1. RAW

The RAW datatype is used to store strings of byte-oriented data.

The difference between a RAW and a VARCHAR2 string is that Oracle does no character set translation on raw data.

Thus if you are retrieving raw data from an Oracle server using ASCII to a machine using the EBCDIC character set, no translation would be done.

The Syntax for the RAW Datatype

```
variable_name RAW(size);
```

size is the size, in bytes, of the variable.

size must be a number between 1 and 32767.

Here are some sample declarations:

```
some_data RAW(255);
```

Like VARCHAR2, the maximum length of a RAW variable is 32767 bytes.

2. LONG RAW

The LONG RAW datatype is just like RAW.

The maximum length of the LONG RAW datatype is 32760 bytes.

The Syntax for the LONG RAW Datatype

```
variable_name LONG RAW(size);
```

size is the size, in bytes, of the variable.
size must be a number between 1 and 32760.
Here are some examples:

```
a_picture LONG RAW(30000);
```

As with a RAW, no character set conversion is performed.
The database version of LONG RAW allows up to 2 gigabytes of data.

26. 1. ROWID

ROWID is a special datatype that enables you to store Oracle's internal key for database records.
Each row in an Oracle table has a unique internal key associated with it.
For example, you might use a cursor to retrieve a number of rows, including the ROWID for each row, and then use that ROWID in a DELETE statement to delete some of the rows.
Using the ROWID results in better performance because it tells Oracle exactly where to find the record.
The Syntax for the ROWID Datatype

```
variable_name ROWID;
employee_row_id ROWID;
```

27. 1. Declare ROWTYPE variable

```
SQL>
SQL>
```

```
SQL>
SQL> -- create demo table
SQL> create table Employee(
  2    ID                  VARCHAR2(4 BYTE)         NOT NULL primary key,
  3    First_Name          VARCHAR2(10 BYTE),
  4    Last_Name           VARCHAR2(10 BYTE),
  5    Start_Date          DATE,
  6    End_Date            DATE,
  7    Salary              Number(8,2),
  8    City                VARCHAR2(10 BYTE),
  9    Description         VARCHAR2(15 BYTE)
 10  )
 11  /

Table created.

SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',   'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('03','James',    'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /
```

```
1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('04','Celia',    'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',   'Black',   to_date('19840115','YYYYMMDD'), to_date('19980808','Y
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('06','Linda',    'Green',   to_date('19870730','YYYYMMDD'), to_date('19960104','Y
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('07','David',    'Larry',   to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('08','James',    'Cat',     to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /
```

```
1 row created.

SQL>
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /


ID   FIRST_NAME LAST_NAME  START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- ---------- ---------- --------- --------- ---------- ---------- ---------------
01   Jason      Martin     25-JUL-96 25-JUL-06    1234.56 Toronto    Programmer
02   Alison     Mathews    21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James      Smith      12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia      Rice       24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert     Black      15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda      Green      30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David      Larry      31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James      Cat        17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.

SQL>
SQL> CREATE OR REPLACE PROCEDURE update_emp (emp_rec employee%ROWTYPE) IS
  2  BEGIN
  3       UPDATE employee
  4       SET    start_date = emp_rec.start_date + 100
  5       WHERE id = emp_rec.id;
  6  END update_emp;
  7  /

Procedure created.

SQL>
SQL>
SQL> DECLARE
  2       a employee%ROWTYPE;
  3  BEGIN
  4       a.id := '01';
```

```
  5       update_emp(a);
  6  END;
  7  /

PL/SQL procedure successfully completed.

SQL> SELECT * FROM employee;


ID   FIRST_NAME LAST_NAME  START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- ---------- ---------- --------- --------- ---------- ---------- ---------------
01   Jason      Martin               25-JUL-06    1234.56 Toronto    Programmer
02   Alison     Mathews    21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James      Smith      12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia      Rice       24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert     Black      15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda      Green      30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David      Larry      31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James      Cat        17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.
```

27. 2. Reference attribute in a ROWTYPE variable

```
SQL>
SQL>
SQL>
SQL> -- create demo table
SQL> create table Employee(
  2    ID                 VARCHAR2(4 BYTE)         NOT NULL primary key,
  3    First_Name         VARCHAR2(10 BYTE),
  4    Last_Name          VARCHAR2(10 BYTE),
  5    Start_Date         DATE,
  6    End_Date           DATE,
  7    Salary             Number(8,2),
```

```
  8    City                VARCHAR2(10 BYTE),
  9    Description         VARCHAR2(15 BYTE)
 10  )
 11  /

Table created.

SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',   'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('03','James',    'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('04','Celia',    'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /

1 row created.
```

```
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',    'Black',    to_date('19840115','YYYYMMDD'), to_date('19980808','Y
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('06','Linda',     'Green',    to_date('19870730','YYYYMMDD'), to_date('19960104','Y
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('07','David',     'Larry',    to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('08','James',     'Cat',      to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /

1 row created.

SQL>
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /
```

```
ID   FIRST_NAME LAST_NAME  START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- ---------- ---------- --------- --------- ---------- ---------- ---------------
01   Jason      Martin     25-JUL-96 25-JUL-06    1234.56 Toronto    Programmer
02   Alison     Mathews    21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James      Smith      12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia      Rice       24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert     Black      15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda      Green      30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David      Larry      31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James      Cat        17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.

SQL>
SQL> CREATE OR REPLACE PROCEDURE update_emp (emp_rec employee%ROWTYPE) IS
  2  BEGIN
  3       UPDATE employee
  4       SET    start_date = emp_rec.start_date + 100
  5       WHERE id = emp_rec.id;
  6  END update_emp;
  7  /

Procedure created.

SQL>
SQL>
SQL> DECLARE
  2       a employee%ROWTYPE;
  3  BEGIN
  4       a.id := '01';
  5       update_emp(a);
  6  END;
  7  /

PL/SQL procedure successfully completed.

SQL> SELECT * FROM employee;
```

```
ID   FIRST_NAME LAST_NAME  START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- ---------- ---------- --------- --------- ---------- ---------- ---------------
01   Jason      Martin               25-JUL-06    1234.56 Toronto    Programmer
02   Alison     Mathews    21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James      Smith      12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia      Rice       24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert     Black      15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda      Green      30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David      Larry      31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James      Cat        17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.
```

27. 3. Select value into the %ROWTYPE type variable

```
SQL>
SQL>
SQL> Using the %ROWTYPE Clause on the Table Name
SQL>
SQL>
SQL> -- create demo table
SQL> create table Employee(
  2    ID                 VARCHAR2(4 BYTE)         NOT NULL,
  3    First_Name         VARCHAR2(10 BYTE),
  4    Last_Name          VARCHAR2(10 BYTE),
  5    Start_Date         DATE,
  6    End_Date           DATE,
  7    Salary             Number(8,2),
  8    City               VARCHAR2(10 BYTE),
  9    Description        VARCHAR2(15 BYTE)
 10  )
 11  /

Table created.
```

```
SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',   'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('03','James',    'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('04','Celia',    'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',   'Black',   to_date('19840115','YYYYMMDD'), to_date('19980808','Y
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /
```

```
1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('06','Linda',    'Green',   to_date('19870730','YYYYMMDD'), to_date('19960104','Y
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('07','David',    'Larry',   to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('08','James',    'Cat',     to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /

1 row created.

SQL>
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /

ID   FIRST_NAME           LAST_NAME            START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- -------------------- -------------------- --------- --------- ---------- ---------- ---------------
01   Jason                Martin               25-JUL-96 25-JUL-06    1234.56 Toronto    Programmer
02   Alison               Mathews              21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James                Smith                12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia                Rice                 24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert               Black                15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
```

```
06   Linda                Green                30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David                Larry                31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James                Cat                  17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.

SQL>
SQL>
SQL> declare
  2      r_emp employee%ROWTYPE;
  3  begin
  4      select * into r_emp from employee where id = '01';
  5      DBMS_OUTPUT.put_line(r_emp.first_Name);
  6  end;
  7  /
Jason

PL/SQL procedure successfully completed.
```

27. 4. Using a Weak REF CURSOR and %ROWTYPE

```
SQL> --
SQL>
SQL> -- create demo table
SQL> create table Employee(
  2    ID                  VARCHAR2(4 BYTE)         NOT NULL,
  3    First_Name          VARCHAR2(10 BYTE),
  4    Last_Name           VARCHAR2(10 BYTE),
  5    Start_Date          DATE,
  6    End_Date            DATE,
  7    Salary              Number(8,2),
  8    City                VARCHAR2(10 BYTE),
  9    Description         VARCHAR2(15 BYTE)
 10  )
 11  /
```

```
Table created.

SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',   'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('03','James',    'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('04','Celia',    'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',   'Black',   to_date('19840115','YYYYMMDD'), to_date('19980808','Y
```

```
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('06','Linda',    'Green',   to_date('19870730','YYYYMMDD'), to_date('19960104','Y
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('07','David',    'Larry',   to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('08','James',    'Cat',     to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /

1 row created.

SQL>
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /

ID   FIRST_NAME LAST_NAME  START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- ---------- ---------- --------- --------- ---------- ---------- ---------------
01   Jason      Martin     25-JUL-96 25-JUL-06    1234.56 Toronto    Programmer
02   Alison     Mathews    21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
```

```
03   James      Smith      12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia      Rice       24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert     Black      15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda      Green      30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David      Larry      31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James      Cat        17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.

SQL>
SQL>
SQL>
SQL> create or replace procedure p_report(i_where_tx VARCHAR2)
  2  is
  3      c_emp_ref sys_refcursor;
  4      v_emp_rec employee%ROWTYPE;
  5  begin
  6      DBMS_OUTPUT.put_line('-----------------------');
  7      open c_emp_ref for 'select * from employee where ' ||i_where_tx;
  8      loop
  9          fetch c_emp_ref into v_emp_rec;
 10          exit when c_emp_ref%notfound;
 11          DBMS_OUTPUT.put_line(v_emp_rec.id||' '|| v_emp_rec.first_Name||' - '|| v_emp_rec.salary);
 12      end loop;
 13      DBMS_OUTPUT.put_line('----------------------');
 14  end p_report;
 15  /

Procedure created.

SQL> begin
  2         p_report('salary > 2000');
  3  end;
  4  /

PL/SQL procedure successfully completed.
```

