1. Here's the SQL commands to create the tables with appropriate integrity constraints and enter some sample data:

Create Customer table

CREATE TABLE Customer (

cust\_id INT PRIMARY KEY,

cust\_name VARCHAR(50),

annual\_revenue DECIMAL,

cust\_type VARCHAR(20),

CONSTRAINT chk\_cust\_id CHECK (cust\_id >= 100),

CONSTRAINT chk\_cust\_type CHECK (cust\_type IN ('manufacturer', 'wholesaler', 'retailer', 'distributor'))

);

Create Truck table

CREATE TABLE Truck (

truck\_no INT PRIMARY KEY,

driver\_name VARCHAR(50)

);

Create City table

CREATE TABLE City (

city\_name VARCHAR(50) PRIMARY KEY,

population INT

);

Create Shipment table

CREATE TABLE Shipment (

shipment\_no INT PRIMARY KEY,

cust\_id INT,

weight DECIMAL,

truck\_no INT,

start\_city VARCHAR(50),

destination\_city VARCHAR(50),

FOREIGN KEY (cust\_id) REFERENCES Customer(cust\_id),

FOREIGN KEY (truck\_no) REFERENCES Truck(truck\_no),

FOREIGN KEY (start\_city) REFERENCES City(city\_name),

FOREIGN KEY (destination\_city) REFERENCES City(city\_name)

);

Inserting sample data into tables

INSERT INTO Customer (cust\_id, cust\_name, annual\_revenue, cust\_type) VALUES

(100, 'Customer1', 100000, 'manufacturer'),

(101, 'Customer2', 200000, 'wholesaler'),

(102, 'Customer3', 300000, 'retailer');

INSERT INTO Truck (truck\_no, driver\_name) VALUES

(1, 'John Doe'),

(2, 'Jane Smith'),

(3, 'Michael Johnson');

INSERT INTO City (city\_name, population) VALUES

('CityA', 1000000),

('CityB', 500000),

('CityC', 750000);

INSERT INTO Shipment (shipment\_no, cust\_id, weight, truck\_no, start\_city, destination\_city) VALUES

(1, 100, 500, 1, 'CityA', 'CityA'),

(2, 101, 300, 2, 'CityB', 'CityC'),

(3, 102, 400, 3, 'CityC', 'CityB');

Now, here are the SQL commands for your queries:

a) Give the details for those shipments where the start-city and destination-city are the same.

SELECT \* FROM Shipment WHERE start\_city = destination\_city;

b) Give the driver names who participated in the maximum number of shipments.

SELECT driver\_name, COUNT(\*) AS num\_shipments

FROM Shipment

JOIN Truck ON Shipment.truck\_no = Truck.truck\_no

GROUP BY driver\_name

ORDER BY num\_shipments DESC

LIMIT 1;

c) Give the name of the city that never appears as a destination city.

SELECT city\_name

FROM City

WHERE city\_name NOT IN (SELECT destination\_city FROM Shipment);

d) Give the name of the cities whose population is more than the average of all the cities.

SELECT city\_name

FROM City

WHERE population > (SELECT AVG(population) FROM City);

e) Give the shipment details where the starting character of the customer name and the last character of the driver name are 'A'.

SELECT \*

FROM Shipment

JOIN Customer ON Shipment.cust\_id = Customer.cust\_id

JOIN Truck ON Shipment.truck\_no = Truck.truck\_no

WHERE LEFT(Customer.cust\_name, 1) = 'A' AND RIGHT(Truck.driver\_name, 1) = 'A';

2. Here's the SQL commands to create the tables with appropriate integrity constraints and enter some sample data:

Create Employee table

CREATE TABLE Employee (

empno INT PRIMARY KEY,

emp\_name VARCHAR(50),

salary DECIMAL,

hiredate DATE,

city VARCHAR(50)

);

Create Department table

CREATE TABLE Department (

deptno INT PRIMARY KEY,

dname VARCHAR(50),

city VARCHAR(50)

);

Create Manager table

CREATE TABLE Manager (

mgrno INT PRIMARY KEY,

deptno INT,

manager\_name VARCHAR(50),

salary DECIMAL,

hiredate DATE,

city VARCHAR(50),

FOREIGN KEY (deptno) REFERENCES Department(deptno)

);

Create Empmgr table

CREATE TABLE Empmgr (

empno INT,

mgrno INT,

FOREIGN KEY (empno) REFERENCES Employee(empno),

FOREIGN KEY (mgrno) REFERENCES Manager(mgrno)

