Use the handout of three tables -- Vehicle, Own, Person -- to answer these.

0. Assume that Vehicle only has the first two rows shown on the handout and that Person only has the first two rows shown on the handout. Show or otherwise describe the result on the natural join of Vehicle and Person.

Natural join results in a pairing of rows in which a row of one table is paired with a row of another table if the two constituent rows are equal along all the same-named (and typed) attributes in the two tables. There are ZERO same-named attributes between Vehicle and Person, so all pairs of rows satisfy the constraint that ZERO same-named rows have the same value.

<table>
<thead>
<tr>
<th>VRN</th>
<th>Ma</th>
<th>Mo</th>
<th>Color</th>
<th>SSN</th>
<th>Name</th>
<th>Addr</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Honda</td>
<td>Hawk</td>
<td>Red</td>
<td>abc</td>
<td>Dave</td>
<td>Birch</td>
<td>xxx</td>
</tr>
<tr>
<td>123</td>
<td>Honda</td>
<td>Hawk</td>
<td>Red</td>
<td>bcd</td>
<td>Mary</td>
<td>Grove</td>
<td>yyy</td>
</tr>
<tr>
<td>234</td>
<td>Mazda</td>
<td>RX7</td>
<td>Blue</td>
<td>abc</td>
<td>Dave</td>
<td>Birch</td>
<td>xxx</td>
</tr>
<tr>
<td>234</td>
<td>Mazda</td>
<td>RX7</td>
<td>Blue</td>
<td>bcd</td>
<td>Mary</td>
<td>Grove</td>
<td>yyy</td>
</tr>
</tbody>
</table>

Give relational algebra expressions for each of the following queries specified in English (assuming the three tables, with all rows, of the handout).
1. Write a query that returns the Name and Phone of all Persons owning VRN=123.

\[ T_{Name, \ \text{Phone}} (\sigma_{VRN=123} (\langle \text{Own} \bowtie \text{Person} \rangle)) \]

could have parenthesized individual tables

\[ T_{Name, \ \text{Phone}} (\sigma_{VRN=123} (\langle \text{Own} \bowtie \text{Person} \rangle)) \]

rename SSNs to distinguish them

\[ \theta \text{(join makes join condition explicit)} \]

This shorthand would be acceptable

\[ T_{Name, \ \text{Phone}} (\sigma_{VRN=123} (\langle \text{Own} \bowtie \text{Person} \rangle)) \]

Push selects inward

\[ T_{Name, \ \text{Phone}} (\langle \sigma_{VRN=123} \text{Own} \bowtie \text{Person} \rangle) \]

Generally, "pushing" projections almost never a good idea

\[ T_{Name, \ \text{Phone}} (\langle \sigma_{VRN=123} \text{Own} \bowtie (T_{Name, \ \text{Person}}) \rangle) \]

\[ \\text{Wrong} \]

No basis for join
1. Own

<table>
<thead>
<tr>
<th>VRN</th>
<th>SSN</th>
<th>SSN</th>
</tr>
</thead>
<tbody>
<tr>
<td>012</td>
<td>abc</td>
<td>def</td>
</tr>
<tr>
<td>123</td>
<td>bcd</td>
<td>abc</td>
</tr>
<tr>
<td>234</td>
<td>cde</td>
<td>bcd</td>
</tr>
</tbody>
</table>

\[ \text{VRN} = 123 \]

What if answer had been

\[ \Pi_{\text{Name}, \text{Phone}} (\bigotimes_{\text{VRN} = 123} (\text{Own} \times \text{Person})) \]

\[ \begin{array}{c|c|c}
\text{VRN} & \text{SSN} & \text{SSN} \\
012 & abc & def \\
123 & bcd & abc \\
234 & cde & bcd & cde \\
\end{array} \]

\[ \Rightarrow 012 \ abc \ def \]

These would pass select \(=123\)
2. Write a query to show the VRN and Mo of each owned Vehicle and the Name and Addr of the Person who owns it.

