REPORT

Global Solution Package (Pvt.) Ltd

Module: Advanced Database Management Systems

Course: Management Information Systems (Special)

Batch : 19.1 UGC

Team Members

Name	St.ID	Email
H.M.R.K. Herath	21019019	hmrkherath@students.nsbm.lk
C.M.D. Liyanage	23006035	cbdliyanage@students.nsbm.lk
I.P.R. Kumarasinghe	21017868	iprkumarasinghe@students.nsbm.lk
P.W. Hettiarachchi	21010139	pwheettiarachchi@students.nsbm.lk
P.G.I.D. Gallage	22018035	pgidgallage@students.nsbm.lk

Content **Page no.**

INTRODUCTION TO THE COMPANY

Overview to company

Company name :	Global Packaging Solution
Company category :	Polythene Manufacturing Factory
Location :	Borupana, Dehiwala
Site visited :	2020.12.02

Introduction to the Research

The research project was carried out with the industrial collaboration of Global Packaging Solution. The company that sites a polythene manufacturing process which provides a greater contribution to the national income growth by providing high quality plastic products to Sri Lankan market. Company assets their own manufacturing factory, two warehouses and one sales office which are located at Borupana and Dehiwala.

Proposed User Scenario

When a customer gives an order, the administrative department checks for the availability of raw materials and approve the order. Then according to the user requirements, the product is manufactured. The manufacturing process is completed at the factory located in Borupana which also located with the main warehouse at the same premises.

The raw materials needed to manufacture polythene is comes as tiny beads. First, they mix the raw materials to get the polythene pulp using the mixing machine and then that pulp goes through a film blowing machine which can convert the polythene pulp into a polythene roll.

These rolls are going through a cutting machine so that the handle cut is removed at the moment. This handle cut is considered as recycling waste and convert to raw materials again using special kind of machine.

The polythene bag which removed the handle cut is the pass to a separate machine to attach the gusset and then pass to the next level to the printing machine. Here they print what the user expect to have in the order.

The final process is quality checking and packaging. Here the employees check whether the end products are manufactured with enough quality and remove the exceptions to recycle again. The polybags that ready to sell are well packed and stored in the warehouse that located at the same premises.

The packages that need to sell are transported to the warehouse that located in Dehiwala where they have their Sales Outlet.

Here the Sales department check for the date to hand over the finished order and contact the customer to visit the store. Then the customer can have their order right after he/she made the full payment. Then a bill is generated and given to the customer and the sales department updates the order details in their records.

For the supplier side, the Global Packaging Solution have both international and local suppliers itself to get raw materials. When they run out of raw materials the administrative department contacts the supplier and makes an order for the raw materials make the payment and generates the invoice. These details are stored in the records.

When the raw materials arrived at the factory location, they are stored in the warehouse which is located at the same location.

Sales department, Production, HR department, Administrative department, and are the Warehouse are basic types of departments managed under the company. All the departments are managed by a manager and every manger is specific to that department. Every employee has their own records that are stored in order to manage their salary and workload.

At the current state company handles the accounts and all other monitory values using a software called Quick Labs but due to increase of employees, orders, and management issues they expect to have a system which supports their business and to expand it.

Company Processes

- Film Blowing
- Cutting
- Printing
- Quality Checking
- Packing

Raw Material Categories

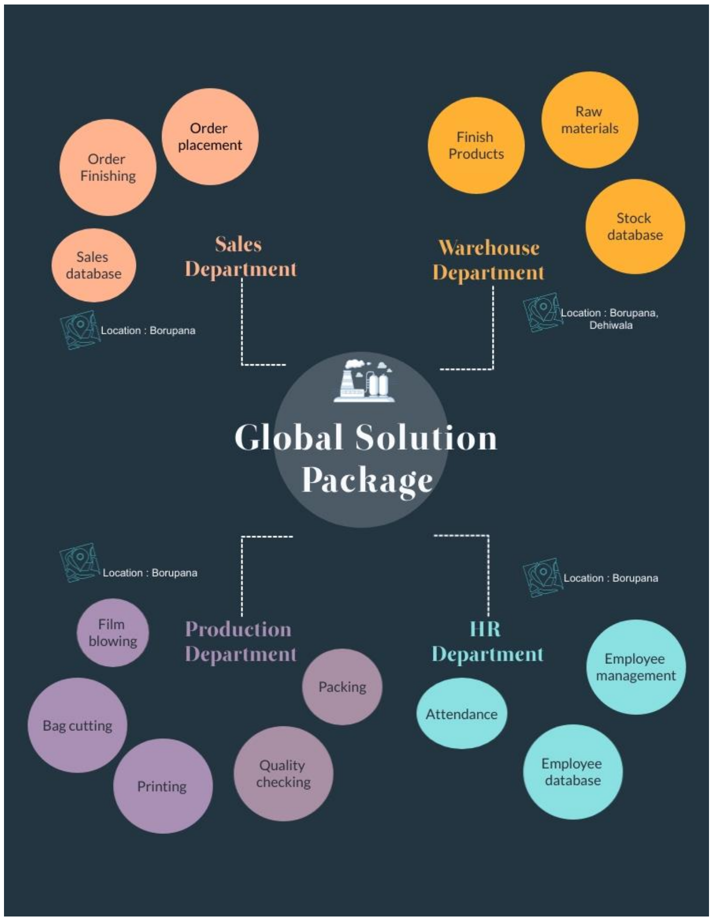
- High Density Polythene (HDPE)
- Low Density Polythene (LDP)
- Polypropylene (PP)

Types of Departments in the company

- Production Department
- Sales Department
- Administration Department
- HR Department

Product Categories	
Product	**Sizes**
Chicken bag	177+100x330 mm
Small shopping bag	250+125x415 mm
Large shopping bag	278+127x508 mm
Jumbo bag	445+190x762 mm
Small grocery bag	203x127 mm
Large grocery bag	178x54 mm

