



PGECcons
PostgreSQL Enterprise Consortium

商用DBからPostgreSQLへの移行検証結果 ～安心してPostgreSQLをエンタープライズ領域で利用するために～

2014年12月12日
PostgreSQL エンタープライズ・コンソーシアム
技術部会

<http://www.pgecons.org/>

自己紹介

- 名前
 - 小山田 政紀(おやまた まさのり)
- 所属企業
 - 株式会社富士通ソーシャルサイエンスラボトリ
(略称: 富士通SSL)
 - 富士通SSLはWG2に参画 本年度のWG2主査企業
(WG2に参画し、3年目)
- 経歴
 - 2010/4 入社
 - 2010/7～10
 - 社内システムの開発等
 - 2010/11～
 - OSS調査・検証(Eucalyptus、KVM等)
 - OSSミドルウェアサポート(PostgreSQL担当)
- 得意分野
 - PostgreSQL
 - OSSの検証作業



FUJITSU 富士通ソーシャルサイエンスラボトリ

目次

1. はじめに(本日本お伝えしたいこと)
2. PGEConsについて
3. 技術部会の活動内容
4. PostgreSQL技術紹介
5. さいごに

1. はじめに(本日お伝えしたいこと)

- ① 企業の基幹業務へのPostgreSQLの導入が加速している
- ② ミッションクリティカル性の高い領域へのPostgreSQLの普及促進、課題解決に奮闘している企業連携の団体(PGECons)があること
- ③ PGEConsの活動内容と活動成果のご紹介

PostgreSQLのエンタープライズにおける活用

【PGEConsで紹介してきた活用事例】

■ 住友電気工業様

- 2005年より社内標準DBをPostgreSQLに

■ ヤマハモーターソリューション様

- 世界各地の拠点の基幹業務システムに採用

■ 株式会社キャム様

- クラウド・SaaS型 統合基幹業務システム「CAM MACS」

■ 株式会社スポットライト様

- スマートフォンを使った共通来店ポイントサービス「スマポ」

PostgreSQLのエンタープライズにおける活用

【その他の導入事例】

■ 英気象庁

- 年間サポート費用と保守費用の負担が大きいため、商用DBMSへの依存度を低下させることを狙って、PostgreSQLを活用

- ITメディア様の記事より

<http://techtarget.itmedia.co.jp/tt/news/1407/24/news01.html>

■ 仏社会保障システム

- メインフレームからの移行

- Let's postgres の記事より

http://lets.postgresql.jp/documents/case/Bull_Case_in_France/

エンタープライズ領域へのPostgreSQLの普及が進んでいる

2. PGECons (PostgreSQL Enterprise Consortium) について

2. PGEECons について

■ PGEEConsの発足と目的

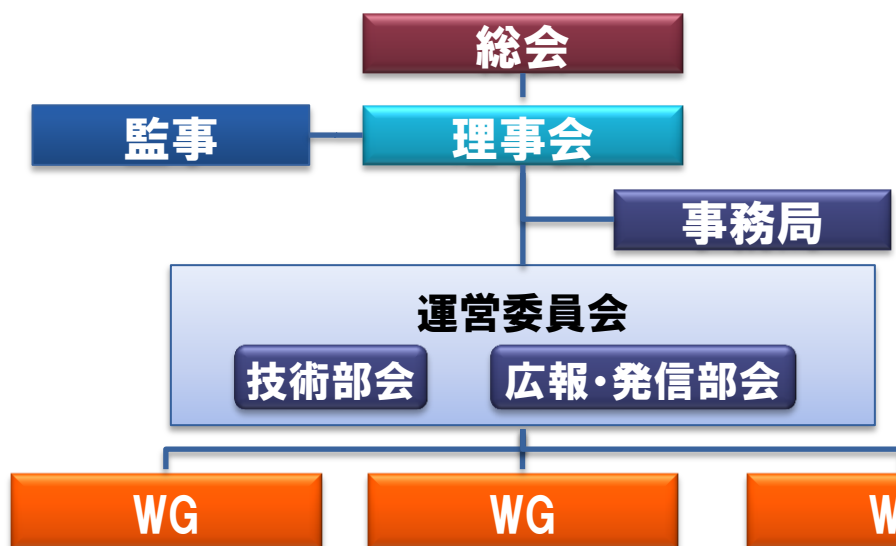
- 2012年4月11日発足
- ミッションクリティカル性の高い**エンタープライズ領域への PostgreSQLの普及を推進**するため、各種ツールや PostgreSQL本体に関する**利用技術情報**の収集と提供および、その整備などの**活動を企業ベースで展開**する

活動項目	概要
情報発信	会員の導入実績を基に、PostgreSQLおよび周辺ツールに関する情報を集約し、情報発信サイトや 세미나等を通じて提供する
共同検証	エンタープライズ領域への適用に向けて必要となる情報を、実証を通じて充実を図る
開発コミュニティへのフィードバック	よりミッションクリティカル性の高い領域への適用に向けた技術的な課題を集約し、開発コミュニティに要望を発信する
開発プロジェクト支援	会員間での機能拡張に関する連携開発や、必要な周辺ツールの開発プロジェクト支援を進める

平成26年度 体制・会員構成

■ 会員は法人およびそれに準ずる団体で構成

平成26年度体制



理事長	: 日本電信電話株式会社
運営委員長	: 日本電気株式会社
運営副委員長	: 株式会社 日立製作所
技術部会長	: 富士通株式会社
広報・発信部会長	: 日本ヒューレットパッカード株式会社
広報・発信副部会長	: 株式会社アシスト
事務局長	: SRA OSS, Inc. 日本支社
監事	: 税理士法人ジャストスタッフ

種別		概要	総会議決権
正会員	理事	理事会に参加、理事長および運営委員長は理事のうちから就任する	あり
	運営委員	運営委員会に参加、部会長およびWG長は運営委員から就任する	
		ワーキンググループ(WG)に参加し、活動に貢献	
一般会員		メーリングリストやWebなどから、活動情報を取得することが可能	無し

平成26年度 参加会員一覧(1/2)

(2014年12月12日 現在)

■ 47社(正会員 17社、一般会員30社)が活動

参加法人名 (順不同)			
正会員 (17社)	運営委員	(株)アシスト	日本ヒューレット・パカード(株)
		SRA OSS, Inc. 日本支社	(株)日立製作所
		NECソリューションイノベータ(株)	(株)日立ソリューションズ
		日本電気(株)	富士通(株)
		日本電信電話(株)	(株)富士通ソーシアルサイエンスラボラトリ
		TIS(株)	特定非営利活動法人 LPI-Japan
		サイオステクノロジー(株)	フューチャーアーキテクト(株)
		NTTソフトウェア(株)	ZEKKO(株)
		大日本印刷(株)	

平成26年度 参加会員一覧(2/2)

(2014年12月12日 現在)

	参加法人名（順不同）		
一般会員 (30社)	(株)アイ・ティ・プロデュース	(株)キャム	日本アイ・ビー・エム(株)
	岩通ソフトシステム(株)	キーウェアソリューションズ(株)	(株)フィックスターズ
	(株)エクサ	ジャパンシステム(株)	(株)マインド
	SFKメディカル(株)	住友電気工業(株)	ミュートック(株)
	(株)エニブラ	住友電工情報システム(株)	(株)メトロシステムズ
	(株)エム・オー・エム・テクノロジー	(株)seiwa	ヤマハモーターソリューション(株)
	エンタープライズDB(株)	(株)中電シーティーアイ	ローリーコンサルティング(株)
	(株)オンザマーク	(株)デジタル・ヒュージ・テクノロジー	(株)アイ・アイ・エム
	関電システムソリューションズ(株)	(株)ニイズ	(株)インフォメーションクリエイティブ
	クオリカ(株)	東芝ソリューション(株)	(株)ギークフィード