);

Inserting sample data into tables

INSERT INTO Employee (empno, emp\_name, salary, hiredate, city) VALUES

(1, 'Employee1', 50000, '2005-01-15', 'CityA'),

(2, 'Employee2', 60000, '2006-03-20', 'CityB'),

(3, 'Employee3', 70000, '2006-07-10', 'CityC');

INSERT INTO Department (deptno, dname, city) VALUES

(1, 'Department1', 'CityA'),

(2, 'Department2', 'CityB'),

(3, 'Department3', 'CityC');

INSERT INTO Manager (mgrno, deptno, manager\_name, salary, hiredate, city) VALUES

(101, 1, 'Manager1', 80000, '2004-05-10', 'CityA'),

(102, 2, 'Manager2', 90000, '2003-09-25', 'CityB'),

(103, 3, 'Manager3', 100000, '2002-12-15', 'CityC');

INSERT INTO Empmgr (empno, mgrno) VALUES

(1, 101),

(2, 102),

(3, 103);

Now, here are the SQL commands for your queries:

a) Give the employee names who earn more than their manager.

SELECT e.emp\_name

FROM Employee e

JOIN Empmgr em ON e.empno = em.empno

JOIN Employee mgr ON em.mgrno = mgr.empno

WHERE e.salary > mgr.salary;

b) Give the name of the managers who have no employee under their supervision from the same city as of them.

SELECT manager\_name

FROM Manager

WHERE mgrno NOT IN (

SELECT DISTINCT mgrno

FROM Empmgr

JOIN Employee ON Empmgr.empno = Employee.empno

JOIN Manager ON Empmgr.mgrno = Manager.mgrno

WHERE Employee.city = Manager.city

);

c) Give the name of the manager who has the maximum number of employees under his supervision.

SELECT manager\_name

FROM Manager

WHERE mgrno IN (

SELECT mgrno

FROM Empmgr

GROUP BY mgrno

ORDER BY COUNT(empno) DESC

LIMIT 1

);

d) Give the name of the employees who have been hired in the year 2006.

SELECT emp\_name

FROM Employee

WHERE EXTRACT(YEAR FROM hiredate) = 2006;

e) Give the name of the city where no manager is located.

SELECT DISTINCT city

FROM Department

WHERE city NOT IN (SELECT DISTINCT city FROM Manager);

3. Here's the SQL commands to create the tables with appropriate integrity constraints and enter some sample data:

Create Employee table

CREATE TABLE Employee (

empno INT PRIMARY KEY,

empname VARCHAR(50),

city VARCHAR(50)

);

Create Project table

CREATE TABLE Project (

pno INT PRIMARY KEY,

pname VARCHAR(50)

);

Create Part table

CREATE TABLE Part (

partno INT PRIMARY KEY,

partname VARCHAR(50),

color VARCHAR(50)

);

Create Use table

CREATE TABLE Use (

pno INT,

partno INT,

FOREIGN KEY (pno) REFERENCES Project(pno),

FOREIGN KEY (partno) REFERENCES Part(partno)

);

Create Works table

CREATE TABLE Works (

empno INT,

pno INT,

FOREIGN KEY (empno) REFERENCES Employee(empno),

FOREIGN KEY (pno) REFERENCES Project(pno)

);

Inserting sample data into tables

INSERT INTO Employee (empno, empname, city) VALUES

(1, 'John Doe', 'CityA'),

(2, 'Jane Smith', 'CityB'),

(3, 'Sarah Johnson', 'CityC');

INSERT INTO Project (pno, pname) VALUES

(101, 'Project1'),

(102, 'Project2'),

(103, 'Project3');

INSERT INTO Part (partno, partname, color) VALUES

(201, 'Part1', 'Red'),

(202, 'Part2', 'Blue'),

(203, 'Part3', 'Green');

INSERT INTO Use (pno, partno) VALUES

(101, 201),

(101, 202),

(102, 201),

(102, 203),

(103, 201),

(103, 202),

(103, 203);

INSERT INTO Works (empno, pno) VALUES

(1, 101),

(2, 102),

(3, 103),

(1, 102),

(2, 103);

Now, here are the SQL commands for your queries:

a) Give the name of the employees who are working on more than 1 project.

SELECT empname

FROM Employee

WHERE empno IN (

SELECT empno

FROM Works

GROUP BY empno

HAVING COUNT(pno) > 1

);

b) Give the name of the city where the maximum employees are located.

