



Relational Model

André Restivo

Index

Introduction

Values

Primary Keys

Unique Keys

Foreign Keys

Constraints

Operations

Conversion

Example

Introduction

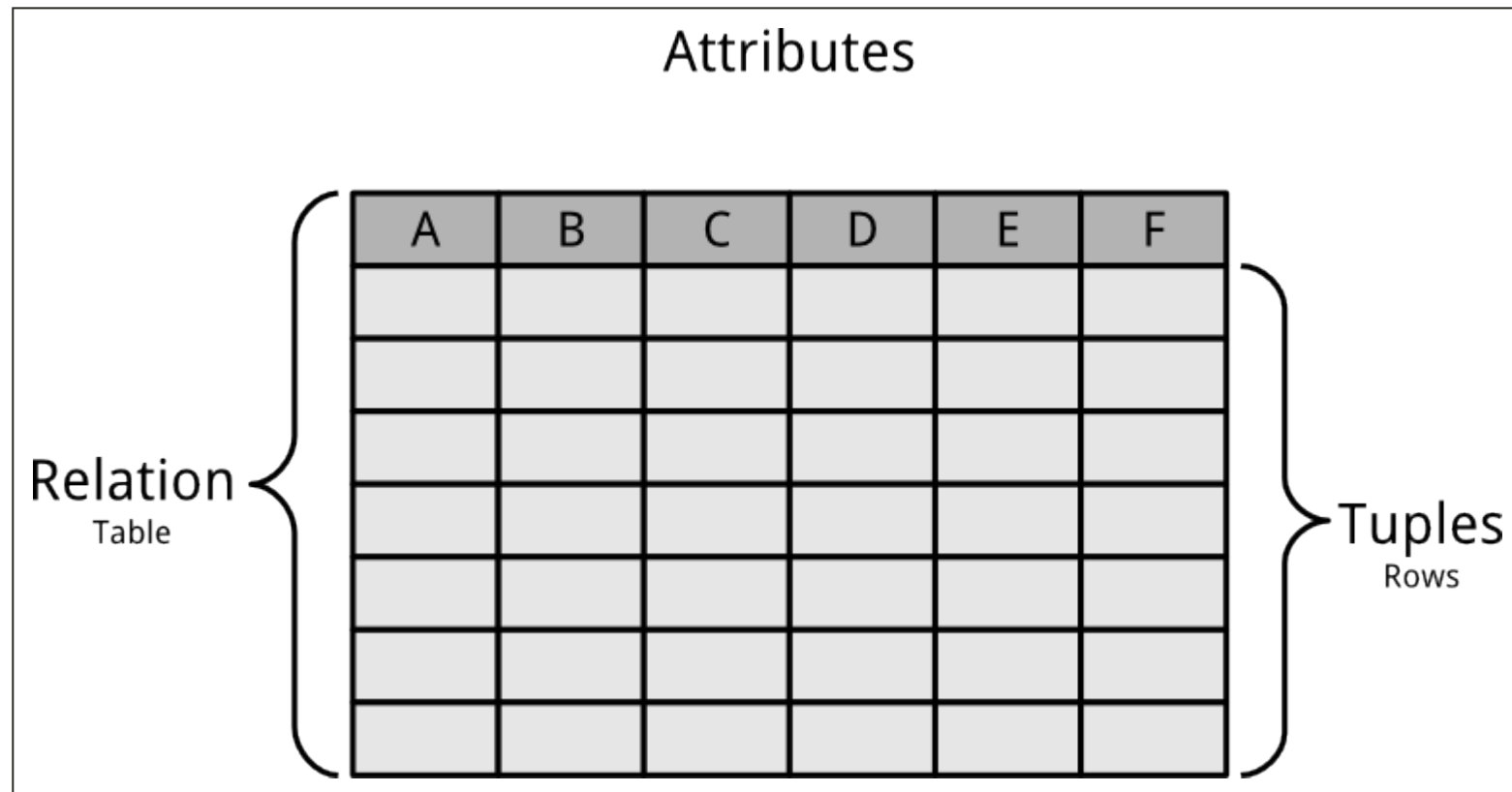
Relational Model

- A simple Database model based on first-order predicate logic.
- First formulated and proposed in 1969 by Edgar F. Codd.
- But still the most used model for databases.

