# Postal Management System

**Submitted By:**

**Manisha Tanwar**
**Sriharsha Vemugunta**

## Acknowledgement

We respect and thank Professor Stephen J Frein , for providing us an opportunity to do the project work in Database Systems course and giving us all support and guidance, which made us complete the project duly.
We are extremely thankful to him for his expert advice and encouragement throughout this project by providing such a nice support.

## Table of Contents

## <u>Requirement Overview</u>

Description: A database system about how postal management system works. The management system focuses on manual processes for mail and package intake and delivery, usually with a front desk worker or back office employee receiving.

Below are the major entities that will interact to make this system working.

<u>Customer</u>: For every person who comes in as a Sender, Postal Management will capture the Customer's full name, phone numbers, emails, address details including address line, city, state, country and zip code and auto generated customer Id to unique identify any customer. We also capture the Receiver details as a different customer with the same attributes as defined for Sender.

<u>Transaction:</u> For every item that a Customer/Sender requests to send will be a enclosed in terms of a transaction that is unique for a particular Sender, Receiver and item details to be send across.

For each item to be delivered, we will record the itemType, itemCategory, deliveryType, transaction start date, sender's detail, receiver's detail and unique id of the employee who performed this transaction and the store id where this transaction is performed.

ItemType can take different values like Delicate, Document, Standard.

ItemCategory can take different values like Small, Medium, Large.

Delivery Type can take different values like Standard, Overnight, and Urgent.

Depending upon the details entered system will calculate the charges and the delivery date.

<u>Store:</u> We will record every store name, and address details including the address line, city, state, zip code and country and an auto generated unique id that will uniquely identify the store. A store can employ multiple employees.

<u>Employee</u>: We will capture all the employees working in the different store throughout the country. There may be multiple employees working for a store. Every employee will have SSN that will uniquely identify each employee. Also, we will capture employee full name, salary, emails, phone numbers and address details including address line, city, state, zip code, country.

An employee can either be a Full time employee or part time employee with fixed annual salary or it can be a part time with hourly salary.

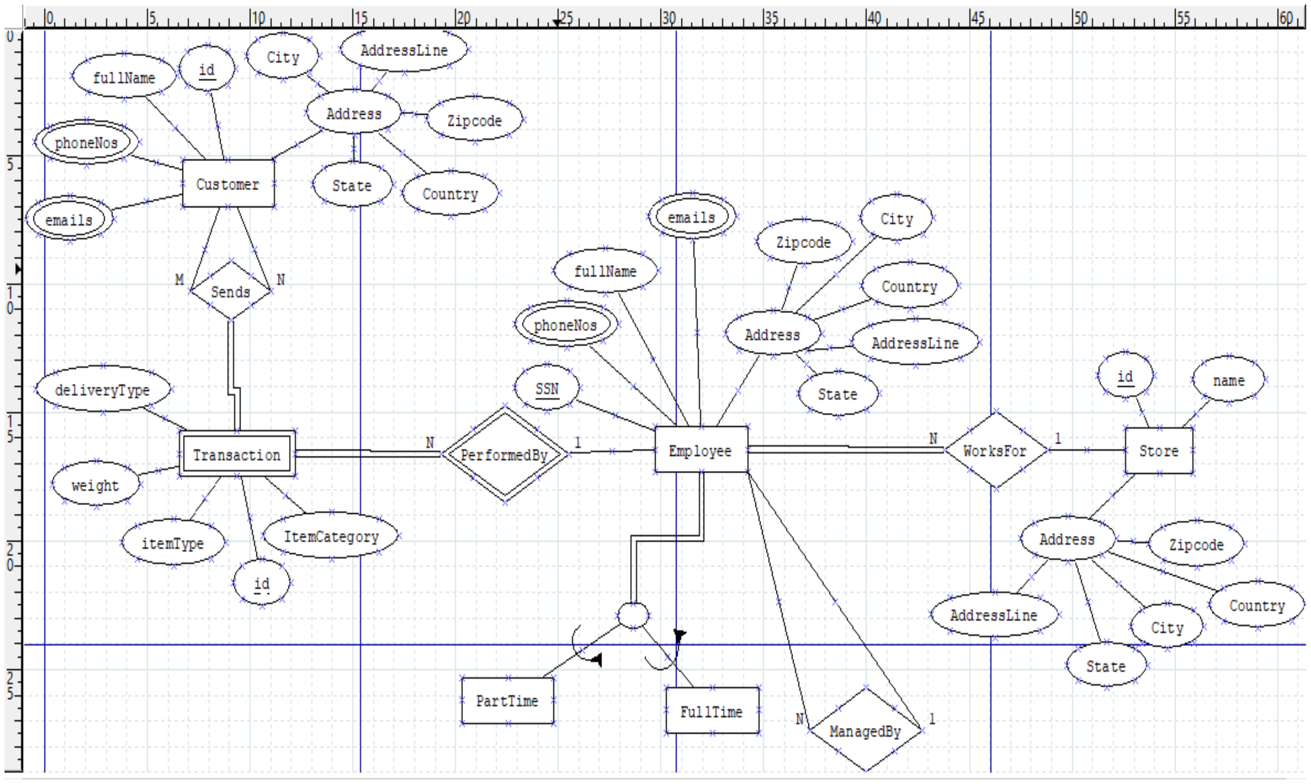
<u>Manager:</u> There is a hierarchy among the employees. An employee can report to a Manager, and Manager can have multiple employees working under him.

Services provided:

1. Create, update, view and delete Store details.
2. Add, update, view and delete employee(s) to Store(s).
3. Manage employee and manager hierarchy. Assign, un-assign manager and employee.
4. Sender can come in make a transaction in the store that will involve shipping an item from one store to the receiver’s address.
5. Store manager manages multiple employees at a store and oversees the transaction.
6. Sender can update the receiver’s address for the item shipment.
7. Sender can request the delivery type (Standard or Priority) only before the item is not assigned a stamp. Sender can view the status of transaction.
8. Calculate the bill of transaction for the sender based on his/her preferences like Item type (ItemType, ItemCategory, DeliveryType)
9. Generate different report based on the different criteria:

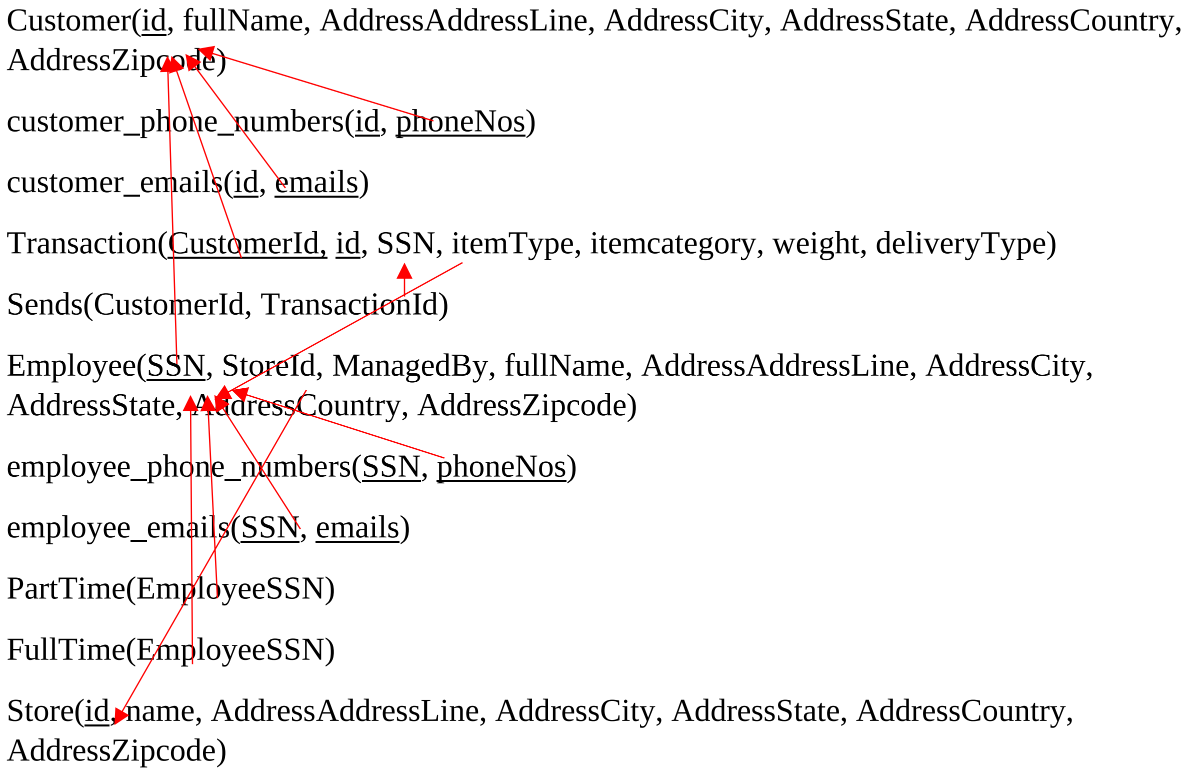
    - Number of transactions performed in a particular store between a date range.
    - Number of transactions performed by a particular employee in a specific store.
    - Number of items shipped from a particular Customer.

