Part I: Multiple Choice Questions (4p each).

1. **Normalization process …**
   [d) All of above]
2. **In a first normal form database table…**
   [a) The rows are atomic] L-Note page 6
3. Insertion anomalie exists when…
   [d) None of the above] c!
4. **No non-key attribute is dependent on another non-key attribute in…**
   [c) 3NF]
5. **When there are anomalies in the relation…**
   [d) The relation is decomposed]
6. **Update anomalie exists when…**
   [a) The process has to visit multiple rows]
7. **The attribute are functionally dependent to one another when…**
   [a) They are in the same database table] a) 65% c) 35%
8. **Null values are not desirable in the relations because…**
   [a) Both c and d]
9. The "car" relation contains…
   [b) Insertion anomalie only] c!
10. **The "car" relation has a…**
    [a) A single column primary key]
11. The "car" relation has…
    [d) Three candidate Keys]
12. **The "car" relation's primary key is**
    [a) Customer-Code + Make]
13. The "car" relation has no anomalies when it is normalized to…
    [b) 3NF] a!
14. When "car" relation is normalized to 2NF ___ new relations are produced.
    [a) 3] b!
15. When "car" relation is decomposed one of the new relations have the columns...
    [c) Make, Price, Customer-Code] b!
16. **In the "car" relation which of the functional dependency is true?**
    [b) Customer-Code ← Customer-Name]
17. **Which of the following queries finds the total number of "Mercedes" cars sold?**
    [b) Select COUNT (Prod-Year) from cars where Make = "Mercedes"]
18. **Which of the following queries gives the total amount of money made from car sales on 1/4/2008?**
    [d) Select SUM (Price-(Price*Discount/100)) from cars where Date = #1/4/2008#]
19. **Which of the following queries finds the most expensive car price in "cars"?**
    [b) Select MAX (Price) from cars]
20. **Which of the following queries finds the total number of car Mr. Mark Soulsberg bought?**
    [c) Select COUNT(Price) from cars where Customer-Name='Mark Soulsberg']

Part II: T/F questions (2p each).

1. **Structured Query Language (SQL) is a Procedural Language.**
   [F] page 1 'SQL'
2. **Do while loops are valid procedures in SQL.**
   [F] L-Note page 1
3. **The "car" relations is in 1NF.**
   [T] 1NF
4. In a composite primary key, each key attribute is called 'partial key'.
   [T] L-Note page 8
5. **Primary Key columns can contain repeating data.**
   [F] must not contain duplicate data
6. When a relation is decomposed, each partial key is placed on each of the new relation created.
   [T]
7. **In 2NF all attributes are dependent on the primary key.**
   [T] L-Note page 6-7
8. **The aggregate function "SUM" finds the total number of rows selected by the query.**
   [F] COUNT ←→ SUM or number ←→ sum
9. To build relationship between the newly created relations after decomposition process, partial key columns are used.
   [T]
10. **Transitive dependencies are not allowed in 3NF.**
    [T] L-Note page 6.

Table:

The following is a section of a relation that keeps the necessary information of second hand car sale transactions.

Table name: cars.

Prod-Year	Make	Price	Date	Customer-Code	Customer-Name	Discount %
2006	Ford	24000	17.04.2008	008	John White	4
2007	Mercedes	54000	19.04.2008	019	Mary Finch	6
….	….	….				
….	….	….				
2008	Toyota	260000	28.04.2008	002	Ali Veli	2
2007	Mercedes	570000	09.06.2008	057	Fatma Duran	7
2008	Ford	290000	11.06.2008	019	Mary Finch	3

ASSUME NO CUSTOMER BUYS THE SAME MAKE OF CAR AGAIN.

Customer-Code *→ Customer-Name
Prod-Year *→ Prod-Year
Make, Date → Customer-Code
Customer-Code, Make → Price, Discount %, Prod-Year, Date
Customer-Code, Date, Prod-Year → Make, Price, Discount %
Make, Price, Date → Customer-Code, Prod-Year, Discount %
Customer-Code, Make, Prod-Year → Price, Discount %, Date # fine its extra, customer and make fine
Customer-Code, Date → Make, Price, Discount %, Prod-Year #maybe make 2 transaction in same day
Price, Date → Discount %, Make, Discount %, Prod-Year #maybe same price date, make different
Customer-Code, Prod-Year → Make, Date #maybe customer buy same prod-year
Make, Price → Discount %, Date #maybe make sell same price in future

[Candidate key]

- **Make, Date**
- **Customer-Code, Make**
- **Customer-Code, Date, Prod-Year**

[One of Candidate Key is Primary Key]

- **Customer-Code + Make** #75%
- Customer-Code + Date + Prod-Year #60%

{Primary Key}

- **Customer-Code**

[Note]

It looks like every time we add new customer buy car from Make. Not delete no update.