Unit 1:
Relational Databases

1.1. Fundamentals
1.2. The Relational Data Model
1.3. Interpretation of a Relational Database

UD 1.3 Interpretation of a Relational DB

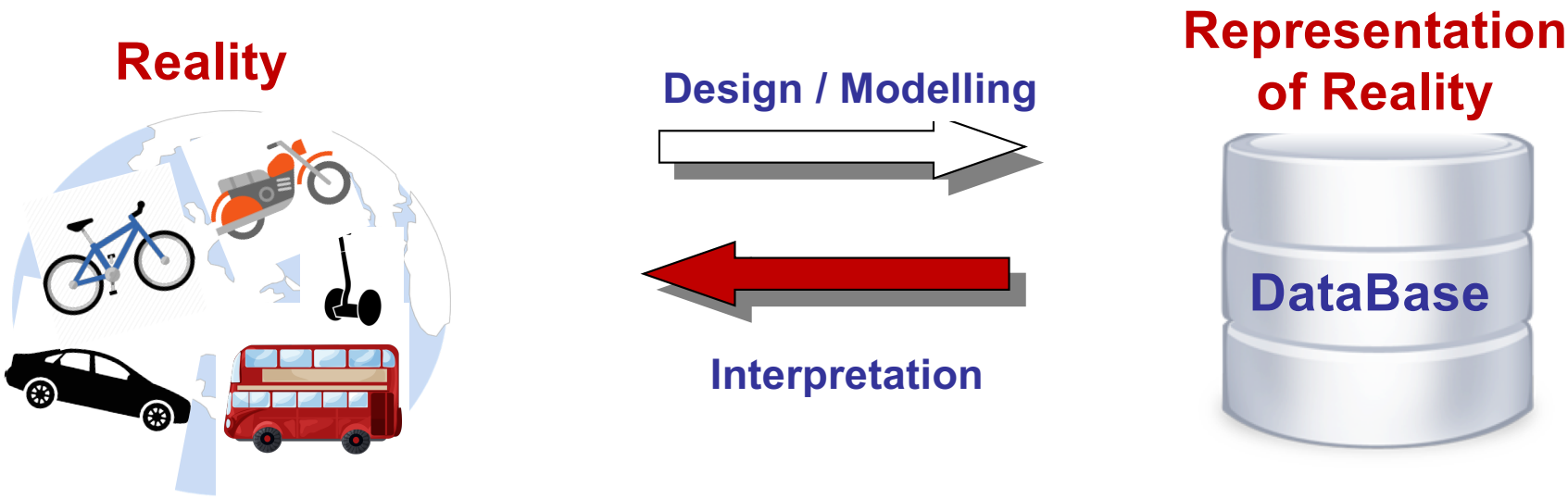
1 Introduction: Representation of reality

2 The “Music Library” Database

3 Interpreting database schemas

4 Examples

1 Introduction: Representation of reality



- For each object in reality about which we want to have information we define a relation whose attributes denote the most significant properties of these objects (code, name, …) in such a way that each tuple which is present in this relation must be interpreted as a particular instance of an object.
- In order to represent associations between objects we use explicit references through attributes which identify each object.
- Associations between objects where the cardinality is many-to-many require an extra relation

UD 1.3 Interpretation of a Relational DB

1 Introduction: Representation of reality

2 The “Music Library” Database

3 Interpreting database schemas

4 Examples

Music Library

Canción (song)

cod: song code (id).

título: Song title.

duración: Length of the song.

Está (is_in)

It stores what songs are included in each record, where "***can***" is the code of a song appearing in the record "**cod**".

Companyia (company)

cod: record company (record label) code.

nombre: company name.

dir: Address of the company.

fax: Fax number of the company.

tfno.: Phone number of the company.

Disco (record)

cod: record code (id).

nombre: record name.

fecha: Publishing date.

cod_comp: Code of the record company which has published this record.

cod_gru: Code of the music group (band) which has recorded this record.

Music Library

Grupo (group/band)

cod: Group (band) code.

nombre: Name of the group.

fecha: Date of the group foundation.

país: Country where the group was created.

Artista (artist)

dni: artist id.

nombre: name of the artist.

Club (fan club)

cod: fan club code (id).

nombre: name of the club.

sede: Address of the main office.

num: number of members of the club.

cod_gru: code of the group which the club is fan of.

Pertenece (belongs_to)

It contains the group members information: The artist "**dni**" is member of the group "**cod**" performing the function "funcion" (e.g. plays the guitar, sings,…).

Music Library

Canción (***cod***: integer, *título*: char(30), *duración*: real)

PK:{cod}

NNV:{título}

Compañía (***cod***: char(3), *nombre*: char(30), *dir*: char(30), *fax*: char(15), *tfno*: char(15))

PK:{cod}

NNV:{nombre}

Disco (***cod***: char(3), *nombre*: char(30), *fecha*: date, *cod_comp*: char(3), *cod_gru*: char(3)

PK:{cod}

FK:{cod_comp} → Compañia

NNV:{cod_comp}

FK:{cod_gru} → Grupo

NNV:{cod_gru}

Esta (***can***: integer, ***cod***: char(3))

PK:{can,cod}

FK:{can} → Cancion

FK:{cod} → Disco

Music Library

Grupo (***cod***: char(3), *nombre*: char(30), *fecha*: date, *país* :char(10))

