# Basi di dati

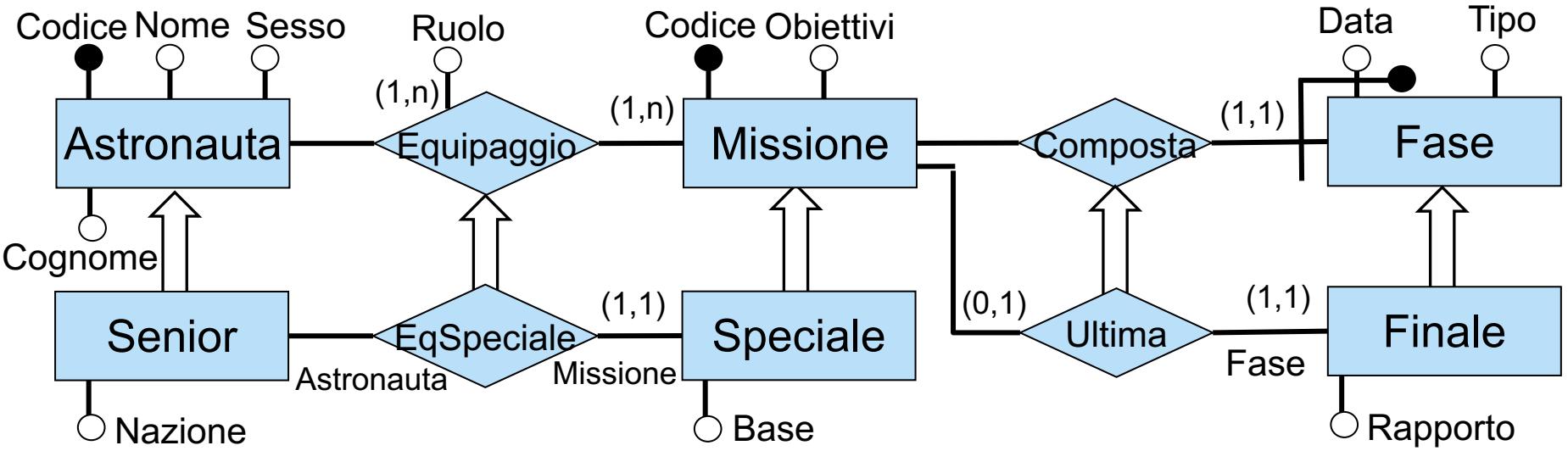
## Soluzioni dei problemi proposti nell'appello del 7-02-2022

*Maurizio Lenzerini*

Anno Accademico 2021/22

# Problema 1 – Schema concettuale

Schema concettuale:

