Answer Key									
Internal Assessment Test III– September 2022									
Sub:	**DATABASE MANAGEMENT SYSTEMS**							**Sub Code:**	**20MCA21**
Date:	**24-09-2022**	Duration:	90 mins	Max Marks:	50	**Sem:**	**II**	**Branch:**	**MCA**

CMR INSTITUTE OF TECHNOLOGY

USN | | | | | | | | | | |

Internal Assessment Test III – September 2022

Sub:	**Database Management System**							**Sub Code:**	**20MCA21**
Date:	**24-09-2022**	**Duration:**	**90 min's**	**Max Marks:**	**50**	**Sem:**	**II**	**Branch:**	**MCA**

Note : Answer FIVE FULL Questions, choosing ONE full question from each Module

	PART I	MARKS	OBE	
			CO	RBT
1	What are Views in SQL? Discuss on methodologies to implement views in SQL. Explain with an example. **OR**	[10]	CO3	L2
2	List and explain transaction isolation levels.	[10]	CO5	L2
3	**PART II** Explain in brief database backup and recovery from catastrophic Failure. **OR**	[10]	CO5	L1
4	What are triggers in SQL? Explain about Triggers in SQL with Suitable example.	[10]	CO3	L2
5	**PART III** Explain Schema Changes statements in SQL. **OR**	[10]	CO3	L1
6	Why concurrency control and recovery are needed in DBMS? Explain lost update and Dirty read problems with example.	[10]	CO5	L3
7	**PART IV** Explain two phase locking concurrency control protocol. **OR**	[10]	CO5	L3

8	Explain in detail about PL/SQL and Stored procedures.	[10]	CO3	L3
9	**PART V** Define Transaction. Explain ACID properties of a transaction in detail along with state diagram. **OR**	[10]	CO5	L2
10	Explain the transaction support in SQL	[10]	CO5	L2

1. **What are Views in SQL? Discuss on methodologies to implement views in SQL. Explain with an example.**

   View: Any relation that is not part of the logical model, but is made visible to a user as a virtual relation

   • View Definition: create view as ; where is any legal query expression.

   Example: create view sampview as select instid, instname, deptname from instructor; • Views having group by clause:

   create or replace view instview (deptname, sumsal) as select deptname, sum(salary) from instructor group by deptname;

   Materialised View: Certain database systems allow view relations to be stored, but they make sure that, if the actual relations used in the view definition change, the view is kept up-to-date

   Materialised view maintenance (or View Maintenance): The process of keeping the materialized view up-to-date

```
SQL> select * from instructor;

INSTI INSTNAME            DEPTNAME              SALARY
----- ------------------- ------------------- ---------
1002  raja                mca                   80000
1004  ramesh              ece                   66950
1005  suresh              eee                   66950
1007  rahul singh         mba                   77250
1001  kannan              mca                   56650
```

```
SQL> create view instview as
  2  select instid, instname, deptname
  3  from instructor;

View created.
```

```
SQL> select * from instview;

INSTI  INSTNAME                DEPTNAME
-----  ---------------------   ----------
1002   raja                    mca
1004   ramesh                  ece
1005   suresh                  eee
1007   rahul singh             mba
1001   kannan                  mca
```

View names may appear in a query any place where a relation name may appear

The attribute names of a view can be specified explicitly

```
SQL> create view deptsal as
  2  select deptname, sum(salary)
  3  from instructor
  4  group by deptname;
select deptname, sum(salary)
                 *
ERROR at line 2:
ORA-00998: must name this expression with a column alias
```

```
SQL> create or replace view instsal(deptname, sumsal) as
  2  select deptname, sum(salary)
  3  from instructor
  4  group by deptname;

View created.
```

2. **List and explain transaction isolation levels.**

Levels of isolation

Level 0 isolation does not overwrite the dirty reads of higher-level transactions

Level 1 isolation has no lost updates n Level 2 isolation has no lost updates and no dirty reads

Level 3 (true) isolation has repeatable reads

In addition to level 2 properties Snapshot isolation

3. **Explain in brief database backup and recovery from catastrophic Failure.**

Certain events can be classified as catastrophic failures. These include the damage caused by natural disasters, such as flooding in a computer room.

