オンプレミスデータベースの AWS移行ガイド

アマゾン ウェブ サービス ジャパン株式会社

内容についての注意点

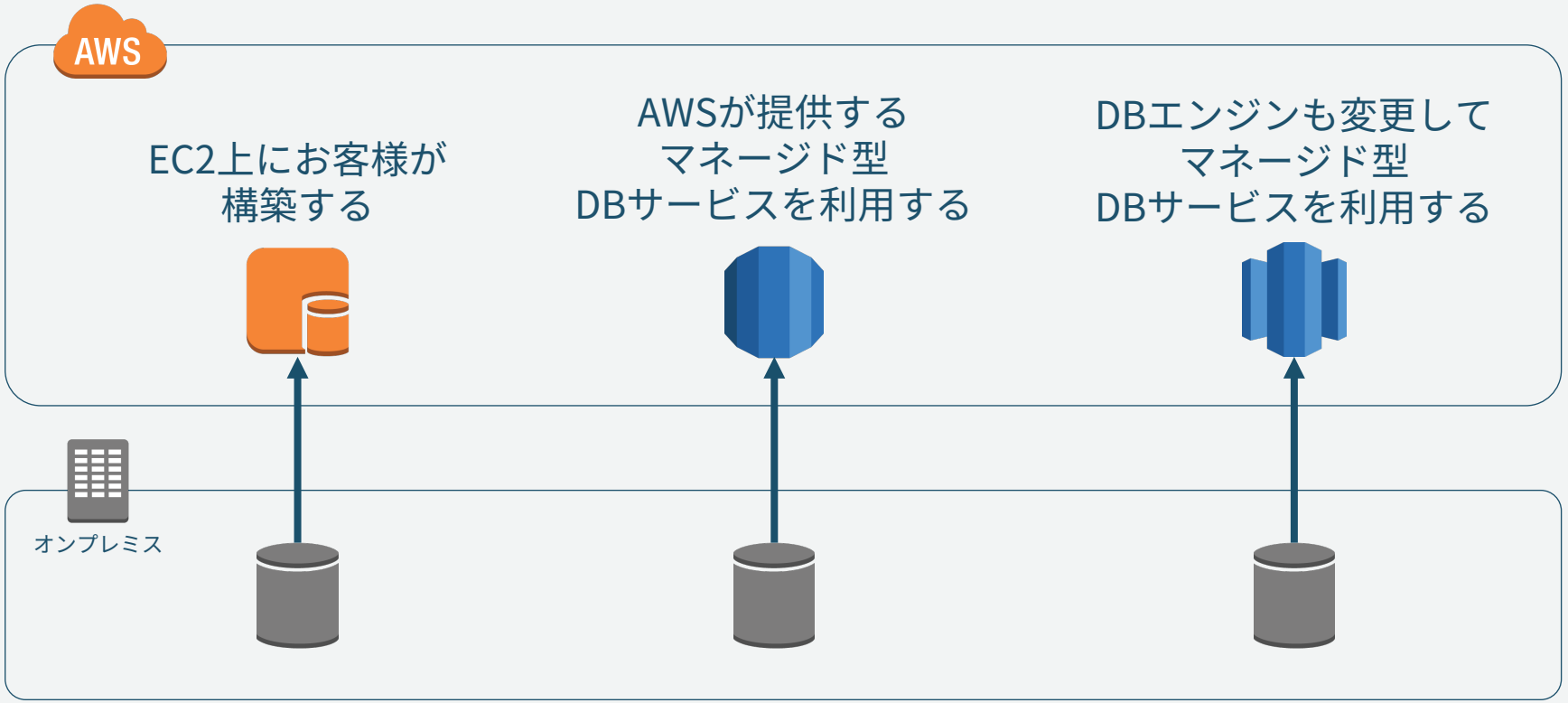
本資料では2017年10月31日時点のサービス内容および価格についてご説明しています。
最新の情報はAWS公式ウェブサイト (http://aws.amazon.com/) にてご確認ください。

- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます
- 価格は税抜表記となっています。日本居住者のお客様が東京リージョンを使用する場合、別途消費税をご請求させていただきます

AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at http://aws.amazon.com/agreement/. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

aws

オンプレミスデータベースの移行とは



データベースエンジンを変更せずに AWSに移行したいお客様の声

- クラウドにシステム全体を移行するため、データベースもクラウドに移行したい
- 障害復旧やパッチ適用などの管理の手間を減らしたい

などなど

データベースエンジンを変更して AWSに移行したいお客様の声

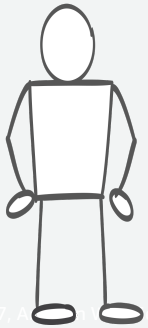
- クラウドにシステム全体を移行するのであれば、RDBMSもクラウドネイティブなものにしたい
- RDBMSもオートスケールさせたいが商用DBがCPUライセンスだとそれができない
- IT予算の多くを商用DBライセンスが占めている

などなど

移行にあたっての不安や悩み

業務部門 / アプリ開発部門

- アプリケーションへの影響は
  どれくらいあるのだろうか
- 移行のための業務停止は
  できるだけ短くしたい



IT部門 / インフラ管理部門

- 業務部門に何をガイドして
  よいのか分からない
- 従来の管理手法が
  どう変わるのだろうか
- 移行のための費用は
  あまりかけたくない

移行にあたって決めなくてはいけないこと

1. What?
   対象システムは？
2. Why?
   移行理由は？
3. How?
   移行戦略は？
4. Where?
   移行先は？
5. When?
   期限は？
6. Who?
   担当者は？

AWSの考えるクラウドへの6つの移行戦略 (1/3)

Re-Host | ホスト変更

サーバーやアプリケーションを
オンプレミス環境からクラウドに
そのまま持ってくる

オンプレミスのPostgreSQLを
そのままEC2に持ってくる

Re-Platform | プラットフォーム変更

クラウド移行の一環として
プラットフォームの
アップグレードを行なう

オンプレミスの MySQL 5.5 を
EC2に構築した5.6に移行する

AWSの考えるクラウドへの6つの移行戦略 (2/3)

Re-Purchase | 買い換え

クラウド対応したライセンス
またはアプリケーションに
買い換える

オンプレミスのOracleを
ライセンス込みの
RDS for Oracle に買い換える

Refactor | 書き換え

クラウド環境で最適に動作する
ようにアプリケーションを
書き換える

オンプレミスの SQL Server を
AuroraやRedshiftに変更する

AWSの考えるクラウドへの6つの移行戦略 (3/3)

