CHAPTER 6: THE RELATIONAL ALGEBRA AND RELATIONAL CALCULUS

Answers to Selected Exercises

6.15 - Show the result of each of the sample queries in Section 6.5 as it would apply to the database state in Figure 3.6.

Answer:

(QUERY 1) Find the name and address of all employees who work for the 'Research'
department.
Result: FNAME LNAME ADDRESS
John Smith 731 Fondren, Houston, TX
Franklin Wong 638 Voss, Houston, TX
Ramesh Narayan 975 Fire Oak, Humble, TX
Joyce English 5631 Rice, Houston, TX

(QUERY 2) For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and
birth date.
Result:
PNUMBER DNUM LNAME ADDRESS BDATE
10 4 Wallace 291 Berry, Bellaire, TX 20-JUN-31
30 4 Wallace 291 Berry, Bellaire, TX 20-JUN-31

(QUERY 3) Find the names of all employees who work on all the projects controlled by
department number 5.
Result: (empty because no tuples satisfy the result).
LNAME FNAME

(QUERY 4) Make a list of project numbers for projects that involve an employee whose
last name is 'Smith' as a worker or as a manager of the department that controls the project.
Result: PNO
1
2

(QUERY 5) List the names of all employees with two or more dependents.
Result: LNAME FNAME
Smith John
Wong Franklin
(QUERY 6) List the names of employees who have no dependents.
Result: LNAME FNAME
Zelaya Alicia
Narayan Ramesh
English Joyce
Jabbar Ahmad
Borg James
(QUERY 7) List the names of managers who have at least one dependent.
Result: LNAME FNAME
Wallace Jennifer
Wong Franklin

6.16 - Specify the following queries on the COMPANY relational database schema shown in Figure 3.5, using the relational operators discussed in this chapter. Also show the result of each query as it would apply to the database state of Figure 3.6.

(a) Retrieve the names of employees in department 5 who work more than 10 hours per week on the 'ProductX' project.

(b) List the names of employees who have a dependent with the same first name as themselves.

(c) Find the names of employees that are directly supervised by 'Franklin Wong'.

(d) For each project, list the project name and the total hours per week (by all employees) spent on that project.

(e) Retrieve the names of employees who work on every project.

(f) Retrieve the names of employees who do not work on any project.

(g) For each department, retrieve the department name, and the average salary of employees working in that department.

(h) Retrieve the average salary of all female employees.

(i) Find the names and addresses of employees who work on at least one project located in Houston but whose department has no location in Houston.

(j) List the last names of department managers who have no dependents.

Answers:

In the relational algebra, as in other languages, it is possible to specify the same query
in multiple ways. We give one possible solution for each query. We use the symbol s for
SELECT, P for PROJECT, J for EQUIJOIN, * for NATURAL JOIN, and f for FUNCTION.

(a) EMP_W_X <-- (s PNAME='ProductX' (PROJECT)) J (PNUMBER),(PNO) (WORKS_ON)
EMP_WORK_10 <-- (EMPLOYEE) J (SSN),(ESSN) (s HOURS>10 (EMP_W_X))
RESULT <-- P LNAME,FNAME (s DNO=5 (EMP_WORK_10))
Result:
LNAME FNAME
Smith John
English Joyce

(b) E <-- (EMPLOYEE) J (SSN,FNAME),(ESSN,DEPENDENT_NAME) (DEPENDENT)
R <-- P LNAME,FNAME (E)
Result (empty):
LNAME FNAME

(c) WONG_SSN <-- P SSN (s FNAME='Franklin' AND LNAME='Wong' (EMPLOYEE))
WONG_EMPS <-- (EMPLOYEE) J (SUPERSSN),(SSN) (WONG_SSN)
RESULT <-- P LNAME,FNAME (WONG_EMPS)
Result:
LNAME FNAME
Smith John
Narayan Ramesh
English Joyce

(d) PROJ_HOURS(PNO,TOT_HRS) <-- PNO f SUM HOURS (WORKS_ON)
RESULT <-- P PNAME,TOT_HRS ((PROJ_HOURS) J (PNO),(PNUMBER) (PROJECT))
Result:
PNAME TOT_HRS
ProductX 52.5
ProductY 37.5
ProductZ 50.0
Computerization 55.0
Reorganization 25.0
Newbenefits 55.0

(e) PROJ_EMPS(PNO,SSN) <-- P PNO,ESSN (WORKS_ON)