27. 5. Defining and using a cursor-oriented record

```
SQL>
SQL>
SQL> create table product(
  2     product_id number(4)     not null,
  3     product_description varchar2(20) not null
  4  );

Table created.

SQL>
SQL> insert into product values (1,'Java');

1 row created.

SQL> insert into product values (2,'Oracle');

1 row created.

SQL> insert into product values (3,'C#');

1 row created.

SQL> insert into product values (4,'Javascript');

1 row created.

SQL> insert into product values (5,'Python');

1 row created.

SQL>
SQL>
SQL>
SQL>
SQL> DECLARE
  2    CURSOR csr_hrc IS
  3      SELECT * FROM product ORDER BY 1;
  4    myRecord csr_hrc%ROWTYPE;
  5  BEGIN
  6    OPEN csr_hrc;
```

```
  7
  8    LOOP
  9      FETCH csr_hrc INTO myRecord;
 10      EXIT WHEN csr_hrc%NOTFOUND;
 11      dbms_output.put_line(to_char(myRecord.product_id)||' '||myRecord.product_description);
 12    END LOOP;
 13    CLOSE csr_hrc;
 14  END;
 15  /
1 Java
2 Oracle
3 C#
4 Javascript
5 Python

PL/SQL procedure successfully completed.
```

27. 6. INSERT statement involving entire records

```
SQL>
SQL>
SQL> create table product(
  2     product_id number(4)     not null,
  3     product_description varchar2(20) not null
  4  );

Table created.

SQL>
SQL> insert into product values (1,'Java');

1 row created.

SQL> insert into product values (2,'Oracle');
```

```
1 row created.

SQL> insert into product values (3,'C#');

1 row created.

SQL> insert into product values (4,'Javascript');

1 row created.

SQL> insert into product values (5,'Python');

1 row created.

SQL>
SQL>
SQL>
SQL>
SQL>
SQL> DECLARE
  2    TYPE myRecord IS RECORD
  3      (product_id NUMBER,
  4       product_description VARCHAR2(20));
  5    v_example_rec myRecord;
  6  BEGIN
  7    v_example_rec.product_id :=99;
  8    v_example_rec.product_description :='Web Analyst';
  9    INSERT INTO product VALUES v_example_rec;
 10    COMMIT;
 11  END;
 12  /

PL/SQL procedure successfully completed.

SQL>
```

27. 7. UPDATE statement involving entire records

```
SQL>
SQL>
SQL>
SQL> create table product(
  2     product_id number(4)     not null,
  3     product_description varchar2(20) not null
  4  );

Table created.

SQL>
SQL> insert into product values (1,'Java');

1 row created.

SQL> insert into product values (2,'Oracle');

1 row created.

SQL> insert into product values (3,'C#');

1 row created.

SQL> insert into product values (4,'Javascript');

1 row created.

SQL> insert into product values (5,'Python');

1 row created.

SQL>
SQL>
SQL>
SQL> DECLARE
  2    TYPE myRecord IS RECORD
  3      (product_id NUMBER,
  4       product_description VARCHAR2(20));
```

```
  5    v_example_rec myRecord;
  6  BEGIN
  7    v_example_rec.product_id :=99;
  8    v_example_rec.product_description :='Web';
  9    UPDATE product SET ROW =v_example_rec WHERE product_id =99;
 10    COMMIT;
 11  END;
 12  /

PL/SQL procedure successfully completed.
```

27. 8. Use rowtype with object table

```
SQL>
SQL> CREATE OR REPLACE TYPE address AS OBJECT
  2              (line1 VARCHAR2(20),
  3               line2 VARCHAR2(20),
  4               city VARCHAR2(20),
  5               state_code VARCHAR2(2),
  6               zip VARCHAR2(13),
  7    MEMBER FUNCTION get_address RETURN VARCHAR2,
  8    MEMBER PROCEDURE set_address
  9              (addressLine1 VARCHAR2,
 10               addressLine2 VARCHAR2,
 11               address_city VARCHAR2,
 12               address_state VARCHAR2,
 13               address_zip VARCHAR2)
 14  );
 15  /

Type created.

SQL> CREATE OR REPLACE TYPE BODY address AS
  2    MEMBER FUNCTION get_address RETURN VARCHAR2
  3    IS
```

```
  4    BEGIN
  5      RETURN (SELF.line1||' '||SELF.line2||' '||SELF.city||', '||SELF.state_code||' '||SELF.zip);
  6    END get_address;
  7    MEMBER PROCEDURE set_address (addressLine1 VARCHAR2,
  8                  addressLine2 VARCHAR2,
  9                  address_city VARCHAR2,
 10                  address_state VARCHAR2,
 11                  address_zip VARCHAR2)
 12    IS
 13    BEGIN
 14      line1 :=addressLine1;
 15      line2 :=addressLine2;
 16      city :=address_city;
 17      state_code :=address_state;
 18      zip :=address_zip;
 19    END set_address;
 20  END;
 21  /

Type body created.

SQL>
SQL> CREATE TABLE address_master OF address;

Table created.

SQL>
SQL> INSERT INTO address_master VALUES (address('19 J','Reading Rd','Vancouver','NJ','00000'));

1 row created.

SQL>
SQL> select * from address_master;

LINE1                LINE2                CITY                 ST
-------------------- -------------------- -------------------- --
ZIP
-------------
19 J                 Reading Rd           Vancouver            NJ
00000
```

```
1 row selected.

SQL>
SQL> declare
  2    cursor c1 is select VALUE(a)from address_master a;
  3    v_add address_master%ROWTYPE;
  4  begin
  5    open c1;
  6    loop
  7      fetch c1 into v_add;
  8      exit when c1%notfound;
  9      dbms_output.put_line(v_add.line1);
 10    end loop;
 11    close c1;
 12  end;
 13  /
19 J

PL/SQL procedure successfully completed.
```

27. 9. rowtype.iterations

```
SQL>
SQL> create table job_parameters
  2  ( jobid number primary key,
  3    iterations number,
  4    table_idx number );

Table created.

SQL>
SQL>
SQL> create or replace procedure bind( p_job in number )
```

```
  2  as
  3      l_rec job_parameters%rowtype;
  4  begin
  5      select * into l_rec from job_parameters where jobid = p_job;
  6      for i in 1 .. l_rec.iterations
  7      loop
  8          execute immediate
  9          'insert into t' || l_rec.table_idx || ' values ( :x )' using i;
 10          commit;
 11      end loop;
 12      delete from job_parameters where jobid = p_job;
 13  end;
 14  /

Procedure created.
```

27. 10. Use table column type as the record attribute type

```
SQL>
SQL> CREATE TABLE employee
  2  (employee_id         NUMBER(7),
  3   last_name           VARCHAR2(25),
  4   first_name          VARCHAR2(25),
  5   userid              VARCHAR2(8),
  6   start_date          DATE,
  7   comments            VARCHAR2(255),
  8   manager_id          NUMBER(7),
  9   title               VARCHAR2(25),
 10   department_id       NUMBER(7),
 11   salary              NUMBER(11, 2),
 12   commission_pct      NUMBER(4, 2)
 13  );

Table created.
```

```
SQL>
SQL> INSERT INTO employee VALUES (1, 'V', 'Ben', 'cv',to_date('03-MAR-90 8:30', 'dd-mon-
yy hh24:mi'),NULL, NULL, 'PRESIDENT', 50, 2500, NULL);

1 row created.

SQL> INSERT INTO employee VALUES (2, 'N', 'Haidy', 'ln', '08-MAR-
90', NULL,1, 'VP, OPERATIONS', 41, 1450, NULL);

1 row created.

SQL> INSERT INTO employee VALUES (3, 'N', 'Molly', 'mn', '17-JUN-91',NULL, 1, 'VP, SALES', 31, 1400, NULL);

1 row created.

SQL> INSERT INTO employee VALUES (4, 'S', 'Mark', 'mq', '07-APR-
90',NULL, 1, 'VP, FINANCE', 10, 1450, NULL);

1 row created.

SQL> INSERT INTO employee VALUES (5, 'R', 'AUDRY', 'ar', '04-MAR-
90',NULL, 1, 'VP, ADMINISTRATION', 50, 1550, NULL);

1 row created.

SQL> INSERT INTO employee VALUES (6, 'U', 'MOLLY', 'mu', '18-JAN-
91',NULL, 2, 'WAREHOUSE MANAGER', 41, 1200, NULL);

1 row created.

SQL> INSERT INTO employee VALUES (7, 'M', 'ROBERTA', 'rm', '14-MAY-
90',NULL, 2, 'WAREHOUSE MANAGER', 41, 1250, NULL);

1 row created.

SQL> INSERT INTO employee VALUES (8, 'B', 'BEN', 'ry', '07-APR-
90', NULL, 2,'WAREHOUSE MANAGER', 41, 1100, NULL);

1 row created.
```

```
SQL> INSERT INTO employee VALUES (9, 'C', 'Jane', 'ac', '09-FEB-
92',NULL, 2, 'WAREHOUSE MANAGER', 41, 1300, NULL);

1 row created.

SQL> INSERT INTO employee VALUES (10, 'H', 'Mart', 'mh', '27-FEB-
91', NULL, 2,'WAREHOUSE MANAGER', 41, 1307, NULL);

1 row created.

SQL>
SQL>
SQL>
SQL> DECLARE
  2     TYPE type_emp_comp_rec IS RECORD
  3        (emp_last_name_txt      employee.last_name%TYPE,
  4         emp_first_name_txt     employee.first_name%TYPE,
  5         emp_salary_num         employee.salary%TYPE,
  6         emp_commission_pct_num employee.commission_pct%TYPE );
  7     lv_emp_comp_rec1 type_emp_comp_rec;
  8     lv_emp_comp_rec2 type_emp_comp_rec;
  9  BEGIN
 10     lv_emp_comp_rec1.emp_last_name_txt  := 'TREZZO';
 11     lv_emp_comp_rec2.emp_first_name_txt := 'JOE';
 12  END;
 13  /

PL/SQL procedure successfully completed.
```