PK:{cod}

NNV:{nombre}

UNI:{nombre}

Artista (***dni***: char(10),*nombre*: char(30))

PK:{dni}

NNV:{nombre}

Club (***cod***: char(3), *nombre*: char(30), *sede*: char(30), *num*: integer, *cod_gru*: char(3))

PK:{cod}

FK:{cod_gru} → Grupo

NNV:{cod_gru}

UNI:{cod_gru}

NNV:{nombre}

Pertenece (***dni***: char(10), ***cod***: char(3), *función*: char(15))

PK:{dni,cod}

FK:{dni} → Artista

FK:{cod} → Grupo

UD 1.3 Interpretation of a Relational DB

1 Introduction: Representation of reality

2 The “Music Library” Database

3 Interpreting database schemas

4 Examples

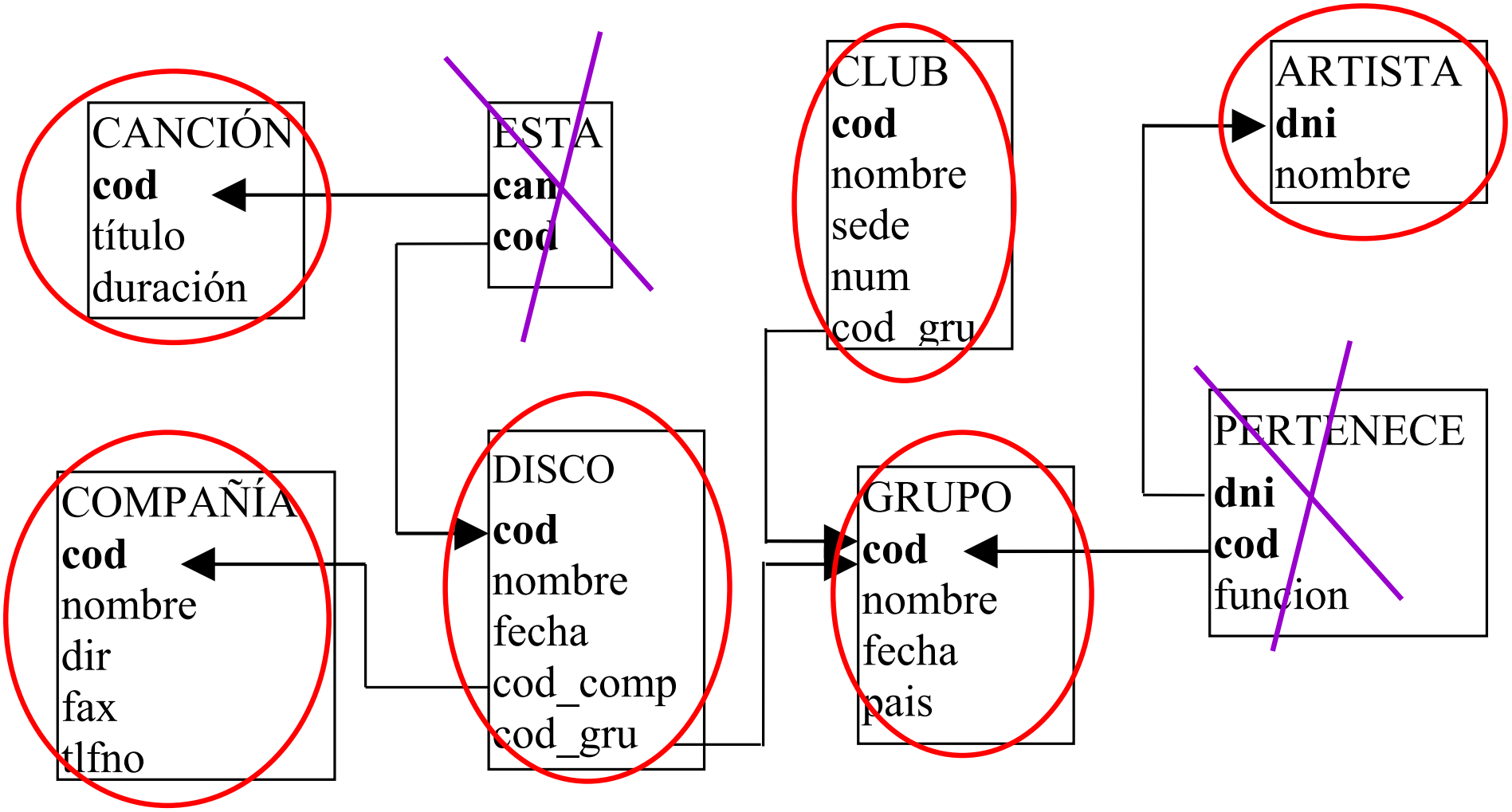
3 Interpreting database schemas

Tables:

Objects are represented by tables for which none of the attributes of their PK refer to other tables (they have existence on their own).

