Exercise 5.1.1

As a set:

speed
2.66
2.10
1.42
2.80
3.20
2.20
2.00
1.86
3.06

Average = 2.37

As a bag:

speed
2.66
2.10
1.42
2.80
3.20
3.20
2.20
2.20
2.00
2.80
1.86
2.80
3.06

Average = 2.48

Exercise 5.1.2

As a set:

hd
250
80
320
200
300
160

Average = 218

As a bag:

hd
250
250
80
250
250
320
200
250
250
300
160
160
80

Average = 215

Exercise 5.1.3a

As a set:

bore
15
16
14
18

As a bag:

bore
15
16
14
16
15
15
14
18

Exercise 5.1.3b

π_{bore}(Ships $\bowtie$ Classes)

Exercise 5.1.4a

For bags:

On the left-hand side:

Given bags R and S where a tuple *t* appears *n* and *m* times respectively, the union of bags R and S will have tuple *t* appear *n* + *m* times. The further union of bag T with the tuple *t* appearing *o* times will have tuple *t* appear *n* + *m* + *o* times in the final result.

On the right-hand side:

Given bags S and T where a tuple *t* appears *m* and *o* times respectively, the union of bags R and S will have tuple *t* appear *m* + *o* times. The further union of bag R with the tuple *t* appearing *n* times will have tuple *t* appear *m* + *o* + *n* times in the final result.

For sets:

This is a similar case when dealing with bags except the tuple *t* can only appear at most once in each set. The tuple *t* only appears in the result if all the sets have the tuple *t*. Otherwise, the tuple *t* will not appear in the result. Since we cannot have duplicates, the result only has at most one copy of the tuple *t*.

Exercise 5.1.4b

For bags:

On the left-hand side:

Given bags R and S where a tuple t appears n and m times respectively, the intersection of bags R and S will have tuple t appear $min(n, m)$ times. The further intersection of bag T with the tuple t appearing o times will produce tuple t $min(o, min(n, m))$ times in the final result.

On the right-hand side:

Given bags S and T where a tuple t appears m and o times respectively, the intersection of bags R and S will have tuple t appear $min(m, o)$ times. The further intersection of bag R with the tuple t appearing n times will produce tuple t $min(n, min(m, o))$ times in the final result.

The intersection of bags R,S and T will yield a result where tuple t appears $min(n,m,o)$ times.

For sets:

This is a similar case when dealing with bags except the tuple t can only appear at most once in each set. The tuple t only appears in the result if all the sets have the tuple t. Otherwise, the tuple t will not appear in the result.

Exercise 5.1.4c

For bags:

On the left-hand side:

Given that tuple r in R, which appears m times, can successfully join with tuples in S, which appears n times, we expect the result to contain mn copies. Also given that tuple t in T, which appears o times, can successfully join with the joined tuples of r and s, we expect the final result to have mno copies.

On the right-hand side:

Given that tuple s in S, which appears n times, can successfully join with tuple t in T, which appears o times, we expect the result to contain no copies. Also given that tuple r in R, which appears m times, can successfully join with the joined tuples of s and t, we expect the final result to have mno copies.

The order in which we perform the natural join does not matter for bags.

For sets:

This is a similar case when dealing with bags except the joined tuples can only appear at most once in each result. If there are tuples r,s,t in relations R,S,T that can successfully join, then the result will contain a tuple with the schema of their joined attributes.

Exercise 5.1.4d

For bags:

Suppose a tuple t occurs n and m times in bags R and S respectively. In the union of these two bags $R \cup S$, tuple t would appear $n + m$ times. Likewise, in the union of these two bags $S \cup R$, tuple t would appear $m + n$ times. Both sides of the relation yield the same result.

For sets:

A tuple t can only appear at most one time. Tuple t might appear each in sets R and S one or zero times. The combinations of number of occurrences for tuple t in R and S respectively are (0,0), (0,1), (1,0), and (1,1). Only when tuple t appears in both sets R and S will the union $R \cup S$ have the tuple t. The same reasoning holds when we take the union $S \cup R$.

Therefore the commutative law for union holds.

Exercise 5.1.4e

For bags:

Suppose a tuple t occurs n and m times in bags R and S respectively. In the intersection of these two bags $R \cap S$, tuple t would appear $\min(n,m)$ times. Likewise in the intersection of these two bags $S \cap R$, tuple t would appear $\min(m,n)$ times. Both sides of the relation yield the same result.

