FINAL PROJECT PROPOSAL

INTEGRATED SUPPLY CHAIN AND FINANCIAL MANAGEMENT SYSTEM

CATERPILLAR INDIA PRIVATE LIMITED

Nihith Nath Kandikattu

SUBMITTED TO Prasanna Venkatesh

December 6th, 2019

Table of Contents

Project Overview………………………………………………………………………………… 3

Task 0: Repository………………………………………………………………………………. 5

Task 1: Primary Keys and Foreign Keys………………………………………………………. 6

Task 2: EER Diagram……………………………………………………………………………. 7

Task 3:

a) Relational Schema…………………………………………………………………………….8

b) Degree of Relationships and Cardinalities…………………………………………………9

c) Supertype and Subtype Relation…………………………………………………………….9

d) Constraints……………………………………………………………………………………

e) Anomalies………………………………………………………………………………………

Task 4:

a) Functional Dependencies……………………………………………………………………10

b) Normalization………………………………………………………………………………..11

Task 5: SQL Queries…………………………………………………………………………….13

Roles of Team Members……………………………………………………………………….19

PROJECT OVERVIEW

Introduction

The team has built a database schema for a smart database for a company that makes and supplies things. It will make everything run smoother - from designing products to getting materials from different places. The database will help organize how each department works and keep track of money matters, like invoices, in a careful way. The goal is to create a system that works well for the company, making sure everything is done right and everyone knows what they are supposed to do.

Objective

This project involves the effective management of employees across various departments, warehouses, and production lines, ensuring everyone is distinctly allocated to a specific entity. The focus also extends to overseeing the complete product lifecycle, from conceptualization in design departments to the manufacturing phase on production lines. Additionally, the project aims to streamline the flow of raw materials, whether originating from warehouses or directly supplied by vendors. A crucial aspect involves tracking and processing vendor-supplied invoices with clear sender identification to ensure seamless financial operations within the accounting department. Overall, the project aims to enhance organizational efficiency by addressing personnel allocation, product lifecycle oversight, raw material management, and financial processing.

Data

The project includes data on eight tables (including Employee, Invoice, Product, Product Line, Raw Material, Supply_schedule, Vendor and Warehouse). This has been used in the creation of the repository.

Business Rules

1) Each product must be produced by one specific production line.
2) Each production line can produce only one type of product.
3) For repair purposes, production lines may produce no products.

4) Each vendor can supply many raw materials to any number of warehouses.
5) Raw materials are supplied by any number of warehouses, which are supplied by any number of vendors.
6) Raw materials may also be directly supplied by vendors.
7) 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.

8) The company has departments, warehouses, and production lines.
9) The company designs and produces products.

10) Each department, warehouse, and production line have multiple employees.
11) Each employee works in only one department, warehouse, or production line.

12) Only Employees within the design department design products.
13) Each designer can design multiple products.
14) Each product has exactly one designer.

15) Vendors submit an invoice when they supply a raw material.
16) Invoices are processed by the accounting department.

17) If a warehouse shutdowns or production line repairs, no employees will be scheduled to work there.

TASK 0: REPOSITORY

Based on the data received, the following is the repository which contains the metadata and is a baseline for this project. There are fourteen entities/relations that have been created and each entity/relation consists of multiple attributes.

Product

Data	Datatype	Precision	Scale	Metadata Description
ProductNumber	VARCHAR2	50		ProductNumber is a field or attribute within a database table that serves as a unique identifier for each product.
ProductName	VARCHAR2	50		ProductName stores the name or title of a product
ProductType	VARCHAR2	20		ProductType is an attribute within a database table that categorizes products based on their type or classification.
DesignerID	VARCHAR2	10		DesignerID serves as a unique identifier for each designer.
Price	NUMBER	5	2	Price is an attribute within a database table that stores the monetary value associated with a product.
Cost	NUMBER	5	2	Cost stores information about the expenditure associated with producing or acquiring a product.
Color	VARCHAR2	10		Color is a field or attribute within a database table that stores information about the color of a product.
Weight (lbs)	NUMBER	4	2	Weight stores numerical values representing the mass or heaviness of a product.

ProductRawMaterial

Data	Datatype	Precision	Metadata Description
RawmaterialName	VARCHAR2	20	RawmaterialName is a field or attribute within a database table that stores the name or title of a raw material.
ProductNumber	VARCHAR2	50	ProductNumber is essential for distinguishing and accessing individual product records within the database.

ProductProductionLine

Data	Datatype	Precision	Metadata Description
ProductNumber	VARCHAR2	50	ProductNumber is essential for distinguishing and accessing individual product records within the database.
LineNumber	VARCHAR2	10	LineNumber is an attribute for distinguishing and accessing individual records within the database.

Employee

Data	Datatype	Precision	Metadata Description
EmployeeID	VARCHAR2	10	EmployeeID is crucial for distinguishing and accessing individual employee records within the database.
FirstName	VARCHAR2	50	FirstName is part of the overall database schema and holds textual data representing the given or first name of each record (e.g., person or employee).
LastName	VARCHAR2	50	LastName attribute within a database table stores the last or family name of an individual.
Position	VARCHAR2	50	Position stores information about the job title, role, or position held by an individual.
Salary	NUMBER	10	Salary is a field or attribute within a database table that stores information about the monetary compensation associated with an individual's employment
WorkCategory	VARCHAR2	50	WorkCategory represents the category or type of work associated with an entity.

Warehouse Employee

Data	Datatype	Precision	Metadata Description
EmployeeID	VARCHAR2	10	EmployeeID is related to only Warehouse Employees.
WarehouseNumber	VARCHAR2	10	WarehouseNumber is a database field serving as a unique identifier for individual warehouses.

Department Employee

Data	Datatype	Precision	Metadata Description
EmployeeID	VARCHAR2	10	EmployeeID is only related to the department employees.
DepartmentID	VARCHAR2	10	DepartmentID acts as a unique identifier for individual departments, facilitating organization and retrieval of department-specific information.

Line Employee

Data	Datatype	Precision	Metadata Description
EmployeeID	VARCHAR2	10	EmployeeID is related to only Line Employees
LineNumber	VARCHAR2	10	LineNumber is a field or attribute within a database table that denotes the position or sequence of a record within the dataset.

Production Line

Data	Datatype	Precision	Metadata Description
LineNumber	VARCHAR2	10	LineNumber is a field or attribute within a database table that denotes the position or sequence of a record within the dataset.
LineCapacity (items/hour)	NUMBER	4	LineCapacity stores numerical values indicating the quantity or volume that a specific production line can handle
PhoneNumber	VARCHAR2	15	PhoneNumber is an attribute within a database table that stores numerical values representing a phone number associated with an entity, such as a person or a business.
StreetAddress	VARCHAR2	50	StreetAddress is a database field that stores the textual representation of the street or physical address linked to an entity.
Location	VARCHAR2	50	Location represents a field storing information about the geographical or physical location associated with an entity.
Zipcode	VARCHAR2	10	ZipCode is an attribute capturing numerical data representing the postal code associated with an entity.

Supply Schedule

Data	Datatype	Precision	Metadata Description
SupplyCode	VARCHAR2	10	SupplyCode is a field or attribute within a database table that serves as a unique identifier for a supply source or entity.
ProductNumber	VARCHAR2	50	ProductNumber is a field that is essential for distinguishing and accessing individual product records within the database.
RawmaterialName	VARCHAR2	50	RawmaterialName is an attribute that is essential for organizing and retrieving information about individual raw materials within the database.
WarehouseNumber	VARCHAR3	10	WarehouseNumber is a field or attribute within a database table that serves as a unique identifier for each warehouse.
VendorNumber	VARCHAR4	10	VendorNumber is typically assigned as a unique numerical or alphanumeric code to ensure that each vendor has a distinct identification for various database operations, such as querying, updating, or referencing vendor information.
SupplyDate	DATE		SupplyDate is a field or attribute within a database table that stores information about the date on which a supply was received or provided.

Warehouse

Data	Datatype	Precision	Metadata Description
WarehouseNumber	VARCHAR2	10	WarehouseNumber is a database field serving as a unique identifier for individual warehouses.
StreetAddress	VARCHAR2	50	StreetAddress is a database field that stores the textual representation of the street or physical address linked to an entity.
CityName	VARCHAR2	20	CityName is a field or an attribute storing textual data representing the name of a city associated with an entity.
PhoneNumber	VARCHAR2	15	PhoneNumber in DBMS is a database field storing numerical data representing the phone number associated with an entity.

Vendor

Data	Datatype	Precision	Metadata Description
VendorNumber	VARCHAR2	10	VendorNumber is a unique identifier for individual vendors or suppliers.
VendorName	VARCHAR2	50	VendorName is an attribute or field storing textual data representing the name of a vendor or supplier associated with an entity.
StreetAddress	VARCHAR2	50	StreetAddress is a database field that stores the textual representation of the street or physical address linked to an entity.
CityName	VARCHAR2	20	CityName is a field or an attribute storing textual data representing the name of a city associated with an entity.
PhoneNumber	VARCHAR2	15	PhoneNumber is a database field storing numerical data representing the phone number associated with an entity.

Invoice

Data	Datatype	Precision	Metadata Description
InvoiceNumber	VARCHAR2	10	InvoiceNumber in a DBMS is a database field serving as a unique identifier for individual invoices, facilitating tracking and management of financial transactions.
TotalAmount	VARCHAR2	5	TotalAmount is the field that holds the numerical value representing the total cost or sum related to a financial transaction.
VendorNumber	VARCHAR2	10	VendorNumber is a unique identifier for individual vendors or suppliers.
DepartmentID	VARCHAR2	10	DepartmentID acts as a unique identifier for individual departments, facilitating organization and retrieval of department-specific information.

Vendor_Payment

Data	Datatype	Precision	Metadata Description
VendorNumber	VARCHAR2	10	VendorNumber is a unique identifier for individual vendors or suppliers.
VendorPaymentType	VARCHAR2	20	VendorPaymentType is a database field representing the payment method or type associated with a vendor or supplier.

Department

Data	Datatype	Precision	Metadata Description
DepartmentID	VARCHAR2	10	DepartmentID acts as a unique identifier for individual departments, facilitating organization and retrieval of department-specific information.
DepatmentName	VARCHAR2	50	DepartmentName is a database field storing textual data representing the name of a department, facilitating organization and retrieval of department-specific information.
StreetAddress	VARCHAR2	50	StreetAddress is a database field that stores the textual representation of the street or physical address linked to an entity.
CityName	VARCHAR2	20	CityName stores textual data representing the name of a city associated with an entity.
NumberOfEmployees	NUMBER	2	NumberOfEmployees represents the numerical count of employees associated with a particular entity or department.
PhoneNumber	VARCHAR2	15	PhoneNumber is a database field storing numerical data representing the phone number associated with an entity.

TASK 1: PRIMARY KEYS AND FOREIGN KEYS

The following table has a list of primary and foreign keys identified based on the fourteen entities/relations.

Entity	Primary Key(s)	Foreign Key(s)	Comments
Product	ProductNumber	DesignerID, LineNumber	Product has many side in relations with department_table,ProductionLine so it will have DesignerID, LineNumber as foreign keys
ProductRawMaterial	RawMaterialName, ProductNumber	ProductNumber	ProductNumber here is a foreign key as it is considered as composite primary key.
ProductProductionLine	ProductNumber	ProductNumber, LineNumber	ProductNumber is both primarykey and foreignkey
Production Line	LineNumber	-	Production line entity does not have a foreign key and line number is the primary key
Supply_Schedule	SupplyCode	ProductNumber, RawMaterialName, WarehouseNumber, VendorNumber	This is an associative entity created after removing a ternary relationship between rawmaterial,vendor,Warehouse
Warehouse	WarehouseNumber	-	This entity does not have a foreign key and warehouse number is the primary key
Vendor	VendorNumber	-	This entity does not have a foreign key and vendorNumber is the primary key
Invoice	InvoiceNumber	VendorNumber, DepartmentID	The invoice table has many side and gets foreign from vendor table and Department
Vendor_Payment	VendorNumber	-	This entity is created after normalization to remove transitive dependency in the invoice table
Department	DepartmentID	-	Department entity has no foreign key
Employee	EmployeeID	-	Employee entity is a supertype entity
WarehouseEmployee	WEmployeeID	WEmployeeID, WarehouseNumber	Warehouse employee entity has many side and get foreign key from Employee, WEmployeeID is both primary and foreign key
DepartmentEmployee	DEmployeeID	DEmployeeID, DepartmentID	Department employee entity has many side and get foreign key from Employee, DEmployeeID is both primary and foreign key
LineEmployee	LEmployeeID	LEmployeeID, LineNumber	Line employee entity has many side and get foreign key from

			Employee, LEmployeeID is both primary and foreign key

TASK 2: EER DIAGRAM

The following EER diagram is a representation of all the seven relations.

