4
RELATIONAL ALGEBRA AND CALCULUS

Exercise 4.1 Explain the statement that relational algebra operators can be *composed*. Why is the ability to compose operators important?

Answer 4.1 Every operator in relational algebra accepts one or more relation instances as arguments and the result is always an relation instance. So the argument of one operator could be the result of another operator. This is important because, this makes it easy to write complex queries by simply composing the relational algebra operators.

Exercise 4.2 Given two relations $R1$ and $R2$, where $R1$ contains N1 tuples, $R2$ contains N2 tuples, and N2 > N1 > 0, give the minimum and maximum possible sizes (in tuples) for the resulting relation produced by each of the following relational algebra expressions. In each case, state any assumptions about the schemas for $R1$ and $R2$ needed to make the expression meaningful:

> (1) $R1 \cup R2$, (2) $R1 \cap R2$, (3) $R1 - R2$, (4) $R1 \times R2$, (5) $\sigma_{a=5}(R1)$, (6) $\pi_a(R1)$, and (7) $R1/R2$

Answer 4.2 Answer omitted.

Exercise 4.3 Consider the following schema:

> Suppliers(<u>*sid:* `integer`</u>, *sname:* `string`, *address:* `string`)
> Parts(<u>*pid:* `integer`</u>, *pname:* `string`, *color:* `string`)
> Catalog(<u>*sid:* `integer`, *pid:* `integer`</u>, *cost:* `real`)

The key fields are underlined, and the domain of each field is listed after the field name. Therefore *sid* is the key for Suppliers, *pid* is the key for Parts, and *sid* and *pid* together form the key for Catalog. The Catalog relation lists the prices charged for parts by Suppliers. Write the following queries in relational algebra, tuple relational calculus, and domain relational calculus:

1. Find the *name*s of suppliers who supply some red part.
2. Find the *sid*s of suppliers who supply some red or green part.
3. Find the *sid*s of suppliers who supply some red part or are at 221 Packer Street.
4. Find the *sid*s of suppliers who supply some red part and some green part.
5. Find the *sid*s of suppliers who supply every part.
6. Find the *sid*s of suppliers who supply every red part.
7. Find the *sid*s of suppliers who supply every red or green part.
8. Find the *sid*s of suppliers who supply every red part or supply every green part.
9. Find pairs of *sid*s such that the supplier with the first *sid* charges more for some part than the supplier with the second *sid*.
10. Find the *pid*s of parts supplied by at least two different suppliers.
11. Find the *pid*s of the most expensive parts supplied by suppliers named Yosemite Sham.
12. Find the *pid*s of parts supplied by every supplier at less than $200. (If any supplier either does not supply the part or charges more than $200 for it, the part is not selected.)

Answer 4.3 In the answers below RA refers to Relational Algebra, TRC refers to Tuple Relational Calculus and DRC refers to Domain Relational Calculus.

1. ■ RA

$$\pi_{sname}(\pi_{sid}((\pi_{pid}\sigma_{color='red'}Parts) \bowtie Catalog) \bowtie Suppliers)$$

■ TRC

$$\{T \mid \exists T1 \in Suppliers(\exists X \in Parts(X.color =' red' \wedge \exists Y \in Catalog$$
$$(Y.pid = X.pid \wedge Y.sid = T1.sid)) \wedge T.sname = T1.sname)\}$$

■ DRC

$$\{\langle Y\rangle \mid \langle X, Y, Z\rangle \in Suppliers \wedge \exists P, Q, R(\langle P, Q, R\rangle \in Parts$$
$$\wedge R =' red' \wedge \exists I, J, K(\langle I, J, K\rangle \in Catalog \wedge J = P \wedge I = X))\}$$

■ SQL

```
SELECT S.sname
FROM   Suppliers S, Parts P, Catalog C
WHERE  P.color='red' AND C.pid=P.pid AND C.sid=S.sid
```

2. - RA

$$\pi_{sid}(\pi_{pid}(\sigma_{color='red' \vee color='green'} Parts) \bowtie catalog)$$

- TRC

$$\{T \mid \exists T1 \in Catalog(\exists X \in Parts((X.color = \text{`}red' \vee X.color = \text{`}green')$$
$$\wedge X.pid = T1.pid) \wedge T.sid = T1.sid)\}$$

- DRC

$$\{\langle X \rangle \mid \langle X, Y, Z \rangle \in Catalog \wedge \exists A, B, C(\langle A, B, C \rangle \in Parts$$
$$\wedge (C =' red' \vee C =' green') \wedge A = Y)\}$$

