
Use Google to search on the questions
Use Google Doc to cheat with my friend and we will got C togther
imgflip.com

Databases - Tutorial 05
From ERD to Relational Schema

Hamza Salem - Innopolis University

Contents

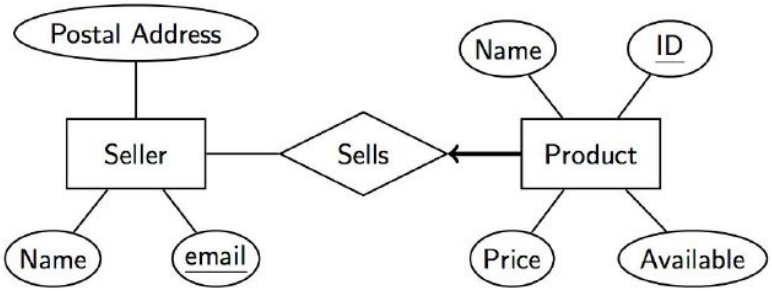
- From ERD to Relational Schema

ERD to Relational Schema

Step 1. Convert all entity sets into tables:

- Entity set name -> Table name

- Entity set attributes -> Table columns



Seller

name	email	address
...	...	...

Product

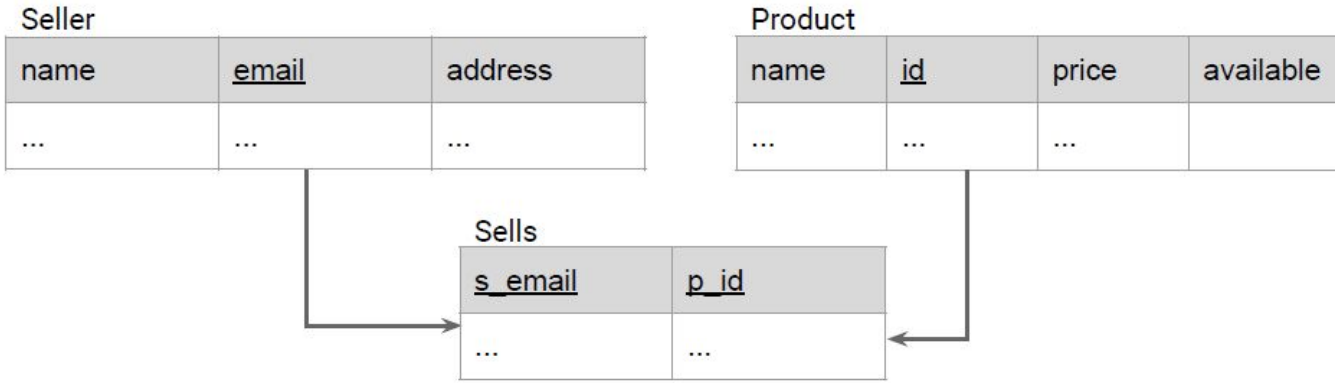
name	id	price	available
...	...	...	

ERD to Relational Schema

Step 2. Create relationships between entity sets:

- Each product has exactly one seller and each seller may sell multiple products

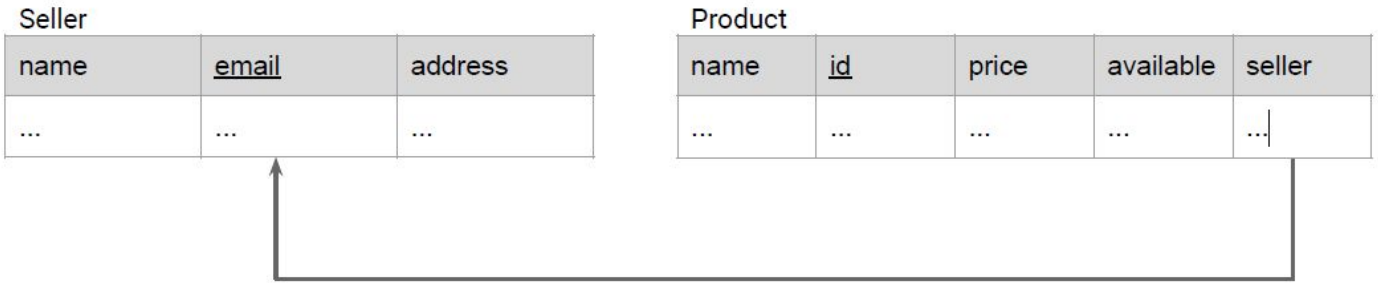
Option 1: introduce a new table that will hold correspondence between products and sellers



