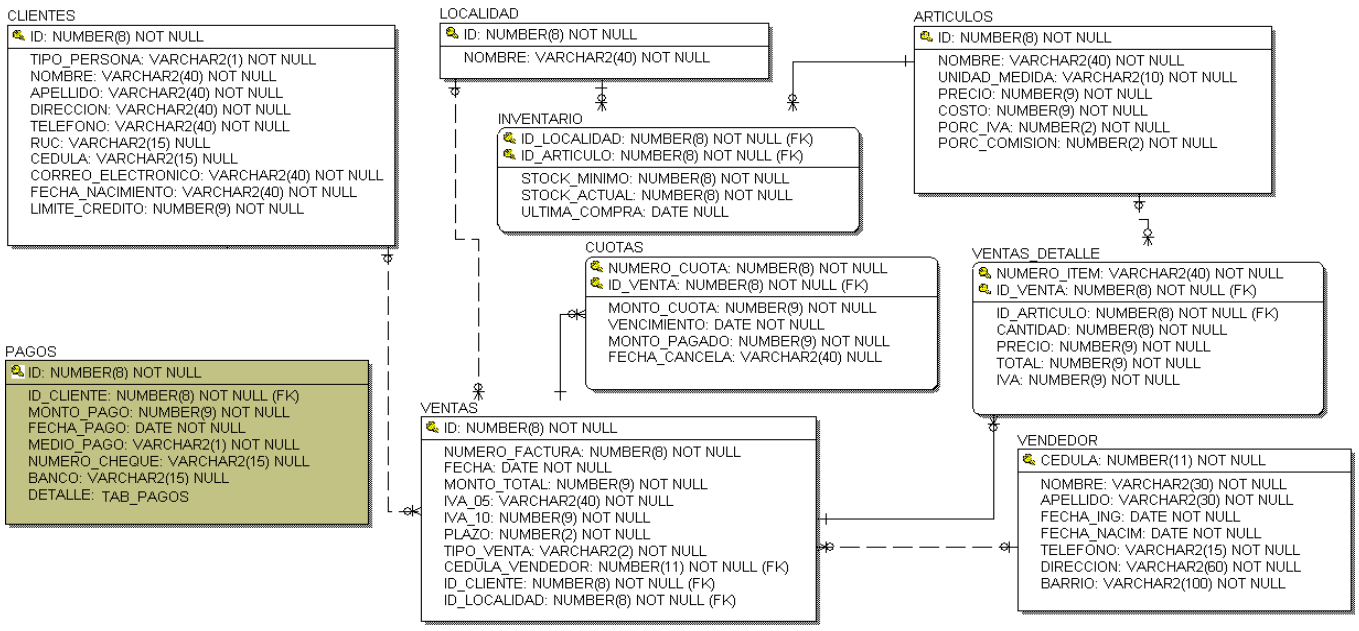
| Carrera | CIENCIAS INFORMÁTICAS | | | | Asignatura: | BASE DE DATOS 2 | | | | FECHA | Tema1 | 35 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Examen: | 3er Parcial | | 1er FINAL | | Sección (marcar) | A | | B | | 26/11/2015 | Tema2 | 30 | |
| Profesores | | | Alumno | | | | | | | | Tema3 | 15 | |
| Severino Marco Leiva Fernández | | | Cédula | | | | | | | | Tema 4 | 20 | |
| Carmen N. Martínez Wenninger | | | | | | | | | | | TOTAL | 100 | |




A continuación se muestra un sistema de venta de artículos que opera a crédito. Se pide le siguiente:

**TEMA 1: [35P] Cree los siguientes objetos en la Base de Datos**

1. El tipo **TAB_ PAGOS** como un VARRAY de **10** elementos compuesto de los siguientes atributos (**5P**)

| ID_VENTA | NUMERO_CUOTA | MONTO_PAGADO |
|---|---|---|
| NUMBER(8) | NUMBER(8) | NUMBER(9) |

2. El tipo **TIPO_PAGOS** como un objeto con los siguientes atributos (**5P** especificación, **15P** Miembro, **5P** Map)

| ID_PAGO | ID_CLIENTE | MONTO_PAGO | FECHA_PAGO | MEDIO | CHEQUE | BANCO | DETALLE |
|---|---|---|---|---|---|---|---|
| NUMBER(8) | NUMBER(8) | NUMBER(9) | DATE | VARCHAR2(1) | VARCHAR2(15) | VARCHAR2(15) | TAB_PAGOS |

Y los siguientes métodos:

**- ASIGNAR_DETALLE:** Este método verifica el atributo MONTO_PAGADO y recorre las **CUOTAS** pendientes de pago correspondientes al CLIENTE en orden descendente de vencimiento. Por cada cuota:
Verifica si el MONTO_PAGADO >= MONTO_CUOTA, entonces actualiza el monto_pagado y la fecha_cancela de la CUOTA, e incorpora el ID de la venta cuya cuota se pagó, en el varray DETALLE. Si el saldo del MONTO_PAGADO no llega a cubrir una cuota, igual asigna dicho saldo y actualiza el monto pagado, pero no así la fecha de cancelación de la cuota.

