1. Criar um bloco PL/SQL anônimo para imprimir a tabuada abaixo:

5 X 1 = 5
5 X 2 = 10
...
5 X 10 = 50

```
DECLARE
 V_N CONSTANT NUMBER(2) := 5;
BEGIN
 FOR I IN 1..10 LOOP
  DBMS_OUTPUT.PUT_LINE(V_N || ' X ' || I || ' = ' || V_N*I);
 END LOOP;
END;
/
```

2. Criar um bloco PL/SQL anônimo para imprimir as tabuadas abaixo:

1 X 1 = 1
1 X 2 = 2
...
10 X 9 = 90
10 X 10 = 100

```
BEGIN
 FOR I IN 1..10 LOOP
  FOR J IN 1..10 LOOP
   DBMS_OUTPUT.PUT_LINE(I || ' X ' || J || ' = ' || I*J);
  END LOOP;
 END LOOP;
END;
/
```

3. Criar um bloco PL/SQL para apresentar os anos bissextos entre 2000 e 2100. Um ano será bissexto quando for possível dividi-lo por 4, mas não por 100 ou quando for possível dividi-lo por 400.

```
DECLARE
 V_ANO NUMBER(4);
BEGIN
 FOR V_ANO IN 2000..2100 LOOP
  IF (MOD(V_ANO,4) = 0 AND MOD(V_ANO,100) != 0) OR (MOD(V_ANO,400) = 0) THEN
   DBMS_OUTPUT.PUT_LINE(V_ANO);
  END IF;
 END LOOP;
END;
/
```

4. Criar um bloco PL/SQL para atualizar a tabela abaixo, conforme segue:

Produtos categoria A deverão ser reajustados em 5%
Produtos categoria B deverão ser reajustados em 10%
Produtos categoria C deverão ser reajustados em 15%

PRODUTO

CODIGO	CATEGORIA	VALOR
1001	A	7.55
1002	B	5.95
1003	C	3.45

```
CREATE TABLE PRODUTO (
CODIGO NUMBER(4),
CATEGORIA CHAR(1),
VALOR NUMBER(4,2));

INSERT INTO PRODUTO VALUES (1001,'A',7.55);
INSERT INTO PRODUTO VALUES (1002,'B',5.95);
INSERT INTO PRODUTO VALUES (1003,'C',3.45);

--SOLUÇÃO 1
DECLARE
 CURSOR C_PRODUTO IS SELECT * FROM PRODUTO;
 V_PRODUTO PRODUTO%ROWTYPE;
BEGIN
 FOR V_PRODUTO IN C_PRODUTO LOOP
  IF V_PRODUTO.CATEGORIA = 'A' THEN
   UPDATE PRODUTO SET VALOR = VALOR * 1.05 WHERE CODIGO = V_PRODUTO.CODIGO;
  ELSIF V_PRODUTO.CATEGORIA = 'B' THEN
   UPDATE PRODUTO SET VALOR = VALOR * 1.10 WHERE CODIGO = V_PRODUTO.CODIGO;
  ELSE
   UPDATE PRODUTO SET VALOR = VALOR * 1.15 WHERE CODIGO = V_PRODUTO.CODIGO;
  END IF;
 END LOOP;
END;
/


--SOLUÇÃO 2
DECLARE
 CURSOR C_PRODUTO IS SELECT * FROM PRODUTO;
 V_PRODUTO PRODUTO%ROWTYPE;
 V_REAJUSTE NUMBER(3,2);
BEGIN
 FOR V_PRODUTO IN C_PRODUTO LOOP
  IF V_PRODUTO.CATEGORIA = 'A' THEN
   V_REAJUSTE := 1.05;
  ELSIF V_PRODUTO.CATEGORIA = 'B' THEN
   V_REAJUSTE := 1.10;
  ELSE
   V_REAJUSTE := 1.15;
  END IF;
   UPDATE PRODUTO SET VALOR = VALOR * V_REAJUSTE WHERE CODIGO =
V_PRODUTO.CODIGO;
 END LOOP;
END;
/
```

5. Criar um bloco PL/SQL para imprimir a sequência de Fibonacci: 1 1 2 3 5 8 13 21 34 55

```
DECLARE
 V_A NUMBER(2) := 1;
 V_B NUMBER(2) := 1;
 V_C NUMBER(2) := 0;
BEGIN
 FOR V_I IN 1..11 LOOP
  V_A := V_B;
  V_B := V_C;
  DBMS_OUTPUT.PUT_LINE(V_C);
  V_C := V_A + V_B;
 END LOOP;
END;
/
```