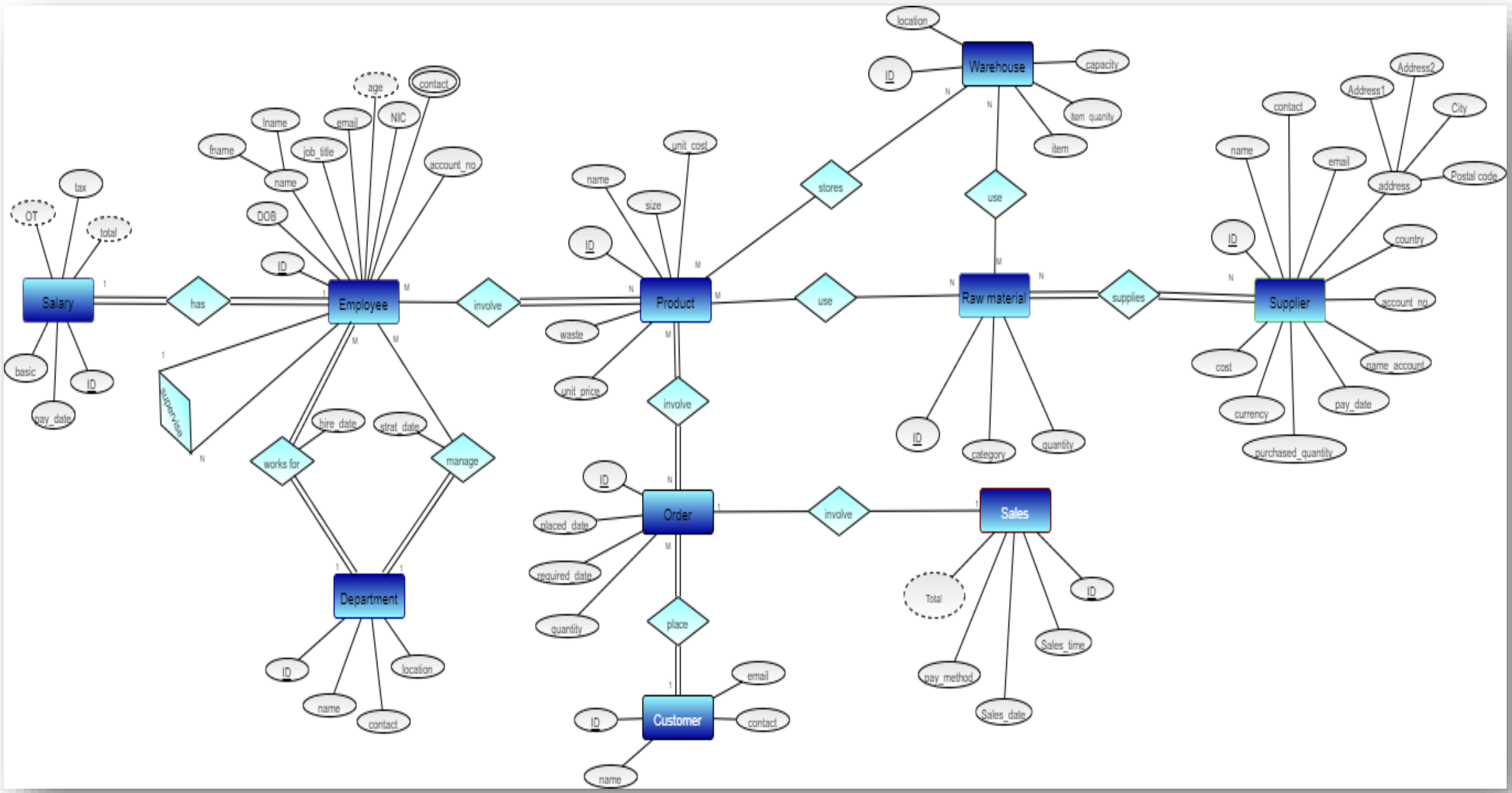
Company Overview



Used tool: Free Concept Map Maker - Create Concept Maps Online | Visme

EXTENDED ENTITY RELATIONSHIP DIAGRAM

: Link to ER diagram (.txt) and editable vector image(.svg) also attached



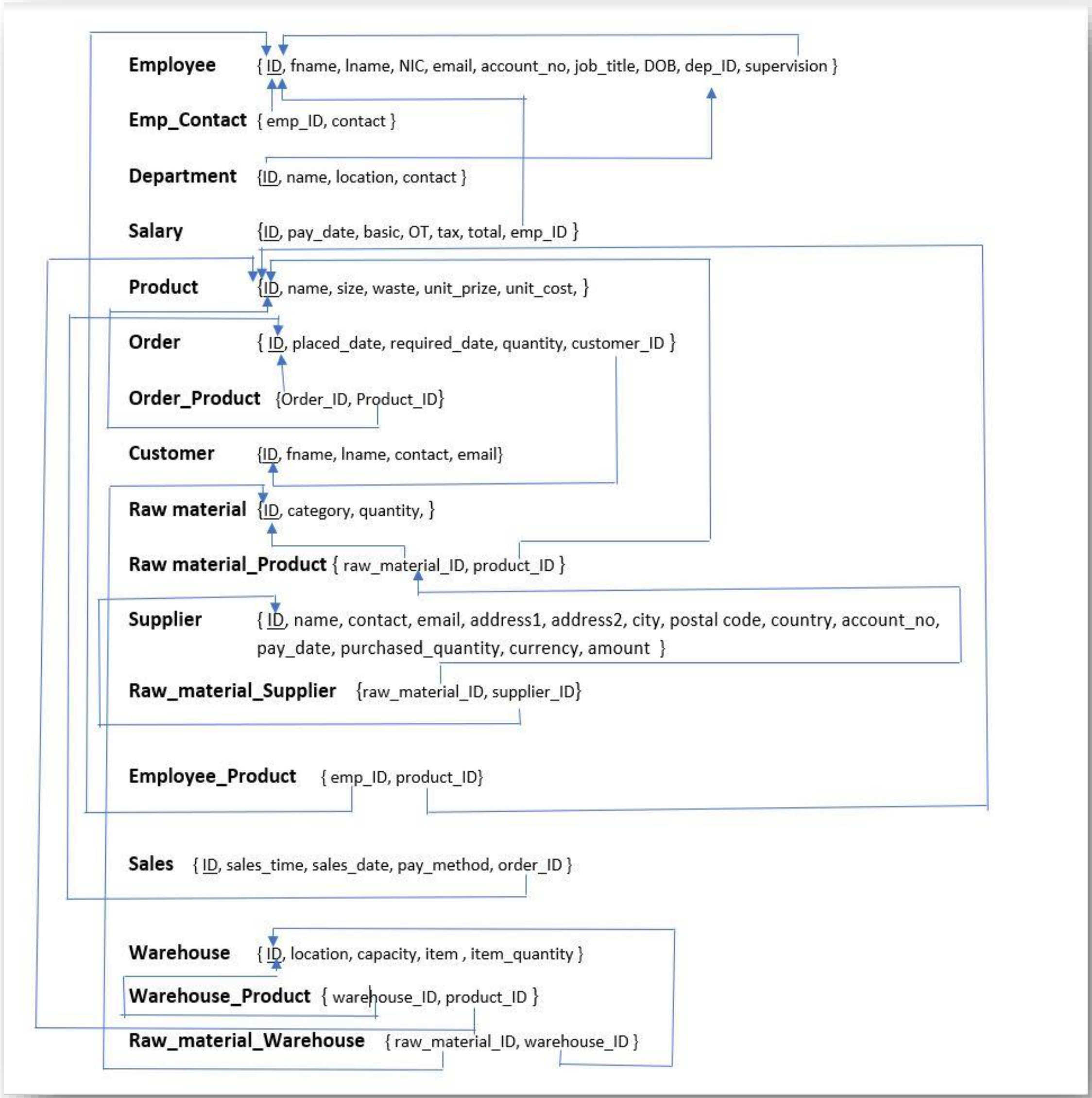
Tools used: Draw.io / diagrams.net

ADDITIONAL ASSUMPTIONS

- All employees have their bank account in the same bank.
- Customers are not directly connect with employees. Automated system provides filling forms and everything.
- Warehouse keeps both finish products and raw materials. But only finished products quantity is considered as items.
- Customer needs to place an order before purchase the product.

- Every customer, and department have one contact number for each. Not consider as a multivalued attribute.
- Customer name is identified as the company which gives the order.
- If customer needs their order to be ready at separate dates, the system generates separate dates for the given dates.

RELATIONAL SCHEMA



Employee { ID, fname, lname, NIC, email, account_no, job_title, DOB, dep_ID, supervision }

Emp_Contact { emp_ID, contact }

Department {ID, name, location, contact }

Salary {ID, pay_date, basic, OT, tax, total, emp_ID }

Product {ID, name, size, waste, unit_prize, unit_cost, }

Order { ID, placed_date, required_date, quantity, customer_ID }

Order_Product {Order_ID, Product_ID}

Customer {ID, fname, lname, contact, email}

Raw material {ID, category, quantity, }

Raw material_Product { raw_material_ID, product_ID }

Supplier { ID, name, contact, email, address1, address2, city, postal code, country, account_no, pay_date, purchased_quantity, currency, amount }

Raw_material_Supplier {raw_material_ID, supplier_ID}

Employee_Product { emp_ID, product_ID}

Sales { ID, sales_time, sales_date, pay_method, order_ID }

Warehouse { ID, location, capacity, item , item_quantity }

Warehouse_Product { warehouse_ID, product_ID }

Raw_material_Warehouse { raw_material_ID, warehouse_ID }

NORMALIZATION

Employee table

Emp_ID	FName	Lname	Job title	Email	NIC	Gender	Acc_no	DOB	Salary_ID	Dep_ID

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not any transitive dependencies

Emp_Contact table

Emp_ID	Contact_no

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not any transitive dependencies

Salary table

Salary id	Salary category	Date	Basic	OT	Tax	Total salary

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is not in 3NF ,because it has transitive dependency. Salary category is a non-prime attribute. Basic salary depends on salary category. Because of that we can take salary category and basic columns to another separate table.

Salarycategory table

Salary category ID	Salary category	Basic

Salary table

Salary ID	Date	OT	Tax	Total salary	Salary category id

Department table

Dep_number	Name	Location	Contact _No

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not any transitive dependencies

Supplier table

id	Fname	Lname	Contact no	email	Address 1	Address 2	city	Postal code	country	Acc	bank	Pay-no	Pay _date	Purchased Qty	currency	amount

This table is in 1NF because it has This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is not in 2NF.because it has partial dependencies

Supplier_id, pay_no $\longrightarrow$ purchased_date, Purchased_qty, currency, cost

Supplier Id $\longrightarrow$ Fname, Lname, contact_no, email,Adreess1,Address2,City,Postal code, country, Account_no

Sup_id	Fname	Lname	Contact_no	email	Adrees1	Address2	city	Pos_code	Country	Acc_no

Sup_id	Pay_no	Purchased date	Purchased_qty	Currency	cost

This table is not in 3NF ,because Postal code depends on Country

Postalcode_Country table

Postal code	country

Sup_id	Fname	Lname	Contact no	email	Address 1	Address 2	city	country	Acc_no	Postal code

Sup_id	Pay_no	Purchased date	Purchased_qty	currency	cost

Customer table

Customer_ID	name	Contact no	E-mail	Sale_id

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not any transitive dependencies

Order table

Order_ID	Placed date	Required date	quantity	Customer_Id

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not any transitive dependencies

Product table

ID	Name	size	Unit price	Unit cost	waste

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not transitive dependencies

Order -product table

Order_id	Product_id

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not transitive dependencies

Sales Table

Sale_id	Sales_date	Sale time	Pay_method	Order_id	Paid_amount	discount

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is not in 3NF because it has transitive dependencies

Sale id	Sales date	Sale item	Pay method	Oder id	Paid-amount id

Paid_amount_id	Paid amount	discount

Raw material table

Raw_material_id	category	quantity

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not transitive dependencies

Supplier_raw material table

Supplier_id	Raw material ID

This table is in 1NF because it has not any multivalued attributes or nester relationships

This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not transitive dependencies

Warehouse table

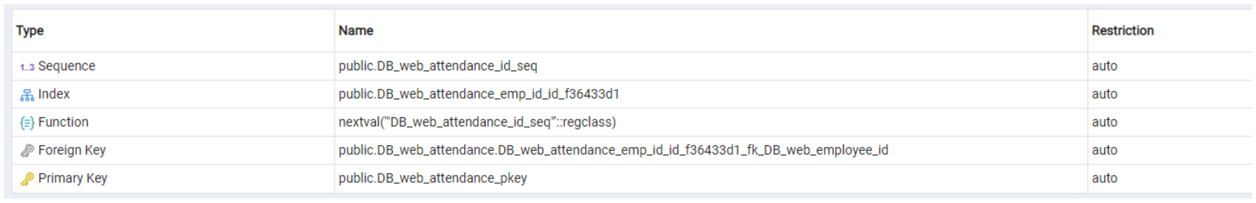
Warehouse_id	location	capacity	Item-quantity	Item_name

This table is in 1NF because it has not any multivalued attributes or nester relationships

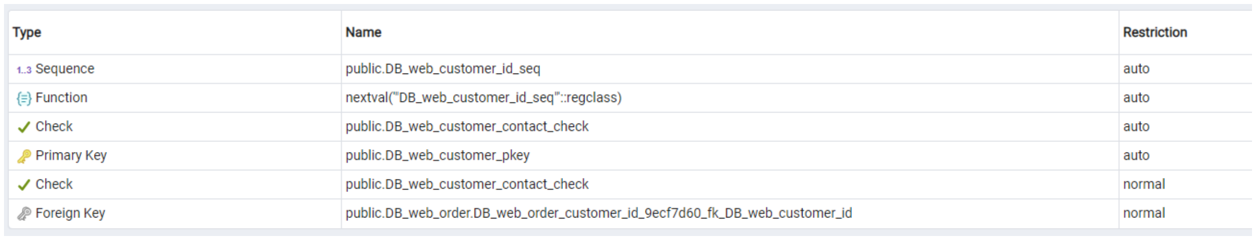
This table is in 2NF because it has not any partial dependencies.

This table is in 3NF because it has not transitive dependencies

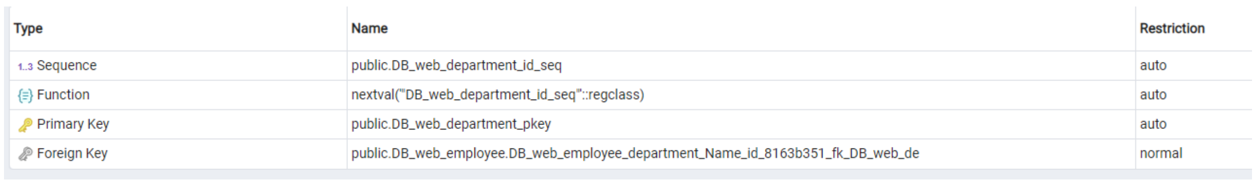
Database Dictionary screen shots



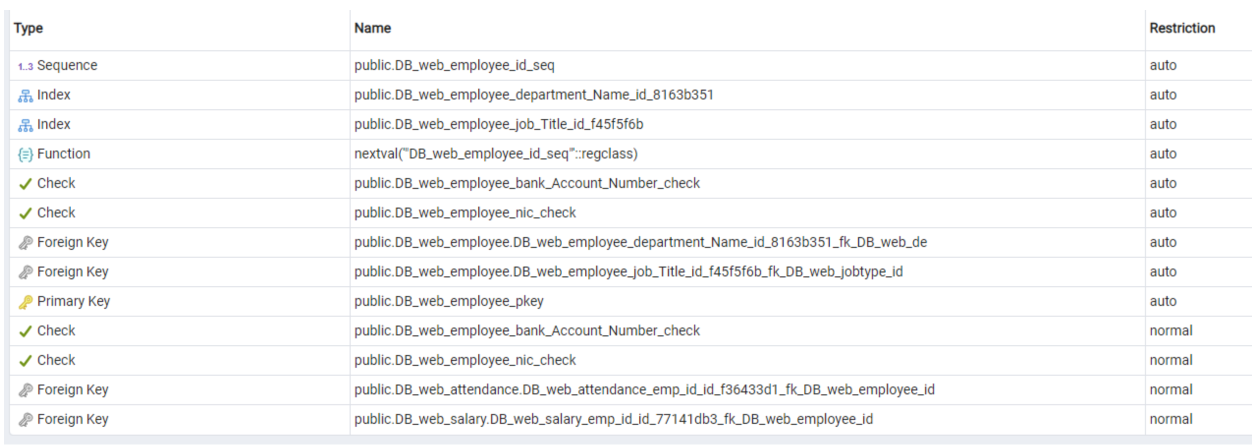
Type	Name	Restriction
Sequence	public.DB_web_attendance_id_seq	auto
Index	public.DB_web_attendance_emp_id_id_f36433d1	auto
Function	nextval('"DB_web_attendance_id_seq"'::regclass)	auto
Foreign Key	public.DB_web_attendance.DB_web_attendance_emp_id_id_f36433d1_fk_DB_web_employee_id	auto
Primary Key	public.DB_web_attendance_pkey	auto