27. 11. Select * into table%rowtype

```
SQL>
SQL> create table T (
  2   c1 number,
  3   c2 number );
```

```
Table created.

SQL>
SQL> insert into T values (1,2);

1 row created.

SQL>
SQL> create or replace procedure WITH_ROWTYPE is
  2     r T%ROWTYPE;
  3  begin
  4     select * into r from T where rownum = 1;
  5  end;
  6  /

Procedure created.
```

28. 1. Variables Based on Database Columns

You base a declaration on a table column without having to know definitively what that column's datatype is.

Your variables will automatically be changed as the table column.

You use the %type attribute in variable declarations to define variable based on table column.

```
avg_rate    employee.pay_rate%type</th>
dept_id     INTEGER;</td>
sub_dept_id dept_id%type;  -- datatype based on a variable</td>
area_id     dept_id%type := 9141; -- used with an initialization clause</td>
```

28. 2. PL/SQL allows you to use the %type attribute in a nesting variable declaration.

```
dept_sales    INTEGER;
area_sales    dept_sales%type;
group_sales     area_sales%type;
regional_sales  area_sales%type;
corporate_sales  regional_sales%type;
```

28. 4. The %TYPE Command Illustrated

```
SQL>
SQL>
SQL>
SQL> -- create demo table
SQL> create table Employee(
  2    ID                  VARCHAR2(4 BYTE)         NOT NULL primary key,
  3    First_Name          VARCHAR2(10 BYTE),
  4    Last_Name           VARCHAR2(10 BYTE),
  5    Start_Date          DATE,
  6    End_Date            DATE,
  7    Salary              Number(8,2),
  8    City                VARCHAR2(10 BYTE),
  9    Description         VARCHAR2(15 BYTE)
 10  )
 11  /

Table created.

SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.
```

```
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',    'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('03','James',     'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('04','Celia',     'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',    'Black',   to_date('19840115','YYYYMMDD'), to_date('19980808','Y
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary, City,        Description)
  2                values('06','Linda',     'Green',   to_date('19870730','YYYYMMDD'), to_date('19960104','Y
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.
```

```
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary, City,        Description)
  2                values('07','David',    'Larry',   to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary, City,        Description)
  2                values('08','James',    'Cat',     to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /

1 row created.

SQL>
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /

ID   FIRST_NAME           LAST_NAME            START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- -------------------- -------------------- --------- --------- ---------- ---------- ---------------
01   Jason                Martin               25-JUL-96 25-JUL-06    1234.56 Toronto    Programmer
02   Alison               Mathews              21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James                Smith                12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia                Rice                 24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert               Black                15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda                Green                30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David                Larry                31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James                Cat                  17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.

SQL>
SQL>
SQL>
SQL> declare
```

```
  2    v_ename employee.first_Name%TYPE;
  3    v_emp_rec employee%ROWTYPE;
  4  begin
  5    v_ename := 'Smith';
  6    v_emp_rec.first_Name := 'Chan';
  7  end;
  8  /

PL/SQL procedure successfully completed.
```

28. 5. Using %TYPE and %ROWTYPE on Row Objects

```
SQL>
SQL> CREATE OR REPLACE TYPE address AS OBJECT
  2              (line1 VARCHAR2(20),
  3               line2 VARCHAR2(20),
  4               city VARCHAR2(20),
  5               state_code VARCHAR2(2),
  6               zip VARCHAR2(13),
  7    MEMBER FUNCTION get_address RETURN VARCHAR2,
  8    MEMBER PROCEDURE set_address
  9              (addressLine1 VARCHAR2,
 10               addressLine2 VARCHAR2,
 11               address_city VARCHAR2,
 12               address_state VARCHAR2,
 13               address_zip VARCHAR2)
 14  );
 15  /

Type created.

SQL> CREATE OR REPLACE TYPE BODY address AS
  2    MEMBER FUNCTION get_address RETURN VARCHAR2
  3    IS
  4    BEGIN
  5      RETURN (SELF.line1||' '||SELF.line2||' '||SELF.city||', '||SELF.state_code||' '||SELF.zip);
  6    END get_address;
  7    MEMBER PROCEDURE set_address (addressLine1 VARCHAR2,
```

```
  8                  addressLine2 VARCHAR2,
  9                  address_city VARCHAR2,
 10                  address_state VARCHAR2,
 11                  address_zip VARCHAR2)
 12    IS
 13    BEGIN
 14      line1 :=addressLine1;
 15      line2 :=addressLine2;
 16      city :=address_city;
 17      state_code :=address_state;
 18      zip :=address_zip;
 19    END set_address;
 20  END;
 21  /

Type body created.

SQL>
SQL> CREATE TABLE address_master OF address;

Table created.

SQL>
SQL> INSERT INTO address_master VALUES (address('19 J','Reading Rd','Vancouver','NJ','00000'));

1 row created.

SQL>
SQL> select * from address_master;

LINE1                LINE2                CITY                 ST
-------------------- -------------------- -------------------- --
ZIP
-------------
19 J                 Reading Rd           Vancouver            NJ
00000


1 row selected.
```

```
SQL>
SQL> DECLARE
  2    v_line1 address_master.line1%TYPE;
  3  BEGIN
  4    SELECT line1 INTO v_line1 FROM address_master WHERE city ='Vancouver';
  5    DBMS_OUTPUT.PUT_LINE(v_line1);
  6  END;
  7  /
19 J

PL/SQL procedure successfully completed.
```

28. 6. Column type parameters

```
SQL>
SQL> CREATE TABLE employee_locker (
  2    emp_id        NUMBER        NOT NULL  PRIMARY KEY,
  3    name          VARCHAR2(30)  NOT NULL,
  4    room_number   VARCHAR2(30)  NOT NULL,
  5    occupied_dt   DATE,
  6    checkout_date DATE );

Table created.

SQL>
SQL> INSERT INTO employee_locker( emp_id, name, room_number, occupied_dt, checkout_date )VALUES( 1, 'Java',
 '10A', TRUNC(SYSDATE), TRUNC(SYSDATE) );

1 row created.

SQL> INSERT INTO employee_locker( emp_id, name, room_number, occupied_dt )VALUES( 2, 'SQL', '12A', TRUNC(SY
SDATE) );

1 row created.

SQL> INSERT INTO employee_locker( emp_id, name, room_number, occupied_dt, checkout_date )VALUES( 3, 'Oracle
', '12B', TRUNC(SYSDATE), TRUNC(SYSDATE) );
```

```
1 row created.

SQL> INSERT INTO employee_locker( emp_id, name, room_number, occupied_dt )VALUES( 4, 'PC', '10A', TRUNC(SYS
DATE+1) );

1 row created.

SQL> INSERT INTO employee_locker( emp_id, name, room_number, occupied_dt )VALUES( 5, 'JavaScript', '12A', T
RUNC(SYSDATE+1) );

1 row created.

SQL> INSERT INTO employee_locker( emp_id, name, room_number, occupied_dt )VALUES( 6, 'C', '12B', TRUNC(SYSD
ATE+1) );

1 row created.

SQL> INSERT INTO employee_locker( emp_id, name, room_number, occupied_dt )VALUES( 7, 'C++', '10A', TRUNC(SY
SDATE+2) );

1 row created.

SQL> INSERT INTO employee_locker( emp_id, name, room_number, occupied_dt )VALUES( 8, 'Python', '12A', TRUNC
(SYSDATE+2) );

1 row created.

SQL> INSERT INTO employee_locker( emp_id, name, room_number, occupied_dt )VALUES( 9, 'C#', '12B', TRUNC(SYS
DATE+2) );

1 row created.

SQL>
SQL>
SQL>
SQL> CREATE OR REPLACE PROCEDURE update_bill(
  2    emp_id_in  employee_locker.emp_id%TYPE,
  3    room_number_in  employee_locker.room_number%TYPE )
  4  IS
  5  BEGIN
```

```
  6    dbms_output.put_line('bill updated for emp_id = '||emp_id_in||', room_number = '||room_number_in);
  7  END update_bill;
  8  /

Procedure created.

SQL>
SQL> set serveroutput on size 500000
SQL>
SQL> DECLARE
  2     CURSOR employee_locker_cur IS
  3        SELECT emp_id, room_number
  4        FROM employee_locker WHERE occupied_dt = TRUNC(SYSDATE);
  5     employee_locker_rec employee_locker_cur%ROWTYPE;
  6  BEGIN
  7     OPEN employee_locker_cur;
  8     LOOP
  9        FETCH employee_locker_cur INTO employee_locker_rec;
 10        EXIT WHEN employee_locker_cur%NOTFOUND;
 11        update_bill (employee_locker_rec.emp_id, employee_locker_rec.room_number);
 12     END LOOP;
 13     CLOSE employee_locker_cur;
 14  END;
 15  /
bill updated for emp_id = 1, room_number = 10A
bill updated for emp_id = 2, room_number = 12A
bill updated for emp_id = 3, room_number = 12B

PL/SQL procedure successfully completed.
```

