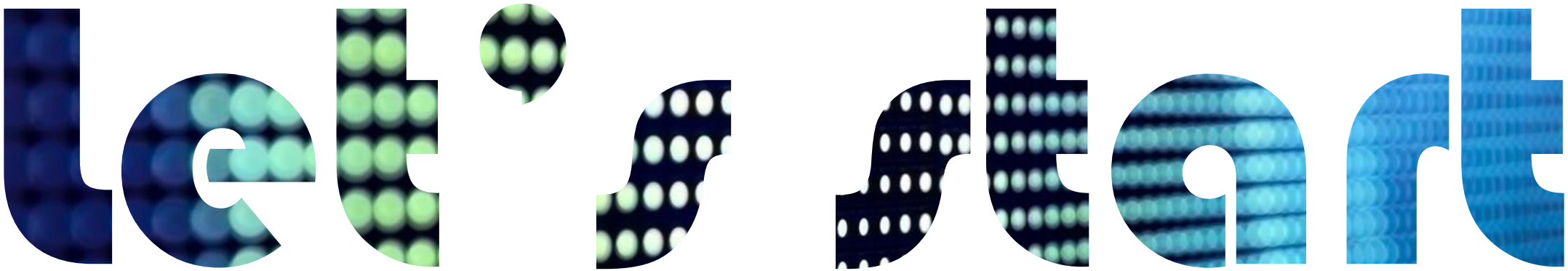
Lancaster University



Lancaster University

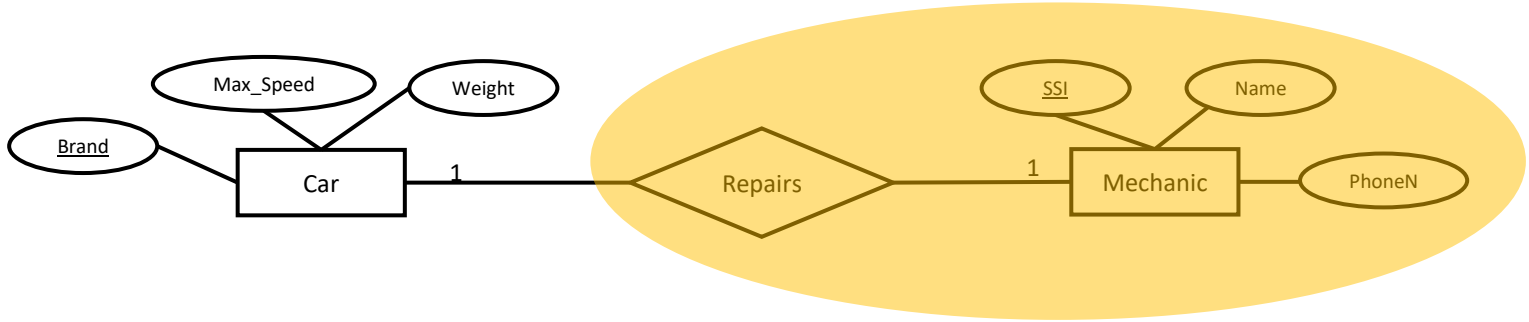
SCC.201
Database Management Systems

2023 - Week 3 – Relational Model to SQL

Uraz C Turker & Ricki Boswell

Recall

- 1 to 1 relations



Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

SSI	Name	Phone_Number	Brand
87542702	Tom	75315567	Toyota..
68201937	Uraz	75335521	Hyundai.
23139827	Nick	75315544	BMW..

Car(Brand:*string*,Weight:*integer*,Length:*real*,Max_Speed:*integer*)

Mec_Rep(SSI:*string*,Name:*string*,Phone:*string*,Brand:*string*)

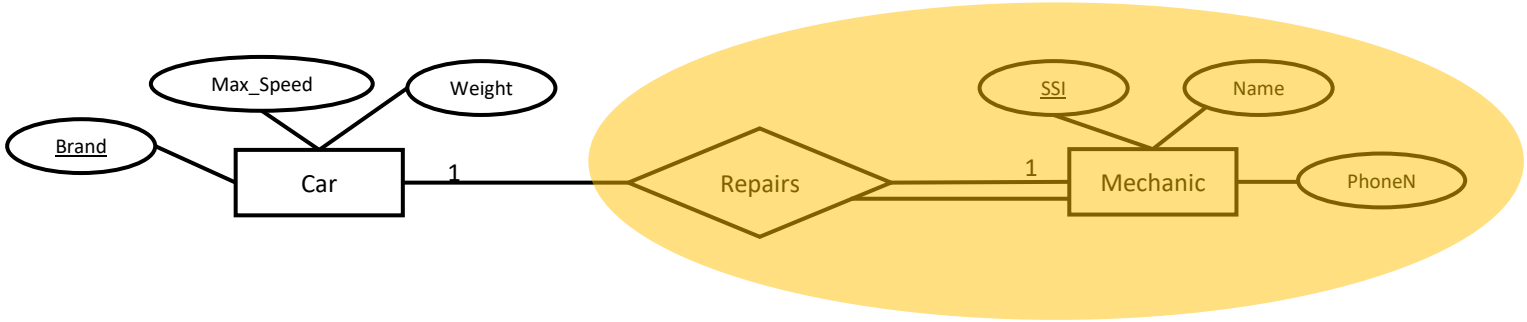
IC for CAR: Primary key Brand

IC's for Mec_Rep : Primary key SSI, Foreign key Brand referencing CAR.

On deleting car tuple SET NULL/DEFAULT, Brand is UNIQUE.

Recall

- 1 to 1 relations



Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

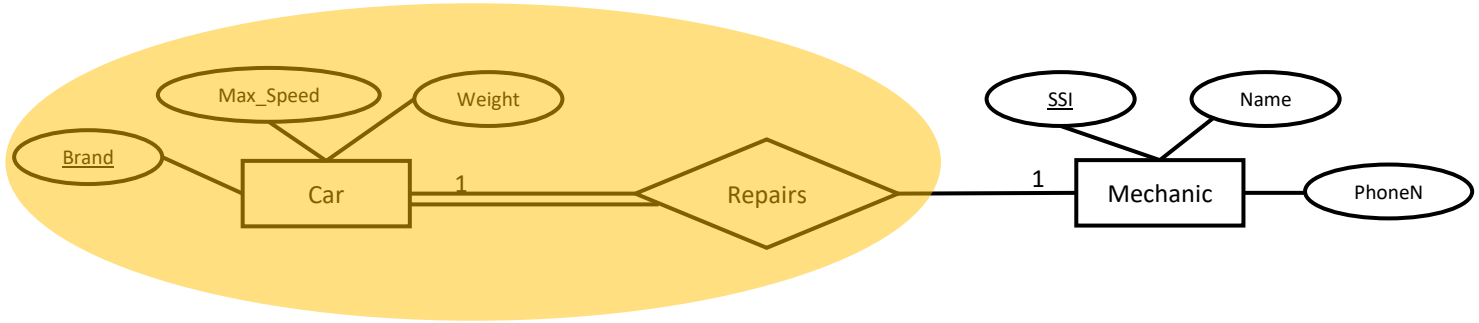
SSI	Name	Phone_Number	Brand
87542702	Tom	75315567	Toyota..
68201937	Uraz	75335521	Hyundai.
23139827	Nick	75315544	BMW..

Car(Brand:string,Weight:integer,Length:real,Max_Speed:integer)
Mec_Rep(SSI:string,Name:string,Phone:string,Brand:string)
IC for CAR: Primary key Brand

IC's for Mec_Rep: Primary key SSI, Foreign key Brand referencing CAR.
On delete CASCADE/REJECT, BRAND CANNOT BE NULL, Brand is UNIQUE

Recall

- 1 to 1 relations



Brand	Weight	Length	Max_Speed	SSI
BMW 3.21	1400	3.21	200	87542702
Toyota_Corolla	1300	3.18	200	68201937
Hyundai E.GLS	1400	3.16	210	23139827

SSI	Name	Phone_Number
87542702	Tom	75315567
68201937	Uraz	75335521
23139827	Nick	75315544

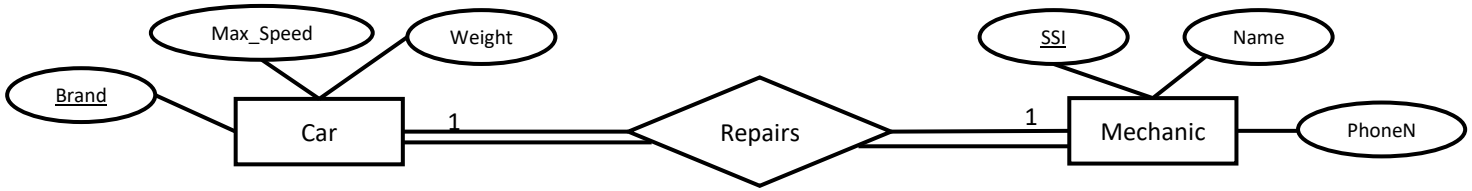
Car_Rep(Brand:*string*,Weight:*integer*,Length:*real*,Max_Speed:*integer*,SSI: *string*)

IC's for Car_Rep: Primary key Brand, Foreign key SSI referencing Mec. On delete CASCADE/REJECT, SSI CANNOT BE NULL, SSI is UNIQUE

Mec (SSI:*string*,Name:*string*,Phone:*string*) IC for Mec: Primary key SSI

Recall

- 1 to 1 relations



Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

Car(Brand:string,Weight:integer,Length:real,Max_Speed:integer)

SSI	Brand
87542702	Toyota..
68201937	Hyundai.
23139827	BMW..

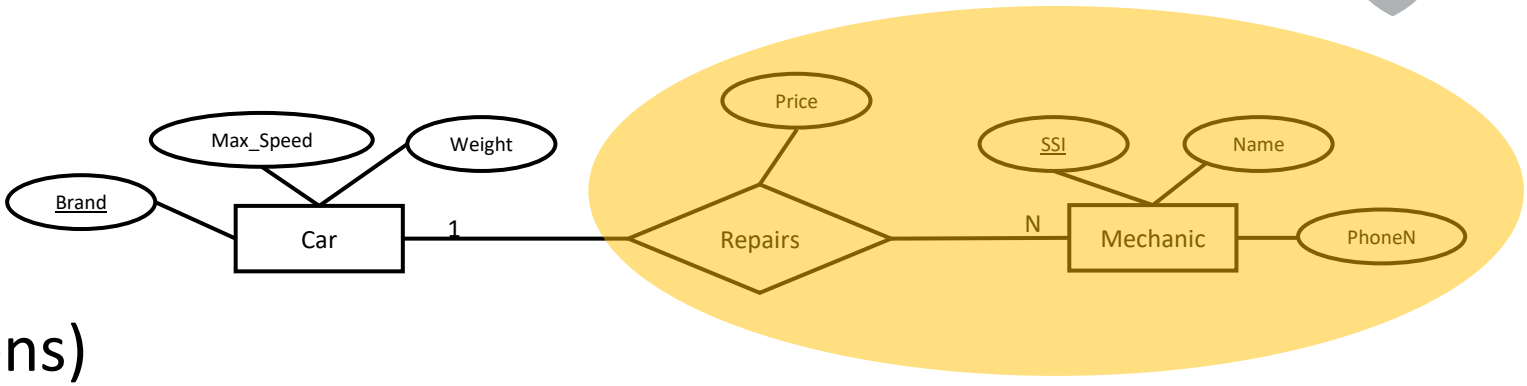
Car_Rep_Mec(Brand:*string*, SSI: *string*)

SSI	Name	Phone_Number
87542702	Tom	75315567
68201937	Uraz	75335521
23139827	Nick	75315544

Mec (SSI:*string*,Name:*string*,Phone:*string*)

IC's for Car_Rep_Mec: Primary key SSI, Foreign Key SSI referencing Mec, Foreign Key Brand referencing Car, On delete CASCADE/Reject, BRAND CANNOT BE NULL, Brand is UNIQUE

Recall



- 1 to Many relations

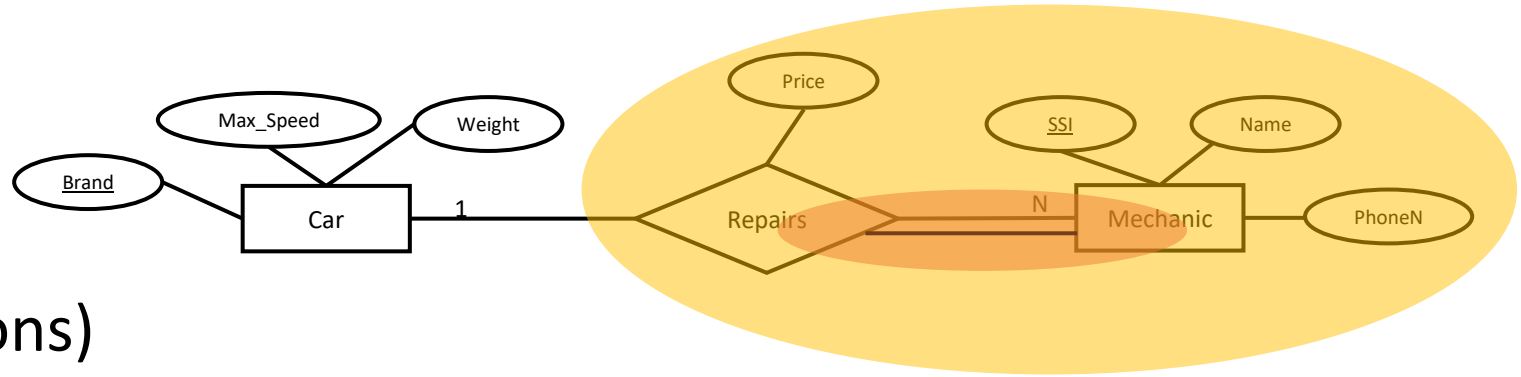
(Same for Many to 1 relations)

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

Car (Brand:*string*,Weight: *integer*, Length:*real*, Max_Speed:*integer*) IC: Primary key Brand.
Mec_R (Brand:*string*,Price: *integer*, SSI:*integer*, Name:*string*, Phone_Number:*string*) IC's: Primary key SSI, Foreign Key Brand referencing Car.

Recall



- 1 to Many relations

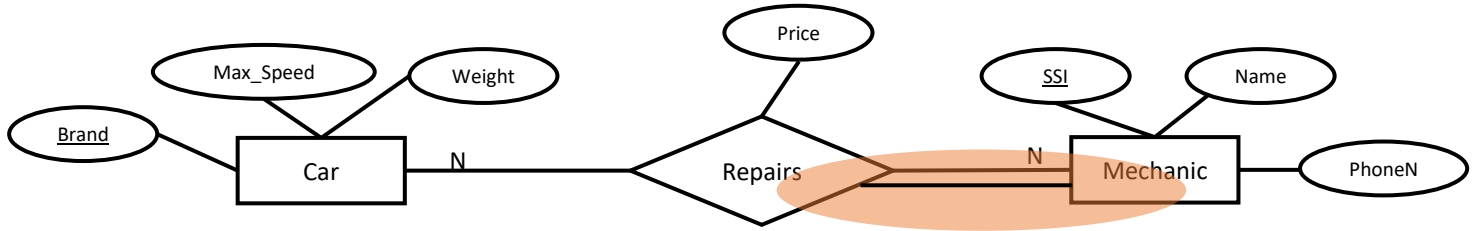
(Same for Many to 1 relations)

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544
BMW 3.21	11	23761281	Alex	73828732

Car (Brand:*string*,Weight: *integer*, Length:*real*, Max_Speed:*integer*) IC: Primary key Brand.
Mec_R (Brand:*string*,Price: *integer*, SSI:*integer*, Name:*string*, Phone_Number:*string*) IC's: Primary key SSI, Foreign Key Brand referencing Car, On delete CASCADE/REJECT, BRAND CANNOT BE NULL

Recall



- Many to Many (N-N, N-M, X-Y,….)

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

Car (Brand:*string*,Weight: *integer*, Length:*real*, Max_Speed:*integer*) IC: Primary key Brand.

Price	SSI	Brand
10	87542702	BMW 3.21
23	68201937	Toyota_Corolla
12	23139827	Hyundai E.GLS

Rep(Price:*Integer*,SSI:*integer*,Brand:*string*).
IC's: Primary Key {SSI,BRAND}
Foreign Key SSI referencing Mec,
Foreign Key Brand referencing Car,
BRAND CANNOT NULL, ON DELETE CASCADE.

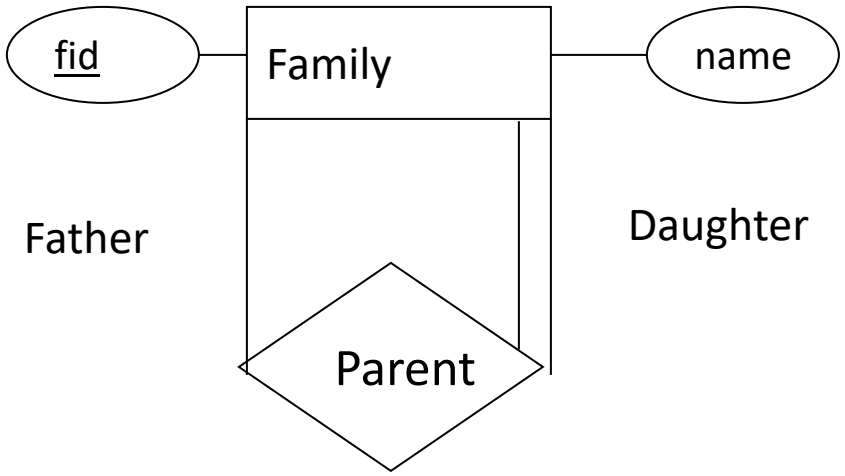
SSI	Name	Phone_Number
87542702	Tom	75315567
68201937	Uraz	75335521
23139827	Nick	75315544

Mec (SSI:*string*, Name:*string*, Phone_Number:*string*,) IC: Primary key SSI.

