PL/SQL プログラミング I
～研修受講後のスキルアップサポート～

対応バージョン：Oracle 10gR1 ～ 12cR1

本資料は、アシスト Oracle 研修をご受講いただいたお客様からのご質問や、研修ではご案内できなかった情報などを FAQ にまとめたものです。研修受講後のスキルアップの一助として、是非お役立てください。
※ご利用上の注意事項は最後のページにまとめられております。ご確認のうえ、ご利用ください。

第 1 章　PL/SQL 概要

1 Q. 無名の PL/SQL ブロックを実行するには、権限が必要ですか。

A. 無名の PL/SQL ブロックの実行に権限は必要ありません。

※ストアド・サブプログラムを作成する場合は CREATE PROCEDURE、CREATE TRIGGER などのシステム権限が必要です。

2 Q. 現行インスタンスで実行された無名の PL/SQL ブロックを確認する方法

A. 無名の PL/SQL ブロックは、実行するたびに解析が行なわれるため、CPU 負荷の原因となる可能性があります。

サイズの大きい無名の PL/SQL ブロックは、ストアド化し、データベースに格納することをお薦めいたします。

現行インスタンスで実行されたことがある無名の PL/SQL ブロックは、管理者ユーザーでログイン後、以下の SQL 文で確認することができます。

例）
```
SQL> SELECT substrb(sql_text,1,80)
  2  FROM   v$sqlarea
  3  WHERE  command_type = 47
  4  AND    length(sql_text) > 100  ←サイズを指定してください。
  5  ;
```

3 Q. ブロック内のネスト最大数

A. 255 までネストできます。

第 2 章　PL/SQL の基本記述

4 Q. 暗黙カーソル属性を使用する際の注意事項

A. 暗黙カーソル属性はセッションに 1 つだけ存在するもので、直前に使用された暗黙カーソルの情報を参照します。

暗黙カーソルは直前に行われた SQL 文の実行後、自動的にクローズされるため、暗黙カーソル属性は参照したい SQL 文（SELECT...INTO 文や DML 文）の直後に使用することをお薦めいたします。

5	Q.	結果セットを1行飛ばしにFETCHすることはできますか。
	A.	できません。Oracleは必ず結果セットを上から1行ずつFETCHします。
6	Q.	カーソルを一度にいくつまでオープンできますか。
	A.	初期化パラメータOPEN_CURSORSで指定したの値までカーソルをオープンする事ができます。 ※初期化パラメータOPEN_CURSORSの詳細については「リファレンス」マニュアルをご参照ください。
7	Q.	データベースから取り出された行が1行もない場合に、エラーを発生させずに処理することは できますか。

	A.	FETCH INTO文を使用することで可能になります。
		
		例)
		…(省略)…
		OPEN カーソル名;
		LOOP
		FETCH カーソル名 INTO 変数名;
		IF カーソル名%ROWCOUNT = 0 THEN
		処理 ; ←0行の場合の対処を記述
		EXIT; ←LOOPを抜ける。
		END IF;
		…(省略)…

8	Q.	PL/SQL内でROWIDを使用してSQLを高速に処理する方法

	A.	SQLで特定の1行にアクセスする場合、WHERE句に"WHERE 列名 = ～ "と
		指定するよりも、"WHERE ROWID = ～ "とした方が高速にSQLを処理できます。
		
		ROWIDとは行のアドレスで、データが格納される物理的な場所を示しているため、
		データベース全体で行を一意に識別できます。
		そのため、ROWIDをWHERE句で指定すると、対象行が格納されている場所に直接
		アクセスできるので、特定の1行へ最も高速にアクセスできます。
		
		PL/SQLでは、ROWIDデータ型を使用して効率よく、特定の行にアクセスするSQLを
		実行させることが可能です。
		
		例)
		DECLARE
		CURSOR cur_emp IS SELECT rowid,ename,sal,deptno FROM emp;
		BEGIN
		FOR emp_rec IN cur_emp LOOP
		UPDATE emp
		SET sal=sal*1.1
		WHERE rowid = emp_rec.rowid; ← ROWIDの指定
		END LOOP;
		END;
		