The main technique used to handle catastrophic failures including disk crash is that of database backup. The whole database and the log are periodically copied onto a cheap storage medium such as magnetic tapes. In case of a catastrophic system failure, the latest backup copy can be reloaded from the tape to the disk, and the system can be restarted. To avoid losing all the effects of transactions that have been executed since the last backup, it is customary to back up the system log by periodically copying it to magnetic tape. The system log is usually substantially smaller than the database itself and hence can be backed up more frequently. When the system log is backed up, users do not lose all transactions they have performed since the last database backup. All committed transactions recorded in the portion of the

system log that has been backed up can have their effect on the database reconstructed. A new system log is started after each database backup operation. Hence, to recover from disk failure, the database is first recreated on disk from its latest backup copy on tape. Following that, the effects of all the committed transactions whose operations have been entered in the backed-up copy of the system log are reconstructed.

4. What are triggers in SQL? Explain about Triggers in SQL with Suitable example.

- A trigger is a statement that the system executes automatically as a side effect of a modification to the database.
- Triggers are stored programs, which are automatically executed or fired when some events occur
- Two requirements to design a trigger mechanism:

(i) Specify when a trigger is to be executed.

- This is broken up into an event that causes the trigger to be checked and a condition that must be satisfied for trigger execution to proceed.

(ii) Specify the actions to be taken when the trigger executes

- Once we enter a trigger into the database, the database system takes on the responsibility of executing it whenever the specified event occurs and the corresponding condition is satisfied

- Need for Triggers
- used to implement certain integrity constraints that cannot be specified using the constraint mechanism of SQL
- useful mechanisms for alerting humans for starting certain tasks automatically when certain conditions are met
- Example: a warehouse wishes to maintain a minimum inventory of each item
- Triggers in SQL
  - for each row
  - Referencing new row as when statement

- Syntax:

CREATE [OR REPLACE] TRIGGER trigger_name

{BEFORE | AFTER } triggering_event ON table_name

[FOR EACH ROW]

[WHEN condition]

DECLARE

declaration statements

BEGIN

executable statements

EXCEPTION

exception_handling statements

END;

```
SQL> create table sample
  2  (myid number(5),
  3  ename varchar2(20)
  4  , salary number(5,2)
  5  );

Table created.

SQL> create or replace trigger samptrig
  2  before
  3  insert on sample
  4  for each row
  5  begin
  6     :new.ename := upper(:new.ename);
  7  end;
  8  /

Trigger created.
```

```
SQL> insert into sample values (1001, 'raja', 5000);

1 row created.
```

```
SQL> select * from sample;

     MYID ENAME                          SALARY
--------- -------------------- ----------
     1001 RAJA                          5000
```

5. Explain Schema Changes statements in SQL .

schema evolution commands available in SQL

used to alter a schema by adding or dropping tables, attributes, constraints, and other schema elements.

This can be done while the database is operational and does not require recompilation of the database schema.

Certain checks must be done by the DBMS to ensure that the changes do not affect the rest of the database and make it inconsistent.

- The DROP Command

used to drop named schema elements, such as tables, domains, or constraints.

DROP SCHEMA command can be used if a whole schema is no longer needed

two drop behavior options: CASCADE and RESTRICT

- CASCADE option: remove the database schema and all its tables, domains, and other elements
- RESTRICT option: the schema is dropped only if it has no elements in it; otherwise, the DROP command will not be executed

Eg.) DROP SCHEMA COMPANY CASCADE;

The DROP command can also be used to drop other types of named schema elements, such as constraints or domains.

DROP TABLE command not only deletes all the records in the table if successful, but also removes the table definition from the catalog.

If it is desired to delete only the records but to leave the table definition for future use, then the DELETE command should be used instead of DROP TABLE.

The ALTER Command

The definition of a base table or of other named schema elements can be changed by using the ALTER command.

For base tables, the possible alter table actions include:

- adding or dropping a column (attribute),
- changing a column definition, and
- adding or dropping table constraints.

```
alter table employee add column job varchar(12);
alter table employee drop column address;
alter table department alter column mgr_ssn drop default;
alter table employee drop constraint empsuperfk;
```

6. Why concurrency control and recovery are needed in DBMS? Explain lost update and Dirty read problems with example.

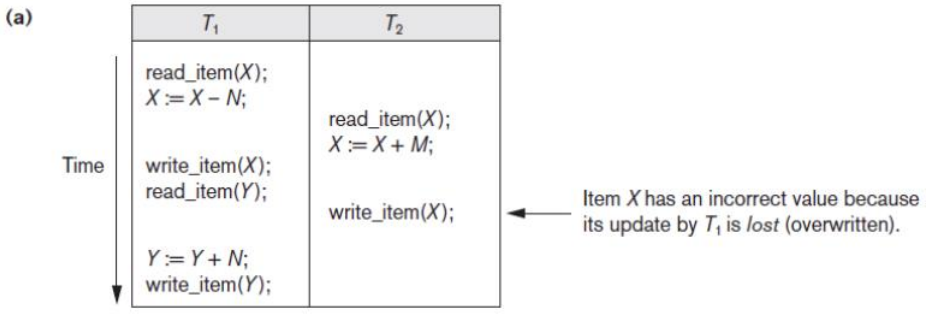
Transactions submitted by various users may execute concurrently

