

# Basi di dati

**Giuseppe De Giacomo**

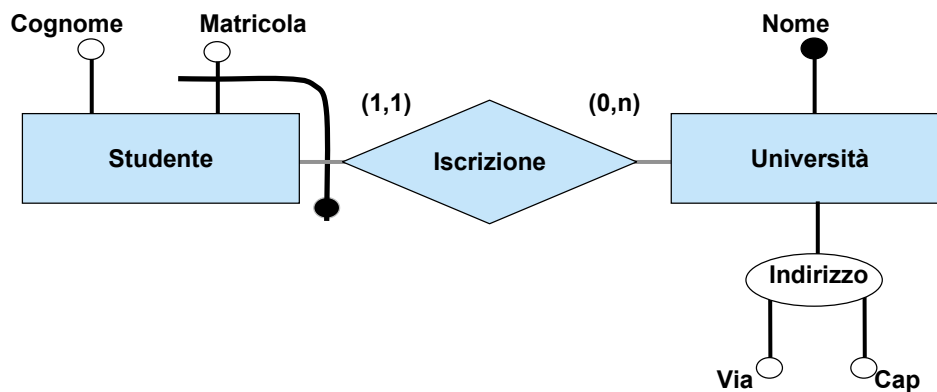
**Dipartimento di Informatica e Sistemistica "Antonio Ruberti"  
Università di Roma "La Sapienza"**

Anno Accademico 2006/07  
Canale M-Z

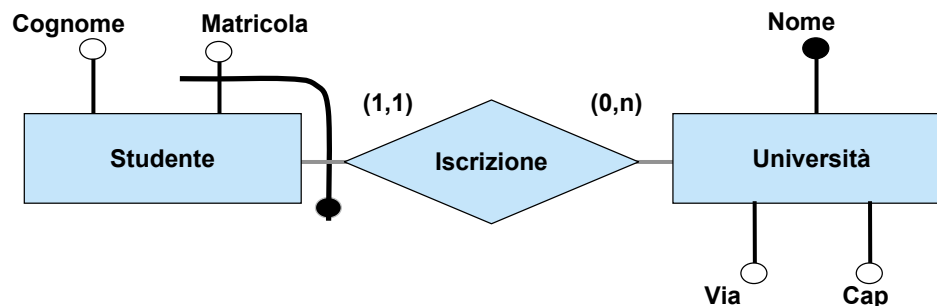
*Appunti della lezione del 21/11/2006 sulla traduzione in relazionale di ER  
ristrutturato*

<http://www.dis.uniroma1.it/~deggiacomo/didattica/basidati/>

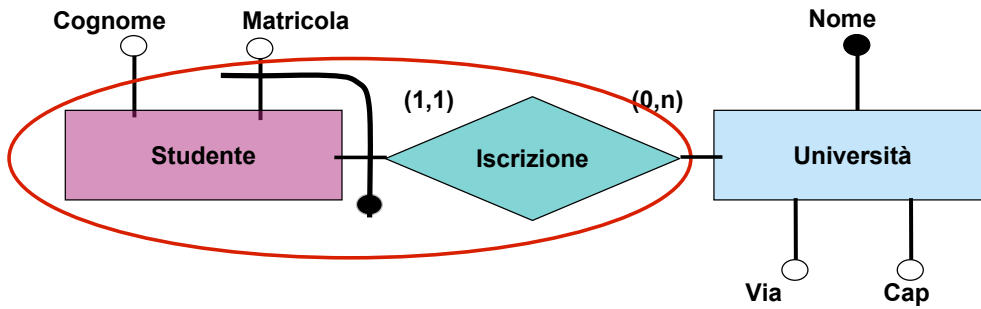
ER



ER ristrutturato



ER ristrutturato



SQL

```

create table StudenteIscrizione(
  matricola varchar(20) not null,
  cognome varchar (30) not null,
  universita varchar(30) not null,
  primary key(matricola, universita),
  foreign key (universita) references
    Universita(nome)
)
create table Universita(
  nome varchar(30) primary key,
  via varchar(40) not null,
  cap varchar(10) not null,
)
  