Recall

Family(fId:*integer*, Name:*string*, fatherOf:*integer*, daughterOf:*integer*)

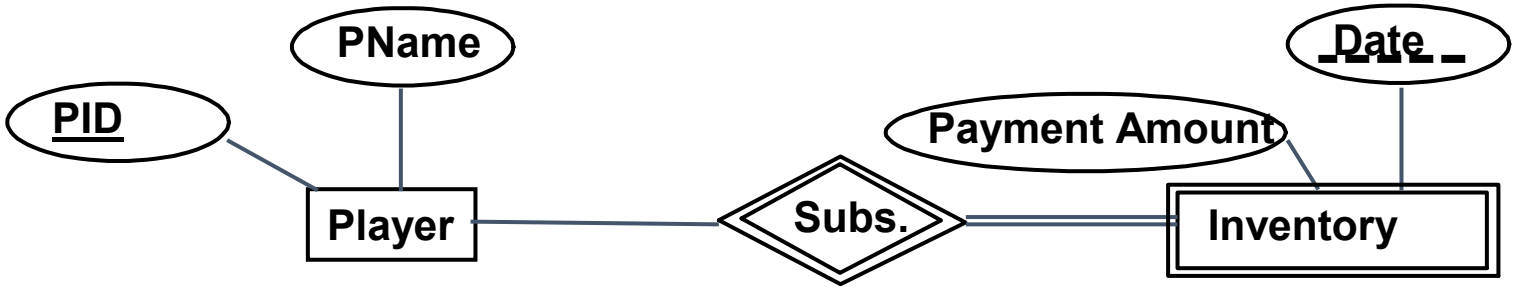
IC's: Primary key fId foreign key, fatherOf referencing Family. Foreign key daughterOf referencing Family, daughterOf CANNOT Null, daughterOf is UNIQUE.



Recall

Player(PID:integer,PName:string

- In a weak-entity set, the existence of an entity depends on the existence of an entity in the entity set in relation!

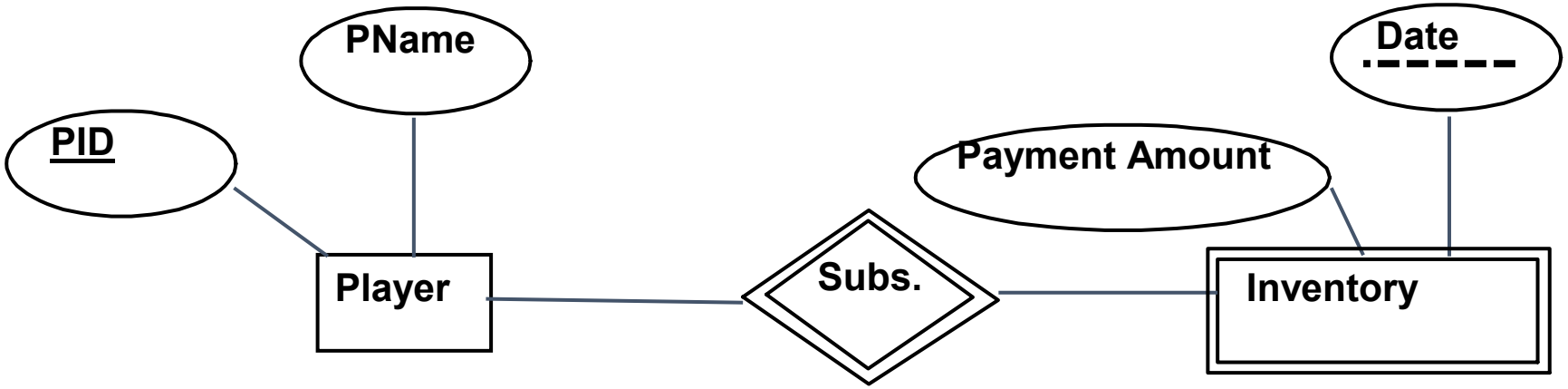


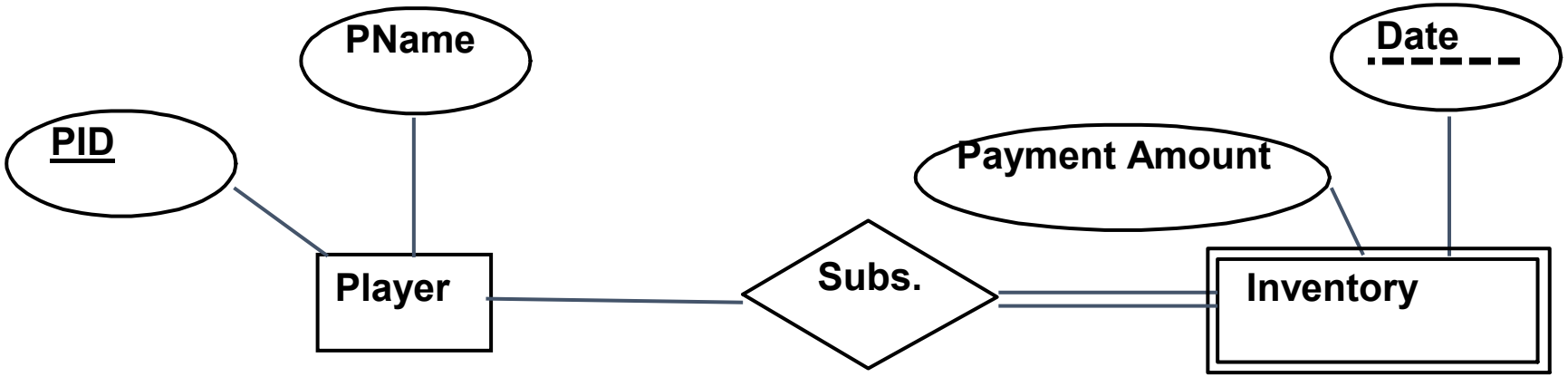
- If a player is deleted from the game server, the inventory information must be deleted.

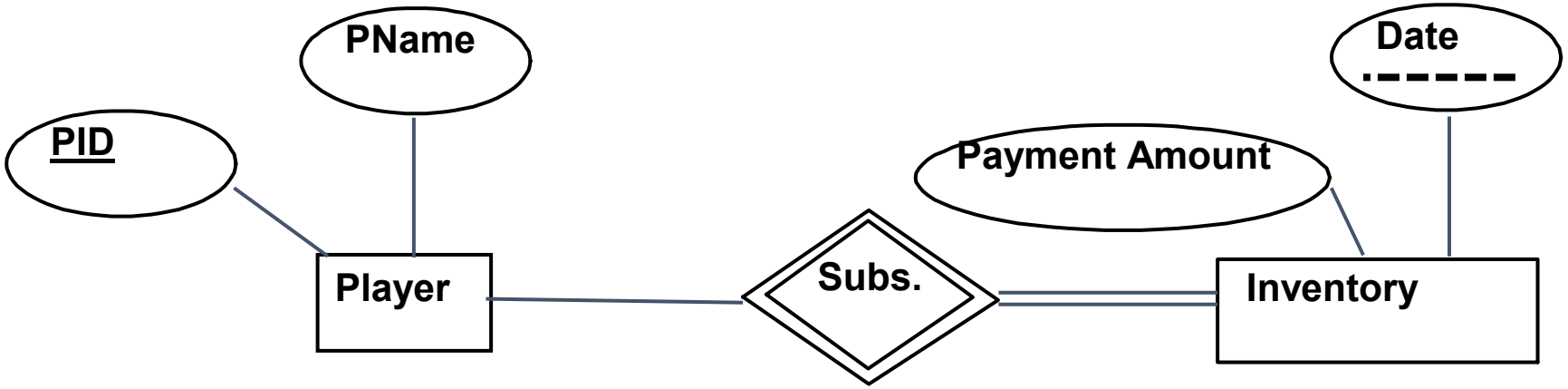
Inv_Subs(PID:integer,PaymentAmound:string,Date:date)

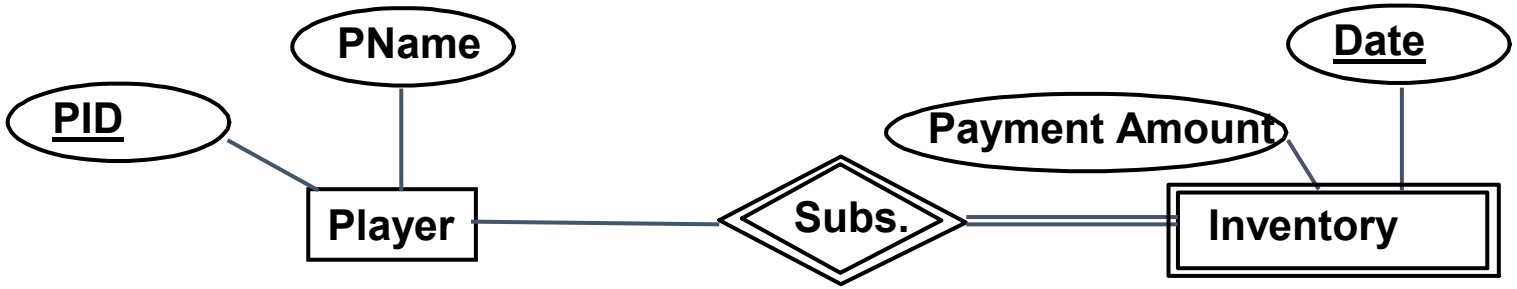
IC's: Primary key :{PID,DATE}, foreign key PID, referencing Player, on delete:cascade, PID cannot be null

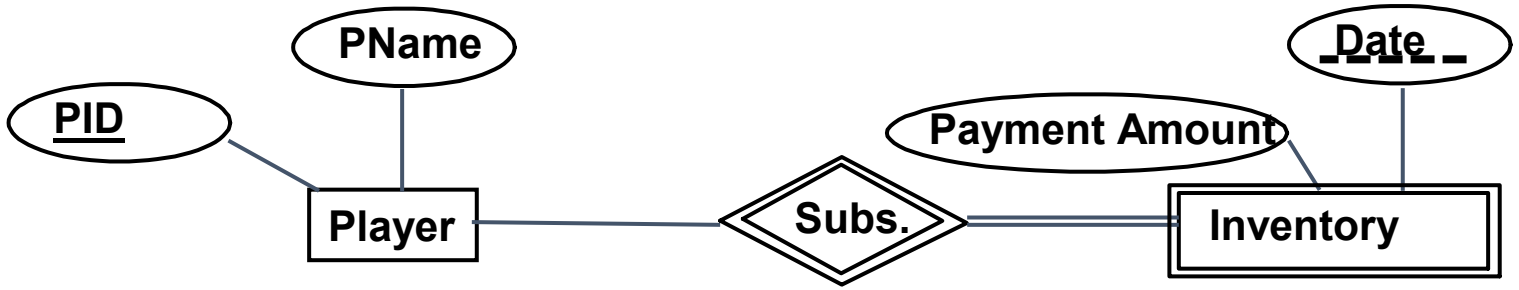
Lancaster University



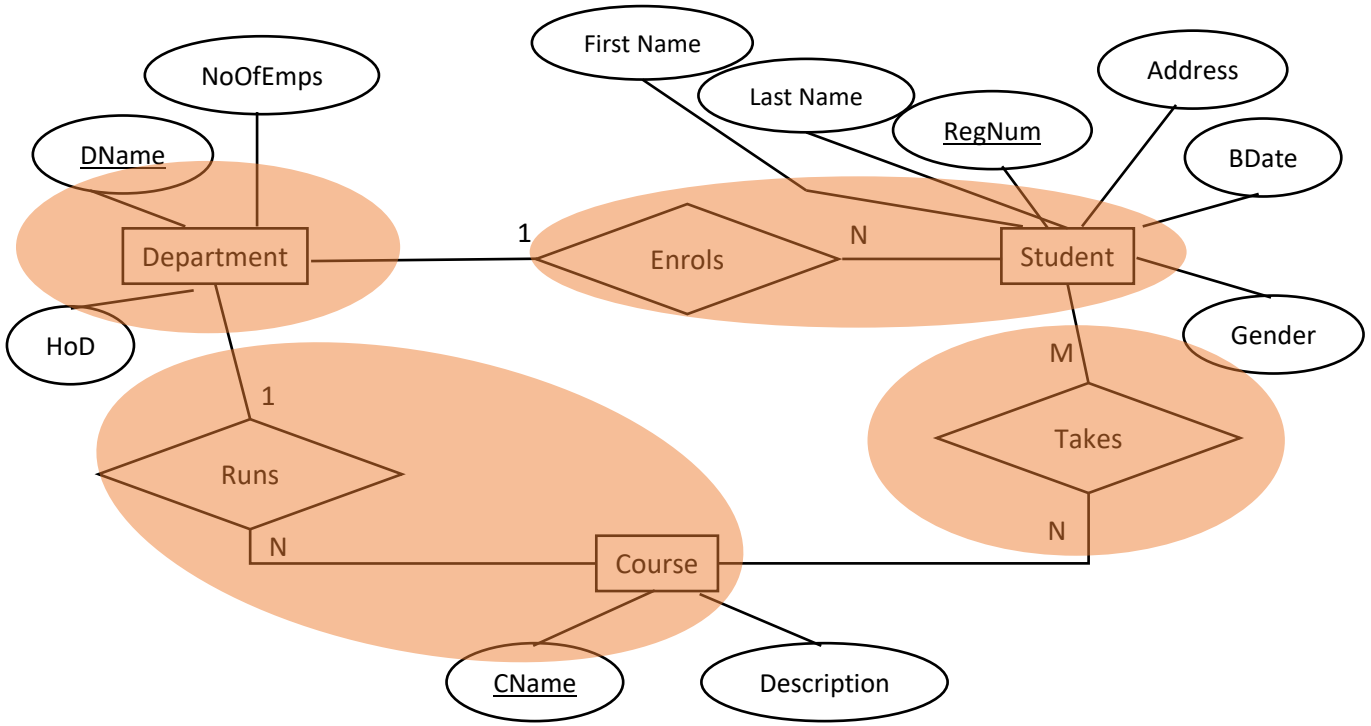

PName
Date
PID
Payment Amount
Player
Subs.
Inventory


PName
PID
Player
Subs.
Payment Amount
Date
Inventory


PName
Date
PID
Payment Amount
Player
Subs.
Inventory

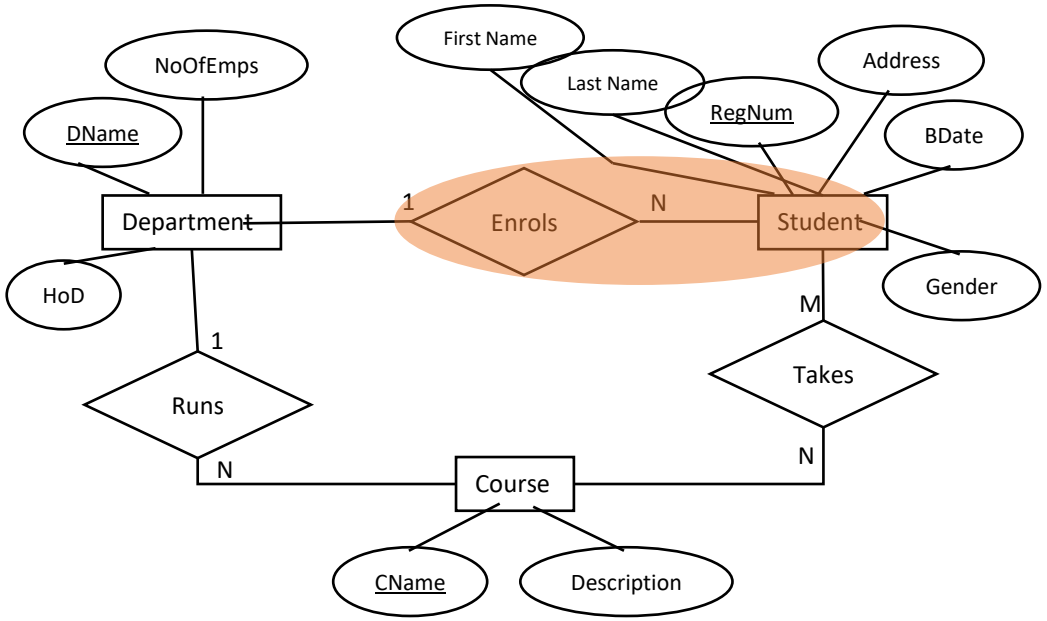

PName
PID
Player
Subs.
Payment Amount
Date
Inventory

Recall


First Name
NoOfEmps
Last Name
Address
RegNum
DName
BDate
Department
1
Enrols
N
Student
HoD
Gender
M
1
Takes
Runs
N
N
Course
CName
Description

