Oracle

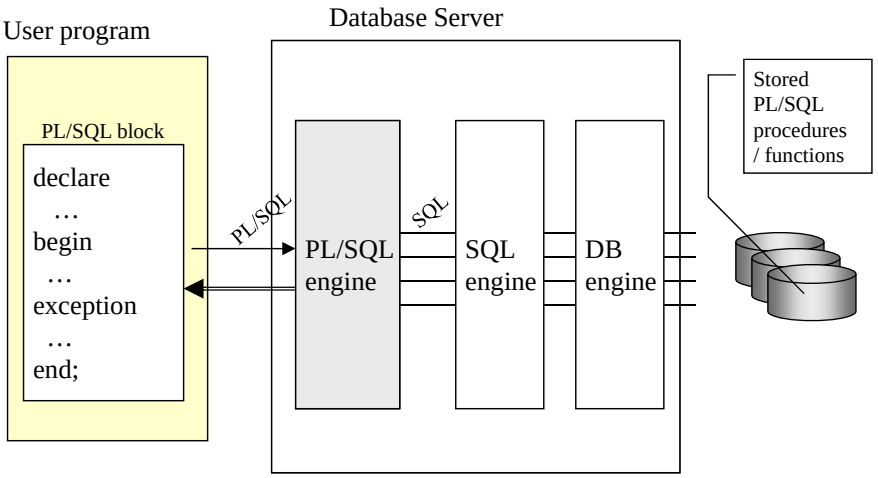
Oracle

PL/SQL Procedural Language

based on
Günther Stürner: Oracle7 - A User's and Developer's Guide
Michael R. Ault: Oracle 7.0 Administration & Management
Oracle8 online documentation
Feuerstein et al: Oracle PL/SQL Language