		※CURRENT OF カーソルも同じようにROWIDを使用するため、高速です。

9	Q.	現在オープンされているカーソルの確認を行うことはできますか。
	A.	V$OPEN_CURSOR ビューでオープンされている以下のカーソルを確認できます。 ・暗黙カーソル ・明示的にオープンされている静的カーソル ・オープンされ、まだクローズされていない動的カーソル ※オープンはされているが、まだ解析されていないカーソルはカウントされませんので ご注意ください。 ※V$OPEN_CURSOR ビューの詳細については「リファレンス」マニュアルをご参照ください。 ※動的カーソルについては「PL/SQL プログラミングⅡ」コースでご紹介しています。
10	Q.	識別子とデータベース・オブジェクトの名前が重複していた場合は、どちらが優先されるのですか。
	A.	データベース・オブジェクトの名前は、変数名や仮パラメータ名より優先されます。
11	Q.	IF 文や検索 CASE 文で複数の条件を指定する方法
	A.	AND、OR 演算子を使用することで、複数の条件を指定できます。 また、条件の指定順番を意識することで、無駄な審査を省けます。 ・A 条件 AND B 条件　⇒ A、B 条件を共に満たす必要がある。 ※FALSE と評価されやすい条件を先に指定すると無駄な審査を省けます。 ・A 条件 OR　B 条件　⇒ A、B 条件のどちらか一方が満たされればよい。 ※TRUE と評価されやすい条件を先に指定すると無駄な審査を省けます。 例) …省略… IF var1=1 AND var2=2 THEN DBMS_OUTPUT.PUT_LINE('TRUE'); ELSE DBMS_OUTPUT.PUT_LINE('FALSE'); END IF; IF var1=3 OR var2=1 THEN DBMS_OUTPUT.PUT_LINE('TRUE'); ELSE DBMS_OUTPUT.PUT_LINE('FALSE'); END IF; …省略…
12	Q.	GOTO 文などの制御文はなぜ使用が推奨されていないのですか。
	A.	PL/SQL のようなプログラムはブロック構造化言語であり、上から下に処理を進めていきます。 しかし、GOTO 文や EXIT 文はそれを無視し、無条件で制御を自由に移すことができます。 このような記述はプログラムの原則に反するばかりか、どのような過程で処理が行われているのか 非常に分かりづらいプログラムになってしまいます。 そのため、順次制御などを行う際は必ず理由をつけて処理を行うなど見やすく、 分かりやすいプログラムを作成することを心掛けることを推奨します。

13	Q.	カーソル FOR ループ内で例外が発生したとき、カーソルは明示的にクローズする必要がありますか。
	A.	カーソル FOR ループ内で例外が発生した場合、カーソルは自動的にクローズされます。 例外処理部にてカーソルを明示的にクローズする処理は必要ありません。
14	Q.	複数のカーソルに全く同じ問合せ文が関連づけられていた場合、同じカーソルを使用するのでしょうか。
	A.	複数のカーソルに同一の問合せ文が関連付けられていたとしても、 それぞれ異なるカーソル名と関連付けられるために 共有メモリー上に異なる作業領域を確保します。共有はされません。
15	Q.	PL/SQL 内で宣言した変数をセッションを通して使用することは可能でしょうか。
	A.	変数の有効範囲は、自ブロック、およびサブブロック内となるため、基本的にはできません。 変数をセッションを通して使用したい場合は、パッケージを作成する必要があります。 ※パッケージについては、「PL/SQL プログラミングⅡ」コースでご紹介しています。
16	Q.	カーソルに対応付ける問合せ文の選択リストに、式や関数を指定する場合の注意事項。
	A.	カーソルに対応付ける問合せ文の選択リストに式や関数を指定する場合は、 式や関数に別名を定義してください。 例) CUROSOR カーソル名 IS SELECT 式や関数 AS 別名 FROM 表名; 別名を定義しなかった場合、%ROWTYPE などによる変数使用時にフィールド名を指定できません。
17	Q.	ユーザー定義の例外名に事前定義の例外名を指定した場合、どちらが有効になるのですか。
	A.	ユーザー定義の例外名を指定する際に、TOO_MANY_ROWS などの事前定義の例外名を 指定することができます。この場合、明示的に宣言された例外名が、事前定義の 例外名よりも優先されます。 ※事前定義の例外名の前に「STANDARD.」と記述することで、明示的に同名の例外が 定義されている場合でも、両者を区別して使うことはできます。ただし、プログラムを 判読しにくくなるため、なるべく同一名にしないことをお薦めいたします。 例) …省略… EXCEPTION WHEN no_data_found ←明示的に定義された例外 THEN DBMS_OUTPUT.PUT_LINE('NOTHING'); WHEN STANDARD.no_data_found ←事前定義の例外 THEN NULL; END;

