Model Question Paper- II with effect from 2022

CBCS SCHEME

Fourth Semester B.E Degree Examination 2024-25

Database Management System (BCS403)

TIME: 03 Hours *Max.Marks:100*

1. *Note: Answer any **FIVE** full questions, choosing at least **ONE** question from each **MODULE***
2. *M: Marks, L: Bloom's level, C: Course outcomes.*

		Module - 1	M	L	C
Q.1	a	What is a Database? Explain the three schema architecture with neat diagram.	8	L2	CO1
	b	What are the advantages of using DBMS approach? Explain	8	L2	CO1
	c	Explain the following terms. 1. Data Dictionary 2. Weak Entity	4	L2	CO1
		OR			
Q.2	a	Explain the categories of Data Models.	8	L2	CO1
	b	Explain the component modules of DBMS & their interactions with diagram.	8	L2	CO1
	c	What are the responsibilities of DBA & database designers?	4	L2	CO1
		Module - 2			
Q.3	a	Explain the different types of update operations on relational database. How basic operation deals with constraint violation.	6	L2	CO2
	b	Explain Unary relational operations with examples.	6	L2	CO2
	c	What is an Integrity Constraint? Explain the importance of Referential Integrity Constraint.	8	L2	CO2
		OR			
Q.4	a	Explain the following relational algebra operation. JOIN, DIFFERENCE, SELECT, UNION	10	L3	CO2
	b	Discuss the E.R to Relational mapping algorithm with example for each step.	6	L3	CO2
	c	Explain the relational algebra operation for set theory with examples.	4	L2	CO2
		Module - 3			
Q.5	a	Illustrate insert, delete, update, alter & drop commands in SQL.	6	L4	CO3

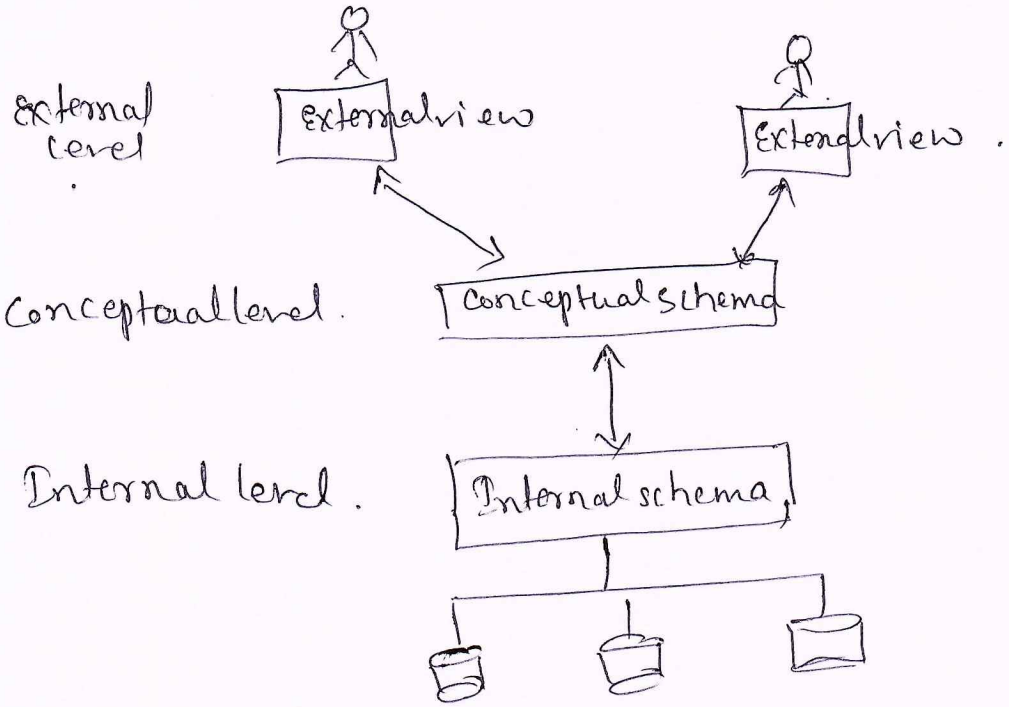
Model Question Paper- II with effect from 2022

