



オンプレ DB をクラウド上の PostgreSQL へ ～ DB移行時のSQLアセスメントを自動化・省力化するには？～



State-of-the-Art Solution for Database Migration
DB・アプリ移行を促進するキーソリューション

株式会社インサイトテクノロジー
趙 明春（チョウ メイシュン）

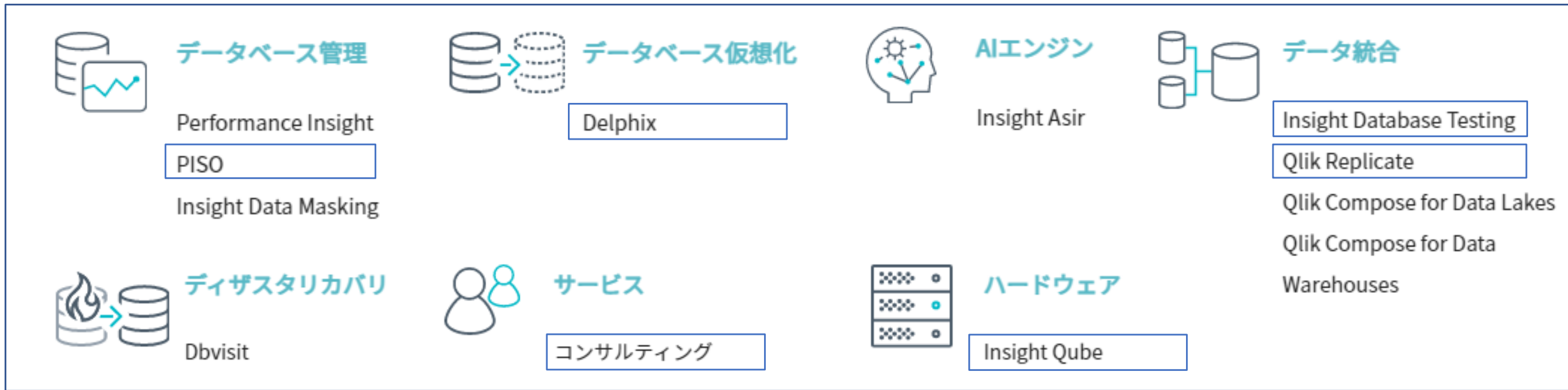
- ・ 弊社事業・製品紹介
- ・ PostgreSQLへの取り組み
- ・ DB監査とPISO製品の概要
- ・ Insight Database Testing(IDT)の紹介
- ・ AWS Schema Conversion Tool(SCT)の紹介
- ・ 「IDT + SCT」移行ソリューション
 - ・ オンプレDBからクラウドPostgreSQLへのキーソリューション
 - ・ 移行モチベーションと移行時課題
 - ・ 移行コスト評価プロセス
- ・ IDTの事例
 - ・ テスト網羅性向上、テスト工数削減