Note: This EER diagram is not the normalized version.

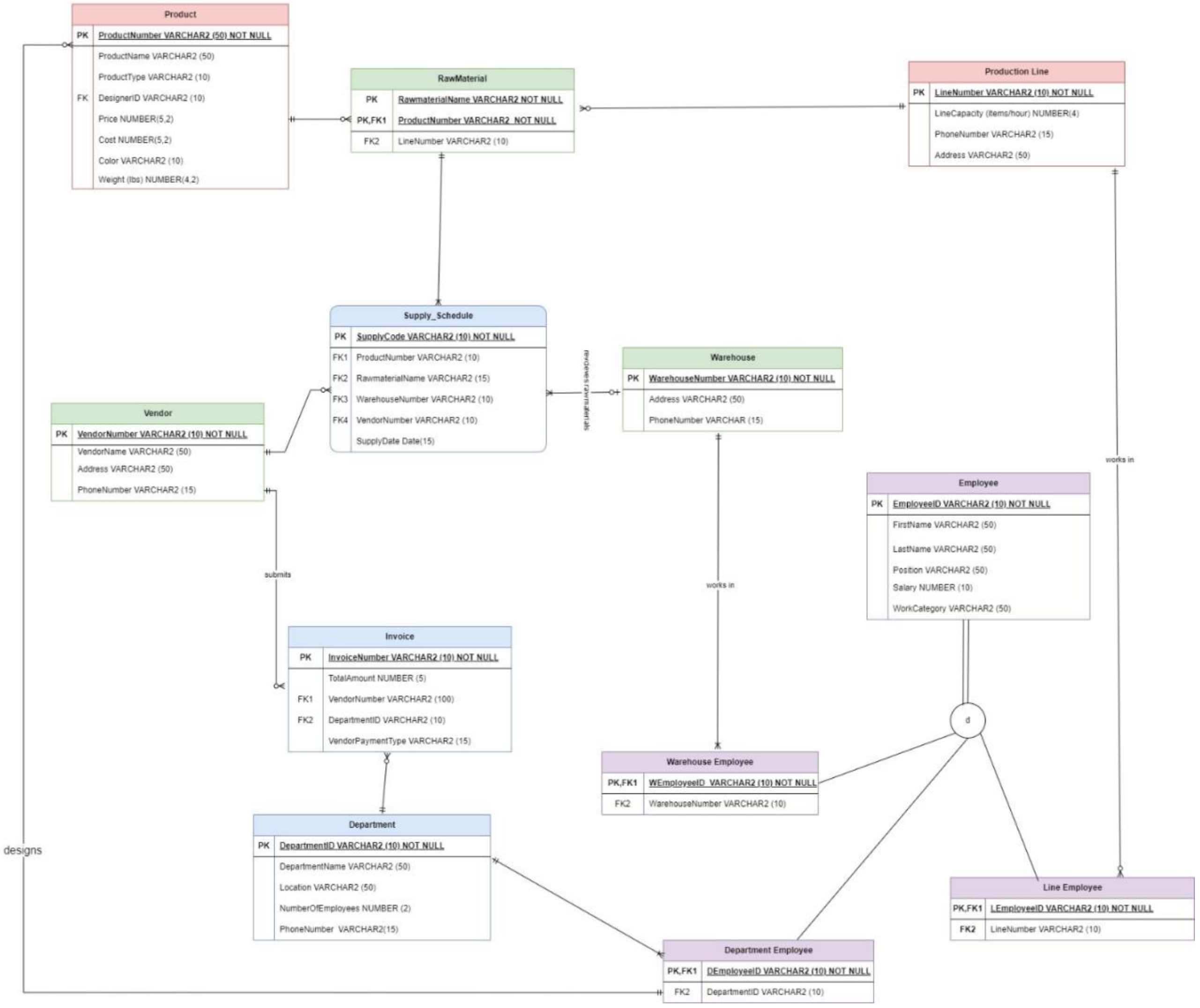
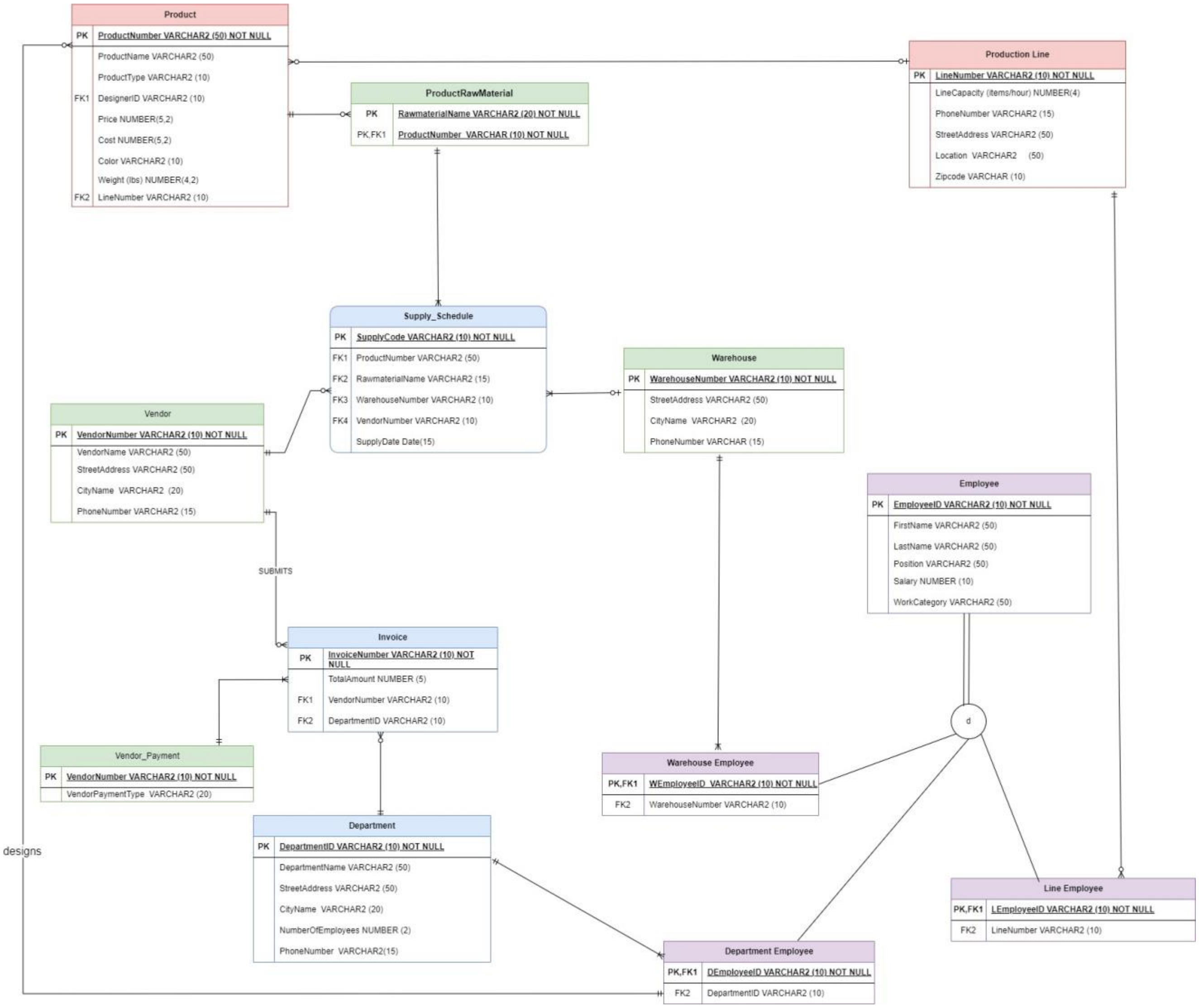

Product
PK ProductNumber VARCHAR2 (50) NOT NULL
ProductName VARCHAR2 (50)
ProductType VARCHAR2 (10)
FK DesignerID VARCHAR2 (10)
Price NUMBER(5,2)
Cost NUMBER(5,2)
Color VARCHAR2 (10)
Weight (lbs) NUMBER(4,2)
RawMaterial
PK RawmaterialName VARCHAR2 NOT NULL
PK,FK1 ProductNumber VARCHAR2 NOT NULL
FK2 LineNumber VARCHAR2 (10)
Production Line
PK LineNumber VARCHAR2 (10) NOT NULL
LineCapacity (items/hour) NUMBER(4)
PhoneNumber VARCHAR2 (15)
Address VARCHAR2 (50)
Supply_Schedule
PK SupplyCode VARCHAR2 (10) NOT NULL
FK1 ProductNumber VARCHAR2 (10)
FK2 RawmaterialName VARCHAR2 (15)
FK3 WarehouseNumber VARCHAR2 (10)
FK4 VendorNumber VARCHAR2 (10)
SupplyDate Date(15)
recieves rawmaterials
Warehouse
PK WarehouseNumber VARCHAR2 (10) NOT NULL
Address VARCHAR2 (50)
PhoneNumber VARCHAR (15)
Vendor
PK VendorNumber VARCHAR2 (10) NOT NULL
VendorName VARCHAR2 (50)
Address VARCHAR2 (50)
PhoneNumber VARCHAR2 (15)
works in
Employee
PK EmployeeID VARCHAR2 (10) NOT NULL
FirstName VARCHAR2 (50)
LastName VARCHAR2 (50)
Position VARCHAR2 (50)
Salary NUMBER (10)
WorkCategory VARCHAR2 (50)
submits
works in
Invoice
PK InvoiceNumber VARCHAR2 (10) NOT NULL
TotalAmount NUMBER (5)
FK1 VendorNumber VARCHAR2 (100)
FK2 DepartmentID VARCHAR2 (10)
VendorPaymentType VARCHAR2 (15)
d
Warehouse Employee
PK,FK1 WEmployeeID VARCHAR2 (10) NOT NULL
FK2 WarehouseNumber VARCHAR2 (10)
Department
PK DepartmentID VARCHAR2 (10) NOT NULL
DepartmentName VARCHAR2 (50)
Location VARCHAR2 (50)
NumberOfEmployees NUMBER (2)
PhoneNumber VARCHAR2(15)
designs
Line Employee
PK,FK1 LEmployeeID VARCHAR2 (10) NOT NULL
FK2 LineNumber VARCHAR2 (10)
Department Employee
PK,FK1 DEmployeeID VARCHAR2 (10) NOT NULL
FK2 DepartmentID VARCHAR2 (10)

Fig: EER Diagram Before Normalization

EER diagram after normalization,



Product

PK	ProductNumber VARCHAR2 (50) NOT NULL
	ProductName VARCHAR2 (50)
	ProductType VARCHAR2 (10)
FK1	DesignerID VARCHAR2 (10)
	Price NUMBER(5,2)
	Cost NUMBER(5,2)
	Color VARCHAR2 (10)
	Weight (lbs) NUMBER(4,2)
FK2	LineNumber VARCHAR2 (10)

ProductRawMaterial

PK	RawmaterialName VARCHAR2 (20) NOT NULL
PK,FK1	ProductNumber VARCHAR (10) NOT NULL

Production Line

PK	LineNumber VARCHAR2 (10) NOT NULL
	LineCapacity (items/hour) NUMBER(4)
	PhoneNumber VARCHAR2 (15)
	StreetAddress VARCHAR2 (50)
	Location VARCHAR2 (50)
	Zipcode VARCHAR (10)

Supply_Schedule

PK	SupplyCode VARCHAR2 (10) NOT NULL
FK1	ProductNumber VARCHAR2 (50)
FK2	RawmaterialName VARCHAR2 (15)
FK3	WarehouseNumber VARCHAR2 (10)
FK4	VendorNumber VARCHAR2 (10)
	SupplyDate Date(15)

Warehouse

PK	WarehouseNumber VARCHAR2 (10) NOT NULL
	StreetAddress VARCHAR2 (50)
	CityName VARCHAR2 (20)
	PhoneNumber VARCHAR (15)

Vendor

PK	VendorNumber VARCHAR2 (10) NOT NULL
	VendorName VARCHAR2 (50)
	StreetAddress VARCHAR2 (50)
	CityName VARCHAR2 (20)
	PhoneNumber VARCHAR2 (15)

Employee

PK	EmployeeID VARCHAR2 (10) NOT NULL
	FirstName VARCHAR2 (50)
	LastName VARCHAR2 (50)
	Position VARCHAR2 (50)
	Salary NUMBER (10)
	WorkCategory VARCHAR2 (50)