```

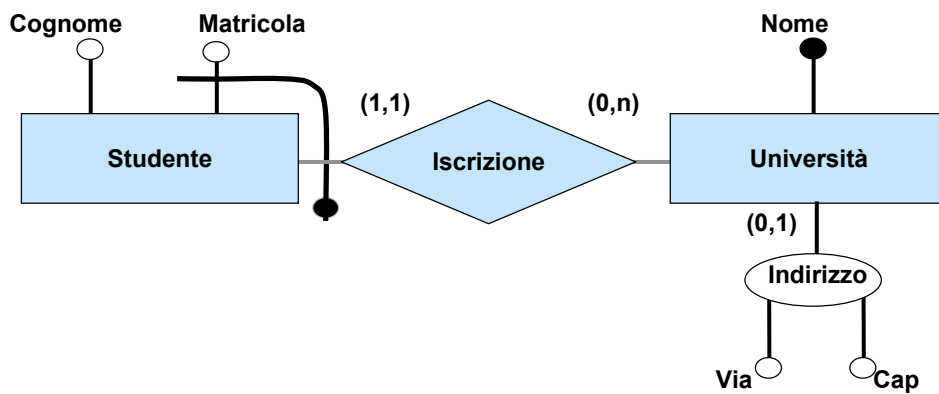
Schema Relazionale

```

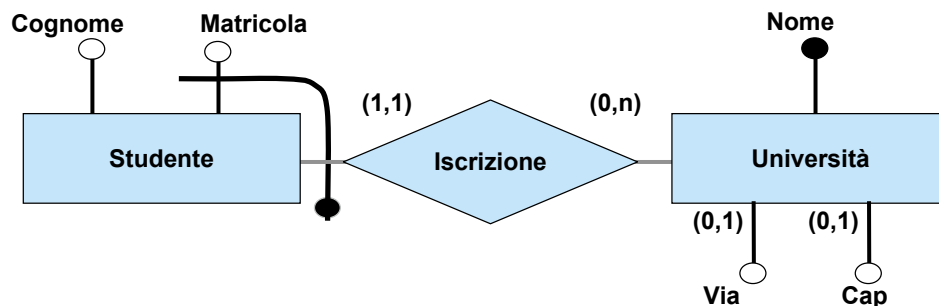
StudenteIscrizione (matricola, cognome, universita)
FK: StudenteIscrizione[universita] ⊆ Universita[nome]

Universita(nome,nabitanti)
  
```

ER

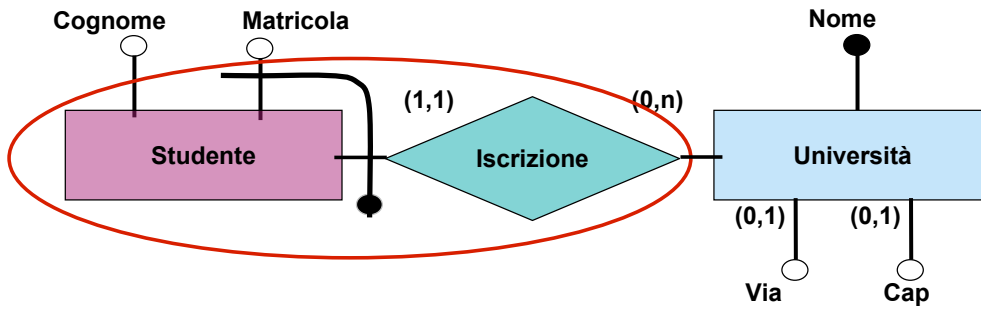


ER ristrutturato



Vincolo Estremo: Via e Cap o sono

### ER ristrutturato



### SQL

```
create table Studentelscrizione(
  matricola varchar(20) not null,
  cognome varchar (30) not null,
  universita varchar(30) not null,
  primary key(matricola, universita),
  foreign key (universita) references
  Universita(nome)
)
```

```
create table Universita(
  nome varchar(30) primary key,
  via varchar(40),
  cap varchar(10),
  check ((nome is null and cap is null) or
  (via is not null and cap is not null)
  )
)
```

