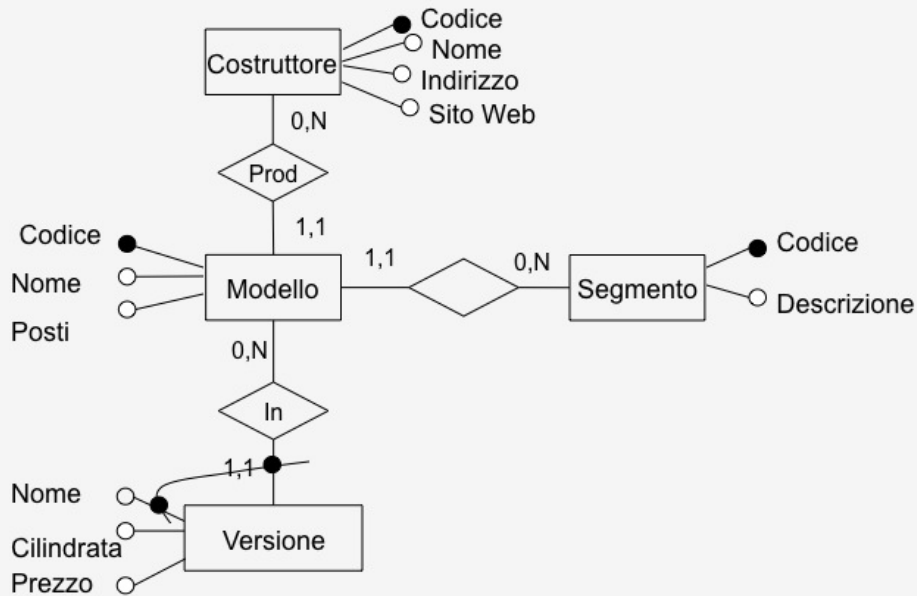


Domanda 1

Con riferimento alla Domanda 2 della prova di autovalutazione del 22 novembre 2013:

1. progettare lo schema logico relazionale corrispondente allo schema concettuale definito al punto precedente, mostrando i nomi delle relazioni, quelli degli attributi e i vincoli di chiave e di integrità referenziale;
2. mostrare un'istanza della base di dati progettata al punto precedente, utilizzando i dati nell'esempio (o anche parte di essi, purché si riescano a mostrare gli aspetti significativi).

Domanda 1



Domanda 1

Costruttori	Codice	Nome	Indirizzo	Sito Web
	1	Fiat	Corso Giovanni ...	www.fiat.it
	2	Toyota	Via Kiiciro Toyoda...	www.toyota.it
	3	Renault	Via Tiburtina ...	www.renault.it

Modelli	Codice	Nome	Posti	Segmento	Costruttore
	11	Panda	4	B	1
	12	Punto	5	C	1
	13	Aygo	4	B	2

Versioni	Modello	Nome	Cilindrata	Prezzo
	11	1.1	1098	8.900
	11	1.2 4x4	1250	13.000
	12	1.2 base	1212	11.100

Segmenti	Codice	Descrizione
	A	supercompatte
	B	compatte
	C	medie

Domanda 2

Considerare lo schema concettuale seguente:



Valutare la convenienza dell'introduzione di una relationship fra Cittadino e Regione, ridondante in quanto derivabile dalla concatenazione delle altre tre relationship, in presenza di un carico applicativo che includa come operazioni principali le seguenti:

1. modifica della residenza di un cittadino (assumere per semplicità che il trasferimento sia relativo a comuni di regioni diverse), con frequenza $f_1 = 1.000$
2. ricerca della regione di residenza di un cittadino, con frequenza $f_2 = 100.000$

Considerare solo il costo dell'accesso alle entità e assumere il costo delle scritture pari al triplo del costo delle letture.

Soluzione

con ridondanza $c_1 \times f_1 + c_2 \times f_2 = 6 \times 1.000 + 2 \times 100.000 = 206.000$

- c_1 è pari a 6 perché si deve scrivere Cittadino e si debbono leggere Comune, Provincia e Regione (un'occorrenza ciascuna)
- c_2 è pari a 2 perché è sufficiente leggere un'occorrenza di Cittadino e una di Regione

senza ridondanza $c_1 \times f_1 + c_2 \times f_2 = 4 \times 1.000 + 4 \times 100.000 = 404.000$

- c_1 è pari a 4 perché si deve scrivere Cittadino e si deve leggere Comune (un'occorrenza ciascuna)
- c_2 è pari a 4 perché si deve leggere un'occorrenza di ciascuna delle entità