Principles

- Relational Model:
 - Set of **relations** (tables)
 - Composed by **tuples** and **attributes** (rows and columns)
- Subject to **constraints**
- Relation:
 - Relation name
 - Attributes
 - Constraints

Relation



Tuples

- The lines of a relation.
- Ordered sequence of values.
- Tuples do not have a specific order between them.
- Tuple values are atomic (no composite or multi-value).

Notation

The notation used to represent a basic relation is the following:

Relation Name	attribute_1	attribute_2	...	attribute_n
---------------	-------------	-------------	-----	-------------

Example for an employee:

Employee	id	name	address	telephone
----------	----	------	---------	-----------

Values

Domain

- The set of possible values for a given attribute.
- Can be considered a **constraint** on the value of the attribute.
- Examples:
 - integer values
 - email addresses
 - text
- For certain attributes, **null** can be a possible value. Others don't allow the null value (not null constraint).

Notation

Notation for not null attributes:

Relation Name	attribute_1	attribute_2 (NN)	...	attribute_n
---------------	-------------	------------------	-----	-------------

Primary Keys

Definition

- Primary keys are a set of **one or more** attributes that uniquely define a tuple within a relation.
- They cannot have **repeated** values.
- They cannot have **null** values.
- There can **only** be **one** primary key in each relation.

Table with a Primary Key

Primary
Key
↓

A	B	C	D	E	F
1					
2					
4					
5					
6					
8					
9					

Notation

Notation for primary keys:


Relation Name	primary_key	attribute_1	...	attribute_n
---------------	-------------	-------------	-----	-------------

Example for an employee:

Employee	id	name	address	telephone
----------	----	------	---------	-----------

Double Primary Key

Primary Key



A	B	C	D	E	F
1	a				
1	b				
2	a				
2	c				
2	d				
3	b				
3	d				

Notation

Notation for double primary keys:

Relation Name	primary_key_1	primary_key_2	attribute_1	...	attribute_n
---------------	---------------	---------------	-------------	-----	-------------

Example for a phone list:

Phone	person	number	type
-------	--------	--------	------

Unique Keys

Definition

- Unique keys are similar to primary keys but allow null values.
- A relation can have several unique keys.

Table with an Unique Key

Primary Key		Unique Key			
A	B	C	D	E	F
1	a		20		
1	b		30		
2	a		10		
2	c		15		
2	d		50		
3	b		5		
3	d		55		

Notation

Notation for unique keys:

Relation Name	primary_key	attribute_1 (UK)	...	attribute_n
---------------	-------------	------------------	-----	-------------

Example for an employee:

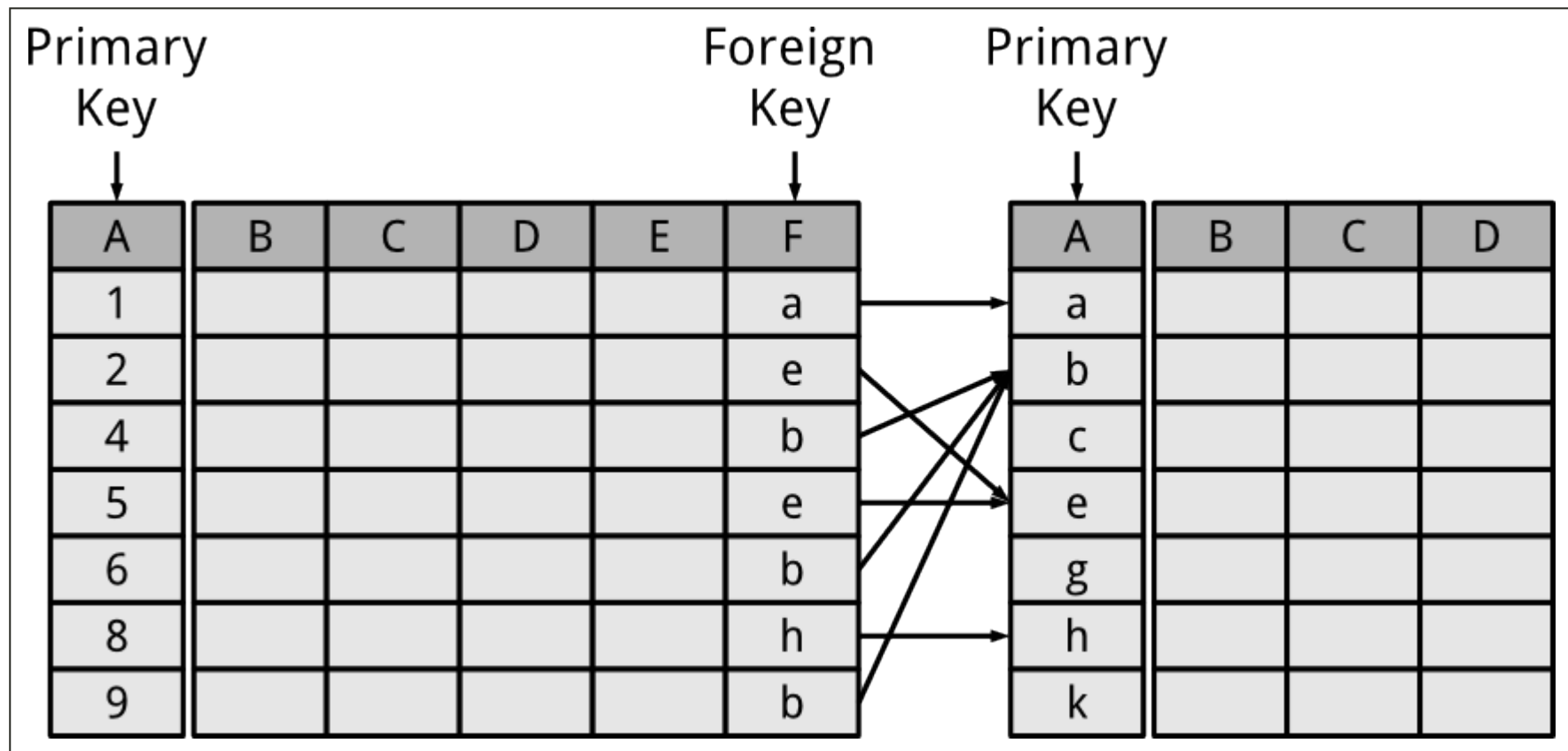
Employee	id	name	username (UK)	phone	address
----------	----	------	---------------	-------	---------

Foreign Keys

Definition

- An attribute (or set of attributes) that always matches a key attribute in another relation.
- Can be used to cross-reference relations.
- Possible to use the values of attributes in the referenced relation to restrict the domain of one or more attributes in the referencing relation.

Table with a Foreign Key



Notation

Notation for foreign keys:

Relation Name	primary_key	attribute_1	attribute_2	#foreign_key → referenced_relation
---------------	-------------	-------------	-------------	------------------------------------

Example for an employee and a department:

Employee	id	name	address	phone	#dep_id → Department
----------	----	------	---------	-------	----------------------

Department	id	name (UK)
------------	----	-----------

Constraints

Summary of Constraints

- Primary key values cannot be null.
- Key values (primary and unique) cannot have repeated values.
- Values have to belong to the attribute domain.
- Attributes with a not null constraint cannot have null values.
- Foreign key attributes can only have values that exist in the referenced relation.

Operations

List of Operations

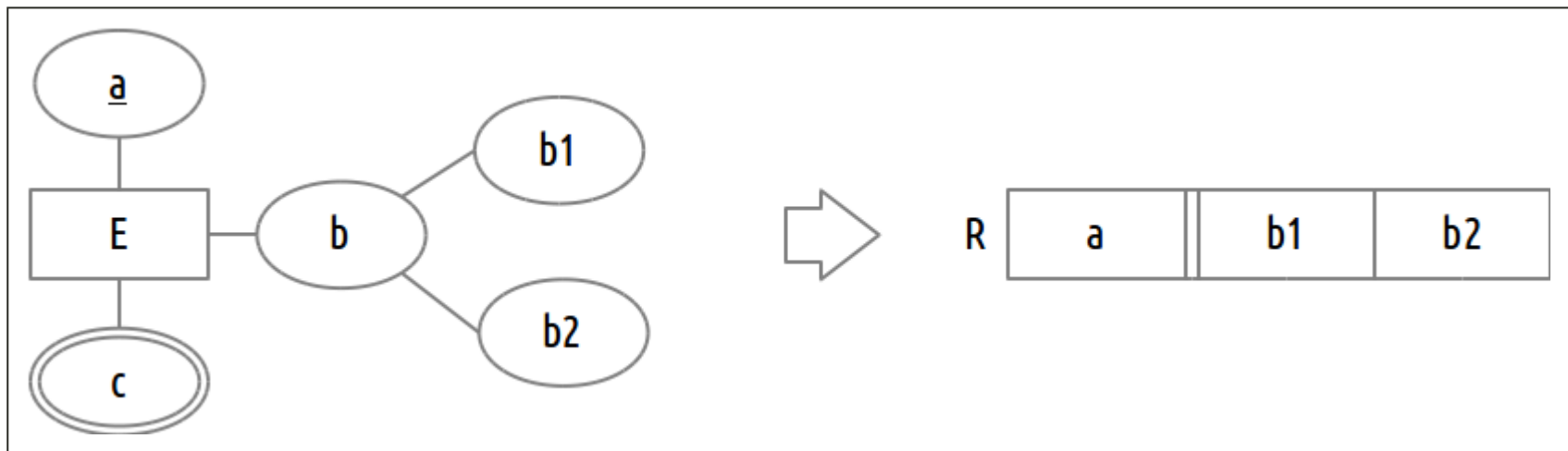
- **Insertion:** Inserts a new tuple into a relation.
- **Deletion:** Deletes one or more tuples from a relation.
- **Update:** Updates one or more tuples.

