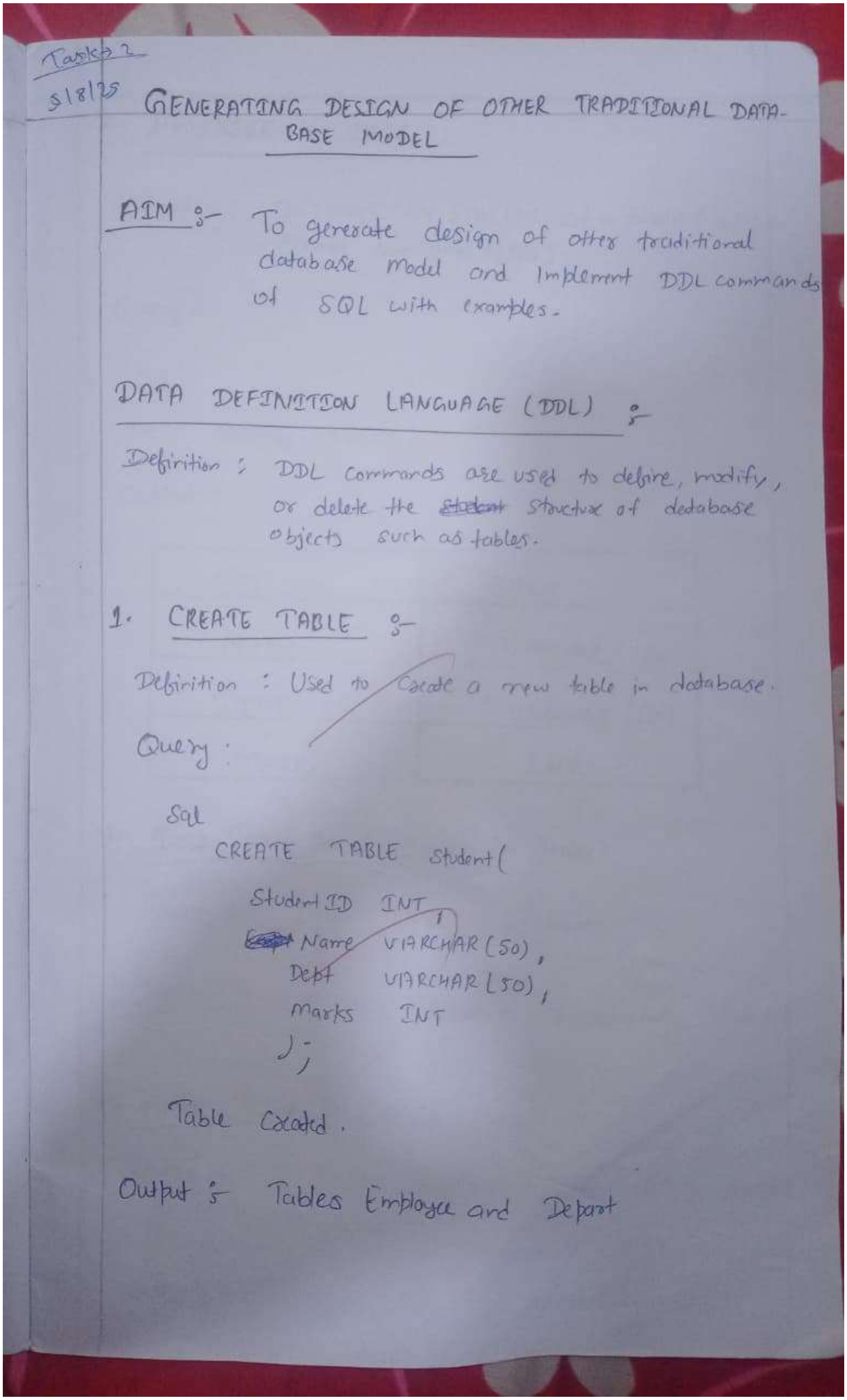

Task-2

5/8/25

GENERATING DESIGN OF OTHER TRADITIONAL DATABASE MODEL

AIM :- To gererate design of other traditional database model and Implement DDL commands of SQL with examples.

DATA DEFINITION LANGUAGE (DDL) :-

Definition : DDL commands are used to define, modify, or delete the ~~Student~~ structure of database objects such as tables.

1. CREATE TABLE :-

Definition : Used to Create a new table in database.

Query :

Sql

```
CREATE TABLE Student(
    StudentID INT,
    Name  VARCHAR(50),
      Dept  VARCHAR(50),
      marks  INT
    );
```

Table Created.

Output :- Tables Employee and Depart

2. DESCRIBE or DESC

Definition :- Displays the structure of a table (column names and data types).

Query :-

Sal

DESC ~~[illegible]~~ students ;

Output :

ID	INT
NAME	VARCHAR (100)
DEPT	VARCHAR (50)
MARKS	INT

3. Drop Table :- (Deletes the table)

Query :- DROP TABLE students ;

OUTPUT :- Table students succesfully deleted.

4. Alter Table :- (Adds fields in a table)

Query :- > Alter table Students ADD subject VARCHAR (50);

Output :- Subject field is succesfully added.

II DML Queries :-

* Insert Into :- (inserts new rows in table)

Query :-

```
> INSERT INTO Students (ID , name, Dept,
    marks) VALUES (1 , 'RAM', 'CSE', 80);
```

Output :-

```
1 row inserted to Student.
```

* SELECT :- (Retrives data from one or more tables).