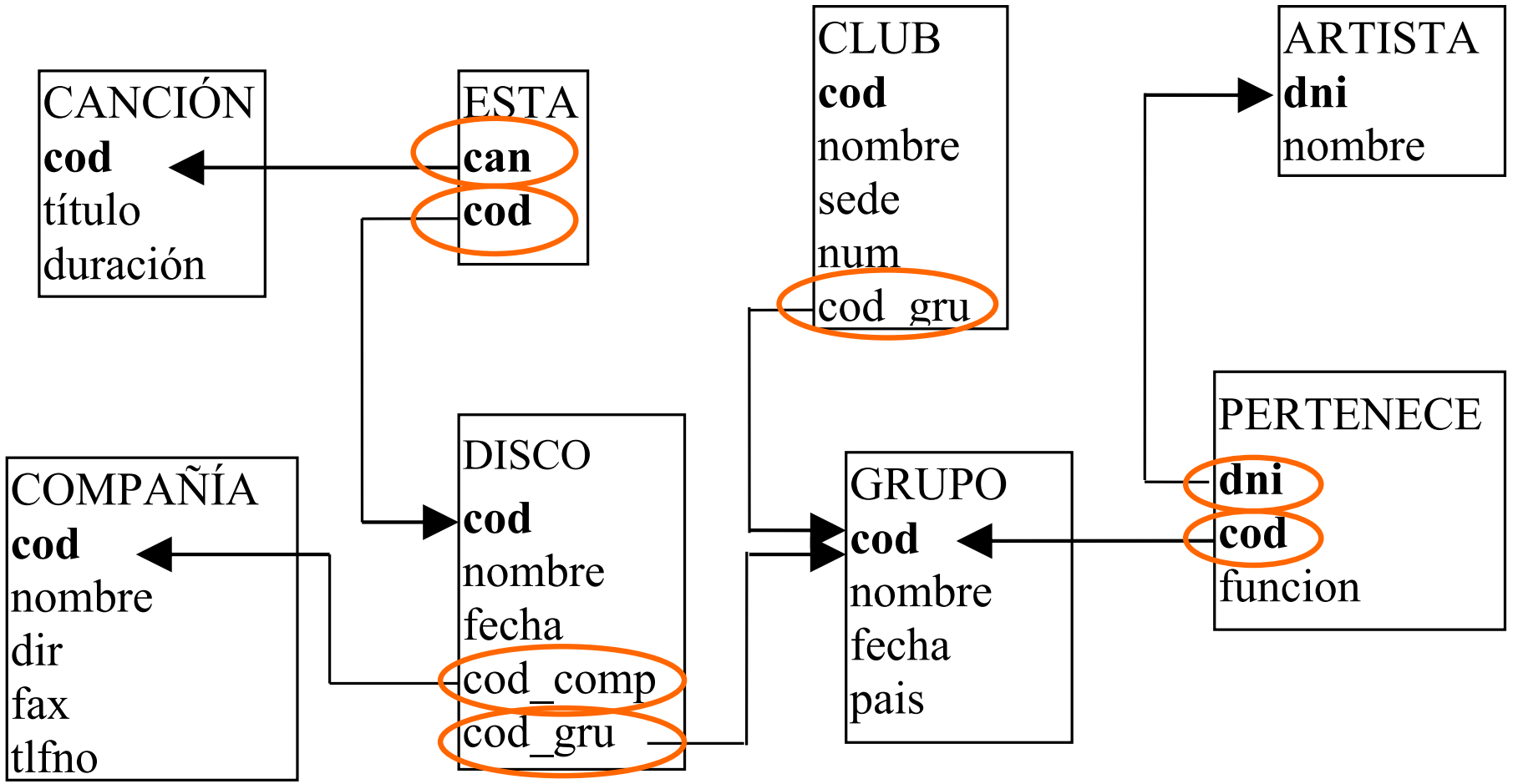
This is not true for hierarchical relationships (specialization).

The rest of tables represent relationships (non-entity tables).


CANCIÓN
cod
título
duración
ESTA
can
cod
CLUB
cod
nombre
sede
num
cod_gru
ARTISTA
dni
nombre
COMPAÑÍA
cod
nombre
dir
fax
tlfno
DISCO
cod
nombre
fecha
cod_comp
cod_gru
GRUPO
cod
nombre
fecha
pais
PERTENECE
dni
cod
funcion

Attributes

- Represent properties of objects (if they are not FK)
- If they are FK, they represent relationships between objects

