストプラクティスに従って既存のコードをふるいにかけ、テスト中に不愉快な驚きを避けることができます。

データをモデリングすることで、Visual Expert は、オラクルが提供する改善を利用するために近代化すべき**設計の古さを**特定するのに役立ちます。例えば

- 一部のテーブルでは、一意なシーケンス/識別子の代わりに複合識別子を使用している。
- 制約によって宣言されない関係もある（歴史的にはコードによって処理される）。
- おそらく、非常に長い、関係のないテキスト・フィールドはCLOBに変換できるのではないだろうか？

このように、**Visual Expert は**、コードの変更に加え、**構造的な変更について**考える材料を提供します。ダイアグラムには、計画された変更（新しい関係、新しいタイプ）を注釈として加えることができます。これらの要素は、次の段階の影響分析で使用されます。

ステップ4：変更の影響分析

目的：後退を避けるために、変化のすべての結果を予測する。

何を変更するか（置き換える関数、変更する列など）が決まったら、**その変更がどこに影響するかを**評価する必要がある。これが影響分析であり、移行中に何かを壊さないようにするために非常に重要である。

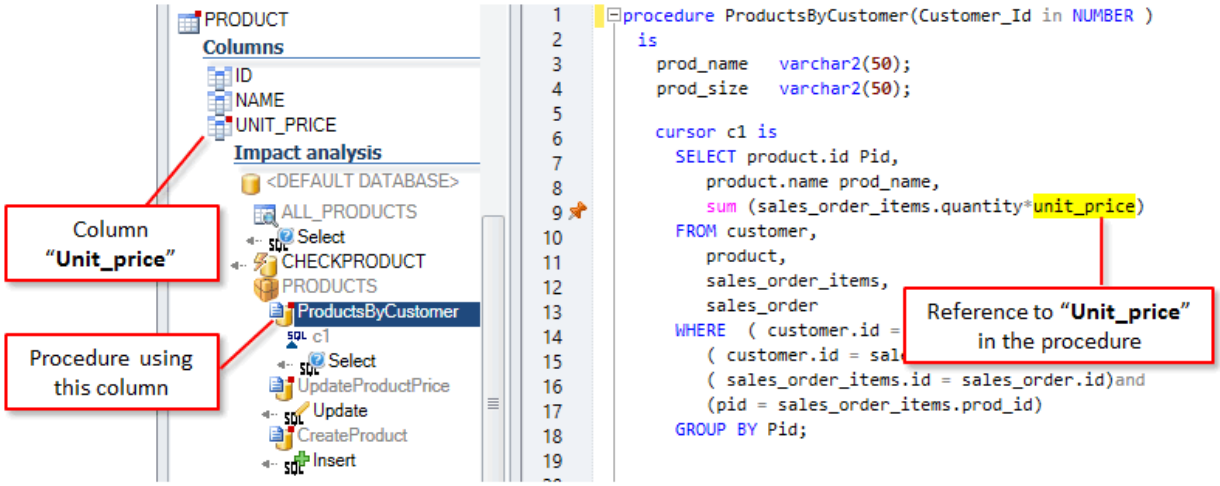
- 例えば、廃止された関数を置き換えるということは、その関数を呼び出すすべての手続きのコードを修正することを意味する。しかし、これらのプロシージャーは他の場所でも呼び出される可能性があります。これらすべての変更の結果をチェックする必要がある。
- 同様に、LONGカラムをCLOBに変更するには、このカラムを操作するプロシージャを変更し、上流/下流のコンポーネント（ビュー、インターフェースなど）をチェックする必要があります。

ツールがなければ、これらの影響をマッピングするのは手間がかかる。複数の依存関係（誰が何を呼び出し、誰がどのテーブル/カラムを使うか）のスレッドを手作業でたどらなければならない。チームが恐れているのは、**参照を見逃して**、テストや本番で、更新されていなかったコードの一部に問題があることを発見することだ。

ビジュアル・エキスパートは、*「X**を変更したらどうなるか*」といった質問に、**インタラクティブな影響分析で**答えます：

- オブジェクト（関数や列など）を選択して影響度分析を実行すると、Visual Expert は**そのオブジェクトを参照するすべてのオブジェクトを**一覧表示します。例えば *「'customer_name'列を変更した場合、どのプロシージャ、トリガー、ビュー、パッケージが影響を受けるか？どのスクリプトがそれを参照しているか*?

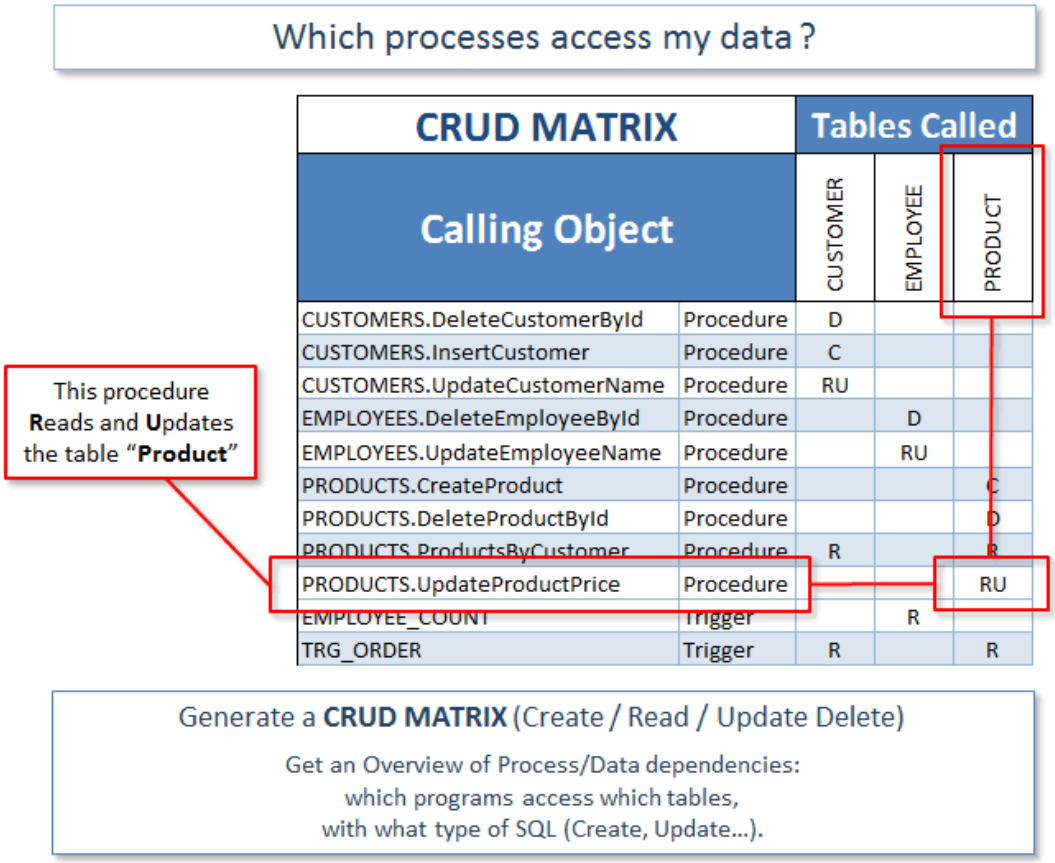
Where is a column used?



