Integrated Supply Chain and Financial Management System

OVERVIEW

This project aims to develop an all-encompassing Database Schema for Enterprise Resource Planning (ERP) tailored for a manufacturing and supply chain enterprise. The primary focus is on refining the journey from design to production, regulating internal departments, and optimizing the management of materials from diverse suppliers. A pivotal aspect involves skillfully handling invoices to ensure sender details are crystal clear, facilitating precise processing by the accounting department. The objective is to deliver a resilient database schema adept at efficiently meeting the operational needs of a modern manufacturing entity, emphasizing peak performance, meticulous employee management, and precise financial oversight. Key components include streamlined employee administration, product life cycle oversight, efficient raw material flow, and seamless tracking and processing of supplier invoices.

INITIAL TIMELINE DECIDED

Please find a visual of the timeline below. Further details are listed afterward for each step of the timeline.

	10/4/2023	10/18/2023	10/25/2023	11/8/2023	11/23/2023	12/3/2023
Week 1						
Weeks 2-3						
Week 4						
Weeks 5-6						
Weeks 7-8						
Week 9						
Week 10						

Week 1: INITIATION, PLANNING AND PROPOSAL

- Create a preliminary draft for the project proposal.
- Finalize the project proposal.
- Develop a detailed project plan, including a timeline and deliverables.
- Submit the finalized project proposal.

Week 2 - 3: ANALYSIS, METADATA COMPILATION

- Perform a comprehensive analysis of project objectives and requirements using existing data.
- Compile metadata, including entity details, attributes, data types, and relationships.
- Refine the metadata as necessary, adding attributes and metadata as required, and making any needed improvements.

Week 4: ESTABLISHING BUSINESS RULES AND CREATE EER DIAGRAM

- Align the project with established business rules to ensure seamless integration with the database design and the EER model.

- Create the project structure with the above-defined metadata and business rules and the listed tasks requirements.
- Establish a systematic organization for easy interpretation of the EER diagram.

Week 5 - 6: RELATION AND NORMALIZATION

- Document the steps taken and define the relationships, and integrity constraints involved.
- Create relations, normalize them, and describe the anomalies found.

Week 7 - 8: SQL QUERIES REVISION, TESTING, AND QUALITY CONTROL

- Create SQL queries to make tables with the provided data
- Conduct a thorough review of the created project to ensure it adheres to established guidelines.
- Revise and refine the project based on feedback and suggestions from reviewers.
- Engage in formal and deliberate testing for Quality Control.

Week 9: PROJECT SUBMISSION

Week 10: PROJECT PRESENTATION

THE ACTUAL TIMELINE

Week 1: INITIATION, PLANNING AND PROPOSAL

- Created a preliminary draft for the project proposal.
- Finalized and submitted the project proposal.
- Developed a detailed project plan, including a timeline and deliverables.

Week 2 - 3: TASK 0 - ANALYSIS, METADATA AND BUSINESS RULES COMPILATION

- Performed a comprehensive analysis of project objectives and requirements using existing data.
- Compiled metadata, including entity details, attributes, data types, and relationships.
- Refined the metadata as necessary, adding attributes and metadata as required, and making any needed improvements.
- Aligned the project with established business rules to ensure seamless integration with the database design.

Week 4: TASK 1 and TASK 2 – PRIMARY KEY, FOREIGN KEY, EER DIAGRAM

- Identified the primary key, and foreign key and began putting the EER diagram together

Week 5: COMPLETED TASK 2, EER DIAGRAM

- Finalized the EER diagram.
- Made sure that all the relationships were according to the business rules, and that the final diagram was devoid of any redundant entities, relationships, or attributes.

Week 6: TASK 3 – RELATIONAL SCHEMA, INTEGRITY CONSTRAINTS, RELATIONSHIPS

- Transformed the EER diagram into a relational schema. Identified the binary and ternary relationships and established the relevant integrity constraints.

Week 7 - 8: TASK 4 – FUNCTIONAL DEPENDENCIES, ANOMALIES, NORMALIZATION

- We listed the functional dependencies for each relation, identified the anomalies, and normalized the relations wherever required.
- Explanations for every step taken were created.
- Started working on the project paper.

Week 9: TASK 5 – SQL QUERIES FOR THE NORMALIZED TABLES, COMPLETION OF PAPER

- Carried out the SQL queries for each relation, created tables with the normalized data, and designed three meaningful SQL queries on the data.
- Conducted a thorough review of the created project to ensure it adheres to established guidelines.
- Revised and refined the project based on feedback and suggestions from reviewers.
- Engaged in formal and deliberate testing for Quality Control.

Week 10: PROJECT PRESENTATION

TASK 0: REPOSITORY OF META DATA AND BUSINESS RULES

CONTENTS OF REPOSITORY STRUCTURE:
The repository's structure outlined below will encompass essential components, including metadata, associated metadata types, and Business Rules.

Table 1: METADATA

Metadata for Integrated Supply Chain and Financial Management System			
Data Item		**Metadata**	
Name Source	**Type**	**Description**	
Position	VARCHAR	Worker's Position	Employee
Salary	NUM	Worker's Salary	
ID	NUM	Worker's ID	
first name	VARCHAR	Worker's first name	
last name	VARCHAR	Worker's last name	
Name	VARCHAR	Department Name	Department
Location	NUM	Department Location	
Phone Number	NUM	Department Phone no	
Number of Employees	NUM	Total Employees in department	
Product Number	NUM	Product Number	Product
Name	VARCHAR	Product Name	
Price	NUM	Product Price	
Cost	NUM	Product Cost	
Dimensions	NUM	Product Dimension	
Colour	VARCHAR	Product Colour	
Weight	NUM	Product Weight	
Raw Material	VARCHAR	Product raw material	
Line Number	NUM	Production line number	Production Line
Line Capacity	NUM	Production Line capacity	
Company Name	VARCHAR	Company Name	Vendor
Address	VARCHAR	Company Address	
Speciality	VARCHAR	Company Speciality	
Phone number	NUM	Company Phone number	
Invoice Number	NUM	Invoice Number	Invoice
Total Amount	NUM	Invoice Amount	
Address	NUM	Warehouse Address	Warehouse
Phone Number	VARCHAR	Warehouse Phone Number	