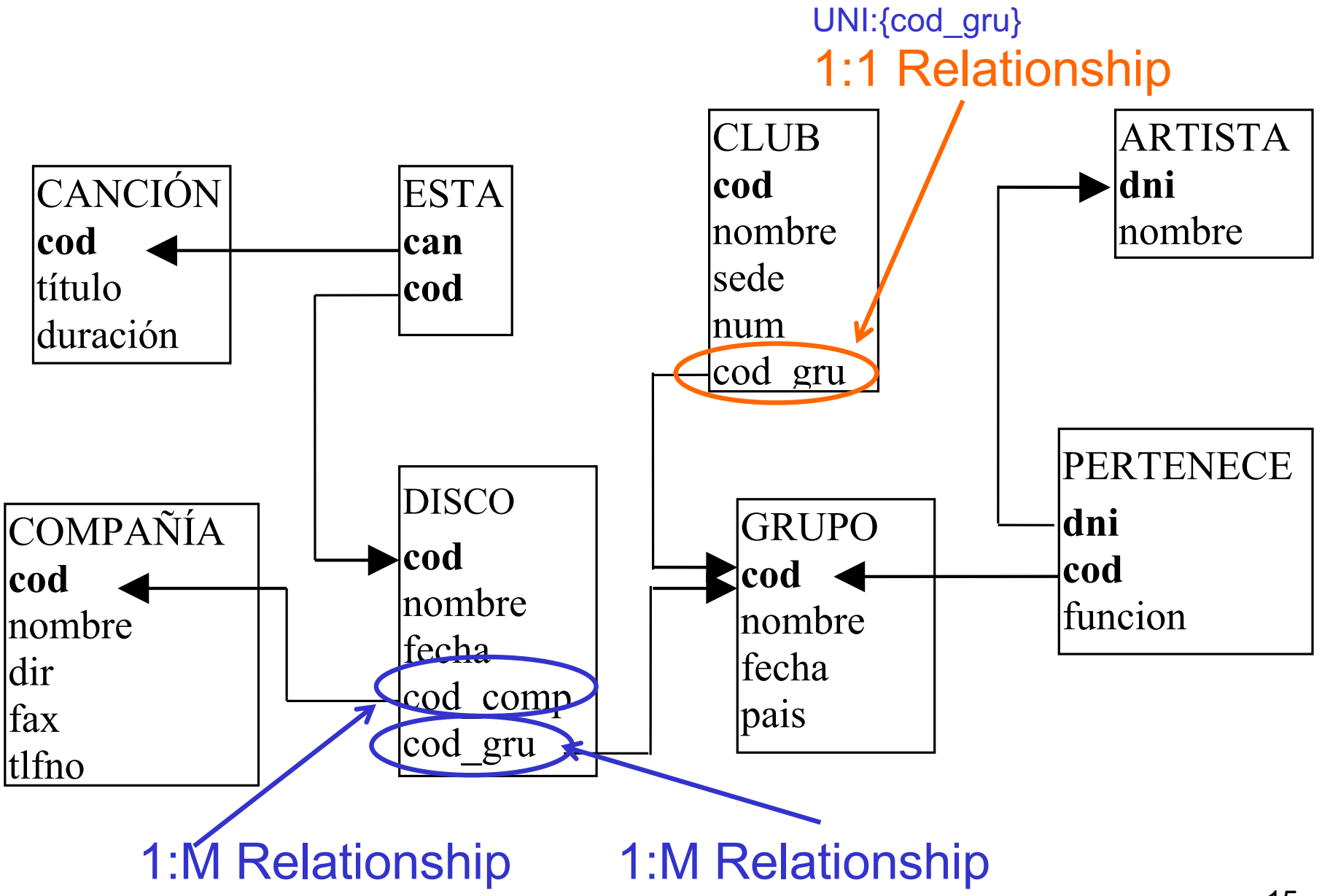

CANCIÓN
cod
título
duración
ESTA
can
cod
CLUB
cod
nombre
sede
num
cod_gru
ARTISTA
dni
nombre
PERTENECE
dni
cod
funcion
COMPAÑÍA
cod
nombre
dir
fax
tlfno
DISCO
cod
nombre
fecha
cod_comp
cod_gru
GRUPO
cod
nombre
fecha
pais

Relationship types

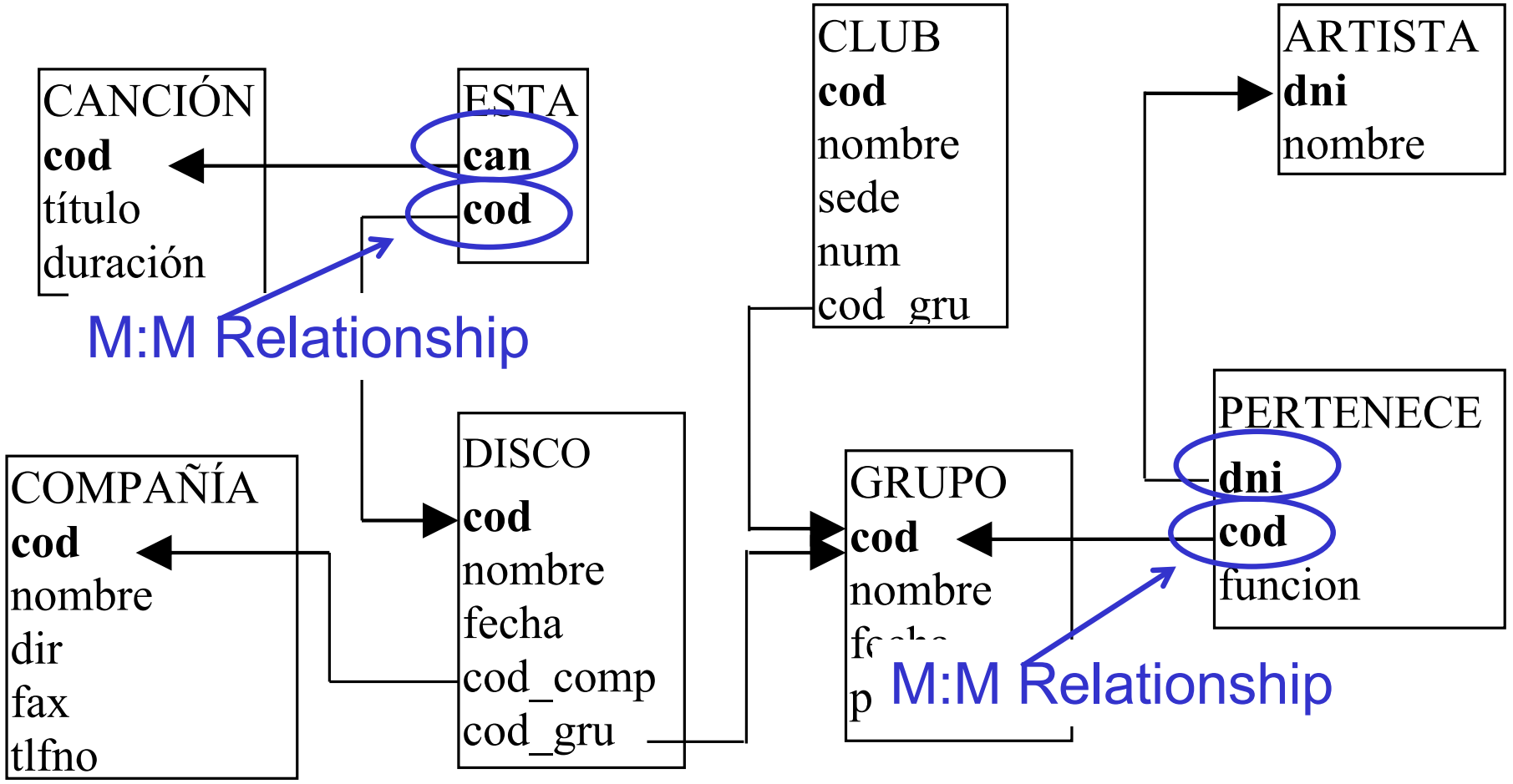
1. The FK is in a table representing an object
   - FK doesn't have UNI constraint:
     1:M relationship (one-to-many)
   - FK has UNI constraint
     1:1 relationship (one-to-one), or 0:1