## ERD Model



fullName
id
City
AddressLine
Address
Zipcode
phoneNos
Customer
State
Country
emails
Sends
M
N
deliveryType
Transaction
weight
itemType
ItemCategory
id
PerformedBy
N
1
emails
Zipcode
City
fullName
Country
phoneNos
Address
AddressLine
SSN
State
Employee
WorksFor
N
1
Store
id
name
Address
Zipcode
AddressLine
Country
City
State
PartTime
FullTime
ManagedBy
N
1

**Relational Schema**



Customer(id, fullName, AddressAddressLine, AddressCity, AddressState, AddressCountry, AddressZipcode)

customer_phone_numbers(id, phoneNos)

customer_emails(id, emails)

Transaction(CustomerId, id, SSN, itemType, itemcategory, weight, deliveryType)

Sends(CustomerId, TransactionId)

Employee(SSN, StoreId, ManagedBy, fullName, AddressAddressLine, AddressCity, AddressState, AddressCountry, AddressZipcode)

employee_phone_numbers(SSN, phoneNos)

employee_emails(SSN, emails)

PartTime(EmployeeSSN)

FullTime(EmployeeSSN)

Store(id, name, AddressAddressLine, AddressCity, AddressState, AddressCountry, AddressZipcode)

## Data Dictionary

**Customer**: Contains information about a Customer who wants to send a package using the postal management system.

| Attribute Name | Description | Datatype | Domain | Nullable | PK | FK |
|---|---|---|---|---|---|---|
| ID | Customer Unique Identifier | NUMBER(10,0) | All | No | Yes | No |
| NAME | Customer Full Name | VARCHAR2(25) | All | No | No | No |
| ADDREESS_LINE | Customer Street Address | VARCHAR2(255) | All | Yes | No | No |
| CITY | Customer City | VARCHAR2(50) | All | Yes | No | No |
| COUNTRY | Customer Country | VARCHAR2(50) | All | Yes | No | No |
| STATE | Customer State | VARCHAR2(50) | All | Yes | No | No |
| ZIP_CODE | ZipCode of the area | VARCHAR2(50) | All | Yes | No | No |

**Customer Email**: Contains information about Customer emails.

| Attribute Name | Description | Datatype | Domain | Nullable | PK | FK |
|---|---|---|---|---|---|---|
| CUSTOMER_ID | Customer Unique Identifier | NUMBER(10,0) | All | No | No | Yes |
| EMAILID | Customer Email ID | VARCHAR2(50) | All | Yes | No | No |

**Customer Phone Numbers**: Contains information about Customer phone numbers.

| Attribute Name | Description | Datatype | Domain | Nullable | PK | FK |
|---|---|---|---|---|---|---|
| CUSTOMER_ID | Customer Unique Identifier | NUMBER(10,0) | All | No | No | Yes |
| PHONE_NUMBER | Customer Phone number | NUMBER(10) | All | No | No | No |

**Store**: Contains information about Stores that process the packages

| Attribute Name | Description | Datatype | Domain | Nullable | PK | FK |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| ID | Store Unique Identifier | NUMBER(10,0) | All | No | Yes | No |
| NAME | Store name | VARCHAR2(100) | All | No | No | No |
| ADDREESS_LINE | Store Street Address | VARCHAR2(255) | All | Yes | No | No |
| CITY | Store City | VARCHAR2(50) | All | Yes | No | No |
| COUNTRY | Store Country | VARCHAR2(50) | All | Yes | No | No |
| STATE | Store State | VARCHAR2(50) | All | Yes | No | No |
| ZIP_CODE | ZipCode of the Store | VARCHAR2(50) | All | Yes | No | No |

**Employee**: Contains information about Employees that process the packages

| Attribute Name | Description | Datatype | Domain | Nullable | PK | FK |
|---|---|---|---|---|---|---|
| EMPLOYEE_TYPE | Employee Type (Full time or Part time) | CHAR(4) | {“Full”, ”Part”} | No | No | No |
| SSN | Employee Unique identifier | CHAR(9) | {111-11-1111 – 999-99-9999} | No | Yes | No |
| NAME | Employee full name | VARCHAR2(100) | All | No | No | No |
| ADDREESS_LINE | Employee Street Address | VARCHAR2(255) | All | Yes | No | No |
| CITY | Employee City | VARCHAR2(50) | All | Yes | No | No |
| COUNTRY | Employee Country | VARCHAR2(50) | All | Yes | No | No |
| STATE | Employee State | VARCHAR2(50) | All | Yes | No | No |
| ZIP_CODE | Zip Code of the Employee | VARCHAR2(50) | All | Yes | No | No |
| STORE | Store unique identifier | NUMBER(10,0) | All | No | No | Yes |
| SALARY | Salary for full time employees | NUMBER(8,2) | All | Yes | No | No |
| HOURLY_RATE | Hourly rate for part time employees | NUMBER(8,2) | All | Yes | No | No |

**Employee Email**: Contains information about Employee emails.

| Attribute Name | Description | Datatype | Domain | Nullable | PK | FK |
|---|---|---|---|---|---|---|
| EMPLOYEE_SSN | Employee Unique Identifier | CHAR(9) | {111-11-1111 – 999-99-9999} | No | No | Yes |
| EMAILID | Employee Email ID | VARCHAR2(50) | All | Yes | No | No |

**Employee Phone Numbers**: Contains information about Employee phone numbers.

| Attribute Name | Description | Datatype | Domain | Nullable | PK | FK |
|---|---|---|---|---|---|---|
| EMPLOYEE_SSN | Employee Unique Identifier | CHAR(9) | {111-11-1111 – 999-99-9999} | No | No | Yes |
| PHONE_NUMBER | Employee Phone number | CHAR(11) | All | No | No | No |

**Employee Manager**: Contains information about Employee Manager

| Attribute Name | Description | Datatype | Domain | Nullable | PK | FK |
|---|---|---|---|---|---|---|
| MANAGER_ID | Manager unique identifier | CHAR(9) | All | No | No | Yes |
| EMPLOYEE_ID | Employee unique identifier | CHAR(9) | All | No | No | Yes |

**Transaction**: Contains information about transactions made by the customer

| Attribute Name | Description | Datatype | Domain | Nullable | PK | FK |
|---|---|---|---|---|---|---|
| ID | Transaction Unique Identifier | NUMBER(10,0) | All | No | Yes | No |
| CATEGORY | Item categorized based on size | VARCHAR2(10) | {'small','medium','large'} | No | No | No |
| CHARGES | Charges calculated for delivering the item | NUMBER(10,2) | All | No | No | No |
| DELIVERY_DATE | Estimated date of delivery | DATE | All | Yes | No | No |
| DELIVERY_TYPE | Type of delivery | VARCHAR2(10) | {'standard','overnight','urgent'} | No | No | No |

| | | | | | | |
|---|---|---|---|---|---|---|
| ITEM_TYPE | Type of Item | VARCHAR2(10) | {'delicate','normal','document'} | No | No | No |
| START_DATE | Transaction start date | DATE | {SYSDATE} | No | No | No |
| WEIGHT | Weight of the package | NUMBER(4,3) | All | Yes | No | No |
| PERFORMED_BY | Who performs the transaction | CHAR(9) | All | No | No | Yes |
| RECEIVER | The receiver of this item | NUMBER(10,0) | All | No | No | Yes |
| SENDER | The sender of this item | NUMBER(10,0) | All | No | No | Yes |

## DDL SQL

### 1.Customer

```
create table customer (
id number(10,0)  CONSTRAINT customer_pk PRIMARY KEY,
name varchar2(25) CONSTRAINT customer_uq_name UNIQUE
                                   CONSTRAINT customer_nn_name NOT NULL,
addreess_line varchar2(255),
city varchar2(50),
country varchar2(50),
state varchar2(50),
zip_code varchar2(50)
);
```

### 2.Customer Email

```
create table customer_emails (
customer_id number(10,0) CONSTRAINT customer_nn_id NOT NULL,
emailid varchar2(50),
CONSTRAINT customer_fk_email FOREIGN KEY (customer_id) REFERENCES
customer(id) ON DELETE CASCADE
);
```

### 3. Customer phone numbers

```
create table customer_phone_numbers (
customer_id number(10,0) CONSTRAINT customer_pno_nn_id NOT NULL,
phone_number number(10) CONSTRAINT customer_pno_uq UNIQUE CONSTRAINT
customer_pno_nn NOT NULL,
CONSTRAINT customer_fk_pno FOREIGN KEY (customer_id) REFERENCES
customer(id) ON DELETE CASCADE
);
```

