Name: Ayush Jain
SAP ID: 60004200132
Div: B1
Computer Engineering

Solutions:

→ Q. 3) 1) Integrity constraints are a set of rules. It is used to maintain the quality of information.

2) It ensure that the data insertion, updating and other processes have to be performed in such a way that data integrity is not affected.

3) Types of Integrity constraints:

a) Domain constraints:

(i) It can be defined as the defination of a valid set of values for an attribute.

(2) The data-type of domain includes string, character, integer, time, date, currency, etc. The value of attribute must be available in the corresponding domain.

(3) Example:

ID	NAME	AGE
100	Jack	18
101	Herry	19
102	Roman	A

→ Not allowed. Because age is an integer attribute.

b) Entity integrity constraints:

(i) It states that primary key value can't be NULL.

(2) This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, than we can't identify those rows.

(3) A table can contain a null value other than the primary key field.

(4) Example :

EMPLOYEE

Emp-ID	Emp_Name	Salary
100	Jack	10000
101	Harry	50000
	John	15000

↓ Not allowed. Because primary key can't contain NULL value.

c) Referential Integrity Constraints :

(1) It is specified between two tables.

(2) In this constraints, if a foreign key in Table 1 refers to the Primary key of table 2, then every value of the foreign key in Table 1 must be NULL or be available in Table 2.

(3) Example :

(Table 1)

Emp-id	Name	Age	D-NO → Foreign Key
1	Jack	18	11
2	Harry	40	24
3	John	31	18 → Not allowed as D-No 18 is not defined
4	Devil	27	13 as a primary key of table 2 and in table 1, D-No is foreign key defined.

Relationships ↕

Primary key →

D-NO	D-Location
11	Mumbai
24	Delhi
13	Goa

(Table 2)

d) Key constraints :

(1) Keys are the entity set that is used to identify an entity within its entity set uniquely.

(2) An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique and null value in the relational table.

(3) Example :

ID	Name	Age
100	Tom	20
101	Jerry	23
102	Jack	21
101	John	27

↓

Not allowed. Because all rows must be unique.

→ Q. 4) 1) Database Normalization is a technique of organizing the data in the database.

2) It can be considered as a process of analysing the given relation schemas based on their Functional Dependencies and primary keys to achieve the following properties:

(1) Minimizing redundancy.

(2) minimizing the insertion, deletion and update anomalies.

(3) Ensuring data is stored in correct table.

3) It is a multi-step process that puts data into tabular-form by removing duplicated data from the relational tables.

4) The various forms of Normalization are described below:

a) First Normal Form (1NF):

(1) It states that domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute.

(2) Example:

Un-Normalized Table:

Student	Age	Subject
Jack	15	Java, C
Kane	16	HTML, PHP

Normalized Table: Any row must not have a column in which more than ~~one~~ one value is saved.

Student	Age	Subject
Jack	15	Java
Jack	15	C
Kane	16	HTML
Kane	16	PHP

b) Second Normal Form (2NF):

(1) A relation is said to be in 2NF, if it is already in 1NF and each and every attribute fully depends on the primary key of the relation.

(2) Example:

Student - Project Table

Stud-id	Proj-id	Stud-Name	Proj-Name
100	001	Rooney	Cloud
200	002	Kane	Servers

Stud-name depends on Stud-id and Proj-name depends on Proj-id. The above table can be normalized to 2NF as shown below:

Student Table in 2NF

Stud-ID	Stud-Name	Proj-ID
100	Rooney	001
200	Kane	002

Project Table in 2NF

Proj-ID	Proj-Name
001	Cloud
002	Servers

c) Third Normal Form (3NF):

(1) A relation is said to be in 3NF, if it is already in 2NF and there exists no transitive dependencies in that relation.

(2) Example:

Below Table not in 3NF :

Stud-ID	Stud_Name	City	ZIP
100	Rooney	Manchester	4001
200	Kane	Stoke	4002

Stud-ID is only primary key attribute. City can be identified by Stud-id as well as ZIP.

Stud-ID -> ZIP -> City, so there exists transitive dependency.

Hence 3NF table is below:

Student-Detail:

Stud-ID	Stud_Name	ZIP
100	Rooney	4001
200	Kane	4002

Zip_Code:

ZIP	City
4001	Manchester
4002	Stoke.

d) Boyce-Codd Normal form (BCNF):

(1) It is an extension of 3NF in strict way. A relationship is said to be in BCNF if it is already in 3NF and for any non-trivial functional dependency, X -> A, then X must be a super-key.

