Student Database Management System

Submitted in partial fulfilment of requirement

for the completion of

DATABASE MANAGEMENT SYSTEMS LABORATORY

B.Tech Computer Science Engineering

(AIML-A)

By

1. J.HARIHARAN (22071A6622)

2. M.KHYATHI (22071A6638)

3. M.SPOORTHY (22071A6640)

4. S. RISHIT (22071A6655)

2023-2024

Under the guidance of
Mrs.K.ARCHANA
Asst.Prof DEPT OF CSE(AIML & IOT)

VALLURIPALLI NAGESWARARAO
VIGNANA JYOTHI INSTITUTE OF ENGINEERING & TECHNOLOGY
(AUTONOMOUS INSTITUTE)
NAAC ACCREDITED WITH 'A GRADE
VignanaJyothi Nagar, Bachupally, Nizampet (s.o), Hyderabad 500090
Phone no: 040-23042758/59/60, Fax: 040-23042761
Email:postbox@vnrvjiet.ac.in Website: www.vnrvjiet.ac.in

VNR Vignana Jyothi Institute of Engineering and Technology (Affiliated to J.N.T.U, Hyderabad)
Bachupally(v), Hyderabad, Telangana, India.

CERTIFICATE

This is to certify that the project entitled **"Student Database Management System"** submitted in partial fulfilment for the course of Database Management Systems Laboratory (22PC2AM201) being offered for the award of B.Tech (CSE-AIML-A) by VNR VJIET is a result of the bonafide work carried out by **22071A6622, 22071A6638, 22071A6640, 22071A6655** during the year 2023-2024. This has not been submitted for any other certificate or course. This work is carried out under supervision and has not been submitted to any other University/Institute for award of any degree/diploma.

Ms.K.Archana
Assistant Professor
CSE - AIML & IoT Department
VNRVJIET

Dr.N.Sandhya
Professor and Head
CSE - AIML & IoT Department
VNRVJIET

DECLARATION

This is to certify that our project report titled **"Student Database Management System"** submitted to Vallurupalli Nageswara Rao Institute of Engineering and Technology in complete fulfilment of requirement for the award of Bachelor of Technology in CSE-AIML is a bona fide report to the work carried out by us under the guidance and supervision of **Ms.K.Archana**, Assistant Professor, Department of CSE-AIML & IoT, Vallurupalli Nageswara Rao Institute of Engineering and Technology. To the best of our knowledge, this has not been submitted in any form to other university or institution for the award of any degree or diploma.

22071A6622
J.Hariharan
CSE-AIML-A

22071A6638
M.Khyathi
CSE-AIML-A

22071A6640
M.Spoorthy
CSE-AIML-A

22071A6655
S.Rishit
CSE-AIML-A

Table of Contents:

Introduction of the Database System:

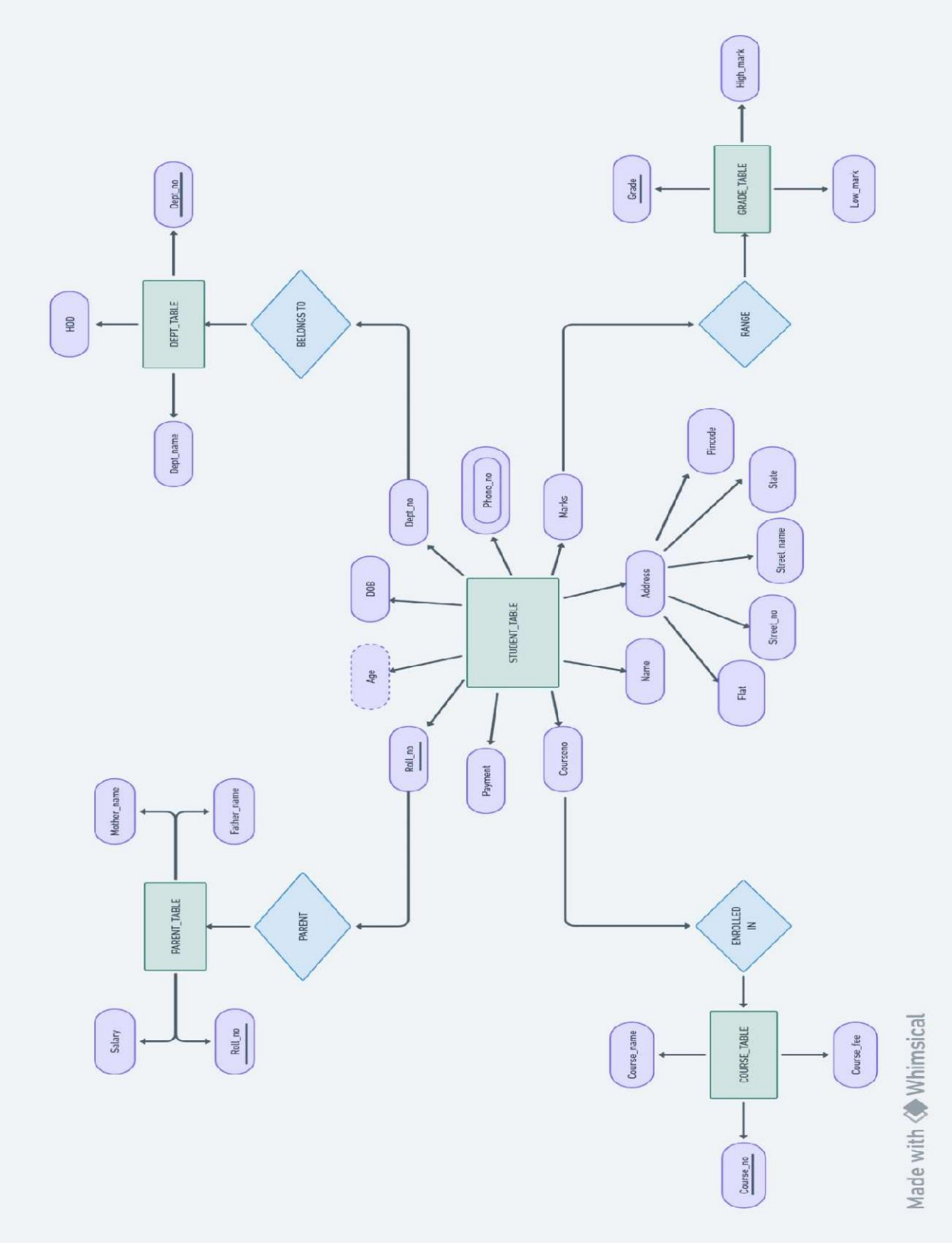
A database system is a software application that allows you to organize, store, and retrieve large amounts of data efficiently. It helps you manage and manipulate data in a structured way, making it easier to access and analyze information.

In a database system, data is organized into tables, which consist of rows and columns. Each row represents a record or an instance of data, while each column represents a specific attribute or characteristic of that data. This structured approach allows you to establish relationships between different tables, enabling you to retrieve and combine data from multiple sources.

Database systems are used in various industries and applications, such as e-commerce, banking, healthcare, and more. They provide a reliable and secure way to store and manage data, ensuring data integrity and consistency. By using database systems, organizations can make informed decisions, improve efficiency, and enhance their overall operations.

The ideology behind database systems is all about organizing and managing data effectively. It focuses on principles like data integrity, consistency, and security. Database systems use data models to structure and represent data, ensuring that it's stored in a logical way. They also prioritize data independence, separating the way data is stored from the applications that use it. Additionally, database systems emphasize data security, protecting sensitive information through authentication, encryption, and access control. Overall, the ideology of database systems aims to provide efficient data management and secure data storage.

ER Diagram Modelling:



Entities, Attributes and Relationships:

Entity: An entity is an object that exists. It doesn't have to do anything; it just has to exist. In database administration, an entity can be a single thing, person, place, or object. Data can be stored about such entities. A design tool that allows database administrators to view the relationships between several entities is called the entity relationship diagram (ERD).

The following Entities are used in our Database:

1. Student
2. Parent
3. Course
4. Department
5. Grade

Attribute: An attribute defines the information about the entity that needs to be stored. If the entity is an student, attributes could include name, rollno, deptno, marks, and address. An entity will havezero or more attributes, and each of those attributes apply only to that entity.s

Relationship: A relationship, in the context of databases, is a situation that exists between two relational database tables when one table has a foreign key that references the primary key of the other table. Relationships allow relational databases to split and store data in different tables, while linking disparate data items.

Following is a detailed description of every Entity, its Attributes and Relations in between them as employed in our Database:

1). Student:

Each Student Entity has a **Rollno**, **Name, DOB, Deptno, Payment, Age, Phone_numbers, course_number, marks, Address** associated with it.

