“Año de la recuperación y consolidación de la economía peruana”

UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS

Facultad de Ingeniería de Sistemas e Informática

Escuela profesional de Ingeniería de Software

Tarea 4

Asignatura: Base de Datos II

Docente: Jorge Luis Chávez Soto

Alumno:

- Chavez Gave, Jose Luis

Fecha: 13 de octubre

Lima - Perú

2025

1. **Creación de Tablespaces** y carga de datos

```
-- Tabla S (Suppliers)
CREATE TABLE S (
  S# VARCHAR2(2) PRIMARY KEY,
  SNAME VARCHAR2(50),
  STATUS NUMBER,
  CITY VARCHAR2(50)
);

-- Tabla P (Partes)
CREATE TABLE P (
  P# VARCHAR2(2) PRIMARY KEY,
  PNAME VARCHAR2(50),
  COLOR VARCHAR2(20),
  WEIGHT NUMBER,
  CITY VARCHAR2(50)
);

-- Tabla SP (Envíos)
CREATE TABLE SP (
  S# VARCHAR2(2),
  P# VARCHAR2(2),
  QTY NUMBER,
  PRIMARY KEY (S#, P#),
  FOREIGN KEY (S#) REFERENCES S(S#),
  FOREIGN KEY (P#) REFERENCES P(P#)
```

```
);

-- Tabla J (Proyectos)
CREATE TABLE J (
  J# VARCHAR2(2) PRIMARY KEY,
  JNAME VARCHAR2(50),
  CITY VARCHAR2(50)
);

-- Tabla SPJ (Envíos a Proyectos)
CREATE TABLE SPJ (
  S# VARCHAR2(2),
  P# VARCHAR2(2),
  J# VARCHAR2(2),
  QTY NUMBER,
  PRIMARY KEY (S#, P#, J#),
  FOREIGN KEY (S#) REFERENCES S(S#),
  FOREIGN KEY (P#) REFERENCES P(P#),
  FOREIGN KEY (J#) REFERENCES J(J#)
);
```

2. Desarrollo de procedimientos y funciones

a. Obtenga el color y ciudad para las partes que no son de París, con un peso mayor de diez.

```
CREATE OR REPLACE PROCEDURE partes_fuera_paris_peso_alto
IS
```

```
BEGIN
  FOR r IN (
    SELECT COLOR, CITY
    FROM P
    WHERE CITY <> 'Paris' AND WEIGHT > 10
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Color: ' || r.COLOR || ', Ciudad: ' ||
r.CITY);
  END LOOP;
END;
```

b. Para todas las partes, obtenga el número de parte y el peso de dichas partes en gramos.

```
CREATE OR REPLACE FUNCTION peso_en_gramos(p_numero
VARCHAR2) RETURN NUMBER IS
  v_peso_libras NUMBER;
BEGIN
  SELECT WEIGHT INTO v_peso_libras FROM P WHERE P# =
p_numero;
  RETURN v_peso_libras * 453.592;
END;
```

c. Obtenga el detalle completo de todos los proveedores.

```
CREATE OR REPLACE PROCEDURE detalle_proveedores IS
BEGIN
  FOR r IN (SELECT * FROM S) LOOP
    DBMS_OUTPUT.PUT_LINE('S#: ' || r.S# || ', Nombre: ' || r.SNAME ||
', Estado: ' || r.STATUS || ', Ciudad: ' || r.CITY);
```

```
  END LOOP;
END;
```

d. Obtenga todas las combinaciones de proveedores y partes para aquellos proveedores y partes co-localizados.

```
CREATE OR REPLACE PROCEDURE
proveedores_partes_colocalizados IS
BEGIN
  FOR r IN (
    SELECT S.S#, P.P#
    FROM S JOIN P ON S.CITY = P.CITY
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Proveedor: ' || r.S# || ', Parte: ' || r.P#);
  END LOOP;
END;
```

e. Obtenga todos los pares de nombres de ciudades de tal forma que el proveedor localizado en la primera ciudad del par abastece una parte almacenada en la segunda ciudad del par.

```
CREATE OR REPLACE PROCEDURE pares_ciudades_abastecimiento
IS
BEGIN
  FOR r IN (
    SELECT DISTINCT S.CITY AS CIUDAD_PROVEEDOR, P.CITY
AS CIUDAD_PARTE
    FROM S
    JOIN SP ON S.S# = SP.S#
    JOIN P ON SP.P# = P.P#
```

```
    WHERE S.CITY <> P.CITY
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Proveedor en: ' ||
r.CIUDAD_PROVEEDOR || ' → Parte en: ' || r.CIUDAD_PARTE);
  END LOOP;
END;
```

f. Obtenga todos los pares de número de proveedor tales que los dos proveedores del par estén co-localizados.

```
CREATE OR REPLACE PROCEDURE proveedores_colocalizados IS
BEGIN
  FOR r IN (
    SELECT DISTINCT A.S# AS PROV1, B.S# AS PROV2
    FROM S A, S B
    WHERE A.S# < B.S# AND A.CITY = B.CITY
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Par: ' || r.PROV1 || ' - ' || r.PROV2);
  END LOOP;
END;
```