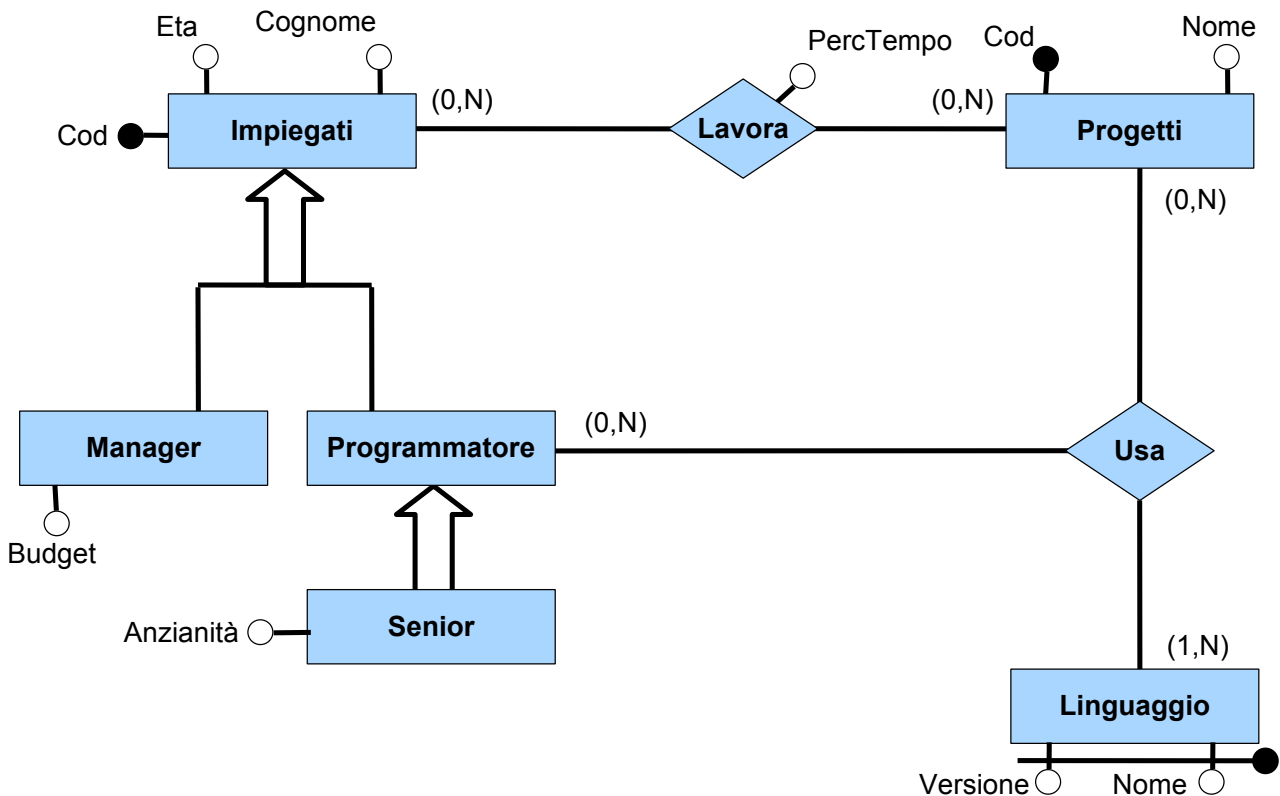
### Schema Relazionale

Studentelscrizione (matricola, cognome, universita)  
 FK: Studentelscrizione[universita] ⊆ Università[nome]