- SQL

```
SELECT C.sid
FROM   Catalog C, Parts P
WHERE  (P.color = 'red' OR P.color = 'green')
       AND P.pid = C.pid
```

3. - RA

$$\rho(R1, \pi_{sid}((\pi_{pid}\sigma_{color='red'} Parts) \bowtie Catalog))$$
$$\rho(R2, \pi_{sid}\sigma_{address='221PackerStreet'} Suppliers)$$
$$R1 \cup R2$$

- TRC

$$\{T \mid \exists T1 \in Catalog(\exists X \in Parts(X.color = \text{`}red' \wedge X.pid = T1.pid)$$
$$\wedge T.sid = T1.sid)$$
$$\vee \exists T2 \in Suppliers(T2.address =' 221PackerStreet' \wedge T.sid = T2.sid)\}$$

- DRC

$$\{\langle X \rangle \mid \langle X, Y, Z \rangle \in Catalog \wedge \exists A, B, C(\langle A, B, C \rangle \in Parts$$
$$\wedge C =' red' \wedge A = Y)$$
$$\vee \exists P, Q(\langle X, P, Q \rangle \in Suppliers \wedge Q =' 221PackerStreet')\}$$

- SQL

```
SELECT S.sid
FROM   Suppliers S
WHERE  S.address = '221 Packer street'
       OR S.sid IN ( SELECT C.sid
                     FROM   Parts P, Catalog C
                     WHERE  P.color='red' AND P.pid = C.pid )
```

4. ■ RA

$$\rho(R1, \pi_{sid}((\pi_{pid}\sigma_{color='red'}Parts) \bowtie Catalog))$$
$$\rho(R2, \pi_{sid}((\pi_{pid}\sigma_{color='green'}Parts) \bowtie Catalog))$$
$$R1 \cap R2$$

■ TRC

$$\{T \mid \exists T1 \in Catalog(\exists X \in Parts(X.color = \text{`}red' \wedge X.pid = T1.pid)$$
$$\wedge \exists T2 \in Catalog(\exists Y \in Parts(Y.color =' green' \wedge Y.pid = T2.pid)$$
$$\wedge T2.sid = T1.sid) \wedge T.sid = T1.sid)\}$$

■ DRC

$$\{\langle X\rangle \mid \langle X, Y, Z\rangle \in Catalog \wedge \exists A, B, C(\langle A, B, C\rangle \in Parts$$
$$\wedge C =' red' \wedge A = Y)$$
$$\wedge \exists P, Q, R(\langle P, Q, R\rangle \in Catalog \wedge \exists E, F, G(\langle E, F, G\rangle \in Parts$$
$$\wedge G =' green' \wedge E = Q) \wedge P = X)\}$$

■ SQL

```
SELECT C.sid
FROM   Parts P, Catalog C
WHERE  P.color = 'red' AND P.pid = C.pid
       AND EXISTS ( SELECT P2.pid
                    FROM   Parts P2, Catalog C2
                    WHERE  P2.color = 'green' AND C2.sid = C.sid
                           AND P2.pid = C2.pid )
```

5. ■ RA

$$(\pi_{sid,pid}Catalog)/(\pi_{pid}Parts)$$

■ TRC

$$\{T \mid \exists T1 \in Catalog(\forall X \in Parts(\exists T2 \in Catalog$$
$$(T2.pid = X.pid \wedge T2.sid = T1.sid)) \wedge T.sid = T1.sid)\}$$

- DRC

$$\{\langle X \rangle \mid \langle X, Y, Z \rangle \in Catalog \land \forall \langle A, B, C \rangle \in Parts$$
$$(\exists \langle P, Q, R \rangle \in Catalog(Q = A \land P = X))\}$$

- SQL

```
SELECT C.sid
FROM   Catalog C
WHERE  NOT EXISTS (SELECT P.pid
                   FROM   Parts P
                   WHERE  NOT EXISTS (SELECT C1.sid
                                      FROM   Catalog C1
                                      WHERE  C1.sid = C.sid
                                             AND C1.pid = P.pid))
```

6. - RA

$$(\pi_{sid,pid}Catalog)/(\pi_{pid}\sigma_{color='red'}Parts)$$

- TRC

$$\{T \mid \exists T1 \in Catalog(\forall X \in Parts(X.color \neq \text{`}red'$$
$$\lor \exists T2 \in Catalog(T2.pid = X.pid \land T2.sid = T1.sid))$$
$$\land T.sid = T1.sid)\}$$