For sets:

A tuple t can only appear at most one time. Tuple t might appear each in sets R and S one or zero times. The combinations of number of occurrences for tuple t in R and S respectively are (0,0), (0,1), (1,0), and (1,1). Only when tuple t appears in at least one of the sets R and S will the intersection $R \cap S$ have the tuple t. The same reasoning holds when we take the intersection $S \cap R$.

Therefore the commutative law for intersection holds.

Exercise 5.1.4f

For bags:

Suppose a tuple t occurs n times in bag R and tuple u occurs m times in bag S. Suppose also that the two tuples t, u can successfully join. Then in the natural join of these two bags $R \bowtie S$, the joined tuple would appear nm times. Likewise in the natural join of these two bags $S \bowtie R$, the joined tuple would appear mn times. Both sides of the relation yield the same result.

For sets:

An arbitrary tuple t can only appear at most one time in any set. Tuples u, v might appear respectively in sets R and S one or zero times. The combinations of number of occurrences for tuples u, v in R and S respectively are (0,0), (0,1), (1,0), and (1,1). Only when tuple u exists in R

and tuple v exists in S will the natural join R $\bowtie$ S have the joined tuple. The same reasoning holds when we take the natural join S $\bowtie$ R.

Therefore the commutative law for natural join holds.

Exercise 5.1.4g

For bags:

Suppose tuple t appears m times in R and n times in S. If we take the union of R and S first, we will get a relation where tuple t appears $m + n$ times. Taking the projection of a list of attributes L will yield a resulting relation where the projected attributes from tuple t appear $m + n$ times. If we take the projection of the attributes in list L first, then the projected attributes from tuple t would appear m times from R and n times from S. The union of these resulting relations would have the projected attributes of tuple t appear $m + n$ times.

For sets:

An arbitrary tuple t can only appear at most one time in any set. Tuple t might appear in sets R and S one or zero times. The combinations of number of occurrences for tuple t in R and S respectively are (0,0), (0,1), (1,0), and (1,1). Only when tuple t exists in R or S (or both R and S) will the projected attributes of tuple t appear in the result.

Therefore the law holds.

Exercise 5.1.4h

For bags:

Suppose tuple t appears u times in R, v times in S and w times in T. On the left hand side, the intersection of S and T would produce a result where tuple t would appear $\min(v , w)$ times. With the addition of the union of R, the overall result would have $u + \min(v , w)$ copies of tuple t. On the right hand side, we would get a result of $\min(u + v, u + w)$ copies of tuple t. The expressions on both the left and right sides are equivalent.

For sets:

An arbitrary tuple t can only appear at most one time in any set. Tuple t might appear in sets R,S and T one or zero times. The combinations of number of occurrences for tuple t in R, S and T respectively are (0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0) and (1,1,1). Only when tuple t appears in R or in both S and T will the result have tuple t.

Therefore the distributive law of union over intersection holds.

Exercise 5.1.4i

Suppose that in relation R,u tuples satisfy condition C and$v$ tuples satisfy condition D. Suppose also that$w$ tuples satisfy both conditions C and D where$w \leq \min(v, w)$. Then the left hand side will return those w tuples. On the right hand side, $\sigma_C(R)$ produces$u$ tuples and $\sigma_D(R)$ produces$v$ tuples. However, we know the intersection will produce the same$w$ tuples in the result.

When considering bags and sets, the only difference is bags allow duplicate tuples while sets only allow one copy of the tuple. The example above applies to both cases.

Therefore the law holds.

Exercise 5.1.5a

For sets, an arbitrary tuple$t$ appears on the left hand side if it appears in both R,S and not in T. The same is true for the right hand side.

As an example for bags, suppose that tuple$t$ appears one time each in both R,T and two times in S. The result of the left hand side would have zero copies of tuple$t$ while the right hand side would have one copy of tuple$t$.

Therefore the law holds for sets but not for bags.

Exercise 5.1.5b

For sets, an arbitrary tuple$t$ appears on the left hand side if it appears in R and either S or T. This is equivalent to saying tuple$t$ only appears when it is in at least R and S or in R and T. The equivalence is exactly the right side’s expression.

As an example for bags, suppose that tuple$t$ appears one time in R and two times each in S and T. Then the left hand side would have one copy of tuple$t$ in the result while the right hand side would have two copies of tuple$t$.

Therefore the law holds for sets but not for bags.