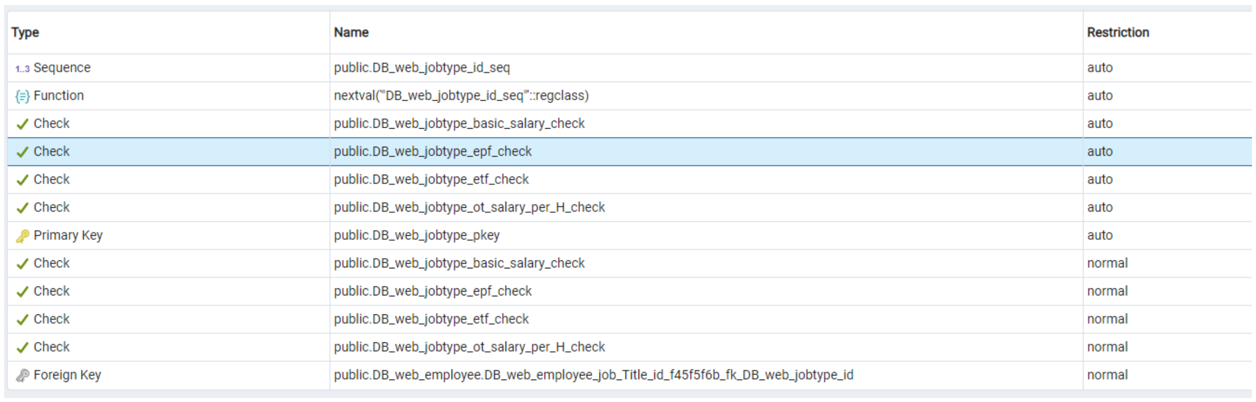
Type	Name	Restriction
Sequence	public.DB_web_customer_id_seq	auto
Function	nextval('"DB_web_customer_id_seq"'::regclass)	auto
✓ Check	public.DB_web_customer_contact_check	auto
Primary Key	public.DB_web_customer_pkey	auto
✓ Check	public.DB_web_customer_contact_check	normal
Foreign Key	public.DB_web_order.DB_web_order_customer_id_9ecf7d60_fk_DB_web_customer_id	normal



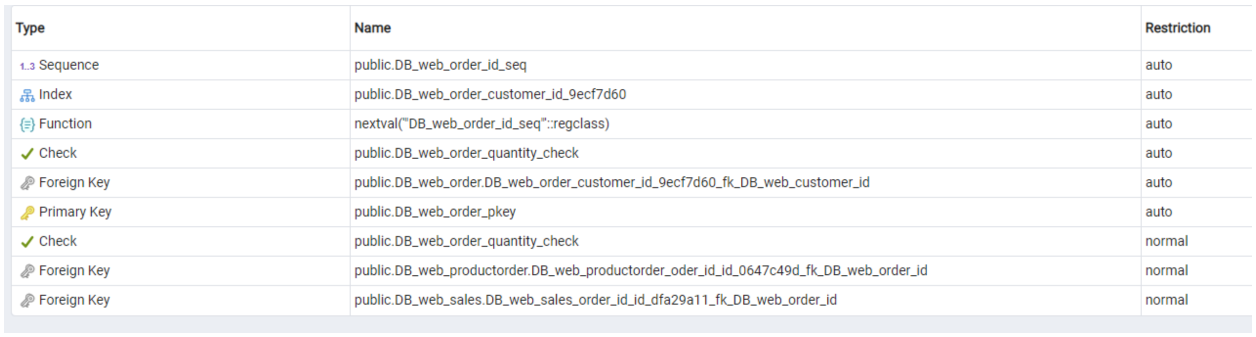
Type	Name	Restriction
Sequence	public.DB_web_department_id_seq	auto
Function	nextval('"DB_web_department_id_seq"'::regclass)	auto
Primary Key	public.DB_web_department_pkey	auto
Foreign Key	public.DB_web_employee.DB_web_employee_department_Name_id_8163b351_fk_DB_web_de	normal



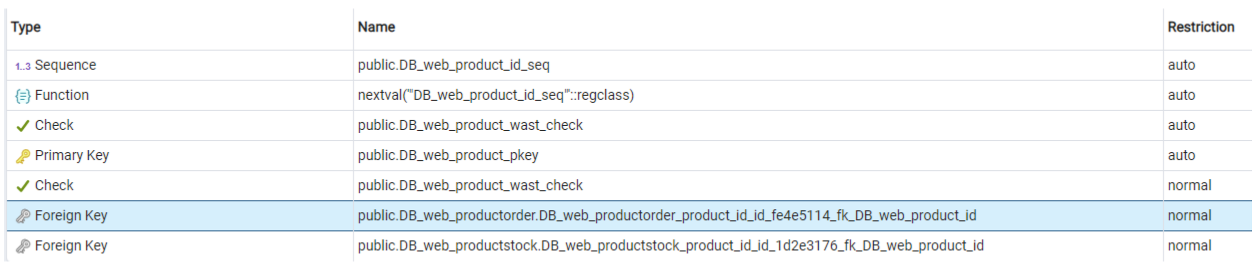
Type	Name	Restriction
Sequence	public.DB_web_employee_id_seq	auto
Index	public.DB_web_employee_department_Name_id_8163b351	auto
Index	public.DB_web_employee_job_Title_id_f45f5f6b	auto
Function	nextval('"DB_web_employee_id_seq"'::regclass)	auto
✓ Check	public.DB_web_employee_bank_Account_Number_check	auto
✓ Check	public.DB_web_employee_nic_check	auto
Foreign Key	public.DB_web_employee.DB_web_employee_department_Name_id_8163b351_fk_DB_web_de	auto
Foreign Key	public.DB_web_employee.DB_web_employee_job_Title_id_f45f5f6b_fk_DB_web_jobtype_id	auto
Primary Key	public.DB_web_employee_pkey	auto
✓ Check	public.DB_web_employee_bank_Account_Number_check	normal
✓ Check	public.DB_web_employee_nic_check	normal
Foreign Key	public.DB_web_attendance.DB_web_attendance_emp_id_id_f36433d1_fk_DB_web_employee_id	normal
Foreign Key	public.DB_web_salary.DB_web_salary_emp_id_id_77141db3_fk_DB_web_employee_id	normal



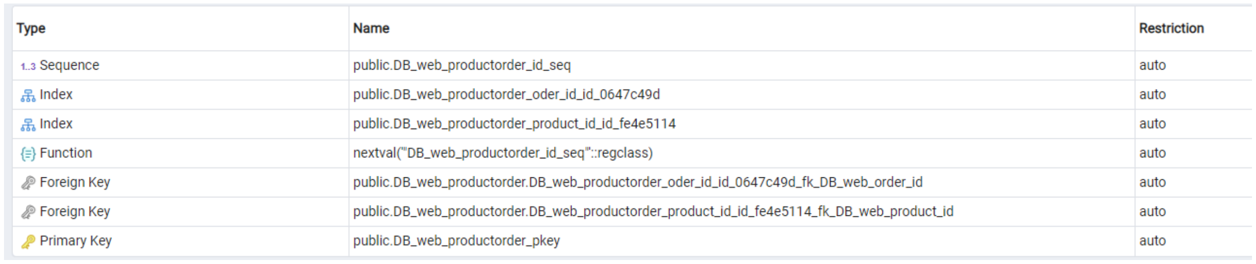
Type	Name	Restriction
Sequence	public.DB_web_jobtype_id_seq	auto
Function	nextval('"DB_web_jobtype_id_seq"'::regclass)	auto
✓ Check	public.DB_web_jobtype_basic_salary_check	auto
✓ Check	public.DB_web_jobtype_epf_check	auto
✓ Check	public.DB_web_jobtype_etf_check	auto
✓ Check	public.DB_web_jobtype_ot_salary_per_H_check	auto
Primary Key	public.DB_web_jobtype_pkey	auto
✓ Check	public.DB_web_jobtype_basic_salary_check	normal
✓ Check	public.DB_web_jobtype_epf_check	normal
✓ Check	public.DB_web_jobtype_etf_check	normal
✓ Check	public.DB_web_jobtype_ot_salary_per_H_check	normal
Foreign Key	public.DB_web_employee.DB_web_employee_job_Title_id_f45f5f6b_fk_DB_web_jobtype_id	normal



Type	Name	Restriction
Sequence	public.DB_web_order_id_seq	auto
Index	public.DB_web_order_customer_id_9ecf7d60	auto
Function	nextval('"DB_web_order_id_seq"'::regclass)	auto
Check	public.DB_web_order_quantity_check	auto
Foreign Key	public.DB_web_order.DB_web_order_customer_id_9ecf7d60_fk_DB_web_customer_id	auto
Primary Key	public.DB_web_order_pkey	auto
Check	public.DB_web_order_quantity_check	normal
Foreign Key	public.DB_web_productorder.DB_web_productorder_oder_id_id_0647c49d_fk_DB_web_order_id	normal
Foreign Key	public.DB_web_sales.DB_web_sales_order_id_id_dfa29a11_fk_DB_web_order_id	normal