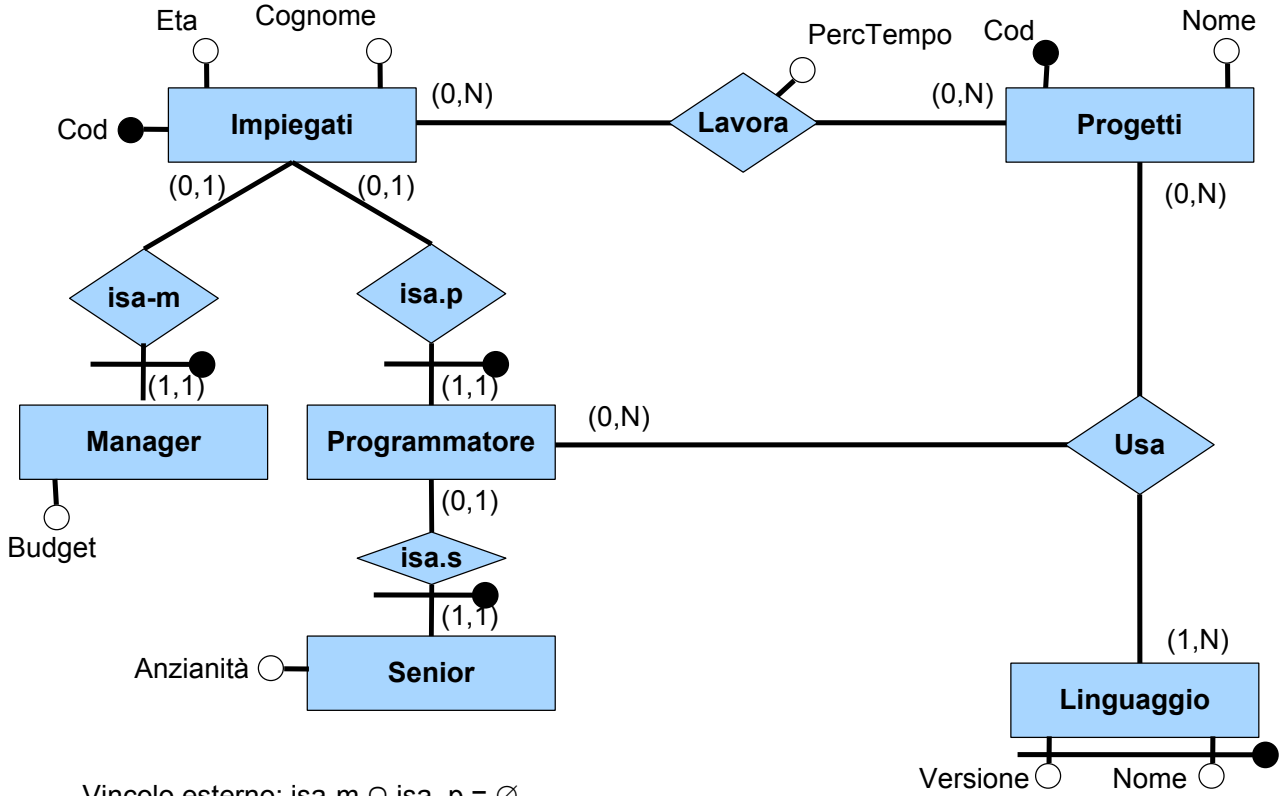
Università(nome, nabitanti)

**Vincolo Estremo: Via e Cap o sono  
 peresenti entrambi o mancano entrambi**

## Esercizio: ristrutturare e tradurre in relazionale



## Esercizio: soluzione - ER ristrutturato



### SQL

```
create table Impiegati(
  cod varchar(20) primary key,
  eta int not null,
  nome varchar(30) not null,
  cognome varchar(30) not null,
)
```

```
create table Progetti (
  cod varchar(10) primary key,
  nome varchar(30) not null
)
```

```
create table Lavora(
  impiegato varchar(20) not null,
  progetto varchar(10) not null,
  perctempo real not null,
  foreign key (impiegato) references Impiegati(cod),
  foreign key (progetto) references Progetti(cod),
  primary key (impiegato, progetto)
)
```

...

## SQL

...

```
create table Manager(  
  cod varchar(20) primary key,  
  budget real not null,  
  foreign key (cod) references Impiegato(cod)  
)
```

```
create table Programmatore(  
  cod varchar(20) primary key,  
  foreign key (cod) references Impiegato(cod)  
)
```