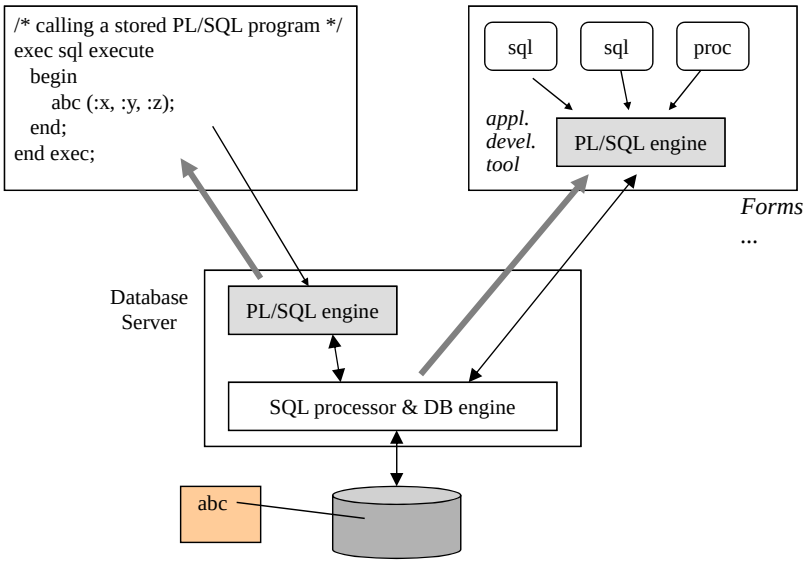
Oracle

Processing PL/SQL Block



Oracle

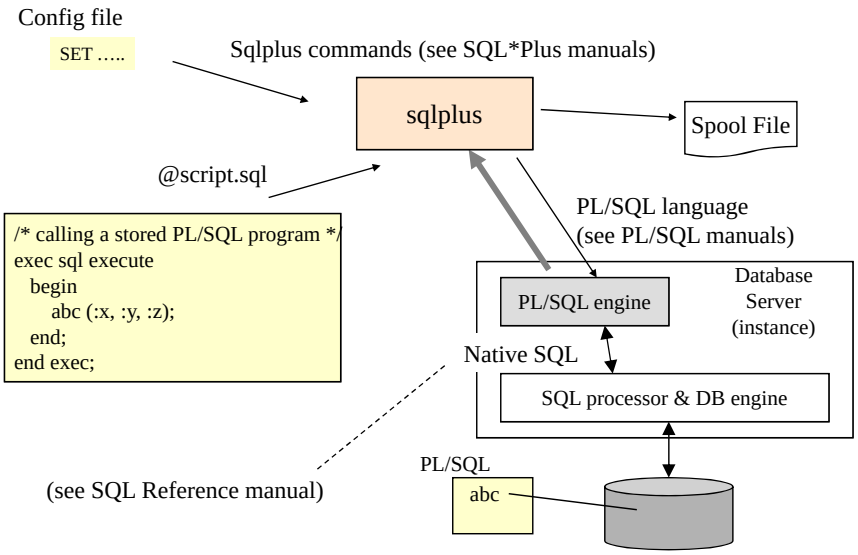
PL/SQL in Client/Server



Helia / Martti Laiho 1998-2006

Oracle

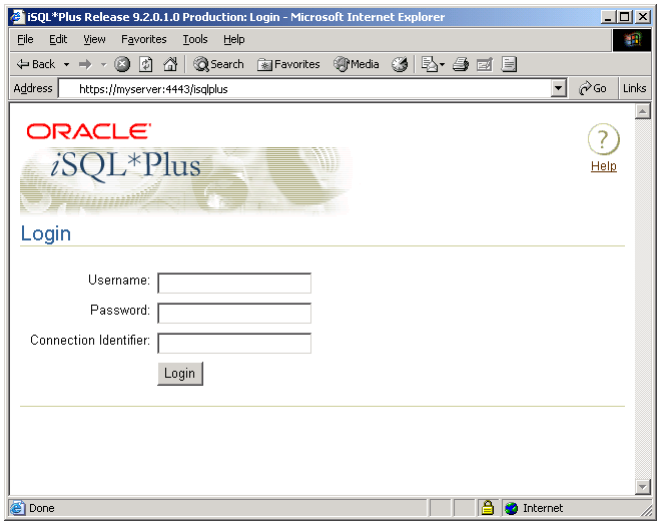
SQL*Plus



Helia / Martti Laiho 1998-2006

Oracle

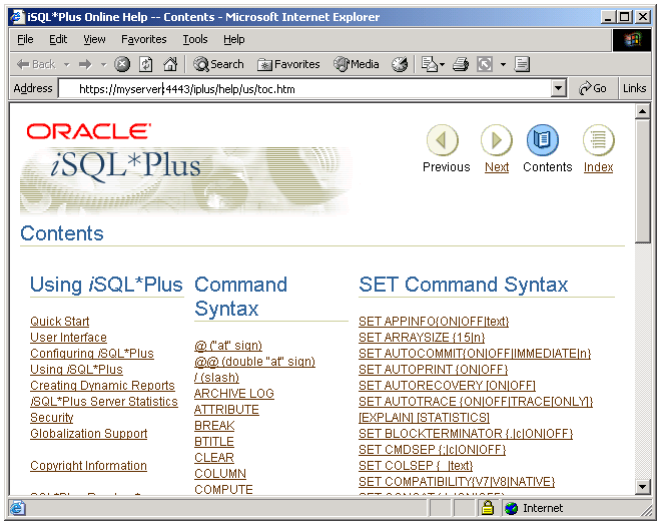
*i*SQL*Plus



ks SQL*Plus User's Guide and Reference

Oracle

*i*SQL*Plus Help



Oracle

Types of PL/SQL programs

- 4GL procedures or functions
  - client side triggers of Oracle Forms, Menu, Reports
- Anonymous PL/SQL blocks
  - 3GL embedded SQL
  - SQL*Plus scripts (host or client)
- Stored PL/SQL programs
  - Created procedures, functions, packages
  - stored in database in compiled form
- Database triggers
  - (remote operations secured by 2PC)

Oracle

Declarative part

- Variable and constant declarations
  - of Oracle data types, with NOT NULL constraint and initial value
  - PL/SQL arrays and records
- Cursor definitions
  - Declare *c1* (<parameters>) select … <parameters>;
- User-defined exceptions
  - *myException* exception;
  - *max_op_cursors* exception;
    user defined error names overriding Oracle error codes
- Pragma definitions (compiler instructions)
  - pragma exception_init('*max_op_cursors*', -1000)
  - RESTRICT_REFERENCES
  - SERIALLY_REUSABLE
  - AUTONOMOUS_TRANSACTION

Oracle

Identifiers, variables

Identifiers:

Names of PL/SQL objects: constants, variables, labels, exceptions, functions, procedures, cursors, record types, ...

