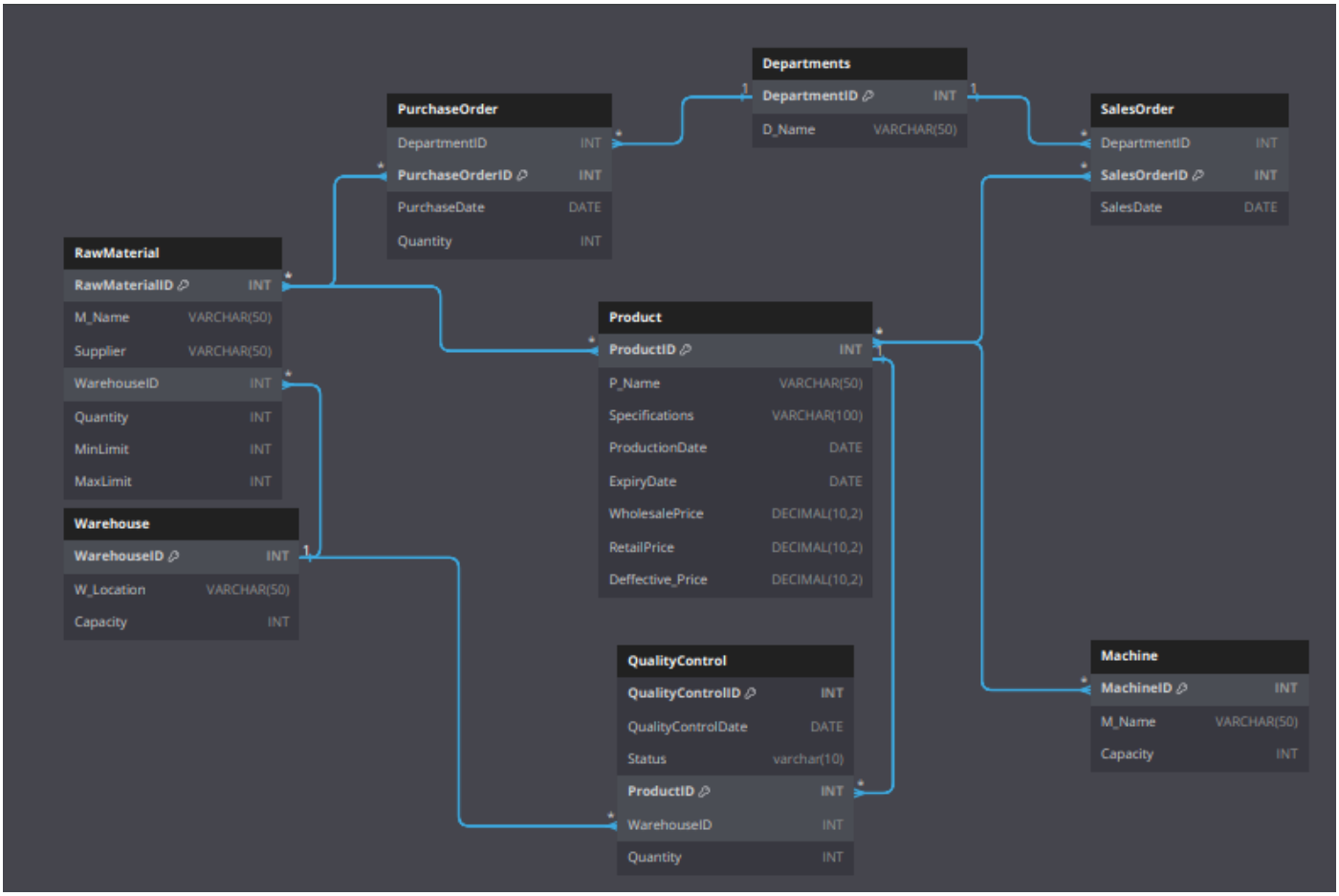
BDBL-501

HW-S23

الاسم	الرقم	الصف
يزن جوريه	Yazan_174681	C1
يزن حبيب	Yazan_154417	C1
عبدالرحمن شوا	Abdulrahman_243065	C1

مخطط الـ ERD:



شرح الكيانات ضمن الجدول و العلاقات بينها:

المنتجات Product:

- ProductID: هو معرف فريد لكل منتج.
- P_Name: اسم المنتج.
- Spesification وهي توفر مواصفات أو تفاصيل المنتج.
- ProductionDate: يمثل تاريخ إنتاج المنتج.
- Expiry Date: يشير إلى تاريخ انتهاء صلاحية المنتج.
- Wholesale Price: سعر الجملة للمنتج.
- Retail Price: سعر التجزئة للمنتج.
- Deffective_Price: يحدد سعر المنتج المعيب.

