Question 1.  [6 marks]

Consider the following schema. SIN stands for Social Insurance Number.

Relations

- Person(SIN, name)
- Doctor(SIN, speciality)
- Caresfor(doctor, patient)

Integrity constraints

- Doctor[SIN] ⊆ Person[SIN]
- Caresfor[doctor] ⊆ Doctor[SIN]
- Caresfor[patient] ⊆ Person[SIN]
- (σ_{doctor=patient} Caresfor) = ∅

Part (a)  [3 marks]

Suppose relation Caresfor has 4 tuples. What can you say about the minimum number of tuples in relation Person?

Solution:

There must be 3 tuples in Person. We can make a valid instance with that few because the schema does not prevent two people from caring for each other. Also, we can chain together CaresFor relationships, for example A cares for B and B cares for C.

Give a valid instance of the database that demonstrates your answer.

Person:

<table>
<thead>
<tr>
<th>SIN</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Barney</td>
</tr>
<tr>
<td>B</td>
<td>Wilma</td>
</tr>
<tr>
<td>C</td>
<td>BamBam</td>
</tr>
</tbody>
</table>

Doctor:

<table>
<thead>
<tr>
<th>SIN</th>
<th>specialty</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>pediatrics</td>
</tr>
<tr>
<td>B</td>
<td>pediatrics</td>
</tr>
<tr>
<td>C</td>
<td>pediatrics</td>
</tr>
</tbody>
</table>

Caresfor:

<table>
<thead>
<tr>
<th>doctor</th>
<th>patient</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
</tr>
</tbody>
</table>

Part (b)  [3 marks]

Use the notation \( R = ∅ \) to express the following new constraint: A doctor’s patient can’t be his or her doctor.

Solution:

\[
[σ_{c_1.doctor=c_2.patient∧c_2.patient=c_1.doctor}(ρ_{c_1} CaresFor × ρ_{c_2} CaresFor)] = ∅
\]
Question 2. [14 marks]

Here is part of the schema from assignment 1:

Relations

- Object(CN, date, name, description, type, length, width, height, who)
- Donor(DID, surname, firstname, address, email)
- Donation(NID, date, DID)
- Contains(NID, CN)
- Staff(SID, surname, firstname, address, email, type, date)

Integrity constraints

- Object[who] ⊆ Staff[SID]
- Contains[NID] ⊆ Donation[NID]
- Contains[CN] ⊆ Object[CN]
- Donation[DID] ⊆ Donor[DID]

Answer the following questions in relational algebra, using only the basic operators Π, σ, △, ×, , −, ρ.

Part (a) [7 marks]

For each staff member who has catalogued at least one object, find the first object that he or she catalogued. If there are ties, report them all. Report the SID and CN, as well as the DID of the donor who donated the item.

Solution:

Catalogued(SID, CN, date) := Πwho,CN,date Object
NotFirst(SID, CN, date) := Πc1.SID,c1.CN,c1.dateσc1.SID=c2.SID∧c1.date>c2.date(ρc1.Catalogued×ρc2.Catalogued)
First(SID, CN, date) := Catalogued − NotFirst
Answer(SID, CN, DID) := ΠSID,CN,DID(σContains.NID=Donation.NID(First △ Contains)×Donation)
Part (b)  [7 marks]

Find donations that meet the following conditions: they contain no items whose type is “couch”, and no
items that were catalogued by a temp (i.e., a staff member whose type is “temp”). Report the donation’s
NID and the donor’s email address.

Solution:

\[
\text{HasCouch}(NID) := \sigma_{\text{type} = \text{couch}}(\text{Contains} \bowtie \text{Object})
\]

\[
\text{HasTempCataloguedItem}(NID) := \\
\Pi_{NID} \sigma_{\text{who} = \text{SID} \land \text{Staff.type} = \text{temp}}((\text{Contains} \bowtie \text{Object}) \times \text{Staff})
\]

\[
\text{BareAnswer}(NID) := \Pi_{NID}(\text{Donation} - \text{HasCouch} - \text{HasTempCataloguedItem})
\]

\[
\text{Answer}(NID, email) := \Pi_{NID, email}(\text{BareAnswer} \bowtie \text{Donation} \bowtie \text{Donor})
\]
Question 3.  [11 marks]

Here is part of a schema you used for one of your Lecture Prep exercises.

Relations

- Employee(eid, name, salary, dept)
- Department(did, name, division)
- Sales(eid, day, amount)

Integrity constraints

- Employee[dept] ⊆ Department[did]
- Sales[eid] ⊆ Employee[eid]

Part (a)  [2 marks]

Suppose table Sales has this content:

<table>
<thead>
<tr>
<th>eid</th>
<th>day</th>
<th>amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2013-11-02</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>2013-11-03</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>2013-11-05</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>2013-11-06</td>
<td>129</td>
</tr>
<tr>
<td>5</td>
<td>2013-11-01</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>2013-11-02</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>2013-11-06</td>
<td>129</td>
</tr>
<tr>
<td>6</td>
<td>2013-11-07</td>
<td>18</td>
</tr>
<tr>
<td>7</td>
<td>2013-11-01</td>
<td>18</td>
</tr>
<tr>
<td>7</td>
<td>2013-11-02</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>2013-11-01</td>
<td>28</td>
</tr>
<tr>
<td>8</td>
<td>2013-11-02</td>
<td>129</td>
</tr>
</tbody>
</table>

Below, show the output of the following query.

```sql
SELECT eid
FROM Sales s1
WHERE EXISTS
  (SELECT eid
   FROM Sales s2
   WHERE s1.eid <> s2.eid AND s1.day = s2.day AND s1.amount = s2.amount);
```

Solution:

<table>
<thead>
<tr>
<th>eid</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
</tbody>
</table>

(4 rows)
Part (b) [2 marks]
Complete each of the following two queries so that they will run without error:

```sql
SELECT eid, min(amount), count(day)
FROM Sales

SELECT count(eid), day, sum(amount)
FROM Sales
```

Solution:

```sql
SELECT eid, min(amount), count(day)
FROM Sales
GROUP BY eid;

<table>
<thead>
<tr>
<th>eid</th>
<th>min</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>28</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>18</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>2</td>
</tr>
</tbody>
</table>

(5 rows)

SELECT count(eid), day, sum(amount)
FROM Sales
GROUP BY day;

<table>
<thead>
<tr>
<th>count</th>
<th>day</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2013-11-02</td>
<td>155</td>
</tr>
<tr>
<td>1</td>
<td>2013-11-07</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>2013-11-03</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>2013-11-01</td>
<td>58</td>
</tr>
<tr>
<td>1</td>
<td>2013-11-05</td>
<td>25</td>
</tr>
<tr>
<td>2</td>
<td>2013-11-06</td>
<td>258</td>
</tr>
</tbody>
</table>

(6 rows)
```

Part (c) [2 marks]
Complete this query so that it will report the eid of employees whose total sales in the Sales table exceeds 1,000. Do not use subqueries.

```sql
SELECT eid
FROM Sales
```

Solution:

```sql
SELECT eid
FROM Sales
GROUP BY eid
HAVING sum(amount) > 1000;
```

Part (d) [5 marks]
Write a query in SQL to find the eid of employees whose department name is "Widgets" and who have never made a sale.

Solution:

```sql
(SELECT eid FROM Employee, Department WHERE did=dept AND department.name='Widgets')
except
(SELECT eid FROM Sales);
```