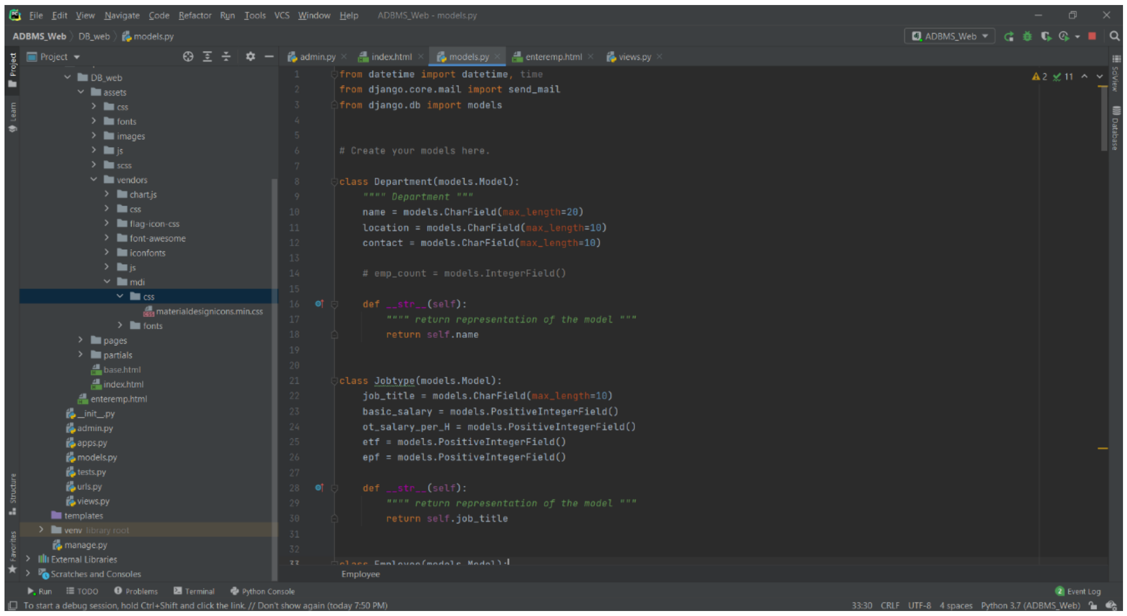


Type	Name	Restriction
Sequence	public.DB_web_product_id_seq	auto
Function	nextval('"DB_web_product_id_seq"'::regclass)	auto
Check	public.DB_web_product_wast_check	auto
Primary Key	public.DB_web_product_pkey	auto
Check	public.DB_web_product_wast_check	normal
Foreign Key	public.DB_web_productorder.DB_web_productorder_product_id_id_fe4e5114_fk_DB_web_product_id	normal
Foreign Key	public.DB_web_productstock.DB_web_productstock_product_id_id_1d2e3176_fk_DB_web_product_id	normal



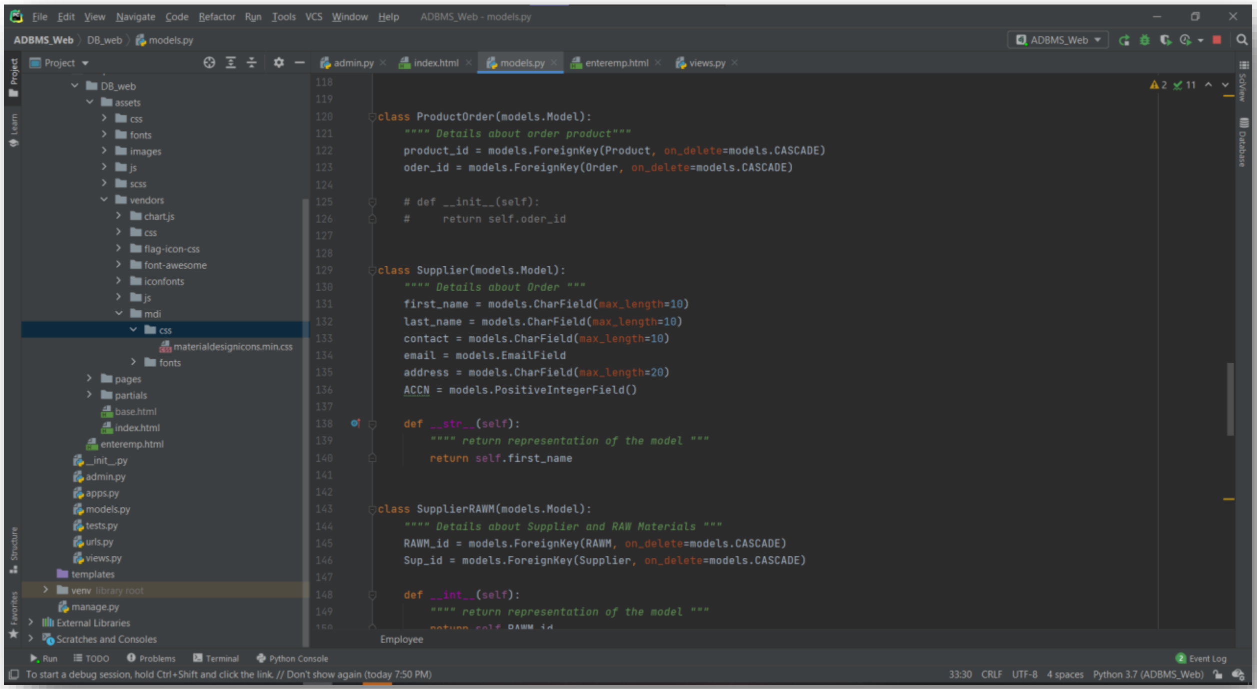
Type	Name	Restriction
Sequence	public.DB_web_productorder_id_seq	auto
Index	public.DB_web_productorder_oder_id_id_0647c49d	auto
Index	public.DB_web_productorder_product_id_id_fe4e5114	auto
Function	nextval('"DB_web_productorder_id_seq"'::regclass)	auto
Foreign Key	public.DB_web_productorder.DB_web_productorder_oder_id_id_0647c49d_fk_DB_web_order_id	auto
Foreign Key	public.DB_web_productorder.DB_web_productorder_product_id_id_fe4e5114_fk_DB_web_product_id	auto
Primary Key	public.DB_web_productorder_pkey	auto

Create Table Statements screen shots

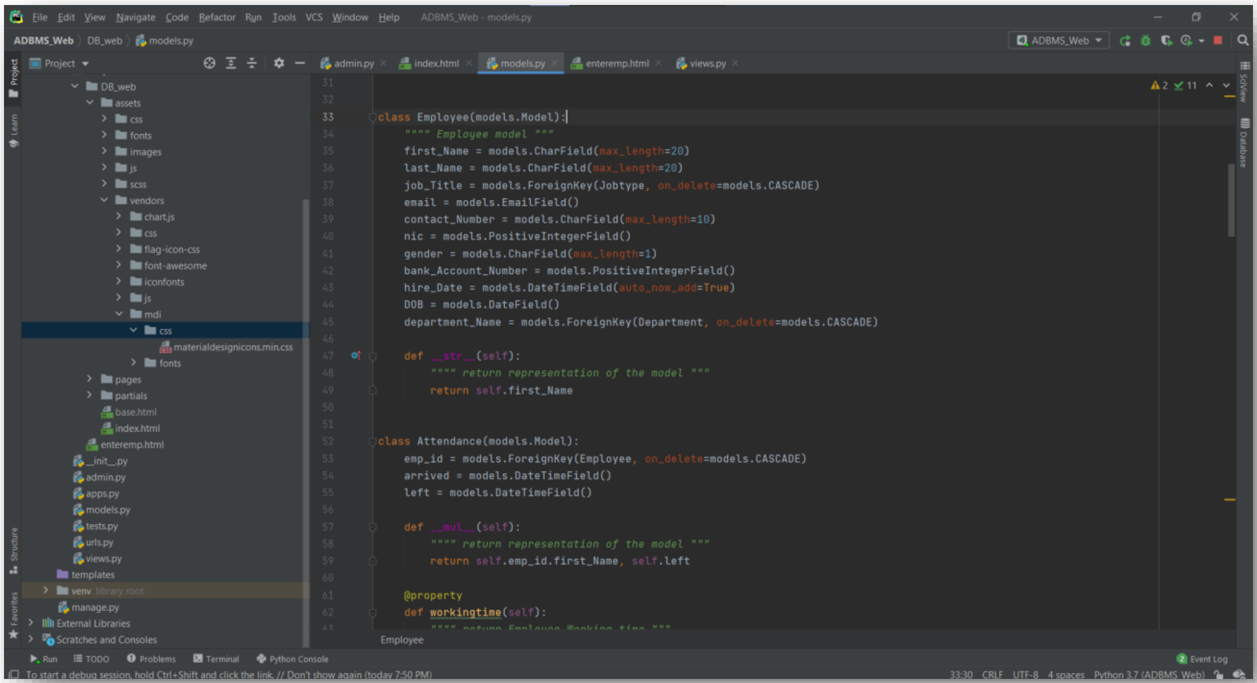


```python
from datetime import datetime, time
from django.core.mail import send_mail
from django.db import models


# Create your models here.

class Department(models.Model):
    """" Department """
    name = models.CharField(max_length=20)
    location = models.CharField(max_length=10)
    contact = models.CharField(max_length=10)

    # emp_count = models.IntegerField()

    def __str__(self):
        """" return representation of the model """
        return self.name


class Jobtype(models.Model):
    job_title = models.CharField(max_length=10)
    basic_salary = models.PositiveIntegerField()
    ot_salary_per_H = models.PositiveIntegerField()
    etf = models.PositiveIntegerField()
    epf = models.PositiveIntegerField()

    def __str__(self):
        """" return representation of the model """
        return self.job_title


class Employee(models.Model):
```

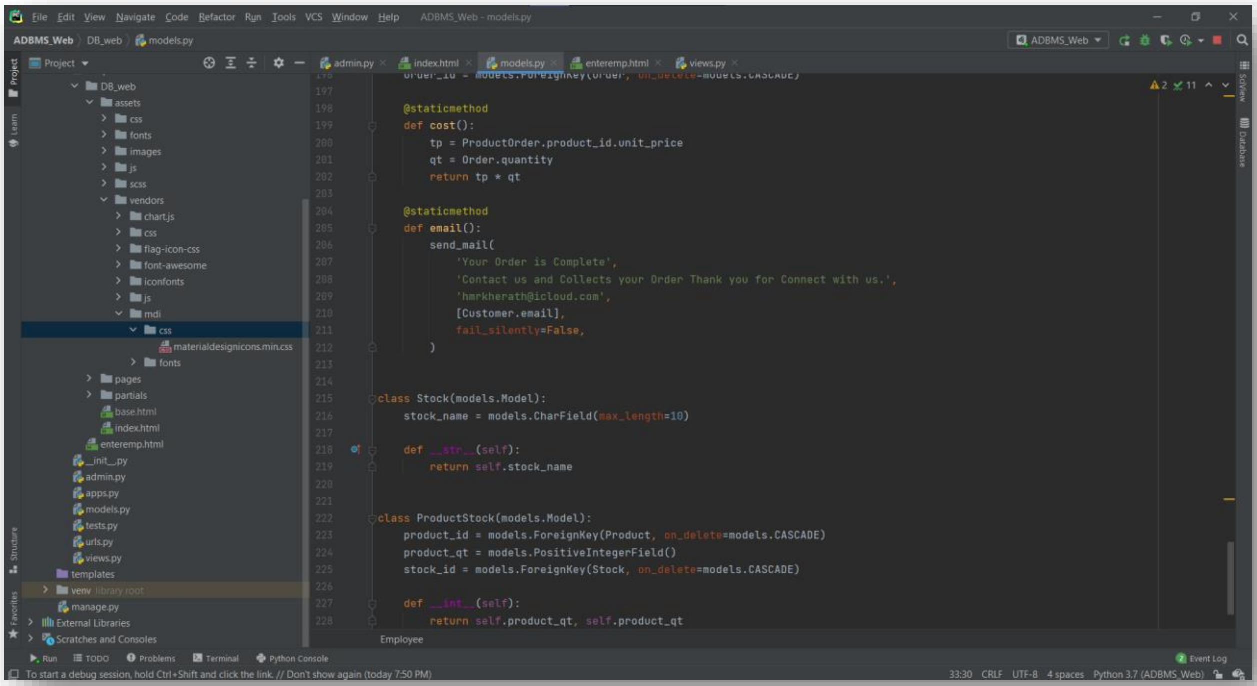
To start a debug session, hold Ctrl+Shift and click the link. // Don't show again (today 7:50 PM)