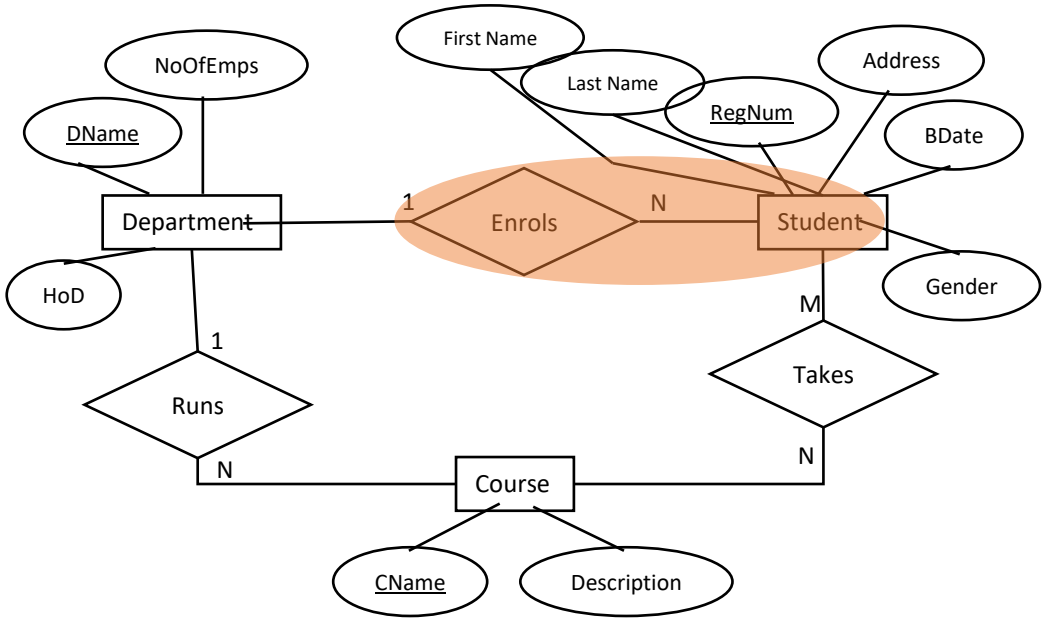
Exercise solutions

- Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName))
-
-
-



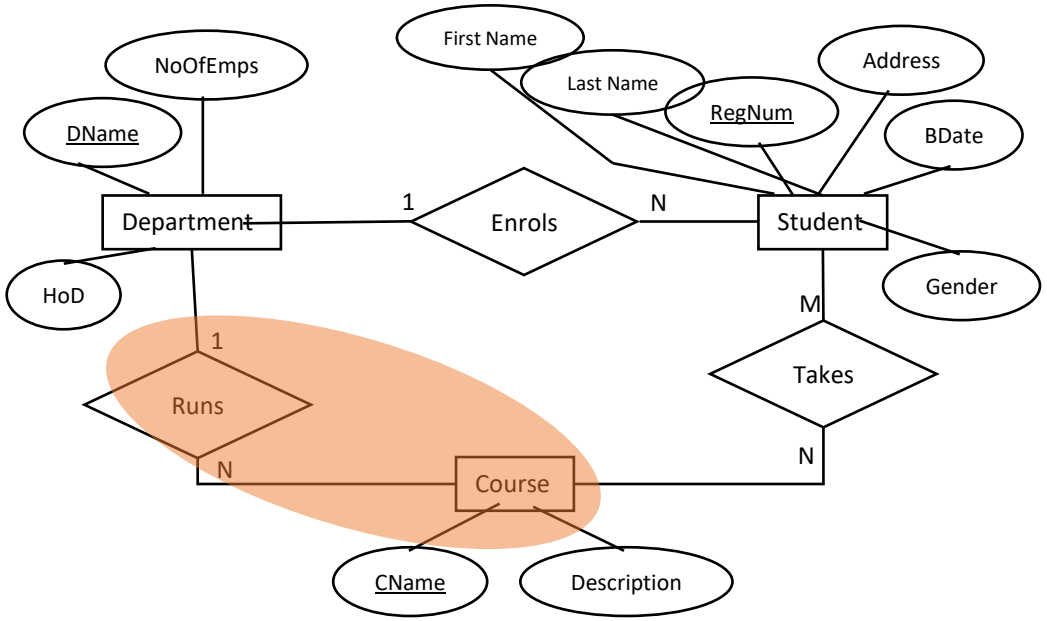
Exercise solutions

- Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName))
- StudentsEnrol(firstName TEXT, lastName TEXT, RegNumber INTEGER NOT NULL, Address TEXT, BDate TEXT, Gender TEXT, DepName TEXT, PRIMARY KEY (RegNumber), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL)
-
-



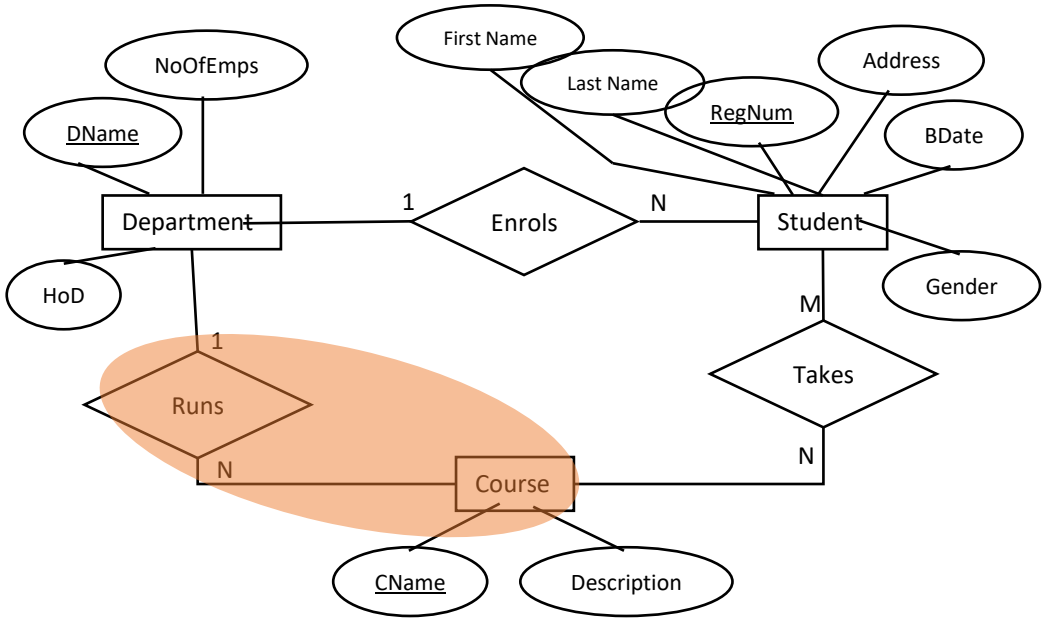
Exercise solutions

- Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName))
- StudentsEnrol(firstName TEXT, lastName TEXT, RegNumber INTEGER NOT NULL, Address TEXT, BDate TEXT, Gender TEXT, DepName TEXT, PRIMARY KEY (RegNumber), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL)
-
-



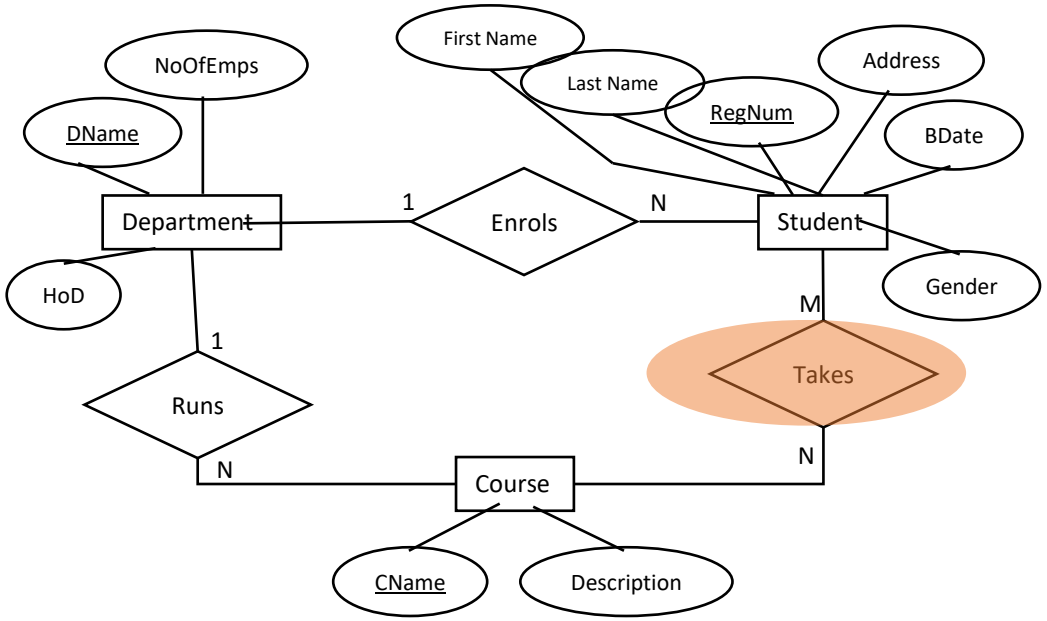
Exercise solutions

- Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName))
- StudentsEnrol(firstName TEXT, lastName TEXT, RegNumber INTEGER NOT NULL, Address TEXT, BDate TEXT, Gender TEXT, DepName TEXT, PRIMARY KEY (RegNumber), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL)
- CourseRuns(CName TEXT NOT NULL, Desc TEXT, DepName TEXT, PRIMARY KEY (CName), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL)
-



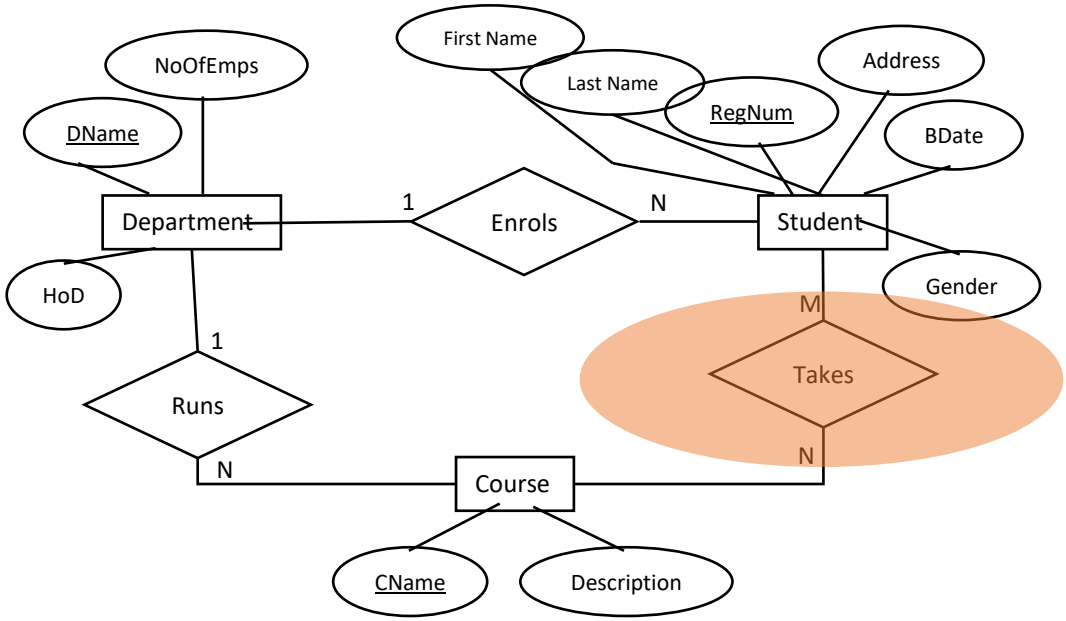
Exercise solutions

- Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName))
- StudentsEnrol(firstName TEXT, lastName TEXT, RegNumber INTEGER NOT NULL, Address TEXT, BDate TEXT, Gender TEXT, DepName TEXT, PRIMARY KEY (RegNumber), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL)
- CourseRuns(CName TEXT NOT NULL, Desc TEXT, DepName TEXT, PRIMARY KEY (CName), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL)
-



Exercise solutions

- Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName))
- StudentsEnrol(firstName TEXT, lastName TEXT, RegNumber INTEGER NOT NULL, Address TEXT, BDate TEXT, Gender TEXT, DepName TEXT, PRIMARY KEY (RegNumber), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL)
- CourseRuns(CName TEXT NOT NULL, Desc TEXT, DepName TEXT, PRIMARY KEY (CName), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL)
- StTakesCourse(CName TEXT NOT NULL, RegNumber INTEGER NOT NULL, PRIMARY KEY(CNAME,RegNumber), FOREIGN KEY (CName) REFERENCES CourseRuns(CName), FOREIGN KEY (RegNumber) REFERENCES StudentsEnrol(RegNumber))



The SQL Query Language

Lancaster University

- Developed by IBM (system R)
- Need for a standard since it is
- Standards:
  - SQL-86
  - SQL-89 (minor revision)
  - SQL-92 (major revision, curre
  - SQL-99 (major extensions)

```
SELECT S.rating, MIN (S.age)
FROM Sailors S
WHERE S.age >= 18
GROUP BY S.rating
HAVING COUNT (*) > 1
```

```
void myDisplay()
{
    game.display();
}

// Define the reshape function
void myReshape(int width, int height)
{
    game.reshape(width, height);
}

// Define the mouse click events
void myMouseClick(int button, int state, int x, int y)
{
    game.mouseClick(button, state, x, y);
}
```

Creating Relations in SQL

Lancaster University

```
CREATE TABLE Students
        (sid TEXT,
         name TEXT,
         login TEXT,
         age INTEGER,
         gpa REAL);
```

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Shero	shero@cs	18	3.2

Creates the Students' relation. Observe that each field's type (domain) is specified and enforced by the DBMS whenever tuples are added or modified.

Built-in data types varies interpreter to another.

TEXT, VARCHAR(Length), REAL, INTEGER, and BLOB are the most common data types.

Some interpreters accepts INT and INTEGER, some accepts DATA and BOOLEAN, but others don't.

Creating Relations in SQL

Lancaster University

As another example, the Enrolled table holds information about students' courses and grades.

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Shero	shero@cs	18	3.2

```
CREATE TABLE Enrolled
    (sid TEXT,
     cid TEXT,
     grade TEXT);
```

sid	cid	grade
53666	Carnatic101	C
53666	Reggae203	B
53650	Topology112	A
53666	History105	B

Primary and Candidate Keys in SQL

Lancaster University

"For a given student and course, there is a single grade in Enrolled."

sid	cid	grade
53666	Carnatic101	C
53666	Reggae203	B
53650	Topology112	A
53666	History105	B

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Shero	shero@cs	18	3.2

```
CREATE TABLE Enrolled
  (sid VARCHAR(20)
   cid  VARCHAR(20),
   grade VARHAR(2),
   PRIMARY KEY  (sid,cid) );
```

Primary and Candidate Keys in SQL

Lancaster University

Possibly many *candidate keys*, one of which is chosen as the *primary key*.

Create Enrolled table

"**Students can take only one course, and receive a single grade for that course;** further, no two students in a course receive the same grade."

```
CREATE TABLE Enrolled
  (sid VARCHAR(20)
   cid  VARCHAR(20),
   grade VARCHAR(2),
   PRIMARY KEY (sid),
   UNIQUE (cid, grade) );
```

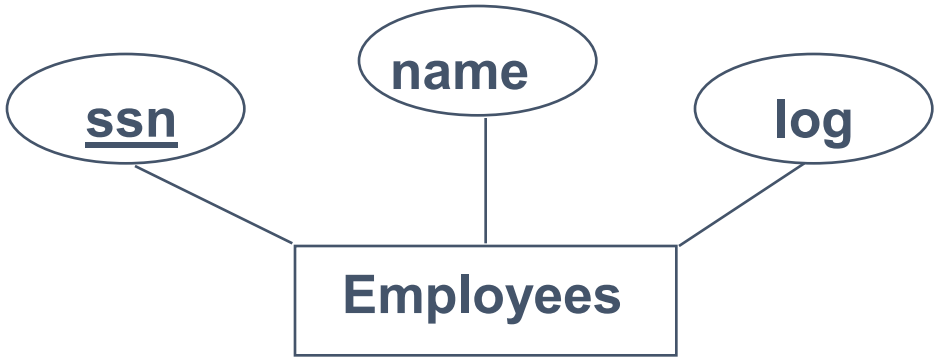
sid	cid	grade
53834	Carnatic101	C
53831	Reggae203	B
53650	Topology112	A
53666	History105	B

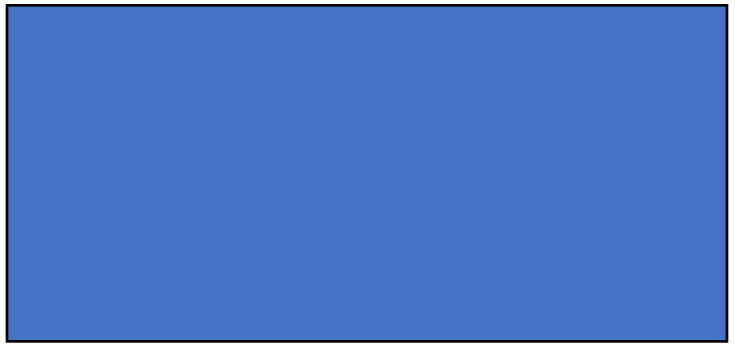
sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Shero	shero@cs	18	3.2

Logical DB Design: ER to Relational

Lancaster University

Entity sets to tables.

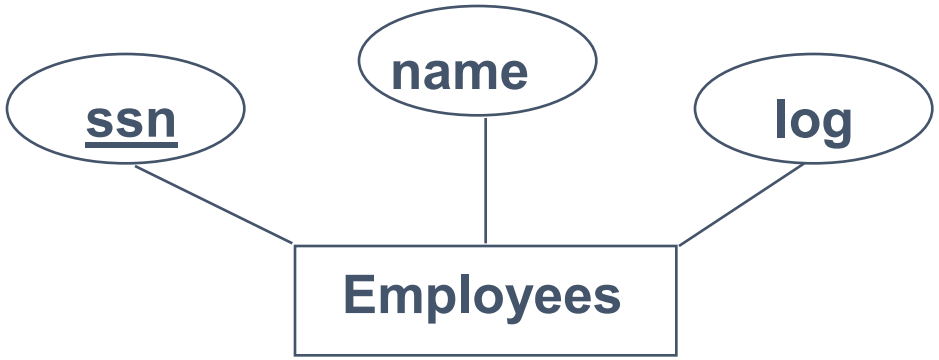




Logical DB Design: ER to Relational

Lancaster University

Entity sets to tables.

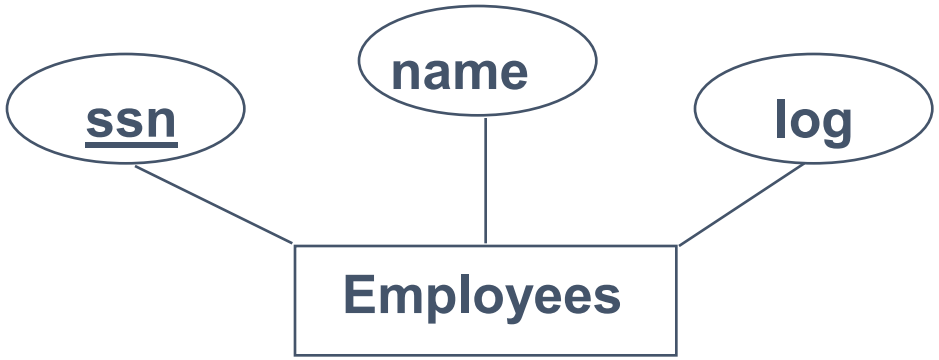



```
CREATE TABLE Employees
    (ssn VARCHAR(11),
    name VARCHAR(20),
    log  INTEGER,
```

Logical DB Design: ER to Relational

Lancaster University

Entity sets to tables.