「データ × AI」 から得た Insight で新しく豊かな社会を創造し続けます

データ活用からお客様が求める
「本質」を得られる
「インサイト・テクノロジー」の実現へ



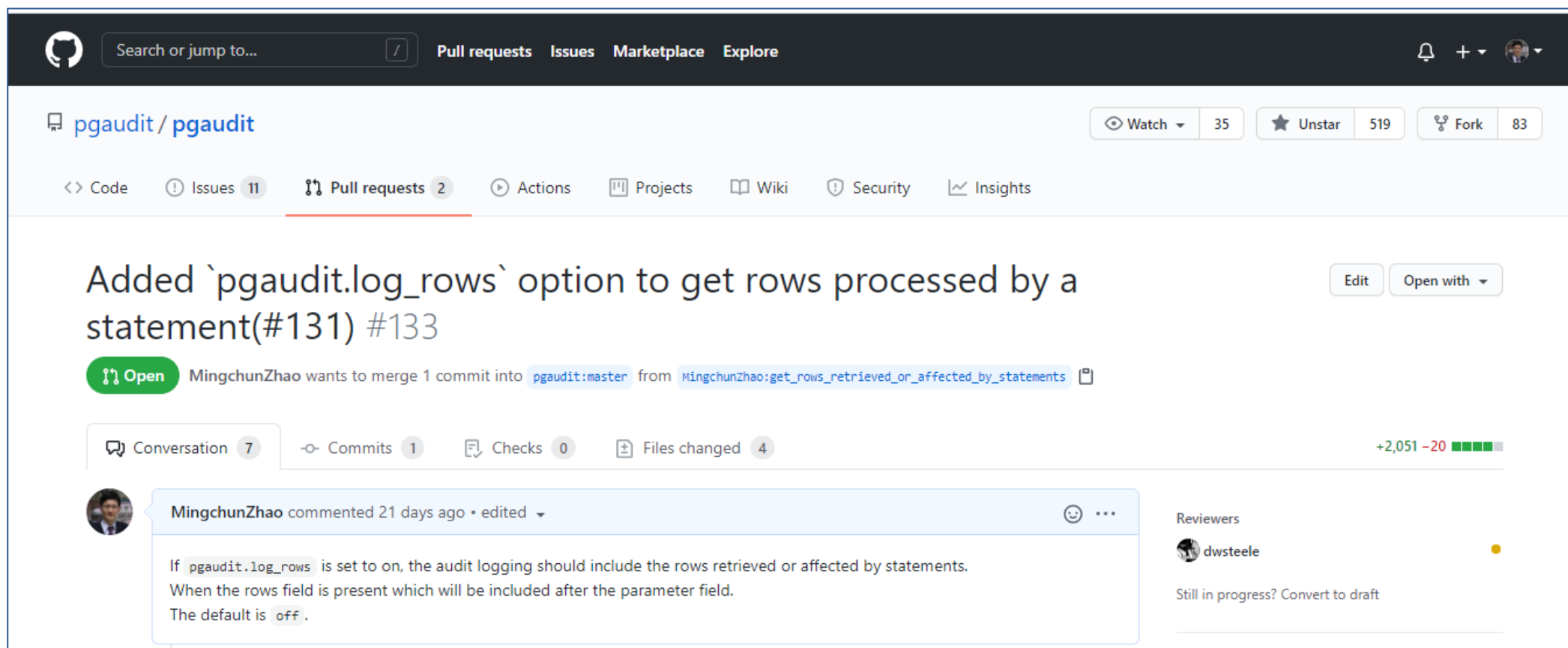
製品群とPostgreSQL向け製品



- PISO: データベース監査ツール
- Delphix: データベース仮想化ツール(Database as Code)
本番環境と同等のテスト用データを短時間で作成し、「テストサイクルの短縮」「効率的なバグ修正による品質向上」を支援
- Insight Database Testing: 本セッションの主役、DB・アプリ移行アセスメント・テスト自動化ツール
- Qlik Replicate: マルチデータベース対応のリアルタイムレプリケーションツール
データ活用基盤の再構築において、既存システムに負荷をかけずに迅速で安全なデータ統合／移行を実現
- Insight Qube: 世界最速のデータベースマシンをコンセプトにしたプラットフォーム
企業でのデータ活用にフォーカス、マルチデータベースによる運用性、コストパフォーマンス、拡張性の向上を実現

PostgreSQLへの取り組み

- 独自のPostgreSQL監査ロギング拡張機能を自社開発
- 自社製品でデータ蓄積・管理に用いるデータベースをPostgreSQLへ移行
- AWS PostgreSQLの監査ロギング拡張機能pgAuditのOSSコントリビュート



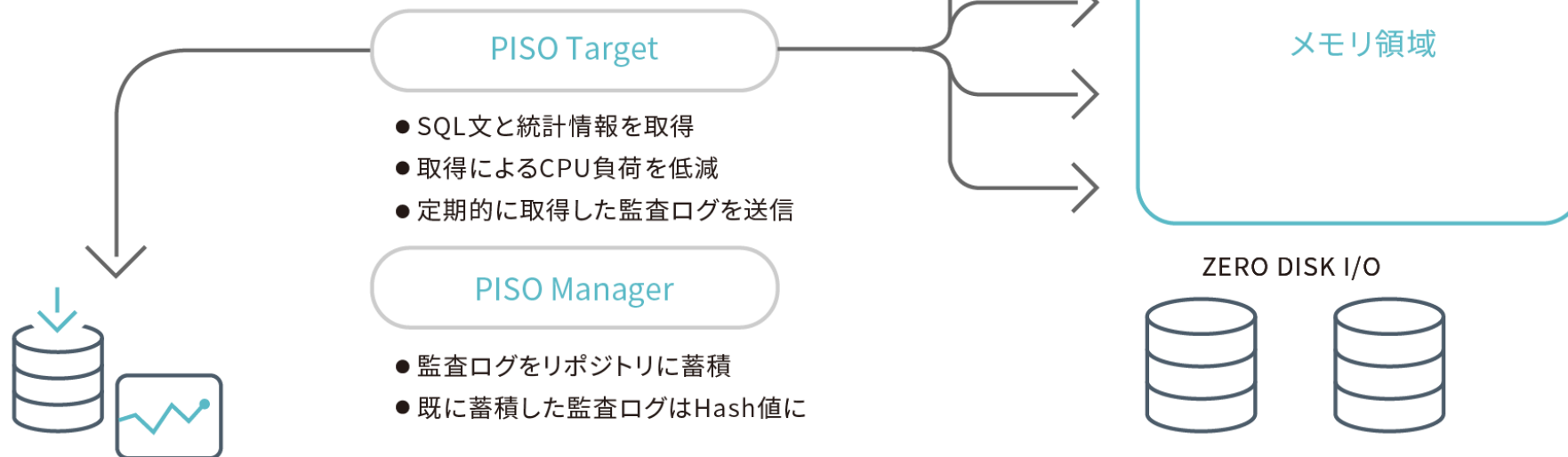
データベース監査ツール PISO

- メモリ参照型でDB性能を劣化させず、低負荷でアクセスログを取得
- 特権ユーザーも含めたあらゆるアクセス経路のログを取得し、保全、警告、検索
- マルチデータベース、クラウドに対応
- データベース監査市場にて10年以上連続シェアNo.1、国内導入企業数は700社以上

