CHAPTER 3

Relational Model

Exercises

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

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

3.5 Answer:

e. $\Pi_{person\text{-}name}\ ((employee\ \bowtie\ manages)$
$\bowtie_{(manager\text{-}name = employee2.person\text{-}name \wedge employee.street = employee2.street \wedge employee.city = employee2.city)} (\rho_{employee2}\ (employee)))$

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.

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

If people may not work for any company:

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

g. $\Pi_{person\text{-}name}\ (works)\ -\ (\Pi_{works.person\text{-}name}\ (works$
$\bowtie_{(works.salary\ \leq works2.salary \wedge works2.company\text{-}name = \text{“Small Bank Corporation”})}$
$\rho_{works2}(works)))$

3.7 Answer:

a. The left outer theta join of $r(R)$ and $s(S)$ (r ⟕$_\theta$ s) can be defined as
$(r \bowtie_\theta s) \cup ((r - \Pi_R(r \bowtie_\theta s)) \times (null, null, \ldots, null))$
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 (*works*) and the *manages* relation, we must use several steps. First we identify the tuples of *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 *works* and insert the tuples of t_2.

$$t_1 \leftarrow \Pi_{works.person\text{-}name, company\text{-}name, salary} (\sigma_{works.person\text{-}name = manager\text{-}name}(works \times manages))$$

$$t_2 \leftarrow \Pi_{person\text{-}name, company\text{-}name, 1.1 * salary}(t_1)$$

$$works \leftarrow (works - t_1) \cup t_2$$

3.13 Answer:

a. $\{t \mid \exists\, q \in r\ (q[A] = t[A])\}$
b. $\{t \mid t \in r \wedge t[B] = 17\}$
c. $\{t \mid \exists\, p \in r\ \exists\, q \in s\ (t[A] = p[A] \wedge t[B] = p[B] \wedge t[C] = p[C] \wedge t[D] = q[D] \wedge t[E] = q[E] \wedge t[F] = q[F])\}$
d. $\{t \mid \exists\, p \in r\ \exists\, q \in s\ (t[A] = p[A] \wedge t[F] = q[F] \wedge p[C] = q[D]\}$

3.14 Answer:

a. $\{< t > \mid \exists\, p,\ q\ (< t, p, q > \in r_1)\}$
b. $\{< a, b, c > \mid\ < a, b, c > \in r_1 \wedge b = 17\}$
c. $\{< a, b, c > \mid\ < a, b, c > \in r_1 \vee < a, b, c > \in r_2\}$
d. $\{< a, b, c > \mid\ < a, b, c > \in r_1 \wedge < a, b, c > \in r_2\}$
e. $\{< a, b, c > \mid\ < a, b, c > \in r_1 \wedge < a, b, c > \notin r_2\}$
f. $\{< a, b, c > \mid \exists\, p,\ q\ (< a, b, p > \in r_1 \wedge < q, b, c > \in r_2)\}$

3.19 Answer: To insert the tuple ("Johnson", 1900) into the view *loan-info*, we can do the following:-
$borrower \leftarrow (\text{"Johnson"}, \perp_k) \cup borrower$

$loan \leftarrow (\perp_k, \perp, 1900) \cup loan$
such that $\perp_k$ 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.