Database

Mattia Cozzi
cozzimattia@gmail.com

a.s. 2023/2024

Contenuti

Basi di dati

Nelle applicazioni informatiche sono presenti informazioni che è necessario memorizzare in modo permanente per renderle utilizzabili per elaborazioni. Inoltre, quando molti utenti lavorano su un insieme di dati, è necessario avere una sola copia dei dati, sempre aggiornata, che consenta l'accesso simultaneo a più utenti. Questo compito è svolto dai database (basi di dati).

Le basi di dati sono raccolte di dati progettati in modo tale da poter essere utilizzati in maniera ottimizzata da diverse applicazioni e diversi utenti.

Il sistema che gestisce i dati e la loro organizzazione è detto DBMS (DataBase Management System).

Basi di dati

Nelle applicazioni informatiche sono presenti informazioni che è necessario memorizzare in modo permanente per renderle utilizzabili per elaborazioni. Inoltre, quando molti utenti lavorano su un insieme di dati, è necessario avere una sola copia dei dati, sempre aggiornata, che consenta l'accesso simultaneo a più utenti. Questo compito è svolto dai database (basi di dati).

Le basi di dati sono raccolte di dati progettati in modo tale da poter essere utilizzati in maniera ottimizzata da diverse applicazioni e diversi utenti.

Il sistema che gestisce i dati e la loro organizzazione è detto DBMS (DataBase Management System).

Basi di dati

Nelle applicazioni informatiche sono presenti informazioni che è necessario memorizzare in modo permanente per renderle utilizzabili per elaborazioni. Inoltre, quando molti utenti lavorano su un insieme di dati, è necessario avere una sola copia dei dati, sempre aggiornata, che consenta l'accesso simultaneo a più utenti. Questo compito è svolto dai database (basi di dati).

Le basi di dati sono raccolte di dati progettati in modo tale da poter essere utilizzati in maniera ottimizzata da diverse applicazioni e diversi utenti.

Il sistema che gestisce i dati e la loro organizzazione è detto DBMS (DataBase Management System).

Funzioni di un DBMS

Un DBMS deve:

- gestire grandi quantità di dati, senza essere un "collo di bottiglia";
- garantire la condivisione dei dati, ad esempio coordinando gli accessi;
- garantire la persistenza dei dati e la loro integrità, controllando gli accessi;
- avere un'interfaccia grafica per l'amministrazione dei dati.

Funzioni di un DBMS

Un DBMS deve:

- gestire grandi quantità di dati, senza essere un "collo di bottiglia";
- garantire la condivisione dei dati, ad esempio coordinando gli accessi;
- garantire la persistenza dei dati e la loro integrità, controllando gli accessi;
- avere un'interfaccia grafica per l'amministrazione dei dati.

Funzioni di un DBMS

Un DBMS deve:

- gestire grandi quantità di dati, senza essere un "collo di bottiglia";
- garantire la condivisione dei dati, ad esempio coordinando gli accessi;
- garantire la persistenza dei dati e la loro integrità, controllando gli accessi;
- avere un'interfaccia grafica per l'amministrazione dei dati.

Funzioni di un DBMS

Un DBMS deve:

- gestire grandi quantità di dati, senza essere un "collo di bottiglia";
- garantire la condivisione dei dati, ad esempio coordinando gli accessi;
- garantire la persistenza dei dati e la loro integrità, controllando gli accessi;
- avere un'interfaccia grafica per l'amministrazione dei dati.

Esempi di DBMS

Commerciali:

- Oracle;
- MS SQL Server;
- MS Access (inserito nel pacchetto MS Office).

Open source:

- MySQL;
- PostgreSQL;
- Base (inserito nel pacchetto LibreOffice).

Esempi di DBMS

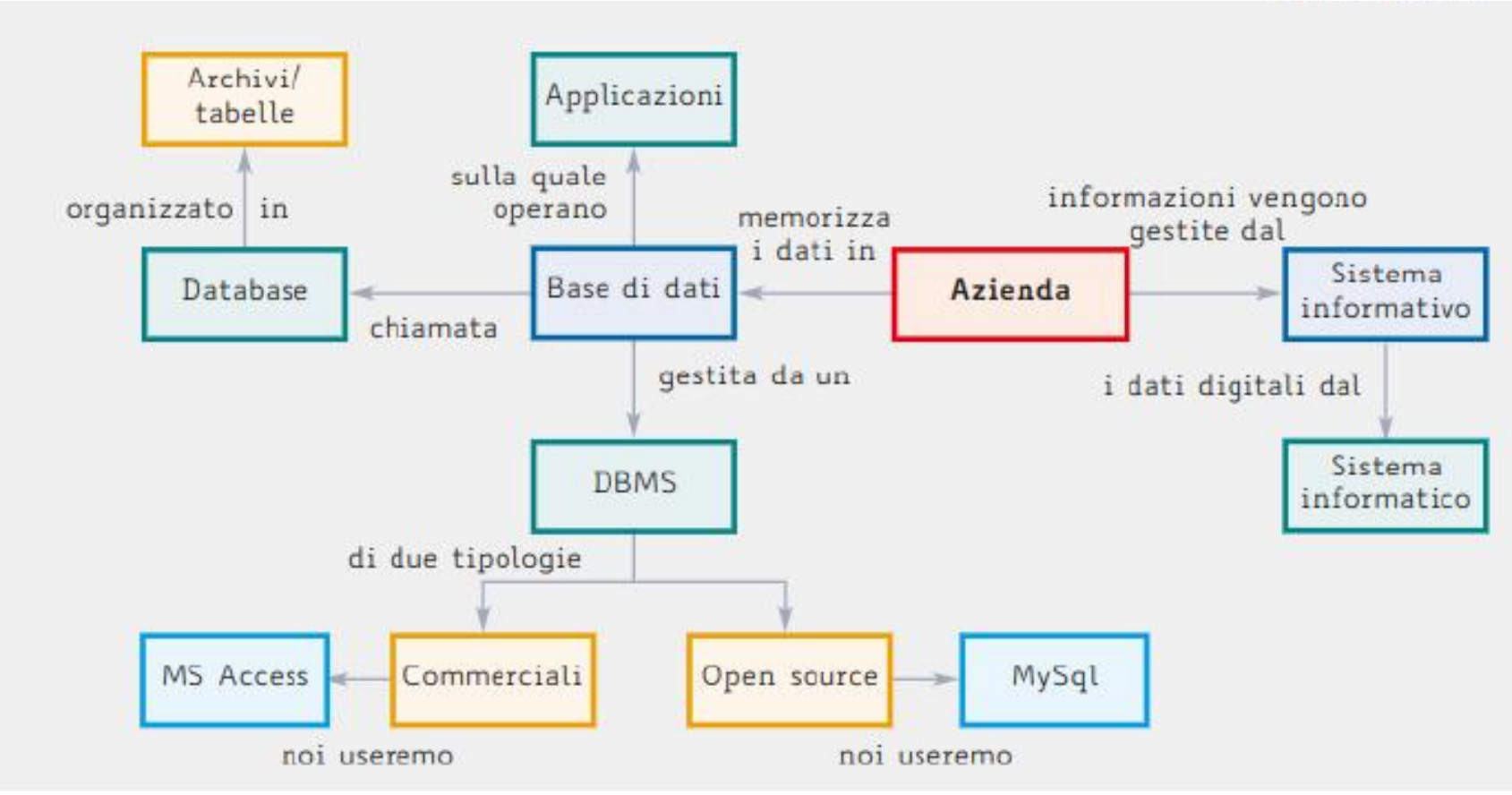
Commerciali:

- Oracle;
- MS SQL Server;
- MS Access (inserito nel pacchetto MS Office).

Open source:

- MySQL;
- PostgreSQL;
- Base (inserito nel pacchetto LibreOffice).

Schema riassuntivo



Fasi della progettazione di un database

La creazione di un database è un'operazione complessa, che viene svolta con una metodologia precisa:

1. modellazione dei dati (creazione delle tabelle):
   - 1.1 analisi;
   - 1.2 progettazione concettuale mediante il modello E-R (COSA);
   - 1.3 progettazione logica mediante lo schema logico o modello relazionale (COME);
2. modellazione funzionale:
   - 2.1 implementazione;
   - 2.2 realizzazione delle applicazioni.

Fasi della progettazione di un database

La creazione di un database è un'operazione complessa, che viene svolta con una metodologia precisa:

1. modellazione dei dati (creazione delle tabelle):
   - 1.1 analisi;
   - 1.2 progettazione concettuale mediante il modello E-R (COSA);
   - 1.3 progettazione logica mediante lo schema logico o modello relazionale (COME);
2. modellazione funzionale:
   - 2.1 implementazione;
   - 2.2 realizzazione delle applicazioni.

Fasi della progettazione di un database

La creazione di un database è un'operazione complessa, che viene svolta con una metodologia precisa:

1. modellazione dei dati (creazione delle tabelle):
   - 1.1 analisi;
   - 1.2 progettazione concettuale mediante il modello E-R (COSA);
   - 1.3 progettazione logica mediante lo schema logico o modello relazionale (COME);
2. modellazione funzionale:
   - 2.1 implementazione;
   - 2.2 realizzazione delle applicazioni.

Fasi della progettazione di un database

La creazione di un database è un'operazione complessa, che viene svolta con una metodologia precisa:

1. modellazione dei dati (creazione delle tabelle):
   - 1.1 analisi;
   - 1.2 progettazione concettuale mediante il modello E-R (COSA);
   - 1.3 progettazione logica mediante lo schema logico o modello relazionale (COME);
2. modellazione funzionale:
   - 2.1 implementazione;
   - 2.2 realizzazione delle applicazioni.

Fasi della progettazione di un database

La creazione di un database è un'operazione complessa, che viene svolta con una metodologia precisa:

1. modellazione dei dati (creazione delle tabelle):
   - 1.1 analisi;
   - 1.2 progettazione concettuale mediante il modello E-R (COSA);
   - 1.3 progettazione logica mediante lo schema logico o modello relazionale (COME);

2. modellazione funzionale:
   - 2.1 implementazione;
   - 2.2 realizzazione delle applicazioni.

Modello relazionale

Il modello maggiormente utilizzato per rappresentare i dati è il modello relazionale, che si realizza mediante tabelle.

Le colonne delle tabelle rappresentano campi o proprietà, le righe rappresentano i diversi record.

Nr. fatt.	Data	Nome	Cognome	Nr. cliente	Via	Nr. civico	CAP	Città	Nr. pos.	Articolo	Nr. Articolo	Quantità	Prezzo in €
123	29.01.2018	Gero nimo	Scatandro	11	Piazza Roma	1	00188	Roma	1	Monitor	2-0023-D	10	200
123	29.01.2018	Gero nimo	Scatandro	11	Piazza Roma	1	00188	Roma	2	Tappetino per il mouse	4-0023-D	12	0,50
123	29.01.2018	Gero nimo	Scatandro	11	Piazza Roma	1	00188	Roma	3	Sedia da ufficio	5-0023-D	1	120
124	30.01.2018	Gioia	Rinace	12	Via Roma	2	00188	Roma	1	Computer portatile	1-0023-D	2	1200
124	30.01.2018	Gioia	Rinace	12	Via Roma	2	00188	Roma	2	Cuffie	3-0023-D	2	75

Modello relazionale

Il modello maggiormente utilizzato per rappresentare i dati è il modello relazionale, che si realizza mediante tabelle.

Le colonne delle tabelle rappresentano campi o proprietà, le righe rappresentano i diversi record.