Table 2: Business Rules

BUSINESS RULES	
B1	**CLUSTER 1 – ENTITIES: PRODUCT, PRODUCTION LINE**
B1-1	Each product must be produced by one specific production line.
B1-2	Each production line can produce only one type of product.
B1-3	For repair purposes, production lines may produce no products.
B2	**CLUSTER 2 – ENTITIES: VENDOR, WAREHOUSE, RAW MATERIAL**
B2-1	Each vendor can supply many raw materials to any number of warehouses.
B2-2	Raw materials are supplied by any number of warehouses, which are supplied by any number of vendors.
B2-3	Raw materials may also be directly supplied by vendors.
B2-4	Each warehouse can be supplied with any number of raw materials from more than one vendor, but each warehouse must be supplied with at least one raw material.
B3	**CLUSTER 3 – ENTITIES: EMPLOYEE, WAREHOUSE, DEPARTMENT**
B3-1	The company has departments, warehouses, and production lines.
B3-2	The company designs and produces products.
B3-3	Each department, warehouse, and production line have multiple employees.
B3-4	Each employee works in only one department, warehouse, or production line.
B3-5	Only Employees within the design department design products.
B3-6	Each designer can design multiple products.
B3-7	Each product has exactly one designer
B4	**CLUSTER 4 – ENTITIES: INVOICE, DEPARTMENT**
B4-1	Vendors submit an invoice when they supply a raw material.
B4-2	Invoices are processed by the accounting department

TASK 1 and TASK 2 – REPRESENTATION OF PRIMARY KEY, FOREIGN KEY, EER DIAGRAM

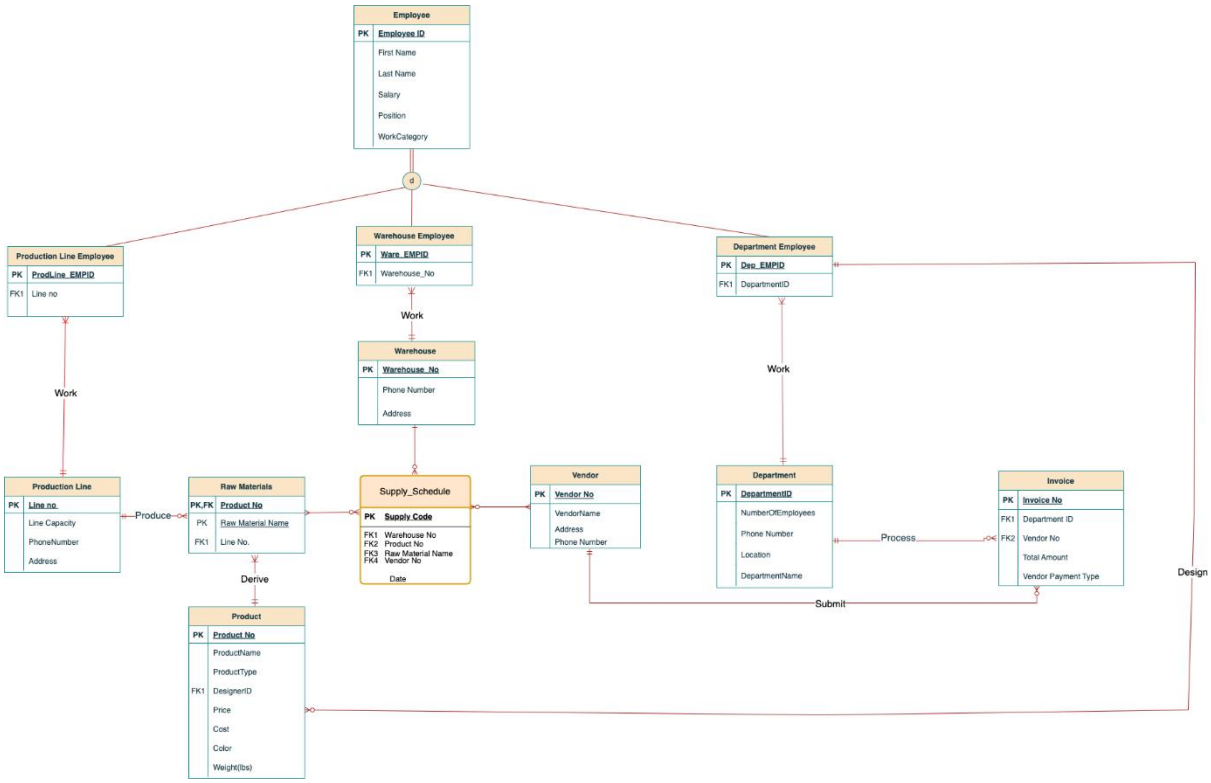


FIGURE: EER DIAGRAM

In the above diagrams, the primary keys of each entity are denoted by PK and are underlined by a solid line. Whereas, if there is a foreign key it is denoted by FK. Some of the attributes have more than one foreign keys. And the entity 'Raw Material' has a composite primary key.