WGの活動風景

■ 検討会



■ 実機検証

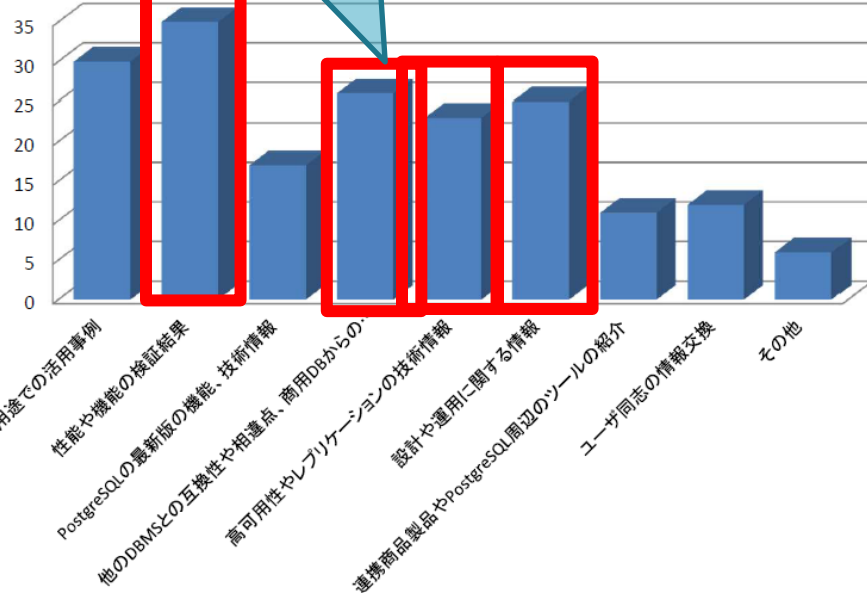


3. 技術部会の活動内容

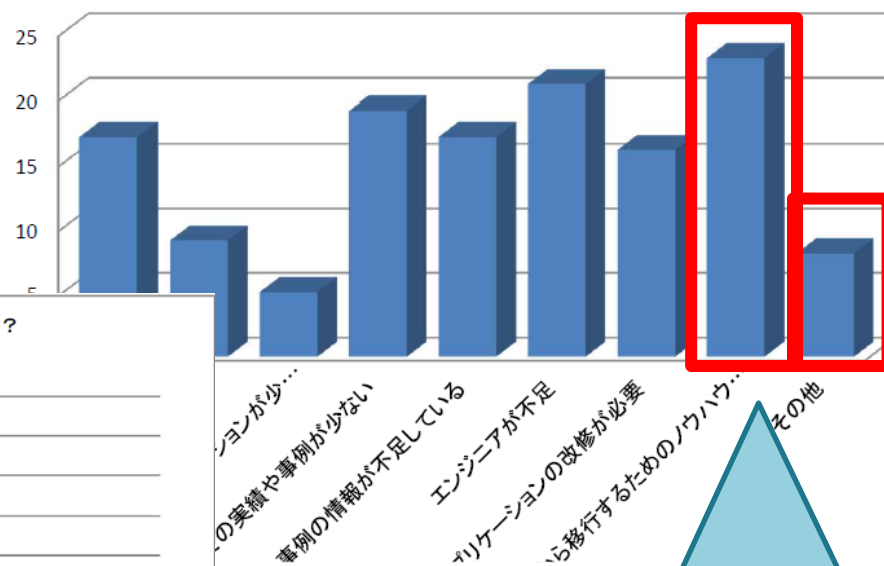
過去セミナーのアンケートよりテーマを選定

PGEConsへの**期待**
「性能や移行、高可用性、
設計・運用に関する
情報発信」

Q6. PGEConsに何を期待していますか？またどんな情報を発信してほしいですか？
(複数回答可)



Q5. PostgreSQLをよりミッションクリティカル性の高いエンタープライズ領域で採用するための課題は何だとお考えですか？(複数回答可)



PostgreSQL採用の**課題**
「個々のコメントから具体的テーマを選定」

技術部会の活動内容と方針

■ 技術部会

- 性能WG (WG1): 性能向上に向けた機能の活用指針を作成
 - 性能向上を狙ったハードウェア活用、パーティショニングを観点に測定
 - バージョンアップによる差を定点観測
 - 実測結果と共に、性能ボトルネックを解決するノウハウをご紹介
- 移行WG (WG2): DB移行ガイドの作成
 - DB移行時の各プロセスのガイドの詳細化や広範囲化を中心に活動
 - PostgreSQL活用の課題である、移行費用見積のベースとなる成果物を作成
- 設計運用WG (WG3): 設計や運用面の技術情報を整備
 - 可用性をテーマにエンタープライズ領域に求められる技術情報の整備・検証の実施
 - PostgreSQLの安定運用に必要な技術ノウハウをご紹介

WG1(性能WG)の活動テーマ

大規模DBを見据えたPostgreSQLの性能検証

■ 大規模DBの特徴

使用ユーザー数が多い

データ量が多い

高性能が求められる

※1「定点観測」と呼び毎年実施

※2 2014年度に検証予定

■ 対応方法

スケールアップ

スケールアウト

スキーマ・クエリの最適化

ストレージ高速化

■ 主な検証内容

- 多コアCPU検証 ※1
- 多同時接続検証 ※1
- WAL書き込み改善 ※2

- コネクションプール (pgpool-II)
- クラスタDB(Postgres-XC, Postgres-XL ※2)

- パーティショニングの有効ケース
- 多パーティションでのオーバヘッド

- SSDを有効活用するデータ配置

PostgreSQLの性能向上に向けた機能の活用指針を作成

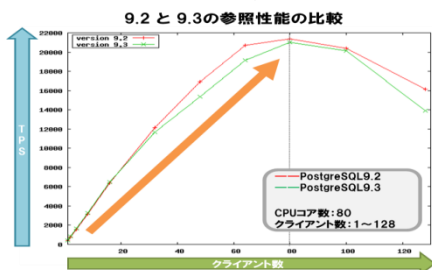
WG1 活動内容

ダウンロード可能

2013年度活動成果 ～多数のデータと詳細な解説の報告書(67p)を作成～

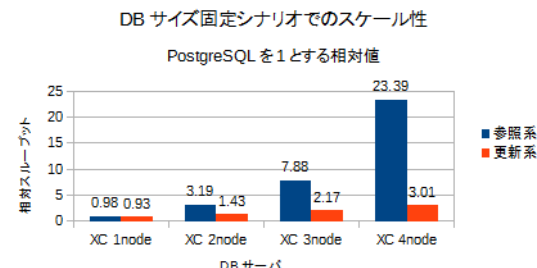
■ 定点観測

CPUコア数1～80に対して、良好なスケール性を確認。



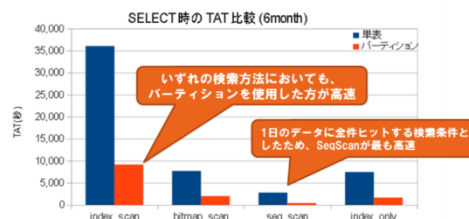
■ Postgres-XC検証

スケールアウトによる性能向上を確認。特に参照系におけるスケールが顕著。



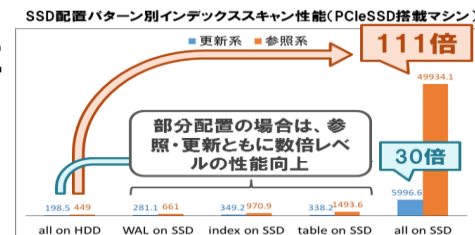
■ パーティショニング検証

約180パーティションの表の検索性能が高速であることを確認。データロードも動的関数の使用により効率化可能。



■ SSD検証

更新系処理の高速化に非常に有効であることを確認。全データをSSDに配置すると特に顕著



大規模DBを見据えた性能評価結果を公開

WG2(移行WG)の活動テーマ・活動内容

WG2の活動内容は、後程詳細に説明します。

WG3(設計・運用WG)の活動テーマ

■ 可用性

- PostgreSQLの代表的なシステム構成(シングル・HA・レプリケーション)の特徴・適用領域の整理と検証
- 災害対策を想定したシステム構成の拡張と検証(※1)

■ バックアップ

- バックアップ手法の整理と運用例、および検証

■ 監視

- 死活・性能監視に必要な情報と分析手法の整理、対処方法の検証

■ セキュリティ・監査(※1)