2. The FK is in a non-entity (object) table. The PK is composed of two FK
   M:M relationship (many-to-many)

Relationship types



Relationship types



Constraints

Non null value

- If a FK from S to R has a NNV constraint, then every object in S is associated with one object in R.
- If a FK from S to R does not have a NNV constraint, then every object in S is not necessarily associated with any object in R

Uniqueness

- If a FK from S to R has a uniqueness constraint, then every object in R can at most be associated with one object in S.

UD 1.3 Interpretation of a Relational DB

1 Introduction: Representation of reality

2 The “Music Library” Database

3 Interpreting database schemas

4 Examples

Geographical Information

RIVER (rcod: d_rcod, name: d_nom, length: d_long, mcod: d_mcod)

PK: {rcod}

FK: {mcod} -> SEA

SEA (mcod: d_mcod, name: d_nom, details: d_det)

PK: {mcod}

PROVINCE (pcod: d_pcod, name: d_nom, extension: d_ext)

PK: {pcod}

CROSSES (rcod: d_rcod, pcod: d_pcod, km: d_km)

PK: {pcod,rcod}

FK: {pcod} -> PROVINCE

FK: {rcod} -> RIVER

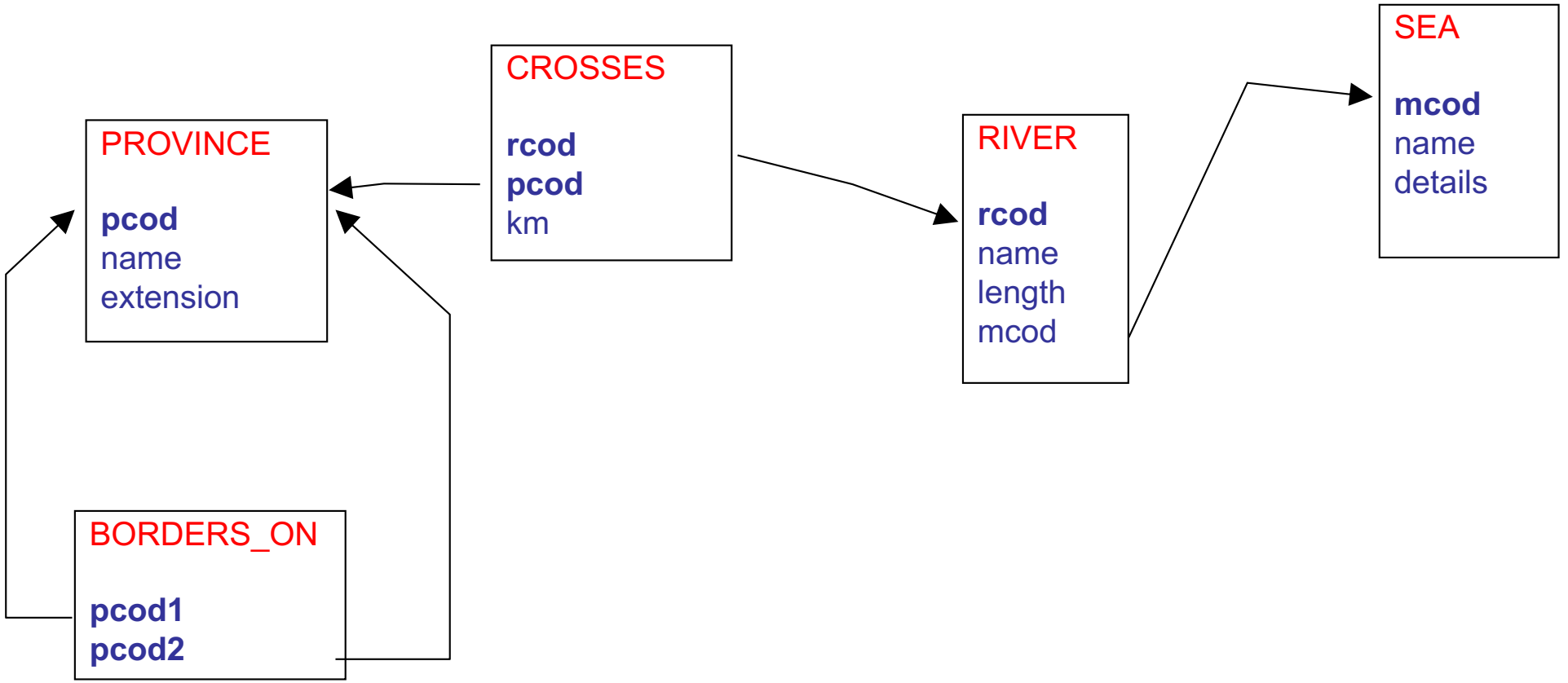
BORDERS_ON (pcod1: d_pcod, pcod2: d_pcod)

