INTRODUCTION:

We are creating a database of University Management system like vtop . In that students and faculty and department head are able to use the database. To create more interaction between student and faculty this database is created. This database also included with teacher details to course that faculty teaches.

PROJECT CONTAINS:

1. Problem Statement

2. Module 1:

- ➔ Analysis
- ➔ Identify the following
- ➔ Entity (Strong /weak)
- ➔ Different types of attributes
- ➔ Relationship
- ➔ Cardinality
- ➔ Participation

3. Module 2:

- ➔ ER diagram of database.

4. Module 3:

- ➔ ER to Table Mapping.

5. Module 4:

➔ Normaliation 1NF,2NF,3NF on tables.

6. Module 5:

➔ Implementation

Create and insert
Alter, Delete and update
Primary key and foreign key constraint
Select with Where clause
Order by clause
Like clause
Is null/ is not null
Any five Aggregate functions
Any five date functions
Any three numeric functions
Any five String Functions
Group by and having
Join more than two tables

7. Module 6:

➔ Query Optimization
➔ Query tree and query graph

8. Module 7:

➔ Indexing
➔ index table.

9: Module 8:

➔ Implement B tree and hash index on the relations you have designed.

10. Conclusion .

==========

1. Problem Statement :

This University database handle safely they store the data of students information which includes student_ID which represents the students uniquely, E-mail, ph_number, Name, F.name, L.name . The university has different departments in which Each department has department_name, department_ID , HOD , contact_no, E-mail . The students studying in the university need pay the fee to the department.

Students enroll the courses each course have course ID ,course name ,credits , type of course like [ETH,ELA,EPJ]. Course_ID is used to uniquely identify the course . Each course is handled by the department. Each department must have atleast one course . Department conduct exams and in each exam Name , number of students attended, number of students absent, names of students in malpractice , date of exam need to be stored.

Faculties teaches the students. Each faculty has Name, F.name L.name , Teacher_ID , phone number , E-mail , Experience these to be stored . A Teacher in university must teach a course .

Students buys the books in library we need to keep track the details of book name , price ,book author, date of book buyed , Book_ID which uniquely represent the Books .

A student can buy many books as they wish there is a chance some student did not buy any books his student belongs to the department.

Every student has Mentor to solve the problems of the students. Name , F.name , L.name , E-mail , Mentor_ID , ph_number. A mentor can handle many students .

E_mail, PH_no mentioned above must be university given mail and single attribute.

2. Analysis:

i) Entity (Strong /weak)

a) Mentor
b) Student
c) Books
d) Course
e) Teacher
f) Department
g) Exam (weak entity)

ii) Different type of attributes:

- Mentor : [ID, Name, Email, Phone number]
  Name is a composite attribute as First_name,Last_name.

- Books : [Author , Name, Price, Book id]

- Student : [Phone no, Course_ids, Email, Student id, Name]
  In this course_ids is the multivalued attribute.
  Name is a composite attribute as First_name, Last_name.

- Department : [HOD, D_ID, D_NAME, Contact, Email]

- Exam : [Absent, No. of students, Dot, Malpractice]

- Course : [Name, Course_id, Credits, Type]

- Teacher : [Experience, Teacher_ID, Phone no, Email, Name]
  Name is a composite attribute as First_name,Last_name.

iii) Relationship, Participation, Cardinality :

> Mentor , student Relationship

The relationship between mentor and student is the mentor is appointed to guide the students through the academic year. Every mentor has more than one students.

Total participation on both sides Because Mentor cannot exist without guiding a student vice versa.

This is 1:M Relation a Mentor can guide many students and a student can only have one mentor.

> Student buys books.

Students buys the books every student need not buy a book.

partial participation on both sides. Because student has no restriction that buy a book..

This is M:N Relation a book can be bought by many students and vice versa.

> Student pay fee to department.

Students are paying the fee to the department. This is the relation between the student and department.

Total participation of sudent and a department is partial participation. Because student cannot exist without paying fee.

This is M:1 Relation a department can have many students but the vice versa not possible.

> Department conduct exam.

The department conducts the exam to the students this information is also stored.

Total participation of Exam and a Department is partial participation. Because Exam cannot exist without Department.

This is 1:M Relation a Department can conduct many exams and a vice versa not possible.

> Teacher teaches to student.

The relation between teacher and student is teacher teaches the students.

Total participation of Student and a Teacher is partial participation. Because Student cannot exist without taught by the teacher.

This is M:N Relation a teacher can teach to many students and a student can be taught by many teachers.

> Student takes courses.

The relation between student and courses is student takes the courses.

Total participation of Student and a course is partial participation. Because Student cannot exist without taking a course.

This is M:N Relation a student can take many courses and a course can be took by many students.

> Department handles courses.

The relation between Department and courses is Department handles the courses.

Total participation on both sides because a department cannot exist without a course and vice versa.

This is 1:M Relation a DEPARTMENT can have many courses and a course can be handled by many Department.

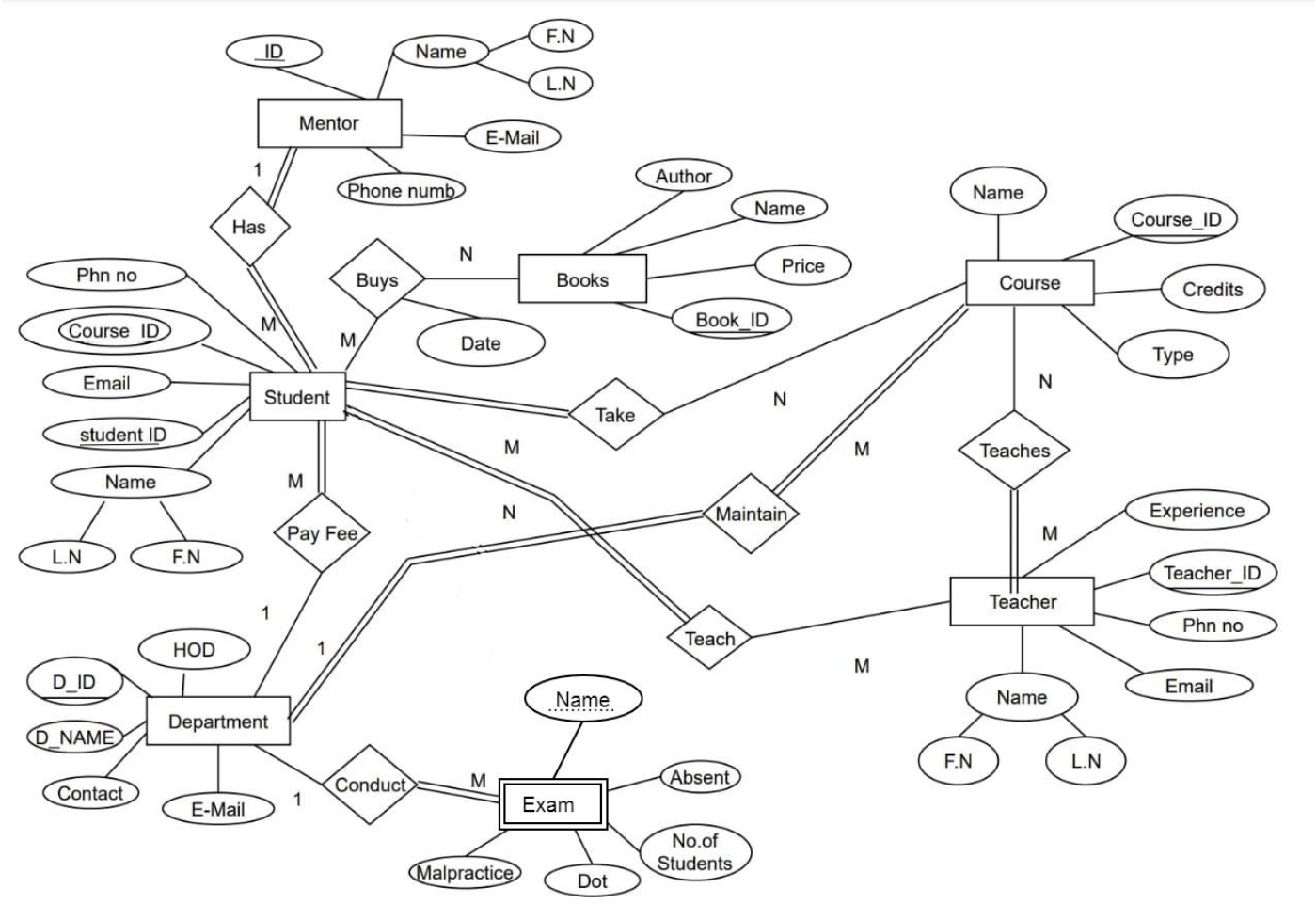
> Teacher teaches a course.

The relation between Teacher and courses is Teacher teaches the courses.

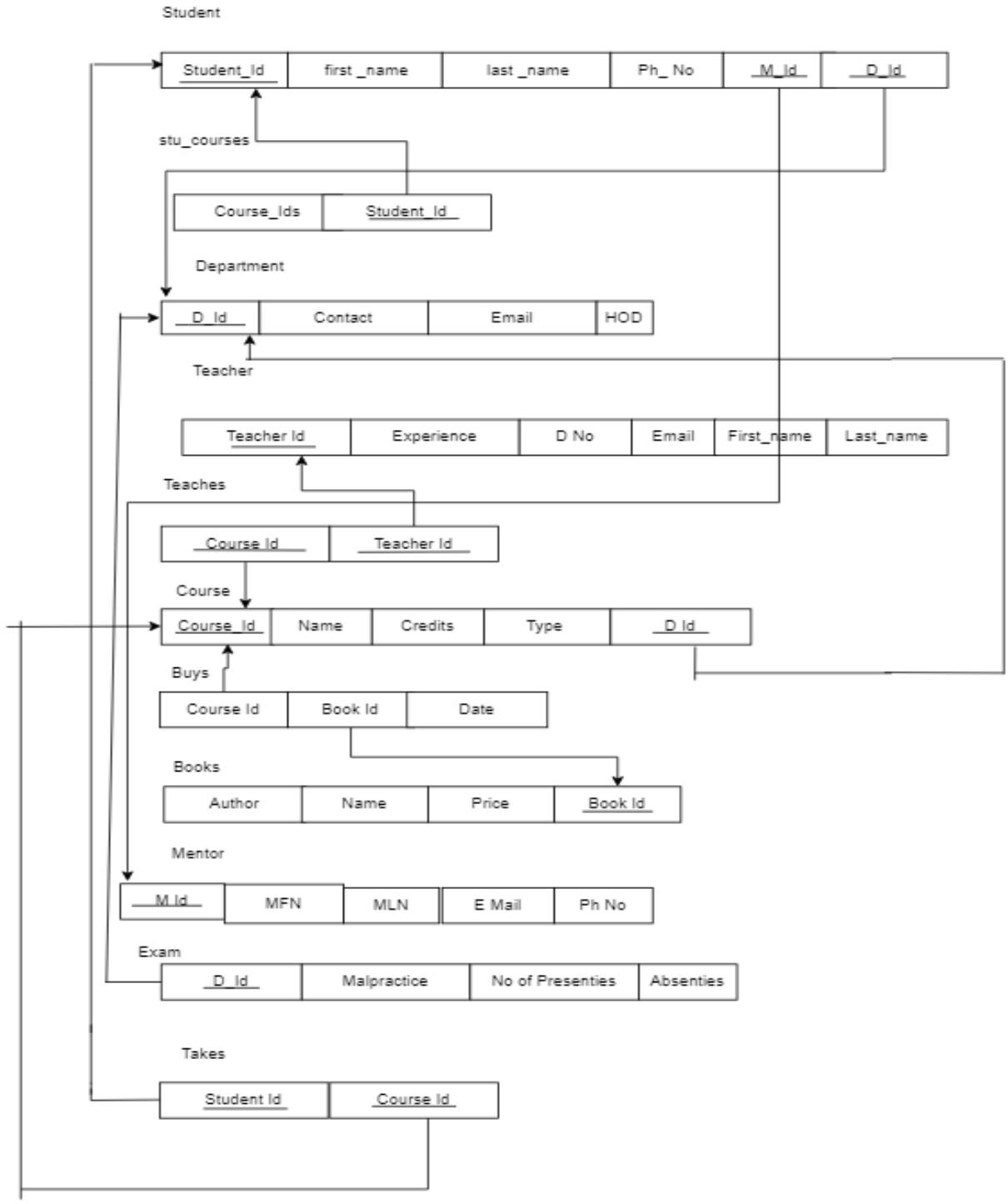
Total participation of Teacher and a course is partial participation. Because Teacher cannot exist without teaching a course.

This is M:N Relation a teacher can teach many courses and a course can be taught by many students.