アクセスログの記録 -Direct Memory Access -

パフォーマンスを劣化させないための作り

- ✓ データベースに依存することなくメモリから直接情報を取得
- ✓ ログを監視対象データベース（ディスク）に書き込まない

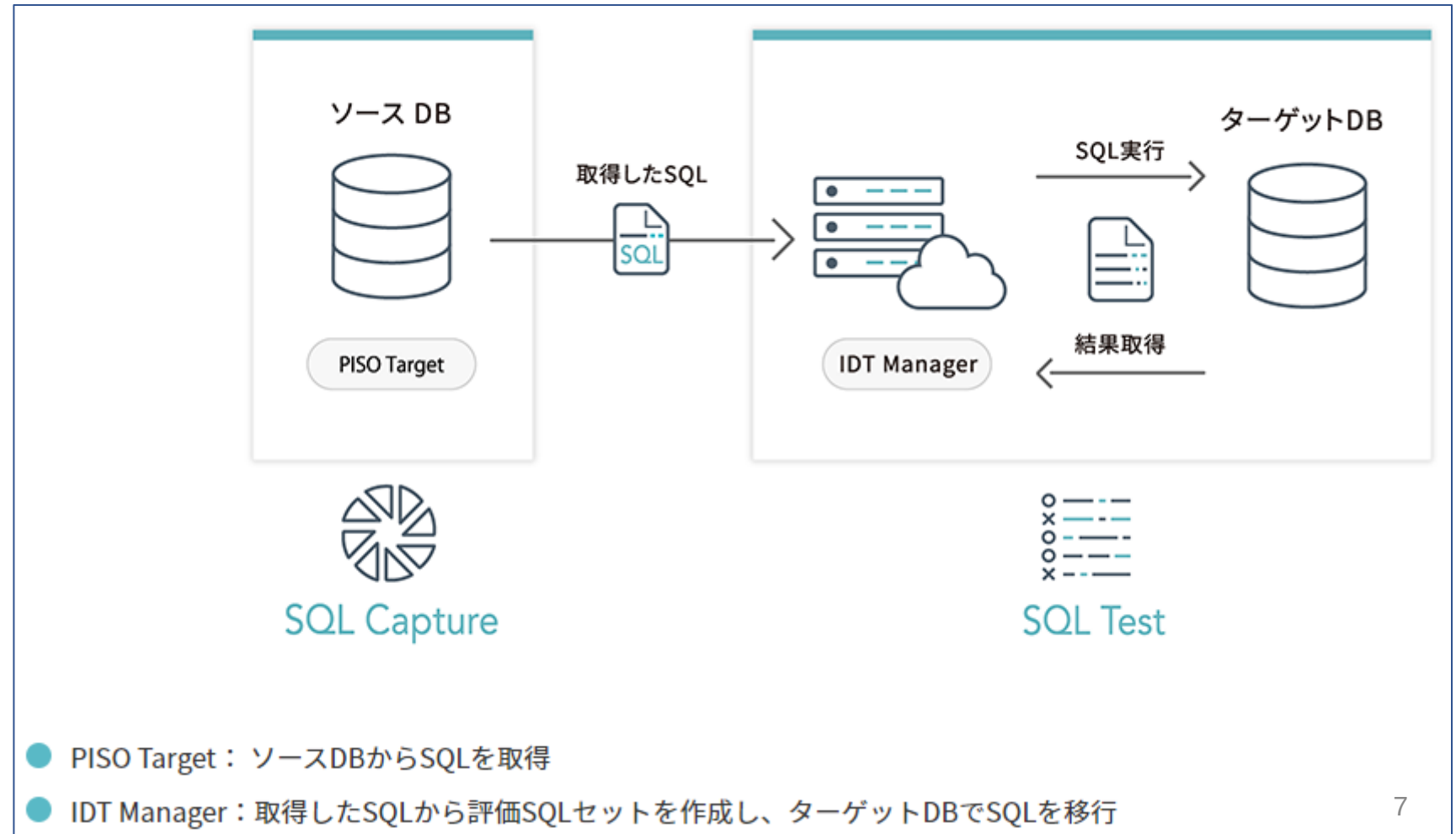


Insight Database Testing(IDT)とは

- データベース/アプリケーションの移行アセスメント・テスト自動化ツール
- マルチデータベースとクラウドサポート、移行の作業コスト削減を支援

IDTの機能

- SQL Capture
- SQL Test
- SQL 改修

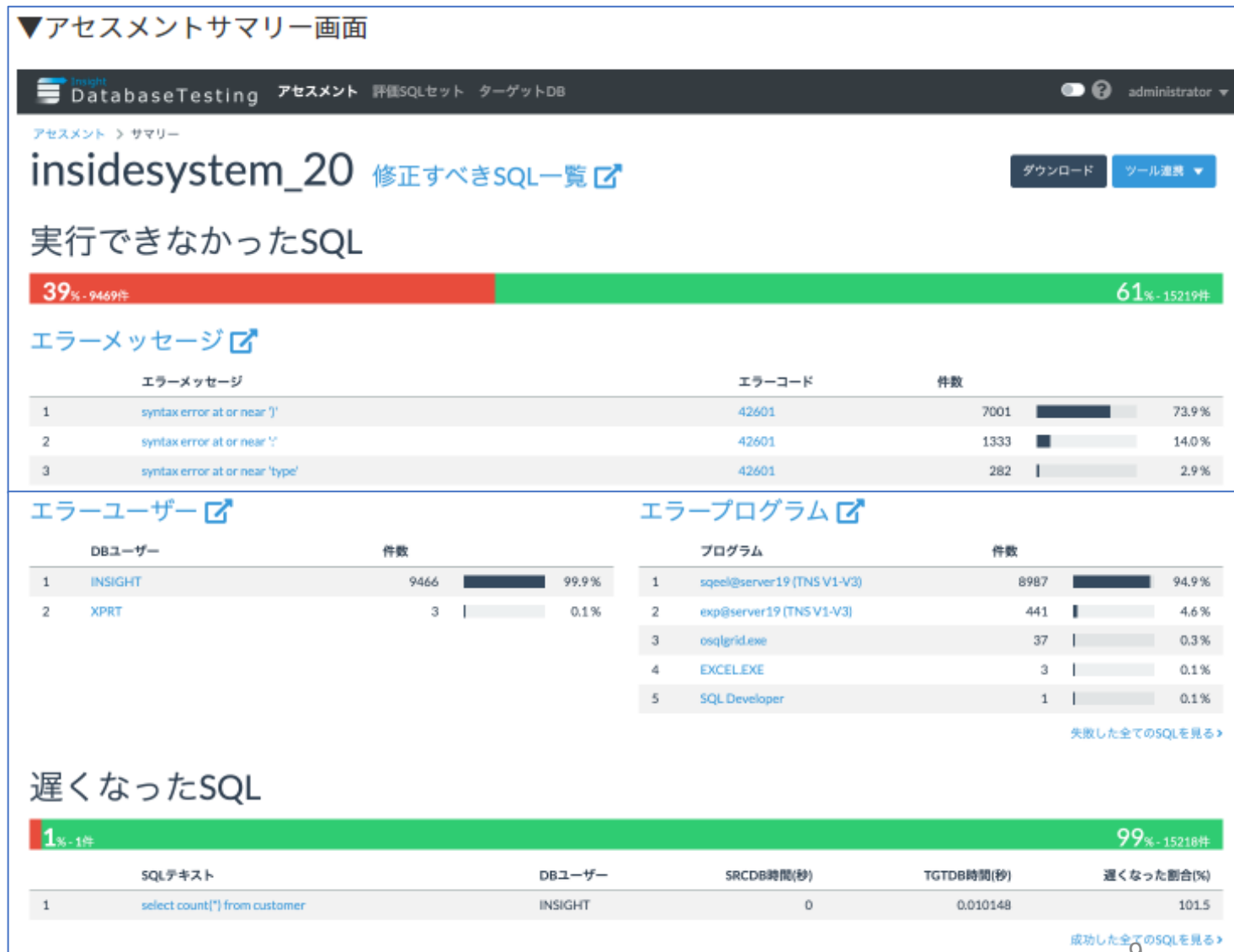


- 本番環境のSQLを自動的にキャプチャーし、重複するSQLをグループ化して要約表示
- 弊社独自のDirect Memory Access機能によりDB性能にほとんど影響を与えない

SQL Capture		
SQL	SQL Text	実行回数
	SQL開始／終了時間	処理件数
	オブジェクト	バインド変数
プログラム	プログラム名	プロセスID
ユーザー	OS／DBユーザー名	ログイン／アウト時間

SQL Test

- SQLを移行先の環境や
設定変更後の環境で
実行・評価
- SQLテストで実施
できるテスト項目
 - SQL互換性テスト
(実行 or パース)
 - パフォーマンス比較
(実行時間比較)
 - SQL修正結果の動作確認



- 実行エラーになったSQL、実行時間が遅くなったSQLを改修、テスト
- Test Reportで失敗SQLをドリルダウンし、GUI上でSQLを変更しそのまま実行しエラー修正

▼SQL詳細画面

DatabaseTesting アセスメント 評価SQLセット ターゲットDB administrator

アセスメント > サマリー > SQL一覧 > SQL詳細

PI Hash:2329725775

失敗
subquery in FROM must have an alias

SQLの修正 ソースSQL 詳細情報 同一のSQL

```
select
  active.sup asup,
  total.sup sup
from
  (
    select