TASK 3 CONVERSION TO RELATIONAL SCHEMA

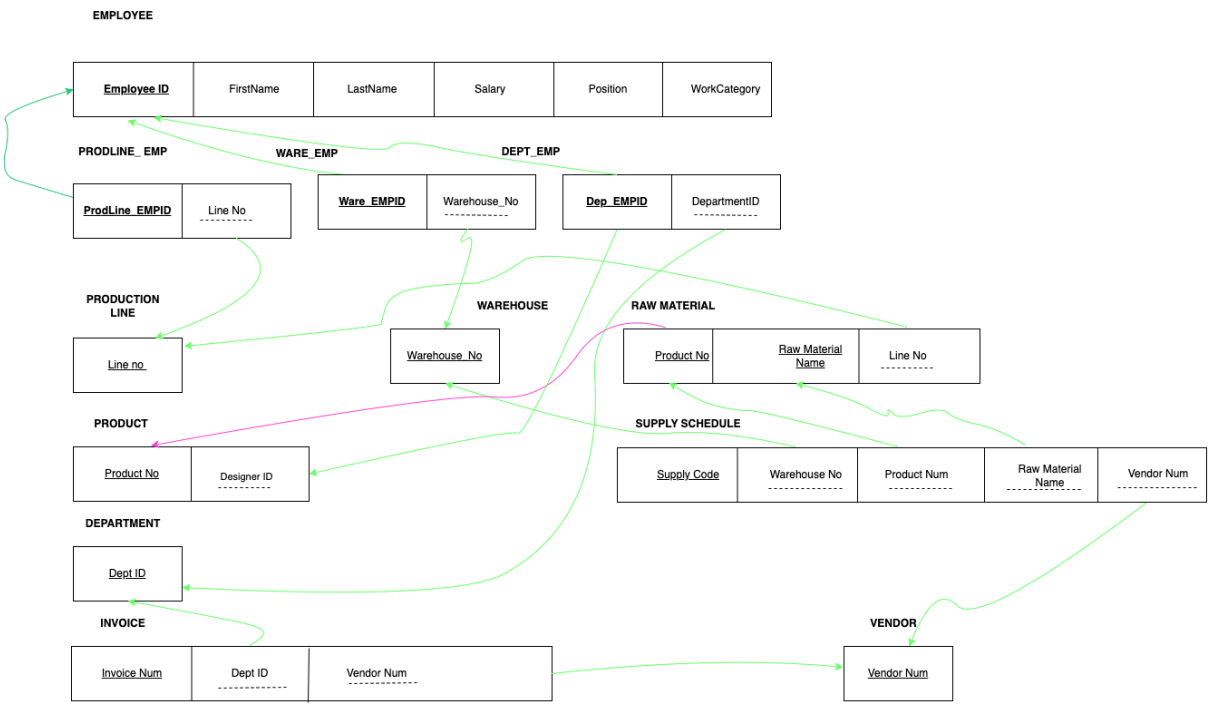


FIGURE: RELATION SCHEMA

LIST OF ALL THE IDENTIFIED RELATIONSHIPS

1) **UNARY RELATIONSHIP**
   There is no Unary relationship.

2) **TERNARY RELATIONSHIP**

This used to be the only ternary relationship until I brought the associative entity 'supply schedule'.

(Vendor - Warehouse - Raw Material):

- **Business Rule:** "Vendor warehouse and raw material participate in a ternary relationship with supply schedule as an associative entity."
- **Description:** Involves vendors, warehouses, and raw materials in a ternary relationship. This captures the logistics of raw material supply from vendors to warehouses based on the business rules governing these entities.

After bringing the associative entity 'supply schedule' we have now three binary relationships between supply schedule and vendor, supply schedule and warehouse, and supply schedule and raw materials.

3) **BINARY RELATIONSHIPS**

1. (Production Line - Raw Material)
2. (Product - Raw Material)
3. (Production Line Employee - Production Line)
4. (Warehouse Employee - Warehouse)
5. (Department Employee - Department)
6. (Vendor - Invoice)
7. (Product - Department)
8. (Department - Invoice)
9. (Supply Schedule - Vendor)
10. (Supply Schedule – Raw Materials)
11. (Supply Schedule – Warehouse)

INTEGRITY CONSTRAINTS

There are three integrity constraints which are also depicted in the above relational schema.

1)DOMAIN INTEGRITY CONSTRAINTS – A domain definition usually consists of the following components: domain name, meaning, data type, size (or length), and allowable values or allowable range (if applicable). Below table table shows the domain definastion.

Domain Constraints – Domain definition is listed below:

Attribute	Domain Name	Description	Domain
ProductNumber	Product Number	Set of all product numbers	NUM
ProductType	Product Type	Set of all product type	VARCHAR
ProductName	Product Name	Set of all possible product names	VARCHAR
DesignerID	Designer IDs	Set of all possible Designer ID	VARCHAR
Price	Product Price	Set of all possible product price	NUM
Cost	Product Cost	Set of all possible product cost	NUM
Color	Colors	Set of all possible colors	VARCHAR
Weight (lbs)	Weights	Set of weights	NUM
LineNumber	Line Number	Set of all possible line numbers	VARCHAR
LineCapacity(items/hour)	Line Capacity	Set of all line capacity	VARCHAR
PhoneNumber	Phone Number	Set of phone numbers	NUM
Address	Full Address	Set of all address	VARCHAR
RawmaterialName	Rawmaterial Name	Set of all possible raw material names	VARCHAR
WarehouseNumber	Warehouse Number	Set of all possible warehouse number	VARCHAR
VendorNumber	Vendor Number	Set of all possible vendor number	VARCHAR
VendorName	Vendor Name	Set of all possible vendor name	VARCHAR
SupplyCode	Supply Code	Set of all possible supply code	VARCHAR
VendorID	Vendor IDs	Set of all possible vendor IDs	VARCHAR
DepartmentID	Department IDs	Set of all possible department IDs	VARCHAR
DepartmentName	Department Name	Set of all possible department names	VARCHAR
Location	Full Location	Set of all possible locations	VARCHAR
NumberOfEmployees	Number of Employees	Set of all possible number of employees	NUM
EmployeeID	Employee ID	Set of all possible employee IDs	VARCHAR
FirstName	First Name	Set of all first name	VARCHAR
LastName	Last Name	Set of all last name	VARCHAR
Position	Designation	Set of all possible positions	VARCHAR
Salary	Salary Details	Set of all possible salaries	NUM

InvoiceNumber	Invoice Number	Set of all possible invoice numbers	VARCHAR
Total Amount ($)	TotalAmount	Set of total amounts	NUM
VendorPaymentType	Vendor Payment Type	Set of all possible vendor payment type	VARCHAR

FIGURE: TABLE FOR DOMAIN DEFINITION FOR THE DOMAIN ASSOCIATED WITH THE ATTRIBUTES

2) ENTITY INTEGRITY CONSTRAINT

- **Explanation:**
  - Entity constraints ensure the uniqueness of primary keys within a table, allowing each entity to be uniquely identified.
- **Example:**
  - In the "Employee" entity, we enforce an entity constraint that ensures each employee has a unique "EmployeeID."