- DRC

$$\{\langle X \rangle \mid \langle X, Y, Z \rangle \in Catalog \land \forall \langle A, B, C \rangle \in Parts$$
$$(C \neq \text{`}red' \lor \exists \langle P, Q, R \rangle \in Catalog(Q = A \land P = X))\}$$

- SQL

```
SELECT C.sid
FROM   Catalog C
WHERE  NOT EXISTS (SELECT P.pid
                   FROM   Parts P
                   WHERE  P.color = 'red'
                   AND (NOT EXISTS (SELECT C1.sid
                                    FROM   Catalog C1
                                    WHERE  C1.sid = C.sid AND
                                           C1.pid = P.pid)))
```

7. - RA

$$(\pi_{sid,pid}Catalog)/(\pi_{pid}\sigma_{color='red' \lor color='green'}Parts)$$

- TRC

$$\{T \mid \exists T1 \in Catalog(\forall X \in Parts((X.color \neq `red' \\ \wedge X.color \neq `green') \vee \exists T2 \in Catalog \\ (T2.pid = X.pid \wedge T2.sid = T1.sid)) \wedge T.sid = T1.sid)\}$$

- DRC

$$\{\langle X \rangle \mid \langle X, Y, Z \rangle \in Catalog \wedge \forall \langle A, B, C \rangle \in Parts \\ ((C \neq `red' \wedge C \neq `green') \vee \exists \langle P, Q, R \rangle \in Catalog \\ (Q = A \wedge P = X))\}$$

- SQL

```
SELECT C.sid
FROM   Catalog C
WHERE  NOT EXISTS (SELECT P.pid
                   FROM   Parts P
                   WHERE  (P.color = 'red' OR P.color = 'green')
                   AND (NOT EXISTS (SELECT C1.sid
                                    FROM   Catalog C1
                                    WHERE  C1.sid = C.sid AND
                                           C1.pid = P.pid)))
```

8. - RA

$$\rho(R1, ((\pi_{sid,pid}Catalog)/(\pi_{pid}\sigma_{color='red'}Parts))) \\ \rho(R2, ((\pi_{sid,pid}Catalog)/(\pi_{pid}\sigma_{color='green'}Parts))) \\ R1 \cup R2$$

- TRC

$$\{T \mid \exists T1 \in Catalog((\forall X \in Parts \\ (X.color \neq `red' \vee \exists Y \in Catalog(Y.pid = X.pid \wedge Y.sid = T1.sid)) \\ \vee \forall Z \in Parts(Z.color \neq `green' \vee \exists P \in Catalog \\ (P.pid = Z.pid \wedge P.sid = T1.sid))) \wedge T.sid = T1.sid)\}$$

- DRC

$$\{\langle X \rangle \mid \langle X, Y, Z \rangle \in Catalog \wedge (\forall \langle A, B, C \rangle \in Parts \\ (C \neq `red' \vee \exists \langle P, Q, R \rangle \in Catalog(Q = A \wedge P = X)) \\ \vee \forall \langle U, V, W \rangle \in Parts(W \neq `green' \vee \langle M, N, L \rangle \in Catalog \\ (N = U \wedge M = X)))\}$$

- SQL

```
SELECT C.sid
FROM   Catalog C
WHERE  (NOT EXISTS (SELECT P.pid
                    FROM   Parts P
                    WHERE  P.color = 'red' AND
                    (NOT EXISTS (SELECT C1.sid
                                 FROM   Catalog C1
                                 WHERE  C1.sid = C.sid AND
                                        C1.pid = P.pid))))
       OR ( NOT EXISTS (SELECT P1.pid
                        FROM   Parts P1
                        WHERE  P1.color = 'green' AND
                        (NOT EXISTS (SELECT C2.sid
                                     FROM   Catalog C2
                                     WHERE  C2.sid = C.sid AND
                                            C2.pid = P1.pid))))
```

9. - RA

$$\rho(R1, Catalog)$$
$$\rho(R2, Catalog)$$
$$\pi_{R1.sid,R2.sid}(\sigma_{R1.pid=R2.pid \wedge R1.sid \neq R2.sid \wedge R1.cost > R2.cost}(R1 \times R2))$$