      trunc(sur(sup), 1) sup
    from
      (
        select
          case
            sb.usendo
            when 5 then (
              1 + (trunc(sysdate - sh.CREATED_DATE) * 0.01) + (trunc(sysdate - sh.LAST_UPDATED) * (1 * 0.03))
            ) * decode(sh.STATUS_NO, 3, 0.7, 6, 0.5, 8, 0.8, 10, 0.9, 1)
            when 4 then (
              2 + (trunc(sysdate - sh.CREATED_DATE) * 0.01) + (trunc(sysdate - sh.LAST_UPDATED) * (2 * 0.03))
            )
          end
        from
          sh
        where
          sh.STATUS_NO = 3
      )
    )
```

整形する 設定 SQL実行 SQLの保存

ミッション: Oracle DatabaseをAWS上のPostgreSQLへ

- 背景
 - オンプレDBをRDSやAuroraなどのマネージドデータベースへ移行したいと考えるお客様が非常に多くなっている
- 移行のモチベーション
 - クラウドへの移行
 - オンプレDBサーバーの稼働を自社管理するコストを削減
 - クラウドのセキュリティの方が安全という評価
 - 性能が足りなくなってきたときに容易にスケールアップ
 - DBエンジン変更
 - Oracle Databaseのライセンスコストの増加に対する備え
 - Oracle DatabaseからPostgreSQLへの移行事例が増えている
- 「IDT + SCT」を融合した移行ソリューション
 - Insight Database Testing(IDT)
 - AWS Schema Conversion Tool (SCT)
 - AWS DBへの移行に伴うアプリケーション修正コストの評価を劇的に高速化