Nr. fatt.	Data	Nome	Cognome	Nr. cliente	Via	Nr. civico	CAP	Città	Nr. pos.	Articolo	Nr. Articolo	Quantità	Prezzo in €
123	29.01.2018	Geronimo	Scatandro	11	Piazza Roma	1	00188	Roma	1	Monitor	2-0023-D	10	200
123	29.01.2018	Geronimo	Scatandro	11	Piazza Roma	1	00188	Roma	2	Tappetino per il mouse	4-0023-D	12	0,50
123	29.01.2018	Geronimo	Scatandro	11	Piazza Roma	1	00188	Roma	3	Sedia da ufficio	5-0023-D	1	120
124	30.01.2018	Gioia	Rinace	12	Via Roma	2	00188	Roma	1	Computer portatile	1-0023-D	2	1200
124	30.01.2018	Gioia	Rinace	12	Via Roma	2	00188	Roma	2	Cuffie	3-0023-D	2	75

Modello Entità-Relazione

Questo modello permette di modella graficamente il mondo reale sotto forma di entità e relazioni tra esse.

Le entità sono gli oggetti principali su cui vengono raccolte le informazioni. Ogni entità rappresenta graficamente un oggetto, concreto o astratto, del mondo reale.

Ogni entità avrà un nome, che permette di identificare ogni istanza (ogni "esemplare") di quella classe.

Le relazioni tra entità verranno rappresentate mediante linee che collegano le entità.

Modello Entità-Relazione

Questo modello permette di modella graficamente il mondo reale sotto forma di entità e relazioni tra esse.

Le entità sono gli oggetti principali su cui vengono raccolte le informazioni. Ogni entità rappresenta graficamente un oggetto, concreto o astratto, del mondo reale.

Ogni entità avrà un nome, che permette di identificare ogni istanza (ogni "esemplare") di quella classe.

Le relazioni tra entità verranno rappresentate mediante linee che collegano le entità.

Modello Entità-Relazione

Questo modello permette di modella graficamente il mondo reale sotto forma di entità e relazioni tra esse.

Le entità sono gli oggetti principali su cui vengono raccolte le informazioni. Ogni entità rappresenta graficamente un oggetto, concreto o astratto, del mondo reale.

Ogni entità avrà un nome, che permette di identificare ogni istanza (ogni "esemplare") di quella classe.

Le relazioni tra entità verranno rappresentate mediante linee che collegano le entità.

Modello Entità-Relazione

Questo modello permette di modella graficamente il mondo reale sotto forma di entità e relazioni tra esse.

Le entità sono gli oggetti principali su cui vengono raccolte le informazioni. Ogni entità rappresenta graficamente un oggetto, concreto o astratto, del mondo reale.

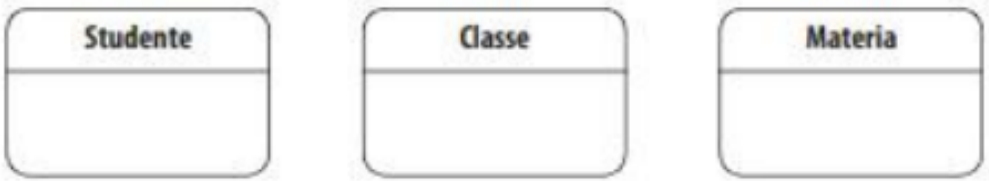
Ogni entità avrà un nome, che permette di identificare ogni istanza (ogni "esemplare") di quella classe.

Le relazioni tra entità verranno rappresentate mediante linee che collegano le entità.

Rappresentazione delle entità

Le entità possono essere rappresentate in notazione classica:

oppure in notazione UML (Unified Modelling Language):



Tipi di entità

Possiamo distinguere tra entità forti (che non richiedono altre entità per essere identificata) e entità deboli.

Ad esempio, in un database ospedaliero, "Paziente" è entità forte, "Esame" è entità debole.

Esistono inoltre entità associative, usate per associare due o più entità tra loro, cioè per risolvere delle associazioni multiple.

Ad esempio, in un database di classe, ogni "Docente" insegna in diverse "Classi", e serve pertanto l'entità "Orario" per associare ogni docente alle sue classi.

Tipi di entità

Possiamo distinguere tra entità forti (che non richiedono altre entità per essere identificata) e entità deboli.

Ad esempio, in un database ospedaliero, “Paziente” è entità forte, “Esame” è entità debole.

Esistono inoltre entità associative, usate per associare due o più entità tra loro, cioè per risolvere delle associazioni multiple.

Ad esempio, in un database di classe, ogni “Docente” insegna in diverse “Classi”, e serve pertanto l’entità “Orario” per associare ogni docente alle sue classi.

Tipi di entità

Possiamo distinguere tra entità forti (che non richiedono altre entità per essere identificata) e entità deboli.

Ad esempio, in un database ospedaliero, “Paziente” è entità forte, “Esame” è entità debole.

Esistono inoltre entità associative, usate per associare due o più entità tra loro, cioè per risolvere delle associazioni multiple.

Ad esempio, in un database di classe, ogni “Docente” insegna in diverse “Classi”, e serve pertanto l’entità “Orario” per associare ogni docente alle sue classi.

Tipi di entità

Possiamo distinguere tra entità forti (che non richiedono altre entità per essere identificata) e entità deboli.

Ad esempio, in un database ospedaliero, “Paziente” è entità forte, “Esame” è entità debole.

Esistono inoltre entità associative, usate per associare due o più entità tra loro, cioè per risolvere delle associazioni multiple.

Ad esempio, in un database di classe, ogni “Docente” insegna in diverse “Classi”, e serve pertanto l’entità “Orario” per associare ogni docente alle sue classi.

Istanze

Ogni istanza di una certa entità è caratterizzato da un insieme di valori che descrivono le sue proprietà.

Tutte le istanze di una certa entità hanno gli stessi attributi, ma con valori diversi per poterle distinguere.

Ad esempio, er l'entità "Studente", gli attributi possono essere "Nome", "Cognome", "Codice fiscale", "Data di nascita", "Sezione", ecc.

Istanze

Ogni istanza di una certa entità è caratterizzato da un insieme di valori che descrivono le sue proprietà.

Tutte le istanze di una certa entità hanno gli stessi attributi, ma con valori diversi per poterle distinguere.

Ad esempio, er l'entità "Studente", gli attributi possono essere "Nome", "Cognome", "Codice fiscale", "Data di nascita", "Sezione", ecc.

Istanze

Ogni istanza di una certa entità è caratterizzato da un insieme di valori che descrivono le sue proprietà.

Tutte le istanze di una certa entità hanno gli stessi attributi, ma con valori diversi per poterle distinguere.

Ad esempio, er l'entità "Studente", gli attributi possono essere "Nome", "Cognome", "Codice fiscale", "Data di nascita", "Sezione", ecc.

Attributi

Attributo fondamentale di ogni istanza è il suo identificatore univoco, detto chiave.

I tipi di attributi sono mostrati in figura.

TIPO	DESCRIZIONE	ESEMPI
Semplice (atomico)	Non è ulteriormente scomponibile, elementare.	Progressivo, Cognome, Età, Peso
Composto	È costituito da un insieme di componenti.	Telefono, Indirizzo, Data
Opzionale (parziale)	È possibile la sua assenza, cioè potrebbe non esistere in qualche istanza.	Telefono, Coniugato, Interesse
Obbligatorio (totale)	È l'opposto di opzionale e deve sempre essere presente un suo valore in ogni istanza.	Codice_Fiscale, Cognome, Data_Nascita
Costante (statico)	I valori non possono essere cambiati per tutto il "ciclo di vita" dell'attributo.	Codice_Fiscale, Cognome, Nome
Modificabile (dinamico)	È l'opposto di costante, cioè i suoi valori possono venire modificati.	Età, Peso, Indirizzo
Calcolato	Il valore è calcolato con un algoritmo.	Stipendio, Importo_Fattura, Età
Esplicito	È l'opposto di calcolato.	Data_Nascita, Prezzo, Peso
Unico (univoco)	Tutte le istanze della classe hanno valore diverso.	Codice_Fiscale, Partita_IVA
Generico (multivalore)	È l'opposto di unico.	Materie_Insegnate, Lingue_Parlate, Colore
Temporale	Alcuni attributi hanno una validità temporale, cioè dopo un certo tempo non hanno più significatività se non per l'archivio storico.	Stipendio, Giacenza, Prezzo, Mansione, Scadenza

Attributi

Attributo fondamentale di ogni istanza è il suo identificatore univoco, detto chiave.

I tipi di attributi sono mostrati in figura.

TIPO	DESCRIZIONE	ESEMPI
Semplice (atomico)	Non è ulteriormente scomponibile, elementare.	Progressivo, Cognome, Età, Peso
Composto	È costituito da un insieme di componenti.	Telefono, Indirizzo, Data
Opzionale (parziale)	È possibile la sua assenza, cioè potrebbe non esistere in qualche istanza.	Telefono, Coniugato, Interesse
Obbligatorio (totale)	È l'opposto di opzionale e deve sempre essere presente un suo valore in ogni istanza.	Codice_Fiscale, Cognome, Data_Nascita
Costante (statico)	I valori non possono essere cambiati per tutto il "ciclo di vita" dell'attributo.	Codice_Fiscale, Cognome, Nome
Modificabile (dinamico)	È l'opposto di costante, cioè i suoi valori possono venire modificati.	Età, Peso, Indirizzo
Calcolato	Il valore è calcolato con un algoritmo.	Stipendio, Importo_Fattura, Età
Esplicito	È l'opposto di calcolato.	Data_Nascita, Prezzo, Peso
Unico (univoco)	Tutte le istanze della classe hanno valore diverso.	Codice_Fiscale, Partita_IVA
Generico (multivalore)	È l'opposto di unico.	Materie_Insegnate, Lingue_Parlate, Colore
Temporale	Alcuni attributi hanno una validità temporale, cioè dopo un certo tempo non hanno più significatività se non per l'archivio storico.	Stipendio, Giacenza, Prezzo, Mansione, Scadenza

Dominio degli attributi

In fase di progettazione è importante stabilire quali specifiche devono rispettare gli attributi:

- tipo di dato: intero, decimale, carattere (stringa di caratteri), più i tipi speciali per data, ora e valori logici (booleani);
- lunghezza e obbligatorietà;
- se il valore NULL è supportato e se esiste un valore di default;
- intervallo: indica i limiti inferiori e superiori dei valori.

Sugli attributi possono anche essere impostati specifici vincoli, particolari restrizioni o controlli sui valori ammessi.

Il controllo della correttezza degli attributi è detto validazione.

Dominio degli attributi

In fase di progettazione è importante stabilire quali specifiche devono rispettare gli attributi:

- tipo di dato: intero, decimale, carattere (stringa di caratteri), più i tipi speciali per data, ora e valori logici (booleani);
- lunghezza e obbligatorietà;
- se il valore NULL è supportato e se esiste un valore di default;
- intervallo: indica i limiti inferiori e superiori dei valori.

Sugli attributi possono anche essere impostati specifici vincoli, particolari restrizioni o controlli sui valori ammessi.

Il controllo della correttezza degli attributi è detto validazione.

Dominio degli attributi

In fase di progettazione è importante stabilire quali specifiche devono rispettare gli attributi:

- tipo di dato: intero, decimale, carattere (stringa di caratteri), più i tipi speciali per data, ora e valori logici (booleani);
- lunghezza e obbligatorietà;
- se il valore NULL è supportato e se esiste un valore di default;
- intervallo: indica i limiti inferiori e superiori dei valori.

Sugli attributi possono anche essere impostati specifici vincoli, particolari restrizioni o controlli sui valori ammessi.

Il controllo della correttezza degli attributi è detto validazione.