Consists of letter, digits 0..9, dollar sign ($), underscore (_), number sign (#)
Cannot include whitespaces (space, tab, carriage return)
Must start with a letter A..Z
Not case-sensitive (except quoted identifiers)
Up to 30 characters long

Declaring variables:

<variable name> <datatype> [CONSTANT] [NOT NULL]
[{:= | DEFAULT} <initial value>]

Oracle

Delimiters

In addition to typical SQL delimiters

**	exponentiation
||	concatenation
:=	assignment
=	equal to
<> !=	not equal to
^= ~=	
<< >>	label
:	host variable
%	attribute indicator / anchoring %TYPE {<cursor>|<table>}%ROWTYPE <cursor>%<attribute>
@	remote database indicator
..	Range indicator
=>	named parameter assignment <parameter> => <value>

Oracle

Predefined PL/SQL Datatypes

- Number Types
  - NUMBER (fixed point 38 digits or floating point in range [1E-130 .. 1.0E126))
  - BINARY_FLOAT, BINARY_DOUBLE (IEEE comptatible)
  - BINARY_INTEGER and Subtypes
    - NATURAL, NATURALN, POSITIVE, POSITIVEN, SIGNTYPE
  - ISO compatible Subtypes
    - INT[EGER], SMALLINT, DEC[IMAL],
    - FLOAT, REAL, DOUBLE PRECISION
- Character Types
  - CHAR[ACTER], NCHAR[ACTER], VARCHAR, VARCHAR2, STRING
  - LONG, LONG RAW, RAW, ROWID, UROWID
- Boolean Types
  - BOOLEAN
- Date, Time and Interval Types
  - DATE, TIMESTAMP [WITH [LOCAL] TMEZONE]
  - INTERVAL YEAR TO MONTH, INTERVAL DAY TO SECOND

Oracle

.. Datatypes, records, arrays

- Subtypes of Predefined PL/SQL datatypes
  - SUBTYPEs
- Data type copy
  - *myvar1 employee.emp_name*%TYPE;
  - *myvar2 myvar1*%TYPE;
- Row type copy of columns and data types for a record
  - *emp_rec employee*%ROWTYPE;
  - *emp_rec.salary* := 1000;
- Single dimensional arrays
  - type *string_arr* is table of varchar2(100) indexed by binary_integer;
  - *textline string_arr* ;

Oracle

.. Records

```
TYPE  emp_rec_type  IS  RECORD (
 no       smallint,
 name  varchar2(50),
 birthdate date
) ;


emp_rec  emp_rec_type;


emp_rec.no := 100;
emp_rec.name := ‘Jones Tom’;
emp_rec.birthdate := to_date (‘1980-10-10’,’YYYY-MM-DD’);
```


Oracle

Executable part

- Assignments (:=) arithmetic expressions
- Conditional processing
  - IF … THEN … ELSE … END IF
- Unconditional branching
  - GOTO label … <<label>>
- Loop structures
  - LOOP … END LOOP
  - FOR … END LOOP
  - WHILE … END LOOP
  - Cursor loops
- Cursor control
- EXECUTE IMMEDIATE SQL-statement-string
- Raising exceptions

EXIT WHEN condition

Oracle

Native Dynamic SQL

EXECUTE IMMEDIATE <SQL statement string>
[INTO { <define variable list> | <record> | <object variable> }]
[USING [IN | OUT | INOUT <bind argument list>] ;|

Oracle

Cursor processing

Cursor status attributes:

%FOUND
%NOTFOUND
%ROWCOUNT
%ISOPEN

Examples:

if c1%found then …

if c1%isopen then

```
Declare
  cursor C1 is
    select * from T
    order by …;
  c1_rec   C1%rowtype;

Begin
    …
   open C1;
   …
   loop
      fetch  C1  into  c1_rec;
      exit when C1%notfound;
   end loop;

   total_count := C1%rowcount;
   close C1;
```