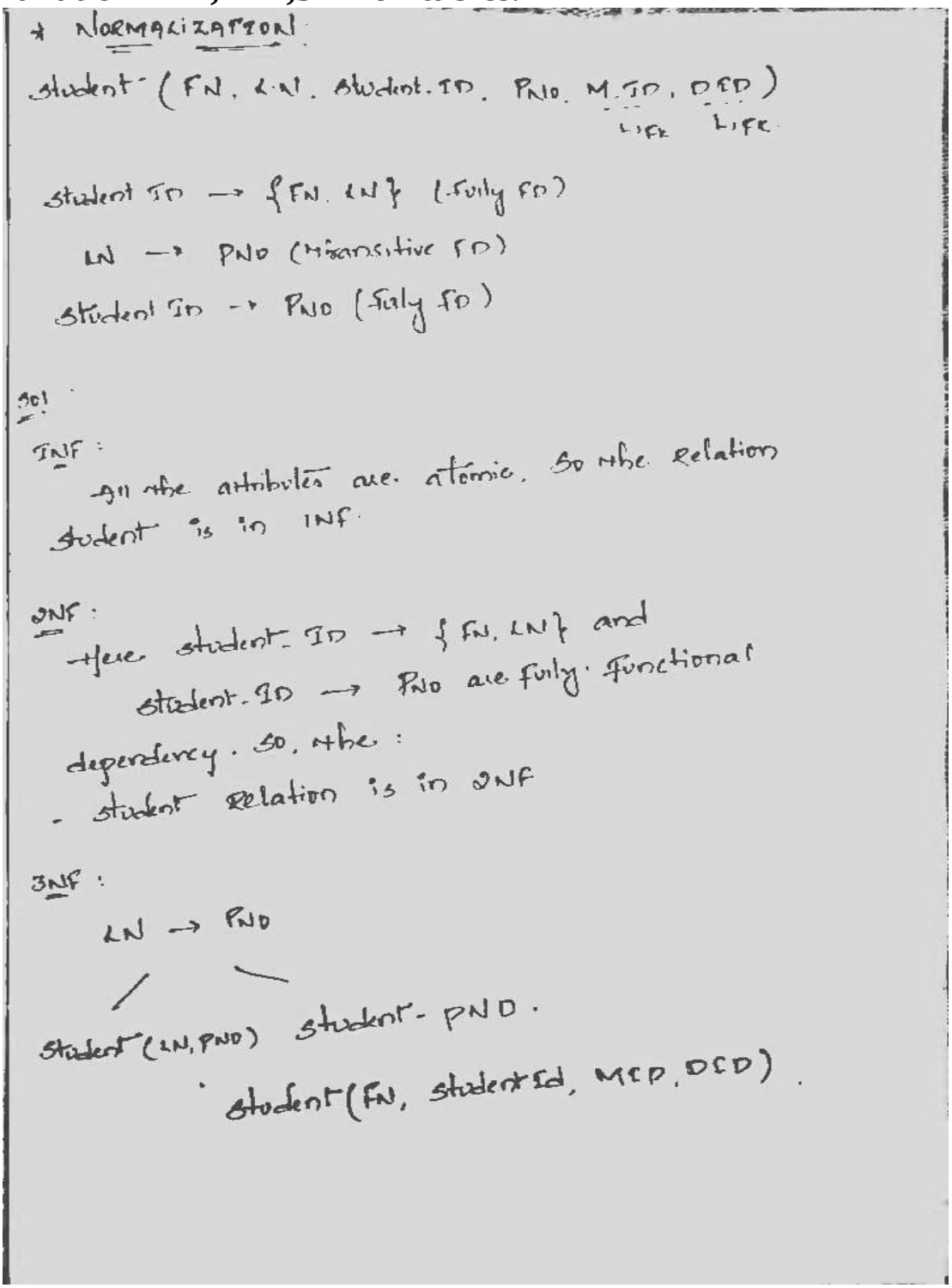
3. ER Diagram:



4. ER to Table Mapping


Student
Student_Id
first _name
last _name
Ph_ No
M_Id
D_Id
stu_courses
Course_Ids
Student_Id
Department
D_Id
Contact
Email
HOD
Teacher
Teacher Id
Experience
D No
Email
First_name
Last_name
Teaches
Course Id
Teacher Id
Course
Course_Id
Name
Credits
Type
D Id
Buys
Course Id
Book Id
Date
Books
Author
Name
Price
Book Id
Mentor
M Id
MFN
MLN
E Mail
Ph No
Exam
D_Id
Malpractice
No of Presenties
Absenties
Takes
Student Id
Course Id

4. Normaliation 1NF,2NF,3NF on tables.



* NORMALIZATION:

Student (FN, LN, Student.ID, PNO, MID, DID)
LIFE LIFE

Student ID → {FN, LN} (Fully FD)

LN → PNO (Transitive FD)

Student ID → PNO (Fully FD)

So!

1NF:

All the attributes are atomic. So the Relation Student is in 1NF.

2NF:

Here student_ID → {FN, LN} and

Student.ID → PNO are fully functional dependency. So, the:

- student Relation is in 2NF

3NF:

LN → PNO

/ \

Student(LN, PNO) student-PNO.

Student(FN, StudentId, MID, DID).

Teacher (Experience, Teacher ID, DNo, E-mail, TFN, TLN)

Experince → {TFN, TLN}
Teacher ID → Email
Teacher ID → DNO

1NF :-

Here Their is NO Multivalid Attribute All are Atomic - Hence it is in the 1NF.

2NF :-

Teacher ID → EMail is a Fully Functinal Dependency and Their is no Partial Functinal dependency. So, the Teacher relation is in 2NF.

3NF :-

Experience → {TFN, TLN}

Now it is the Transitive FD. So, Now we want to Divide the Relation into Two parts.

Experience → {TFN, TLN}

Teacher (Experience, TFN, TLN)

Teacher (Teacher ID, DNO, Email)

Department(HOD, Contact, email, DID)

~~HOD → DID~~

DID → HOD

HOD → {Contact, email}

1NF :-

Here All the Attributes are in 1NF. Atomic Attributes No Multivalied Attributes So, the Department Relation is in 1st NORMAL FORM.

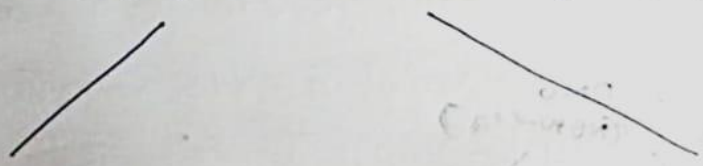
2NF :-

DID → HOD is Fully FD and

HOD → { Contact, email } is Transitive FD, So, Here their is no partial functinal Dependency. So, it is in 2NF.

3NF :-

HOD → { Contact, email } is TFD.



Department (HOD, Contact , email) Department (DID

Mentor (MID, MFN, MLN, E-mail, DNO)

1NF :- MID → {MFN, MLN}
Email → DNO

Here their is NO MVA. All Are Atomic Attributs Hence they are in 1NF.

2NF :-

Here In the FD's Their is NO partial FD's. Hence the "Mentor" Relation is in 2NF.

3NF :-

Here Transitive FD's Email → Phno

Hence Divide the Table into two parts.

Email → PNO

Mentor1 (Email, PNO)

Mentor2 (MID, MFN, MLN)

Books (Author, Name, price, BookID)

1NF:- FD:- Authore → Name
Name → price
BookID → Name

Here All the Attributes in Books are in Atomic. Their is no MVA. Hence the Relation is in 1NF.

2NF :-

Here Their is NO partial Functinal Dependency. Hence the Books Relation is in 2NF.

3NF :-

Here

Author → Name and Name → Price

are in Transitive Functinal Dependency

Author → Name and Name → Price

Books 1 (Author, Name)

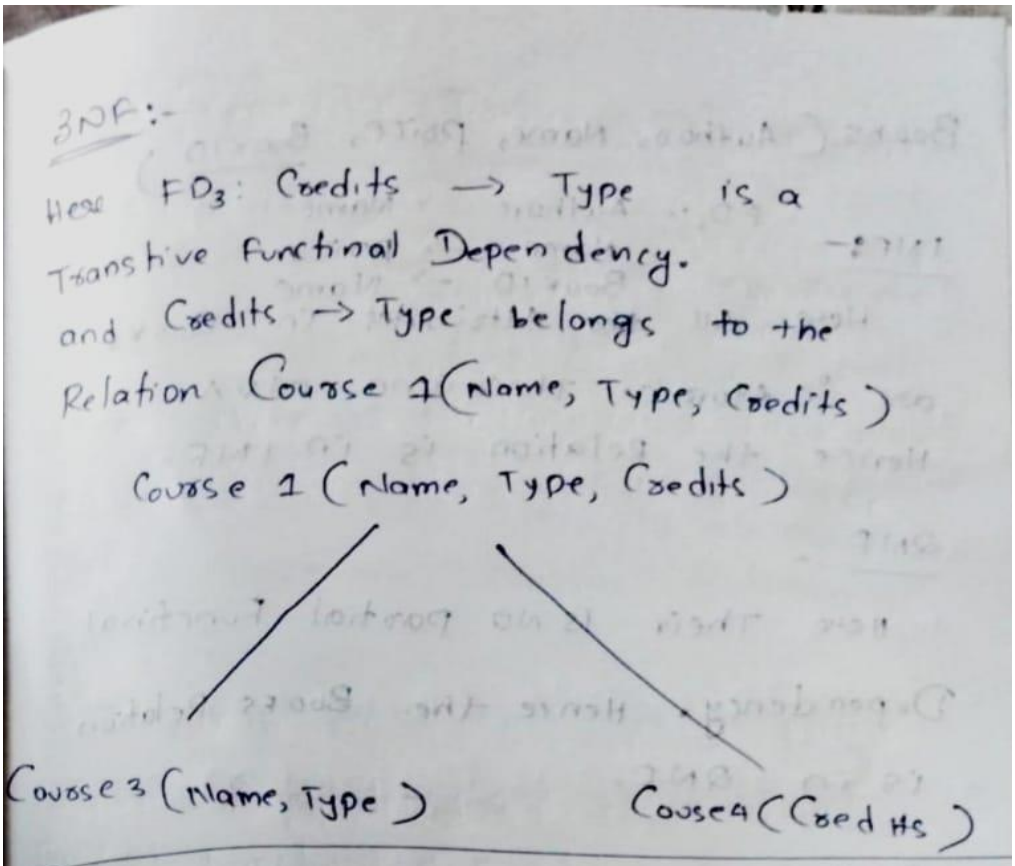
Books 2 (Name, Price)

Books 3 (BookID)

3NF:-

Here FD_3: Credits → Type is a Transitive Functinal Dependency.

and Credits → Type belongs to the Relation Course 1(Name, Type, Credits)



Course (Name → PK, CourseID → PK, Credits, Type, DID → FK)

Here {Name, Course ID} is PK.

② FD's

FD_1: Name → {Type, Credits}

FD_2: {CourseID, Name} → Credits

FD_3: Credits → Type

1NF :-

All the Attributes are in Relation "Course" are Atomic. So the Course is in 1NF

FD_1 ~~and~~ FD_2 are partial/functional dependency They are not in 2NF.

2NF :-

Here FD_1 is a partial FD.

So, Divide the Relation in two parts

Name → {Type, Credits}

Course 1 (Name, Type, Credits) Course 2 (CourseID, DID)

Takes(Student ID, Course ID)

Student ID → Course ID

1NF :-

Here their is no multivalied Attribute All are Composite Hence it is in 1NF.

2NF :-

Here their is no partial FD, because both are key attributes Hence they are in 2NF.

3NF :-

Here Their is no Transitive FD Hence It is in 3NF.

Buys (CourseID , BookID , Date)

CourseID → Date

1NF :- BookID → Date

Here All the Attributes are Atomic and no multivalied Hence it is in first normal form.

2NF :-

Here

✓ CourseID → Date (Partial FD)

✓ BookID → Date (Partial FD)

CourseID → Date , BookID → Date

Buy1(CourseID, Date)

Buy2(BookID, Date)

Buys3 ( );

3NF :-

Here there is NO Transitive FD Hence it is 3NF.

6. Implementation:

Create and insert

STUDENT:

```
CREATE TABLE STUDENT (
STUDENT_ID VARCHAR2(10) NOT NULL,
PH_NO NUMBER,
FIRST_NAME VARCHAR2(15),
LAST_NAME VARCHAR2(15),
E_MAIL VARCHAR2(15),
PRIMARY KEY(STUDENT_ID)
)
```

- ```
  INSERT INTO STUDENT (STUDENT_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('20BIT0001',9870541230,'SAI','PRANEETH','20BIT0001@GMAIL.COM')
  ```
- ```
  INSERT INTO STUDENT (STUDENT_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('20BIT0002',9806541230,'RAM','DEERAJ','20BIT0002@GMAIL.COM')
  ```
- ```
  INSERT INTO STUDENT (STUDENT_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('20BIT0003',9870541230,'SIN','ARI','20BIT0003@GMAIL.COM')
  ```
- ```
  INSERT INTO STUDENT (STUDENT_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('20BIT0004',9876441230,'DIVS','DIYA','20BIT0004@GMAIL.COM')
  ```
- ```
  INSERT INTO STUDENT (STUDENT_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('20BIT0005',9872541230,'HEY','RAMYA','20BIT0005@GMAIL.COM')
  ```
- ```
  INSERT INTO STUDENT (STUDENT_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('20BIT0006',9376541230,'NANI','PAVAN','20BIT0006@GMAIL.COM')
  ```
- ```
  INSERT INTO STUDENT (STUDENT_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('20BIT0007',9856541230,'KAVYA','GOKUL','20BIT0007@GMAIL.COM')
  ```
  ```

STUDENT_ID	PH_NO	FIRST_NAME	LAST_NAME	E_MAIL	M_ID	D_ID
20BIT0001	9870541230	SAI	PRANEETH	20BIT0001@GMAIL.COM	-	-
20BIT0002	9806541230	RAM	DEERAJ	20BIT0002@GMAIL.COM	-	-
20BIT0003	9870541230	SIN	ARI	20BIT0003@GMAIL.COM	-	-
20BIT0004	9876441230	DIVS	DIYA	20BIT0004@GMAIL.COM	-	-
20BIT0005	9872541230	HEY	RAMYA	20BIT0005@GMAIL.COM	-	-
20BIT0006	9376541230	King	PAVAN	20BIT0006@GMAIL.COM	-	-
20BIT0007	9856541230	KAVYA	GOKUL	20BIT0007@GMAIL.COM	-	-

Download CSV

7 rows selected.

COURSE:

```
CREATE TABLE COURSE (
COURSE_ID VARCHAR2(10) NOT NULL,
CREDITS NUMBER,
NAME VARCHAR2(15),
TYPE VARCHAR2(15),
PRIMARY KEY(COURSE_ID)
)
```

```
INSERT INTO COURSE (COURSE_ID,CREDITS,NAME,TYPE) values ('ITE1004',4,'DSA','ETH,ELA')
INSERT INTO COURSE (COURSE_ID,CREDITS,NAME,TYPE) values ('ITE1003',3,'DBMS','ETH')
INSERT INTO COURSE (COURSE_ID,CREDITS,NAME,TYPE) values ('ITE1001',4,'DLM','ETH,ELA')
INSERT INTO COURSE (COURSE_ID,CREDITS,NAME,TYPE) values ('ITE1014',3,'HCI','ETH,EPJ')
INSERT INTO COURSE (COURSE_ID,CREDITS,NAME,TYPE) values ('ITE1005',2,'SOFTWARE','ETH')
INSERT INTO COURSE (COURSE_ID,CREDITS,NAME,TYPE) values ('CSE1002',3,'OOPS','ELA')
INSERT INTO COURSE (COURSE_ID,CREDITS,NAME,TYPE) values ('CSE1014',4,'JAVA','ETH,ELA')
```

COURSE_ID	CREDITS	NAME	TYPE	D_ID
ITE1004	4	DSA	ETH,ELA	-
ITE1003	3	DBMS	ETH	-
ITE1001	4	DLM	ETH,ELA	-
ITE1014	3	HCI	ETH,EPJ	-
ITE1005	2	SOFTWARE	ETH	-
CSE1002	3	OOPS	ELA	-
CSE1014	4	JAVA	ETH,ELA	-

Download CSV

7 rows selected.

TEACHER:

```
CREATE TABLE TEACHER (
TEACHER_ID VARCHAR2(10) NOT NULL,
PH_NO NUMBER,
FIRST_NAME VARCHAR2(15),
LAST_NAME VARCHAR2(15),
E_MAIL VARCHAR2(15),
EXPERIENCE NUMBER,
PRIMARY KEY(TEACHER_ID)
)
```

- ```
  INSERT INTO TEACHER (TEACHER_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL,EXPERIENCE)
  VALUES ('TEACH1',9087654321,'DR.SUDHEER ','REDDY ','TEACH1@vit.ac.in ',20)
  ```
- ```
  INSERT INTO TEACHER (TEACHER_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL,EXPERIENCE)
  VALUES ('TEACH2',8567654321,'DR.RAMA','SUNDHAR ','TEACH2@vit.ac.in',15)
  ```
- ```
  INSERT INTO TEACHER (TEACHER_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL,EXPERIENCE)
  VALUES ('TEACH3',8967643210,'DR.RAVI','KUMAR ','TEACH3@vit.ac.in ',10)
  ```
- ```
  INSERT INTO TEACHER (TEACHER_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL,EXPERIENCE)
  VALUES ('TEACH4',7967643210,'DR.SRINIVAS','REDDY ','TEACH4@vit.ac.in ',30)
  ```
- ```
  INSERT INTO TEACHER (TEACHER_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL,EXPERIENCE)
  VALUES ('TEACH5',8867643210,'DR.RASHEED','REDDHY ',' TEACH5@vit.ac.in',5)
  ```
  ```

- INSERT INTO TEACHER (TEACHER_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL,EXPERIENCE)

VALUES ('TEACH6',8888643210,'DR.SURAJ',' REDDY',' TEACH6@vit.ac.in',23)
- INSERT INTO TEACHER (TEACHER_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL,EXPERIENCE)

VALUES ('TEACH7',8123643210,'DR.SAI','REDDY ',' TEACH7@vit.ac.in',1)

TEACHER_ID	PH_NO	FIRST_NAME	LAST_NAME	E_MAIL	EXPERIENCE
TEACH1	9087654321	DR.SUDHEER	REDDY	TEACH1@vit.ac.in	20
TEACH2	8567654321	DR.RAMA	SUNDHAR	TEACH2@vit.ac.in	15
TEACH3	8967643210	DR.RAVI	KUMAR	TEACH3@vit.ac.in	10
TEACH4	7967643210	DR.SRINIVAS	REDDY	TEACH4@vit.ac.in	30
TEACH5	8867643210	DR.RASHEED	REDDHY	TEACH5@vit.ac.in	5
TEACH6	8888643210	DR.SURAJ	REDDY	TEACH6@vit.ac.in	23
TEACH7	8123643210	DR.SAI	REDDY	TEACH7@vit.ac.in	1

Download CSV

7 rows selected.

BOOKS:

```
CREATE TABLE BOOKS (
BOOK_ID VARCHAR2(10) NOT NULL,
PRICE NUMBER,
NAME VARCHAR2(15),
AUTHOR VARCHAR2(15),
PRIMARY KEY(BOOK_ID)
)
```

```
INSERT INTO BOOKS (BOOK_ID,PRICE,NAME,AUTHOR) values ('1',1500,'WEB_TECH','RAVI')
INSERT INTO BOOKS (BOOK_ID,PRICE,NAME,AUTHOR) values ('2',1300,'DBMS','SAI')
INSERT INTO BOOKS (BOOK_ID,PRICE,NAME,AUTHOR) values ('3',1400,'SOCIAL','DEERAJ')
INSERT INTO BOOKS (BOOK_ID,PRICE,NAME,AUTHOR) values ('4',1250,'OOPS','RAMYA')
INSERT INTO BOOKS (BOOK_ID,PRICE,NAME,AUTHOR) values ('5',1340,'HCI','JASHWANTH')
INSERT INTO BOOKS (BOOK_ID,PRICE,NAME,AUTHOR) values ('6',900,'AOD','GOKUL')
```

```
INSERT INTO BOOKS (BOOK_ID,PRICE,NAME,AUTHOR) values ('7',890,'STATS','DIVYA')
```

BOOK_ID	PRICE	NAME	AUTHOR
1	1500	WEB_TECH	RAVI
2	1300	DBMS	SAI
3	1400	SOCIAL	DEERAJ
4	1250	OOPS	RAMYA
5	1340	HCI	JASHWANTH
6	900	King	GOKUL
7	890	STATS	DIVYA

Download CSV

7 rows selected.

MENTOR:

```
CREATE TABLE MENTOR (
MENTOR_ID VARCHAR2(10) NOT NULL,
PH_NO NUMBER,
FIRST_NAME VARCHAR2(15),
LAST_NAME VARCHAR2(15),
E_MAIL VARCHAR2(15),
PRIMARY KEY(MENTOR_ID)
)

INSERT INTO MENTOR (MENTOR_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
VALUES ('MEN1',9996665789,'DR.Surendhar','reddy','MEN1@vit.ac.in')
```

- ```
  INSERT INTO MENTOR (MENTOR_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('MEN2',7806665789,'DR.chandrababu','naidu','MEN2@vit.ac.in')
  ```

- ```
  INSERT INTO MENTOR (MENTOR_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('MEN3',7006665700,'Dheeran','guptha','MEN3@vit.ac.in')
  ```

- INSERT INTO MENTOR (MENTOR_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('MEN4',9996665700,'DR.SRIRAM','CHANDRA','MEN4@vit.ac.in')

- INSERT INTO MENTOR (MENTOR_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('MEN5',9008765700,'DR.Sheker','Reddy','MEN5@vit.ac.in')

- INSERT INTO MENTOR (MENTOR_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('MEN6',9118765700,'Dr.routhu','praneeth','MEN6@vit.ac.in')

- INSERT INTO MENTOR (MENTOR_ID,PH_NO,FIRST_NAME,LAST_NAME,E_MAIL)
  VALUES ('MEN7',8118765700,'DR.rama','devi','MEN7@vit.ac.in')

MENTOR_ID	PH_NO	FIRST_NAME	LAST_NAME	E_MAIL
MEN1	9996665789	DR.Surendhar	reddy	MEN1@vit.ac.in
MEN2	7806665789	DR.chandrababu	naidu	MEN2@vit.ac.in
MEN3	7006665700	Dheeran	guptha	MEN3@vit.ac.in
MEN4	9996665700	DR.SRIRAM	CHANDRA	MEN4@vit.ac.in
MEN5	9008765700	DR.Sheker	Reddy	MEN5@vit.ac.in
MEN6	9118765700	BIMAL	praneeth	MEN6@vit.ac.in
MEN7	8118765700	DR.rama	devi	MEN7@vit.ac.in

Download CSV

7 rows selected.

DEPARTMENT:

```
CREATE TABLE DEPARTMENT (
DEPARTMENT_ID VARCHAR2(10) NOT NULL,
FIRST_NAME VARCHAR2(15),
LAST_NAME VARCHAR2(15),
PH_NO NUMBER,
```

```
HOD VARCHAR2(15),
E_MAIL VARCHAR(25),
PRIMARY KEY(DEPARTMENT_ID)
)
```

- INSERT INTO DEPARTMENT (DEPARTMENT_ID,D_NAME,PH_NO,HOD,E_MAIL)
  VALUES ('BIT1','IT',9876504321,'VISWANATHAN','BIT1@GMAIL.COM')
- INSERT INTO DEPARTMENT (DEPARTMENT_ID,D_NAME,PH_NO,HOD,E_MAIL)
  VALUES ('CSE1','CSE',8706543219,'RAM','CSE1@GMAIL.COM')
- INSERT INTO DEPARTMENT (DEPARTMENT_ID,D_NAME,PH_NO,HOD,E_MAIL)
  VALUES ('BCE1','CSE_SPEC',8706512349,'PARIMALA','BCE1@GMAIL.COM')
- INSERT INTO DEPARTMENT (DEPARTMENT_ID,D_NAME,PH_NO,HOD,E_MAIL)
  VALUES ('EEE1','ELECTRONICS',9102345678,'VIJAYAN','EEE1@GMAIL.COM')
- INSERT INTO DEPARTMENT (DEPARTMENT_ID,D_NAME,PH_NO,HOD,E_MAIL)
  VALUES ('ECE1','ECE',8976504321,'BANSAL','ECE1@GMAIL.COM')
- INSERT INTO DEPARTMENT (DEPARTMENT_ID,D_NAME,PH_NO,HOD,E_MAIL)
  VALUES ('AER1','AERONATICAL',7896504321,'KERITH','AER1@GMAIL.COM')
- INSERT INTO DEPARTMENT (DEPARTMENT_ID,D_NAME,PH_NO,HOD,E_MAIL)
  VALUES ('MIS1','M_TECH',9871234560,'CHERITH','MIS1@GMAIL.COM')

DEPARTMENT_ID	D_NAME	PH_NO	HOD	E_MAIL
BIT1	IT	9876504321	VISWANATHAN	BIT1@GMAIL.COM
CSE1	CSE	8706543219	RAM	CSE1@GMAIL.COM
BCE1	CSE_SPEC	8706512349	PARIMALA	BCE1@GMAIL.COM
EEE1	ELECTRONICS	9102345678	VIJAYAN	EEE1@GMAIL.COM
ECE1	ECE	8976504321	BANSAL	ECE1@GMAIL.COM
AER1	AERONATICAL	7896504321	KERITH	AER1@GMAIL.COM
MIS1	M_TECH	9871234560	CHERITH	MIS1@GMAIL.COM

Download CSV

7 rows selected.

EXAM:

```
CREATE TABLE EXAM (
EXAM_ID VARCHAR2(15) NOT NULL,
NO_OF_STU NUMBER,
DOE VARCHAR2(15),
NO_OF_ABS NUMBER,
MALPRACTICE NUMBER,
PRIMARY KEY(EXAM_ID)
)
```

- ```
  INSERT INTO EXAM (EXAM_ID,NO_OF_STU,DOE,NO_OF_ABS,MALPRACTICE)
  VALUES ('EX101',40,'2021-01-11',2,0)
  ```
- ```
  INSERT INTO EXAM (EXAM_ID,NO_OF_STU,DOE,NO_OF_ABS,MALPRACTICE)
  VALUES ('EX102',50,'2021-05-01',0,0)
  ```
- ```
  INSERT INTO EXAM (EXAM_ID,NO_OF_STU,DOE,NO_OF_ABS,MALPRACTICE)
  VALUES ('EX103',60,'2021-04-01',1,2)
  ```
- ```
  INSERT INTO EXAM (EXAM_ID,NO_OF_STU,DOE,NO_OF_ABS,MALPRACTICE)
  VALUES ('EX104',40,'2021-03-01',3,0)
  ```
- ```
  INSERT INTO EXAM (EXAM_ID,NO_OF_STU,DOE,NO_OF_ABS,MALPRACTICE)
  VALUES ('EX105',70,'2021-02-01',4,1)
  ```
- ```
  INSERT INTO EXAM (EXAM_ID,NO_OF_STU,DOE,NO_OF_ABS,MALPRACTICE)
  VALUES ('EX106',30,'2021-11-01',2,0)
  ```
- ```
  INSERT INTO EXAM (EXAM_ID,NO_OF_STU,DOE,NO_OF_ABS,MALPRACTICE)
  VALUES ('EX107',45,'2021-01-01',1,0)
  ```
  ```

EXAM_ID	NO_OF_STU	NO_OF_ABS	MALPRACTICE	D_ID	DOE
EX101	40	2	0	-	-
EX102	50	0	0	-	-
EX103	60	1	2	-	-
EX104	40	3	0	-	-
EX105	70	4	1	-	-
EX106	30	5	0	-	-
EX107	45	1	0	-	-

Download CSV

7 rows selected.

TEACHES:

```
CREATE TABLE TEACHES (
COURSE_ID VARCHAR2(10),
TEACHER_ID VARCHAR2(10),
constraint FK_TEACHES_COURSE_ID foreign key (COURSE_ID) references
COURSE(COURSE_ID),
constraint FK_TEACHES_TEACHES_ID foreign key (TEACHER_ID) references
TEACHER(TEACHER_ID),
CONSTRAINT PK_TEACHES PRIMARY KEY(COURSE_ID,TEACHER_ID)
)
```

```
INSERT INTO TEACHES(COURSE_ID,TEACHER_ID) VALUES ('ITE1001','TEACH1')
INSERT INTO TEACHES(COURSE_ID,TEACHER_ID) VALUES ('ITE1003','TEACH2')
INSERT INTO TEACHES(COURSE_ID,TEACHER_ID) VALUES ('ITE1004','TEACH3')
INSERT INTO TEACHES(COURSE_ID,TEACHER_ID) VALUES ('ITE1005','TEACH4')
INSERT INTO TEACHES(COURSE_ID,TEACHER_ID) VALUES ('CSE1002','TEACH5')
INSERT INTO TEACHES(COURSE_ID,TEACHER_ID) VALUES ('CSE1014','TEACH6')
INSERT INTO TEACHES(COURSE_ID,TEACHER_ID) VALUES ('ITE1014','TEACH7')
```

COURSE_ID	TEACHER_ID
CSE1002	TEACH5
CSE1014	TEACH6
ITE1001	TEACH1
ITE1003	TEACH2
ITE1004	TEACH3
ITE1005	TEACH4

Download CSV
6 rows selected.

BUYS:

```
CREATE TABLE BUYS (
COURSE_ID VARCHAR2(10),
BOOK_ID VARCHAR2(10),
DATE_BUY DATE,
constraint FK_BUYS_COURSE_ID foreign key (STUDENT_ID) references
COURSE(STUDENT_ID),
constraint FK_BUYS_BOOK_ID foreign key (BOOK_ID) references
BOOKS(BOOK_ID),
CONSTRAINT PK_BUYS PRIMARY KEY(COURSE_ID,BOOK_ID)
)
```

```
INSERT INTO BUYS (STUDENT_ID,BOOK_ID,DATE_BUY) VALUES ('20BIT0001','1','01-JAN-21')
INSERT INTO BUYS (STUDENT_ID,BOOK_ID,DATE_BUY) VALUES ('20BIT0002','2','11-FEB-21')
INSERT INTO BUYS (STUDENT_ID,BOOK_ID,DATE_BUY) VALUES ('20BIT0003','3','21-MAR-21')
INSERT INTO BUYS (STUDENT_ID,BOOK_ID,DATE_BUY) VALUES ('20BIT0004','4','13-APR-21')
INSERT INTO BUYS (STUDENT_ID,BOOK_ID,DATE_BUY) VALUES ('20BIT0005','5','14-MAY-21')
INSERT INTO BUYS (STUDENT_ID,BOOK_ID,DATE_BUY) VALUES ('20BIT0006','6','01-JUN-21')
INSERT INTO BUYS (STUDENT_ID,BOOK_ID,DATE_BUY) VALUES ('20BIT0007','7','11-JUL-21')
```

STUDENT_ID	BOOK_ID	DATE_BUY
20BIT0001	1	01-JAN-21
20BIT0002	2	11-FEB-21
20BIT0003	3	21-MAR-21
20BIT0004	4	13-APR-21
20BIT0005	5	14-MAY-21
20BIT0006	6	01-JUN-21

Download CSV
6 rows selected.

TAKES:

```
CREATE TABLE TAKES (
COURSE_ID VARCHAR2(10),
STUDENT_ID VARCHAR2(10),
constraint FK_TAKES_COURSE_ID foreign key (COURSE_ID) references
COURSE(COURSE_ID),
constraint FK_TAKES_STUDENT_ID foreign key (STUDENT_ID) references
STUDENT(STUDENT_ID),
CONSTRAINT PK_TAKES PRIMARY KEY(COURSE_ID,STUDENT_ID)
)
```

```
INSERT INTO TAKES(COURSE_ID,STUDENT_ID) VALUES ('ITE1001','20BIT0001')
INSERT INTO TAKES(COURSE_ID,STUDENT_ID) VALUES ('ITE1003','20BIT0002')
INSERT INTO TAKES(COURSE_ID,STUDENT_ID) VALUES ('ITE1005','20BIT0004')
INSERT INTO TAKES(COURSE_ID,STUDENT_ID) VALUES ('CSE1002','20BIT0005')
INSERT INTO TAKES(COURSE_ID,STUDENT_ID) VALUES ('CSE1014','20BIT0006')
INSERT INTO TAKES(COURSE_ID,STUDENT_ID) VALUES ('ITE1014','20BIT0007')
INSERT INTO TAKES(COURSE_ID,STUDENT_ID) VALUES ('ITE1004','20BIT0003')
```

COURSE_ID	STUDENT_ID
CSE1002	20BIT0005
CSE1014	20BIT0006
ITE1001	20BIT0001
ITE1003	20BIT0002
ITE1004	20BIT0003
ITE1005	20BIT0004

Download CSV

6 rows selected.

STU_COURSES:

```
CREATE TABLE STU_COURSES (
COURSE_IDS VARCHAR2(10),
STUDENT_ID VARCHAR2(10),
constraint FK_STU_COURSES_STUDENT_ID foreign key (STUDENT_ID)
references STUDENT(STUDENT_ID),
CONSTRAINT PK_STU_COURSES PRIMARY KEY(COURSE_IDS,STUDENT_ID)
)
```

```
INSERT INTO STU_COURSES(COURSE_IDS,STUDENT_ID) VALUES ('ITE1001','20BIT0001')
INSERT INTO STU_COURSES(COURSE_IDS,STUDENT_ID) VALUES ('ITE1003','20BIT0002')
INSERT INTO STU_COURSES(COURSE_IDS,STUDENT_ID) VALUES ('ITE1004','20BIT0003')
INSERT INTO STU_COURSES(COURSE_IDS,STUDENT_ID) VALUES ('ITE1005','20BIT0004')
INSERT INTO STU_COURSES(COURSE_IDS,STUDENT_ID) VALUES ('CSE1002','20BIT0005')
INSERT INTO STU_COURSES(COURSE_IDS,STUDENT_ID) VALUES ('CSE1014','20BIT0006')
INSERT INTO STU_COURSES(COURSE_IDS,STUDENT_ID) VALUES ('ITE1014','20BIT0007')
```

COURSE_IDS	STUDENT_ID
CSE1002	20BIT0005
CSE1014	20BIT0006
ITE1001	20BIT0001
ITE1003	20BIT0002
ITE1004	20BIT0003
ITE1005	20BIT0004

Download CSV

6 rows selected.

Statement 22

▷

```
select table_name from user_tables
```

TABLE_NAME
BOOKS
BUYS
COURSE
DEPARTMENT
EXAM
MENTOR
STUDENT
STU_COURSES
TAKES
TEACHER
TEACHES

11 rows selected.

Alter, Delete and update

ALTER:

ALTER TABLE STUDENT MODIFY(STUDENT_ID VARCHAR2(11));

ALTER TABLE COURSE MODIFY(COURSE_ID VARCHAR2(11));

ALTER TABLE MENTOR MODIFY(E_MAIL VARCHAR2(31) NOT NULL);

ALTER TABLE TEACHER MODIFY(PH_NO NUMBER(10));

ALTER TABLE STUDENT ADD (D_ID VARCHAR2(10));

```
ALTER TABLE MENTOR MODIFY(E_MAIL VARCHAR2(31) NOT NULL)
```

```
Table altered.
```

```
ALTER TABLE STUDENT MODIFY(STUDENT_ID VARCHAR2(11))
```

```
Table altered.
```

```
ALTER TABLE COURSE MODIFY(COURSE_ID VARCHAR2(11))
```

```
Table altered.
```

```
ALTER TABLE TEACHER MODIFY(PH_NO NUMBER(20))
```

```
ALTER TABLE TEACHER MODIFY(E_MAIL VARCHAR2(30))
```

```
Table altered.
```

```
ALTER TABLE STUDENT MODIFY(E_MAIL VARCHAR2(30))
```

```
Table altered.
```

DELETE:

```
DELETE FROM TAKES WHERE STUDENT_ID='20BIT0007';

DELETE FROM TEACHES WHERE TEACHER_ID='TEACH7';

DELETE FROM STU_COURSES WHERE STUDENT_ID='20BIT0007';

DELETE FROM BUYS WHERE BOOK_ID='7';
```

```
DELETE FROM TAKES WHERE STUDENT_ID='20BIT0007'
```

```
1 row(s) deleted.
```

```
DELETE FROM TEACHES WHERE TEACHER_ID='TEACH7'
```

```
1 row(s) deleted.
```

```
DELETE FROM STU_COURSES WHERE STUDENT_ID='20BIT0007'
```

```
1 row(s) deleted.
```

```
DELETE FROM BUYS WHERE BOOK_ID='7'
```

```
1 row(s) deleted.
```

UPDATE:

UPDATE STUDENT SET FIRST_NAME = 'King' WHERE STUDENT_ID = '20BIT0006';

UPDATE BOOKS SET NAME = 'King' WHERE BOOK_ID = '6';

UPDATE EXAM SET NO_OF_ABS = 5 WHERE EXAM_ID = 'EX106';

UPDATE MENTOR SET FIRST_NAME = 'BIMAL' WHERE MENTOR_ID = 'MEN6';

```
UPDATE STUDENT SET FIRST_NAME = 'King'  WHERE STUDENT_ID = '20BIT0006'
```

```
1 row(s) updated.
```

```
UPDATE BOOKS SET NAME = 'King'  WHERE BOOK_ID = '6'
```

```
1 row(s) updated.
```

```
UPDATE EXAM SET NO_OF_ABS = 5  WHERE EXAM_ID = 'EX106'
```

```
1 row(s) updated.
```

```
UPDATE MENTOR SET FIRST_NAME = 'BIMAL'  WHERE MENTOR_ID = 'MEN6'
```

```
1 row(s) updated.
```

Primary key and foreign key constraint

ALTER TABLE STUDENT ADD (CONSTRAINT STU_FK_A FOREIGN KEY (M_ID) REFERENCES MENTOR(MENTOR_ID));

ALTER TABLE STUDENT ADD (CONSTRAINT STU_FK_D_ID FOREIGN KEY (D_ID) REFERENCES DEPARTMENT(DEPARTMENT_ID));

ALTER TABLE COURSE ADD (CONSTRAINT COURSE_FK_D_ID FOREIGN KEY (D_ID) REFERENCES DEPARTMENT(DEPARTMENT_ID));

ALTER TABLE EXAM ADD (CONSTRAINT EXAM_FK_D_ID FOREIGN KEY (D_ID) REFERENCES DEPARTMENT(DEPARTMENT_ID));

```
ALTER TABLE STUDENT ADD (CONSTRAINT STU_FK_D_ID FOREIGN KEY (D_ID)  REFERENCES DEPARTMENT(DEPARTMENT_ID))

Table altered.

ALTER TABLE COURSE ADD (CONSTRAINT COURSE_FK_D_ID FOREIGN KEY (D_ID)  REFERENCES DEPARTMENT(DEPARTMENT_ID))

Table altered.

ALTER TABLE EXAM ADD (CONSTRAINT EXAM_FK_D_ID FOREIGN KEY (D_ID)  REFERENCES DEPARTMENT(DEPARTMENT_ID))

Table altered.
```

Primary keys are already added.

Select with Where clause

SELECT FIRST_NAME,E_MAIL FROM STUDENT WHERE STUDENT_ID='20BIT0004';

SELECT NO_OF_STU,NO_OF_ABS FROM EXAM WHERE EXAM_ID='EX103';

SELECT PH_NO ,FIRST_NAME FROM MENTOR WHERE MENTOR_ID='MEN2';

SELECT FIRST_NAME,E_MAIL FROM TEACHER WHERE TEACHER_ID='TEACH2';

```
SELECT FIRST_NAME,E_MAIL FROM STUDENT WHERE STUDENT_ID='20BIT0004'
```

FIRST_NAME	E_MAIL
DIVS	20BIT0004@GMAIL.COM

```
SELECT NO_OF_STU,NO_OF_ABS FROM EXAM WHERE EXAM_ID='EX103'
```

NO_OF_STU	NO_OF_ABS
60	1

```
SELECT PH_NO ,FIRST_NAME FROM MENTOR WHERE MENTOR_ID='MEN2'
```

PH_NO	FIRST_NAME
7806665789	DR.chandrababu

```
SELECT FIRST_NAME,E_MAIL FROM TEACHER WHERE TEACHER_ID='TEACH2'
```

FIRST_NAME	E_MAIL
DR.RAMA	TEACH2@vit.ac.in

Order by clause

SELECT NAME,CREDITS,TYPE FROM COURSE ORDER BY NAME ASC;

SELECT FIRST_NAME,PH_NO FROM MENTOR ORDER BY MENTOR_ID DESC;

SELECT MALPRACTICE FROM EXAM ORDER BY EXAM_ID ASC;

```
SELECT NAME,CREDITS,TYPE FROM COURSE ORDER BY NAME ASC
```

NAME	CREDITS	TYPE
DBMS	3	ETH
DLM	4	ETH,ELA
DSA	4	ETH,ELA
HCI	3	ETH,EPJ
JAVA	4	ETH,ELA
OOPS	3	ELA
SOFTWARE	2	ETH

7 rows selected.

```
SELECT FIRST_NAME,PH_NO FROM MENTOR ORDER BY MENTOR_ID DESC
```

FIRST_NAME	PH_NO
DR.rama	8118765700
BIMAL	9118765700
DR.Sheker	9008765700
DR.SRIRAM	9996665700
Dheeran	7006665700
DR.chandrababu	7806665789
DR.Surendhar	9996665789

7 rows selected.

```
SELECT FIRST_NAME,LAST_NAME FROM MENTOR WHERE FIRST_NAME LIKE '%A%' ORDER BY LAST_NAME
```

FIRST_NAME	LAST_NAME
DR.SRIRAM	CHANDRA
BIMAL	praneeth

2 rows selected.

Like clause

SELECT FIRST_NAME,LAST_NAME FROM MENTOR WHERE FIRST_NAME LIKE '%A%' ORDER BY LAST_NAME;

SELECT FIRST_NAME,LAST_NAME FROM STUDENT WHERE LAST_NAME LIKE '%S%' ORDER BY LAST_NAME;

SELECT FIRST_NAME,LAST_NAME FROM TEACHER WHERE FIRST_NAME LIKE '%E%' ORDER BY LAST_NAME;

```
SELECT FIRST_NAME,LAST_NAME FROM MENTOR WHERE FIRST_NAME LIKE '%A%' ORDER BY LAST_NAME
```

FIRST_NAME	LAST_NAME
DR.SRIRAM	CHANDRA
BIMAL	praneeth

```
2 rows selected.
```

```
SELECT FIRST_NAME,LAST_NAME FROM STUDENT WHERE LAST_NAME LIKE '%S%' ORDER BY LAST_NAME
```

```
no data found
```

```
SELECT FIRST_NAME,LAST_NAME FROM TEACHER WHERE FIRST_NAME LIKE '%E%' ORDER BY LAST_NAME
```

FIRST_NAME	LAST_NAME
DR.RASHEED	REDDHY
DR.SUDHEER	REDDY

```
2 rows selected.
```

Is null/ is not null

SELECT PH_NO,DEPARTMENT_ID FROM DEPARTMENT WHERE PH_NO IS NULL;

SELECT NAME,TYPE FROM COURSE WHERE TYPE IS NULL;

SELECT PH_NO,DEPARTMENT_ID FROM DEPARTMENT WHERE PH_NO IS NOT NULL;

SELECT NAME,TYPE FROM COURSE WHERE TYPE IS NOT NULL;

```
SELECT PH_NO,DEPARTMENT_ID FROM DEPARTMENT WHERE PH_NO IS NULL
```

```
no data found
```

```
SELECT NAME,TYPE FROM COURSE WHERE TYPE IS NULL
```

```
no data found
```

```
SELECT PH_NO,DEPARTMENT_ID FROM DEPARTMENT WHERE PH_NO IS NOT NULL
```

PH_NO	DEPARTMENT_ID
9876504321	BIT1
8706543219	CSE1
8706512349	BCE1
9102345678	EEE1
8976504321	ECE1
7896504321	AER1
9871234560	MIS1

7 rows selected.