Invoice

PK	InvoiceNumber VARCHAR2 (10) NOT NULL
	TotalAmount NUMBER (5)
FK1	VendorNumber VARCHAR2 (10)
FK2	DepartmentID VARCHAR2 (10)

Vendor_Payment

PK	VendorNumber VARCHAR2 (10) NOT NULL
	VendorPaymentType VARCHAR2 (20)

Warehouse Employee

PK,FK1	WEmployeeID VARCHAR2 (10) NOT NULL
FK2	WarehouseNumber VARCHAR2 (10)

Department

PK	DepartmentID VARCHAR2 (10) NOT NULL
	DepartmentName VARCHAR2 (50)
	StreetAddress VARCHAR2 (50)
	CityName VARCHAR2 (20)
	NumberOfEmployees NUMBER (2)
	PhoneNumber VARCHAR2(15)

Line Employee

PK,FK1	LEmployeeID VARCHAR2 (10) NOT NULL
FK2	LineNumber VARCHAR2 (10)

Department Employee

PK,FK1	DEmployeeID VARCHAR2 (10) NOT NULL
FK2	DepartmentID VARCHAR2 (10)

Fig: EER Diagram after Normalization

Below is the detailed schematic view of the Dependencies Pre-Normalization

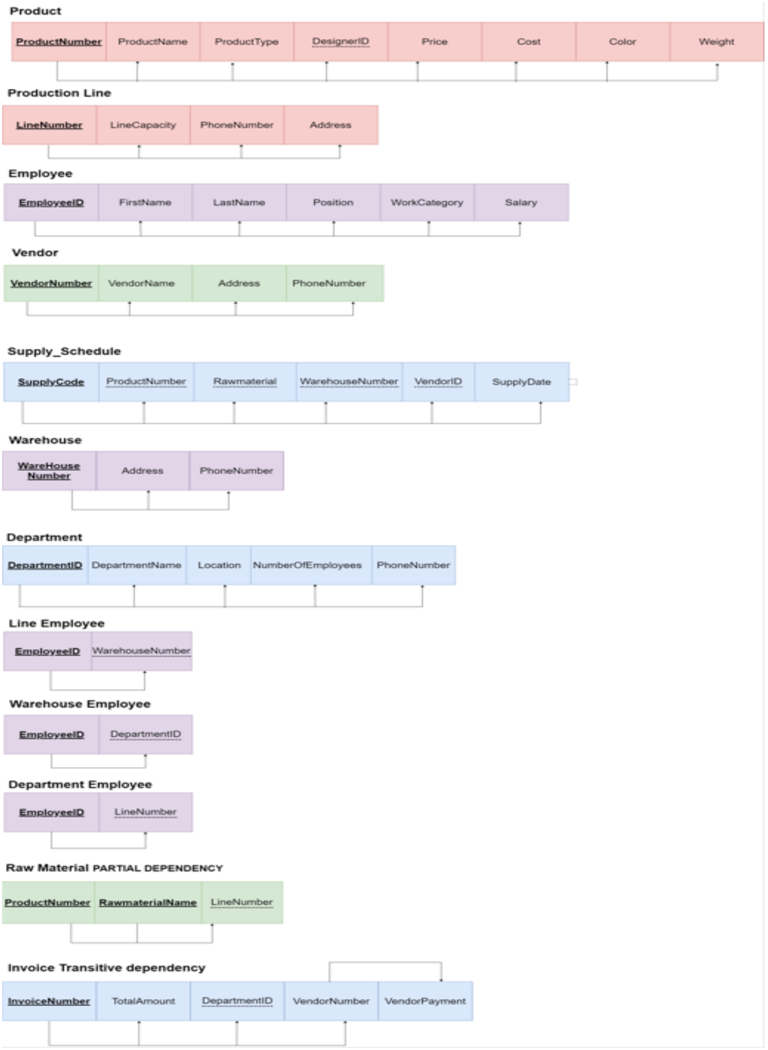


Fig: Dependencies Pre-Norm

Below is the detailed schematic view of the Dependencies Post-Normalization

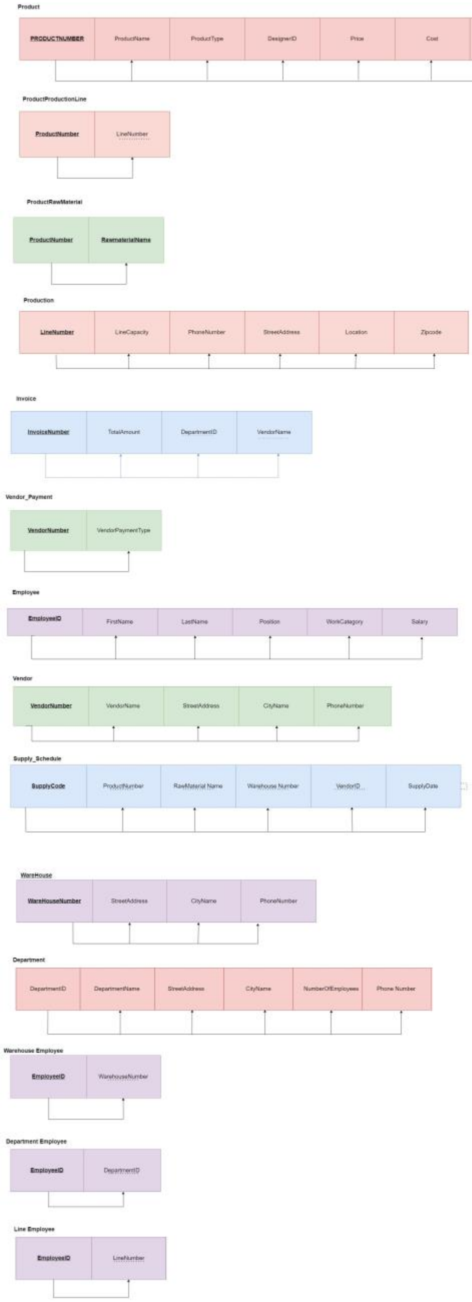


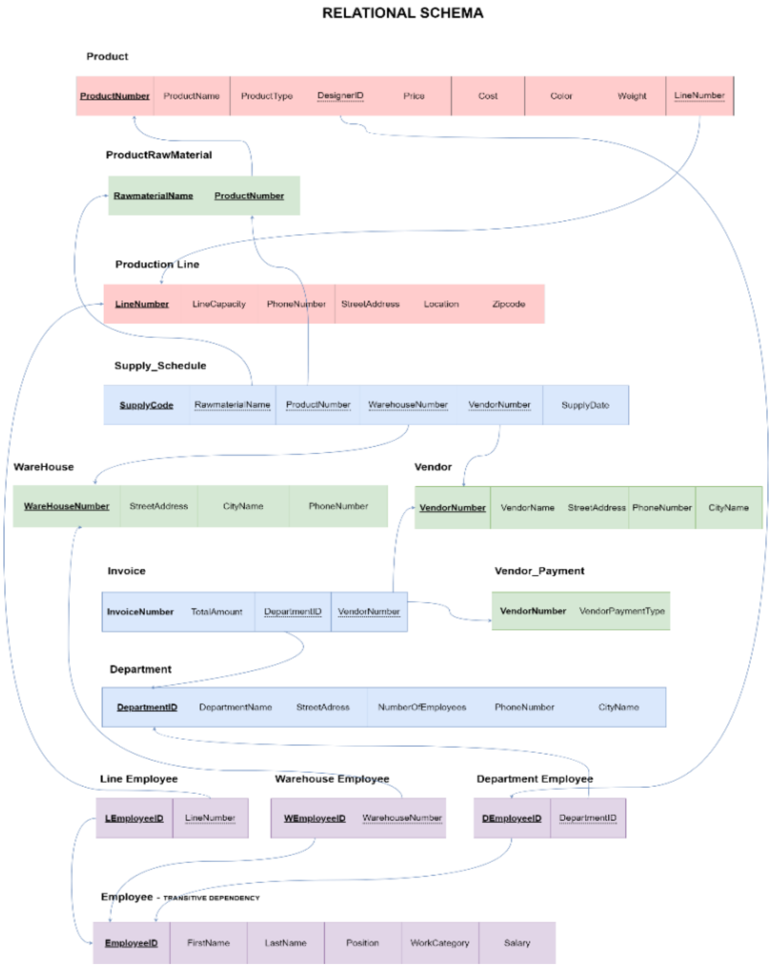
Fig: Dependencies afterNormalization

TASK 3:

a) RELATIONAL SCHEMA

Using the below three-step procedure, the EER diagram (from Task 2) has been translated to relations:
i. Regular entity mapping
ii. Binary relationship mapping
iii. Associative entity mapping

The mapping for weak entities, unary / ternary, supertype & subtype relationships, and other relationships not found in the EER diagram has been removed from the previously described procedure.


RELATIONAL SCHEMA
Product
ProductNumber | ProductName | ProductType | DesignerID | Price | Cost | Color | Weight | LineNumber
ProductRawMaterial
RawmaterialName | ProductNumber
Production Line
LineNumber | LineCapacity | PhoneNumber | StreetAddress | Location | Zipcode
Supply_Schedule
SupplyCode | RawmaterialName | ProductNumber | WarehouseNumber | VendorNumber | SupplyDate
WareHouse
WareHouseNumber | StreetAddress | CityName | PhoneNumber
Vendor
VendorNumber | VendorName | StreetAddress | PhoneNumber | CityName
Invoice
InvoiceNumber | TotalAmount | DepartmentID | VendorNumber
Vendor_Payment
VendorNumber | VendorPaymentType
Department
DepartmentID | DepartmentName | StreetAdress | NumberOfEmployees | PhoneNumber | CityName
Line Employee
LEmployeeID | LineNumber
Warehouse Employee
WEmployeeID | WarehouseNumber
Department Employee
DEmployeeID | DepartmentID
Employee - TRANSITIVE DEPENDENCY
EmployeeID | FirstName | LastName | Position | WorkCategory | Salary

b) DEGREE OF RELATIONSHIPS AND CARDINALITIES

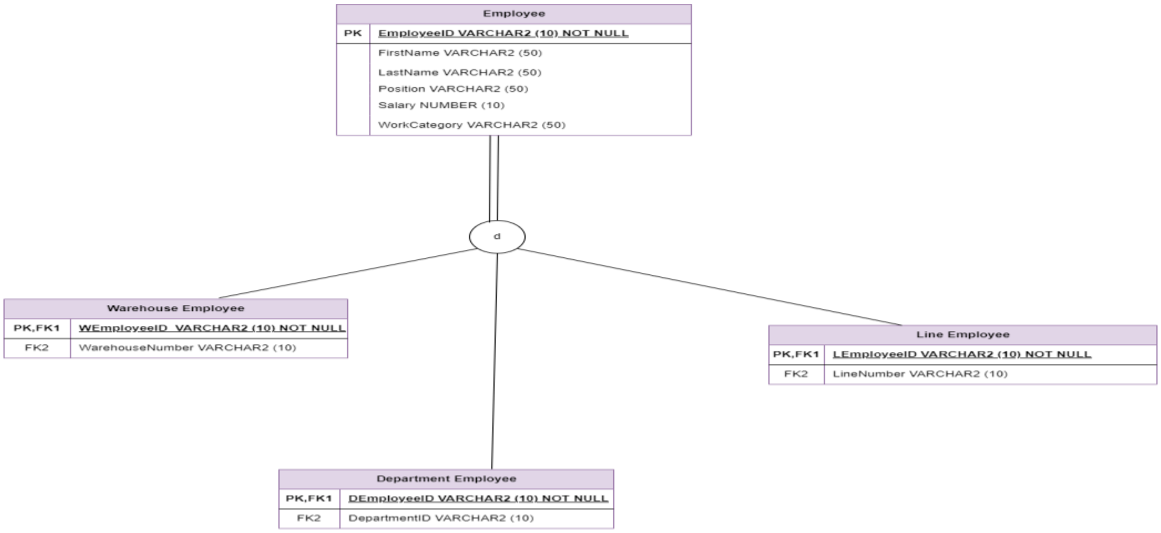
In the above EER model, there is no unary relationship i.e., entities of the same type which are related to each other. Also, there is one ternary relationship, in cluster 2 where an associate entry Supply Schedule is added as shown in the below figure. There are multiple binary relationships that exist within the EER model, which have been mapped out into relations.