Retire | 廃止

サーバーやアプリケーションを
廃止する

平行運用していた
古いシステムをDBごと
このタイミングで廃止する

Retain | 保持

オンプレミス環境を
引きつづき使用する

Oracle9iに依存している
アップグレードできない
パッケージアプリケーションが
ある

クラウド移行戦略を移行の複雑さで比較

- Retain | 保持
- Retire | 廃止
- Re-Host | ホスト変更
- Re-Purchase | 買い換え
- Re-Platform | プラットフォーム変更
- Refactor | 書き換え

aws

移行にあたって決めなくてはいけないこと

1. What?
   対象システムは？
2. Why?
   移行理由は？
3. How?
   移行戦略は？
4. Where?
   移行先は？
5. When?
   期限は？
6. Who?
   担当者は？

AWSが提供するデータベースサービス

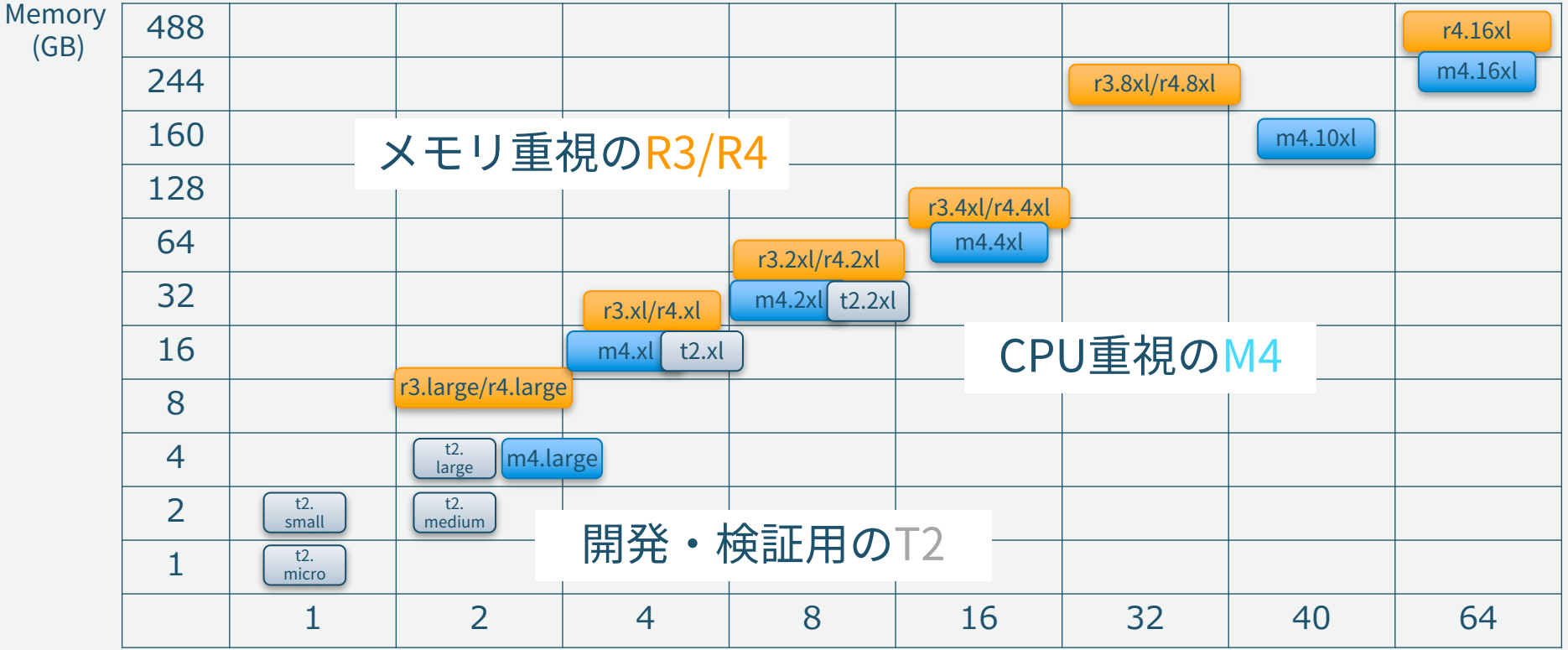
 Amazon RDS	完全マネージド型で、セットアップ、運用、拡張が容易なリレーショナル・データベースサービス
 Amazon DynamoDB	完全マネージド型で、高速なパフォーマンス、シームレスな拡張性と高い信頼性のNoSQLサービス
 Amazon Redshift	完全マネージド型で、高速で管理も万全なペタバイト規模のデータウェアハウスサービス
 Amazon ElastiCache	完全マネージド型で、セットアップ、運用、拡張が容易なキャッシュサービス

aws

Amazon RDS

- 構築
  - 数クリック or APIでDBサーバーを操作
  - 初期費用なし、時間単位の従量課金
- 親和性
  - 6種類のエンジンをサポート
  - 既存アプリケーションの変更不要
- 運用
  - 可用性向上のための機能
  - モニタリング、障害検出/復旧、パッチ、スケーリングが容易
- セキュリテイ
  - VPC、セキュリティグループ、暗号化等に対応

DBインスタンスタイプ



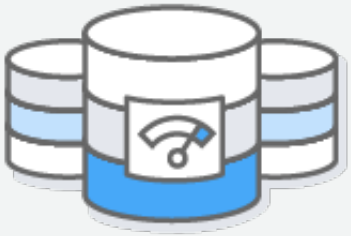
※DBエンジンによって使用できるインスタンスの種類が異なる
※図には記載していない旧世代インスタンスも選択可能