- 利用できる機能の整理と実用レベルのバランス調査と検証

※1 2014年度に検証予定

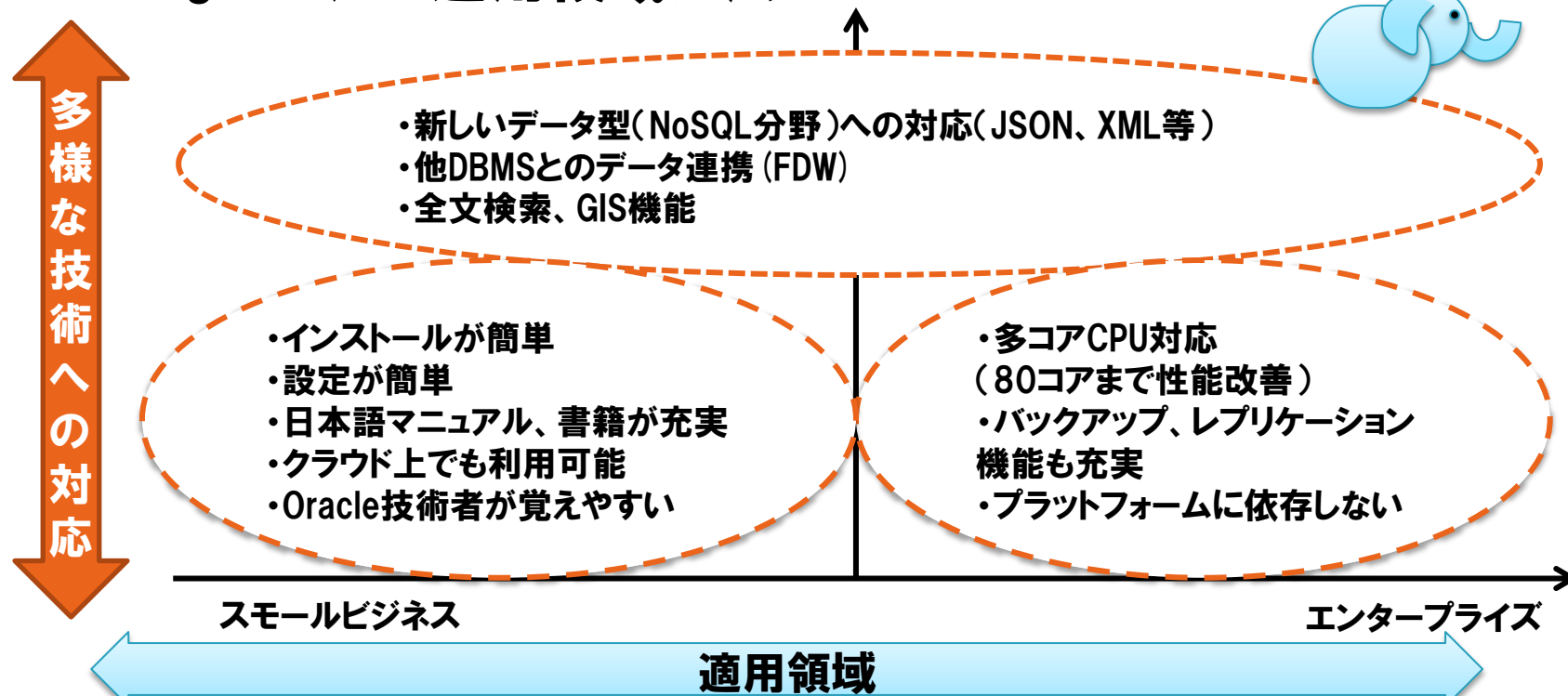
PostgreSQL運用に必要な技術ノウハウ提供を目指す

4. PostgreSQL技術紹介

WG2(移行WG)成果物のご紹介

商用DBからPostgreSQLに移行するメリット

- ベンダ・ロックインの回避
- 運用コストの削減
- PostgreSQLの適用領域が広い



適用領域でデータベースを使い分けず、データベースを
PostgreSQLに統一することが可能

4.1. 2012/2013年度の成果物のご紹介

WG2(移行WG)活動内容

活動テーマ: 異種DBMSからPostgreSQLへの移行

課題認識

- ・ 異種DBMSシステムをPostgreSQLへ移行するプロセスが確立していないことが、普及を妨げる大きな障壁と認識
 - ・ 移行作業をどのように進めればよいか分からない。
 - ・ 初期段階で移行に必要なトータルコストを算出できない。
 - ・ 過去の経験則や点在するノウハウに依存しているのが現状



活動目標

- ・ 異種DBMSからPostgreSQLへの移行を検討する際のガイドラインを提示する。(難易度判断、留意すべき事項、移行手順)



成果物

- ・ 「異種DBMSからPostgreSQLへの移行ガイド」を作成

「異種DBMSからPostgreSQLへの移行ガイド」の文章構成

■ 移行作業の全体像を解説

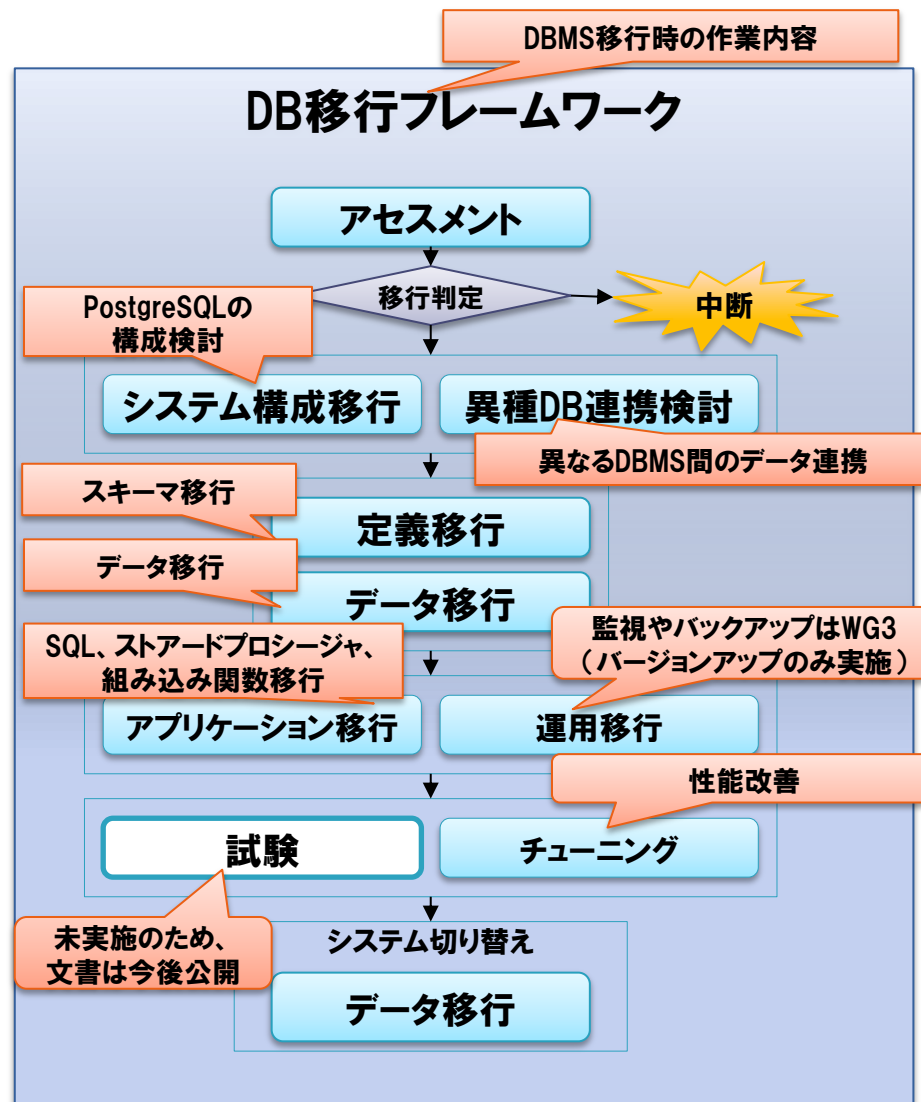
- DB移行フレームワーク編 (21ページ)

■ 移行作業に含まれる作業内容、手順の調査

- システム構成調査編 (29ページ)
- 異種DB間連携調査編 (18ページ)
- スキーマ移行調査編 (25ページ+別表)
- データ移行・文字コード変換編 (49ページ)
- ストアドプロシージャ移行調査編 (34ページ)
- アプリケーション移行調査編 (10ページ)
- SQL移行調査編 (18ページ+別表)
- 組み込み関数移行調査編 (15ページ+別表)
- チューニング編 (30ページ+別表)
- バージョンアップ編 (39ページ+別表7)

■ 移行作業を試行する検証

- データ移行調査および実践編 (60ページ+別表3)
- アプリケーション移行実践編 (25ページ+別表)



「異種DBMSからPostgreSQLへの移行ガイド」の各文章

ダウンロード可能

項番	作成文書（作成年度）	記載概要
1	DB移行フレームワーク編（2012年）	移行作業全体の解説
2	システム構成調査編（2012年）	DBMSの一般的なシステム構成とPostgreSQLの構成
3	異種DB間連携調査編（2012年）	異種DBMSとPostgreSQLのデータ連携方法
4	スキーマ移行調査編（2012年）	異種DBMSとPostgreSQLのスキーマの違いと書き換え方針
5	SQL移行調査編（2012年）	異種DBMSとPostgreSQLのSQL互換性調査結果と書き換え方針
6	ストアドプロシージャ移行調査編（2012、2013年）	異種DBMSのストアドプロシージャをPostgreSQLに移行する際の手書き換え方針
7	組み込み関数移行調査編（2012年）	Oracle, PostgreSQLの組み込み関数互換性調査結果と書き換え方針
8	チューニング（2013年）	PostgreSQLのチューニング手法を記載
9	バージョンアップ編（2013年）	PostgreSQLのバージョンアップ手法と検証結果
10	データ移行調査および実践編（2012年） データ移行・文字コード変換編（2013年） ※1	異種DBMSからPostgreSQLへのデータ移行時の注意点と実施結果
11	アプリケーション移行調査および実践編（2012年）	APIやトランザクションの差異と対処方法。および異種DBMSからPostgreSQLに移行した際のアプリケーションの移行結果

■ 赤字部分を本日紹介

※1「データ移行調査および実践編」としてまとめて紹介

スキーマ移行調査編（1/2）

ダウンロード可能

■ 異種DBMSとPostgreSQLのスキーマの違いは？

- Oracle, PostgreSQLのDDL互換性を調査
- 各種DDLの相違点をもとにした移行方針を解説
- 組み込みデータ型対応表（一部抜粋）

属性	Oracle			PostgreSQL				備考
	データ型	単位	備考	データ型	サイズ	単位	備考	
文字	VARCHAR2	byte	4000 バイト	varchar(n)			上限付き可変長	PostgreSQLは文字数指定だが、Oracleはバイト数指定、文字数指定をセマンティクスで区別するので注意が必要（デフォルトはバイト数指定）
		char	4000 バイト	varchar(n)			上限付き可変長	
	NVARCHAR2	char	4000 バイト	varchar(n)			上限付き可変長	8iはDBのキャラクターセットでバイト数が文字数が異なる。（9iは文字数指定）
	CHAR	byte	2000 バイト	char(n)			空白で埋められた固定長	PostgreSQLは文字数指定だが、Oracleはバイト数指定、文字数指定をセマンティクスで区別するので注意が必要（デフォルトはバイト数指定）
		char	2000 バイト	char(n)			空白で埋められた固定長	
	NCHAR	char	2000 バイト	char(n)			空白で埋められた固定長	8iはDBのキャラクターセットでバイト数が文字数が異なる。（9iは文字数指定）
	LONG（下位互換）		2G-1 バイト	text			可変長(最大1GB)	1GBを超える場合は、ラージオブジェクトを使用するか、外部ファイルに保存する。
	CLOB		4G-1 バイト	text			可変長(最大1GB)	
	NCLOB		4G-1 バイト	text			可変長(最大1GB)	