ALL_PROJS(PNO) <-- P PNUMBER (PROJECT)
EMPS_ALL_PROJS <-- PROJ_EMPS -:- ALLPROJS (* DIVISION operation *)
RESULT <-- P LNAME,FNAME (EMPLOYEE * EMP_ALL_PROJS)
Result (empty):
LNAME FNAME
(f) ALL_EMPS <-- P SSN (EMPLOYEE)
WORKING_EMPS(SSN) <-- P ESSN (WORKS_ON)
NON_WORKING_EMPS <-- ALL_EMPS - WORKING_EMPS (*
DIFFERENCE
*)
RESULT <-- P LNAME,FNAME (EMPLOYEE * NON_WORKING_EMPS)
Result (empty):
LNAME FNAME

(g) DEPT_AVG_SALS(DNUMBER,AVG_SAL) <-- DNO f AVG SALARY
(EMPLOYEE)
RESULT <-- P DNUMBER,AVG_SAL (DEPT_AVG_SALS * DEPARTMENT
)
Result:
DNUMBER AVG_SAL
Research 33250
Administration 31000
Headquarters 55000
(h) RESULT(AVG_F_SAL) <-- f AVG SALARY (s SEX='F' (EMPLOYEE))
Result:
AVG_F_SAL
31000
(i) E_P_HOU(SSN) <--
P ESSN (WORKS_ON J(PNO),(PNUMBER) (s PLOCATION='Houston'
(PROJECT)))
D_NO_HOU <--
P DNUMBER (DEPARTMENT) - P DNUMBER (s
DLOCATION='Houston' (DEPARTMENT))
E_D_NO_HOU <-- P SSN (EMPLOYEE J(PNO),(DNUMBER) (D_NO_HOU))
RESULT_EMPS <-- E_P_HOU - E_D_NO_HOU (* this is set DIFFERENCE *)
RESULT <-- P LNAME,FNAME,ADDRESS (EMPLOYEE * RESULT_EMPS)
Result:
LNAME FNAME ADDRESS
Wallace Jennifer 291 Berry, Bellaire, TX
(j) DEPT_MANAGERS(SSN)<-- P MGRSSN (DEPARTMENT)
EMPS_WITH_DEPENDENTS(SSN) <-- P ESSN (DEPENDENT)
RESULT_EMPS <-- DEPT_MANAGERS - EMPS_WITH_DEPENDENTS
RESULT <-- P LNAME,FNAME (EMPLOYEE * RESULT_EMPS)

Result:
LNAME FNAME
Borg James

6.17 – No solution provided.

6.18 - Consider the LIBRARY relational schema shown in Figure 6.14, which is used to keep track of books, borrowers, and book loans. Referential integrity constraints are shown as directed arcs in Figure 6.14, as in the notation of Figure 3.7. Write down relational expressions for the following queries on the LIBRARY database:

(a) How many copies of the book titled The Lost Tribe are owned by the library branch whose name is "Sharpstown"?

(b) How many copies of the book titled The Lost Tribe are owned by each library branch?

(c) Retrieve the names of all borrowers who do not have any books checked out.

(d) For each book that is loaned out from the "Sharpstown" branch and whose DueDate is today, retrieve the book title, the borrower's name, and the borrower's address.

(e) For each library branch, retrieve the branch name and the total number of books loaned out from that branch.

(f) Retrieve the names, addresses, and number of books checked out for all borrowers
who have more than five books checked out.

(g) For each book authored (or co-authored) by "Stephen King", retrieve the title and
the number of copies owned by the library branch whose name is "Central".

Answer:
(Note: We will use S for SELECT, P for PROJECT, * for NATURAL JOIN, - for SET DIFFERENCE, F for AGGREGATE FUNCTION)

(a) A <-- BOOKCOPIES * LIBRARY-BRANCH * BOOK
RESULT <-- P No_Of_Copies (S BranchName='Sharpstown' and Title='The Lost

Tribe'
(A))
Note: A better query would be to do the SELECTs before the JOIN as follows:
A <-- P No_Of_Copies ((S BranchName='Sharpstown' (LIBRARY-BRANCH))
*
(BOOKCOPIES * (S Title='The Lost Tribe'
(BOOK))))

(b) P BranchID,No_Of_Copies ((S Title='The Lost Tribe' (BOOK)) *
BOOKCOPIES)

(c) NO_CHECKOUT_B <-- P CardNo (BORROWER) - P CardNo
(BOOK_LOANS)
RESULT <-- P Name (BORROWER * NO_CHECKOUT_B)

(d) S <-- P BranchId (S BranchName='Sharpstown' (LIBRARY-BRANCH))
B_FROM_S <-- P BookId,CardNo ((S DueDate='today' (BOOKLOANS)) * S)
RESULT <-- P Title,Name,Address (BOOK * BORROWER * B_FROM_S)