A column has changed: what's the Impact on my code?

Find all references to a column or table.
Make sure your application never breaks when your schema evolves.

- このツールは、E/R　ダイアグラムや**マトリックスの**形で依存関係を表すことができる。例えば、**CRUDマトリックスでは**、データを操作する関数とデータを相互参照し、各プロシージャで実行される操作のタイプ（Create/Read/Update/Delete）を指定します。



- Visual Expert は、手作業で何時間もかかる作業を数秒で完了します。そのため、チームは、各変更の準備（所要時間の見積もり、変更するオブジェクトのリスト）を行うだけでなく、見落としのリスクを排除することができます。

ステップ5：変更を加える

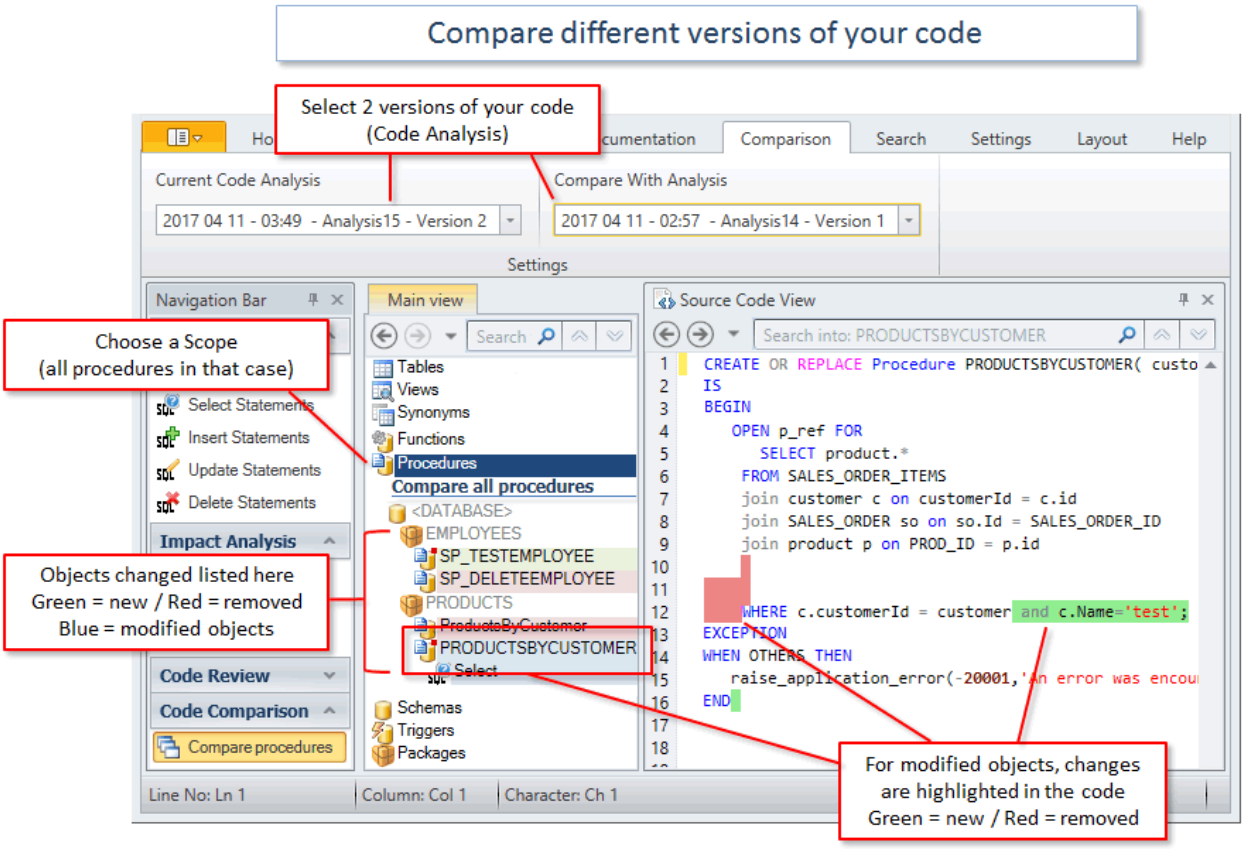
目的：変更を一貫して安全に適用する。

分析フェーズの後には、**変更**フェーズ（コードのリファクタリング、スキーマの修正）がやってくる。

Visual Expertは解析ツールであり、編集ツールではありません。そのため、修正は開発者が通常使用するコード・エディターで行うことになる。とはいえ、Visual Expertは実装時に役立つだろう：

- チームはこれを、**変更を追跡するためのダッシュボードとして**使うことができる。変更を行った後、新しいコードを分析し、古い要素が残っていないことを確認することができる。この**品質チェックは**、実装中に定期的に行うことができる。

- Visual Expert は、初期スキーマと移行スキーマの間で使用できる**コード/スキーマ比較**機能も提供している。このツールはコードとオブジェクトの構造を理解し、削除または追加されたオブジェクト、プロシージャコードの変更、カラムタイプの変更など、**構造化された差分の**レポートを生成します。この**チェック**リストにより、計画された各ポイントが確実に処理されていることを確認し、意図しない逸脱を検出することができる。