- TRC

$$\{T \mid \exists T1 \in Catalog(\exists T2 \in Catalog$$
$$(T2.pid = T1.pid \wedge T2.sid \neq T1.sid$$
$$\wedge T2.cost < T1.cost \wedge T.sid2 = T2.sid)$$
$$\wedge T.sid1 = T1.sid)\}$$

- DRC

$$\{\langle X, P\rangle \mid \langle X, Y, Z\rangle \in Catalog \wedge \exists P, Q, R$$
$$(\langle P, Q, R\rangle \in Catalog \wedge Q = Y \wedge P \neq X \wedge R < Z)\}$$

- SQL

```
SELECT C1.sid, C2.sid
FROM   Catalog C1, Catalog C2
WHERE  C1.pid = C2.pid AND C1.sid ≠ C2.sid
       AND C1.cost > C2.cost
```

10. ■ RA

$$\rho(R1, Catalog)$$
$$\rho(R2, Catalog)$$
$$\pi_{R1.pid}\sigma_{R1.pid=R2.pid \wedge R1.sid \neq R2.sid}(R1 \times R2)$$

■ TRC

$$\{T \mid \exists T1 \in Catalog(\exists T2 \in Catalog$$
$$(T2.pid = T1.pid \wedge T2.sid \neq T1.sid)$$
$$\wedge T.pid = T1.pid)\}$$

■ DRC

$$\{\langle X \rangle \mid \langle X, Y, Z \rangle \in Catalog \wedge \exists A, B, C$$
$$(\langle A, B, C \rangle \in Catalog \wedge B = Y \wedge A \neq X)\}$$

■ SQL

```
SELECT  C.pid
FROM    Catalog C
WHERE   EXISTS (SELECT  C1.sid
                FROM    Catalog C1
                WHERE   C1.pid = C.pid AND C1.sid ≠ C.sid )
```

11. ■ RA

$$\rho(R1, \pi_{sid}\sigma_{sname='YosemiteSham'}Suppliers)$$
$$\rho(R2, R1 \bowtie Catalog)$$
$$\rho(R3, R2)$$
$$\rho(R4(1 \rightarrow sid, 2 \rightarrow pid, 3 \rightarrow cost), \sigma_{R3.cost<R2.cost}(R3 \times R2))$$
$$\pi_{pid}(R2 - \pi_{sid,pid,cost}R4)$$

■ TRC

$$\{T \mid \exists T1 \in Catalog(\exists X \in Suppliers$$
$$(X.sname =' YosemiteSham' \wedge X.sid = T1.sid) \wedge \neg(\exists S \in Suppliers$$
$$(S.sname =' YosemiteSham' \wedge \exists Z \in Catalog$$
$$(Z.sid = S.sid \wedge Z.cost > T1.cost))) \wedge T.pid = T1.pid)$$

■ DRC

$$\{\langle Y \rangle \mid \langle X, Y, Z \rangle \in Catalog \wedge \exists A, B, C$$
$$(\langle A, B, C \rangle \in Suppliers \wedge C =' YosemiteSham' \wedge A = X)$$
$$\wedge \neg(\exists P, Q, R(\langle P, Q, R \rangle \in Suppliers \wedge R =' YosemiteSham'$$
$$\wedge \exists I, J, K(\langle I, J, K \rangle \in Catalog(I = P \wedge K > Z))))\}$$

- SQL

```
SELECT C.pid
FROM   Catalog C, Suppliers S
WHERE  S.sname = 'Yosemite Sham' AND C.sid = S.sid
       AND C.cost ≥ ALL (Select C2.cost
                         FROM  Catalog C2, Suppliers S2
                         WHERE S2.sname = 'Yosemite Sham'
                               AND C2.sid = S2.sid)
```

Exercise 4.4 Consider the Supplier-Parts-Catalog schema from the previous question. State what the following queries compute:

1. $\pi_{sname}(\pi_{sid}((\sigma_{color='red'}Parts) \bowtie (\sigma_{cost<100}Catalog)) \bowtie Suppliers)$
2. $\pi_{sname}(\pi_{sid}((\sigma_{color='red'}Parts) \bowtie (\sigma_{cost<100}Catalog) \bowtie Suppliers))$
3. $(\pi_{sname}((\sigma_{color='red'}Parts) \bowtie (\sigma_{cost<100}Catalog) \bowtie Suppliers)) \ \cap$

   $$(\pi_{sname}((\sigma_{color='green'}Parts) \bowtie (\sigma_{cost<100}Catalog) \bowtie Suppliers))$$