28. 7. Select value into a column type variable

```
SQL>
SQL> CREATE TABLE accounts(
  2    account_id  NUMBER  NOT NULL  PRIMARY KEY,
  3    balance  NUMBER );

Table created.
```

```
SQL>
SQL> INSERT INTO accounts(account_id,balance )VALUES(1,1000 );

1 row created.

SQL> INSERT INTO accounts(account_id,balance )VALUES(2,800 );

1 row created.

SQL>
SQL>
SQL> CREATE OR REPLACE FUNCTION account_balance(
  2    account_id_in IN accounts.account_id%TYPE)
  3  RETURN accounts.balance%TYPE
  4  IS
  5    l_balance  accounts.balance%TYPE;
  6  BEGIN
  7    SELECT balance
  8      INTO l_balance
  9      FROM accounts
 10     WHERE account_id = account_id_in;
 11
 12    RETURN l_balance;
 13
 14  END account_balance;
 15  /

Function created.

SQL>
SQL> CREATE OR REPLACE PROCEDURE apply_balance(
  2    account_id_in IN accounts.balance%TYPE,
  3    balance_in IN accounts.balance%TYPE)
  4  IS
  5  BEGIN
  6    UPDATE accounts
  7       SET balance = balance - balance_in
  8     WHERE account_id = account_id_in;
  9  END apply_balance;
```

```
 10  /

Procedure created.

SQL>
SQL>
SQL>
SQL>
SQL> DECLARE
  2     account_id  accounts.account_id%TYPE := 2;
  3     balance_remaining  accounts.balance%TYPE;
  4  BEGIN
  5     LOOP
  6
  7        balance_remaining := account_balance (account_id);
  8
  9
 10        EXIT WHEN balance_remaining < 1000;
 11
 12
 13        apply_balance (account_id, balance_remaining);
 14     END LOOP;
 15  END;
 16  /

PL/SQL procedure successfully completed.
```

30. 1. Records

A record is a group of related data items stored in attributes, each with its own name and datatype.

Records types are used in PL/SQL code (for example, as parameters of functions/procedures), but not in any SQL (views, table definitions, stored datatypes, and so on).

A record type can be defined either explicitly or implicitly.

The way to reference fields in the record type variables is by using variable.attribute.

An explicit declaration means that you first define your own datatype and then create a variable of that type.

Explicit Record Type

```
SQL> -- create demo table
SQL> create table Employee(
  2    ID                  VARCHAR2(4 BYTE)         NOT NULL,
  3    First_Name          VARCHAR2(10 BYTE),
  4    Last_Name           VARCHAR2(10 BYTE),
  5    Start_Date          DATE,
  6    End_Date            DATE,
  7    Salary              Number(8,2),
  8    City                VARCHAR2(10 BYTE),
  9    Description         VARCHAR2(15 BYTE)
 10  )
 11  /
SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',   'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('03','James',    'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
```

```
           Salary,  City,       Description)
  2                values('04','Celia',    'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',   'Black',   to_date('19840115','YYYYMMDD'), to_date('19980808','Y
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('06','Linda',    'Green',   to_date('19870730','YYYYMMDD'), to_date('19960104','Y
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('07','David',    'Larry',   to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('08','James',    'Cat',     to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /

1 row created.

SQL>
```

```
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /

ID   FIRST_NAME           LAST_NAME            START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- -------------------- -------------------- --------- --------- ---------- ---------- ---------------
02   Alison               Mathews              21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James                Smith                12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia                Rice                 24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert               Black                15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda                Green                30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David                Larry                31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James                Cat                  17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

7 rows selected.

SQL>
SQL>
SQL>
SQL> declare
  2      type emp_ty is record (emp_tx VARCHAR2(256),deptNo employee.id%TYPE);
  3      v_emp_rty emp_ty;
  4  begin
  5      select id ||' '||first_Name,  id into v_emp_rty from employee where id='01';
  6      DBMS_OUTPUT.put_line('Emp:'||v_emp_rty.emp_tx||'('||v_emp_rty.deptno||')');
  7  end;
  8  /
Emp:01 Jason(01)

PL/SQL procedure successfully completed.
```

30. 2. Accessing Individual Record elements

```
SQL>
SQL>
```

```
SQL> DECLARE
  2    TYPE hrc_company_rec IS RECORD
  3      (hrc_company_id NUMBER,
  4       product_description VARCHAR2(20),
  5       company_short_name VARCHAR2(30));
  6    v_example_rec hrc_company_rec;
  7  BEGIN
  8    v_example_rec.hrc_company_id :=1001;
  9
 10    v_example_rec.product_description :='C';
 11
 12    v_example_rec.company_short_name :='O Inc.';
 13
 14    dbms_output.put_line(to_number(v_example_rec.hrc_company_id)||' '||
 15
 16    v_example_rec.product_description||' '||
 17
 18    v_example_rec.company_short_name);
 19  END;
 20  /
1001 C O Inc.

PL/SQL procedure successfully completed.
```

30. 3. Accessing an entire record

```
SQL> DECLARE
  2    TYPE hrc_company_rec IS RECORD
  3      (hrc_company_id NUMBER,
  4       product_description VARCHAR2(20),
  5       company_short_name VARCHAR2(30));
  6    v_example_rec1 hrc_company_rec;
  7    v_example_rec2 hrc_company_rec;
  8  BEGIN
  9    v_example_rec1.hrc_company_id :=1001;
```

```
 10
 11    v_example_rec1.product_description :='C';
 12
 13    v_example_rec1.company_short_name :='O Inc.';
 14
 15    v_example_rec2 :=v_example_rec1;
 16
 17    dbms_output.put_line(to_number(v_example_rec2.hrc_company_id)||' '||
 18    v_example_rec2.product_description||' '||
 19
 20    v_example_rec2.company_short_name);
 21  END;
 22  /
1001 C O Inc.

PL/SQL procedure successfully completed.

SQL>
```

30. 4. Testing for equality of records

```
SQL> DECLARE
  2    TYPE hrc_company_rec IS RECORD
  3      (hrc_company_id NUMBER,
  4       product_description VARCHAR2(20),
  5       company_short_name VARCHAR2(30));
  6    v_example_rec1 hrc_company_rec;
  7    v_example_rec2 hrc_company_rec;
  8  BEGIN
  9    v_example_rec1.hrc_company_id :=1001;
 10
 11    v_example_rec1.product_description :='CEO/COO';
 12
 13    v_example_rec1.company_short_name :='O Inc.';
 14
```

```
15    v_example_rec2.hrc_company_id :=1002;
16
17    v_example_rec2.product_description :='VP ';
18
19    v_example_rec2.company_short_name :='Office Inc.';
20
21    IF ((v_example_rec1.hrc_company_id =v_example_rec2.hrc_company_id) AND
22       (v_example_rec1.product_description =v_example_rec2.product_description) AND
23       (v_example_rec1.company_short_name =v_example_rec2.company_short_name)) THEN
24      dbms_output.put_line('Both example records are identical.');
25    ELSE
26      dbms_output.put_line('The two example records are different.');
27    END IF;
28  END;
29  /
The two example records are different.

PL/SQL procedure successfully completed.
```

30. 5. Record Variables Based on Tables

A record type variable based on a table means that each field in the record has the exact same name and datatype as the columns in the specified table.

The %rowtype attribute is used to define a record based on a table.

The %rowtype is similar to the %type.

The %type refers to only a single variable.

The %rowtype refers to an entire table row.

```
TYPE emp_sales_rec is record
(id       INTEGER,
 name      VARCHAR2(32),
 rate      employee.salary%type);
```

30. 6. Record Variables

A record is a collection of individual fields that represents a row in a table.

These fields are unique and each has its own values.

By using records, you can group like data into one structure and then manipulate this structure as one entity or logical unit.

You declare a record in the declaration portion of a PL/SQL block, subprogram, or package.

```
SQL>
SQL>
SQL> DECLARE
  2    TYPE hrc_company_rec IS RECORD
  3      (hrc_company_id NUMBER,
  4       product_description VARCHAR2(20),
  5       company_short_name VARCHAR2(30));
  6    v_example_rec hrc_company_rec;
  7  BEGIN
  8    /*Do some processing */
  9    null;
 10  END;
 11  /

PL/SQL procedure successfully completed.

SQL>
SQL>
```

30. 7. Create Record based on table column type

```
SQL> set serveroutput on
SQL> set echo on
SQL> -- create demo table
```

```
SQL> create table Employee(
  2    ID                  VARCHAR2(4 BYTE)         NOT NULL,
  3    First_Name          VARCHAR2(10 BYTE),
  4    Salary              Number(8,2)
  5  )
  6  /

Table created.

SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Salary)
  2               values ('01','Jason', 1234.56)
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Salary)
  2                values('02','Alison', 6661.78)
  3  /

1 row created.

SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /


ID   FIRST_NAME     SALARY
---- ---------- ----------
01   Jason         1234.56
02   Alison        6661.78

SQL>
SQL>
SQL> SET ECHO ON
SQL>
SQL> DECLARE
```

```
  2      TYPE EmployeeTypee IS RECORD (
  3          id employee.id%type,
  4          name employee.first_name%type,
  5          salary employee.salary%type
  6      );
  7
  8      emp EmployeeTypee;
  9  BEGIN
 10      SELECT * INTO emp
 11      FROM employee
 12      WHERE id = 2;
 13
 14      DBMS_OUTPUT.PUT_LINE('emp_record.id: ' || emp.id);
 15      DBMS_OUTPUT.PUT_LINE('emp_record.id: ' || emp.name);
 16      DBMS_OUTPUT.PUT_LINE('emp_record.id: ' || emp.salary);
 17  END;
 18  /
emp_record.id: 02
emp_record.id: Alison
emp_record.id: 6661.78

PL/SQL procedure successfully completed.
```