aws


MySQL
Amazon
Aurora
PostgreSQL

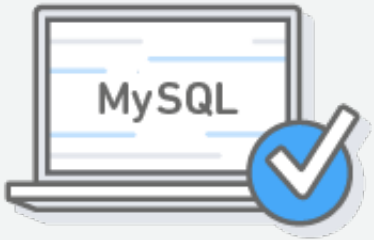
Amazon Aurora

Amazonがクラウド時代に再設計したデータベース

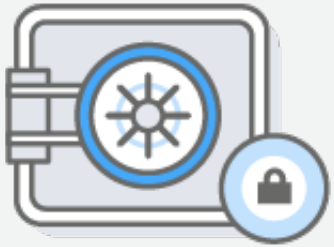


ハイパフォーマンス

スケーラブル



OSS-DB互換



セキュリティにも配慮

フルマネージド

高可用性・高耐久性

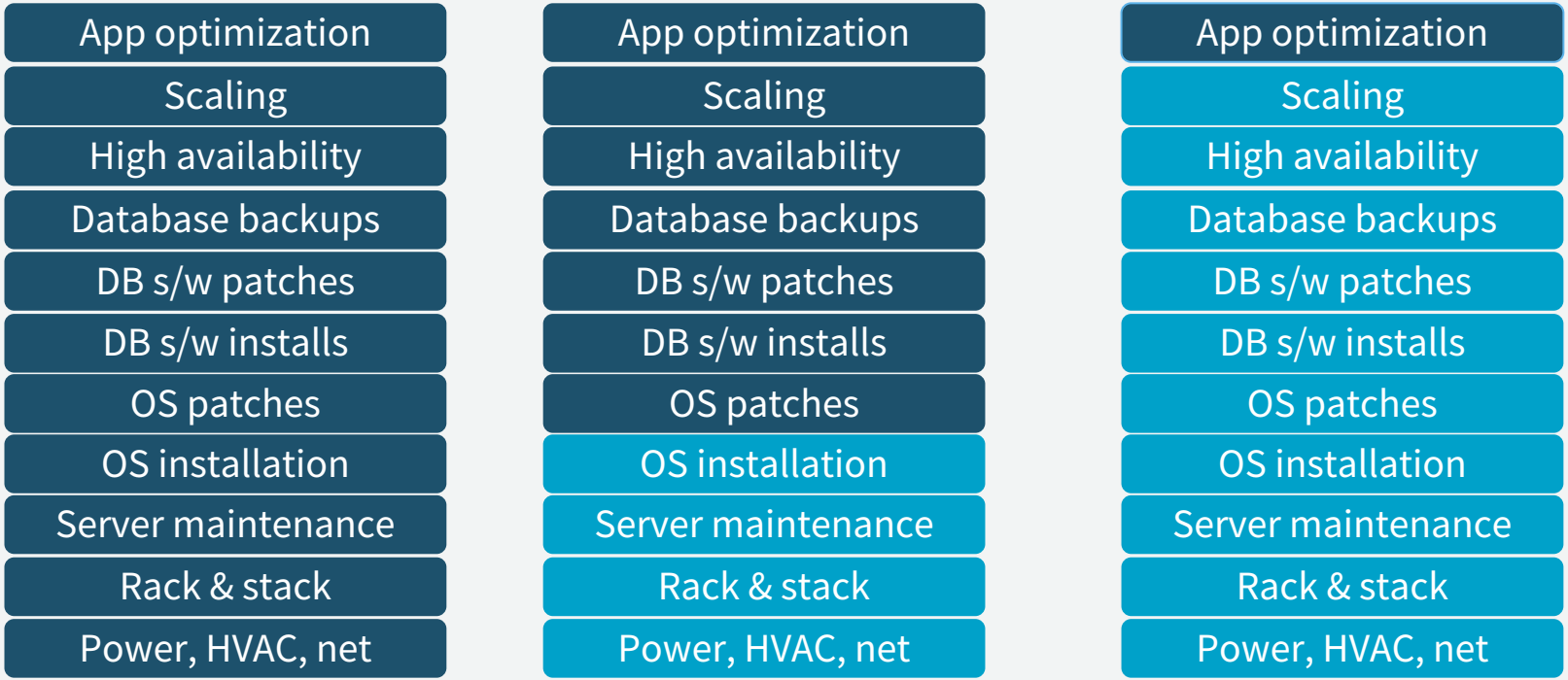
Amazon Redshift

フルマネージドのデータウェアハウスサービス

- 160GBから最大2PBまで拡張可能
  - 最小1ノードから最大128ノードまで拡張可能
- 超並列 (MPP) & カラムナ型DBエンジンによる高速処理
- ほかのAWSサービスとの高い親和性
  - S3からのロード、S3へのアンロード、
    S3上のデータを外部表として直接参照可能
  - DynamoDB / EMR / EC2 (SSH) とのデータ連携可能
- 従来のDWHの1/10のコストを時間単位の従量課金で

aws

オンプレミスとの管理負担を比較



お客様がご担当する作業　　AWSが提供するマネージド機能

3つのRDBMSの特性概要

	Aurora MySQL	Aurora PostgreSQL	Redshift
OLTP性能	★★★	★★★	-
OLAP性能	-	★★	★★★
目標レスポンス時間	数ミリ秒から数十秒	数ミリ秒から数十分	数秒から数時間
結合方法 *1	ネステッドループ *2	ネステッドループ、 ハッシュ、ソートマージ	ハッシュ、 ソートマージ
インデックス *1	Bツリー、空間、 全文	Bツリー、関数、空間、 全文、ゾーンマップ	ゾーンマップ
制約 *1	NOT NULL、PK、 UNIQUE、FK	NOT NULL、PK、 UNIQUE、FK、CHECK	NOT NULL *3

*1 移行を前提としているため、Oracle Databaseにない機能は記載省略
*2 Block Nested Loop 結合と Batched Key Access 結合を含む
*3 PK、UNIQUE、FKは定義できるが強要されない

aws

Statspack/AWRから現行ワークロードを評価

- 短時間のSQLが大量に流れているのか、
  長時間のSQLが少量流れているのかを見極める
- SQL ordered by Elapsed Time を調査
  - Elapsed Time は期間中の合計時間
  - Elapsed Time per Exec が短い = 短時間のSQLが大量
  - Elapsed Time per Exec が長い = 長時間のSQLが少量

