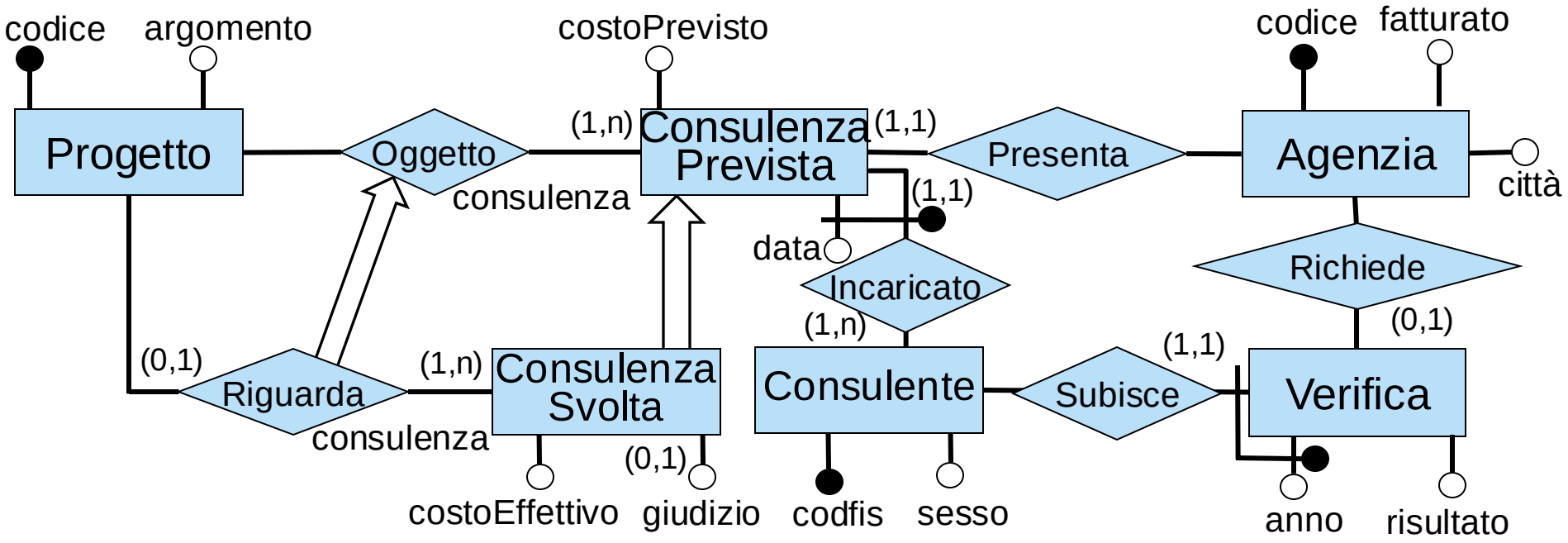
Basi di dati

Soluzioni dei problemi proposti nell'appello del 15-01-2025 Compito B

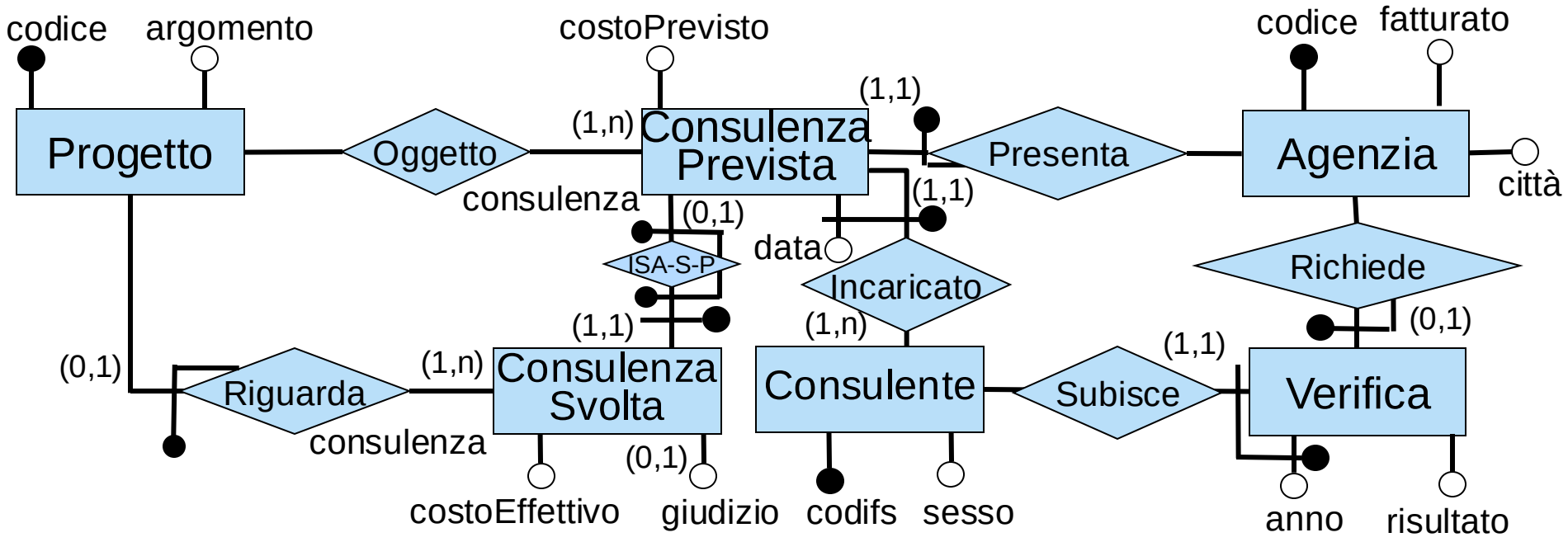
Maurizio Lenzerini

Anno Accademico 2024/25

Problema 1 – Schema ER



Problema 1 – Schema ER ristrutturato



Vincolo esterno:

In ogni istanza I dello schema concettuale, per ogni tupla <Progetto:p,ConsulenzaSvolta:c> in Istanze(I,Riguarda) tale che <ConsulenzaSvolta:c,ConsulenzaPrevista:v> è in Istanze(I,ISA-S-P), la tupla <Progetto:p, SpedizionePrevista:v> è in Istanze(I,Oggetto)

Problema 2 – Traduzione diretta

Progetto(codice,argomento)
ConsulenzaPrevista(data,consulente,costoPrevisto)
  foreign key: ConsulenzaPrevista[consulente] $\subseteq$ Consulente[codfis]
  inclusione: ConsulenzaPrevista[data,consulente] $\subseteq$ Oggetto[data,consulente]
  foreign key: ConsulenzaPrevista[data,consulente] $\subseteq$ Presenta[data,consulente]
Agenzia(codice,fatturato,città)
ConsulenzaSvolta(data,consulente,costoEffettivo,giudizio*)
  foreign key: ConsulenzaSvolta[data,consulente] $\subseteq$ ConsulenzaPrevista[data,consulente]