スキーマ移行調査編（2/2）

ダウンロード可能

■ Ora2Pg ※1を使い、Oracleのスキーマを移行する際に、手作業で修正必要な部分を解説

分類	Oracle型	PostgreSQL対応型	単位	Ora2Pgによる変換例 (Oracle→変換後)	移行時の修正方法
	データ型	データ型			
文字 (固定長)	NCHAR (n)	char (n)	n:文字数	NCHAR (10) → char ※2 charはchar (1) と同等	Oracleを確認して、文字数を指定する。 例:char→char (10) に書き換える
真数	NUMBER (n,m)	numeric (n,m)	n:桁数 m:小数点以下の桁数	NUMBER (5) → numeric (5) NUMBER (10) → numeric (10)	精度とスケールに応じてPostgreSQLのデータ型を指定する。 ※3 numeric型は、非常に大きな桁数で数値を格納でき、正確な計算を行える。 ただし PostgreSQLのnumeric は算術演算パフォーマンスが悪いため、大きな桁数が必要ない場合は、integer, bigint型を使用する。 例:numeric (5) →intger 例:numeric (10) →bigint
		decimal (n,m)	n:桁数 m:小数点以下の桁数		
		integer			
		bigint			
日時	TIMESTAMP	timestamp [(p)] [without time zone]	p:ミリ秒小数点以下の桁数	TIMESTAMP (6) → timestamp TIMESTAMP (6) WITH TIME ZONE→ timestamp	TIMESTAMP型のミリ秒以下の桁数を指定する。タイムゾーンを持つ場合は、データ型にタイムゾーンを指定する。 例: timestamp → timestamp (6) 例: timestamp → timestamp (6) with time zone
	TIMESTAMP WITH TIMEZONE	timestamp [(p)] with time zone	p:ミリ小秒小数点以下の桁数		

※1 Oracleのスキーマ、データをPostgreSQLに投入可能な形式に変換するツール

SQL移行調査編（1/2）

ダウンロード可能

■ 異種DBMSのSQLはPostgreSQLでも動くのか？

- Oracle, SQL Server, PostgreSQLのSQL互換性を調査
- 互換性がないSQLの書き換え方針を調査
- SQL差異表（一部抜粋）

SQL機能	標準SQL	説明	PostgreSQL 9.2.1		Oracle 11g R2		SQL Server 2008 R2	
			対応	備考	対応	備考	対応	備考
SELECT								
WITH句	○	WITH 問い合わせ（共通テーブル式）	○	WITH 句の中で更新系コマンドが使用可能	○		○	列の別名に独自の構文あり
DISTINCT	○	重複している行を取り除く	○	DISTINCT ON は PostgreSQL 特有	○		○	
UNIQUE	×	重複している行を取り除く	×		○	Oracle 特有	×	
TOP句	×	取得する行数の指定	×		×		○	SQL Server 特有
FROM句	○	テーブルの指定	○	FROM 句を省略可能	○	FROM 句は省略不可 サブクエリの別名を省略可	○	FROM 句を省略可能
CONNECT BY / START WITH 句	×	階層問い合わせ	×		○	Oracle 特有	×	
JOIN 句	○	テーブルの結合	○		○	独自の結合演算子(+)あり	○	
WINDOW句	○	OVER 句で参照するウィンドウ	○		×		×	
LIMIT 句	×	取得する行数などの指定	○		×		×	
FETCH 句	○	取得する行数などの指定	○		×		×	
SELECT FOR UPDATE	○	更新ロックを取得する （標準 SQL ではカーソルのオプションとしてのみ有効）	○	任意の SELECT で使用可能	○	サブクエリでは使用不可	○	カーソルでのみ使用可能
SELECT FOR SHARE	×	共有ロックを取得する	○	PostgreSQL 特有	×		×	
更新系								
INSERT		行の挿入		RETURNING句を使用可能				TOP句を使用可能

SQL移行調査編（2/2）

ダウンロード可能

■ 代表的な書き換え例

- SELECT結果の行数を制限するSQLはDBMS毎に異なる。

Oracle	SQL Server	PostgreSQL
SELECT * FROM (SELECT * FROM tbl ORDER BY id) WHERE ROWNUM <= 10	SELECT TOP 10 * FROM tbl ORDER BY id	SELECT * FROM tbl ORDER BY id LIMIT 10 OFFSET 0

※PostgreSQL(8.4以降)は標準SQLの構文に対応

SELECT * FROM tbl **OFFSET 0 ROW FETCH FIRST 10 ROWS ONLY**

□ MERGE

- PostgreSQLでは未実装。WITH句を使った書き換え方法を紹介

□ CONNECT BY

- Oracle独自の階層問い合わせ
- 再帰SQL(標準SQL準拠)を使った書き換え方法を紹介

組み込み関数移行調査編（1/2）

ダウンロード可能

■ 異種DBMSの関数はPostgreSQLでも動くのか？

- Oracle, PostgreSQLの組み込み関数互換性を調査
- 組み込み関数対応表（一部抜粋）

Oracle関数	説明	PostgreSQL関数	対応可否
日時ファンクション			
ADD_MONTHS (date, integer)	日付dateに月数integerを加えて戻す。	+ 演算子を使って書き換え可能 例: select date '2013-03-22' + interval '1 months'	○
CURRENT_DATE	セッション・タイムゾーンの現在の日付を戻す。	current_date, current_timestamp	○
CURRENT_TIMESTAMP	セッション・タイムゾーンの現在の日付および時刻をTIMESTAMP WITH TIME ZONEデータ型の値で戻す。	current_timestamp	○
DBTIMEZONE	データベースのタイムゾーンの値を戻す。		×
EXTRACT (element FROM date)	日時式または期間式から指定された日時フィールドの値を抽出して戻す。	extract(field from timestamp)	○
FROM_TZ(timestamp, time_zone_value)	タイムスタンプ値およびタイムゾーンをTIMESTAMP WITH TIME ZONE値に変換する。		×
LAST_DAY(d)	d を含む月の最後の日付を戻す。	last_date(date)	○
LOCALTIMESTAMP	セッション・タイムゾーンの現在の日付および時刻をTIMESTAMPデータ型の値で戻す。	localtimestamp	○
MONTHS_BETWEEN(d1, d2)	d1 と d2 の間の月数を戻す。	months_between(d1, d2)	○
NEW_TIME(d, z1, z2)	時間帯z1の日時がdの時点の時間帯z2の日時を戻す。		×
NEXT_DAY(d, char)	charで指定した曜日で日付d以降の最初の日付を戻す。	next_day(date, text)	○
NUMTODSINTERVAL(n, 'char_expr')	n をINTERVAL DAY TO SECOND リテラルに変換する。		×
NUMTOYMINTERVAL(n, 'char_expr')	n をINTERVAL YEAR TO MONTH リテラルに変換する。		×
ROUND(d, fmt)	d を書式 fmt で指定した単位に丸めた結果を戻す。	round(date, text)	○
SESSIONTIMEZONE	セッションのタイムゾーンの値を戻す。		×

組み込み関数移行調査編（2/2）

ダウンロード可能

■ 代表的な書き換え例

□ ADD_MONTHS : 月を演算する関数

```
SQL> SELECT ADD_MONTHS('2013/3/22',1) FROM DUAL;  
ADD_MONT  
-----  
13-04-22
```

□ PostgreSQL に ADD_MONTHS 関数は存在しない。

□ 算術演算子を使った書き換え

```
postgres=# SELECT date '2013-03-22' + interval '1 months';  
?column?  
-----  
2013-04-22 00:00:00  
(1 行)
```

□ Oracle互換関数ライブラリ(Orafce)を利用する方法も

バージョンアップ (1/3)

ダウンロード可能

■ PostgreSQL移行後のバージョンアップはどのように実施すべきか？

□ 各バージョンアップ手法のメリット・デメリットを整理

※ バージョンアップ元:既存PostgreSQL、バージョンアップ先:新規PostgreSQL

No.	手法(使用ツール)	説明	対応バージョンアップ
1	データベースクラスタの継続利用	データベースクラスタをそのまま残し、PostgreSQLのバイナリのみ新規バージョンに置き換える。	マイナー
2	バックアップ・リストア (pg_dump/pg_restore)	既存PostgreSQLからpg_dumpで取得したバックアップを、psqlまたはpg_restoreを用いて新規PostgreSQLにリストアする。	マイナー/メジャー
3	バージョンアップツールの利用 (pg_upgrade/コピーモード)	既存データベースクラスタのデータを、新規PostgreSQLのデータベースクラスタにコピーするモード	マイナー/メジャー
4	バージョンアップツールの利用 (pg_upgrade/リンクモード)	既存データベースクラスタと新規データベースクラスタをハードリンクで繋ぎ、データを共有するモード	マイナー/メジャー
5	レプリケーション (Slony-I)	異なるメジャーバージョン間でも、PostgreSQLのレプリケーション構成を実現できるSlony-Iを用いて、バージョンアップを行う。	マイナー/メジャー
6	ベースバックアップとアーカイブログ (pg_receivexlog、 pg_basebackup、pg_upgrade)	以下のように各手法を組み合わせ、移行日に大規模なデータコピーが発生することを回避した手法。(ベースバックアップを利用して、バージョンアップの事前試行も可能) 事前:ベースバックアップ(pg_basebackup) 都度:アーカイブログ転送(pg_receivexlog) 移行日:PITR、バージョンアップ(pg_upgrade)	マイナー/メジャー