移行にあたって決めなくてはいけないこと

1. What?
   対象システムは？
2. Why?
   移行理由は？
3. How?
   移行戦略は？
4. Where?
   移行先は？
5. When?
   期限は？
6. Who?
   担当者は？

期限と担当者を決めるには工数見積もりが必要
工数 = 移行先との違いの量

aws

AWSが提供する移行支援サービス

- AWS TCO Calculator
- AWS Application Discovery Service

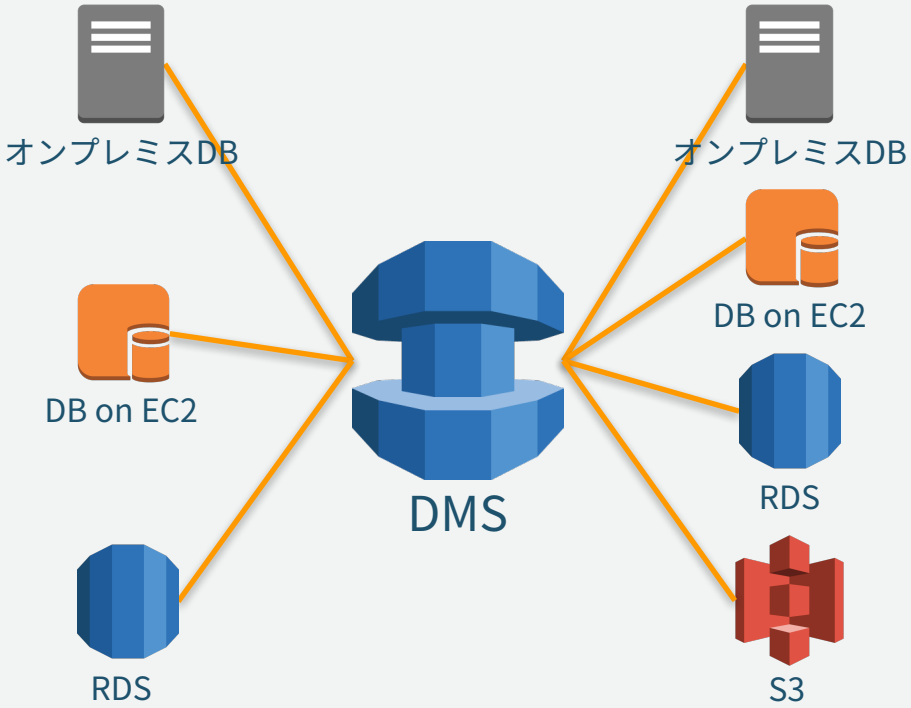
- VM Import/Export
- AWS Server Migration Service
- AWS Database Migration Service
- AWS Schema Conversion Tool
- AWS Storage Gateway
- AWS Snowball
- VMware on AWS

- AWS CloudWatch
- AWS Config
- AWS CloudFormation
- AWS CloudTrail
- AWS Service Catalog
- AWS Trusted Advisor
- AWS サポート

AWS Database Migration Service とは

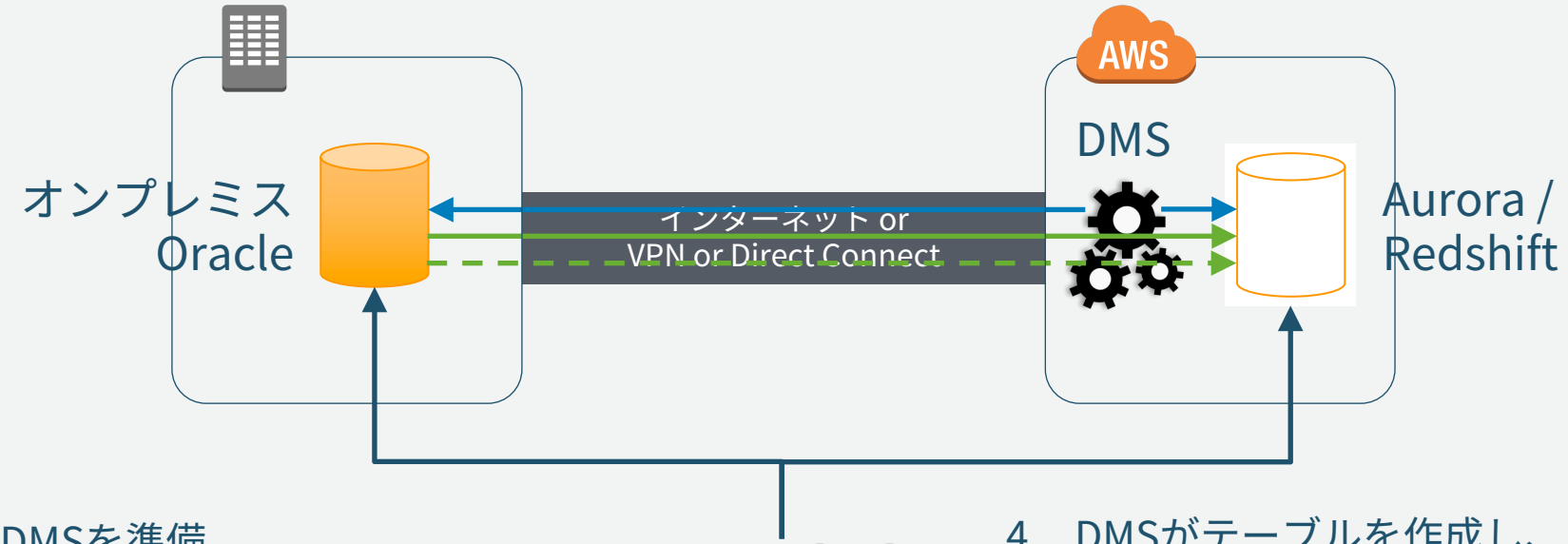
既存のデータベースを
最小限のダウンタイムで
マイグレーションする
サービス

同種はもちろん
異種プラットフォームの
移行にも対応



※オンプレミス to オンプレミスは非対応

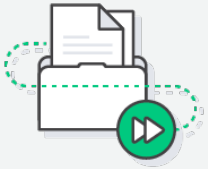
移行中もアプリケーションは稼働したまま



1. DMSを準備
2. DMSがソースDBと
   ターゲットDBに接続
3. 対象のテーブル、スキーマ
   などを選択
4. DMSがテーブルを作成し、
   データをロードし、
   レプリケーション開始
5. 任意のタイミングで
   アプリケーションを
   ターゲットDB側に切り替え

AWS Database Migration Service の特徴

最小限のダウンタイム
オンラインでの
継続的レプリケーション対応



簡単なセットアップ
ソースDBへの変更はほぼ不要

使用が簡単
MCで数回クリックするだけ

高い信頼性
マルチAZ可能なインスタンス

豊富な
対応プラットフォーム
Oracle, Microsoft SQL Server,
SAP ASE, MySQL, MariaDB,
PostgreSQL, Aurora, Redshift,
S3, MongoDB, DynamoDB

低コスト
c4.largeインスタンスで
0.196USD/時間

aws

対応データベース詳細

プラットフォーム	ソース	ターゲット
Oracle Database	10g R2, 11g, 12c	10g, 11g, 12c
Microsoft SQL Server	2005, 2008, 2012, 2014	2005, 2008, 2012, 2014
SAP ASE	15.7以降	15.7以降
MySQL / MariaDB / Aurora	5.5以降	5.5以降
PostgreSQL	9.4以降	9.3以降
Redshift	-	すべて
S3	-	すべて
MongoDB	2.6.x, 3.x	-
DynamoDB	-	すべて

aws

設定の流れ

1. DMSインスタンスの作成
   - インスタンスサイズ、VPC、マルチAZ、パブリックアクセス
     （ストレージサイズ、サブネットグループ、AZ、セキュリティグループ、KMSキー）
2. エンドポイントの設定
   - ソースとターゲットそれぞれのRDBMSプラットフォーム、ホスト名、ポート、SSL有効無効、ユーザー名、パスワード
3. タスクの定義
   - 移行タイプ
     （ターゲットテーブルが存在していた場合、LOB対応、最大LOBサイズ、ログの有効化）