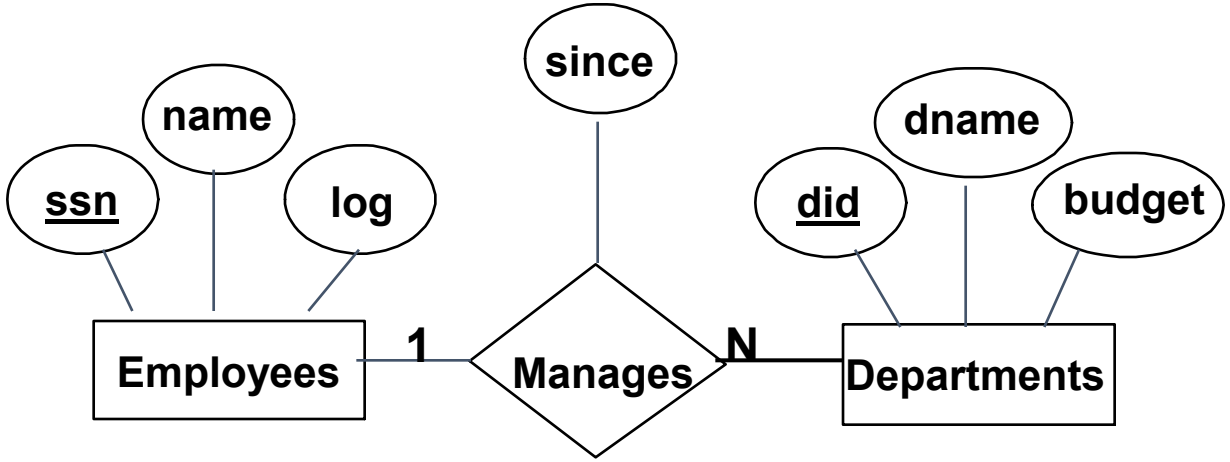


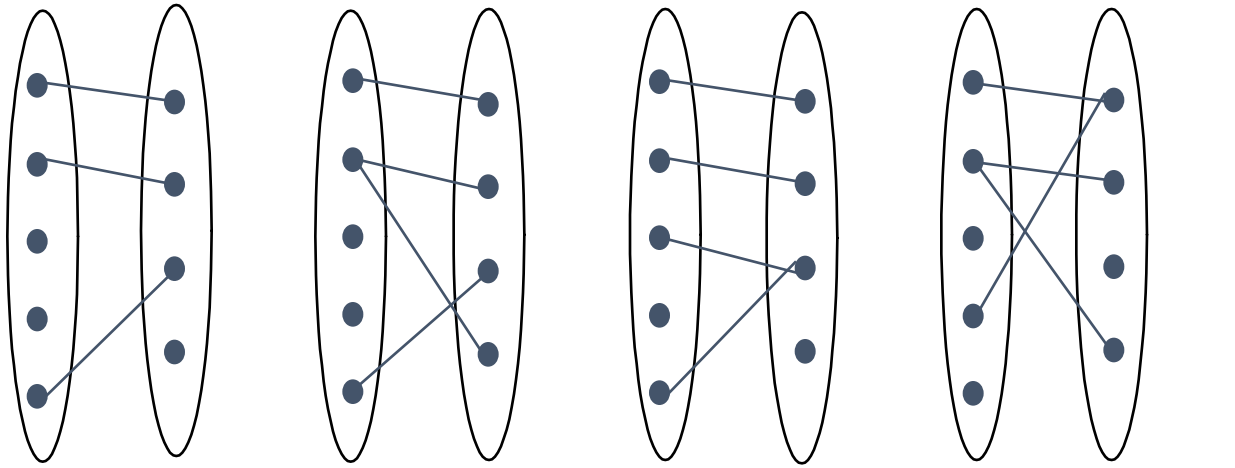

```
CREATE TABLE Employees
    (ssn VARCHAR(11),
    name VARCHAR(20),
    log  INTEGER,
    PRIMARY KEY  (ssn));
```

Review: Key Constraints

Lancaster University

- Each dept has at most one manager, according to the *key constraint* on Manages.

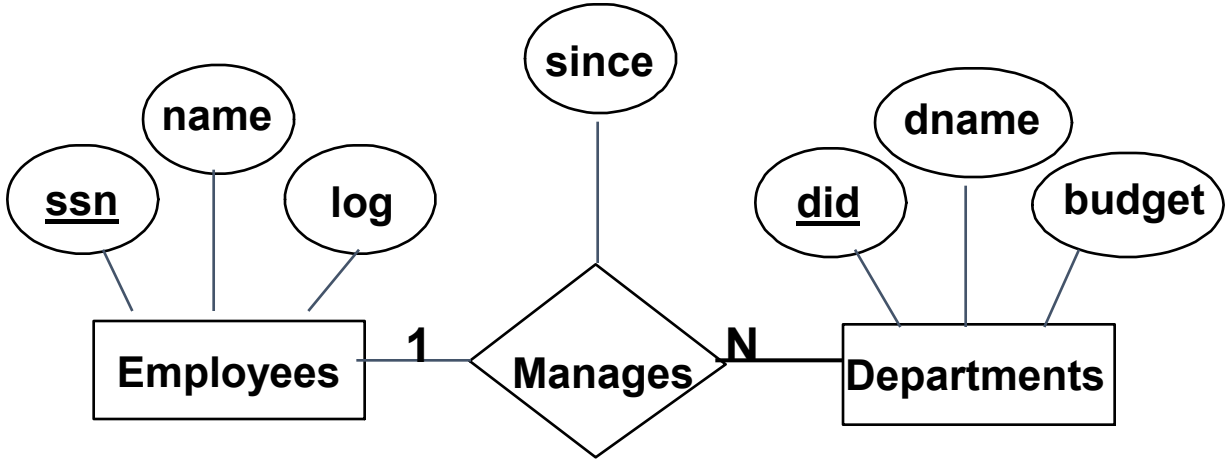




Translation to relational model?

Review: Key Constraints

- Each dept has at most one manager, according to the *key constraint* on Manages.

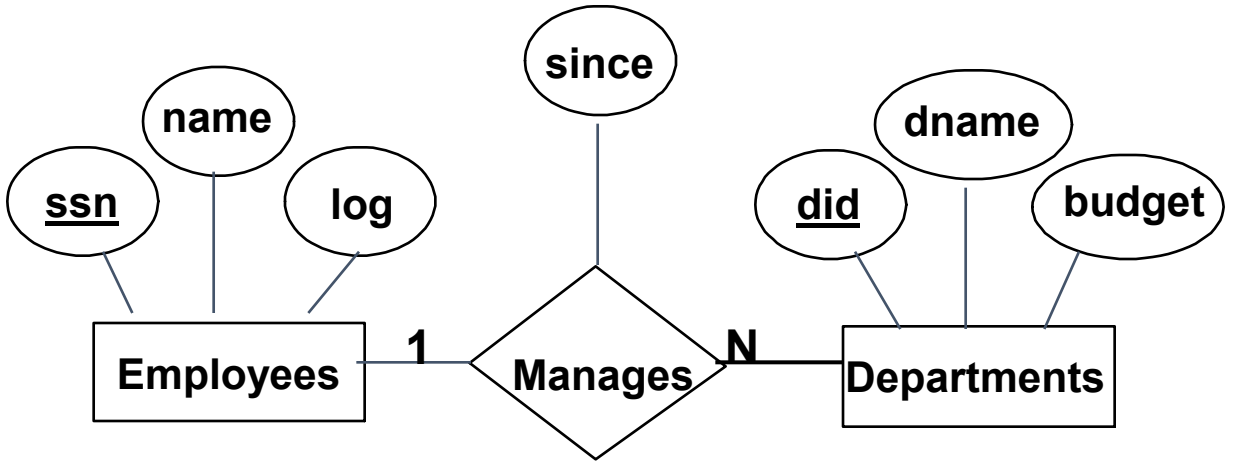


Translation to relational model?

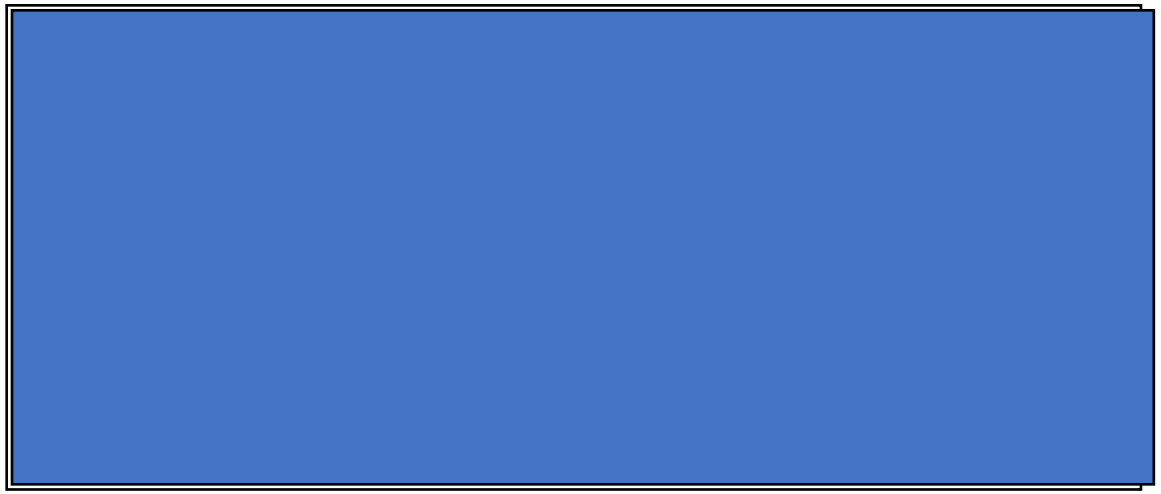
Review: Key Constraints

Lancaster University

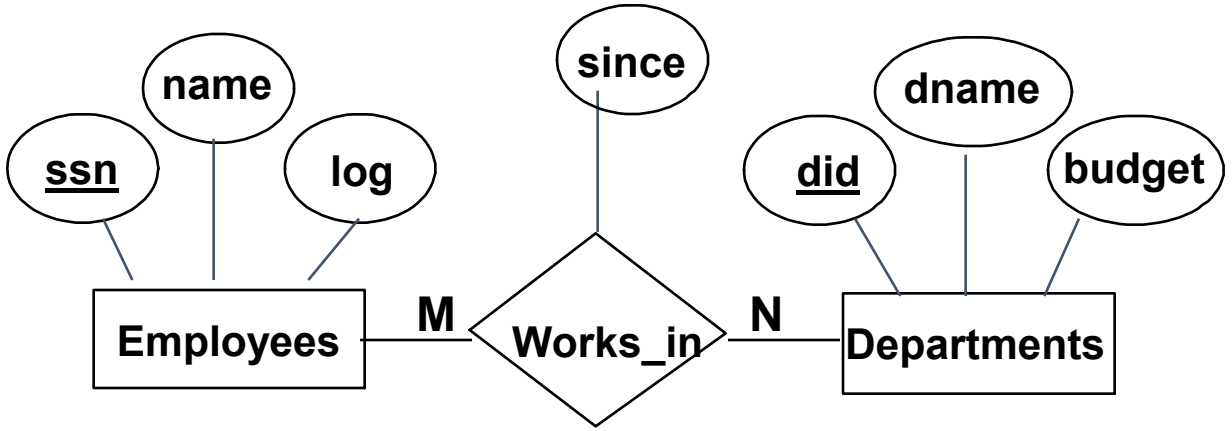
- Each dept has at most one manager, according to the *key constraint* on Manages.

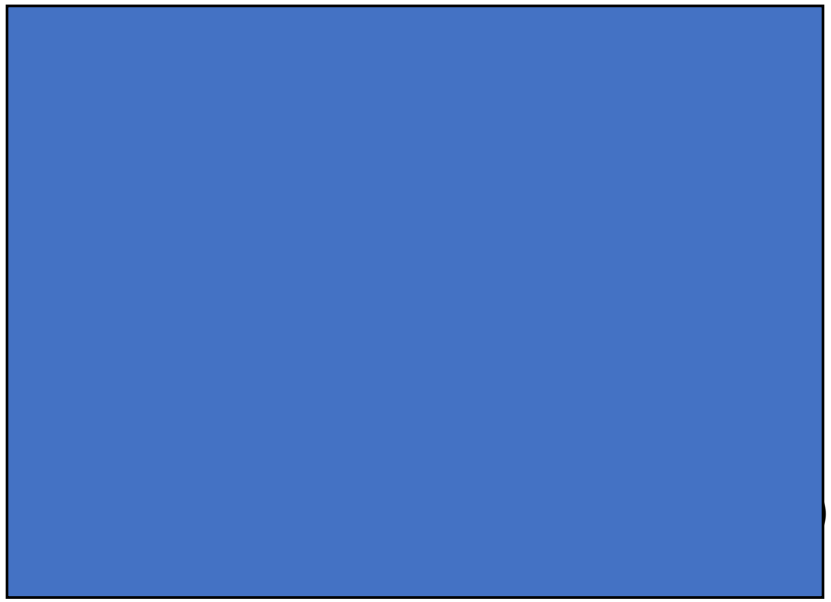


1. Since each department has a unique manager, we could instead combine Manages and Departments.

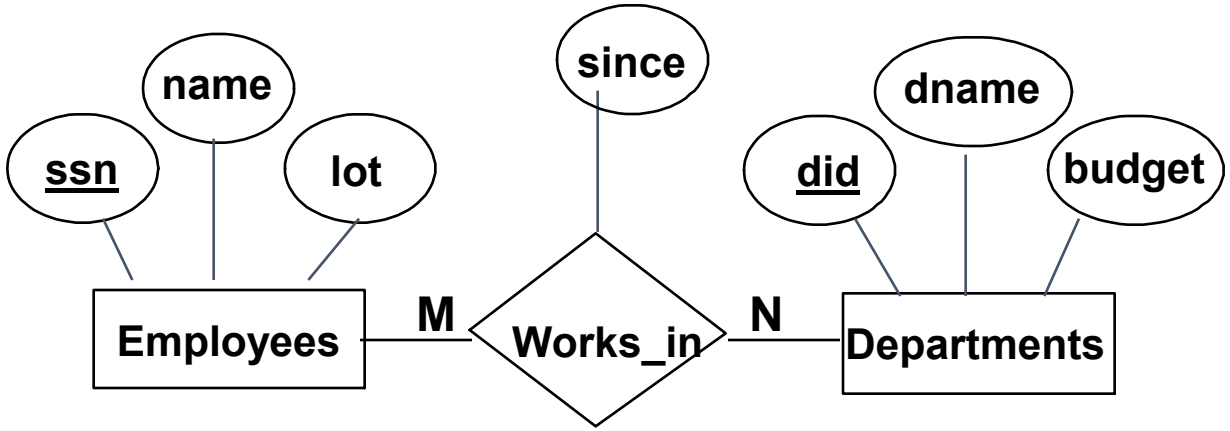


Relationship Sets to Tables



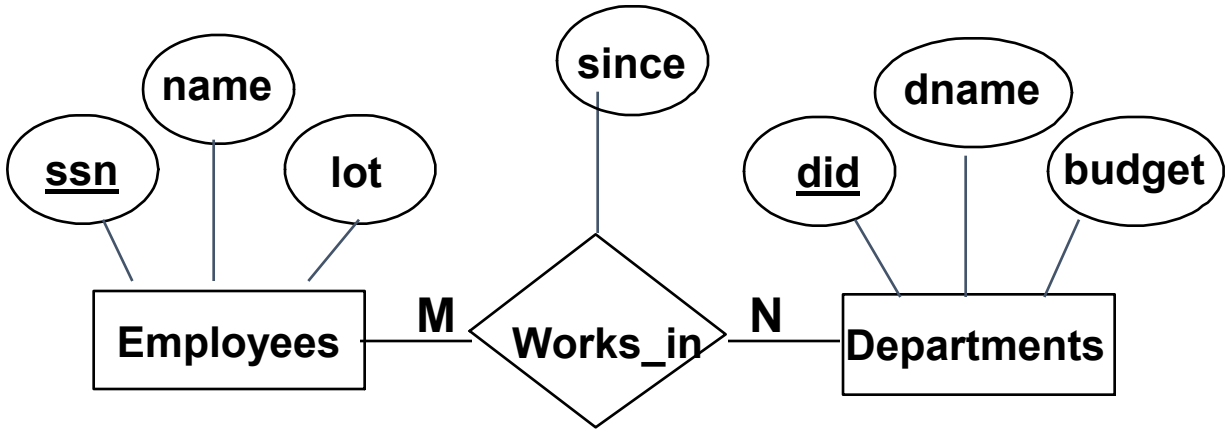


Relationship Sets to Tables

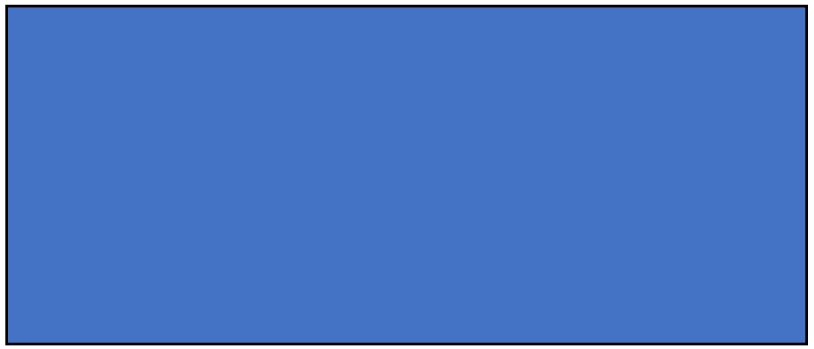


CREATE TABLE Works_In(

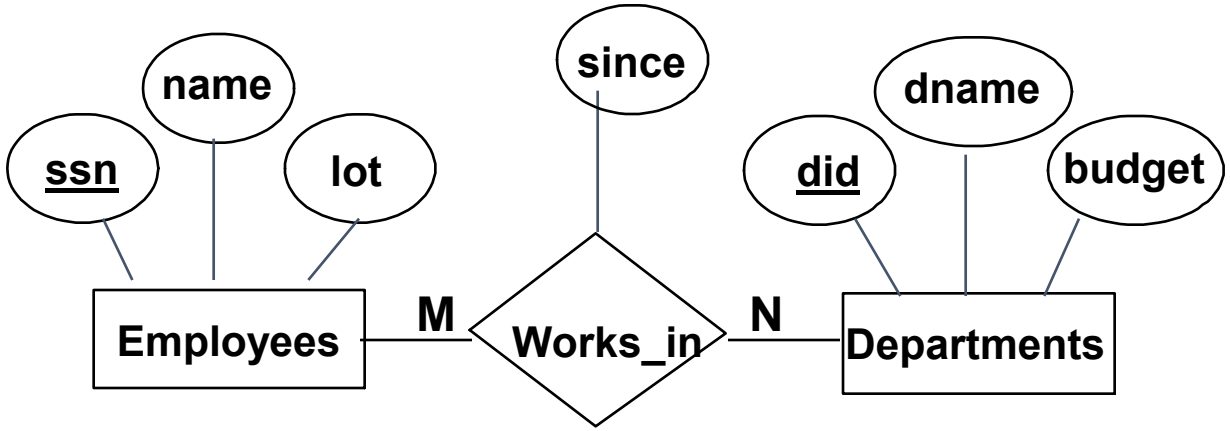
Relationship Sets to Tables




```
CREATE TABLE Works_In(
 ssn  VARCHAR(1),
  did  INTEGER,
  since  TEXT,
```