30. 12. Inserts and updates using record variables

```
SQL> -- create demo table
SQL> create table Employee(
  2    ID                  VARCHAR2(4 BYTE)         NOT NULL,
  3    First_Name          VARCHAR2(10 BYTE),
  4    Salary              Number(8,2)
  5  )
  6  /

Table created.
```

```
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /

no rows selected

SQL>
SQL>
SQL>
SQL> create or replace procedure p_insertNewEmp(i_deptno VARCHAR)
  2  is
  3      v_emp_rec employee%ROWTYPE;
  4  begin
  5      select max(id)+1 into v_emp_rec.id from employee;
  6
  7      v_emp_rec.first_Name:='Emp#'||v_emp_rec.id;
  8      v_emp_rec.id :=i_deptno;
  9      v_emp_rec.salary := v_emp_rec.salary+1;
 10      insert into employee values v_emp_rec;
 11  end;
 12  /

Procedure created.

SQL>
SQL> call p_insertNewEmp('01');

Call completed.

SQL>
SQL> select * from employee;

ID   FIRST_NAME               SALARY
---- -------------------- ----------
01   Emp#
```

30. 15. Looping through Records in a Cursor

```
SQL>
SQL> -- create demo table
SQL> create table Employee(
  2    ID                  VARCHAR2(4 BYTE)         NOT NULL,
  3    First_Name          VARCHAR2(10 BYTE),
  4    Last_Name           VARCHAR2(10 BYTE),
  5    Start_Date          DATE,
  6    End_Date            DATE,
  7    Salary              Number(8,2),
  8    City                VARCHAR2(10 BYTE),
  9    Description         VARCHAR2(15 BYTE)
 10  )
 11  /

Table created.

SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',   'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
```

```
  2                values('03','James',    'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary,  City,       Description)
  2                values('04','Celia',    'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',   'Black',   to_date('19840115','YYYYMMDD'), to_date('19980808','Y
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary, City,        Description)
  2                values('06','Linda',    'Green',   to_date('19870730','YYYYMMDD'), to_date('19960104','Y
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary, City,        Description)
  2                values('07','David',    'Larry',   to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary, City,        Description)
```

```
  2                values('08','James',    'Cat',     to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /

1 row created.

SQL>
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /

ID   FIRST_NAME           LAST_NAME            START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- -------------------- -------------------- --------- --------- ---------- ---------- ---------------
01   Jason                Martin               25-JUL-96 25-JUL-06    1234.56 Toronto    Programmer
02   Alison               Mathews              21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James                Smith                12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia                Rice                 24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert               Black                15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda                Green                30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David                Larry                31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James                Cat                  17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.

SQL>
SQL>
SQL> declare
  2      cursor c_emp is select * from employee;
  3      r_emp c_emp%ROWTYPE;
  4  begin
  5      open c_emp;
  6      loop
  7          fetch c_emp into r_emp;
  8          exit when c_emp%NOTFOUND;
  9          DBMS_OUTPUT.put_line(r_emp.first_Name);
 10      end loop;
 11      close c_emp;
 12  end;
```

```
 13  /
Jason
Alison
James
Celia
Robert
Linda
David
James

PL/SQL procedure successfully completed.
```

30. 16. Looping through multiple records

```
SQL>
SQL> -- create demo table
SQL> create table Employee(
  2    ID                  VARCHAR2(4 BYTE)         NOT NULL,
  3    First_Name          VARCHAR2(10 BYTE),
  4    Last_Name           VARCHAR2(10 BYTE),
  5    Start_Date          DATE,
  6    End_Date            DATE,
  7    Salary              Number(8,2),
  8    City                VARCHAR2(10 BYTE),
  9    Description         VARCHAR2(15 BYTE)
 10  )
 11  /

Table created.

SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
```

```
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',   'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('03','James',    'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('04','Celia',    'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',   'Black',   to_date('19840115','YYYYMMDD'), to_date('19980808','Y
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary, City,        Description)
  2                values('06','Linda',    'Green',   to_date('19870730','YYYYMMDD'), to_date('19960104','Y
```

```
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary, City,        Description)
  2                values('07','David',    'Larry',   to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary, City,        Description)
  2                values('08','James',    'Cat',     to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /

1 row created.

SQL>
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /

ID   FIRST_NAME           LAST_NAME            START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- -------------------- -------------------- --------- --------- ---------- ---------- ---------------
01   Jason                Martin               25-JUL-96 25-JUL-06    1234.56 Toronto    Programmer
02   Alison               Mathews              21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James                Smith                12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia                Rice                 24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert               Black                15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda                Green                30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David                Larry                31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James                Cat                  17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.
```

```
SQL>
SQL>
SQL> declare
  2      cursor c_emp is select * from employee;
  3      r_emp c_emp%ROWTYPE;
  4  begin
  5      open c_emp;
  6      loop
  7          fetch c_emp into r_emp;
  8          exit when c_emp%NOTFOUND;
  9          DBMS_OUTPUT.put_line(r_emp.first_Name);
 10      end loop;
 11      close c_emp;
 12  end;
 13  /
Jason
Alison
James
Celia
Robert
Linda
David
James

PL/SQL procedure successfully completed.
```

30. 17. Perform a field-by-field comparison

```
SQL>
SQL> -- create demo table
SQL> create table Employee(
  2    ID                 VARCHAR2(4 BYTE)         NOT NULL,
  3    First_Name         VARCHAR2(10 BYTE),
  4    Last_Name          VARCHAR2(10 BYTE),
  5    Start_Date         DATE,
  6    End_Date           DATE,
```

```
  7    Salary              Number(8,2),
  8    City                VARCHAR2(10 BYTE),
  9    Description         VARCHAR2(15 BYTE)
 10  )
 11  /

Table created.

SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',   'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('03','James',    'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('04','Celia',    'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /
```

```
1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',    'Black',    to_date('19840115','YYYYMMDD'), to_date('19980808','Y
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('06','Linda',     'Green',    to_date('19870730','YYYYMMDD'), to_date('19960104','Y
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('07','David',     'Larry',    to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                      End_Date,
           Salary, City,        Description)
  2                values('08','James',     'Cat',      to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /

1 row created.

SQL>
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /
```

```
ID   FIRST_NAME           LAST_NAME            START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- -------------------- -------------------- --------- --------- ---------- ---------- ---------------
01   Jason                Martin               25-JUL-96 25-JUL-06    1234.56 Toronto    Programmer
02   Alison               Mathews              21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James                Smith                12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia                Rice                 24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert               Black                15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda                Green                30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David                Larry                31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James                Cat                  17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.

SQL>
SQL>
SQL>
SQL> create or replace function f_isDuplicate_yn(i_emp1_rec employee%ROWTYPE, i_emp2_rec employee%ROWTYPE)
  2  return VARCHAR2
  3  is
  4      v_out_tx VARCHAR2(1):='N';
  5  begin
  6      if  i_emp1_rec.first_Name=i_emp2_rec.first_Name
  7      and i_emp1_rec.id=i_emp2_rec.id
  8      and i_emp1_rec.last_name=i_emp2_rec.last_Name
  9      then
 10          v_out_tx:='Y';
 11      end if;
 12
 13      return v_out_tx;
 14  end;
 15  /

Function created.
```

30. 18. Passing Variables without Copying

Variables can serve as input/output parameters.