6. Criar uma procedure que deverá receber o código de um cliente e a partir deste dado imprimir o seu nome, e seu e-mail. Os dados deverão ser obtidos a partir de uma tabela chamada CLIENTE com as seguintes colunas (COD_CLI,NOME_CLI,EMAIL_CLI). Exemplo:

```
CLIENTE
-----------------------------------------------
COD_CLI  NOME_CLI           EMAIL_CLI
-----------------------------------------------
10       BEATRIZ BERNARDES  bb@dominio.com..br
-----------------------------------------------
```

```
CREATE TABLE CLIENTE (
COD_CLI NUMBER(4) PRIMARY KEY,
NOME_CLI VARCHAR2(30),
EMAIL_CLI VARCHAR2(30));

INSERT INTO CLIENTE VALUES (10,'BEATRIZ BERNARDES','bb@dominio.com.br');

CREATE OR REPLACE PROCEDURE MOSTRA_CLIENTE (
  P_COD_CLI NUMBER) IS
  V_CLIENTE CLIENTE%ROWTYPE;
BEGIN
  SELECT * INTO V_CLIENTE FROM CLIENTE WHERE COD_CLI = P_COD_CLI;
  DBMS_OUTPUT.PUT_LINE(V_CLIENTE.NOME_CLI || ' - ' || V_CLIENTE.EMAIL_CLI);
END MOSTRA_CLIENTE;
/
```

7. Criar uma procedure que receberá um RA, um NOME e quatro notas conforme a sequência: (RA,NOME,A1,A2,A3,A4). A partir destes valores deverá efetuar o cálculo da média somando o maior valor entre A1 e A2 às notas A3 e A4 e dividindo o valor obtido por três (achando a média). Se a média for menor que 6 (seis) o aluno estará REPROVADO e se a média for igual ou superior a 6 (seis) o aluno estará APROVADO. A procedure deverá inserir os valores acima numa tabela denominada ALUNO com as seguintes colunas RA,NOME,A1,A2,A3,A4,MEDIA,RESULTADO. Exemplo:

```
ALUNO
-----------------------------------------------------------
RA   NOME           A1   A2   A3   A4   MEDIA  RESULTADO
-----------------------------------------------------------
123  ANTONIO ALVES  6.5  3.5  9.5  5.0  7.0    APROVADO
-----------------------------------------------------------
```

```
CREATE TABLE ALUNO (
RA NUMBER(9),
NOME VARCHAR2(30),
NOTA1 NUMBER(3,1),
NOTA2 NUMBER(3,1),
NOTA3 NUMBER(3,1),
NOTA4 NUMBER(3,1),
MEDIA NUMBER(3,1),
RESULTADO VARCHAR2(15));


CREATE OR REPLACE PROCEDURE CALCULA_MEDIA (
 RA NUMBER,
 NOME VARCHAR2,
 N1 NUMBER,
 N2 NUMBER,
 N3 NUMBER,
 N4 NUMBER) IS
 V_MAIOR NUMBER(3,1);
 V_MEDIA NUMBER(3,1);
 V_RESULTADO VARCHAR2(15);
```

```
BEGIN
 IF N1 > N2 THEN V_MAIOR := N1;
 ELSE V_MAIOR := N2;
 END IF;
 V_MEDIA := (V_MAIOR + N3 + N4)/3;
 IF V_MEDIA < 6 THEN V_RESULTADO := 'REPROVADO';
 ELSE V_RESULTADO := 'APROVADO';
 END IF;
 INSERT INTO ALUNO VALUES (RA,NOME,N1,N2,N3,N4,V_MEDIA,V_RESULTADO);
END CALCULA_MEDIA;
/
```

8. Criar uma procedure que receberá o CÓDIGO de um PRODUTO. A partir deste dado deverá consultar uma tabela denominada PRODUTO e verificar a que CATEGORIA o produto pertence. Com base nesta informação deverá informar qual o valor (em Reais) do IPI consultando para isso uma tabela denominada ALIQUOTA. As tabelas PRODUTO e ALIQUOTA estão parcialmente representadas a seguir:

```
PRODUTO
-----------------------
COD_PRO  VALOR  COD_CAT
-----------------------
1001    120.00  A
1002    250.00  B
-----------------------

ALIQUOTA
---------------
COD_CAT     IPI
---------------
A           10
B           15
---------------
```