SELECT city

FROM Employee

GROUP BY city

ORDER BY COUNT(empno) DESC

LIMIT 1;

c) Give the part names, which are used in all the projects.

SELECT partname

FROM Part

WHERE partno IN (

SELECT partno

FROM Use

GROUP BY partno

HAVING COUNT(DISTINCT pno) = (SELECT COUNT(\*) FROM Project)

);

d) Give the name of the projects in which no employees are working whose name starts with ‘S’.

SELECT pname

FROM Project

WHERE pno NOT IN (

SELECT pno

FROM Works

JOIN Employee ON Works.empno = Employee.empno

WHERE empname LIKE 'S%'

);

e) Give the name of the project which uses the maximum number of red parts.

SELECT pname

FROM Project

WHERE pno IN (

SELECT pno

FROM Use

JOIN Part ON Use.partno = Part.partno

WHERE color = 'Red'

GROUP BY pno

ORDER BY COUNT(\*) DESC

LIMIT 1

);

f) Give the name of the projects which have used exactly 3 parts.

SELECT pname

FROM Project

WHERE pno IN (

SELECT pno

FROM Use

GROUP BY pno

HAVING COUNT(\*) = 3

);

4. Here's the SQL commands to create the tables with appropriate integrity constraints and enter some sample data:

Create Branch table

CREATE TABLE Branch (

branch\_name VARCHAR(50) PRIMARY KEY,

branch\_city VARCHAR(50),

assets DECIMAL

);

Create Customer table

CREATE TABLE Customer (

customer\_name VARCHAR(50) PRIMARY KEY,

street VARCHAR(100),

city VARCHAR(50)

);

Create Loan table

CREATE TABLE Loan (

branch\_name VARCHAR(50),

loan\_no INT PRIMARY KEY,

amount DECIMAL,

FOREIGN KEY (branch\_name) REFERENCES Branch(branch\_name)

);

Create Borrower table

CREATE TABLE Borrower (

customer\_name VARCHAR(50),

loan\_no INT,

PRIMARY KEY (customer\_name, loan\_no),

FOREIGN KEY (customer\_name) REFERENCES Customer(customer\_name),

FOREIGN KEY (loan\_no) REFERENCES Loan(loan\_no)

);

Create Account table

CREATE TABLE Account (

branch\_name VARCHAR(50),

account\_no INT PRIMARY KEY,

balance DECIMAL,

FOREIGN KEY (branch\_name) REFERENCES Branch(branch\_name)

);

Create Depositor table

CREATE TABLE Depositor (

customer\_name VARCHAR(50),

account\_no INT,

PRIMARY KEY (customer\_name, account\_no),

FOREIGN KEY (customer\_name) REFERENCES Customer(customer\_name),

FOREIGN KEY (account\_no) REFERENCES Account(account\_no)

);

Inserting sample data into tables

INSERT INTO Branch (branch\_name, branch\_city, assets) VALUES

('Branch1', 'CityA', 500000),

('Branch2', 'CityB', 750000),

('Branch3', 'CityC', 1000000);

INSERT INTO Customer (customer\_name, street, city) VALUES

('Customer1', 'Street1', 'CityA'),

('Customer2', 'Street2', 'CityB'),

('Customer3', 'Street3', 'CityC');

INSERT INTO Loan (branch\_name, loan\_no, amount) VALUES

('Branch1', 101, 15000),

('Branch2', 102, 20000),

('Branch3', 103, 25000);

INSERT INTO Borrower (customer\_name, loan\_no) VALUES

('Customer1', 101),

('Customer2', 102),

('Customer3', 103);

INSERT INTO Account (branch\_name, account\_no, balance) VALUES

('Branch1', 201, 10000),

('Branch2', 202, 30000),

('Branch3', 203, 40000);

INSERT INTO Depositor (customer\_name, account\_no) VALUES

('Customer1', 201),

('Customer2', 202),

('Customer3', 203);

Now, here are the SQL commands for your queries:

a) Give the customer name who has an account but not any loan.

SELECT customer\_name

FROM Customer

WHERE customer\_name IN (

SELECT customer\_name

FROM Depositor

EXCEPT

SELECT customer\_name

FROM Borrower

);

b) Give the name of the customer who has either (10000-20000) or (30000-40000) in their account.