Entity 1	Entity 2	Cardinality	Description
Product	ProductRawMaterial	1:M	"PRODUCT" relates to "ProductRawMaterial" in a one-to-many setup, signifying that each product can be made up with multiple raw materials.
Product	ProductionLine	M:1	"PRODUCT" to “ProductionLine” has a many-to-one relationship, indicating that multiple products can be processed in a single production line.
Product	DepartmentEmployee	M:1	"PRODUCT" links to the “DepartmentEmployee” in a many-to-one setup, meaning multiple products can be manufactured with a single department employee.
ProductRawMaterial	Supply_Schedule	1:M	"ProductRawMaterial" “Supply_Schedule” exhibits a one-to-many relationship, indicating that each raw material can have multiple entries in the supply schedule.
Supply_Schedule	Warehouse	M:1	"Supply_Schedule" for a “Warehouse” reflects a many-to-one relationship, implying that multiple entries in the supply schedule can be associated with a single warehouse.
Supply_Schedule	Vendor	M:1	"Supply_Schedule" is a many-to-one relationship with the "Vendor," implying that multiple supply schedules can be associated with a single vendor.
Vendor	Invoice	1:M	"Vendor" has a one-to-many relationship with "Invoice," indicating that a single vendor can be associated with multiple invoices.
Vendor_Payment	Invoice	1:M	"Vendor_Payment" to "Invoice" showcases a one-to-many relationship, meaning a single vendor payment can be linked to multiple invoices.
Invoice	Department	M:1	"Invoice" to "Department" demonstrates a many-to-one relationship, indicating that multiple invoices can be associated with a single department.

Warehouse	Warehouse Employee	1:M	"Each warehouse employs multiple warehouse employees."
Production Line	Line Employee	1:M	"A production line employs multiple line employees."
Department	Department Employee	1:M	"Each department has multiple employees."

c) SUPERTYPE SUBTYPE RELATION

In the model, we get one supertype subtype relation from cluster 3. The below figure shows the supertype subtype relation.


Employee
PK
EmployeeID VARCHAR2 (10) NOT NULL
FirstName VARCHAR2 (50)
LastName VARCHAR2 (50)
Position VARCHAR2 (50)
Salary NUMBER (10)
WorkCategory VARCHAR2 (50)
d
Warehouse Employee
PK,FK1
WEmployeeID VARCHAR2 (10) NOT NULL
FK2
WarehouseNumber VARCHAR2 (10)
Line Employee
PK,FK1
LEmployeeID VARCHAR2 (10) NOT NULL
FK2
LineNumber VARCHAR2 (10)
Department Employee
PK,FK1
DEmployeeID VARCHAR2 (10) NOT NULL
FK2
DepartmentID VARCHAR2 (10)

d) CONSTRAINTS

There are three types of integrity constraints that are applicable and have been verified with respect to the dataset provided:

i. **Domain constraints**: All the data tables for the mentioned entities/relations have allowable values according to the specified data types.
ii. **Entity integrity**: None of the primary keys identified for the relations have null values.
iii. **Referential integrity**: Every foreign key assigned in each of the above relations has a corresponding primary key in the relation on the other side.

e) Anamolies:

Insertion Anamoly:

When adding a new raw material for making Product B in the Raw Material table, we're required to include a Line Number, even if it's not needed specifically for this product.

Deletion Anamoly:

When trying to stop making a product, it's hard to remove its details without taking out the Line Number linked to it. Similarly, if a production line is shut down or being fixed, it's tough to delete its data without also removing the connected product information.

TASK 4:

a) Identifying the dependencies

1. FULL DEPENDENCY

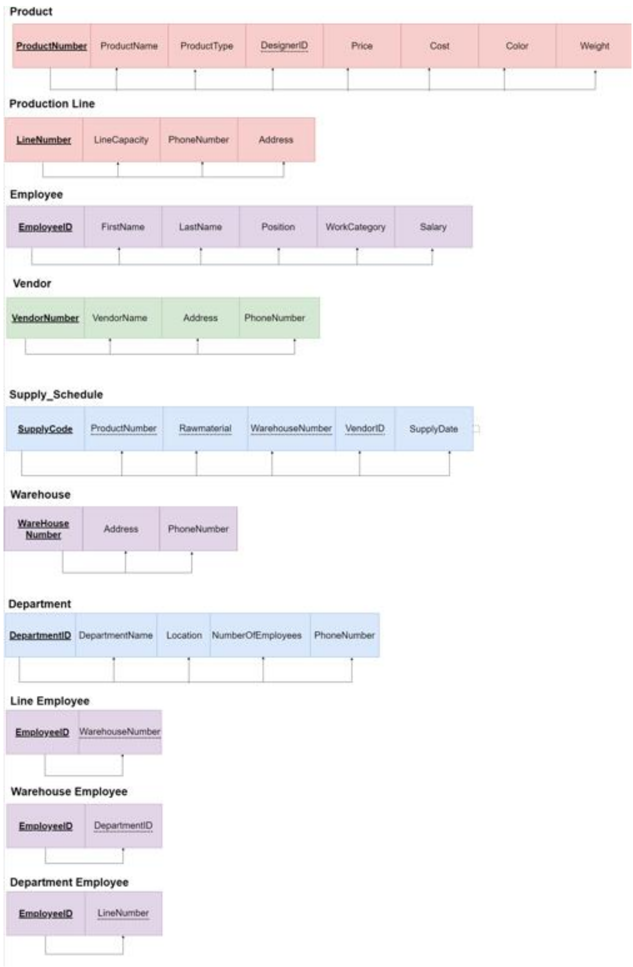


Fig: Full Dependencies

2. PARTIAL DEPENDENCY

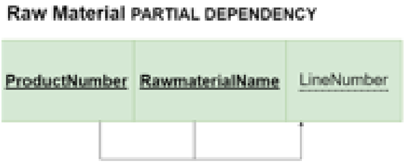


Fig: Partial Dependency of Raw Material

3. TRANSITIVE DEPENDENCY

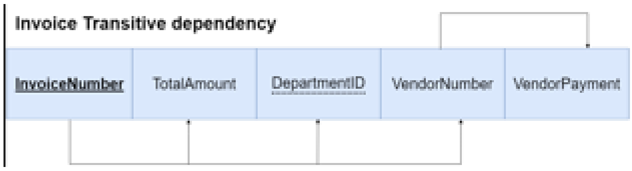


Fig: Transitive Dependency

b) Normalization

To remove anomalies from the relations, a normalization process has been undertaken. The three-step process includes:

i. First Normal Form (1NF): This step ensures that there are no multi-valued attributes in the relations/tables. In this project, there are no multi-valued attributes in any of the relations. Hence, the relations are in the 1NF form.

ii. Second Normal Form (2NF): In addition to ensuring that the relations/tables are in 1NF, this step also requires the relations/tables to not have any partial dependencies. In the relations mentioned above there is one partial dependency (Raw Material). To normalize we eliminate that convert into two tables as shown below.

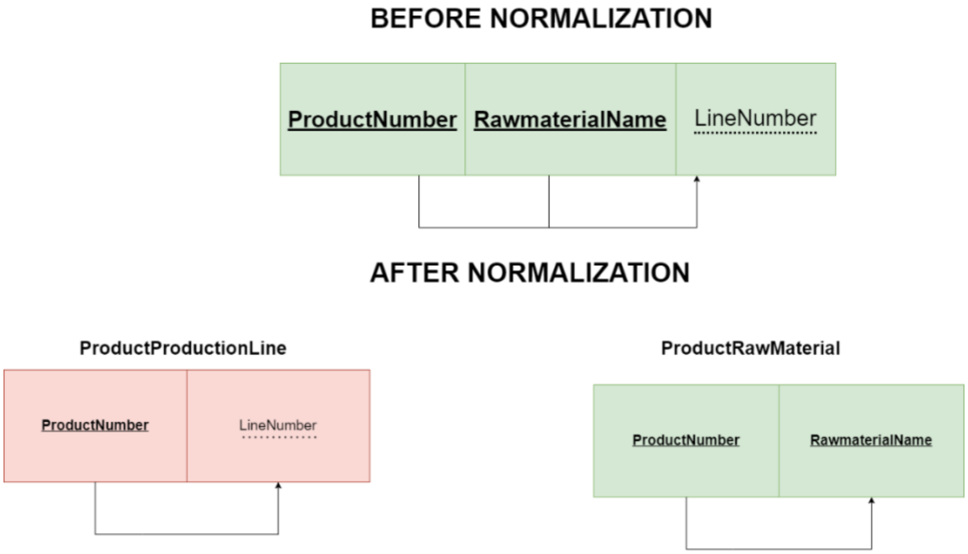


Fig: 2NF process for Product

Hence, the relations are now all in 2NF. The Table ProuctProductionLine has mandatory-mandatory cardinality with Products table, so we have added the LineNumber attribute to Product Table.

We have made a change to the way products are connected to production lines in our database. Instead of having a separate table, we added a "LineNumber" directly to the Products table to simplify the relationship between products and production lines.

iii. Third Normal Form (3NF): Along with the relations/tables being in the 2NF form, the 3NF step requires removal of any transitive dependencies. In the above, we get transitive dependency from the Invoice. To normalize we eliminate into two tables as shown below.

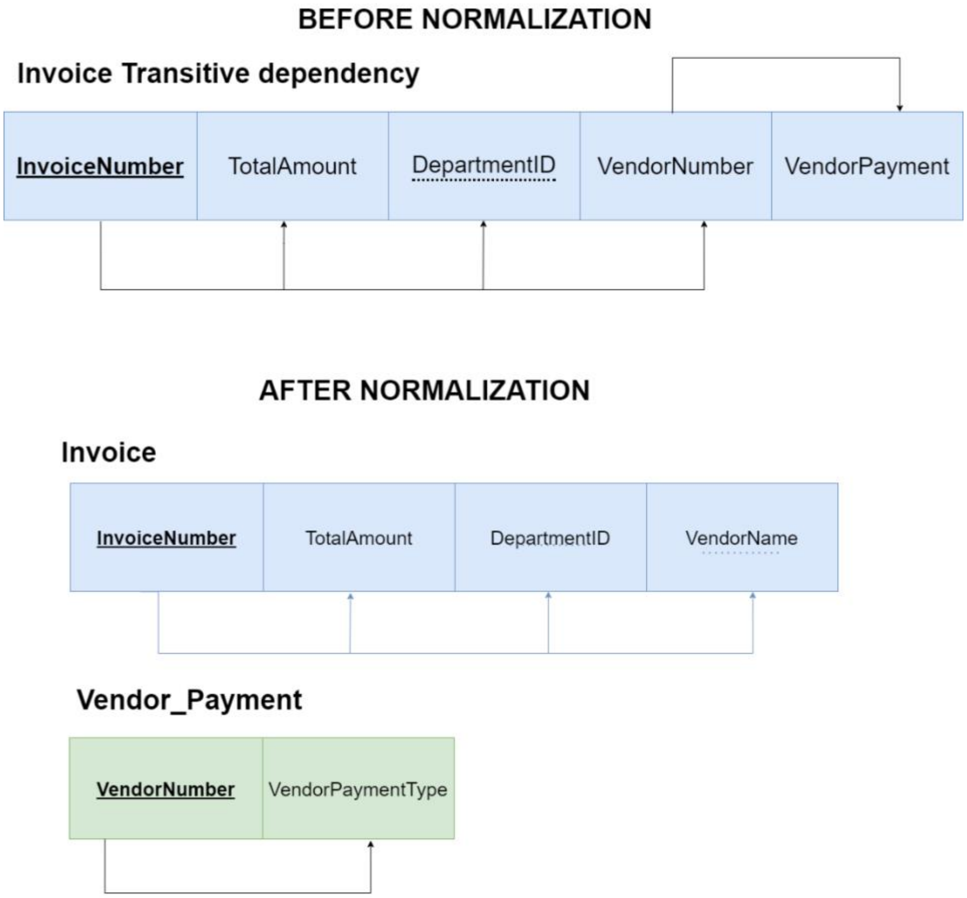


Fig: 3NF process for Invoice

These three steps ensure that the relations and data are normalized. After normalization the ER diagram look like below figure