PK: {pcod1,pcod2}

FK: {pcod1} -> PROVINCE

FK: {pcod2} -> PROVINCE

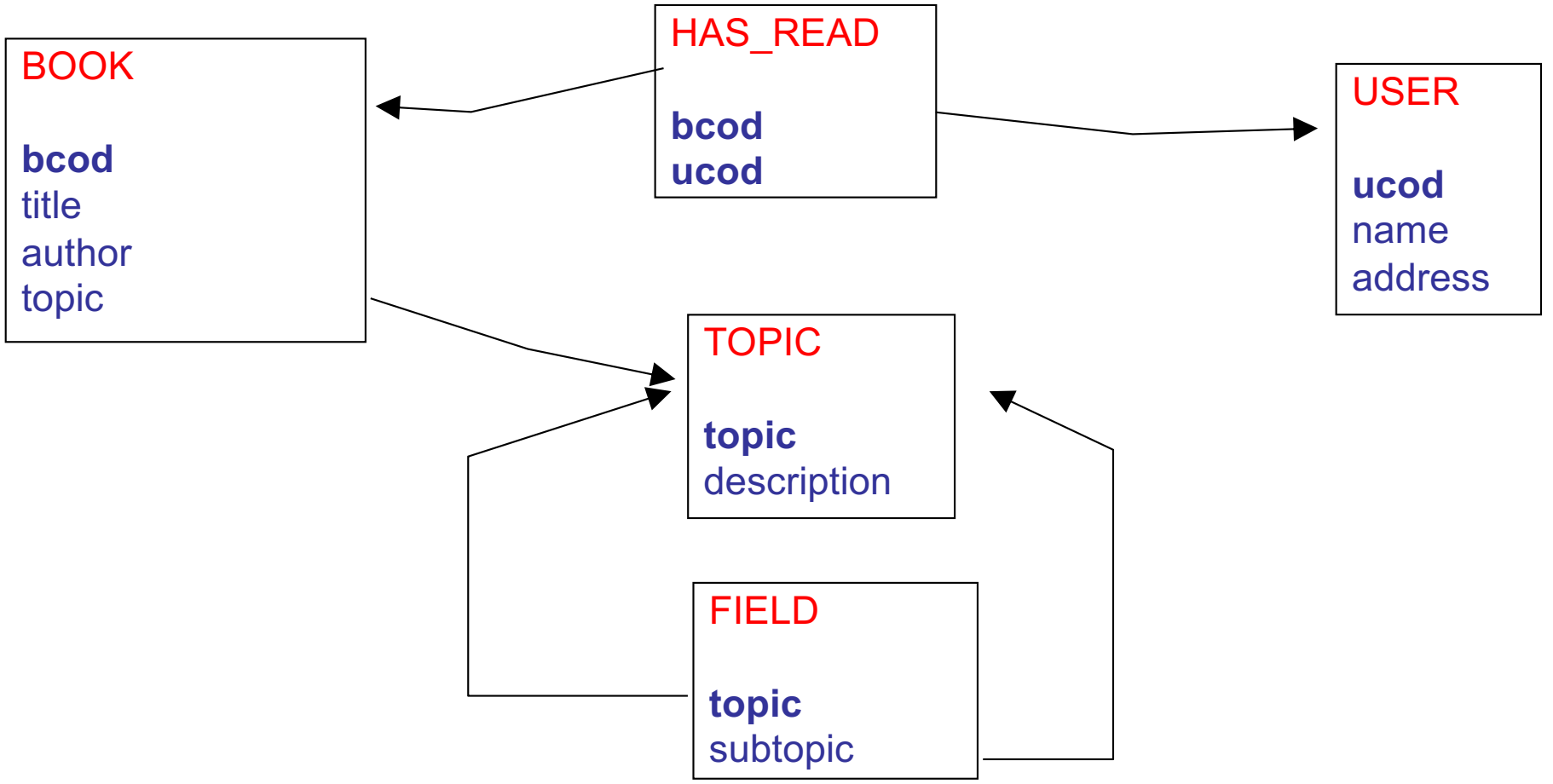
Geographical Information



Schema: Geographical Information

1. Can a river flow into two seas?
2. Can a river cross two provinces?
3. Can a river cross the same province twice?
4. Can a province border on itself?
5. How many seas, as a maximum, can a river flow into? And the minimum?