ステップ 6: 移行したデータベースの技術文書

目的：完全で、最新かつ使用可能なドキュメントを自動的に生成する。

移行が完了したら、最新の**技術文書を**作成する必要があります。これは、メンテナンスのための備品目録としても、変更内容の記録としても役立ちます。

しかし、回路図や何千行ものコードを手作業で文書化するのは現実的ではないし、プロジェクトの終わりには時間がないことが多い。時間がかかり、採算が合わず、ミスも起こりやすい。そのため、ドキュメンテーションを自動化することが不可欠です。

デフォルトでは、ドキュメンテーションには

1. データベースの構造（回路図、ダイアグラム）、
2. データ辞書（列、型）、
3. PL/SQLコード（モジュール、パラメータ、呼び出しのリストと説明）、
4. また、移籍前と移籍後の比較もできるかもしれない。

Visual　Expertは、コード・ドキュメントを自動的に生成します。解析リポジトリから、アプリケーションの詳細を記述した**HTMLページを**生成することができる。チームの新人は、コードがどのように動作するかを推測する必要はない。

- 各パッケージ/プロシージャー/トリガーについて、フォーマットされたソースコード、パラメータのリスト、説明、そして**リファレンス**（「このプロシージャーは、このようなテーブルを呼び出し、このような他のオブジェクトから呼び出される...」など）が指定されます。これらの参照は、ドキュメントの中でクリック可能なハイパーリンクとなっており、ウェブサイトのように**ナビゲートできるようになっています**。

- 例えば、移行中に変更されたすべてのオブジェクトのリストとその新しいステータス、コンポーネント間の相互作用を示す**CRUD**マトリックスや**コールダイアグラムなど**です。

Visual Expertは、マイグレーション後の**データモデル・ドキュメンテーションにも**役立ちます**。**

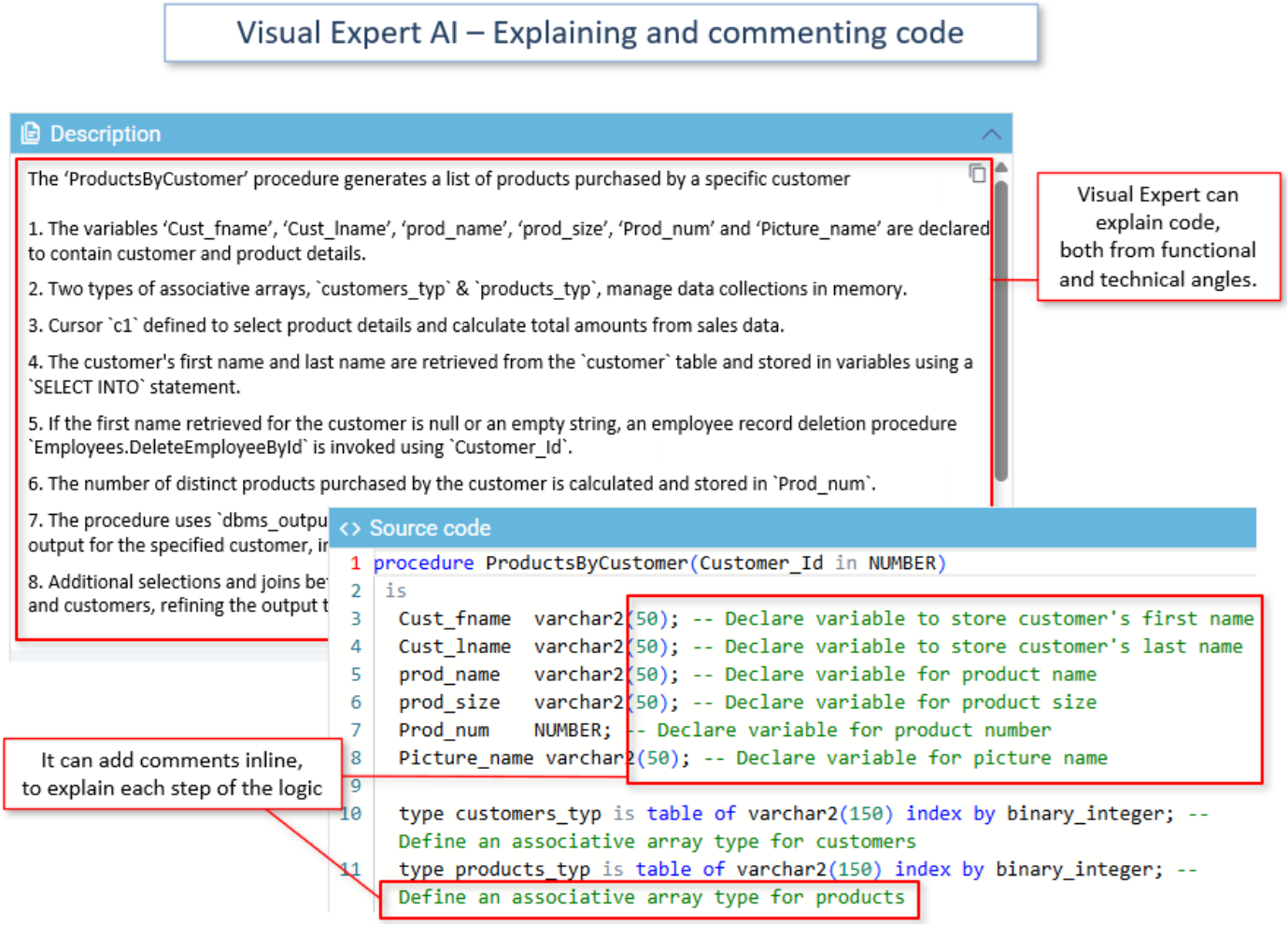
- 移行が完了すると、彼は**エンティティ関係図を更新**し、システム全体の**最新の視覚的図を**作成する。

- このダイアグラムは調整可能である。たとえば、**テーマ別のサブダイアグラムに**分割して、参照しやすくすることもできる。各サブダイヤグラムは、機能領域（請求、顧客管理など）をカバーすることができる。

- ダイアグラムが満足のいくものであれば、Visual　Expertは、最終的なドキュメントに統合するために、ダイアグラムを**PDFまたは高解像度の画像としてエクスポート**することができます。