	b	Explain informal design guidelines for relational schema design.	4	L2	CO3
	c	What is Functional dependency? Explain the inference rules for functional dependency with proof.	10	L3	CO4
		OR			
Q.6	a	Consider two sets of functional dependency. F={A→C, AC→D, E→AD, E→H} E= {A→CD, E→AH}. Are they Equivalent?	10	L3	CO4
	b	Explain the types of update anomalies in SQL with an example.	10	L2	CO3
		Module - 4			
Q.7	a	Demonstrate transaction states & additional operations.	10	L3	CO4
	b	Demonstrate working of Assertion & Triggers in database? Explain with an example.	10	L2	CO3
		OR			
Q.8	a	Demonstrate the System Log in database transaction.	6	L2	CO4
	b	Discuss the ACID properties of database transaction.	4	L2	CO4
	c	Explain stored procedure language in SQL with an example.	10	L2	CO3
		Module - 5			
Q.9	a	Explain the Two phase locking protocol used for concurrency control.	8	L3	CO5
	b	Define Schedule? Illustrate with an example.	4	L2	CO5
	c.	Why Concurrency control is needed? Demonstrate with an example.	8	L3	CO5
		OR			
Q.10	a	What is NOSQL? Explain the CAP theorem.	6	L2	CO5
	b	What are document based NOSQL systems? basic operations CRUD in MongoDB.	8	L2	CO5
	c	What is NOSQL Graph database? Explain Neo4j.	6	L2	CO5

Model Question Paper - II with effect from 2022-25
DBMS (BCS403)

Q1 a) Database is an organized collection of Related data stored electronically & managed using a database managment system.

Three schema Architechure



* External level (view level)
* Highest level of abstraction
* shows only required data to users.
* Different users get different views.

Example: student see marks & attendence.

Adavantage: Security, Data hiding, user friendly.

* Conceptual level (Logical level)
* Describe overall database structure
* Define entities, relationship, constraints.
* Independent of physical storage.

example: Define table student (ID, name, Dept).

③ Internal level (physical level)
* lowest level.
* Describe how data is stored physically
* Include file structure, indexing, storage blocks

1b: Advantage of using DBMS approach.

① Data Redundancy control: Elimates duplicate date
* save storage space

② Data Consistency: maintain same value across database

③ Data Sharing: ~~Maintains same~~ multiple same value across database

④ Improved Security: Authetication & authorization

⑤ Back up & Recovery: protect Data from loss

⑥ Data Independence: change in physical storage do not affect application

⑦ Bettel Data Accen: Query language like SQL provides easy retrieval.

1c) i) Data Dictonary: A data dictonary stores metadata about Database objects.
It contains: table name, attribute names, datatypes, constrain
Example: Information about student table column

Student	name	ID	fees,	percenta

ii) Fact entity: A weak entity depends on a strong entity for existence.

Characteristics: No primary key, uses foreign key from strong entity

Example: Employee → strong entity
Dependent → weak entity.

29 Data model is collection of concept used to describe the structure of database. It defines how data is stored, organised, related and manipulated in a database system.

Data models are mainly classified into 3 types.

① Conceptual Data Model (High level Data Model)

The conceptual Data model Describe the datasets at very high level It focuses on understanding the real world system & user requirement.

* Describes entities, attributes, Relationships.
* Easy to understand
* Independent of DBMS & storage details.

Example: college database student is entity. course is entity, enrollement shows relationship

② Logical Data Model (Representational Data Model)

It describe how data is logically structured in a database but does not describe how data is physically stored.