4. $(\pi_{sid}((\sigma_{color='red'}Parts) \bowtie (\sigma_{cost<100}Catalog) \bowtie Suppliers)) \ \cap$

   $$(\pi_{sid}((\sigma_{color='green'}Parts) \bowtie (\sigma_{cost<100}Catalog) \bowtie Suppliers))$$

5. $\pi_{sname}((\pi_{sid,sname}((\sigma_{color='red'}Parts) \bowtie (\sigma_{cost<100}Catalog) \bowtie Suppliers)) \ \cap$

   $$(\pi_{sid,sname}((\sigma_{color='green'}Parts) \bowtie (\sigma_{cost<100}Catalog) \bowtie Suppliers)))$$

Answer 4.4 The statements can be interpreted as:

1. Find the Supplier names of the suppliers who supply a red part that costs less than 100 dollars.
2. This Relational Algebra statement does not return anything because of the sequence of projection operators. Once the sid is projected, it is the only field in the set. Therefore, projecting on sname will not return anything.
3. Find the Supplier names of the suppliers who supply a red part that costs less than 100 dollars and a green part that costs less than 100 dollars.
4. Find the Supplier ids of the suppliers who supply a red part that costs less than 100 dollars and a green part that costs less than 100 dollars.
5. Find the Supplier names of the suppliers who supply a red part that costs less than 100 dollars and a green part that costs less than 100 dollars.

Exercise 4.5 Consider the following relations containing airline flight information:

Flights(*flno:* `integer`, *from:* `string`, *to:* `string`,
distance: `integer`, *departs:* `time`, *arrives:* `time`)
Aircraft(*aid:* `integer`, *aname:* `string`, *cruisingrange:* `integer`)
Certified(*eid:* `integer`, *aid:* `integer`)
Employees(*eid:* `integer`, *ename:* `string`, *salary:* `integer`)

Note that the Employees relation describes pilots and other kinds of employees as well; every pilot is certified for some aircraft (otherwise, he or she would not qualify as a pilot), and only pilots are certified to fly.

Write the following queries in relational algebra, tuple relational calculus, and domain relational calculus. Note that some of these queries may not be expressible in relational algebra (and, therefore, also not expressible in tuple and domain relational calculus)! For such queries, informally explain why they cannot be expressed. (See the exercises at the end of Chapter 5 for additional queries over the airline schema.)

1. Find the *eid*s of pilots certified for some Boeing aircraft.
2. Find the *name*s of pilots certified for some Boeing aircraft.
3. Find the *aid*s of all aircraft that can be used on non-stop flights from Bonn to Madras.
4. Identify the flights that can be piloted by every pilot whose salary is more than \$100,000.
5. Find the names of pilots who can operate planes with a range greater than 3,000 miles but are not certified on any Boeing aircraft.
6. Find the *eid*s of employees who make the highest salary.
7. Find the *eid*s of employees who make the second highest salary.
8. Find the *eid*s of employees who are certified for the largest number of aircraft.
9. Find the *eid*s of employees who are certified for exactly three aircraft.
10. Find the total amount paid to employees as salaries.
11. Is there a sequence of flights from Madison to Timbuktu? Each flight in the sequence is required to depart from the city that is the destination of the previous flight; the first flight must leave Madison, the last flight must reach Timbuktu, and there is no restriction on the number of intermediate flights. Your query must determine whether a sequence of flights from Madison to Timbuktu exists for *any* input Flights relation instance.

Answer 4.5 In the answers below RA refers to Relational Algebra, TRC refers to Tuple Relational Calculus and DRC refers to Domain Relational Calculus.

1. - RA

$$\pi_{eid}(\sigma_{aname=\text{`}Boeing'}(Aircraft \bowtie Certified))$$

   - TRC

$$\{C.eid \mid C \in Certified \wedge \exists A \in Aircraft(A.aid = C.aid \wedge A.aname = \text{`}Boeing'))\}$$

   - DRC

$$\{\langle Ceid \rangle \mid \langle Ceid, Caid \rangle \in Certified \wedge \exists Aid, AN, AR(\langle Aid, AN, AR \rangle \in Aircraft \wedge Aid = Caid \wedge AN = \text{`}Boeing'))\}$$

   - SQL

```
SELECT C.eid
FROM   Aircraft A, Certified C
WHERE  A.aid = C.aid AND A.aname = 'Boeing'
```