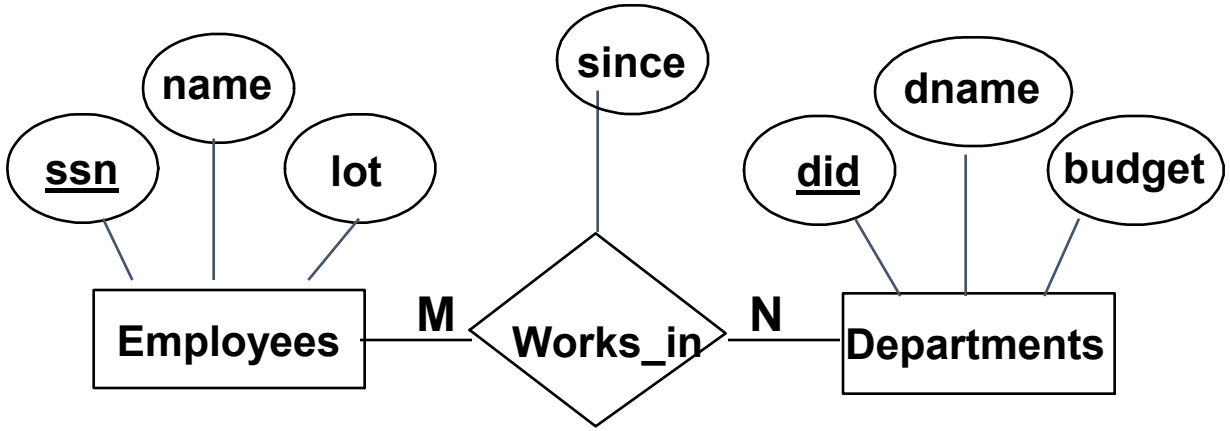


Relationship Sets to Tables




```
CREATE TABLE Works_In(
 ssn  VARCHAR(1),
  did  INTEGER,
  since  TEXT,
  PRIMARY KEY (ssn, did),
```

Relationship Sets to Tables

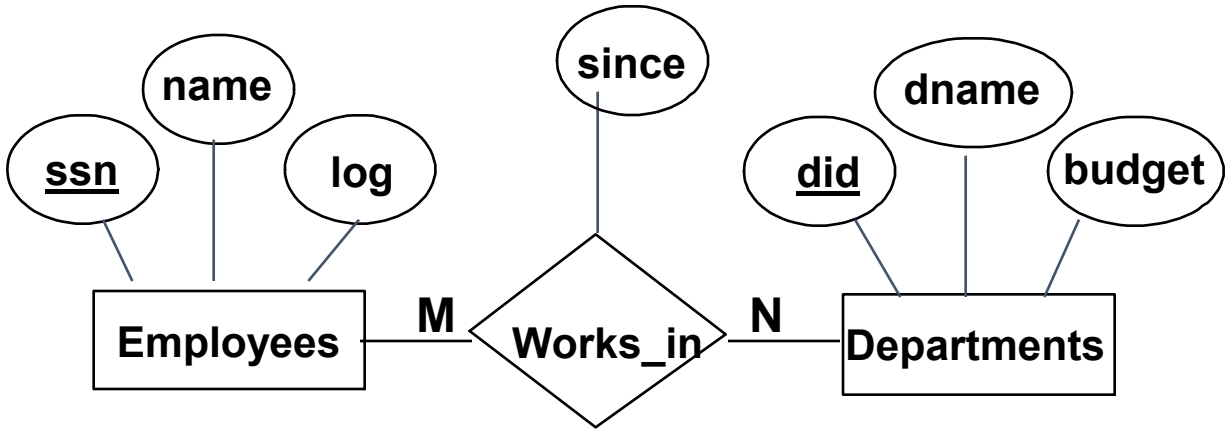



```
CREATE TABLE Works_In(
 ssn  VARCHAR(1),
  did  INTEGER,
  since  TEXT,
  PRIMARY KEY (ssn, did),
  FOREIGN KEY (ssn)
       REFERENCES Employees(ssn),
```

Relationship Sets to Tables

Lancaster University

- In translating a relationship set to a relation, attributes of the relation must include:
  - Keys for each participating entity set (as foreign keys).
  - All descriptive attributes.




```
CREATE TABLE Works_In(
 ssn  VARCHAR(1),
  did  INTEGER,
  since  TEXT,
  PRIMARY KEY (ssn, did), FOREIGN KEY (ssn)
      REFERENCES Employees(ssn),
  FOREIGN KEY Departments (did)
       REFERENCES Departments (did) );
```

Foreign Keys in SQL

Lancaster University

```
CREATE TABLE Enrolled
  (sid VARCHAR(20),  cid VARCHAR(20),  grade VARCHAR(2),
   PRIMARY KEY  (sid,cid),
   FOREIGN KEY (sid) REFERENCES Students (sid) );
```

Enrolled

sid	cid	grade
53666	Carnatic101	C
53666	Reggae203	B
53650	Topology112	A
53666	History105	B

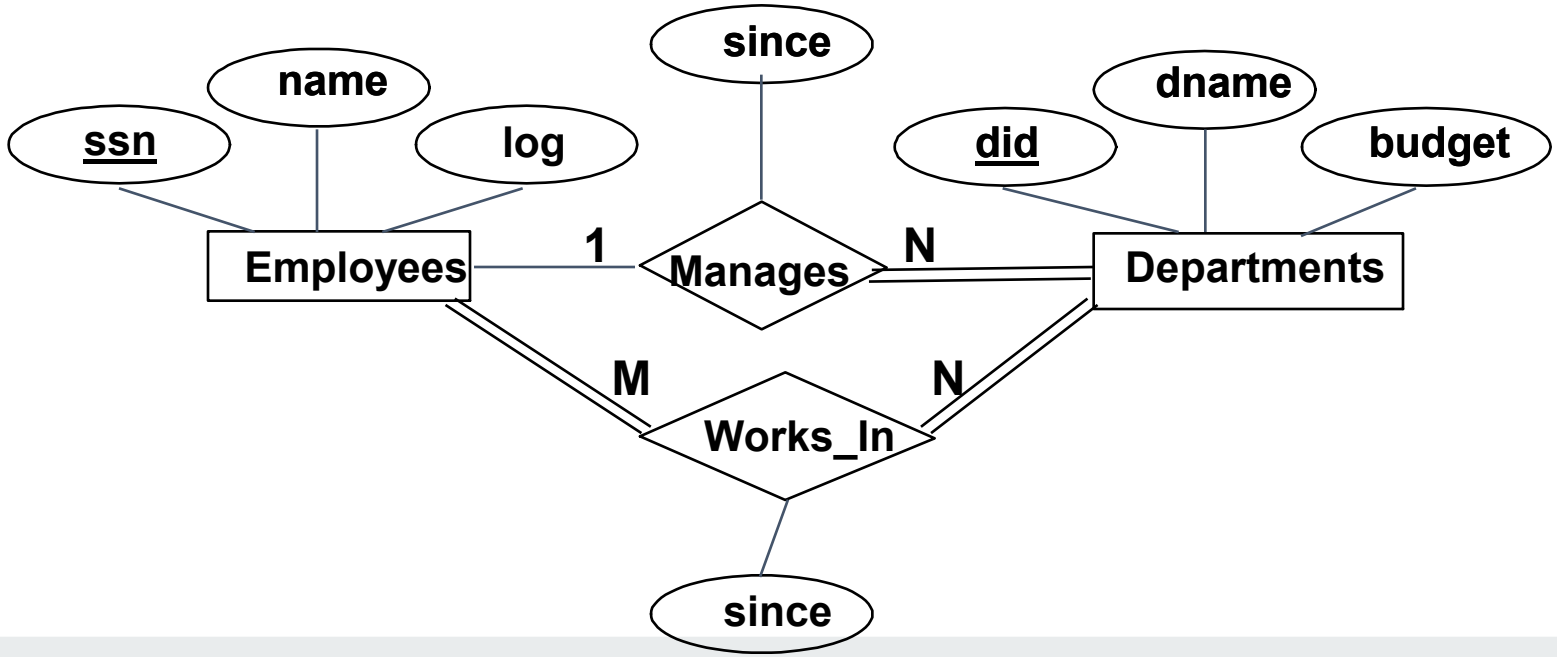
Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

Review: Participation Constraints

Lancaster University

- Does every department have a manager?
  - If so, this is a *participation constraint*: the participation of Departments in Manages is said to be *total* (vs. *partial*).
    - Every *did* value in Departments table must appear in a row of the Manages table (with a non-null *ssn* value!)

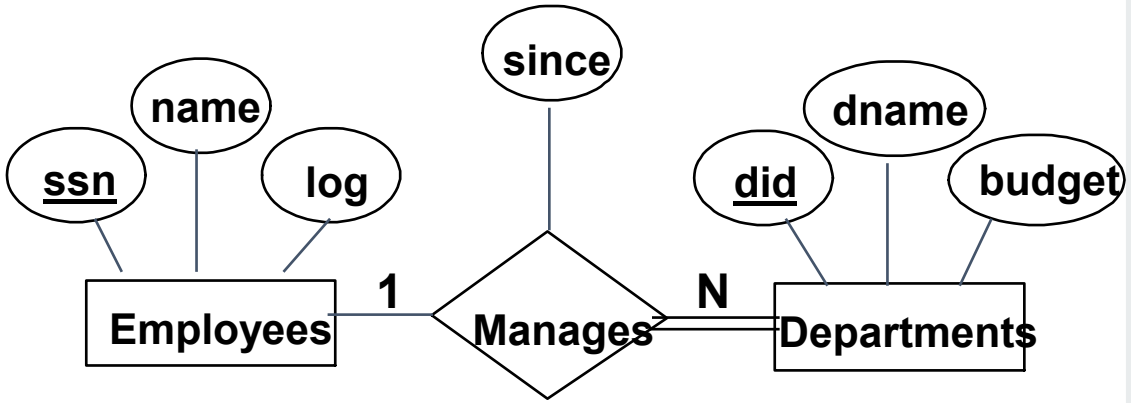


Participation Constraints in SQL

Lancaster University

- We can capture participation constraints involving one entity set in a binary relationship, but little else (without resorting to CHECK constraints).

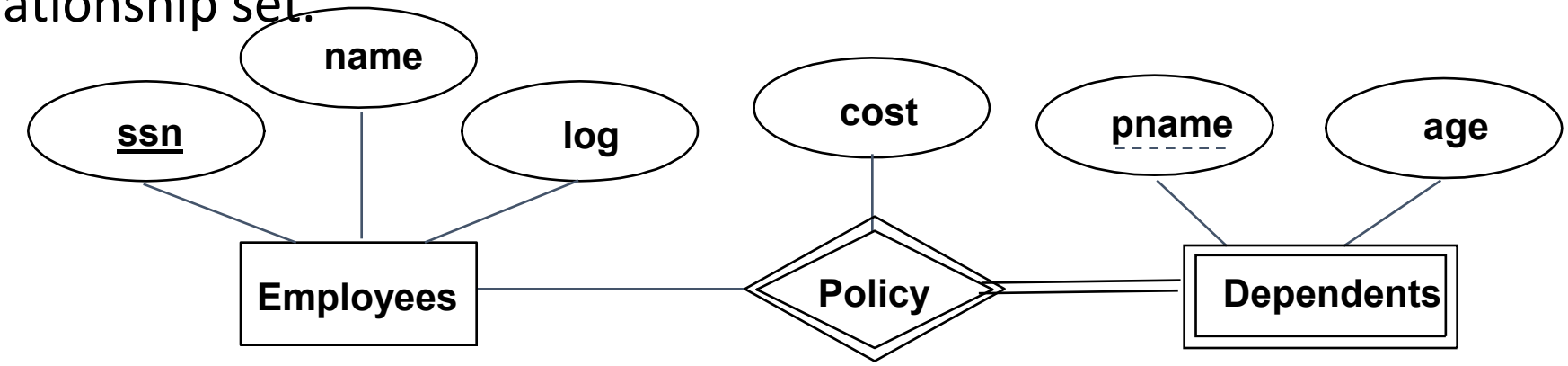
```
CREATE TABLE Dept_Mgr(
  did INTEGER,
  dname VARCHAR(20),
  budget REAL,
  ssn VARCHAR(11) NOT NULL,
  since DATE,
  PRIMARY KEY (did),
  FOREIGN KEY (ssn) REFERENCES Employees(ssn),
    ON DELETE CASCADE)
```



Review: Weak Entities

Lancaster University

- A *weak entity* can be identified uniquely only by considering the primary key of another (*owner*) entity.
  - Owner entity set and weak entity set must participate in a one-to-many relationship set (1 owner, many weak entities).
  - Weak entity set must have total participation in this *identifying* relationship set.

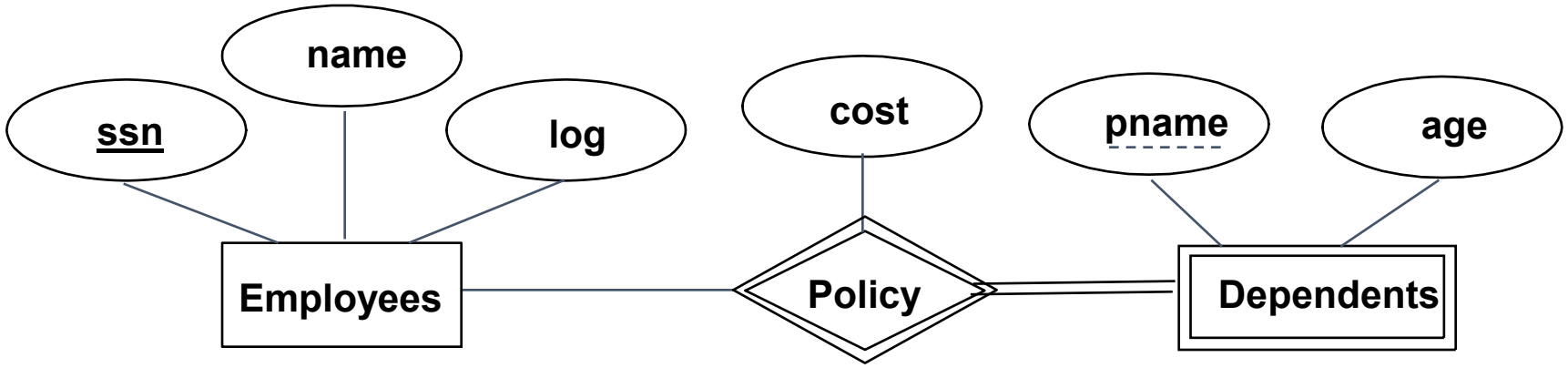


Weak Entities

Lancaster University

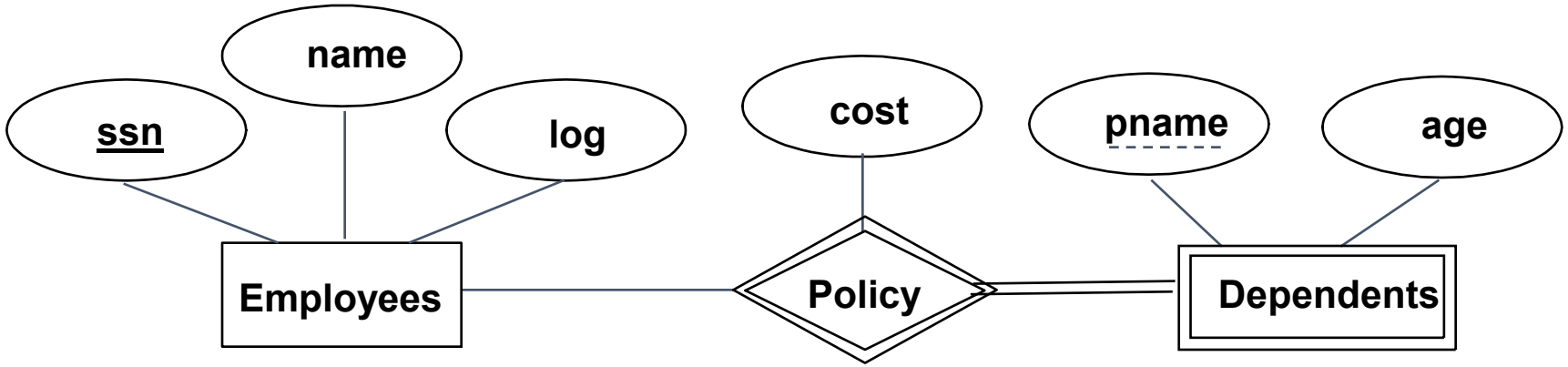
Weak entity set and identifying relationship set are translated into a single table.

- When the owner entity is deleted, all owned weak entities must also be deleted.