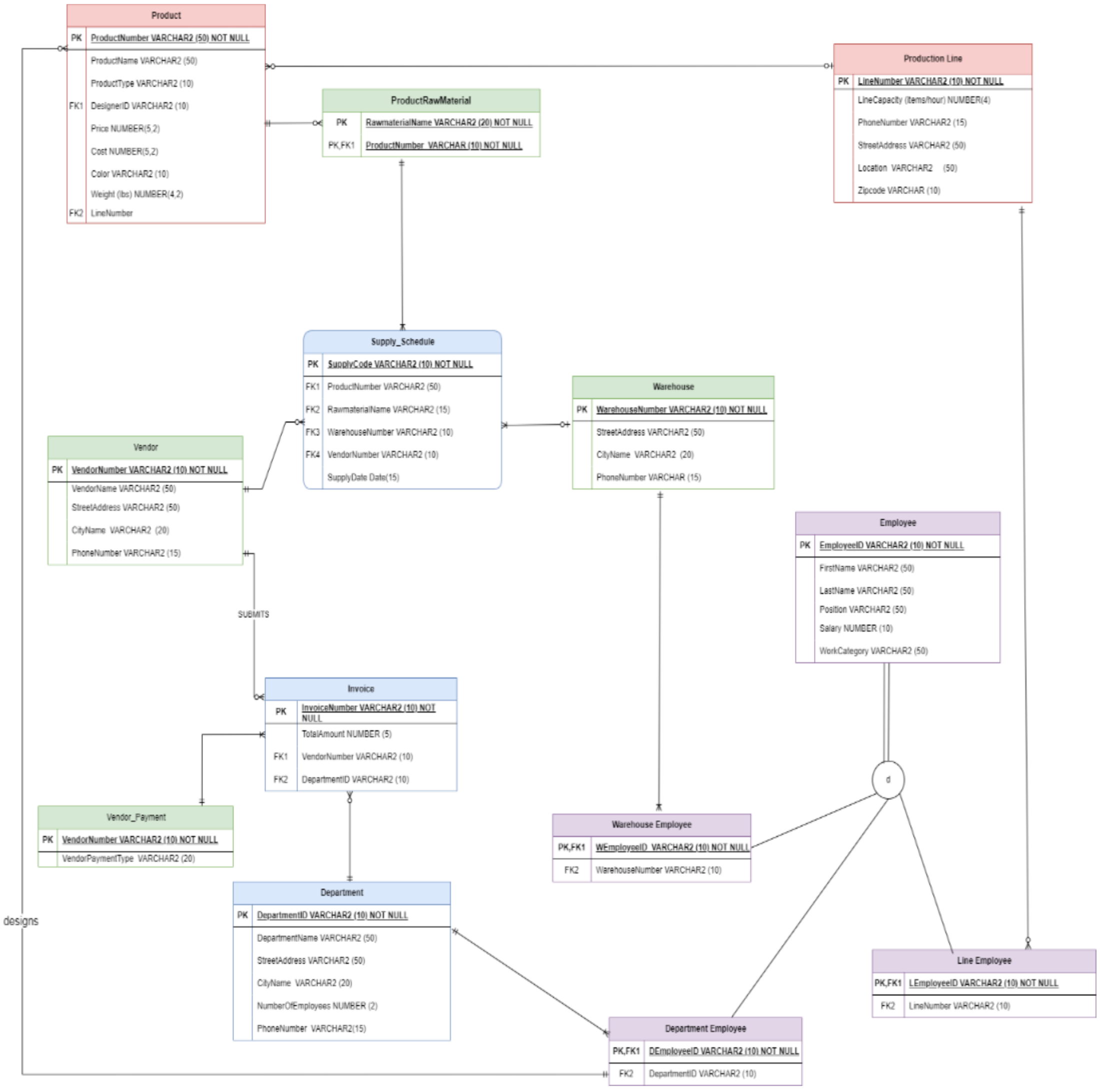
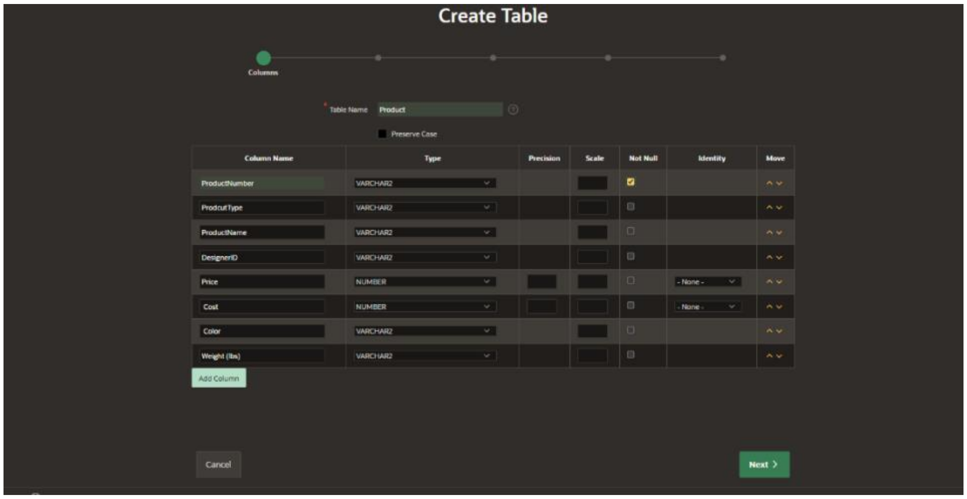

Product
PK ProductNumber VARCHAR2 (50) NOT NULL
ProductName VARCHAR2 (50)
ProductType VARCHAR2 (10)
FK1 DesignerID VARCHAR2 (10)
Price NUMBER(5,2)
Cost NUMBER(5,2)
Color VARCHAR2 (10)
Weight (lbs) NUMBER(4,2)
FK2 LineNumber
Production Line
PK LineNumber VARCHAR2 (10) NOT NULL
LineCapacity (Items/hour) NUMBER(4)
PhoneNumber VARCHAR2 (15)
StreetAddress VARCHAR2 (50)
Location VARCHAR2 (50)
Zipcode VARCHAR (10)
ProductRawMaterial
PK RawmaterialName VARCHAR2 (20) NOT NULL
PK,FK1 ProductNumber VARCHAR (10) NOT NULL
Supply_Schedule
PK SupplyCode VARCHAR2 (10) NOT NULL
FK1 ProductNumber VARCHAR2 (50)
FK2 RawmaterialName VARCHAR2 (15)
FK3 WarehouseNumber VARCHAR2 (10)
FK4 VendorNumber VARCHAR2 (10)
SupplyDate Date(15)
Warehouse
PK WarehouseNumber VARCHAR2 (10) NOT NULL
StreetAddress VARCHAR2 (50)
CityName VARCHAR2 (20)
PhoneNumber VARCHAR (15)
Vendor
PK VendorNumber VARCHAR2 (10) NOT NULL
VendorName VARCHAR2 (50)
StreetAddress VARCHAR2 (50)
CityName VARCHAR2 (20)
PhoneNumber VARCHAR2 (15)
Employee
PK EmployeeID VARCHAR2 (10) NOT NULL
FirstName VARCHAR2 (50)
LastName VARCHAR2 (50)
Position VARCHAR2 (50)
Salary NUMBER (10)
WorkCategory VARCHAR2 (50)
SUBMITS
Invoice
PK InvoiceNumber VARCHAR2 (10) NOT NULL
TotalAmount NUMBER (5)
FK1 VendorNumber VARCHAR2 (10)
FK2 DepartmentID VARCHAR2 (10)
d
Vendor_Payment
PK VendorNumber VARCHAR2 (10) NOT NULL
VendorPaymentType VARCHAR2 (20)
Warehouse Employee
PK,FK1 WEmployeeID VARCHAR2 (10) NOT NULL
FK2 WarehouseNumber VARCHAR2 (10)
designs
Department
PK DepartmentID VARCHAR2 (10) NOT NULL
DepartmentName VARCHAR2 (50)
StreetAddress VARCHAR2 (50)
CityName VARCHAR2 (20)
NumberOfEmployees NUMBER (2)
PhoneNumber VARCHAR2(15)
Line Employee
PK,FK1 LEmployeeID VARCHAR2 (10) NOT NULL
FK2 LineNumber VARCHAR2 (10)
Department Employee
PK,FK1 DEmployeeID VARCHAR2 (10) NOT NULL
FK2 DepartmentID VARCHAR2 (10)

Fig: The ER diagram after Post Normalization

TASK 5: SQL QUERIES

I. SQL ‘CREATE’ Query for building every relation with Entity Integrity

As a first step, the tables have been created on Apex. The process requires building a table for each entity, assigning column names (attributes), type and Precision (Precision). Once this information is appended, identification of primary keys and foreign keys is necessary. This completes the process for creating the table. Following is a screenshot of a table called ‘Product’.



Create Table

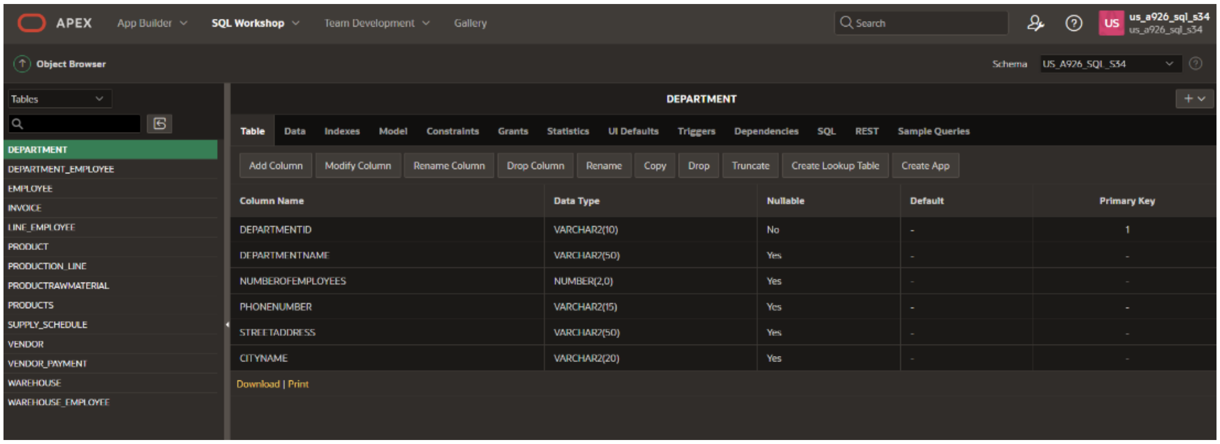
Table Name: Product

■ Preserve Case

Column Name	Type	Precision	Scale	Not Null	Identity	Move
ProductNumber	VARCHAR2			☑		^ v
ProdcutType	VARCHAR2			□		^ v
ProductName	VARCHAR2			□		^ v
DesignerID	VARCHAR2			□		^ v
Price	NUMBER			□	- None -	^ v
Cost	NUMBER			□	- None -	^ v
Color	VARCHAR2			□		^ v
Weight (lbs)	VARCHAR2			□		^ v

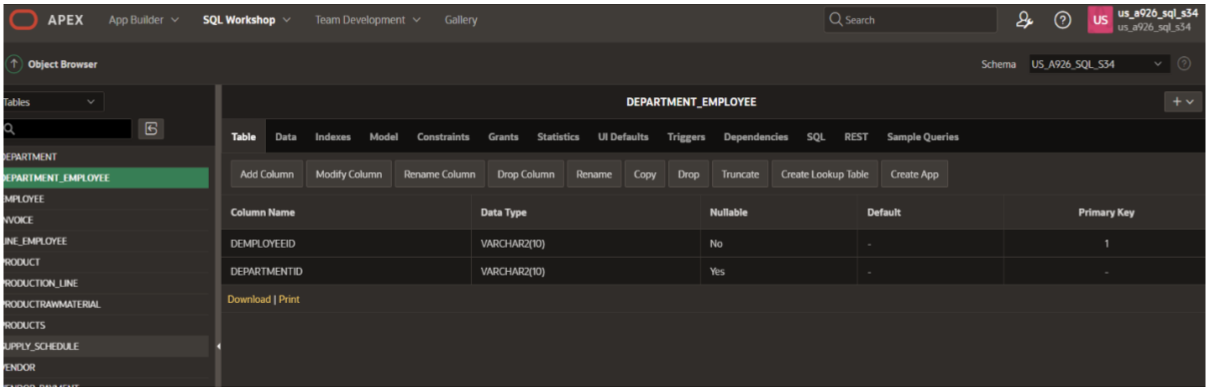
The second step is to load the appropriate data, which is a straight-forward method. Post the creation of tables, the data is loaded using the ‘Load Data’ button within each table. Following is the screenshot of the same. The data is now loaded within each table. As seen in the following screenshot, each data for each column has been correctly assigned to the right attribute/field.

