Written Homework 1 Solutions:

Answer 1:
1.1 The query would yield customer name, email and contact number from the Customers table for all the entries which contain the substring “New York” as a part of the value in cust_address column.

1.2 The query output will have the columns Name which would represent the restaurant_name, Cuisine which would represent the restaurant_cuisines, Address would represent the restaurant_address and Contact would represent the restaurant_contact, Rating which would represent the restaurant_rating. The output will only have entries with restaurant_rating >= 3.5 and contain the substring “Lebanese” in the restaurant_cuisines column. Finally, the entries will be ordered in descending order of restaurant rating. All the data is from Restaurants table.

1.3 No.

Correct Query:

```
SELECT cust_name, cust_email, cust_contact, SUM(order_amount) AS order_total
FROM Customers AS c
INNER JOIN
        Orders AS o
ON c.cust_id = o.cust_id
WHERE AND o.order_status = ‘Delivered’
          AND EXTRACT(YEAR FROM o.order_date) = 2021
GROUP_BY c.cust_id
HAVING order_total > 500
```

1.4 The query would yield data from the FoodItems table. The query would return the item_id, item_name, item_cuisine and item_cost for all the items which are served in the restaurant with id “89758”.

1.5 One way to accomplish this would be to add a column “item_id” to the orders table which would tell us the item for each order. However, this is not the correct solution since an order can contain multiple items.

We can create a relationship between Orders and FoodItems. The relationship table would look like:
Order_FoodItems (id, order_id, item_id).
The first column “id” would be the primary key and auto-increment as well. “Order_id” will be a foreign key referencing Orders table and “item_id” will be a foreign key referencing FoodItems table. To clarify further, for order_id = ‘4559’, the table might look like:

id	order_id	item_id
1	4559	23
2	4559	13
3	4000	34

The SQL query to get FoodItems ordered as a part of order_id = ‘4559’ would be like:

```
SELECT * FROM FoodItems as f
INNER JOIN
        Order_FoodItems as of
ON
        f.item_id = o.item_id
WHERE f.order_id = ‘4559’
```

Answer 2:
2a. Inner Join

course_id	course_name	instructor_id	instructor_name
C001	Programming Languages	I001	Dan Fisher
C002	Software Engineering	I003	Steve Simpson
C003	Operating Systems	I002	Zoe Underwood
C004	Data Analytics	I003	Steve Simpson
C005	Business Analytics	I001	Dan Fisher

2b. Left Join

course_id	course_name	instructor_id	instructor_name
C001	Programming Languages	I001	Dan Fisher
C002	Software Engineering	I003	Steve Simpson

C003	Operating Systems	I002	Zoe Underwood
C004	Data Analytics	I003	Steve Simpson
C005	Business Analytics	I001	Dan Fisher

2c. Right Join

course_id	course_name	instructor_id	instructor_name
C001	Programming Languages	I001	Dan Fisher
C002	Software Engineering	I003	Steve Simpson
C003	Operating Systems	I002	Zoe Underwood
C004	Data Analytics	I003	Steve Simpson
C005	Business Analytics	I001	Dan Fisher
		I004	Natalie Lawrence

2d. Full Join

course_id	course_name	instructor_id	instructor_name
C001	Programming Languages	I001	Dan Fisher
C002	Software Engineering	I003	Steve Simpson
C003	Operating Systems	I002	Zoe Underwood
C004	Data Analytics	I003	Steve Simpson
C005	Business Analytics	I001	Dan Fisher
		I004	Natalie Lawrence

Answer 3: Query 1 will give the correct output

Answer 4:

a. max: n + m, min: n
b. max: m, min: 0
c. max: n * m, min: 0
d. max: m^2, min: m, we also accept min:0, max m
e. max: n, min: n-m
f. max: m, min: 0

Answer 5:
5.1 The user of SQL has no idea how the data is physically represented in the machine. He or she relies entirely on the relation abstraction for querying. Physical data independence is therefore assured. Since a user can define views, logical data independence can also be achieved by using view definitions to hide changes in the conceptual schema.