➔ **It has 20 tuples**.

2). Parent:

Each Parent Entity has **Rollno**, **Salary, Mother_name, Father_name**

➔ **It has 20 tuples**.

➔ **Each parent detail is associated with Student rollno**.

3). Course:

The course entity has **Course_no, Course_name, Course_fee**.

➔ **It has 5 tuples**.
➔ Each student is enrolled in a course where **Course_no** is a primary key.

4). Department:

The department entity has **Dept_no, Dept_name, HOD**.

➔ **It has 5 tuples**.

5). Grade:
The grade entity has **grade, low_mark, high_mark**

➔ **It has 6 tuples.**

Process of Normalization:

Normalization: Normalization is the process of minimizing redundancy from a relation or set of relations. Redundancy in relation may cause insertion, deletion and update-based anomalies. So, it helps to minimize the redundancy in relations. Normal forms are used to eliminate or reduce redundancy in database tables.

There exist three main types of Normal forms, each being associated with a increasing degree of Normalization:

➔ **First Normal Form:** A relation is in first normal form if every

attribute in that relation is singled valued attribute.
Ex: In our database, consider the attribute **Phone Number** under entity **Student.** The numbers are multi-valued. Hence, it need to be converted to 1NF. So we converted this phone number to phone number1 and phone number2. Now, it is in 1NF.

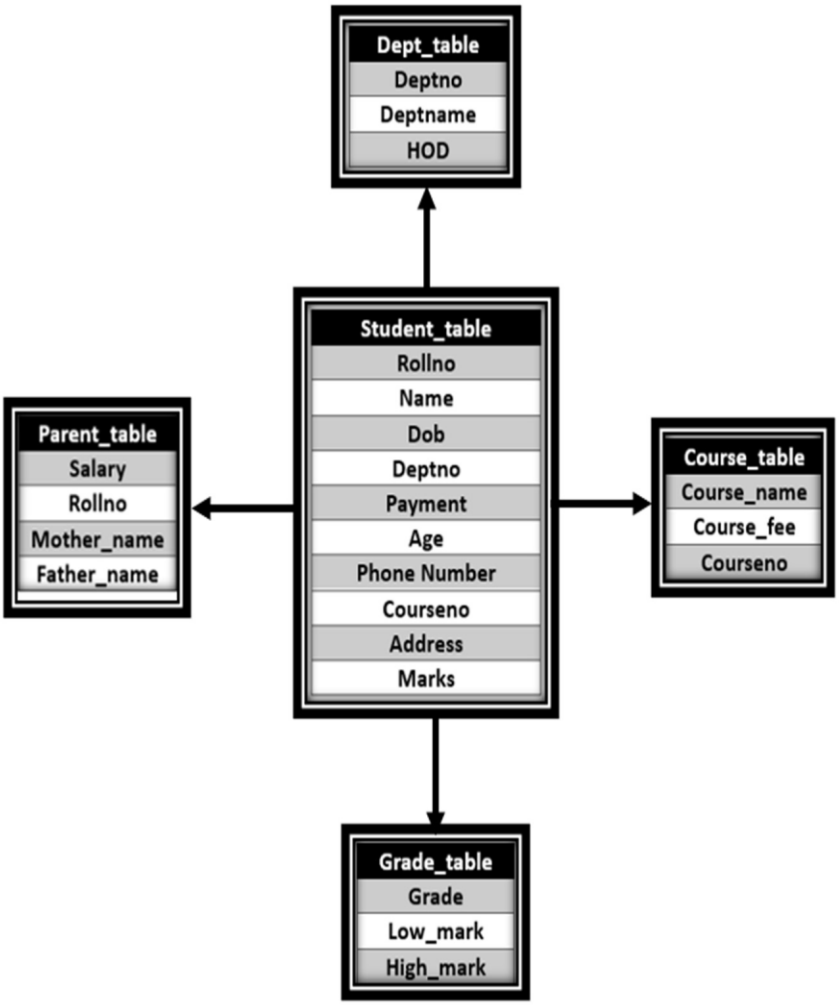
➔ **Second Normal Form:** A relation is in second normal form if it has No Partial Dependency, i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.
Ex: In our database, consider all attributes under the entity **Student.**
There exist no partial dependencies throughout them. Hence, in 2NF.

