Task No 12:Deleting record from Instructor Table

```
DELETE FROM Teaches
WHERE Instructor_ID = 'I002';
Delete From Instructor
where Instructor_ID='I002';
select*from Instructor;

SELECT * FROM Student;


SELECT * FROM Course;
```

Action Output

#	Time	Action	Messag
1	02:09:29	Delete From Instructor where Instructor_ID='I002'	Error C
2	02:10:37	DELETE FROM Teaches WHERE Instructor_ID = 'I002'	1 row(s
3	02:10:37	Delete From Instructor where Instructor_ID='I002'	1 row(s

12. Delete Data (DML)

Deleting a record from the **Instructor** table.

```
DELETE FROM Instructor
WHERE Instructor_ID = 'I003';
```

The **Instructor** table after the deletion:

Instructor_ID	Name	Department	Email
I001	Dr. Steve	Computer Science	steve@university.edu
I002	Dr. Mary	Mathematics	mary@university.edu
I004	Dr. Alice	Biology	alice@university.edu
I005	Dr. Bob	Engineering	bob@university.edu

Explanation:Shows an error as the entry is not deleted from child table(Teaches) .It will work if the entry is deleted from the Teaches Table first

Task No 13:Performing Natural Join Operation

```
select Instructor.name ,Teaches.Course_ID
from Instructor
natural join Teaches ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell

name	Course_ID
Dr. Mary	C102
Dr. John	C103
Dr. Alice	C104
Dr. Bob	C105

storage off for 3 months.

100% Normal text

Performing a **Natural Join** betv
which instructor teaches which

SELECT Instructor.Name, Teach
FROM Instructor
NATURAL JOIN Teaches;

Output:

Name	Course_ID
Dr. Steve	C101

Hi, Fahmid!

Manage your Google Account

Show more accounts

98% of 15 GB used

14.79 GB of 15 GB

Get storage Clean up space

Privacy Policy • Terms of Service

Explanation: Natural Join connect multiple tables based on the common columns

Task No 14:Performing Drop Table Command

```
select *from Instructor;
select *from Teaches;
select Instructor.name ,
from Instructor

natural join Teaches ;

drop table Teaches;
select *from Teaches;
```

Output

Action Output

#	Time	Action	Duration / Fetch
68	21:29:06	select *from Teaches LIMIT	0.000 sec / 0.00
69	21:32:48	select Instructor.name ,Tea	0.000 sec / 0.00
70	21:33:41	select Instructor.name ,Tea	0.000 sec / 0.00
71	22:21:14	drop table Teaches	0.156 sec
72	22:21:29	select *from Teaches LIMIT	0.015 sec

Hi, Fahmid!

Manage your Google Account

Show more accounts

98% of 15 GB used

14.79 GB of 15 GB

Get storage Clean up space

Privacy Policy • Terms of Service

INSERT INTO Takes (Student_ID, Course_ID, Sem
VALUES (7, 'C102', 'Spring', 2023);

This will fail because **Student_ID = 7** does not e

Task No 15:Foreign Key Constraints and Update

```
INSERT INTO Takes (Student_ID, Course_ID, Semester, Year)
VALUES (7, 'C102', 'Spring', 2023);
```

Output

Action Output

#	Time	Action	Message
69	21:32:48	select Instructor.name ,Teaches.Course_ID from Instructor natural jo...	4 row(s)
70	21:33:41	select Instructor.name ,Teaches.Semester from Instructor natural joi...	4 row(s)
71	22:21:14	drop table Teaches	0 row(s)
72	22:21:29	select *from Teaches LIMIT 0, 10000	Error Co
73	22:24:37	INSERT INTO Takes (Student_ID, Course_ID, Semester, Year) VA...	Error Co

For example, if we try to insert a row into the **Takes** table that r
non-existing student, the command will fail due to the foreign k
constraint.