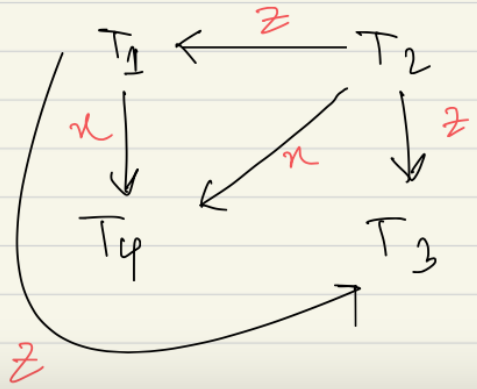
5.2 B, D
5.3 A, C

Answer 6:

6.1

(i) $S = r_1(x)\ w_2(z)\ r_2(x)\ w_1(z)\ r_3(z)\ w_4(x)$

T_1	$r_1(x)$			$w_1(z)$		
T_2		$w_2(z)$	$r_2(x)$			
T_3					$r_3(z)$	
T_4						$w_4(x)$



NOTE:

TYPES OF CONFLICTS:

$r_x(i) - w_y(i)$

$w_x(i) - r_y(i)$

$w_x(i) - w_y(i)$

6.2 The graph is serializable because it contains no cycle (DAG).

6.3

T_2,T_1,T_3,T_4

T_2,T_1,T_4,T_3

6.4
Concurrency can be thought of as running a schedule consisting of a set of transactions in some order maintaining the data Isolation and Consistency properties. The best criterion for concurrent transactions is to have a serializable schedule. This is because running a set of transactions concurrently is analogous to running those transactions in a serialized order (because of isolation and consistency).

Strict 2-PL guarantees conflict serializability. However, for this schedule, the conflict graph results in a cycle, therefore, is not serializable. Thus, the given schedule cannot run concurrently under strict 2-PL.

Answer 7:
7.1 A, B
7.2 A, B
7.3 B
7.4 A, B, C
7.5 D

Answer 8:
8.1

T1: Slock A; R(A); Slock B; R(B); B=A+B; Xlock B; W(B); unlock A; unlock B
T2: Slock B; R(B); Slock A; R(A); A=A+B; Xlock A; W(A); unlock B; unlock A

8.2

T1	T2
S(A)	
R(A)	
S(B)	
R(B)	
B = A + B	
	S(B)
	R(B)
	S(A)
	R(A)

	A = A + B
X(B)	
wait	X(A)
wait	wait

Answer 9:

Part A: 30%.
Explanation: The CDF graph shows that 30% of requests have a latency of 1-2 microseconds or less. Such a latency is only possible when the requests are served from DRAM.

Part B: 50% DRAM, 30% flash.
Explanation: The CDF shows that 50% of requests have a latency of 1-2 microseconds, therefore they are served from DRAM. The next 30% exhibit latencies up to 200 microseconds, which would be characteristic of flash latencies. The rest are served over the Internet, which incurs tens-hundreds of milliseconds of latency.

Part C: (c) Use more flash on the phone.
Explanation: the average latency right now is roughly: 0.5 * 1 microsecond + 0.3 * 100 microseconds + 0.2 * 100 milliseconds ~= 20 milliseconds. This latency is completely dominated by the network latency.
(a) only shifts the latency between DRAM and flash. Since the network completely dominates the latency this has a very negligible impact on the total average latency.
(b) would be more impactful, because it reduces the percentage of requests going over the network, but it reduces the average latency due to the network by a factor of 25%.
(c) has an even greater impact, because it halves the percentage of requests going over the network, which is almost like halving the entire latency.
(d) similar to b, reduces the latency due to the network by a factor of 25%, but that is less than the effect of c, which almost halves the end-to-end latency.

Answer 10:
10.1: Throughput of Systems A, B and C are 1.333, 1.143, 0.888 jobs/sec respectively
10.2: System A performs the best by a factor of 1.166 and 1.5 compared to Systems B and C respectively.
10.3: Speedup = 45/25 = 1.8 times
10.4: The average latency of the jobs in each of the systems after fixing the performance issue in System C is 0.75 sec, 0.875 sec and 0.625 sec in systems A, B and C respectively.