(2) Example:

In a 3NF example, Stud-ID is super-key in Student-detail relation and ZIP is a super key in zip codes relations.

So Stud-ID -> Stud_Name, zip and

Zip -> City

Confirms, that both relations are in BCNF.

→ Q. 5

a) i) Shadow Paging is a recovery technique that is used to recover database.

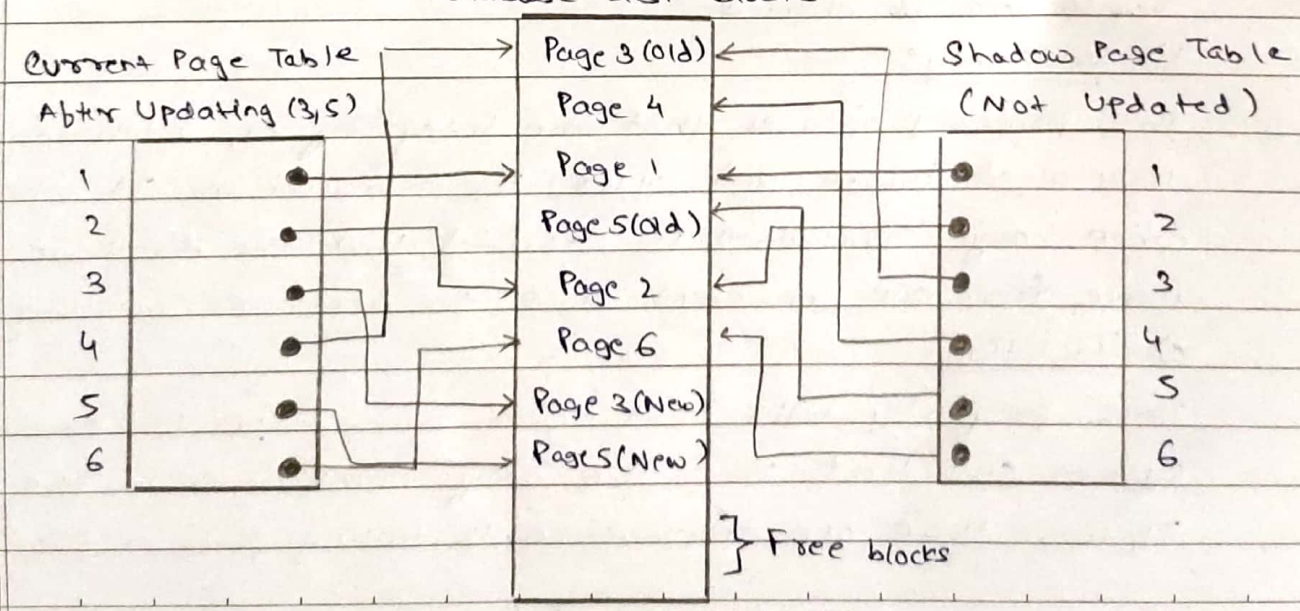
2) In this recovery technique, database is considered as made up of fixed size of logical units of storage which are reffered as pages. Pages are mapped into physical blocks of storage, with the help of ~~storage~~ page table which allow one entry for each logical page of database. This method uses two page tables named current page table and shadow page table.

3) The entries which are present in current page table are used to point to most recent databases pages on disk.

4) Shadow page table is used when the transcation starts ~~wit~~ which is copying current page table.

5) After this, shadow page table gets saved on disk and current page table is going to be used for transcation.

6) Entries present in current page table may be changed during execution but in shadow page table it never gets changed. After, transcation, both tables becomes identical.



Q. 5

→ b) 1) A transaction is a set of related changes which is used to achieved some of the ACID properties.

2) A transaction can be defined as a group of tasks. A single task is the minimum processing unit which cannot be divided further.

3) The database is inconsistent during the transaction. It goes into a consistent state only when the transaction has occurred successfully. It is very important to have successful transaction.

4) For maintaing the integrity of data in the database, certain properties are followed by all the transaction that take place in database.

5) The ACID Properties of a Transaction are:

a) Atomicity:

This property states that the transaction should either occur completely or doesn't occur at all. Each transaction is treated as unit and the execution is completed else the transaction is aborted.

b) Consistency:

This property ensures that the integrity of database is maintained before and after the transaction. It ensures that when any transaction is executed that the database should move from one consistent state to another consistent state.

c) Isolation:

This property tells that each transaction is executed in the system such that it is the only transaction in the system. If more than one transaction is taking place in parallel, then

the occurrence of one transcation will not affect the other transcation.

d) Durability:

This property ensures that once the changes are made in database these changes persist in the database even if any system failure occurs. These changes are saved permanently in non-volatile memory.