バージョンアップ (2/3)

ダウンロード可能

■ 各バージョンアップ手法の実機検証の結果

No.	手法(使用ツール)	実施バージョンアップ	サービスの停止	サービス停止時間	手順の難易度	異なるサーバ間のバージョンアップ	推奨
1	データベースクラスタの継続利用	マイナー	必要	9秒	易	不可	○
2	バックアップ・リストア (pg_dump/pg_restore)	メジャー	必要	289秒	易	可能	○
3	バージョンアップツールの利用 (pg_upgrade/コピーモード)	メジャー	必要	40秒	易	不可	○
4	バージョンアップツールの利用 (pg_upgrade/リンクモード)	メジャー	必要	19秒	易	不可	○
5	レプリケーション(Slony-I)	メジャー	必要	48秒(※)	難	可能	△
6	ベースバックアップとアーカイブログ (pg_receivexlog、 pg_basebackup、 pg_upgrade)	メジャー	必要	28秒	並	可能	○

- ※ 主キーがないテーブルは、Slony-Iではレプリケーションできない。
本検証では、主キーが無いテーブルには、pg_dump/pg_restoreを利用。
- バージョンアップ時のデータを生成には、JdbcRunnerの付属テーブル構造の1つである「Tiny TPC-C」を利用
- JdbcRunner : <http://hp.vector.co.jp/authors/VA052413/jdbcrunner/>

バージョンアップ (3/3)

ダウンロード可能

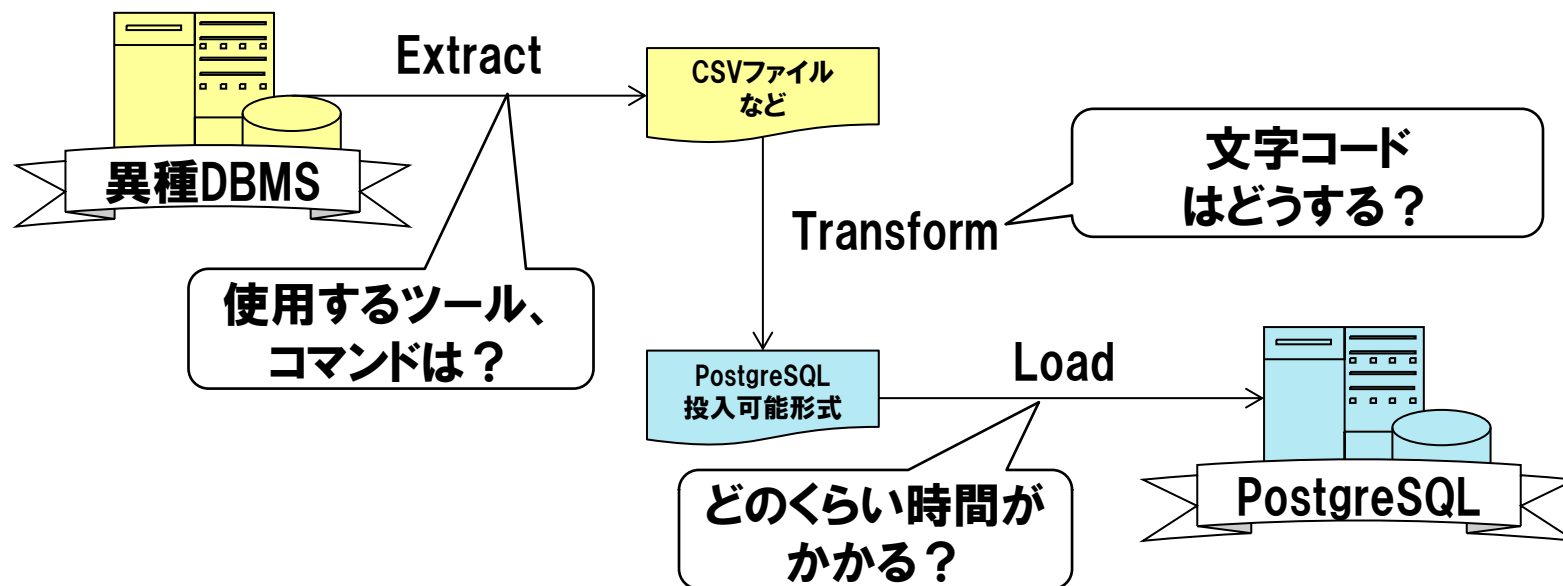
■ 各バージョンアップ手法のメリット、デメリット

項番	手法(使用ツール)	メリット	デメリット・注意点
1	データベースクラスタの継続利用	・バイナリの追加インストールだけでマイナーバージョンアップできる。 ・新規PostgreSQLのための領域を確保しなくて良い。	・メジャーバージョンアップでは利用できない。 ・PostgreSQLの停止を伴う
2	バックアップ・リストア (pg_dump/pg_restore)	一般的な(枯れた)手法であるため、参考資料が豊富であり、難易度も高くない。	・バックアップ・リストア中にデータ差分が発生した場合に再度バックアップとリストアを実行する必要がある。 ・PostgreSQLの停止を伴う。(バックアップ+リストア時間) ・データ量が増えると、実行時間も長くなる。
3	バージョンアップツールの利用 (pg_upgrade/コピーモード)	・「バックアップ・リストア(pg_dump/pg_restore)」と比較すると、高速にバージョンアップが可能。 ・バージョンアップ時の変換等、挙動について注意点が明記されている。	・PostgreSQLの停止を伴う。(バージョンアップ時間) ・サーバ間のデータ移行ができない
4	バージョンアップツールの利用 (pg_upgrade/リンクモード)	「バージョンアップツールの利用(pg_upgrade/コピーモード)」よりも、高速にバージョンアップが可能。	・基本的に旧バージョンでのデータベースクラスタはできない。PostgreSQLの停止を伴う。(バージョンアップ時間) ・サーバ間のデータ移行ができない
5	レプリケーション (Slony-I)	稼働中のPostgreSQLのデータを移行できる (負荷はかかるため試験は必須)。	・レプリケーションソフトウェアであるため、バージョンアップ向けのドキュメントが整備されているとは言い難い。 ・実施手順が他の手法に比べて複雑。 ・レプリケーションできないデータが存在する (ラージオブジェクトや主キーがないテーブルのデータ等)
6	ベースバックアップとアーカイブログ (pg_receivexlog, pg_basebackup, pg_upgrade)	・ダウンタイムを削減することが可能。 ・試験的にバージョンアップを行うことが行えるため、バージョンアップ中のトラブルを未然に防ぐことが可能。	アーカイブログが蓄積(物理バックアップ時点からの差分)されすぎると、その適用が長時間になる可能性がある。

データ移行調査および実践編（1/4）

ダウンロード可能

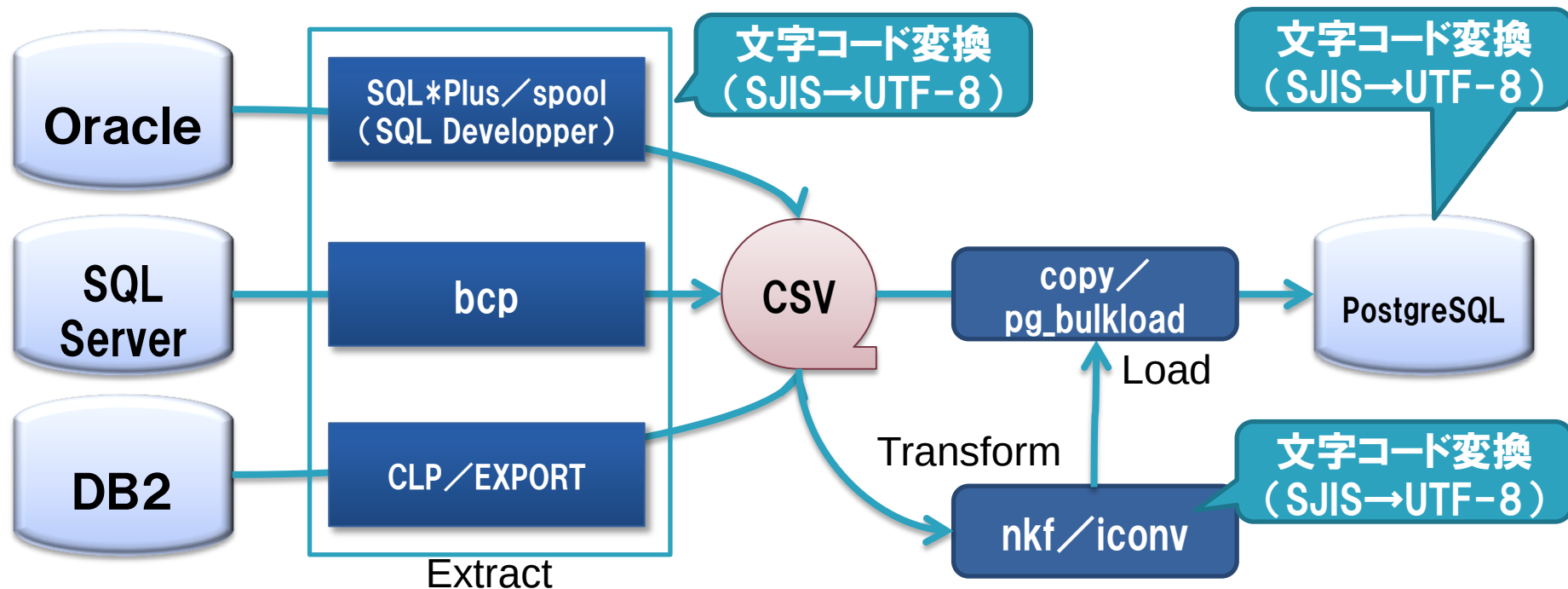
- 異種DBMSのデータをPostgreSQLに持ってくるにはどうすればいい？
 - 異種DBMSからPostgreSQLへデータ移行を行う場合に発生する作業や注意点について調査
 - 実際にデータ移行を試行し、調査結果の妥当性を検証