إدارة الجودة QualityControle:

- QualityControl: تمثل عملية مراقبة الجودة للمنتج.
- Quality ControlID: هو المعرف الخاص بمراقبة الجودة.
- Quality ControlDate: يخزن هذا الحقل تاريخ مراقبة الجودة.
- Status: تشير إلى حالة المنتج.

- ProductID: معرف المنتج.
- WarehouseID: يشير إلى معرف المستودع.

المواد الخام Raw Material :
- RawMaterialID: هو معرف فريد لكل صنف من المواد الخام.
- M Name: يمثل اسم المادة الخام.
- Supplier: يحتوي على اسم المورد الذي يتم شراء المادة الخام منه.
- WarehouseID: يمثل معرف المستودع الذي يتم تخزين المادة الخام فيه.
- Quantity: يقوم هذا الحقل بتخزين كمية المادة الخام المتوفرة في المستودع.
- MinLimit: يحدد الحد الأدنى لكمية المواد الخام المطلوبة في المستودع.
- MaxLimit: يشير إلى الحد الأقصى لكمية المواد الخام التي يمكن تخزينها في المستودع.

المستودع Warehouses :
- Warehouse: يخزن فيه اسم المستودع.
- WarehouseID: هو المعرف الخاص بالمستودع.

الأقسام Departments:
- DepartmentID: وهو معرف فريد لكل قسم.
- D_Name: يمثل هذا الحقل اسم القسم.

أمر شراء PurchaseOrder:
- DepartmentID: يمثل معرف القسم المتعلق بأمر الشراء.
- Purchase OrderID: هو معرف فريد لكل أمر شراء.
- PurchaseDate: يخزن هذا الحقل تاريخ أمر الشراء.
- Quantity: يحدد كمية المادة الخام المراد شراؤها.

أمر مبيع SalesOrder:
- DepartmentID: وهو معرف فريد لكل قسم.
- SalesOrderID: وهو معرف فريد لكل أمر مبيعات.
- SalesDate: يقوم هذا الحقل بتخزين تاريخ أمر المبيعات.

الآلات Machine:
- MachineID: هو معرف فريد لكل جهاز.
- M Name: هذا الحقل يحمل اسم الجهاز.
- Capacity: تحدد سعة الآلة.

هذه نظرة عامة على مخطط قاعدة البيانات المحدد. ويتضمن جداول مختلفة مثل المواد الخام وطلب الشراء والمنتج ومراقبة الجودة والأقسام وطلب المبيعات. تقوم هذه الجداول بتخزين معلومات حول المواد الخام وأوامر الشراء والمنتجات وعمليات مراقبة الجودة والأقسام وأوامر المبيعات. وهي مترابطة باستخدام علاقات المفتاح الأساسي والمفتاح الخارجي، مما يسمح بإدارة فعالة للبيانات المتعلقة بعمليات المنظمة في مجالات المشتريات والإنتاج ومراقبة الجودة والمخزون والمبيعات.

و هذه العلاقات كالتالي:

1- يرتبط جدول المنتجات مع جدول المواد الخام بعلاقة M-M Relation و هي علاقة متعدد لمتعدد أي انه يمكن لكل منتج أن تتألف من أكثر من مادة خام وكل مادة خام يمكن أن تستخدم في أكثر من منتج.

2- يرتبط جدول المنتجات مع جدول الآلات بعلاقة M-M Relation و هي علاقة متعدد لمتعدد أي انه يمكن لكل منتج أن تنتج من مجموعة من الآلات و يمكن لمجموعة من الآلات أن تنتج أكثر من منتج.

3- يرتبط جدول المنتجات مع جدول المخازن بعلاقة M-M Relation و هي علاقة متعدد لمتعدد أي انه يمكن لكل منتج أن تخزن في أكثر من جدول و يمكن لجدول أن يخزن أكثر من مادة.