Query :-

sql

```
SELECT * FROM Students
```

Output :-

ID	name	Dept	Marks
1	RAM	CSE	80
2	Sam	ECE	70
3	Tom	CIVIL	90

* **Update** :- (Modifies existing data)

Query :-

```
> UPDATE Students SET ID = 20 WHERE
  Marks = 80;
```

output :

```
1 row updated.
```

* **Delete** :- (Delete one or more rows from a table)

Query :-

```
> DELETE FROM Students WHERE ID=2;
```

output :-

```
1 row deleted;
```

* **SELECT** : (Retrives specefic record that satisfy the conditions)

Query :-

```
> SELECT * FROM Students WHERE Name =
  'Sam';
```

Output :-

ID	Name	Dept	Marks
1	RAM	CSE	80

VEL TECH	
EX NO.	2
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	-
TOTAL (20)	15
[illegible] WITH DATE	8/9/25

RESULT : Therefore , DDL and DML commands Using My SQL has been implemented Successfully .

Task->2.1
Date->12/8/25

DDL AND DML COMMANDS WITH CONSTRAINTS

AIM :- To design and Implement a database for a Sports Event Management System that manages information about teams, Coaches, players and matches using SQL DDL.

STEPS :- 1. Identify Entities

- Team
- Coach
- Player
- Match

2. Identify Attributes

- Team -> (TeamID, Name, HomeGround, CoachID)
- Coach -> (CoachID, Name, Experience, TeamID)
- Player -> (PlayerID, Name, Position, TeamID)
- Match -> (MatchID, matchDate, HomeTeamID, AwayTeamID)

3. Identify Relationships

- Team-Coach -> One-to-One (each team has one coach).
- Team- Player -> One-to-many (to team has many players)
- Match - Team -> Many-to-Many (each match has two teams).

4. Reframe Relation with Keys & Constrainsts

-> Primary Keys -> TeamID, CoachID, PlayersID matchID.

-> Foreign keys -> CoachID in Team, TeamID in Player, HomeTeamID / AwayTeamID in Match

* Constraints!

Unique Coach per Team

A team Cannot play against itself.

(1) DDL COMMANDS :-

* Table for Teams

```
CREATE TABLE Team (
    Team_ID . INT PRIMARY KEY,
  TeamName   VARCHAR(50) NOT NULL,
   HomeGround  VARCHAR (50),
   CoachID    INT UNIQUE,
   FOREIGN KEY (CoachID) REFERENCES Coach (CoachID)
 );
```

* Table for Coaches

```
CREATE TABLE Coach (
   CoachID   INT PRIMARY KEY,
   CoachName   VARCHAR (50) NOT NULL,
    Experience  INT,
   TeamID     INT UNIQUE,
    FOREIGN KEY (Team ID) REFERENCES Team (TeamID)
 );
```

* Table for Players

```
 CREATE TABLE Player (
    PlayerID  INT PRIMARY KEY,
    PlayerName  VARCHAR(50) NOT NULL,
     Position     VARCHAR(30),
     TeamID       INT,
     FOREIGN KEY (TeamID) REFERENCES Team (Team ID)
 );
```

Table for matches

```
CREATE TABLE Match (
  matchID  INT PRIMARY KEY,
  MatchDate   DATE NOT NULL,
HomeTeamID  INT,
AwayTeam ID  INT,
FOREIGN KEY (HomeTeam ID) REFERENCES Team (TeamID),
FOREIGN KEY (Away Team ID) REFERENCES Team (TeamID),
CHECK (HomeTeam ID <> AwayTeamID)
);
```

```
SQL > DESC team;
```

Name	Null?	Type
TEAM ID	NOT NULL	NUMBER (38)
NAME	NOT NULL	VARCHAR2(50)
HOMEGROUND		VARCHAR2(50)
COACH ID		NUMBER (38)

```
SQL > DESC player;
```

~~RESULTS~~

Name	Null?	Type
PLAYER ID	NOT NULL	NUMBER (38)
NAME	NOT NULL	VARCHAR2(50)
POSITION		VARCHAR2(30)
TEAM ID		NUMBER (38)

SQL> desc Coach;

Name	Null ?	Type
MATCHID	NOT NULL	NUMBER (38)
MATCHDATE	NOT NULL	DATE
HOMETEAMID		NUMBER (38)
AWAYTEAMID		NUMBER (38)

SQL > desc match;

Name	Null ?	Type
MATCHID	NOT NULL	NUMBER (38)
MATCHDATE	NOT NULL	DATE
HOMETEAMID		NUMBER (38)
AWAYTEAMID		NUMBER (38)

DML COMMANDS :-

(I) INSERT INTO Team Values (1, 'worriors', StadiumA, NULL)

(II) Select * From team;

Team ID	Name	Homeground	Coach ID
1	Worriers	StadiumA	NULL

(iii) UPDATE :-

```
Update Team Set 'Worriers' = 'Hustler' where
team ID = 1 ;
```

Team ID	Name	Home Ground	Coach ID
1	Hustler	Stadium A	NULL

(iv) DELETE :-

```
DELETE  FROM  team
  2  WHERE  TEAMID = 1;

1 row deleted ;
```

VEL TECH	
EX NO.	2.1
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	
TOTAL (20)	15
SIGN WITH DATE	CA 12/8h

RESULT :-

Thus, design and Implementation of a database for a Sports' Event Management System by using SQL DDL is Successfully Completed