第3章　ストアド・サブプログラム		
18	Q.	ストアド・サブプログラムを作成する時に定義する「IS」「AS」に違いはありますか。
	A.	違いはありません。
19	Q.	ストアド・サブプログラムが実行されるとメモリ上のどの領域にロードされるのですか。
	A.	共有プールのライブラリ・キャッシュ内にロードされます。 ※ライブラリ・キャッシュについては「データベース・アーキテクチャ」コースでご紹介しています。
20	Q.	ファンクション内にDML文を記述した場合に、SQLで使用できないのはどんな場合ですか。
	A.	DML文を含んだファンクションをSQL文から呼び出す場合、以下の制限が発生します。 ・DML文を含んだファンクションを問合せ文から呼び出すことはできません。 ・DMLから呼び出されるファンクションは、そのDML文によって変更された表に対して 問合せやDML処理を行うことはできません。 ※ただしファンクション内で行う処理が違う表であれば変更できます。
21	Q.	リモート・データベースのストアド・サブプログラムを実行する方法

	A.	通常のオブジェクト同様、リモート・データベースのストアド・サブプログラムを
		実行する場合は、データベース・リンクを使用します。データベース・リンクとは
		リモート・データベースにアクセスするためのデータベース・オブジェクトです。
		
		例)リモート・データベース上のremote_proプロシージャを実行する。
		※tokyo.oracleというデータベース・リンクを予め作成済み。
		
		`/* SQL*PlusのEXECUTEコマンドで実行する */`
		`SQL> EXECUTE remote_pro@tokyo.oracle;`
		
		`/* ローカル・データベースのサブプログラム内でコールする */`
		`SQL> CREATE OR REPLACE PROCEDURE local_pro`
		`SQL> IS`
		`SQL> BEGIN`
		`SQL>  remote_pro@tokyo.oracle;`
		`SQL> END;`

22	Q.	ストアド・プロシージャ作成時に、OR REPLACE句を使って、同名のストアド・ファンクションと 置き換えることはできますか。
	A.	できません。OR REPLACE句による置き換えは、同じタイプ(プロシージャやファンクション)で ある必要があります。異なるタイプのプログラムに置き換えようとすると、エラーが発生します。