### 4. Store

```
create table store (
 id number(10,0) CONSTRAINT store_pk PRIMARY KEY,
 name varchar2(100) CONSTRAINT store_nn_name NOT NULL,
 addreess_line varchar2(255),
 city varchar2(50),
 country varchar2(50),
 state varchar2(50),
 zip_code varchar2(50)
 );
```

## 5. Employee

```
create table employee (
employee_type char(4) CONSTRAINT employee_nn_type NOT NULL,
ssn char(9) CONSTRAINT employee_pk PRIMARY KEY,
name varchar2(100) CONSTRAINT employee_uq_name UNIQUE
                                CONSTRAINT employee_nn_name NOT NULL,
addreess_line varchar2(255),
city varchar2(50),
country varchar2(50),
state varchar2(50),
zip_code varchar2(50),
store number(10,0),
salary number(8,2),
hourly_rate number(8,2),
CONSTRAINT employee_fk_store FOREIGN KEY (store) REFERENCES store(id) ON
DELETE SET NULL
);
```

## 6. Employee email

```
create table employee_emails (
employee_ssn char(9)  CONSTRAINT employee_email_nn_ssn NOT NULL,
 emailid varchar2(50),
 CONSTRAINT employee_fk_email FOREIGN KEY (employee_ssn) REFERENCES
employee(ssn) ON DELETE CASCADE
 );
```

## 7. Employee Phone number

```
 create table employee_phone_numbers (
 employee_ssn char(9) CONSTRAINT employee_pno_nn_ssn NOT NULL,
 phone_number char(11) CONSTRAINT employee_pno_uq UNIQUE
                                        CONSTRAINT employee_pno_nn NOT NULL,
 CONSTRAINT employee_fk_pno FOREIGN KEY (employee_ssn) REFERENCES
employee(ssn) ON DELETE CASCADE
 );
```

## 8. Employee Manager

```
 create table employee_manager (
manager_id char(9) CONSTRAINT manager_ssn_nn NOT NULL ,
 CONSTRAINT employee_manager__managerId_fk FOREIGN KEY (manager_id)
REFERENCES employee(ssn) ON DELETE CASCADE,
employee_id char(9) CONSTRAINT employee_ssn_nn NOT NULL,
 CONSTRAINT employee_manager__empId_fk FOREIGN KEY (employee_id)
REFERENCES employee(ssn) ON DELETE CASCADE
 );
```

## 9. Transaction

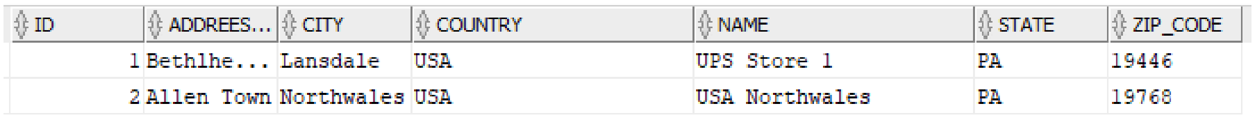
```
create table transaction (
 id number(10,0) CONSTRAINT transaction_pk PRIMARY KEY,
 category varchar2(10) CONSTRAINT transaction_category_chk check (category IN
('small','medium','large')),
 charges number(10,2)  CONSTRAINT transaction_charges_nn NOT NULL,
 delivery_date date ,
 delivery_type varchar2(10) CONSTRAINT transaction_del_type_chk check (delivery_type
IN ('standard','overnight','urgent')),
 item_type varchar2(10) CONSTRAINT transaction_type_chk check (item_type IN
('delicate','normal','document')),
 start_date date DEFAULT SYSDATE,
 weight number(4,3),
performed_by char(9) CONSTRAINT transaction_employee_nn NOT NULL,
CONSTRAINT transaction_fk_employee FOREIGN KEY (performed_by) REFERENCES
employee(ssn) ON DELETE CASCADE,
 receiver number(10,0) CONSTRAINT transaction_receiver_nn NOT NULL,
CONSTRAINT transaction_fk_receiver FOREIGN KEY (receiver) REFERENCES
customer(id) ON DELETE CASCADE,
 sender number(10,0) CONSTRAINT transaction_sender_nn NOT NULL, CONSTRAINT
transaction_fk_sender FOREIGN KEY (sender) REFERENCES customer(id) ON DELETE
CASCADE

 );
```

**DML:**

**Store:**

INSERT INTO "MANI_DBA"."STORE" (ID, NAME, ADDREESS_LINE, CITY, COUNTRY, STATE, ZIP_CODE) VALUES ('1', 'UPS Store 1', 'Bethlhem Pik', 'Lansdale', 'USA', 'PA', '19446');
INSERT INTO "MANI_DBA"."STORE" (ID, NAME, ADDREESS_LINE, CITY, COUNTRY, STATE, ZIP_CODE) VALUES ('2', 'USA Northwales', 'Allen Town', 'Northwales', 'USA', 'PA', '19768');



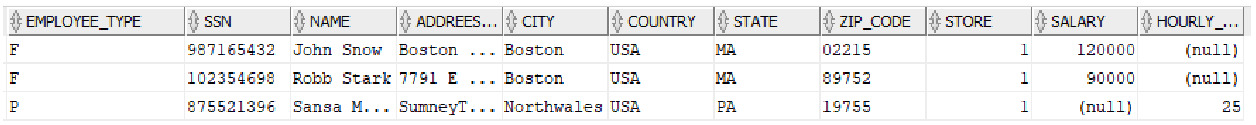
| ID | ADDREES... | CITY | COUNTRY | NAME | STATE | ZIP_CODE |
|---|---|---|---|---|---|---|
| 1 | Bethlhe... | Lansdale | USA | UPS Store 1 | PA | 19446 |
| 2 | Allen Town | Northwales | USA | USA Northwales | PA | 19768 |

**Employee:**

INSERT INTO "MANI_DBA"."EMPLOYEE" (EMPLOYEE_TYPE, SSN, NAME, ADDREESS_LINE, CITY, COUNTRY, STATE, ZIP_CODE, STORE, SALARY) VALUES ('F', '987165432', 'John Snow', 'Boston MA. 1185 Boylston St.', 'Boston', 'USA', 'MA', '02215', '1', '120000');

INSERT INTO "MANI_DBA"."EMPLOYEE" (EMPLOYEE_TYPE, SSN, NAME, ADDREESS_LINE, CITY, COUNTRY, STATE, ZIP_CODE, STORE, SALARY) VALUES ('F', '102354698', 'Robb Stark', '7791 E Osborn Rd', 'Boston', 'USA', 'MA', '89752', '1', '90000');

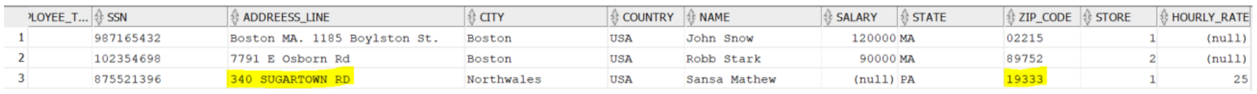
INSERT INTO "MANI_DBA"."EMPLOYEE" (EMPLOYEE_TYPE, SSN, NAME, ADDREESS_LINE, CITY, COUNTRY, STATE, ZIP_CODE, STORE, HOURLY_RATE) VALUES ('P', '875521396', 'Sansa Mathew', 'SumneyTown Pike Rd.', 'Northwales', 'USA', 'PA', '19755', '1', '25');



| EMPLOYEE_TYPE | SSN | NAME | ADDREES... | CITY | COUNTRY | STATE | ZIP_CODE | STORE | SALARY | HOURLY_... |
|---|---|---|---|---|---|---|---|---|---|---|
| F | 987165432 | John Snow | Boston ... | Boston | USA | MA | 02215 | 1 | 120000 | (null) |
| F | 102354698 | Robb Stark | 7791 E ... | Boston | USA | MA | 89752 | 1 | 90000 | (null) |
| P | 875521396 | Sansa M... | SumneyT... | Northwales | USA | PA | 19755 | 1 | (null) | 25 |

UPDATE "SVEMUGUN"."EMPLOYEE" SET ADDREESS_LINE='340 SUGARTOWN RD', ZIP_CODE='19333' WHERE SSN='875521396';

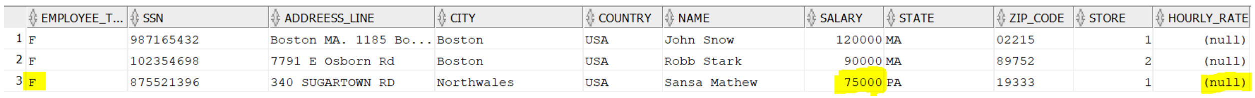
COMMIT;



| | LOYEE_T... | SSN | ADDREESS_LINE | CITY | COUNTRY | NAME | SALARY | STATE | ZIP_CODE | STORE | HOURLY_RATE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | 987165432 | Boston MA. 1185 Boylston St. | Boston | USA | John Snow | 120000 | MA | 02215 | 1 | (null) |
| 2 | | 102354698 | 7791 E Osborn Rd | Boston | USA | Robb Stark | 90000 | MA | 89752 | 2 | (null) |
| 3 | | 875521396 | 340 SUGARTOWN RD | Northwales | USA | Sansa Mathew | (null) | PA | 19333 | 1 | 25 |

