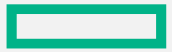


Hewlett Packard
Enterprise

Babelfish の話

Noriyoshi Shinoda

February 22, 2022

SPEAKER
篠田典良（しのだのりよし）

- 所属
  - 日本ヒューレット・パッカード合同会社
- 現在の業務
  - PostgreSQL をはじめ、Oracle Database, Microsoft SQL Server, Vertica 等 RDBMS 全般に関するシステムの設計、移行、チューニング、コンサルティング
  - Oracle ACE（2009 年 4 月～）
  - オープンソース製品に関する調査、検証
- 関連する URL
  - 「PostgreSQL 虎の巻」シリーズ
    http://h30507.www3.hp.com/t5/user/viewprofilepage/user-id/838802
  - 「NewSQL」はPostgreSQLやMySQLを代替するのか？
    https://atmarkit.itmedia.co.jp/ait/articles/2112/07/news008.html
  - Oracle ACE ってどんな人？
    http://www.oracle.com/technetwork/jp/database/articles/vivadeveloper/index-1838335-ja.html

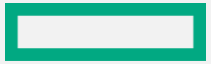
SPEAKER
篠田典良（しのだのりよし）

- PostgreSQL Unconference #15 (2020/07/30)
  - 検知できない破壊の話
- PostgreSQL Unconference #20 (2021/02/02)
  - プロセス障害の話
- PostgreSQL Unconference #26 (2021/08/24)
  - エラーが出ない話
- PostgreSQL Unconference #29 (2021/12/21)
  - 文字コードの話
- PostgreSQL Unconference #31 (2022/2/22)
  - Babelfish の話
- スライドはこちら

  https://github.com/nori-shinoda/documents#readme

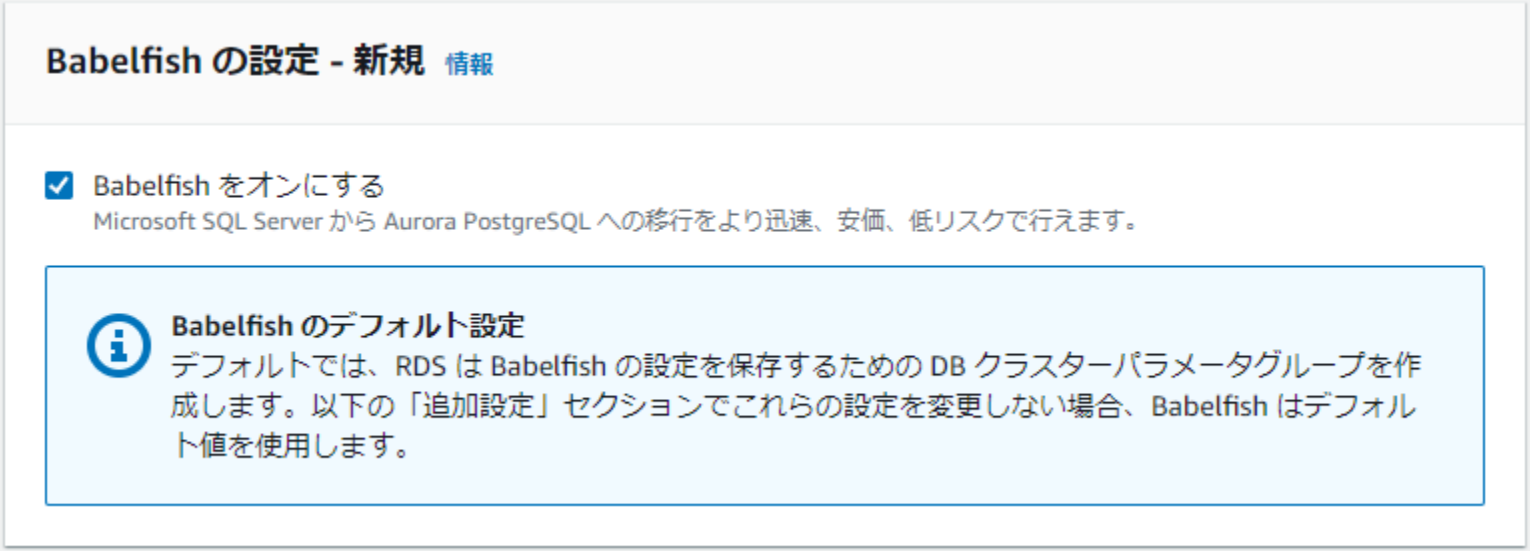
  https://www.slideshare.net/noriyoshishinoda