NOTA: Os valores do IPI estão representados em porcentagem (10%, 15%, etc.)

```
CREATE TABLE PRODUTO (
COD_PRO NUMBER(4) PRIMARY KEY,
VALOR NUMBER(6,2),
COD_CAT CHAR(1));

CREATE TABLE ALIQUOTA (
COD_CAT CHAR(1) PRIMARY KEY,
IPI NUMBER(2));

INSERT INTO PRODUTO VALUES (1001,120,'A');
INSERT INTO PRODUTO VALUES (1002,250,'B');

INSERT INTO ALIQUOTA VALUES ('A',10);
INSERT INTO ALIQUOTA VALUES ('B',15);

CREATE OR REPLACE PROCEDURE CALCULA_IPI (
 P_COD_PRO NUMBER) IS
 V_VALOR PRODUTO.VALOR%TYPE;
 V_IPI ALIQUOTA.IPI%TYPE;
 V_VALOR_IPI NUMBER(6,2);
BEGIN
 SELECT VALOR INTO V_VALOR FROM PRODUTO WHERE COD_PRO = P_COD_PRO;
 SELECT A.IPI INTO V_IPI FROM PRODUTO P INNER JOIN ALIQUOTA A
  ON P.COD_CAT = A.COD_CAT WHERE COD_PRO = P_COD_PRO;
 V_VALOR_IPI := V_VALOR * (V_IPI/100);
 DBMS_OUTPUT.PUT_LINE(V_VALOR_IPI);
END CALCULA_IPI;
/
```

9. Uma empresa oferece um bônus a seus funcionários com base no lucro liquido (tabela LUCRO) obtido durante o ano e no valor do salário do funcionário (tabela SALARIO). O bônus é calculado conforme a seguinte formula: BONUS = LUCRO * 0.01 + SALARIO * 0.05. Crie uma procedure que receba o ano (tabela LUCRO) e número de matricula do funcionário e devolva (imprima) o valor do seu respectivo bônus.

```
LUCRO
-----------------
ANO   VALOR
-----------------
2007  1200000
2008  1500000
2009  1400000
-----------------

SALARIO
-----------------
MATRICULA  VALOR
-----------------
1001       2500
1002       3200
-----------------
```

```
CREATE TABLE LUCRO (
ANO NUMBER(4),
VALOR NUMBER(9,2));

CREATE TABLE SALARIO (
MATRICULA NUMBER(4),
VALOR NUMBER(7,2));

INSERT INTO LUCRO VALUES (2007,1200000);
INSERT INTO LUCRO VALUES (2008,1500000);
INSERT INTO LUCRO VALUES (2009,1400000);

INSERT INTO SALARIO VALUES (1001,2500);
INSERT INTO SALARIO VALUES (1002,3200);

CREATE OR REPLACE PROCEDURE CALCULA_BONUS (
 P_ANO LUCRO.ANO%TYPE,
 P_MAT SALARIO.MATRICULA%TYPE) IS
 V_VL_LUCRO LUCRO.VALOR%TYPE;
 V_VL_SALARIO SALARIO.VALOR%TYPE;
 V_BONUS NUMBER(7,2);
BEGIN
 SELECT VALOR INTO V_VL_LUCRO FROM LUCRO
  WHERE ANO = P_ANO;
 SELECT VALOR INTO V_VL_SALARIO FROM SALARIO
  WHERE MATRICULA = P_MAT;
 V_BONUS := V_VL_LUCRO * 0.01 + V_VL_SALARIO * 0.05;
 DBMS_OUTPUT.PUT_LINE ('Valor do Bonus: ' || V_BONUS);
END;
/

EXEC CALCULA_BONUS (2007,1001);
```

10. Criar uma função que deverá receber um número inteiro e retornar se o mesmo é primo ou não. (Lembrete: Números primos são divisíveis somente por eles mesmos e por um).

```
CREATE OR REPLACE FUNCTION PRIMO (
 V_N NUMBER)
 RETURN VARCHAR2 IS
 V_SQRT NUMBER(4);
 V_DIVISOR NUMBER(4);
 V_RESULTADO VARCHAR2(12) := 'É PRIMO';
```

```
BEGIN
 V_SQRT := SQRT(V_N);
 FOR V_I IN 2..V_SQRT LOOP
  IF MOD(V_N,V_I) = 0 AND V_N <> V_I THEN
   V_RESULTADO := 'NÃO É PRIMO';
   V_DIVISOR := V_I;
  END IF;
 END LOOP;
 RETURN V_RESULTADO;
END;
/
```