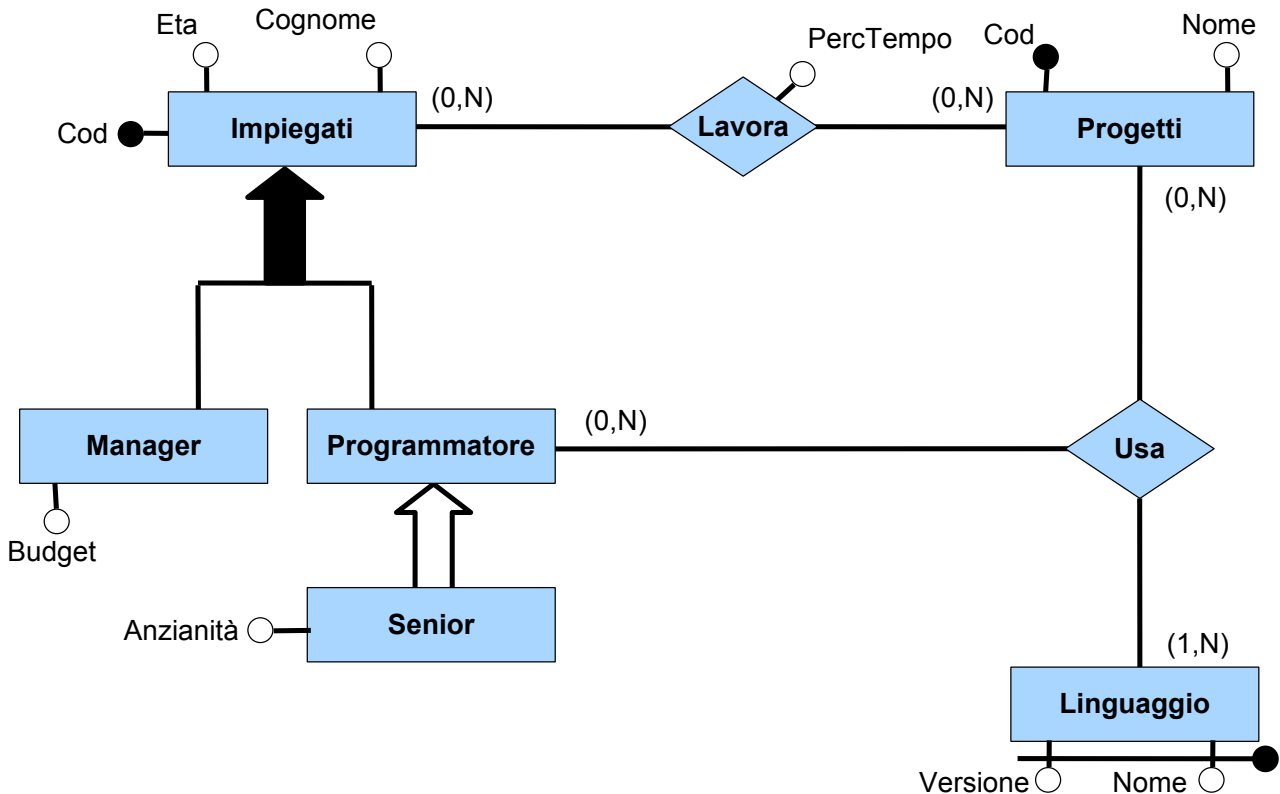
```
create table Senior(  
  cod varchar(20) primary key,  
  anzinita int not null,  
  foreign key (cod) references Programmatore(cod)  
)
```

```
create table Usa(  
  programmatore varchar(20) not null,  
  progetto varchar(20) not null,  
  nomeL varchar(30) not null,  
  versioneL varchar(30) not null,  
  primary key  
  (programmatore, progetto,nomeL,verisoneL),  
  foreign key (programmatore)  
  references Programmatore(cod),  
  foreign key (progetto) references Progetto(cod),  
  foreign key (nomeL,versioneL)
```

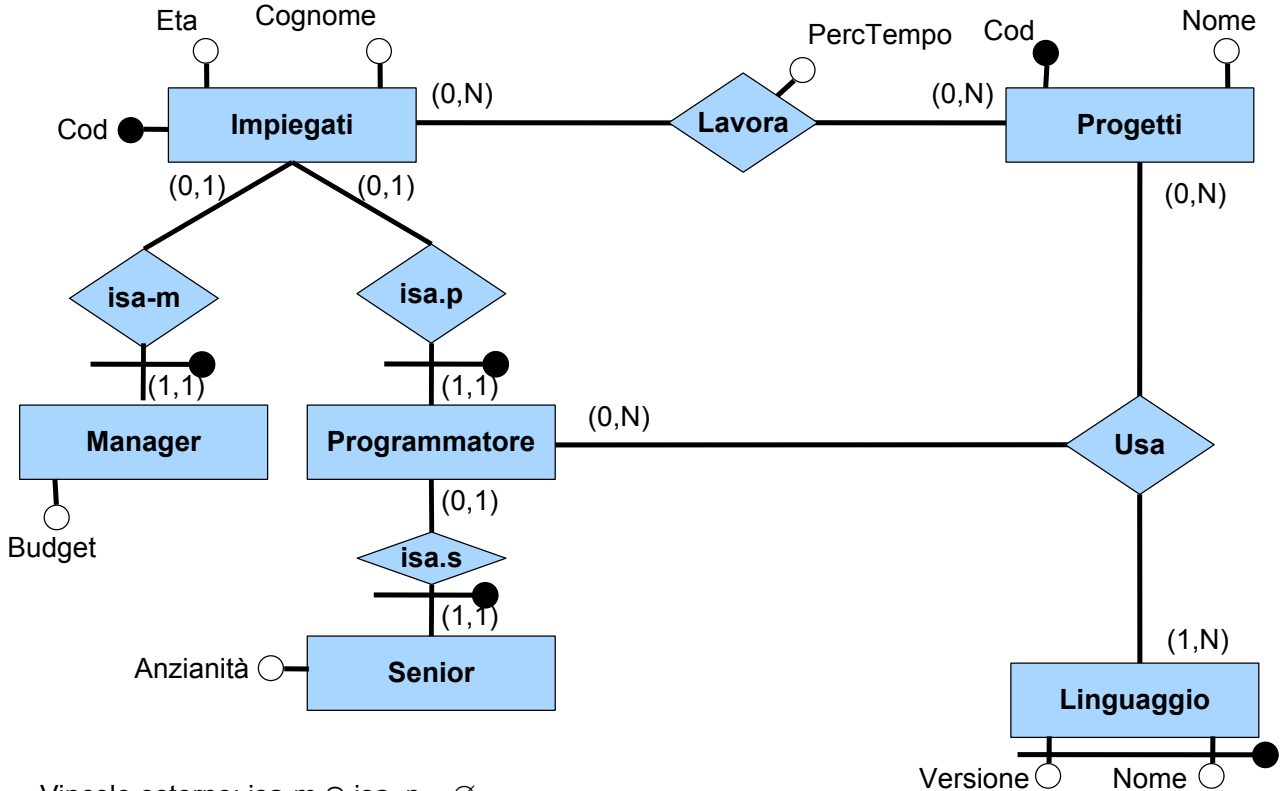
```
create table Linguaggio(  
  nome varchar(30) not null,  
  versione varchar(30) not null,  
  primary key (nome, versione),  
  check ((nome,versione) in (  
    select nomeL,versioneL  
    form Usa  
  )  
)
```

```
create assertion DisjointISA check (  
  not exists (  
    select *  
    from Manager m,Programmatore p  
    where m.cod = p.cod  
  )  
)
```