Dominio degli attributi

In fase di progettazione è importante stabilire quali specifiche devono rispettare gli attributi:

- tipo di dato: intero, decimale, carattere (stringa di caratteri), più i tipi speciali per data, ora e valori logici (booleani);
- lunghezza e obbligatorietà;
- se il valore NULL è supportato e se esiste un valore di default;
- intervallo: indica i limiti inferiori e superiori dei valori.

Sugli attributi possono anche essere impostati specifici vincoli, particolari restrizioni o controlli sui valori ammessi.

Il controllo della correttezza degli attributi è detto validazione.

Dominio degli attributi

In fase di progettazione è importante stabilire quali specifiche devono rispettare gli attributi:

- tipo di dato: intero, decimale, carattere (stringa di caratteri), più i tipi speciali per data, ora e valori logici (booleani);
- lunghezza e obbligatorietà;
- se il valore NULL è supportato e se esiste un valore di default;
- intervallo: indica i limiti inferiori e superiori dei valori.

Sugli attributi possono anche essere impostati specifici vincoli, particolari restrizioni o controlli sui valori ammessi.

Il controllo della correttezza degli attributi è detto validazione.

Dominio degli attributi

In fase di progettazione è importante stabilire quali specifiche devono rispettare gli attributi:

- tipo di dato: intero, decimale, carattere (stringa di caratteri), più i tipi speciali per data, ora e valori logici (booleani);
- lunghezza e obbligatorietà;
- se il valore NULL è supportato e se esiste un valore di default;
- intervallo: indica i limiti inferiori e superiori dei valori.

Sugli attributi possono anche essere impostati specifici vincoli, particolari restrizioni o controlli sui valori ammessi.

Il controllo della correttezza degli attributi è detto validazione.

Esempio di definizione degli attributi

Per l'entità "Persona":

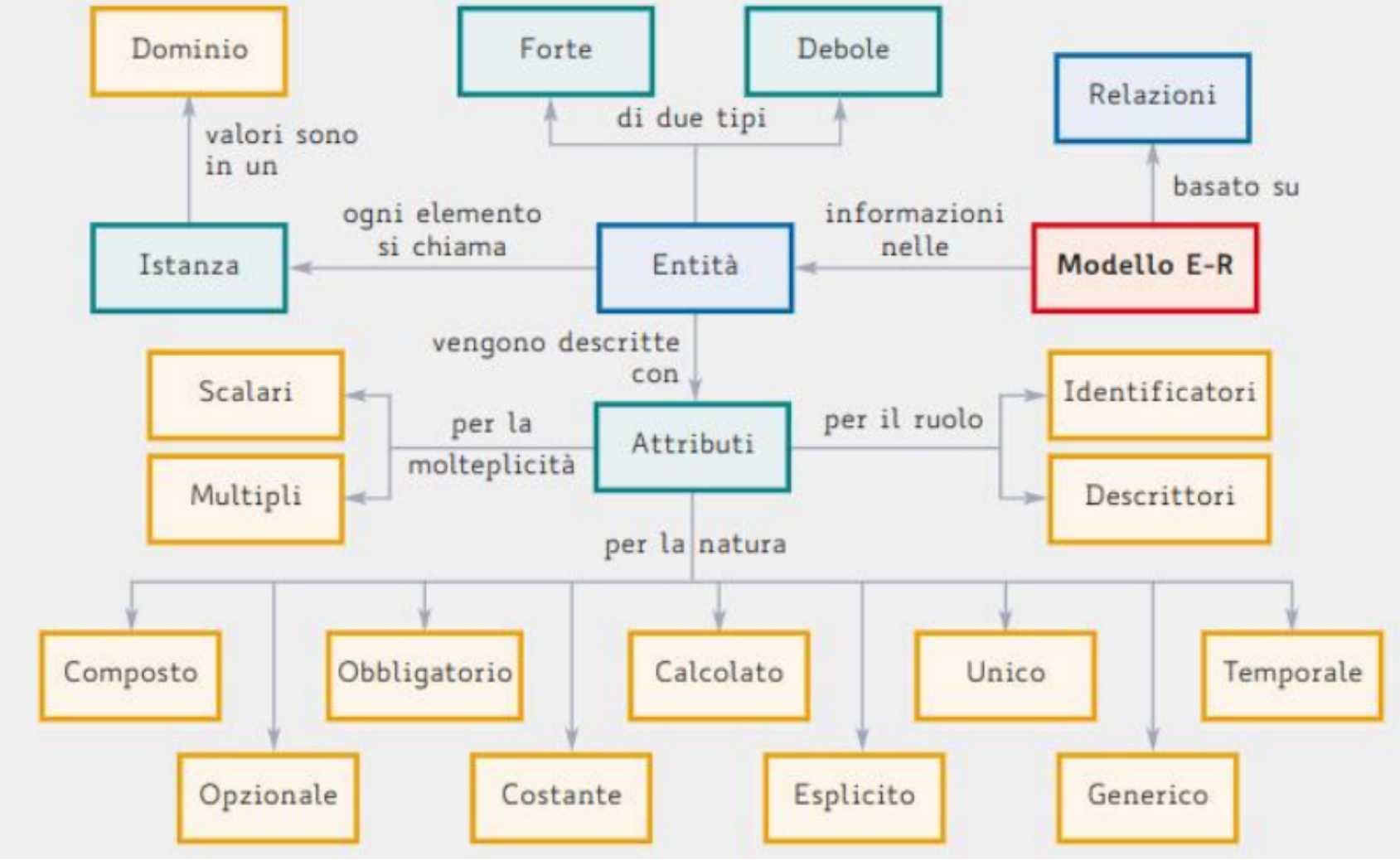
- Nome: stringa(20), obbligatorio, non NULL
- Cognome: stringa(20), obbligatorio, non NULL
- Cod_Fiscale: stringa(16)
- Titolo_di_Studio: stringa(50)
- Data_di_Nascita: giorno - mese - anno
- Peso: numerico
- Anzianità_Servizio: numerico

Rappresentazione degli attributi

Gli attributi possono essere rappresentati in notazione classica:

oppure in notazione UML:

Schema riassuntivo

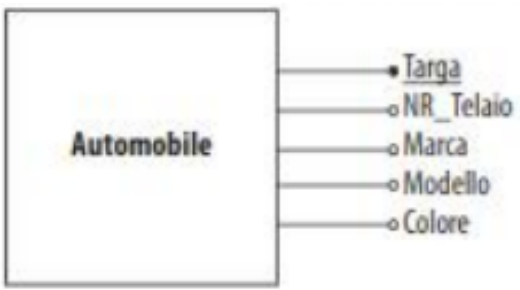


Attributi chiave

Gli attributi chiave sono degli identificatori univoci di un'istanza di un'entità. Ogni attributo chiave (*primary key*):

- deve essere obbligatorio, unico, esplicito (vedi tabella precedente);
- può essere semplice o composto;
- non deve essere modificabile;
- non può avere valore NULL.

Rappresentazione degli attributi chiave:

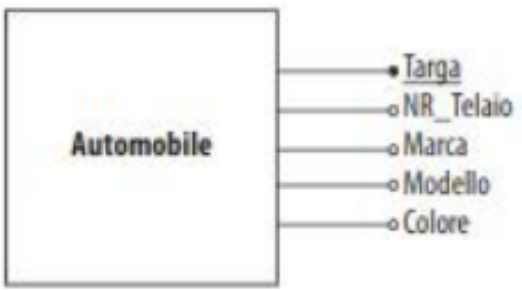


Attributi chiave

Gli attributi chiave sono degli identificatori univoci di un'istanza di un'entità. Ogni attributo chiave (*primary key*):

- deve essere obbligatorio, unico, esplicito (vedi tabella precedente);
- può essere semplice o composto;
- non deve essere modificabile;
- non può avere valore NULL.

Rappresentazione degli attributi chiave:

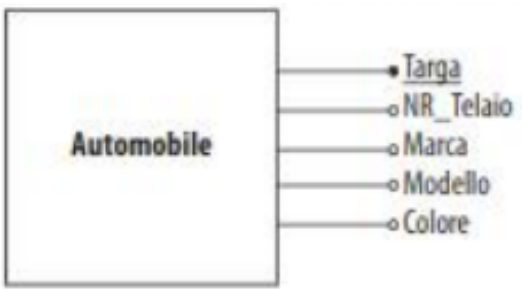


Attributi chiave

Gli attributi chiave sono degli identificatori univoci di un'istanza di un'entità. Ogni attributo chiave (*primary key*):

- deve essere obbligatorio, unico, esplicito (vedi tabella precedente);
- può essere semplice o composto;
- non deve essere modificabile;
- non può avere valore NULL.

Rappresentazione degli attributi chiave:

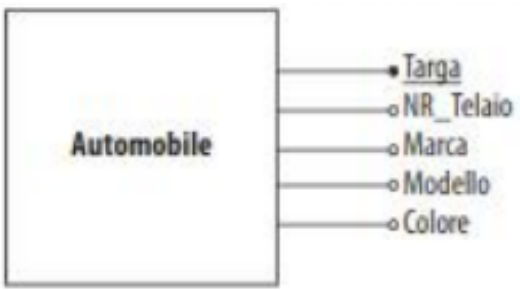


Attributi chiave

Gli attributi chiave sono degli identificatori univoci di un'istanza di un'entità. Ogni attributo chiave (*primary key*):

- deve essere obbligatorio, unico, esplicito (vedi tabella precedente);
- può essere semplice o composto;
- non deve essere modificabile;
- non può avere valore NULL.

Rappresentazione degli attributi chiave:

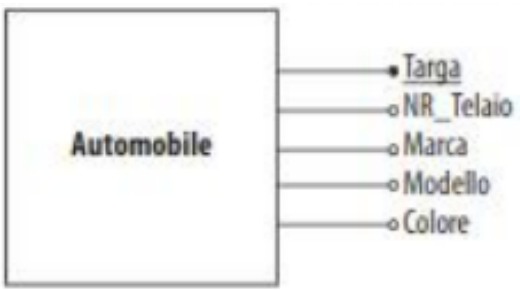


Attributi chiave

Gli attributi chiave sono degli identificatori univoci di un'istanza di un'entità. Ogni attributo chiave (*primary key*):

- deve essere obbligatorio, unico, esplicito (vedi tabella precedente);
- può essere semplice o composto;
- non deve essere modificabile;
- non può avere valore NULL.

Rappresentazione degli attributi chiave:



Chiavi artificiali

Una chiave artificiale è un attributo che assegna un codice univoco ad ogni istanza.

Esempio: la numerazione sequenziale delle tessere rilasciate da una associazione.

La maggior parte dei DBMS ammette il tipo contatore: gli attributi definiti come contatore si autoincrementano di 1 per ogni record che viene aggiunto.

Alle chiavi artificiali viene solitamente dato un nome che inizia con ID_ (ID_Studente, ID_Auto).

Chiavi artificiali

Una chiave artificiale è un attributo che assegna un codice univoco ad ogni istanza.

Esempio: la numerazione sequenziale delle tessere rilasciate da una associazione.

La maggior parte dei DBMS ammette il tipo contatore: gli attributi definiti come contatore si autoincrementano di 1 per ogni record che viene aggiunto.

Alle chiavi artificiali viene solitamente dato un nome che inizia con ID_ (ID_Studente, ID_Auto).

Chiavi artificiali

Una chiave artificiale è un attributo che assegna un codice univoco ad ogni istanza.

Esempio: la numerazione sequenziale delle tessere rilasciate da una associazione.

La maggior parte dei DBMS ammette il tipo contatore: gli attributi definiti come contatore si autoincrementano di 1 per ogni record che viene aggiunto.