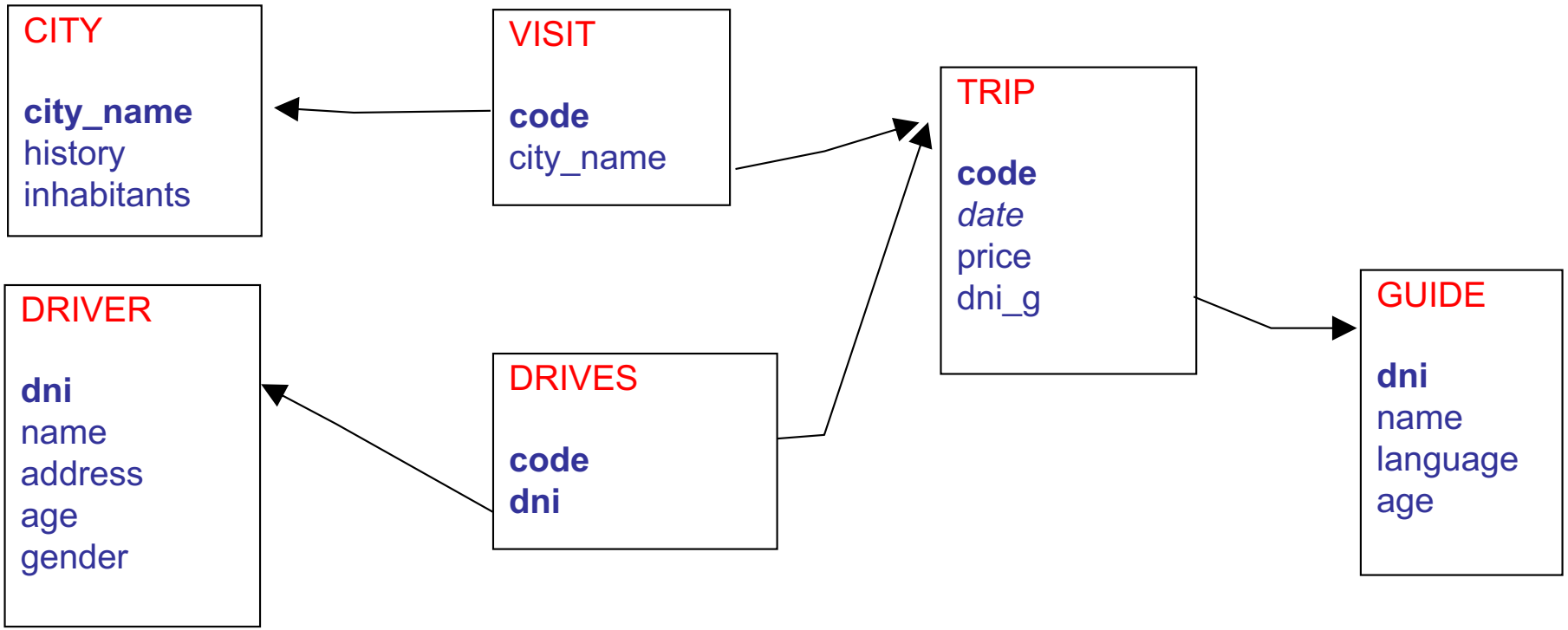
Library


BOOK
bcod
title
author
topic
HAS_READ
bcod
ucod
USER
ucod
name
address
TOPIC
topic
description
FIELD
topic
subtopic

Library

1. Can a user read more than one book?
2. Can a user read the same book more than once?
3. Can a book have more than one author?
4. Can a book have more than one topic?
5. Can a topic be a subtopic of itself?
6. Can a book be read by two different users?

Travel Agency



Travel Agency

1. Can the same trip visit the same city twice?
2. Can a guide speak two languages?
3. Can a driver be in two trips at the same time (date)?
4. Can a driver be also a guide?
5. How many drivers are there, as a minimum, in each trip?
6. Can a guide participate in more than one trip?

Cycling race

Team

teamname: name of the team.
director: name of the team director.

Cyclist

cnum: cyclist number assigned to the cyclist during the race.
name: cyclist name.
age: age of the cyclist.
teamname: name of the cyclist team.

Stage

stagenum: stage number (in the race).
km: How many kilometers the stage has.
departue: name of the city where the stage starts.
arrival: name of the city where the stage finish.
cnum: number of the cyclist who has won the stage.

Cycling race

Climb

climbname: name of the climb.
height: maximum height in the pass.
category: category of the pass: 1ª/primera (first), especial (special), ….
slope: slope of the climb (in %).
stagename: stage number where the climb is climbed.
cnum: number of the cyclist who has won the climb

Jersey

code: code of the jersey.
type: indicates the prize level of the jersey.
color: color of the prize.
prize: how much money the cyclist wins if he finishes wearing this jersey.

Wear

The cyclist with number '*cnum*' has worn the jersey identified by '***code***' at the stage with number '***stagenum***'.