UPDATE "SVEMUGUN"."EMPLOYEE" SET EMPLOYEE_TYPE='F', SALARY='75000', HOURLY_RATE=NULL WHERE SSN='875521396';

COMMIT;



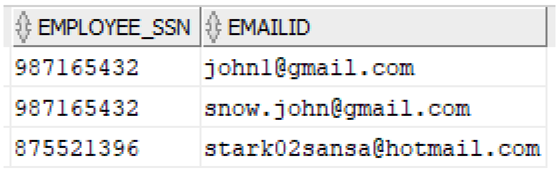
|  | EMPLOYEE_T... | SSN | ADDREESS_LINE | CITY | COUNTRY | NAME | SALARY | STATE | ZIP_CODE | STORE | HOURLY_RATE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | F | 987165432 | Boston MA. 1185 Bo... | Boston | USA | John Snow | 120000 | MA | 02215 | 1 | (null) |
| 2 | F | 102354698 | 7791 E Osborn Rd | Boston | USA | Robb Stark | 90000 | MA | 89752 | 2 | (null) |
| 3 | F | 875521396 | 340 SUGARTOWN RD | Northwales | USA | Sansa Mathew | 75000 | PA | 19333 | 1 | (null) |

## Employee_emails:

INSERT INTO "MANI_DBA"."EMPLOYEE_EMAILS" (EMPLOYEE_SSN, EMAILID) VALUES ('987165432', 'john1@gmail.com');

INSERT INTO "MANI_DBA"."EMPLOYEE_EMAILS" (EMPLOYEE_SSN, EMAILID) VALUES ('987165432', 'snow.john@gmail.com');

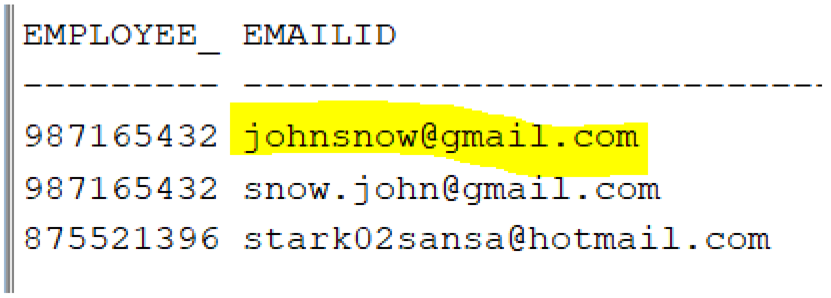
INSERT INTO "MANI_DBA"."EMPLOYEE_EMAILS" (EMPLOYEE_SSN, EMAILID) VALUES ('875521396', 'stark02sansa@hotmail.com');



| EMPLOYEE_SSN | EMAILID |
|---|---|
| 987165432 | john1@gmail.com |
| 987165432 | snow.john@gmail.com |
| 875521396 | stark02sansa@hotmail.com |

UPDATE "SVEMUGUN"."EMPLOYEE_EMAILS" SET EMAILID='johnsnow@gmail.com' WHERE EMPLOYEE_SSN='987165432' AND EMAILID='john1@gmail.com';

COMMIT;



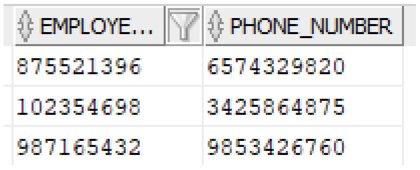
```
EMPLOYEE_ EMAILID
--------- ----------------------------
987165432 johnsnow@gmail.com
987165432 snow.john@gmail.com
875521396 stark02sansa@hotmail.com
```

## Employee_phonenumbers:

INSERT INTO "MANI_DBA"."EMPLOYEE_PHONE_NUMBERS" (EMPLOYEE_SSN, PHONE_NUMBER) VALUES ('875521396', '6574329820');

INSERT INTO "MANI_DBA"."EMPLOYEE_PHONE_NUMBERS" (EMPLOYEE_SSN, PHONE_NUMBER) VALUES ('102354698', '3425864875');

INSERT INTO "MANI_DBA"."EMPLOYEE_PHONE_NUMBERS" (EMPLOYEE_SSN, PHONE_NUMBER) VALUES ('987165432', '9853426760');

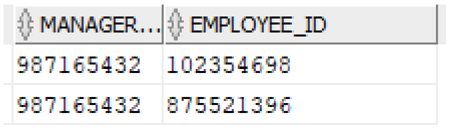


| EMPLOYE... | PHONE_NUMBER |
|---|---|
| 875521396 | 6574329820 |
| 102354698 | 3425864875 |
| 987165432 | 9853426760 |

## Employee_Manager:

INSERT INTO "MANI_DBA"."EMPLOYEE_MANAGER" (MANAGER_ID, EMPLOYEE_ID) VALUES ('987165432', '102354698');

INSERT INTO "MANI_DBA"."EMPLOYEE_MANAGER" (MANAGER_ID, EMPLOYEE_ID) VALUES ('987165432', '875521396');

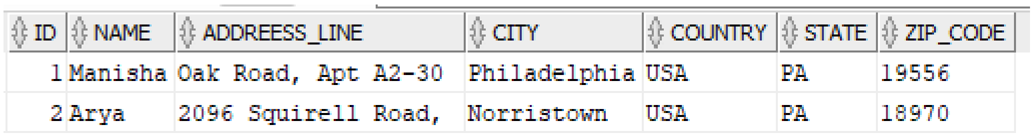


| MANAGER... | EMPLOYEE_ID |
|---|---|
| 987165432 | 102354698 |
| 987165432 | 875521396 |

## Customer:

INSERT INTO "MANI_DBA"."CUSTOMER" (ID, NAME, ADDREESS_LINE, CITY, COUNTRY, STATE, ZIP_CODE) VALUES ('1', 'Manisha', 'Oak Road, Apt A2-30', 'Philadelphia', 'USA', 'PA', '19556');

INSERT INTO "MANI_DBA"."CUSTOMER" (ID, NAME, ADDREESS_LINE, CITY, COUNTRY, STATE, ZIP_CODE) VALUES ('2', 'Arya', '2096 Squirell Road, ', 'Norristown', 'USA', 'PA', '18970');



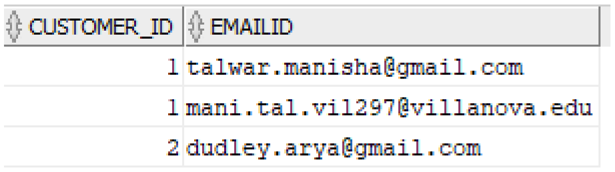
| ID | NAME | ADDREESS_LINE | CITY | COUNTRY | STATE | ZIP_CODE |
|---|---|---|---|---|---|---|
| 1 | Manisha | Oak Road, Apt A2-30 | Philadelphia | USA | PA | 19556 |
| 2 | Arya | 2096 Squirell Road, | Norristown | USA | PA | 18970 |

**Customer_emails:**

INSERT INTO "MANI_DBA"."CUSTOMER_EMAILS" (CUSTOMER_ID, EMAILID) VALUES ('1', 'talwar.manisha@gmail.com');

INSERT INTO "MANI_DBA"."CUSTOMER_EMAILS" (CUSTOMER_ID, EMAILID) VALUES ('1', 'mani.tal.vil297@villanova.edu');

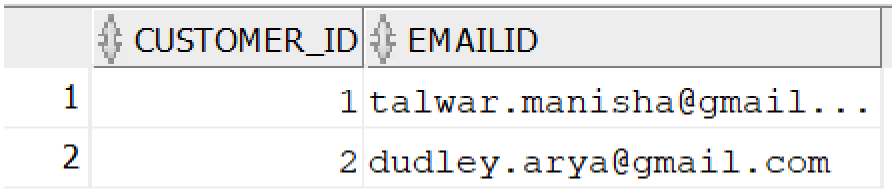
INSERT INTO "MANI_DBA"."CUSTOMER_EMAILS" (CUSTOMER_ID, EMAILID) VALUES ('2', 'dudley.arya@gmail.com');



| CUSTOMER_ID | EMAILID |
|---|---|
| 1 | talwar.manisha@gmail.com |
| 1 | mani.tal.vil297@villanova.edu |
| 2 | dudley.arya@gmail.com |

DELETE FROM "SVEMUGUN"."CUSTOMER_EMAILS" WHERE CUSTOMER_ID='1' AND EMAILID='mani.tal.vil297@villanova.edu';

COMMIT;