  inclusione: ConsulenzaSvolta[data,consulente] $\subseteq$ Riguarda[data,consulente]
Oggetto(progetto,data,consulente)
  foreign key: Oggetto[progetto] $\subseteq$ Progetto[codice]
  foreign key: Oggetto[data,consulente] $\subseteq$ ConsulenzaPrevista[data,consulente]
Riguarda(progetto,data,consulente)
  foreign key: Riguarda[progetto,data,consulente] $\subseteq$ Oggetto[progetto,data,consulente]
  foreign key: Riguarda[data,consulente] $\subseteq$ ConsulenzaSvolta[data,consulente]
Presenta(agenzia,data,consulente)
  foreign key: Presenta[agenzia] $\subseteq$ Agenzia[codice]
  foreign key: Presenta[data,consulente] $\subseteq$ ConsulenzaPrevista[data,consulente]
Agenzia(codice,fatturato,città)
Consulente(codfis,sesso)
  inclusione: Consulente[codfis] $\subseteq$ ConsulenzaPrevista[consulente]
Verifica(anno,consulente,risultato)
  foreign key: Verifica[consulente] $\subseteq$ Consulente[codice]
Richiede(anno,consulente,agenzia)
  foreign key: Richiede[anno,consulente] $\subseteq$ Verifica[anno,consulente]
  foreign key: Richiede[agenzia] $\subseteq$ Agenzia[codice]