Exercise 5.1.5c

For sets, an arbitrary tuple$t$ appears on the left hand side if it satisfies condition C, condition D or both condition C and D. On the right hand side,$\sigma_C(R)$ selects those tuples that satisfy condition C while $\sigma_D(R)$ selects those tuples that satisfy condition D. However, the union operator will eliminate duplicate tuples, namely those tuples that satisfy both condition C and D. Thus we are ensured that both sides are equivalent.

As an example for bags, we only need to look at the union operator. If there are indeed tuples that satisfy both conditions C and D, then the right hand side will contain duplicate copies of those tuples. The left hand side, however, will only have one copy for each tuple of the original set of tuples.

Exercise 5.2.1a

A+B	A^2	B^2
1	0	1
5	4	9
1	0	1
6	4	16
7	9	16

Exercise 5.2.1b

B+1	C-1
1	0
3	3
3	4
4	3
1	1
4	3

Exercise 5.2.1c

A	B
0	1
0	1
2	3
2	4
3	4

Exercise 5.2.1d

B	C
0	1
0	2
2	4
2	5
3	4
3	4

Exercise 5.2.1e

A	B
0	1
2	3
2	4
3	4

Exercise 5.2.1f

B	C
0	1
2	4
2	5
3	4
0	2

Exercise 5.2.1g

A	SUM(B)
0	2
2	7
3	4

Exercise 5.2.1h

B	AVG(C)
0	1.5
2	4.5
3	4

Exercise 5.2.1i

A
0
2
3

Exercise 5.2.1j

A	MAX(C)
2	4

Exercise 5.2.1k

A	B	C
2	3	4
2	3	4
0	1	⊥
0	1	⊥
2	4	⊥
3	4	⊥

Exercise 5.2.1l

A	B	C
2	3	4
2	3	4
⊥	0	1
⊥	2	4
⊥	2	5
⊥	0	2

Exercise 5.2.1m

A	B	C
2	3	4
2	3	4
0	1	⊥
0	1	⊥
2	4	⊥
3	4	⊥
⊥	0	1
⊥	2	4
⊥	2	5
⊥	0	2

Exercise 5.2.1n

A	R.B	S.B	C
0	1	2	4
0	1	2	5
0	1	3	4
0	1	3	4
0	1	2	4
0	1	2	5
0	1	3	4
0	1	3	4
2	3	⊥	⊥
2	4	⊥	⊥
3	4	⊥	⊥
⊥	⊥	0	1
⊥	⊥	0	2

Exercise 5.2.2a

Applying the δ operator on a relation with no duplicates will yield the same relation. Thus δ is idempotent.

Exercise 5.2.2b

The result of π_L is a relation over the list of attributes L. Performing the projection again will return the same relation because the relation only contains the list of attributes L. Thus π_L is idempotent.

Exercise 5.2.2c

The result of σ_C is a relation where condition C is satisfied by every tuple. Performing the selection again will return the same relation because the relation only contains tuples that satisfy the condition C. Thus σ_C is idempotent.

Exercise 5.2.2d

The result of γ_L is a relation whose schema consists of the grouping attributes and the aggregated attributes. If we perform the same grouping operation, there is no guarantee that the expression would make sense. The grouping attributes will still appear in the new result. However, the aggregated attributes may or may not appear correctly. If the aggregated attribute is given a different name than the original attribute, then performing γ_L would not make sense because it contains an aggregation for an attribute name that does not exist. In this case, the resulting

relation would, according to the definition, only contain the grouping attributes. Thus, $_L$ is not idempotent.

Exercise 5.2.2e

The result of τ is a sorted list of tuples based on some attributes L. If L is not the entire schema of relation R, then there are attributes that are not sorted on. If in relation R there are two tuples that agree in all attributes L and disagree in some of the remaining attributes not in L, then it is arbitrary as to which order these two tuples appear in the result. Thus, performing the operation multiple times can yield a different relation where these two tuples are swapped. Thus, idempotent.

Exercise 5.2.3

If we only consider sets, then it is possible. We can take π_A(R) and do a product with itself. From this product, we take the tuples where the two columns are equal to each other.

If we consider bags as well, then it is not possible. Take the case where we have the two tuples (1,0) and (1,0). We wish to produce a relation that contains tuples (1,1) and (1,1). If we use the classical operations of relational algebra, we can either get a result where there are no tuples or four copies of the tuple (1,1). It is not possible to get the desired relation because no operation can distinguish between the original tuples and the duplicated tuples. Thus it is not possible to get the relation with the two tuples (1,1) and (1,1).