```
INSERT INTO Takes (Student_ID, Course_ID, Semester, Year)
VALUES (7, 'C102', 'Spring', 2023);
```

This will fail because **Student_ID = 7** does not exist in the **Stude**

16. SELECT Command

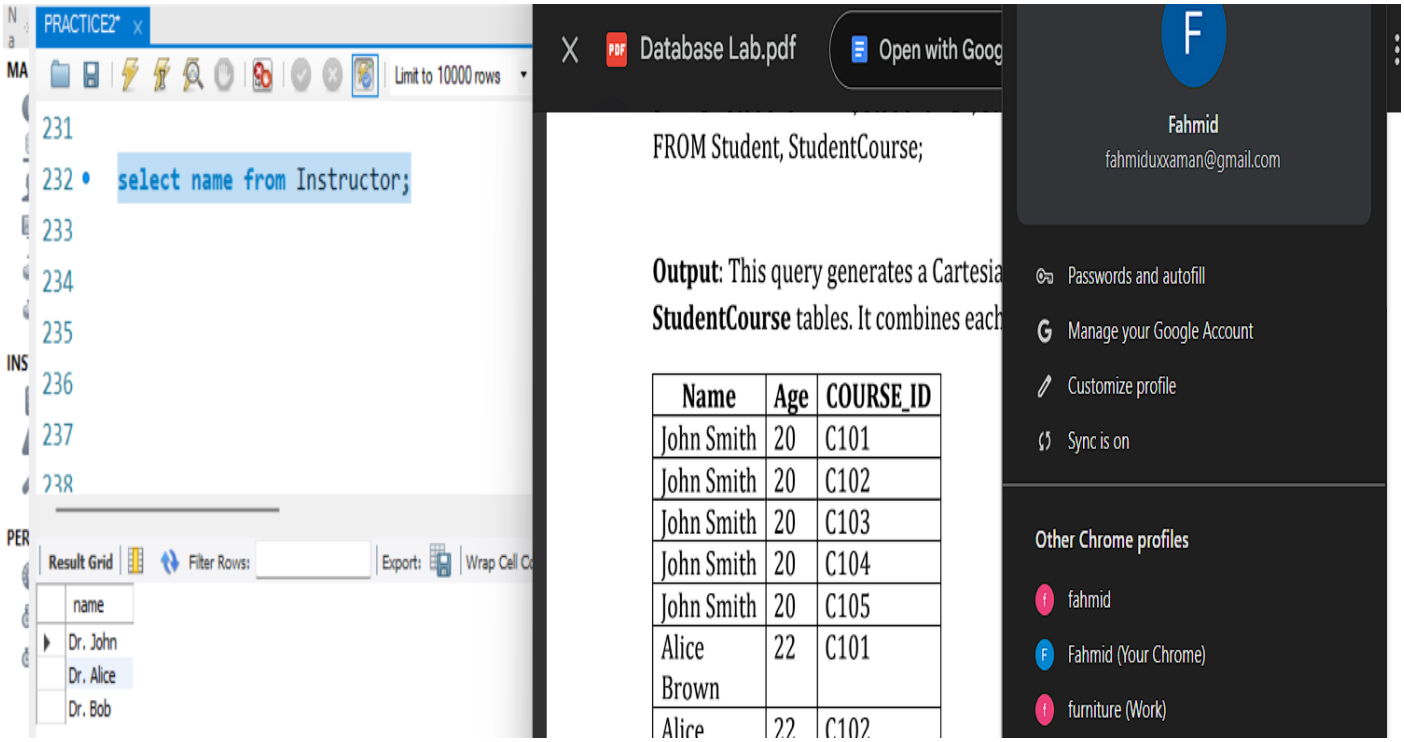
```
SELECT name
FROM instructor;
```

Output: This command selects the **name** column from the **Instructor** table.

Name

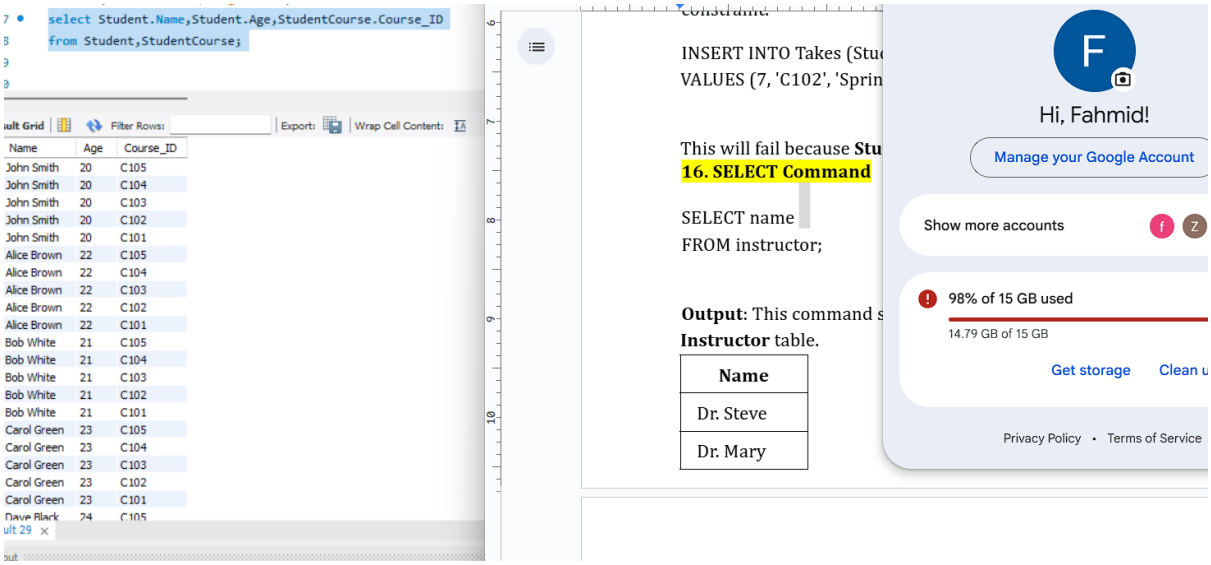
Explanation: An error occurs because the student_ID being inserted into the Takes table does not exist in the parent table Student. Since student_ID is defined as a foreign key referencing the Student table, the database enforces referential integrity. Therefore, inserting a record with a student_ID (e.g., 7) that does not exist in the Student table violates the foreign key constraint and results in an error.

Task No 16:Performing Select Command



Explanation: Shows the remaining entries of Instructor Table as I have done delete operation earlier

Task No 17:Performing Cross Join on Student and StudentCourse Table



Explanation : Cross Join (also known as a Cartesian product) combines every row from the first table with every row from the second table, creating all possible combinations

Task No 18:Cross Join (Alternative Syntex)