## Esercizio: ristrutturare e tradurre in relazionale



## Esercizio: soluzione - ER ristrutturato



Vincolo esterno:  $isa\_m \cap isa\_p = \emptyset$

Vincolo esterno: ogni impiegato partecipa a *isa\_m* o a *isa\_p*

### SQL

```
create table Impiegati(
  cod varchar(20) primary key,
  eta int not null,
  nome varchar(30) not null,
  cognome varchar(30) not null,
)
```

```
create table Progetti (
  cod varchar(10) primary key,
  nome varchar(30) not null
)
```

```
create table Lavora(
  impiegato varchar(20) not null,
  progetto varchar(10) not null,
  perctempo real not null,
  foreign key (impiegato) references Impiegati(cod),
  foreign key (progetto) references Progetti(cod),
  primary key (impiegato, progetto)
)
```

...

## SQL

```
...
create table Manager(
  cod varchar(20) primary key,
  budget real not null,
  foreign key (cod) references Impiegato(cod)
)

create table Programmatore(
  cod varchar(20) primary key,
  foreign key (cod) references Impiegato(cod)
)

create table Senior(
  cod varchar(20) primary key,
  anzinita int not null,
  foreign key (cod) references Programmatore(cod)
)

create table Usa(
  programmatore varchar(20) not null,
  progetto varchar(20) not null,
  nomeL varchar(30) not null,
  versioneL varchar(30) not null,
  primary key
    (programmatore, progetto,nomeL,versioneL),
  foreign key (programmatore)
    references Programmatore(cod),
  foreign key (progetto) references Progetto(cod),
  foreign key (nomeL,versioneL)
```

## SQL

```
create table Impiegati(
  cod varchar(20) primary key,
  eta int not null,
  nome varchar(30) not null,
  cognome varchar(30) not null,
  check ( cod in (select cod from Manager)
    or
    cod in (select cod from Programmatore)
  )
)
...
```

Soluzione alternativa

```
create table Linguaggio(
  nome varchar(30) not null,
  versione varchar(30) not null,
  primary key (nome, versione),
  check ((nome,versione) in (
    select nomeL,versioneL
    form Usa
  )
)

create assertion DisjointISA check (
  not exists (
    select *
    from Manager m,Programmatore p
    where m.cod = p.cod
  )
)

create assertion CompleteISA check (
  not exists (
    select cod
    from Impiegato
  except
    ( select cod
      from Manager
    union
    select cod
    from Programmatore
  )
)
```

```
...
create assertion DisjointISA check (
  not exists (
    select *
    from Manager m,Programmatore p
    where m.cod = p.cod
  )
)

create assertion CompleteISA check (
  not exists (
    select cod
    from Impiegato
  except
    ( select cod
      from Manager
    union
    select cod
    from Programmatore
  )
)
```