g. Obtenga el número total de proveedores.

```
CREATE OR REPLACE FUNCTION total_proveedores RETURN
NUMBER IS
  v_total NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_total FROM S;
  RETURN v_total;
END;
```

h. Obtenga la cantidad mínima y la cantidad máxima para la parte P2.

```
CREATE OR REPLACE PROCEDURE min_max_p2 IS
  v_min NUMBER;
  v_max NUMBER;
BEGIN
  SELECT MIN(QTY), MAX(QTY) INTO v_min, v_max
  FROM SP
  WHERE P# = 'P2';

  DBMS_OUTPUT.PUT_LINE('Cantidad mínima: ' || v_min);
  DBMS_OUTPUT.PUT_LINE('Cantidad máxima: ' || v_max);
END;
```

i. Para cada parte abastecida, obtenga el número de parte y el total despachado.

```
CREATE OR REPLACE PROCEDURE total_despachado_por_parte IS
BEGIN
  FOR r IN (
    SELECT P#, SUM(QTY) AS TOTAL
    FROM SP
    GROUP BY P#
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Parte: ' || r.P# || ' → Total: ' ||
r.TOTAL);
  END LOOP;
END;
```

j. Obtenga el número de parte para todas las partes abastecidas por más de un proveedor.

```
CREATE OR REPLACE PROCEDURE partes_multi_proveedor IS
BEGIN
  FOR r IN (
    SELECT P#
    FROM SP
    GROUP BY P#
    HAVING COUNT(DISTINCT S#) > 1
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Parte abastecida por múltiples
proveedores: ' || r.P#);
  END LOOP;
END;
```

k. Obtenga el nombre de proveedor para todos los proveedores que abastecen la parte P2.

```
CREATE OR REPLACE PROCEDURE proveedores_para_p2 IS
BEGIN
  FOR r IN (
    SELECT DISTINCT S.SNAME
    FROM S
    JOIN SP ON S.S# = SP.S#
    WHERE SP.P# = 'P2'
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Proveedor de P2: ' || r.SNAME);
  END LOOP;
END;
```

l. Obtenga el nombre de proveedor de quienes abastecen por lo menos una parte.

```
CREATE OR REPLACE PROCEDURE proveedores_activos IS
BEGIN
  FOR r IN (
    SELECT DISTINCT S.SNAME
    FROM S
    WHERE EXISTS (
      SELECT 1 FROM SP WHERE SP.S# = S.S#
    )
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Proveedor activo: ' || r.SNAME);
  END LOOP;
END;
```

m. Obtenga el número de proveedor para los proveedores con estado menor que el máximo valor de estado en la tabla S.

```
CREATE OR REPLACE PROCEDURE proveedores_estado_menor IS
  v_max_status NUMBER;
BEGIN
  SELECT MAX(STATUS) INTO v_max_status FROM S;

  FOR r IN (
    SELECT S# FROM S WHERE STATUS < v_max_status
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Proveedor con estado menor: ' || r.S#);
  END LOOP;
END;
```

n. Obtenga el nombre de proveedor para los proveedores que abastecen la parte P2 (aplicar EXISTS en su solución).

```
CREATE OR REPLACE PROCEDURE proveedores_exist_p2 IS
BEGIN
  FOR r IN (
    SELECT SNAME
    FROM S
    WHERE EXISTS (
      SELECT 1 FROM SP WHERE SP.S# = S.S# AND SP.P# = 'P2'
    )
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Proveedor que abastece P2 (EXISTS):
' || r.SNAME);
  END LOOP;
END;
```

o. Obtenga el nombre de proveedor para los proveedores que no abastecen la parte P2.

```
CREATE OR REPLACE PROCEDURE proveedores_sin_p2 IS
BEGIN
  FOR r IN (
    SELECT SNAME
    FROM S
    WHERE NOT EXISTS (
      SELECT 1 FROM SP WHERE SP.S# = S.S# AND SP.P# = 'P2'
    )
  ) LOOP
```

```
    DBMS_OUTPUT.PUT_LINE('Proveedor que NO abastece P2: ' ||
r.SNAME);
  END LOOP;
END;
```

p. Obtenga el nombre de proveedor para los proveedores que abastecen todas las partes.

```
CREATE OR REPLACE PROCEDURE proveedores_todas_partes IS
BEGIN
  FOR r IN (
    SELECT S.SNAME
    FROM S
    WHERE NOT EXISTS (
      SELECT P.P#
      FROM P
      MINUS
      SELECT SP.P#
      FROM SP
      WHERE SP.S# = S.S#
    )
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Proveedor que abastece todas las
partes: ' || r.SNAME);
  END LOOP;
END;
```

q. Obtenga el número de parte para todas las partes que pesan más de 16 libras ó son abastecidas por el proveedor S2, ó cumplen con ambos criterios.

```
CREATE OR REPLACE PROCEDURE partes_peso_o_s2 IS
BEGIN
  FOR r IN (
    SELECT DISTINCT P.P#
    FROM P
    LEFT JOIN SP ON P.P# = SP.P#
    WHERE P.WEIGHT > 16 OR SP.S# = 'S2'
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Parte que cumple condición: ' || r.P#);
  END LOOP;
END;
```