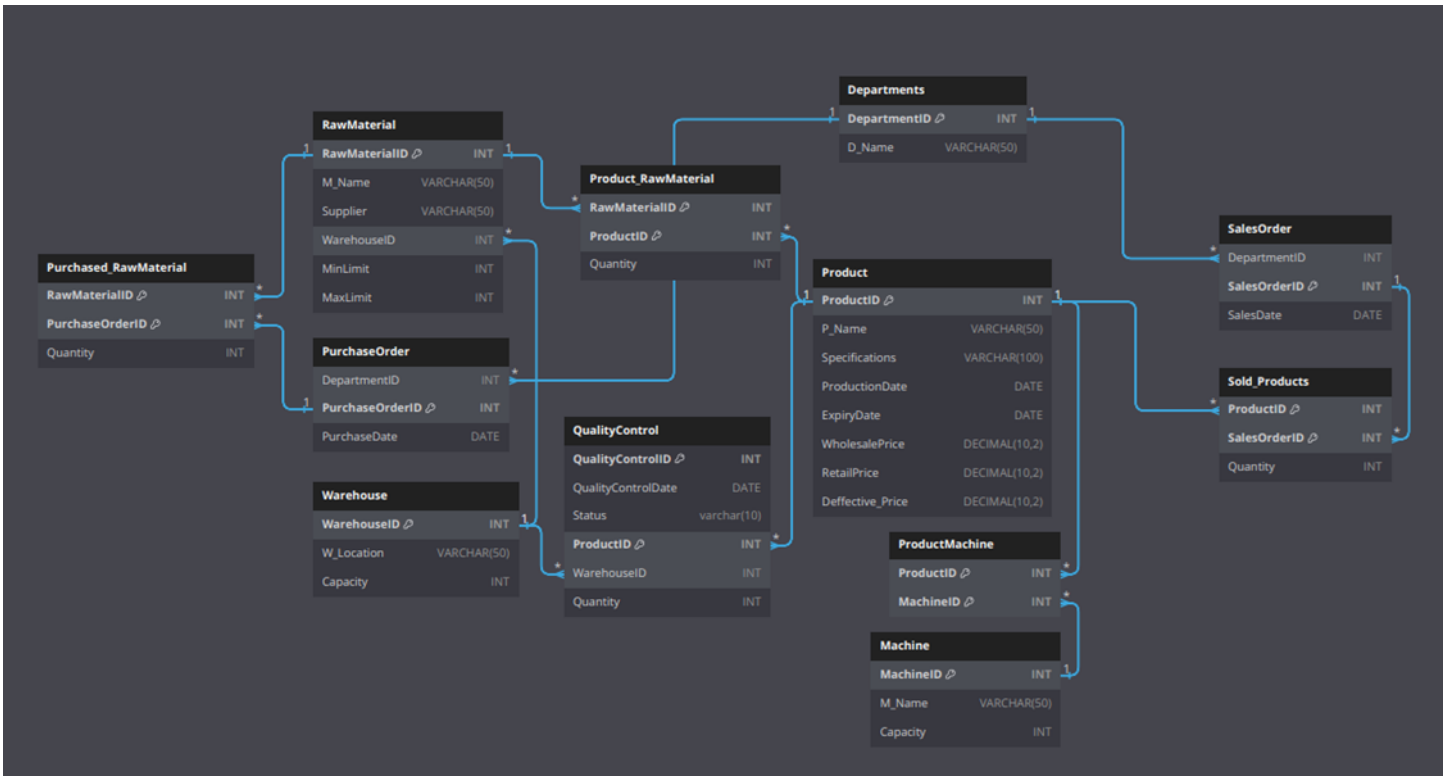
4- يرتبط جدول المنتجات مع جدول أمر البيع بعلاقة M-M Relation و هي علاقة متعدد لمتعدد أي انه يمكن لكل منتج أن تباع من خلال أكثر من أمر بيع و يمكن لأمر البع أن يحتوي أكثر من منتج.

5- يرتبط جدول المواد الخام مع جدول أمر الشراء بعلاقة M-M Relation و هي علاقة متعدد لمتعدد أي انه يمكن لكل مادة أن يتم شراؤها من خلال أكثر من أمر شراء و يمكن لأمر الشراء أن يحتوي أكثر من مادة.

6- يرتبط جدول المواد الخام مع جدول المخاز بعلاقة واحد لمتعدد حيث أنه يوجد مخزن وحيد للمواد الخام.

7- يرتبط جدول الأقسام مع كل من أمر البيع و أمر الشراء بعلاقة واحد لمتعدد حيث أنه يمكن للقسم الواحد أن يقوم باكثر من أمر بيع أو شراء و لكن لا يمكن أن يتم أمر الشراء من خلال أكثر من قسم.

ننتقل بعد ذلك لتحقيق هذه العلاقات في المخطط :



قمنا بإضافة عدة كيانات لتحقيق العلاقات و هي :

- Product_RawMaterial :

و هذا الجدول يحقق العلاقة بين جدول المنتجات و جدول المواد الأولية و يتألف من :

- RawMaterialID و يحتوي معرف المادة الأولية و هومفتاح اساسي
- ProductID و يحتوي معرف المنتج و هو مفتاح اساسي
- Quantity يحتوي على كمية المادة الأولية المخصصة لمنتج معين

و الهدف من هذا الجدول هو تخصيص مجموعة من المواد الأولية لمنتج معين و سيستخدم في عملية الإنتاج.