Weak Entities

```
CREATE TABLE  Dep_Policy (
  pname  VARCHAR(20),
  age  INTEGER,
  cost  REAL,
```



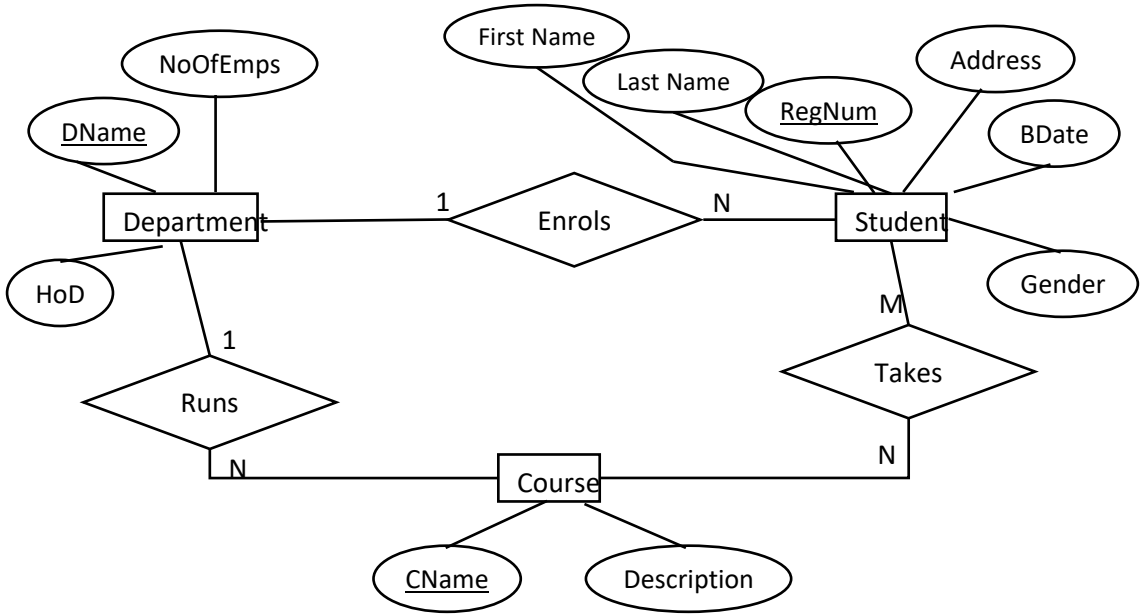
Referential Integrity in SQL/92

- SQL/92 supports all 4 options on deletes and updates.
  - Default is NO ACTION (*delete/update is rejected*)
  - CASCADE (also delete all tuples that refer to deleted tuple)
  - SET NULL / SET DEFAULT (sets foreign key value of referencing tuple)

DRAW ER DIAGRAM FOR THIS

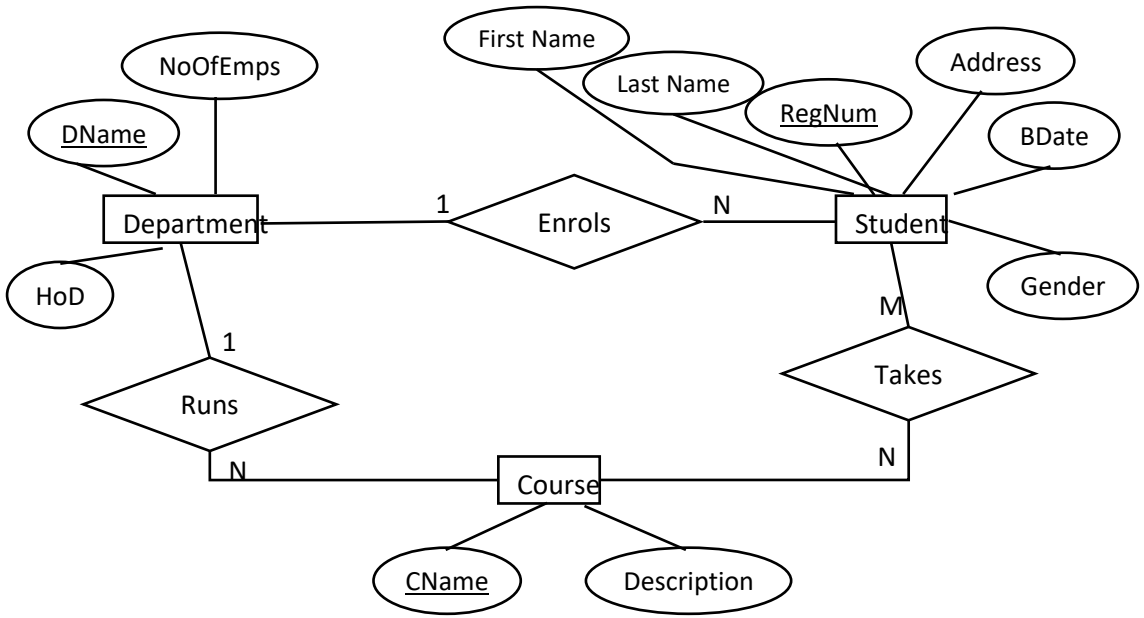
```
CREATE TABLE Enrolled
  (sid VARCHAR(20),
   cid VARCHAR(20),
   grade VARCHAR(2),
   PRIMARY KEY (sid,cid),
   FOREIGN KEY (sid)
     REFERENCES Students (sid)
          ON DELETE CASCADE
          ON UPDATE SET DEFAULT );
```

Example



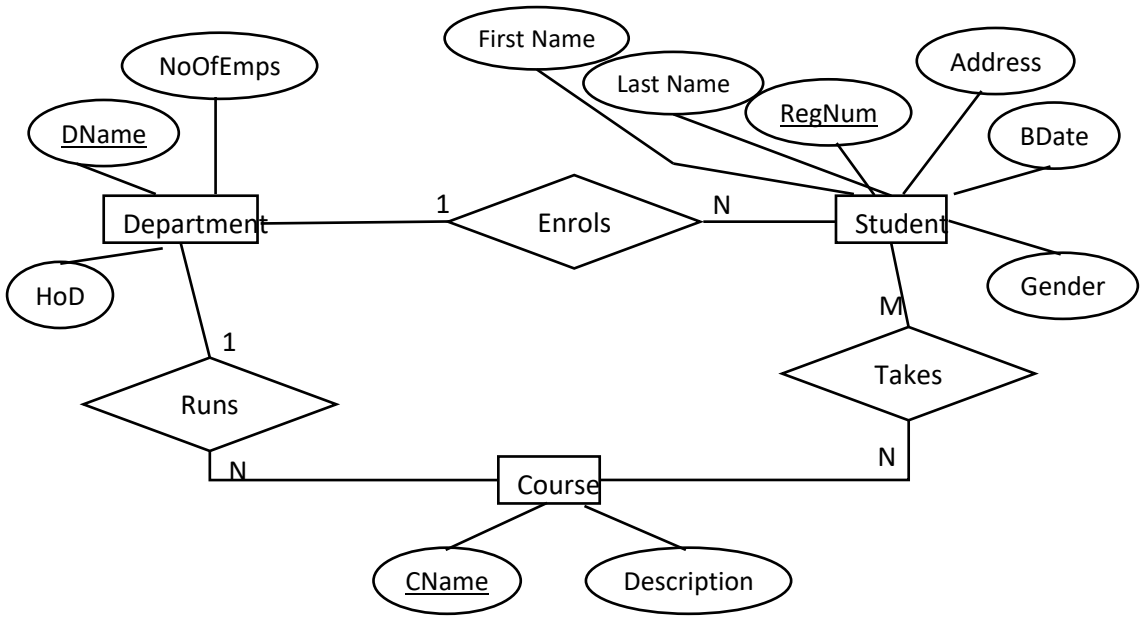
- CREATE TABLE Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName));

Example



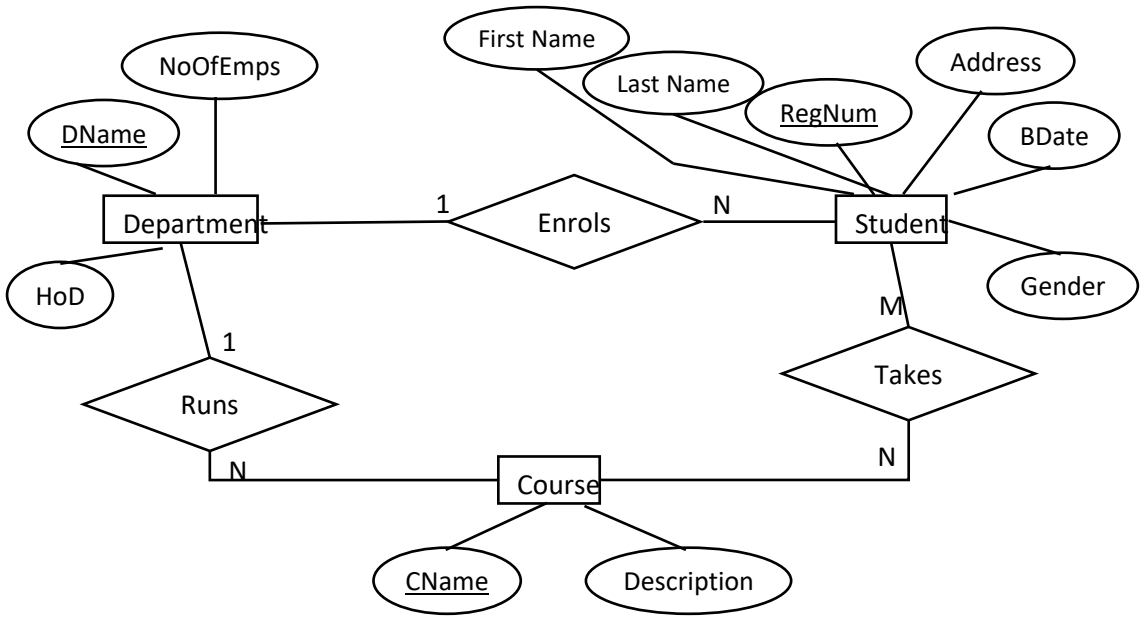
- CREATE TABLE Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName));
- CREATE TABLE StudentsEnrol(firstName TEXT, lastName TEXT, RegNumber INTEGER NOT NULL, Address TEXT, BDate TEXT, Gender TEXT, DepName TEXT, PRIMARY KEY (RegNumber), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL);
-
-

Example



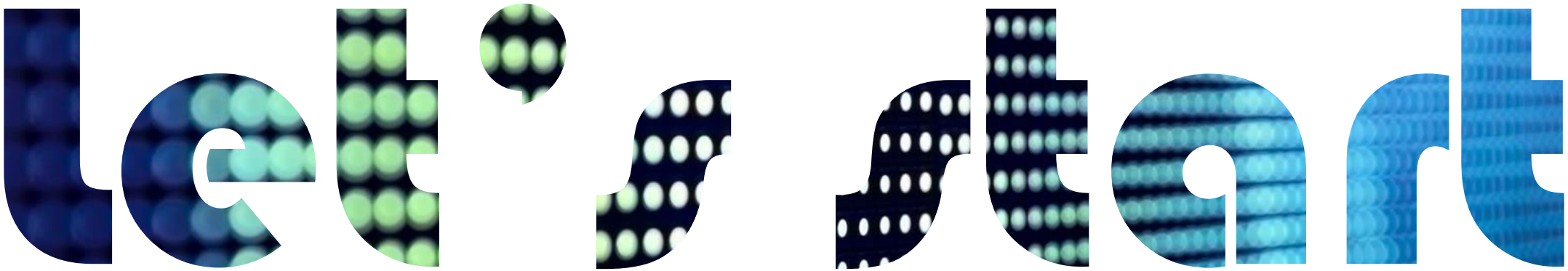
- CREATE TABLE Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName));
- CREATE TABLE StudentsEnrol(firstName TEXT, lastName TEXT, RegNumber INTEGER NOT NULL, Address TEXT, BDate TEXT, Gender TEXT, DepName TEXT, PRIMARY KEY (RegNumber), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL);
- CREATE TABLE CourseRuns(CName TEXT NOT NULL, Desc TEXT, DepName TEXT, PRIMARY KEY (CName), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL);
-

Example



- CREATE TABLE Department(DName TEXT NOT NULL, HoD TEXT, NoOfEmp INTEGER, PRIMARY KEY(DName));
- CREATE TABLE StudentsEnrol(firstName TEXT, lastName TEXT, RegNumber INTEGER NOT NULL, Address TEXT, BDate TEXT, Gender TEXT, DepName TEXT, PRIMARY KEY (RegNumber), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL);
- CREATE TABLE CourseRuns(CName TEXT NOT NULL, Desc TEXT, DepName TEXT, PRIMARY KEY (CName), FOREIGN KEY(DepName) REFERENCES Department(DName) ON DELETE SET NULL);
- CREATE TABLE StTakesCourse(CName TEXT NOT NULL, RegNumber INTEGER NOT NULL, PRIMARY KEY(CNAME,RegNumber), FOREIGN KEY (CName) REFERENCES CourseRuns(CName), FOREIGN KEY (RegNumber) REFERENCES StudentsEnrol(RegNumber));

Lancaster University



Lancaster University

SCC.201
Database Management Systems

2023 - Week 3 – Relational Model to SQL

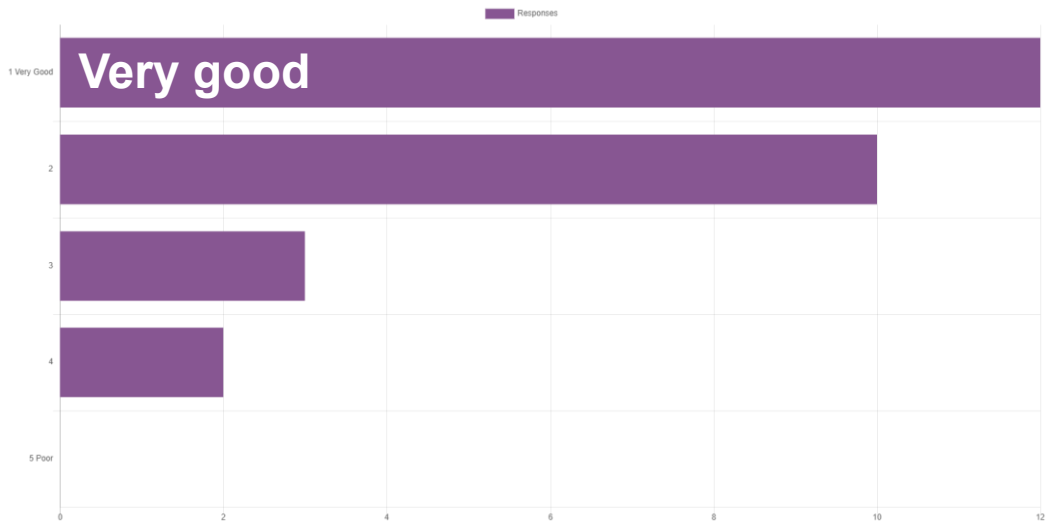
Uraz C Turker & Ricki Boswell

Please read chapters 5 and 7.1 from “Fundamentals of Database Systems by Elmasri”

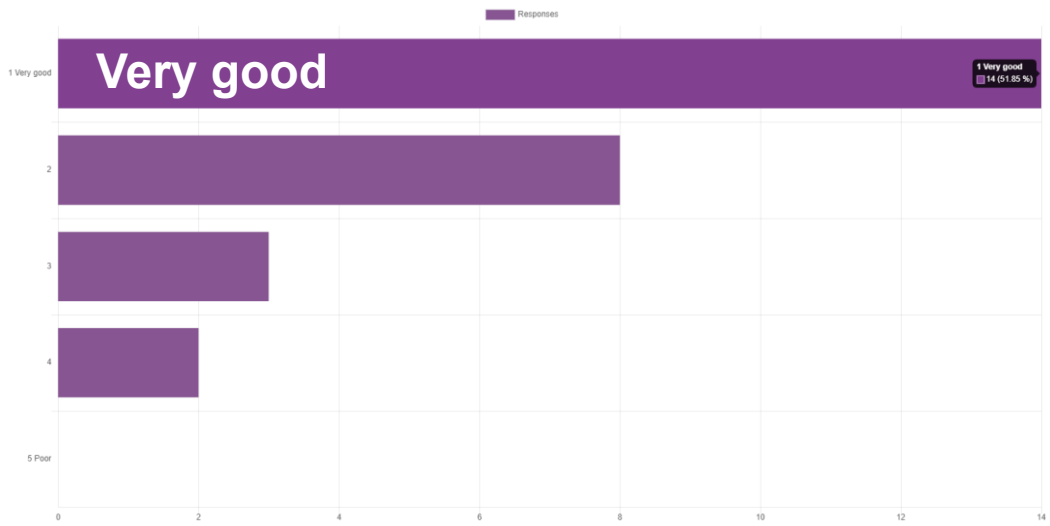
Evaluation results

Lancaster University

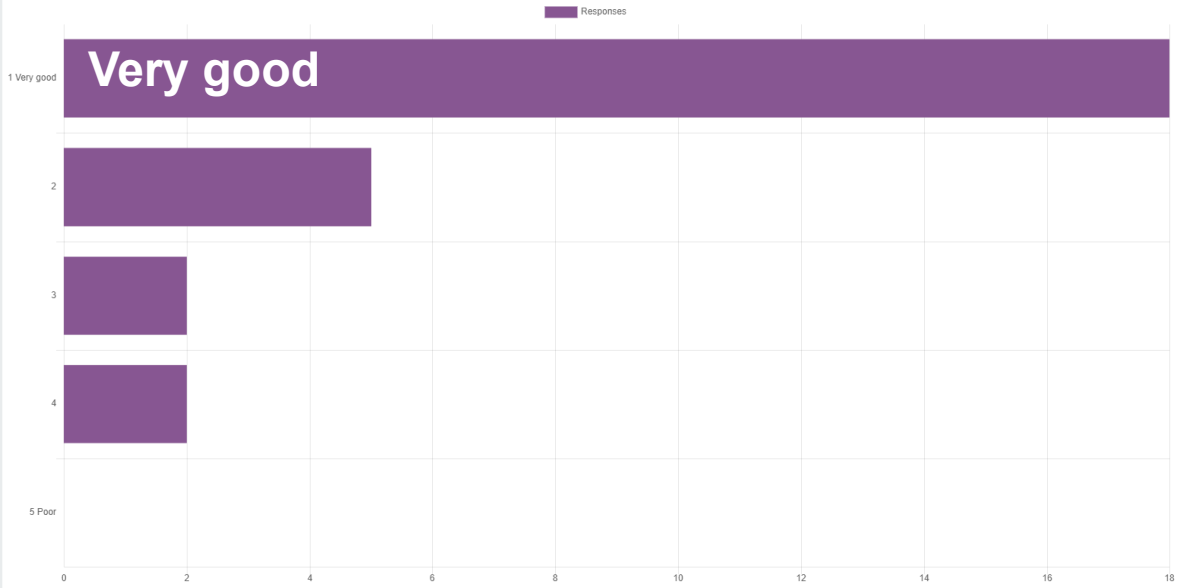
The quality of teaching is



The module overall so far



Helpfulness of the staff is



How did you found the module as a whole?