Answer 11:

Step	Memory Data	Memory Log	Disk Data	Disk Log
1	A=0		A=0, B=1	
2	A=0, B=1		A=0, B=1	
3 (Log not flushed to disk yet)	A=0, B=2	Update B oldval=1 newval=2	A=0, B=1	
4 (Log not flushed to disk yet)	A=1, B=2	Update B oldval=1 newval=2 Update A oldval=0 newval=1	A=0, B=1	
5 (Log Flushed to Disk)	A=1 B=2		A=0, B=1	Update B oldval=1 newval=2 Update A oldval=0 newval=1 commit
6 (Data Flushed to Disk)	A=1 B=2		A=1 B=2	Update B oldval=1 newval=2 Update A oldval=0 newval=1 commit

Client: Entity which wants to run a transaction on the database server.
Database server: Entity which runs the Database management software to handle the database

Q2. Durability.
If there is a crash after the commit has been issued and before the log records have been written to the disk, the client considers the transaction to have been successful although when the database server crashes and restarts it will not be able to rebuild the database correctly due to the missing WAL logs. The database server hence cannot identify that this transaction ever occurred.
This is the reason we do not reply commit to the client until we have synced the in-memory write-ahead-log to disk.

Q3. Consistency
If there is a crash after writing data to the disk and before the log can be written to the disk then in the event of a database server crash, the transaction fails midway and is not complete but the data on the disk already updated with no WAL record to roll it back. The client thinks that the transaction has failed but we have already altered data on the disk. This is not valid state of the database and hence consistency is affected.
Answer 12 (Aashish + Asaf):
Part A: Yes, we can simply run 2PL, where any read/write always tries to lock the entire table. This would produce serial (and serializable) schedules, because transactions would have to be executed one after the other serially. However, this would also be very inefficient, because there would be no concurrency.

Part B: Yes, once again we can simply run 2PL, but now the reads also acquire an exclusive lock rather than a shared lock. This would ensure serializability (similar to 2PL), since all the schedules produced by this algorithm are a subset of the regular 2PL with exclusive and shared locks, and all schedules produced by 2PL are serializable. However, it would enable less concurrency, because if any transactions read the same value, they would not be able to do so concurrently, so the only transactions that can execute concurrently are those that operate on non-overlapping values.

Part C: No it does not, because it can produce schedules with anomalies.
Very simple example (there are many others):
T1: W1(A), R1(A)
T2: W2(A), W2(B)

The following schedule is possible with this algorithm:
W1(A), W2(A), W2(B), R1(B)

But this schedule has a circular conflict graph (T1-->T2-->T1).

In this schedule, W1(A) is able to obtain the lock (no other lock is holding A). Then when W2(A) arrives, it is able to also obtain the lock on A, because it has a higher ID. It writes the value to A,

and is also able to execute W2(B) (no lock on B), and finally unlocks because the transaction is done. Then R1(A) arrives and again can obtain the lock (because no other transaction is holding it).

Answer 13:
Only S1 is serializable

Answer 14:
14.1:

```
SELECT employee_id
    , employee_name
    , department_name
    , (salary - avg(salary) OVER (PARTITION BY department_name))/(avg(salary) OVER
(PARTITION BY department_name))*100 AS rel_diff
FROM employees;
```

14.2:
`ROW_NUMBER` - assigns a number to each row in order, regardless of any ties
`RANK` - assigns same rank to ties, and skips ahead to next number in position, eg. 1, 2, 2, 4 (3 is skipped because of the tie)
`DENSE_RANK` - assigns same rank to ties, but continues with the sequence, eg. 1, 2, 2, 3

14.3:

```
SELECT MAX(salary) OVER (PARTITION BY department_name ORDER BY joining_date
ROWS BETWEEN UNBOUNDED PRECEDING AND 1 PRECEDING)
```