The associated sql code for the tables we created are below:



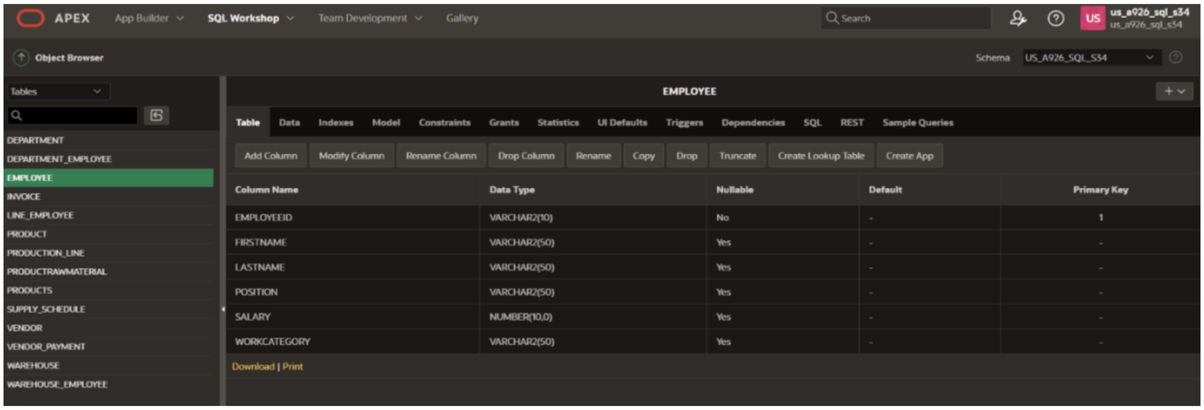
Column Name	Data Type	Nullable	Default	Primary Key
DEPARTMENTID	VARCHAR2(10)	No	-	1
DEPARTMENTNAME	VARCHAR2(50)	Yes	-	-
NUMBEROFEMPLOYEES	NUMBER(2,0)	Yes	-	-
PHONENUMBER	VARCHAR2(15)	Yes	-	-
STREETADDRESS	VARCHAR2(50)	Yes	-	-
CITYNAME	VARCHAR2(20)	Yes	-	-

Fig: DEPARTMENT



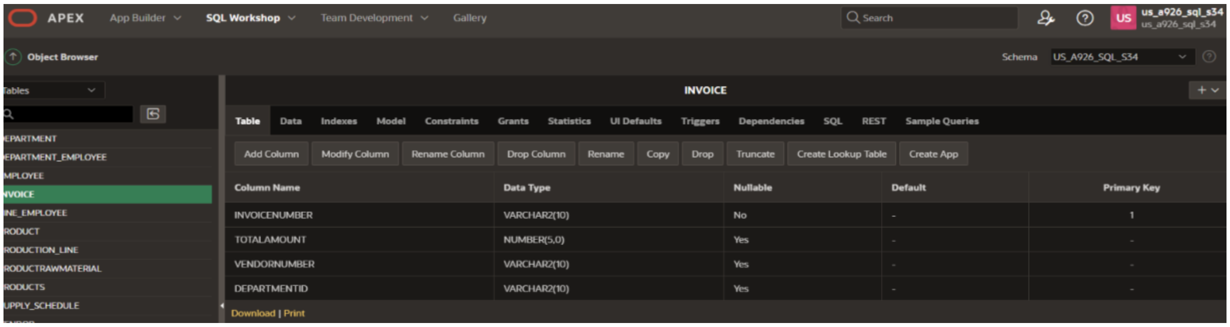
Column Name	Data Type	Nullable	Default	Primary Key
DEMPLOYEEID	VARCHAR2(10)	No	-	1
DEPARTMENTID	VARCHAR2(10)	Yes	-	-

Fig: DEPARTMENT_EMPLOYEE



Column Name	Data Type	Nullable	Default	Primary Key
EMPLOYEEID	VARCHAR2(10)	No	-	1
FIRSTNAME	VARCHAR2(50)	Yes	-	-
LASTNAME	VARCHAR2(50)	Yes	-	-
POSITION	VARCHAR2(50)	Yes	-	-
SALARY	NUMBER(10,0)	Yes	-	-
WORKCATEGORY	VARCHAR2(50)	Yes	-	-

Fig: EMPLOYEE

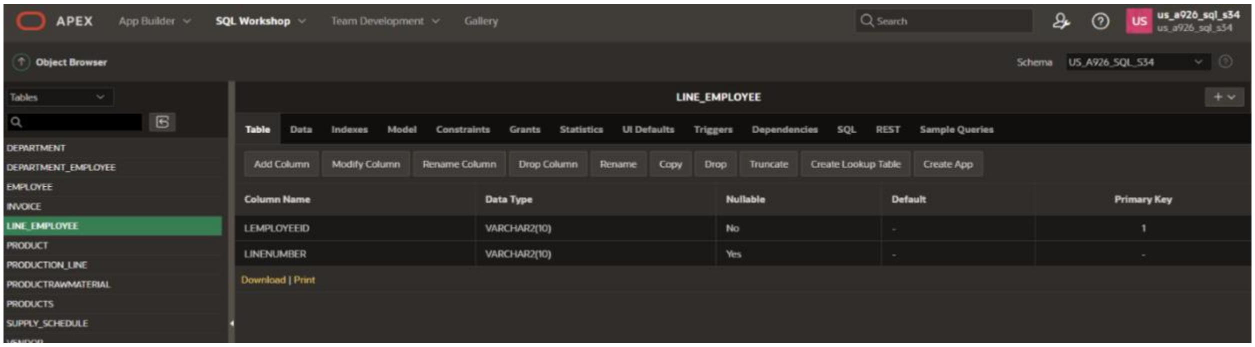


Schema US_A926_SQL_S34

INVOICE

Column Name	Data Type	Nullable	Default	Primary Key
INVOICENUMBER	VARCHAR2(10)	No	-	1
TOTALAMOUNT	NUMBER(5,0)	Yes	-	-
VENDORNUMBER	VARCHAR2(10)	Yes	-	-
DEPARTMENTID	VARCHAR2(10)	Yes	-	-

Fig: INVOICE

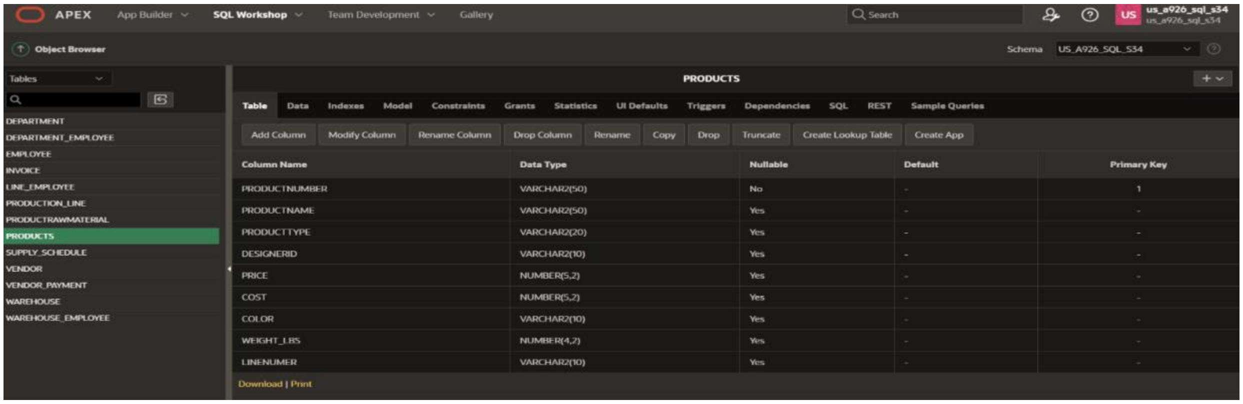


Schema US_A926_SQL_S34

LINE_EMPLOYEE

Column Name	Data Type	Nullable	Default	Primary Key
LEMPLOYEEID	VARCHAR2(10)	No	-	1
LINENUMBER	VARCHAR2(10)	Yes	-	-

Fig: LINE_EMPLOYEE

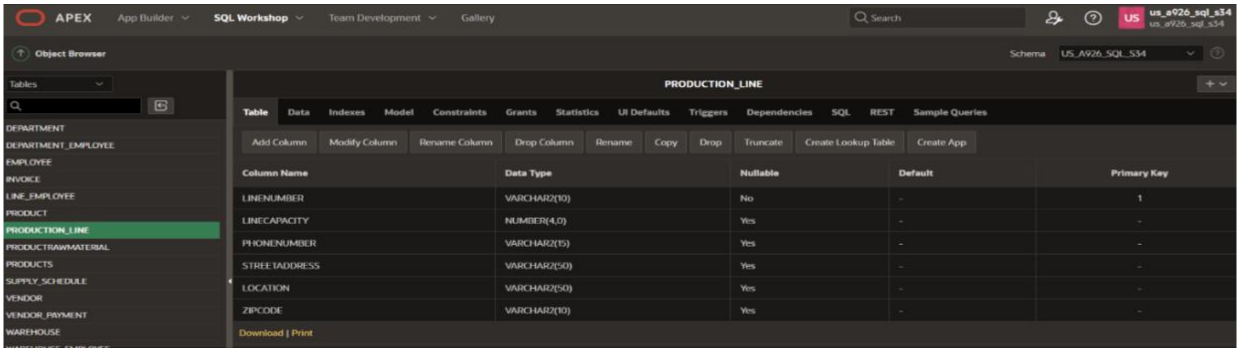


Schema US_A926_SQL_S34

PRODUCTS

Column Name	Data Type	Nullable	Default	Primary Key
PRODUCTNUMBER	VARCHAR2(50)	No	-	1
PRODUCTNAME	VARCHAR2(50)	Yes	-	-
PRODUCTTYPE	VARCHAR2(20)	Yes	-	-
DESIGNERID	VARCHAR2(10)	Yes	-	-
PRICE	NUMBER(5,2)	Yes	-	-
COST	NUMBER(5,2)	Yes	-	-
COLOR	VARCHAR2(10)	Yes	-	-
WEIGHT_LBS	NUMBER(4,2)	Yes	-	-
LINENUMER	VARCHAR2(10)	Yes	-	-

Fig: PRODUCTS

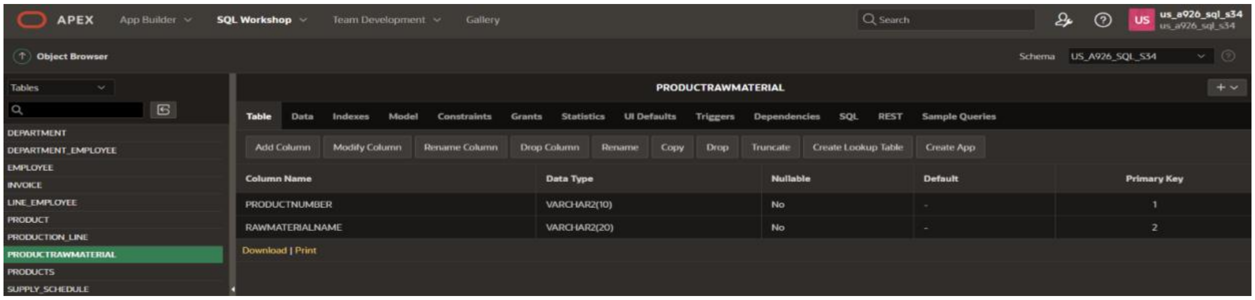


Schema US_A926_SQL_S34

PRODUCTION_LINE

Column Name	Data Type	Nullable	Default	Primary Key
LINENUMBER	VARCHAR2(10)	No	-	1
LINECAPACITY	NUMBER(4,0)	Yes	-	-
PHONENUMBER	VARCHAR2(15)	Yes	-	-
STREETADDRESS	VARCHAR2(50)	Yes	-	-
LOCATION	VARCHAR2(50)	Yes	-	-
ZIPCODE	VARCHAR2(10)	Yes	-	-

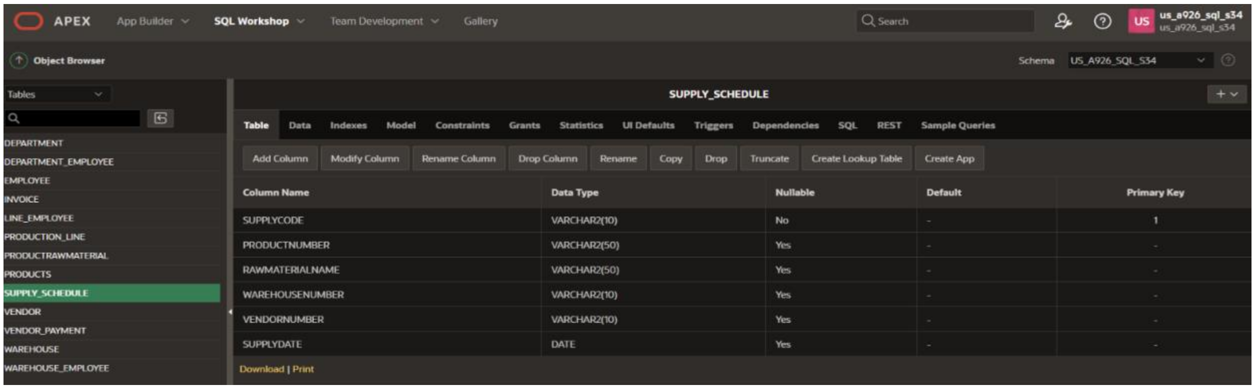
Fig: PRODUCTION_LINE



PRODUCTRAWMATERIAL

Column Name	Data Type	Nullable	Default	Primary Key
PRODUCTNUMBER	VARCHAR2(10)	No	-	1
RAWMATERIALNAME	VARCHAR2(20)	No	-	2