Exercise 5.3.1

a) Answer(model) ← PC(model,speed,_,_,_) AND speed ≥ 3.00

b) Answer(maker) ← Laptop(model,_,_,hd,_,_) AND Product(maker,model,_) AND hd ≥ 100

c) Answer(model,price) ← PC(model,_,_,_,price) AND Product(maker,model,_) AND maker= ’ B ’
Answer(model,price) ← Laptop(model,_,_,_,price) AND Product(maker,model,_) AND maker= ’ B ’
Answer(model,price) ← Printer(model,_,_,price) AND Product(maker,model,_) AND maker= ’ B ’

d) Answer(model) ← Printer(model,color,type,_) AND color= ’ true ’ AND type= ’ laser

e) PCMaker(maker) ← Product(maker,_,type) AND type= ’ pc ’
LaptopMaker(maker) ← Product(maker,_,type) AND type= ’ laptop ’
Answer(maker) ← LaptopMaker(maker) AND NOT PCMaker(maker)

f) Answer(hd) ← PC(model1,_,_,hd,_) AND PC(model2,_,_,hd,_) AND model1 <> model2

g) Answer(model1,model2) ← PC(model1,speed, ram,_,_) AND PC(model2,_speed,ram,_,_) AND model1 < model2

h) FastComputer(model) ← PC(model,speed,_,_,_) AND speed ≥ 2.80
FastComputer(model) ← Laptop(model,speed,_,_,_,_) AND speed ≥ 2.80
Answer(maker) ← Product(maker,model1,_) AND Product(maker,model2,_) AND FastComputer(model1) AND FastComputer(model2) AND model1 <> model2

i) Computers(model,speed) ← PC(model,speed,_,_,_)
Computers(model,speed) ← Laptop(model,speed,_,_,_,_)
SlowComputers(model) ←Computers(model,speed) AND Computers(model1,speed1) AND speed < speed1
FastestComputers(model) ← Computers(model,_) AND NOT SlowComputers(model)
Answer(maker) ← FastestComputers(model) AND Product(maker,model,_)

j) PCs(maker,speed) ← PC(model,speed,_,_,_) AND Product(maker,model,_)
Answer(maker) ← PCs(maker,speed) AND PCs(maker,speed1) AND PCs(maker,speed2) AND speed <> speed1 AND speed <> speed2 AND speed1 <> speed2

k) PCs(maker,model) ← Product(maker,model,type) AND type= ’pc’
Answer(maker) ← PCs(maker,model) AND PCs(maker,model1) AND PCs(maker,model2) AND PCs(maker,model3) AND model <> model1 AND model <> model2 AND model1 <> model2 AND (model3 = model OR model3 = model1 OR model3 = model2)

Exercise 5.3.2

a) Answer(class,country) ← Classes(class,_,country,_,bore,_) AND bore ≥ 16

b) Answer(name) ← Ships(name,_,launched) AND launched < 1921

c) Answer(ship) ← Outcomes(ship,battle,result) AND battle= ’Denmark Strait’ AND result = ‘sunk’

d) Answer(name) ← Classes(class,_,_,_,_,displacement) AND Ships(name,class,launched) AND displacement > 35000 AND launched > 1921

e) Answer(name,displacement,numGuns) ← Classes(class,_,_,numGuns,_,displacement) AND Ships(name,class,_) AND Outcomes (ship,battle,_) AND battle= ’Guadalcanal’ AND ship=name

f) Answer(name) ← Ships(name,_,_)
Answer(name) ← Outcomes(name,_,_) AND NOT Answer(name)

g) MoreThanOne(class) ← Ships(name,class,_) AND Ships(name1,class,_) AND name <> name1
Answer(class) ← Classes(class,_,_,_,_,_) AND NOT MoreThanOne(class)

h) Battleship(country) ← Classes(_,type,country,_,_,_) AND type= ’bb’
Battlecruiser(country) ← Classes(_,type,country,_,_,_) AND type= ’bc’
Answer(country) ← Battleship(country) AND Battlecruiser(country)

i) Results(ship,result,date) ← Battles(name,date) AND Outcomes(ship,battle,result) AND battle=name
Answer(ship) ← Results(ship,result,date) AND Results(ship,_,date1) AND result= ’damaged’ AND date < date1

Exercise 5.3.3

Answer(x,y) ← R(x,y) AND≠z z

Exercise 5.4.1a

Answer(a,b,c) ← R(a,b,c)
Answer(a,b,c) ← S(a,b,c)

Exercise 5.4.1b

Answer(a,b,c) ← R(a,b,c) AND S(a,b,c)