実行

aws

移行タイプ

- 既存のデータを移行する (FullLoad)
  - 現時点でソースDBに入っているすべてのデータを
    ターゲットDBに移行する
- データ変更のみをレプリケートする
  (Change Data Capture / CDC)
  - ソースDBに対する変更データをキャプチャし、
    ターゲットに適用する
  - アプリケーションは稼働したまま移行可能
- 既存のデータを移行して、継続的な変更をレプリケート

aws

FullLoadの仕組み ターゲットがRedshift以外

1. DMSインスタンスがソースDBから8†テーブル並列に、各テーブルから1万†行ずつSELECT
2. 必要に応じてDMSインスタンスがデータを変換し、ターゲットDBにINSERT
    - メモリーだけで処理しきれない場合は、DMSインスタンスのEBSがスワップ領域として使われる
    - 行長が長く、行長×1万行×8テーブルがDMSインスタンスのメモリーサイズを超える場合は読み取り行数単位を小さく

† デフォルト値であり、ユーザーが変更可能

aws

CDCの仕組み

1. DMSインスタンスがソースDBのトランザクションログ（バイナリログ、WAL、REDOログ）を5秒間隔でキャプチャ
2. DMSインスタンスがSQLに変換し、トランザクションCOMMIT順にターゲットDBに実行
3. ターゲットDBに接続できない、遅い場合 or FullLoadが完了していない場合は、DMSインスタンス内にキューイング

東京リージョンでの料金

インスタンス

+

ストレージ

+

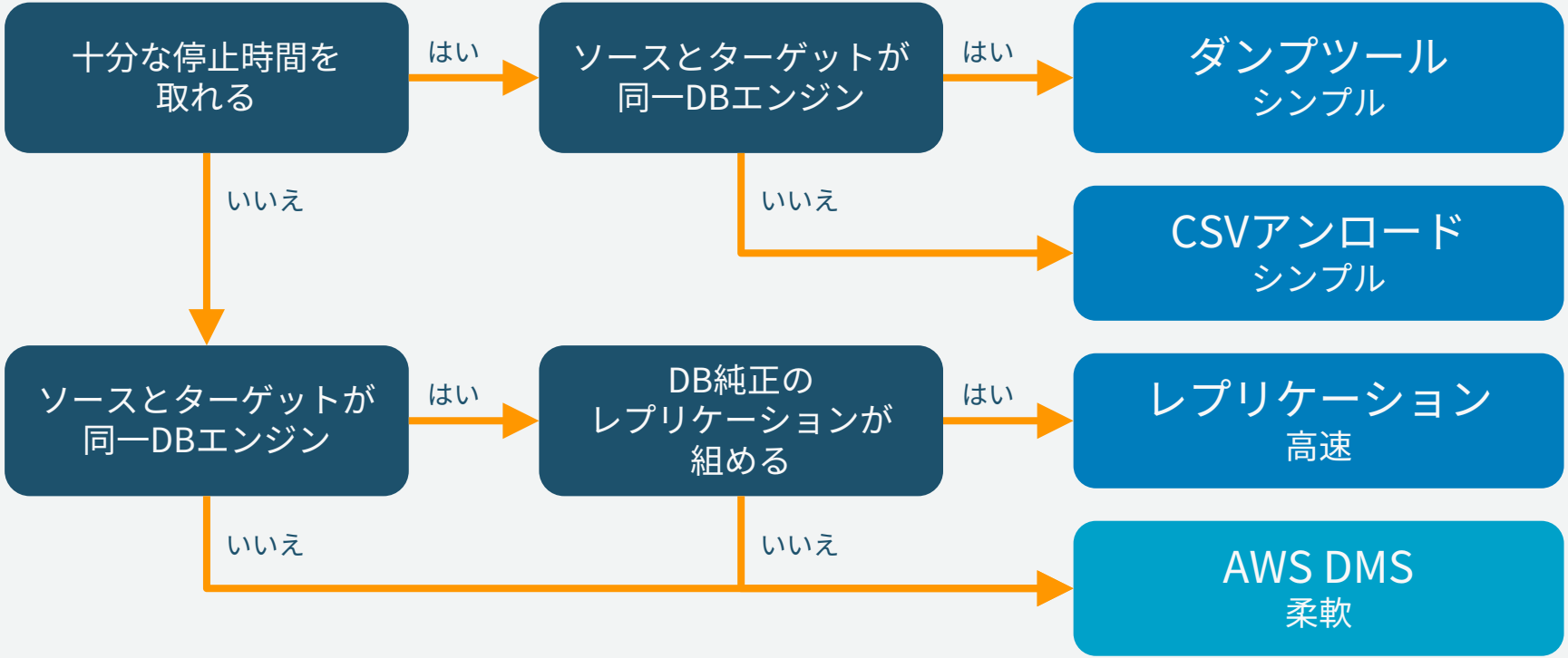
データ転送

t2.micro: 0.028USD/時 から
c4.4large: 1.564USD/時 までの8種類
マルチAZの場合は2倍

0.138USD/GB/月
マルチAZの場合は2倍

受信: 無料
同一AZ内への送信: 無料
別AZへの送信: 0.010USD/GB
別リージョンへの送信: 0.090USD/GB
インターネットへの送信: 0.14USD/GB以下