クラウド移行のネックは？

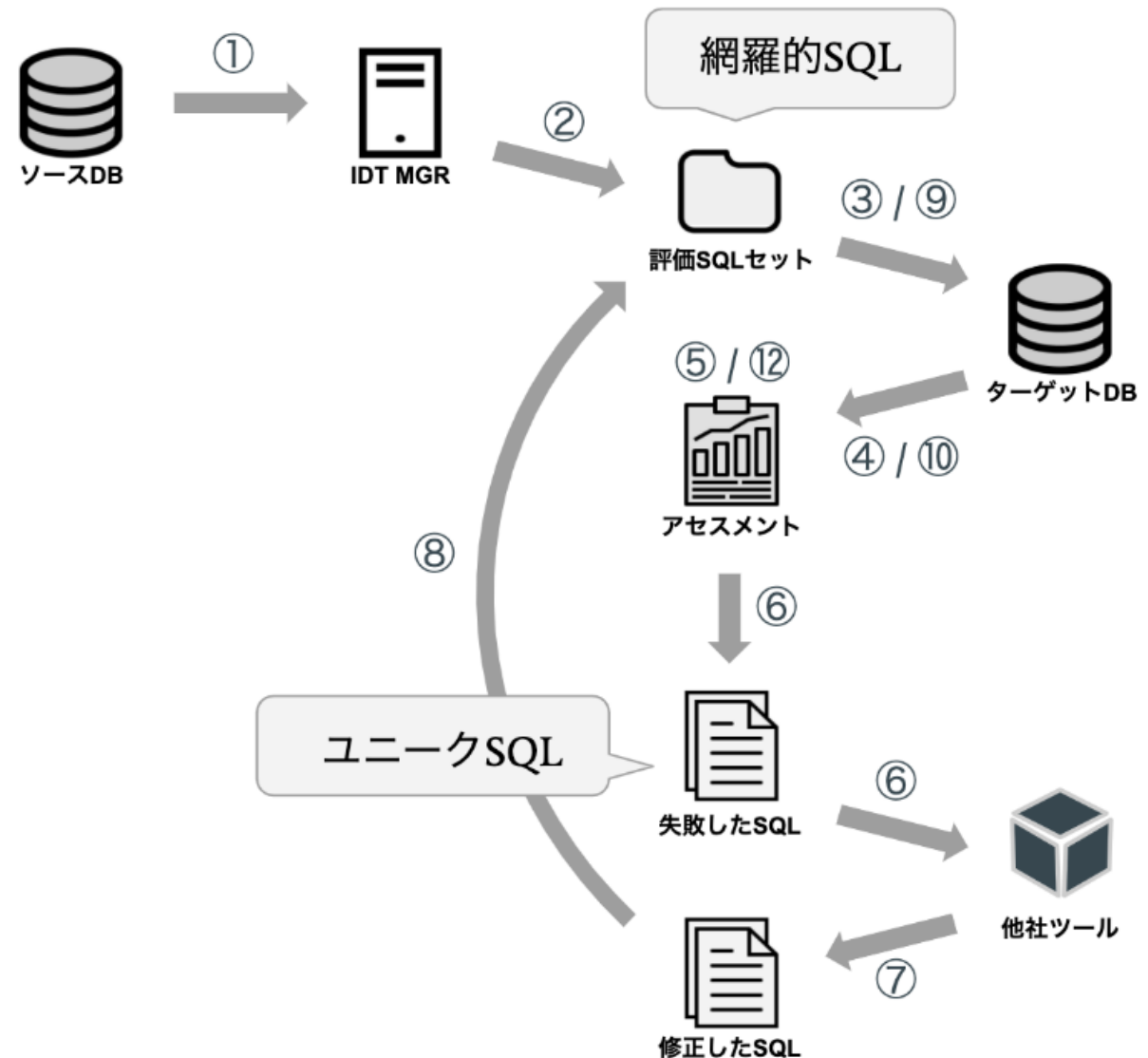
- データベースの移行とは
 - テーブルなどのスキーマ移行
 - テーブルに含まれるデータ移行
 - それだけで十分でしょうか？
- アプリケーションの移行がネック
 - データベースを利用するアプリケーション移行こそがクラウド移行の鍵
- アプリケーション移行の課題
 - DBにより使用できるSQLの方言が異なる
 - 特にOracle固有のSQLを多く使っているアプリケーションは修正コストが非常に大きい
 - アプリケーションで実行されるSQL全文そのものがソースコードに書かれていない
 - ORマッパー経由で実行されていたり、コードで文字列連結により組み立てられていたり
 - 複雑な場合、実際にどのようなSQL文となって実行されるか確認が難しい
 - アプリケーションが巨大すぎてどこからSQLが発行されているかわからない
 - SQLが把握できたとしても、そのSQLをPostgreSQLで実行できるかわからない

アプリケーション移行コストの見積もりの課題

- ・ 非現実的な見積もりとなるケース
 - ・ 実際に実行されているSQLがわからない
 - ・ そのSQLがPostgreSQLで実行可能かわからない
 - ・ アプリケーションコードに精通したエンジニアが工数をかけて調査が必要
- ・ 実際に発行されるSQLが特定できないと、
 - ・ リスクヘッジを重ねた非現実的な見積もりベースのコストとなり
 - ・ データベースのクラウド移行自体をあきらめた話も。。。

IDTを使ったアプリケーション移行評価の流れ

- ソースDBのSQLを取得
[SQL Capture機能]
- 評価SQLセットを作成
ターゲットDBで実行したいSQLセット
- ターゲットDBを選択
テーブルなどオブジェクトが
事前に用意されている前提
※IDTでは行えません
- アセスメントを実行
評価SQLセットをターゲットDBで実行
[SQL Test機能]
- 実行結果の確認
実行できたSQLの数、
実行エラーになったSQL、
遅くなったSQL
を一覧で表示
- SQLの詳細確認と実行
アセスメントサマリー画面から
ドリルダウンし、
SQLの詳細画面に遷移、
SQLの編集と実行、
および実行したSQLを保存
[SQL 改修機能]



AWSスキーマ変換ツール(Schema Conversion Tool、SCT)



- AWSより無償で提供され**AWSへのDB移行を強力に後押し**
- ソースDBのスキーマと大部分のカスタムコードを自動変換
 - ビュー、ストアドプロシージャ、関数などをターゲットDBと互換性のある形式に自動変換
- 異種データベースを簡単に移行できる
- IDTで実行されない部分にSCTを利用
 - AWSへのDB移行に伴う、アプリケーション移行影響調査を劇的に省力化