Entity Relationship to Relational

Step 1. Entity to Relation

- Every entity in the entity-relationship model becomes a relation in the relational model.
- Composite attributes are separated into several attributes.
- Key attributes become primary keys.
- Multi-valued attributes (see step 5).

Step 1. Entity to Relation

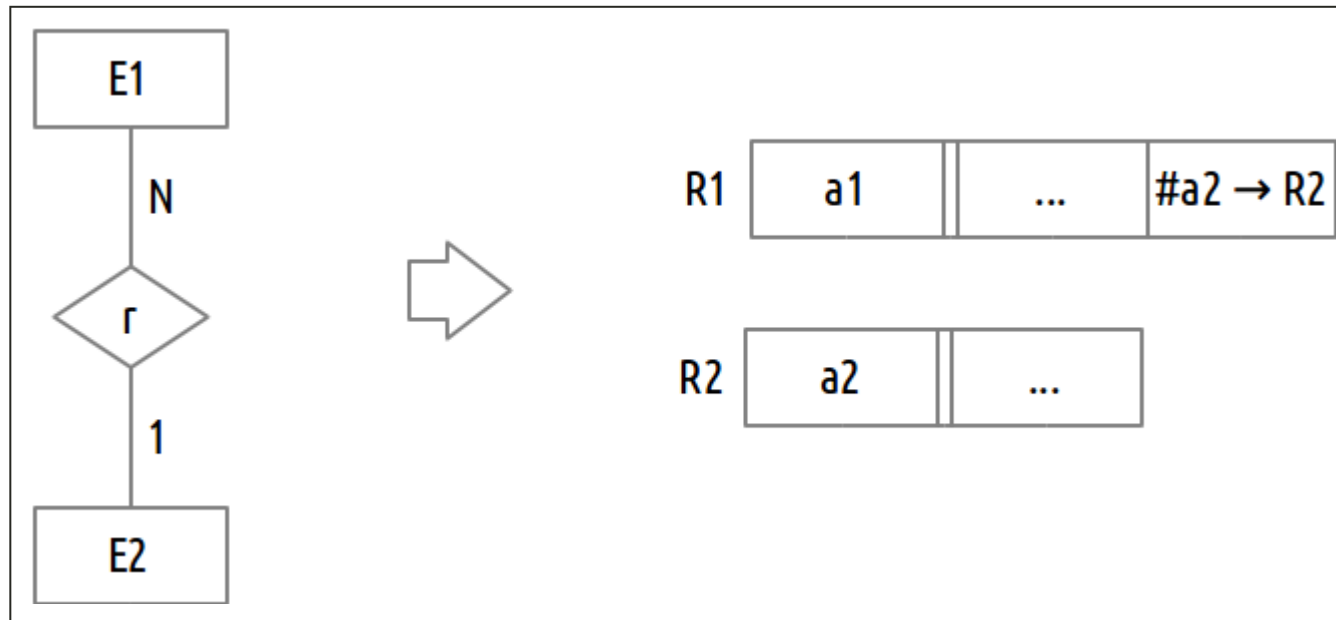


Step 2. Many-to-one relationships

- Add a foreign key from the relation in the many side of the relationship to the relation in the one side.
- If the participation of the entity on the many side of the relationship is total, add a not null constraint to the foreign key.