Babelfish とは？



Babelfish とは？
SQL Server 互換機能

- Microsoft SQL Server 互換機能を提供
- AWS Aurora for PostgreSQL のオプション
  - Aurora for PostgreSQL 13.4 のデータベース作成時に指定

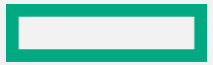


- Babelfish という名前
  - ダグラス・アダムスの「銀河ヒッチハイク・ガイド」に出てくる自動翻訳できる魚
  - AltaVista Babelfish Translation 等でも使われた

Babelfish とは？
Babelfish の特徴

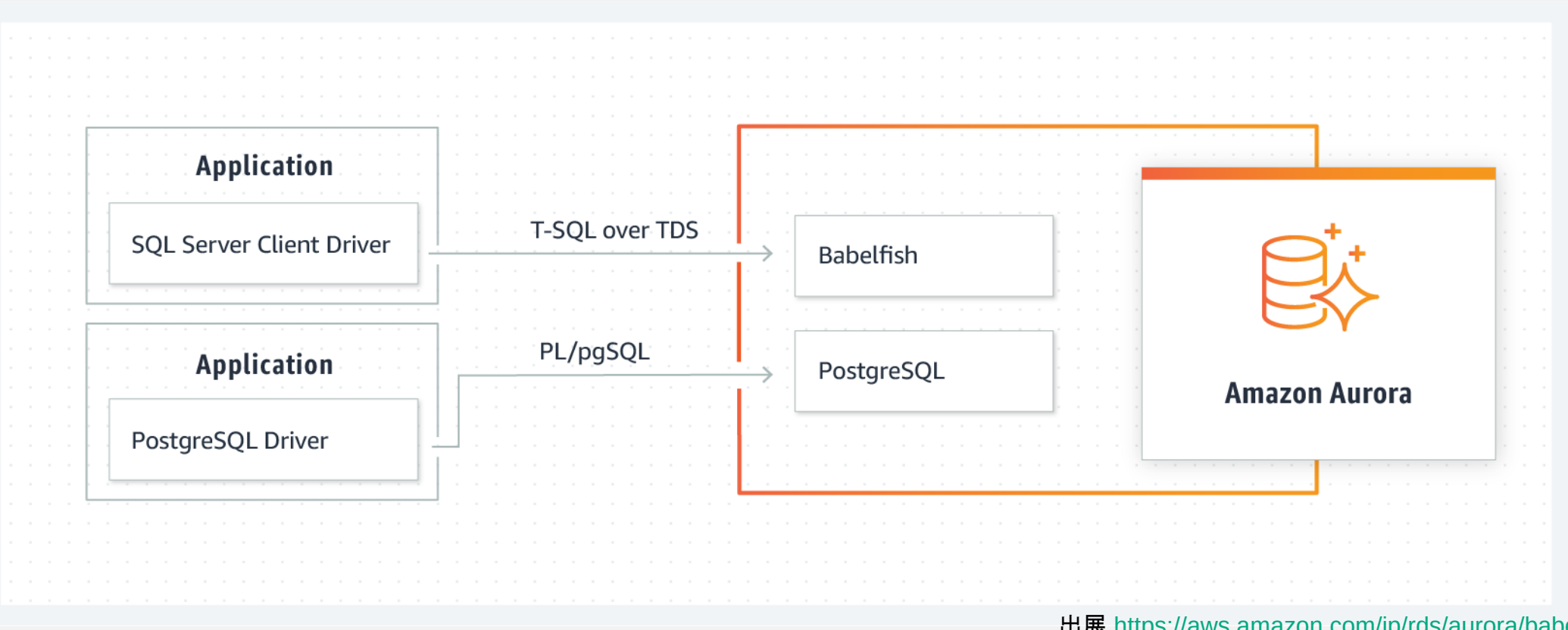
– SQL Server をプロトコル・レベルでエミュレーション

Product	互換機能	Client
orafce Extension	Oracle Database 関数	PostgreSQL
EDB Postgres Advanced Server	Oracle Database 構文、関数、オブジェクト、SQL 動作	EDB Postgres Client
Oracle SQL Translation Framework	Sybase 互換構文、関数等	Oracle Database Client
SQL Server Migration Assistant (SSMA)	Oracle Database 関数	SQL Server Client
Babelfish	SQL Server 構文、関数、オブジェクト	SQL Server Client



