2023 Timed Practice Paper 1

Q1 : Step By Step

Conditions	1	2	3	4	5	6	7	8
No Accident	T	T	T	T	F	F	F	F
NCD Protector	T	T	F	F	T	T	F	F
Luxury Car	T	F	T	F	T	F	T	F
NCD Discount	x	x	x	x	x	x		
Free Road-side Assistance	x				x			

Merge the conditions that have the same outcomes

Conditions	1+5	7+8	2	3	4	6
No Accident	-	F	T	T	T	F
NCD Protector	T	F	T	F	F	T
Luxury Car	T	-	F	T	F	F
NCD Discount	x		x	x	x	x
Free Road-side Assistance	x					

2+4 vs 2+6

2+4

Conditions	1+5	7+8	2+4	3	6
No Accident	-	F	T	T	F
NCD Protector	T	F	-	F	T
Luxury Car	T	-	F	T	F
NCD Discount	x		x	x	x
Free Road-side Assistance	x				

2+6

Conditions	1+5	7+8	2+6	3+4
No Accident	T	F	-	T
NCD Protector	T	F	T	F
Luxury Car	T	T	F	-
NCD Discount	x		x	x
Free Road-side Assistance	x			

1 (a) Design a decision table to take into account of all the possibilities.

No Accident	T	T	T	T	F	F	F	F
NCD Protector	T	T	F	F	T	T	F	F
Luxury Car	T	F	T	F	T	F	T	F
NCD Discount	x	x	x	x	x	x		
Free Road-side Assistance	x				x			

4

(b) Simplify the decision table in (a)

No Accident	-	-	T	F
NCD Protector	T	T	F	F
Luxury Car	T	F	-	-
NCD Discount	x	x	x	
Free Road-side Assistance	x			

2

(c) Using a variable to represent each condition and an OUTPUT statement to represent the action to be taken. Translate the decision table into pseudocode

```
if ncd_prot or no_accident:
    print("NCD Discount")
if ncd_prot and luxury:
    print("Road Side Assistance")
if not no_accident and not ncd_prot:
    print("No Benefits)
```

2

2 Post order list: `[3, 2, 6, 10, 9, 5, 17, 14, 13, 11]`

In order list: `[2, 3, 5, 6, 9, 10, 11, 13, 14, 17]`

(a) Algorithm

FUNCTION `pre_order(in_order, post_order)`

```
- pop the last item in post_order list, that is the root
node
- find the index of the root node in the in_order list
- left of index is the left subtree, right of index is
the right subtree
- build the in_order and post_order lists for left and
right subtree
- left = pre_order( in_order_for_left_subtree,
post_order_for_left_subtree)
```

4

```
- right = pre_order( in_order_for_right_subtree,
post_order_for_right_subtree)
   - Return root + left + right // concatenate operation
ENDDUNCTION
```

(b)

```
          11
      5       13
   2    9       14
    3 6 10       17
```

2

(c) **i** Any
- Find the height of the current node
- Find the number of edges from the current node to the furthers leaf node
- Find the longest number of traversals from the current node to a leaf node

1

ii - 2,3 1

iii - 6, 11 1

iv - O(N) 1

v

```
FUNCTION insert(current, new_data) RETURNS BOOLEAN
 IF new_data < Data[current] THEN                    a
  IF Left[current] = -1 THEN                         b
   Free ← GetNextFree()                              c
   Data[Free] ← new_data
   Left[Free] ← -1
   Right[Free] ← -1
   Left[current] ← Free
  ELSE                                               b
   Insert(Left[current])                             d
  ENDIF
 ELSE                                                a
  IF Right[current] = -1 THEN                        b
   Free ← GetNextFree()                              c
   Data[Free] ← new_data
   Left[Free] ← -1
   Right[Free] ← -1
   Right[current] ← Free
  ELSE                                               b
   Insert(Right[current])                            d
  ENDIF
 ENDIF
ENDFUNCTION
// GetNextFree() will return the next free index in
the array
```

a- [1] for if < data, [1] for else
b- [1] for Left[] = -1 [1] for else
c- [1] getNextFreeIndex(), [1] init data,left, right, [1] Left[cur] = free
d- [1] recursive call

8

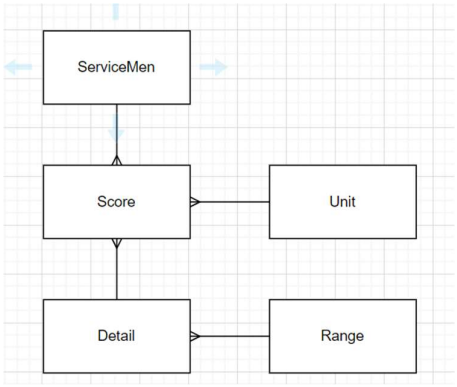
Base case a AND b for both [4]

Insert [3]

Recusive case d [1]

Q3

(a)



[6]

- **ServiceMen, Range , Unit[1]**
- **Score with relationship with ServiceMen [2] , Detail with relationships with Score and Range [3]**

(b)

ServiceMen (NRIC, Name, Address, Contact, BloodGroup, NOK)

Range (RangeID, RangeName, Address, Contact)

Unit(Unit, CO)

Detail (DetailID, RangeID*, DetailNumber, Date, StartTime)

Score (NRIC*, DetailID*, ShooterNumber, Score, Unit*)

OR

Detail (RangeID*, DetailNumber, Date, StartTime)

Score (NRIC*, RangeID*, DetailNumber, Date ,ShooterNumber, Score, Unit*)

[4]

- **ServiceMen, Unit, Range with correct PK [2]**
- **Score with PK with FK[1]**
- **Detail with PK with FK[1]**

(c)

Answers must be based on these concepts:

3NF ->

- **A non-key attribute cannot depend on another non-key attribute**
- **2NF**

2NF ->

- **A non-key attribute must be dependent wholly on the key attributes(composite key)**
- **1NF**

1NF ->

- **All attributes must be atomic with no repeating group of attributes**
- **Isomorphic (same structure)**
  - **All the data must be able to be organised into**
    - **rows with the same number of columns,**
    - **all the values in each column must be atomic and of the same data type.**
    - **Each row must be unique**

Any 2 [2]

- **(Not 1NF) NOT isomorphic ,The range data (name,address,contact,..) are not in a row together with the rest of the data.**
- **(Not 2NF) Address, Contact depends only on Range Name.**
- **(Not 3NF) Score depends only on the shooter in the detail**

(d) [2]

- **Reduce Redundant data**
- **Reduce Insert/Update/Delete anomalies**

(e)

(i) [2]

```
SELECT Score.Score FROM
Score
WHERE Score.NRIC = "S7652344Z"
```

- 1m: Correct column selected
- 1m: Correct WHERE

(ii) [2]

```
SELECT Score.NRIC FROM
Score
GROUP BY Score.NRIC
HAVING SUM(Score) > 31
```

- 1m: Correct GROUP BY
- 1m: Correct HAVING

(f)

[2]

- **Can store new data with new data types without changing the schema of the database, example video and other sensor data**
- **Can re-purpose data for new use cases, example instead of filtering servicemen for combat and leader roles, a new use case could be the quality and robustness of the rifles used in the shooting sessions.**
- **Scalable to include new ranges**
- **Archive old data using hierarchical storage**

(g)

- Private vs Public
- Need Authentication and Authorisation for Private

(h)

- NRIC , Address , contact details are being used in a different scenario , example used by Insurance companies to solicit business or employment agencies to hire part time staffs
- Data are not deleted when the NS men are no longer liable for NS.
- Data are not back-up and are lost when there is a hardware failure on the disk.