The problem with using records as parameters is that they are just too big, and they require a lot of memory.
By using NOCOPY, you are only passing a pointer to the variable rather than copying the values.
NOCOPY is particularly useful when passing record variables.

```
SQL>
SQL> -- create demo table
SQL> create table Employee(
  2    ID                  VARCHAR2(4 BYTE)         NOT NULL,
  3    First_Name          VARCHAR2(10 BYTE),
  4    Last_Name           VARCHAR2(10 BYTE),
  5    Start_Date          DATE,
  6    End_Date            DATE,
  7    Salary              Number(8,2),
  8    City                VARCHAR2(10 BYTE),
  9    Description         VARCHAR2(15 BYTE)
 10  )
 11  /

Table created.

SQL>
SQL> -- prepare data
SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2               values ('01','Jason',    'Martin',  to_date('19960725','YYYYMMDD'), to_date('20060725','Y
YYYMMDD'), 1234.56, 'Toronto',  'Programmer')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
  2                values('02','Alison',   'Mathews', to_date('19760321','YYYYMMDD'), to_date('19860221','Y
YYYMMDD'), 6661.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                    End_Date,
           Salary,  City,       Description)
```

```
  2                values('03','James',    'Smith',   to_date('19781212','YYYYMMDD'), to_date('19900315','Y
YYYMMDD'), 6544.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary,  City,       Description)
  2                values('04','Celia',    'Rice',    to_date('19821024','YYYYMMDD'), to_date('19990421','Y
YYYMMDD'), 2344.78, 'Vancouver','Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary,  City,       Description)
  2                values('05','Robert',   'Black',   to_date('19840115','YYYYMMDD'), to_date('19980808','Y
YYYMMDD'), 2334.78, 'Vancouver','Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary, City,        Description)
  2                values('06','Linda',    'Green',   to_date('19870730','YYYYMMDD'), to_date('19960104','Y
YYYMMDD'), 4322.78,'New York',  'Tester')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary, City,        Description)
  2                values('07','David',    'Larry',   to_date('19901231','YYYYMMDD'), to_date('19980212','Y
YYYMMDD'), 7897.78,'New York',  'Manager')
  3  /

1 row created.

SQL> insert into Employee(ID,  First_Name, Last_Name, Start_Date,                     End_Date,
           Salary, City,        Description)
```

```
  2                 values('08','James',     'Cat',     to_date('19960917','YYYYMMDD'), to_date('20020415','Y
YYYMMDD'), 1232.78,'Vancouver', 'Tester')
  3  /

1 row created.

SQL>
SQL>
SQL>
SQL> -- display data in the table
SQL> select * from Employee
  2  /

ID   FIRST_NAME           LAST_NAME            START_DAT END_DATE      SALARY CITY       DESCRIPTION
---- -------------------- -------------------- --------- --------- ---------- ---------- ---------------
01   Jason                Martin               25-JUL-96 25-JUL-06    1234.56 Toronto    Programmer
02   Alison               Mathews              21-MAR-76 21-FEB-86    6661.78 Vancouver  Tester
03   James                Smith                12-DEC-78 15-MAR-90    6544.78 Vancouver  Tester
04   Celia                Rice                 24-OCT-82 21-APR-99    2344.78 Vancouver  Manager
05   Robert               Black                15-JAN-84 08-AUG-98    2334.78 Vancouver  Tester
06   Linda                Green                30-JUL-87 04-JAN-96    4322.78 New York   Tester
07   David                Larry                31-DEC-90 12-FEB-98    7897.78 New York   Manager
08   James                Cat                  17-SEP-96 15-APR-02    1232.78 Vancouver  Tester

8 rows selected.

SQL>
SQL>
SQL>
SQL> create or replace procedure p_generateNewEmp(io_emp in out nocopy employee%ROWTYPE)
  2  is
  3  begin
  4      select max(id)+1 into io_emp.id from employee;
  5      io_emp.first_Name:='Emp#'||io_emp.id;
  6  end;
  7  /

Procedure created.

SQL>
```

```
SQL> declare
  2      v_emp_rec employee%ROWTYPE;
  3  begin
  4      v_emp_rec.id:=10;
  5      p_generateNewEmp(v_emp_rec);
  6      DBMS_OUTPUT.put_line('Generated:'||v_emp_rec.id||' '||v_emp_rec.first_Name);
  7  end;
  8  /
Generated:9 Emp#9

PL/SQL procedure successfully completed.
```

31. 1. Using Objects in PL/SQL

```
SQL>
SQL>
SQL> CREATE Or Replace TYPE ProductType AS OBJECT (
  2    id          NUMBER,
  3    name        VARCHAR2(15),
  4    description VARCHAR2(22),
  5    price       NUMBER(5, 2),
  6    days_valid  NUMBER
  7  )
  8  /

Type created.

SQL>
SQL>
SQL> CREATE TABLE object_products OF ProductType
  2  /

Table created.

SQL>
```

```
SQL> INSERT INTO object_products (
  2    id, name, description, price, days_valid
  3  ) VALUES (
  4    1, 'AAA', 'BBB', 2.99, 5
  5  );

1 row created.

SQL>
SQL> select * from object_products;

 ID NAME            DESCRIPTION                 PRICE DAYS_VALID
--- --------------- ---------------------- ---------- ----------
  1 AAA             BBB                         2.99          5

SQL>
SQL> CREATE OR REPLACE PACKAGE product_package AS
  2    TYPE ref_cursor_typ IS REF CURSOR;
  3    FUNCTION get_products RETURN ref_cursor_typ;
  4    PROCEDURE insert_product (
  5      p_id          IN object_products.id%TYPE,
  6      p_name        IN object_products.name%TYPE,
  7      p_description IN object_products.description%TYPE,
  8      p_price       IN object_products.price%TYPE,
  9      p_days_valid  IN object_products.days_valid%TYPE
 10    );
 11  END product_package;
 12  /

Package created.

SQL>
SQL>
SQL> CREATE OR REPLACE PACKAGE BODY product_package AS
  2    FUNCTION get_products
  3    RETURN ref_cursor_typ IS
  4      products_ref_cursor ref_cursor_typ;
  5    BEGIN
  6      OPEN products_ref_cursor FOR
  7        SELECT VALUE(op)
```

```
  8         FROM object_products op;
  9       RETURN products_ref_cursor;
 10     END get_products;
 11
 12     PROCEDURE insert_product (
 13       p_id          IN object_products.id%TYPE,
 14       p_name        IN object_products.name%TYPE,
 15       p_description IN object_products.description%TYPE,
 16       p_price       IN object_products.price%TYPE,
 17       p_days_valid  IN object_products.days_valid%TYPE
 18     ) AS
 19       product ProductType :=
 20         ProductType(
 21           p_id, p_name, p_description, p_price, p_days_valid
 22         );
 23     BEGIN
 24       INSERT INTO object_products VALUES (product);
 25       COMMIT;
 26     EXCEPTION
 27       WHEN OTHERS THEN
 28         ROLLBACK;
 29     END insert_product;
 30   END product_package;
 31   /

Package body created.

SQL>
SQL> CALL product_package.insert_product(101, 'AAA', 'AAA and AAA', 1.50, 20);

Call completed.

SQL>
SQL> SELECT product_package.get_products
  2  FROM dual;

GET_PRODUCTS
--------------------
CURSOR STATEMENT : 1
```

```
CURSOR STATEMENT : 1

VALUE(OP)(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
---------------------------------------------------
PRODUCTTYPE(1, 'AAA', 'BBB', 2.99, 5)
PRODUCTTYPE(101, 'AAA', 'AAA and AAA', 1.5, 20)


SQL>
SQL> select * from object_products;

 ID NAME            DESCRIPTION                 PRICE DAYS_VALID
--- --------------- ----------------------- ---------- ----------
  1 AAA             BBB                          2.99          5
### AAA             AAA and AAA                   1.5         20

SQL>
```

32. 1. SYS.XMLTYPE type column

```
SQL> CREATE TABLE testxml (id NUMBER(3), dt SYS.XMLTYPE);

Table created.

SQL>
SQL> DECLARE
  2       x clob;
  3  BEGIN
  4       x := '<?xml version="1.0"?>
  5       <customer><name>C C</name><title>Golfer</title>
  6       </customer>';
  7
  8      INSERT INTO testxml VALUES (123,
  9       sys.xmltype.createxml(x)
 10       );
```

```
 11  end;
 12  /

PL/SQL procedure successfully completed.

SQL> SELECT *
  2  FROM testxml
  3  /

ID     DT
--------------------------------------------------------------------------------
123    <?xml version="1.0"?><customer><name>C C</name><title>Golfer</title></customer>


SQL>
```

32. 2. Using an anonymous PL/SQL script like the following one.

```
SQL>
SQL>
SQL> CREATE TABLE testxml (id NUMBER(3), dt SYS.XMLTYPE);

Table created.

SQL>
SQL>
SQL> DECLARE
  2      x VARCHAR2(1000);
  3  BEGIN
  4
  5  INSERT INTO testxml VALUES (222,
  6  sys.xmltype.createxml(
  7  '<?xml version="1.0"?>
  8  <customer>
  9      <name>T J</name>
```

```
 10      <title>Plumber</title>
 11  </customer>'));
 12  end;
 13  /

PL/SQL procedure successfully completed.

SQL>
SQL>
SQL> SELECT *
  2  FROM testxml
  3  /

ID      DT
-----------------------------------------------------------------------------------------
222     <?xml version="1.0"?><customer><name>T J</name><title>Plumber</title></customer>


SQL>
```

40. 1. Blob locator

```
SQL>
SQL>
SQL> CREATE TABLE myBlob
  2  (id NUMBER PRIMARY KEY,
  3   blob_data BLOB);

Table created.

SQL> DECLARE
  2    blobValue BLOB;
  3  BEGIN
```

```
  4    INSERT INTO myBlob VALUES (3,HEXTORAW('AAAAAAAAAAAAAAAAAAA'));
  5    SELECT blob_data
  6    INTO blobValue
  7    FROM myBlob
  8    WHERE id =3;
  9
 10  END;
 11  /

PL/SQL procedure successfully completed
```

40. 2. Insert into returning to blob type variable

```
SQL> CREATE TABLE myBlob
  2  (id NUMBER PRIMARY KEY,
  3   blob_data BLOB);

Table created.

SQL>
SQL>
SQL> DECLARE
  2    blobValue BLOB;
  3  BEGIN
  4    INSERT INTO myBlob VALUES (4,HEXTORAW('AAAAAAAAAAAAAAAAAAA'))
  5    RETURNING blob_data INTO blobValue;
  6
  7  END;
  8  /

PL/SQL procedure successfully completed.
```

40. 3. Stored procedure to display the contents of the col_blob and col_clob columns

```
SQL>
SQL> create table myTable
  2  (key NUMBER PRIMARY KEY
  3  ,col_blob BLOB
  4  ,col_clob CLOB);

Table created.

SQL>
SQL> INSERT INTO myTable(key, col_blob, col_clob) VALUES(1, HEXTORAW('101F'), 'ZYXW');

1 row created.

SQL>
SQL> INSERT INTO myTable(key, col_blob, col_clob) VALUES(2, HEXTORAW('111101F'), 'ABCD');

1 row created.

SQL>
SQL> CREATE OR REPLACE PROCEDURE sp_inmyTable
  2  IS
  3          v_key1          myTable.key%TYPE;
  4          blobValue1         myTable.col_blob%TYPE;
  5          clobValue1         myTable.col_clob%TYPE;
  6          v_key2          myTable.key%TYPE;
  7          blobValue2         myTable.col_blob%TYPE;
  8          clobValue2         myTable.col_clob%TYPE;
  9          v_buffer        VARCHAR2(1000);
 10          v_offset        NUMBER;
 11          v_amount        NUMBER;
 12  BEGIN
 13          SELECT key, col_blob, col_clob
 14          INTO v_key1, blobValue1, clobValue1
 15          FROM myTable
 16          WHERE key = 1;
 17
 18          v_amount := 80;
 19          v_offset := 1;
 20          DBMS_LOB.READ (clobValue1, v_amount, v_offset, v_buffer);
 21          DBMS_OUTPUT.PUT_LINE ('Clob Contents => ' ||  v_buffer);
```

```
 22
 23          v_amount := 80;
 24          v_offset := 1;
 25          DBMS_LOB.READ (blobValue1, v_amount, v_offset, v_buffer);
 26          DBMS_OUTPUT.PUT_LINE ('Blob Contents => ' ||  v_buffer);
 27  END;
 28  /

Procedure created.

SQL>
SQL> exec sp_inmyTable
Clob Contents => ZYXW
Blob Contents => 101F

PL/SQL procedure successfully completed.
```