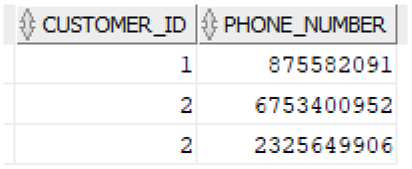
| | CUSTOMER_ID | EMAILID |
|---|---|---|
| 1 | 1 | talwar.manisha@gmail... |
| 2 | 2 | dudley.arya@gmail.com |

**Customer_phonenumbers:**

INSERT INTO "MANI_DBA"."CUSTOMER_PHONE_NUMBERS" (CUSTOMER_ID, PHONE_NUMBER) VALUES ('1', '875582091');

INSERT INTO "MANI_DBA"."CUSTOMER_PHONE_NUMBERS" (CUSTOMER_ID, PHONE_NUMBER) VALUES ('2', '6753400952');

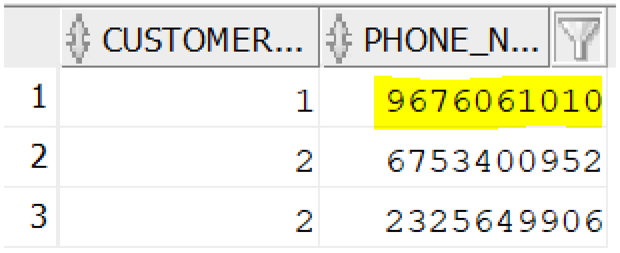
INSERT INTO "MANI_DBA"."CUSTOMER_PHONE_NUMBERS" (CUSTOMER_ID, PHONE_NUMBER) VALUES ('2', '2325649906');



| CUSTOMER_ID | PHONE_NUMBER |
|---|---|
| 1 | 875582091 |
| 2 | 6753400952 |
| 2 | 2325649906 |

UPDATE "SVEMUGUN"."CUSTOMER_PHONE_NUMBERS" SET PHONE_NUMBER='9676061010' WHERE CUSTOMER_ID='1';

COMMIT;



| | CUSTOMER... | PHONE_N... |
|---|---|---|
| 1 | 1 | 9676061010 |
| 2 | 2 | 6753400952 |
| 3 | 2 | 2325649906 |

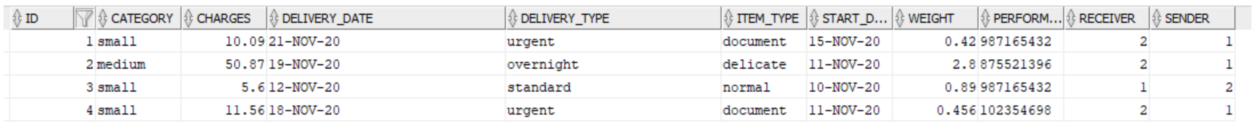
**Transaction:**

INSERT INTO "MANI_DBA"."TRANSACTION" (ID, CATEGORY, CHARGES, DELIVERY_TYPE, ITEM_TYPE, WEIGHT, PERFORMED_BY, RECEIVER, SENDER) VALUES ('1', 'small', '10.09', 'urgent', 'document', '0.420', '987165432', '2', '1');

INSERT INTO "MANI_DBA"."TRANSACTION" (ID, CATEGORY, CHARGES, DELIVERY_TYPE, ITEM_TYPE, WEIGHT, PERFORMED_BY, RECEIVER, SENDER) VALUES ('2', 'medium', '50.87', 'overnight', 'delicate', '2.800', '875521396', '2', '1');

INSERT INTO "MANI_DBA"."TRANSACTION" (ID, CATEGORY, CHARGES, DELIVERY_TYPE, ITEM_TYPE, WEIGHT, PERFORMED_BY, RECEIVER, SENDER) VALUES ('3', 'small', '5.60', 'standard', 'normal', '0.890', '987165432', '1', '2');

INSERT INTO "MANI_DBA"."TRANSACTION" (ID, CATEGORY, CHARGES, DELIVERY_TYPE, ITEM_TYPE, WEIGHT, PERFORMED_BY, RECEIVER, SENDER) VALUES ('4', 'small', '11.56', 'urgent', 'document', '0.456', '102354698', '2', '1');

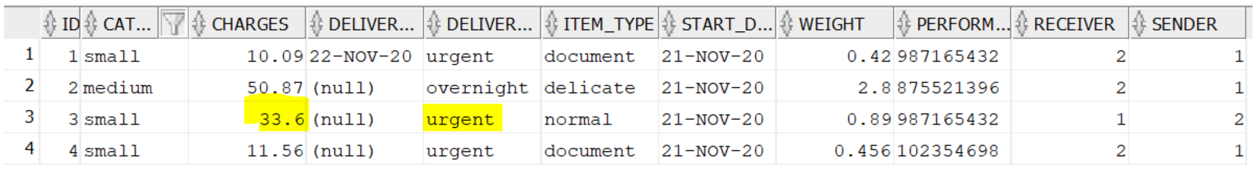


| ID | CATEGORY | CHARGES | DELIVERY_DATE | DELIVERY_TYPE | ITEM_TYPE | START_D... | WEIGHT | PERFORM... | RECEIVER | SENDER |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | small | 10.09 | 21-NOV-20 | urgent | document | 15-NOV-20 | 0.42 | 987165432 | 2 | 1 |
| 2 | medium | 50.87 | 19-NOV-20 | overnight | delicate | 11-NOV-20 | 2.8 | 875521396 | 2 | 1 |
| 3 | small | 5.6 | 12-NOV-20 | standard | normal | 10-NOV-20 | 0.89 | 987165432 | 1 | 2 |
| 4 | small | 11.56 | 18-NOV-20 | urgent | document | 11-NOV-20 | 0.456 | 102354698 | 2 | 1 |

UPDATE "MANI_DBA"."TRANSACTION" SET DELIVERY_DATE =to_date('22-Nov-20') WHERE id=1;

UPDATE "SVEMUGUN"."TRANSACTION" SET DELIVERY_TYPE ='urgent', CHARGES=CHARGES*3 WHERE id=3;

COMMIT;

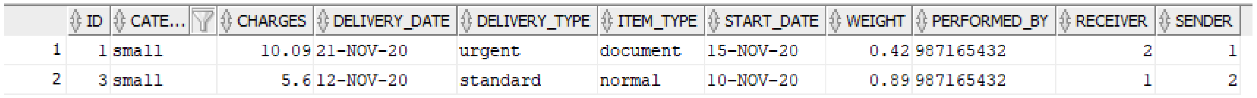


| | ID | CAT... | CHARGES | DELIVER... | DELIVER... | ITEM_TYPE | START_D... | WEIGHT | PERFORM... | RECEIVER | SENDER |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | small | 10.09 | 22-NOV-20 | urgent | document | 21-NOV-20 | 0.42 | 987165432 | 2 | 1 |
| 2 | 2 | medium | 50.87 | (null) | overnight | delicate | 21-NOV-20 | 2.8 | 875521396 | 2 | 1 |
| 3 | 3 | small | 33.6 | (null) | urgent | normal | 21-NOV-20 | 0.89 | 987165432 | 1 | 2 |
| 4 | 4 | small | 11.56 | (null) | urgent | document | 21-NOV-20 | 0.456 | 102354698 | 2 | 1 |

## Queries

1. Find recent transactions details performed by an employee in a store.

Select * FROM transaction  t WHERE performed_by= 987165432 ORDER BY delivery_date DESC;



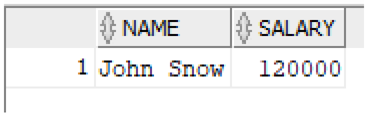
| | ID | CATE... | CHARGES | DELIVERY_DATE | DELIVERY_TYPE | ITEM_TYPE | START_DATE | WEIGHT | PERFORMED_BY | RECEIVER | SENDER |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | small | 10.09 | 21-NOV-20 | urgent | document | 15-NOV-20 | 0.42 | 987165432 | 2 | 1 |
| 2 | 3 | small | 5.6 | 12-NOV-20 | standard | normal | 10-NOV-20 | 0.89 | 987165432 | 1 | 2 |