3)REFERENTIAL INTEGRITY CONSTRAINT:

Referential Integrity is a principle that stipulates that any foreign key value, located on the many sides of a relationship, must correspond to a primary key value in the table on the one side. Alternatively, the foreign key can be null. This rule is crucial for maintaining the consistency of relationships in a database. For instance, when it comes to deletion rules:

- **Restrict:** This rule prohibits the deletion of the "parent" side if related rows exist in the "dependent" side.
- **Cascade:** It involves the automatic deletion of rows on the "dependent" side that correspond to the "parent" side row being deleted.
- **Set-to-Null:** This rule entails setting the foreign key on the dependent side to null if a deletion occurs on the parent side. However, it's important to note that this is not allowed for weak entities.

Referential integrity constraints are visually represented using arrows that point from the dependent table to the parent table. This graphical representation helps to depict the direction of the relationship and ensures that foreign keys align with corresponding primary keys, thus maintaining the integrity of the database structure.

TASK 4 DEPENDENCIES, ANOMALIES, NORMALIZATION

FUNCTIONAL DEPENDENCIES WILL BE LISTED BELOW BEFORE THAT LET'S DESCRIBE NORMALIZATION:

NORMALIZATION PROCESS

Normalization is the process used to organize and structure a relational database in such a way that data redundancy is minimized, and data integrity is preserved. The goal of normalization is to remove the functional dependency, and well-organized tables to reduce data duplication and anomalies.

1. **First Normal Form (1NF):**
    - Ensures that each column in a table contains atomic (indivisible) values and that each entry in a column is of the same data type. It eliminates multivalued attributes and ensures a unique primary key for each record.
2. **Second Normal Form (2NF):**
    - Extends 1NF by ensuring that all non-key attributes are fully functionally dependent on the entire primary key. This means that no partial dependencies exist, and every column in a table depends on the entire primary key.
3. **Third Normal Form (3NF):**
    - Extends 2NF by eliminating transitive dependencies. In 3NF, no non-key attribute should depend on another non-key attribute. Each attribute should be dependent only on the primary key.

FUNCTIONAL DEPENDENCIES AND NORMALIZATION

Partial, full, and transitive dependencies are shown below in the relations and required normalization.

FIGURE: NORMALIZATION

NORMALIZATION STEPS TAKEN

The entities, ‘Employee’, ‘Product’, ‘Production line’, ‘Warehouse’, ‘Vendor’, ‘Department’, ‘Supply Schedule’ are in 3NF already, which means they only have complete or full dependencies.

However, ‘Raw Material’ has partial dependency as ‘Line Number’ comes from only ‘Product Number’. But in ‘Raw Materials’ we have composite primary keys so for it to be a complete dependency, line number should have come from both the primary keys.

So, to normalize, we break the relations into two: One only with the ‘Product Number’ and ‘Rawmaterial name’ and the other relation with ‘Product Number’ and ‘Line Number’.

‘Invoice’ also has transitive dependency because the ‘Vendor No’ which is not a primary key gives the attribute ‘Vendor Payment Type’ in the relation.

So, to normalize the relation, we break it into two: One containing the ‘Invoice No’ as the primary key and the other containing ‘Vendor No’ as the primary key.

Hence, after this, all the relations are in 3NF and are normalized.