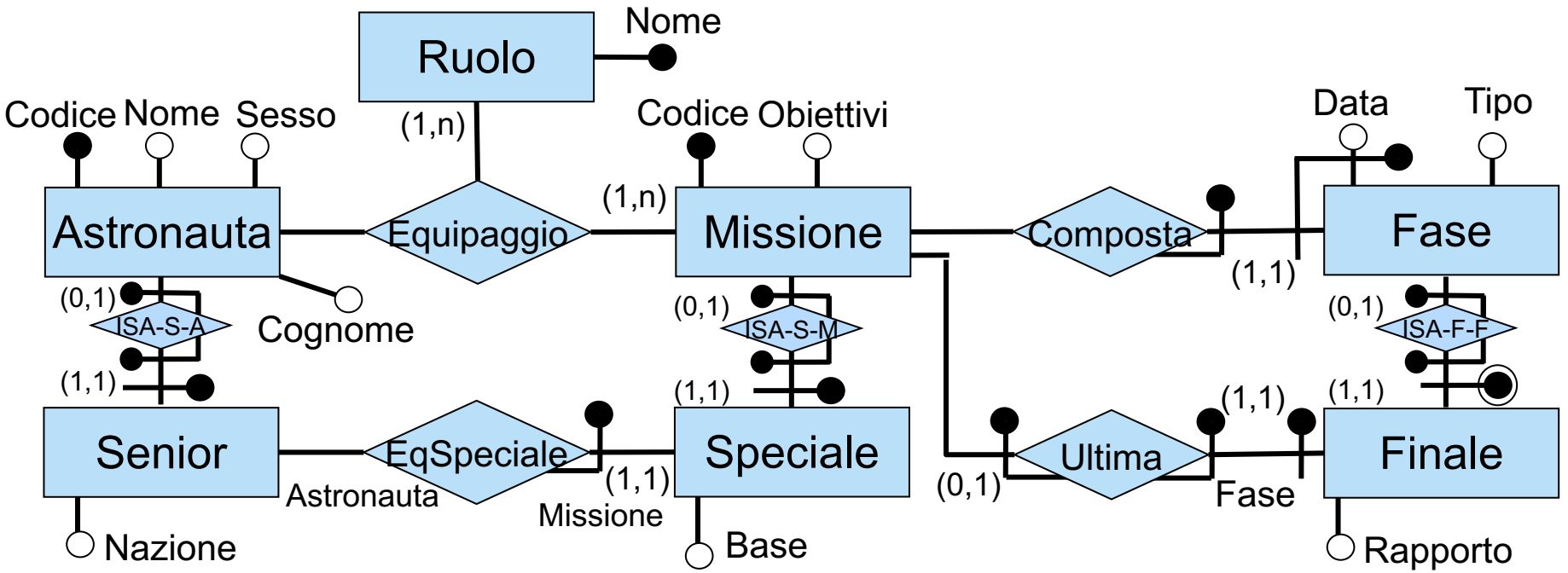



Vincolo esterno – Per ogni istanza I dello schema concettuale:

Per ogni m,f1,f2,d1,d2, se <Missione:m,Fase:f1> è in Istanze(I,Composta), <Missione:m,Fase:f2> è in Istanze(I,Ultima) e <f1,d1> e <f2,d2> sono in Istanze(I,Data), allora d2 > d1.

# Problema 2 – Schema concettuale ristrutturato

## Schema concettuale ristrutturato:




Vincoli esterni – Per ogni istanza I dello schema concettuale:

1. Per ogni s,p,a,m, se <Astronauta:s,Missione:p> è in Istanze(I,Eq-Speciale), <Senior:s,Astronauta:a> è in Istanze(I,ISA-S-A) e <Speciale:p,Missione:m> è in Istanze(I,ISA-S-M), allora esiste almeno un r in Istanze(I,Ruolo) tale che <Astronauta:a,Missione:m,Ruolo:r> è in Istanze(I,Equipaggio).
2. Per ogni m,f,s, se <Missione:m,Fase:f> è in Istanze(I,Ultima) e <Finale:f,Fase:s> è in Istanze(I,ISA-F-F), allora <Missione:m,Fase:s> è in Istanze(I,Composta).
3. Per ogni m,f1,f2,d1,d2, se <Missione:m, Fase:f1> è in Istanze(I,Composta), <Missione:m, Fase:f2> è in Istanze(I,Ultima), <Finale:f2,Fase:f3> è in Istanze(I,ISA-F-F) e <f1,d1> e <f3,d2> sono in Istanze(I,Data), allora d2 > d1.

# Problema 2 – Traduzione diretta

Schema logico prodotto dalla traduzione diretta:

Astronauta(<u>codice</u>,nome,cognome,sesso)
Senior(<u>codice</u>,nazione)
  foreign key: Senior[codice] $\subseteq$ Astronauta[codice]
Equipaggio(<u>astronauta</u>,<u>missione</u>,<u>ruolo</u>)
  foreign key: Equipaggio[astronauta] $\subseteq$ Astronauta[codice]
  foreign key: Equipaggio[missione] $\subseteq$ Missione[codice]
  foreign key: Equipaggio[ruolo] $\subseteq$ Ruolo[nome]
Missione(<u>codice</u>,obiettivi)
  inclusione: Missione[codice] $\subseteq$ Equipaggio[missione]
Ruolo(<u>nome</u>)
  inclusion: Ruolo[nome] $\subseteq$ Equipaggio[ruolo]
Speciale(<u>codice</u>,base)
  foreign key: Speciale[codice] $\subseteq$ Missione[codice]
  foreign key: Speciale[codice] $\subseteq$ EqSpeciale[codice]
EqSpeciale(astronauta,<u>missione</u>)
  foreign key: EqSpeciale[missione] $\subseteq$ Speciale[codice]
  foreign key: EqSpeciale[astronauta] $\subseteq$ Senior[codice]
  inclusione: EqSpeciale[astronauta,missione] $\subseteq$ Equipaggio[astronauta,missione]
Fase(<u>data</u>,<u>missione</u>,tipo)
  foreign key: Fase[missione] $\subseteq$ Missione[codice]
Finale(data,<u>missione</u>,rapporto)
  foreign key: Finale[data,missione] $\subseteq$ Fase[data,missione]
  vincolo esterno  data > all (select data from Fase where Fase.missione = missione and
                   (Fase.data,Fase.missione) not in (select data, missione from Finale)