- Purchased_RawMterial :

و هذا الجدول يحقق العلاقة بين جدول أمر الشراء و جدول المواد الأولية و يتألف من :

- RawMaterialID و يحتوي معرف المادة الأولية و هومفتاح اساسي
- PurchaseOrderID و يحتوي معرف أمر الشراءو هو مفتاح اساسي
- Quantity يحتوي على كمية المادة الأولية المشتراة

و تمثل هذة الكميات مع الكميات الموجودة في جدول Product_RawMaterial الرصيد الكلي للمواد الأولية ضمن مخزن المواد الأولية.

- إدارة الجودة QualityControle:

يربط هذا الجدول جدول المنتجات بجدول المخازن و يحتوي:

- QualityControl: تمثل عملية مراقبة الجودة للمنتج.
- Quality ControlID: هو المعرف الخاص بمراقبة الجودة.
- Quality ControlDate: يخزن هذا الحقل تاريخ مراقبة الجودة.
- Status: تشير إلى حالة المنتج.
- ProductID: معرف المنتج.
- WarehouseID: يشير إلى معرف المستودع.

لا يمكن لمادة أن تدخل المخزن دون المرور على إدارة الجودة للتأكد من سلامتها و يمثل الإدراج أو التعديل على هذا الجدول تعديل بشكل مباشر على رصيد المواد.

- Sold_Products :

و هذا الجدول يحقق العلاقة بين جدول أمر المبيع و جدول المنتجات و يتألف من :

- ProductID و يحتوي معرف المنتج و هومفتاح اساسي
- SalesOrderID و يحتوي معرف أمر البيع و هو مفتاح اساسي
- Quantity يحتوي على كمية المنتج المباع

عمليات إنشاء الجداول:

1. Create Table Product:

```
--create products table
Create Table Product (
    Productid Int  Generated By Default As Identity,
    P_Name Varchar(50) NOT NULL,
    Specifications VARCHAR(100) NOT NULL,
    ProductionDate DATE Not Null,
    ExpiryDate DATE Not Null,
    WholesalePrice DECIMAL(10,2) Not Null,
    Retailprice Decimal(10,2) Not Null,
    Deffective_Price Decimal(10,2),
    PRIMARY KEY (Productid)
);
```

2. Create Table Department:

```
--create departments table (sales,purchasement)
Create Table Departments (
    Departmentid Int Generated By Default As Identity ,
    D_Name Varchar(50) Not Null,
    Primary Key(Departmentid)
);
```

3. Create Table Purchaseorder:

```
-- Create the PurchaseOrder table
Create Table Purchaseorder (
    Departmentid Int Not Null,
    Purchaseorderid Int Generated By Default As Identity,
    Purchasedate Date Not Null,
    Primary Key(Purchaseorderid),
    FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
    );
```

4. Create Table SalesOrder:

```
 -- Create the SalesOrder table
Create Table Salesorder (
    Departmentid Int Not Null,
    Salesorderid Int Generated By Default As Identity,
    Salesdate Date Not Null,
    Primary Key(Salesorderid),
    FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);
```

5. Create Table Warehouse:

```
--Create warehouses table(Rawmaterial,validProducts,invalidProducts)
Create Table Warehouse (
    Warehouseid Int Generated By Default As Identity,
    W_Location Varchar(50) Not Null,
    Capacity Int Not Null,
    Primary Key(Warehouseid)
);
```

6. Create Table Rawmaterial:

```
--create raw material table
Create Table Rawmaterial (
    Rawmaterialid Int Generated By Default As Identity,
    M_Name Varchar(50) Not Null,
    Supplier Varchar(50) Not Null,
    Warehouseid Int Not Null,
    Minlimit Int Not Null,
    Maxlimit Int Not Null,
    Primary Key(Rawmaterialid),
    FOREIGN KEY (WarehouseID) REFERENCES Warehouse(WarehouseID)
);
```

7. Create Table Machine:

```
--create the table of machines
Create Table Machine (
    Machineid Int Generated By Default As Identity,
    M_Name Varchar(50) Not Null,
    Capacity Int Not Null,
    Primary Key(Machineid)
);
```

8. Create Table Purchased_Rawmaterial:

```
    --Create procedures tables

   --Create Purchased_Raw table (PurchasedOrder-rawmaterial M-M relation table)
Create Table Purchased_Rawmaterial(
 Rawmaterialid Int Not Null,
 Purchaseorderid Int Not Null,
 Quantity INT Not Null,
 PRIMARY KEY (RawMaterialID,PurchaseOrderID),
 FOREIGN KEY (RawMaterialID) REFERENCES RawMaterial(RawMaterialID),
 FOREIGN KEY (PurchaseOrderID) REFERENCES PurchaseOrder(PurchaseOrderID)
);
```

