Q1)

```
CREATE TABLE employee (
   ID INT PRIMARY KEY,                  -- Primary key for employee
   person_name VARCHAR(100) NOT NULL,     -- Employee's name
   street VARCHAR(100),                  -- Street address
   city VARCHAR(100)                    -- City
);

-- Create the company table
CREATE TABLE company (
   company_name  VARCHAR(100) PRIMARY KEY,  -- Primary key for company
   city VARCHAR(100)                    -- City where the company is located
);

-- Create the works table
CREATE TABLE works (
   ID INT,                          -- Foreign key to employee table
   company_name VARCHAR(100),             -- Foreign key to company table
   salary DECIMAL(10, 2),                -- Salary of the employee
   PRIMARY KEY (ID, company_name),        -- Composite primary key
   FOREIGN KEY (ID) REFERENCES employee(ID) ON DELETE CASCADE,
   FOREIGN KEY (company_name) REFERENCES company(company_name) ON DELETE CASCADE
);

-- Create the manages table
CREATE TABLE manages (
   ID INT,                          -- Foreign key to employee table
   manager_id INT,                     -- Foreign key to employee table (self-referential)
```

```
  PRIMARY KEY (ID, manager_id),          -- Composite primary key
   FOREIGN KEY (ID) REFERENCES employee(ID) ON DELETE CASCADE,
   FOREIGN KEY (manager_id) REFERENCES employee(ID) ON DELETE SET NULL
);
```

Q2)

a)

```
SELECT DISTINCT depositor.ID
FROM depositor
WHERE depositor.ID NOT IN (
   SELECT borrower.ID
   FROM borrower
);
```

b)

```
SELECT customer.ID
FROM customer
WHERE customer.customer_street = (
   SELECT customer.customer_street
   FROM customer
   WHERE customer.ID = '12345'
)
AND customer.customer_city = (
   SELECT customer.customer_city
   FROM customer
   WHERE customer.ID = '12345'
);
```

c)

```
SELECT DISTINCT branch.branch_name
FROM branch
JOIN account ON branch.branch_name = account.branch_name
JOIN depositor ON account.account_number = depositor.account_number
JOIN customer ON depositor.ID = customer.ID
WHERE customer.customer_city = 'Harrison';
```

Q3)

A)

```
SELECT day, qty,
SUM(qty) OVER (ORDER BY day) AS cumQty
FROM demand;
```

B)

```
WITH RankedDays AS (
   SELECT product,  day,  qty,
   ROW_NUMBER() OVER (PARTITION BY product ORDER BY qty ASC) AS RN
   FROM demand
)
SELECT  product, day,   qty, RN
FROM   RankedDays
WHERE   RN <= 2;
```