Deal with blob data

```
SQL>
SQL> create table demo
  2  ( id           int primary key,
  3    theBlob      blob
  4  )
  5  /

Table created.

SQL>
SQL> create or replace directory my_files as 'c:\temp\';

Directory created.

SQL>
SQL> create sequence blob_seq;
```

```
Sequence created.

SQL>
SQL> create or replace function clean( p_raw in blob,
  2                  p_from_byte in number default 1,
  3                  p_for_bytes in number default 4000 )
  4  return varchar2 as
  5      l_tmp varchar2(8192) default utl_raw.cast_to_varchar2( dbms_lob.substr(p_raw,p_for_bytes,p_from_by
te));
  6      l_char   char(1);
  7      l_return varchar2(16384);
  8      l_whitespace varchar2(25) default chr(13) || ' ' || chr(9);
  9      l_ws_char    varchar2(50) default 'rnt';
 10
 11  begin
 12      for i in 1 .. length(l_tmp)
 13      loop
 14          l_char := substr( l_tmp, i, 1 );
 15          if ( ascii(l_char) between 32 and 127 )
 16          then
 17              l_return := l_return || l_char;
 18              if ( l_char = '\' ) then l_return := l_return || '\';
 19              end if;
 20          elsif ( instr( l_whitespace, l_char ) > 0 )
 21          then
 22              l_return := l_return || '\' || substr( l_ws_char, instr(l_whitespace,l_char), 1 );
 23          else
 24              l_return := l_return || '.';
 25          end if;
 26      end loop;
 27      return substr(l_return,1,4000);
 28  end;
 29  /

Function created.

SQL>
SQL> select id,dbms_lob.getlength(theBlob) len,clean(theBlob,30,40) piece,
  2         dbms_lob.substr(theBlob,40,30) raw_data
```

```
  3  from demo
  4  /

no rows selected
```

41. 1. BFILE type variable

```
SQL> CREATE TABLE myBFile
  2  (id NUMBER PRIMARY KEY,
  3   bfile_data BFILE);

Table created.

SQL>
SQL> DECLARE
  2    bfileValue BFILE;
  3  BEGIN
  4    INSERT INTO myBFile VALUES (1,BFILENAME('BFILE_DIR','test.bmp'))
  5    RETURNING bfile_data INTO bfileValue;
  6
  7  END;
  8  /

PL/SQL procedure successfully completed.
```

41. 2. Close the BFILE

```
SQL>
SQL> CREATE TABLE myBlob
  2  (id NUMBER PRIMARY KEY,
  3   blob_data BLOB);

Table created.

SQL>
SQL> DECLARE
```

```
  2    bfile_pointer BFILE;
  3    blob_pointer BLOB;
  4    bfile_offset NUMBER :=1;
  5    blob_offset NUMBER :=1;
  6    tot_len INTEGER;
  7  BEGIN
  8
  9    INSERT INTO myBlob VALUES (5,EMPTY_BLOB());
 10
 11    SELECT blob_data INTO blob_pointer FROM myBlob WHERE id =5 FOR UPDATE;
 12
 13    bfile_pointer :=bfilename('BFILE_DIR','test.bmp');
 14
 15    dbms_lob.fileopen(bfile_pointer,dbms_lob.file_readonly);
 16
 17    dbms_lob.OPEN(blob_pointer,dbms_lob.lob_readwrite);
 18
 19    dbms_lob.LOADBLOBFROMFILE(blob_pointer,bfile_pointer,dbms_lob.lobmaxsize,bfile_offset,blob_offset);
 20
 21    tot_len :=DBMS_LOB.GETLENGTH(blob_pointer);
 22
 23    dbms_lob.close(blob_pointer);
 24
 25    dbms_lob.fileclose(bfile_pointer);
 26
 27    DBMS_OUTPUT.PUT_LINE(TO_CHAR(tot_len));
 28  END;
 29  /
DECLARE
*
ERROR at line 1:
ORA-22288: file or LOB operation FILEOPEN failed
The system cannot find the path specified.
ORA-06512: at "SYS.DBMS_LOB", line 523
ORA-06512: at line 15
```

41. 3. Open the target CLOB and the source BFILE

```
SQL>
SQL> CREATE TABLE facebook (
  2     name VARCHAR2(80),
  3     photo BLOB,
  4     directions CLOB,
  5     description NCLOB,
  6     web_page BFILE);

Table created.

SQL>
SQL> CREATE DIRECTORY bfile_data AS 'c:\xxx';

Directory created.

SQL>
SQL> DECLARE
  2     myBFile BFILE := BFILENAME('BFILE_DATA','TanneryFalls.directions');
  3     directions CLOB;
  4     destination_offset INTEGER := 1;
  5     source_offset INTEGER := 1;
  6     language_context INTEGER := DBMS_LOB.default_lang_ctx;
  7     warning_message INTEGER;
  8  BEGIN
  9     DELETE FROM facebook WHERE name='Falls';
 10
 11     INSERT INTO facebook (name,directions) VALUES ('Falls',EMPTY_CLOB(  ));
 12
 13     SELECT directions INTO directions FROM facebook WHERE name='Falls';
 14
 15
 16     DBMS_LOB.OPEN(directions, DBMS_LOB.LOB_READWRITE);
 17     DBMS_LOB.OPEN(myBFile);
 18
 19     DBMS_LOB.LOADCLOBFROMFILE(directions, myBFile,
 20                               DBMS_LOB.LOBMAXSIZE,
```

```
 21                                destination_offset, source_offset,
 22                                NLS_CHARSET_ID('US7ASCII'),
 23                                language_context, warning_message);
 24
 25     IF warning_message = DBMS_LOB.WARN_INCONVERTIBLE_CHAR THEN
 26          dbms_output.put_line(
 27             'Warning! Some characters couldn''t be converted.');
 28     END IF;
 29
 30     DBMS_LOB.CLOSE(directions);
 31     DBMS_LOB.CLOSE(myBFile);
 32  END;
 33  /
DECLARE
*
ERROR at line 1:
ORA-22288: file or LOB operation FILEOPEN failed
The system cannot find the path specified.
ORA-06512: at "SYS.DBMS_LOB", line 716
ORA-06512: at line 17


SQL>
```

42. 1. Clob type column and clob type variable

```
SQL>
SQL> CREATE TABLE test_clob(
  2      clob_id    NUMBER,
  3      clob_object CLOB
  4  );

Table created.

SQL>
SQL> DECLARE
  2        lv_clob_pointer_txt CLOB;
  3  BEGIN
```

```
  4        INSERT INTO test_clob (clob_id, clob_object)VALUES (1, '1234567890')
  5        RETURNING clob_object INTO lv_clob_pointer_txt;
  6  END;
  7  /

PL/SQL procedure successfully completed.
```

42. 2. Insert into returning into clob

```
SQL>
SQL> CREATE TABLE myClob
  2  (id NUMBER PRIMARY KEY,
  3   clob_data CLOB);

Table created.

SQL>
SQL>
SQL> DECLARE
  2    clobValue CLOB;
  3  BEGIN
  4    INSERT INTO myClob VALUES (104,EMPTY_CLOB())
  5    RETURNING clob_data INTO clobValue;
  6
  7  END;
  8  /

PL/SQL procedure successfully completed.
```

42. 3. Save data to clob type column

```
SQL>
SQL> set echo on
SQL>
SQL> create global temporary table myTable
  2  ( id    int primary key,
```

```
  3    c_lob clob,
  4    b_lob blob
  5  )
  6  /

Table created.

SQL>
SQL> create sequence myTable_seq;

Sequence created.

SQL>
SQL> create or replace function to_blob( p_cname in varchar2,
  2                    p_tname in varchar2,
  3                    p_rowid in rowid ) return blob
  4  as
  5      l_blob blob;
  6      l_id   int;
  7  begin
  8      select myTable_seq.nextval into l_id from dual;
  9
 10      execute immediate
 11         'insert into myTable (id,b_lob)
 12          select :id, to_lob( p_cname )
 13            from ' || p_tname ||
 14         ' where rowid = :rid '
 15       using IN l_id, IN p_rowid;
 16
 17      select b_lob into l_blob from myTable where id = l_id ;
 18
 19      return l_blob;
 20  end;
 21  /

Function created.
```