Babelfish とは？
Babelfish の特徴

– エンドポイントを2つ持ち、オブジェクトに相互アクセス可能
  – PostgreSQL (5432/TCP)、と Babelfish (1433/TCP)



出展 https://aws.amazon.com/jp/rds/aurora/babelfish/

Babelfish とは？
Babelfish の構成

- オープンソース版
  - 修正された PostgreSQL と4つのエクステンションから構成される

パッケージ	説明	備考
PostgreSQL 13.4	Babelfish 用に修正された PostgreSQL	
babelfishpg_common	Babelfish 共通関数用エクステンション	
babelfishpg_money	money / smallmoney 型用エクステンション	小数点以下4桁をサポート
babelfishpg_tds	TDS プロトコル・サポート用エクステンション	
babelfishpg_tsql	Transact SQL サポート用エクステンション	

- インストール方法
  https://babelfishpg.org/docs/installation/compiling-babelfish-from-source/

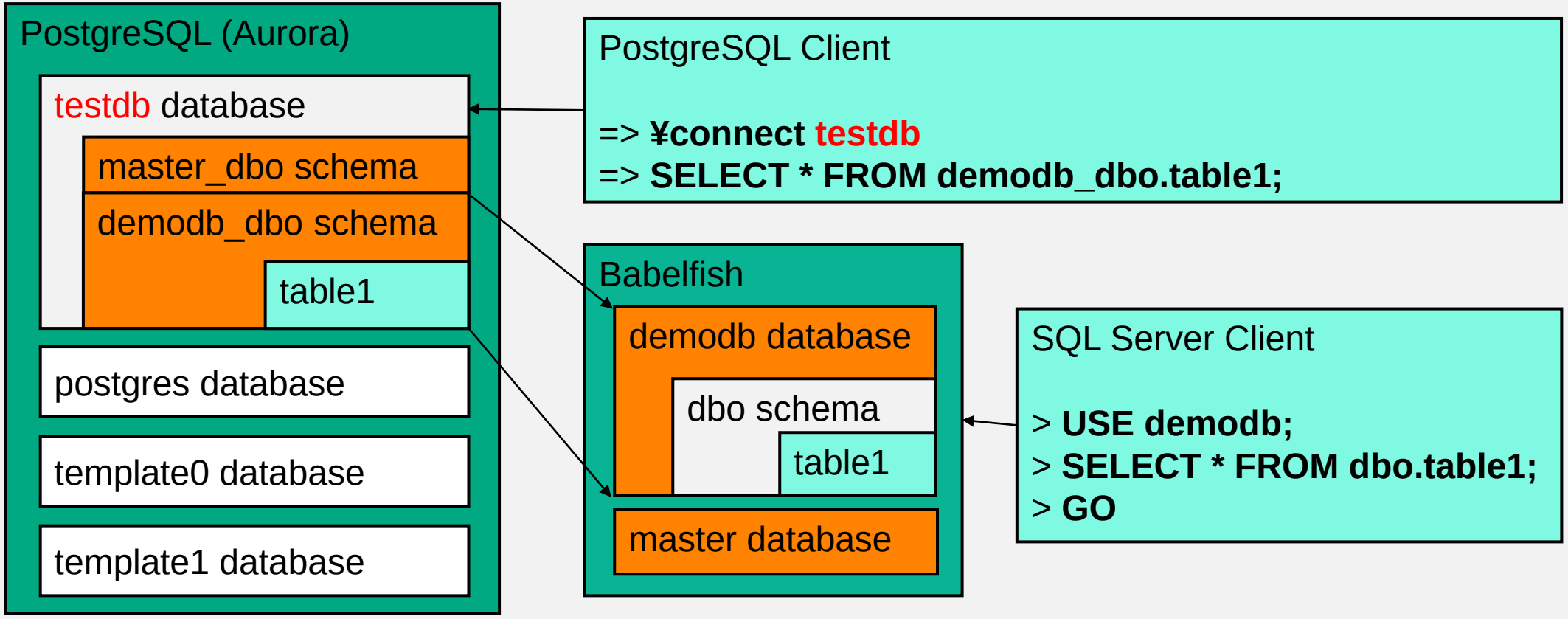
Babelfish とは？
データベースとスキーマの構成

- データベースとスキーマの関係
  - SQL Server では接続後もデータベースを変更可能（USE 文）
  - カレント以外のデータベースにもアクセス可能（Azure では異なる）
- SQL Server 接続例