23	Q.	パラメータの数やデータ型の違いで、同名のプログラムを作成できますか。
	A.	単体のストアド・サブプログラムでは行えませんが、パッケージを使用すれば可能です。 ※パッケージについては「PL/SQL プログラミングⅡ」コースでご紹介しています。
24	Q.	他のユーザーが所有する表にアクセスするストアド・サブプログラムの注意事項。
	A.	他のユーザーが所有する表にアクセスするストアド・サブプログラムを作成/実行するためには、 表を所有するユーザーから直接オブジェクト権限を付与する必要があります。ロール経由で オブジェクト権限を与えるだけではエラーが発生します。
第4章　パラメータ		
25	Q.	ストアド・サブプログラムに設定したパラメータの情報を確認する方法。
	A.	ストアド・サブプログラムに設定したパラメータに関する情報は、USER_ARGUMENTS ビューや SQL*Plus の DESCRIBE コマンドで、確認することができます。 ただし、パラメータのデフォルト値については、上記の方法で確認することができないため、 USER_SOURCE ビューを使って直接ソースコードを確認する必要があります。 ※USER_ARGUMENTS ビュー、USER_SOURCE ビューの詳細については、「リファレンス」マニュアルを ご参照ください。
26	Q.	パラメータのデータ型に%ROWTYPE を指定することはできますか。
	A.	できます。パラメータのデータ型に%ROWTYPE を指定することで、ストアド・サブプログラム間で 結果セットのデータを受け渡すことができます。
第5章　ファンクション		
27	Q.	ファンクション名と列名の優先順位について
	A.	列名と同名のファンクションをパラメータを指定せずに実行した場合、 列名の方が優先して認識されます。 この場合、ファンクションをスキーマ名で修飾することで、ファンクションとして 認識させることができます。 ※ただし、ソースコードの判読が難しくなるため、列名と同名のファンクションは 作成しないことをお薦めします。
第6章　トリガー		
28	Q.	データベース・トリガー内でデータベース・リンクを使用してリモート・データベースに 対して処理を行なうことはサポートされていますか。
	A.	通常のストアド・サブプログラム同様に、データベース・トリガー内でも データベース・リンクを使用することは可能です。

29	Q.	INSERT、UPDATE を行った時刻を同じ表の列に記録するには、どうすればよいですか。

	A.	BEFORE 行トリガーと SYSDATE 関数を使用します。
		
		例)DEPT 表に INSERT または UPDATE を行った時間を LAST_DML 列に記録する
		
		CREATE OR REPLACE TRIGGER dept_tri
		BEFORE INSERT OR UPDATE ON dept
		FOR EACH ROW
		BEGIN
		　SELECT SYSDATE INTO :NEW.LAST_DML FROM dual;
		END;
		/

30	Q.	トリガーを、「毎週日曜日」のように、スケジュールに従って定期的に実行できますか。
	A.	トリガーは、スケジュールに応じて起動させることはできません。 スケジュールに応じてプログラムを動作させる際には、DBMS_SCHEDULER パッケージを使用します。 ※DMBS_SCHEDULER パッケージについては「PL/SQL プログラミングⅡ」コースでご紹介しています。
31	Q.	トリガー内でコールされたストアド・サブプログラム内にトランザクション制御文が 含まれていた場合は、エラーが発生しますか。
	A.	トリガーが実行された時にエラーが発生します。 これはトリガーが起動されたトランザクションとコールされたストアド・サブプログラムの トランザクションが同一のためです。 それぞれのトランザクションを独立させたい場合は、自立型トランザクション （PRAGMA AUTONOMOUS_TRANSACTION）を指定してください。
32	Q.	無限ループを解決するにはどうすればよいですか。
	A.	無限ループを解決するには、管理者ユーザーでデータベースに接続し、 無限ループが発生している対象セッションを強制的に切断します。 ※セッションの強制切断については「データベース・マネジメント」コースでご紹介しています。

第7章　ストアド・サブプログラムの管理

33 Q. 既に削除されてしまった表の依存オブジェクト（削除された表を参照するオブジェクト）を確認する方法

A. 既に削除されてしまった表の依存オブジェクトは、USER_OBJECTS ビューと
USER_DEPENDENCIES ビューを使って確認できます。

削除された表の依存オブジェクトは、USER_PBJRCTS ビューの STATUS 列の値が
INVALID となっています。また、USER_DEPENDENCIES ビューには既に削除された表の
依存オブジェクトの情報が存在しません。
そのため、上記を問合わせる SELECT 文に MINUS 演算子を使用することで、
既に削除された表の依存オブジェクトの情報を確認できます。

例)削除された表を参照するオブジェクトを確認する。

```
SQL> SELECT object_name,object_type
  2  FROM user_objects
  3  WHERE status = 'INVALID'
  4  MINUS
  5  SELECT name,type
  6  FROM user_dependencies
  7  WHERE referenced_type = 'TABLE';

OBJECT_NAM OBJECT_TYPE
---------- ------------------
EXE_PRO    PROCEDURE
NEWSAL_PRO PROCEDURE
```