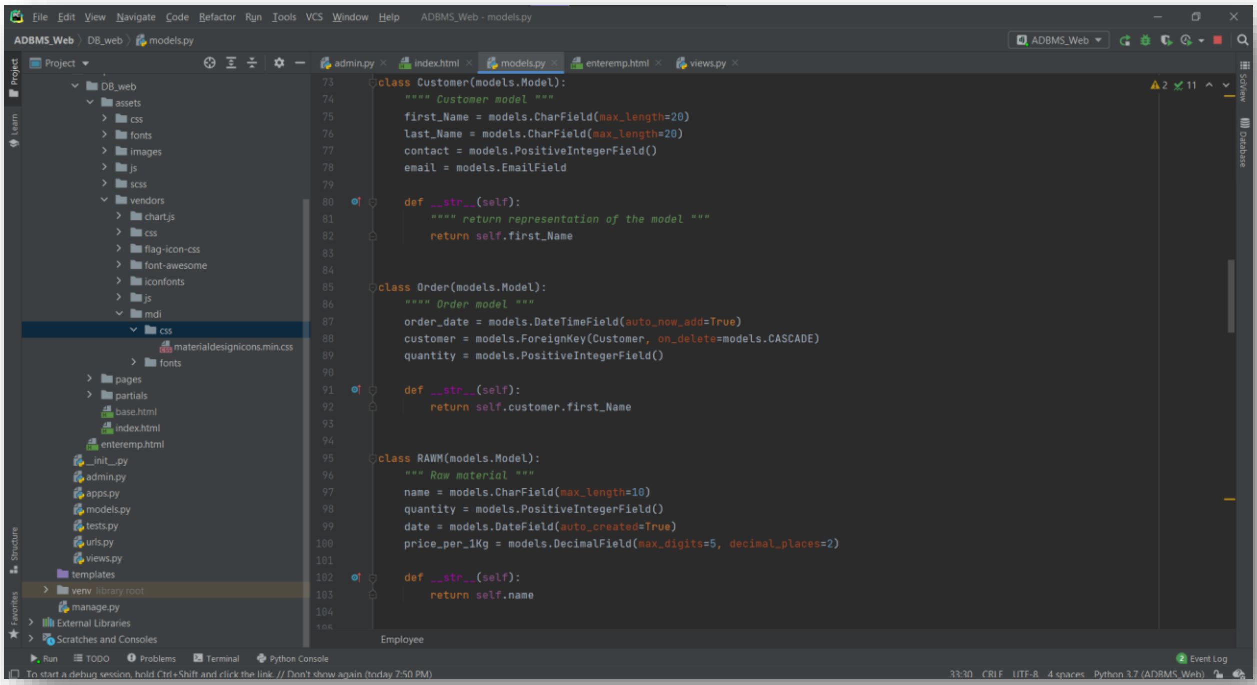
```python
class ProductOrder(models.Model):
    """" Details about order product"""
    product_id = models.ForeignKey(Product, on_delete=models.CASCADE)
    oder_id = models.ForeignKey(Order, on_delete=models.CASCADE)

    # def __init__(self):
    #     return self.oder_id


class Supplier(models.Model):
    """" Details about Order """
    first_name = models.CharField(max_length=10)
    last_name = models.CharField(max_length=10)
    contact = models.CharField(max_length=10)
    email = models.EmailField
    address = models.CharField(max_length=20)
    ACCN = models.PositiveIntegerField()

    def __str__(self):
        """" return representation of the model """
        return self.first_name


class SupplierRAWM(models.Model):
    """" Details about Supplier and RAW Materials """
    RAWM_id = models.ForeignKey(RAWM, on_delete=models.CASCADE)
    Sup_id = models.ForeignKey(Supplier, on_delete=models.CASCADE)

    def __int__(self):
        """" return representation of the model """
        return self.RAWM_id
```



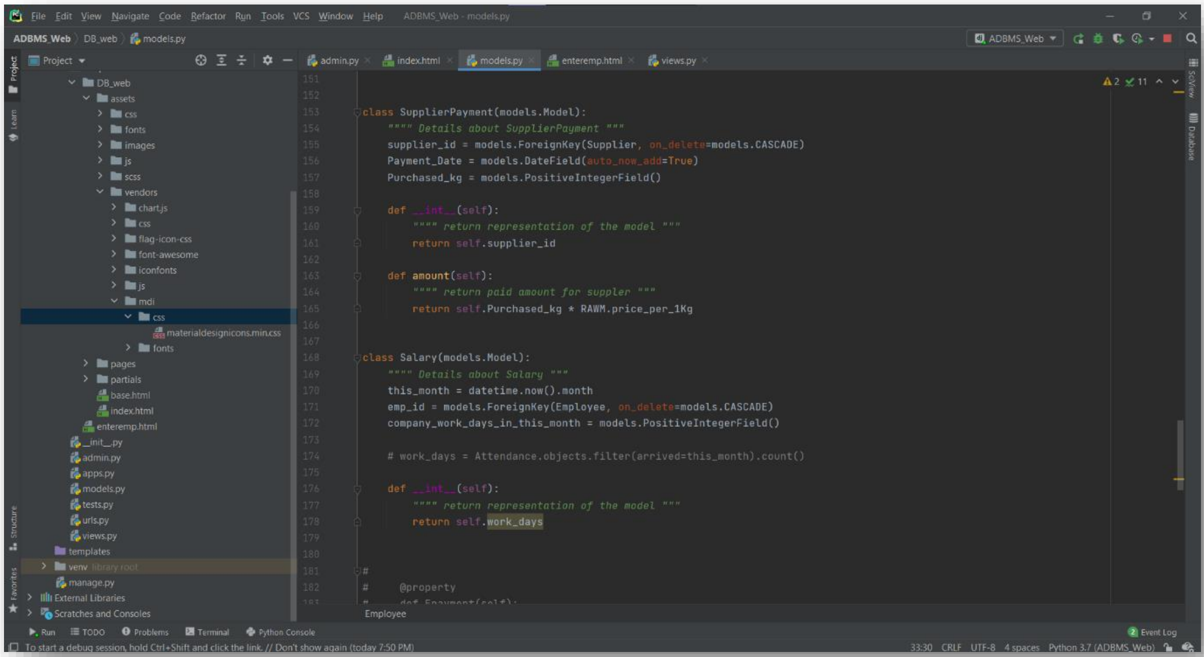
```python
class Employee(models.Model):
    """" Employee model """
    first_Name = models.CharField(max_length=20)
    last_Name = models.CharField(max_length=20)
    job_Title = models.ForeignKey(Jobtype, on_delete=models.CASCADE)
    email = models.EmailField()
    contact_Number = models.CharField(max_length=10)
    nic = models.PositiveIntegerField()
    gender = models.CharField(max_length=1)
    bank_Account_Number = models.PositiveIntegerField()
    hire_Date = models.DateTimeField(auto_now_add=True)
    DOB = models.DateField()
    department_Name = models.ForeignKey(Department, on_delete=models.CASCADE)

    def __str__(self):
        """" return representation of the model """
        return self.first_Name


class Attendance(models.Model):
    emp_id = models.ForeignKey(Employee, on_delete=models.CASCADE)
    arrived = models.DateTimeField()
    left = models.DateTimeField()

    def __mul__(self):
        """" return representation of the model """
        return self.emp_id.first_Name, self.left

    @property
    def workingtime(self):
        """" return Employee Working time """
```



```python
    order_id = models.ForeignKey(Order, on_delete=models.CASCADE)

    @staticmethod
    def cost():
        tp = ProductOrder.product_id.unit_price
        qt = Order.quantity
        return tp * qt

    @staticmethod
    def email():
        send_mail(
            'Your Order is Complete',
            'Contact us and Collects your Order Thank you for Connect with us.',
            'hmrkherath@icloud.com',
            [Customer.email],
            fail_silently=False,
        )


class Stock(models.Model):
    stock_name = models.CharField(max_length=10)

    def __str__(self):
        return self.stock_name


class ProductStock(models.Model):
    product_id = models.ForeignKey(Product, on_delete=models.CASCADE)
    product_qt = models.PositiveIntegerField()
    stock_id = models.ForeignKey(Stock, on_delete=models.CASCADE)

    def __int__(self):
        return self.product_qt, self.product_qt
```