技術文書の域を超えるために、ビジュアル・また、次のようなことができます：

1. コードの**ビジネス上の目的を説明する**
2. その**内部論理を**説明せよ。
3. コードに**コメントを**生成する



データ・モデルの概要、詳細なコード・ドキュメント、機能的な説明を組み合わせることで、Visual Expert はデータベースのあらゆる側面をカバーします。これらのツールのおかげで労力は最小限に抑えられ、チームに過度の負担をかけることなく、**プロジェクト全体の質が向上します**。

ステップ7：移行後の最適化

目的：移行したデータベースのパフォーマンスと保守性を向上させる。

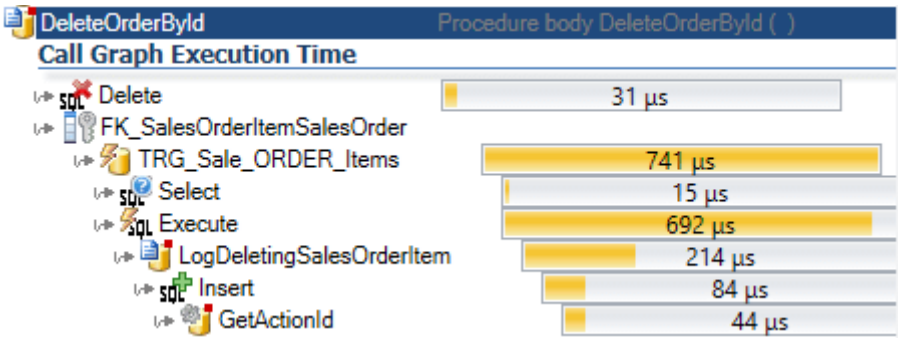
移行を成功させるには、バージョンのアップグレードにとどまりません。データベースも最適に機能しなければならない。しかし、上位バージョンに移行することで、**思わぬ問題が見つかる**こともあります。移行はまた、以前のバージョンでは対処できなかったパフォーマンス上の問題（インデックスの欠落、クエリの遅さなど）に対処する機会でもあります。

これらの移行後の最適化は、時間がないために見過ごされることがあります。数百あるクエリやプロシージャのうち、どれを最初に最適化すべきか？

Visual Expertには、Oracleによって生成された実行統計情報を収集して分析する**パフォーマンス最適化**モジュールが統合されています。これらの移行後の問題を修正することで、データベースのパフォーマンスと保守性が向上します。

- 平均実行時間が最も長いプロシージャーや、最も頻繁に実行されるプロシージャーを特定することができる。また、プロシージャ内の各SQLで消費される時間を示すこともできます。この情報は、例えば、最もコストのかかる10個のクエリをターゲットにするために使用することができます。

- 関数を呼び出す文字列を含む処理を分解し、**呼び出しと実行時間のグラフを生成します。**これにより、どの関数が処理を遅くしているかを視覚的に特定することができる：

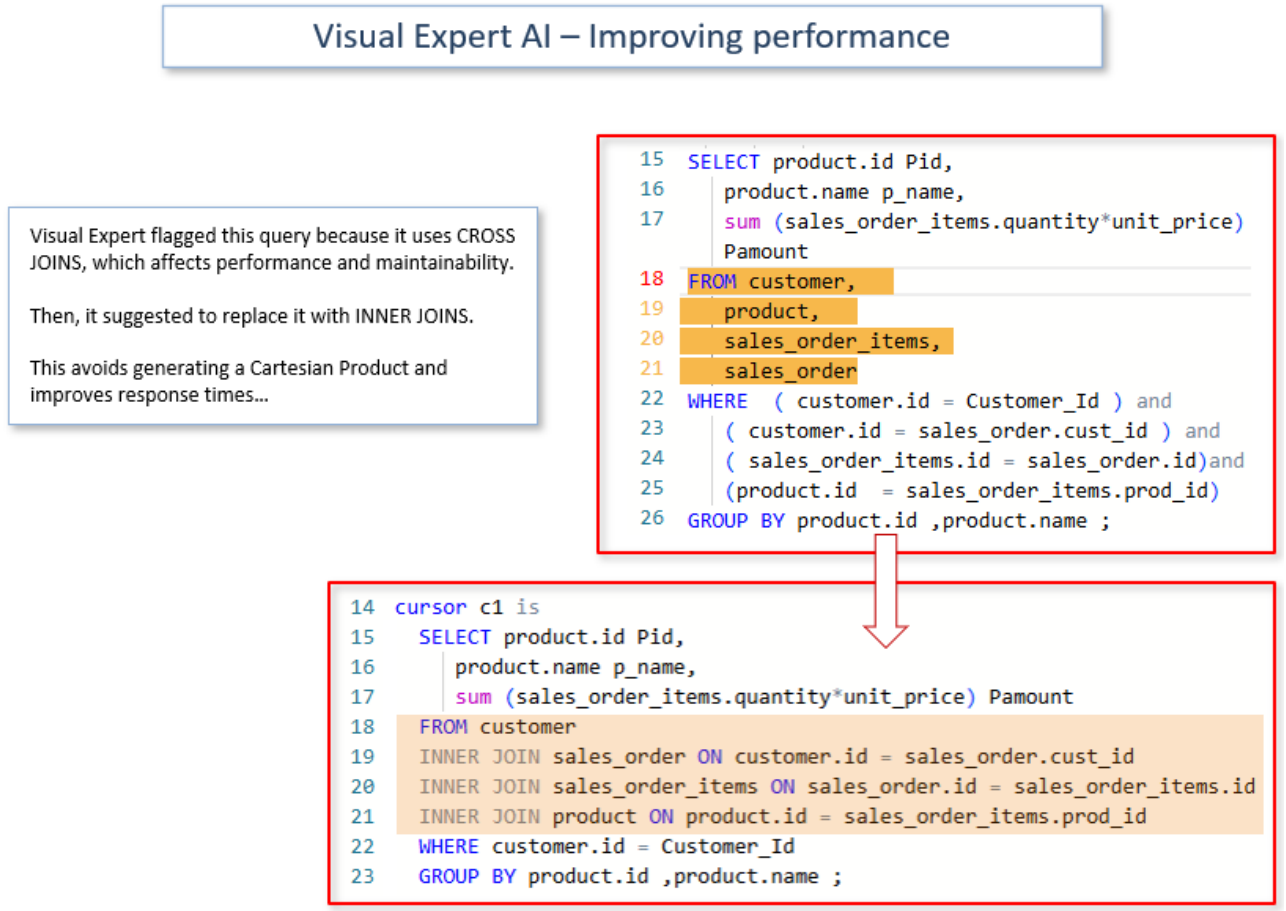


- Visual　Expertはまた、コードの品質をチェックし、**パフォーマンスに影響する**特定の**バッドプラクティス**（SQL結合の代わりにPL/SQLでネストされたループを使用するなど）を検出する。