WELL STRUCTURED – FINAL NORMALIZED RELATION SCHEMA

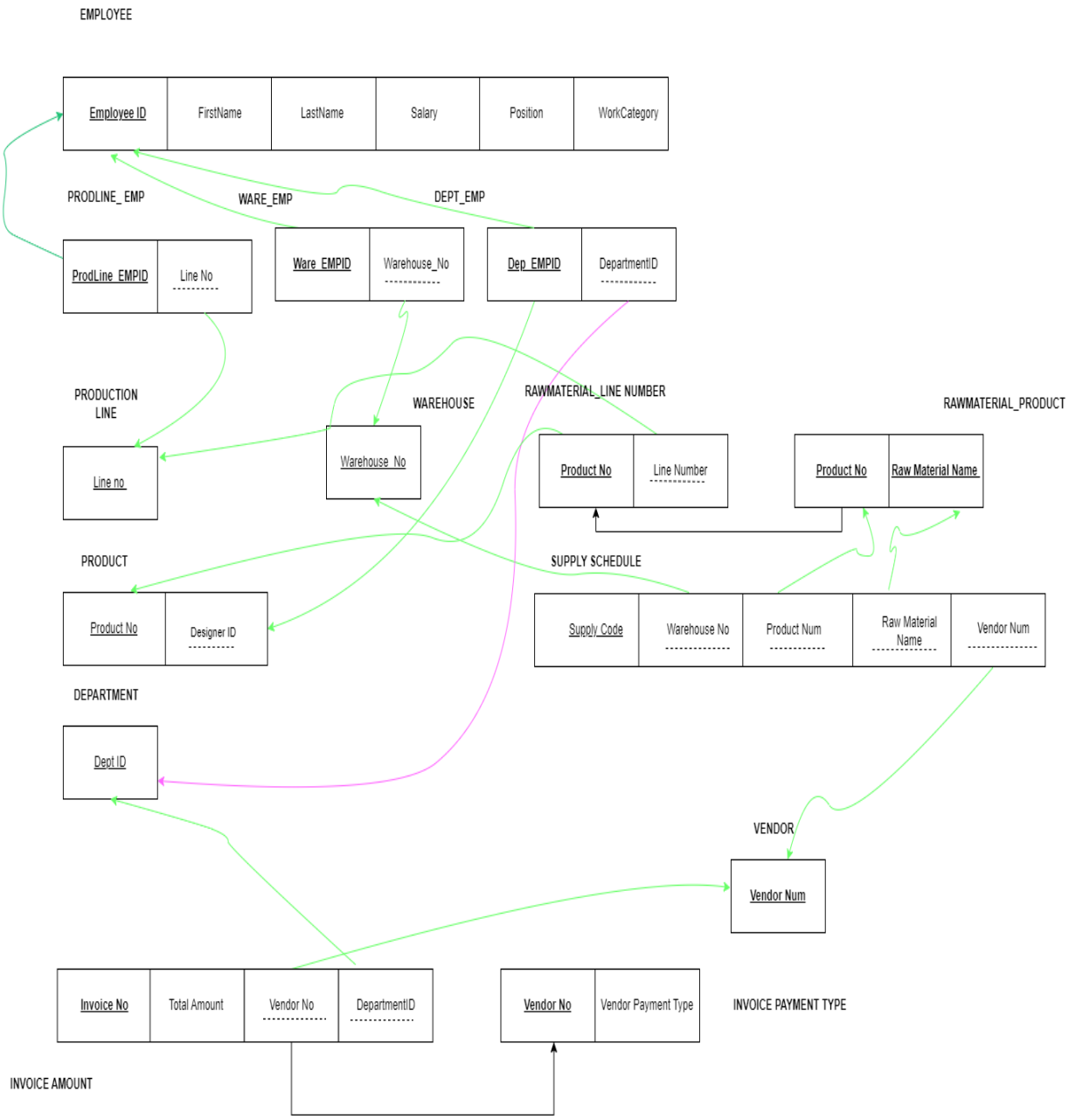


FIGURE: WELL-STRUCTURED NORMALIZED RELATION SCHEMA

THE ANOMALIES NOTICED BEFORE NORMALIZATION ARE LISTED BELOW:

Since, the entities, ‘Employee’, ‘Product’, ‘Production line’, ‘Warehouse’, ‘Vendor’, ‘Department’, and ‘Supply Schedule’ were in 3NF already they do not have deletion, modification, or insertion anomalies.

However, ‘raw-material’ and ‘invoice’ had a few anomalies which are listed below:

RAW MATERIAL

1. **Modification anomaly:**

   - If we need to update the ‘**LineNumber’** for a specific ‘**ProductNumber’** and ‘**RawmaterialName’**, we will have to update multiple records with the same ‘**LineNumber’**, leading to redundancy and increasing the likelihood of errors.

2. **Deletion anomaly:**

   - There is deletion anomaly concerning ‘Line number’ which is a foreign key referencing production line relation. So, to delete the record, we will have to set the value of Line no as null.
   - To delete the product number which is a primary key as well as a foreign key so we cannot set it to null. Hence, we will delete the product number as cascade which is deleting from both the tables.

3. **Insertion anomaly:**

   - There will be insertion anomaly, if we need to add a new product number, we will also have to enter the raw material name and the line number repeatedly.

INVOICE

1. **Update Anomaly:**

   - If the payment type for a vendor changes, we will have to update multiple records in the invoice table to reflect this change accurately. For instance, if the payment type for **V001** changes from 'Cash' to 'Credit,' we will have to update all invoices associated with **V001**. If we do not update all the relevant records could lead to inconsistencies where some invoices will still show the old payment type.

2. **Insertion Anomaly:**

- If we add a new invoice, we will have to add the vendor number and vendor payment type which is getting repeated and already existing in the table. Hence, it is causing insertion anomaly.

3. Deletion Anomaly:

- There is deletion anomaly because the vendor number is a foreign key that is referencing another relation. So, to delete a record, we will have to set its value to null.

TASK 5: TABLES OF ENTITIES AND THE SQL QUERIES

1)**DEPARTMENT**

DEPARTMENT TABLE

Oracle APEX
US_A925_SQL_S30.DEPARTMENT

Column Name	Data Type	Nullable	Default	Primary Key
DEPARTMENTID	VARCHAR2(100)	No		1
DEPARTMENTNAME	VARCHAR2(250)	Yes		
STREET	VARCHAR2(250)	Yes		
CITY	VARCHAR2(250)	Yes		
NUMBEROFEMPLOYEES	NUMBER(2,0)	Yes		
PHONENUMBER	VARCHAR2(250)	Yes		

SQL QUERY FOR DEPARTMENT

EMPLOYEES

Table | Data | Indexes | Model | **Constraints** | Grants | Statistics | UI Defaults | Triggers | Dependencies | **SQL**

```
CREATE TABLE  "EMPLOYEES"
   (    "FIRSTNAME" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "LASTNAME" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "POSITION" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "SALARY" NUMBER(10,2),
        "WORKCATEGORY" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "DEPARTMENTID" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "WAREHOUSENUMBER" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "LINENUMBER" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "EMPLOYEEID" VARCHAR2(100) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
         CONSTRAINT "EMPLOYEES_PK" PRIMARY KEY ("EMPLOYEEID")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
```

2) EMPLOYEE

EMPLOYEE TABLE

Oracle APEX
US_A925_SQL_S30.EMPLOYEES

Column Name	Data Type	Nullable	Default	Primary Key
EMPLOYEEID	VARCHAR2(100)	No		1
FIRSTNAME	VARCHAR2(250)	Yes		
LASTNAME	VARCHAR2(250)	Yes		
POSITION	VARCHAR2(100)	Yes		
SALARY	NUMBER(10,2)	Yes		
WORKCATEGORY	VARCHAR2(100)	Yes		
DEPARTMENTID	VARCHAR2(100)	Yes		
WAREHOUSENUMBER	VARCHAR2(100)	Yes		
LINENUMBER	VARCHAR2(100)	Yes		

SQL QUERY FOR EMPLOYEE

EMPLOYEES

Table | Data | Indexes | Model | Constraints | Grants | Statistics | UI Defaults | Triggers | Dependencies | SQL

```
CREATE TABLE  "EMPLOYEES"
   (    "FIRSTNAME" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "LASTNAME" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "POSITION" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "SALARY" NUMBER(10,2),
        "WORKCATEGORY" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "DEPARTMENTID" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "WAREHOUSENUMBER" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "LINENUMBER" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "EMPLOYEEID" VARCHAR2(100) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
         CONSTRAINT "EMPLOYEES_PK" PRIMARY KEY ("EMPLOYEEID")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
```