➔ **Third Normal Form:** A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form.
Ex: In our database, consider all attributes under the entity Student**.**
There exist no transitive dependencies throughout them. Also, theyare in 2NF and hence, are also in 3NF.

Apart from the entity cart, in the entire database, **there exist no attributes which exhibit any feature of redundancy.** Furthermore, each attribute satisfies all the above-mentioned Normal forms, thereby eliminating the need for any further Normalization. Thereby, we can move on to the actual representation of the Database schema.

Database Schema:

The following is an overview of the entire database schema, with individual table schemas defined.



Data Definition Language:

In total, there exist six tables in our database design. Each table and its associated DDL commands have been listed below:

1). Student Table:

CREATE TABLE STUDENT_TABLE(ROLLNO NUMBER(4) CONSTRAINT RID PRIMARY KEY,NAME VARCHAR2(20), DOB DATE, DEPTNO NUMBER(2), PAYMENT VARCHAR2(3), AGE NUMBER(2), PHONE_NUMBER1 NUMBER(10) CONSTRAINT PH1 NOT NULL. PHONE_NUMBER2 NUMBER(10), COURSENO NUMBER(4), ADDRESS VARCHAR2(25) DEFAULT 'HYDERABAD', MARKS NUMBER(3), CONSTRAINT POINTS CHECK(MARKS<=100);

DESC STUDENT_TABLE;

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
ROLLNO                                    NOT NULL NUMBER(4)
NAME                                               VARCHAR2(10)
DOB                                                DATE
DEPTNO                                             NUMBER(2)
PAYMENT                                            VARCHAR2(3)
AGE                                                NUMBER(2)
PHONE_NUMBER1                             NOT NULL NUMBER(10)
PHONE_NUMBER2                                      NUMBER(10)
COURSENO                                           NUMBER(4)
ADDRESS                                            VARCHAR2(25)
MARKS                                              NUMBER(3)
```

2). Grade Table:

CREATE TABLE GRADE_TABLE(GRADE VARCAHR2(1) CONSTRAINT GID PRIMARY KEY, LOW_MARK NUMBER(3), HIGH_MARK NUMBER(3));

DESC GARDE_TABLE;

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
GRADE                                     NOT NULL VARCHAR2(1)
LOW_MARK                                           NUMBER(3)
HIGH_MARK                                          NUMBER(3)
```

3). Department Table:

CREATE TABLE DEPT_TABLE(DEPTNO NUMBER(2) CONSTRAINT DNO PRIMARY KEY, DEPTNAME VARCHAR2(5), HOD VARCHAR2(10));

DESC DEPT_TABLE;

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
DEPTNO                                    NOT NULL NUMBER(2)
DEPTNAME                                           VARCHAR2(5)
HOD                                                VARCHAR2(10)
```

4). Parent Table:

CREATE TABLE PARENT_TABLE(SALARY NUMBER(6), ROLLNO NUMBER(4) CONSTRAINT ROLL PRIMARY KEY, MOTHER_NAME VARCHAR2(10), FATHER_NAME VARCHAR2(10));

DESC PARENT_TABLE;

```
SQL> desc parent_table;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SALARY                                             NUMBER(6)
 ROLLNO                                    NOT NULL NUMBER(4)
 MOTHER_NAME                                        VARCHAR2(10)
 FATHER_NAME                                        VARCHAR2(10)
```

5). Course Table:

CREATE TABLE COURSE_TABLE(COURSE_NAME VARCHAR2(6), COURSE_FEE NUMBER(6), COURSENO NUMBER(4) CONSTRAINT CNO PRIMARY KEY);

DESC COURSE_TABLE;