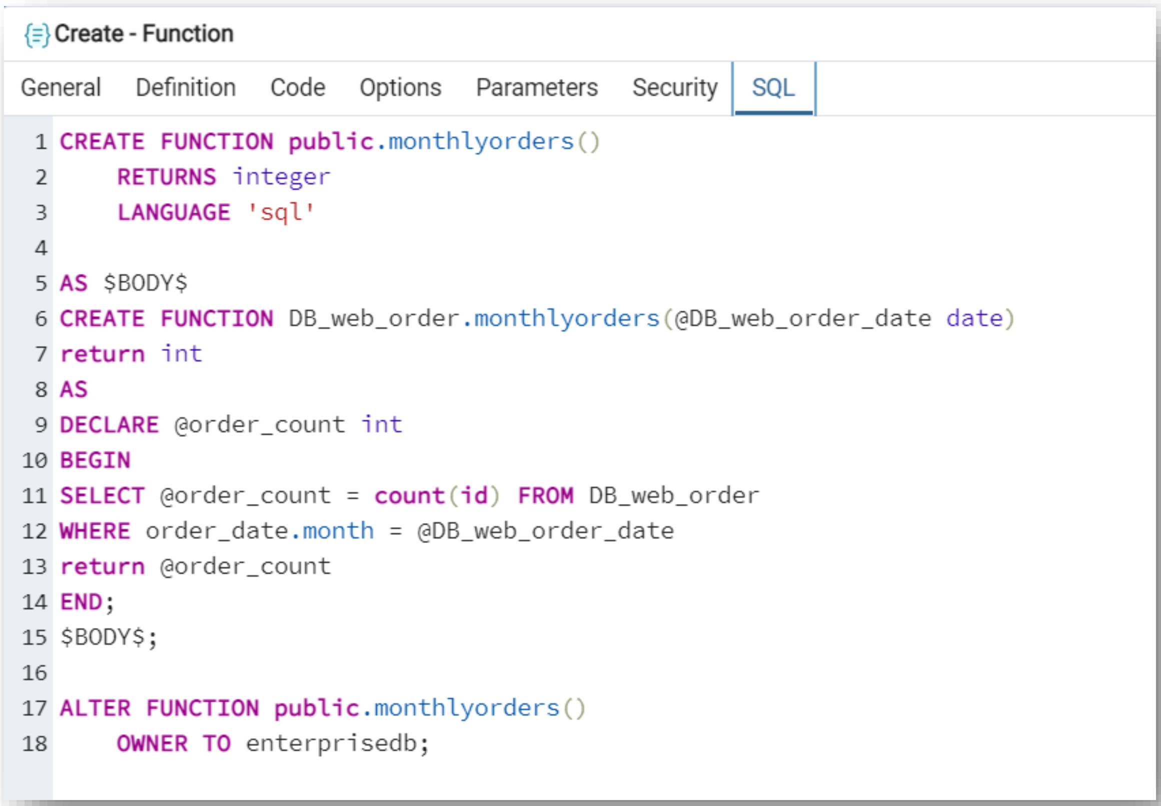
```python
class Customer(models.Model):
    """" Customer model """
    first_Name = models.CharField(max_length=20)
    last_Name = models.CharField(max_length=20)
    contact = models.PositiveIntegerField()
    email = models.EmailField

    def __str__(self):
        """" return representation of the model """
        return self.first_Name


class Order(models.Model):
    """" Order model """
    order_date = models.DateTimeField(auto_now_add=True)
    customer = models.ForeignKey(Customer, on_delete=models.CASCADE)
    quantity = models.PositiveIntegerField()

    def __str__(self):
        return self.customer.first_Name


class RAWM(models.Model):
    """ Raw material """
    name = models.CharField(max_length=10)
    quantity = models.PositiveIntegerField()
    date = models.DateField(auto_created=True)
    price_per_1Kg = models.DecimalField(max_digits=5, decimal_places=2)

    def __str__(self):
        return self.name
```



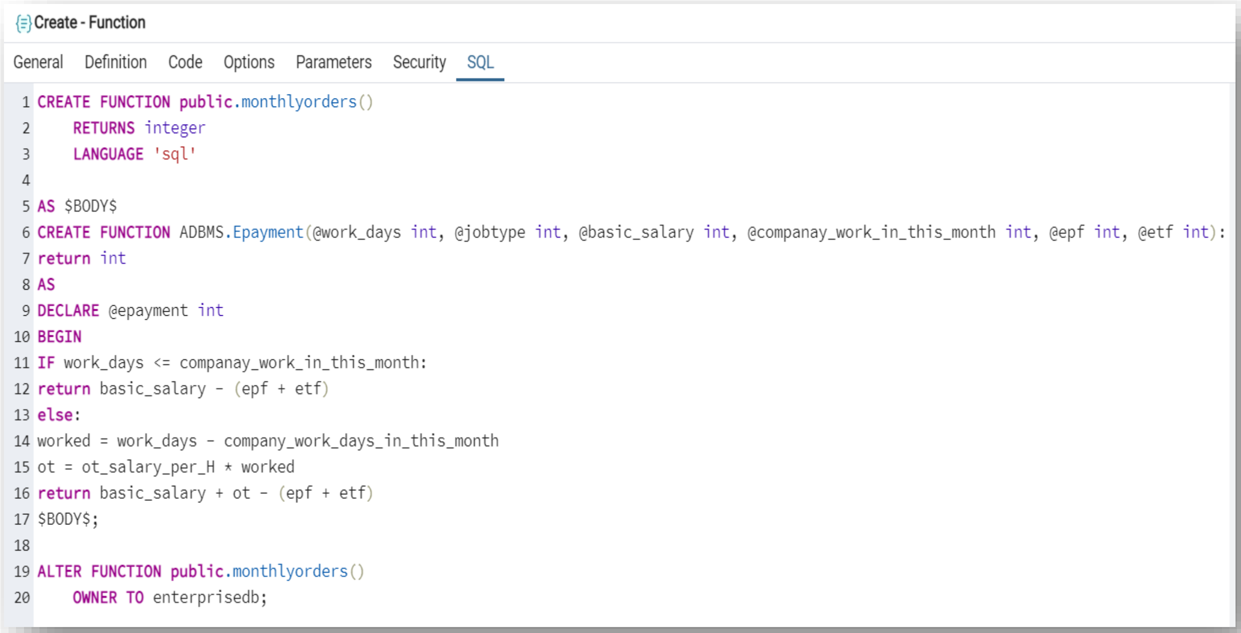
```
class SupplierPayment(models.Model):
    """" Details about SupplierPayment """
    supplier_id = models.ForeignKey(Supplier, on_delete=models.CASCADE)
    Payment_Date = models.DateField(auto_now_add=True)
    Purchased_kg = models.PositiveIntegerField()

    def __int__(self):
        """" return representation of the model """
        return self.supplier_id

    def amount(self):
        """" return paid amount for suppler """
        return self.Purchased_kg * RAWM.price_per_1Kg


class Salary(models.Model):
    """" Details about Salary """
    this_month = datetime.now().month
    emp_id = models.ForeignKey(Employee, on_delete=models.CASCADE)
    company_work_days_in_this_month = models.PositiveIntegerField()

    # work_days = Attendance.objects.filter(arrived=this_month).count()

    def __int__(self):
        """" return representation of the model """
        return self.work_days


#
#     @property
```

To start a debug session, hold Ctrl+Shift and click the link. // Don't show again (today 7:50 PM)

Create Functions Statements screen shots



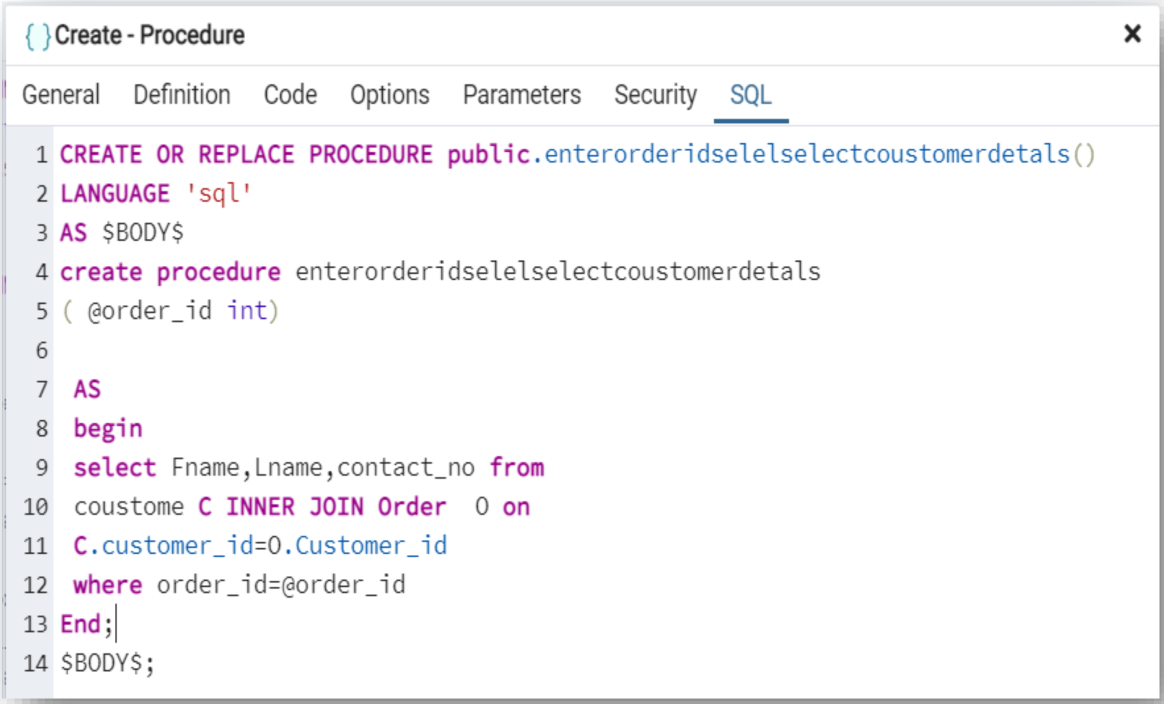
```
CREATE FUNCTION public.monthlyorders()
    RETURNS integer
    LANGUAGE 'sql'

AS $BODY$
CREATE FUNCTION DB_web_order.monthlyorders(@DB_web_order_date date)
return int
AS
DECLARE @order_count int
BEGIN
SELECT @order_count = count(id) FROM DB_web_order
WHERE order_date.month = @DB_web_order_date
return @order_count
END;
$BODY$;

ALTER FUNCTION public.monthlyorders()
    OWNER TO enterprisedb;
```


Create - Function

General Definition Code Options Parameters Security SQL

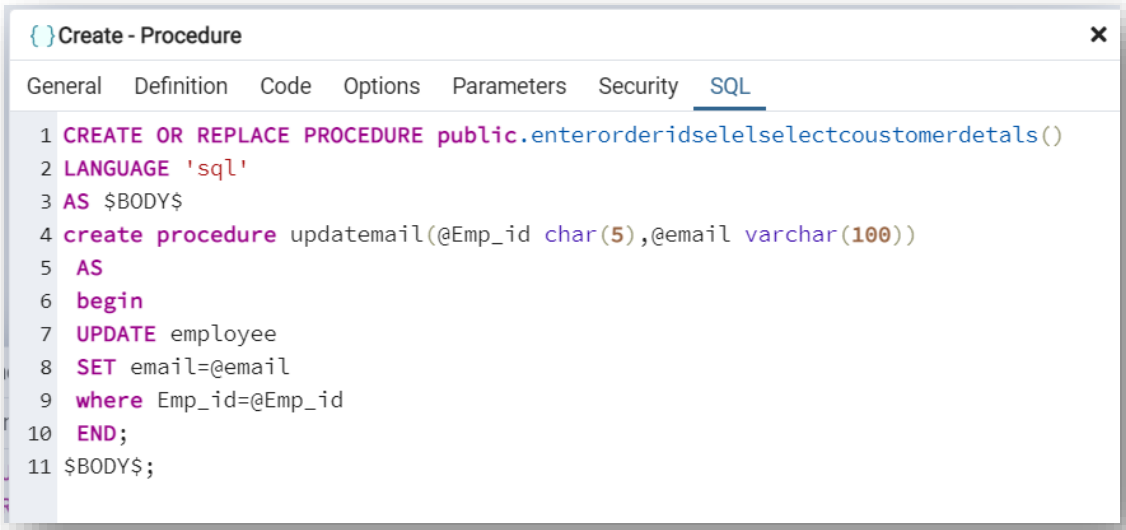
```
CREATE FUNCTION public.monthlyorders()
    RETURNS integer
    LANGUAGE 'sql'

AS $BODY$
CREATE FUNCTION ADBMS.Epayment(@work_days int, @jobtype int, @basic_salary int, @companay_work_in_this_month int, @epf int, @etf int):
return int
AS
DECLARE @epayment int
BEGIN
IF work_days <= companay_work_in_this_month:
return basic_salary - (epf + etf)
else:
worked = work_days - company_work_days_in_this_month
ot = ot_salary_per_H * worked
return basic_salary + ot - (epf + etf)
$BODY$;

ALTER FUNCTION public.monthlyorders()
    OWNER TO enterprisedb;
```