9. Create Table Sold_Products:

```
--create sold_produt table (salesorder-products M-M relation table)
   Create Table Sold_Products(
 Productid Int Not Null,
 Salesorderid Int Not Null,
 Quantity Int Not Null,
 PRIMARY KEY (ProductID,SalesOrderID),
 FOREIGN KEY (ProductID) REFERENCES Product(ProductID),
 FOREIGN KEY (SalesOrderID) REFERENCES SalesOrder(SalesOrderID)
);
```

10. Create Table Product_Rawmaterial:

```
--Raw-product M-M relation table
Create Table Product_Rawmaterial(
  Rawmaterialid Int Not Null,
  Productid Int Not Null,
  Quantity Int Not Null,
  PRIMARY KEY (RawMaterialID,ProductID),
    FOREIGN KEY (RawMaterialID) REFERENCES RawMaterial(RawMaterialID),
    FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
);
```

11. Create Table Productmachine:

```
--create productmachine table (M-M relation)
Create Table Productmachine (
    Productid Int Not Null,
    Machineid Int Not Null,
    PRIMARY KEY (ProductID, MachineID),
    FOREIGN KEY (ProductID) REFERENCES Product(ProductID),
    FOREIGN KEY (MachineID) REFERENCES Machine(MachineID)
);
```

12. Create Table Qualitycontrol:

```
-- Create the QualityControl table
Create Table Qualitycontrol (
    Qualitycontrolid Int Generated By Default As Identity ,
    Qualitycontroldate Date Not Null,
    Status Varchar(10) Check (Status In ('VALID', 'INVALID')) Not Null,
    Productid Int Not Null,
    Warehouseid Int Not Null,
    Quantity INT Not Null,
    PRIMARY KEY (ProductID, QualityControlID),
    FOREIGN KEY (ProductID) REFERENCES Product (ProductID),
    FOREIGN KEY (WarehouseID) REFERENCES Warehouse (WarehouseID)
    );
```

تعليمات ادخال بيانات الجداول:

1. Insert Data Into Products Table:

```
-- Insert data into Products table
Insert Into Product (P_Name, Specifications, Productiondate, Expirydate, Wholesaleprice, Retailprice, Deffective_Price)
Values ('Product1', 'Spec1', '1/1/2022', '1/1/2023', 50.00, 75.00, 30.00);
Insert Into Product (P_Name, Specifications, Productiondate, Expirydate, Wholesaleprice, Retailprice, Deffective_Price)
VALUES ('Product2', 'Spec2', '1/1/2022', '1/1/2023', 40.00, 60.0, 20.00);
```

2. Insert Data Into Departmemts Table:

```
-- Insert data into Departments table
INSERT INTO Departments (D_Name) VALUES ('Sales');
INSERT INTO Departments (D_Name) VALUES ('Purchasement');
```

3. Insert Data Into PurchaseOrder Table:

```
-- Insert data into PurchaseOrder table
INSERT INTO Purchaseorder (Departmentid, Purchasedate) VALUES (2, '1/1/2023');
```

4. Insert Data Into SalesOrder Table:

```
-- Insert data into SalesOrder table
INSERT INTO Salesorder (Departmentid, Salesdate) VALUES (1, '1/1/2023');
```

5. Insert Data Into Warehouse Table:

```
-- Insert data into Warehouse table
Insert Into Warehouse (W_Location, Capacity) Values ('Location1', 1000);
Insert Into Warehouse (W_Location, Capacity) Values ('Location2', 1000);
INSERT INTO Warehouse (W_Location, Capacity) VALUES ('Location3', 1000);
```

6. Insert Data Into Rawmaterial Table:

```
-- Insert data into Rawmaterial table
Insert Into Rawmaterial (M_Name, Supplier, Warehouseid, Minlimit, Maxlimit) Values ('Material1', 'Supplier1', 1, 100, 500);
Insert Into Rawmaterial (M_Name, Supplier, Warehouseid, Minlimit, Maxlimit) Values ('Material2', 'Supplier2', 1, 100, 500);
INSERT INTO Rawmaterial (M_Name, Supplier, Warehouseid, Minlimit, Maxlimit) VALUES ('Material3', 'Supplier1', 1, 100, 500);
```

7. Insert Data Into Machine Table:

```
-- Insert data into Machine table
INSERT INTO Machine (M_Name, Capacity) VALUES ('Machine1', 100);
```

