Cap. 2 - Il Modello Relazionale

Concetti e definizioni

Introduzione

- Modello proposto nel 1970 da E.F. Codd in:
  - "A relational model for large shared data banks"
    Communications of the ACM Vol. 13, n. 6, pagg. 377-387
- Prime apparizioni nel mercato solo nel 1981
- Caratterizzato da un alto livello di astrazione
  - proposto per superare le limitazioni precedenti
  - caratterizzato da una elevata indipendenza dei dati
  - ha richiesto l'individuazione di realizzazioni efficienti e di hardware adeguato.

I fattori del successo

- Il modello relazionale si fonda su due concetti:
  - La relazione
    - Definizione formale
    - Ereditata dalla teoria degli insiemi
    - Utile per completare il modello con una precisa teoria
  - La tabella
    - Semplice ed intuitiva
    - Rappresentazione grafica
    - Utile nella comunicazione con gli utenti

Relazione: definizione

- Dati n>0 domini non necessariamente distinti

  $D_1, D_2, \ldots, D_n$

  una relazione matematica **r** sui domini D_i è un sottoinsieme, anche vuoto, del prodotto cartesiano

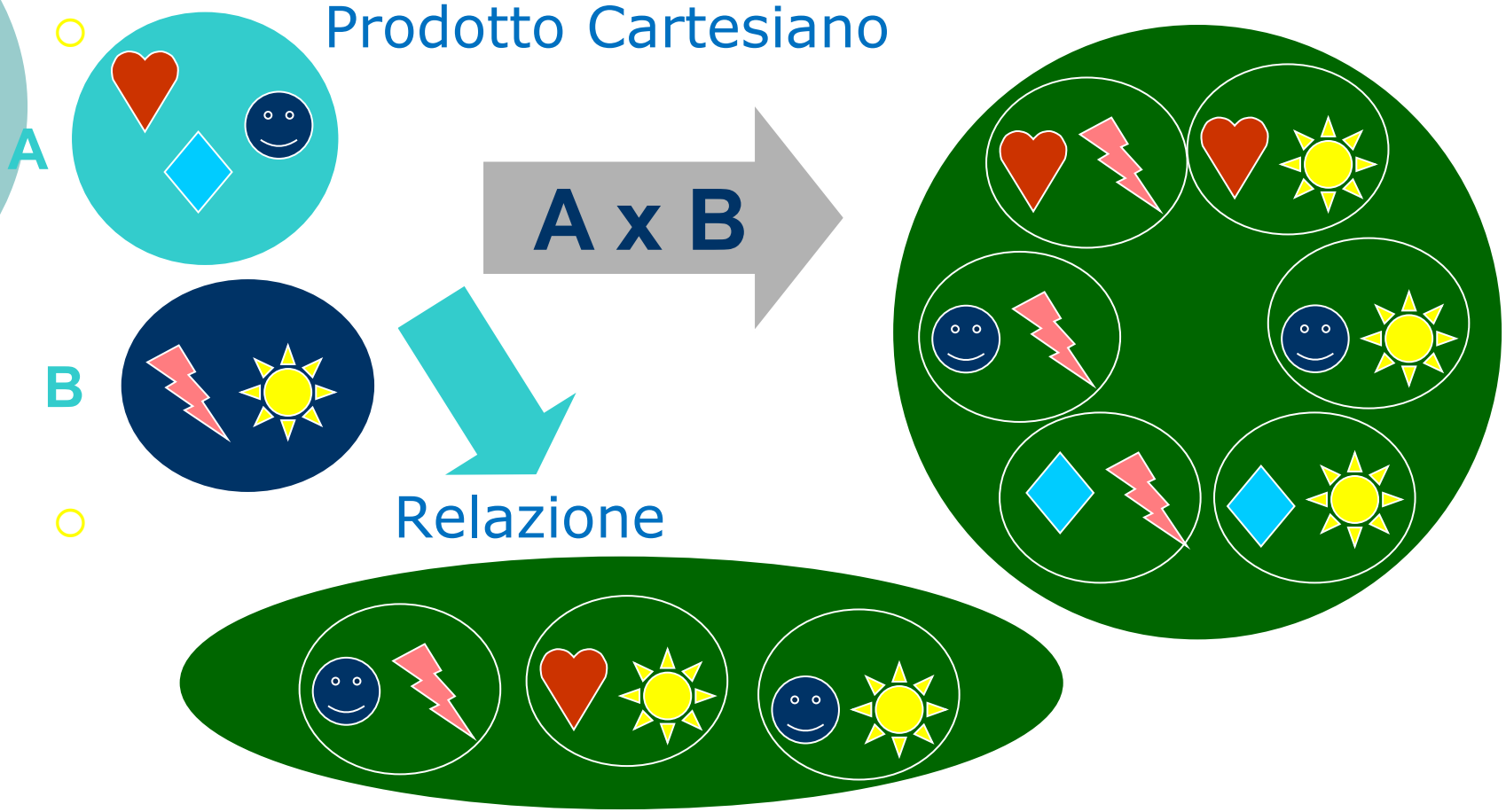
  $r \subseteq D_1 \times D_2 \times \ldots \times D_n$