(e) R(BranchId,Total) <-- BranchId FCOUNT(BookId,CardNo) (BOOK_LOANS)
RESULT <-- P BranchName,Total (R * LIBRARY_BRANCH)

(f) B(CardNo,TotalCheckout) <-- CardNo F COUNT(BookId) (BOOK_LOANS)
B5 <-- S TotalCheckout > 5 (B)
RESULT <-- P Name,Address,TotalCheckout (B5 * BORROWER)

(g) SK(BookId,Title) <-- (sAuthorName='Stephen King' (BOOK_AUTHORS)) *
BOOK
CENTRAL(BranchId) <-- sBranchName='Central' (LIBRARY_BRANCH)
RESULT <-- P Title,NoOfCopies (SK * BOOKCOPIES * CENTRAL)

6.19 – 6.21: No solutions provided.

6.22 – Consider the two tables *T*1 and *T*2 shown in Figure 6.15. Show the results of the following operations:

Answers:
(a)
P Q R A B C
10 a 5 10 b 6
10 a 5 10 b 5
25 a 6 25 c 3
(b)

P Q R A B C
15 b 8 10 b 6
15 b 8 10 b 5
(c)
P Q R A B C
10 a 5 10 b 6
10 a 5 10 b 5
15 b 8 null null null
25 a 6 25 c 3
(d)
P Q R A B C
15 b 8 10 b 6
null null null 25 c 3
15 b 8 10 b 5
(e)
P Q R
10a 5
15 b 8
25 a 6
10b 6
25 c 3
10b 5
(f)
P Q R A B C
10 a 5 10 b 5

6.23 – No solution provided.

6.24 - Specify queries (a), (b), (c), (e), (f), (i), and (j) of Exercise 6.16 in both tuple and domain relational calculus.

Answer:

(a) Retrieve the names of employees in department 5 who work more than 10 hours per
week on the 'ProductX' project.

Tuple relational Calculus:

{ e.LNAME, e.FNAME | EMPLOYEE(e) AND e.DNO=5 AND (EXISTS p) (EXISTS w)
(WORKS_ON(w) AND PROJECT(p) AND e.SSN=w.ESSN AND w.PNO=p.PNUMBER AND
p.PNAME='ProductX' AND w.HOURS>10) }

Domain relational Calculus:

{ qs | EMPLOYEE(qrstuvwxyz) AND z=5 AND (EXISTS a) (EXISTS b) (EXISTS e)
(EXISTS f)
(EXISTS g) (WORKS_ON(efg) AND PROJECT(abcd) AND t=e AND f=b AND

a='ProductX' AND
g>10) }
(b) List the names of employees who have a dependent with the same first name as themselves.
Tuple relational Calculus:
{ e.LNAME, e.FNAME | EMPLOYEE(e) AND (EXISTS d) (DEPENDENT(d) AND
e.SSN=d.ESSN
AND e.FNAME=d.DEPENDENT_NAME) }
Domain relational Calculus:
{ qs | (EXISTS t) (EXISTS a) (EXISTS b) (EMPLOYEE(qrstuvwxyz) AND DEPENDENT(abcde)
AND a=t AND b=q) }

(c) Find the names of employees that are directly supervised by 'Franklin Wong'.
Tuple relational Calculus:
{ e.LNAME, e.FNAME | EMPLOYEE(e) AND (EXISTS s) (EMPLOYEE(s) AND
s.FNAME='Franklin' AND s.LNAME='Wong' AND e.SUPERSSN=s.SSN) }
Domain relational Calculus:
{ qs | (EXISTS y) (EXISTS a) (EXISTS c) (EXISTS d) (
EMPLOYEE(qrstuvwxyz) AND
EMPLOYEE(abcdefghij) AND a='Franklin' AND c='Wong' AND y=d) }
(e) Retrieve the names of employees who work on every project.
Tuple relational Calculus:
{ e.LNAME, e.FNAME | EMPLOYEE(e) AND (FORALL p) (
NOT(PROJECT(p)) OR
(EXISTS w) (
WORKS_ON(w) AND p.PNUMBER=w.PNO AND w.ESSN=e.SSN)) }
Domain relational Calculus:
{ qs | (EXISTS t) (EMPLOYEE(qrstuvwxyz) AND (FORALL b) (
NOT(PROJECT(abcd)) OR
(EXISTS e) (EXISTS f) (WORKS_ON(efg) AND e=t AND f=b)) }
(f) Retrieve the names of employees who do not work on any project.
Tuple relational Calculus:
{ e.LNAME, e.FNAME | EMPLOYEE(e) AND NOT(EXISTS w) (
WORKS_ON(w)
AND
w.ESSN=e.SSN) }
Domain relational Calculus:
{ qs | (EXISTS t) (EMPLOYEE(qrstuvwxyz) AND NOT(EXISTS a) (
WORKS_ON(abc) AND a=t)
) }