```
SELECT NAME,TYPE FROM COURSE WHERE TYPE IS NOT NULL
```

NAME	TYPE
DSA	ETH,ELA
DBMS	ETH
DLM	ETH,ELA
HCI	ETH,EPJ
SOFTWARE	ETH
OOPS	ELA
JAVA	ETH,ELA

7 rows selected.

Any five Aggregate functions

SELECT AVG(NO_OF_STU), AVG(DISTINCT NO_OF_STU) FROM EXAM;

SELECT MIN (NO_OF_STU), MIN (DISTINCT NO_OF_STU) FROM EXAM;

SELECT STDDEV(CREDITS), STDDEV(DISTINCT CREDITS) FROM COURSE;

SELECT VARIANCE(NO_OF_ABS), VARIANCE(DISTINCT NO_OF_ABS) FROM EXAM;

SELECT COUNT(MALPRACTICE), COUNT(DISTINCT MALPRACTICE) FROM EXAM;

select max(NO_OF_ABS) from EXAM;

select avg(NO_OF_STU) as "Average NUMBER of STUDENTS" from EXAM;

```
SELECT AVG(NO_OF_STU), AVG(DISTINCT NO_OF_STU) FROM EXAM
```

AVG(NO_OF_STU)	AVG(DISTINCTNO_OF_STU)
47.85714285714285714285714285714285714286	49.16666666666666666666666666666666666667

```
SELECT MIN (NO_OF_STU), MIN (DISTINCT NO_OF_STU) FROM EXAM
```

MIN(NO_OF_STU)	MIN(DISTINCTNO_OF_STU)
30	30

```
SELECT STDDEV(CREDITS), STDDEV(DISTINCT CREDITS) FROM COURSE
```

STDDEV(CREDITS)	STDDEV(DISTINCTCREDITS)
.7559289460184544544290330724683601216316	1

```
SELECT VARIANCE(NO_OF_ABS), VARIANCE(DISTINCT NO_OF_ABS) FROM EXAM
```

VARIANCE(NO_OF_ABS)	VARIANCE(DISTINCTNO_OF_ABS)
3.23809523809523809523809523809523809524	3.5

```
select max(NO_OF_ABS) from EXAM
```

MAX(NO_OF_ABS)
5

```
select avg(NO_OF_STU) as "Average NUMBER of STUDENTS" from EXAM
```

Average NUMBER of STUDENTS
47.85714285714285714285714285714285714286

Any five date functions

- SELECT ADD_MONTHS(DATE_BUY, 1) FROM BUYS;

```
1  SELECT  ADD_MONTHS( DATE_BUY, 1 ) FROM BUYS;
```

ADD_MONTHS(DATE_BUY,1)
01-FEB-21
11-MAR-21
21-APR-21
13-MAY-21
14-JUN-21
01-JUL-21

Download CSV
6 rows selected.

- SELECT EXTRACT(YEAR FROM TO_DATE(DATE_BUY, 'DD-Mon-YYYY')) YEAR FROM BUYS;

```
1  SELECT EXTRACT( YEAR FROM TO_DATE( DATE_BUY,  'DD-Mon-YYYY' ) ) YEAR FROM BUYS;
```

YEAR
21
21
21
21
21
21

Download CSV
6 rows selected.

❖ SELECT LAST_DAY(DATE_BUY) FROM BUYS;

```
1   SELECT LAST_DAY(DATE_BUY) FROM BUYS;
```

LAST_DAY(DATE_BUY)
31-JAN-21
28-FEB-21
31-MAR-21
30-APR-21
31-MAY-21
30-JUN-21

Download CSV
6 rows selected.

❖ SELECT TO_CHAR(ROUND(TO_DATE(DATE_BUY, 'DD-Mon-YYYY')), 'DD-Mon-YYYY') rounded_result FROM BUYS;

```
1   SELECT TO_CHAR(  ROUND( TO_DATE( DATE_BUY,  'DD-Mon-YYYY' ) ), 'DD-Mon-YYYY' )
2   rounded_result FROM BUYS;
```

ROUNDED_RESULT
01-Jan-0021
11-Feb-0021
21-Mar-0021
13-Apr-0021
14-May-0021
01-Jun-0021

Download CSV
6 rows selected.

❖ SELECT TO_CHAR(DATE_BUY, 'YYYY-MM-DD') FROM BUYS;

```
1  SELECT TO_CHAR( DATE_BUY, 'YYYY-MM-DD' ) FROM BUYS;
```

TO_CHAR(DATE_BUY,'YYYY-MM-DD')
2021-01-01
2021-02-11
2021-03-21
2021-04-13
2021-05-14
2021-06-01

Download CSV

6 rows selected.

Any three numeric functions

❖ SELECT NAME, ceil(PRICE) from Books;
❖ SELECT GREATEST(PRICE) FROM BOOKS;
❖ SELECT NAME, floor(sqrt(PRICE)) from Books;
❖ SELECT ROUND(PRICE, 2) FROM BOOKS;

NAME	CEIL(PRICE)
WEB_TECH	1500
DBMS	1300
SOCIAL	1400
OOPS	1250
HCI	1340
King	900
STATS	890

Download CSV

7 rows selected.

GREATEST(PRICE)
1500
1300
1400
1250
1340
900
890

Download CSV

NAME	FLOOR(SQRT(PRICE))
WEB_TECH	38
DBMS	36
SOCIAL	37
OOPS	35
HCI	36
King	30
STATS	29

Download CSV

7 rows selected.

ROUND(PRICE,2)
1500
1300
1400
1250
1340
900
890

Download CSV

Any five String Functions

- select NAME,length(NAME) as "BOOK NAME LENGTH" from Books;
- select trim(D_Name),trim(HOD) from DEPARTMENT;
- select substr(FIRST_Name,1,2) from STUDENT;
- select reverse(substr(LAST_Name,1,3)) from TEACHER;
- select concat(FIRST_Name,LAST_NAME) from TEACHER;

NAME	BOOK NAME LENGTH
WEB_TECH	8
DBMS	4
SOCIAL	6
OOPS	4
HCI	3
King	4
STATS	5

Download CSV

7 rows selected.

TRIM(D_NAME)	TRIM(HOD)
IT	VISWANATHAN
CSE	RAM
CSE_SPEC	PARIMALA
ELECTRONICS	VIJAYAN
ECE	BANSAL
AERONATICAL	KERITH
M_TECH	CHERITH

Download CSV

7 rows selected.

SUBSTR(FIRST_NAME,1,2)
SA
RA
SI
DI
HE
Ki
KA

Download CSV

7 rows selected.

REVERSE(SUBSTR(LAST_NAME,1,3))
DER
NUS
MUK
DER
DER
ER
DER

Download CSV

7 rows selected.

CONCAT(FIRST_NAME,LAST_NAME)
DR.SUDHEER REDDY
DR.RAMASUNDHAR
DR.RAVIKUMAR
DR.SRINIVASREDDY
DR.RASHEEDREDDHY
DR.SURAJ REDDY
DR.SAIREDDY

Download CSV

7 rows selected.

Group by and having

- SELECT EXAM_ID AS "EXAM", FLOOR(SUM(MALPRACTICE)) AS "Total Students in Malpractice", COUNT(EXAM_ID) AS "Number of EXAMS" FROM EXAM GROUP BY EXAM_ID;

EXAM	Total Students in Malpractice	Number of EXAMS
EX101	0	1
EX102	0	1
EX103	2	1
EX104	0	1
EX105	1	1
EX106	0	1
EX107	0	1

Download CSV

7 rows selected.

- SELECT EXAM_Id AS "EACH EXAM",FLOOR(SUM(NO_OF_STU)) as "NUMBER OF STUDENTS ",Count(NO_OF_ABS) AS "NUMBER OF ABSENT" FROM EXAM GROUP BY EXAM_ID HAVING COUNT(MALPRACTICE)>0;

EACH EXAM	NUMBER OF STUDENTS	NUMBER OF ABSENT
EX103	60	1
EX105	70	1
EX106	30	1
EX102	50	1
EX104	40	1
EX107	45	1
EX101	40	1

Download CSV

7 rows selected.

Join more than two tables

- select STUDENT.FIRST_Name,Books.Name from STUDENT,BOOKS;

KAVYA	SOCIAL
KAVYA	OOPS
KAVYA	HCI
KAVYA	King
KAVYA	STATS

Download CSV

49 rows selected.

- select STUDENT.FIRST_Name,Books.Name from STUDENT NATURAL JOIN BOOKS;

KAVYA	SOCIAL
KAVYA	OOPS
KAVYA	HCI
KAVYA	King
KAVYA	STATS

Download CSV

49 rows selected.

- SELECT STUDENT.FIRST_Name,BUYS.BOOK_ID,STU_COURSES.COURSE_IDS from BUYS, STUDENT , STU_COURSES where BUYS.STUDENT_Id=STUDENT.STUDENT_Id and STUDENT.STUDENT_Id=STU_COURSES.STUDENT_Id ;

FIRST_NAME	BOOK_ID	COURSE_IDS
HEY	5	CSE1002
King	6	CSE1014
SAI	1	ITE1001
RAM	2	ITE1003
SIN	3	ITE1004
DIVS	4	ITE1005
KAVYA	7	ITE1014

Download CSV

7 rows selected.

7. Query Optimization Query tree and query graph

- ```
  SELECT  FIRST_NAME,NAME,D_NAME
   FROM
   MENTOR,STUDENT,DEPARTMENT
  WHERE
  MENTOR_ID=M_ID  AND  D_ID= DEPARTMENT_ID  AND
FIRST_NAME=”SAI”;
  ```