- Nella teoria relazionale dei dati si fa l'ulteriore ipotesi:
  - **i domini sono "a valori atomici"**

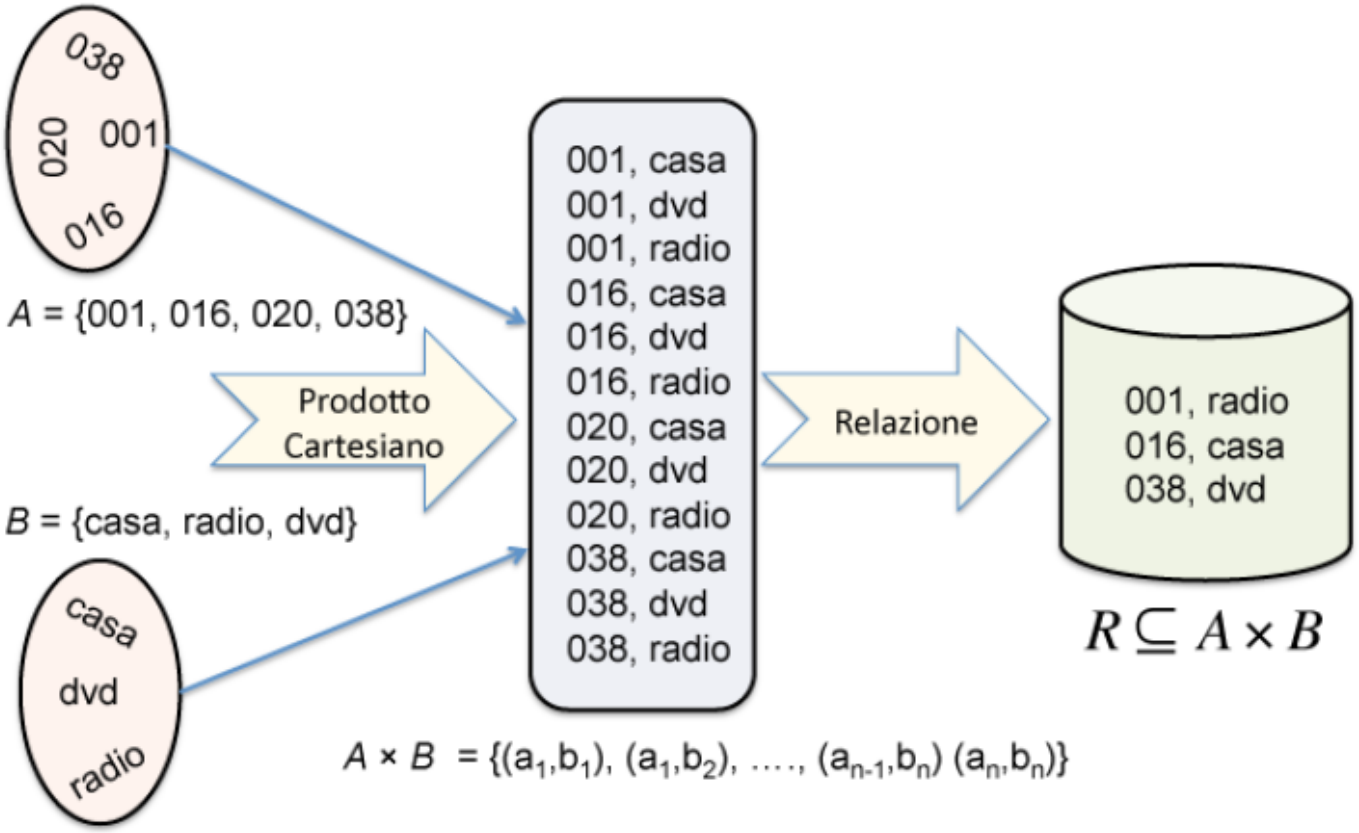
Definizioni

Relazione: schematizzazione grafica

Definizioni



Relazione: schematizzazione grafica



Relazione

E’ vista come un insieme di ennuple ordinate:

$t = (v_1, v_2, \ldots, v_n) \mid v_1 \in D_1, v_2 \in D_2, \ldots, v_n \in D_n$

- Ciascuna ennupla si chiama, nella terminologia relazionale, **tupla**
- Il numero **n** è detto **grado** del **prodotto cartesiano** e della **relazione**
- Il numero di tuple della relazione viene detto **cardinalità della relazione**

Definizioni

Esempio (Relazione)

Siano dati i seguenti domini:

***Codice* = {001, 004, 005}**
***Nome* = {*Mel, Pedro, Federico*}**
***Cognome* = {*Almodovar, Gibson, Fellini*},**
***Nazionalità*= {*Italia, Spagna, Australia*}.**

Una relazione su questi domini è una generica :

***r* ⊆ Codice x Nome x Cognome x Nazionalità**

Possibili relazioni sono dunque:

***r*1 = {(001, *Pedro, Almodovar, Spagna*)}**
***r*2 = {(001, *Pedro, Almodovar, Spagna*);**
(004, *Mel, Gibson, Australia*)}

Considerazioni

- Il valore di **n** è **finito**
  - Rappresentazione finita delle informazioni
- **La cardinalità dei domini può essere considerata infinita**