8. Insert Data Into Purchased_Rawmaterial Table:

```
-- Insert data into Purchased_Rawmaterial table
Insert Into Purchased_Rawmaterial (Rawmaterialid, Purchaseorderid, Quantity) Values (1, 1, 200);
Insert Into Purchased_Rawmaterial (Rawmaterialid, Purchaseorderid, Quantity) Values (2, 1, 200);
INSERT INTO Purchased_Rawmaterial (Rawmaterialid, Purchaseorderid, Quantity) VALUES (3, 1, 200);
```

9. Insert Data Into Sold_Products Table:

```
-- Insert data into Sold_Products table
INSERT INTO Sold_Products (Productid, Salesorderid, Quantity) VALUES (1, 1, 50);
```

10. Insert Data Into Product_Rawmaterial Table:

```
-- Insert data into Product_Rawmaterial table
Insert Into Product_Rawmaterial (Rawmaterialid, Productid, Quantity) Values (1, 1, 100);
Insert Into Product_Rawmaterial (Rawmaterialid, Productid, Quantity) Values (2, 1, 100);
INSERT INTO Product_Rawmaterial (Rawmaterialid, Productid, Quantity) VALUES (3, 2, 100);
```

11. Insert Data Into Productmachine Table:

```
-- Insert data into Productmachine table
INSERT INTO Productmachine (Productid, Machineid) VALUES (1, 1);
```

إنشاء الإجرائية وتنفيذها:

```
--Create a New object Represent Mulible Raw material Fror each Product
CREATE OR REPLACE TYPE RAW_MATERIAL AS OBJECT (
    RawMaterialID NUMBER,
    Quantity NUMBER
```

```
);
/
-- Create a new collection type based on the RAW_MATERIAL object
CREATE OR REPLACE TYPE RawMaterialArray AS TABLE OF RAW_MATERIAL;
/

CREATE OR REPLACE PROCEDURE ManufactureProduct (
    p_productID NUMBER,
    p_quantities RawMaterialArray,
    p_warehouseID NUMBER
) AS
    v_totalCapacity NUMBER;
    v_quantitiesCount NUMBER := p_quantities.Count; -- Get the count directly from
the collection
BEGIN
    -- Check if the required raw materials are available and the total capacity of the
warehouse is sufficient
    FOR i IN 1..v_quantitiesCount LOOP
        SELECT SUM(CASE WHEN pr.Quantity < p_quantities(i).Quantity THEN 1 ELSE 0
END)
        INTO v_totalCapacity
        FROM Product_RawMaterial pr
        JOIN RawMaterial r ON pr.RawMaterialID = r.RawMaterialID
        WHERE pr.ProductID = p_productID
        AND pr.RawMaterialID = p_quantities(i).RawMaterialID;

        IF v_totalCapacity > 0 THEN
            DBMS_OUTPUT.PUT_LINE('Insufficient quantity of Raw Material');
            RETURN;
        END IF;
    END LOOP;
-- At this point, we know that sufficient quantities are available, so we can proceed
with the manufacturing process
    FOR i IN 1..v_quantitiesCount LOOP
        UPDATE Product_RawMaterial
        SET Quantity = Quantity - p_quantities(i).Quantity
        WHERE ProductID = p_productID
        AND RawMaterialID = p_quantities(i).RawMaterialID;
    END LOOP;

    -- Insert the manufactured product into QualityControl
    INSERT INTO QualityControl (QualityControlDate, Status, ProductID, WarehouseID,
Quantity)
    VALUES (SYSDATE, 'VALID', p_productID, p_warehouseID, 50);
```

```
    -- Proper concatenation of the message
    DBMS_OUTPUT.PUT_LINE('Product ' || p_productID || ' manufactured
successfully.');

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('Insufficient warehouse capacity.');
END;
/
```

شرح الكود:

```
SET SERVEROUTPUT ON;
DECLARE
Quantities Rawmaterialarray := Rawmaterialarray(
Raw_Material(1, 3),
RAW_MATERIAL(2, 50)
);
Begin
ManufactureProduct(1, quantities, 2);
END;
```

يقوم الكود بتهيئة متغير الكميات مع مجموعة من كائنات المواد الخام ومن ثم يستدعي وظيفة ManufactureProduct، وتمرير معرف المنتج، ومتغير الكميات ورقم المخزن الخاص بالمواد الأولية. ستحتوي الدالة ManufactureProduct على المنطق الخاص بمعالجة المواد الخام، وإجراء التصنيع اللازم الخطوات وإخراج المنتج النهائي.