データ移行調査および実践編（2/4）

ダウンロード可能

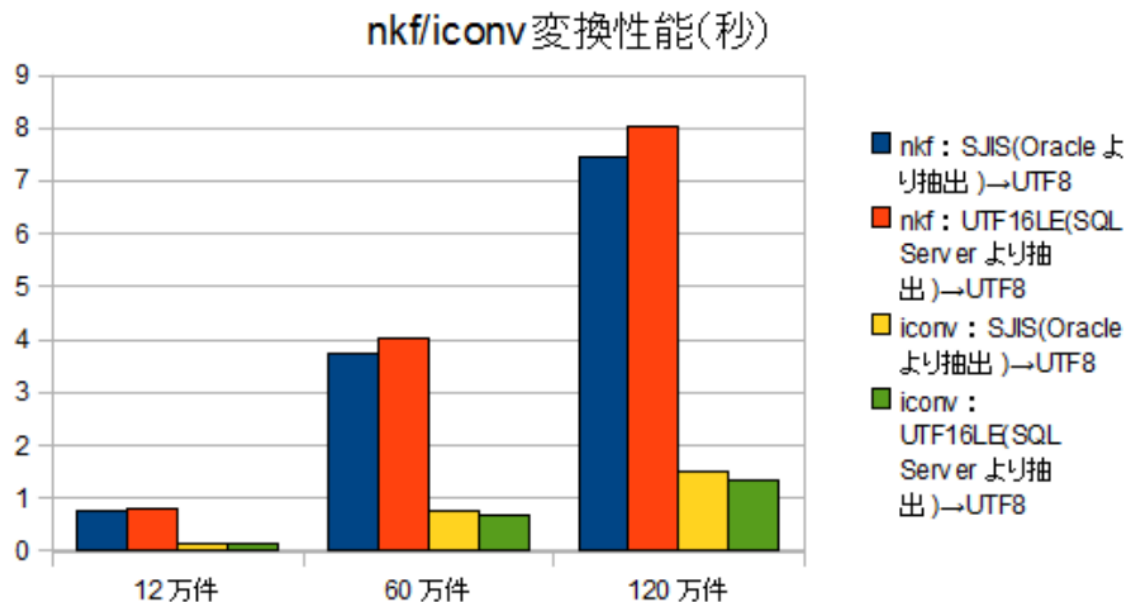
- 郵便番号のデータを利用(約12万件)
- Oracle、SQL Server、DB2からデータを抽出しPostgreSQLに投入
- 各DBMSのデータをファイル(CSV)に抽出
- コード変換のタイミング別検証
(データ抽出時／出力ファイル／データ投入時)



データ移行調査および実践編（3/4）

ダウンロード可能

検証項目	検証結果
データの抽出／投入	- DBMS、ツール、文字コード変換のタイミングに関わらず、データ移行に成功 - 出力時にデータを””でデータを囲うなど、ツールにより出力形式が異なる結果となった → 今回は、形式の違いがデータ投入の成否に影響しなかった - データ抽出時の文字コードはDBMSによる差を確認 - Oracle(SQL*Plus)、DB2(CLP/EXPORT)はUTF-8出力 - SQL Server(bcp)はUTF-16出力
ファイルの文字コード変換	nkfとiconvには大きな変換性能差



データ移行調査および実践編（4/4）

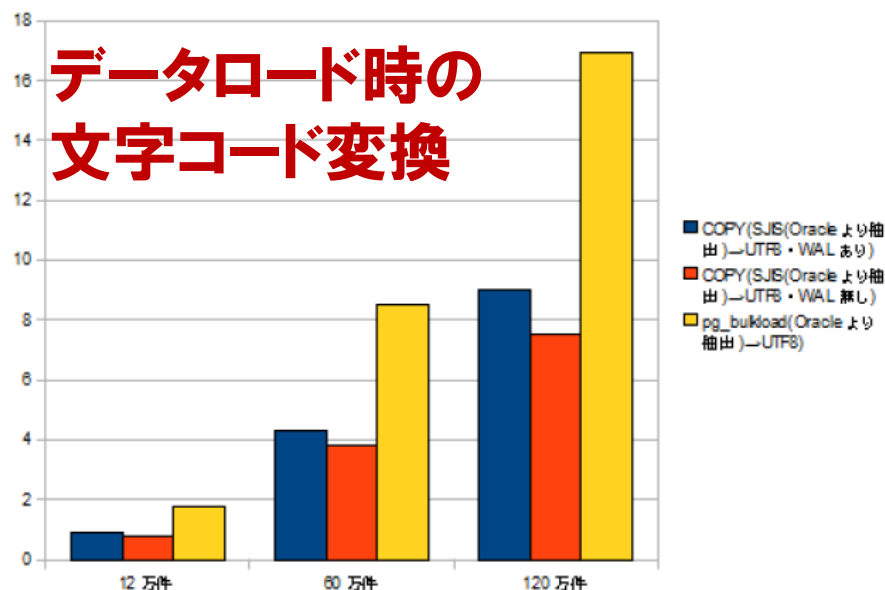
ダウンロード可能

■ 検証結果

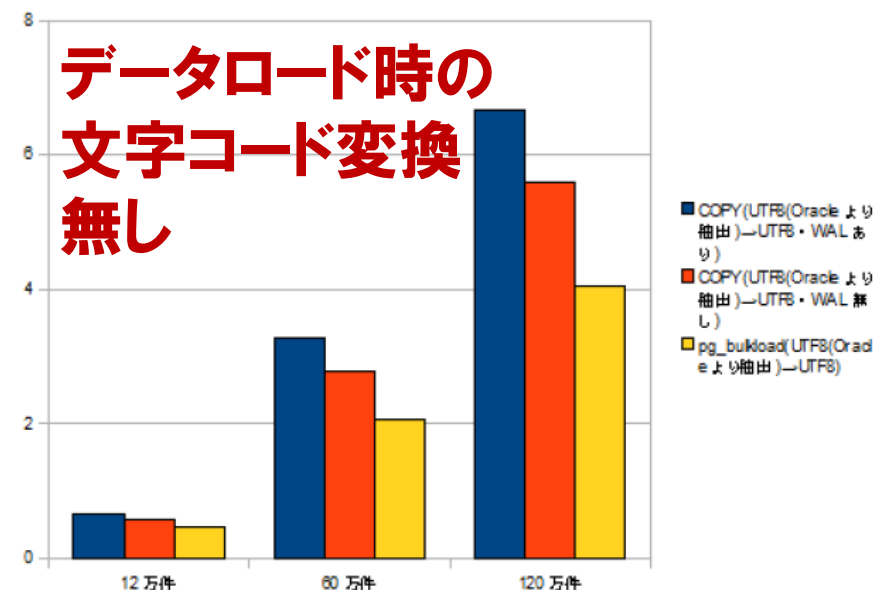
□ 文字コード変換によるロード性能への影響

- COPY(WAL出力)
- COPY(WAL出力抑止)
- pg_bulkload

データ投入時に文字コード変換した場合の投入処理時間（インデックス定義なし）



事前に文字コード変換した場合のデータ投入処理時間（インデックス定義なし）



アプリケーション移行実践編（1/5）

ダウンロード可能

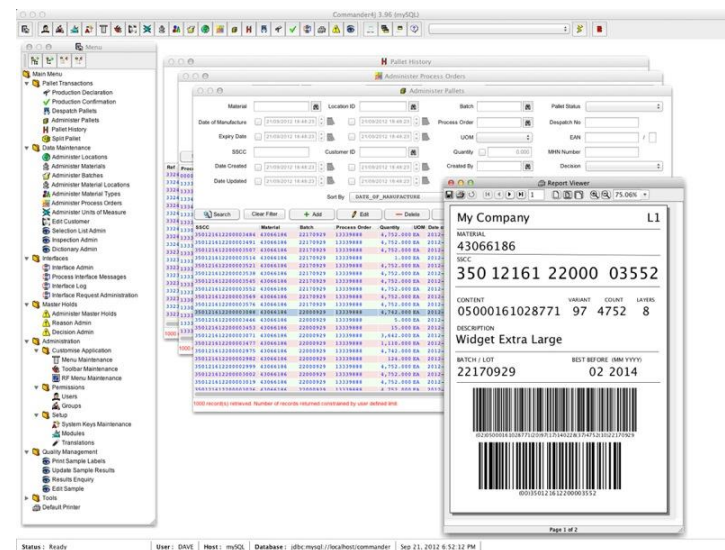
■ 実際に異種DBMSからPostgreSQLへアプリケーション移行を行うと、どういう問題が発生するか？

