ASSIGNMENT - 1 DBMS

Name :- G. Abhamay Saa Karthik

VTU NO :- VTU 31764

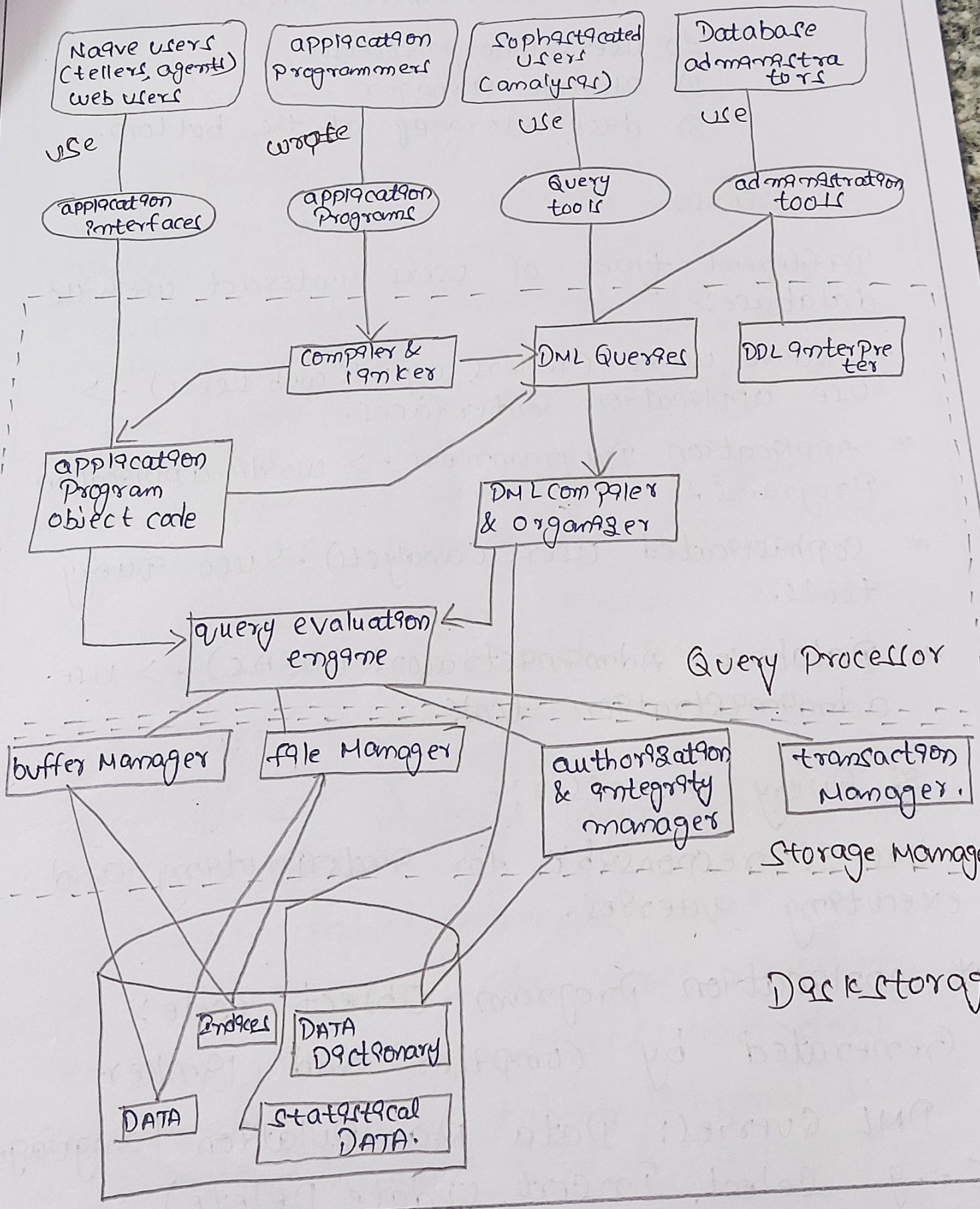
Subject :- DBMS

5

A25

ASSIGNMENT-I

1) Explain the detail about database system architecture with a neat diagram?



DATABASE SYSTEM ARCHITECTURE:-

The database system architecture can be divided into three main parts:

1) users, Query Processor
2) Storage Manager
3) disk storage at the bottom.