Problema 2 – Ristrutturazione dello schema logico

L'indicazione di progetto induce un accorpamento tra la tabella ConsulenzaPrevista e la tabella ConsulenzaSvolta debolmente accoppiate (e l'aggiornamento della definizione della tabella Riguarda):

ConsulenzaPrevista(<u>data</u>,<u>consulente</u>,costoPrevisto,costoEffettivo*,giudizio*)
- foreign key: ConsulenzaPrevista[consulente] $\subseteq$ Consulente[codfis]
- inclusione: ConsulenzaPrevista[data,consulente] $\subseteq$ Oggetto[data,consulente]
- foreign key: ConsulenzaPrevista[data,consulente] $\subseteq$ Presenta[data,consulente]
- vincolo di tupla: se giudizio is not null allora costoEffettivo is not null
- vincolo esterno (select data, consulente
  from ConsulenzaPrevista
  where costoEffettivo is not null) $\subseteq$ Riguarda[data,consulente]

Riguarda(<u>progetto</u>,data,consulente)
- foreign key: Riguarda[progetto,data,consulente] $\subseteq$ Oggetto[progetto,data,consulente]
- foreign key: Riguarda[data,consulente] $\subseteq$ (select data, consulente
  from ConsulenzaPrevista
  where costoEffettivo is not null)

La tabella ConsulenzaSvolta si può ricostruire attraverso la seguente vista:

view ConsulenzaSvolta(data,consulente,costoEffettivo,giudizio) =
$\text{PROJ}_{\text{data,consulente,costoEffettivo,giudizio}}(\text{SEL}_{\text{costoEffettivo is not null}}(\text{ConsulenzaPrevista}))$

Problema 3 – testo e soluzione

(3.1) Scrivere una query in algebra relazionale che calcoli il codice fiscale di tutti i consulenti stimati, dove un consulente stimato è un consulente c tale che ogni consulenza prevista che ha c come incaricato è stata svolta.

(3.2) Scrivere una query in SQL che per ogni consulente stimato calcoli il codice fiscale ed il numero di verifiche fiscali che ha subito.

(3.1)

$REN_{consulente \leftarrow codfis}(PROJ_{codfis}(Consulente)) –$
$PROJ_{consulente}(SEL_{costoEffettivo\ is\ null}(ConsulenzaPrevista))$

(3.2)

```
with Stimato as
(select codfis from Consulente
 except
 select consulente from ConsulenzaPrevista where costoEffettivo is null)
select s.codfis, count(risultato)
from Stimato s left join Verifica v on s.codifs = v.consulente
group by s.codfis
```

Problema 4

4.1
Un vincolo di integrità è una condizione che si esprime a livello di uno schema S di basi di dati e che deve essere soddisfatta da tutte le istanze di S.
4.2
Siccome la relazione Q(A,B,C,D) non può essere soggetta ad aggiornamenti, dobbiamo considerare solo cancellazioni ed inserimenti. Notiamo che se una base di dati soddisfa il vincolo di esclusione A → ¬B, nessuna operazione di cancellazione su Q può lasciare la base di dati in uno stato in cui il vincolo stesso è violato. Rimane quindi da considerare l'operazione di inserimenti. Ovviamente, l'inserimento di una tupla t = <a,b,c,d> in Q può causare una violazione del vincolo di esclusione e questo succede sotto questa condizione Z: l'inserimento assegna ad A e B di t lo stesso valore, oppure il nuovo valore di t.A è già nell'attributo B di qualche tupla t' di Q, oppure il nuovo valore di t.B è già nell'attributo A di qualche tupla t' di Q.
La soluzione 4.a, consiste nel definire un trigger che realizza l'inserimento se e solo se la condizione Z non è soddisfatta.