- 異種DBMSで稼動するアプリケーションをPostgreSQL(9.2.2)上に移行して評価を行った。

■ 移行対象アプリケーション

□ Commander4J

- バーコードラベルを作成するJavaのOSSアプリケーション
- 対応DB: Oracle, SQLServer, MySQL
- ステップ数: 71Ks
- テーブル数: 39
- SQL数: 3390
- ストアドプロシージャ: なし



アプリケーション移行実践編（2/5）

ダウンロード可能

- SQL変更箇所の抽出に db_syntax_diff を使用
- db_syntax_diff
 - Oracle DatabaseからPostgreSQLへアプリケーションを移行する際に修正が必要となる箇所を報告するツール
 - NTT OSSセンタで開発されたOSS
 - ライセンスは The PostgreSQL License
 - <https://github.com/db-syntax-diff> より取得可能
 - アプリケーションのソースコードを指定して実行すると、修正が必要なSQLのパターンとマッチングして報告。
 - 修正が必要なSQLのパターンはXML形式で定義するため、必要に応じてチェックパターンのカスタマイズが可能

アプリケーション移行実践編（3/5）

ダウンロード可能

■ db_syntax_diff の出力例（一部抜粋）

/home/postgres/b4/src/com/commander4j/db/JDBTable.java	83	7	SQL	WARNING	user_idのデータ・ディクショナリ・ビューまたは動的パフォーマンス・ビュー
/home/postgres/b4/src/com/commander4j/db/JDBTable.java	83	15	SQL	WARNING	user_commentのデータ・ディクショナリ・ビューまたは動的パフォーマンス
/home/postgres/b4/src/com/commander4j/db/JDBDespatch.java	428	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/db/JDBQuery.java	67	13	SQL	ERROR LV1	全列指定は未サポートです。SELECT句にすべてのカラム名を指定してく
/home/postgres/b4/src/com/commander4j/db/JDBQuery.java	77	23	SQL	LOW1	ROWNUM擬似列は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameLoca	239	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameLoca	248	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameLoca	297	0	FUNC	WARNING	このまま移行する場合、引数のデータ型の確認が必要です。なぜなら、C
/home/postgres/b4/src/com/commander4j/app/JInternalFrameQMIn	449	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameQMIn	466	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameQMIn	485	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameProc	238	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameMHN	252	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameMHN	261	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameMHN	314	0	FUNC	WARNING	このまま移行する場合、引数のデータ型の確認が必要です。なぜなら、C
/home/postgres/b4/src/com/commander4j/app/JInternalFrameMHN	1115	7	SQL	LOW2	FROMの省略は未サポートです。
/home/postgres/b4/src/com/commander4j/app/JInternalFrameQMF	78	82	SQL	WARNING	User_Data_1のデータ・ディクショナリ・ビューまたは動的パフォーマンス・ビ
/home/postgres/b4/src/com/commander4j/app/JInternalFrameQMF	78	95	SQL	WARNING	User_Data_2のデータ・ディクショナリ・ビューまたは動的パフォーマンス・ビ
/home/postgres/b4/src/com/commander4j/app/JInternalFrameQMF	166	80	SQL	WARNING	USER_DATA_1のデータ・ディクショナリ・ビューまたは動的パフォーマンス
/home/postgres/b4/src/com/commander4j/app/JInternalFrameQMF	166	92	SQL	WARNING	USER_DATA_2のデータ・ディクショナリ・ビューまたは動的パフォーマンス
/home/postgres/b4/src/com/commander4j/app/JInternalFrameMate	230	7	SQL	LOW2	FROMの省略は未サポートです。

アプリケーション移行実践編（4/5）

ダウンロード可能

■ 移行実施の手順

- db_syntax_diffをJavaソースコードと環境構築用のSQLファイル(スキーマ定義DDL、データ投入DML)に対して実行
- db_syntax_diffの実行結果をもとに、SQL文を修正
- PostgreSQL用JDBCドライバの追加
- DBMS別分岐処理への PostgreSQL 用処理の追加
 - DBMS別に分岐している処理にPostgreSQL用JDBCドライバでの分岐と処理を追加

ソースの修正例(**赤字が追加箇所**)

```
if (hst.getDatabaseParameters().getJdbcDriver().equals("com.mysql.jdbc.Driver")) {  
    jComboBoxJdbcDriver.setSelectedIndex(1);  
}  
if (hst.getDatabaseParameters().getJdbcDriver().equals("oracle.jdbc.driver.OracleDriver")) {  
    jComboBoxJdbcDriver.setSelectedIndex(2);  
}  
[省略]  
//PG add ->  
if (hst.getDatabaseParameters().getJdbcDriver().equals("org.postgresql.Driver")) {  
    jComboBoxJdbcDriver.setSelectedIndex(5);  
} //PG add end
```

- 修正後アプリケーションの動作確認
 - Commander4j の基本機能が正常動作するか？

アプリケーション移行実践編（5/5）

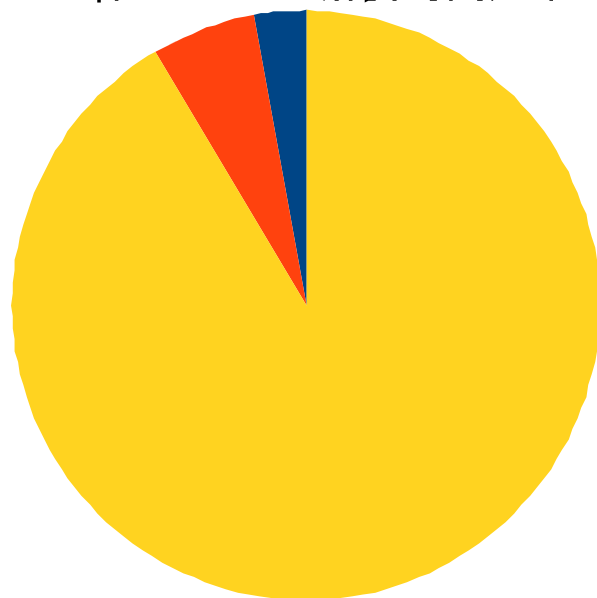
ダウンロード可能

■ 移行実施結果

□ 試験工程が全工程の90%以上を占めた

- ツールでSQL修正箇所を自動検出し、修正作業が短く済んだ。
- 修正有無に関わらず、関連するSQLはすべて試験を実施した。
- db_syntax_diff で検出できなかった箇所は試験工程で対応した。

各フェーズの所要時間比率



項番	作業項目	所要時間比率
#1	AP修正(PostgreSQLへの接続)	2.8%
#2	AP修正(db_syntax_diffに基づく修正)	5.8%
#3	試験およびエラー修正	91.4%