2. Find all full time employees name and salary working for a store whose salary is > 95000

Select  name, salary

FROM employee

WHERE employee_type='F' AND store=1 AND salary > 95000;



| | NAME | SALARY |
|---|---|---|
| 1 | John Snow | 120000 |

3. Display all employees name and status working under a particular manager in a store.

Select e.name, e.employee_type

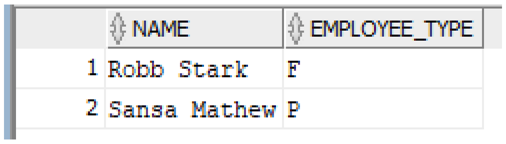
FROM employee e

WHERE e.ssn IN (

Select em.employee_id FROM employee_manager em

WHERE em.manager_id= 987165432

) AND e.store=1



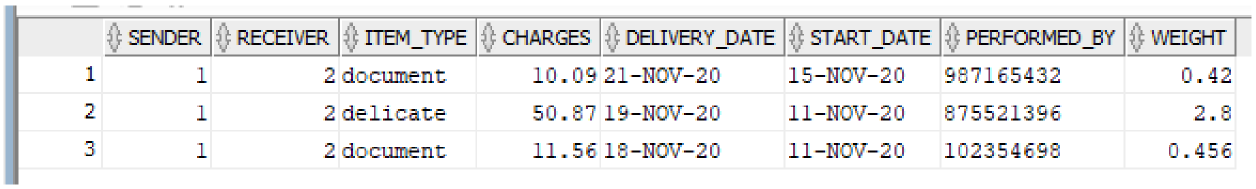
| | NAME | EMPLOYEE_TYPE |
|---|---|---|
| 1 | Robb Stark | F |
| 2 | Sansa Mathew | P |

4. Find all the transactions performed between a date range.

Select t.sender, t.receiver, t.item_type, t.charges, t.delivery_date, t.start_date, t.performed_by,t.weight

FROM transaction t

WHERE t.start_date BETWEEN to_date('11-Nov-20') AND to_date('18-Nov-20');



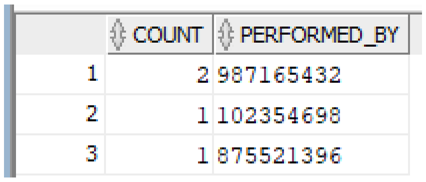
| | SENDER | RECEIVER | ITEM_TYPE | CHARGES | DELIVERY_DATE | START_DATE | PERFORMED_BY | WEIGHT |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | document | 10.09 | 21-NOV-20 | 15-NOV-20 | 987165432 | 0.42 |
| 2 | 1 | 2 | delicate | 50.87 | 19-NOV-20 | 11-NOV-20 | 875521396 | 2.8 |
| 3 | 1 | 2 | document | 11.56 | 18-NOV-20 | 11-NOV-20 | 102354698 | 0.456 |

5. Find number of transactions performed by each employee.

Select COUNT(t.sender) as count, t.performed_by

FROM transaction t

GROUP BY t.performed_by

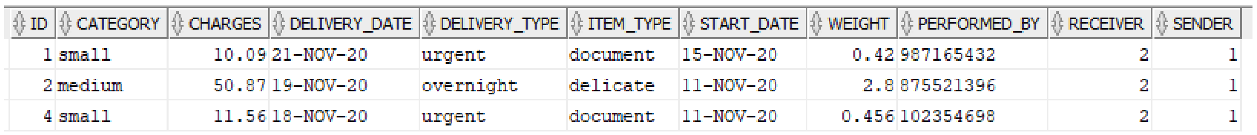


| | COUNT | PERFORMED_BY |
|---|---|---|
| 1 | 2 | 987165432 |
| 2 | 1 | 102354698 |
| 3 | 1 | 875521396 |

6. Find the transaction that are left to be delivered.

Select * FROM transaction  t

WHERE t.delivery_date > SYSDATE + 1;

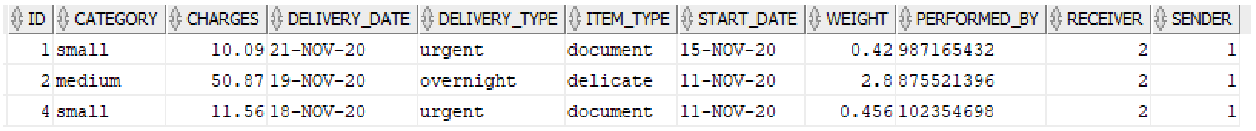


| ID | CATEGORY | CHARGES | DELIVERY_DATE | DELIVERY_TYPE | ITEM_TYPE | START_DATE | WEIGHT | PERFORMED_BY | RECEIVER | SENDER |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | small | 10.09 | 21-NOV-20 | urgent | document | 15-NOV-20 | 0.42 | 987165432 | 2 | 1 |
| 2 | medium | 50.87 | 19-NOV-20 | overnight | delicate | 11-NOV-20 | 2.8 | 875521396 | 2 | 1 |
| 4 | small | 11.56 | 18-NOV-20 | urgent | document | 11-NOV-20 | 0.456 | 102354698 | 2 | 1 |

7. Find all the urgent and overnight deliveries requested by a customer/sender.

Select * FROM transaction  t

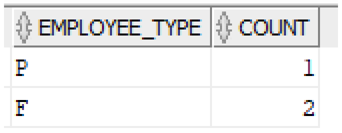
WHERE t.delivery_type IN ('urgent', 'overnight') AND sender=1;



| ID | CATEGORY | CHARGES | DELIVERY_DATE | DELIVERY_TYPE | ITEM_TYPE | START_DATE | WEIGHT | PERFORMED_BY | RECEIVER | SENDER |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | small | 10.09 | 21-NOV-20 | urgent | document | 15-NOV-20 | 0.42 | 987165432 | 2 | 1 |
| 2 | medium | 50.87 | 19-NOV-20 | overnight | delicate | 11-NOV-20 | 2.8 | 875521396 | 2 | 1 |
| 4 | small | 11.56 | 18-NOV-20 | urgent | document | 11-NOV-20 | 0.456 | 102354698 | 2 | 1 |

8. How many employees are working full time/part time?

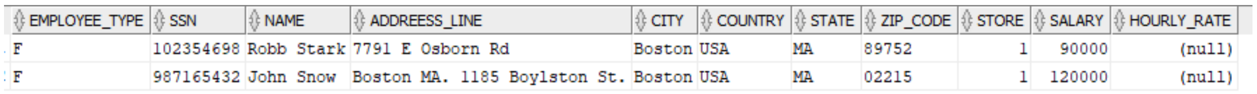
```
Select e.employee_type, count(e.ssn) AS count
FROM employee e
GROUP BY e.employee_type;
```



| EMPLOYEE_TYPE | COUNT |
|---|---|
| P | 1 |
| F | 2 |

9. Top 2 employees who delivers the more number of packages

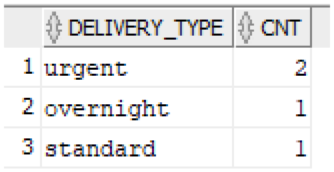
```
SELECT * FROM employee WHERE ssn in
(SELECT performed_by FROM
(SELECT performed_by,count(id) AS cnt
FROM transaction
GROUP BY performed_by
ORDER BY cnt desc)
WHERE rownum<=2);
```



| EMPLOYEE_TYPE | SSN | NAME | ADDREESS_LINE | CITY | COUNTRY | STATE | ZIP_CODE | STORE | SALARY | HOURLY_RATE |
|---|---|---|---|---|---|---|---|---|---|---|
| F | 102354698 | Robb Stark | 7791 E Osborn Rd | Boston | USA | MA | 89752 | 1 | 90000 | (null) |
| F | 987165432 | John Snow | Boston MA. 1185 Boylston St. | Boston | USA | MA | 02215 | 1 | 120000 | (null) |

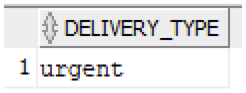
10. Display number of transactions for each delivery type and what is the most frequent delivery type customers choose?

```
SELECT delivery_type,count(id) AS cnt
FROM transaction
GROUP BY delivery_type
ORDER BY cnt DESC;
```



|  | DELIVERY_TYPE | CNT |
|---|---|---|
| 1 | urgent | 2 |
| 2 | overnight | 1 |
| 3 | standard | 1 |

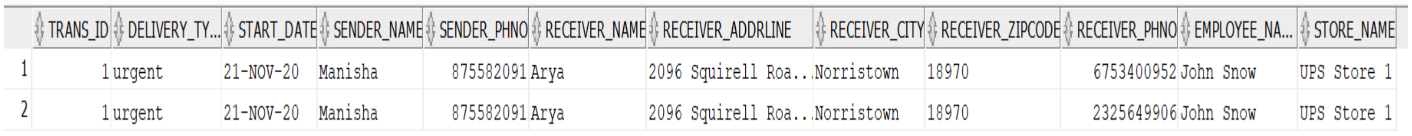
```
SELECT delivery_type FROM
(SELECT delivery_type,count(id) AS cnt
FROM transaction
GROUP BY delivery_type
ORDER BY cnt DESC)
WHERE rownum=1;
```