Alle chiavi artificiali viene solitamente dato un nome che inizia con ID_ (ID_Studente, ID_Auto).

Chiavi artificiali

Una chiave artificiale è un attributo che assegna un codice univoco ad ogni istanza.

Esempio: la numerazione sequenziale delle tessere rilasciate da una associazione.

La maggior parte dei DBMS ammette il tipo contatore: gli attributi definiti come contatore si autoincrementano di 1 per ogni record che viene aggiunto.

Alle chiavi artificiali viene solitamente dato un nome che inizia con ID_ (ID_Studente, ID_Auto).

Chiavi esterne

Le chiavi esterne (*foreign keys*) sono utilizzate per realizzare i collegamenti tra entità, risolvendo quindi i collegamenti uno-a-molti.

Se le chiavi esterne sono chiavi artificiali, vengono nominate in modo simile a quanto si fa per le chiavi primarie, con il prefisso id_ (minuscolo).

Chiavi esterne

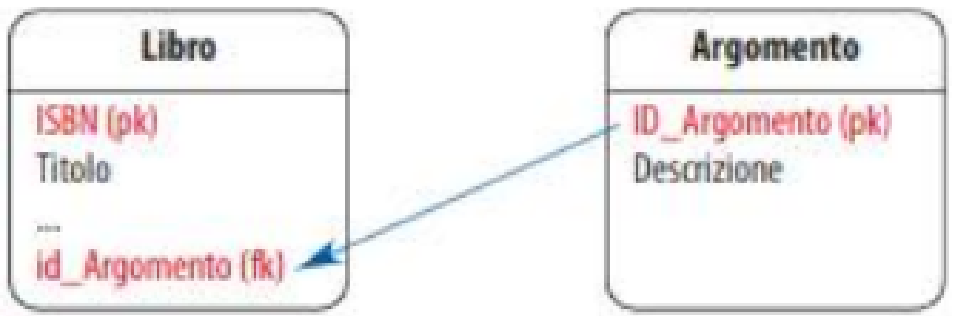
Le chiavi esterne (*foreign keys*) sono utilizzate per realizzare i collegamenti tra entità, risolvendo quindi i collegamenti uno-a-molti.

Se le chiavi esterne sono chiavi artificiali, vengono nominate in modo simile a quanto si fa per le chiavi primarie, con il prefisso id_ (minuscolo).

Chiavi esterne

Le chiavi esterne (*foreign keys*) sono utilizzate per realizzare i collegamenti tra entità, risolvendo quindi i collegamenti uno-a-molti.

Se le chiavi esterne sono chiavi artificiali, vengono nominate in modo simile a quanto si fa per le chiavi primarie, con il prefisso id_ (minuscolo).



Relazioni

Ogni associazione binaria ha:

- un'entità di partenza e una di arrivo;
- una descrizione che ne esprima il significato;
- eventuali attributi caratterizzanti.

Rappresentazione delle relazioni:

Relazioni

Ogni associazione binaria ha:

- un'entità di partenza e una di arrivo;
- una descrizione che ne esprima il significato;
- eventuali attributi caratterizzanti.

Rappresentazione delle relazioni:

Relazioni

Ogni associazione binaria ha:

- un'entità di partenza e una di arrivo;
- una descrizione che ne esprima il significato;
- eventuali attributi caratterizzanti.

Rappresentazione delle relazioni:

Relazioni

Ogni associazione binaria ha:

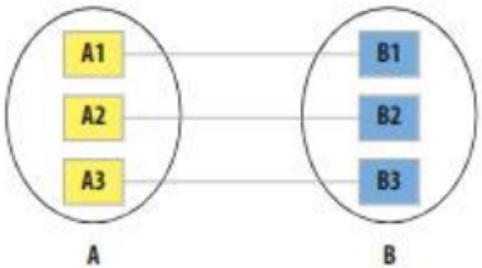
- un'entità di partenza e una di arrivo;
- una descrizione che ne esprima il significato;
- eventuali attributi caratterizzanti.

Rappresentazione delle relazioni:

Cardinalità delle relazioni

Una relazione può essere:

- uno-a-uno (1,1), espressa da una funzione biettiva tra due insiemi che contengono le istanze da associare;

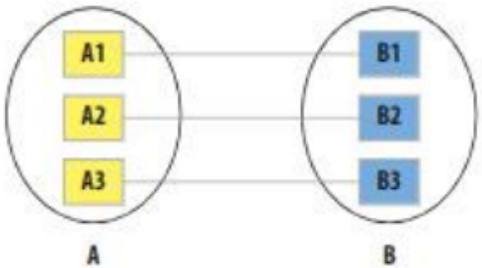


- uno-a-molti (1, n);
- molti-a-molti (n, n).

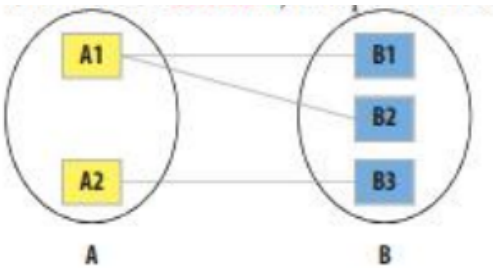
Cardinalità delle relazioni

Una relazione può essere:

- uno-a-uno (1,1), espressa da una funzione biettiva tra due insiemi che contengono le istanze da associare;



- uno-a-molti (1, n);

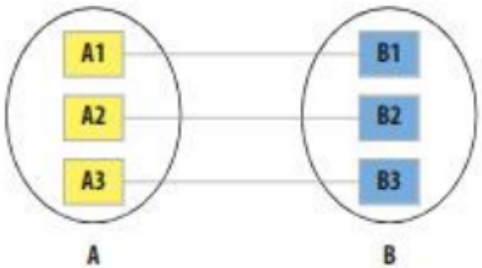


- molti-a-molti (n, n).

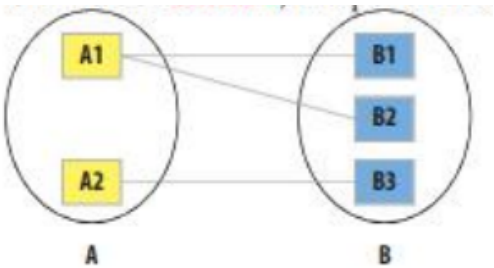
Cardinalità delle relazioni

Una relazione può essere:

- uno-a-uno (1,1), espressa da una funzione biettiva tra due insiemi che contengono le istanze da associare;

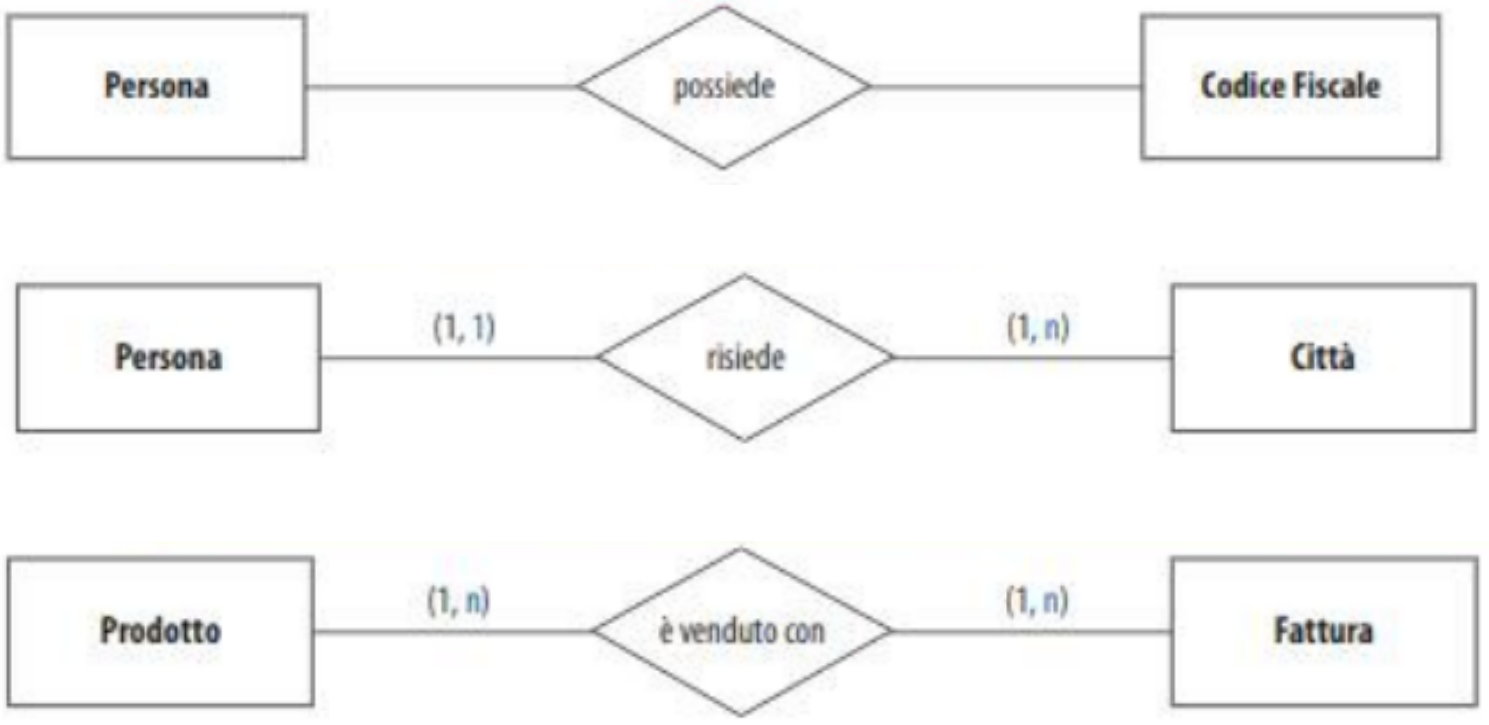


- uno-a-molti (1, n);