2. - RA

$$\pi_{ename}(\sigma_{aname=\text{`}Boeing'}(Aircraft \bowtie Certified \bowtie Employees))$$

   - TRC

$\{E.ename \mid E \in Employees \wedge \exists C \in Certified$
$(\exists A \in Aircraft(A.aid = C.aid \wedge A.aname = \text{`}Boeing' \wedge E.eid = C.eid))\}$

   - DRC

$\{\langle EN \rangle \mid \langle Eid, EN, ES \rangle \in Employees \wedge$
$\exists Ceid, Caid(\langle Ceid, Caid \rangle \in Certified \wedge$
$\exists Aid, AN, AR(\langle Aid, AN, AR \rangle \in Aircraft \wedge$
$Aid = Caid \wedge AN = \text{`}Boeing' \wedge Eid = Ceid)\}$

   - SQL

```
SELECT E.ename
FROM   Aircraft A, Certified C, Employees E
WHERE  A.aid = C.aid AND A.aname = 'Boeing' AND E.eid = C.eid
```

3. ■ RA

$\rho(BonnToMadrid, \sigma_{from='Bonn' \wedge to='Madrid'}(Flights))$
$\pi_{aid}(\sigma_{cruisingrange>distance}(Aircraft \times BonnToMadrid))$

■ TRC

$\{A.aid \mid A \in Aircraft \wedge \exists F \in Flights$
$(F.from = 'Bonn' \wedge F.to = 'Madrid' \wedge A.cruisingrange > F.distance)\}$

■ DRC

$\{Aid \mid \langle Aid, AN, AR\rangle \in Aircraft \wedge$
$(\exists FN, FF, FT, FDi, FDe, FA(\langle FN, FF, FT, FDi, FDe, FA\rangle \in Flights \wedge$
$FF = 'Bonn' \wedge FT = 'Madrid' \wedge FDi < AR))\}$

■ SQL

```
SELECT A.aid
FROM   Aircraft A, Flights F
WHERE  F.from = 'Bonn' AND F.to = 'Madrid' AND
       A.cruisingrange > F.distance
```

4. ■ RA

$\pi_{flno}(\sigma_{distance<cruisingrange \wedge salary>100{,}000}(Flights \bowtie Aircraft \bowtie$
$Certified \bowtie Employees)))$

■ TRC $\{F.flno \mid F \in Flights \wedge \exists A \in Aircraft \exists C \in Certified$
$\exists E \in Employees(A.cruisingrange > F.distance \wedge E.salary > 100,000 \wedge$
$A.aid = C.aid \wedge E.eid = C.eid)\}$

■ DRC

$\{FN \mid \langle FN, FF, FT, FDi, FDe, FA\rangle \in Flights \wedge$
$\exists Ceid, Caid(\langle Ceid, Caid\rangle \in Certified \wedge$
$\exists Aid, AN, AR(\langle Aid, AN, AR\rangle \in Aircraft \wedge$
$\exists Eid, EN, ES(\langle Eid, EN, ES\rangle \in Employees$
$(AR > FDi \wedge ES > 100,000 \wedge Aid = Caid \wedge Eid = Ceid)\}$

■ SQL

```
SELECT E.ename
FROM   Aircraft A, Certified C, Employees E, Flights F
WHERE  A.aid = C.aid AND E.eid = C.eid AND
       distance < cruisingrange AND salary > 100,000
```

5. ■ RA $\rho(R1, \pi_{eid}(\sigma_{cruisingrange>3000}(Aircraft \bowtie Certified)))$
$\pi_{ename}(Employees \bowtie (R1 - \pi_{eid}(\sigma_{aname=\text{'}Boeing'}(Aircraft \bowtie Certified))))$

■ TRC
$\{E.ename \mid E \in Employees \wedge \exists C \in Certified(\exists A \in Aircraft$
$(A.aid = C.aid \wedge E.eid = C.eid \wedge A.cruisingrange > 3000))\wedge$
$\neg(\exists C2 \in Certified(\exists A2 \in Aircraft(A2.aname = \text{'}Boeing' \wedge C2.aid = A2.aid \wedge C2.eid = E.eid)))\}$