1. users:-

Different types of users interact with the database:-

* Naive users (tellers, agents, web users) -> use application interfaces.
* Application programmers -> write application programs.
* Sophisticated users (analysts) -> use query tools.
* Database Administrators (DBAs) -> use administration tools.

2. Query Processor:-

This is responsible for interpreting and executing queries.

* Application Program Object code: Generated by compiler and linker.
* DML Queries: Data Manipulation language (e.g., Select, Insert, Update, Delete).

* DDL interpreter: interprets schema definitions (tables, constraints).
* DML compiler and organizer: checks syntax and optimizes queries.
* Query Evaluation Engine: Executes optimized queries.

3. STORAGE MANAGER:-

The storage manager controls how data is stored and retrieved.

* Buffer Manager: Manages disk IIo by storing frequently accessed data in memory.
* File Manager:- Manages allocation of space and file structures.
* Authorization & integrity Manager: Ensures data security & integrity constraints.
* Transaction Manager: Ensures consistency, concurrency control, and recovery.

4. DISK STORAGE:-

This is the physical layer where actual data resides.

* DATA:- Tables & records.
* Indices:- used for fast searching
* Data Dictionary:- stores metadata (information about tables, schemas, users).
* Statistical Data:- used for query optimization.

JOINED RELATIONS :-

2) Explain in Detail about various queries & joints with the suitable examples?

-> Join operations take two relations and return as a result another relation.

These additional operations are typically used as subquery expressions in the from clause.

JOIN CONDITION :- Defines which tuples in the two relations match, and what attributes are present in the result of the join.

JOIN type :- defines how tuples in each relation that do not match any tuple in the other relation (based on the join condition) are treated.

JOIN TYPES
Inner join
left outer join
right outer join
full outer join

JOIN Conditions
natural
on <predicate>
using $\langle A_1, A_2, \cdots A_n \rangle$

DATABASE EXAMPLES - JOIN

INSTRUCTOR

ID	name	dept_name
10101	Srinivasan	Comp. Sci
12121	Wu	Finance
15151	Mozart	Music.

2) Explain in detail about various Queries & joints with the suitable examples?

A) Queries in DBMS

Queries are used to retrieve, insert, update, & delete data from a database using SQL.

Types of Queries:

1. Select Query:- Used to fetch data from a table.

SQL:

SELECT name, age FROM students;

Fetches name & age of all students.

2. Insert Query:- used to add new records.

SQL

INSERT INTO students (id, name, age)

3. Update Query:- Used to modify existing records.

SQL

UPDATE students SET age = 21, WHERE id = 1;

4. Delete Query:- Used to remove records.

SQL

DELETE FROM students WHERE id = 1;

INNER JOIN

-> Returns only the matching rows from both tables based on a given condition.

```
SELECT instructor.ID, name, course_id
 FROM instructor
INNER JOIN teacher
ON instructor.ID = teacher.ID;
```

ID	name	course_id
10101	Srinivasan	CS-101
12121	Wu	FIN-201

LEFT OUTER JOIN

-> Returns all rows from the left table and the matching rows from the right table.

```
SELECT instructor.ID, name, course_id
 FROM instructor
RIGHT JOIN teacher
ON instructor.ID = teacher.ID;
```

ID	name	course_id
10101	Srinivasan	CS-101
12121	Wu	FIN-201
76766	NULL	BIO-101

RIGHT OUTER JOIN

-> Returns all rows from the right table and the matching rows from the left table.

```
SELECT instructor.ID, name, course_id
FROM instructor
RIGHT JOIN teaches
ON instructor.ID = teaches.ID;
```

ID	name	course_id
10101	Srinivasan	CS-101
12121	Wu	FIN-201
76766	NULL	BIO-101

FULL OUTER JOIN

-> Returns all rows from both tables, matching rows where available, and filling NULLs where there's no match.