Exercise 5.4.1c

Answer(a,b,c) ← R(a,b,c) AND NOT S(a,b,c)

Exercise 5.4.1d

Union(a,b,c) ← R(a,b,c)
Union(a,b,c) ← S(a,b,c)
Answer(a,b,c) ← Union(a,b,c) AND NOT T(a,b,c)

Exercise 5.4.1e

J(a,b,c) ← R(a,b,c) AND NOT S(a,b,c)
K(a,b,c) ← R(,a,b,c) AND NOT T(a,b,c)
Answer(a,b,c) ← J(a,b,c) AND K(a,b,c)

Exercise 5.4.1f

Answer(a,b) ← R(a,b,_)

Exercise 5.4.1g

J(a,b) ← R(a,b,_)
K(a,b) ← S(_,a,b)
Answer(a,b) ← J(a,b) AND K(a,b)

Exercise 5.4.2a

Answer(x,y,z) ← R(x,y,z) AND x = y

Exercise 5.4.2b

Answer(x,y,z) ← R(x,y,z) AND x < y AND y < z

Exercise 5.4.2c

Answer(x,y,z) ← R(x,y,z) AND x < y
Answer(x,y,z) ← R(x,y,z) AND y < z

Exercise 5.4.2d

Change: NOT(x < y OR x > y)
To: x ≥ y AND x ≤ y

The above simplifies to x = y

Answer(x,y,z) ← R(x,y,z) AND x = y

Exercise 5.4.2e

Change: NOT((x < y OR x > y) AND y < z)
NOT(x < y OR x > y) OR y ≥ z
(x ≥ y AND x ≤ y) OR y ≥ z
To: x = y OR y ≥ z

Answer(x,y,z) ← R(x,y,z) AND x = y
Answer(x,y,z) ← R(x,y,z) AND y ≥ z

Exercise 5.4.2f

Change: NOT((x < y OR x < z) AND y < z)
NOT(x < y OR x < z) OR y ≥ z
To: (x ≥ y AND x ≥ z) OR y ≥ z

Answer(x,y,z) ← R(x,y,z) AND x ≥ y AND x ≥ z
Answer(x,y,z) ← R(x,y,z) AND yz ≥

Exercise 5.4.3a

Answer(a,b,c,d) ← R(a,b,c) AND S(b,c,d)

Exercise 5.4.3b

Answer(b,c,d,e) ← S(b,c,d) AND T(d,e)

Exercise 5.4.3c

Answer(a,b,c,d,e) ← R(a,b,c) AND S(b,c,d) AND T(d,e)

Exercise 5.4.4

a) Answer(rx,ry,rz,sx,sy,sz) ← R(rx,ry,rz) AND S(sx,sy,sz) AND rx = sy

b) Answer(rx,ry,rz,sx,sy,sz) ← R(rx,ry,rz) AND S(sx,sy,sz) AND rx < sy AND ry < sz

c) Answer(rx,ry,rz,sx,sy,sz) ← R(rx,ry,rz) AND S(sx,sy,sz) AND rx < sy
   Answer(rx,ry,rz,sx,sy,sz) ← R(rx,ry,rz) AND S(sx,sy,sz) AND ry < sz

d) Answer(rx,ry,rz,sx,sy,sz) ← R(rx,ry,rz) AND S(sx,sy,sz) AND rx = sy

e) Answer(rx,ry,rz,sx,sy,sz) ← R(rx,ry,rz) AND S(sx,sy,sz) AND rx = sy
   Answer(rx,ry,rz,sx,sy,sz) ← R(rx,ry,rz) AND S(sx,sy,sz) AND ry ≥ sz

f) Answer(rx,ry,rz,sx,sy,sz) ← R(rx,ry,rz) AND S(sx,sy,sz) AND rx ≥ sy AND rx ≥ s
   Answer(rx,ry,rz,sx,sy,sz) ← R(rx,ry,rz) AND S(sx,sy,sz) AND ry ≥ sz

Exercise 5.4.5a

$R1 := \pi_{x,y}(Q \bowtie R)$

Exercise 5.4.5b

$R1 := \rho_{R1(x,z)}(Q)$
$R2 := \rho_{R2(z,y)}(Q)$
$R3 := \pi_{x,y}(R1 \bowtie_{(R1.z = R2.z)} R2)$

Exercise 5.4.5c

$R1 := \pi_{x,y}(Q \bowtie R)$
$R2 := \sigma_{x<y}(R1)$