ERD to Relational Schema

Step 2. Create relationships between entity sets:

- Each product has exactly one seller and each seller may sell multiple product



ERD to Relational Schema

Step 3. Implement the schema in SQL

- You may use http://sqlfiddle.com/ if you don’t have a local installation

```
CREATE TABLE Seller (
name VARCHAR(30),
email VARCHAR(30) PRIMARY KEY,
address VARCHAR(200)
);
```

```
CREATE TABLE Product (
name VARCHAR(60),
id INTEGER PRIMARY KEY,
price INTEGER,
available INTEGER,
seller VARCHAR(30)
);
```

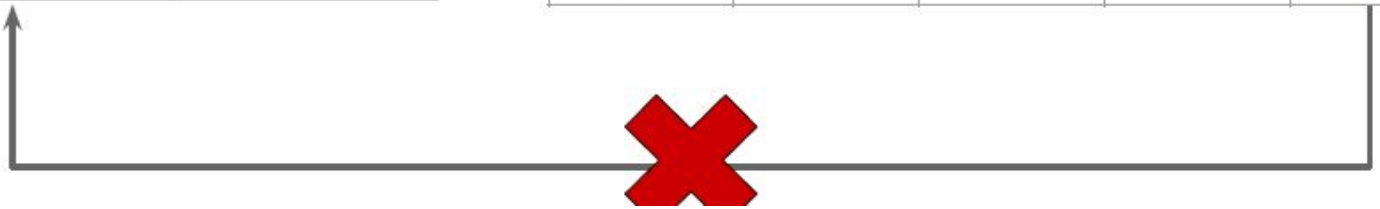
ERD to Relational Schema

Seller

name	email	address
Bulls Inc.	pr@bulls.com	123 Street

Product

name	id	price	available	seller
Teapot	1	100	600	b@b.com



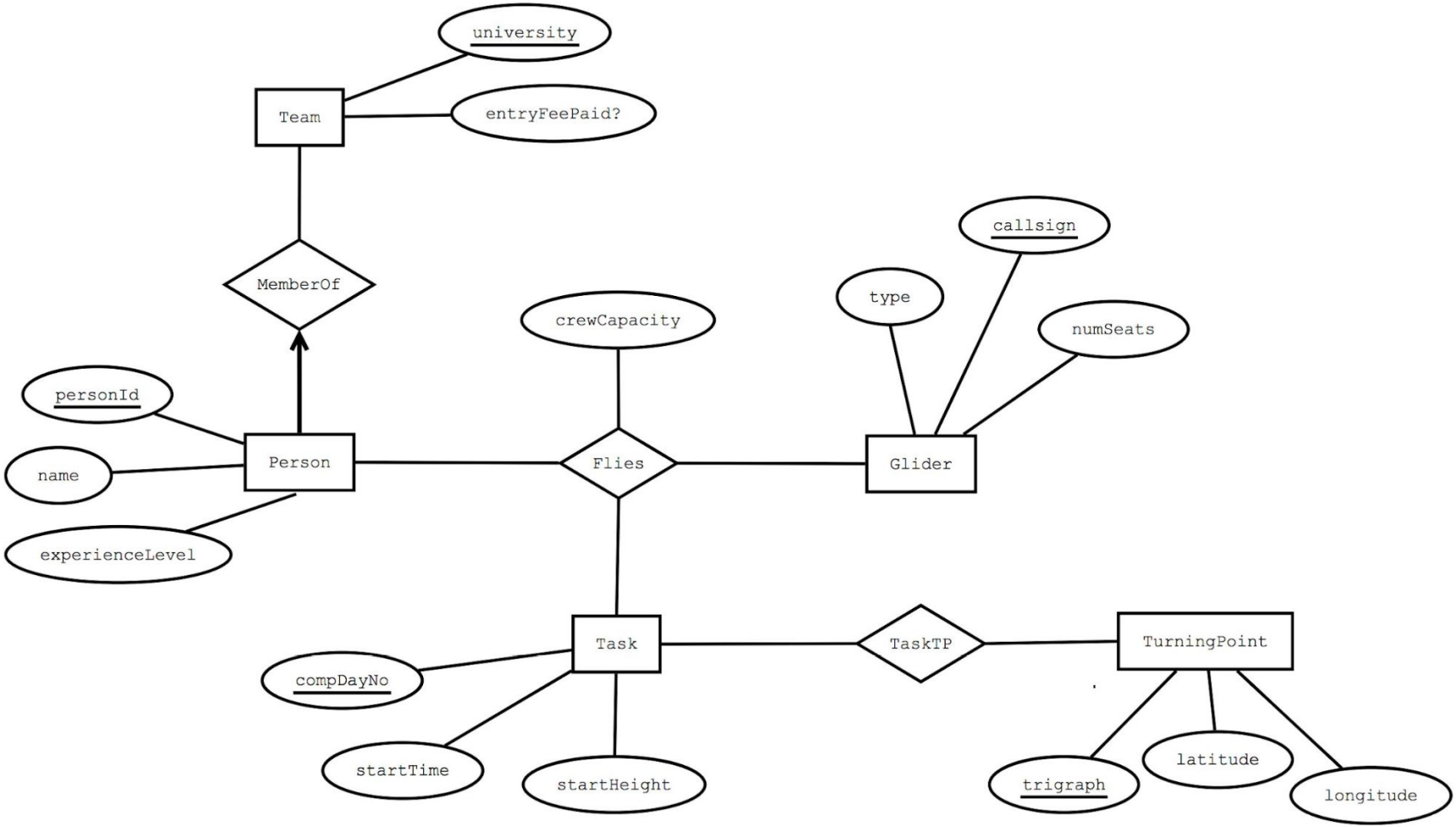
ERD to Relational Schema

Step 3. Implement the schema in SQL

- You may use http://sqlfiddle.com/ if you don’t have a local installation

```
CREATE TABLE Seller (
name VARCHAR(30),
email VARCHAR(30) PRIMARY KEY,
address VARCHAR(200)
);
```

```
CREATE TABLE Product (
name VARCHAR(60),
id INTEGER PRIMARY KEY,
price INTEGER,
available INTEGER,
seller VARCHAR(30) REFERENCES Seller(email)
);
```


university
Team
entryFeePaid?
MemberOf
callsign
type
crewCapacity
numSeats
personId
Person
Flies
Glider
name
experienceLevel
Task
TaskTP
TurningPoint
compDayNo
startTime
startHeight