  - Si pensi all'insieme dei cognomi delle persone
  - Può essere utile considerare che esista una n-upla non presente nella relazione
- I domini possono essere
  - Tutti dello stesso tipo
  - Di tipo diverso
  - Non tutti dello stesso tipo

Proprietà di una relazione

1. non è definito alcun ordinamento tra le tuple di una relazione;
2. ogni tupla è distinta da tutte le altre;
3. esiste una corrispondenza di tipo posizionale tra i valori interni ad una tupla ed i relativi domini:

$t = (v_1, v_2, \ldots, v_n) \mid v_1 \in D_1, v_2 \in D_2, \ldots, v_n \in D_n$

Considerazioni

è possibile eliminare la proprietà 3 se:

- ad ogni valore di un dominio D_i di una relazione si associa un **attributo** A_i che permette di identificare e qualificare il ruolo del dominio
  - **dom : A -> D**

- Da quanto detto la definizione di tupla si modifica in:

  $t = \{\langle v_1, A_1\rangle, \langle v_2, A_2\rangle, \ldots, \langle v_n, A_n\rangle\}$

  per cui:

  3. **non esiste alcun ordinamento all'interno di una tupla.**

Rappresentazione di una relazione

- Una relazione può essere rappresentata naturalmente attraverso le **tabelle** dove
  - ogni riga è una tupla
  - ogni colonna è data dai valori relativi ad un certo attributo che ne è l'intestazione.