```
select Student.Name,Student.Age,StudentCourse.Course_ID
from Student cross join StudentCourse;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	Age	Course_ID
John Smith	20	C105
John Smith	20	C104
John Smith	20	C103
John Smith	20	C102
John Smith	20	C101
Alice Brown	22	C105
Alice Brown	22	C104
Alice Brown	22	C103
Alice Brown	22	C102
Alice Brown	22	C101
Bob White	21	C105
Bob White	21	C104
Bob White	21	C103
Bob White	21	C102
Bob White	21	C101
Carol Green	23	C105
Carol Green	23	C104
Carol Green	23	C103
Carol Green	23	C102
Carol Green	23	C101
Dave Black	24	C105

Result 30

File Edit View Insert Format Tools Extensions ...

100% Normal text

Dave Black		C10

18. Cross Join (Alternat

SELECT Student.NAME, S
FROM Student CROSS JO

Output: This will yield th
it's using a different synt

Name	Age	COUI
John Smith	20	C10
John Smith	20	C10
John Smith	20	C10
John Smith	20	C104

fahmiduxzaman@gmail.com

Hi, Fahmid!

Manage your Google Account

Show more accounts

98% of 15 GB used

14.79 GB of 15 GB

Get storage Clean up space

Privacy Policy • Terms of Service

Task No 19:Performing Explicit Inner Join operation

```
INSERT INTO Teaches (Instructor_ID, Course_ID, Semester, Year) VALU
('I002', 'C102', 'Spring', 2023),
('I003', 'C103', 'Fall', 2023),
('I004', 'C104', 'Spring', 2023),
('I005', 'C105', 'Fall', 2023);
select Name,Course_ID
from Instructor,Teaches
where Instructor.Instructor_ID=Teaches.Instructor_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	Course_ID
Dr. Mary	C102
Dr. John	C103
Dr. Alice	C104
Dr. Bob	C105

Result Grid | Form Editor

100% | Normal text

which course)

CREATE TABLE Teaches (
Instructor_ID VARCHAR(10),
Course_ID VARCHAR(10),
Semester VARCHAR(10),
Year INT,
PRIMARY KEY (Instructor_ID, Cours
KEY (Instructor_ID) REFERENCES In
KEY (Course_ID) REFERENCES Cour

Instructor_ID	Course_ID	Se
I001	C101	Fall
I002	C102	Sprin

fahmiduxxaman@gmail.com

Hi, Fahmid!

Manage your Google Account

Show more accounts

98% of 15 GB used

14.79 GB of 15 GB

Get storage | Clean up space

Task No 20:Performing Natural Join

```
select Name,Course_ID
From Instructor Natural Join Teaches;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	Course_ID
Dr. Mary	C102
Dr. John	C103
Dr. Alice	C104
Dr. Bob	C105

20. Natural Join Simplif

SELECT name, course_id
FROM instructor NATUR

Output: This is the simpl
same output as the previ

Name	course_id
Dr. Steve	C101
Dr. Mary	C102
Dr. John	C103
Dr. Alice	C104
Dr. Bob	C105

Hi, Fahmid!

Manage your Google Account

Show more accounts

98% of 15 GB used

14.79 GB of 15 GB

Get storage Clean up space

Privacy Policy • Terms of Service

Task No 21:Performing Multiple Natural Join

```
Select Name,Course_ID,Age
from Instructor natural join Teaches natural join Student;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	Course_ID	Age

Explanation:It shows empty columns as theres no common columns among the three tables

Task No 22:Renaming Table

```
alter table Student Rename To Student_Info;
select *from Student_Info;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Student_ID	Name	Age	Major	Email
1	John Smith	20	Computer Science	john@email.com
2	Alice Brown	22	Mathematics	alice@email.com
3	Bob White	21	Physics	bob@email.com
4	Carol Green	23	Biology	carol@email.com
5	Dave Black	24	Engineering	dave@email.com
6	Eve White	25	Chemistry	eve@email.com
NULL	NULL	NULL	NULL	NULL

ALTER TABLE Student RE

Output: After renaming t
name is now changed in

Table Name
Students_Info

Now, you would refer to t
future operations.

23. STRING Manipulati

SELECT CONCAT(Name,

fahmiduxxaman@gmail.com

Hi, Fahmid!

Manage your Google Account

Show more accounts

98% of 15 GB used

14.79 GB of 15 GB

Get storage Clean up space

Privacy Policy • Terms of Service

Task No 23: String Manipulation

```
alter table Student Rename To Student_Info;
select *from Student_Info;
select concat(Name,'-',Major)As Student_Info
from Student_Info;
```

Student_Info
John Smith-Computer Science
Alice Brown-Mathematics
Bob White-Physics
Carol Green-Biology
Dave Black-Engineering
Eve White-Chemistry

23. STRING Manipulation Exa

SELECT CONCAT(Name, ' - ', M

FROM Student;

Output: This command uses th
and **Major** columns with a sep
Student_Info.

Student_Info
John Smith - Computer Science
Alice Brown - Mathematics
Bob White - Physics

fahmiduxzaman@gmail.com

Hi, Fahmid!