(i) Find the names and addresses of employees who work on at least one project located
in Houston but whose department has no location in Houston.
Tuple relational Calculus:
{ e.LNAME, e.FNAME, e.ADDRESS | EMPLOYEE(e) AND (EXISTS p) (EXISTS w) (
WORKS_ON(w) AND PROJECT(p) AND e.SSN=w.ESSN AND w.PNO=p.PNUMBER AND
p.PLOCATION='Houston' AND NOT(EXISTS l) (DEPT_LOCATIONS(l) AND e.DNO=l.DNUMBER
AND l.DLOCATION='Houston')) }
Domain relational Calculus:
{ qsv | (EXISTS t) (EXISTS z) (EMPLOYEE(qrstuvwxyz) AND (EXISTS b) (EXISTS c)
(EXISTS e)
(EXISTS f) (WORKS_ON(efg) AND PROJECT(abcd) AND t=e AND f=b AND c='Houston' AND
NOT(EXISTS h) NOT(EXISTS i) (DEPT_LOCATIONS(hi) AND z=h AND i='Houston'
)) }
(j) List the last names of department managers who have no dependents.
Tuple relational Calculus:
{ e.LNAME | EMPLOYEE(e) AND (EXISTS d) (DEPARTMENT(d) AND e.SSN=d.MGRSSN AND
NOT(EXISTS x) (DEPENDENT(x) AND e.SSN=x.ESSN)) }
Domain relational Calculus:
{ s | (EXISTS t) (EMPLOYEE(qrstuvwxyz) AND (EXISTS c) (DEPARTMENT(abcd)
AND t=c
AND NOT(EXISTS e) (DEPENDENT(efghi) AND e=t)) }

6.25 - No solution provided.

6.26 Specify queries c, d, and e of Exercise 6.18 in both tuple and domain relational calculus.

Answer:

(c) For each section taught by professor King, retrieve the course number, semester,
year, and number of students who took the section.
This query cannot be done in basic relational calculus as it requires a COUNT function.

(d) Retrieve the name and transcript of each senior student (Class=5) majoring in
COSC. Transcript includes course name, course number, credit hours, semester, year,
and grade for each course completed by the student.
Tuple relational Calculus:
{s.Name, c.CourseName, c.CourseNumber, c.CreditHours, t.Semester, t.Year, g.Grade |
STUDENT(s)
AND COURSE(c) AND SECTION(t) AND GRADE_REPORT(g) AND
s.Class=5 AND
s.Major='COSC' AND s.StudentNumber=g.StudentNumber AND
g.SectionIdentifier=t.SectionIdentifier
AND t.CourseNumber=c.CourseNumber}
Domain relational Calculus:
{aefgklp | (EXISTS b) (EXISTS c) (EXISTS d) (EXISTS n) (EXISTS o) (EXISTS j)
(EXISTS i)
(STUDENT(abcd) AND COURSE(efgh) AND SECTION(ijklm) AND GRADE_REPORT(nop) AND
c=5 AND d='COSC' AND b=n AND i=o AND j=f)}
(e) Retrieve the names and major departments of all straight A students (students who have a grade of A in all their courses).
Tuple relational Calculus:
{ s.Name, s.Major | STUDENT(s) AND (FORALL g) (
NOT(GRADE_REPORT(g)) OR
NOT(s.StudentNumber=g.StudentNumber) OR g.Grade='A') }
Domain relational Calculus:
{ ad | (EXISTS b) (STUDENT(abcd) AND (FORALL e) (FORALL g) (
NOT(GRADE_REPORT(efg)) OR NOT(b=e) OR g='A')) }

6.27 - In a tuple relational calculus query with *n* tuple variables, what would be the typical minimum number of join conditions? Why? What is the effect of having a smaller number of join conditions?

Answer:
Typically, there should be at least (n-1) join conditions; otherwise, a Cartesian product with one of the range relations would be taken, which usually does not make sense.

6.28 - Rewrite the domain relational calculus queries that followed Q0 in Section 6.7 in the style of the abbreviated notation of Q0A, where the objective is to

minimize the number of domain variables by writing constants in place of variables wherever possible.