Create Procedure Statements screen shots


Create - Procedure

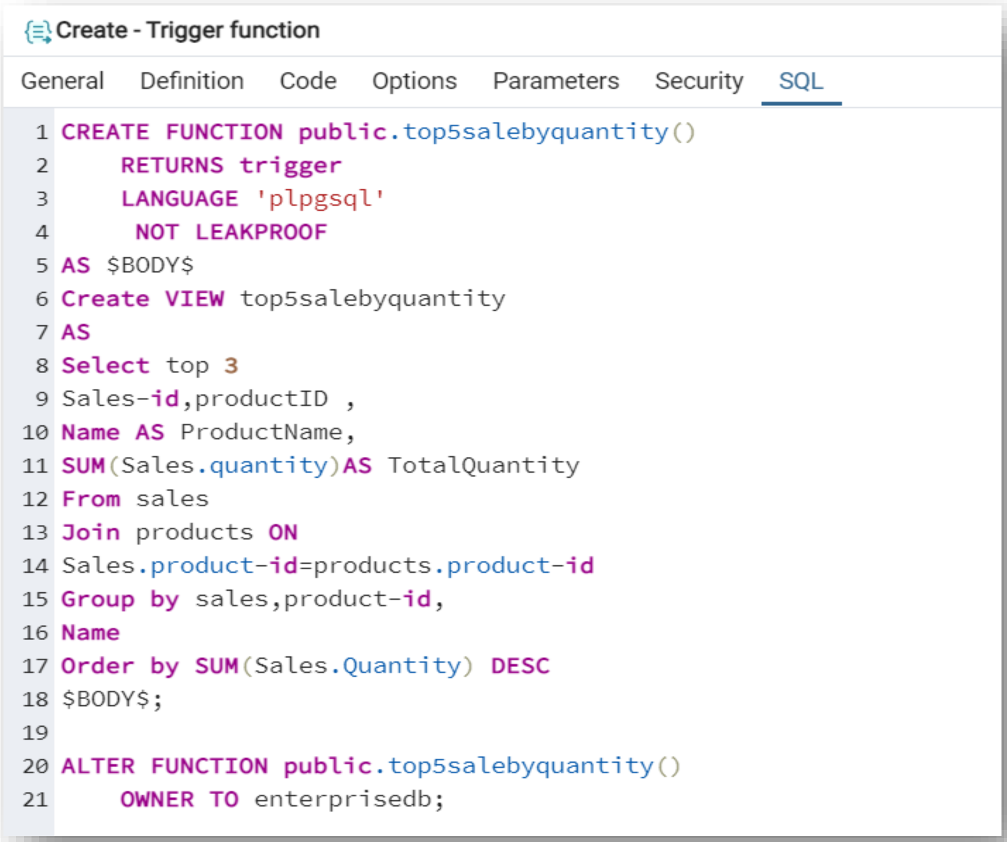
General Definition Code Options Parameters Security SQL

```
CREATE OR REPLACE PROCEDURE public.enterorderidselelselectcoustomerdetals()
LANGUAGE 'sql'
AS $BODY$
create procedure enterorderidselelselectcoustomerdetals
( @order_id int)

 AS
 begin
 select Fname,Lname,contact_no from
 coustome C INNER JOIN Order  O on
 C.customer_id=O.Customer_id
 where order_id=@order_id
End;
$BODY$;
```

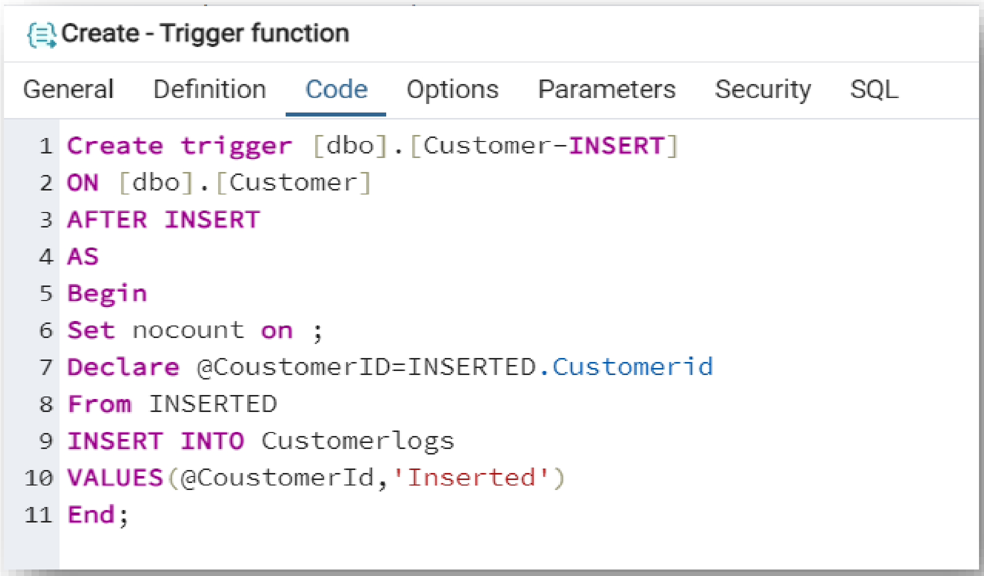


```
CREATE OR REPLACE PROCEDURE public.enterorderidselelselectcoustomerdetals()
LANGUAGE 'sql'
AS $BODY$
create procedure updatemail(@Emp_id char(5),@email varchar(100))
 AS
 begin
 UPDATE employee
 SET email=@email
 where Emp_id=@Emp_id
 END;
$BODY$;
```

Create Triggers Statements screen shots

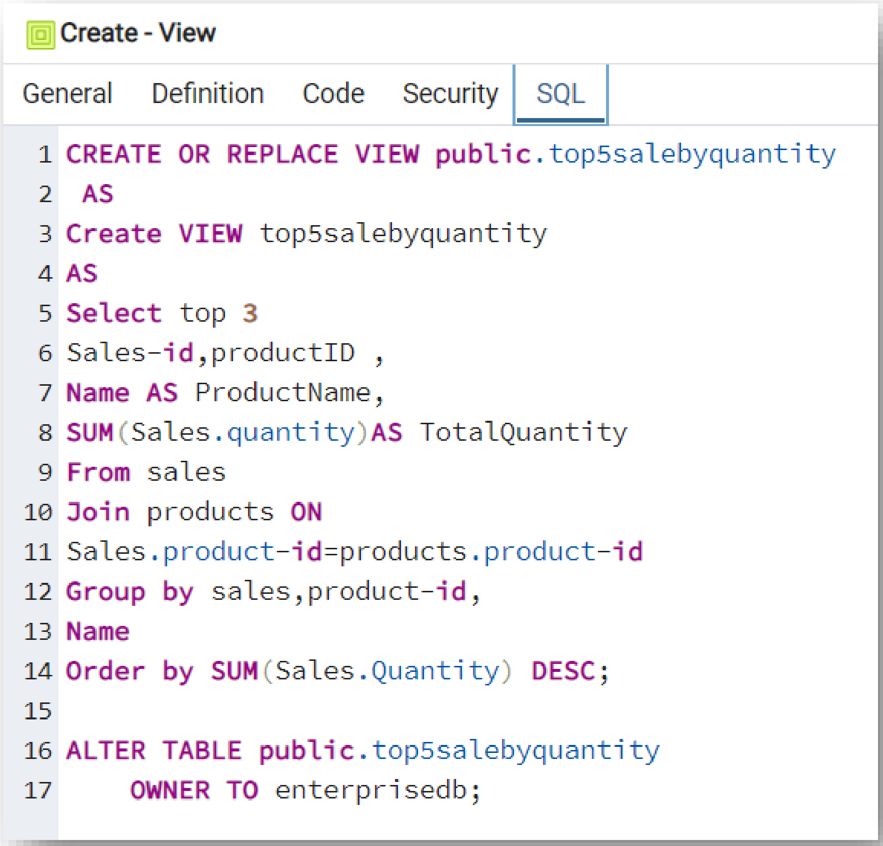


```
CREATE FUNCTION public.top5salebyquantity()
    RETURNS trigger
    LANGUAGE 'plpgsql'
     NOT LEAKPROOF
AS $BODY$
Create VIEW top5salebyquantity
AS
Select top 3
Sales-id,productID ,
Name AS ProductName,
SUM(Sales.quantity)AS TotalQuantity
From sales
Join products ON
Sales.product-id=products.product-id
Group by sales,product-id,
Name
Order by SUM(Sales.Quantity) DESC
$BODY$;

ALTER FUNCTION public.top5salebyquantity()
    OWNER TO enterprisedb;
```

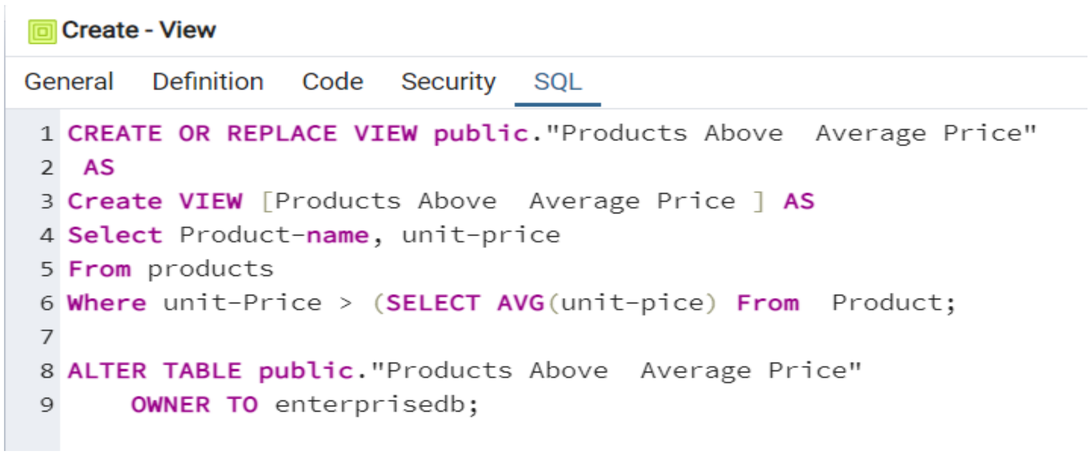


```
Create trigger [dbo].[Customer-INSERT]
ON [dbo].[Customer]
AFTER INSERT
AS
Begin
Set nocount on ;
Declare @CoustomerID=INSERTED.Customerid
From INSERTED
INSERT INTO Customerlogs
VALUES(@CoustomerId,'Inserted')
End;
```