- Access and update the same database items
- Some form of concurrency control is needed

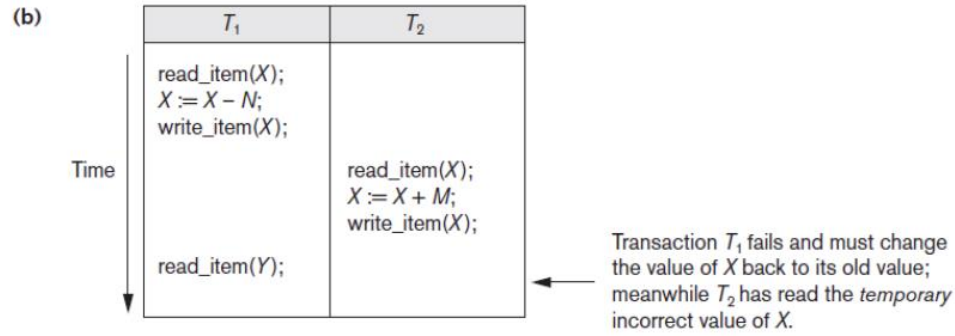
The lost update problem

- Occurs when two transactions that access the same database items have operations interleaved
- Results in incorrect value of some database items

The Lost Update Problem



The Temporary Update Problem



Temporary Update problem is also called as Dirty read.

7. Explain two phase locking concurrency control protocol.

Purpose of Concurrency Control

- To enforce Isolation (through mutual exclusion) among conflicting transactions.
- To preserve database consistency through consistency preserving execution of transactions.
- To resolve read-write and write-write conflicts.

Example: In concurrent execution environment if T1 conflicts with T2 over a data item A, then the existing concurrency control decides if T1 or T2 should get the A and if the other transaction is rolled-back or waits.

Two-Phase Locking Techniques

Locking is an operation which secures (a) permission to Read or (b) permission to Write a data item for a transaction. Example: Lock (X). Data item X is locked in behalf of the requesting transaction.

Unlocking is an operation which removes these permissions from the data item. Example: Unlock (X). Data item X is made available to all other transactions.

Lock and Unlock are Atomic operations.

Two-Phase Locking Techniques: Essential components

Two locks modes (a) shared (read) and (b) exclusive (write).

Shared mode: shared lock (X). More than one transaction can apply share lock on X for reading its value but no write lock can be applied on X by any other transaction.

Exclusive mode: Write lock (X). Only one write lock on X can exist at any time and no shared lock can be applied by any other transaction on X.

Conflict matrix

	Read	Write
Read	Y	N
Write	N	N

Lock Manager: Managing locks on data items.

Lock table: Lock manager uses it to store the identify of transaction locking a data item, the data item, lock mode and pointer to the next data item locked. One simple way to implement a lock table is through linked list.

Transaction ID	Data item id	lock mode	Ptr to next data item
T1	X1	Read	Next

Two-Phase Locking Techniques: Essential components

The following code performs the lock operation:

```
B: if LOCK (X) = 0 (*item is unlocked*)
   then LOCK (X) ← 1 (*lock the item*)
   else begin
            wait (until lock (X) = 0) and
            the lock manager wakes up the transaction);
   goto B
   end;
```

The following code performs the unlock operation:

```
LOCK (X) ← 0 (*unlock the item*)
if any transactions are waiting then
        wake up one of the waiting the transactions;
```

Lock conversion

Lock upgrade: existing read lock to write lock

```
if Ti has a read-lock (X) and Tj has no read-lock (X) (i ≠ j) then
    convert read-lock (X) to write-lock (X)
else
   force Ti to wait until Tj unlocks X
```

Lock downgrade: existing write lock to read lock

```
Ti has a write-lock (X)    (*no transaction can have any lock on X*)
 convert write-lock (X) to read-lock (X)
```

Two Phases: (a) Locking (Growing) (b) Unlocking (Shrinking).

Locking (Growing) Phase: A transaction applies locks (read or write) on desired data items one at a time.

Unlocking (Shrinking) Phase: A transaction unlocks its locked data items one at a time.

Requirement: For a transaction these two phases must be mutually exclusively, that is, during locking phase unlocking phase must not start and during unlocking phase locking phase must not begin.

