MAI172: Advance Database Technologies

Register Number: 2448513

Name: Deshmukh Pratik Bhushanrao

Experiment Number and Name: 2: Use of integrity constraints and referential integrity

Date: 18/07/2024 Time: 9.45 to 11.45

Scenario:

Consider an application for an educational institute. Every department has several instructors but an instructor can be associated with only one department. One among the instructors would act as head of the department. Instructors are allocated to take classes in multiple courses. There are many instructors for a given course. Various relations and their sample data are provided below:

Department(DeptId,DName,HODId)

DeptId	DName	HODId
1	Electronics	11
2	Mechanical	12

Instructor(InstrId,Name,DeptId,EMail)

InstrId	Name	DeptId	EMail
11	John	1	aaa@zz.com
12	Mark	2	bbb@zz.com
13	Jane	1	ccc@zz.com
14	Joe	2	ddd@zz.com

Allocation (CId,InstrId)

CId	InstrId
101	11
101	13
102	11
102	12
103	14
103	12

Course (CId,CName,Credit)

CId	CName	Credit
101	Microprocessors	5
102	Programming	3
103	Thermodynamics	3

Tasks:

1. **Create** a table as per the format given above and identify the candidate key, primary key and foreign keys of each entity and create.

Queries for creating tables:

For Department Table:

create table Department (DeptId int primary key, DName varchar (20), HODId int);

Output:

	Field	Type	Null	Key	Default	Extra
▶	DeptId	int	NO	PRI	NULL	
	DName	varchar(20)	YES		NULL	
	HODId	int	YES		NULL	

For Instructor Table:

create table Instructor (InstrId int primary key, Name varchar (20), DeptId int, Email varchar (20), foreign key (DeptId) references Department (DeptId));

Output:

	Field	Type	Null	Key	Default	Extra
▶	InstrId	int	NO	PRI	NULL	
	Name	varchar(20)	YES		NULL	
	DeptId	int	YES	MUL	NULL	
	Email	varchar(20)	YES		NULL	

For Course Table:

create table Course (CId int primary key, CName varchar (20), Credit int);

Output:

	Field	Type	Null	Key	Default	Extra
▶	CId	int	NO	PRI	NULL	
	CName	varchar(20)	YES		NULL	
	Credit	int	YES		NULL	

For Allocation Table:

create table Allocation (CId int references Course (CId), InstrId int references Instructor (InstrId));

Output:

	Field	Type	Null	Key	Default	Extra
▶	CId	int	YES	MUL	NULL	
	InstrId	int	YES	MUL	NULL	

Queries for inserting values in tables:

For Inserting Data In Department Table:

insert into Department values (1, "Electronics", 11), (2, "Mechanical", 12);

Output:

	DeptId	DName	HODId
▶	1	Electronics	11
	2	Mechanical	12
*	NULL	NULL	NULL

For Inserting Data In Instructor Table:

insert into Instructor values (11, "John", 1, "aaa@zz.com"),
(12, "Mark", 2, "bbb@zz.com"),
(13, "Jane", 1, "ccc@zz.com"),
(14, "Joe", 2, "ddd@zz.com");

Output:

	InstrId	Name	DeptId	Email
▶	11	John	1	aaa@zz.com
	12	Mark	2	bbb@zz.com
	13	Jane	1	ccc@zz.com
	14	Joe	2	ddd@zz.com
*	NULL	NULL	NULL	NULL

For Inserting Data In Course Table:

insert into Course values (101, "Microprocessor", 5),
(102, "Programming", 3),
(103, "Thermodynamics", 3);

Output:

	CId	CName	Credit
▶	101	Microprocessor	5
	102	Programming	3
	103	Thermodynamics	3
*	NULL	NULL	NULL

For Inserting Data In Allocation Table:

insert into Allocation values (101, 11), (101, 13), (102, 11), (102, 12), (103, 14), (103, 12);

Output:

	CId	InstrId
▶	101	11
	101	13
	102	11
	102	12
	103	14
	103	12

2. **Display** Course Name, Instructor Name and Department Name all attributes in single table.

   Query:

   select c.CName as Course_Name, i.Name as Instructor_Name, d.DName as Department_Name from Allocation a join Course c on a.CId = c.Cid join Instructor i on a.InstrId = i.InstrId join Department d on i.DeptId = d.DeptId;

   Output:

Course_Name	Instructor_Name	Department_Name
Microprocessor	John	Electronics
Programming	John	Electronics
Programming	Mark	Mechanical
Thermodynamics	Mark	Mechanical
Microprocessor	Jane	Electronics
Thermodynamics	Joe	Mechanical