DMS以外の移行方法



AWS Schema Conversion Tool とは

ソースDBのスキーマ、ビュー、

ファンクション、ストアド

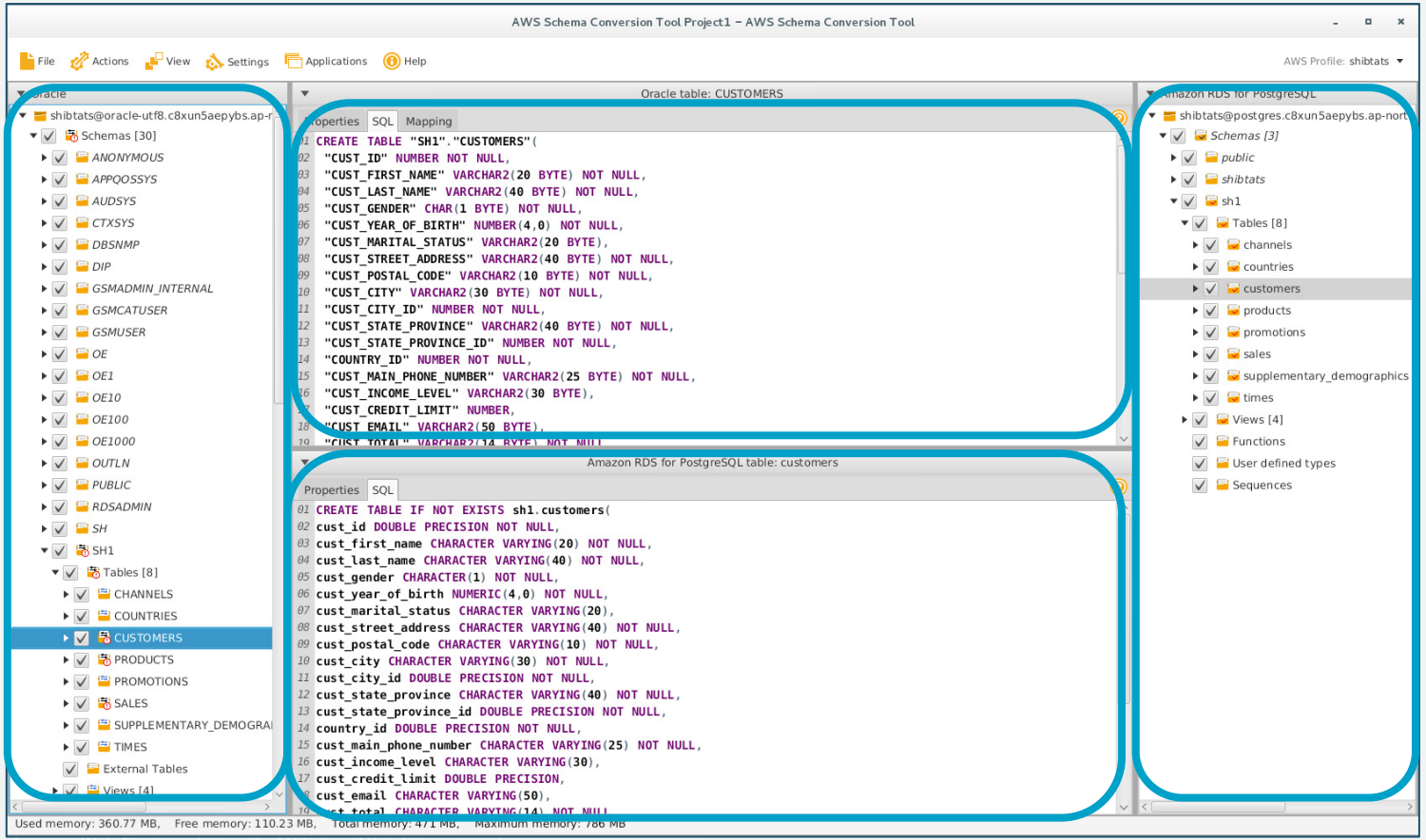
プロシージャの大部分を

自動的にターゲットDB

互換フォーマットに変換できる

デスクトップアプリケーション

メインビュー



AWS Schema Conversion Tool の特徴

手動変換の補助
自動変換できない個所とその理由を明示

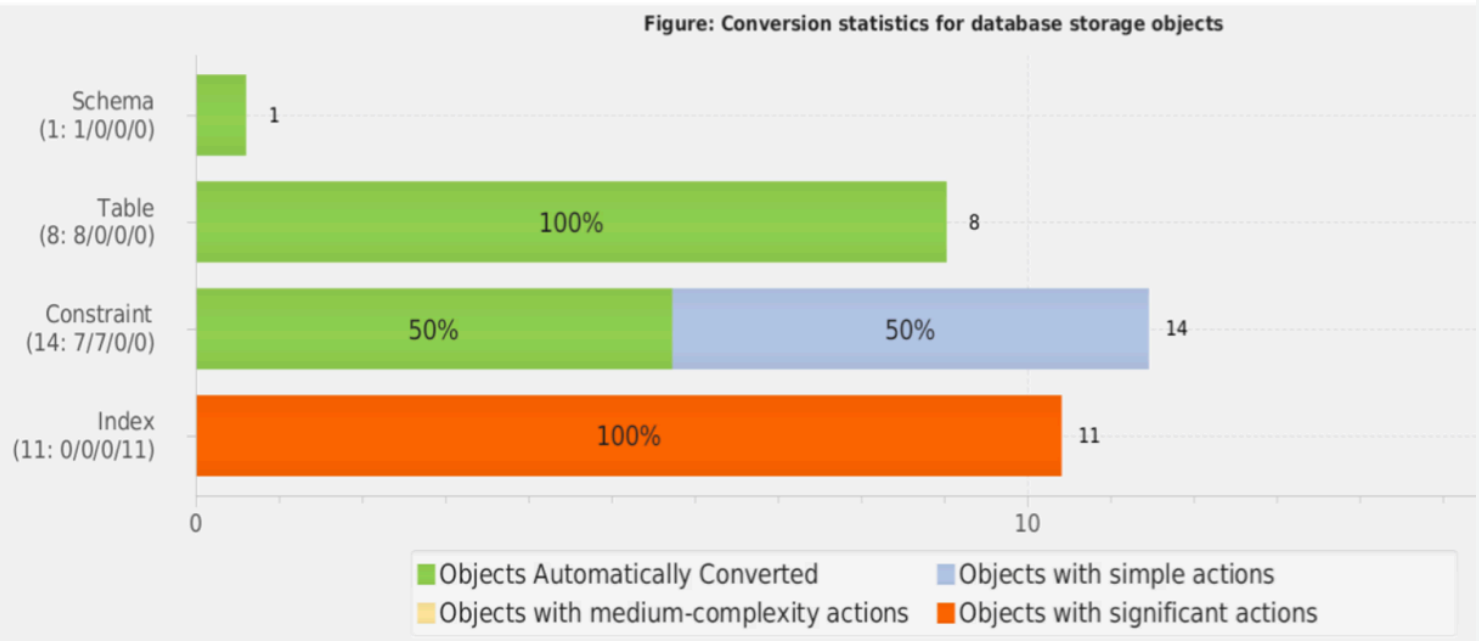
評価レポートの作成
何割のオブジェクトが自動変換可能か
などのPDFレポートを数クリックで
作成でき、変換工数の事前見積もりを
補助