Cycling race

TEAM (**teamname**: char(25), director: char(30))
PK:{teamname}

CYCLIST (**cnum**: integer, name: char(30), age: integer, teamname :char(25))
PK:{cnum} FK:{teamname}→ TEAM
NNV:{teamname} NNV:{name}

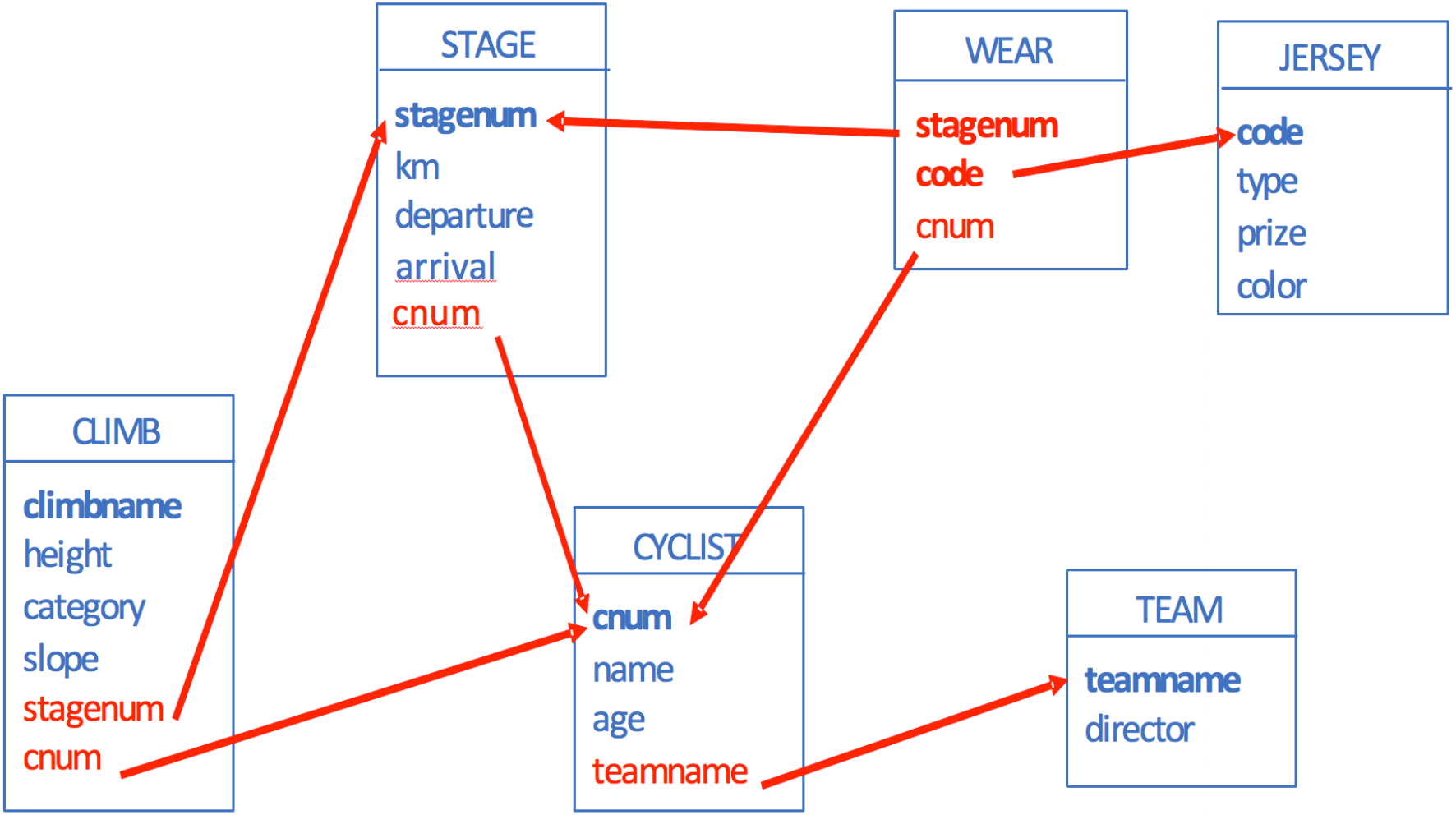
STAGE (**stagenum**: integer, km: integer, departure: char(35), arrival: char(35),
cnum: Integer)
PK:{stagenum} FK:{cnum}→ CYCLIST

JERSEY (**code**: char(3, type: char(30), prize: integer, color: char(25))
PK:{code}

CLIMB (**climbname**: char(30), height:I nteger, category: char(1),
slope:real, stagenum:integer, cnum:integer)
PK:{climbname} FK:{stagenum}→ STAGE
FK:{cnum}→ CYCLIST NNV:{stagenum}

WEAR (**stagenum**: integer, **code**: char(3), cnum: integer,)
PK:{stagenum,code}
FK:{stagenum}→ STAGE FK:{cnum}→ CYCLIST
FK:{code}→ JERSEY NNV:{cnum}

Cycling race



Schema: Cycling race

1. Can a cyclist belong to more than one team?
2. Can a cyclist wear more than one jercey during the race ("tour")?
3. Can a cyclist wear more than one jersey in the same stage?
4. Can a climb appear in more than one stage?
5. Can a cyclist win more than one stage?