- Non tutte le tabelle rappresentano relazioni:

  lo sono se e solo se sono soddisfatte le proprietà 1 e 2.

Tabelle e Relazioni

- La seguente tabella corrisponde ad una relazione con i seguenti

ATTRIBUTO DOMINIO
Codice {001, 004, 005, 006}
Nome {Mel, Pedro, Federico}
Cognome {Almodovar, Gibson, Fellini},
Nazionalità {Italia, Spagna, Australia}.

Codice	Nome	Cognome	Nazionalità
001	Pedro	Almodovar	Spagna
004	Federico	Fellini	Italia
005	Mel	Gibson	Australia


Schema di Relazione

Dato un insieme di nomi di attributi

$$X = \{A_1, A_2, \ldots, A_n\},$$

si definisce "schema di relazione" ***un nome*** *R,* ***seguito da un insieme di nomi di attributi*** *X:*

$$R(X) = R(A_1, A_2, \ldots, A_n)$$

Definizioni

Esempi di Schema di Relazione

- *Sono esempi di schema di relazione:*

AUTORI (Codice,Nome,Cognome,Nazionalità)

FILM (Autore, Titolo, Anno)

Relazione su uno schema

- Si definisce **relazione r** su uno schema di relazione R(X) **una istanza di R(X)**

- **Schema**: componente intensionale
- **Istanza**: componente estensionale

Notazione importante

Sia i una tupla definita su un insieme di attributi X. Con la notazione $t_i[A]$ definiamo il valore della tupla i relativamente all'attributo $A \subseteq X$.

È possibile estendere la stessa notazione ad un sottoinsieme di attributi $Y \subseteq X$: in questo caso con il termine $t_i[Y]$ indichiamo il valore della tupla i ristretta ai soli attributi Y.

Esempio

NomeStudente	NomeEsame	Voto	Lode
Pippo	TSI	30	SI
Paolo	TM	28	NO
Marta	EI	18	SI
Maria	AI	33	NO

Considerando la seconda tupla di questa relazione si ha:

t_2 [*NomeStudente*] = *Paolo*

t_2 [*NomeStudente, Voto*] = (*Paolo,* 28).

Informazioni incomplete

- Il modello relazionale impone strutture rigide alle informazioni:
  - Una relazione è un insieme di tuple omogenee (sullo stesso schema)
  - Per alcune tuple può accadere che non sia definito il valore di alcuni attributi

Come gestire l'assenza di informazione?

- Riempendo i campi con valori opportuni?
- E come sceglierli?

Informazioni incomplete

- In questi casi, si è soliti estendere i domini delle relazioni con un valore speciale, detto NULL

$$D_i = D_i \textbf{ U } \text{NULL}.$$

Con il valore **NULL** si intende prendere in considerazione una **assenza di informazione** che può essere dovuta a diversi fattori:

1. il dato esiste ma non è stato fornito (**valore sconosciuto**)
2. il dato non esiste, in quanto non è applicabile alla specifica tupla (**valore inesistente**)
3. il dato non c'è, ma non si sa se è sconosciuto oppure inesistente (**valore senza informazione**)

Esempio

Ipotesi: ogni studente è dotato di cellulare, mentre il professore è dotato di cellulare e di telefono di ufficio.

Nome	Ruolo	Cellulare	Ufficio
Giacomo	STUD	335123123	NULL
Antonio	PROF	335123131	0817683826
Marta	STUD	NULL	NULL
Annarita	NULL	334123123	NULL

- nella prima tupla NULL al telefono ufficio per uno studente, indica la non applicabilità dell'informazione;
- la terza tupla, presenta NULL nel campo cellulare di Marta, una *STUD,* pertanto il valore è sconosciuto.
- l'ultima tupla, non potendo stabilire se Annarita è *STUD* o *PROF*, non si può dire nulla (senza informazione).

Basi di dati e vincoli di integrità

Definizioni

Un vincolo di integrità o *integrity constraint*, è una **regola** che **ogni** istanza di uno schema di relazione **deve rispettare** affinchè i suoi dati siano corrispondenti al modello della realtà che una BD cattura.