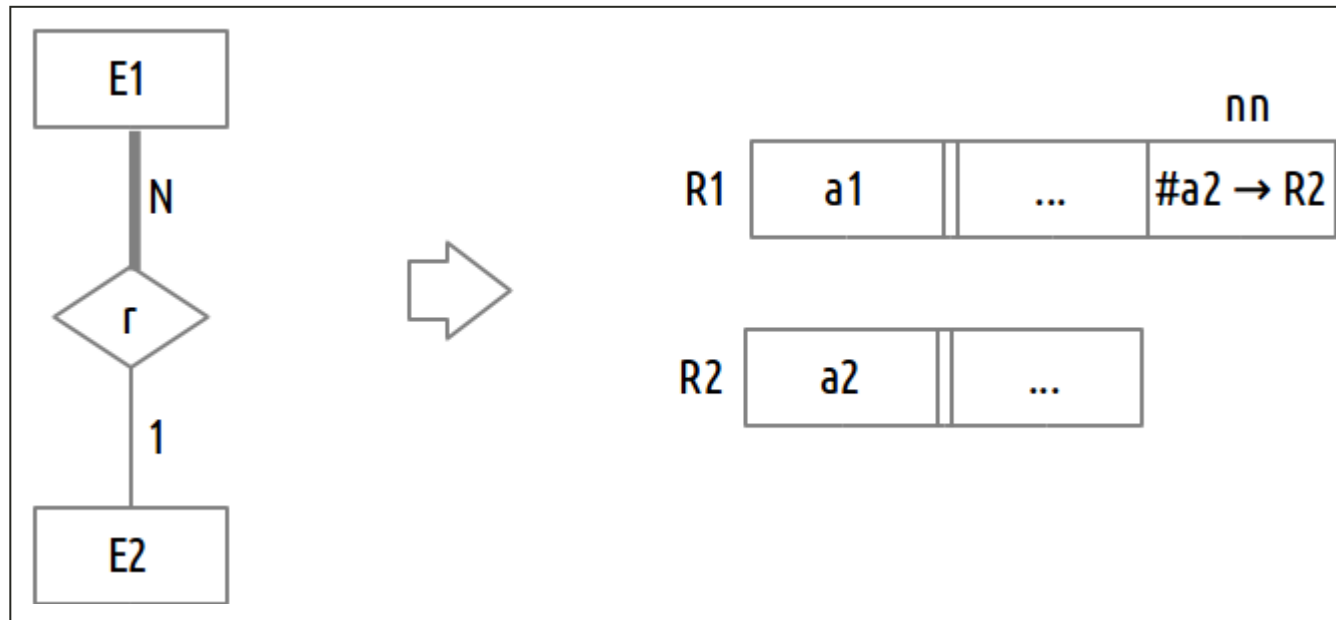
Step 2. Many-to-one relationships

Foreign key always in the many side



Step 2. Many-to-one relationships

If the many side has total participation in the relationship

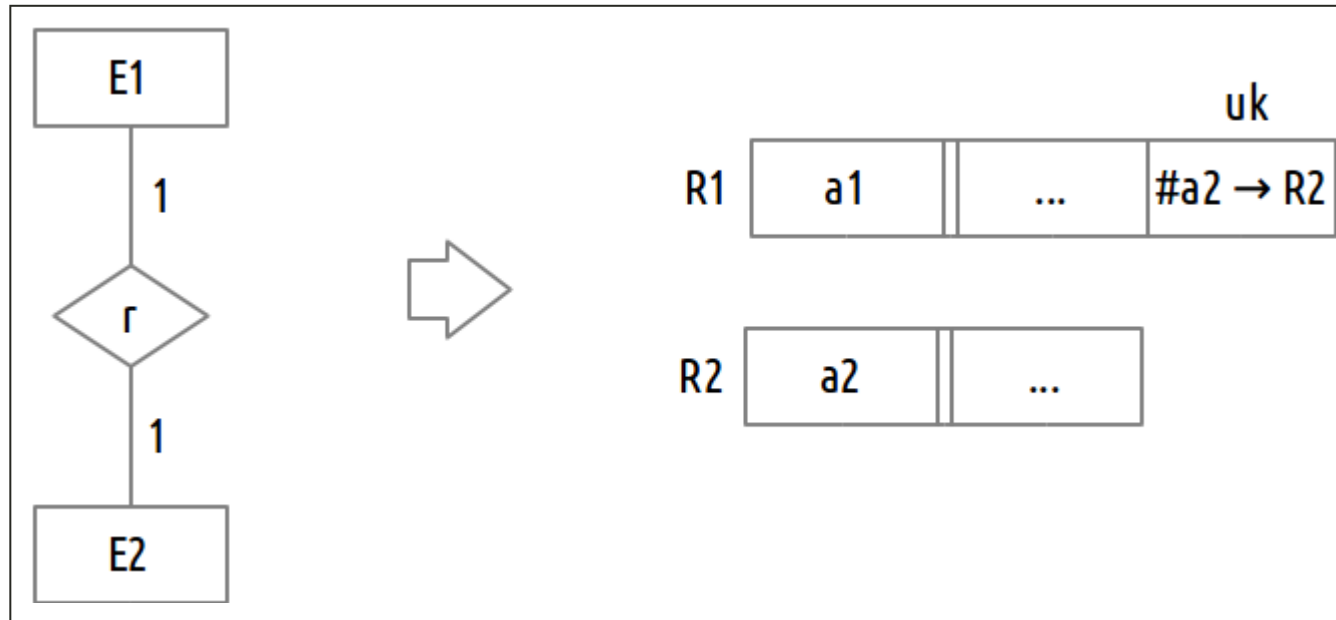


Step 3. One-to-one relationships

- Add a foreign key from one of the relations to the other.