Oracle

Cursor FOR loop

```
FOR <record> IN
        { <cursor name> | (SELECT … ) }
LOOP
        <executable statements>
END LOOP ;
```

<record> is defined automatically as <cursor name>%ROWTYPE
FOR loop does automatically cursor open, fetch, and close operations
Example:

```
FOR emp_rec IN SELECT * FROM emp
LOOP
    IF emp_rec.dept = 'D123' THEN
        salary_raise(emp_rec.emp_id, 15) ;
    END IF ;
END LOOP ;
```


Oracle

Exception Handling

```
BEGIN
        …
EXCEPTION
When <exception name> then
        <statements>
...
When OTHERS then
        err_code := sqlcode;
        err_text := sqlerrm;
        insert into program_erors
            (error_code, error_text, error_time)
        values (err_code, err_text, sysdate);
END;
```

After exception processing the PL/SQL block is ended and control returns to the calling module or next statement after the END in case of local exceptions of a BEGIN-END block.

Oracle

Raising Exceptions

Exception can be raised by

- PL/SQL engine
- RAISE statement

  RAISE <exception name>
- calling RAISE_APPLICATION_ERROR

Exception handler can process and then re-raise the current exception by

RAISE ;

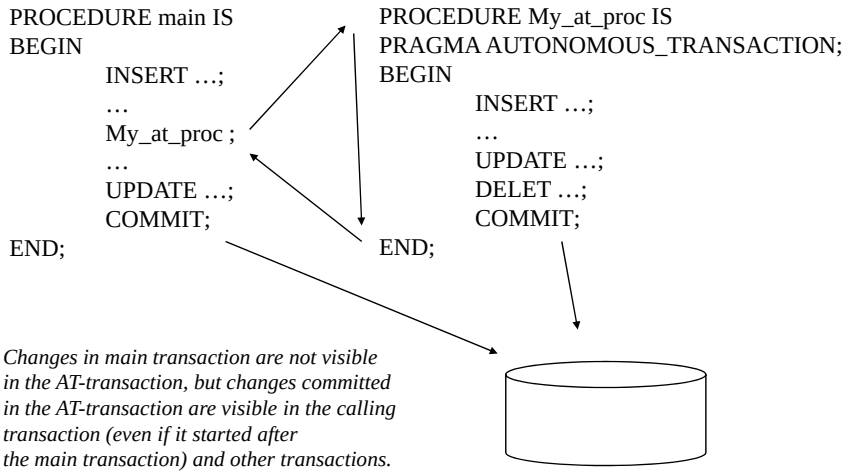
Oracle

Exceptions with PL/SQL mnemonics

Exception name:	for error code:
• Cursor_already_open	ORA-06511
• dup_val_on_index	ORA-00001
• invalid_cursor	ORA-01001
• invalid_number	ORA-01722
• login_defined	ORA-01017
• no_data_found	ORA-01403
• not_logged_on	ORA-01012
• program_error	ORA-06501
• storage_error	ORA-06500
• timeout_on_resource	ORA-00051
• too_many_rows	ORA-01422
• transaction_backed_out	ORA-00061
• value_error	ORA-06502
• zero_divide	ORA-01476