trigraph
latitude
longitude

As always, let’s first begin with entities

```
create table Team (
university varchar(50),
entryFeePaid integer,
primary key (university)
);
create table Person (
personId integer,
name varchar(30),
experienceLevel varchar(15),
primary key (personId),
);
```

```
create table Glider (
callsign varchar(5),
type varchar(10),
numSeats integer,
primary key (callsign)
);
create table Task (
compDayNo integer,
startTime timestamp,
startHeight integer,
primary key (compDayNo)
);
```

```
create table TurningPoint
(
trigraph varchar(3),
latitude varchar(10),
longitude varchar(10),
primary key (trigraph)
)
```

Let’s refine these entities to include one-to-many relationships (MemberOf):

```
create table Team (
university varchar(50),
entryFeePaid integer,
primary key (university)
);
create table Person (
personId integer,
name varchar(30),
experienceLevel varchar(15),
university varchar(50) not
null,
primary key (personId),
foreign key (university)
references Team
);
```

```
create table Glider (
callsign varchar(5),
type varchar(10),
numSeats integer,
primary key (callsign)
);
create table Task (
compDayNo integer,
startTime timestamp,
startHeight integer,
primary key (compDayNo)
);
```

```
create table TurningPoint
(
trigraph varchar(3),
latitude varchar(10),
longitude varchar(10),
primary key (trigraph)
)
```