```
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
COURSE_NAME                                        VARCHAR2(6)
COURSE_FEE                                         NUMBER(6)
COURSENO                                  NOT NULL NUMBER(4)
```

Inserting Values:

Following are a few sample queries being inserted into each of the tables to build up a mock database model.

1). Student Table:

INSERT INTO STUDENT_TABLE VALUES(1, ‘ALEX’, ‘04-APR-02', 1, ‘YES’, 19, 8765432109, NULL, 55, ‘789 BROADWAY’, 90);

1 ROW INSERTED

--SIMILARLY 19 MORE ROWS HAS BEEN INSERTED.

2). Grade Table:

INSERT INTO GRADE_TABLE VALUES(F, 0, 50);

1 ROW INSERTED

--SIMILARLY 5 MORE ROWS HAS BEEN INSERTED.

3). DepartmentTable:

INSERT INTO DEPT_TABLE VALUES(1, ‘CSE’, JOHN’);

1 ROW INSERTED.

--SIMILARLY 4 MORE ROWS HAS BEEN INSERTED.

4). Parent Table:

INSERT INTO PARENT_TABLE VALUES(1000, 1, ‘MEERA’, ‘ARJUN’);

1 ROW INSERTED

-- SIMILARLY 19 MORE ROWS HAS BEEN INSERTED.

5). Course Table:

INSERT INTO COURSE_TABLE VALUES(‘CPP’, 2000, 51);

1 ROW INSERTED.

--SIMILARLY 4 MORE ROWS HAS BEEN INSERTED.

Sample Table Overviews:

1. Student Table

ROLL NO	NAME	DOB	DEPT NO	PAYMENT	AGE	PHONE NUMBER 1	PHONE NUMBER 2	COURSE NO	ADDRESS	MARKS
1	Alex	04-Apr-02	1	Yes	19	8765432109	NULL	55	789 Broadway	90
2	Sophia	12-DEC-00	3	No	20	2345678901	6789012345	52	345 Oak St	75
3	William	25-FEB-99	5	Yes	22	8901234567	NULL	54	234 Main St	85
4	Bob	19-SEP-93	4	No	26	4445556666	8889990000	54	Pune	75
5	Alice	05-FEB-95	3	Yes	23	1112223333	5556667777	53	Delhi	95
6	Ava	14-SEP-04	2	No	21	4567890123	5678901234	53	678 Elm St	65
7	James	30-MAR-00	4	Yes	20	6789012345	NULL	51	123 Pine St	95
8	Olivia	08-NOV-02	1	No	19	9012345678	7890123456	55	456 Maple St	80
9	Ethan	17-JUN-99	3	Yes	22	3456789012	NULL	52	Hyderabad	70
10	Emma	20-DEC-01	5	No	21	5678901234	4567890123	54	890 Elm St	60
11	Noah	05-JAN-00	2	Yes	20	1234567890	NULL	53	345 Maple St	85
12	Mason	05-DEC-05	5	Yes	20	6789012345	NULL	54	789 Oak St	95
13	Amelia	14-APR-02	2	No	19	8901234567	7890123456	53	234 Pine St	80
14	Logan	14-FEB-02	4	No	21	2345678901	NULL	51	345 Oak St	75
15	Evelyn	05-MAY-99	2	Yes	22	7890123456	6789012345	53	789 Pine St	95
16	Harper	17-JAN-00	3	Yes	20	5678901234	NULL	52	123 Maple St	85
17	Elijah	20-DEC-02	5	No	19	1234567890	4567890123	54	456 Elm St	80
18	Avery	18-OCT-01	1	No	21	3456789012	2345678901	55	890 Oak St	60
19	Liam	25-FEB-99	1	Yes	22	2345678901	NULL	55	123 Elm St	90
20	Raason	25-DEC-00	4	Yes	20	6789512345	NULL	54	750 Oak St	95

2. Grade Table

GRADE	LOW_MARK	HIGH_MARK
F	0	50
E	51	60
D	61	70
C	71	80
B	81	90
A	91	100

3. Department Table

DEPTNO	DEPTNAME	HOD
1	CSE	John
2	AIML	Mark
3	DS	Ram
4	CYS	Uma
5	IOT	Sita

4. Parent Table

SALARY	ROLLNO	MOTHER_NAME	FATHER_NAME
1000	1	Meera	Arjun
2000	2	Zoya	Ishan
3000	3	Chaitra	Moksh
4000	4	Harika	Nithin
5000	5	Ganga	Parin
6000	6	Olivia	Rishi
7000	7	Lakshmi	Swami
8000	8	Uma	Veer
9000	9	Shaila	Shankar
10000	10	Sitha	Ramesh
11000	11	Meera	Ishan
12000	12	Moksha	Anish
13000	13	Lalitha	Siva
14000	14	Shailu	Aakash
15000	15	Aravinda	Ashok
16000	16	Ramya	Rishi
17000	17	Bhavitha	Swamy
18000	18	Uma	Harish
19000	19	Rishika	Roopesh
20000	20	Devika	Dinesh

5. Course Table

COURSE	COURSE_FEE	COURSENO
CPP	2000	51
Python	1000	52
Java	1500	53
C	1200	54
DBMS	3000	55

Sample Queries:

Query 1: Delete student information where Roll Number is 20

delete from student_table where rollno=20;

Output:

```
SQL> delete from student_table where rollno=20;

1 row deleted.
```

Query 2: Display HOD Names in the ascending order

select hod from dept_table order by hod asc;

Output:

```
HOD
----------
John
Mark
Ram
Sita
Uma
```

Query 3:Display Average Salary of parents

select avg(salary) from parent_table;

Output:

```
AVG(SALARY)
-----------
      10500
```

Query 4: Display Average Marks of Students from each Department

select deptno,avg(marks) from student_table group by deptno;

Output:

```
   DEPTNO AVG(MARKS)
--------- ----------
        1         80
        2      81.25
        5         80
        4 81.6666667
        3      81.25
```

Query 5: Drop the Constraint Ph1 in Student_table

alter table student_table drop constraint ph1;

Output:

```
SQL> desc student_table;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
 ROLLNO                                    NOT NULL NUMBER(4)
 NAME                                               VARCHAR2(10)
 DOB                                                DATE
 DEPTNO                                             NUMBER(2)
 PAYMENT                                            VARCHAR2(3)
 AGE                                                NUMBER(2)
 PHONE_NUMBER1                                      NUMBER(10)
 PHONE_NUMBER2                                      NUMBER(10)
 COURSENO                                           NUMBER(4)
 ADDRESS                                            VARCHAR2(25)
 MARKS                                              NUMBER(3)
```

Query 6: Display all the Student Names who secured marks between 75 and 85

select name from student_table where marks between 75 and 85;

Output:

```
NAME
----------
Sophia
William
Bob
Olivia
Noah
Amelia
Logan
Harper
Elijah
```

Query 7: Display RollNo and Names of all students whose name ends with letter a

select rollno,name from student_table where name like '%a';

Output:

```
    ROLLNO NAME
---------- ----------
         2 Sophia
         6 Ava
         8 Olivia
        10 Emma
        13 Amelia
```

Query 8: Display RollNo of Students who belongs to DeptNo 4 or 5

select rollno from student_table where deptno=4 or deptno=5;

Output:

```
    ROLLNO
----------
         3
         4
         7
        10
        12
        14
        17
```

Query 9: insert course JS fee =2000, course number is 52

insert into course_table values('js',2000,52);

Output:

```
ERROR at line 1:
ORA-00001: unique constraint (22071A6622.CNO) violated
```

Query 10:Generate the output as EMMA-CourseNO is 54

select name||'-'||' courseno is '|| courseno from student_table where name='Emma';

Output:

```
NAME||'-'||'COURSENOIS'||COURSENO
--------------------------------
Emma- courseno is 54
```

Query 11:Find the Difference between highest and lowest age

select max(age)-min(age) as difference from student_table;

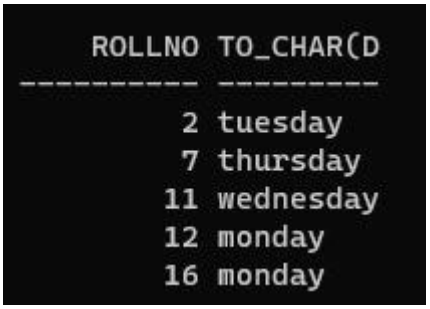
Output:

```
DIFFERENCE
----------
         7
```

Query 12:Write a query which will return the day of week Student born 20 Years ago

select rollno,to_char(dob,'day') from student_table where age=20;

Output:

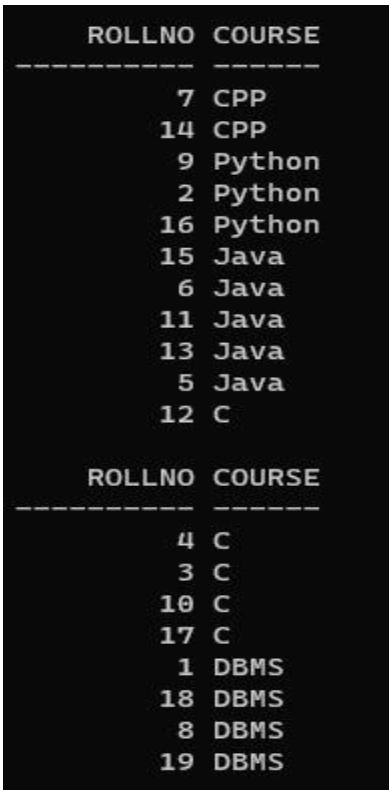


ROLLNO	TO_CHAR(D
2	tuesday
7	thursday
11	wednesday
12	monday
16	monday

Query 13: display course name opted by all students

select s.rollno,c.course_name from student_table s ,course_table c where s.courseno=c.courseno;

Output:



ROLLNO	COURSE
7	CPP
14	CPP
9	Python
2	Python
16	Python
15	Java
6	Java
11	Java
13	Java
5	Java
12	C

ROLLNO	COURSE
4	C
3	C
10	C
17	C
1	DBMS
18	DBMS
8	DBMS
19	DBMS

Query 14: Display the RollNo of Student whose parent information is present but not student information

select rollno from parent_table where rollno not in (select rollno from student_table);

Output:

ROLLNO
20

Query 15:Derive age of all students from DOB

SQL> select name,months_between(sysdate,dob) as age from student_table;

Output:

NAME	AGE
Alex	260.47143
Sophia	276.213365
William	297.79401
Bob	362.987559
Alice	346.439172
Ava	231.148849
James	284.63272
Olivia	253.342397
Ethan	294.052075
Emma	263.955301
Noah	287.439172

NAME	AGE
Mason	216.439172
Amelia	260.148849
Logan	262.148849
Evelyn	295.439172
Harper	287.052075
Elijah	251.955301
Avery	266
Liam	297.79401

Query 16:Display Student name and mother name using Joints

select student_table.name as student_name, parent_table.mother_name from student_table join parent_table on student_table.rollno = parent_table.rollno;

Output:

STUDENT_NA	MOTHER_NAM
Alex	Meera
Sophia	Zoya
William	Chaitra
Bob	Harika
Alice	Ganga
Ava	Olivia
James	Lakshmi
Olivia	Uma
Ethan	Shaila
Emma	Sitha
Noah	Meera

STUDENT_NA	MOTHER_NAM
Mason	Moksha
Amelia	Lalitha
Logan	Shailu
Evelyn	Aravinda
Harper	Ramya
Elijah	Bhavitha
Avery	Uma
Liam	Rishika

Query 17: Display student names whose parents earn more than parent of student whose roll no is 13

select student_table.name from student_table,parent_table where parent_table.salary>(select salary from parent_table where rollno=13) and student_table.rollno=parent_table.rollno;

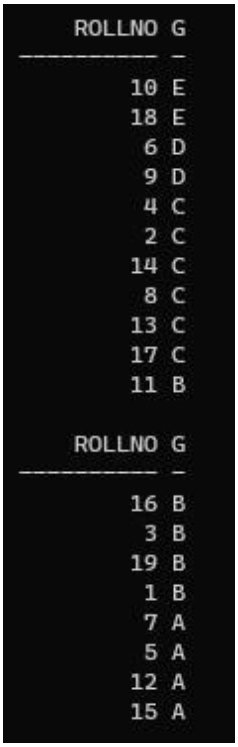
Output:

```
NAME
----------
Logan
Evelyn
Harper
Elijah
Avery
Liam
```

Query 18: Give grade of all students based on their marks

select rollno,grade from student_table,grade_table where marks between low_mark and high_mark;

Output:



```
    ROLLNO G
---------- -
        10 E
        18 E
         6 D
         9 D
         4 C
         2 C
        14 C
         8 C
        13 C
        17 C
        11 B

    ROLLNO G
---------- -
        16 B
         3 B
        19 B
         1 B
         7 A
         5 A
        12 A
        15 A
```

Conclusion:

In conclusion, the "Student Database Management System" project has successfully demonstrated how a database management system can be used to efficiently manage and handle complex data related to students. The system has been designed to be user-friendly and robust, ensuring data integrity and security.

The implementation of this system has streamlined the process of storing, retrieving, and managing student data, thereby reducing manual workload and increasing productivity. It has also provided a platform for easy access to student information, which is crucial for decision-making processes in an educational institution.

Furthermore, the system is scalable and can be extended to incorporate additional features as per the evolving needs of the institution. This adaptability makes it a sustainable solution for managing student data.

Overall, the "Student Database Management System" serves as a valuable tool for educational institutions, enhancing their data management capabilities and contributing to their digital transformation journey.