■ DRC
$\{\langle EN\rangle \mid \langle Eid, EN, ES\rangle \in Employees\wedge$
$\exists Ceid, Caid(\langle Ceid, Caid\rangle \in Certified\wedge$
$\exists Aid, AN, AR(\langle Aid, AN, AR\rangle \in Aircraft\wedge$
$Aid = Caid \wedge Eid = Ceid \wedge AR > 3000))\wedge$
$\neg(\exists Aid2, AN2, AR2(\langle Aid2, AN2, AR2\rangle \in Aircraft\wedge$
$\exists Ceid2, Caid2(\langle Ceid2, Caid2\rangle \in Certified$
$\wedge Aid2 = Caid2 \wedge Eid = Ceid2 \wedge AN2 = \text{'}Boeing')))\}$

■ SQL

```
SELECT E.ename
FROM   Certified C, Employees E, Aircraft A
WHERE  A.aid = C.aid AND E.eid = C.eid AND A.cruisingrange > 3000
AND E.eid NOT IN ( SELECT C2.eid
FROM Certified C2, Aircraft A2
WHERE C2.aid = A2.aid AND A2.aname = 'Boeing' )
```

6. ■ RA

The approach to take is first find all the employees who do not have the highest salary. Subtract these from the original list of employees and what is left is the highest paid employees.

$\rho(E1, Employees)$
$\rho(E2, Employees)$
$\rho(E3, \pi_{E2.eid}(E1 \bowtie_{E1.salary>E2.salary} E2)$
$(\pi_{eid}E1) - E3$

■ TRC

$\{E1.eid \mid E1 \in Employees\wedge\neg(\exists E2 \in Employees(E2.salary > E1.salary))\}$

■ DRC

$\{\langle Eid1\rangle \mid \langle Eid1, EN1, ES1\rangle \in Employees \wedge$
$\neg(\exists Eid2, EN2, ES2(\langle Eid2, EN2, ES2\rangle \in Employees \wedge ES2 > ES1))\}$

- SQL

```
SELECT E.eid
FROM   Employees E
WHERE  E.salary = ( Select MAX (E2.salary)
                    FROM  Employees E2 )
```

7. - RA

The approach taken is similar to the solution for the previous exercise. First find all the employees who do not have the highest salary. Remove these from the original list of employees and what is left is the highest paid employees. Remove the highest paid employees from the original list. What is left is the second highest paid employees together with the rest of the employees. Then find the highest paid employees of this new list. This is the list of the second highest paid employees.

$\rho(E1, Employees)$
$\rho(E2, Employees)$
$\rho(E3, \pi_{E2.eid}(E1 \bowtie_{E1.salary > E2.salary} E2)$
$\rho(E4, E2 \bowtie E3)$
$\rho(E5, E2 \bowtie E3)$
$\rho(E6, \pi_{E5.eid}(E4 \bowtie_{E1.salary > E5.salary} E5)$
$(\pi_{eid} E3) - E6$

- TRC

$\{E1.eid \mid E1 \in Employees \wedge \exists E2 \in Employees(E2.salary > E1.salary$
$\wedge \neg(\exists E3 \in Employees(E3.salary > E2.salary)))\}$

- DRC

$\{\langle Eid1\rangle \mid \langle Eid1, EN1, ES1\rangle \in Employees \wedge$
$\exists Eid2, EN2, ES2(\langle Eid2, EN2, ES2\rangle \in Employees(ES2 > ES1)$
$\wedge \neg(\exists Eid3, EN3, ES3(\langle Eid3, EN3, ES3\rangle \in Employees(ES3 > ES2))))\}$

- SQL

```
SELECT E.eid
FROM   Employees E
WHERE  E.salary = (SELECT MAX (E2.salary)
                   FROM   Employees E2
                   WHERE  E2.salary ≠ (SELECT MAX (E3.salary)
                                       FROM   Employees E3 ))
```

8. This cannot be expressed in relational algebra (or calculus) because there is no operator to count, and this query requires the ability to count up to a number that depends on the data. The query can however be expressed in SQL as follows:

```
SELECT Temp.eid
FROM   ( SELECT    C.eid AS eid, COUNT (C.aid) AS cnt,
         FROM      Certified C
         GROUP BY  C.eid) AS Temp
WHERE  Temp.cnt = ( SELECT   MAX (Temp.cnt)
                    FROM     Temp)
```

9. ■ RA

   The approach behind this query is to first find the employees who are certified for at least three aircraft (they appear at least three times in the Certified relation). Then find the employees who are certified for at least four aircraft. Subtract the second from the first and what is left is the employees who are certified for exactly three aircraft.