- ```
  SELECT  NAME,DATE_BUY,FIRST_NAME
  FROM
  BOOKS,BUYS,STUDENT
  WHERE
  BOOK_ID=B_ID  AND  S_ID= STUDENT_ID  AND
  FIRST_NAME=”DEERAJ”  AND  PRICE>1000;
  ```

(1)

π First_Name, Name, D_Name

(σ Mentor_ID = M_ID AND D_ID = DEPARTMENT_ID

AND First_Name = "Sai".)

(Mentor x student x Department).

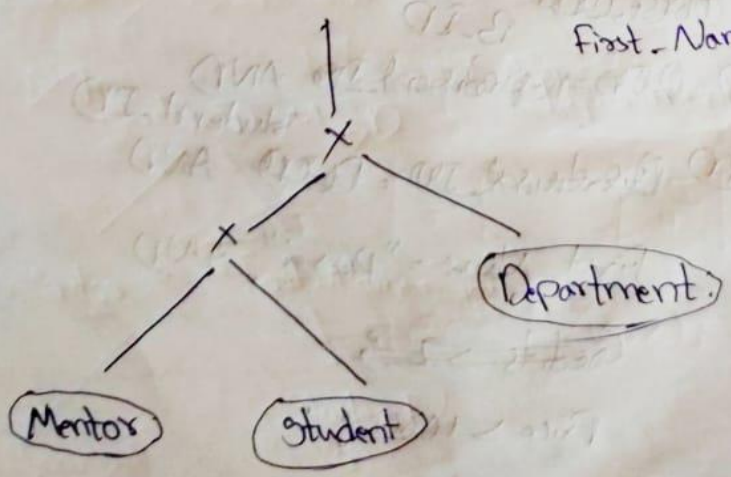
(a)

π First_Name, Name, D_Name

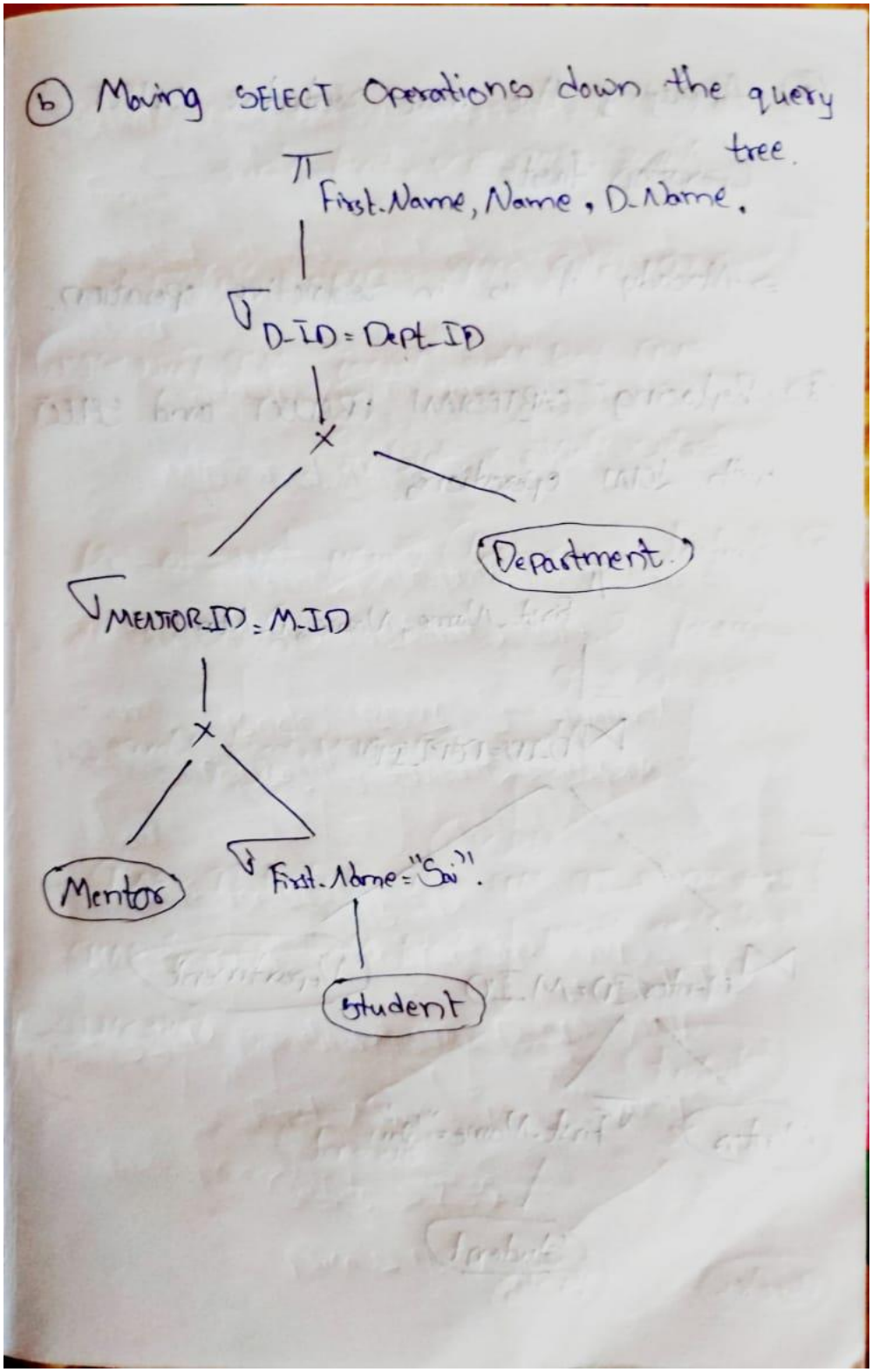
|

σ MentorID = M_ID AND DID = Dept_ID AND

First_Name = "Sai".)



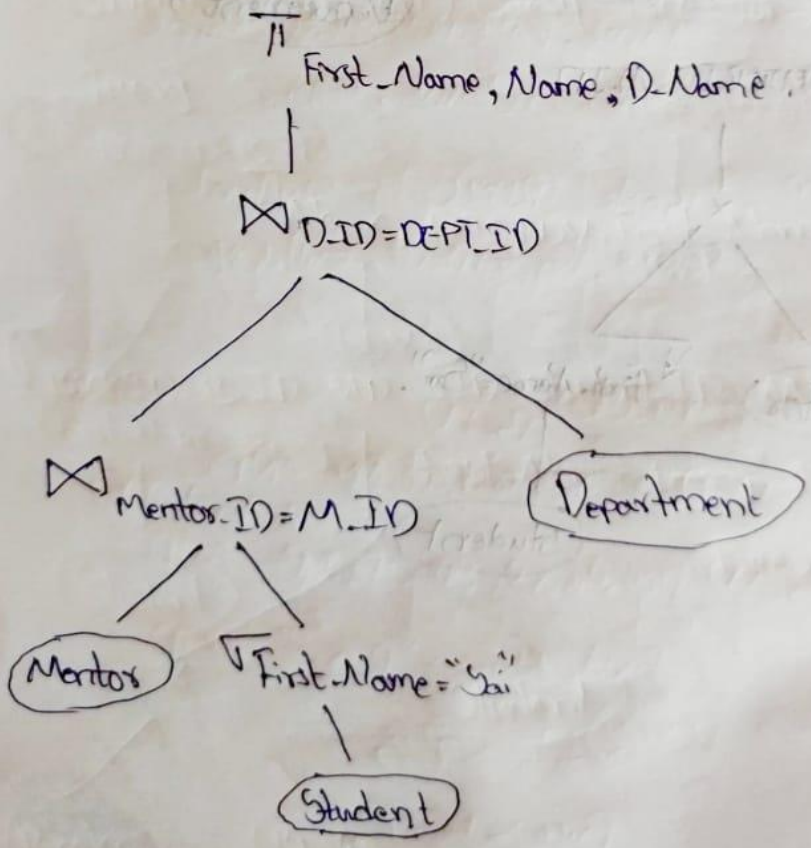
(b) Moving SELECT Operations down the query tree.



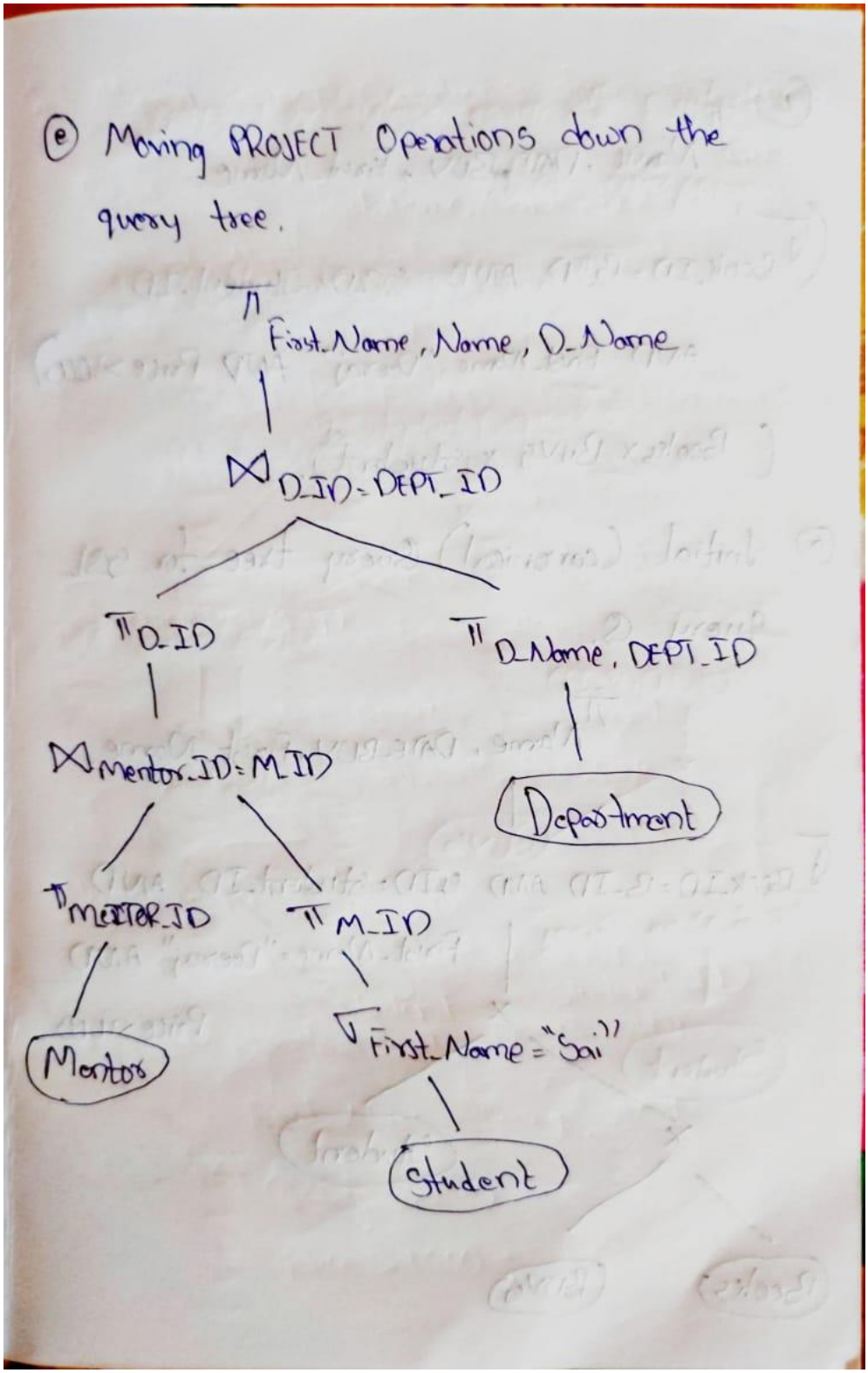
(c) Applying the more restrictive SELECT Operation first.

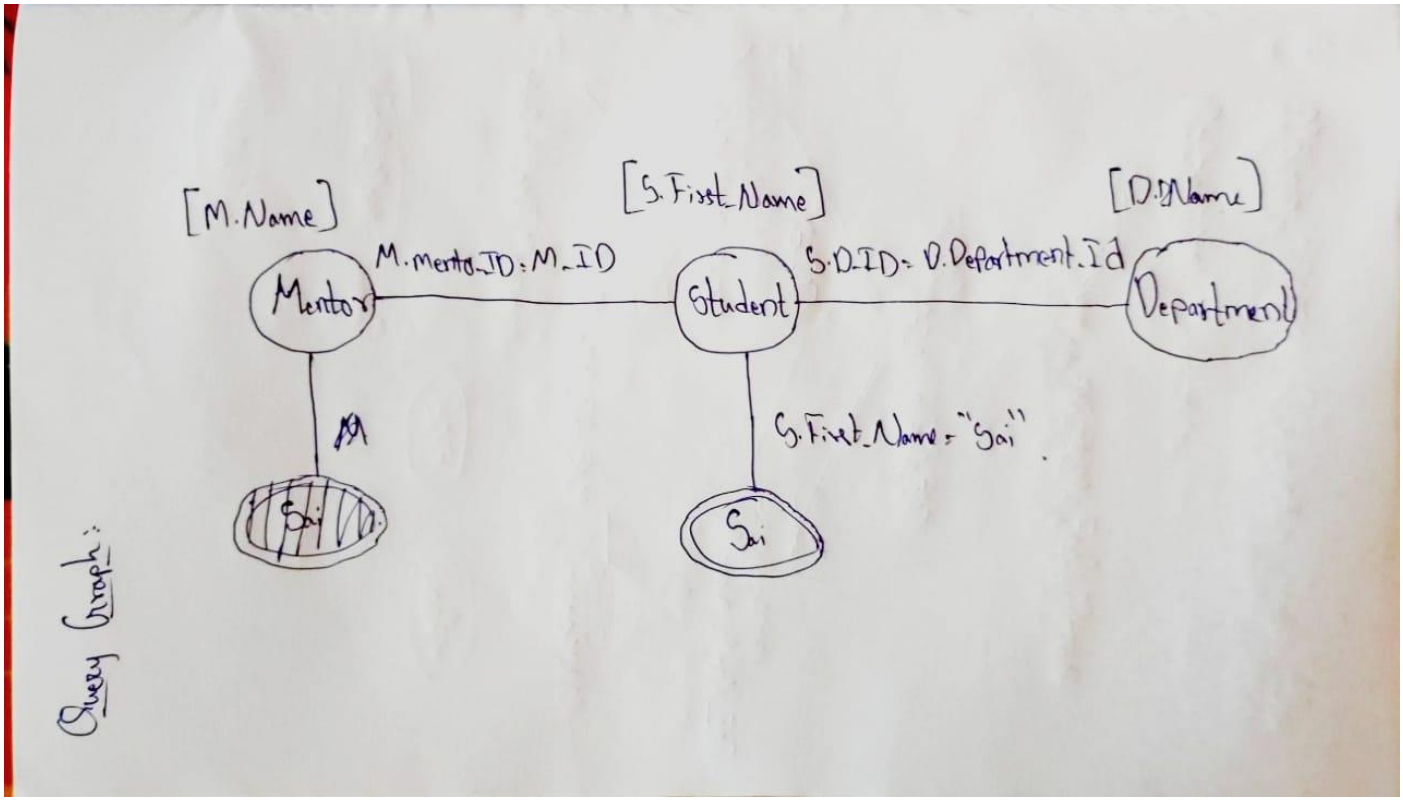
⇒ Already it is in restrictive operation.

(d) Replacing CARTESIAN PRODUCT and SELECT with JOIN operations.



(e) Moving PROJECT Operations down the query tree.




[M.Name]
[S.First_Name]
[D.DName]
M.Mentor_ID = M_ID
S.D_ID = D.Department_Id
Mentor
Student
Department
M
S.First_Name = "Sai"
Sai
Sai

Query Graph:

(2) $\pi_{Name, DATE\_BUY, First\_Name}$

$(\sigma_{Book\_ID = B\_ID \text{ AND } SID = Student\_ID}$

AND First_Name = "Deeraj" AND Price > 1000)

(Books x BUYS x student).

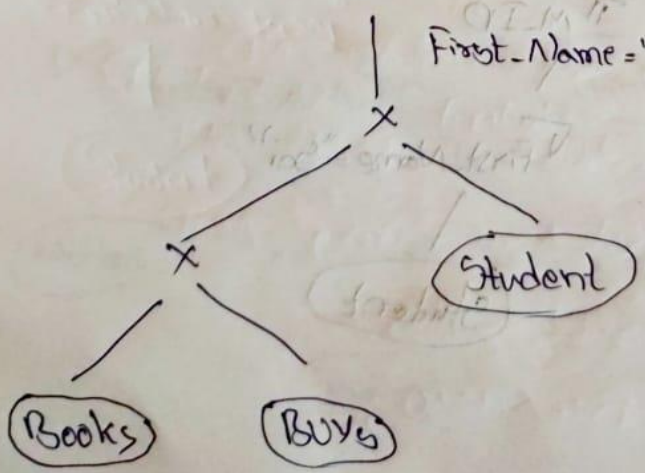