Next step is to model many-to-many binary relationships TaskTP. To do so we need to introduce a separate table that will connect entities:

```
create table TaskTP (
compDayNo integer,
trigraph varchar(3),
primary key (compDayNo, trigraph),
foreign key (compDayNo) references Task,
foreign key (trigraph) references TurningPoint
);
```

Ternary many-to-many relationship Flies is modelled in a similar way (note the inclusion of an attribute):

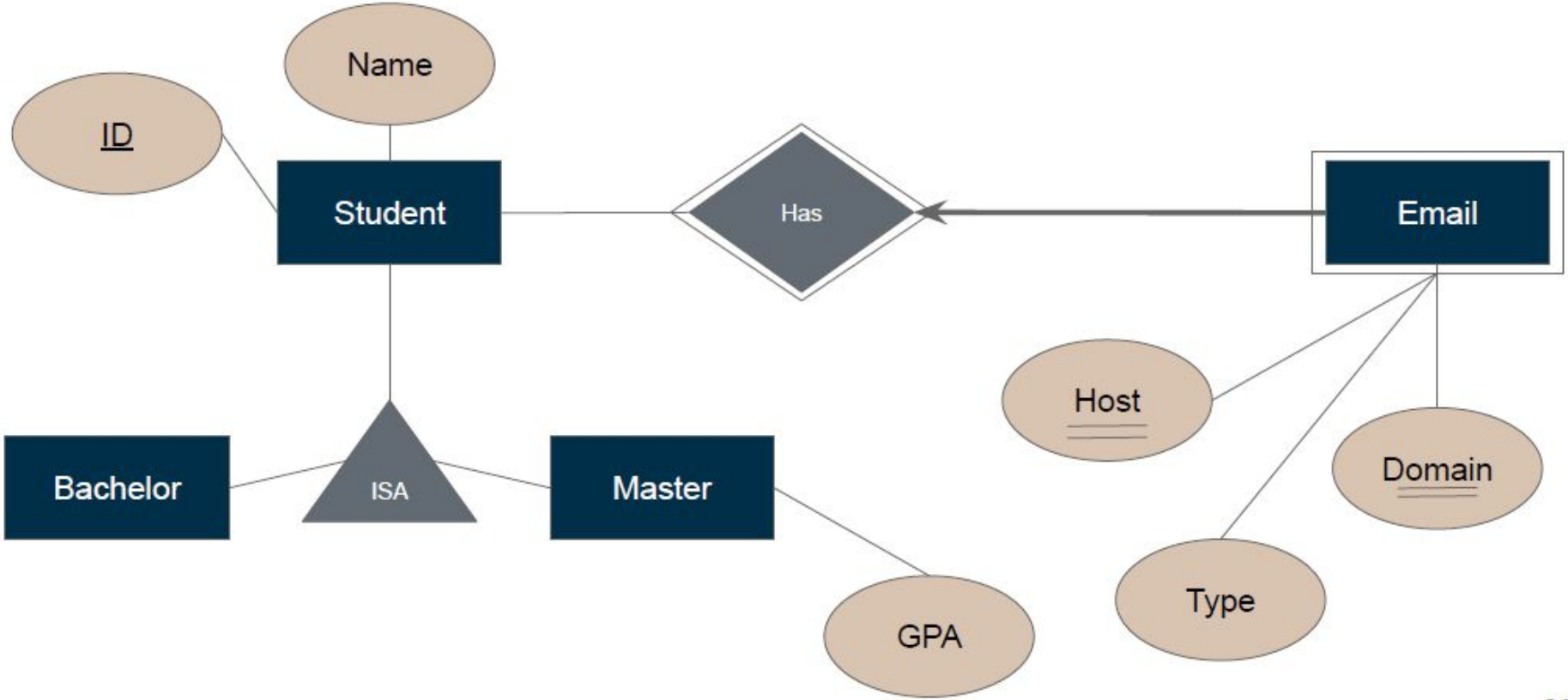
```
create table Flies (
personId integer,
callsign varchar(5),
compDayNo integer,
crewCapacity integer,
primary key (personId, callsign, compDayNo),
foreign key (personId) references Person,
foreign key (callsign) references Glider,
foreign key (compDayNo) references Task
);
```

