CHAPTER 3

Relational Model

Exercises

3.1 Answer: One solution is described below. Many alternatives are possible. Underlined attributes indicate the primary key.

student (student-id, name, program)
course (courseno, title, syllabus, credits)
course-offering (courseno, secno, year, semester, time, room)
instructor (instructor-id, name, dept, title)
enrols (student-id, courseno, secno, semester, year, grade)
teaches (courseno, secno, semester, year, instructor-id)
requires (maincourse, prerequisite)

3.5 Answer:

e. $\Pi_{\text{person-name}}((\text{employee} \bowtie \text{manages})$

$f. The following solutions assume that all people work for exactly one company. If one allows people to appear in the database (e.g. in employee) but not appear in works, the problem is more complicated. We give solutions for this more realistic case later.

$\Pi_{\text{person-name}}((\sigma_{\text{company-name} \neq \text{"First Bank Corporation"}}(\text{works})))$

If people may not work for any company:

$\Pi_{\text{person-name}}(\text{employee}) - \Pi_{\text{person-name}}((\sigma_{\text{company-name} = \text{"First Bank Corporation"}}(\text{works})))$

g. $\Pi_{\text{person-name}}(\text{works}) - (\Pi_{\text{works.person-name}}(\text{works})$

$\bowtie (\text{works.salary} \leq \text{works2.salary} \land \text{works2.company-name} = \text{"Small Bank Corporation"})$
3.7 Answer:

a. The left outer theta join of \( r(\mathcal{R}) \) and \( s(\mathcal{S}) \) can be defined as
\[
(r \bowtie_{\theta} s) \cup ((r - \Pi_{R}(r \bowtie_{\theta} s)) \times (\text{null}, \text{null}, \ldots, \text{null}))
\]
The tuple of nulls is of size equal to the number of attributes in \( S \).

3.8 Answer:

c. The update syntax allows reference to a single relation only. Since this update requires access to both the relation to be updated (\texttt{works}) and the manages relation, we must use several steps. First we identify the tuples of \texttt{works} to be updated and store them in a temporary relation \((t_1)\). Then we create a temporary relation containing the new tuples \((t_2)\). Finally, we delete the tuples in \( t_1 \), from \texttt{works} and insert the tuples of \( t_2 \).

\[
t_1 \leftarrow \Pi_{\texttt{works}.\texttt{person-name},\texttt{company-name},\texttt{salary}} (\sigma_{\texttt{works}.\texttt{person-name} = \texttt{manager-name}}(\texttt{works} \times \texttt{manages}))
\]
\[
t_2 \leftarrow \Pi_{\texttt{person-name},\texttt{company-name},1.1*\texttt{salary}}(t_1)
\]
\[
\texttt{works} \leftarrow (\texttt{works} - t_1) \cup t_2
\]

3.13 Answer:

a. \{ \langle t \rangle \mid \exists q \in r \ (q[A] = t[A]) \}\n
b. \{ \langle t \rangle \mid t \in r \land t[B] = 17 \}\n
c. \{ \langle t \rangle \mid \exists p \in r \ 
\exists q \in s \ (t[A] = p[A] \land t[B] = p[B] \land t[C] = p[C] \land t[D] = q[D] \land t[E] = q[E] \land t[F] = q[F]) \}\n
d. \{ \langle t \rangle \mid \exists p \in r \ 
\exists q \in s \ (t[A] = p[A] \land t[F] = q[F] \land p[C] = q[D]) \}\n
3.14 Answer:

a. \{ \langle < t > \mid \exists p, q \ (< t, p, q > \in r_1) \}\n
b. \{ \langle < a, b, c > \mid < a, b, c > \in r_1 \land b = 17 \}\n
c. \{ \langle < a, b, c > \mid < a, b, c > \in r_1 \lor < a, b, c > \in r_2 \}\n
d. \{ \langle < a, b, c > \mid < a, b, c > \in r_1 \land < a, b, c > \in r_2 \}\n
e. \{ \langle < a, b, c > \mid < a, b, c > \in r_1 \land < a, b, c > \notin r_2 \}\n
f. \{ \langle < a, b, c > \mid \exists p, q \ (< a, b, p > \in r_1 \land < q, b, c > \in r_2) \}\n
3.19 Answer: To insert the tuple (“Johnson”, 1900) into the view \textit{loan-info}, we can do the following:-

\[
borrower \leftarrow (\text{“Johnson”,} \bot) \cup borrower
\]
\[
loan \leftarrow (\bot, \bot, 1900) \cup loan
\]
such that \( \bot \) is a new marked null not already existing in the database.

Note: no commercial database system supports marked nulls, but strings (or other values) that cannot possibly occur as real values can be used to simulate marked nulls.