Oracle

Autonomous Transactions

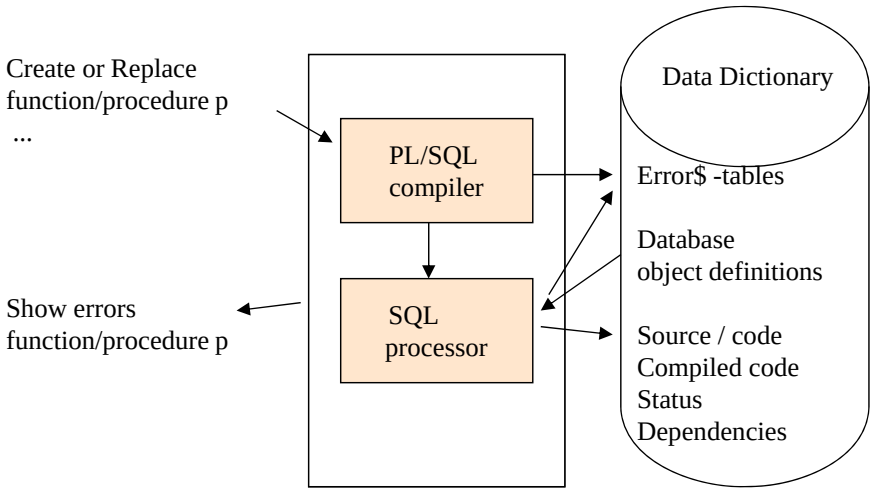


Changes in main transaction are not visible in the AT-transaction, but changes committed in the AT-transaction are visible in the calling transaction (even if it started after the main transaction) and other transactions.

Helia / Martti Laiho 1998-2006

Oracle

Developing a stand-alone procedure



Helia / Martti Laiho 1998-2006

Oracle

Create Function

```
SQL >   CREATE  OR  REPLACE  FUNCTION <fname>
                (p1 <datatype>,  p2<datatype>, … )
        RETURN  <datatype>
        IS  <PL/SQL block>
```

The PL/SQL code is compiled by PL/SQL compiler
parsed by SQL-processor and stored in the data dictionary

Any errors are stored in the data dictionary table Error$
to accessed by views USER_ERRORS, ALL_ERRORS or DBA_ERRORS
or
SQL> SHOW ERRORS FUNCTION <fname>

Oracle

An example function

```
SQL> CREATE OR REPLACE FUNCTION DayOfWeek (indate IN DATE)
  2  RETURN SMALLINT
  3  IS BEGIN
  4           RETURN TO_NUMBER(TO_CHAR(indate,'D'));
  5  END;
  6  /

Function created.

SQL> SELECT DayOfWeek(SYSDATE) FROM DUAL;

DAYOFWEEK(SYSDATE)
------------------
                 3
```

Note: DUAL is a virtual single line table which is typically used for reporting current system data

Oracle

Create Procedure

SQL:

```
CREATE  OR  REPLACE  PROCEDURE <pname>
          (p1  IN <datatype>, p2 OUT  <datatype>, … )
IS
 <PL/SQL block>
```

```
SQL> SHOW  ERRORS
or
SQL> SHOW  ERRORS  PROCEDURE <pname>
or
SQL> select * from user_errors  where name = '<pname>';
```


Oracle

Create Trigger

```
CREATE OR REPLACE  <trigger name>
{ BEFORE | AFTER | INSTEAD OF}
{ {<trigger event> ON
     { NESTED TABLE <nested_table_column> OF <view>
       | <table> | <view>}
     [ <referencing clause> ]
     [FOR EACH ROW [WHEN <trigger condition> ] ] }
 | <trigger event> ON DATABASE
}
<trigger body> ;
```

INSERT
UPDATE
DELETE

CREATE| ALTER|DROP
LOGON | LOGOF
START | SHUTDOWN
SERVERERROR

Oracle

PL/SQL Packages

- Encapsulation of modules
- Administration of procedures
- Managing access privileges
- Declaring global variables and constants
  - available in the same Oracle session
- Hiding private procedures, ..
- Overloading

Oracle

Creating PL/SQL Package

Specification part declaring the visible objects

```
Create or Replace Package pack1 as
        procedure p1 (a in number, …);
        function  f1 (af  in number) return number;
        var_1 …;
        my_ex  exception;
        cursor c1;
end pack1;
```

Creating the code and hidden objects

Initialization block to be executed on the first call by a new user →

```
Create or Replace Package Body pack1 as
        procedure p1 (a in number, …);
                <code>
        function  f1 (af  in number) return number;
                <code>
        procedure p_private ( …);
                <code>
begin
        var_1 :=  <value>;
         . . .
end pack1;
```