11. Criar uma função que deverá receber um valor correspondente à temperatura em graus Fahrenheit e retornar o equivalente em graus Celsius. Fórmula para conversão: C = (F - 32) / 1.8

```
CREATE OR REPLACE FUNCTION F_TO_C (
 P_F NUMBER)
 RETURN NUMBER IS
 V_C NUMBER(4,1);
BEGIN
 V_C := (P_F - 32) / 1.8;
 RETURN V_C;
END F_TO_C;
/
```

12. Criar uma função que deverá receber o número de matrícula de um funcionário e retornar o seu nome e o nome de seu departamento, conforme as seguintes tabelas:

```
FUNCIONARIO
-----------------------------
MATRICULA  NOME     COD_DEPTO
-----------------------------
1001       ANTONIO  1
1002       BEATRIZ  2
1003       CLAUDIO  1
-----------------------------

-----------------------------
DEPARTAMENTO
-----------------------------
COD_DEPTO  NOME_DEPTO
-----------------------------
1          ENGENHARIA
2          INFORMATICA
-----------------------------
```

```
CREATE TABLE FUNCIONARIO (
MATRICULA NUMBER(4),
NOME VARCHAR2(30),
COD_DEPTO NUMBER(2));

CREATE TABLE DEPARTAMENTO (
COD_DEPTO NUMBER(2),
NOME_DEPTO VARCHAR2(20));

INSERT INTO FUNCIONARIO VALUES (1001,'ANTONIO',1);
INSERT INTO FUNCIONARIO VALUES (1002,'BEATRIZ',2);
INSERT INTO FUNCIONARIO VALUES (1003,'CLAUDIO',1);

INSERT INTO DEPARTAMENTO VALUES (1,'ENGENHARIA');
INSERT INTO DEPARTAMENTO VALUES (2,'INFORMATICA');
```

```
CREATE OR REPLACE FUNCTION CONSULTA_FUNC(
 P_MATRICULA NUMBER)
 RETURN VARCHAR2 IS
 V_NOME FUNCIONARIO.NOME%TYPE;
 V_NOME_DEPTO DEPARTAMENTO.NOME_DEPTO%TYPE;
 V_SAIDA VARCHAR2(100);
BEGIN
 SELECT NOME INTO V_NOME FROM FUNCIONARIO WHERE MATRICULA = P_MATRICULA;
 SELECT NOME_DEPTO INTO V_NOME_DEPTO FROM FUNCIONARIO
  INNER JOIN DEPARTAMENTO
  ON FUNCIONARIO.COD_DEPTO = DEPARTAMENTO.COD_DEPTO
  WHERE MATRICULA = P_MATRICULA;
 V_SAIDA := (V_NOME || ' - ' || V_NOME_DEPTO);
 RETURN V_SAIDA;
END CONSULTA_FUNC;
/
```

13. Criar uma trigger para implementar uma restrição para que o salário do funcionário (ver tabela a seguir) não possa ser reduzido.

```
FUNCIONARIO
-----------------------------
MATRICULA  NOME     SALARIO
-----------------------------
1001       ANTONIO  2500
1002       BEATRIZ  1800
1003       CLAUDIO  1500
-----------------------------
```

```
CREATE OR REPLACE TRIGGER VERIFICA_SALARIO
 BEFORE UPDATE OF SALARIO ON FUNCIONARIO
 FOR EACH ROW
 WHEN (NEW.SALARIO < OLD.SALARIO)
BEGIN
 RAISE_APPLICATION_ERROR (-20508, 'O salário não pode ser reduzido');
END;
/
```

14. Criar uma trigger para impedir que o salário do funcionário seja reajustado acima de 20% (vinte por cento). Utilize como base a mesma tabela do exercício anterior.

```
CREATE OR REPLACE TRIGGER AUMENTO
 BEFORE UPDATE OF SALARIO ON FUNCIONARIO
 FOR EACH ROW
BEGIN
 IF (:NEW.SALARIO - :OLD.SALARIO) > :OLD.SALARIO * 0.20 THEN
  RAISE_APPLICATION_ERROR (-20512, 'O aumento não deve ser maior que 20%');
 END IF;
END;
/
```