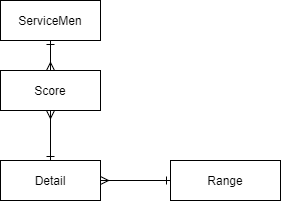
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | 1. 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 |  |  |  |  1. 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 |  |  |  | | | | 4  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") | | 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  Algorithm  pre\_order(in\_fix, post\_fix)  - pop the last item in post\_fix list, that is the root node  - find the index of the root node in the in\_fix list  - left of index is the left subtree, right of index is the right subtree  - Repeat on the right subtree until all the nodes have been identified  - Repeat on the left subtree until all the nodes have been identified | | 4 |
|  |  | See jupyter notebook for Python code | |  |
|  |  |  | [ 2,3,5,6,9,10] 11 [ **13**,14,17 ]   * Right-subtree * [ ] 13 [**14**,17]   + Right-sub tree   + 14 [17]     - Right-subtree     - 17 * Left-subtree * [ 2,3 ] 5 [ 6,**9**,10]   + Right-sub tree   + [6] 9 [10]     - Right-subtree     - 10     - Left-subtree     - 6   + Left-subtree   + **2** [3]     - Right-subtree     - 3 |  |
|  | **(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 to 16 | 1 |
|  |  | **iv** | * O(N) | 1 |
|  |  |  | |  |
|  |  | **v** | FUNCTION insert(current, new\_data) RETURNS BOOLEAN  IF new\_data < Data[current] THEN  IF Left[current] = -1 THEN  Free 🡨 GetNextFree()  Data[Free] 🡨 new\_data  Left[Free] 🡨 -1  Right[Free] 🡨 -1  Left[current] 🡨 Free  ELSE  Insert(Left[current])  ENDIF  ELSE  IF Right[current] = -1 THEN  Free 🡨 GetNextFree()  Data[Free] 🡨 new\_data  Left[Free] 🡨 -1  Right[Free] 🡨 -1  Right[current] 🡨 Free  ELSE  Insert(Right[current])  ENDIF  ENDIF  ENDFUNCTION  // GetNextFree() will return the next free index in the array | a  b  c  b  d  a  b  c  b  d |
|  |  |  | 1. [1] for if < data, [1] for else 2. [1] for Left[] = -1 [1] for else 3. [1] getNextFreeIndex(), [1] init data,left, right, [1] Left[cur] = free 4. [1] recursive call   Base case a AND b for both [4]  Insert [3]  Recusive case d [1] | 8 |
|  |  |  | |  |

**Q3**

**(a)**

****

Unit

**[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]**

**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), PK identifies unique rows**

**Any 2 [2]**

* **No valid PK, Not in Isormophic (similar) form (not 1NF)**
* **Address, Contact depends only on Range Name**
* **Score depends only on the shooter in the detail (Not 3NF)**

**(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.