To model one-to-many relationship, first approach - primary key is from the “many values” table, foreign keys reference both.

```
create table MemberOf (
personId integer,
university varchar(15),
primary key (personId),
foreign key (personId) references Person,
foreign key (university) references Team
);
```

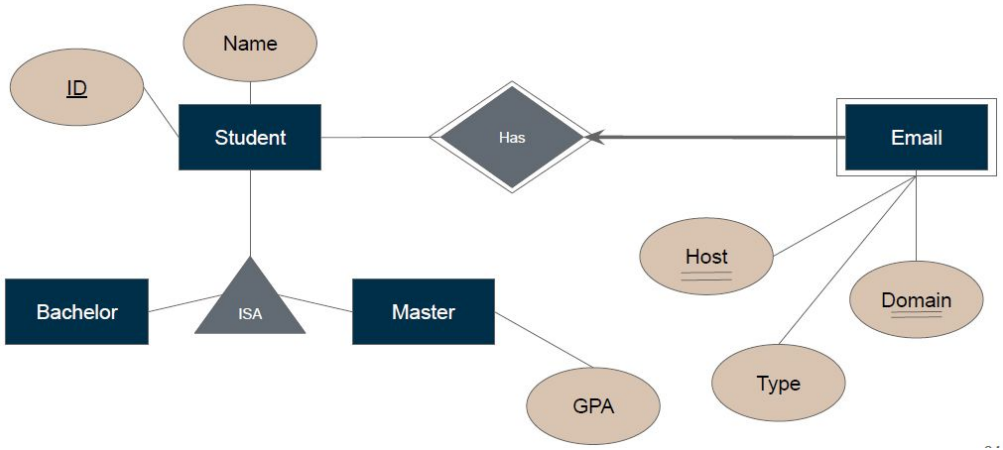
To model one-to-many relationship, second approach - reference on the main table.

```
create table PersonsInTeam (
personId integer,
name varchar(30)
experienceLevel integer
university varchar(15),
primary key (personId),
foreign key (university) references Team
);
```


Name
ID
Student
Has
Email
Host
Domain
Type
Bachelor
ISA
Master
GPA

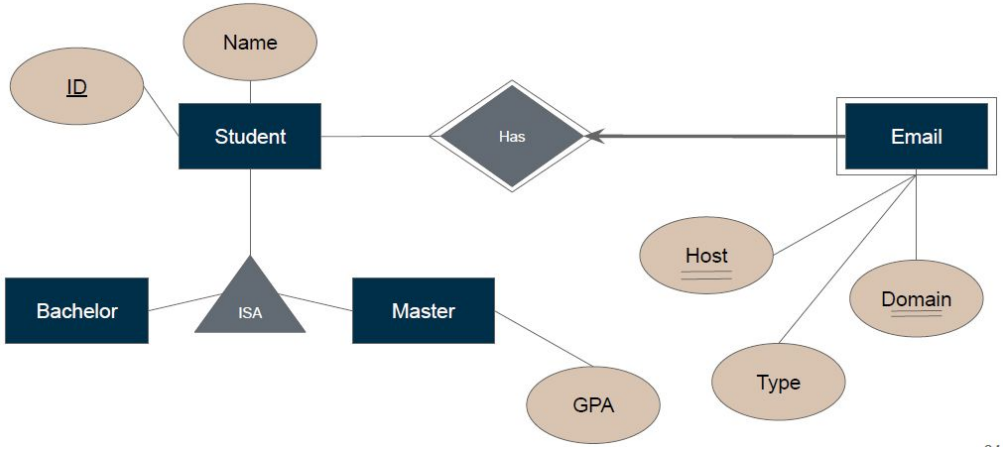
Let’s model ISA relationship. To do so we need to declare tables for the superclass and for each subclass with superclass primary key and the subclass extra attributes

