Relational Databases
Assignment 1

Caltech CS 121
January 2023

Dallas Taylor

Part A

2.5

a. $\Pi_{\text{person\_name}}(\sigma_{\text{company\_name}=\text{"First Bank Corporation"}}(\text{works}))$

b. $\Pi_{\text{person\_name, city}}(\text{employee} \bowtie \sigma_{\text{company\_name}=\text{"First Bank Corporation"}}(\text{works}))$

c. $\Pi_{\text{person\_name, street, city}}(\text{employee} \bowtie \sigma_{\text{company\_name}=\text{"First Bank Corporation"} \wedge \text{ salary}>\$10,000}(\text{works}))$

d. $\Pi_{\text{person\_name}}(\text{employee} \bowtie \text{company} \bowtie \text{works})$

e. $\text{company} \div \Pi_{\text{city}}(\sigma_{\text{company\_name}=\text{"Small Bank Corporation"}}(\text{works})$

2.6

NEW QUERY:

$\Pi_{\text{customer\_name, customer\_city, loan\_number, amount}}(\text{borrower} \bowtie \text{loan} \bowtie \text{customer})$

a. Jackson does not appear in the result because there is no tuple for him in the customer relation. Thus, when we perform the natural join with customer, Jackson is removed from the data set.

b. In order to have Jackson in the result, I would modify the data set such that we now add a tuple for Jackson in the customer relation: customer(Jackson, customer_street, customer_city). These could be set to *null* or *unknown*, if desired. We know that he already has tuples in the borrower and loan relations.

c. I would use left outer join on customer:

$$\Pi_{\text{customer\_name, customer\_city, loan\_number, amount}}(\text{borrower} \bowtie \text{loan} ⟕ \text{customer})$$

2.7

a.

$$\begin{aligned}\text{works} \quad \leftarrow \quad & \Pi_{\text{person\_name, company\_name, (salary*1.1)}}(\sigma_{\text{company\_name}=\text{"First Bank Corporation"}}(\text{works})) \cup \\ & \sigma_{\text{company\_name} \neq \text{"First Bank Corporation"}}(\text{works})\end{aligned}$$

b.

$$\begin{aligned}\text{managers} \quad \leftarrow \quad & \Pi_{\text{(manager\_name as person\_name)}}(\text{manages}) \\ \text{works} \quad \leftarrow \quad & \Pi_{\text{person\_name, company\_name, (salary * 1.1)}}(\sigma_{\text{salary * 1.1} \leq \$100,000}(\text{works} \bowtie \text{managers})) \cup \\ & \Pi_{\text{person\_name, company\_name, (salary * 1.03)}}(\sigma_{\text{salary * 1.1} > \$100,000}(\text{works} \bowtie \text{managers})) \cup \\ & \text{works} - \text{works} \bowtie \text{managers}\end{aligned}$$

c.

$$\text{works} \quad \leftarrow \quad \text{works} - \sigma_{\text{company\_name}=\text{"Small Bank Corporation"}}(\text{works})$$

Dallas Taylor

2.8

a.

$$\begin{aligned} \text{accounts} &\leftarrow {}_{\text{account\_number}}\mathbf{G}_{\textbf{count}(\text{customer\_name})\ \textbf{as}\ n}(\text{depositor}) \\ &\quad \Pi_{\text{account\_number}}(\sigma_{n>2}(\text{accounts})) \end{aligned}$$

b. For the following problem, I will represent the keys account_number and customer_name as a_n and c_n, respectively.

$$\Pi_{\text{t1.a\_n}}(\sigma_{\left(\substack{\text{t1.c\_n}\neq\text{t2.c\_n}\ \wedge\ \text{t1.c\_n}\neq\text{t3.c\_n}\ \wedge\ \text{t2.c\_n}\neq\text{t3.c\_n},\\ \text{t1.a\_n}=\text{t2.a\_n}\ \wedge\ \text{t1.a\_n}=\text{t3.a\_n}\ \wedge\ \text{t2.a\_n}=\text{t3.a\_n}}\right)}(\rho_{t1}(\text{depositor}) \times \rho_{t2}(\text{depositor}) \times \rho_{t3}(\text{depositor})))$$

2.9

a.

$$\begin{aligned} \text{employee\_counts} &\leftarrow {}_{\text{company\_name}}\mathbf{G}_{\textbf{count}(\text{person\_name})\ \textbf{as}\ \text{count}}(\text{works}) \\ \text{max\_count} &\leftarrow \mathbf{G}_{\textbf{max}(\text{count})\ \textbf{as}\ \text{max}}(\text{employee\_counts}) \\ &\quad \Pi_{\text{company\_name}}(\sigma_{\text{count = max}}(\text{max\_count} \times \text{employee\_count})) \end{aligned}$$

b.