Left join:

```
SELECT instructor.ID, name, course_id
FROM instructor
LEFT JOIN teaches ON instructor.ID = teaches.ID
```

UNION:-

```
SELECT instructor.ID, name, course_id
FROM instructor
RIGHT JOIN teaches ON instructor.ID =
teaches.ID;
```

EQUI JOIN

A type of INNER JOIN that uses an equality (=) operator to match rows.

```
SELECT instructor.ID, name, course_id
FROM instructor, teaches
WHERE instructor.ID = teaches.ID;
```

ID	name	Course_id
10101	Srinivasan	CS-101
12121	Wu	FIN-201

CROSS JOIN (cartesian product)

Returns the cartesian product of two tables every row from the first table joined with every row from the second table.

```
SELECT name, course_id
  FROM instructor
CROSS JOIN teaches;
```

Name	Course_id
Srinivasan	CS-101
Srinivasan	FIN-201
Srinivasan	BIO-101
Wu	CS-101
Wu	FIN-201
Wu	BIO-101
Mozart	CS-101
Mozart	FIN-201
Mozart	BIO-101.

IT

ASSIGNMENT - 3 - Unit-3

1. Normalization and its various types of Normalization

A) Normalization and its various Types

Normalization is a process in Database Management System (DBMS) used to organise data in a database to reduce redundancy (duplicate data) and improve data integrity

(i) First Normal Form (1NF)

-> Each cell must contain atomic (indivisible) values

-> No repeating groups or arrays allowed

Ex:-

St_ID	St_Name	St_Phone	St-group
123	Rakesh	9840723512	Physics
125	Smitha	9135462193 1341278963	Stat
126	Archana	7365432101 8694532101	Chemistry

New table:-

St_ID	St_Name	St_phone	St_group
123	Rakesh	9840735112	physics
125	Smatha	9135462193	stats
125	Smatha	1341278963	stats.

Second Normal form (2NF)

-> In the 2NF, first table must be in 1NF.

-> In the 2NF, all non key attributes are full functionally dependent on the Primary key.

-> Every Non key attribute should be FFD on key attribute

If P->A holds, then there should not be any proper subset of q of P

Q->A.

St_ID	St_Name	Prof_ID	Prof_Name	Grade
101	ABC	2	Sameer	4
102	xyz	3	Naranian	6
103	PQR	1	Sushmatha	5

St_ID -> Primary key
and No multi valued are there so Satisfies 1NF.

fD

St_ID	St_Name
101	ABC
102	xyz
103	PQR

fD

Prof_ID	Prof_Name
1	Sushmitha
2	Sameer
3	Narayan.

GRADES:-

FFD

St_ID	(PK) Prof_ID	Grade
101	2	4
102	3	6
103	1	5

3. Third Normal Form (3NF)

-> A relation will be in 3NF
-> It not contains any transitive dependency.

The Non-key attributes should not have inter dependencies among them, and the Non-key attributes should fully functionally depend on key attribute

Then it is called 3NF Principle.

-> By using 3NF to achieve data integrity and Data Duplication.

Transitive dependency

If A -> B (B functionally dependent on A)
B -> C
A -> C (C is indirectly dependent on A)
It is called T.D.

Ex:-

Student_ID	Std_Name	DeptName	Dept_Head
101	Alice	CSE	Dr. Rao
102	Bob	ECE	Dr. Kumar
103	Carol	CSE	Dr. Rao

Student table:-

Student_ID	Student_Name	DeptName
101	Alice	CSE
102	Bob	ECE
103	Carol	CSE

Dept_table:-

Dept_Name	Dept_Head.
CSE	Dr. RAO
ECE	Dr. Kumar
CSE	Dr. RAO

4. BCNF (Boyce - Codd Normal form)

-> It is advanced version of 3NF
-> It is an 3NF
-> For every FD A->B A->Super key
-> A should be super key of a table

Ex:-

Student	Course	Teacher
Ramesh	Physics	Kishore
Kumar	Chemistry	Ramu
Sruthi	maths	Sanjeev
Vinay	Physics	Kishore.

- eys -> {Student, Course} ->

To eliminate redundant data we move to 2 tables

Student	Course
Ramesh	physics
kumar	chemistry
Sruthi	maths
vinay	Physics

Course	Teacher
Physics	kishore
chemistry	RamU
Maths	SanjeeV

4NF (Fourth Normal Form)

-> First relation in BCNF (or) 3.5NF

-> It may not contain more than one multivalued attributes.

For a dependency

A -> B

-> If for a single value of "A" multiple values of B exists, then the relation will be multi-valued dependency.