Definizioni

- **Schema di una Base di Dati:**

  È costituito da:

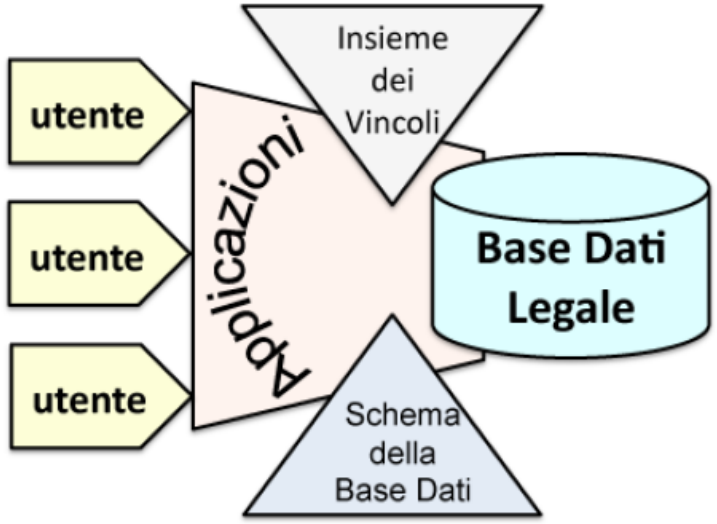
  - *il nome della base di dati B_D,*
  - *dagli schemi di relazione $R_1(X_1), R_2(X_2), \ldots, R_n(X_n)$,*
  - *un insieme IC di regole di integrità.*

- **Base di Dati Relazionale:**

  E' una istanza di uno schema di base di dati che soddisfa l'insieme IC di regole.

Considerazioni

- Se una istanza soddisfa tutti i vincoli di integrità specificati nello schema della base di dati, si parla allora di **istanza legale della base di dati**.
- Compito del DBMS è verificare i vincoli di integrità generando istanze corrette.



Tipi di vincoli

- Si possono imporre vincoli che coinvolgono una singola relazione:
  - Vincoli intra-relazionali
    - sui valori di un attributo (**vincoli di dominio**)
    - su più attributi della tupla (**vincoli di tupla**)

      Interessano tutte le tuple, l'una indipendentemente dalle altre
    - di chiave (per l'**identificazione univoca** di una tupla)
- Oppure imporre vincoli che coinvolgono più relazioni:
  - Vincoli inter-relazionali
    - consentono di verificare la validità dei valori degli attributi inseriti in una relazione per correlarla ad un'altra.

Vincoli intra-relazionali

- vincoli di integrità **intra-relazionale** sono vincoli espressi attraverso condizioni logiche che devono essere soddisfatte all'interno di una singola relazione:
  - **Vincolo di dominio**.
    - Una regola che deve essere soddisfatta dai valori di un fissato attributo di una relazione
  - **Vincolo di tupla**.
    - Una condizione logica che coinvolge più attributi all'interno della stessa tupla.

Definizioni

Esempio

NomeStudente	NomeEsame	Voto	Lode
Pippo	TSI	30	SI
Paolo	TM	28	NO
Marta	EI	18	SI
Maria	AI	33	NO

$18 \leq$ Voto ≤ 30	(**Voto** ≥ 18) AND (**Voto** ≤ 30)	**dominio**
La lode è ammissibile se voto è uguale a 30	NOT (**Lode** = 'SI' AND **Voto** $\neq 30$)	**tupla**
NomeStudente non può essere NULL	NOT(**NomeStudente** = NULL)	**dominio**

Definizioni

Superchiave

Sia dato uno schema di relazione R(X) e sia SK un sottoinsieme di attributi di X.

Diciamo che SK è una superchiave di una relazione r sullo schema R(X) se vale:

$$\forall t_i, t_j \in r,\ i \neq j \rightarrow t_i[SK] \neq t_j[SK]$$

Chiave

- Un sottoinsieme K di attributi X è chiave per r se è una **superchiave minimale** di r (cioè togliendo un qualsiasi attributo da K, K non è più superchiave).