 - If one of the entities has total participation in the relationship, add the foreign key to that relation and add a not null constraint to it.
 - If none of the entities has a total participation, pick the one expected to have less tuples.
- Add a unique key constraint to the foreign key.

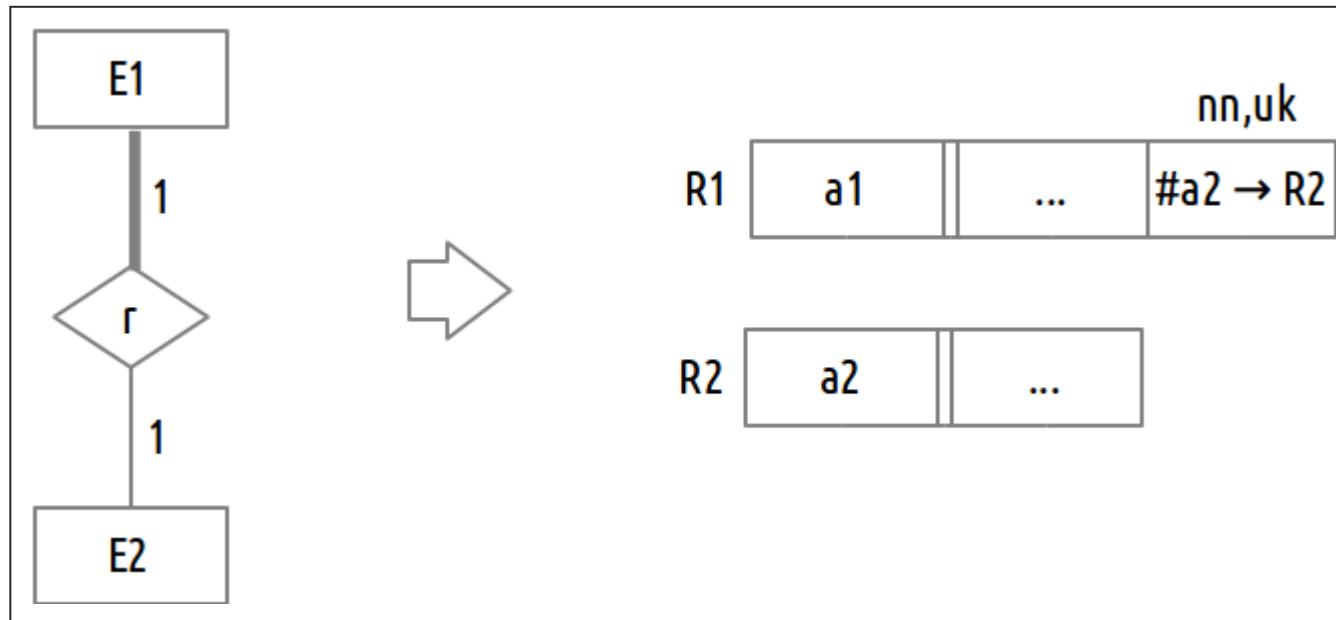
Step 3. One-to-one relationships

When none of the entities has a total participation in the relationship choose.