Actions
Another Evaluation after 2 weeks.

Lancaster University

Things will be done

- Preparing Q&A (Video tutorials)
- More examples
- Refining slides
- “Use cases behind creating ER diagrams, context behind database can help us approach better”

Things already started:

- Upload lecture slides before the lecture
  - I am uploading DRY contend without
    - Providing Q & A’s
    - Providing animations
    - Providing Recall Sections

What will you learn today?

Lancaster University

- Relational Model
  - Review, Relational Algebra
- Reintroduction to SQL

Curriculum Design: Outline Syllabus

This module builds upon knowledge gained in Part I by providing a theoretical background to the design, implementation and use of database management systems, both for data designers and application developers. It takes into account all relevant aspects related to information security in the design, development and use of database systems. The course consists of a number of related sections, which range from single lectures to multi-lecture streams, depending on the required depth of coverage. The sections are as follows.

Introduction : we begin with a brief history of how the need for database management systems (DBMS) grew over time and how they are applied in day to day scenarios.

Database Design: before making use of a DBMS, we must capture our requirements : what data do we actually wish to model? We make use of the Extended Entity-Relationship (EER) model which is both a technique and a notation for designing the data in a DBMS independent way.

The Relational Model: now the de-facto standard for DBMS, this was a revolutionary step taken in 1970. We extensively examine the Model, looking at relational database systems, the model itself and the normalisation process, the relational algebra (the mathematical theory that underpins the model), the three schema architecture and schema definition in SQL. Finally, we look at how we can map the EER model into an equivalent Relational Model. The resultant database is then examined in terms of access rights and privileges.

A (re)Introduction to SQL: SQL is the de-facto standard for DBMS query languages. We look at both the DDL (data definition language) and DML (data manipulation language). We introduce the use of views, a powerful mechanism for providing privacy and security. We look at the Discretionary Access Control (DAC) features that allow the granting and withholding of access rights and privileges.

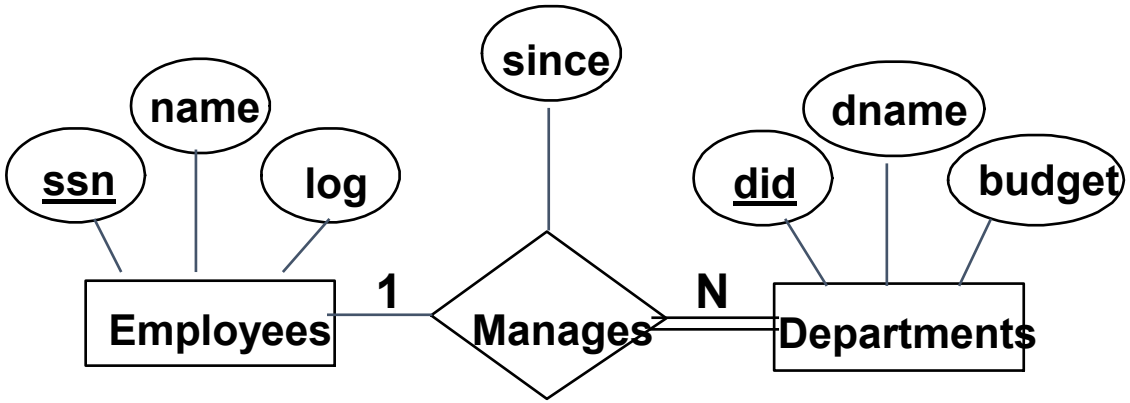
Accessing relational DBMS via Java: we explore the facilities of the JDBC and show how we can write applications in Java which connect with a relational DBMS (in practice, MySQL).

The Physical Model: as Computer Scientists, our students need an awareness of the techniques that allow rapid access to stored data. In this section, we examine the physical data organisation and associated access methods. We show under what circumstances the organisations can be applied, and we look at how queries can be optimised.

Transaction processing and concurrency control: a huge part of DBMS in practice is the need to support transactions and concurrency, allowing huge numbers of users to access the DBMS at any one time while still ensuring the consistency of the data. This stream examines the problems and solutions in depth.

Participation Constraints in SQL

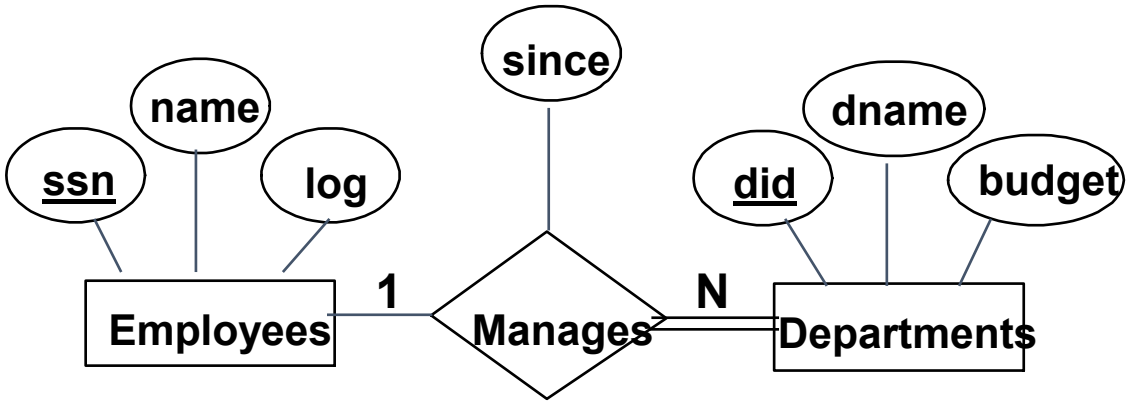
- “Employee can manage one department, a department must be managed by many employees”



Participation Constraints in SQL

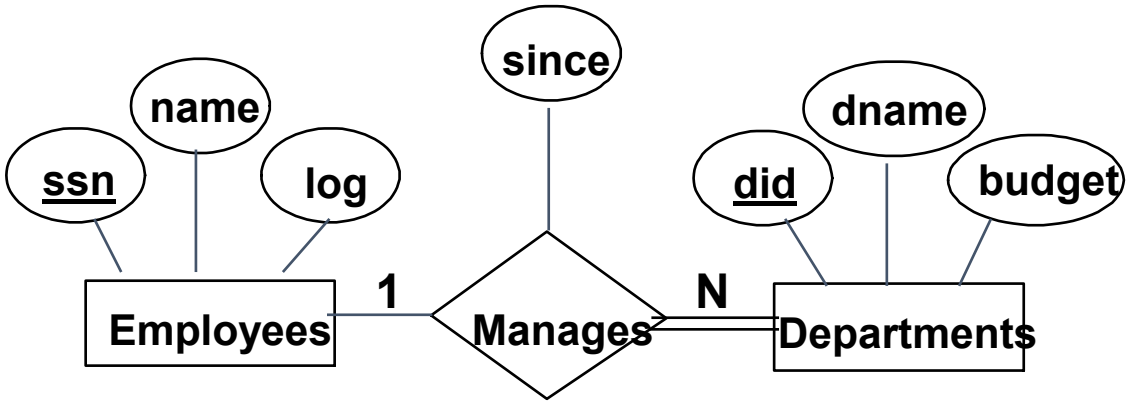
Lancaster University

- “Employee must manage one department, a department may be managed by many employees”



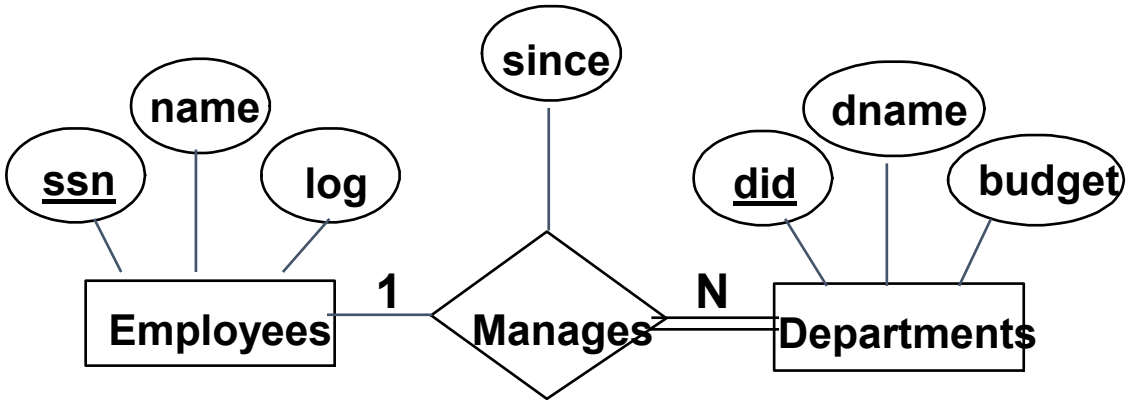
Participation Constraints in SQL

- “Employee can manage many departments, a department may be managed by many employees”

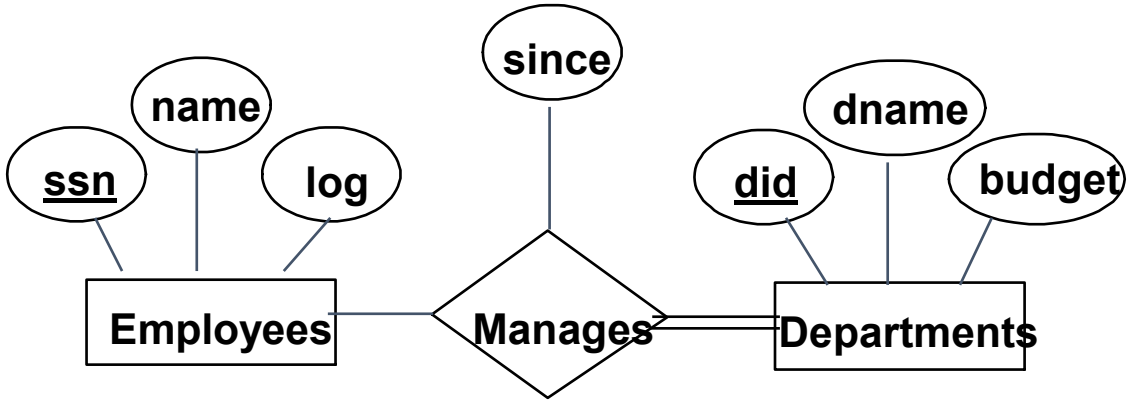


Participation Constraints in SQL

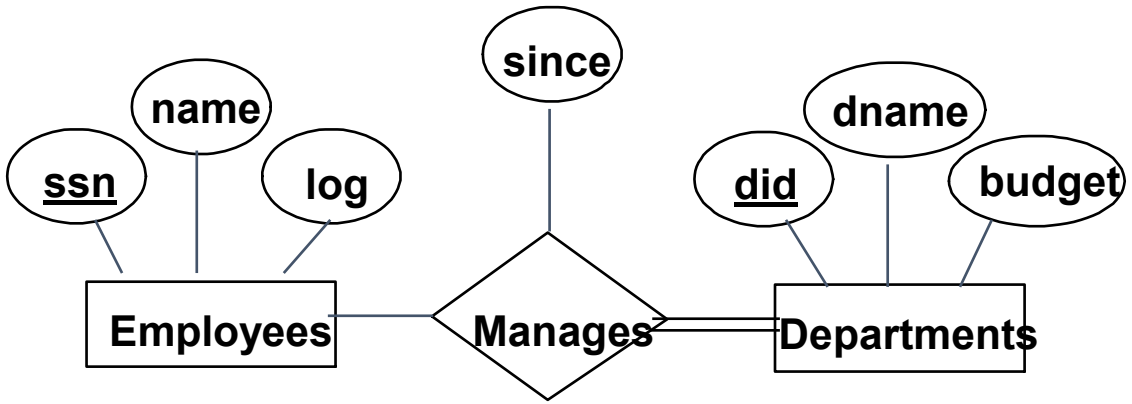
- “Employee can manage many departments, a department must be managed by one employee”



- “Employee must manage department(s) but department may be managed by employee(s).”

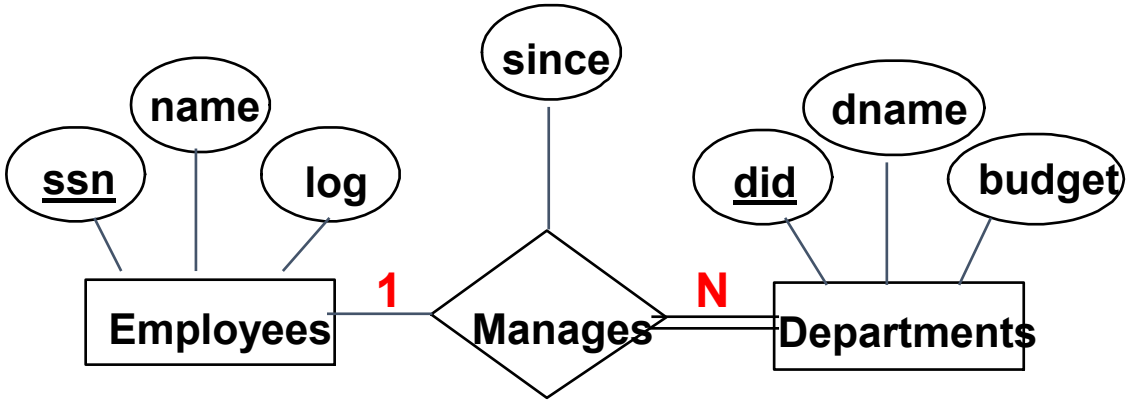


- “Employee can manage department(s) but department must be managed by employee(s).”



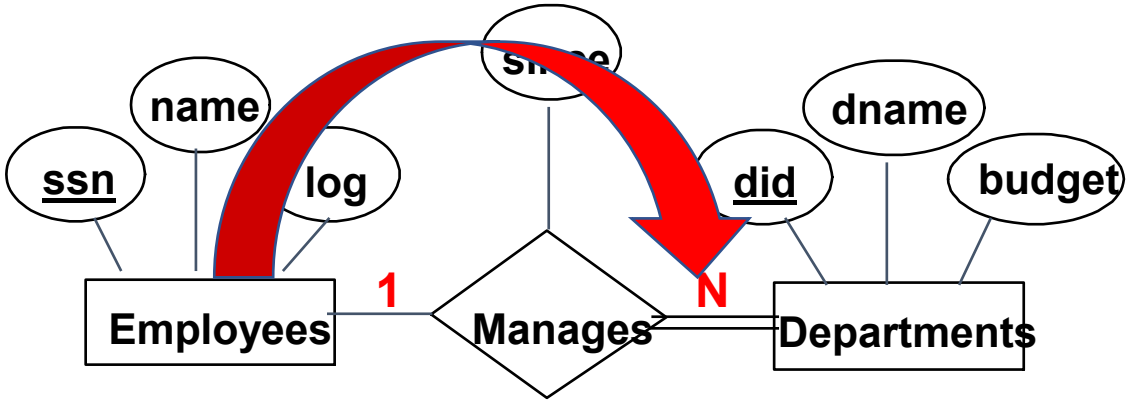
RECALL

- Multiplicty constraints are given in the OPPOSITE ends of the relation



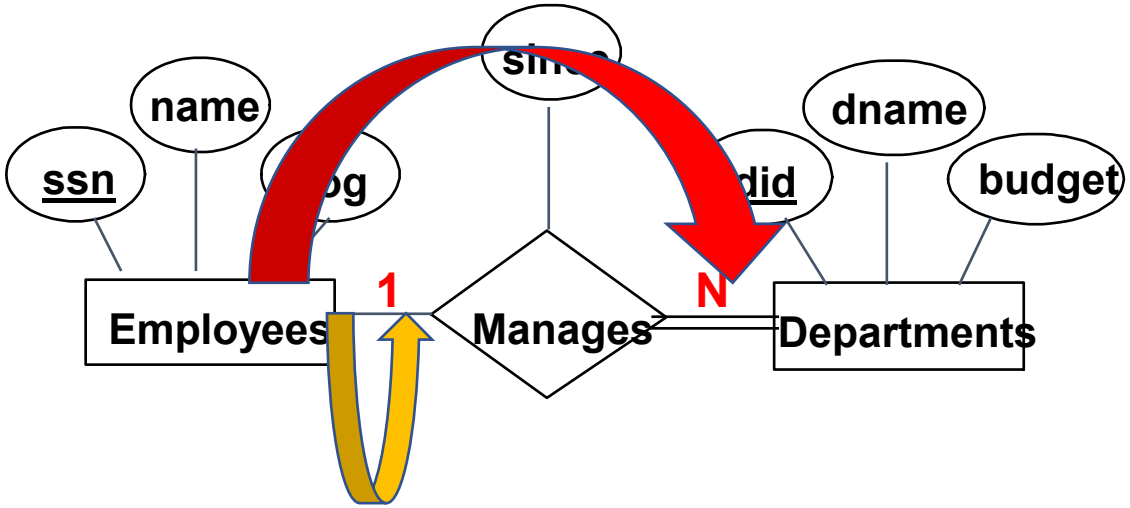
RECALL

- Multiplicty constraints are given in the OPPOSITE ends of the relation
- Employee manages MANY departments.