يستخدم الكود **SET SERVEROUTPUT ON;** للسماح لتعليمات الخرج بإظهار الرسائل عند نجاح العملية أو فشلها.

```
CREATE OR REPLACE TYPE RAW_MATERIAL AS OBJECT (
RawMaterialID NUMBER,
Quantity NUMBER
);
```

هذا الكود يعرف نوع كيان جديد يسمى Raw_Material وله سمتين:

- RawMaterialID
- Quantity

و هذا الكيان سيتم استخدامه ليمثل مادة أولية مع كميتها

```
CREATE OR REPLACE TYPE RawMaterialArray AS TABLE OF RAW_MATERIAL;
```

يعرف هذا الكود مصفوفة مبنية على كيان Raw_Material و ستستخدم لتخزين أكثر من مادة أولية.

```
FOR i IN 1..v_quantitiesCount LOOP
SELECT SUM(CASE WHEN pr.Quantity < p_quantities(i).Quantity THEN 1
ELSE 0 END)
INTO v_totalCapacity
FROM Product_RawMaterial pr
JOIN RawMaterial r ON pr.RawMaterialID = r.RawMaterialID
WHERE pr.ProductID = p_productID
AND pr.RawMaterialID = p_quantities(i).RawMaterialID;

IF v_totalCapacity > 0 THEN
DBMS_OUTPUT.PUT_LINE('Insufficient quantity of Raw Material');
RETURN;
END IF;
END LOOP;
```

يمثل هذا الكود :

1- **FOR i IN 1..v_quantitiesCount LOOP** و هذا السطر ينشئ حلقة تكرارية تبدأ من 1 حتى عدد عناصر v_quantitiesCount و التي تمثل عدد المواد الأولية.

2- داخل الحلقة الأكواد :

```
SELECT SUM(CASE WHEN pr.Quantity < p_quantities(i).Quantity THEN
1 ELSE 0 END)
INTO v_totalCapacity
```

و هذه التعليمة تحسب مجموع نتائج دالة شرطيه و هذا الدالة الشرطية و التي تعيد 1 إذا كانت الكمية المتوفرة للمنتج من المواد الأولية أقل من الكمية المطلوبة و إلا فإنها تعيد 0 و تضيفة للمتغير v_totalCapacity

- الكود :

```
FROM Product_RawMaterial pr
JOIN RawMaterial r ON pr.RawMaterialID = r.RawMaterialID
WHERE pr.ProductID = p_productID
AND pr.RawMaterialID = p_quantities(i).RawMaterialID;
```

هذا الاستعلام هيأ الجداول لاستخدامها و يهيأ الشروط للدمج و الفلترة و هو يربط جدول Product_Rawmaterial مع جدول Rawmaterial من خلال معرفات المواد ثم يفلتر البيانات بناء على معرف المنتج و معرف المادة الخام من P_quantiteis مصفوفة المواد الخام المدخلة.

- الكود :

```
IF v_totalCapacity > 0 THEN
DBMS_OUTPUT.PUT_LINE('Insufficient quantity of Raw Material');
RETURN;
END IF;
```

يقوم هذا الكود باختبار v_totalCapacity في حال كانت أكبر من 1 هذا يعني عدم كفاية أحد المواد المطلوبة للمنتج.

```
FOR i IN 1..v_quantitiesCount LOOP
UPDATE Product_RawMaterial
SET Quantity = Quantity - p_quantities(i).Quantity
WHERE ProductID = p_productID
AND RawMaterialID = p_quantities(i).RawMaterialID;
END LOOP;
```

يقوم هذا الكود بتحديث جدول Product_RawMaterial بالكميات الجديدة بعد عملية التصنيع.

```
INSERT INTO QualityControl (QualityControlDate, Status, ProductID,
WarehouseID, Quantity)
VALUES (SYSDATE, 'VALID', p_productID, p_warehouseID, 50);
```