Step 3. One-to-one relationships

When one of the entities has a total participation in the relationship choose that one.

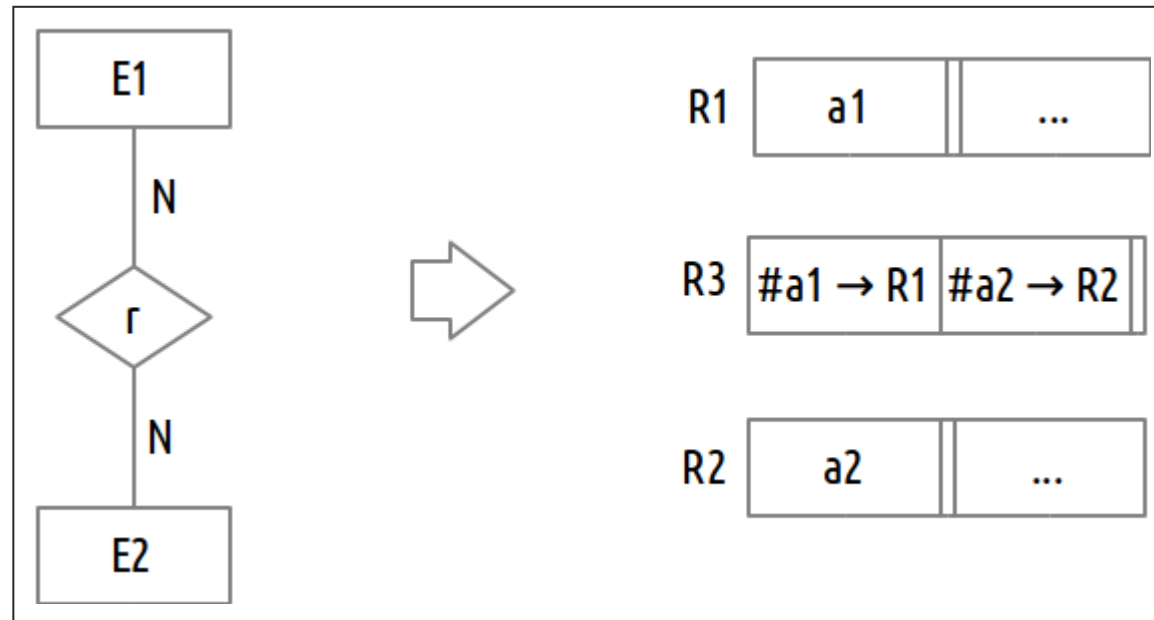


Step 4. Many-to-many relationships

- Add a new relation to the model.
- Add foreign keys to the new relation referencing both relations.
- Select both foreign keys as the (double) primary key of the new relation.