$ **sqlcmd -U user1 -P passwd1 -d master**	← 初期接続データベースは master
1> **USE AdventureWorksDW;**	← AdventureWorksDW データベースへ切り替え
2> **SELECT * FROM dbo.DimAccount;**	← dbo スキーマ内の DimAccount テーブルを参照
3> **SELECT * FROM AdventureWorks.Sales.CustomerAddress;**	← AdventureWorks データベースのテーブルを参照
4> **GO**	

Babelfish とは？
データベースとスキーマの構成

– データベースとスキーマの関係



Babelfish とは？
データベースとスキーマの構成

- Babelfish の対応
  - 複数データベースを使用するかは、Babelfish 初期化時に決定する。
  - GUC babelfishpg_tsql.migration_mode による制御

設定値	説明	備考
single-db	初期接続したデータベースのみ利用可能	デフォルト値
multi-db	USE 文の使用や、異なるデータベースの参照／更新が可能	

Babelfish の非互換部分

Babelfish の非互換部分
トランザクション分離レベル

- トランザクション分離レベルは PostgreSQL と同じ
  - 対応していない分離レベルを指定してもエラーにはならない

LEVEL	SQL Server	PostgreSQL	Babelfish
READ UNCOMMITED	YES	NO	NO
READ COMMITTED	YES	YES	YES
REPEATABLE READ	YES	YES	YES
SNAPSHOT	YES	NO	NO
SERIALIZABLE	YES	YES	YES

Babelfish の非互換部分
トランザクション分離レベル

– 以下の例は READ COMMITTED として動作する

```
1> BEGIN TRANSACTION;
2> UPDATE data1 SET c2='after updated' WHERE c1=100;
3> GO
```

```
1> BEGIN TRANSACTION;
2> SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
3> SELECT * FROM data1 WHERE c1=100;
4> GO
c1                   c2
-------------------- ------------
                 100 before update
(1 rows affected)
```

Babelfish の非互換部分
更新と検索の同時実行

- SQL Server では更新中のタプルに対する検索（SELECT）処理はトランザクションの確定まで待機
- データベース属性 allow_snapshot_isolation と read_committed_snapshot を on に変更することで、PostgreSQL と同様の動作になる
- Babelfish では上記設定が有効になった状態で提供されている（変更不可）

```
1> SELECT snapshot_isolation_state_desc, is_read_committed_snapshot_on FROM sys.databases
    WHERE name='master';
2> GO
snapshot_isolation_state_desc   is_read_committed_snapshot_on
------------------------------ ------------------------------
ON                                                           1
(1 rows affected)
```

Babelfish の非互換部分
文字列型と Collation

– char / varchar / text 型に日本語を格納できない

```
1> CREATE TABLE encode1(c1 varchar(10), c2 nvarchar(10));
2> INSERT INTO encode1 VALUES ('漢字', N'漢字');
3> SELECT * FROM encode1;
4> GO
(1 rows affected)
c1         c2
---------- ----------
??         漢字
```

– Collation に日本語サポートが無い

```
1> SELECT name FROM sys.fn_helpcollations() WHERE name LIKE '%japan%';
2> GO
name
---------------------------------
(0 rows affected)
```

Babelfish の非互換部分
シーケンス

- シーケンスの開始番号は PostgreSQL と同じ
  - SQL Server では START WITH を省略した場合の開始番号はデータ型の最小値
- NEXT VALUE FOR 構文はサポートされない（NEXTVAL 関数を使う）

```
1> CREATE SEQUENCE seq1 START WITH 1;
2> SELECT NEXT VALUE FOR seq1;
3> GO
Msg 33557097, Level 16, State 1, Server BABELFISH, Line 1
'NEXT VALUE FOR' is not currently supported in Babelfish
1> SELECT NEXTVAL('seq1');
2> GO
```

- CREATE SEQUENCE 文、ALTER SEQUENCE 文は互換性あり

Babelfish の非互換部分
シノニム

– シノニムはサポートされていない

```
1> CREATE SYNONYM syn1 FOR dbo.table1;
2> GO
Msg 33557097, Level 16, State 1, Server BABELFISH, Line 1
'CREATE SYNONYM' is not currently supported in Babelfish
```

