```
-- entities
CREATE TABLE Service (
    ABN BIGINT PRIMARY KEY,
    ServiceName VARCHAR(50),
    Owed DECIMAL,
    Email VARCHAR(50)
);

CREATE TABLE Employee (
    EmpID INTEGER PRIMARY KEY,
    Name VARCHAR(100),
    HomeAddress VARCHAR(100),
    DateOfBirth DATE,
    DepartmentName VARCHAR(100) NOT NULL
);

CREATE TABLE Model(
    Manufacturer VARCHAR(50),
    ModelNumber VARCHAR(50),
    Weight FLOAT,
    Description VARCHAR(200),
    PRIMARY KEY(Manufacturer, ModelNumber)
);




/*
There is a device which is not a phone.
Not prevented, phone IsA device indicates that a phone must be a device but a
device is not necessarily a phone.
No reference to phone table in device table allows it.

A particular device was issued to three employees.
Prevented, a thin arrow from device to employee indicates that a device can be
issued to at most one employee at one time.
    DeviceID INTEGER PRIMARY KEY,
    IssuedTo INTEGER REFERENCES Employee(EmpID) ON DELETE SET NULL
unique primary key DeviceID and reference to EmpID prevent this

A particular employee was issued two devices.
Not prevented, a thin line from employee to device indicates that an employee can
be issued multiple devices.
    IssuedTo INTEGER REFERENCES Employee(EmpID) ON DELETE SET NULL
no unique clause in the statement allows it

A particular device was not issued to any employee.
Not prevented, a thin arrow from device to IssedTo means 0 or 1 device can be
issued to an employee.
    IssuedTo INTEGER REFERENCES Employee(EmpID) ON DELETE SET NULL
No not null clause in the statement allows it

A particular employee has not been issued with any device.
Not prevented, a thin line from Employee to IssuedTo means an Employee can be
issued 0 to many devices.
No clause requires all EmpID must appear at least once in Device(IssuedTo)

A particular device was issued to one employee.
Not prevented, an arrow from Device to IssuedTo means a particular device can at
```

```
most be Issued to one employee.
    IssuedTo INTEGER REFERENCES Employee(EmpID) ON DELETE SET NULL
Allows it

*/

CREATE TABLE Device(
    DeviceID INTEGER PRIMARY KEY,
    PurchaseDate DATE,
    PurchaseCost DECIMAL,
    SerialNumber VARCHAR(50),
    Manufacturer VARCHAR(50) NOT NULL,
    ModelNumber VARCHAR(50) NOT NULL,
    FOREIGN KEY (Manufacturer, ModelNumber) REFERENCES Model(Manufacturer,
ModelNumber) ON DELETE CASCADE,
    IssuedTo INTEGER REFERENCES Employee(EmpID) ON DELETE SET NULL
);


CREATE TABLE Phone (
    DeviceID INTEGER PRIMARY KEY REFERENCES Device(DeviceID) ON DELETE CASCADE,
    Carrier VARCHAR(50),
    Number BIGINT,
    Plan VARCHAR(50)
);




/*
A particular device was repaired twice.
Not prevented, a thin line from device to repair indicates a device can have
multiple repairs.
    DeviceID INTEGER NOT NULL REFERENCES Device(DeviceID)
no unique clause in statement allows it

A particular device was never repaired.
Not prevented, a thin line from device to repair indicates a device can have zero
repair.
    DeviceID INTEGER NOT NULL REFERENCES Device(DeviceID)
no unique clause in statement allows it

One repair fixed two devices.
Prevented, a thick arrow from repair to device indicates one repair can only repair
at least and at most one device.
    RepairID INTEGER PRIMARY KEY
    DeviceID INTEGER NOT NULL REFERENCES Device(DeviceID)
unique RepairID and forign key to DeviceID prevent it

A particular repair is not done on any device.
Prevented, a thick arrow from repair to device indicates one repair can only repair
at least and at most one device.
    DeviceID INTEGER NOT NULL REFERENCES Device(DeviceID)
the not null and forign key to DeviceID prevented it
*/

CREATE TABLE Repair (
    RepairID INTEGER PRIMARY KEY,
    Cost DECIMAL,
    StartDate DATE,
    EndDate DATE,
    FaultReport VARCHAR(150),
    ABN BIGINT NOT NULL REFERENCES Service(ABN),
    DeviceID INTEGER NOT NULL REFERENCES Device(DeviceID)
);

CREATE TABLE EmployeePhoneNumbers(
    EmpID INTEGER REFERENCES Employee(EmpID) ON DELETE CASCADE,
    PhoneNumber BIGINT,
    PRIMARY KEY (EmpID, PhoneNumber)
);

CREATE TABLE Department (
    Name VARCHAR(100) PRIMARY KEY,
    Budget DECIMAL,
    EmpID INT NOT NULL
);

CREATE TABLE OfficeLocations (
    Name VARCHAR(100) REFERENCES Department(Name) ON DELETE CASCADE,
    Address VARCHAR(100),
    PRIMARY KEY (Name, Address)
);




/*
A particular device was used by three employees.
Not prevented, a thin line from Device to UsedBy means a particular device can be
used by 0 to many employees.
    PRIMARY KEY(DeviceID, Employee)
Allows many to many relationship between Device and Employee with usedby relation.
    DeviceID INTEGER REFERENCES Device(DeviceID) ON DELETE CASCADE,
No unique clause in statement allows it

A particular employee used two devices.