- SQLクエリの実行速度を不必要に低下させる、欠落インデックス（インデックスを持たないが、SQLクエリのWhere/Group　by/Order　by/Having句で使用されるデータベース・カラム）を自動的に検出する。

- 最後に、Visual Expert を使用して、検出した遅いクエリを分析することができます。DBA は、クエリの実行プランを表示して、オラクルによってどのように処理されるか（インデックス・トラバーサル、完全スキャン、ハッシュ結合、ソートなど）を確認することができます。このようにして、分析からSQL最適化のためのアクションに素早く移行することができます。

- オブジェクトやクエリがアプリケーションの速度を低下させる場合、Visual　Expertはそのコードを最適化することができます：



このような機能の組み合わせは、全体的な予防的アプローチの一環として、あるいは特定のボトルネックの解消のために、アプリケーションのパフォーマンスを向上させる。

また、Visual　Expertを使用して移行前後のパフォーマンスを測定し、行った最適化の実際の影響を評価することもできます。

ステップ8：移行後の継続的モニタリング

目的長期にわたってPL/SQLコードの品質とパフォーマンスを維持する。

移行が完了したら、技術的な利点を維持することが不可欠です。Visual Expert は、Oracle スキーマと PL/SQL コードの長期的な進化の確保、文書化、および管理を支援します：

- **変更前の体系的な影響分析**

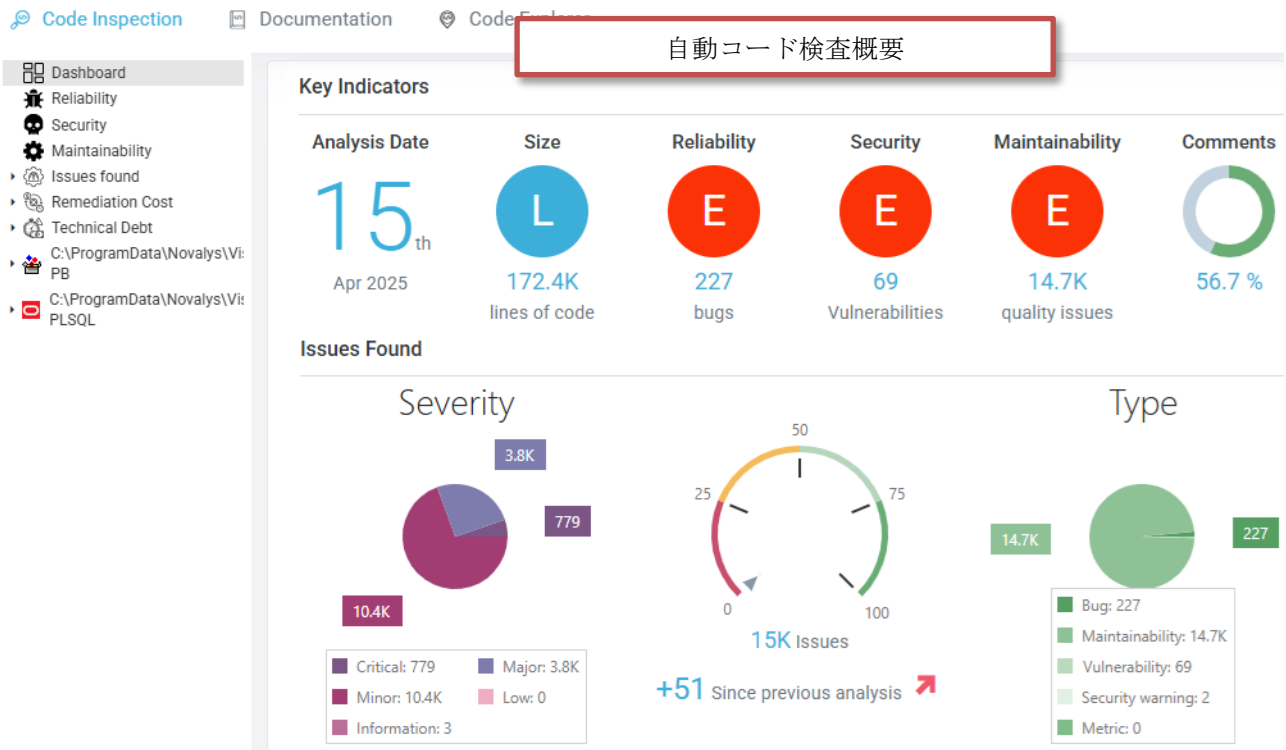
Oracle データベースと PL/SQL コードの重要な変更を行う前に、影響度分析を体系的に実 施することを推奨します。この実施により、依存関係を包括的に考慮し、リグレッション・リスクを最小限に抑えることができます。

- **定期的な性能チェック**

プロシージャやSQLクエリの実行時間を監視することで、新たな問題を検出し、迅速に対処することができます。例えば、パフォーマンスの低下を避けるために新しいインデックスの作成が必要となるようなクエリの修正後に、このような問題が発生する可能性があります。Visual Expert は、このようなケースを検出して DBA に報告し、DBA は必要な対策を講じることができます。