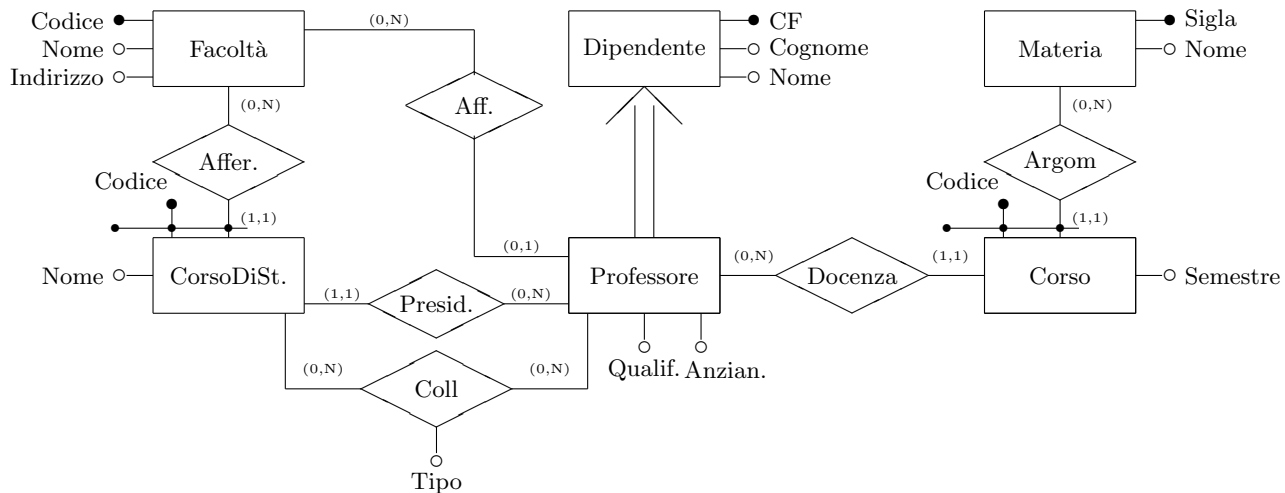
Quindi risulta conveniente introdurre la ridondanza

Domanda 3

Mostrare uno schema concettuale che rappresenti una realtà i cui dati siano organizzati per mezzo del seguente schema relazionale (dove l'asterisco indica la ammissibilità dei valori nulli).

- DIPENDENTE(CodiceFiscale, Cognome, Nome)
- PROFESSORE(CodiceFiscale, Qualifica, Anzianità, Dipartimento*) con vincolo di integrità referenziale fra CodiceFiscale e la relazione DIPENDENTE e fra Dipartimento e la relazione DIPARTIMENTO
- DIPARTIMENTO(Codice, Nome, Indirizzo)
- CORSOdiSTUDIO(Codice, Nome, Dipartimento, Presidente) con vincolo di integrità referenziale fra Dipartimento e la relazione DIPARTIMENTO e fra Presidente e la relazione PROFESSORE
- COLLABORAZIONE(CorsoDiStudio, Dipartimento, Professore, Tipo) con vincolo di integrità referenziale fra CorsoDiStudio, Dipartimento e la relazione CORSOdiSTUDIO e fra Professore e la relazione PROFESSORE
- CORSO(Codice, Materia, Docente, Semestre) con vincolo di integrità referenziale fra Materia e la relazione MATERIA e fra Docente e la relazione PROFESSORE
- MATERIA(Sigla, Nome)

Soluzione



Domanda 4

Con riferimento ad una base di dati con lo schema mostrato nella domanda precedente, scrivere una applicazione Java che stampi, per ogni dipartimento, l'elenco dei professori (con cognome, nome e qualifica) e, per ciascuno di essi, l'elenco dei corsi insegnati (con nome della materia e semestre), con una struttura come la seguente:

```
Dipartimento
  Professore
    Corso
    Corso
    ...
  Professore
  ...
Dipartimento
  Professore
  ...
```

Cenni sulla soluzione

Una possibile soluzione (metodologicamente molto schematica) è nel file Assmt.java allegato. Si notino alcuni punti fondamentali:

- l'interrogazione è una sola, con diversi join, per permettere al DBMS di scegliere la strategia ritenuta migliore (sarebbe meno efficiente avere più interrogazioni, con scansioni nidificate);
- si usano left join perchè possono esistere dipartimenti senza professori e professori senza corsi;
- serve la `ORDER BY` per raggruppare le ennuple per dipartimento e per professore;
- ogni ennupla contiene tutti i dati (dipartimento, professore, corso) e vengono stampati quelli sul dipartimento solo quando il dipartimento è diverso da quello della ennupla precedente; lo stesso per professore.