Not prevented, a thin line from Employee to UsedBy means a particular employee can
use 0 to many devices.
    PRIMARY KEY(DeviceID, Employee)
Allows many to many relationship between Device and Employee with usedby relation.
    Employee INTEGER REFERENCES Employee(EmpID) ON DELETE CASCADE,
No unique clause in statement allows it


A particular device was not used by any employee.
Not prevented, a thin line from Device to UsedBy that can accept zero cardinality.
    DeviceID INTEGER REFERENCES Device(DeviceID) ON DELETE CASCADE,
does not require all DeviceID from Device table must appear once in UsedBy table.
allows it

A particular employee has not used any device.
Not prevented, a thin line from Employee to UsedBy that can accept zero
cardinality.
    Employee INTEGER REFERENCES Employee(EmpID) ON DELETE CASCADE,
does not require all EmpID from Employee table must appear once in UsedBy table.
allows it
```

```
A particular device was used by one employee.
Allowed, a device can be used by 0-n employees (thin line)
    DeviceID INTEGER REFERENCES Device(DeviceID) ON DELETE CASCADE,
    Employee INTEGER REFERENCES Employee(EmpID) ON DELETE CASCADE,
allows one to one relation
*/

-- relationship
CREATE TABLE UsedBy(
    DeviceID INTEGER REFERENCES Device(DeviceID) ON DELETE CASCADE,
    Employee INTEGER REFERENCES Employee(EmpID) ON DELETE CASCADE,
    PRIMARY KEY(DeviceID, Employee)
);




/*
Two models were allocated to the same department.
Allowed, many models can be allocated to the same department (thin line)
    PRIMARY KEY (DepartmentName, Manufacturer, ModelNumber)
allows many to many relationship between Model Manufacturer and Department with
allocation relation.
    DepartmentName VARCHAR(100) REFERENCES Department(name) ON DELETE CASCADE,
no unique clause allows the same department to receive multiple models.


A particular model was allocated to three departments
This is allowed as the ER diagram allows for a model to be allocated to multiple
departments (thin line)
    PRIMARY KEY (DepartmentName, Manufacturer, ModelNumber)
allows many to many relationship between Model Manufacturer and Department with
allocation relation.
    Manufacturer VARCHAR(50),
    ModelNumber VARCHAR(50),
    FOREIGN KEY (Manufacturer, ModelNumber) REFERENCES Model (Manufacturer,
ModelNumber) ON DELETE CASCADE,
no unique clause allows the same model can be allocated to multiple department.

A particular department has not had any model allocated to it.
The ER diagram has a thin line from Model to AllocatedTo, which means it can have 0
to n models allocated to it
    DepartmentName VARCHAR(100) REFERENCES Department(name) ON DELETE CASCADE,
the statement  does not require all department names from Department table must
appear once in AllocatedTo table.


A particular model has not been allocated to any department.
Again, the thin line means could be 0, so this is possible.
    Manufacturer VARCHAR(50),
    ModelNumber VARCHAR(50),
    FOREIGN KEY (Manufacturer, ModelNumber) REFERENCES Model (Manufacturer,
ModelNumber) ON DELETE CASCADE,
the statement does not require all ModelNumber and Manufacturer from Model table
must appear once in AllocatedTo table.
*/


CREATE TABLE AllocatedTo(
    DepartmentName VARCHAR(100) REFERENCES Department(name) ON DELETE CASCADE,
    MaxNumber INTEGER,
    Manufacturer VARCHAR(50),
    ModelNumber VARCHAR(50),
    FOREIGN KEY (Manufacturer, ModelNumber) REFERENCES Model (Manufacturer,
ModelNumber) ON DELETE CASCADE,
    PRIMARY KEY (DepartmentName, Manufacturer, ModelNumber)
);

CREATE TABLE WorksIn (
    EmpID INTEGER REFERENCES Employee(EmpID) ON DELETE CASCADE DEFERRABLE,
    Name VARCHAR(100) REFERENCES Department (Name) ON DELETE CASCADE DEFERRABLE,
    PRIMARY KEY (EmpID, Name),
    Fraction VARCHAR(100)
);

-- given empid must have a row in the worksin table
ALTER TABLE Employee ADD FOREIGN KEY (DepartmentName) REFERENCES Department(Name)
DEFERRABLE,
  ADD CONSTRAINT employee_must_have_at_least_one_department FOREIGN KEY (EmpID,
DepartmentName) REFERENCES WorksIn(EmpID, Name) DEFERRABLE;

-- given name (department primary key) must have a row in the worksin table
ALTER TABLE Department ADD FOREIGN KEY (EmpID) REFERENCES Employee (EmpID)
DEFERRABLE,
  ADD CONSTRAINT department_must_have_at_least_one_employee FOREIGN KEY (EmpID,
Name) REFERENCES WorksIn(EmpID, Name) DEFERRABLE;


/*
When inserting an employee or a department, you have to append any one department
name that employee works in at the employee row, or append any one Empid from the
employee that department has at the department row.
Also, you need to insert the relationship between employees and departments into
wotksin table.

It is very likely you will need to defer forign key check

sample insert statement:

START TRANSACTION;
SET CONSTRAINTS ALL DEFERRED;
INSERT INTO Worksin
VALUES (1, 'Sales', 'chaos');
INSERT INTO Employee
VALUES (
        1,
        'John',
        '123 Main St',
        '1980-01-01',
        'Sales'
    );
INSERT INTO Department
VALUES (
        'Sales',
        100000,
        1
    );
COMMIT;
```

```
*/
```