# Problema 2 – Schema logico ristrutturato

Schema logico ristrutturato (primo passo):

(1) accorpamento tra le tabelle Missione e Speciale debolmente accoppiate

Astronauta(codice,nome,cognome,sesso)
Senior(codice,nazione)
  foreign key: Senior[codice] $\subseteq$ Astronautica[codice]
Equipaggio(astronauta,missione,ruolo)
  foreign key: Equipaggio[astronauta] $\subseteq$ Astronauta[codice]
  foreign key: Equipaggio[missione] $\subseteq$ Missione[codice]
  foreign key: Equipaggio[ruolo] $\subseteq$ Ruolo[nome]
Missione(codice,obiettivi,base*)
  inclusione: Missione[codice] $\subseteq$ Equipaggio[missione]
Ruolo(nome)
  inclusione: Ruolo[nome] $\subseteq$ Equipaggio[ruolo]
EqSpeciale(astronauta,missione)
  foreign key: EqSpeciale[astronauta] $\subseteq$ Senior[codice]
  inclusione: EqSpeciale[missione] $\subseteq$ (select codice from Missione where base si not null)
  inclusione: EqSpeciale[astronauta,missione] $\subseteq$ Equipaggio[astronauta,missione]
Fase(data,missione,tipo)
  foreign key: Fase[missione] $\subseteq$ Missione[codice]
Finale(data,missione,rapporto)
  foreign key: Finale[data,missione] $\subseteq$ Fase[data,missione]
  vincolo esterno  data > all (select data from Fase where Fase.missione = missione and
                   (Fase.data,Fase.missione) not in (select data, missione from Finale)

Vista Speciale = (select codice,base from Missione where base is not null)

# Problema 2 – Schema logico ristrutturato

Schema logico ristrutturato (primo e secondo passo):

(1) accorpamento tra le tabelle Missione e Speciale debolmente accoppiate

(2) ulteriore accorpamento tra le tabelle EqSpeciale e Missione debolmente accoppiate

Astronauta(codice,nome,cognome,sesso)
Senior(codice,nazione)
  foreign key: Senior[codice] $\subseteq$ Astronautica[codice]
Equipaggio(astronauta,missione,ruolo)
  foreign key: Equipaggio[astronauta] $\subseteq$ Astronauta[codice]
  foreign key: Equipaggio[missione] $\subseteq$ Missione[codice]
  foreign key: Equipaggio[ruolo] $\subseteq$ Ruolo[nome]
Missione(codice,obiettivi,base*,senior*)
  foreign key: Missione[senior,codice] $\subseteq$ Equipaggio[astronauta,missione]
  foreign key: Missione[senior] $\subseteq$ Senior[codice]
  vincolo di tupla: base è null se e solo se senior è null
Ruolo(nome)
  inclusione: Ruolo[nome] $\subseteq$ Equipaggio[ruolo]
Fase(data,missione,tipo)
  foreign key: Finale[data,missione] $\subseteq$ Fase[data,missione]
Finale(data,missione,rapporto)
  foreign key: Finale[missione] $\subseteq$ Missione[codice]
  vincolo esterno  data > all (select data from Fase where Fase.missione = missione and
                   (Fase.data,Fase.missione) not in (select data, missione from Finale)

Vista Speciale = (select codice,base from Missione where base is not null)
Vista EqSpeciale = (select astronauta,codice from Missione where base is not null)

# Problema 3 – testo e soluzione

Si consideri una base di dati che comprende le tabelle Persona(codice,età,professione), dove sono ammessi valori nulli per gli attributi età e professione, e Contratto(codice,azienda,anno), dove invece non sono ammessi valori nulli. La prima tabella memorizza il codice (chiave primaria), l'età e la professione di un insieme di persone. Ogni tupla della seconda tabella rappresenta un contratto stipulato da una persona con una azienda in un certo anno. Si noti che nulla vieta che un contratto c abbia un "gemello", ossia un contratto stipulato dalla stessa persona di c con la stessa azienda di c nello stesso anno di c. È noto che la base di dati soddisfa il vincolo di integrità referenziale da Contratto[codice] a Persona[codice]. Scrivere una query in SQL che calcoli quanti sono i contratti nella tabella Contratto che non hanno alcun gemello.