Babelfish の非互換部分
MERGE 文

– MERGE 文はサポートされない

```
1> MERGE dbo.Country2 AS C2
2> USING dbo.Country1 AS C1
3> ON C2.ID = C1.ID
4> WHEN MATCHED
5>    THEN UPDATE SET C2.Country = C1.Country, C2.City = C1.City
6> WHEN NOT MATCHED BY TARGET
7>    THEN INSERT (ID, Country, City) VALUES (C1.ID, C1.Country, C1.City)
8> WHEN NOT MATCHED BY SOURCE
9>    THEN DELETE;
10> GO
Msg 33557097, Level 16, State 1, Server BABELFISH, Line 1
'MERGE' is not currently supported in Babelfish
```

Babelfish の非互換部分
DEFERRABLE 制約

- CASE 句による主キー入れ替え
- 正常に動作させるためには PostgreSQL では制約に DEFERRABLE 句が必要

```
1> CREATE TABLE table1(c1 NUMERIC PRIMARY KEY);
2> INSERT INTO table1 VALUES (1);
3> INSERT INTO table1 VALUES (2);
4> UPDATE table1 SET c1 = CASE c1 WHEN 1 THEN 2 WHEN 2 THEN 1 END;
5> GO

(1 rows affected)
(1 rows affected)
Msg 2627, Level 14, State 1, Server BABELFISH, Line 4
duplicate key value violates unique constraint "table1_pkey"
```

Babelfish の非互換部分
カーソル

- 無視されるカーソル属性
  - FAST_FORWARD, FOR UPDATE
- エラーになるカーソル属性
  - SCROLL, GLOBAL, KEYSET, DYNAMIC, SCROLL_LOCKS, OPTIMISTIC, TYPE_WARNING
- マニュアル上はサポートされないとされているが動作しているように見える構文
  - FETCH PRIOR, FETCH FIRST, FETCH LAST, FETCH ABSOLUTE, FETCH RELATIVE

```
1> DECLARE cursor1 CURSOR DYNAMIC FOR SELECT c2 FROM data1;
2> GO
Msg 33557097, Level 16, State 1, Server BABELFISH, Line 1
'DYNAMIC CURSOR' is not currently supported in Babelfish
```

Babelfish の非互換部分
マニュアル

– 現状ではサポートされない機能

https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/AuroraUserGuide/babelfish-compatibility.html

https://babelfishpg.org/docs/usage/limitations-of-babelfish

Babelfish の互換性

Babelfish の互換性
トランザクション内エラー処理の互換性

– SQL Server ではトランザクション内でエラーが発生しても部分コミット可能
– Babelfish でも同様、DDL がロールバックできる点は元々同じ

```
1> BEGIN TRANSACTION;
2> INSERT INTO data1 VALUES (100, 'normal data');
3> INSERT INTO data1 VALUES (100, 'duplicate data');
4> COMMIT;
5> SELECT * FROM data1;
6> GO
Msg 2627, Level 14, State 1, Server BABELFISH, Line 3
duplicate key value violates unique constraint "data1_pkey"
c1                   c2
-------------------- ----------
                 100 normal data
(1 rows affected)
```

Babelfish の互換性
トランザクションと時刻の互換性

- CURRENT_TIMESTAMP 関数は OS 時刻を返す（≠ トランザクション開始時刻）
- PostgreSQL の NOW 関数も使えるが、こちらはトランザクション間は同一時刻を返す

```
1> BEGIN TRANSACTION;
2> SELECT CURRENT_TIMESTAMP, NOW();
3> GO
current_timestamp       now
----------------------- -----------------------------
2022-02-10 18:58:06.527 2022-02-10 22:34.28.024456+09

1> SELECT CURRENT_TIMESTAMP, NOW();
2> GO
current_timestamp       now
----------------------- -----------------------------
2022-02-10 19:01:06.210 2022-02-10 22:34.28.024456+09
```

Babelfish の互換性
データ型の互換性

– 非互換があるデータ型は独自に作成されている

SQL Server	Babelfish	備考
char / varchar / text	sys.bpchar / sys.varchar / text	text は同一
nchar / nvarchar / ntext	sys.nchar / sys.nvarchar / sys.ntext	
bit / decimal / money / smallmoney / tinyint / bigint / integer / smallint / tinyint / real / float	sys.bit / sys.decimal / sys.money / sys.smallmoney / bigint / integer / smallint / sys.tinyint / real / double	bigint, integer, numeric, smallint, real は同一。float は double にマップ
date / time / datetime2 / datetime / datetimeoffset	date / time without time zone / sys.datetime2 / sys.datetime / sys.datetimeoffset	date は同一
binary / varbinary / image	sys.binary / sys.varbinary / sys.image	