- Un método MAP **ORDENAR_PAGO** que ordena el objeto por ID_PAGO.

3. Cree en la BD la tabla **PAGOS** como una tabla de objeto (OBJECT TABLE) de TIPO_PAGOS : (**5P**)

```
--1 –Varray
CREATE TYPE T_PAGO IS OBJECT
(ID_VENTA NUMBER(8),
 NUMERO_CUOTA NUMBER(8),
 MONTO_PAGADO NUMBER(9));
/
CREATE TYPE TAB_PAGOS IS VARRAY(10) OF T_PAGO;
/
-- Tipo Object y sus métodos

CREATE TYPE TIPO_PAGOS IS OBJECT
(ID_PAGO     NUMBER(8),
 ID_CLIENTE  NUMBER(8),
 MONTO_PAGO  NUMBER(9),
 FECHA_PAGO  DATE,
 MEDIO       VARCHAR2(1),
 CHEQUE      VARCHAR2(15),
 BANCO       VARCHAR2(15),
 DETALLE     TAB_PAGOS,
 MEMBER PROCEDURE ASIGNAR_DETALLE,
 MAP MEMBER FUNCTION ORDENAR_PAGO RETURN NUMBER)
/
CREATE OR REPLACE TYPE BODY TIPO_PAGOS IS
 MEMBER PROCEDURE ASIGNAR_DETALLE IS
```

| Carrera | CIENCIAS INFORMÁTICAS | | | | Asignatura: | BASE DE DATOS 2 | | | | FECHA | Tema1 | 35 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Examen: | 3er Parcial | | 1er FINAL | | Sección (marcar) | A | | B | | 26/11/2015 | Tema2 | 30 | |
| Profesores | | | Alumno | | | | | | | | Tema3 | 15 | |
| Severino Marco Leiva Fernández | | | Cédula | | | | | | | | Tema 4 | 20 | |
| Carmen N. Martínez Wenninger | | | | | | | | | | | TOTAL | 100 | |

```
V_MONTO_PAGADO NUMBER(9) := SELF.MONTO_PAGO;
V_SALDO  NUMBER;
V_FECHA DATE;
V_DETALLE      TAB_PAGOS := TAB_PAGOS();
CURSOR  C_CUOTAS IS
        SELECT C.* FROM CUOTAS C JOIN VENTAS V
        ON V.ID = C.ID_VENTA
        WHERE  NVL(C.MONTO_CUOTA,0) - NVL(C.MONTO_PAGADO,0) > 0
        AND V.ID_CLIENTE = SELF.ID_CLIENTE
        ORDER  BY FECHA_CANCELA DESC
        FOR UPDATE;
V_IND NUMBER := 1;
BEGIN
 IF NVL(V_MONTO_PAGADO,0) > 0 THEN
    FOR REG IN C_CUOTAS LOOP
        V_SALDO:= NVL(REG.MONTO_CUOTA,0) - NVL(REG.MONTO_PAGADO,0);
        IF V_MONTO_PAGADO < V_SALDO THEN
           V_SALDO := V_MONTO_PAGADO;
           V_FECHA := NULL;
        ELSE
           V_FECHA := SYSDATE;
        END IF;
        UPDATE CUOTAS SET MONTO_PAGADO = NVL(MONTO_PAGADO,0)  + V_SALDO,
                      FECHA_CANCELA = V_FECHA
         WHERE CURRENT OF C_CUOTAS;
        V_MONTO_PAGADO := V_MONTO_PAGADO - V_SALDO;
        V_DETALLE.EXTEND();
        V_DETALLE(V_IND) := T_PAGO(REG.ID_VENTA, REG.NUMERO_CUOTA, V_SALDO);
        EXIT WHEN V_MONTO_PAGADO = 0;
        V_IND := V_IND + 1;
     END LOOP;
     SELF.DETALLE := V_DETALLE;
   END IF;
 END;
 MAP MEMBER FUNCTION ORDENAR_PAGO RETURN NUMBER IS
     BEGIN
        RETURN SELF.ID_PAGO;
     END;
END;
/
-- Object Table
CREATE TABLE PAGOS OF TIPO_PAGOS;
```