IDTとSCTを融合したAWSへの移行評価プロセス

- 手順1: 現行DBで実際実行されているSQLの収集
- 手順2: (SCT)現行DB同様のテーブルなどオブジェクトを事前に移行先DBに用意
- 手順3: 収集したSQLを移行先DBで評価
- 手順4: 評価結果の確認
- 手順5: (SCT)実行できなかったSQLをSCTで修正
- 手順6: 修正したSQLを再実行

その結果：

- 多大な工数をかけてソースコードを分析し得ていた実行SQL情報を簡単取得
アプリケーション修正移行コストの見積もりを大幅省力化
- SQLの収集～SQLの評価～修正候補の確認を効率よく行うことが可能
AWSへのアプリケーション移行評価のプロセスをさらに省力化
- 多くのアプリケーションを抱える企業でも、
各アプリケーションの移行難易度の評価が行いやすくなる

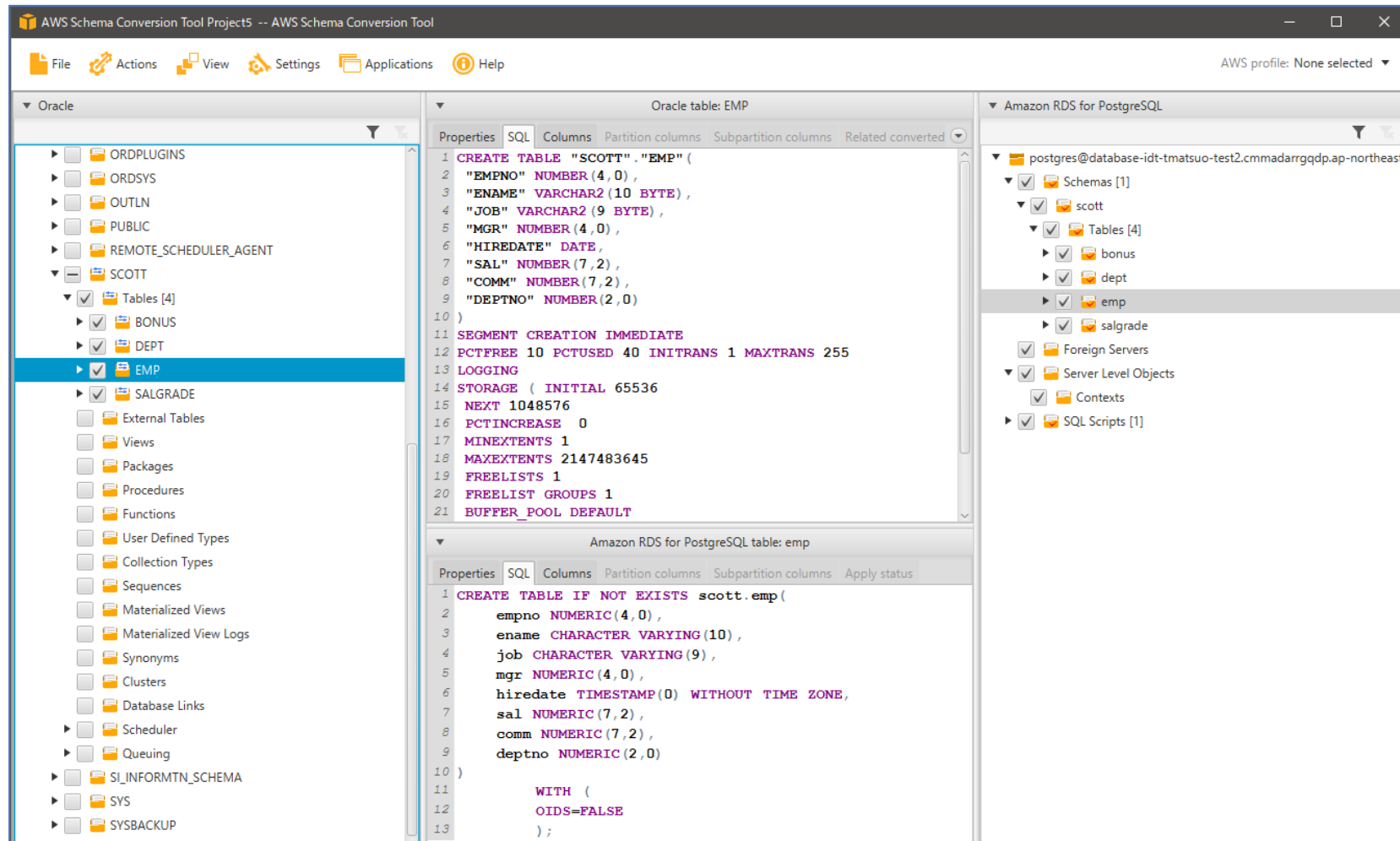
手順1: 現行DBで実際実行されているSQLの収集

- 実行されるSQLを把握困難な状況でも、移行にあたって考慮すべきSQLを効率よく収集可能



手順2: (SCT)現行DB同様のスキーマを事前に移行先DBに用意

- IDTで収集したSQLを実行する前に、実行先DBにテーブルを用意しておく
- SCTのマウス操作のみで簡単にテーブル作成、自身でSQLスクリプトなどを手動作成不要



手順3: 収集したSQLを移行先DBで評価

- 収集SQLを移行先DBに対して実行
- IDTの実行モード
 - 「パース」モード
SQLの文法チェックのみでSQL文は実行しない
OracleとPostgreSQLの方言差異の文法チェックのみ行うケース
 - 「実行」モード
バインド変数を実際に適用したSQL文の実行などを確認・評価したい
- 実行モード以外にもいくつかのオプションを選択し実行可能

新規作成

アセスメント名 ?