$$\begin{aligned} \text{payrolls} &\leftarrow {}_{\text{company\_name}}\mathbf{G}_{\textbf{sum}(\text{salary})\ \textbf{as}\ \text{payroll}}(\text{works}) \\ \text{min\_payroll} &\leftarrow \mathbf{G}_{\textbf{min}(\text{payroll})\ \textbf{as}\ \text{min}}(\text{payrolls}) \\ &\quad \Pi_{\text{company\_name}}(\sigma_{\text{payroll = min}}(\text{payroll} \times \text{min\_payroll})) \end{aligned}$$

c.

$$\begin{aligned} \text{average\_salaries} &\leftarrow {}_{\text{company\_name}}\mathbf{G}_{\textbf{avg}(\text{salary})\ \textbf{as}\ \text{avg\_sal}}(\text{works}) \\ \text{average\_FBC\_salary} &\leftarrow \Pi_{\text{avg\_sal}}(\sigma_{\text{company\_name = "First Bank Corporation"}}(\text{average\_salaries})) \\ &\quad \Pi_{\text{company\_name}}(\sigma_{\text{avg\_sal} > \text{average\_FBC\_salary.avg\_sal}}(\text{average\_salaries})) \end{aligned}$$

Part B

a. To explain my answer I will first rewrite the rule, where l represents the schema *customer_likes* and d represents the schema *customer_drinks*:

$$\begin{aligned}
l \div d &= \Pi_{\text{name}}(l) - \Pi_{\text{name}}((\Pi_{\text{name}}(l) \times d) - \Pi_{\text{name, drink}}(l)) \\
&= \begin{Bmatrix} Archie \\ Eli \\ Pascal \\ Codd \end{Bmatrix} - \Pi_{\text{name}}((\begin{Bmatrix} Archie \\ Eli \\ Pascal \\ Codd \end{Bmatrix} \times \begin{Bmatrix} Espresso \\ Coffee \\ Mocha \end{Bmatrix}) - \begin{Bmatrix} Codd & Espresso \\ Codd & Coffee \\ Codd & Mocha \\ Archie & Espresso \\ ... & ... \\ Codd & Tea \end{Bmatrix}) \\
&= \begin{Bmatrix} Archie \\ Eli \\ Pascal \\ Codd \end{Bmatrix} - \Pi_{\text{name}}(\begin{Bmatrix} Archie & Coffee \\ Pascal & Espresso \end{Bmatrix}) \\
&= \begin{Bmatrix} Archie \\ Eli \\ Pascal \\ Codd \end{Bmatrix} - \begin{Bmatrix} Archie \\ Pascal \end{Bmatrix} = \begin{Bmatrix} Eli \\ Codd \end{Bmatrix}
\end{aligned}$$

We can clearly see that when we perform $(\Pi_{\text{name}}(l) \times d) - \Pi_{\text{name, drink}}(l)$, that we will be left with the tuples of each name, drink combination that is missing from customer_likes (l). Codd likes all of the drinks, and thus all entries with Codd's name will be removed from the cartesian product. The extra entry with tea will be ignored during the set substraction.

b. Note that for this problem, we denote l^* as the relation *customer_likes* with all tuples other than (Codd, Tea).

$$\begin{aligned}
l \div_E d &= (l \div s) - \Pi_{\text{name}}(l - (l \bowtie d)) \\
&= \begin{Bmatrix} Eli \\ Codd \end{Bmatrix} - \Pi_{\text{name}}(l - (l \bowtie \begin{Bmatrix} Espresso \\ Coffee \\ Mocha \end{Bmatrix})) \\
&= \begin{Bmatrix} Eli \\ Codd \end{Bmatrix} - \Pi_{\text{name}}(l - l^*) \\
&= \begin{Bmatrix} Eli \\ Codd \end{Bmatrix} - \{Codd\} = \{Eli\}
\end{aligned}$$

This produces the formal definition:

$$r \div_E s = (r \div s) - \Pi_{R-S}\big(\Pi_{R-S,S}(r) - \Pi_{R-S,S}(r \bowtie s)\big)$$

c. We can get the total number of relevant drinks to be $\mathbf{G}_{\textbf{count}(\text{drink})\ \textbf{as}\ \text{n\_drinks}}(d)$, and we already know how to remove tuples in *customer_likes* with drinks not *customer_drinks* by using $l \bowtie d$. Thus, we can produce the following query:

$$\Pi_{\text{name}}\big({}_{\text{name}}\mathbf{G}_{\textbf{coount}(\text{drink})\ \textbf{as}\ \text{n\_drinks}}(l \bowtie d) \bowtie \mathbf{G}_{\textbf{count}(\text{drink})\ \textbf{as}\ \text{n\_drinks}}(d)\big)$$

Dallas Taylor

Part C

a. Are $\sigma_\theta({}_A\mathbf{G}_F(r))$ and ${}_{\mathrm{A}}\mathbf{G}_F(\sigma_\theta(r))$ equivalent?