   $\rho(R1, Certified)$
   $\rho(R2, Certified)$
   $\rho(R3, Certified)$
   $\rho(R4, Certified)$
   $\rho(R5, \pi_{eid}(\sigma_{(R1.eid=R2.eid=R3.eid)\wedge(R1.aid\neq R2.aid\neq R3.aid)}(R1 \times R2 \times R3)))$
   $\rho(R6, \pi_{eid}(\sigma_{(R1.eid=R2.eid=R3.eid=R4.eid)\wedge(R1.aid\neq R2.aid\neq R3.aid\neq R4.aid)}$
   $(R1 \times R2 \times R3 \times R4)))$
   $R5 - R6$

   ■ TRC

   $\{C1.eid \mid C1 \in Certified \wedge \exists C2 \in Certified(\exists C3 \in Certified$
   $(C1.eid = C2.eid \wedge C2.eid = C3.eid\wedge$
   $C1.aid \neq C2.aid \wedge C2.aid \neq C3.aid \wedge C3.aid \neq C1.aid\wedge$
   $\neg(\exists C4 \in Certified$
   $(C3.eid = C4.eid \wedge C1.aid \neq C4.aid\wedge$
   $C2.aid \neq C4.aid \wedge C3.aid \neq C4.aid))))\}$

   ■ DRC

   $\{\langle CE1\rangle \mid \langle CE1, CA1\rangle \in Certified\wedge$
   $\exists CE2, CA2(\langle CE2, CA2\rangle \in Certified\wedge$
   $\exists CE3, CA3(\langle CE3, CA3\rangle \in Certified\wedge$
   $(CE1 = CE2 \wedge CE2 = CE3\wedge$
   $CA1 \neq CA2 \wedge CA2 \neq CA3 \wedge CA3 \neq CA1\wedge$
   $\neg(\exists CE4, CA4(\langle CE4, CA4\rangle \in Certified\wedge$

$(CE3 = CE4 \wedge CA1 \neq CA4 \wedge$
$CA2 \neq CA4 \wedge CA3 \neq CA4))))\}$

- SQL

```
SELECT C1.eid
FROM   Certified C1, Certified C2, Certified C3
WHERE  (C1.eid = C2.eid AND C2.eid = C3.eid AND
       C1.aid ≠ C2.aid AND C2.aid ≠ C3.aid AND C3.aid ≠ C1.aid)
EXCEPT
SELECT C4.eid
FROM   Certified C4, Certified C5, Certified C6, Certified C7,
WHERE  (C4.eid = C5.eid AND C5.eid = C6.eid AND C6.eid = C7.eid AND
       C4.aid ≠ C5.aid AND C4.aid ≠ C6.aid AND C4.aid ≠ C7.aid AND
       C5.aid ≠ C6.aid AND C5.aid ≠ C7.aid AND C6.aid ≠ C7.aid )
```

This could also be done in SQL using COUNT.

10. This cannot be expressed in relational algebra (or calculus) because there is no operator to sum values. The query can however be expressed in SQL as follows:

```
SELECT SUM (E.salaries)
FROM   Employees E
```

11. This cannot be expressed in relational algebra or relational calculus or SQL. The problem is that there is no restriction on the number of intermediate flights. All of the query methods could find if there was a flight directly from Madison to Timbuktu and if there was a sequence of two flights that started in Madison and ended in Timbuktu. They could even find a sequence of n flights that started in Madison and ended in Timbuktu as long as there is a static (i.e., data-independent) upper bound on the number of intermediate flights. (For large n, this would of course be long and impractical, but at least possible.) In this query, however, the upper bound is not static but dynamic (based upon the set of tuples in the Flights relation).

    In summary, if we had a static upper bound (say k), we could write an algebra or SQL query that repeatedly computes (upto k) joins on the Flights relation. If the upper bound is dynamic, then we cannot write such a query because k is not known when writing the query.

Exercise 4.6 What is *relational completeness*? If a query language is relationally complete, can you write any desired query in that language?

Answer 4.6 Answer omitted.

Exercise 4.7 What is an *unsafe* query? Give an example and explain why it is important to disallow such queries.

Answer 4.7 An $unsafe$ query is a query in relational calculus that has an infinite number of results. An example of such a query is:

$$\{S \mid \neg(S \in Sailors)\}$$

The query is for all things that are not sailors which of course is everything else. Clearly there is an infinite number of answers, and this query is $unsafe$. It is important to disallow $unsafe$ queries because we want to be able to get back to users with a list of all the answers to a query after a finite amount of time.