* Define tables, columns, relations
* Independent of physical storage.

types of logical Models

* Relational Model

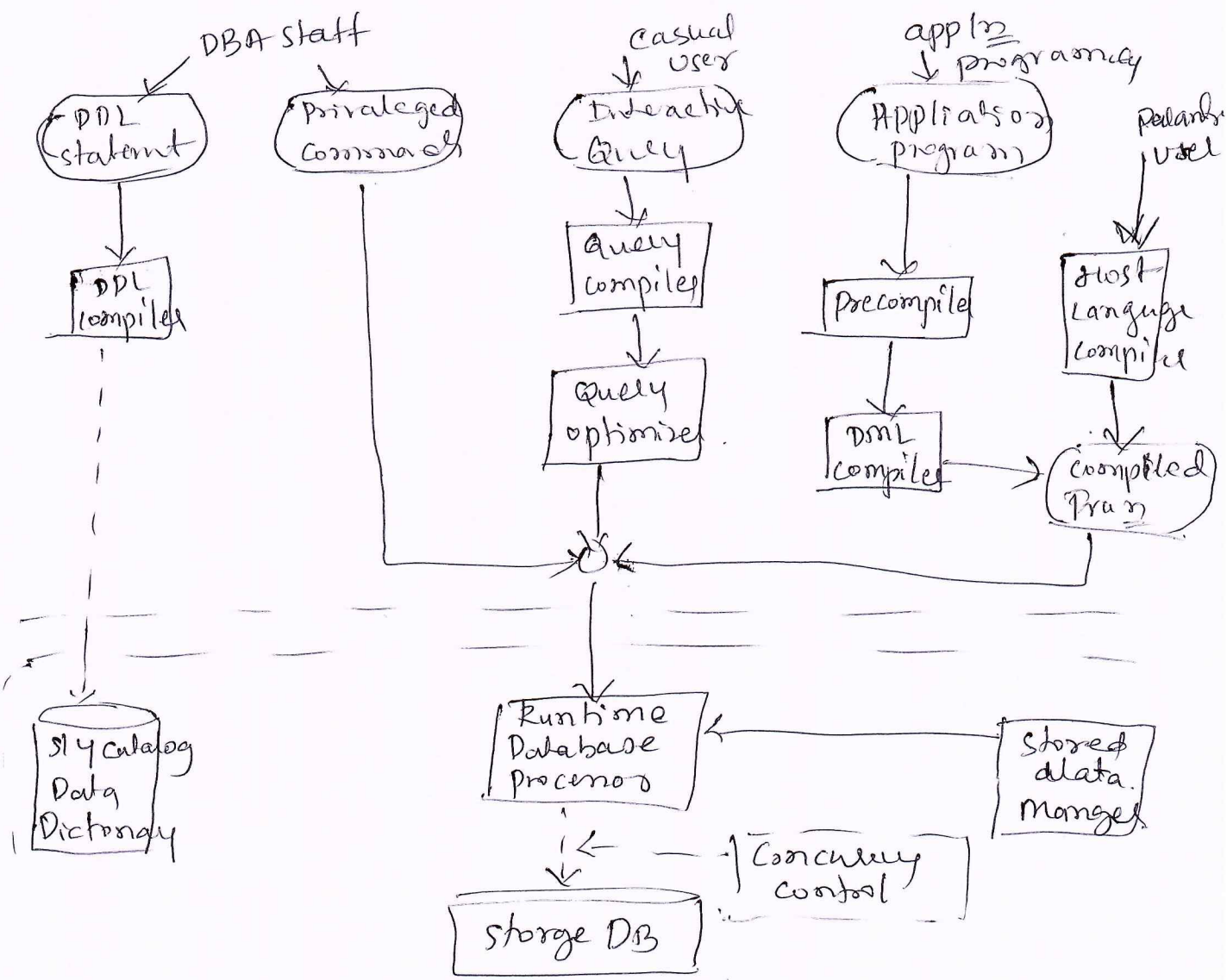
* Hierachical Model
* Network Model.