A is a set of grouping attributes
F is a set of aggregation functions
θ is a predicate using only attributes from A

Yes, these statements are equivalent. In the first case, the aggregate functions are grouped based on A and then tuples are pulled that satisfy the θ predicate, which only uses attributes from A. In the second case, the tuples are first pulled that satisfy the θ predicate, and since we know these use attributes from A, we know the final grouped attributes will be the same as those from the first group.

b. Are $\Pi_A(r–s)$ and $\Pi_A(r)–\Pi_A(s)$ equivalent?

Relations r and s have compatible schemas, e.g. $r(a, b)$ and $s(a, b)$.

No, these statements are not equivalent. This can be seen if you consider the following relations:

$$\begin{aligned}
r &\leftarrow \begin{Bmatrix} a1 & b1 \\ a2 & b1 \end{Bmatrix} \\
s &\leftarrow \begin{Bmatrix} a1 & b1 \\ a2 & b2 \end{Bmatrix} \\
r - s &= \begin{Bmatrix} a2 & b1 \end{Bmatrix} \\
\Pi_A(r - s) &= \{a2\} \\
\Pi_A(r), \Pi_A(s) &= \begin{Bmatrix} a1 \\ a2 \end{Bmatrix}, \begin{Bmatrix} a1 \\ a2 \end{Bmatrix} \\
\Pi_A(r) - \Pi_A(s) &= \{\}
\end{aligned}$$

c. Are $(r ⟗ s) ⟗ t$ and $r ⟗ (s ⟗ t)$ equivalent?

r is a relation with schema (a, b_1) s is a relation with schema (a, b_2) t is a relation with schema (a, b_3)

No, these statements are not equivalent. This can be seen if you consider the following relations:

$$r \leftarrow \begin{Bmatrix} a_1 & b_{11} \\ a_2 & b_{12} \end{Bmatrix} \qquad s \leftarrow \begin{Bmatrix} a_1 & b_{21} \\ a_3 & b_{22} \end{Bmatrix} \qquad t \leftarrow \begin{Bmatrix} a_1 & b_{31} \\ a_2 & b_{32} \end{Bmatrix}$$

$$r ⟗ s = \begin{Bmatrix} a_1 & b_{11} & b_{21} \\ a_2 & b_{12} & \mathit{null} \end{Bmatrix} \qquad s ⟗ t = \begin{Bmatrix} a_1 & b_{21} & b_{31} \\ a_3 & b_{22} & \mathit{null} \end{Bmatrix}$$

$$(r ⟗ s) ⟗ t = \begin{Bmatrix} a_1 & b_{11} & b_{21} & b_{31} \\ a_2 & b_{12} & \mathit{null} & b_{32} \end{Bmatrix} \qquad r ⟗ (s ⟗ t) = \begin{Bmatrix} a_1 & b_{11} & b_{21} & b_{31} \\ a_3 & b_{22} & \mathit{null} & \mathit{null} \end{Bmatrix}$$

d. Are $\sigma_\theta(r ⟗ s)$ and $\sigma_\theta(r) ⟗ s$ equivalent?

θ is a predicate using only attributes from r

Yes, these statements are equivalent. The selection predicate θ does not need to occur in a particular order, as long as it's performed on r, which is on the left side of the left outer join. This is because r's data is preserved.

e. Are $\sigma_\theta(r ⟕ s)$ and $r ⟕ \sigma_\theta(s)$ equivalent?

θ is a predicate using only attributes from s.

No, these statements are not equivalent. This can be seen if you consider the following relations:

$$r \leftarrow \begin{Bmatrix} a_1 & b_{11} \\ a_2 & b_{12} \\ a_3 & b_{13} \end{Bmatrix} \qquad s \leftarrow \begin{Bmatrix} a_1 & b_{21} \\ a_3 & b_{22} \end{Bmatrix}$$

$$r ⟕ s = \begin{Bmatrix} a_1 & b_{11} & b_{21} \\ a_2 & b_{12} & null \\ a_3 & b_{13} & b_{22} \end{Bmatrix} \qquad \sigma_{b_2=b_{21}}(s) = \begin{Bmatrix} a_1 & b_{21} \end{Bmatrix}$$

$$\sigma_{b_2=b_{21}}(r ⟕ s) = \begin{Bmatrix} a_1 & b_{11} & b_{21} \end{Bmatrix} \qquad r ⟕ \sigma_{b_2=b_{21}}(s) = \begin{Bmatrix} a_1 & b_{11} & b_{21} \\ a_2 & b_{12} & null \\ a_3 & b_{13} & null \end{Bmatrix}$$