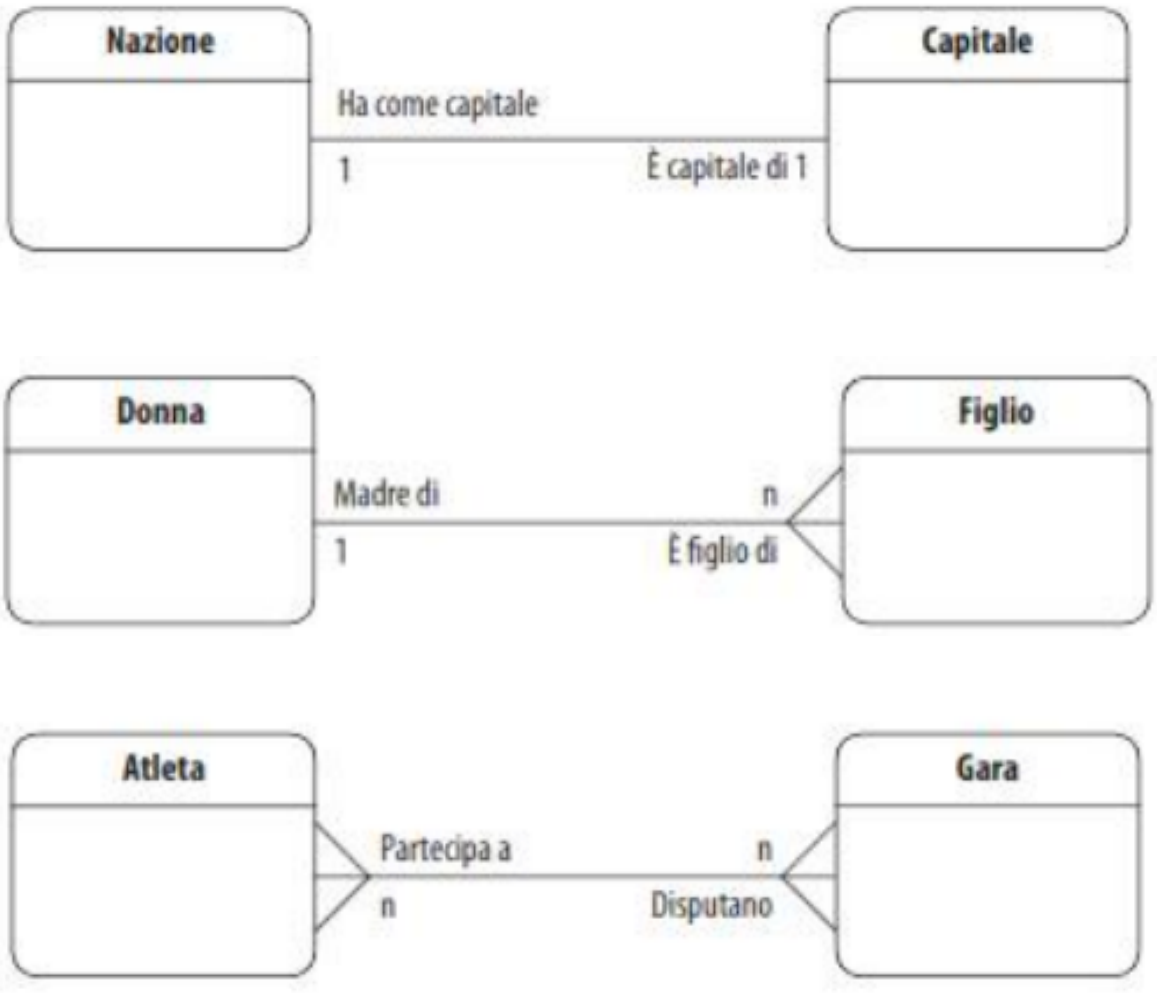


- molti-a-molti (n, n).

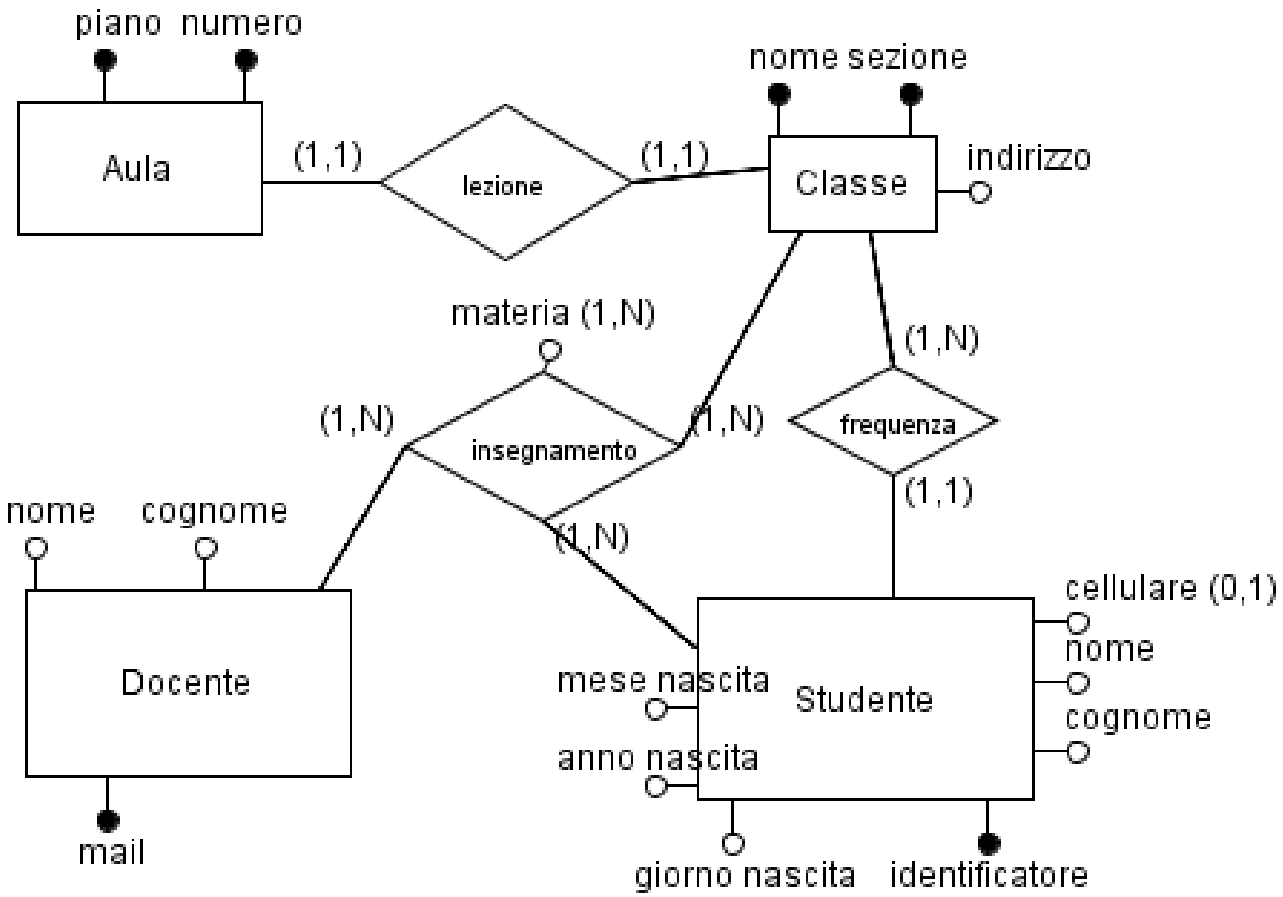
Rappresentazione della cardinalità delle relazioni (classica)



Rappresentazione della cardinalità delle relazioni (UML)



Esempio di diagramma E-R



Realizzazione di un database

La progettazione è solitamente svolta da un database designer, responsabile dell'astrazione dei dati dal mondo reale a partire dall'analisi dei requisiti fino alla corretta modellazione.

Il primo passo è la realizzazione dello schema concettuale, ovvero del diagramma E-R, inizialmente in forma provvisoria, poi via via più raffinata.

È necessario:

- definire gli oggetti del diagramma, eventualmente realizzando un glossario;
- completare le entità con i loro attributi;
- individuare le relazioni esistenti tra le entità.

Realizzazione di un database

La progettazione è solitamente svolta da un database designer, responsabile dell'astrazione dei dati dal mondo reale a partire dall'analisi dei requisiti fino alla corretta modellazione.

Il primo passo è la realizzazione dello schema concettuale, ovvero del diagramma E-R, inizialmente in forma provvisoria, poi via via più raffinata.

È necessario:

- definire gli oggetti del diagramma, eventualmente realizzando un glossario;
- completare le entità con i loro attributi;
- individuare le relazioni esistenti tra le entità.

Realizzazione di un database

La progettazione è solitamente svolta da un database designer, responsabile dell'astrazione dei dati dal mondo reale a partire dall'analisi dei requisiti fino alla corretta modellazione.

Il primo passo è la realizzazione dello schema concettuale, ovvero del diagramma E-R, inizialmente in forma provvisoria, poi via via più raffinata.

È necessario:

- definire gli oggetti del diagramma, eventualmente realizzando un glossario;
- completare le entità con i loro attributi;
- individuare le relazioni esistenti tra le entità.

Realizzazione di un database

La progettazione è solitamente svolta da un database designer, responsabile dell'astrazione dei dati dal mondo reale a partire dall'analisi dei requisiti fino alla corretta modellazione.

Il primo passo è la realizzazione dello schema concettuale, ovvero del diagramma E-R, inizialmente in forma provvisoria, poi via via più raffinata.

È necessario:

- definire gli oggetti del diagramma, eventualmente realizzando un glossario;
- completare le entità con i loro attributi;
- individuare le relazioni esistenti tra le entità.

Realizzazione di un database

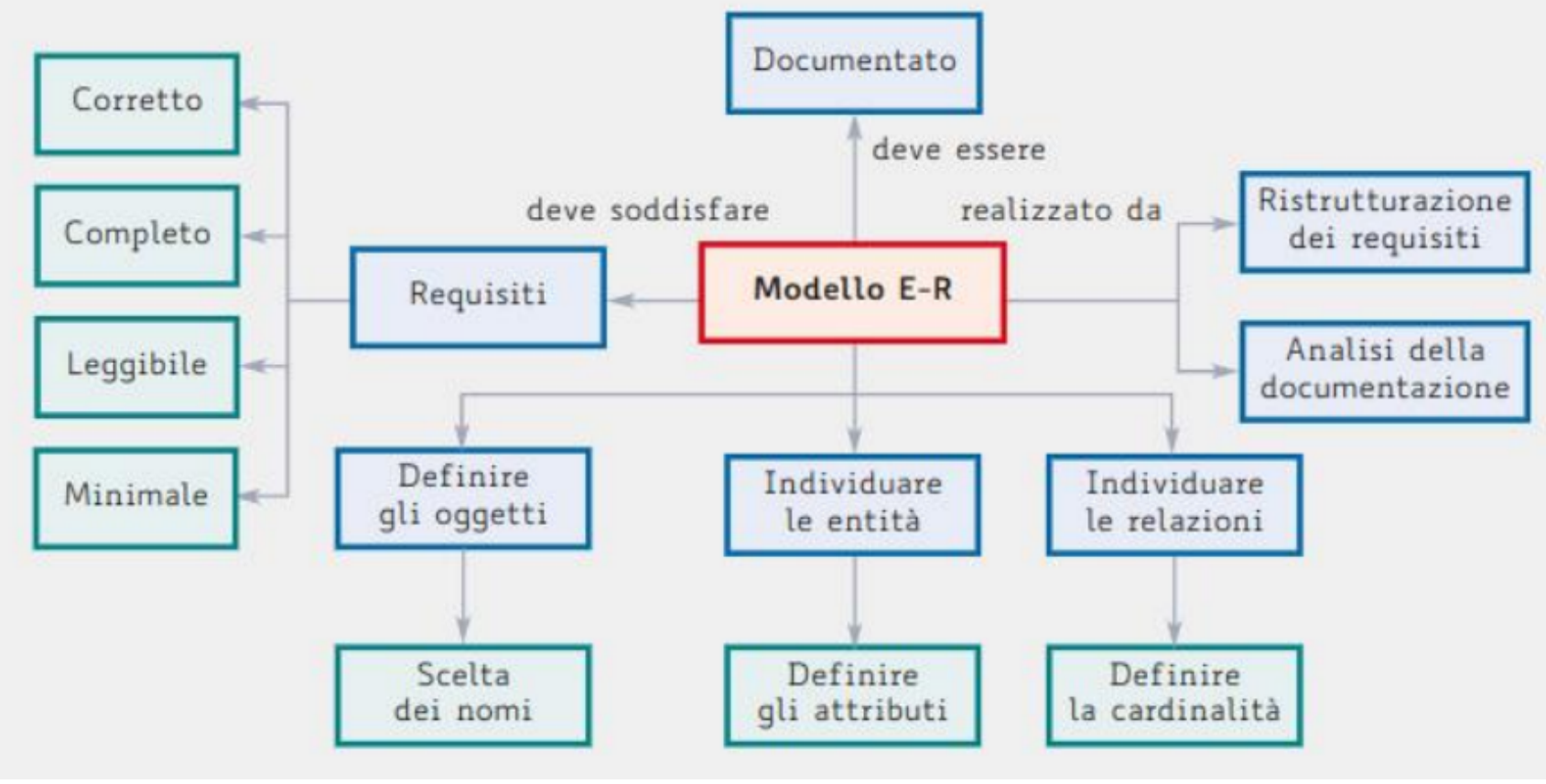
La progettazione è solitamente svolta da un database designer, responsabile dell'astrazione dei dati dal mondo reale a partire dall'analisi dei requisiti fino alla corretta modellazione.