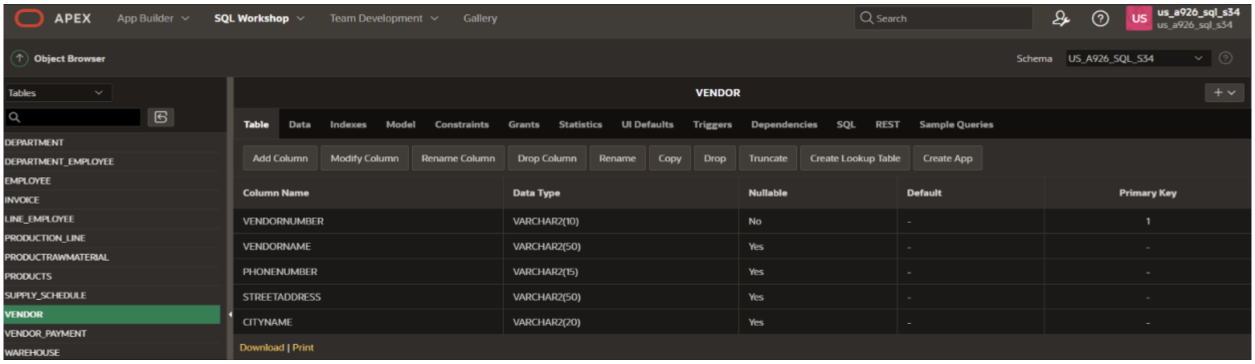
Fig: PRODUCTRAWMATERIAL



SUPPLY_SCHEDULE

Column Name	Data Type	Nullable	Default	Primary Key
SUPPLYCODE	VARCHAR2(10)	No	-	1
PRODUCTNUMBER	VARCHAR2(50)	Yes	-	-
RAWMATERIALNAME	VARCHAR2(50)	Yes	-	-
WAREHOUSENUMBER	VARCHAR2(10)	Yes	-	-
VENDORNUMBER	VARCHAR2(10)	Yes	-	-
SUPPLYDATE	DATE	Yes	-	-

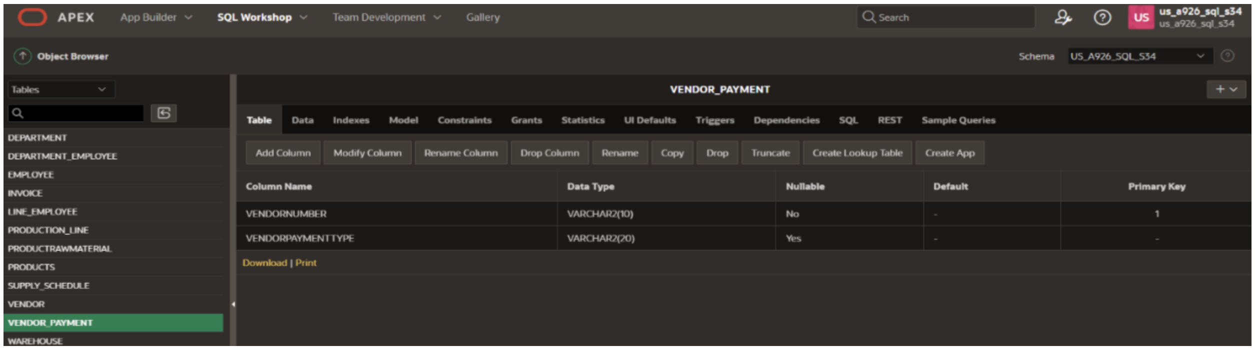
Fig: SUPPLY_SCHEDULE



VENDOR

Column Name	Data Type	Nullable	Default	Primary Key
VENDORNUMBER	VARCHAR2(10)	No	-	1
VENDORNAME	VARCHAR2(50)	Yes	-	-
PHONENUMBER	VARCHAR2(15)	Yes	-	-
STREETADDRESS	VARCHAR2(50)	Yes	-	-
CITYNAME	VARCHAR2(20)	Yes	-	-

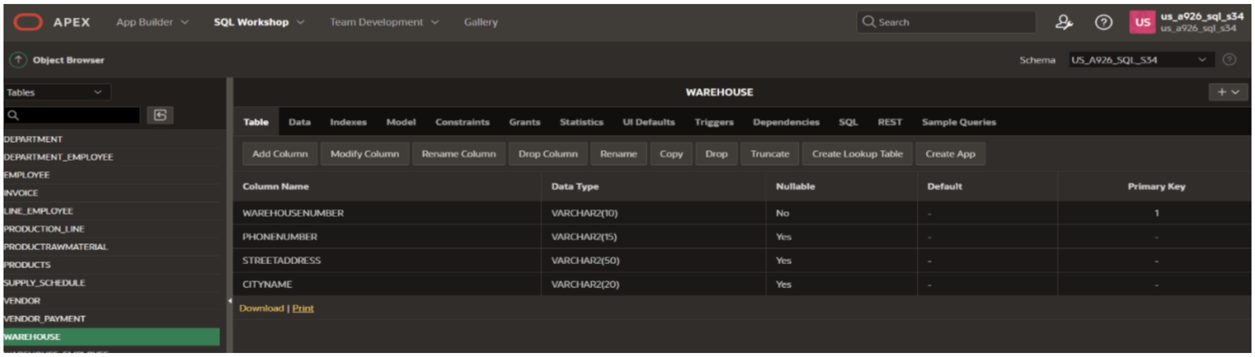
Fig: VENDOR



VENDOR_PAYMENT

Column Name	Data Type	Nullable	Default	Primary Key
VENDORNUMBER	VARCHAR2(10)	No	-	1
VENDORPAYMENTTYPE	VARCHAR2(20)	Yes	-	-

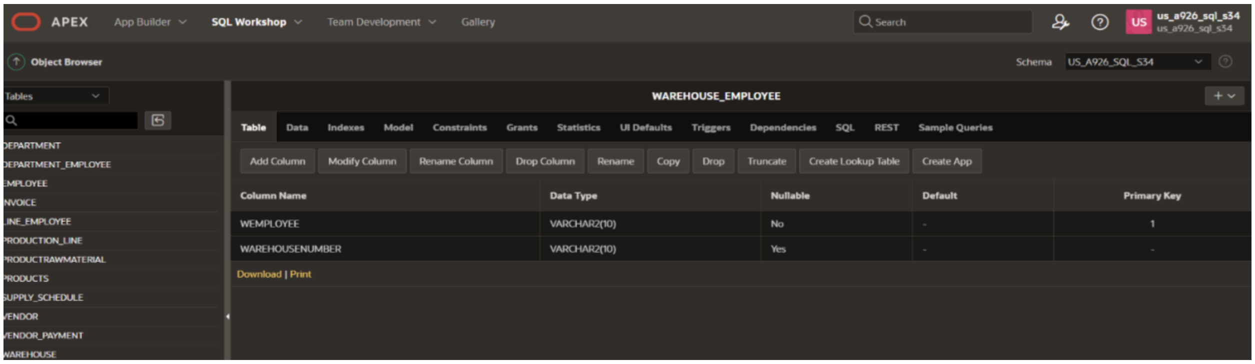
Fig: VENDOR_PAYMENT



WAREHOUSE

Column Name	Data Type	Nullable	Default	Primary Key
WAREHOUSENUMBER	VARCHAR2(10)	No	-	1
PHONENUMBER	VARCHAR2(15)	Yes	-	-
STREETADDRESS	VARCHAR2(50)	Yes	-	-
CITYNAME	VARCHAR2(20)	Yes	-	-

Fig: WAREHOUSE



WAREHOUSE_EMPLOYEE

Column Name	Data Type	Nullable	Default	Primary Key
WEMPLOYEE	VARCHAR2(10)	No	-	1
WAREHOUSENUMBER	VARCHAR2(10)	Yes	-	-

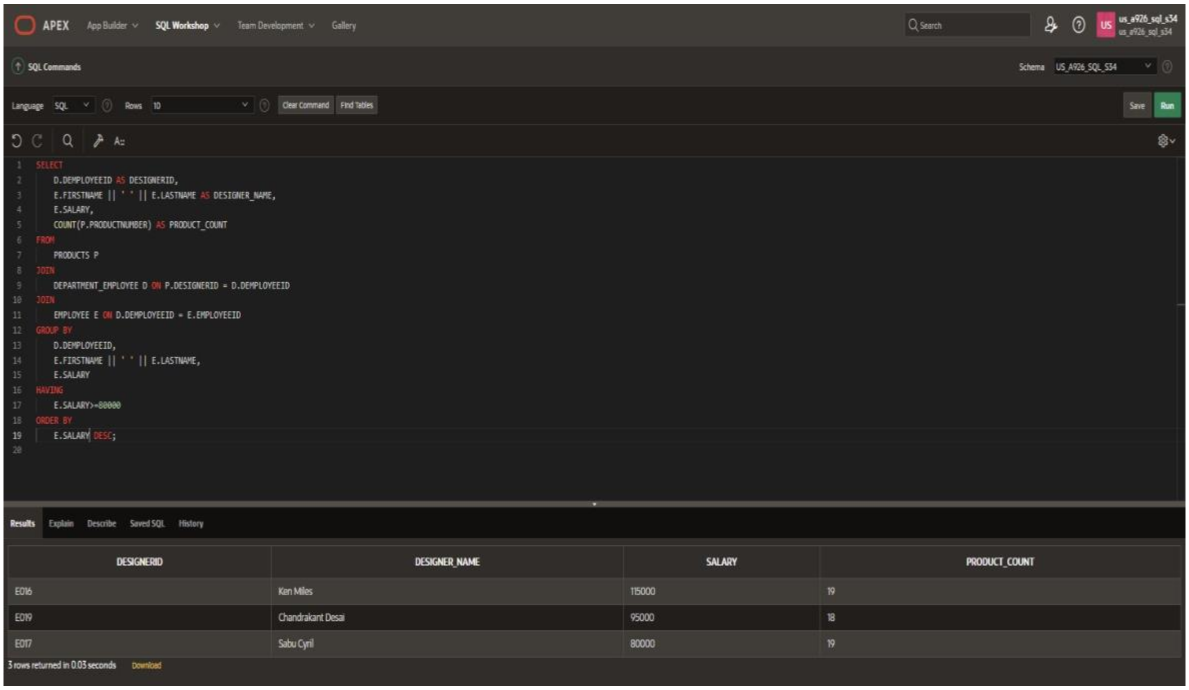
Fig: WAREHOUSE_EMPLOYEE

Following are the SQL queries developed and designed from the above tables we created.

SQL QUERY 1

1.Write a query to get the Designer information such as designer id, designer name and Product count of designers who have salary greater than equal to 80000?