(3) Physical Data Model (Low level Data Model).

It describe how data is stored physically in the system
* Defines file organisation
* Describes indexing technique.

Example: How tables are stored in memory use of B-tree indexing.

2b



① Query manager: Translate SQL Queries & optimize Query execution

② Storage manager: Manage Data storage control indexing and files

③ Transaction Manager: Ensures ACID property and maintain consitency of data.

④ Authorization Manager: control access permissi

⑤ Recovery manager: restores database after failure

The DDL compiler process process schema definition, specified in the DDL, & stores descriptor of schema.

Query optimizer: is concerned with rearranging set of possible recording operations, eliminate redudancy.

2C) Responsibilities of DBA:

* Database Adminstrator (DBA) is responsible for managing, maining and controlling database system

* Database installation & configuration
* Security management
* Backup & recovery
* performance monitoring
* Storage management

Responsibilities of Database Designer:

Database Designer designs the structure of database according to user requirement.

* Requirement Analysis
* Designing database schema

* Defining constraints
* Normalization
* performance planning
* Documentation

39) Update operation in Relational Database are used to modify the contents of database. These are three main update operation

Insert Operation:

used to add new records into relation

student

Rollno	name	mark
101	Ravi	80

possible Constrain violation

* Domain constrain : value must be valid datatype
  Ex: mark can not be text
* key constraint : primary key must be unique
* Referential integrity : foreign key must match parent table

② Delete Operation : Removes records from table

example: Delete from student where Rollno = 101;

constrain voilation: Deleting a record referenced by another table causes referential integrity voilation.

Solution: Reject deletion, cascade delete, set foreign key to NULL

Update Operation: Modifies existing records.

Update student set marks = 90 where Rollno = 102;

Constrain violations:

* changing primary key creates duplicates
* Assigning invalid data type
* Breaking foreign key reference.

DBMS handles violation using:

* Rejecting operation
* Cascading update.

3b) Types of Unary operations:

① Selection operation (σ): which select rows that satify a condition

Syntax: σ condition (Relation)

example: select students with marks greater than 80.

$\sigma_{marks > 80}$ (Student)

② Projection operation (Π): which select specific column from table.

Syntax: Π Attribute (Relation)

Ex: Display only Names & marks.

$\Pi_{name, marks}$ (student.)

③ Rename Operation (ρ): which rename relation or attribute:

Ex: rename student as S.

ρ_S (student)

3c) Integrity Constraints: These are rules applied on database to maintain accuracy, consistency & validity of data.

Importance of Referential Integrity

Referential integrity helps maintain proper relationship between tables. It ensures that childs table records always refer to exiting parent table record

ex: If student table contain student details and marks table contain marks, & Roll number in marks must exist in student table

So maintain relationship between tables

② Prevent Invalid Data entry: It prevent insertion of foreign key value that do not exist in the referenced table.

ex: If roll no 105 is not present in student table, can not be inserted into Marks table

③ Prevent Orphans Records: Orphans records in child tables that do not have matching parent records, referential integrity avoides such record

④ Control Deletion & updating:
referential integrity restrict deletion or updating of parent records when related child records exist DBMS provides option like:

* cascade Delete
* Cascade Update
* Set NULL
* Restrict operation

4a) JOIN operation (⋈) : Join operation combines tup two relations based on a related attribute between them. It is used to retrive related data from multiple tables.

Types of Join:

* Natural Join
* Theta Join
* Equi Join
* Out Join

Ex: student table

SID	Name	DeptID
1	Ravi	D1
2	Meena	D2

Department table

DeptID	DeptName
D1	CSE
D2	ECE

~~Department table~~ Join operation : student ⋈ Department

SID	Name	DeptID	DeptName
1	Ravi	D1	CSE
2	Meena	D2	ECE

(2) Difference operation (−)

Definition : difference operation returns tuples that exist in one relation but not in another

Ex: Relation A

ID
1
2
3

Relation B

ID
2
3

A − B =

ID
1

It removes common records and shows remaining tuples.