Il primo passo è la realizzazione dello schema concettuale, ovvero del diagramma E-R, inizialmente in forma provvisoria, poi via via più raffinata.

È necessario:

- definire gli oggetti del diagramma, eventualmente realizzando un glossario;
- completare le entità con i loro attributi;
- individuare le relazioni esistenti tra le entità.

Schema di progettazione di un database



▸ Esempio di progettazione di un database

Realizzazione dello schema logico

Il modello (o schema) logico viene utilizzato come input per la progettazione vera e propria del database: deve avere quindi il massimo livello di dettaglio e precisione possibile.

Deve contenere tutte le informazioni necessarie per definire le tabelle, riportando la descrizione puntuale e completa del significato di ogni dato memorizzato.

Lo schema logico trasforma le informazioni del modello concettuale in un formato efficiente.

Utilizzeremo un modello relazionale,e dovremo quindi definire le tabelle relazionali.

Realizzazione dello schema logico

Il modello (o schema) logico viene utilizzato come input per la progettazione vera e propria del database: deve avere quindi il massimo livello di dettaglio e precisione possibile.

Deve contenere tutte le informazioni necessarie per definire le tabelle, riportando la descrizione puntuale e completa del significato di ogni dato memorizzato.

Lo schema logico trasforma le informazioni del modello concettuale in un formato efficiente.

Utilizzeremo un modello relazionale,e dovremo quindi definire le tabelle relazionali.

Realizzazione dello schema logico

Il modello (o schema) logico viene utilizzato come input per la progettazione vera e propria del database: deve avere quindi il massimo livello di dettaglio e precisione possibile.

Deve contenere tutte le informazioni necessarie per definire le tabelle, riportando la descrizione puntuale e completa del significato di ogni dato memorizzato.

Lo schema logico trasforma le informazioni del modello concettuale in un formato efficiente.

Utilizzeremo un modello relazionale,e dovremo quindi definire le tabelle relazionali.

Realizzazione dello schema logico

Il modello (o schema) logico viene utilizzato come input per la progettazione vera e propria del database: deve avere quindi il massimo livello di dettaglio e precisione possibile.

Deve contenere tutte le informazioni necessarie per definire le tabelle, riportando la descrizione puntuale e completa del significato di ogni dato memorizzato.

Lo schema logico trasforma le informazioni del modello concettuale in un formato efficiente.

Utilizzeremo un modello relazionale,e dovremo quindi definire le tabelle relazionali.

Mappatura

La traduzione delle tabelle dallo schema concettuale allo schema logico è un'operazione chiamata *mapping* (mappatura).

Un modello logico preciso comprende:

- per ogni entità:
  - l'elenco completo degli attributi;
  - l'indicazione della chiave primaria;
  - l'indicazione di eventuali chiavi esterne;
- per ogni attributo:
  - l'indicazione di opzionalità o di obbligatorietà;
  - l'indicazione del tipo di dati e dei valori ammessi;
- per ogni relazione:
  - l'indicazione della molteplicità minima e massima in entrambe le direzioni;
  - l'indicazione delle regole di integrità referenziale applicabili.

Dallo schema concettuale allo schema logico

Il passaggio dallo schema E-R allo schema logico comprende due fasi:

1. ristrutturazione del diagramma E-R;
2. traduzione del modello E-R nello schema logico e nelle tabelle relazionali.

Dallo schema concettuale allo schema logico

Il passaggio dallo schema E-R allo schema logico comprende due fasi:

1. ristrutturazione del diagramma E-R;
2. traduzione del modello E-R nello schema logico e nelle tabelle relazionali.

Dallo schema concettuale allo schema logico

Il passaggio dallo schema E-R allo schema logico comprende due fasi:

1. ristrutturazione del diagramma E-R;
2. traduzione del modello E-R nello schema logico e nelle tabelle relazionali.

Ristrutturazione

La ristrutturazione avviene secondo le seguenti regole:

1. Partecipazione delle entità alle relazioni: ogni entità creata deve essere collegata a qualche altra entità, altrimenti non sarà possibile collegare quella tabella alle altre.
2. Eliminazione degli attributi composti e aggiunta di attributi semplici: se sono presenti attributi composti, essi vanno eliminati e sostituiti con attributi semplici.
3. Eliminazione degli attributi multivalore e aggiunta di entità: gli eventuali attributi multivalore devono essere "promossi" a entità, creando cioè una nuova entità che contenga i valori dell'attributo e collegandola all'entità che possedeva l'attributo mediante una nuova relazione opportuna.

Ristrutturazione

La ristrutturazione avviene secondo le seguenti regole:

1. Partecipazione delle entità alle relazioni: ogni entità creata deve essere collegata a qualche altra entità, altrimenti non sarà possibile collegare quella tabella alle altre.
2. Eliminazione degli attributi composti e aggiunta di attributi semplici: se sono presenti attributi composti, essi vanno eliminati e sostituiti con attributi semplici.
3. Eliminazione degli attributi multivalore e aggiunta di entità: gli eventuali attributi multivalore devono essere "promossi" a entità, creando cioè una nuova entità che contenga i valori dell'attributo e collegandola all'entità che possedeva l'attributo mediante una nuova relazione opportuna.

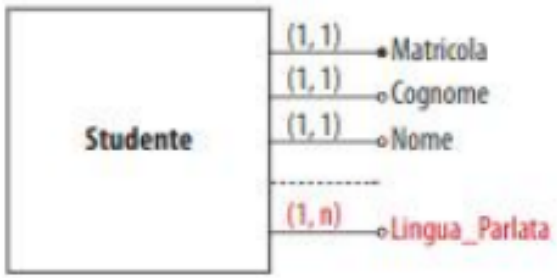
Ristrutturazione

La ristrutturazione avviene secondo le seguenti regole:

1. Partecipazione delle entità alle relazioni: ogni entità creata deve essere collegata a qualche altra entità, altrimenti non sarà possibile collegare quella tabella alle altre.
2. Eliminazione degli attributi composti e aggiunta di attributi semplici: se sono presenti attributi composti, essi vanno eliminati e sostituiti con attributi semplici.
3. Eliminazione degli attributi multivalore e aggiunta di entità: gli eventuali attributi multivalore devono essere "promossi" a entità, creando cioè una nuova entità che contenga i valori dell'attributo e collegandola all'entità che possedeva l'attributo mediante una nuova relazione opportuna.

Esempio: attributo multivalore

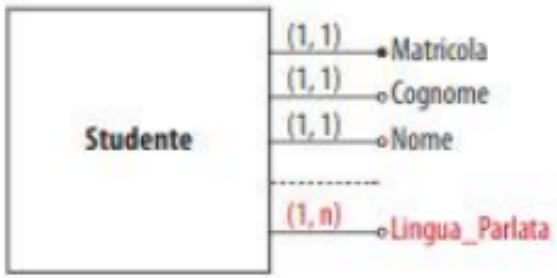
Ad esempio, prima della ristrutturazione:



Dopo la ristrutturazione:

Esempio: attributo multivalore

Ad esempio, prima della ristrutturazione:



Dopo la ristrutturazione:

Semplificazione (1)

Dopo aver ottenuto lo schema ristrutturato, possiamo tradurlo nel modello/schema relazionale.

Si seguono le seguenti regole:

1. Trasformazione delle entità: per ogni entità viene generata una tabella indicando lo schema relazionale, in cui viene riportato un campo per ogni attributo dell'entità.
   Il diagramma E-R:

   viene tradotto nel seguente schema relazionale:

Persona (Codice_Fiscale(pk), Cognome, Nome)

Semplificazione (2)

2. Trasformazione delle relazioni: si procede in base al numero di entità partecipanti e alla loro cardinalità.

   - Unificare le relazioni uno-a-uno: se due entità sono collegate da una relazione (1,1), possono essere ridotte ad un'unica entità che contiene gli attributi sia della prima sia della seconda entità.
   - Eliminare le relazioni complesse, che coinvolgono più di una entità, riducendole a relazioni binarie.
   - Semplificare le relazioni molti-a-molti: le relazioni molti-a-molti non sono supportate nel modello relazionale e devono essere sostituite creando un'entità associativa che va messa in relazione con le due entità originali.

Semplificazione (2)

2. Trasformazione delle relazioni: si procede in base al numero di entità partecipanti e alla loro cardinalità.

   - Unificare le relazioni uno-a-uno: se due entità sono collegate da una relazione (1,1), possono essere ridotte ad un'unica entità che contiene gli attributi sia della prima sia della seconda entità.
   - Eliminare le relazioni complesse, che coinvolgono più di una entità, riducendole a relazioni binarie.
   - Semplificare le relazioni molti-a-molti: le relazioni molti-a-molti non sono supportate nel modello relazionale e devono essere sostituite creando un'entità associativa che va messa in relazione con le due entità originali.

Semplificazione (2)

2. Trasformazione delle relazioni: si procede in base al numero di entità partecipanti e alla loro cardinalità.

   - Unificare le relazioni uno-a-uno: se due entità sono collegate da una relazione (1,1), possono essere ridotte ad un'unica entità che contiene gli attributi sia della prima sia della seconda entità.

   - Eliminare le relazioni complesse, che coinvolgono più di una entità, riducendole a relazioni binarie.

   - Semplificare le relazioni molti-a-molti: le relazioni molti-a-molti non sono supportate nel modello relazionale e devono essere sostituite creando un'entità associativa che va messa in relazione con le due entità originali.

Semplificazione (2)

2. Trasformazione delle relazioni: si procede in base al numero di entità partecipanti e alla loro cardinalità.

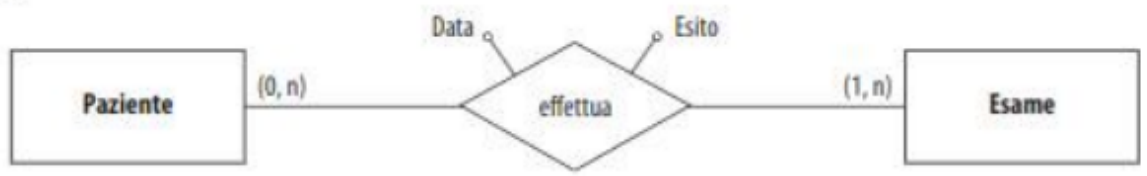
   - Unificare le relazioni uno-a-uno: se due entità sono collegate da una relazione (1,1), possono essere ridotte ad un'unica entità che contiene gli attributi sia della prima sia della seconda entità.

   - Eliminare le relazioni complesse, che coinvolgono più di una entità, riducendole a relazioni binarie.

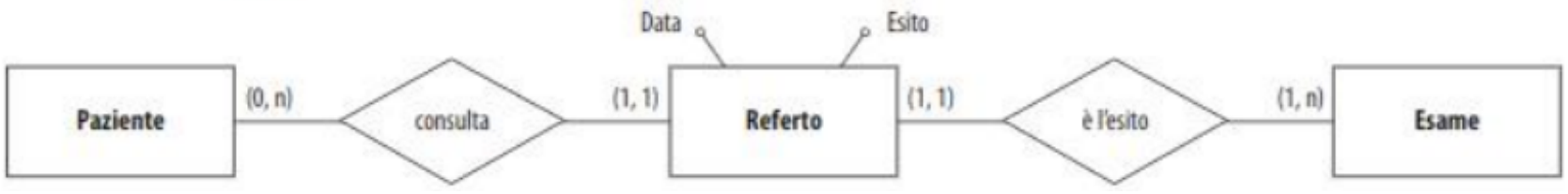
   - Semplificare le relazioni molti-a-molti: le relazioni molti-a-molti non sono supportate nel modello relazionale e devono essere sostituite creando un'entità associativa che va messa in relazione con le due entità originali.