- **コードの品質とセキュリティの定期的な検証**

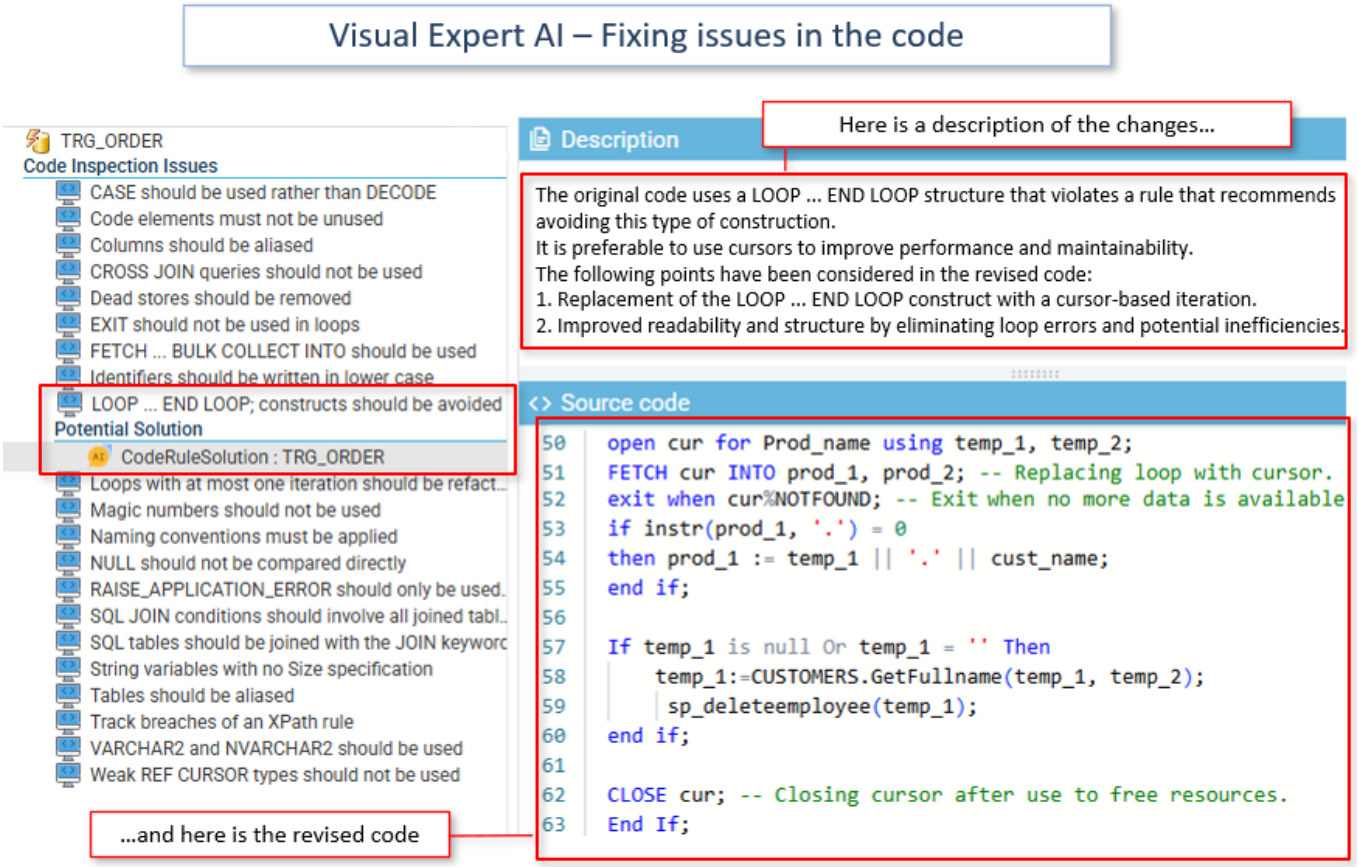
Visual Expertは、自動的かつ定期的にコードを検査し、品質やセキュリティの問題の兆候を検出することができます。例えば、チームが習慣的に非推奨のOracleデータ型や関数を使用している可能性があります。このように定期的に管理することで、長期にわたって高いレベルの品質とセキュリティを維持す



ることができます。

- **コードに見つかった問題の修正**

コード内で発見された各問題に対して、Visual Expertは潜在的な解決策を提案することができます。



- **スキーマとコード変更の文書化されたトラッキング**

  Visual Expertは、スキーマとPL/SQLコードに加えられたすべての変更を追跡し、バージョン管理と変更のトレーサビリティをより簡単にします。

このような継続的なモニタリングにより、メンテナンスコストを削減し、将来の進化を確保することで、チームは移行後の機能改善に集中することができます。

結論

Oracle 11g データベースを 19c/21c に移行するのは複雑なプロジェクトですが、適切なツールを使用すれば大幅に簡素化できます。Visual Expert を使用すると、開発チームと管理チームは、コード解析、スキーマ管理、モデリング、および文書化をカバーする**完全なツールボックスを利用**できます。

スコープの正確な特定、クリーンアップ、新機能への適応、制御された修正、技術文書化、継続的な最適化など、移行プロセスの各段階が容易になります。

最初の移行後、Visual Expert は、定期的な影響分析、パフォーマンス・チェック、および定期的なコード品質管理による実質的なサポートを提供し続けます。これらの機能は、オラクル・データベースの長期的な安定性、品質、およびパフォーマンスを保証し、移行プロジェクトの持続可能な投資収益率を保証します。

Visual Expertとコード編集ツールを組み合わせ、プロジェクト・ステップを厳密に管理することで、技術チームは、リスクを低減し、開発を保護し、Oracleデータベースの将来の保守性を促進しながら、意欲的な移行を成功させることができます。

リソース：

- Visual Expert for Oracle - 製品ページ
- Visual Expert for Oracle - オンライン・ドキュメント

ステージ別概要表

プロジェクト段階	ビジュアル・エキスパートの特徴
1.ボリュームの評価	- メトリクス（コード行数、オブジェクト数）の計算 - オブジェクト・タイプ別のインベントリ - アプリケーション資産の概要 - ER図（データモデル）の自動生成 - 複雑な領域や孤立した領域の視覚的分析
2.クリーニングコード	- 未使用オブジェクトの特定（参照されない） - コードによって一度もアクセスされていないテーブルの特定 - 重複したコード、空のコード、冗長なコードの検出
3.時代遅れの要素の検出	- 廃止された関数/型の使用をチェック - コード解析ルールによる警告 - 最新化すべきオブジェクトの自動識別 - ERモデルを分析し、設計の古さを特定する。 - 進化を定義するためのモデル注釈
4.影響分析	- インタラクティブ・インパクト分析（誰がエレメントを使うのか？） - CRUDマトリックス（誰がどのテーブルから読み込み/どのテーブルに書き込むか？） - 呼び出しと相互参照ダイアグラム
5.変更の実施	- 変更後の検証（すべてのポイントをカバー） - 移行前後の回路図/コード比較
6.技術文書	- コードドキュメントの自動生成 - CRUDマトリックス生成 - コールダイアグラムの生成 - 移行後のドキュメンテーションのためのER図の更新 - データモデルサブダイアグラムのエクスポートと作成
7.マイグレーション後の最適化	- コードパフォーマンス分析（平均時間、頻度） - 関数呼び出し文字列の解析 - 欠落インデックスの検出 - SQLクエリの分析とチューニング（実行プラン） - 最適化を反映してモデルを更新
8 - 移行後の継続的モニタリング	- 変更前の体系的な影響分析 - 定期的な性能チェック - 定期的な品質チェック - スキーマとコードの進化の文書化