③ Select operation (σ)

Select operation retrives rows from a relation that satisfy a specified condition. It is similar to WHERE clause in SQL

syntax: σ condition (Relation)

σ marks > 70 (Student)

Student

ID	marks
1	80
2	65
3	90

Result:

ID	marks
1	80
3	90

④ UNION operation (∪): It combines tuples from two relations and removes duplicates values.

Example: Relation A

ID
1
2

Relation B

ID
2
3

A ∪ B Result

ID
1
2
3

} Union merges data from two relations.

4b) ER model is converted into relational table using mapping rules

Step①: Mapping strong entity

Create a table for each strong entity and primary key of entity becomes primary key of table

Ex: Entity: Student (SID, name, Age)

table: Student (SID, name, Age)

step 2: Mapping weak entity.
Create table including.
* Attribute of weak entity
* primary key of strong entity as foreign key.

Ex: Dependent (Depname, Age) depends on Employee (empID)
table Dependent (EmpID, DepName, Age)

Step3: Mapping Binary 1:1 Relationship
* Add foreign key to either table

Ex: Employee — Manages — Department

Step 4: Mapping 1:M Relationship
~~Create separate relation including primary keys of both entities~~ Add primary key of one side as foreign key in many side.

Ex: ~~student — enroll — course~~

Step 5: Mapping M:N Relationship.
Create seperate relation including primary keys of both entities

Ex: student — enroll — course.

Step 6: Mapping multivalued Attribute.
* Create Separate table

Ex: student has phone Number
table: studentphone (SID, phoneno)

4c) Relational algebra is based on set theory operations

i) UNION (U)
combines tuples from two Relation

Ex: A ∪ B

ii) Intersection (∩): Return tuples common in both relation Ex: A = {1, 2} B = {2, 3}

A ∩ B = {2}

iii) Difference (−): Returns tupples present in one relation but not in another. $A = \{1, 2\}$ $B = \{2, 3\}$

Ex: $A - B = \{1\}$

iv) Cartesian Product (×): Combines each tuple of first relation with every tuple of second relation.

$A = \{1, 2\}$ $B = \{x, 4\}$

$A \times B = \{(1,x), (1,4), (2,x), (2,4)\}$

5a) ~~Illustrate~~

Insert Command: The INSERT command is used to add new records into a table.

Syntax: INSERT into table_name (column1, column2, column3) values (value1, value2, value3);

Ex: insert new record.

insert into student (SID, name, marks) values (101, 'Ravi', 85);

This add new record

Delete Command: Delete command removes records from a table based on a condition

Syntax: Delete from table_name
where condition;

Ex: Delete from student where SID = 101;

without 'where' Clause

Delete from student;

All record in table will be removed

~~UPDA~~ UPDATE Command: which modifies existing records in table

Syntax: UPDATE table_name
SET colum_name = value
WHERE condition;

Ex: UPDATE student SET marks = 100
WHERE SID = 22;

iv) ALTER command: It is used to modify the structure of a table such as adding, deleting or modifying columns.

Syntax: ALTER Table Student
ADD Age INT;
which add a new column.

ALTER table student Drop column age;
which change table structure without deleting table

v) Drop: It delete an entire table or database.
Drop table table-name;
Drop Table student;

5b) Informal design guidelines help in designing good relational database schemas.

① clear Semantics of Relation: Each relation should represent one entity or one relationship.

Ex:
student (SID, Name, coursename, Instructor)
This Bad design

student (SID, name)
course (course name, Instructor) } good design

② Reduce Redundant Information: Duplicate Data should be avoided because it wastes storage and causes inconsistency.

Ex: If department name is repeated for every student, updating department name become difficult.

③ Avoid Update Anomalies:

* Insert anomaly: cannot insert data without other information

Ex: cannot insert department unless student exists

* Deletion Anomaly: Deleting one record remove useful ~~student exists~~ information.