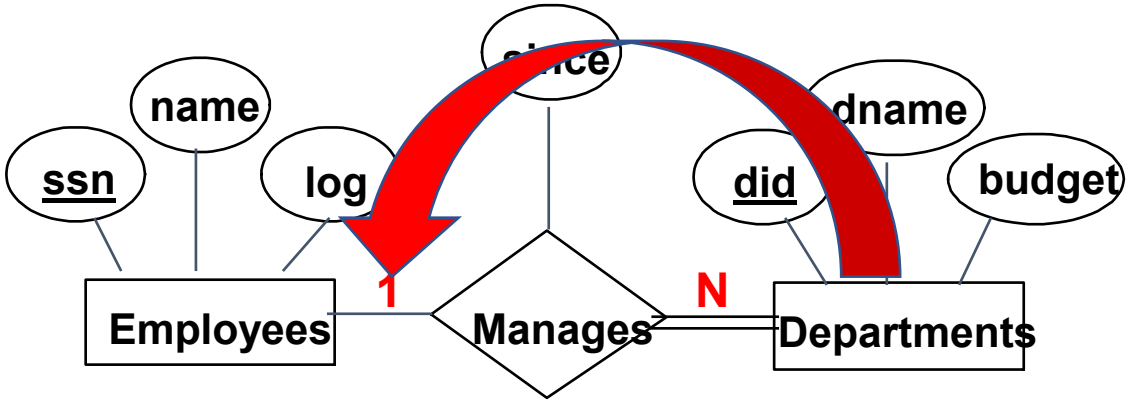
RECALL

- Participation constraints are given BY the entity set.
- Employee MAY manage MANY departments.



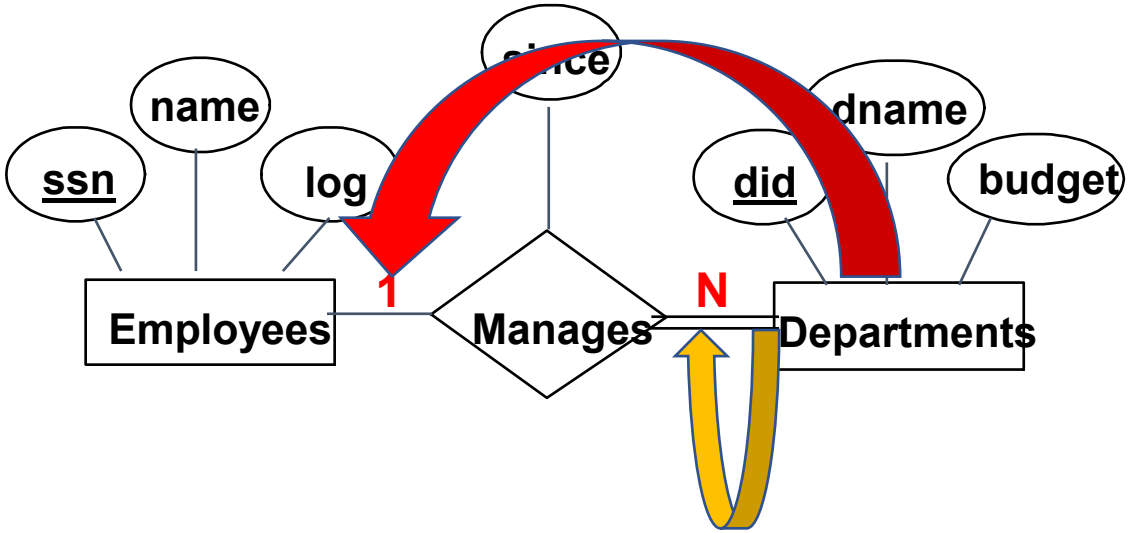
RECALL

- Multiplicty constraints are given in the OPPOSITE ends of the relation, Participation constraints are given BY the entity set.
- Employee may manage many departments.
- Department managed by ONE employee.



RECALL

- Multiplicty constraints are given in the OPPOSITE ends of the relation, Participation constraints are given BY the entity set.
- Employee may manage many departments.
- A department must be managed by one employee.



Recall

Lancaster University

CREATE TABLE

Recall

Lancaster University

```
CREATE TABLE (
        );
```

Recall

CREATE TABLE (Att1 Domain, Att2, Domain,…,
);

Recall

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

Recall

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- Creates a table in a database.

Recall

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- Creates a table in a database.
- Domain: TEXT, INT, INTEGER, REAL, BLOB…

Recall

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- Creates a table in a database.
- Domain: TEXT, INT, INTEGER, REAL, BLOB… ->

Recall

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- Creates a table in a database.
- Domain: TEXT, INT, INTEGER, REAL, BLOB… -> SQL interpreter dependent.

Recall

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- Creates a table in a database.
- Domain: TEXT, INT, INTEGER, REAL, BLOB… -> SQL interpreter dependent.
- IC PRIMARY KEY(Atti, Attj,…)
-
-
-
-

Recall

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- Creates a table in a database.
- Domain: TEXT, INT, INTEGER, REAL, BLOB… -> SQL interpreter dependent.
- IC PRIMARY KEY(Atti, Attj,…)
- IC FOREIGN KEY(Attk,Attl,…)
-
-
-

Recall

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- Creates a table in a database.
- Domain: TEXT, INT, INTEGER, REAL, BLOB… -> SQL interpreter dependent.
- IC PRIMARY KEY(Atti, Attj,…)
- IC FOREIGN KEY(Attk,Attl,…)
- NOT NULL -> The corresponding attribute CANNOT BE NULL in any instance
-
-

Recall

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- Creates a table in a database.
- Domain: TEXT, INT, INTEGER, REAL, BLOB… -> SQL interpreter dependent.
- IC PRIMARY KEY(Atti, Attj,…)
- IC FOREIGN KEY(Attk,Attl,…)
- NOT NULL -> The corresponding attribute CANNOT BE NULL in any instance
- UNIQUE -> The corresponding attribute CANNOT REPEAT in any instance

Recall Foreign Keys

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

Recall Foreign Keys

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

IC FOR MEC_REPAIRS: FOREIGN KEY Brand REFERENCES CAR(Brand)

Recall Foreign Keys

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

IC FOR MEC_REPAIRS: FOREIGN KEY Brand REFERENCES CAR(Brand)

Note that the foreign key attribute (Brand) of the referenced table (MEC_REPAIR) is a PRIMARY KEY attribute for the referencing table (CAR).

Recall Foreign Keys

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

IC FOR MEC_REPAIRS: FOREIGN KEY Brand REFERENCES CAR(Brand)

Note that the foreign key attribute (Brand) of the referenced table (MEC_REPAIR) is a PRIMARY KEY attribute for the referencing table (CAR). We can remove tuples from the referencing table (e.g. remove tuple with Primary Key Hyundai E.GLS).

Recall

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

IC FOR MEC_REPAIRS: FOREIGN KEY Brand REFERENCES CAR(Brand)

Note that the foreign key attribute (Brand) of the referenced table (MEC_REPAIR) is a PRIMARY KEY attribute for the referencing table (CAR). We can remove tuples from the referencing table (e.g. remove tuple with Primary Key Hyundai E.GLS). Since this tuple is in relation to a tuple in MEC_REPAIR, we have to inform DBMS about the outcome of this modification on the referenced table according to the ER diagram/Rules.

Recall Foreign Keys

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

IC FOR MEC_REPAIRS: FOREIGN KEY Brand REFERENCES CAR(Brand)

Note that the foreign key attribute (Brand) of the referenced table (MEC_REPAIR) is a PRIMARY KEY attribute for the referencing table (CAR). We can remove tuples from the referencing table (e.g. remove tuple with Primary Key Hyundai E.GLS). Since this tuple is in relation to a tuple in MEC_REPAIR, we have to inform DBMS about the outcome of this modification on the referenced table according to the ER diagram/Rules.

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
-
-
  -
  -

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
-
  -
  -

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  -
  -

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If CASCADE -> Delete all these tuples in the referenced table (MEC_REPAIR)

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If CASCADE -> Delete all these tuples in the referenced table (MEC_REPAIR) and delete the tuple in the referencing table (CAR).
  -

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If CASCADE -> Delete all these tuples in the referenced table (MEC_REPAIR) and delete the tuple in the referencing table (CAR).
  - If REJECT-> Do NOT allow deletion of the tuple in the referenced and in the referencing table (CAR)

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If SET DEFAULT -> Select all these tuples in the referenced table (MEC_REPAIR)

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If SET DEFAULT -> Select all these tuples in the referenced table (MEC_REPAIR)

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
DEFAULT	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If SET DEFAULT -> Select all these tuples in the referenced table (MEC_REPAIR) and set the foreign key value to a default value (you have to specify this) of these tuples in the referenced table (MEC_REPAIR).

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
DEFAULT	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If SET DEFAULT -> Select all these tuples in the referenced table (MEC_REPAIR) and set the foreign key value to a default value (you have to specify this) of these tuples in the referenced table (MEC_REPAIR). And delete tuples in CAR

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
Toyota_Corolla	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If SET NULL -> Select all these tuples in the referenced table (MEC_REPAIR)

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Toyota_Corolla	1300	3.18	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
NULL	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If SET NULL -> Select all these tuples in the referenced table (MEC_REPAIR) and set the foreign key value to a NULL value of these tuples in the referenced table (MEC_REPAIR).

Referential Integrity

Lancaster University

CREATE TABLE (Att1 Domain, Att2, Domain,…, IC1, IC2, IC3,…);

- ON DELETE (CASCADE/SET NULL/REJECT/SET DEFAULT)
  - Used for an IC Foreign Key / Hierarchical Tables (out of the scope of this module)

CAR

Brand	Weight	Length	Max_Speed
BMW 3.21	1400	3.21	200
Hyundai E.GLS	1400	3.16	210

MEC_REPAIR

Brand	Price	SSI	Name	Phone_Number
BMW 3.21	10	87542702	Tom	75315567
NULL	23	68201937	Uraz	75335521
Hyundai E.GLS	12	23139827	Nick	75315544

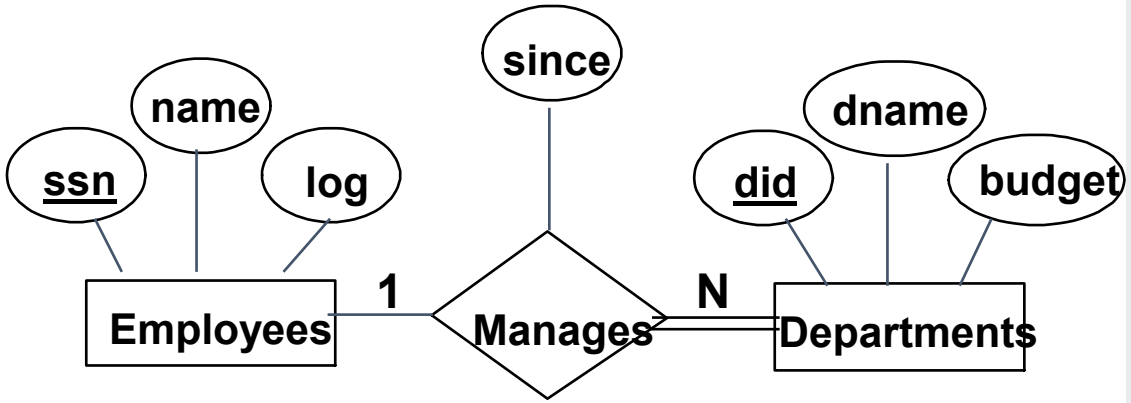
- If a tuple (say 2nd tuple) is to be deleted from referencing table (CAR)
- Get the primary key value of the tuple (Toyota_Corolla).
- Find all the tuples with values (Toyota_Corolla) in the referenced table (MEC_REPAIR)
  - If SET NULL -> Select all these tuples in the referenced table (MEC_REPAIR) and set the foreign key value to a NULL value of these tuples in the referenced table (MEC_REPAIR). And delete the tuples in the referencing table (CAR).

Integrity Constraints (Key,Participation) in SQL

Lancaster University

- We can capture participation constraints involving one entity set in a binary relationship, but little else (without resorting to CHECK constraints).

```
CREATE TABLE Dept_Mgr(
  did INTEGER,
  dname VARCHAR(20),
  budget REAL,
  ssn VARCHAR(11) NOT NULL,
  since DATE,
  PRIMARY KEY (did),
  FOREIGN KEY (ssn) REFERENCES Employees(ssn),
    ON DELETE CASCADE)
```



QUIZ (Formative-Grade yourself)

Lancaster University

- *"We are running an instrument store in Uxbridge, London. We have products with immutable ProductID, and name, price, product picture, and amount_in_Store. Moreover, we want to create a database that keeps transaction information with transaction date, ProductID (customer purchased), and amount_purchased, transactionID.*
- *We can issue one transaction per item, and we will not keep records of a transaction if the purchased items are dated (removed from the store).*"

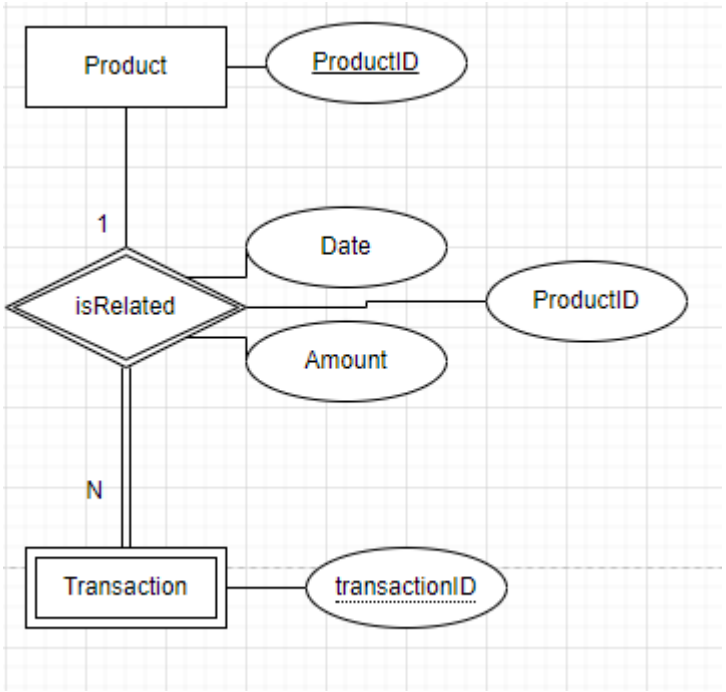
- Draw ERD, Reveal Relational Schema and Integrity Constraints and write SQL code to create the database.

QUIZ (Formative-Grade yourself)

Lancaster University

- *"We are running an instrument store in Uxbridge, London. We have products with immutable ProductID, and name, price, product picture, and amount_in_Store.* ***Moreover, we want to create a database that keeps transaction information with transaction date, ProductID (customer purchased), and amount_purchased, transactionID.***
- ***We can issue one transaction per item, and we will not keep records of a transaction if the purchased items are dated (removed from the store).***"

- Draw ERD, Reveal Relational Schema and Integrity Constraints and write SQL code to create the database.



Product(ProductID INTEGER)
IC: Primary Key ProductID

TransactionIsRelated(ProductID INTEGER, Amount REAL, Date TEXT, P_ProductID INTEGER, TransactionID INTEGER)

IC: Primary Key (P_PruductID,transactionID)
IC: Foreign Key (P_ProductID) References Product ON DELETE CASCADE

Moreover, we want to create a database that keeps transaction information with transaction date, ProductID (customer purchased), and amount_purchased, transactionID.
We can issue one transaction per item, and we will not keep records of a transaction if the purchased items are dated (removed from the store).

```
CREATE TABLE Product(ProductID INTEGER, PRIMARY KEY(ProductId));

CREATE TABLE TransactionIsRelated(ProductID INTEGER, Amount
REAL, Date TEXT, P_ProductID INTEGER, TransactionID
INTEGER,Primary Key (ProductID,TransactionID), Foreign Key
(P_ProductID) References Product(ProductID) ON DELETE CASCADE
);
```

Some more introduction to SQL

Lancaster University

- Inserting and querying

Adding and Deleting Tuples

Lancaster University

One can insert a single tuple using

INSERT INTO <relationalschema_for_the_table> Values <all_the_values>

Directives

INSERT INTO Students (sid, name, login, age, gpa)
VALUES (53688, 'Smith', 'smith@ee', 18, 3.2)

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4

DELETE and SELECT

Lancaster University

The DELETE statement is used to delete existing records in a table.

```
DELETE
FROM Relational Schema
WHERE <Conditions>
```

The SELECT statement is used to select data from a database. The data returned is stored in a result table, called the result set.

```
SELECT  <Attributes>
FROM  Schema
WHERE  <Conditions>
```

Adding and Deleting Tuples

Lancaster University

Can delete all tuples satisfying some condition (e.g., name = Shero):

```
DELETE
FROM Students S
WHERE S.name = 'Jones'
```

Powerful variants of these commands are available; more later!

sid	name	login	age	gpa
53688	Smith	smith@ee	18	3.2

The SQL Query Language

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Shero	shero@cs	18	3.2
53650	Shero	shero@math	19	3.8

Find all students with age 18

```
SELECT *
FROM Students S
WHERE S.age=18
```

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Shero	shero@cs	18	3.2

The SQL Query Language

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Shero	shero@cs	18	3.2
53650	Shero	shero@math	19	3.8

Names and logins of all students with age 18

```
SELECT  S.name, S.login
FROM  Students S
WHERE  S.age=18
```

name	login
Jones	jones@cs
Shero	shero@cs

Querying Multiple Relations

Lancaster University

Where does this selection come from? How does it operate?

Enrolled

sid	cid	grade
53831	Carnatic101	C
53831	Reggae203	B
53650	Topology112	A
53666	History105	B

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53650	Shero	shero@cs	18	3.2

Names and cids of Students who had an A From the cid they are enrolled

```
SELECT  S.name, E.cid
FROM  Students S, Enrolled E
WHERE  S.sid=E.sid AND E.grade="A"
```

S.name	E.cid
Shero	Topology112

Query Languages

Lancaster University

- For manipulation and retrieval of stored data
- Relational model supports simple yet powerful query languages
- Query languages are not as complex as programming languages
- They are specialized for data manipulation and retrieval

Relational Algebra

Lancaster University

- It is a mathematical query language
- Forms the basis of the SQL query language
- Relational Calculus is another mathematical query language but it is declarative rather than operational
- We will concentrate on relational algebra in this course