アプリケーションSQLに対応
アプリケーションソースコードを
スキャンして変換

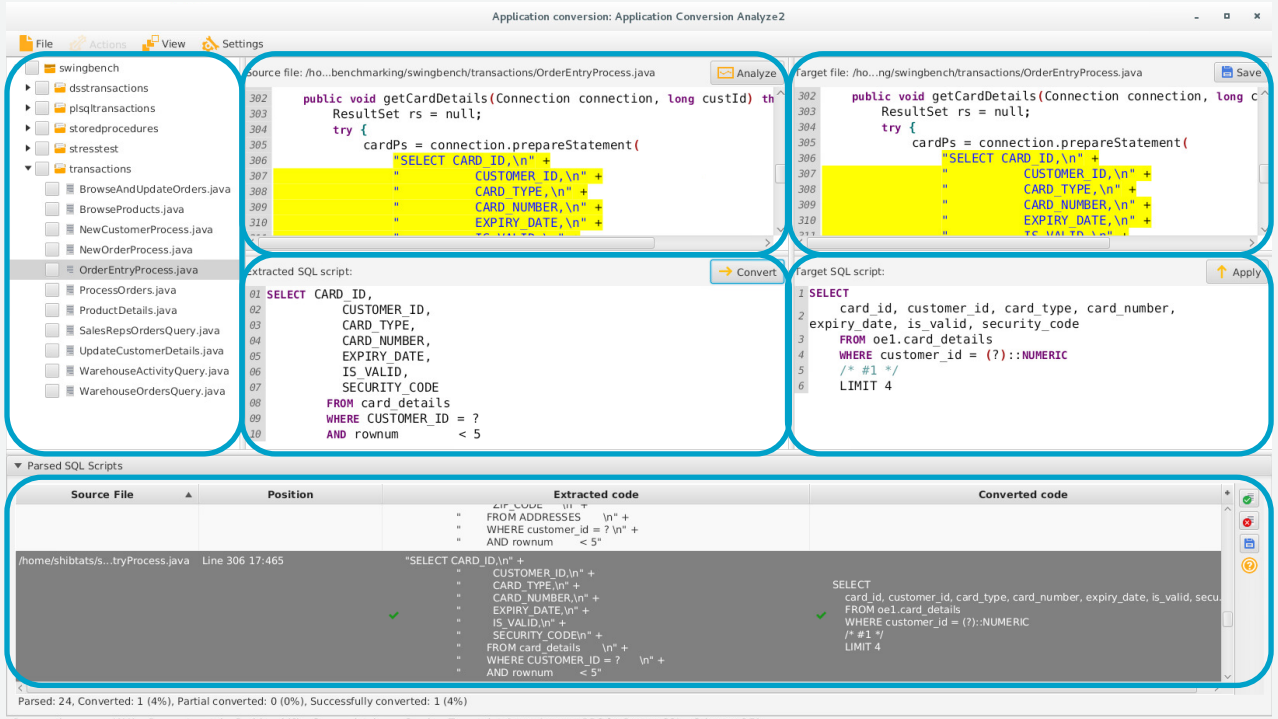
豊富な対応プラットフォーム
Oracle, Microsoft SQL Server, Teradata,
Netezza, Greenplum, Vertica,
MySQL, MariaDB, PostgreSQL,
Aurora, Redshift

評価レポート

完全自動変換できるまたは手動変更工数ごとに
色分けされたグラフなどを含んだレポート



アプリケーションSQLの変換



- (+) 結合
- ROWNUM
- 関数の多く

など

対応データベース詳細

ソース	ターゲット
Oracle Database 10.2以降	Aurora (MySQL、PostgreSQL)、 MySQL、Oracle、PostgreSQL、Redshift
SQL Server 2008以降	Aurora (MySQL、PostgreSQL)、 SQL Server、MySQL、PostgreSQL、Redshift
MySQL 5.5以降	Aurora (PostgreSQL)、MySQL、PostgreSQL
PostgreSQL 9.1以降	Aurora (MySQL)、MySQL、PostgreSQL
Greenplum 4.3以降	Redshift
Netezza 7.0.3以降	Redshift
Teradata 13以降	Redshift
Vertica 7.2.2以降	Redshift

aws

設定の流れ

1. デスクトップ環境にインストール
2. ソースDBの設定
3. スキーマの選択
4. 評価レポートの確認
5. ターゲットDBの設定

- Window, Mac, Fedora, Ubuntu + JRE 8u45以降
- プロジェクト名、OLTP or DW、RDBMSプラットフォーム、ホスト名、ポート、ユーザー名、パスワード、SSL有効無効、JDBCドライバーのパスなど
- RDBMSプラットフォーム、ホスト名、ポート、ユーザー名、パスワード、SSL有効無効、JDBCドライバーのパスなど