Ex: Deleting last student removes department details.

* Update nomality: updating repeated data becomes difficult.

④ Ensure Minimal NULL values: Relations should be designed so that NULL values are minimized

Ex: if ~~em~~ many employee are unmarried, SpouseName becomes NULL.

Better to store spouse details in separate table

⑤ Avoid spurious tuples: Incorrect data may appear when tables are joined improperly.

5c) Functional Dependency (FD): It is relationship between attributes in a relation.

Inference Rules for Functional Dependency. (Armstrong's Axioms).

i) Reflexivity Rule: If Y is subset of X then

$$X \rightarrow Y$$

Prof: If we know all value of X then we automatically know value of Y since Y is part of X

Ex: {SID, Name} → SID.

ii) Augmentation Rule: If $x \rightarrow y$, then;

$$xz \rightarrow yz$$

Prof: If X determines Y, adding Z to both side will

maintain dependency.

(SID, dept) → (name, dept)

iii) Transitivity Rule:

if $x \to y$

$y \to z$

Then

$x \to z$

prof: If x determine y & y determines z, then x indirectly determinez z.

Ex:

SID → DeptID

DeptID → Dept Name

∴ SID → Dept Name

Additional Derived Rules.

Union Rule: if $x \to y$

$x \to z$

then $x \to yz$

Decomposition Rule: if $x \to yz$

then $x \to y$

$x \to z$

Pseudotransitivity Rule: if $x \to y$

$wy \to z$

then $wx \to z$.

6a) Consider two set of functional dependency.

Given

$F = \{ A \to C, AC \to D, E \to AD, E \to H \}$

$E = \{ A \to CD, E \to AH \}$

step 1: Check if $F \subseteq E+$

from E ① $A \to CD$

$E \to AH$

} Derived dependencies of F from E

check $A \rightarrow C$

from E : $A \rightarrow CD$. using decomposition rule $A \rightarrow C$

check $AE \rightarrow D$.

from E : $A \rightarrow CD$ using decomposition Rule $A \rightarrow D$

if $A \rightarrow D$ is true

then $AC \rightarrow D$ is automatically true

$AC \rightarrow D$ satisfied.

Check $E \rightarrow AD$

from E : $E \rightarrow AH$

from earlier result : $A \rightarrow D$

using transitivity : $E \rightarrow A$.

$A \rightarrow D$ $\therefore$ $E \rightarrow D$.

Hence : $E \rightarrow AD$

check $E \rightarrow H$.

Already given in E : $E \rightarrow AH$.

using decomposition : $E \rightarrow H$.

$\therefore$ all dependency of F can be derived.

So $F \subseteq E+$

step 2: check if $E \subseteq F+$

check $A \rightarrow CD$.

from F : $A \rightarrow C$

$AC \rightarrow D$.

from $A \rightarrow C$, we get C.

Thus AC is know.

Hen $AC \rightarrow D$

Hence $A \rightarrow D$

$\therefore$ $A \rightarrow CD$ Satisfied

check E → AH.

from F : E → AD using decomposition : E → A

also : E → H is given directly

∴ E → AH ✓

final Conclusion

since ✓ F ⊆ E+ & E ⊆ F+.

∴ F & E are equivalent functional Dependency set.

6b) update anomalies occure due to poor ~~database~~ design and redundancy.

there 3 typer of update anomalies.

① Insertion Anomaly : which occure when new data cannot be inserted without adding other unnecessary data.

Ex:

SID	name	course	Instructor
1	Ravi	DBMS	Kumar

Suppose a new course 'AI' start but no student enrolled yet.

then cannot insert course detail without student data.

② Delete anomaliy : which occure when deleting a record unintentionally removes useful information

Ex:

SID	Name	course	Instructor
1	Ravi	DBMS	Kumar

if Ravi record is deleted course & Instructor information also lost.