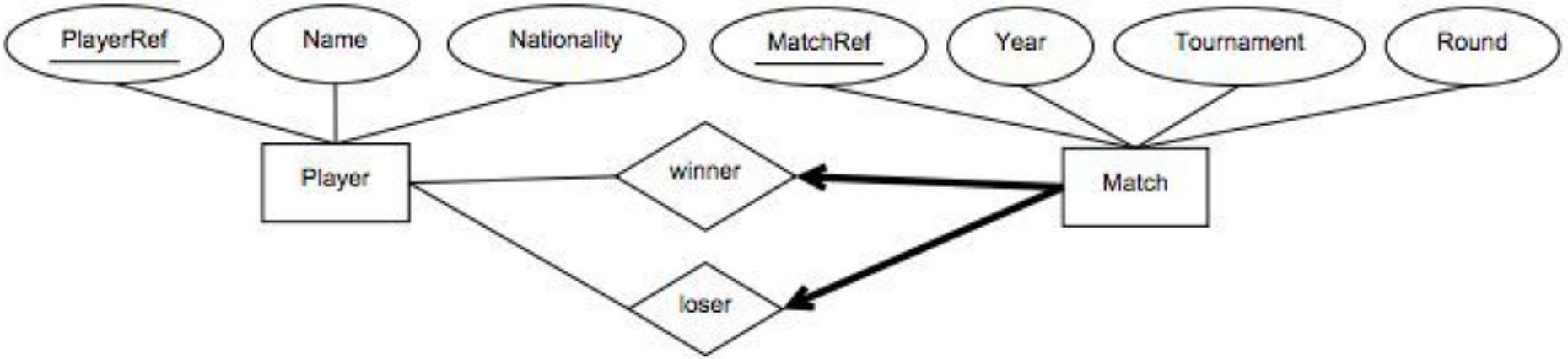
```
create table Student (
ID integer,
Name varchar(30),
primary key ID,
);
create table Masters (
ID integer,
GPA integer,
primary key (ID),
foreign key (ID) references Student
);
```