يقوم هذا لاكود بإدخال المنتج بعد تصنيعة إلى المخزن بعد مروره على جدولQualityControl

```
--Create a New object Represent Mulible Raw material Fror each Product
CREATE OR REPLACE TYPE RAW_MATERIAL AS OBJECT (
    RawMaterialID NUMBER,
    Quantity NUMBER
);
/
-- Create a new collection type based on the RAW_MATERIAL object
CREATE OR REPLACE TYPE RawMaterialArray AS TABLE OF RAW_MATERIAL;
/

CREATE OR REPLACE PROCEDURE ManufactureProduct (
    p_productID NUMBER,
    p_quantities RawMaterialArray,
    p_warehouseID NUMBER
) AS
    v_totalCapacity NUMBER;
    v_quantitiesCount NUMBER := p_quantities.Count; -- Get the count directly from the collection
BEGIN
    -- Check if the required raw materials are available and the total capacity of the warehouse is sufficient
    FOR i IN 1..v_quantitiesCount LOOP
        SELECT SUM(CASE WHEN pr.Quantity < p_quantities(i).Quantity THEN 1 ELSE 0 END)
        INTO v_totalCapacity
        FROM Product_RawMaterial pr
        JOIN RawMaterial r ON pr.RawMaterialID = r.RawMaterialID
        WHERE pr.ProductID = p_productID
        AND pr.RawMaterialID = p_quantities(i).RawMaterialID;

        IF v_totalCapacity > 0 THEN
            DBMS_OUTPUT.PUT_LINE('Insufficient quantity of Raw Material');
            RETURN;
        END IF;
    END LOOP;
```

```
    -- At this point, we know that sufficient quantities are available, so we can proceed with the manufacturing process
    FOR i IN 1..v_quantitiesCount LOOP
        UPDATE Product_RawMaterial
        SET Quantity = Quantity - p_quantities(i).Quantity
        WHERE ProductID = p_productID
        AND RawMaterialID = p_quantities(i).RawMaterialID;
    END LOOP;

    -- Insert the manufactured product into QualityControl
    INSERT INTO QualityControl (QualityControlDate, Status, ProductID, WarehouseID, Quantity)
    VALUES (SYSDATE, 'VALID', p_productID, p_warehouseID, 50);

    -- Proper concatenation of the message
    DBMS_OUTPUT.PUT_LINE('Product ' || p_productID || ' manufactured successfully.');

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('Insufficient warehouse capacity.');
END;
/
```

القادح:

شرح الكود:

يتم تنشيط هذا القادح بعد إجراء عمليات إدراج أو تحديث على الجدول "Purchased_Rawmaterial".
الهدف من هذا القادح هو مراقبة كميات المواد الخام المشتراة وتنفيذ إجراءات معينة إذا ما تجاوزت الكمية حدودًا محددة.

يتكون الكود من عناصر أساسية عدة وهو يهدف للتحقق من توافر الكميات الكافية من المواد الخام قبل تصنيع المنتج.
تم إنشاء الإجراء " "ManufactureProduct وهو يستلم معرف المنتج (ProductID,Quantiteis) قائمة من الكميات المطلوبة من المواد الخام ومعرف المستودع WarehouseID كمعاملات.
يتم استخدام هذا الإجراء لتصنيع المنتج من المواد الخام.
يتم التحقق من توفر الكميات المطلوبة من المواد الخام وقدرة المستودع بشكل عام
إذا كانت الكميات المطلوبة متوفرة وقدرة المستودع كافية، يتم تحديث كميات المواد الخام وإدراج المنتج المصنوع في جدول QualityControl. كما يتم عرض رسالة توضح نجاح عملية التصنيع باستخدام DBMSOUTPUT.
في حالة حدوث أي استثناءات مثل عدم العثور على البيانات المطلوبة، يتم عرض رسالة توضيحية

```
CREATE OR REPLACE TRIGGER Rawmaterialcheck
AFTER INSERT OR UPDATE ON Purchased_Rawmaterial
FOR EACH ROW
DECLARE
    v_minLimit Rawmaterial.MinLimit%TYPE;
    v_maxLimit Rawmaterial.MaxLimit%TYPE;
    v_currentQuantity NUMBER;
BEGIN
    SELECT COALESCE(SUM(Quantity), 0)
```

```
    INTO v_currentQuantity
    FROM Product_Rawmaterial
    WHERE Rawmaterialid = :NEW.RawMaterialID;

    SELECT COALESCE(SUM(Quantity), 0) + v_currentQuantity
    INTO v_currentQuantity
    FROM Purchased_Rawmaterial
    WHERE Rawmaterialid = :NEW.RawMaterialID;

    SELECT Minlimit, Maxlimit
    INTO v_minLimit, v_maxLimit
    FROM Rawmaterial
    WHERE Rawmaterialid = :NEW.RawMaterialID;

    IF v_currentQuantity > v_maxLimit THEN
        INSERT INTO NotificationTable (Message, CreatedDate)
        VALUES ('Raw material quantity for raw material ' || :NEW.RawMaterialID || ' exceeds
the maximum limit', SYSDATE);
    ELSIF v_currentQuantity < v_minLimit THEN
        INSERT INTO NotificationTable (Message, CreatedDate)
        VALUES ('Raw material quantity for raw material ' || :NEW.RawMaterialID || ' is below
the minimum limit', SYSDATE);
    END IF;
END;
```