Question 1 [Short Answers – 45 points]. Consider the following relation schemas

```
R(A,B)
T(B,C)
```

and their instances

r(A,B) = {<a,b>,<a,c>,<c,d>,<b,e>}
t(B,C) = {<b,e>,<d,h>,<b,f>,<a,d>,<a,e>}

(1.a) [10 points] Compute the result of the following relational algebra expression:

$$\pi_{A,C}(r \bowtie_{\text{LEFT}} t)$$

Answer:

A	C
a	e
a	f
c	h
a	NULL
b	NULL

(1.b) [15 points] Express $r \bowtie_{\text{LEFT}} t$ in (1.a) using ***only*** the primary relational algebra operators ($\cup$, $\cap$, $-$, ρ, σ, π, $\times$) according to the definition of $\bowtie_{\text{LEFT}}$.

Answer:

$$r \bowtie_{\text{LEFT}} t = (r \bowtie t) \cup ((r - \pi_{A,B}(r \bowtie t)) \times \{(\text{NULL})\})$$

(1.c) [10 points] Compute the result of the following SQL query:

```
SELECT R.A, COUNT(T.C) AS C_COUNT
FROM R, T
WHERE R.A = T.B
GROUP BY R.A;
```

Answer:

A	C_COUNT
a	4
b	2

(1.d) [10 points] Compute the result of the following SQL query:

```
SELECT T.B, COUNT(T.C) AS C_COUNT
FROM T
GROUP BY T.B
HAVING COUNT(T.C) > 1
```

Answer:

S.B	C_COUNT
b	2
a	2

Question 2 [Relational Algebra – 60 points]. Consider the following database schema computer products:

`Computer(maker,model,category)`

`Model(num,speed,ram,hd,price)`

`Maker(name,address,phone)`

Where

- `maker` indicates the manufacturer of the computer
- `category` takes values such as "`desktop`", "`laptop`", "`server`";
- Following inclusion dependencies hold
  - $\texttt{maker} \subseteq \texttt{name}$
  - $\texttt{model} \subseteq \texttt{num}$

Express following queries in relational algebra:

(2.a) [10 points] "Find *all* the makers who make *some* laptop(s)"

Answer:

$$\pi_{maker}\,(\sigma_{category=\text{“laptop”}}\,(\text{Computer}))$$

(2.b) [15 points] "Find *all* the makers who make *at lease three* different desktop models"

Answer:

$$\begin{aligned}\pi_{maker}(\sigma_{model1 \neq model2\ \wedge\ model2 \neq model3\ \wedge\ model3 \neq model1}\\ (\rho_{\,model1 \leftarrow model}\,(\sigma_{category=\text{“desktop”}}(\text{Computer}))\\ \bowtie \rho_{\,model2 \leftarrow model}\,(\sigma_{category=\text{“desktop”}}(\text{Computer}))\\ \bowtie \rho_{\,model3 \leftarrow model}\,(\sigma_{category=\text{“desktop”}}(\text{Computer}))))\end{aligned}$$

(2.c) [10 points] "Find the phone numbers of *all* the makers who make desktops with speed = 3.2"

Answer:

$\pi_{maker}\,\sigma_{model\,=num}\ (\sigma_{category\,=\,\text{"desktop"}}\ (\text{Computer})\ \times\ \sigma_{speed\,=\,3.2}\ (\text{Model}))$

(2.d) [10 points] "Find the makers who don't make any desktop, and do make some laptop(s)"

Answer:

$(\text{Computer} - \sigma_{category=\text{"desktop"}}(\text{Computer})))\ \cap\ \pi_{maker}(\sigma_{category=\text{"laptop"}}(\text{Computer})$

(2.e) [15 points] "Find the makers who make all models with speed faster than 3.2"

Answer:

$\pi_{maker,\,model}\ (\text{Computer})\ /\ (\rho_{\ model\leftarrow num}\ \pi_{num}(\sigma_{speed>3.2}(\text{Model}))$

Question 3 [SQL – 40 points]. Given the schema in Question 3, express following queries in SQL:

(3.a) [15 points] "Find all laptop models and their makers"

Answer:

```
SELECT    maker, model
FROM      Computer
WHERE     category = "laptop"
```

(3.b) [20 points] "Find all makers who make the most expensive server"

Answer:

```
SELECT    C.maker
FROM      Computer C, Model M
WHERE     C.category = "server" AND
          C.model = M.num AND
          M.price = (SELECT MAX(price) FROM Model)
```

(3.c) [20 points] "Find all desktop models with the highest speed/price ratio, and return them along with their makers"

Answer:

```
SELECT    C.maker, C.model
FROM      Computer C, Model M
WHERE     C.model = M.num AND
          M.speed/M.price ≥ ALL (SELECT (speed/price) FROM Model )
```

Question 5 [True/False questions – 40 points]. For each of the following statements, indicate whether they are true of false. A correct answer is worth **5** points, no answer is worth **0** points, wrong answer is worth **-3** points. The minimum you can receive on this question is 0/40.

(4.a) __T__ In SQL DDL, there can be zero or one `primary key` declarations in each `create table` statement;

(4.b) __F__ Atomicity means that each database transaction executes as if there are no other transactions running;

(4.c) __F__ A relation with 5 attributes can have more than 100 keys and superkeys;

(4.d) __F__ In SQL, relations are sets of tuples and can't have any duplicates;

(4.e) __T__ In SQL, views can be updated just like database relations;

(4.f) __F__ $S \bowtie R = S \times R$, if S and R don't share any attributes, and $S \bowtie R = S \cup R$, if S and R have the same attributes.

(4.g) __F__ Consider relations R(A,B) and S(B,C). If B is a primary key for S, then $R \bowtie S$ ***may*** contain more tuples than R does.

$R \bowtie S$ has, at most, the cardinality of R, since each tuple in R shares its B value with at most one tuple in S (since B is a primary key for S).

(4.h) __F__ Projection operators commute. That is, $\pi_X(\pi_Y(R)) = \pi_Y(\pi_X(R))$, holds for every relation R and all sets of attributes X and Y