Problema 4 (soluzione 4.a)

In questa soluzione definiamo un trigger di tipo before insert per bloccare l'inserimento nel caso in cui lo stato della base di dati soddisfi la condizione Z.

```
create table Q (
   a int not null, b int not null, c int not null, d int not null
);

create function inserisciQ() returns trigger as
$$
begin
   if (new.a = new.b OR
       new.a in (select b from Q) OR
       new.b in (select a from Q)
   then return null; -- inserimento bloccato
   else return new; -- inserimento da eseguire
   end if;
end;
$$ language plpgsql;

create trigger triggerForQ before insert on Q for each row
execute procedure inserisciQ();
```

Problema 4 (soluzione 4.b)

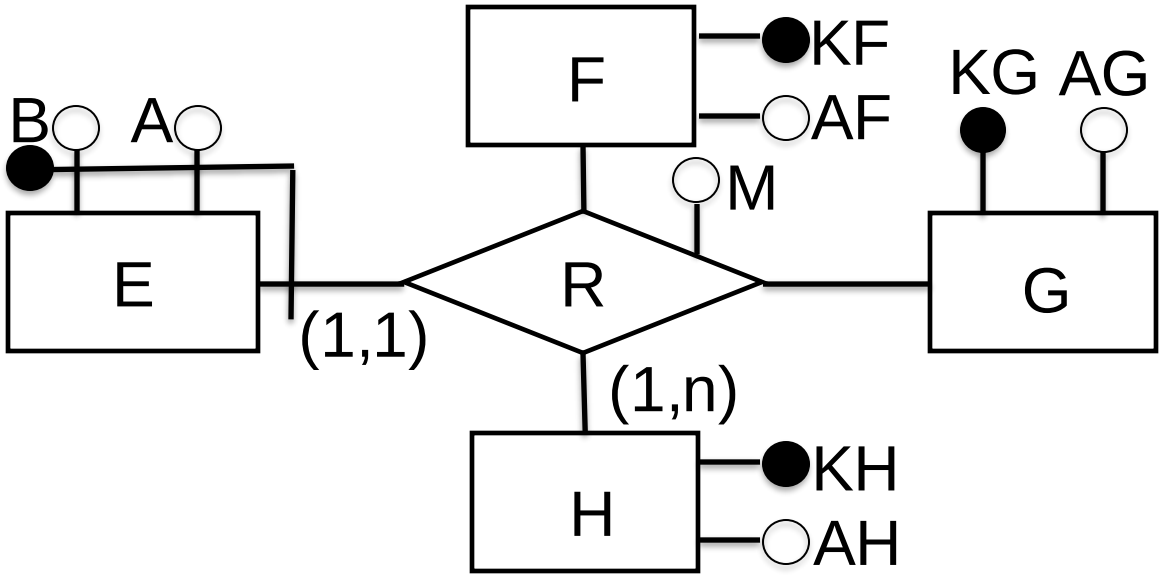
In questa soluzione, anch'essa corretta, definiamo un trigger di tipo after insert per bloccare l'inserimento nel caso in cui lo stato della base di dati non soddisfi il vincolo di esclusione.

```
create table Q (
   a int not null, b int not null, c int not null, d int not null
);

create function inserisciQ() returns trigger as
$$
begin
   if exists (select a from Q intersect select b from Q)
   then rollback;
   else return new;
   end if;
end;
$$ language plpgsql;

create trigger triggerForQ after insert on Q for each statement
execute procedure inserisciQ();
```