St_ID	Course	hobbies
111	maths	Dancing
111	Computer	singing
222	Chemistry	Dancing
444	Sanskrit	Cricket
555	Physics	Hockey.

St_ID	course
111	maths
111	computers
222	Chemistry
444	Sanskrit
555	physics.

St_ID	hobby
111	Dancing
111	singing
222	Dancing
444	Cricket
555	hockey.

ASSIGNMENT - Unit-4

2. Deadlock and its handling?

A) A deadlock occurs on a database when two or more transactions are waiting for each other to release locks on resources, Preventing further progress

Example:

* Transaction T1 locks Resource A and waits for Resource B.
* Transaction T2 locks Resource B and waits for Resource A.

→ Both wait forever -> deadlock

Conditions for Deadlock (coffman conditions)

1. Mutual Exclusion:- only one transaction can use a resource at a time
2. Hold & Wait:- A transaction holds one resource and waits for another
3. No Preemption:- Resources cannot be forcibly taken away.
4. Circular wait:- A circular chain of waiting transactions exists.

Deadlock Handling Techniques

Method	Description
1. Deadlock Prevention	Avoids deadlocks by denying at least one coffman condition Example: Assign resource ordering or use time stamp based methods (wait-die, wound-wait)
2. Deadlock Avoidance	The system checks before granting a lock to ensure no deadlock will occur Example: Banker's algorithm checks safe states.
3. Deadlock detection	The system allows deadlocks to occur but periodically checks for them using a wait-for graph (WFG).
4. Deadlock Recovery	once detected, resolve by rolling back one or more transactions (victim selection).

3. Explain about RAID storage and its types? short.

A) RAID (Redundant Array of Independent Disks) is a data storage technology that combines multiple physical hard drives into a single logical unit.

The main goals are:

-> Improved performance
-> Increased storage capacity
-> Data redundancy (fault tolerance).

RAID is commonly used in servers, data centers, and storage systems to ensure data reliability and faster access.

2. key features

Feature	Description
Redundancy	Data is duplicated or spread to protect against drive failure.
Performance	multiple disks can read/write data simultaneously.

Fault Tolerance	System continues to function even if one drive fails.
Striping	Data is split across multiple disks to improve speed.
Mirroring	Data is copied identically on two or more disks.
Parity	Error checking information is stored to recover data in case of disk failure.

Types of RAID

RAID Level:- Raid 0 (Striping)

Technique used:- Data divided into blocks and spread across disks.

Description:- Fast performance (no redundancy)

Advantages:- High speed.

Disadvantages:- No fault tolerance - if one disk fails, all data lost.

RAID LEVEL:- RAID 1 (Mirroring)

Technique used:-
Same data copied on two disks

Description:- Provided redundancy

Advantages:- High reliability, simple recovery

Disadvantages:- Storage cost doubles.

RAID LEVEL:- Bit level striping with error correction (ECC)

Technique used:- Rarely used.

Description:- Rarely used

Advantages:- Error correction possible.

Disadvantages:- Expensive, complex.

RAID LEVEL:- RAID 3

Technique used:- Byte level striping with dedicated parity disk.

Description:- All drives work together.

Advantages:- Good for large sequential data.

Disadvantages:- Parity disk may be a bottleneck

RAID LEVEL :- RAID 4

Technique used :- Block level striping with dedicated parity disk.

Description :- Parity used for recovery

Advantages :- Fast read performance

Disadvantages :- Single parity disk bottleneck.

RAID LEVEL :- RAID 5

Technique used :- Block level striping with distributed Parity.

Description :- Most common RAID Level

Advantages :- Good balance between performance & redundancy.

Disadvantages :- Complex rebuild if a disk fails.

RAID LEVEL :- RAID 6

Technique used :- LIKE RAID 5 but with two Parity blocks.

Description :- Higher fault tolerance

Advantages :- can tolerance two disk failures

Disadvantages :- Slower write performance.

RAID LEVEL :- RAID 10 (1+0)

Technique used :- combination of RAID 1 & RAID 0 (mirroring + striping)

Description :- very fast & fault-tolerant

Advantages :- High speed & safety

Disadvantages :- Requires at least 4 disks, Costly.