3)INVOICE AMOUNT

TABLE FOR NORMALIZED RELATION INVOICE AMOUNT

Oracle APEX
US_A925_SQL_S30.INVOICEAMOUNT

Column Name	Data Type	Nullable	Default	Primary Key
INVOICENUMBER	VARCHAR2(100)	No		1
TOTAL_AMOUNT_($)	NUMBER(6,2)	Yes		
VENDORNUMBER	VARCHAR2(100)	Yes		
DEPARTMENTID	VARCHAR2(100)	Yes		

SQL QUERY FOR INVOICE AMOUNT

INVOICEAMOUNT

Table | Data | Indexes | Model | Constraints | Grants | Statistics | UI Defaults | Triggers | Dependencies | SQL | REST

```
CREATE TABLE  "INVOICEAMOUNT"
   (    "INVOICENUMBER" VARCHAR2(100) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "TOTAL_AMOUNT_($)" NUMBER(6,2),
        "VENDORNUMBER" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "DEPARTMENTID" VARCHAR2(100) COLLATE "USING_NLS_COMP",
         CONSTRAINT "INVOICE_AMT_PK" PRIMARY KEY ("INVOICENUMBER")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
ALTER TABLE  "INVOICEAMOUNT" ADD CONSTRAINT "INVOICE_AMOUNT_FK" FOREIGN KEY ("DEPARTMENTID")
          REFERENCES  "DEPARTMENTS" ("DEPARTMENTID") ON DELETE SET NULL ENABLE
/
ALTER TABLE  "INVOICEAMOUNT" ADD CONSTRAINT "INVOICE_AMT_FK" FOREIGN KEY ("VENDORNUMBER")
          REFERENCES  "VENDOR" ("VENDORNUMBER") ENABLE
/
```

4)INVOICE PAYMENT

TABLE FOR THE NORMALIZED RELATION INVOICE PAYMENT

Oracle APEX
US_A925_SQL_S30.INVOICEPAYMENT

Column Name	Data Type	Nullable	Default	Primary Key
VENDORNUMBER	VARCHAR2(100)	No		1
VENDORPAYMENTTYPE	VARCHAR2(100)	Yes		

SQL QUERY FOR INVOICE PAYMENT

INVOICEPAYMENT

Table | Data | Indexes | Model | Constraints | Grants | Statistics | UI Defaults | Triggers | Dependencies | SQL | REST

```
CREATE TABLE  "INVOICEPAYMENT"
   (    "VENDORNUMBER" VARCHAR2(100) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "VENDORPAYMENTTYPE" VARCHAR2(100) COLLATE "USING_NLS_COMP",
         CONSTRAINT "INVOICE_PAYMENT_PK" PRIMARY KEY ("VENDORNUMBER")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
```

5)PRODUCT

TABLE FOR PRODUCT

Oracle APEX
US_A925_SQL_S30.PRODUCT

Column Name	Data Type	Nullable	Default	Primary Key
PRODUCTNUMBER	VARCHAR2(100)	No		1
PRODCUTTYPE	VARCHAR2(100)	Yes		
PRODUCTNAME	VARCHAR2(300)	Yes		
DESIGNERID	VARCHAR2(100)	Yes		
PRICE	NUMBER(5,2)	Yes		
COST	NUMBER(4,2)	Yes		
COLOR	VARCHAR2(100)	Yes		
WEIGHT	NUMBER(5,2)	Yes		

SQL QUERY FOR PRODUCT

PRODUCT

Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL

```
CREATE TABLE  "PRODUCT"
   (    "PRODUCTNUMBER" VARCHAR2(100) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "PRODCUTTYPE" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "PRODUCTNAME" VARCHAR2(300) COLLATE "USING_NLS_COMP",
        "EMPLOYEEID" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "PRICE" NUMBER(5,2),
        "COST" NUMBER(4,2),
        "COLOR" VARCHAR2(100) COLLATE "USING_NLS_COMP",
        "WEIGHT" NUMBER(5,2),
         CONSTRAINT "PRODUCT_PK" PRIMARY KEY ("PRODUCTNUMBER")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
ALTER TABLE  "PRODUCT" ADD CONSTRAINT "PRODUCT_FK" FOREIGN KEY ("EMPLOYEEID")
          REFERENCES  "EMPLOYEES" ("EMPLOYEEID") ON DELETE SET NULL ENABLE
/
```

6)PRODUCTION LINE

TABLE FOR PRODUCTION LINE

Oracle APEX
US_A925_SQL_S30.PRODUCTIONLINE

Column Name	Data Type	Nullable	Default	Primary Key
LINENUMBER	VARCHAR2(250)	No		1
LINECAPACITY_(ITEMS/HOUR)	NUMBER(4,0)	Yes		
PHONENUMBER	VARCHAR2(250)	Yes		
STREET	VARCHAR2(250)	Yes		
CITY	VARCHAR2(250)	Yes		
ZIPCODE	VARCHAR2(250)	Yes		

QUERY FOR PRODUCTION LINE

PRODUCTIONLINE

Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL REST

```
CREATE TABLE  "PRODUCTIONLINE"
   (    "LINENUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "LINECAPACITY_(ITEMS/HOUR)" NUMBER(4,0),
        "PHONENUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "STREET" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "CITY" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "ZIPCODE" VARCHAR2(250) COLLATE "USING_NLS_COMP",
         CONSTRAINT "PRODUCTIONLINE_PK" PRIMARY KEY ("LINENUMBER")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
```

7) RAWMATERIAL LINE NUMBER

TABLE FOR NORMALIZED RELATION RAWMATERIAL LINE NUMBER

Oracle APEX
US_A925_SQL_S30.RAWMATERIALINENUMBER

Column Name	Data Type	Nullable	Default	Primary Key
PRODUCTNUMBER	VARCHAR2(250)	No		1
LINENUMBER	VARCHAR2(250)	Yes		