付録

34 Q. 1 つの SQL 文の中で何回も順序を参照している場合、どんな値が戻されますか。

A. 同時に順序にアクセスした場合でも、Oracle によって順序の一意性が保証されています。

よって 1 つの SQL 文の中で複数回 NEXTVAL 列を使用していても、
Oracle は一度だけ順序を増加させ、すべての NEXTVAL 列に同じ値を戻します。
例)

```
SQL> SELECT SEQ1.NEXTVAL,SEQ1.NEXTVAL,SEQ1.NEXTVAL
     FROM dual;

NEXTVAL  NEXTVAL  NEXTVAL
-------  --------  --------
   1560      1560      1560
```

35 Q. 順序をはじめて確認する際に 順序名.NEXTVAL を使用するのはなぜですか。

A. 順序は何らかのアクションによって順序を生成します。しかし、順序を作成時点では
まだアクションがないために初期値が生成されていません。

そのため、順序を増分して次に戻される値を確認できる 順序名.NEXTVAL で値を参照する
必要があります。

36	Q.	順序の値を初期値に戻したり、別の値から再開することはできますか。
	A.	順序は現在の値を直ちに初期値に戻したり、別の値から再開することはできません。 この場合はいったん順序を再作成してください。 ただし、CYCLE オプションを指定して順序を作成することで、最大値に達した後に 自動的に初期値に戻せます。 例)5 づつ増加し、100 に達したら初期値に戻る順序を作成 SQL> CREATE SEQUENCE seq1 2 INCREMENT BY 5 3 START WITH 5 4 MAXVALUE 100 5 CYCLE;
37	Q.	順序の最大値についての注意事項。
	A.	順序の最大値は 28 桁まで指定することができます。 また、最大値を設定しなかった場合、値は無制限となります。 なお、最大値に達した後に値を生成しようとした場合にはエラーが発生します。 ※CYCLE オプションを指定して順序を作成することで、自動的に初期値に戻せます。
38	Q.	作成した順序の次の番号を知る方法
	A.	動的パフォーマンスビュー V$_SEQUENCES の NEXTVALUE 列で次の順序を 確認できます。 例)順序で生成される次の番号を確認する。 ※SYS ユーザーにて参照する必要があります。 SQL> SELECT sequence_name,nextvalue FROM V$_SEQUENCES; SEQUENCE_NAME NEXTVALUE --------------- --------- SEQ1 1260 ※順序作成時に NOCACHE オプションを設定した場合は値が出力されませんので ご注意ください。
39	Q.	NCHAR、NVARCHAR2 型などでサポートされている NLS 文字がどういうものなのか教えて下さい。
	A.	NLS 文字が使用できる環境下では、各国語やキャラクタセットを意識せずに 処理を行うことができます。 NLS 文字が使用できる環境では、Oracle サーバとクライアントでキャラクタ・セットが 異なっていても自動的に変換が行われ、処理を行うことができます。 また NLS の環境下では各国語がサポートされるため、アプリケーションの修正を行うことなく それぞれの国にあった言語、日付、数値などを扱うことができます。

40	Q.	事前定義の内部例外（付-23）に ORA-xx から始まる番号と、符号付きの整数がありますが、 これらは何を意味しますか。
	A.	ORA-xx から始まる番号は、Oracle のエラー番号を意味し、 符号付きの整数は、SQLCODE 関数の戻り値を意味します。
41	Q.	PL/SQL の予約語を確認する方法。
	A.	V$RESERVED_WORDS ビューで確認できます。
42	Q.	自律型トランザクション設定したプログラムにトランザクション制御文(COMMIT、ROLLBACK 文)を 含めないとどうなりますか。
	A.	プログラム終了までに、トランザクションが終了しなかった場合は例外が発生します。
その他		
43	Q.	再帰サブプログラムについて。