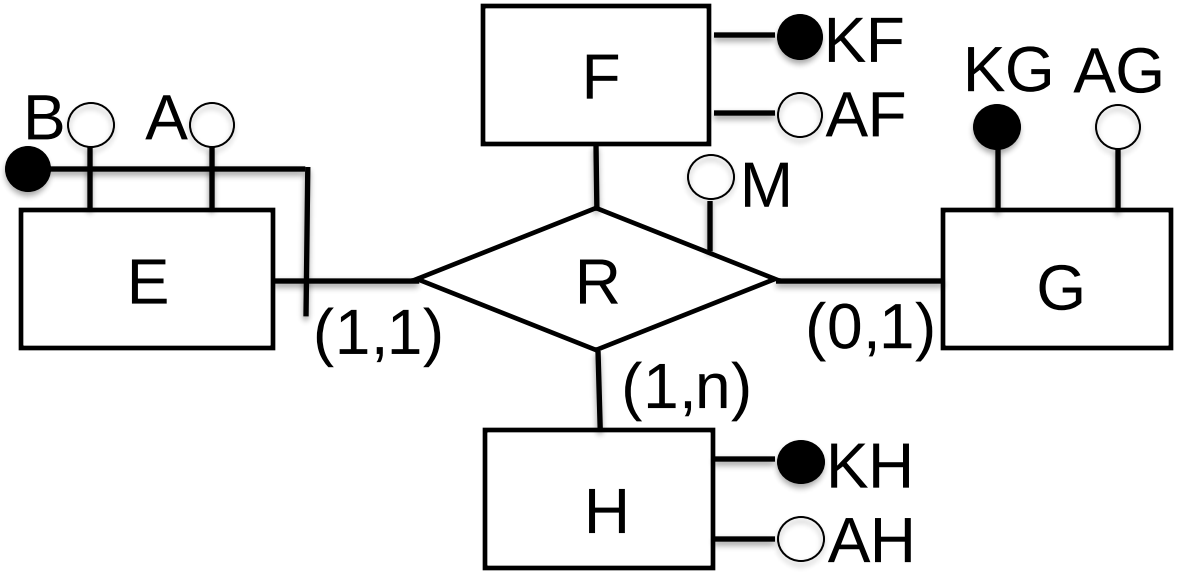
Problema 5



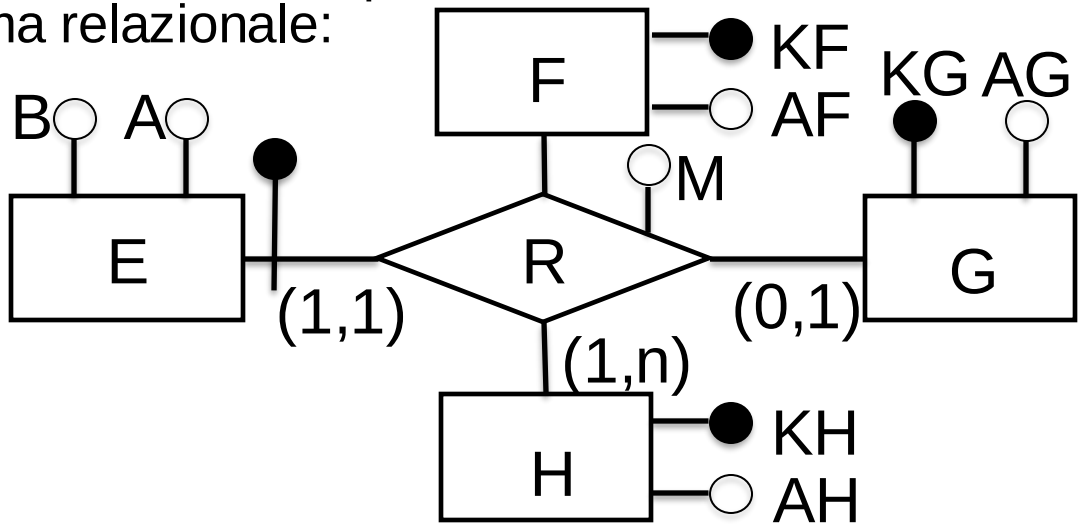
5.1
La ristrutturazione non prevede alcuna operazione. La traduzione diretta produce questo schema relazionale:

F(KF,AF)
G(KG,AG)
H(KH,AH)
 inclusione H[KH] $\subseteq$ E[H]
E(A,B,F,G,H,M)
   foreign key E[F] $\subseteq$ F[KF]
   foreign key E[G] $\subseteq$ G[KG]
   foreign key E[H] $\subseteq$ H[KH]

Problema 5



5.2
Nel caso in cui lo schema di partenza fosse quello sopra, cioè con il vincolo (0,1) associato al ruolo G di R, nello schema concettuale ristrutturato l'entità E avrebbe come identificatore solo il ruolo E di R (vedi figura qui sotto) perché se ci fossero due istanze di E con la stessa partecipazione ad R, ci sarebbero due istanze di R con la stessa istanza di G nel ruolo F, contraddicendo il vincolo 1 di cardinalità massima sul ruolo G.. In questo caso, la traduzione diretta produrrebbe questo schema relazionale:



F(KF,AF)
G(KG,AG)
H(KH,AH)
  inclusione H[KH] $\subseteq$ E[H]
E(B,A,F,G,H,M)
  foreign key E[F] $\subseteq$ F[KF]
  foreign key E[G] $\subseteq$ G[KG]
  foreign key E[H] $\subseteq$ H[KH]