Semplificazione (3)

Ad esempio, prima della semplificazione:



Dopo la semplificazione:



Nello schema relazionale otteniamo:

Paziente (ID_Paziente (pk), Cognome, Nome)
Esame (ID_Esame(pk), descrizione)
Referto (ID_Paziente(pk), ID_esame (pk), Data, Esito)

Schema riassuntivo

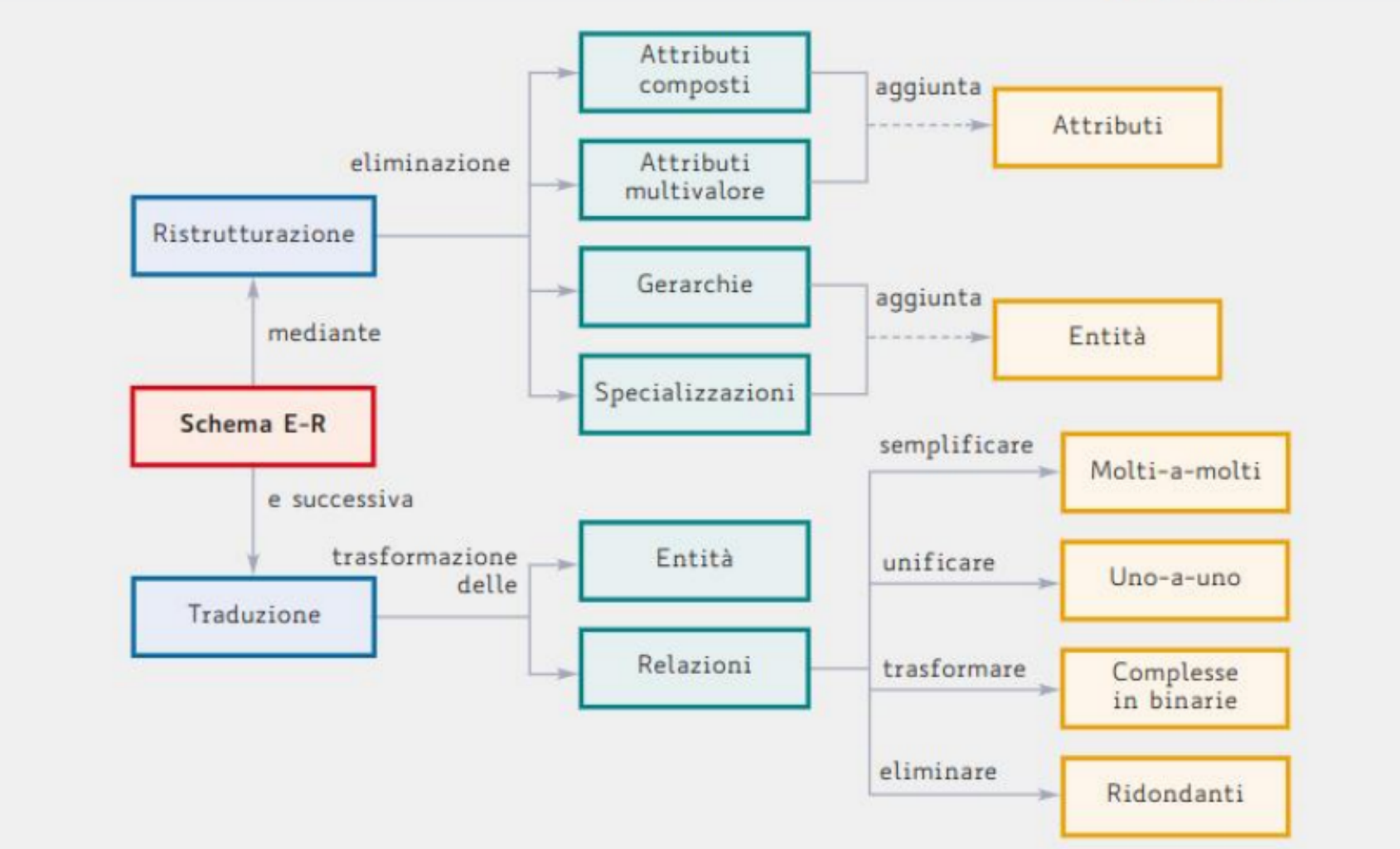


Tabelle relazionali (1)

Il modello relazionale, introdotto da E.F. Codd nel 1970, presenta i dati nella forma di tabelle bidimensionali.

Ogni tabella ha:

- un nome per ogni colonna;
- un elenco di colonne di cui vengono specificati i tipi;
- una chiave primaria che identifica univocamente ogni riga della tabella.

Tra due o più colonne di tabelle ci può essere una relazione: le due tabelle sono allora collegate tramite chiavi esterne.

Tabelle relazionali (1)

Il modello relazionale, introdotto da E.F. Codd nel 1970, presenta i dati nella forma di tabelle bidimensionali.

Ogni tabella ha:

- un nome per ogni colonna;
- un elenco di colonne di cui vengono specificati i tipi;
- una chiave primaria che identifica univocamente ogni riga della tabella.

Tra due o più colonne di tabelle ci può essere una relazione: le due tabelle sono allora collegate tramite chiavi esterne.

Tabelle relazionali (1)

Il modello relazionale, introdotto da E.F. Codd nel 1970, presenta i dati nella forma di tabelle bidimensionali.

Ogni tabella ha:

- un nome per ogni colonna;
- un elenco di colonne di cui vengono specificati i tipi;
- una chiave primaria che identifica univocamente ogni riga della tabella.

Tra due o più colonne di tabelle ci può essere una relazione: le due tabelle sono allora collegate tramite chiavi esterne.

Tabelle relazionali (1)

Il modello relazionale, introdotto da E.F. Codd nel 1970, presenta i dati nella forma di tabelle bidimensionali.

Ogni tabella ha:

- un nome per ogni colonna;
- un elenco di colonne di cui vengono specificati i tipi;
- una chiave primaria che identifica univocamente ogni riga della tabella.

Tra due o più colonne di tabelle ci può essere una relazione: le due tabelle sono allora collegate tramite chiavi esterne.

Tabelle relazionali (1)

Il modello relazionale, introdotto da E.F. Codd nel 1970, presenta i dati nella forma di tabelle bidimensionali.

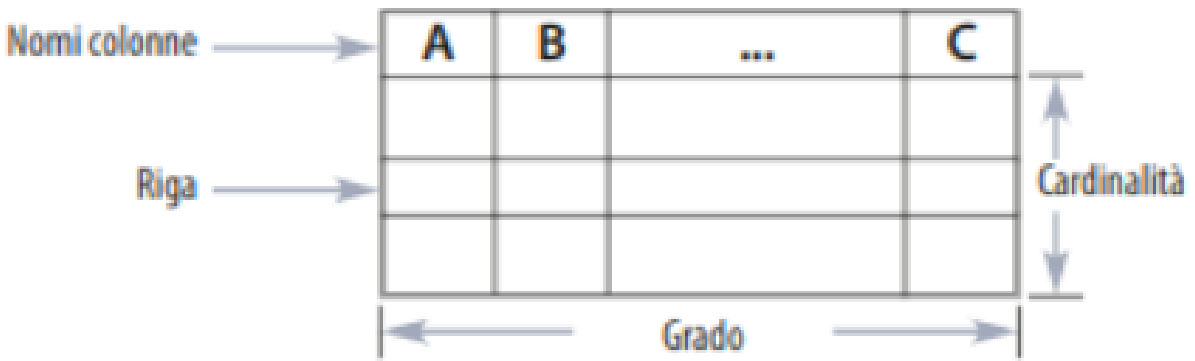
Ogni tabella ha:

- un nome per ogni colonna;
- un elenco di colonne di cui vengono specificati i tipi;
- una chiave primaria che identifica univocamente ogni riga della tabella.

Tra due o più colonne di tabelle ci può essere una relazione: le due tabelle sono allora collegate tramite chiavi esterne.

Terminologia

Le colonne possono anche essere chiamate “campi” o “attributi”, mentre le righe possono anche essere chiamate “record” o “tuple”.



Traduzione (1)

Seguiamo queste quattro regole:

1. ogni entità diventa una tabella;
2. ogni attributo dell'entità diventa una colonna della tabella;
3. ogni istanza di un'entità diventa una riga della tabella;
4. la chiave primaria dell'entità è l'identificatore univoco delle righe di una tabella;
5. le relazioni sono rappresentate con chiavi esterne che collegano righe di due diverse tabelle.

Traduzione (1)

Seguiamo queste quattro regole:

1. ogni entità diventa una tabella;
2. ogni attributo dell'entità diventa una colonna della tabella;
3. ogni istanza di un'entità diventa una riga della tabella;
4. la chiave primaria dell'entità è l'identificatore univoco delle righe di una tabella;
5. le relazioni sono rappresentate con chiavi esterne che collegano righe di due diverse tabelle.

Traduzione (1)

Seguiamo queste quattro regole:

1. ogni entità diventa una tabella;
2. ogni attributo dell'entità diventa una colonna della tabella;
3. ogni istanza di un'entità diventa una riga della tabella;
4. la chiave primaria dell'entità è l'identificatore univoco delle righe di una tabella;
5. le relazioni sono rappresentate con chiavi esterne che collegano righe di due diverse tabelle.

Traduzione (1)

Seguiamo queste quattro regole:

1. ogni entità diventa una tabella;
2. ogni attributo dell'entità diventa una colonna della tabella;
3. ogni istanza di un'entità diventa una riga della tabella;
4. la chiave primaria dell'entità è l'identificatore univoco delle righe di una tabella;
5. le relazioni sono rappresentate con chiavi esterne che collegano righe di due diverse tabelle.

Traduzione (1)

Seguiamo queste quattro regole:

1. ogni entità diventa una tabella;
2. ogni attributo dell'entità diventa una colonna della tabella;
3. ogni istanza di un'entità diventa una riga della tabella;
4. la chiave primaria dell'entità è l'identificatore univoco delle righe di una tabella;
5. le relazioni sono rappresentate con chiavi esterne che collegano righe di due diverse tabelle.

Traduzione (1)

Seguiamo queste quattro regole:

1. ogni entità diventa una tabella;
2. ogni attributo dell'entità diventa una colonna della tabella;
3. ogni istanza di un'entità diventa una riga della tabella;
4. la chiave primaria dell'entità è l'identificatore univoco delle righe di una tabella;
5. le relazioni sono rappresentate con chiavi esterne che collegano righe di due diverse tabelle.

Traduzione (2)

Ad esempio, se nello schema concettuale abbiamo:

nello schema logico otteniamo:

Clienti (ID_Cliente(pk), Ragione_Sociale, Indirizzo, Partita_IVA)
Fatture (Numero(pk), Data, Importo, id_Cliente(fk))

Traduzione (3)

Si è soliti, passando allo schema logico, inserire la chiave primaria nella prima colonna a sinistra, mentre le chiavi esterne nelle ultime colonne a destra.