(a) Initial (canonical) Query tree for SQL query Q.

$\pi_{Name, DATE\_BUY, First\_Name}$

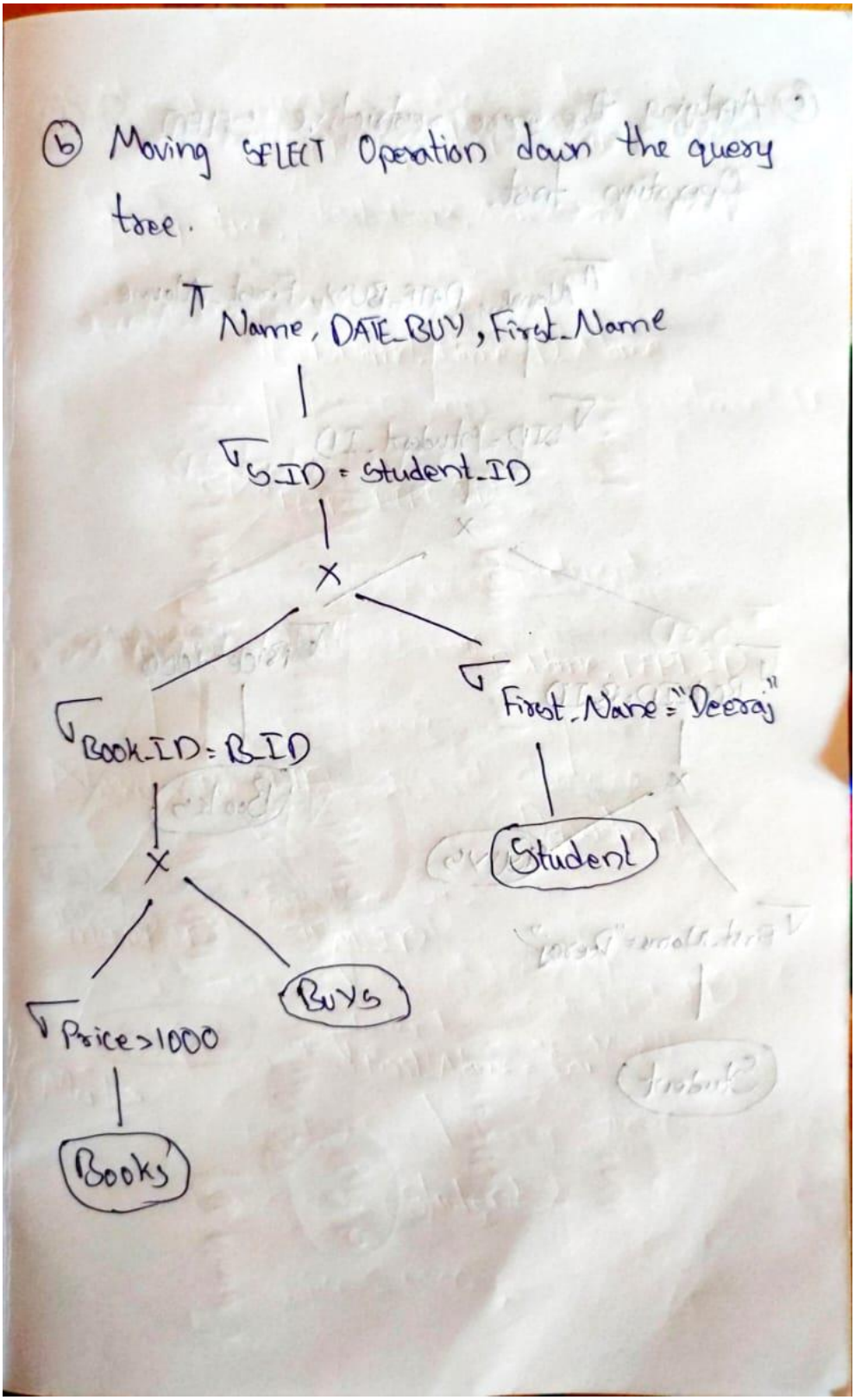
$\sigma_{Book\_ID = B\_ID \text{ AND } SID = Student\_ID \text{ AND }}$

First_Name = "Deeraj" AND

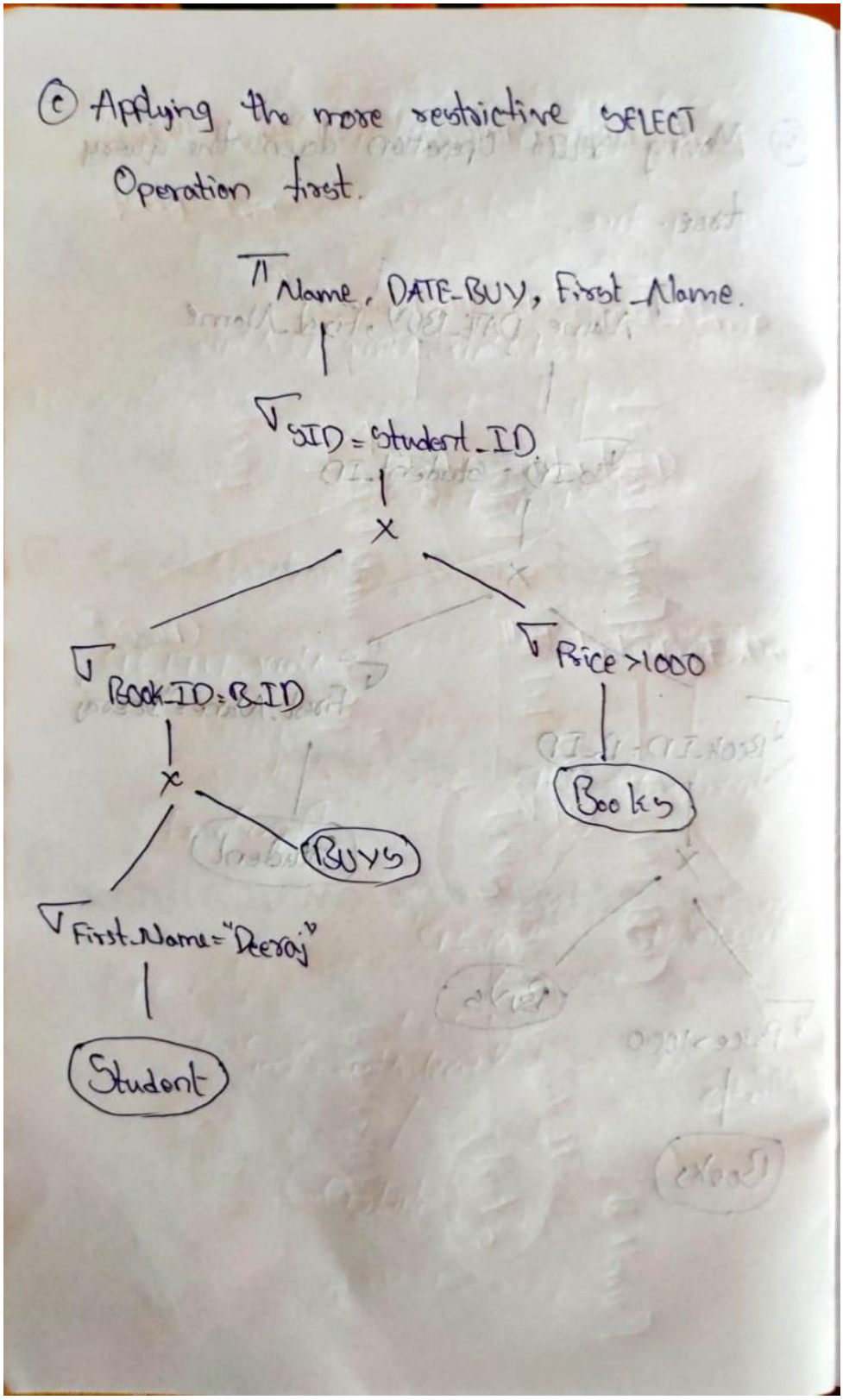
Price > 1000.



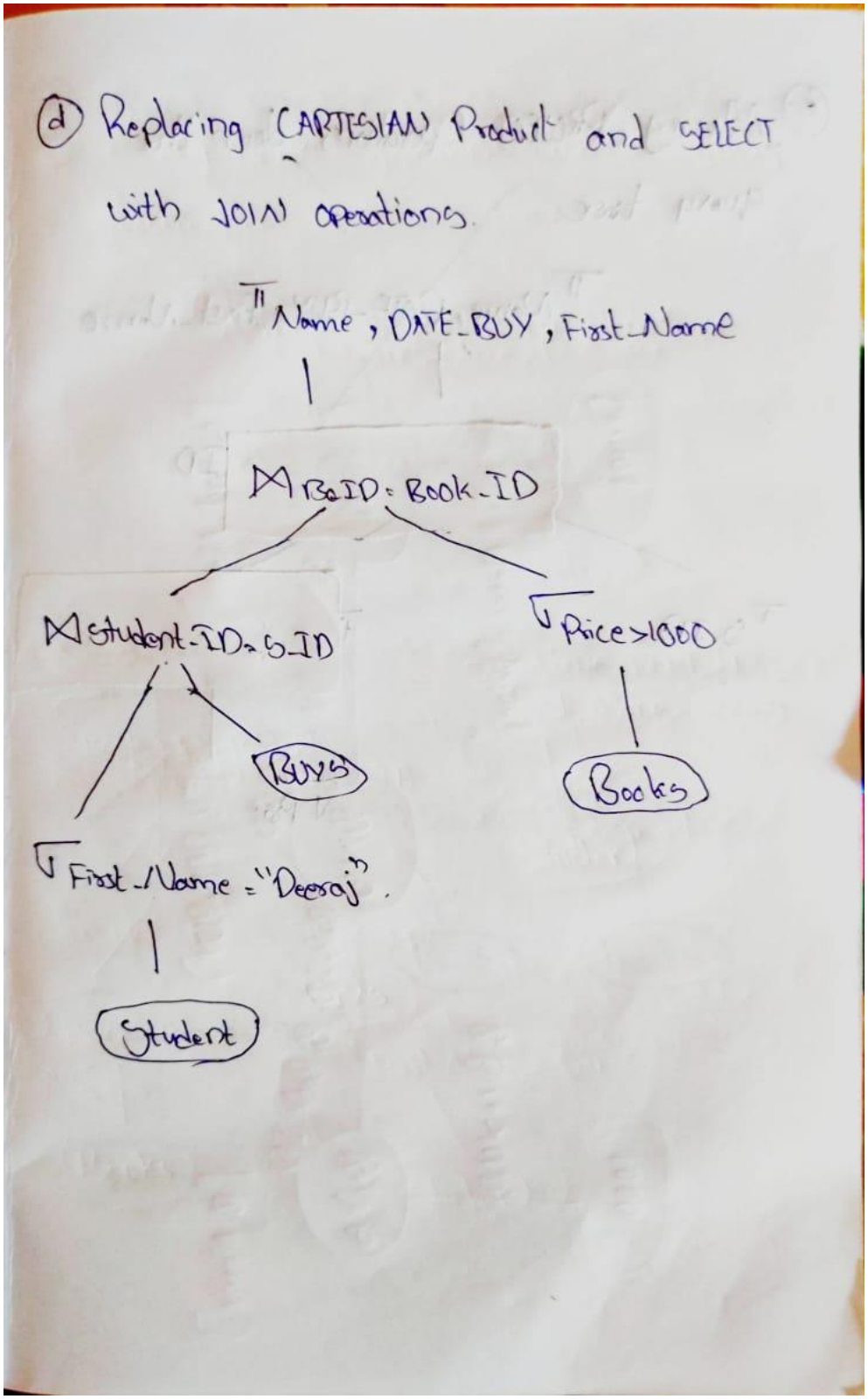
(b) Moving SELECT Operation down the query tree.



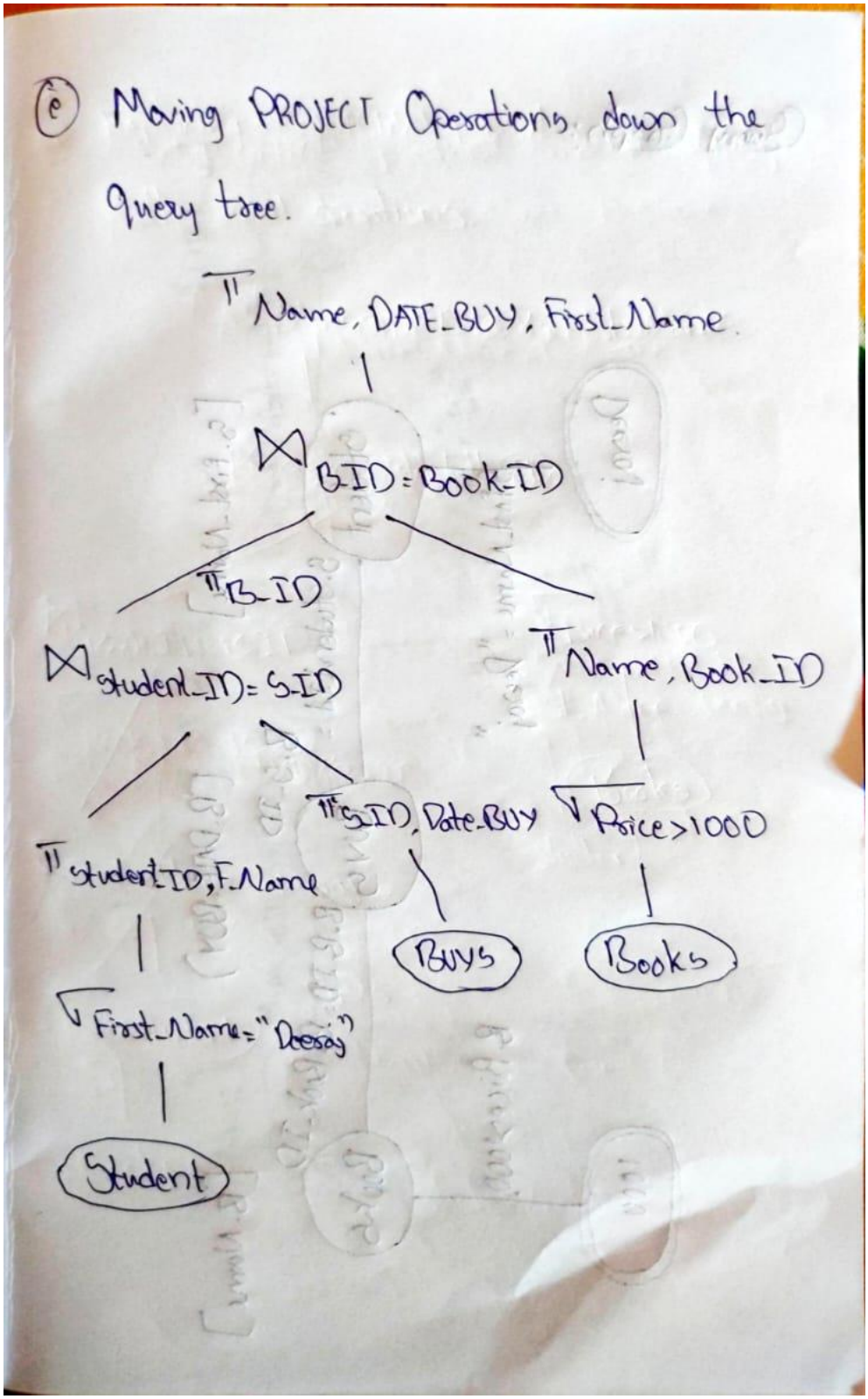
(c) Applying the more restrictive SELECT Operation first.

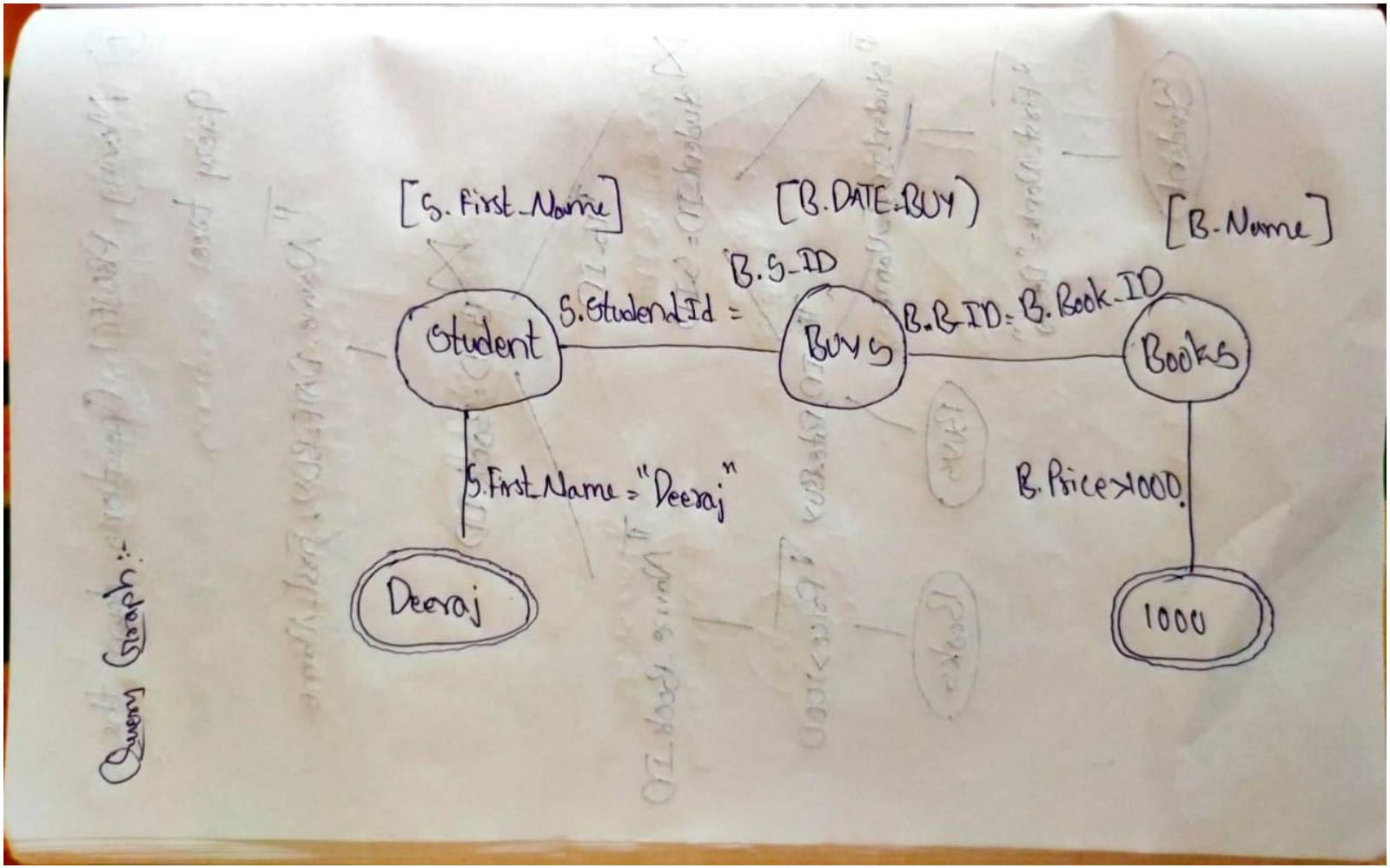


(d) Replacing CARTESIAN Product and SELECT with JOIN operations.

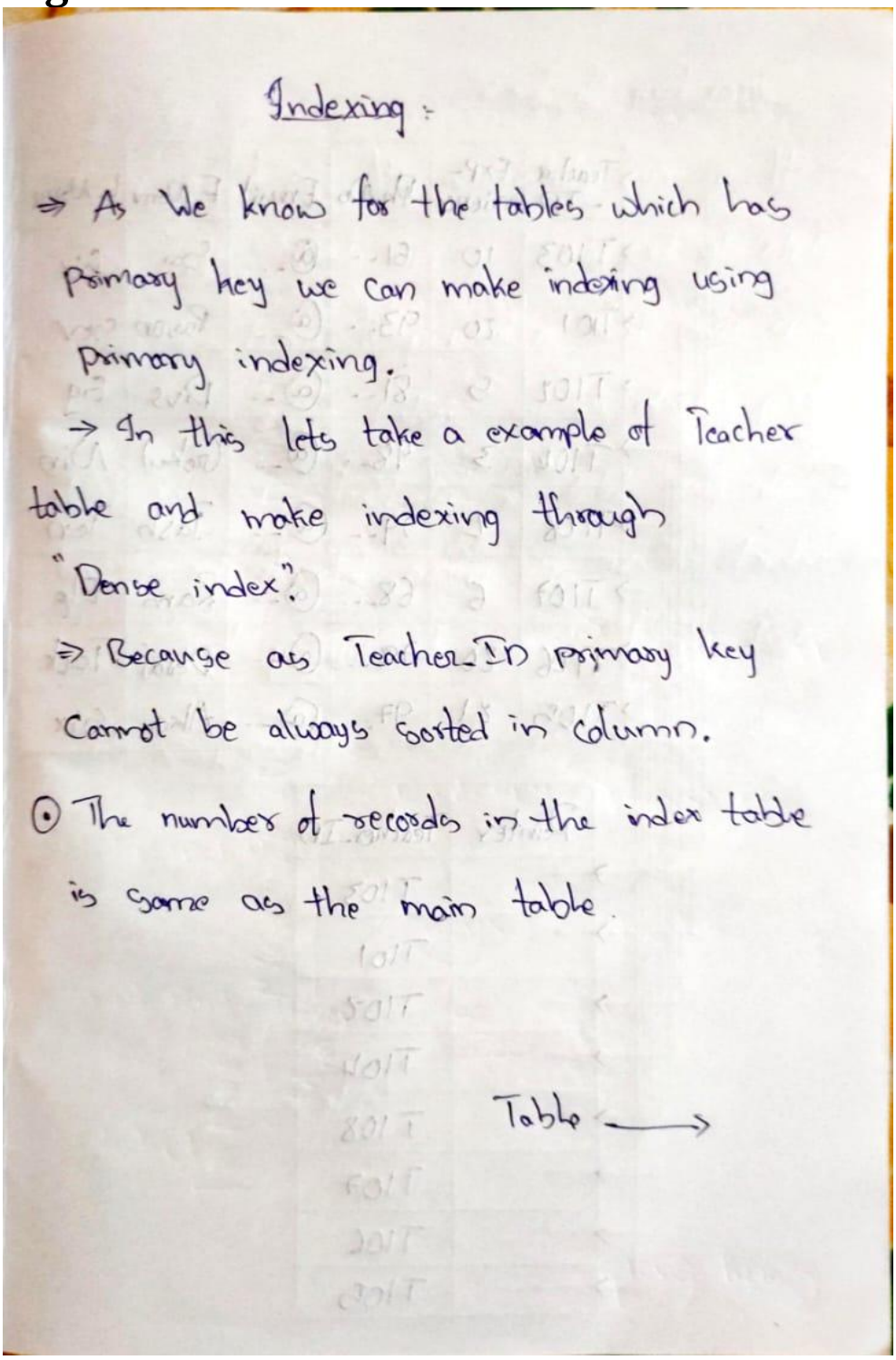


(e) Moving PROJECT Operations down the query tree.




Query Graph :-
[S. First_Name]
[B.DATE-BUY)
[B.Name]
B.S_ID
S.StudentId =
B.B_ID = B.Book_ID
Student
Buys
Books
S.First_Name = "Deeraj"
B.Price > 1000.
Deeraj
1000

8. Indexing index table.



Indexing :

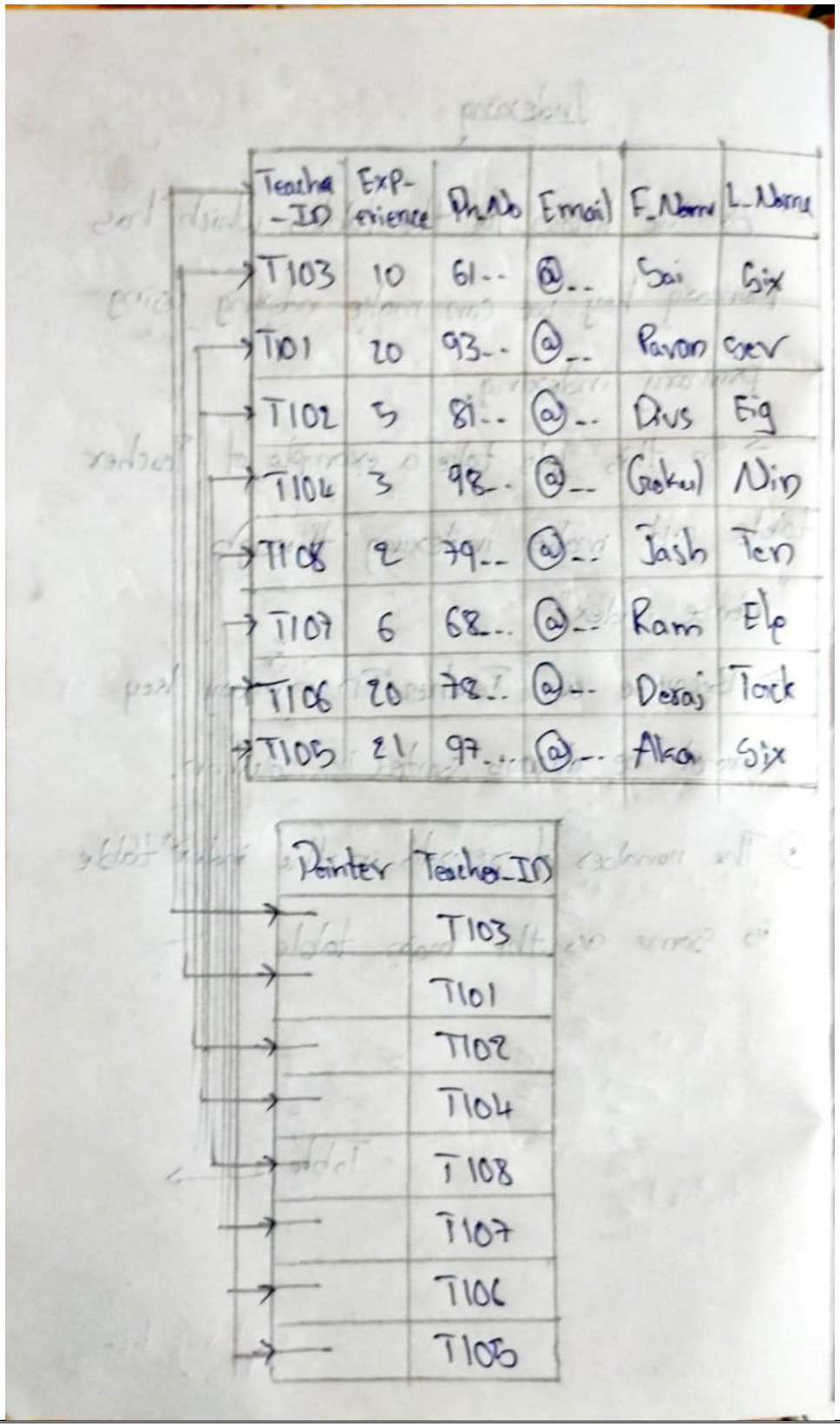
⇒ As we know for the tables which has primary key we can make indexing using primary indexing.

→ In this lets take a example of Teacher table and make indexing through "Dense index".

⇒ Because as Teacher_ID primary key cannot be always sorted in column.

⊙ The number of records in the index table is same as the main table.

Table ⟶