QUERY FOR RAWMATER LINE NUMBER

RAWMATERIALINENUMBER

Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL REST

```
CREATE TABLE  "RAWMATERIALINENUMBER"
   (    "PRODUCTNUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "LINENUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP",
         CONSTRAINT "RAWMATERIALINENUMBER_PK" PRIMARY KEY ("PRODUCTNUMBER")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
ALTER TABLE  "RAWMATERIALINENUMBER" ADD CONSTRAINT "RAWMATERIALINENUMBER_FK" FOREIGN KEY ("PRODUCTNUMBER")
          REFERENCES  "PRODUCT" ("PRODUCTNUMBER") ON DELETE CASCADE ENABLE
/
ALTER TABLE  "RAWMATERIALINENUMBER" ADD CONSTRAINT "RAWMATERIALINENUMBER_FK2" FOREIGN KEY ("LINENUMBER")
          REFERENCES  "PRODUCLINE" ("LINENUMBER") ON DELETE SET NULL ENABLE
/
```

8) RAWMATERIAL PRODUCT

TABLE FOR NORMALIZED RELATION RAWMATERIAL PRODUCT

Oracle APEX
US_A925_SQL_S30.RAWMATERIALPRODUCT

Column Name	Data Type	Nullable	Default	Primary Key
PRODUCTNUMBER	VARCHAR2(250)	No		1
RAWMATERIALNAME	VARCHAR2(250)	No		2

QUERY FOR RAWMATERIAL PRODUCT

RAWMATERIALPRODUCT

Table | Data | Indexes | Model | **Constraints** | Grants | Statistics | UI Defaults | Triggers | Dependencies | **SQL** | REST

```
CREATE TABLE  "RAWMATERIALPRODUCT"
   (    "PRODUCTNUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "RAWMATERIALNAME" VARCHAR2(250) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
         CONSTRAINT "RAWMATERIALPRODUCT_PK" PRIMARY KEY ("PRODUCTNUMBER", "RAWMATERIALNAME")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
ALTER TABLE  "RAWMATERIALPRODUCT" ADD CONSTRAINT "RAWMATERIALPRODUCT_FK1" FOREIGN KEY ("PRODUCTNUMBER")
          REFERENCES  "PRODUCT" ("PRODUCTNUMBER") ON DELETE CASCADE ENABLE
/
```

9) SUPPLY SCHEDULE

TABLE FOR SUPPLU SCHEDULE

Oracle APEX
US_A925_SQL_S30.SUPPLYSCHEDULE

Column Name	Data Type	Nullable	Default	Primary Key
SUPPLYCODE	VARCHAR2(250)	No		1
PRODUCTNUMBER	VARCHAR2(250)	Yes		
RAWMATERIALNAME	VARCHAR2(250)	No		
WAREHOUSENUMBER	VARCHAR2(250)	Yes		
VENDORNUMBER	VARCHAR2(250)	Yes		
DATES	VARCHAR2(250)	Yes		

QUERY FOR SUPPLY SCHEDULE

SUPPLYSCHEDULE

Table | Data | Indexes | Model | **Constraints** | Grants | Statistics | UI Defaults | Triggers | Dependencies | **SQL** | REST | Samp

```
CREATE TABLE  "SUPPLYSCHEDULE"
   (    "SUPPLYCODE" VARCHAR2(250) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "PRODUCTNUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "RAWMATERIALNAME" VARCHAR2(250) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "WAREHOUSENUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "VENDORNUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "DATES" VARCHAR2(250) COLLATE "USING_NLS_COMP",
         CONSTRAINT "SUPPLYSCHEDULE_PK" PRIMARY KEY ("SUPPLYCODE")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
ALTER TABLE  "SUPPLYSCHEDULE" ADD CONSTRAINT "SUPPLYSCHEDULE_FK1" FOREIGN KEY ("PRODUCTNUMBER")
          REFERENCES  "PRODUCT" ("PRODUCTNUMBER") ON DELETE SET NULL ENABLE
/
ALTER TABLE  "SUPPLYSCHEDULE" ADD CONSTRAINT "SUPPLYSCHEDULE_FK2" FOREIGN KEY ("WAREHOUSENUMBER")
          REFERENCES  "WAREHOUSE" ("WAREHOUSENUMBER") ON DELETE SET NULL ENABLE
/
ALTER TABLE  "SUPPLYSCHEDULE" ADD CONSTRAINT "SUPPLYSCHEDULE_FK3" FOREIGN KEY ("VENDORNUMBER")
          REFERENCES  "VENDOR" ("VENDORNUMBER") ON DELETE SET NULL ENABLE
/
ALTER TABLE  "SUPPLYSCHEDULE" ADD CONSTRAINT "SUPPLY_SCHEDUL_FK4" FOREIGN KEY ("PRODUCTNUMBER", "RAWMATERIALNAME")
          REFERENCES  "RAWMATERIALPRODUCT" ("PRODUCTNUMBER", "RAWMATERIALNAME") ENABLE
/
```

10) VENDOR

TABLE FOR VENDOR

Oracle APEX
US_A925_SQL_S30.VENDORS

Column Name	Data Type	Nullable	Default	Primary Key
VENDORNUMBER	VARCHAR2(250)	No		1
VENDORNAME	VARCHAR2(250)	Yes		
STREET	VARCHAR2(250)	Yes		
CITY	VARCHAR2(250)	Yes		
PHONENUMBER	VARCHAR2(250)	Yes		

SQL QUERY FOR VENDOR

VENDORS

Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL

```
CREATE TABLE  "VENDORS"
   (    "VENDORNUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "VENDORNAME" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "STREET" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "CITY" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "PHONENUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP",
         CONSTRAINT "VENDORS_PK" PRIMARY KEY ("VENDORNUMBER")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
```

11) WAREHOUSE

TABLE FOR WAREHOUSE

Oracle APEX
US_A925_SQL_S30.WAREHOUSE

Column Name	Data Type	Nullable	Default	Primary Key
WAREHOUSENUMBER	VARCHAR2(250)	No		1
STREET	VARCHAR2(250)	Yes		
CITY	VARCHAR2(250)	Yes		
PHONE_NUMBER	VARCHAR2(250)	Yes		