- 移行作業項番#1
- 移行作業項番#2
- 移行作業項番#3

4.2. 2014年度の活動内容のご紹介

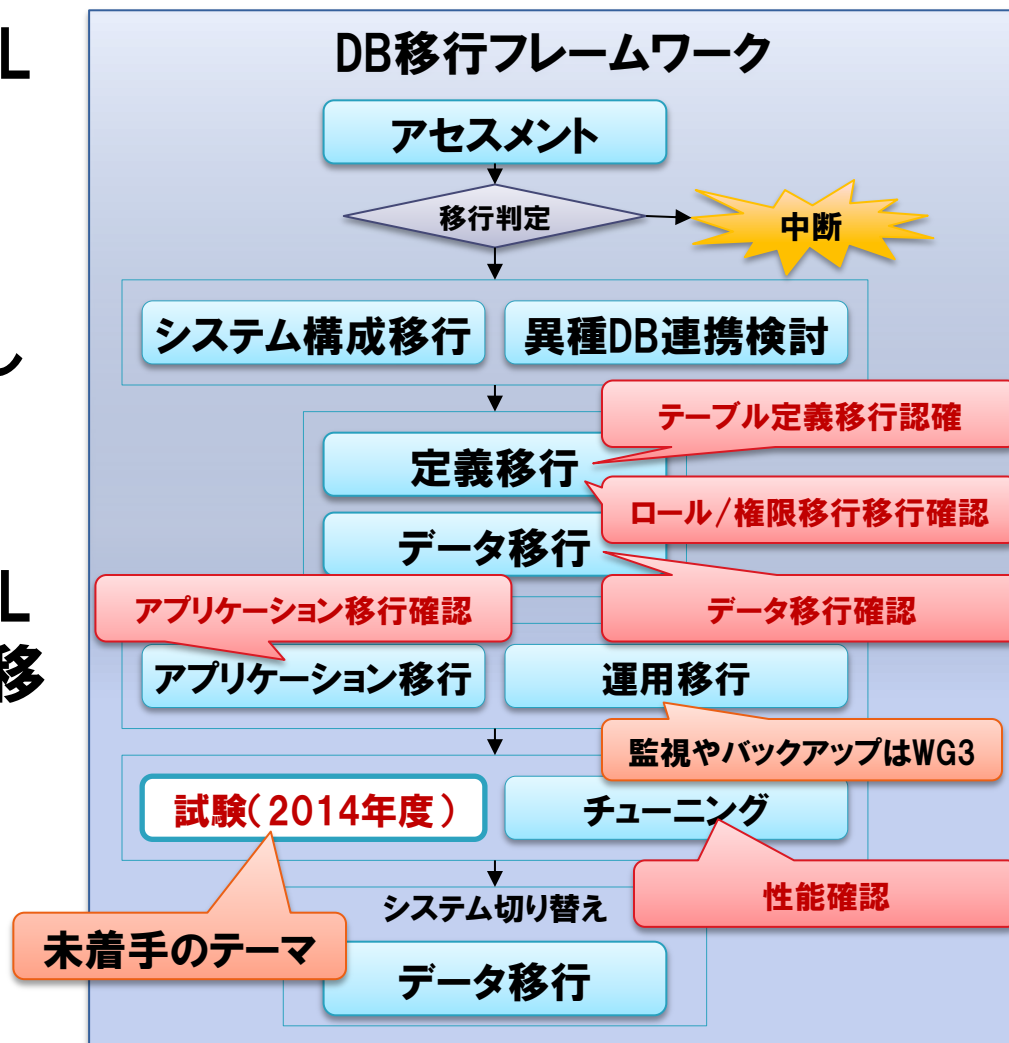
2013年度終了時の課題と2014年度の活動テーマ

■ 「異種DBMSからPostgreSQLへの移行ガイド」で定義した「試験」が未着手

- DB移行後の試験を標準化し、PostgreSQLへの移行を推進したい

■ 「異種DBMSからPostgreSQLへの移行ガイド」を利用した移行事例を提示したい

- 各文章の有用性を確認
- 不足があれば、フィードバック



移行結果確認試験概要

- 現時点で検討している試験は下表の通りです。

No.	試験項目	概要
1	テーブル定義 移行	DDLの移行結果についての試験を行う。 対象DDLは、主にDB初期化時に使用するDDLとする。 ただし、アプリケーションで任意のテーブル作成等が可能である場合は、考慮する。 また、インデックスやパーティショニング等について、PostgreSQLへ移行時に性能問題の発生が懸念される箇所については抽出した後、対処方法の検討を行う。
2	アプリケーション 移行	アプリケーションは下記を想定している。 SQL、ストアドプロシージャ、ドライバ定義、プールライブラリ定義、ORMマッパー (Hibernate) 定義 ※ アプリケーションにレグレッションテストが同梱されている場合は利用を検討する。
3	データ移行	データ移行結果についての試験を行う。 観点としては、件数および、データ内容比較を行う。
4	性能	PostgreSQLへ移行後、性能劣化が無いことを確認する。
5	ロール/権限移 行	ロール/権限が適切に移行されているか確認する。 併せて、OracleとPostgreSQLの権限対応表を可能な限り整備する。

※赤字部分を本日紹介

データ移行確認項目

■ 下表の試験を検討しています。

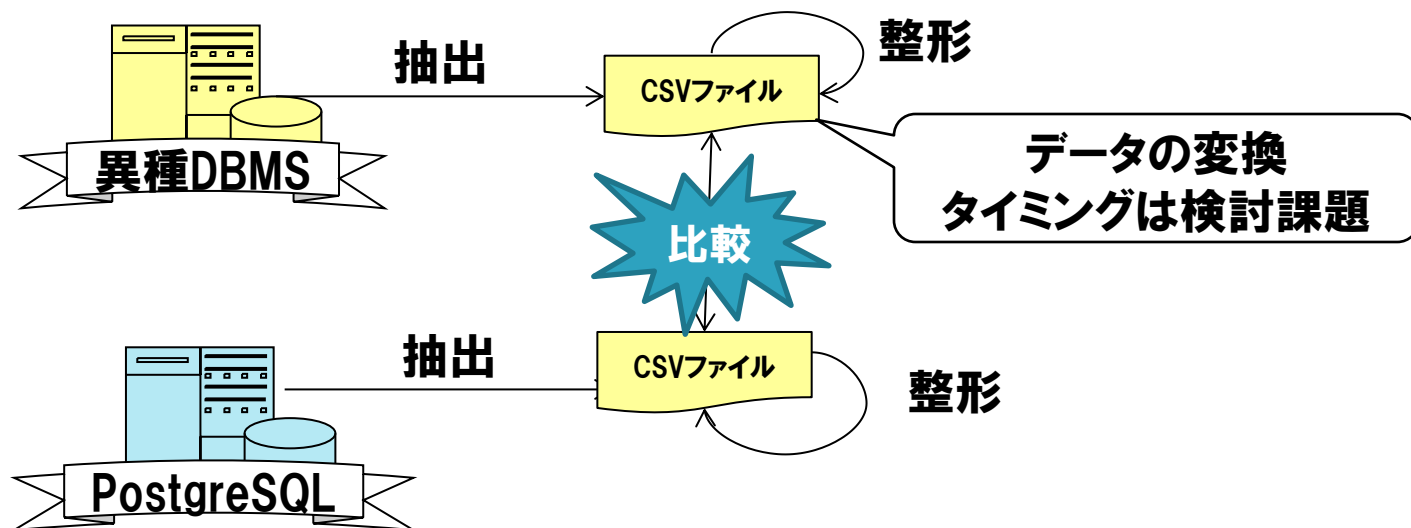
項番	カテゴリ	小項番	作業内容
1	データ件数	データファイル数 (テーブル数)	移行元から出力したCSVファイルと、PostgreSQLから出力したCSVファイル数が同一であることを確認する。
		データ件数	移行元に格納されているテーブル毎のデータ件数と、PostgreSQLに格納されているテーブル毎のデータ件数が同一であることを確認する。
2	データサイズ	データサイズ	想定値と近似値かを確認する。
3	データ内容	データ比較	移行元から出力したCSVファイルと、PostgreSQLから出力したCSVファイルの内容が同一であることを確認する。 CSV出力機能の差異などで差分が発生した場合でも、目視または事前定義した例外項目で問題なしと判断できれば試験は合格とする。
		外字	外字が正しく出力されるか確認する。

■ データ移行確認は、可能な限り自動化を検討

データ移行確認時の問題点

■ 異種DBMSとPostgreSQLから抽出したデータを比較する際の問題点は？

- データ順番（データ抽出時にORDER BYなどを指定するか、出力したCSVファイルをOSコマンド等でソート）
- 日付データの差異（PostgreSQLは、日付データのミリ秒末尾にゼロが補完されないため、異種DBMSのデータ末尾のゼロを削除）
- NULLの文字コード差異（NULLの代替文字に置換） 等



2014年度活動成果予定

■ 前述した試験項目の詳細をご紹介します

- 各試験項目の実施手法や詳細な手順

■ 「異種DBMSからPostgreSQLへの移行ガイド」を利用した移行事例の提示

- 移行対象のアプリケーションとして、「infoScoop」を選択
 - infoScoopは企業情報ポータル(EIP)として動作するOSS
 - Oracle 11g → PostgreSQL 9.4への移行
- infoScoop移行時に発生した課題と解決方法を提示
 - 移行ツール(Ora2Pgやdb_syntax_diff等)の利用結果やアプリケーションの修正箇所等

PGEConsと一緒に活動しませんか？

5. さいごに

成果物の公開

ダウンロード可能

- PGEConsサイトから、無料でダウンロードいただけます
- <http://www.pgecons.org/>



PostgreSQL エンタープライズ・コンソーシアム 技術部会 WG#1
性能ワーキンググループ

2013 年度 WG1 活動報告
大規模 DB を見据えた PostgreSQL の性能検証



PostgreSQL エンタープライズ・コンソーシアム 技術部会 WG#2

異種 DBMS から PostgreSQL への移行ガイド

製作者
関係企業名

© 2014 PostgreSQL Enterprise Consortium



PostgreSQL エンタープライズ・コンソーシアム 技術部会 WG3
設計運用ワーキンググループ (WG3)

2013 年度 WG3 活動報告書
一走り続ける PostgreSQL システム構築のためにー

© 2014 PostgreSQL Enterprise Consortium

2014年度活動スケジュール



会員募集

■ 正会員・一般会員を広く募集いたします

□ WGと一緒に活動を行っていただける団体様
⇒ 正会員

□ PostgreSQLのエンタープライズ領域に興味を持っている団体様 ⇒ 一般会員

お問い合わせ先:

PostgreSQLエンタープライズ・コンソーシアム事務局

メール : jimukyoku@pgecons.org

Web : <http://www.pgecons.org/>

ご清聴ありがとうございました！