| | DELIVERY_TYPE |
|---|---|
| 1 | urgent |

11. Fetch the transactions, customer and employee details of packages on a particular date:

```
SELECT trans.id AS trans_id, trans.delivery_type,trans.start_date, cust_sender.name AS
sender_name, cust_sender_phno.phone_number AS sender_phno, cust_receiver.name AS
receiver_name,cust_receiver.addreess_line AS receiver_addrline ,cust_receiver.city AS
receiver_city,cust_receiver.zip_code AS receiver_zipcode,cust_receiver_phno.phone_number
AS receiver_phno, emp.name AS employee_name,str.name AS store_name
FROM transaction trans,customer cust_sender,customer cust_receiver,employee emp, store str,
customer_phone_numbers cust_sender_phno,customer_phone_numbers cust_receiver_phno
WHERE trans.performed_by=emp.ssn AND trans.sender=cust_sender.id AND
trans.receiver=cust_receiver.id AND emp.store=str.id AND
cust_sender.id=cust_sender_phno.customer_id AND
cust_receiver.id=cust_receiver_phno.customer_id AND trans.delivery_date=to_date(' 22-Nov-
20');
```



| | TRANS_ID | DELIVERY_TY... | START_DATE | SENDER_NAME | SENDER_PHNO | RECEIVER_NAME | RECEIVER_ADDRLINE | RECEIVER_CITY | RECEIVER_ZIPCODE | RECEIVER_PHNO | EMPLOYEE_NA... | STORE_NAME |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | urgent | 21-NOV-20 | Manisha | 875582091 | Arya | 2096 Squirell Roa... | Norristown | 18970 | 6753400952 | John Snow | UPS Store 1 |
| 2 | 1 | urgent | 21-NOV-20 | Manisha | 875582091 | Arya | 2096 Squirell Roa... | Norristown | 18970 | 2325649906 | John Snow | UPS Store 1 |

**<u>Stored Procedure:</u>**

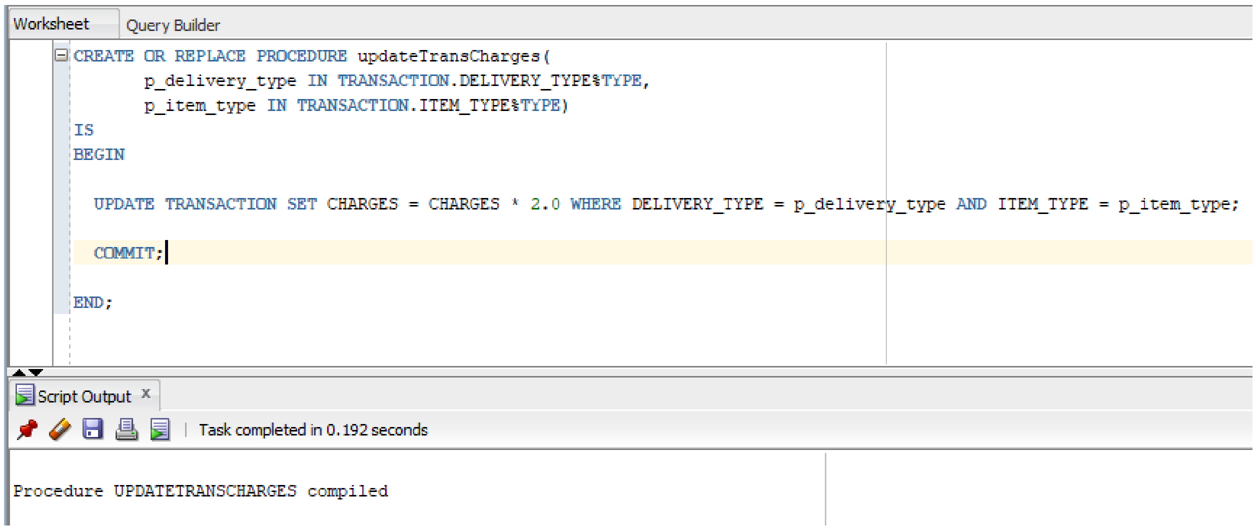
1. Find all the transactions where delivery type is ‘urgent’ and item_type is ‘document’, and double their delivery charges.

<u>Procedure Definition:</u>

```
CREATE OR REPLACE PROCEDURE updateTransCharges(
     p_delivery_type IN TRANSACTION.DELIVERY_TYPE%TYPE,
     p_item_type IN TRANSACTION.ITEM_TYPE%TYPE)
IS
BEGIN
  UPDATE TRANSACTION SET CHARGES = CHARGES * 2.0 WHERE DELIVERY_TYPE
= p_delivery_type AND ITEM_TYPE = p_item_type;
  COMMIT;
END;
```

<u>Procedure invocation:</u>

```
BEGIN
   updateTransCharges('urgent','document');
END;
```



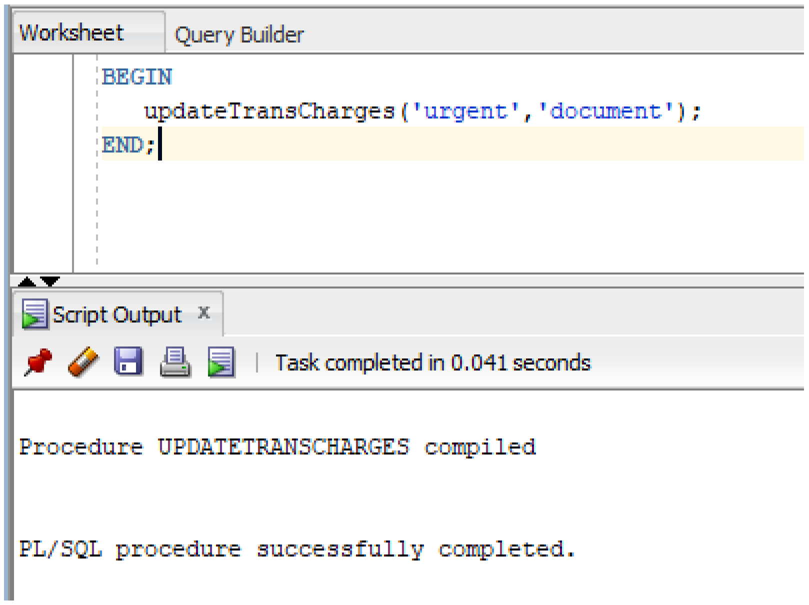
```
CREATE OR REPLACE PROCEDURE updateTransCharges(
       p_delivery_type IN TRANSACTION.DELIVERY_TYPE%TYPE,
       p_item_type IN TRANSACTION.ITEM_TYPE%TYPE)
IS
BEGIN

  UPDATE TRANSACTION SET CHARGES = CHARGES * 2.0 WHERE DELIVERY_TYPE = p_delivery_type AND ITEM_TYPE = p_item_type;

  COMMIT;

END;
```

```
Procedure UPDATETRANSCHARGES compiled
```

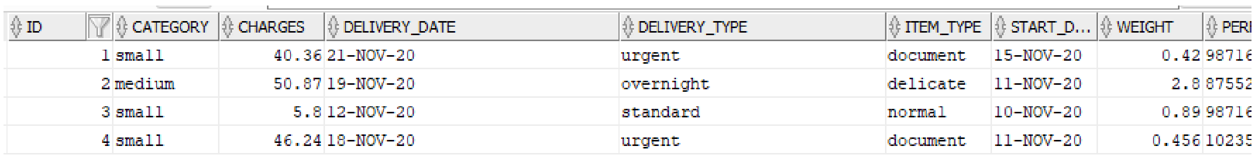


```
BEGIN
   updateTransCharges('urgent','document');
END;
```

Task completed in 0.041 seconds

```
Procedure UPDATETRANSCHARGES compiled


PL/SQL procedure successfully completed.
```