Create View Statements screen shots

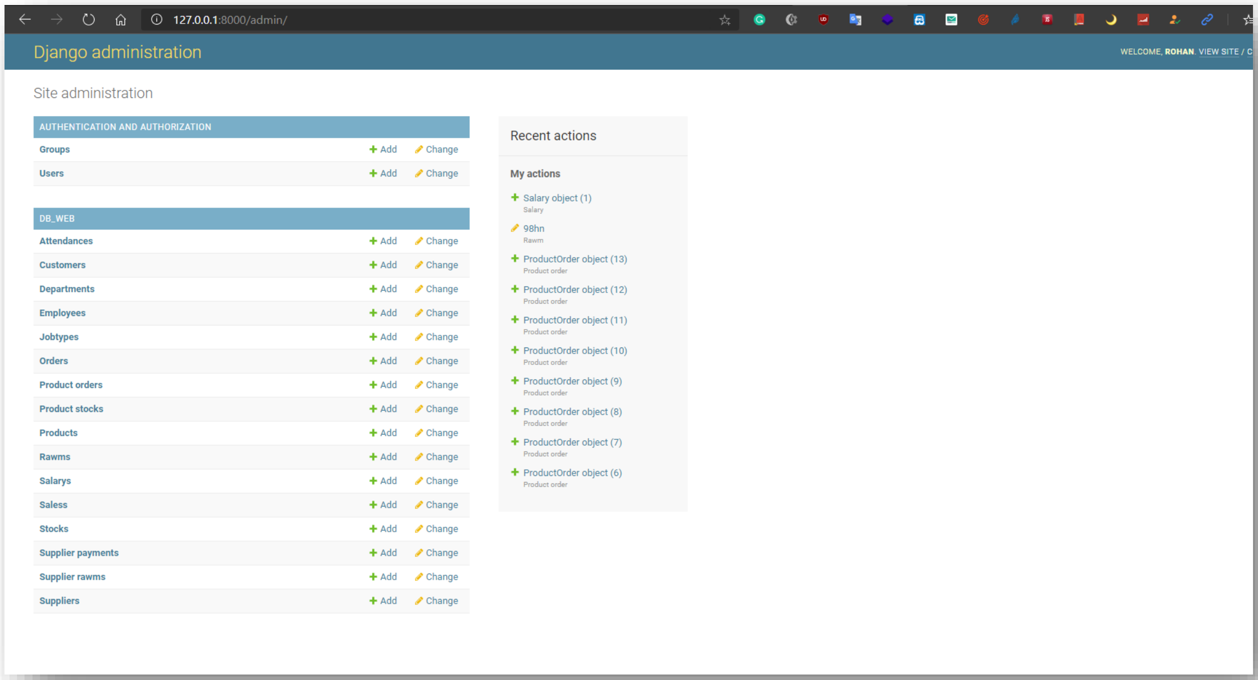


```
CREATE OR REPLACE VIEW public.top5salebyquantity
 AS
Create VIEW top5salebyquantity
AS
Select top 3
Sales-id,productID ,
Name AS ProductName,
SUM(Sales.quantity)AS TotalQuantity
From sales
Join products ON
Sales.product-id=products.product-id
Group by sales,product-id,
Name
Order by SUM(Sales.Quantity) DESC;

ALTER TABLE public.top5salebyquantity
    OWNER TO enterprisedb;
```



```
CREATE OR REPLACE VIEW public."Products Above  Average Price"
 AS
Create VIEW [Products Above  Average Price ] AS
Select Product-name, unit-price
From products
Where unit-Price > (SELECT AVG(unit-pice) From  Product;

ALTER TABLE public."Products Above  Average Price"
    OWNER TO enterprisedb;
```

Admin Panel Screen Shots



AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change

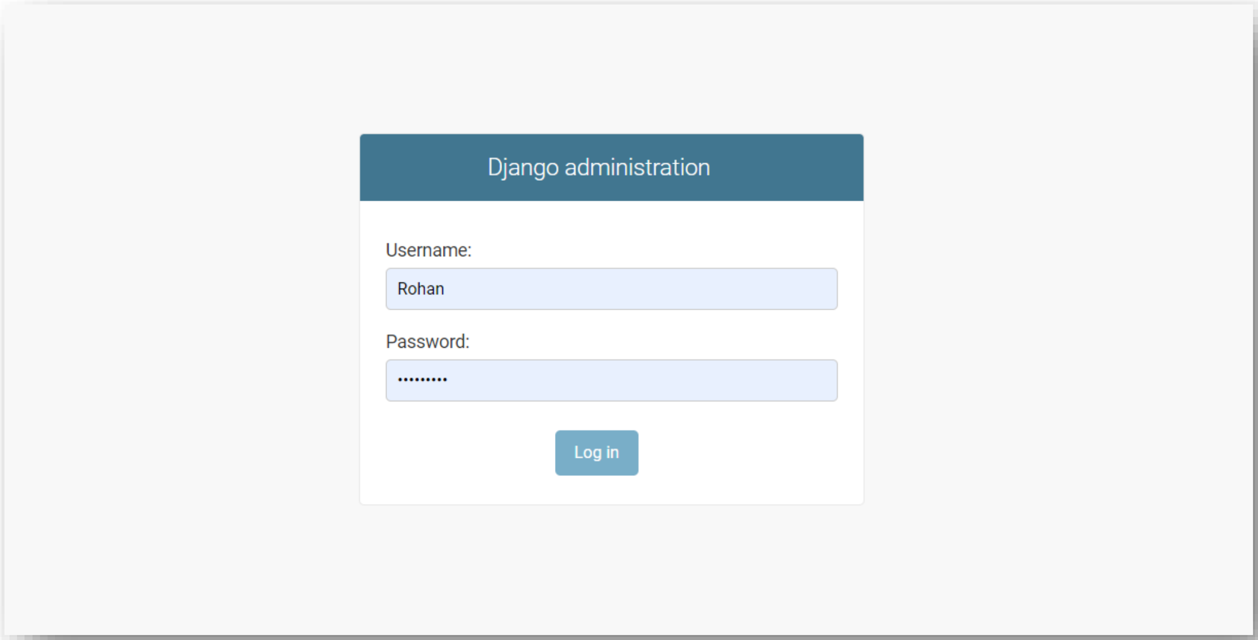
DB_WEB		
Attendances	+ Add	Change
Customers	+ Add	Change
Departments	+ Add	Change
Employees	+ Add	Change
Jobtypes	+ Add	Change
Orders	+ Add	Change
Product orders	+ Add	Change
Product stocks	+ Add	Change
Products	+ Add	Change
Rawms	+ Add	Change
Salarys	+ Add	Change
Saless	+ Add	Change
Stocks	+ Add	Change
Supplier payments	+ Add	Change
Supplier rawms	+ Add	Change
Suppliers	+ Add	Change

Recent actions

My actions

- Salary object (1) — Salary
- 98hn — Rawm
- ProductOrder object (13) — Product order
- ProductOrder object (12) — Product order
- ProductOrder object (11) — Product order
- ProductOrder object (10) — Product order
- ProductOrder object (9) — Product order
- ProductOrder object (8) — Product order
- ProductOrder object (7) — Product order
- ProductOrder object (6) — Product order

User Login Screen Shots

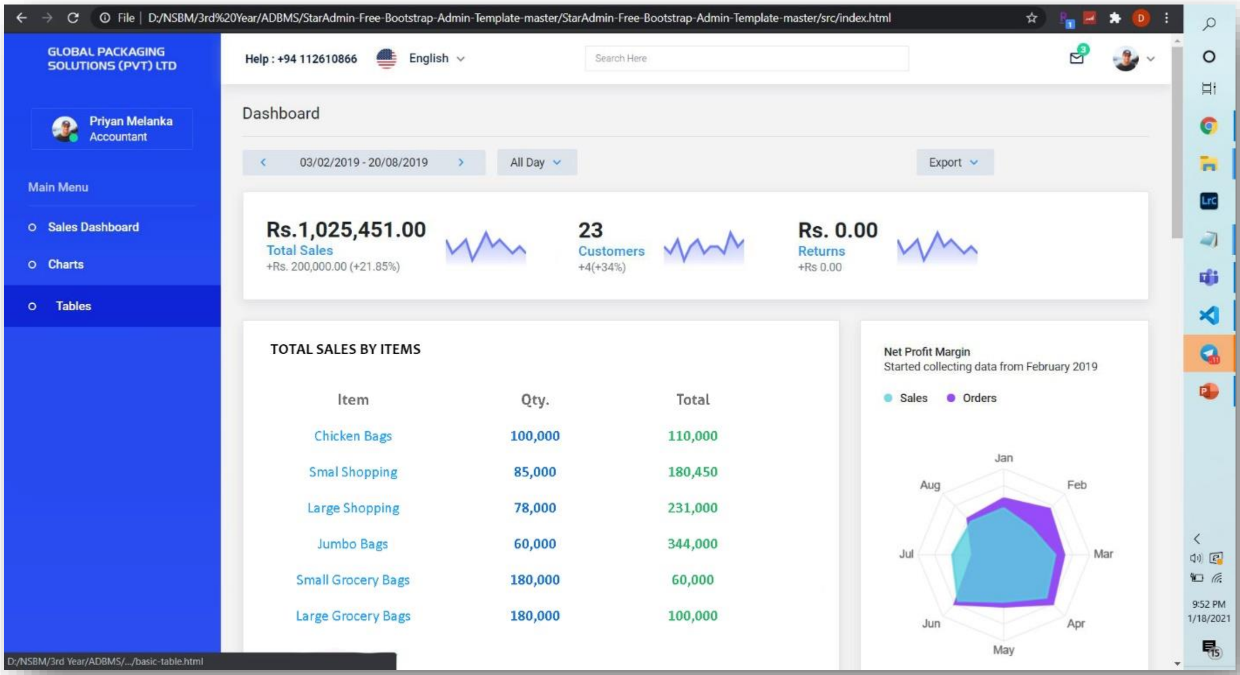


Django administration

Username: Rohan

Password: ....

Dashboard Screen Shots



File | D:/NSBM/3rd%20Year/ADBMS/StarAdmin-Free-Bootstrap-Admin-Template-master/StarAdmin-Free-Bootstrap-Admin-Template-master/src/index.html

GLOBAL PACKAGING SOLUTIONS (PVT) LTD

Help : +94 112610866

Priyan Melanka
Accountant

Dashboard

03/02/2019 - 20/08/2019

Rs.1,025,451.00
Total Sales
+Rs. 200,000.00 (+21.85%)

23
Customers
+4(+34%)

Rs. 0.00
Returns
+Rs 0.00

TOTAL SALES BY ITEMS

Item	Qty.	Total
Chicken Bags	100,000	110,000
Smal Shopping	85,000	180,450
Large Shopping	78,000	231,000
Jumbo Bags	60,000	344,000
Small Grocery Bags	180,000	60,000
Large Grocery Bags	180,000	100,000

Net Profit Margin
Started collecting data from February 2019