SELECT customer\_name

FROM Customer

WHERE customer\_name IN (

SELECT customer\_name

FROM Depositor

JOIN Account ON Depositor.account\_no = Account.account\_no

WHERE balance BETWEEN 10000 AND 20000

UNION

SELECT customer\_name

FROM Depositor

JOIN Account ON Depositor.account\_no = Account.account\_no

WHERE balance BETWEEN 30000 AND 40000

);

c) Which branch has issued the maximum amount of loan?

SELECT branch\_name

FROM Loan

GROUP BY branch\_name

ORDER BY SUM(amount) DESC

LIMIT 1;

d) Give the name of the customer who has the maximum balance in their account.

SELECT customer\_name

FROM Account

JOIN Depositor ON Account.account\_no = Depositor.account\_no

GROUP BY customer\_name

ORDER BY MAX(balance) DESC

LIMIT 1;

e) Give the name of the customer who has opened an account in a city in which he is not located.

SELECT customer\_name

FROM Customer

WHERE city NOT IN (

SELECT city

FROM Account

JOIN Branch ON Account.branch\_name = Branch.branch\_name

JOIN Customer ON Branch.branch\_city = Customer.city

);

5. Here's the SQL commands to create the tables with appropriate integrity constraints and enter some sample data:

Create Aircraft table

CREATE TABLE Aircraft (

aid INT PRIMARY KEY,

type VARCHAR(50)

);

Create Flights table

CREATE TABLE Flights (

flno INT PRIMARY KEY,

aid INT,

company VARCHAR(50),

FOREIGN KEY (aid) REFERENCES Aircraft(aid)

);

Create Employees table

CREATE TABLE Employees (

eid INT PRIMARY KEY,

ename VARCHAR(50),

salary DECIMAL

);

Create Certified table

CREATE TABLE Certified (

eid INT,

aid INT,

PRIMARY KEY (eid, aid),

FOREIGN KEY (eid) REFERENCES Employees(eid),

FOREIGN KEY (aid) REFERENCES Aircraft(aid)

);

Create Schedule table

CREATE TABLE Schedule (

eid INT,

flno INT,

source\_city VARCHAR(50),

destination\_city VARCHAR(50),

date DATE,

FOREIGN KEY (eid) REFERENCES Employees(eid),

FOREIGN KEY (flno) REFERENCES Flights(flno)

);

Inserting sample data into tables

INSERT INTO Aircraft (aid, type) VALUES

(1, 'Type1'),

(2, 'Type2'),

(3, 'Type3');

INSERT INTO Flights (flno, aid, company) VALUES

(101, 1, 'Company1'),

(102, 2, 'Company2'),

(103, 3, 'Company3');

INSERT INTO Employees (eid, ename, salary) VALUES

(1, 'John Doe', 50000),

(2, 'Jane Smith', 60000),

(3, 'Michael Johnson', 70000);

INSERT INTO Certified (eid, aid) VALUES

(1, 1),

(1, 2),

(2, 2),

(3, 3);

INSERT INTO Schedule (eid, flno, source\_city, destination\_city, date) VALUES

(1, 101, 'CityA', 'CityB', '2024-06-01'),

(2, 102, 'CityB', 'CityC', '2024-06-02'),

(3, 103, 'CityC', 'CityD', '2024-06-03');

Now, here are the SQL commands for your queries:

a) Give the name of the employee who has certification on more than 1 type of aircraft.

SELECT ename

FROM Employees

WHERE eid IN (

SELECT eid

FROM Certified

GROUP BY eid

HAVING COUNT(DISTINCT aid) > 1

);

b) Give the name of the employee who has no certification.

SELECT ename

FROM Employees

WHERE eid NOT IN (SELECT eid FROM Certified);

c) Give the name of the city where the maximum flights take off (source\_city).

SELECT source\_city

FROM Schedule

GROUP BY source\_city

ORDER BY COUNT(\*) DESC

LIMIT 1;

d) Give the name of the company that spends the maximum salary for their employees.

SELECT company

FROM Flights

JOIN Schedule ON Flights.flno = Schedule.flno

JOIN Employees ON Schedule.eid = Employees.eid

GROUP BY company

ORDER BY SUM(salary) DESC

LIMIT 1;

e) Give the name of the company that has the maximum types of flights.

SELECT company

FROM Flights

GROUP BY company

HAVING COUNT(DISTINCT aid) = (

SELECT MAX(cnt)

FROM (

SELECT COUNT(DISTINCT aid) as cnt

FROM Flights

GROUP BY company

) AS max\_types

);