ASSESSMENT3

評価SQLセット名 ?

SQL_SET3

ターゲットDB名 ?

RDS_for_PostgreSQL

実行タイプ ?

☒ 実行
 ☐ パース

トランザクション ?

☐ コミット
 ☐ ロールバック
 ☒ 指定なし

同時セッション処理数 ?

1

0秒の仮定値 ?

0.2

秒

バインド変数変換 ?

☐ 名前付きバインド変数書式を変換します。

新規作成

キャンセル

手順4: 評価結果の確認

- 移行先DB(ターゲットDB)での評価結果をチャートなどで確認可能
- 収集したSQLのうち何割が実行できなかったかが一目瞭然
- 結果をドリルダウンすることで実行できなかったSQLを表示できる
- エラーになったSQLをエラーメッセージとともに確認可能

実行できなかったSQL ?

34% - 1件

66% - 2件

エラーメッセージ ?

	エラーメッセージ	エラーコード	件数	
1	syntax error at or near ')	42601	1	100.0%

エラーユーザー ?

	DBユーザー	件数	
1	postgres	1	100.0%

エラープログラム ?

	プログラム	件数	
1	sqlplus@ora18c (TNS V1-V3)	1	100.0%

[失敗した全てのSQLを見る >](#)

失敗

syntax error at or near ')

[SQLの修正 ?](#)

[ソースSQL ?](#)

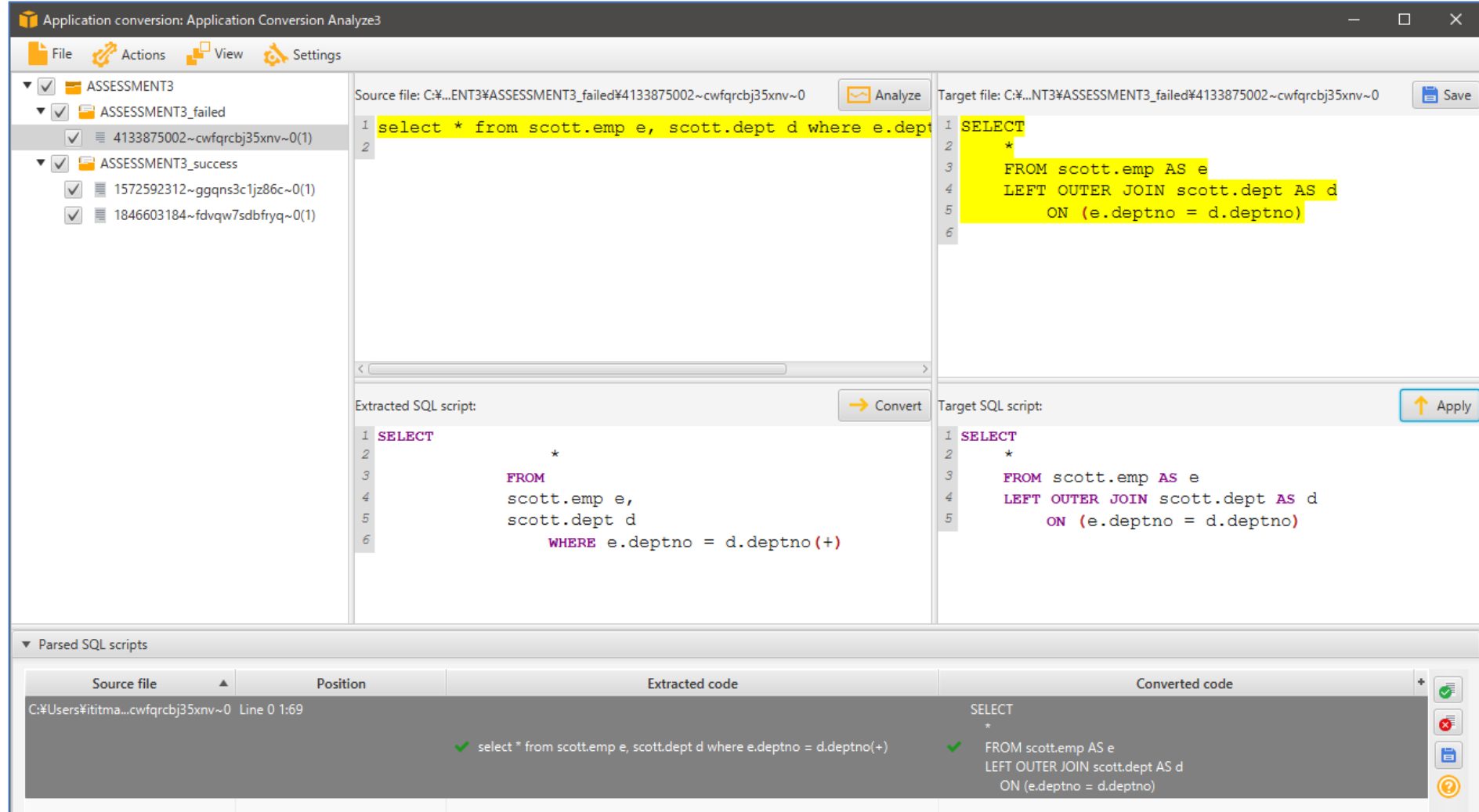
[詳細情報 ?](#)

[同一のSQL ?](#)

```
select
  *|
from
  scott.emp e,
  scott.dept d
where
  e.deptno = d.deptno (+)
```