**TEMA 2: [30P] Controles en la BD**

1. ANTES de **INSERTAR** en la tabla PAGOS: El trigger asignará la columna DETALLE, aplicando el método ASIGNAR_DETALLE (**10P**)
2. ANTES de **INSERTAR** en la tabla VENTAS_DETALLE, actualizará el STOCK correspondiente al INVENTARIO de la localidad de la venta siempre que CANTIDAD <= STOCK_ACTUAL (no permitirá en caso contrario), y asignará también los campos PRECIO (a partir del precio del artículo), TOTAL (PRECIO * CANTIDAD) e IVA (TOTAL * PORC_IVA/100) (**20P**)

```
CREATE OR REPLACE TRIGGER T_BI_PAGOS
BEFORE INSERT ON PAGOS
FOR EACH ROW
DECLARE
  V_PAGO TIPO_PAGOS := TIPO_PAGOS(:NEW.ID_PAGO, :NEW.ID_CLIENTE, :NEW.MONTO_PAGO,
:NEW.FECHA_PAGO,  :NEW.MEDIO, :NEW.CHEQUE, :NEW.BANCO, :NEW.DETALLE);
BEGIN
  V_PAGO.ASIGNAR_DETALLE();
```

| Carrera | CIENCIAS INFORMÁTICAS | | | | Asignatura: | BASE DE DATOS 2 | | | FECHA | Tema1 | 35 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Examen: | 3er Parcial | | 1er FINAL | | Sección (marcar) | A | | B | 26/11/2015 | Tema2 | 30 | |
| Profesores | | | Alumno | | | | | | | Tema3 | 15 | |
| Severino Marco Leiva Fernández | | | Cédula | | | | | | | Tema 4 | 20 | |
| Carmen N. Martínez Wenninger | | | | | | | | | | TOTAL | 100 | |

```
 :NEW.DETALLE := V_PAGO.DETALLE;
END;
/
CREATE OR REPLACE TRIGGER T_BEF_IUD
BEFORE INSERT ON VENTAS_DETALLE
FOR EACH ROW
 DECLARE
   V_PRECIO NUMBER;
   V_STOCK  NUMBER;
   V_PORC   NUMBER(2);
   V_LOCAL  LOCALIDAD.ID%TYPE;
   PROCEDURE P_OBTENER_DATOS(PIDARTICULO NUMBER, PLOCAL NUMBER, PPRECIO OUT
NUMBER, PPORC OUT NUMBER, PSTOCK OUT NUMBER)
   IS
     BEGIN
      SELECT PRECIO, PORC_IVA INTO PPRECIO, PPORC
      FROM ARTICULOS WHERE ID = PIDARTICULO;
      SELECT STOCK_ACTUAL INTO PSTOCK
      FROM   INVENTARIO
      WHERE  ID_ARTICULO = PIDARTICULO AND ID_LOCALIDAD = PLOCAL;
     EXCEPTION
        WHEN NO_DATA_FOUND THEN
            PSTOCK := 0;
            PPRECIO := 0;
            PPORC   := 0;
   END;
   PROCEDURE P_ACTUALIZAR_STOCK(PIDARTICULO NUMBER, PLOCAL NUMBER, PCANTIDAD
NUMBER) IS
       BEGIN
           UPDATE INVENTARIO SET STOCK_ACTUAL = NVL(STOCK_ACTUAL,0) + PCANTIDAD
           WHERE ID_ARTICULO = PIDARTICULO AND ID_LOCALIDAD = PLOCAL;
       END;
 BEGIN
    SELECT ID_LOCALIDAD INTO V_LOCAL FROM VENTAS
    WHERE  ID = :NEW.ID_VENTA;
    P_OBTENER_DATOS(:NEW.ID_ARTICULO, V_LOCAL, V_PRECIO,  V_PORC, V_STOCK);
    IF  V_STOCK >= :NEW.CANTIDAD THEN
        P_ACTUALIZAR_STOCK(:NEW.ID_ARTICULO, V_LOCAL, :NEW.CANTIDAD);
        :NEW.PRECIO := V_PRECIO;
        :NEW.TOTAL  := V_PRECIO * :NEW.CANTIDAD;
        :NEW.IVA    := ROUND(:NEW.TOTAL* V_PORC/100, 0);
   ELSE
        RAISE_APPLICATION_ERROR(-20001, 'La cantidad excede al stock actual');
   END IF;
 END;
/
```