Teacher_ID	Experience	Ph_No	Email	F_Name	L_Name
T103	10	61--	@--	Sai	Six
T101	20	93--	@--	Pavan	Sev
T102	5	81--	@--	Dus	Eig
T104	3	98--	@--	Gokul	Nin
T108	2	79--	@--	Jash	Ten
T107	6	68--	@--	Ram	Elp
T106	20	78--	@--	Desai	Tock
T105	21	97--	@--	Akash	Six

Pointer	Teacher_ID
	T103
	T101
	T102
	T104
	T108
	T107
	T106
	T105

Clustering Index:

=> In this index cannot be a primary key so that they cannot be unique for each record.

⊙ In our database there are no such tables so we cannot make clustering index.

=> In this indexing we want to group the similar rows and put in a block so that the pointer points to each block.

⊙ In this way we can do clustering Indexing.

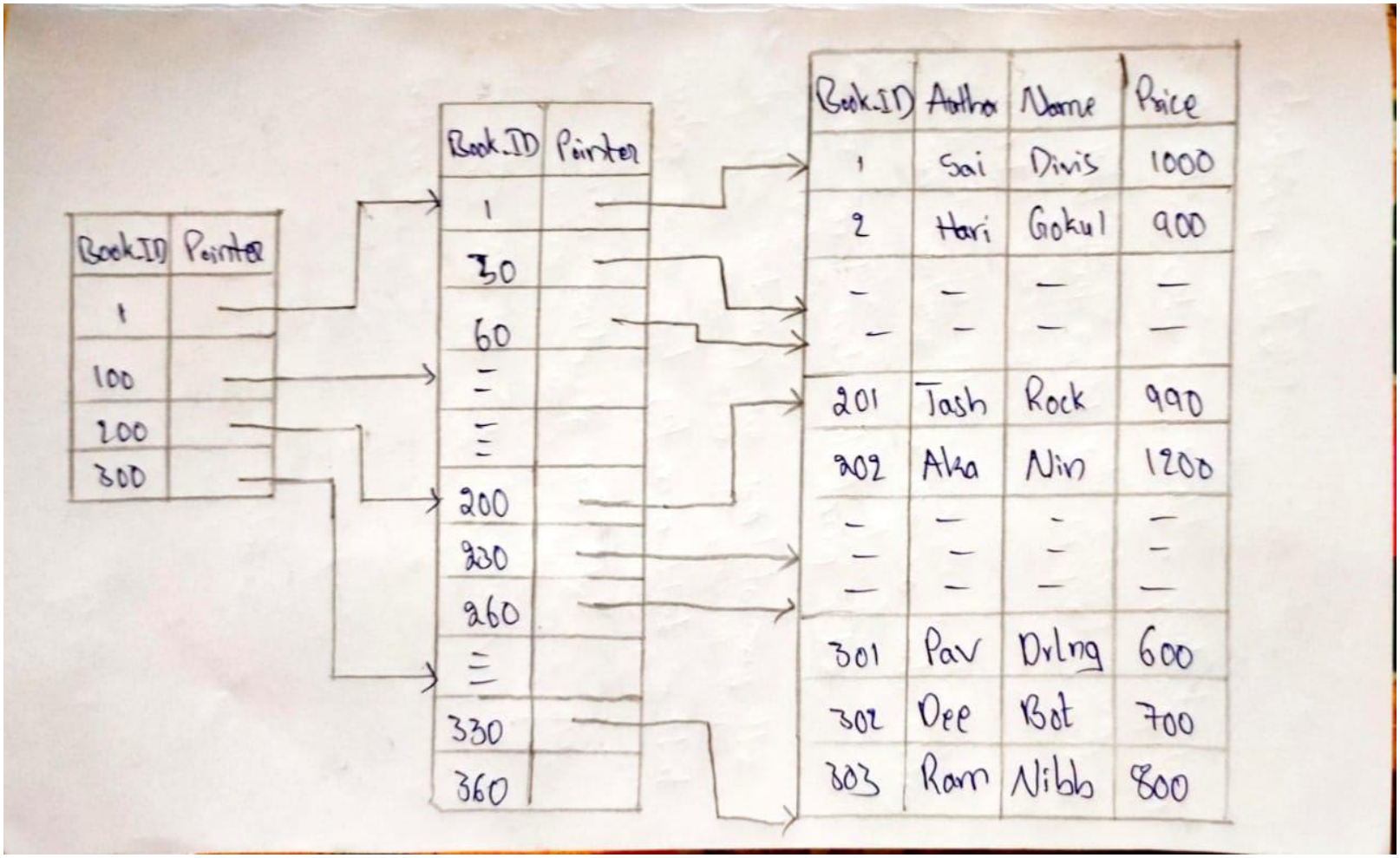
Secondary Indexes

→ If in our database if we have more data to be stored in one table. Then we can use this indexing if we have primary key which is in sorted order.

⊙ In our database Books is the table which there is more data to be stored and we have sorted primary key.

→ So, we can decrease the response time.

Table ⟶


Book_ID
Pointer
1
100
200
300
Book_ID
Pointer
1
30
60
200
230
260
330
360
Book_ID
Author
Name
Price
1
Sai
Divis
1000
2
Hari
Gokul
900
201
Tash
Rock
990
202
Alka
Nin
1200
301
Pav
Drlng
600
302
Dee
Bot
700
303
Ram
Nibb
800