手順5: (SCT)実行できなかったSQLをSCTで修正

- SQLエラーが複雑で難しい場合、SCTで修正
- IDTには、エラーとなったSQLを、SCTで取り込みやすいファイル形式でダウンロードする機能が備えてある
- よって、IDTで収集し評価エラーとなったSQL情報を簡単にSCTに反映可能



The screenshot displays the 'Application conversion: Application Conversion Analyze3' window. The interface is divided into several sections:

- File Explorer:** Shows a tree view of files under 'ASSESSMENT3', including 'ASSESSMENT3_failed' and 'ASSESSMENT3_success'.
- Source file:** Displays the original SQL query: `select * from scott.emp e, scott.dept d where e.deptno = d.deptno(+)`.
- Target file:** Shows the converted SQL query: `SELECT * FROM scott.emp AS e LEFT OUTER JOIN scott.dept AS d ON (e.deptno = d.deptno)`.
- Extracted SQL script:** Shows the SQL query with syntax highlighting.
- Target SQL script:** Shows the converted SQL query with syntax highlighting.
- Parsed SQL scripts:** A table at the bottom showing the mapping between the source file, position, extracted code, and converted code.

Source file	Position	Extracted code	Converted code
C:\Users\%ititma...\cwfqrcbj35xnv~0	Line 0 1:69	<code>select * from scott.emp e, scott.dept d where e.deptno = d.deptno(+)</code>	<code>SELECT * FROM scott.emp AS e LEFT OUTER JOIN scott.dept AS d ON (e.deptno = d.deptno)</code>

手順6: 修正したSQLを再実行

- SCTでの修正結果を、簡単にIDTへ取り込み可能
- 再度IDTでアセスメント実行し、修正後のSQLに対し評価可能

成功
 実行時間 0.000024 秒

[SQLの修正 ?](#)
 [ソースSQL ?](#)
 [詳細情報 ?](#)
 [同一のSQL ?](#)

```

SELECT
  *
FROM
  scott.emp AS e
  LEFT OUTER JOIN scott.dept AS d ON (e.deptno = d.deptno)
    
```

- IDTは、実行時間の目安情報を取得可能、実行速度が遅くなったか比較可能
 - 著しく実行速度が低下している場合、SQLを修正して再実行

遅くなったSQL ?

100% - 3件
0% - 0件

	SQLテキスト	DBユーザー	SRCDDB時間(秒)	TGTDDB時間(秒)	遅くなった割合(%)
1	select * from scott.emp e, scott.dept d where e.deptno = d.deptno	postgres	0.0	4.3e-05	43000000.0
2	SELECT * FROM scott.emp AS e LEFT OUTER JOIN scott.dept AS d ON (e.deptno...	postgres	0.0	2.1e-05	21000000.0
3	select * from scott.emp	postgres	0.0	1.8e-05	18000000.0

[成功した全てのSQLを見る >](#)

「IDT + SCT」移行ソリューションの〇×クイズ

1. アプリケーションコードに書かれたSQLは全てIDTで取得できる
2. IDTのアセスメントに失敗したSQLは、全てSCTを使って修正できる
3. SQLがターゲットDBで実行成功したら、ソースDBと同じ結果となる
 - ・ 更新クエリ(INSERT、UPDATE、DELETE)が同じ結果となる
 - ・ 選択クエリ(SELECT)が同じレコード数・順番となる

[答え]

1. × 実行されるSQLのみ取得。実行頻度の低いSQLの網羅性を高めるため、
 - ・ 一週間～一か月など、ある程度のまとまった期間のSQLを収集
 - ・ 週次・月次バッチで動作するSQL情報も取得する
2. × それでもエラーになるSQLは手動で修正する必要あり
3. × 現在IDTでは実行結果の一致・不一致の確認手段を備えていない。
要確認の留意事項：
 - ・ ORDER BYのない選択クエリ
 - ・ 暗黙の型変換や有効桁数
 - ・ 空文字やnullの扱い

- お客様要件(計3社)
オンプレOracleからAmazon PostgreSQLへの移行
移行におけるアセスメント及び移行コストの見積もり
- お客様はIDTに何を期待されているか？
 - 移行実施前の影響調査段階に、アセスメントツールとして使用したい
 - 非交換SQLの割合や種類を確認し、それに基づき移行コストを見積もりたい
 - 移行前後の性能比較を行い、アプリケーション書き替えコスト評価の1つの指標にしたい
- 弊社から「IDT+SCT」併用ソリューションを提案
 - 変換できなかったSQLの割合から、アプリケーション書き換え負荷を判断
 - 実行モードで取得されるSQL実行時間情報から、性能比較を実施

Insight Database Testing をおためしく下さい

- AWS Marketplace上でIDT、PISOを提供開始、簡単に利用可能
- AWS Marketplace上の製品ページ
 - アプリケーションテストツール「Insight Database Testing」
<https://aws.amazon.com/marketplace/pp/B08JQ8N5XG>
 - マルチデータベースに対応した監査ツール「PISO」
<https://aws.amazon.com/marketplace/pp/B08JQ93XX7>
- IDTの対応環境

ソースDB	Oracle Database / Microsoft SQL Server / PostgreSQL / MySQL
ターゲットDB	Oracle Database / Microsoft SQL Server / PostgreSQL / MySQL ※ 対応 OS および DB に関する詳細情報は、お問い合わせください。
マネジメントサーバー(IDT Manager)	VMware ESXiで動作する仮想アプライアンス vCPU 4コア～ メモリ 8GB ディスク 64GB～

ご清聴ありがとうございました。



記載されている会社名、サービス名、製品名は、株式会社インサイトテクノロジーおよび各社の商標または登録商標です。

Copyright 2021 Insight Technology, Inc. All Rights Reserved.