付録 A - オラクルの非推奨およびサポート対象外の機能 (18c ～ 23c)

オラクル18c

非推奨の機能

- **拡張データ型サポート (EDS)** は DBMS_LOGSTDBY パッケージから廃止されました。EDS がサポートするすべての Oracle データ型は、論理スタンバイまたは Oracle GoldenGate でネイティブにサポートされるようになりました。
- DBMS_LOCK.SLEEP プロシージャは非推奨です。
  を**使用して**ください：代わりに DBMS_SESSION.SLEEP を使用してください。
- *GET_MODEL_**DETAILS**関数は非推奨。
  - 非推奨サブプログラム
    - GET_ASSOCIATION_RULES 関数
    - GET_FREQUENT_ITEMSETS 関数
    - GET_MODEL_DETAILS_AI 関数
    - GET_MODEL_DETAILS_EM 関数
    - GET_MODEL_DETAILS_EM_COMP 関数
    - GET_MODEL_DETAILS_EM_PROJ 関数
    - GET_MODEL_DETAILS_GLM 関数
    - GET_MODEL_DETAILS_GLOBAL 関数
    - GET_MODEL_DETAILS_KM 関数
    - GET_MODEL_DETAILS_NB 関数
    - GET_MODEL_DETAILS_NMF 関数
    - GET_MODEL_DETAILS_OC 関数
    - GET_MODEL_SETTINGS 関数
    - GET_MODEL_SIGNATURE 関数
    - GET_MODEL_DETAILS_SVD 関数
    - GET_MODEL_DETAILS_SVM 関数
    - GET_MODEL_TRANSFORMATIONS 関数
    - GET_MODEL_DETAILS_XML 関数
    - GET_TRANSFORM_LIST 手続き
  - 非推奨データ型
    - DM_CENTROID
    - DM_CENTROIDS
    - DM_CHILD
    - DM_CHILDREN
    - DM_CLUSTER
    - DM_CLUSTERS
    - DM_CONDITIONAL
    - dm_conditionals
    - dm_cost_element
    - DM_COST_MATRIX
    - dm_em_component
    - dm_em_component_set
    - dm_em_プロジェクション
    - dm_em_projection_set
    - DM_GLM_COEFF
    - dm_glm_coeff_set
    - dm_histogram_bin
    - DM_HISTOGRAMS
    - DM_ITEM
    - DM_ITEMS
    - DM_ITEMSET
    - DM_ITEMSETS

        - dm_model_global_detail
        - dm_model_global_details
        - DM_NB_DETAIL
        - DM_NB_DETAILS
        - dm_nmf_属性
        - dm_nmf_attribute_set
        - DM_NMF_FEATURE
        - dm_nmf_feature_set
        - DM_PREDICATE
        - DM_PREDICATES
        - dm_ranked_attribute
        - dm_ranked_attributes
        - DM_RULE
        - DM_RULES
        - DM_SVD_MATRIX
        - dm_svd_matrix_set
        - dm_svm_attribute
        - dm_svm_attribute_set
        - dm_svm_linear_coeff
        - dm_svm_linear_coeff_set
        - DM_TRANSFORM
        - DM_TRANSFORMS

使用してください：代わりにモデル詳細ビューを使用します。

- DBMS_XMLQUERY パッケージは非推奨**です**。
  を**使用して**ください：DBMS_XMLGEN
- DBMS_XMLSAVE パッケージは非推奨です。**VEPLSQLRULE197 (完了)**
  使用します：DBMS_XMLSTORE
- Oracle Multimedia パッケージおよび型は非推奨です**：VEPLSQLRULE201 (動作中)**
    - ORD_AUDIO
    - ORD_DOC
    - ORD_IMAGE
    - ORD_VIDEO
    - オラクルマルチメディア ORDAudio
    - オラクルマルチメディアORDDoc
    - オラクルマルチメディアORDImage
    - オラクル・マルチメディア ORDVideo

使用する：Oracle SecureFiles とサードパーティ製ツール。

非対応機能

- 以下のサブプログラムは DBMS_XDB パッケージからサポートされていません: (Working)
  VEPLSQLRULE202
    - 加算時効マッピング
    - ADDMIMEMAPPING
    - アドケマロックマッピング
    - アドサーブレットマッピング
    - アドサーブ
    - addxmlextension
    - CFG_GET
    - CFG_REFRESH
    - CFG_UPDATE
    - 削除httpexpiremapping
    - 削除
    - デレスケマロクマッピング
    - DELETESERVLET
    - 削除サーブレットマッピング

    - サーブレットの削除
    - deletexmlextension
    - GETFTPPORT
    - GETHTTPPORT
    - ゲットリストエンドポイント
    - SETFTPPORT
    - SETHTTPPORT
    - セットリスト・エンドポイント
    - setlistenerlocalaccess

を使用します：dbms_xdb_config

- UTL_FILE_DIR 初期化パラメータは DBMS_LOGMNR_D パッケージでサポートされていません：**使用してください**：ディレクトリ・オブジェクト
- DBMS_XDB パッケージからサポートされない定数：**VEPLSQLRULE202 (完了)**
    - xdb_endpoint_http
    - xdb_endpoint_http2
    - xdb_protocol_tcp
    - xdb_protocol_tcps