SQL QUERY FOR WAREHOUSE

WAREHOUSE

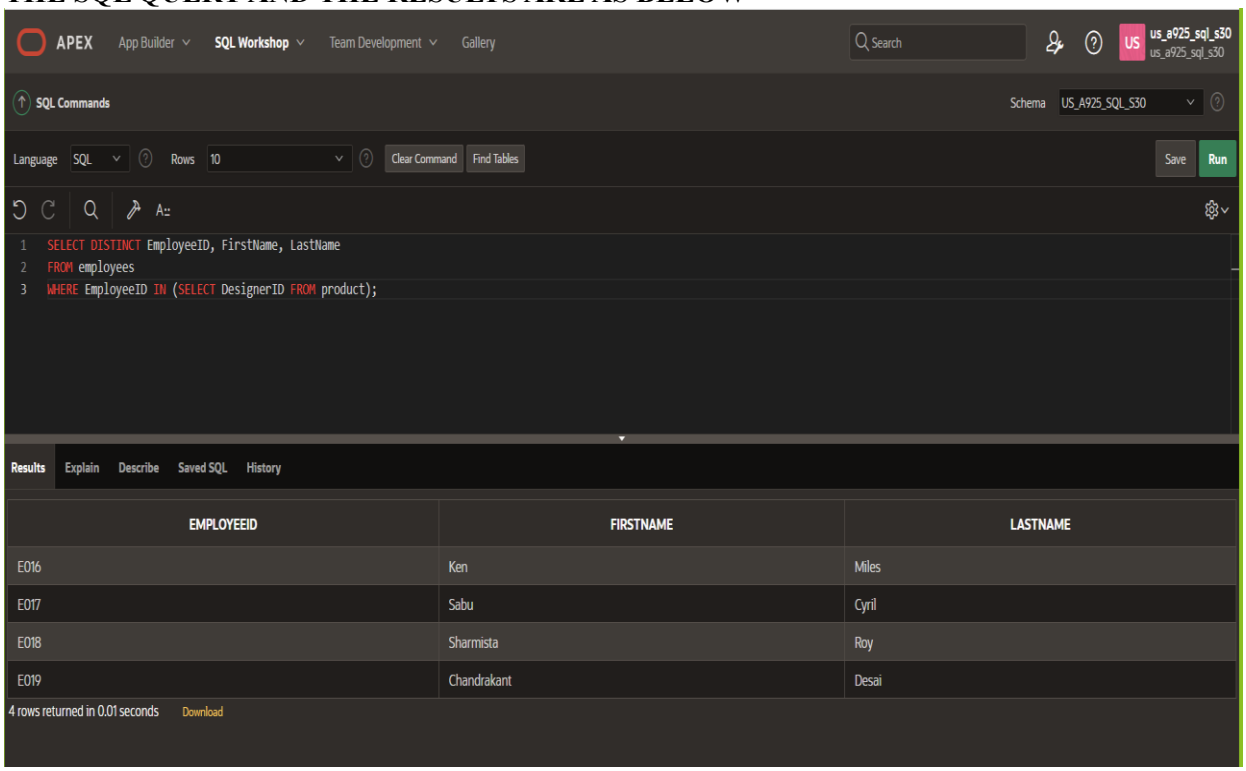
Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL

```
CREATE TABLE  "WAREHOUSE"
   (    "WAREHOUSENUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP" NOT NULL ENABLE,
        "STREET" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "CITY" VARCHAR2(250) COLLATE "USING_NLS_COMP",
        "PHONE_NUMBER" VARCHAR2(250) COLLATE "USING_NLS_COMP",
         CONSTRAINT "WAREHOUSE_PK" PRIMARY KEY ("WAREHOUSENUMBER")
  USING INDEX  ENABLE
   )  DEFAULT COLLATION "USING_NLS_COMP"
/
```

THREE MEANINGFUL QUERIES

1) QUERY: What is the query to extract information of the employees who are only responsible for designing the products ?

THE SQL QUERY AND THE RESULTS ARE AS BELOW



---

2) QUERY: What is the query to fetch distinct details about vendors who supplied raw materials after the date '11/10/23,' including VendorName, PhoneNumber, and the type of raw material they supplied?

THE SQL QUERY AND THE RESULTS ARE AS BELOW

```
SELECT distinct VendorName as supplier, PhoneNumber, RawmaterialName
FROM vendor
INNER JOIN supplyschedule ON vendor.VendorNumber = supplyschedule.VendorNumber
WHERE supplyschedule.Dates > '11/08/2023';
```

SUPPLIER	PHONENUMBER	RAWMATERIALNAME
BlueFab Textiles Ltd.	(555)1234567	Plastic
BlueFab Textiles Ltd.	(555)1234567	Cotton blend
YellowSports Equipments Inc.	(555)2345678	Wool
YellowSports Equipments Inc.	(555)2345678	Metal
YellowSports Equipments Inc.	(555)2345678	Foam
GreenTrend Accessories Co.	(555)3456789	Polyester

---

3) QUERY: What is the query to retrieve details such as ProductName and Price for products manufactured using the raw materials 'Vinyl' and 'Cotton,' with a cost exceeding 10 and is of red color?

THE SQL QUERY AND THE RESULTS ARE AS BELOW

```
SELECT PRODCUTTYPE, ProductName, Color, Cost , Price, Rawmaterialname AS UsedMaterial
FROM product
RIGHT JOIN rawmaterialproduct ON product.ProductNumber = rawmaterialproduct.ProductNumber
WHERE rawmaterialproduct.RawmaterialName IN ('Vinyl', 'Cotton')
  AND product.Cost > 10
  AND product.Color = 'Red';
```

PRODCUTTYPE	PRODUCTNAME	COLOR	COST	PRICE	USEDMATERIAL
Jersey	Galaxy United Jersey	Red	30	60	Vinyl
Gloves	Galaxy United Goalkeeper Gloves	Red	25	50	Cotton
Banners	Galaxy United Banners	Red	20	40	Vinyl
Kits	Galaxy United Maintenance Kits	Red	25	50	Cotton

4 rows returned in 0.01 seconds Download

CITATIONS

1. Modern Database Management, 13th edition (eBook or print versions) By: Jeff Hoffer, Ramesh Venkataraman, Heikki Topi