Possibile soluzione:

```
select count(*)
from (select count(*) as quantiduplicati
      from Contratto
      group by codice, azienda,anno) t
where t.quantiduplicati = 1
```

calcola quante sono le righe di t che hanno il valore dell'attributo "quantiduplicati" uguale ad 1

calcola la tabella t che ha una riga per ogni combinazione diversa di <codice,azienda,anno>, dove ogni riga contiene nell'attributo "quantiduplicati" un valore che indica quante duplicazioni ci sono di quella combinazione

# Problema 4 – testo e soluzione

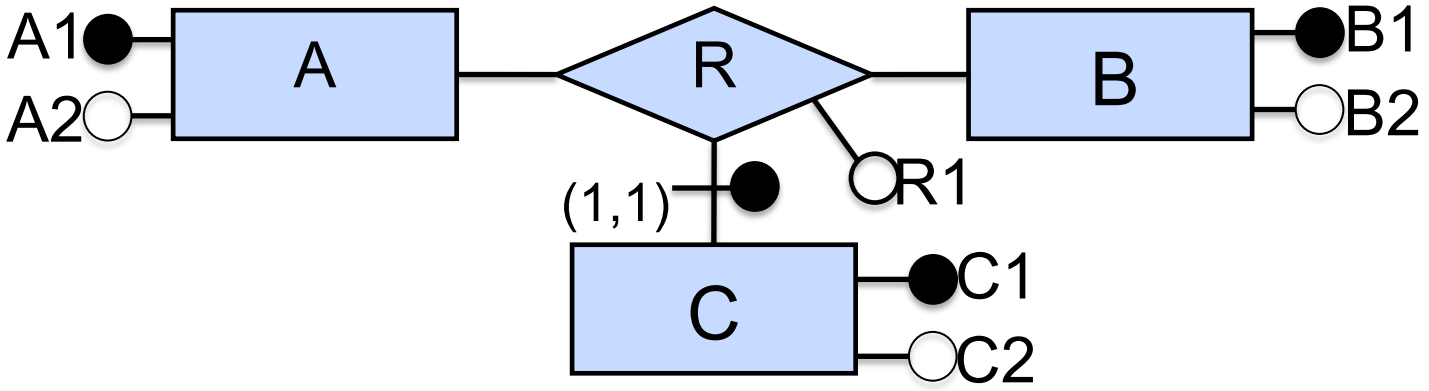
Riferendosi ancora alla base di dati menzionata nel problema 3, scrivere una query in algebra relazionale che calcoli il codice delle persone che hanno stipulato almeno un contratto dopo il 2000 e per le quali se non è nota la professione è nota l'età.

Soluzione:

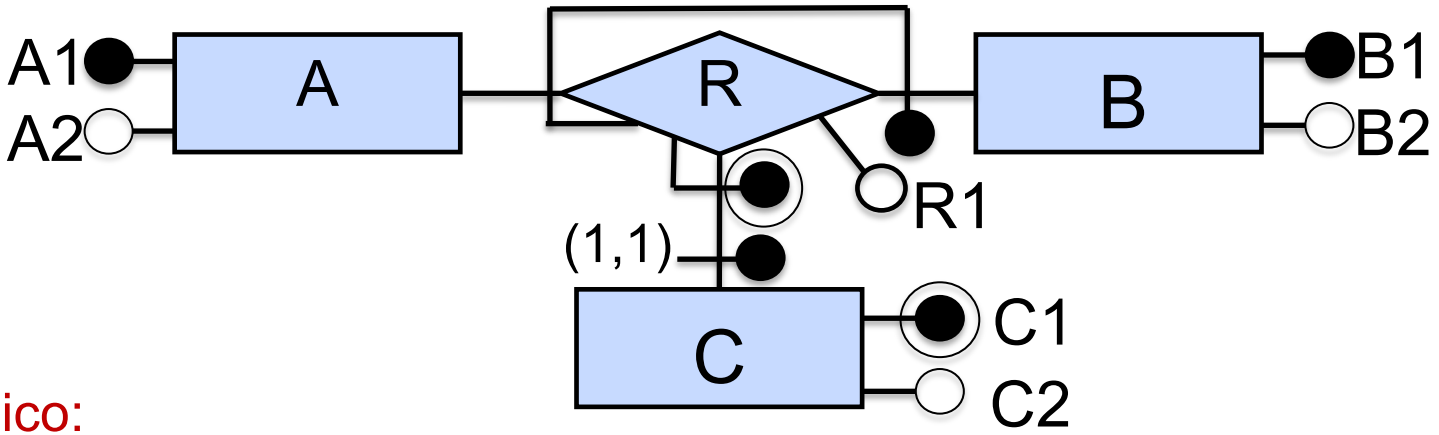
$SEL_{professione\ is\ not\ null\ OR\ età\ is\ not\ null}(Persona)\ JOIN\ SEL_{anno\ >\ 2000}(Contratto)$

# Problema 5 – soluzione

Schema concettuale di partenza:




Schema concettuale ristrutturato:




Schema logico:

A(<u>A1</u>,A2)
B(<u>B1</u>,B2)
C(<u>C1</u>,C2)
foreign key: C[C1] $\subseteq$ R[C]

R(A,B,<u>C</u>,R1)
foreign key: R[A] $\subseteq$ A[A1]
foreign key: R[B] $\subseteq$ B[B1]
foreign key: R[C] $\subseteq$ C[C1]
chiave A,B

# Problema 6 – soluzione

Si noti che:

1. Sia A sia B sono chiavi per R, ma solo A è la chiave primaria e quindi è soggetto al vincolo NOT NULL; si accettano, invece, valori nulli nell'attributo B, perché il vincolo unique non forza a valori diversi da null e sono anche ammesse diverse tuple con il valore null nell'attributo B.
2. Il vincolo di chiave esterna da R[B] a S[C] ha la politica "on delete default" ed il default per l'attributo B è null (visto che non è stato dichiarato alcun default esplicito).
3. Il vincolo di chiave esterna da S[D] a R[A] ha la politica "on delete cascade".

Proprio il punto 2) sopra descritto fa capire che se si dovesse eseguire una cancellazione di una tupla di S e questa cancellazione lasciasse una tupla t di R che non soddisfa il vincolo di foreign key da R[B] a S[C], la politica "on delete default" porrebbe t.B a null. Siccome B è unique in R (e non primary key) questo valore è sempre ammesso ed inoltre non violerà mai il vincolo di foreign key da R[B] a S[C], perché il valore nullo non viola mai un vincolo di foreign key.

Consideriamo allora una base di dati G qualunque, coerente con lo schema L ed assumiamo che si elimini da G una tupla t di R. Indichiamo con X il valore dell'attributo A di t. Per il punto 3) sopra descritto, a fronte di "on delete cascade", verranno eliminate tutte le tuple t' di S tali che il valore dell'attributo D di t' è X. Prendiamo adesso in considerazione tutte le tuple t'' di R tali che nell'attributo B c'è un valore che compare nell'attributo C di S di una delle tuple t' eliminate. Per il punto 2) sopra descritto, a fronte di "on delete set default", in tutte queste tuple t'' il valore dell'attributo B viene modificato in null ed è chiaro che questa modifica non pone alcun problema di violazione di vincoli nella base di dati, come osservato nei punti 1) e 2) sopra descritti. Osserviamo che tutti i vincoli di chiave e tutti i vincoli di foreign key sono chiaramente soddisfatti. Concludiamo che a fronte della cancellazione di una tupla t di R le politiche di violazione dei vincoli definite in L operano altre cancellazioni in modo da portare sempre la base di di dati in uno stato in cui tutti i vincoli sono soddisfatti. Quindi la risposta alla domanda è che non esiste alcuna base di dati G coerente con L tale che la cancellazione dalla base di dati G di una tupla di R provoca un errore.