Tabella Clienti

ID_Cliente	Ragione_Sociale	Indirizzo	Partita_IVA
120	Rossi SRL	Via Roma, 3	10203040501
220	Verdi snc	Via Torino, 23	20304040501
333	Gialli SPA	Via Napoli, 11	40503040501
520	Rossi SRL	Via Genova, 13	30503040501

Istanze della tabella – Tuple – Record

Attributi – Campi

Tabella Fatture

Numero	Data	Importo	id_Cliente
70	01/04/2018	1200,20	120
71	03/04/2018	1450,29	520
72	04/04/2018	1200,20	120
73	05/04/2018	2300,45	220
74	06/04/2018	1200,20	120

Tabelle relazionali (2)

Le tabelle relazionali hanno le seguenti proprietà:

- i valori sono atomici, non ulteriormente scomponibili;
- i valori di una colonna sono dello stesso tipo;
- ogni riga è univoca, diversa dalle altre (garantito dalla presenza di una chiave primaria);
- l'ordine di righe o colonne non è rilevante (permette di aggiungere righe o colonne successivamente alla creazione del database);
- ogni colonna ha un nome univoco.

Tabelle relazionali (2)

Le tabelle relazionali hanno le seguenti proprietà:

- i valori sono atomici, non ulteriormente scomponibili;
- i valori di una colonna sono dello stesso tipo;
- ogni riga è univoca, diversa dalle altre (garantito dalla presenza di una chiave primaria);
- l'ordine di righe o colonne non è rilevante (permette di aggiungere righe o colonne successivamente alla creazione del database);
- ogni colonna ha un nome univoco.

Tabelle relazionali (2)

Le tabelle relazionali hanno le seguenti proprietà:

- i valori sono atomici, non ulteriormente scomponibili;
- i valori di una colonna sono dello stesso tipo;
- ogni riga è univoca, diversa dalle altre (garantito dalla presenza di una chiave primaria);
- l'ordine di righe o colonne non è rilevante (permette di aggiungere righe o colonne successivamente alla creazione del database);
- ogni colonna ha un nome univoco.

Tabelle relazionali (2)

Le tabelle relazionali hanno le seguenti proprietà:

- i valori sono atomici, non ulteriormente scomponibili;
- i valori di una colonna sono dello stesso tipo;
- ogni riga è univoca, diversa dalle altre (garantito dalla presenza di una chiave primaria);
- l'ordine di righe o colonne non è rilevante (permette di aggiungere righe o colonne successivamente alla creazione del database);
- ogni colonna ha un nome univoco.

Tabelle relazionali (2)

Le tabelle relazionali hanno le seguenti proprietà:

- i valori sono atomici, non ulteriormente scomponibili;
- i valori di una colonna sono dello stesso tipo;
- ogni riga è univoca, diversa dalle altre (garantito dalla presenza di una chiave primaria);
- l'ordine di righe o colonne non è rilevante (permette di aggiungere righe o colonne successivamente alla creazione del database);
- ogni colonna ha un nome univoco.

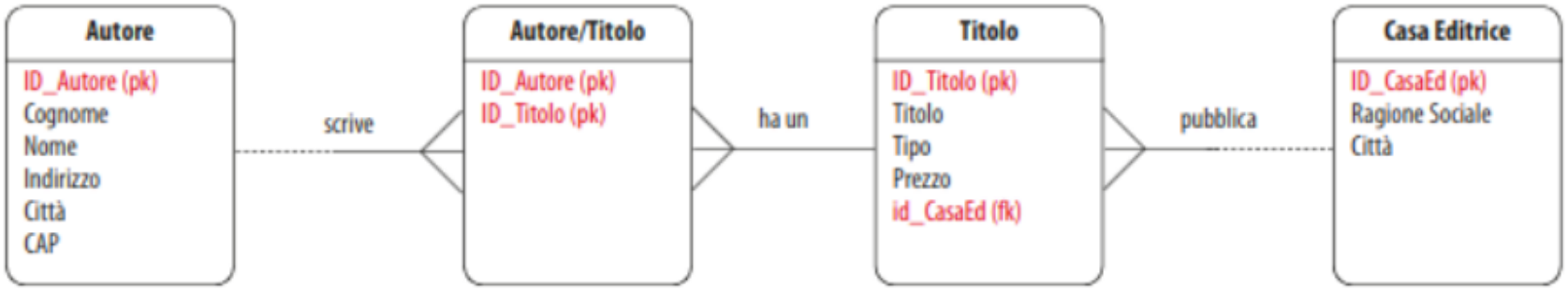
Tabelle relazionali (2)

Le tabelle relazionali hanno le seguenti proprietà:

- i valori sono atomici, non ulteriormente scomponibili;
- i valori di una colonna sono dello stesso tipo;
- ogni riga è univoca, diversa dalle altre (garantito dalla presenza di una chiave primaria);
- l'ordine di righe o colonne non è rilevante (permette di aggiungere righe o colonne successivamente alla creazione del database);
- ogni colonna ha un nome univoco.

Esempio

Per un database bibliografico con il seguente schema concettuale:



si ottiene il seguente schema relazionale:

Autori (ID_Autore(pk), Cognome, Nome, Indirizzo, Città, CAP)
Titoli (ID_Titolo(pk), Titolo, Tipo, Prezzo, id_CasaEd(fk))
Case Editrici (ID_CasaEd(pk), Ragione Sociale, Città)
Autori_Titoli (ID_Autore(pk), ID_Titolo(pk))

Riempire (popolare) le tabelle

AUTORI

ID_Autore	Cognome	Nome	Indirizzo	Città	CAP
172	Bianchi	Mario	Via Garibaldi, 10	Brescia	25100
156	Cerreti	Giovanni	Via Carducci, 12	Milano	20100
268	Rossigni	Giacomo	Piazza Caduti, 20	Torino	10100
884	Tacchi	Alfio	Borgo Trento	Treviso	31100
343	Bertozzi	Filippo	Via del lago, 9	Arenzano	16011
298	Sommo	Norberto	Largo Roma, 3	Como	22100

AUTORI_TITOLI

ID_Autore	ID_Titolo
172	7771010
156	7771213
268	7772123
884	7773244
343	7771356
298	7774744

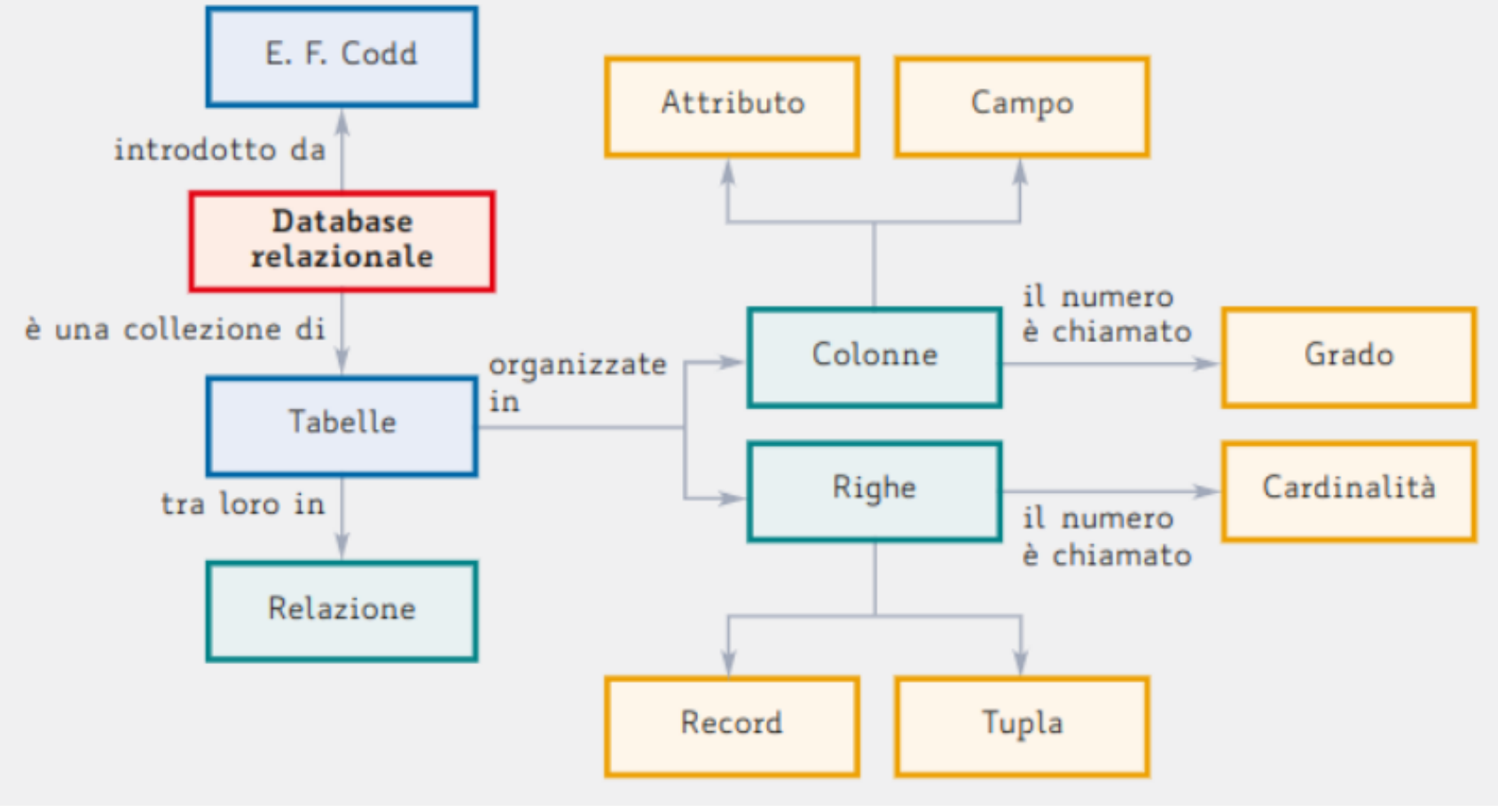
TITOLI

ID_Titolo	Titolo	Tipo	Prezzo	id_CasaEd
7771010	*C++*	Informatica	20.00	91234
7771213	*La cucina mediterranea*	Cucina	18.50	92233
7772123	*Aumentare gli utili*	Economia	36.70	94354
7773244	*L'ultima neve*	Thriller	15.20	95555
7771356	*Stress: il nostro nemico*	Psicologia	18.40	94354
7774744	*Le leggi di Mendel*	Biologia	21.00	95555

CASE EDITRICI

ID_CasaEd	Ragione Sociale	Città
91234	Tecniche OK	Brescia
92233	Nuove Gastronomie	Milano
94354	Hoepli	Milano
94354	Mondo Libri	Torino
95555	New Gothics	Treviso

Schema riassuntivo



Regole di integrità e normalizzazione

▸ Materiale completo sui database

Quando vengono eseguite operazioni sui dati presenti nel database, come inserimento, modifica o cancellazione, è necessario che vengano rispettate le definizioni del database. A tale scopo vengono fissate delle specifiche regole di integrità (cap. 9).

La normalizzazione è un processo che tende a eliminare la ripetizione (ridondanza) dei dati e a migliorarne la consistenza. Questo processo punta a ottenere la tabella in "forma normale" (cap. 10).

Regole di integrità e normalizzazione

▸ Materiale completo sui database

Quando vengono eseguite operazioni sui dati presenti nel database, come inserimento, modifica o cancellazione, è necessario che vengano rispettate le definizioni del database. A tale scopo vengono fissate delle specifiche regole di integrità (cap. 9).

La normalizzazione è un processo che tende a eliminare la ripetizione (ridondanza) dei dati e a migliorarne la consistenza. Questo processo punta a ottenere la tabella in "forma normale" (cap. 10).