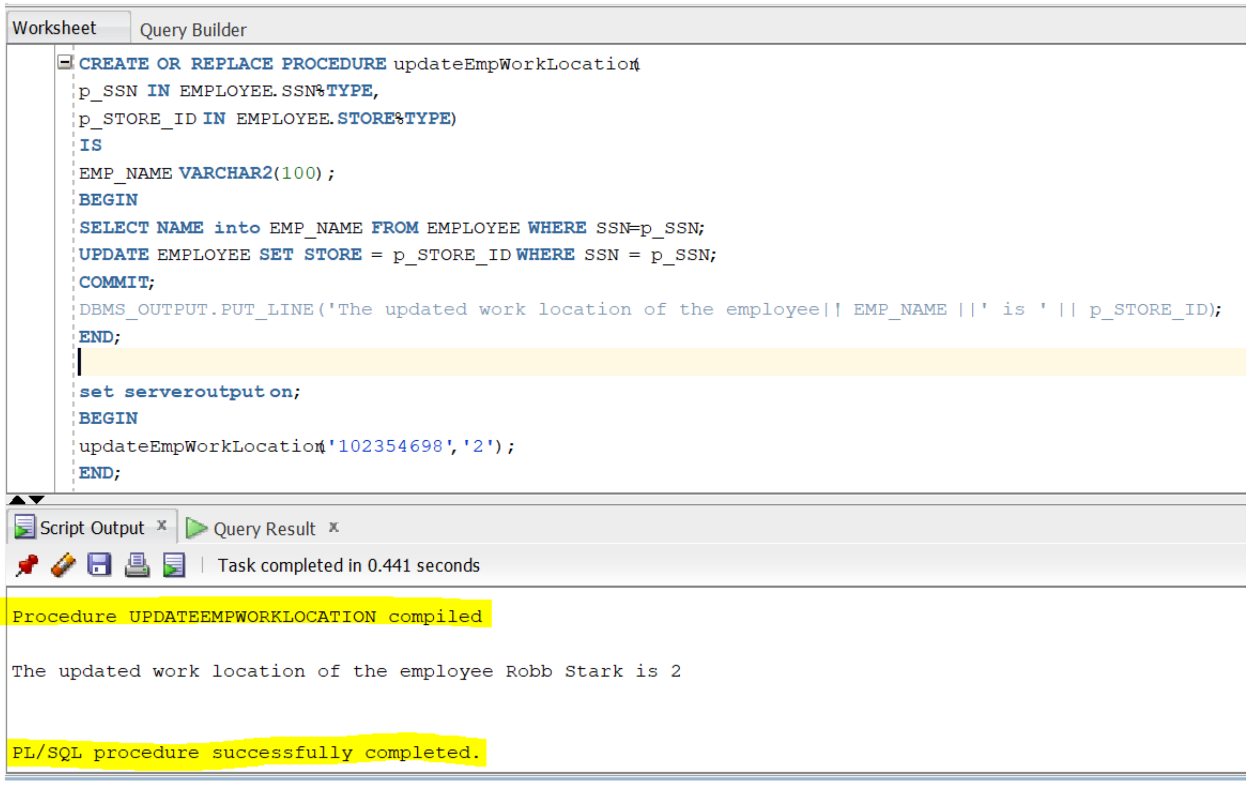
| ID | CATEGORY | CHARGES | DELIVERY_DATE | DELIVERY_TYPE | ITEM_TYPE | START_D... | WEIGHT | PERI |
|---|---|---|---|---|---|---|---|---|
| 1 | small | 40.36 | 21-NOV-20 | urgent | document | 15-NOV-20 | 0.42 | 98716 |
| 2 | medium | 50.87 | 19-NOV-20 | overnight | delicate | 11-NOV-20 | 2.8 | 87552 |
| 3 | small | 5.8 | 12-NOV-20 | standard | normal | 10-NOV-20 | 0.89 | 98716 |
| 4 | small | 46.24 | 18-NOV-20 | urgent | document | 11-NOV-20 | 0.456 | 10235 |

2. Update the work location for an employee

Procedure Definition:

*CREATE OR REPLACE PROCEDURE updateEmpWorkLocation(*
*p_SSN IN EMPLOYEE.SSN%TYPE,*
*p_STORE_ID IN EMPLOYEE.STORE%TYPE)*
*IS*
*EMP_NAME VARCHAR2(100);*
*BEGIN*
*SELECT NAME into EMP_NAME FROM EMPLOYEE WHERE SSN=p_SSN;*
*UPDATE EMPLOYEE SET STORE = p_STORE_ID WHERE SSN = p_SSN;*
*COMMIT;*
*DBMS_OUTPUT.PUT_LINE('The updated work location of the employee ' || EMP_NAME ||' is ' || p_STORE_ID);*
*END;*

Procedure invocation:
*set serveroutput on;*
*BEGIN*
*updateEmpWorkLocation('102354698','2');*
*END;*



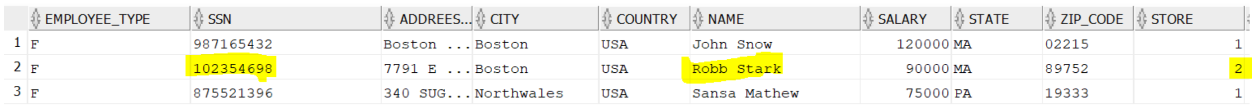
```
CREATE OR REPLACE PROCEDURE updateEmpWorkLocation(
p_SSN IN EMPLOYEE.SSN%TYPE,
p_STORE_ID IN EMPLOYEE.STORE%TYPE)
IS
EMP_NAME VARCHAR2(100);
BEGIN
SELECT NAME into EMP_NAME FROM EMPLOYEE WHERE SSN=p_SSN;
UPDATE EMPLOYEE SET STORE = p_STORE_ID WHERE SSN = p_SSN;
COMMIT;
DBMS_OUTPUT.PUT_LINE('The updated work location of the employee'|| EMP_NAME ||' is ' || p_STORE_ID);
END;

set serveroutput on;
BEGIN
updateEmpWorkLocation('102354698','2');
END;
```

```
Procedure UPDATEEMPWORKLOCATION compiled

The updated work location of the employee Robb Stark is 2


PL/SQL procedure successfully completed.
```



| | EMPLOYEE_TYPE | SSN | ADDREES... | CITY | COUNTRY | NAME | SALARY | STATE | ZIP_CODE | STORE |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | F | 987165432 | Boston ... | Boston | USA | John Snow | 120000 | MA | 02215 | 1 |
| 2 | F | 102354698 | 7791 E ... | Boston | USA | Robb Stark | 90000 | MA | 89752 | 2 |
| 3 | F | 875521396 | 340 SUG... | Northwales | USA | Sansa Mathew | 75000 | PA | 19333 | 1 |

## Application Code

Git-Hub: https://github.com/TmanishaT/CS8490-Database-Systems

Connection properties:

```
spring.jpa.generate-ddl=true
spring.jpa.hibernate.ddl-auto=update

spring.datasource.url=jdbc:oracle:thin:@45.79.135.253:1521/xe
spring.datasource.username= mtanwar
spring.datasource.password= 02205833
spring.datasource.driver-class-name=oracle.jdbc.OracleDriver
## this shows the sql actions in the terminal logs
spring.jpa.show-sql=true
```

```
spring.jpa.generate-ddl=true
spring.jpa.hibernate.ddl-auto=update

spring.datasource.url=jdbc:oracle:thin:@45.79.135.253:1521/xe
spring.datasource.username=svemugun
spring.datasource.password=02158051
spring.datasource.driver-class-name=oracle.jdbc.OracleDriver
## this shows the sql actions in the terminal logs
spring.jpa.show-sql=true
```

Repositories:

**Transaction Repository**

```
@Repository
public interface TransactionRepository extends JpaRepository<Transaction,
Integer>{
   @Query("SELECT t FROM Transaction t WHERE t.sender=:senderId")
   public List<Transaction> findAllBySenderId(@Param("senderId") Integer
senderId);

   @Query("SELECT t FROM Transaction t WHERE t.receiver=:receiverId")
   public List<Transaction> findAllByReceiverId(@Param("receiverId") Integer
receiverId);

   @Query("SELECT t FROM Transaction t WHERE t.itemType=:itemType")
   public List<Transaction> findAllByItemType(@Param("itemType") String
itemType);

   @Query("SELECT t FROM Transaction t WHERE t.startDate=:startDate")
```

```
    public List<Transaction> findAllByStartDate(@Param("startDate") Date
startDate);

    @Query("SELECT t FROM Transaction t WHERE t.deliveryDate=:deliveryDate")
    public List<Transaction> findAllByDeliveryDate(@Param("deliveryDate") Date
deliveryDate);

    @Query("SELECT t FROM Transaction t WHERE t.deliveryType=:deliveryType")
    public List<Transaction> findAllByDeliveryType(@Param("deliveryType") String
deliveryType);

    @Query("SELECT t FROM Transaction t WHERE t.category=:itemCategory")
    public List<Transaction> findAllByItemCategory(@Param("itemCategory") String
itemCategory);

    @Query("SELECT t FROM Transaction t WHERE t.store=:storeId")
    public List<Transaction> findAllByStoreId(@Param("storeId") Integer storeId);

    @Query("SELECT t FROM Transaction t WHERE t.performedBy=:performedBy")
    public List<Transaction> findAllByManagerId(@Param("performedBy") Integer
performedBy);
    }
```

### Store Repository

```
@Repository
public interface StoreRepository extends JpaRepository<Store, Integer> {

    @Query("SELECT s FROM Store s WHERE s.name=:storeName")
    List<Store> findByName(String storeName);

}
```

### Employee Repository

```
@Repository
public interface EmployeeRepository extends JpaRepository<Employee, Long> {

    @Query("SELECT e FROM Employee e WHERE e.name=:name")
    List<Employee> findByName(String name);

    @Query("SELECT e FROM Employee e WHERE e.store=:storeId")
    List<Employee> findByStoreId(Integer storeId);

    @Query("SELECT e FROM Employee e WHERE e.SSN IN (:employeeIds)")
    List<Employee> findBySSNList(List<Long> employeeIds);


}
```

### Customer Repository

```
@Repository
```

```
public interface CustomerRepository extends JpaRepository<Customer, Integer>{
   @Query
   ("SELECT c FROM Customer c WHERE LOWER(c.name) = LOWER(:custName)")
    public List<Customer> findByCustomerName(@Param("custName") String custName);

}
```