**TEMA 3: [15P] Cree la vista materializada. Actualización completa cada 30 minutos a partir de su creación**

Cree la vista materializada V_STOCK que muestra el Stock actual de las localidades del sitio central. El STOCK de Ciudad del Este se obtiene a partir de la tabla remota B_ARTICULOS ubicada en CDE, para la cual deberá usar el DBLINK **db_dist01** (Para probar su sintaxis ejecute la vista con B_ARTICULOS sin el DBLINK, y luego complete con el DBLINK)

| Artículo | Asunción | Fernando de la Mora | Lambaré | Ciudad del Este |
|---|---|---|---|---|
| <nombre artículo> | <stock_actual> | <stock_actual> | <stock_actual> | <stock_actual> |

| Carrera | CIENCIAS INFORMÁTICAS | | | | Asignatura: | BASE DE DATOS 2 | | | | FECHA | Tema1 | 35 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Examen: | 3er Parcial | | 1er FINAL | | Sección (marcar) | A | | B | | 26/11/2015 | Tema2 | 30 | |
| Profesores | | | Alumno | | | | | | | | Tema3 | 15 | |
| Severino Marco Leiva Fernández | | | Cédula | | | | | | | | Tema 4 | 20 | |
| Carmen N. Martínez Wenninger | | | | | | | | | | | TOTAL | 100 | |

```
CREATE MATERIALIZED VIEW V_VENTAS
REFRESH START WITH SYSDATE NEXT SYSDATE + 1/48
AS
SELECT NOMBRE_ARTICULO,
SUM(DECODE(UPPER(NOMBRE), 'ASUNCIÓN', STOCK_ACTUAL,0)) ASUNCION,
SUM(DECODE(UPPER(NOMBRE), 'FERNANDO DE LA MORA', STOCK_ACTUAL,0)) FERNANDO,
SUM(DECODE(UPPER(NOMBRE), 'LAMBARÉ', STOCK_ACTUAL,0)) LAMBARE,
SUM(DECODE(UPPER(NOMBRE), 'CIUDAD DEL ESTE', STOCK_ACTUAL,0)) CIUDAD_DEL_ESTE
FROM
(
 SELECT
  A.NOMBRE NOMBRE_ARTICULO, L.NOMBRE, I.STOCK_ACTUAL FROM
  INVENTARIO I JOIN ARTICULOS A
  ON A.ID = I.ID_ARTICULO
  JOIN  LOCALIDAD L
  ON    L.ID = I.ID_LOCALIDAD
  UNION
SELECT A.NOMBRE, 'CIUDAD DEL ESTE', A.STOCK_ACTUAL
FROM B_ARTICULOS@db_dist01 A) ST
GROUP BY NOMBRE_ARTICULO;
```

**TEMA 4: [20P] Sólo para los que rinden final. Concesión de accesos**

Para conceder el acceso a todos los usuarios  a las tablas de su esquema: Cree en rol R_CONSULTA.

Cree el procedimiento P_ACCESO que recorrerá todas las tablas de su esquema (que no empiezan con ‘B_’) y hará lo siguiente:

- Asignará  a dichas tablas un sinónimo público
- Granteará el derecho de SELECT sobre dichas tablas al rol R_CONSULTA
- Granterá el rol R_CONSULTA a todos los usuarios que empiezan con ‘BASED%’;

```
CREATE ROLE R_CONSULTA;

CREATE OR REPLACE PROCEDURE P_ACCESO IS
 CURSOR C_TABLAS IS
      SELECT TABLE_NAME
      FROM USER_TABLES
      WHERE TABLE_NAME NOT LIKE 'B\_%' ESCAPE '\';
  CURSOR C_USU IS
      SELECT USERNAME FROM ALL_USERS WHERE USERNAME LIKE 'BASED%';
  V_SENTENCIA VARCHAR2(1000);
BEGIN
  FOR REG IN C_TABLAS LOOP
      V_SENTENCIA := 'CREATE PUBLIC SYNONYM '|| REG.TABLE_NAME ||
                     ' FOR '|| REG.TABLE_NAME;
      DBMS_OUTPUT.PUT_LINE(V_SENTENCIA);
      EXECUTE IMMEDIATE V_SENTENCIA;
      V_SENTENCIA := 'GRANT SELECT ON '|| REG.TABLE_NAME || ' TO R_CONSULTA';
      DBMS_OUTPUT.PUT_LINE(V_SENTENCIA);
      EXECUTE IMMEDIATE V_SENTENCIA;
   END LOOP;
   FOR REG IN C_USU LOOP
       V_SENTENCIA := 'GRANT R_CONSULTA TO '|| REG.USERNAME;
       DBMS_OUTPUT.PUT_lINE(V_SENTENCIA);
       EXECUTE IMMEDIATE V_SENTENCIA;
   END LOOP;
END;
/
```