(3) Modification Anomaly: which occurs same data is stored in multiple places & needs multiple updates.

Ex:

SID	Name	course	Instructor
1	Ravi	DBMS	Kumal
2	Meena	DBMS	Kumar

If instructor changes to Ramesh, It must updated in multiple rows. If one row is missed data becomes inconsistent.

7a) Transaction is sequence of database operation performed on a single logical unit of work.

Ex: ↳ Transfer money from Account A to Account B
↳ Add amount to B.

Transaction states: A transaction passes through several during execution

(1) Active state; transaction is executing, database operation like read/write are performed
Ex: money is being deducted from Account A

(2) Partially committed state: All operations are completed, transaction is waiting to make change permanent.
Ex: Amount deducted from A and added to B but not yet saved permanently

(3) Committed State: Transaction is completed succesfully, changes are permantly stored in database
Ex: Money transfer successfully completed

(4) failed state: Transaction cannot continue due to error.
Ex: Invalid data, Hardware failure, system crash

⑤ Aborted state: Transaction is rolled back database returns to previous consistent state.
ex: If money deducted but system crashes, deduction is ~~cafte~~ cancelled

7b) i) Assertion: It is constraint used to enforce conditions across one or more tables.
It ensures database consistency.

Syntax:

Create ASSERTION assertion_name
check (condition);

ex: Suppose total salary of employees should not exceed 100,000

CREATE ASSERTION salary_limit
check (sum(salary) <= 100000);

* automatically checks condition during INSERT, UPDATE & DELETE.

② Triggers: It is stored procedure that automatically execute when a specific database event occurs.

types of trigger:
* BEFORE trigges
* AFTER triggers.

CREATE trigger_name
BEFORE INSERT OR table_name
for each row
beign
statement
end

Ex : create trigger salar_check
Before Insert on employee
for each ROW.
Begin
if new.salary < 5000 then
SIGNAL SQLSTATE '45000'
set message_text = 'Salary too low'
End if
End.

8a) system log (transaction log) : It is file maintained by the DBMS that records all transaction activities it mainly used for recovery & maintaining db. consistency in case of system failure.

The following list s the types of entries called log records → that are written to log by action each perform., T refers to a unique transaction ID that is generated automatically by system and is used to identify each transaction.

① [start_transaction, T], indicate trasaction T has started execution

② [write_item, T, X, old value, new_value] Indicate the transaction T has changed the value of database item X from old_value to new_value

③ [read_item, T, X] Indicate that transaction T has read the value of database item X

④ [commit, T] Indicate transaction T has completed successfully.

⑤ [abort, T] it indicate trasaction T has been aborted

8b) ACID properties ensure reliable trasaction procesing.

① Atomicity: transaction is treated as single unit either all operation are completed or none are executed

Ex: money transfer should either debit & credit both or cancel completely

② Consistency: database moves from one valid state to another valid state

Example: Total balance before & after transaction remains correct.

③ Isolation: Multiple transaction execute independently without affecting each other.

Example: Two user updating same account should not create conflict.

④ Durability: once transaction is committed changes remain permantly even after system crash.

8c) stored procedure: It is precompiled collection of SQL statements stored in the database. It is executed as a single unit.

Syntax:

```
create procedure procedure-name
as
Begin
    SQL statement
end.
```

Exp: Suppose we want to reciene student detail based on department.

```
Create procedure getstudentbydep
@deptname varchar(20)
as
begin
 select * from student
 where department = @deptname;
end.
```

Advantage:

Improve performance.
Reduce network traffic
Reusable code.
Enhances seculity.
maintain data integrity.

9q) Two phase locking is concurrency control method used to maintain data consistency when multiple transaction execute simultaneously. It ensure that transaction are serializable.

Type of locks: ~~shared~~

* shared lock (S_lock): used for reading data, multiple transaction can read simultanoly
* Exclusive lock (X_lock): used for ~~reading~~ writing data, only one transaction can modify data.