Oracle

Oracle Builtin Packages

- Feuerstein 96
- PL/SQL packages

- DBMS_ALERT Notifications of database events
- DBMS_DDL Interface to some SQL DDL statements
- DBMS_JOB Automatic submitting of scheduled database tasks
- DBMS_LOCK Interface for user defined locks using OLM
- DBMS_MAIL Interface to Oracle Mail (Oracle <= 9)
- DBMS_OUTPUT Messaging from PL/SQL programs
- DBMS_PIPE ShMem messaging between Oracle sessions
- DBMS_SESSION Interface to session level commands
- DBMS_SQL Dynamic SQL from PL/SQL
- DBMS_TRANSACTION Interface for transaction options
- DBMS_UTILITY Miscellaneous PL/SQL routines

Helia / Martti Laiho 1998-2006

Oracle

.. Oracle Builtin Packages

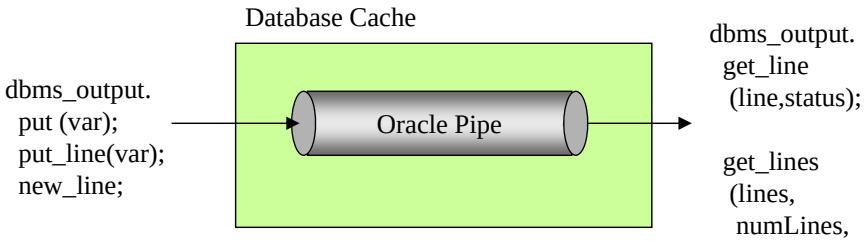
- DBMS_STREAMS
- DBMS_XDB
- DBMS_XML...
- DBMS_RLS
- ...
- HTF functions generating HTML tags
- HTP procedures generating HTML tags
- OWA_... Oracle Web API, cookies etc
- SDO_... Spatial Data Objects ?
- UTL_FILE reading and writing op-sys files
- UTL_HTTP communicating with Web servers
- UTL_SMTP communicating with mail servers

Helia / Martti Laiho 1998-2006

Oracle

Debugging with DBMS_OUTPUT

dbms_output.enable (<buffersize>);



dbms_output.disable;

SQL> set serveroutput on
SQL> execute <proc> ;
… <lines displayed on the terminal>

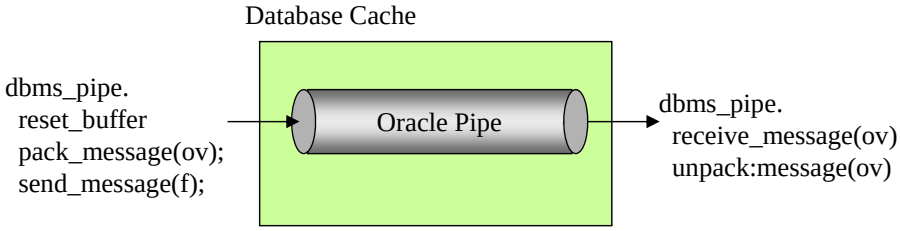
Helia / Martti Laiho 1998-2006

Oracle

DBMS_PIPE

Oracle session A

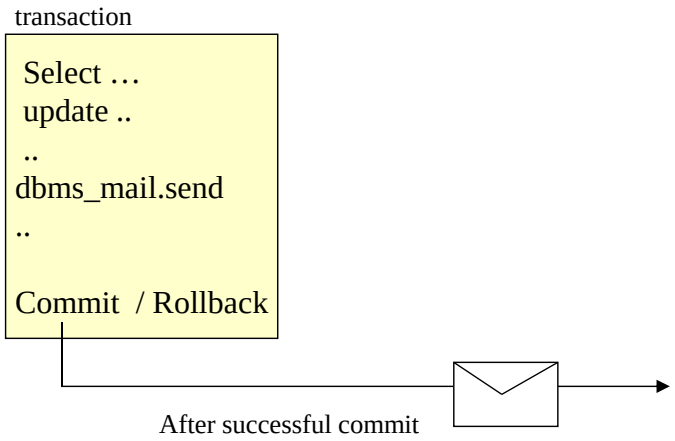
Oracle session B



Helia / Martti Laiho 1998-2006

Oracle

DBMS_MAIL



Oracle

DBMS_LOCK

Package for explicit locking (by integer values)
user defined locks between 1 - 1073741823

Allocate_unique - name associated for integers 1073741824-

Request - request a lock with a given mode

Convert - convert the mode of a lock

Sleep (secs) - suspend the session for a given period of time

Release - releasing the lock