Question 13: Basic relational algebra

a. The tables generated in the calculation are:

\[
\pi_{X,Y}(T_1) : \begin{array}{ccc}
X & Y \\
 x & 15 \\
 x & 10 \\
 x & 15 \\
 y & 15 \\
\end{array}
\]

\[
\pi_{X,D}(T_2) : \begin{array}{ccc}
X & D \\
 x & p \\
 x & t \\
 y & p \\
 y & r \\
\end{array}
\]

\[
R = \pi_{X,Y}(T_1) \bowtie \pi_{X,D}(T_2) : \begin{array}{ccc}
X & Y & D \\
 x & 15 & p \\
 x & 15 & t \\
 x & 10 & p \\
 x & 10 & t \\
 x & 15 & p \\
 x & 15 & t \\
 y & 15 & p \\
 y & 15 & r \\
\end{array}
\]

b. The selection operator \( \sigma_{D=p} \) only depends on the table \( T_2 \). So, applying it early can reduce the size of the tables involved. Hence, a more efficient expression is:

\[
\pi_{X,Y}(T_1) \bowtie \pi_{X,D}(\sigma_{D=p}(T_2))
\]

Question 14: Understanding relational algebra

a. The last name of all those members of staff whose first name is ‘John’.

\[
\text{SELECT lastname} \\
\text{FROM staff} \\
\text{WHERE firstname='John';}
\]

b. The family name of all those members of staff who taught a course with more than 100 students (in any year).

\[
\text{SELECT DISTINCT lastname} \\
\text{FROM staff, lecturing} \\
\text{WHERE staff.sid = lecturing.sid AND numbers > 100;}
\]

c. The name of those courses, which had more than 100 students on them in at least one year, and which are not at level 1.

\[
\text{SELECT name} \\
\text{FROM lecturing, courses} \\
\text{WHERE lecturing.cid = courses.cid AND numbers>100} \\
\text{EXCEPT} \\
\text{SELECT name} \\
\text{FROM courses} \\
\text{WHERE level = 1;}
\]

d. List members of staff with the level 2 courses they taught in 1999.

\[
\text{SELECT lastname, name} \\
\text{FROM staff AS s, lecturing AS l, courses AS c} \\
\text{WHERE s.sid = l.sid AND c.cid = l.cid AND year = 1999 AND level = 2;}
\]
Question 15: From SQL to relational algebra

a. \( \pi_{\text{name}} (\sigma_{\text{year}=1999 \lor \text{year}=2000} \land \text{lastname}='Jung' (\text{staff} \bowtie \text{lecturing} \bowtie \text{courses})) \)

b. \( \pi_{\text{name}} (\sigma_{\text{year}=2001} (\text{lecturing} \bowtie \text{courses})) - \pi_{\text{name}} (\sigma_{\text{year}=2000 \lor \text{year}=1999} (\text{lecturing} \bowtie \text{courses})) \)

c. \( \pi_{\text{name}} (\sigma_{\text{year}=1999} (\text{courses} \bowtie \text{lecturing} \bowtie \rho_{\text{year} \rightarrow \text{year2}, \text{numbers} \rightarrow \text{numbers2}} (\text{lecturing}))) \)

(If you don’t rename the \( \text{year} \) and \( \text{numbers} \) attributes, then you will just get “lecturing” again from the natural join “\( \text{lecturing} \bowtie \text{lecturing} \)”.)

Alternatively, you can project just the \( \text{cid} \) column from the \( \text{lecturing} \) tables so that there is no need for renaming: 
\( \pi_{\text{name}} (\text{courses} \bowtie \pi_{\text{cid}} (\sigma_{\text{year}=1999} (\text{lecturing}))) \bowtie \pi_{\text{cid}} (\sigma_{\text{year}=200} (\text{lecturing})) \)

d. \( \pi_{\text{lastname}} ((\pi_{\text{sid}} (\text{staff}) - \pi_{\text{sid}} (\text{lecturing})) \bowtie \text{staff} ) \)

Exercise 16: Deriving functional dependencies

a. Does not imply \( A \rightarrow D \) because \( A \) alone does not suffice to apply any of the given dependencies. Counterexample:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b_1</td>
<td>c_1</td>
<td>d_1</td>
</tr>
<tr>
<td>a</td>
<td>b_2</td>
<td>c_2</td>
<td>d_2</td>
</tr>
</tbody>
</table>

b. Does imply \( A \rightarrow D \):

i. \( A \rightarrow A \) (trivially)

ii. \( A \rightarrow B \) (assumption)

iii. \( AB \rightarrow C \) (assumption)

iv. \( A \rightarrow C \) (transitivity applied to (i), (ii), and (iii))

v. \( BC \rightarrow D \) (assumption)

vi. \( A \rightarrow D \) (transitivity applied to (iv), (ii), and (iv))

c. Does imply \( A \rightarrow D \). Derivation is almost the same as for (b).

Exercise 17: Suggesting functional dependencies

The pair (\( \text{registration\_number}, \text{date\_of\_inspection} \)) is a candidate key. That means that we have the functional dependency
\( \text{registration\_number}, \text{date\_of\_inspection} \rightarrow X \)

for every other attribute \( X \). For instance, consider \( X = \text{garage} \).

Some possible dependencies:

- \( \text{owner\_contact\_phone} \rightarrow \text{owner} \)
- \( \text{owner\_contact\_phone} \rightarrow \text{owner\_address} \)
- \( \text{registration\_number} \rightarrow \text{model} \)
- \( \text{registration\_number} \rightarrow \text{year\_of\_first\_registration} \)
- \( \text{registration\_number} \rightarrow \text{diesel\_or\_petrol} \)
- \( \text{registration\_number} \rightarrow \text{date\_of\_previous\_MOT} \)
- \( \text{model} \rightarrow \text{diesel\_or\_petrol} \)
- \( \text{garage} \rightarrow \text{garage\_address} \)
- \( \text{garage\_MOT\_licence} \rightarrow \text{garage} \)

Does the contact phone number determine the owner? Yes, as long as we assume that the owner does not give up his/her phone number, which then gets assigned to somebody else.

Car registration numbers (by which we mean the licence plate numbers) can be similarly transferred between vehicles, which might invalidate the stated dependencies.

When we say a model determines whether a car is diesel or petrol, we are assuming that the manufacturer has encoded that information in the model name, and we are using the full model name.

We are also assuming that the garage has not relocated to a new address during the time covered by the database.
Exercise 18: Outer Joins (Optional))

a. Inner join $T_1 \bowtie T_2$:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>11</td>
</tr>
</tbody>
</table>

b. Dangling tuples of: $T_1$:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>6</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>

c. Outer join $T_1 \bowtie T_2$:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>2</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>-</td>
<td>6</td>
<td>7</td>
<td>12</td>
</tr>
</tbody>
</table>

d. Outer join of $T_1 \bowtie T_2$:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>6</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>