\[ \Pi_{\text{VRN}, \text{Mo}, \text{Name}, \text{Addr}} ((\text{Vehicle } \bowtie \text{ Own }) \bowtie \text{ Person}) \]

\[ \Pi_{\text{VRN}, \text{Mo}, \text{Name}, \text{Addr}} ((\text{Vehicle } \bowtie (\text{Own } \bowtie \text{ Person}) ) \]

\[ \Pi_{\text{VRN}, \text{Mo}, \text{Name}, \text{Addr}} ((\text{Vehicle } \bowtie \text{ Person} ) \bowtie \text{ Own}) \]

and of course, can write natural join as theta join that spells out the join condition

\[ \bowtie_{\text{V.VRN} \atop = \text{O.VRN}} \land \text{P.SSN} \atop = \text{O.SSN} \]

Shorthand OK

\[ \Pi_{\text{VRN}, \text{Mo}, \text{Name}, \text{Addr}} (\text{Vehicle } \bowtie \text{ Own } \bowtie \text{ Person}) \]
3. Write a query to show the VRN and Mo of each owned Vehicle by someone on 'Birch' and the Name of the Person who owns it.

$$\Pi_{\text{VRN}, \text{Mo}, \text{Name}} (\sigma_{\text{Addr} = 'Birch'} (\text{Vehicle} \bowtie \text{Own} \bowtie \sigma_{\text{Addr} = 'Birch'} (\text{Person})))$$

Push select inward

$$\Pi_{\text{VRN}, \text{Mo}, \text{Name}} (\text{Vehicle} \bowtie (\text{Own} \bowtie (\sigma_{\text{Addr} = 'Birch'} (\text{Person}))))$$

This is OK, since

$$\sigma_{\text{R} \times \text{S}} (\text{Own} \bowtie \text{Person}) = \sigma_{\text{B}} (\text{R} \times \text{S})$$

but I don't like its style, because 'Birch' is a constant and is not an attribute of either table. Addr='Birch' is best thought of as a select condition
4. Write a query that returns the VRN and Mo of any vehicle that isn't owned.

\[ \left( \Pi_{\text{VRN}} \text{, Vehicle} \right) - \left( \Pi_{\text{VRN}} \text{, (Vehicle } \cap \text{ Own)} \right) \]

Set difference operands must have same format

all vehicles

all owned vehicles

all unowned vehicles

\[ \Pi_{\text{VRN}} \text{, (} \left( \Pi_{\text{VRN}} \text{, Vehicle } - \Pi_{\text{VRN}} \text{, Own} \right) \cap \text{ Vehicle) } \]

VRNs of all unowned vehicles

"join back" to get Mo

(this strategy was illustrated in video)
5. Write a query that returns pairs of SSNs for (different) Persons that live at the same Addr.

\[
\prod_{P1, SSN, P2, SSN} \left( \left( \left( P1 \ldots P1 \right) \bowtie \left( P1 \ldots P2 \right) \right) \right)
\]

\[
\text{OR} \left\{ \begin{aligned}
& P1 = \text{Person} \\
& P2 = \text{Person}
\end{aligned} \right\}
\]

\[
\prod_{P1, SSN, P2, SSN} \left( \left( P1 \bowtie \left( P1 \ldots P2 \right) \right) \right)
\]

\[
\prod_{P1, SSN, P2, SSN} \left( P1 \bowtie \left( P1 \ldots P2 \right) \right)
\]

The versions above will return answers of form

\[
\begin{array}{cc}
P1, SSN & P2, SSN \\
\vdots & \vdots \\
123 & 456 \\
456 & 123 \\
\end{array}
\]

Conceptual duplication

Best to replace '\(\bowtie\)' above with '\(<\)' (or '\(\rangle\)')

Only get

\[
\begin{array}{ccc}
123 & 456 \\
456 & 123 \\
\end{array}
\]
6. Write a query that returns pairs of Names for (different) Persons that live at the same Addr.

