1. Exception Handling

```
DECLARE
   a int:=10;
   b int:=0;
   answer int;

BEGIN
   answer:=a/b;
   dbms_output.put_line('the result after division is'||answer);

exception
   WHEN zero_divide THEN
      dbms_output.put_line('dividing by zero please check the values again');
      dbms_output.put_line('the value of a is '||a);
      dbms_output.put_line('the value of b is '||b);
END;
```

2. Functions

Create a Function

```
create or replace function jk
return number is
tot number(2):=0;
BEGIN
select count(*) into tot from cus;
return tot;
end;
```

Calling a Function

```
DECLARE
   jivi number(2);
BEGIN
   jivi := jk();
   dbms_output.put_line('Total no. of Customers: ' || jivi);
END;
/
```

3. Procedure

Creating a procedure

```
create or replace procedure jaya
(id in number,name in varchar)
is begin
insert into userr values(id,name);
end;
/
```

Calling a Procedure

```
Begin
jaya(101,'Raja');
dbms_output.put_line('Record is inserted');
end;
/
```

4. Triggers

```
CREATE OR REPLACE TRIGGER sal_change
BEFORE DELETE OR INSERT OR UPDATE ON cust
FOR EACH ROW
WHEN (NEW.ID > 0)
DECLARE
  sal_diff number;
BEGIN
  sal_diff := :NEW.sal  - :OLD.sal;
  dbms_output.put_line('Old salary: ' || :OLD.sal);
  dbms_output.put_line('New salary: ' || :NEW.sal);
  dbms_output.put_line('Salary difference: ' || sal_diff);
END;
/
```

Calling a Trigger

```
UPDATE cust
SET sal = sal + 500
WHERE id = 102;
```

5. Cursor

```
DECLARE
CURSOR dm IS SELECT name FROM cust;
lv_name cust.name%type;
BEGIN
OPEN dm;
LOOP
FETCH dm INTO lv_name;
IF dm%NOTFOUND
THEN
EXIT;
END IF;
Dbms_output.put_line('Employee Fetched:'||lv_name);
END LOOP;
Dbms_output.put_line('Total rows fetched is'||dm % rowcount);
CLOSE dm;
END;
/
```

6. Package

Creating a package

```
CREATE PACKAGE citi AS
   FUNCTION p_strng RETURN VARCHAR2;
END citi;
/
```

Creating a package body

```
CREATE OR REPLACE PACKAGE BODY citi AS
 --function implemented
  FUNCTION p_strng RETURN VARCHAR2 IS
    BEGIN
      RETURN 'Software Testing Help!';
    END p_strng;
END citi;
/
```

Calling a package

```
BEGIN
   DBMS_OUTPUT.PUT_LINE (citi.p_strng);
 END;
 /
```