Q. 6

→ a) i) As the name shows, JOIN means to combine something. In case of SQL, JOIN means to combine two or more tables.

ii) Types of JOINS:

a) INNER JOIN:

In SQL, inner join selects record that have matching values in both table as long as the condition is satisfied. It returns the combination of all rows from both the tables where the condition satisfies.

Syntax:

```
Select table1.column1, table1.column2, table2.colum1 .....
From table1
INNER JOIN table2
ON table1.matching_column = table2.matching_column.
```

b) LEFT JOIN

The Left join returns all the values from left table and the matching values from right table.

Syntax:

```
Select table1.column1 , table2.column1 ....
From table1
Left Join table2
ON table1.matching_column = table2.matching_column.
```

c) Right Join

In SQL, Right join returns all the values from the rows of right table and the matched values from the left table.

Syntax:

```
Select table1.column1, table2.column1...
From table1
Right Join table2
ON table1.matching-column = table2.matching column.
```

d) Full Join

In SQL, full join is the result of a combination of both left and right outer join. Join tables have all the records from both the tables.

Syntax:

```
Select table1.column1, table2.column1...
From table1
Full join table2
ON table1.matching-col = table2.matching-column.
```

Q. 6)

→ b) i) Strong EntitySet:

(a) A strong entity is not dependent on any other entity in the schema. A strong entity will always have a primary key.

(b) It is represented by a single rectangle. The relationship of two strong entities is represented by a single diamond.

(c) Various strong entities, when combine together, create a strong entity set.

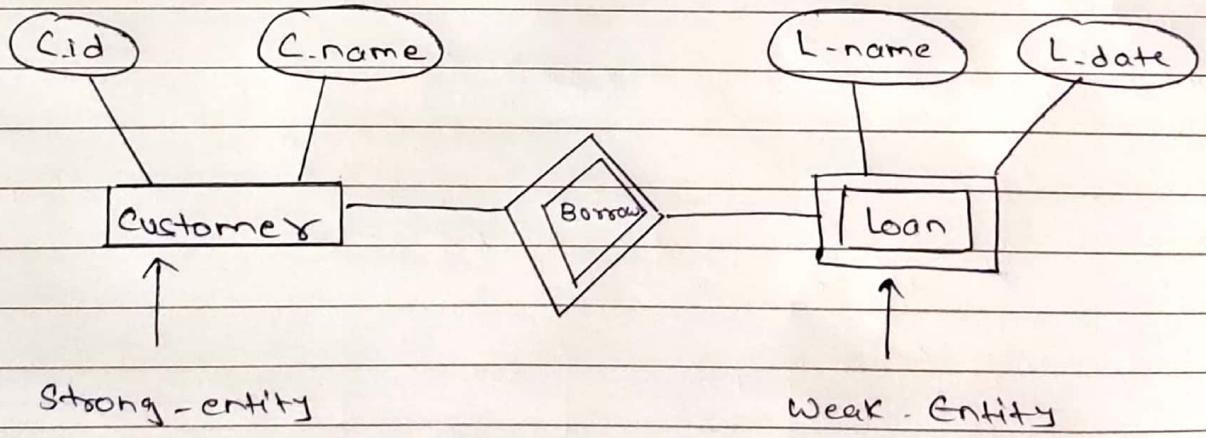
2) Weak EntitySet:

(a) A weak entity is dependent on a strong entity to ensure its existence.

(b) Unlike a strong entity, weak entity does not have primary key

(c) It is represented by a double rectangle. The relation between one strong and one weak entity is represented by a double diamond.

Example:



Q. 1

→ b) i) Fragmentation is a process of dividing the whole or full databases into various subtables or sub-relations so that data can be stored in different system.

2) The small pieces of sub-relations or subtables are called fragments.

3) Horizontal Fragmentation:

a) It refers to the process of dividing a table horizontally by assigning each row of relation to one or more fragments.

b) These fragments are then be assigned to different sides in the distributed system.

c) Example:

Consider an employee table:

E-No.	E-Name	Design	Salary	Dep
101	A	abc	3000	1
102	B	abc	4000	1
103	C	abc	500	2

This employee table can be devided into different fragments like:

$EMP\ 1 := \sigma_{DEP=1}\ EMPLOYEE$

$EMP\ 2 = \sigma_{DEP=2}\ EMPLOYEE$

These two fragments are: T1 fragment of Dep = 1

ENo	E_Name	Design	Salary	Dep
101	A	abc	3000	1
102	B	abc	4000	1

6000420013 2

Q 1

→ a)