\[
\prod_{p1, \text{Name},} \left( \left( c_{p1(\ldots)} \text{Person} \right) \times \left( c_{p2(\ldots)} \text{Person} \right) \right)
\] \[\text{where:} \quad p1, \text{Addr} = p2, \text{Addr} \]
\[\land p1, \text{SSN} < p2, \text{SSN} \]

Still using SSN in join!
(e.g., my father and I shared the same name)
7. Write a query that lists pairs of Vehicles by Ma and Mo that are owned by the same Person.

\[ \text{Person} \xrightarrow{\text{Own}(1)} \text{Vehicle}(1) \xrightarrow{<>} \text{Vehicle}(2) \]

\[ \text{Same Person} \]

\[ \text{Person} \xrightarrow{\text{Own}(2)} \text{Vehicle}(2) \]

\[ \text{Can parenthesize & rename in different ways} \]

But bigger observation – don’t need to involve Person at all, because Own includes a person’s unique ID

\[ \text{Person} \xrightarrow{\text{Own}(1)} \text{Vehicle}(1) \]

\[ \text{Person} \xrightarrow{\text{Own}(2)} \text{Vehicle}(2) \]

\[ \text{Use } < (\text{or } >) \text{ to get rid of conceptual redundancy} \]
For problems like 7, of non-trivial complexity, it's helpful to break them apart

1. What relations are involved?
   Ma, Mo require that Vehicle be involved.
   If only VRNs were needed as the Vehicle info, then only Own would be needed.

2. Decompose problem is often helpful. A simpler problem would be to list pairs of vehicles owned by same person

   $$\begin{align*}
   &\text{Own} \implies \text{Own} \\
   &01(\ldots) \implies \text{Own} \\
   &=02, \text{SSN} \\
   &\land 01, \text{VRN} \\
   &<02, \text{VRN}
   \end{align*}$$

   then recognize that Ma, Mo require too, which implicates Vehicle

   or perhaps start with simpler problem of listing pairs of all different vehicles

   $$\begin{align*}
   &\text{Vehicle} \implies \text{Vehicle} \\
   &\text{Vehicle} \implies \text{Vehicle} \\
   &\land 01, \text{VRN} \\
   &<02, \text{VRN}
   \end{align*}$$

   and build on that
8. Write a query that lists pairs of Persons by SSN and Name that own a Vehicle of the same Ma, Mo and Color.

It's possible for the same person to have two different vehicles with the same Ma, Mo, Color; how would query change if allowed?

```sql
T
PT1.SSN, (< Person XOR Person) Person ≡ different persons
PT1.Name, PT1.SSN < PT2.SSN

PT2.Name

PT1.SSN = 01.SSN
PT2.SSN = 02.SSN

PT1.SSN (01(...)
PT1.VRN = V1.VRN

PT2.SSN (02(...)
PT2.VRN = V2.VRN

V1.Ma = V2.Ma
V1.Mo = V2.Mo
V1.Color = V2.Color

(A V1.VRN < V2.VRN) - if we want to distinguish different vehicles (I think we do)
```

Notice that renaming in inner scope is ok to reference in outer scope. If we wanted to disallow then put all renames in outermost scope:

```sql
C PT1.SSN (...)
```
P1
SSN
123 <
OWN 01
VRN 567

P2
SSN
234
OWN 02
VRN 456

\[ \frac{1}{2} \]

if we have ordered by SSN, then ordering by VRN may cause conflict

use \( <> \) in comparing VRNs for 8
9. Suppose that there was a Price attribute on Vehicle. Write a query in which each car is listed (paired) with each other car that costs less than it does.

\[ \Pi_{v1.urn, v2.urn} (C_{v1(\ldots)} \text{Vehicle} \Join \ C_{v2(\ldots)} \text{Vehicle}) \quad \text{where} \quad v1.\text{Price} < v2.\text{Price} \]

Note: Not all joins are equality joins.
10. Give a left-deep expression tree for the relational algebra query specification of question 3. Your expression tree need not correspond precisely to your answer for question 3.
others possible