Step 4. Many-to-many relationships

Always create a new relation with a double foreign key

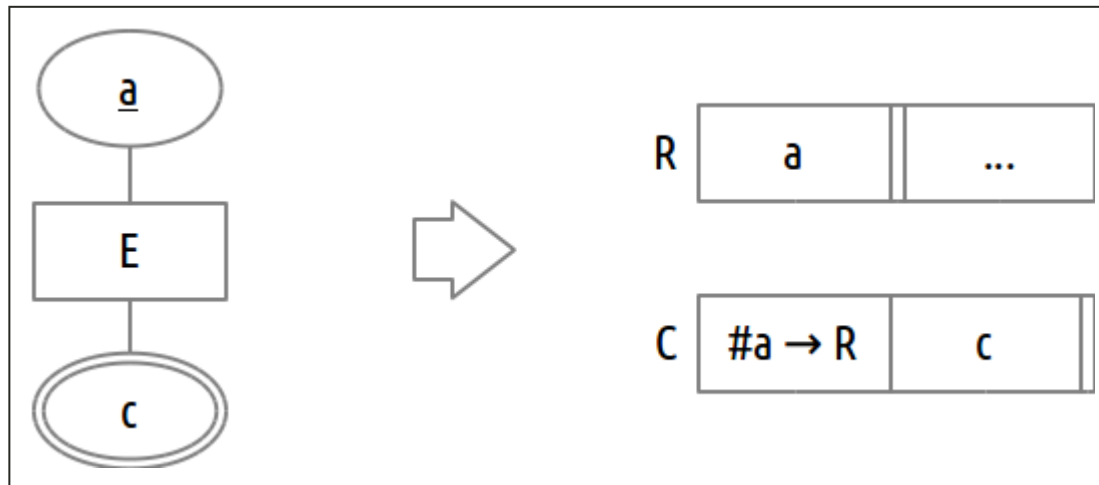


Step 5. Multi-valued attributes

- For each multi-valued attribute create a new relation.
- Add the multi-valued attribute and a foreign key, referencing the relation containing the attribute, to the new relation.
- Both attributes (foreign key and attribute) become the primary key of the new relation.

Step 5. Multi-values attributes

Always create a new relation with a double foreign key



Example

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment
----------	----	------	------	--------	--------	-----------

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment
Department	number	name				

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment
Department	number	name				
Project	number	name				

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment
Department	number	name				
Project	number	name				
Car	plate					

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

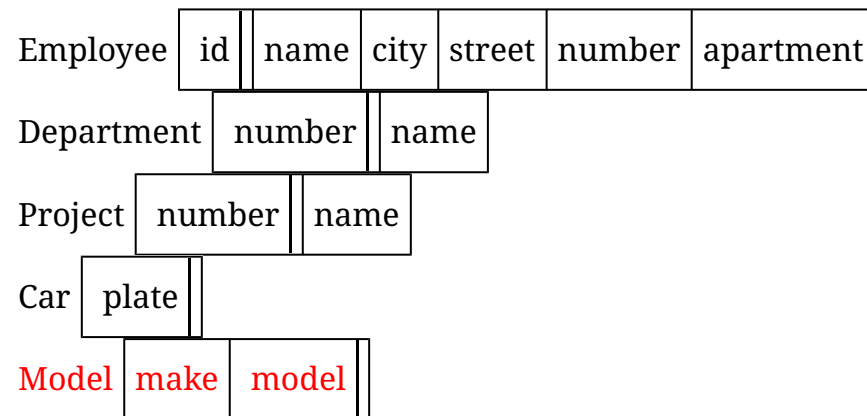
uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p



Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment
Department	number	name	#manager → Employee (UK)			
Project	number	name				
Car	plate					
Model	make	model				

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment
Department	number		name	#manager → Employee (UK)		
Project	number	name				
Car	plate	#used_by → Employee (UK, NN)				
Model	make	model				

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment	#number → Department (NN)
Department	number		name	#manager → Employee (UK)			
Project	number		name				
Car	plate	#used_by → Employee (UK, NN)					
Model	make	model					

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment	#number → Department (NN)
Department	number		name	#manager → Employee (UK)			
Project	number	name	#controlled → Employee				
Car	plate	#used_by → Employee (UK, NN)					
Model	make	model					

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment	#number → Department (NN)
Department	number		name	#manager → Employee (UK)			
Project	number	name	#controlled → Employee				
Car	plate	#used_by → Employee (UK, NN)				#(make, model) → Model (NN)	
Model	make	model					

Example

Employee (id, name, address (city, street, number, apartment))

Department (number, name)

Project (number, name)

Car (plate)

Model (make, model)

manages (Employee, Department) 1:1 p/p

uses (Employee, Car) 1:1 p/t

belongsTo (Employee, Department) N:1 t/p

controls (Employee, Project) 1:N p/p

itsA (Car, Model) N:1 t/p

worksAt (Employee, Project, hours) N:N p/p

Employee	id	name	city	street	number	apartment	#number → Department (NN)
----------	----	------	------	--------	--------	-----------	---------------------------

Department	number	name	#manager → Employee (UK)
------------	--------	------	--------------------------

Project	number	name	#controled → Employee
---------	--------	------	-----------------------

Car	plate	#used_by → Employee (UK, NN)	#(make, model) → Model (NN)
-----	-------	------------------------------	-----------------------------

Model	make	model
-------	------	-------

WorksAt	#id → Employee	#number → Project	hours
---------	----------------	-------------------	-------