を使用します：dbms_xdb_config

- DBMS_XMLSCHEMA パッケージのサブプログラム GENERATESCHEMA および GENERATESCHEMAS : **VEPLSQLRULE203 (完了)**
  注意: 代替なし
- DBMS_XMLTRANSLATIONS パッケージはサポートされません：**VEPLSQLRULE204 (完了)**
  注意: 代替なし

オラクル 19c

非推奨の機能

- DBMS_SESSION.IS_ROLE_ENABLED 関数は非推奨です：**VEPLSQLRULE204**
  を使用してください：DBMS_SESSION.CURRENT_IS_ROLE_ENABLED または SESSION_IS_ROLE_ENABLED を使用してください。
- SERVICE_NAMES パラメータは非推奨です。
  使用: SRVCTL または GDSCTL コマンドラインユーティリティまたは DBMS_SERVICE パッケージ
- RECOVER...SNAPSHOT TIME メソッドは、特定のスナップショットを使用してデータベースをある時点までリカバリするものですが、非推奨となります。
  を使用してください：ALTER DATABASE BEGIN/END BACKUPおよびRECOVER ...時間
  推奨：RMANを使用する。

非対応機能

- PDBのフラットファイル辞書ダンプ
  を使用する：DBMS_LOGMNR.START_LOGMNR（SCN/時刻付き
- DBMS_LOGMNR.START_LOGMNRパッケージのCONTINUOUS_MINEオプション
  注：代替はありません
- Oracle Streams DBMS_STREAMS_ADM パッケージはサポート対象外です：
  VEPLSQLRULE205
  使用するオラクル・ゴールデンゲート
- オラクルマルチメディア**VEPLSQLRULE201**
  使用するOracle SecureFiles + オープンソースまたはサードパーティ製ツール
- LOG_ARCHIVE_DEST_nのMAX_CONNECTIONS属性のサポート終了
  注：ストリーミング・メカニズムをサポートするために削除された
- 拡張データ型サポート（EDS）
  注：現在はネイティブ・サポート
- O7_DICTIONARY_ACCESSIBILITY 初期化パラメータ。
  注：セキュリティ強化のため削除

オラクル21c

非推奨の機能

- オラクル・ウォレット・マネージャー（OWM）
  使用: orapki、mkstore
- SQLNET と DBMS_CRYPTO における SHA-1 の使用 : **VEPLSQLRULE206**
- MD5 と MD4 アルゴリズム : **VEPLSQLRULE206**
- Oracle Enterprise Manager Database Express (EM Express)

非対応機能

- マルチテナント専用アーキテクチャ - CDBのみサポート
- dbms_xmlsave : **veplsqlrule197**
- dbms_xmlquery : **veplsqlrule198**
- DBMS_CRYPTO - アルゴリズム MD4、MD5、RC4 が削除されました : **VEPLSQLRULE206**
- XML DB 関連の変更：
  - パッケージDBMS_XDBT
  - 手続きdbms_xslprocessor.clob2file, dbms_xslprocessor.read2clob
    を使用します：dbms_lob.clob2file, dbms_lob.loadclobfromfile
- ラージ・オブジェクト (LOB) 機能 DBMS_LOB.LOADFROMFILE および LOB バッファリング (BUFFERING_ENABLED) はサポートされません：**VEPLSQLRULE207**
  を使用してください：DBMS_LOB.LoadClobFromFile, DBMS_LOB.LoadBlobFromFile
- DBMS_OBFUSCATION_TOOLKIT - 削除されました：**VEPLSQLRULE208**
  を使用します：DBMS_CRYPTO
- PDBのフラットファイル辞書ダンプ
  を使用します：dbms_logmnr.start_logmnr
- DBMS_APPLY_ADM.SET_PARAMETERプロシージャのOPTIMIZE_PROGRESS_TABLEパラメータ

オラクル **23ai**

非推奨の機能

- mkstoreツール
  使用： orapki
- DBMS_RESULT_CACHE 関数の名前の変更：**VEPLSQLRULE209**
  - BLACK_LIST関数を使用する。BLOCK_LIST関数を使う。
  - BLACK_LIST_ADDプロシージャを使用する。BLOCK_LIST_ADD プロシージャを使用する。
  - BLACK_LIST_CLEAR プロシージャを使用します。BLOCK_LIST_CLEARプロシージャを使用する。
  - BLACK_LIST_REMOVE プロシージャを使用します。BLOCK_LIST_REMOVEプロシージャを使用する。
  - OBJECT_BLACK_LIST 関数を使用します。OBJECT_BLOCK_LIST関数を使う
  - OBJECT_BLACK_LIST_ADD プロシージャを使用する。
    OBJECT_BLOCK_LIST_ADD プロシージャを使用する。
  - OBJECT_BLACK_LIST_CLEAR プロシージャを使用します。OBJECT BLOCK LIST_CLEAR プロシージャを使用する。
  - OBJECT_BLACK_LIST_REMOVE プロシージャを使用します。
    OBJECT_BLOCK_LIST_REMOVE プロシージャを使用する。

- dbms_xmlstore **veplsqlrule210**
  用途: SQL DML と標準的な XQuery および SQL/XML を使用した XML データの保存と管理
- dbms_xmlgen **veplsqlrule211**
  使用: SQL/XML 演算子
- XML DB リポジトリおよび関連するすべてのインターフェイス（oracle.xdb.servlet、oracle.xdb.event、oracle.xdb.spi など）
- dbms_hang_manager : **veplsqlrule212**
  を使用する：dbms_blocker_resolver
- 従来の監査パッケージ/関数**VEPLSQLRULE213**
  - init_cleanup、deinit_cleanup、is_cleanup_initialized
- Oracle OLAP (OLAP DML、Java API、分析ワークスペース)
  使用するOracle Analytic ViewsまたはOracle Essbase

非対応機能

- Oracle Enterprise Manager Database Express（EM Express）
  使用するOCI DB Management、Cloud Control、またはSQL Developer
- サービス属性の値 SESSION_STATE_CONSISTENCY = STATIC with FAILOVER_TYPE = TRANSACTION
  を使用する：
  - failover_type = auto, session_state_consistency = auto
  - failover_type = transaction, session_state_consistency = dynamic
- オラクル・ウォレット・マネージャー（OWM）
  使用法： orapkiコマンドラインツール