Babelfish の互換性
特殊な構文の互換性

– テーブル作成構文
  – SELECT文の結果からテーブルを作成する構文

SQL Server/Bablefish

・実テーブル

```
1> SELECT * INTO data2 FROM data1;
2> GO
```

・一時テーブル

```
1> SELECT * INTO #temp1 FROM data1;
2> GO
```

PostgreSQL

・実テーブル

```
postgres=> CREATE TABLE data2 AS SELECT * FROM data1;
SELECT 1000
```

・一時テーブル

```
postgres=> CREATE TEMPORARY TABLE temp1 AS SELLECT *
    FROM data1;
SELECT 1000
```

Babelfish の互換性
特殊な構文の互換性

– かなり特殊な構文もサポートされている（その通りの動作にはならない場合あり）

・列名のエイリアス
```
> SELECT alias1=count(*) FROM data1;
```
・WITH 句（テーブルオプション）
```
> SELECT * FROM data1 WITH (READUNCOMMITTED) WHERE c1=100;
```
・OPTION 句と結合方法（ヒント）
```
> SELECT count(*) FROM data1 INNER HASH JOIN data2 ON data1.c1=data2.c1
      OPTION (FORCE ORDER);
```
・先頭行
```
> SELECT TOP 10 * FROM data1 ORDER BY 1;
```
・ROW_NUMBER OVER 関数
```
> SELECT c1 FROM (SELECT c1,ROW_NUMBER() OVER (ORDER BY c1) AS rownum FROM data1) AS t
  WHERE rownum BETWEEN 101 AND 200 ORDER BY rownum;
```

Babelfish の互換性
ORDER BY 句の互換性

- SQL Server では NULL は昇順 ORDER BY で最小扱い
- NULLS FIRST / NULLS LAST 句は存在しない

SQL Server/Bablefish

```
1> SELECT * FROM data1 ORDER BY c2;
2> GO
c1          c2
----------- ----------
          0 NULL
        100 data1
        200 data2

(3 rows affected)
```

PostgreSQL

```
postgres=> SELECT * FROM data1 ORDER BY c2;

 c1  |  c2
-----+-------
 100 | data1
 200 | data2
   0 | null

(3 rows)
```

Babelfish の互換性
演算子の互換性

– 文字列の連結演算子

SQL Server/Bablefish

```
1> SELECT '123' + 456 plus1;
2> GO
plus1
-----------
        579
(1 rows affected)
1> SELECT '123' + '456' plus1;
2> GO
plus1
------
123456
(1 rows affected)
```

PostgreSQL

```
postgres=> SELECT '123' + 456 plus1;

 plus1
-------
   579
(1 row)
postgres=> SELECT '123' + '456' plus1;
ERROR:  operator is not unique: unknown + unknown
LINE 1: SELECT '123' + '456' plus1;
                       ^

HINT:  Could not choose a best candidate operator. You
might need to add explicit type casts.
```

Babelfish の互換性
ストアド・プロシージャ

－SQL Server/Babelfish

```
CREATE FUNCTION dbo.ufnGetInventoryStock
    (@ProductID INT) RETURNS INT AS
BEGIN
    DECLARE @ret INT;
    SELECT @ret = SUM(p.Quantity)
        FROM Production.ProductInventory p
        WHERE p.ProductID = @ProductID
            AND p.LocationID = '6';
    IF (@ret IS NULL)
        SET @ret = 0;
    RETURN @ret;
END;
```

PostgreSQL (PL/pgSQL)

```
CREATE FUNCTION dbo.ufnGetInventoryStock
    (_ProductID INT) RETURNS INT AS $$
DECLARE ret INT;
BEGIN
    SELECT INTO ret SUM(p.Quantity)
        FROM Production.ProductInventory p
        WHERE p.ProductID = _ProductID
            AND p.LocationID = '6';
    IF ret IS NULL THEN
        ret := 0;
    END IF;
    RETURN ret;
END; $$ LANGUAGE plpgsql;
```

まとめ

まとめ

- 互換性はかなり高い
- 日本語 Collation 対応ができることが必要
- SQL Server on RDS との差別化が必要では？

THANK YOU

Mail: noriyoshi.shinoda@hpe.com
Twitter: @nori_shinoda