Two-Phase Locking Techniques: The algorithm

T'1	T'2	
read_lock (Y);	read_lock (X);	T1 and T2 follow two-phase
read_item (Y);	read_item (X);	policy but they are subject to
write_lock (X);	Write_lock (Y);	deadlock, which must be
unlock (Y);	unlock (X);	dealt with.
read_item (X);	read_item (Y);	
X:=X+Y;	Y:=X+Y;	
write_item (X);	write_item (Y);	
unlock (X);	unlock (Y);	

8. Explain in detail about PL/SQL and Stored procedures.

PL/SQL programming language: developed by Oracle Corporation in the late 1980s

It is a procedural extension language for SQL

Notable facts about PL/SQL:

is completely portable, high-performance transaction-processing language

provides a built-in, interpreted and OS independent programming environment

Features of PL/SQL:

offers extensive error checking

offers numerous data types

offers a variety of programming structures

supports object-oriented programming

supports structured programming through functions and procedures

Syntax:

```
CREATE [OR REPLACE] PROCEDURE procedure_name
[(parameter_name [IN | OUT | IN OUT] type [, ...])]
{IS | AS}
BEGIN
   Execution section;
[EXCEPTION
  Exception section ]
END;
```

Create a procedure to display a welcome message "Welcome to SQL World"

```
SQL> create or replace procedure welcome
  2  is
  3  begin
  4    dbms_output.put_line('Welcome to SQL world');
  5  end;
  6  /

Procedure WELCOME compiled
```

- Execute the Procedure:

Method-1:

```
SQL> exec welcome;
Welcome to SQL world


PL/SQL procedure successfully completed.
```

Method-2:

```
SQL> begin
  2     welcome;
  3  end;
  4  /
Welcome to SQL world


PL/SQL procedure successfully completed.
```

Write a procedure to raise an increment of 15% for the given instructor id.

```
SQL> create or replace procedure salinc
  2  (empid in instructor.instid%type)
  3  is
  4  begin
  5     update instructor
  6     set salary = salary + salary * 15/100
  7     where instid = empid;
  8  end;
  9  /

Procedure created.
```

```
SQL> exec salinc(1003);

PL/SQL procedure successfully completed.
```

9. Define Transaction. Explain ACID properties of a transaction in detail along with state diagram.

Transaction

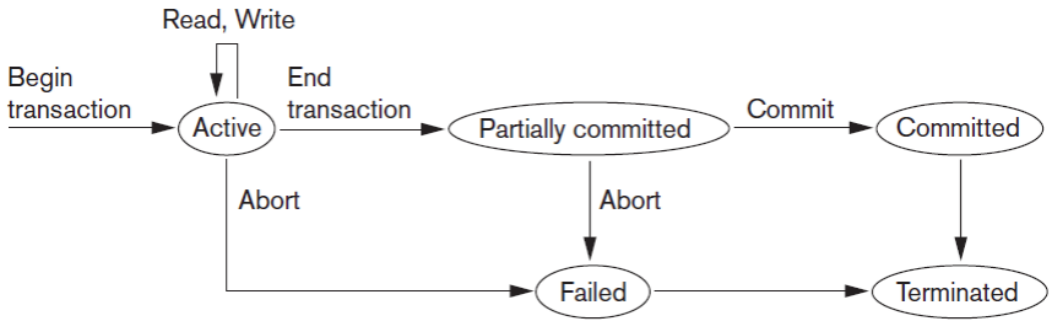
- Describes local unit of database processing

Transaction processing systems

- Systems with large databases and hundreds of concurrent users
- Require high availability and fast response time

ACID properties

- Atomicity
  - Transaction performed in its entirety or not at all
- Consistency preservation
  - Takes database from one consistent state to another
- Isolation
  - Not interfered with by other transactions
- Durability or permanency
  - Changes must persist in the database



10.Explain the transaction support in SQL

No explicit Begin_Transaction statement

Every transaction must have an explicit end statement

- COMMIT
- ROLLBACK

Access mode is READ ONLY or READ WRITE

Diagnostic area size option

- Integer value indicating number of conditions held simultaneously in the diagnostic area

- Isolation level option
  - Dirty read
  - Nonrepeatable read
  - Phantoms

	Type of Violation		
Isolation Level	Dirty Read	Nonrepeatable Read	Phantom
READ UNCOMMITTED	Yes	Yes	Yes
READ COMMITTED	No	Yes	Yes
REPEATABLE READ	No	No	Yes
SERIALIZABLE	No	No	No

- Snapshot isolation
  - Used in some commercial DBMSs
  - Transaction sees data items that it reads based on the committed values of the items in the database snapshot when transaction starts
  - Ensures phantom record problem will not occur