Manage your Google Account

Show more accounts

98% of 15 GB used

14.79 GB of 15 GB

Get storage Clean up space

Privacy Policy • Terms of Service

Task No 24:Sorting Using Order By

```
select Name, Age
from Student_Info
order by Age desc;
```

Name	Age
Eve White	25
Dave Black	24
Carol Green	23
Alice Brown	22
Bob White	21
John Smith	20

24. ORDER BY Example

SELECT Name, Age
FROM Student
ORDER BY Age DESC;

Output: This command r
Student table and order

Name	Age
Dave Black	24
Carol Green	23
Alice Brown	22
Bob White	21

Task No 25:Example of Sorting with both ASC and DESC

```
insert into Student_Info(Student_ID,Name,Age,Major,Email
values
(8,'Zamann',23,'Arts','zamann@gmail.com');
select Name,Age,Major from Student_Info
order by Major asc ,Age desc;
```

Name	Age	Major
Zaman	27	Arts
Zamann	23	Arts
Carol Green	23	Biology
Eve White	25	Chemistry
John Smith	20	Computer Science
Dave Black	24	Engineering
Alice Brown	22	Mathematics
Bob White	21	Physics

FROM Student
ORDER BY Major ASC, Ag

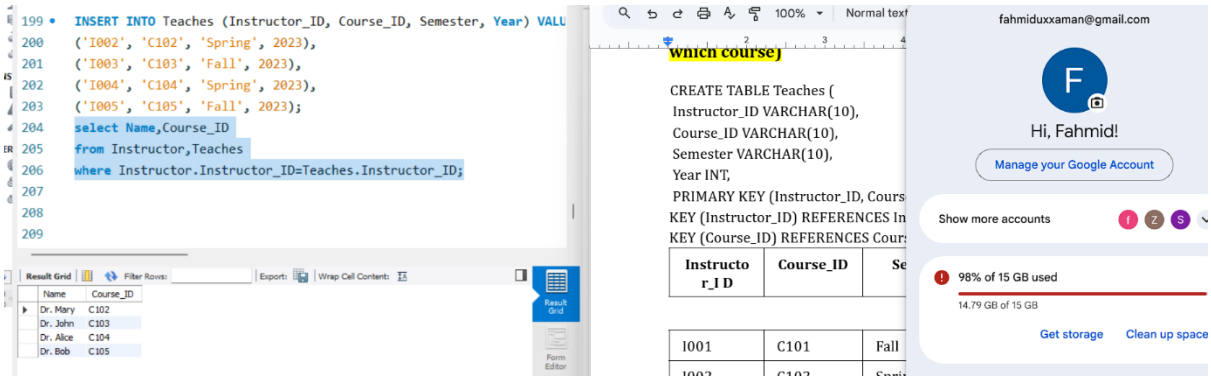
Output: This command s
order, and then by **Age** in

Name	Age	
Bob White	21	Phys
John Smith		Com
Dave Black		Engi
Alice Brown	22	Mat

Explanation : Here order by prioritize the first condition so checks the Majors in ascending order then if theres are equal Major then sorts according to the 2nd condition age in descending order here Arts major is lexicographically smallest so it comes first

and there are two Rows Containing Major Arts so the got sorted by age in descending order according to the 2nd condition

Full Code->
https://github.com/Fahmiduzzaman2003/DBMS_Lab/blob/deab0e18db5ae4a2bf74026c4dad949918834f7d/Assignment2.sql



```
INSERT INTO Takes (Student_ID, Course_ID, Semester, Year)
VALUES (7, 'C102', 'Spring', 2023);
```

Output

Action Output

#	Time	Action	Messag
69	21:32:48	select Instructor.name ,Teaches.Course_ID from Instructor natural jo...	4 row(s)
70	21:33:41	select Instructor.name ,Teaches.Semester from Instructor natural joi...	4 row(s)
71	22:21:14	drop table Teaches	0 row(s)
72	22:21:29	select *from Teaches LIMIT 0, 10000	Error Co
73	22:24:37	INSERT INTO Takes (Student_ID, Course_ID, Semester, Year) VA...	Error Co

For example, if we try to insert a row into the **Takes** table that r
non-existing student, the command will fail due to the foreign k
constraint.

INSERT INTO Takes (Student_ID, Course_ID, Semester, Year)
VALUES (7, 'C102', 'Spring', 2023);

This will fail because **Student_ID = 7** does not exist in the **Stude**

16. SELECT Command

SELECT name
FROM instructor;

Output: This command selects the **name** column from the **Instructor** table.

Name