The snippet shows the code and output for the query1



Schema US_A926_SQL_S34

Language SQL

Rows 10

```
SELECT
    D.DEMPLOYEEID AS DESIGNERID,
    E.FIRSTNAME || ' ' || E.LASTNAME AS DESIGNER_NAME,
    E.SALARY,
    COUNT(P.PRODUCTNUMBER) AS PRODUCT_COUNT
FROM
    PRODUCTS P
JOIN
    DEPARTMENT_EMPLOYEE D ON P.DESIGNERID = D.DEMPLOYEEID
JOIN
    EMPLOYEE E ON D.DEMPLOYEEID = E.EMPLOYEEID
GROUP BY
    D.DEMPLOYEEID,
    E.FIRSTNAME || ' ' || E.LASTNAME,
    E.SALARY
HAVING
    E.SALARY>=80000
ORDER BY
    E.SALARY DESC;
```

DESIGNERID	DESIGNER_NAME	SALARY	PRODUCT_COUNT
E016	Ken Miles	115000	19
E019	Chandrakant Desai	95000	18
E017	Sabu Cyril	80000	19

3 rows returned in 0.03 seconds

The code for the query is below:

```
SELECT

   D.DEMPLOYEEID AS DESIGNERID,

   E.FIRSTNAME || ' ' || E.LASTNAME AS DESIGNER_NAME,

   E.SALARY,

   COUNT(P.PRODUCTNUMBER) AS PRODUCT_COUNT

FROM

   PRODUCTS P

JOIN
```

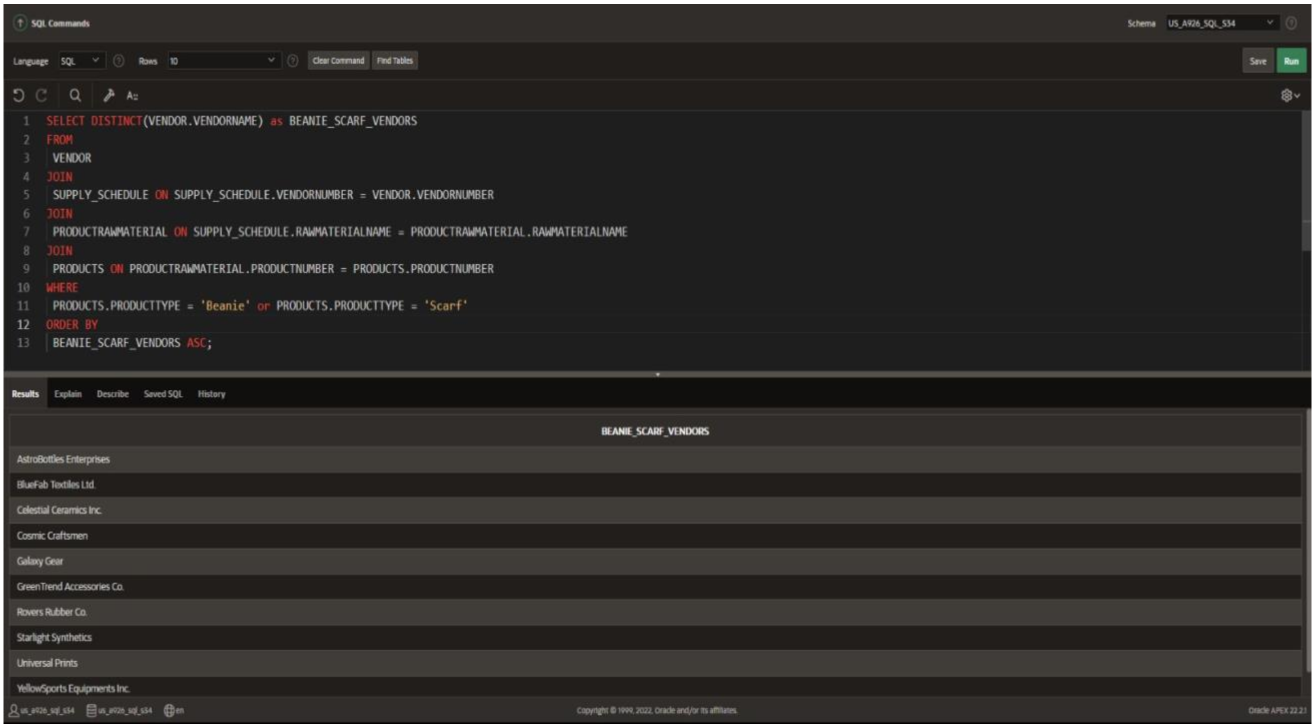
```
    DEPARTMENT_EMPLOYEE D ON P.DESIGNERID = D.DEMPLOYEEID
JOIN
    EMPLOYEE E ON D.DEMPLOYEEID = E.EMPLOYEEID
GROUP BY
    D.DEMPLOYEEID,
    E.FIRSTNAME || ' ' || E.LASTNAME,
    E.SALARY
HAVING
    E.SALARY>=80000
ORDER BY
    E.SALARY DESC;
```

This query joins the PRODUCTS, DEPARTMENT_EMPLOYEE, and EMPLOYEE tables, groups the results by designer ID and name, and then applies a HAVING clause to filter designers who have salary > 80000. The result will include the designer ID, designer name, and the count of products for each qualifying designer.

SQL QUERY 2

2.Write a query to get the vendor details who manufacture beanies or Scarfs and sort them by vendor names?

The snippet shows the code and output for the query2 which



```
SELECT DISTINCT(VENDOR.VENDORNAME) as BEANIE_SCARF_VENDORS
FROM
 VENDOR
JOIN
 SUPPLY_SCHEDULE ON SUPPLY_SCHEDULE.VENDORNUMBER = VENDOR.VENDORNUMBER
JOIN
 PRODUCTRAWMATERIAL ON SUPPLY_SCHEDULE.RAWMATERIALNAME = PRODUCTRAWMATERIAL.RAWMATERIALNAME
JOIN
 PRODUCTS ON PRODUCTRAWMATERIAL.PRODUCTNUMBER = PRODUCTS.PRODUCTNUMBER
WHERE
 PRODUCTS.PRODUCTTYPE = 'Beanie' or PRODUCTS.PRODUCTTYPE = 'Scarf'
ORDER BY
 BEANIE_SCARF_VENDORS ASC;
```

BEANIE_SCARF_VENDORS
AstroBottles Enterprises
BlueFab Textiles Ltd.
Celestial Ceramics Inc.
Cosmic Craftsmen
Galaxy Gear
GreenTrend Accessories Co.
Rovers Rubber Co.
Starlight Synthetics
Universal Prints
YellowSports Equipments Inc.

The code for the query is below:

SELECT DISTINCT(VENDOR.VENDORNAME) as BEANIE_SCARF_VENDORS

FROM

VENDOR

JOIN

SUPPLY_SCHEDULE ON SUPPLY_SCHEDULE.VENDORNUMBER = VENDOR.VENDORNUMBER

JOIN

PRODUCTRAWMATERIAL ON SUPPLY_SCHEDULE.RAWMATERIALNAME = PRODUCTRAWMATERIAL.RAWMATERIALNAME

JOIN

PRODUCTS ON PRODUCTRAWMATERIAL.PRODUCTNUMBER = PRODUCTS.PRODUCTNUMBER

WHERE

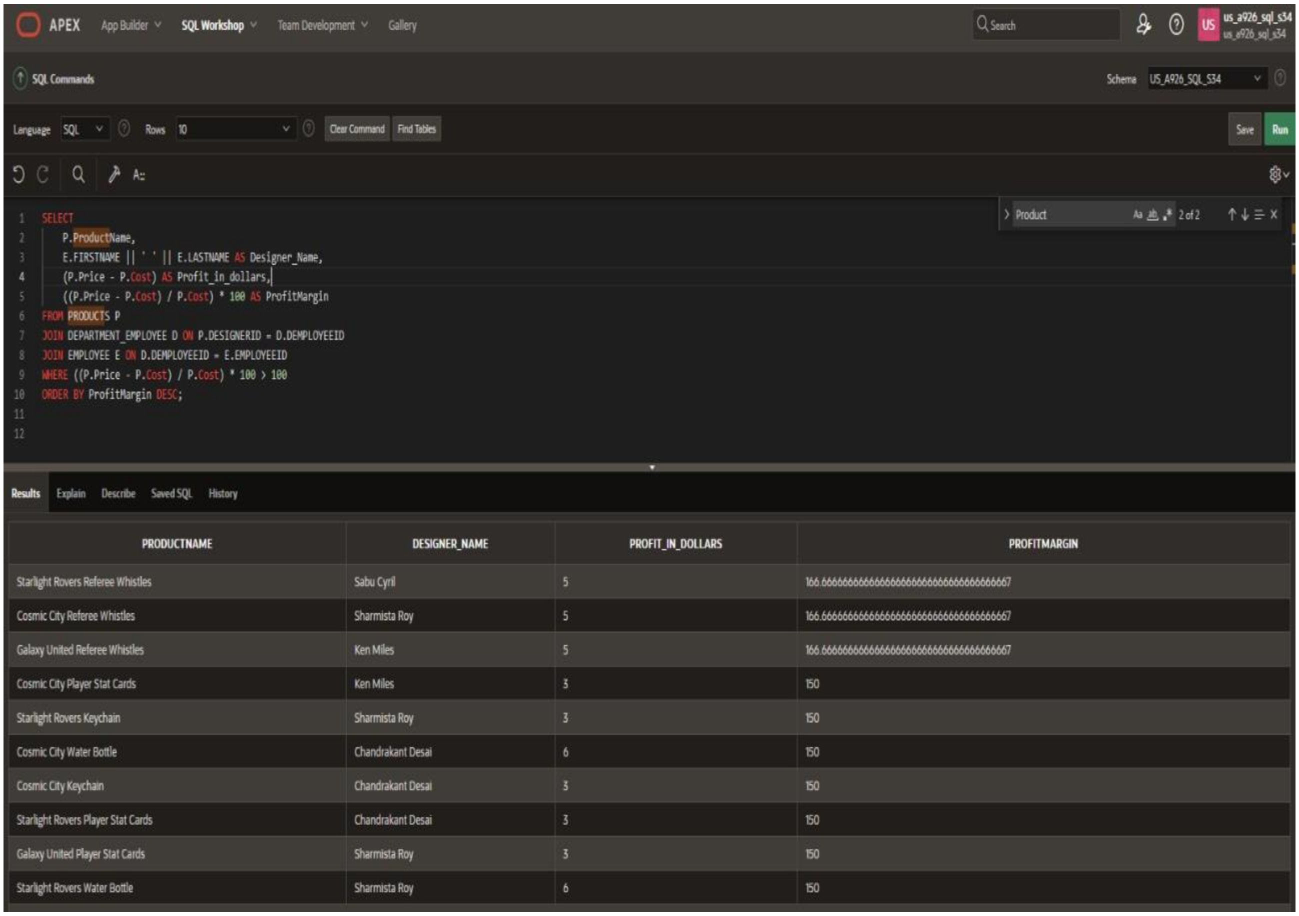
```
PRODUCTS.PRODUCTTYPE = 'Beanie' or PRODUCTS.PRODUCTTYPE = 'Scarf'

ORDER BY

BEANIE_SCARF_VENDORS ASC;
```

The SQL query selects unique vendor names linked to 'Beanie' or 'Scarf' products from interconnected tables, sorting them alphabetically.

SQL QUERY 3

3.Find the products with profit margins greater than 100 % and their designers ?

The snippet shows the code and output for the query 3



```
SELECT
    P.ProductName,
    E.FIRSTNAME || ' ' || E.LASTNAME AS Designer_Name,
    (P.Price - P.Cost) AS Profit_in_dollars,
    ((P.Price - P.Cost) / P.Cost) * 100 AS ProfitMargin
FROM PRODUCTS P
JOIN DEPARTMENT_EMPLOYEE D ON P.DESIGNERID = D.DEMPLOYEEID
JOIN EMPLOYEE E ON D.DEMPLOYEEID = E.EMPLOYEEID
WHERE ((P.Price - P.Cost) / P.Cost) * 100 > 100
ORDER BY ProfitMargin DESC;
```

PRODUCTNAME	DESIGNER_NAME	PROFIT_IN_DOLLARS	PROFITMARGIN
Starlight Rovers Referee Whistles	Sabu Cyril	5	166.6666666666666666666666666666666666667
Cosmic City Referee Whistles	Sharmista Roy	5	166.6666666666666666666666666666666666667
Galaxy United Referee Whistles	Ken Miles	5	166.6666666666666666666666666666666666667
Cosmic City Player Stat Cards	Ken Miles	3	150
Starlight Rovers Keychain	Sharmista Roy	3	150
Cosmic City Water Bottle	Chandrakant Desai	6	150
Cosmic City Keychain	Chandrakant Desai	3	150
Starlight Rovers Player Stat Cards	Chandrakant Desai	3	150
Galaxy United Player Stat Cards	Sharmista Roy	3	150
Starlight Rovers Water Bottle	Sharmista Roy	6	150

The code for the query is below

```
SELECT
    P.ProductName,
    E.FIRSTNAME || ' ' || E.LASTNAME AS Designer_Name,
    (P.Price - P.Cost) AS Profit_in_dollars,
    ((P.Price - P.Cost) / P.Cost) * 100 AS ProfitMargin
FROM PRODUCTS P
JOIN DEPARTMENT_EMPLOYEE D ON P.DESIGNERID = D.DEMPLOYEEID
JOIN EMPLOYEE E ON D.DEMPLOYEEID = E.EMPLOYEEID
WHERE ((P.Price - P.Cost) / P.Cost) * 100 > 100
ORDER BY ProfitMargin DESC;
```

The SQL query retrieves product information, including product name, designer name, profit in dollars, and profit margin for products designed by employees whose profit margin is greater than 100%, ordered by profit margin in descending order.

--------x------------