	A.	自分自身を呼び出すサブプログラムのことを、再帰サブプログラムといいます。
		再帰サブプログラムを使用することによってプログラムの設計を単純化することが
		できる場合があります。
		
		例)階乗計算するためのファンクションを再帰サブプログラムを使って作成する
		CREATE OR REPLACE FUNCTION kaijou(n NUMBER)
		RETURN INTEGER
		IS
		BEGIN
		IF n=1 THEN
		RETURN (1);
		ELSE
		RETURN (n*kaijou(n-1)); ←自分自身をコール
		END IF;
		END;
		
		/* 再帰サブプログラムの呼び出し */
		SQL> SELECT kaijou(4) FROM dual;
		
		KAIJOU(4)
		---------- 	
		24
		
		※再帰サブプログラムでは終了条件を記述しないと、無限ループが発生するため、
		必ず終了条件を記述してください。

44	Q.	SQL*Plus で PL/SQL ブロックを実行すると、画面に「PL/SQL プロシージャが正常に完了しました。」 と表示されないようにする方法。
	A.	SQL*Plus の SET コマンドで可能です。 SET FEEDBACK OFF を SQL*Plus で実行してください。

45	Q.	データベースに登録済みのストアド・サブプログラムのソースを OS ファイルに保存する方法
	A.	1.以下のようにスクリプト・ファイル（source.sql）を作成します。 －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－ SET ECHO OFF　　--ファイル内の実行コマンドを非表示にする SET HEADING OFF　--検索結果の列ヘッダーを非表示にする SET FEEDBACK OFF --問合せ結果のレコード数を非表示にする SET VERIFY OFF　--置換変数を値と置き換える前後の SQL を非表示にする DEFINE exte =　　--'.sql'　→出力ファイルの拡張子を設定する ACCEPT name CHAR PROMPT 'INPUT PROCEDURE or FUNCTION NAME! : 　　--変数を宣言する。 　　--（PROMPT オプションで指定したプロンプトをユーザに表示できる） SPOOL &name&exte　--出力結果を OS ファイルに保存。 　　--（ファイル名は先に宣言した変数を使用する） SET TERM OFF　　--スクリプトからの出力を非表示にする。 SELECT text FROM user_source WHERE name = '&name'; 　　-- ストアド・サブプログラムのソースの検索する。 SET TERM ON SPOOL OFF SET VERIFY ON SET FEEDBACK ON SET HEADING ON SET ECHO ON －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－ 2.SQL*Plus から上記のスクリプト・ファイルを実行します。 SQL> @source.sql INPUT PROCEDURE or FUNCTION NAME! : NEWSAL_PRO 　　↑ 　　ソースを調べたいサブプログラムの名前を入力します(大文字で) ※生成されたファイル名は「ストアド・サブプログラム名.sql」となります
46	Q.	SQL*Plus に PL/SQL を直接記述していた場合、途中で記述をやめる方法。
	A.	新しい行に「.」(ピリオド)のみを入力して Enter キーを押してください。 ※新しい行に「/」(スラッシュ)を入力しても終了になりますが、この場合は PL/SQL を 実行して終了します。ブロックの記述が途中である場合はエラーが返ります。
47	Q.	PL/SQL コードサイズの最大値について
	A.	256MB までです（トリガーは 32KB までです）。 ただし、コードサイズの最大値は内部計算によって求められるため、正確な値を 算出することはできません。あくまでも目安なので、必ずしもこれらの値で作成 できるとは限りませんので、ご了承ください。

※ ご利用上の注意事項※

- 本書の著作権は株式会社アシストに帰属します。
- 本書は参考資料であり、掲載されている情報は予告なしに変更されることがあります。
- 本書で使用している製品の名称は、各社の商標または登録商標です。
- 本資料の内容に関するご質問はご遠慮ください。
- 本資料はお客様の責任のもとでご利用ください。これらの使用によりいかなる損害が生じたとしても、株式会社アシストは一切保証致しかねますので、ご了承ください。