Two Phase of 2PL:

1) Growing phase: Transaction can acquire locks, and cannot release any lock. locks are added but not removed

② Shrinking Phase: Transaction can release locks and cannot acquire new locks.
locks are removed but not added.

Ex:

T1	T2
Lock_X(A)	Lock_S(A)
Read(A)	Read(A)
write(A)	unlock(A)
unlock(A)	

9b) A schedule is sequence in which operation of multiple transaction are executed.
It shows order of read & write operation

① Serial schedule: transaction one after another

Ex:
T1 : read(A)
T2 : write(A)
T2 : Read(B)
T2 : write(B)

② non serial schedule: transaction execute simultaneously.

Ex:
T1 : Read(A)
T2 : Read(B)
T1 : write(A)
T2 : write(B)

③ Serializable schedule: non-serial schedule producing same result as serial schedul.

9c) Need for concurrency control.

Concurrency control ensures that multiple transaction executed simultaneously donot create inconsistency in database.

① prevent lost update problem: occure when two transaction update same data & one update is lost.

Ex: Initial balance = 5000/-

transaction T1 withdraw = 500/-

Transaction T2 deposits = 1000/-

without control.

T1 update balance = 4500

T2 update balance = 6000

correct balance should be 5500. one update lost

② Prevent Dirty Read Problem: Transaction read uncommitted data from another transaction.

Ex: T1 update balance but not commited

T2 read update balance.

if T1 roll backs, T2 used wrong data

③ prevent unrepeatable Read: Transaction read same data twice but get different results.

④ Maintain database consitency: ensure database remains accurate.

Ex: without concurrency control.

T1: read(A)
T2: Read(A)
T1: write(A)
T2: write(A)

} data may be inconsistent

10 a) NoSQL is type of database designed to store and manage large volume of unstructured or semi structured data.

CAP theorem:

① Consistency (C): All users see the same data at the same time.

Ex: If data is updated, every user immediately see update value.

② Availability (A): System alway responds to user requests.

Ex: Even if some server fail, system continues working

③ Partition Tolerance (P) system continue working even if network communication fails between nodes.

CAP combinations

① CA (Consitency + Availabiry)

* Works when network partion does not occne
* Ex: tradition relational DB.

② CP (consitency + Partition Tolerance)

* maintain data accuracy but may reduce availabiliy
* Ex: Mango DB, HBase

③ AP (Availabiliy + Partition Tolerance)

* System remain available but may show temporary inconsistent data.

Ex: Cassandra, DynamoDB.

10 b) Document Based Database store data in the form of documents instead of rows and columns.

CRUD operation in Mongo DB.

① Create (Insert Date)
used to insert new document

Ex:
```
db.Student.insertone ( {
    SID : 101,
    Name : "Ravi",
    Depaltmnt : "CSE"
});
```

② Read (Rebrine.Data): used to fetch documents.
```
db.student.find ({SID:101});
```

③ Update (Modify Date): used to update existing documents
```
db.students.updateOne (
  { SID:101 },
  {$set {Department : "ISE"}}
);
```

④ Delete (Remove Data): used to delete documents
```
db.student.deleteOne ({SID:101});
```

10 c) Graph database store data in the form of nodes, edges, and relationships

Ex: person → friend → Person

Neo4j: Neo4j is a popular graph database that uses graph structures to store and retrieve data efficiently

Features of Neo 4j -

* Uses graph model.
* Support ACID properties
* High performance
* .

Ex :

node

Student : Ravi ;

Relationship

Ravi → friend → Meena

* Cypher Query example ; create node.

```
create (S: student {name : "Ravi"}).
```

* Create Relationship

```
create (a: student {name : "Ravi"}),
(b: student {name : "Meena"}),
(a) → [: Friend] → (b)
```

* Retriere Data.

```
MATCH (S: student)
RETURN S.
```

Applications.

14/2/24