42. 4. Writes into a CLOB

```
SQL> CREATE TABLE facebook (
  2     name VARCHAR2(80),
  3     photo BLOB,
  4     directions CLOB,
  5     description NCLOB,
  6     web_page BFILE);

Table created.

SQL>
SQL> CREATE DIRECTORY bfile_data AS 'c:\xxx';

Directory created.

SQL>
SQL> DECLARE
  2     directions CLOB;
  3     amount BINARY_INTEGER;
  4     offset INTEGER;
  5     first_direction VARCHAR2(100);
  6     more_directions VARCHAR2(500);
  7  BEGIN
  8
  9     DELETE FROM facebook WHERE name='Falls';
 10
 11     INSERT INTO facebook(name,directions)VALUES ('Falls',EMPTY_CLOB());
 12
 13     SELECT directions INTO directions FROM facebook WHERE name='Falls';
 14
 15     DBMS_LOB.OPEN(directions, DBMS_LOB.LOB_READWRITE);
 16
 17     first_direction := 'across the Bridge.';
 18     amount := LENGTH(first_direction);
 19     offset := 1;
 20     DBMS_LOB.WRITE(directions, amount, offset, first_direction);
 21
 22     more_directions := ' right.';
 23     DBMS_LOB.WRITEAPPEND(directions,LENGTH(more_directions), more_directions);
 24
 25     more_directions := ' more.';
```

```
 26     DBMS_LOB.WRITEAPPEND(directions,LENGTH(more_directions), more_directions);
 27
 28     DBMS_LOB.CLOSE(directions);
 29  END;
 30  /

PL/SQL procedure successfully completed.
```

42. 5. Demonstrations of SQL functions being applied to CLOB values

```
SQL>
SQL> CREATE TABLE facebook (
  2     name VARCHAR2(80),
  3     photo BLOB,
  4     directions CLOB,
  5     description NCLOB,
  6     web_page BFILE);

Table created.

SQL>
SQL> CREATE DIRECTORY bfile_data AS 'c:\xxx';

Directory created.

SQL>
SQL> DECLARE
  2     name CLOB;
  3     name_upper CLOB;
  4     directions CLOB;
  5     blank_space VARCHAR2(1) := ' ';
  6  BEGIN
  7     SELECT name, SUBSTR(directions,1,500)
  8     INTO name, directions
  9     FROM facebook
```

```
 10     WHERE name = 'Falls';
 11
 12     name_upper := UPPER(name);
 13
 14     IF name = name_upper THEN
 15        DBMS_OUTPUT.PUT_LINE('We did not need to uppercase the name.');
 16     END IF;
 17
 18     IF INSTR(directions,'Bridge') <> 0 THEN
 19        DBMS_OUTPUT.PUT_LINE('To get to ' || name_upper || blank_space);
 20     END IF;
 21  END;
 22  /
DECLARE
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 7
```

44. 1. Use a SYS.HttpUriType value to retrieve a message from a URL

```
SQL>
SQL>
SQL> DECLARE
  2     WebPageURL sys.HttpUriType;
  3     WebPage CLOB;
  4  BEGIN
  5     WebPageURL := SYS.HttpUriType.createUri('http://g.com/');
  6
  7     WebPage := WebPageURL.getclob();
  8
  9     DBMS_OUTPUT.PUT_LINE((SUBSTR(WebPage,1,60)));
 10  END;
 11  /
DECLARE
*
ERROR at line 1:
ORA-29273: HTTP request failed
```

```
ORA-06512: at "SYS.UTL_HTTP", line 1674
ORA-12545: Connect failed because target host or object does not exist
ORA-06512: at "SYS.HTTPURITYPE", line 34
ORA-06512: at line 7


SQL>
```

45. 1. UROWID type

```
SQL>
SQL> CREATE TABLE employee (
  2     emp_id NUMBER,
  3     last_name VARCHAR2(30),
  4     first_name VARCHAR2(30),
  5     salary NUMBER
  6  );

Table created.

SQL>
SQL>
SQL> INSERT INTO employee (emp_id, last_name, first_name, salary)VALUES (1, 'G', 'J', 100000);

1 row created.

SQL>
SQL> INSERT INTO employee (emp_id, last_name, first_name, salary)VALUES (2, 'G', 'H', 100000);

1 row created.

SQL>
SQL>
SQL>
SQL> DECLARE
  2     employee_rowid UROWID;
  3     employee_salary NUMBER;
  4  BEGIN
```

```
  5     SELECT rowid, salary INTO employee_rowid, employee_salary
  6     FROM employee
  7     WHERE last_name='G' AND first_name='J';
  8
  9     UPDATE employee
 10        SET salary = employee_salary
 11      WHERE last_name='G' AND first_name='J';
 12  END;
 13  /

PL/SQL procedure successfully completed.
```

46. 1. XMLType.CreateXML

```
SQL>
SQL> CREATE TABLE myTable (
  2     fall_id NUMBER,
  3     fall SYS.XMLType
  4  );

Table created.

SQL>
SQL> INSERT INTO myTable VALUES (1, SYS.XMLType.CreateXML(
  2     '<?xml version="1.0"?>
  3      <fall>
  4         <name>myTable</name>
  5         <county>USA</county>
  6         <state>MI</state>
  7         <url>
  8            http://your.com
  9         </url>
 10      </fall>'));

1 row created.

SQL>
SQL> INSERT INTO myTable VALUES (2, SYS.XMLType.CreateXML(
```

```
  2     '<?xml version="1.0"?>
  3      <fall>
  4         <name>Au myTable</name>
  5         <county>USA</county>
  6         <state>MI</state>
  7         <url>
  8            http://your.com
  9         </url>
 10      </fall>'));

1 row created.

SQL>
SQL> INSERT INTO myTable VALUES (3, SYS.XMLType.CreateXML(
  2     '<?xml version="1.0"?>
  3      <fall>
  4         <name>myTable</name>
  5         <county>USA</county>
  6         <state>MI</state>
  7      </fall>'));

1 row created.

SQL>
```

46. 2. XMLType.getStringVal

```
SQL>
SQL> CREATE TABLE myTable (
  2     fall_id NUMBER,
  3     fall SYS.XMLType
  4  );

Table created.

SQL>
SQL> INSERT INTO myTable VALUES (1, SYS.XMLType.CreateXML(
  2     '<?xml version="1.0"?>
```

```
  3      <fall>
  4         <name>myTable</name>
  5         <county>USA</county>
  6         <state>MI</state>
  7         <url>
  8            http://your.com
  9         </url>
 10      </fall>'));

1 row created.


SQL>
SQL> CREATE INDEX by_name
  2     ON myTable f (
  3        SUBSTR(
  4           SYS.XMLType.getStringVal(
  5              SYS.XMLType.extract(f.fall,'/fall/name/text()')
  6           ),1,80
  7        )
  8     );

Index created.
```

46. 3. Extract and display the URL

```
SQL>
SQL> CREATE TABLE myTable (
  2     fall_id NUMBER,
  3     fall SYS.XMLType
  4  );

Table created.

SQL>
```

```
SQL> INSERT INTO myTable VALUES (1, SYS.XMLType.CreateXML(
  2     '<?xml version="1.0"?>
  3      <fall>
  4         <name>myTable</name>
  5         <county>USA</county>
  6         <state>MI</state>
  7         <url>
  8            http://your.com
  9         </url>
 10      </fall>'));

1 row created.



SQL>
SQL>
SQL> <<demo_block>>
  2  DECLARE
  3     fall SYS.XMLType;
  4     url VARCHAR2(80);
  5  BEGIN
  6
  7     SELECT fall INTO demo_block.fall
  8     FROM myTable f
  9     WHERE f.fall_id = 1;
 10
 11     url := fall.extract('/fall/url/text()').getStringVal;
 12     DBMS_OUTPUT.PUT_LINE(url);
 13  END;
 14  /

PL/SQL procedure successfully completed.
```

47. 1. The underlying database datatype for this example is Unicode UTF-8

```
SQL> DECLARE
  2
  3     x CHAR(30 CHAR) := 'The character ?is two-bytes.';
  4  BEGIN
  5     --Find the location of "is" in terms of characters
  6     DBMS_OUTPUT.PUT_LINE(INSTR(x,'is'));
  7
  8     --Find the location of "is" in terms of bytes
  9     DBMS_OUTPUT.PUT_LINE(INSTRB(x,'is'));
 10  END;
 11  /

PL/SQL procedure successfully completed.

SQL>
```

47. 3. INSTR2(x, UNISTR('\D834'))

```
SQL>
SQL> DECLARE
  2    x NCHAR(40) := UNISTR('This is a \D834\DD1E test');
  3  BEGIN
  4    DBMS_OUTPUT.PUT_LINE (INSTR2(x, UNISTR('\D834')));
  5    DBMS_OUTPUT.PUT_LINE (INSTR4(x, UNISTR('\D834')));
  6  END;
  7  /

PL/SQL procedure successfully completed.
```

```
SQL>
```

48. 1. BINARY_DOUBLE Demo

```
SQL>
SQL> DECLARE
  2     bd BINARY_DOUBLE;
  3     bd_area BINARY_DOUBLE;
  4     bd_sine BINARY_DOUBLE;
  5     nm NUMBER;
  6     nm_area NUMBER;
  7     nm_sine NUMBER;
  8     pi_bd BINARY_DOUBLE := 3.1415926536d;
  9     pi_nm NUMBER := 3.1415926536;
 10     bd_begin TIMESTAMP(9);
 11     bd_end TIMESTAMP(9);
 12     bd_wall_time INTERVAL DAY TO SECOND(9);
 13     nm_begin TIMESTAMP(9);
 14     nm_end TIMESTAMP(9);
 15     nm_wall_time INTERVAL DAY TO SECOND(9);
 16  BEGIN
 17     bd_begin := SYSTIMESTAMP;
 18     bd := 1d;
 19     LOOP
 20        bd_area := bd * bd * pi_bd;
 21        bd := bd + 1d;
 22        EXIT WHEN bd > 5000000;
 23     END LOOP;
 24     bd_end := SYSTIMESTAMP;
 25
 26
 27  END;
 28  /

PL/SQL procedure successfully completed.

SQL>
```