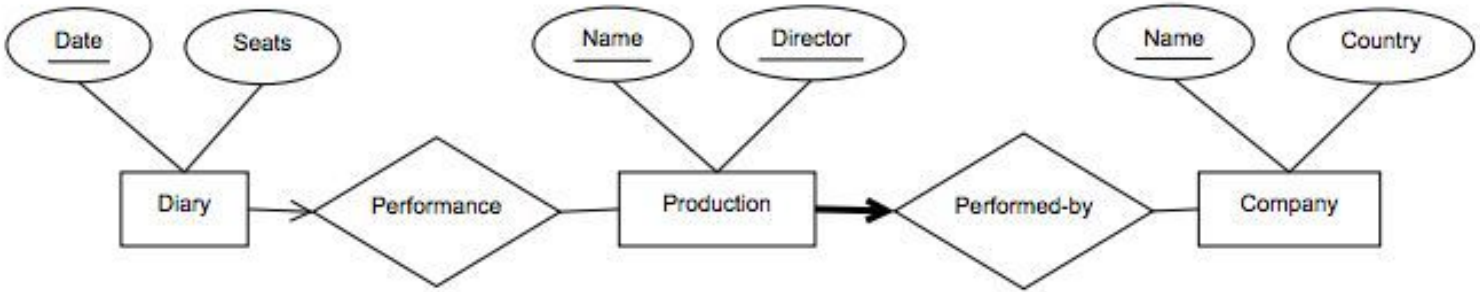


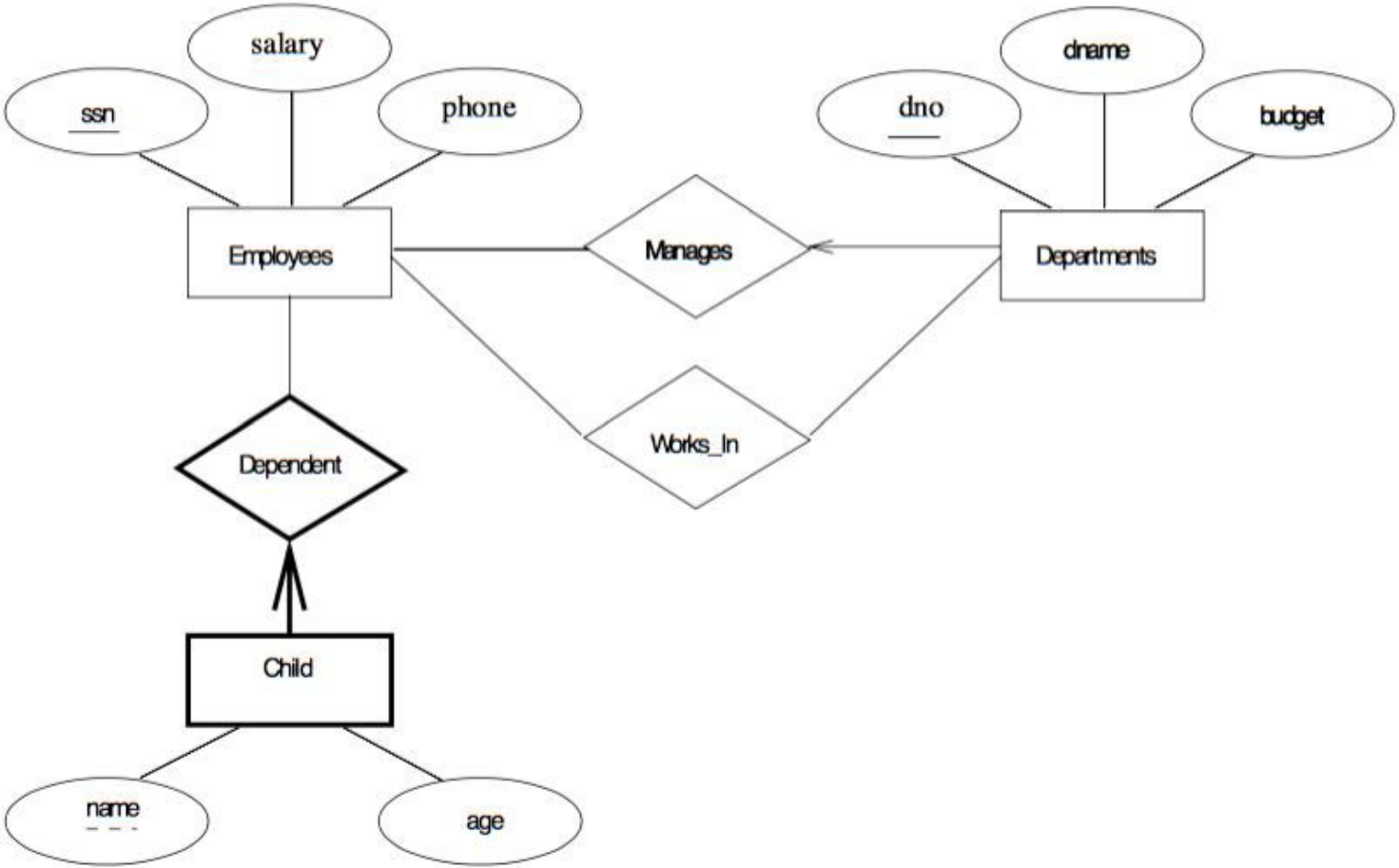
To model a weak entity we need to add fields for the primary key attributes of the identifying owner, declare a foreign key constraint and automatically delete any tuples in the table for which there are no owners

```
create table Email (
Host varchar(30),
Domain varchar(30),
Type varchar (30)
ID integer,
primary key (host, domain, ID),
foreign key (ID) references Student on delete
cascade
);
```




PlayerRef
Name
Nationality
MatchRef
Year
Tournament
Round
Player
winner
Match
loser


Date
Seats
Diary
Performance
Name
Director
Production
Performed-by
Name
Country
Company


salary
ssn
phone
dname
dno
budget
Employees
Manages
Departments
Works_In
Dependent
Child
name
age