aws

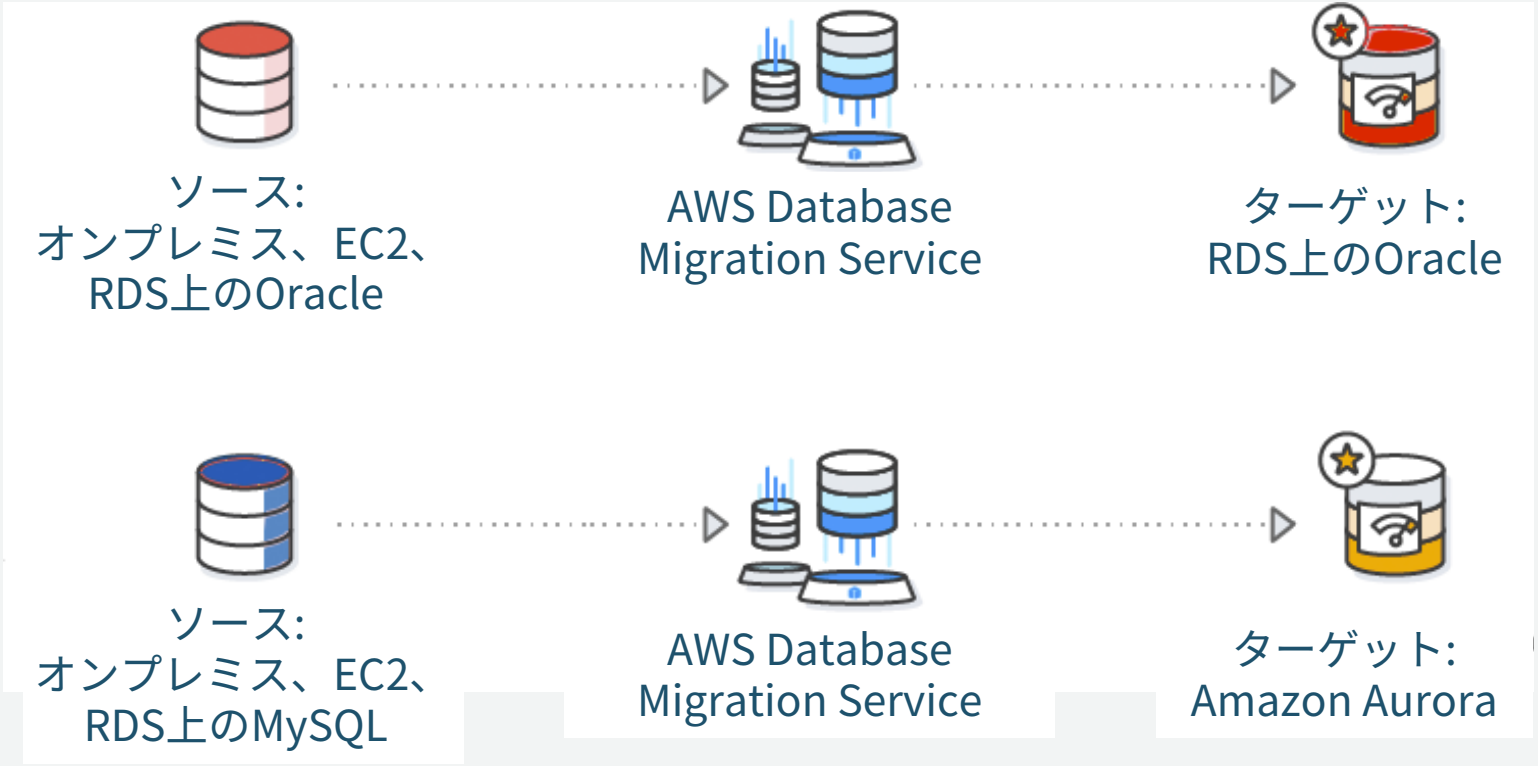
マッピングルールの作成

- SCTのデフォルトマッピングルールを変更可能
  - データタイプの変更
    `INTEGER ← NUMBER(10,0) → NUMERIC(10,0)`
  - オブジェクトを別スキーマに移動
  - オブジェクトの名称変更
  - プレフィックスの追加、削除、置換
  - サフィックスの追加、削除、置換
- データベース、スキーマ、テーブル、列単位で設定

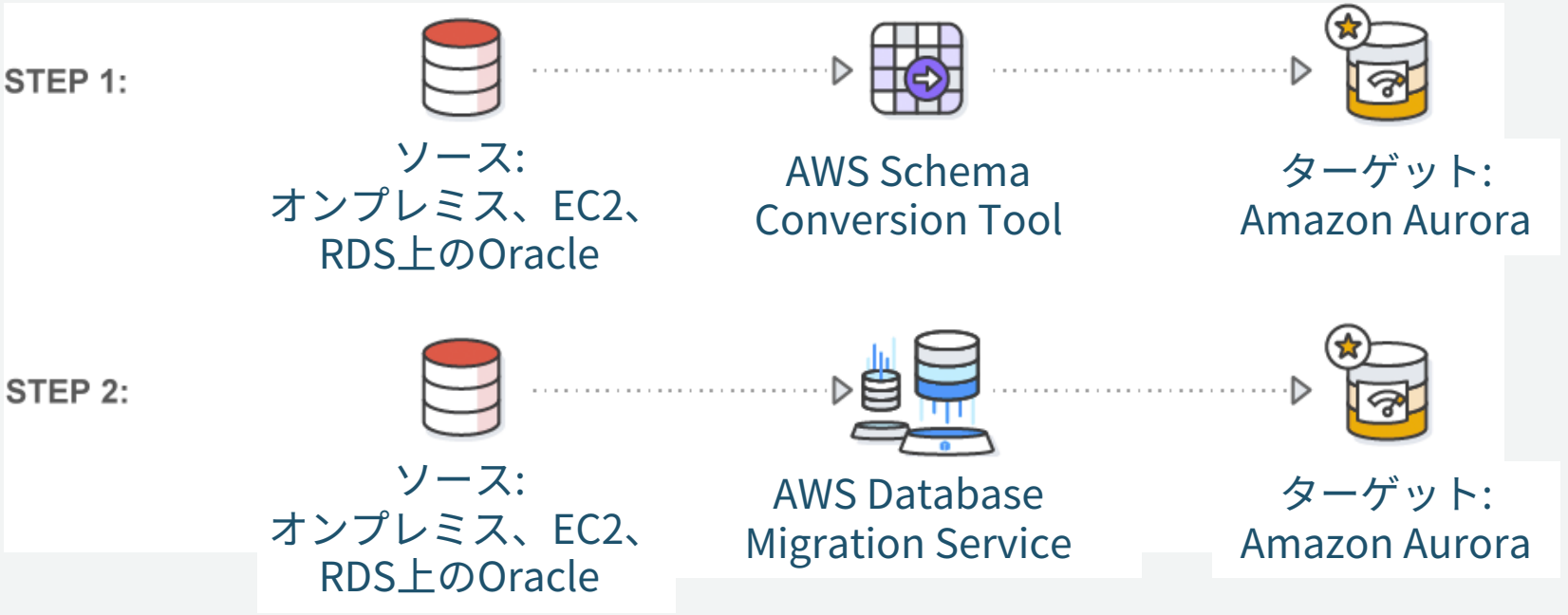
データ移行エージェント

- ソースDBのデータをCSV形式でアンロードするエージェント
- S3にアップロードするまで、RedshiftへCOPYするまで自動化することも可能
- アンロードしたファイルをSnowballなどでAWSへ。DMSでは時間が掛かりすぎる場合の代替案
- 現在対応しているソースDB:Greenplum、Netezza、Oracle、Teradata、Vertica

同種DB間で、ダウンタイムを最小限に移行



異なるDB間で、ダウンタイムを最小限に移行



データベースの統合

ソース:
オンプレミス、EC2、RDS上
の複数のMySQL

AWS Database
Migration Service

ターゲット:
Amazon Aurora

継続的なデータレプリケーション

ソース:
Amazon Aurora

AWS Database
Migration Service

ターゲット:
異なるリージョンのAurora、
テスト用Aurora、
オンプレミスMySQL