Definizioni

Esempio

Consideriamo lo schema di relazione:

STUDENTI (Matricola,Nome,Cognome,Nascita,CorsoDiStudio)

- **Superchiavi**
  - Matricola, Nome, Cognome
  - Nome, Cognome, Nascita
  - Matricola, Nome
  - Matricola
  - Matricola, Nome, Cognome, Nascita, CorsoDiStudio
- **Superchiavi minimali (chiavi)**
  - Nome, Cognome, Nascita
  - Matricola
- **Chiave primaria**
  - Matricola

Note

- in una relazione esiste sempre almeno una (super) chiave - la tupla è sicuramente una superchiave per la proprietà di unicità delle tuple;
- in generale, in una relazione è possibile individuare chiavi differenti.
- Tra tutte le possibili chiavi, si sceglie sempre una chiave detta chiave primaria della relazione.

Integrità dell'entità

Nessun valore di una chiave primaria può essere nullo

Un DBMS relazionale in presenza di una nuova tupla per r con valore NULL nel campo della chiave primaria non permette l'inserimento della tupla in r.

Esempio schema di relazione con chiave primaria

Dall'esempio precedente relativo alla relazione STUDENTI, avendo scelto Matricola come chiave primaria, abbiamo la notazione:

STUDENTI (<u>Matricola</u>, Nome, Cognome, Nascita, CorsoDiStudio)

Vincoli inter-relazionali

- In una base di dati solitamente si distribuisce l'informazione su relazioni differenti in modo da esprimere differenti concetti in differenti relazioni evitando in tal modo:
  - ridondanze nei dati
  - eventuali inconsistenze nell'aggiornamento dei dati.

- La distribuzione delle informazioni richiede un meccanismo che permetta di associare dati presenti in una tabella con quelli di un'altra tabella .

Esempio

Definizioni

STUDENTI

Matricola	Nome	Cognome	Indirizzo
150	Alex	Parisi	Via dei Palloni, 30
151	Martina	Stellina	Via del Cielo, 40
142	Giovanni	Senzaterra	Via Crociate, 30

CORSI

Codice	Corso
TSI	Tecnologia dei Sistemi Informatici
EI	Elementi di Informatica
A1	Analisi Matematica 1

CARRIERE

MatStudente	CodiceCorso	Data	Voto
150	TSI	10/10/04	30
150	A1	10/09/05	28
151	TSI	10/10/05	28


Esempio

- **MatStudente** (di **CARRIERE**) è definito sullo stesso dominio dell'attributo **Matricola** (chiave primaria di STUDENTI)
- **CodiceCorso** (di **CARRIERE**) è definito sullo stesso dominio dell'attributo **Codice** (chiave primaria di CORSI).
- **MatStudente** e **CodiceCorso sono dette chiavi esterne di CARRIERE in quanto "estranee" al concetto espresso dalla relazione CARRIERE**

- Per l'integrità dei dati:
  - ogni valore di **MatStudente** in **CARRIERE** deve essere anche presente come valore di **Matricola** in STUDENTE
  - ogni valore di **CodiceCorso** in **CARRIERE** deve essere anche presente come valore di **Codice** in CORSI

Integrità referenziale

Date r_1 ed r_2, non necessariamente differenti, con r_1 dotata di chiave esterna FK relativa alla chiave primaria PK della relazione r_2.

Si dice che tra r_1 ed r_2 sussiste un vincolo di integrità referenziale se, ogni occorrenza di FK in $t_1 \in r_1$

- è **NULL** oppure
- $\exists$ $\mathbf{t_2 \in r_2 \mid t_1[FK] = t_2[PK]}$

Definizioni

Concetto di integrità referenziale

- Per ogni occorrenza a valore non nullo della chiave esterna nella tabella *referente* deve essere presente nella tabella *riferita* un ugual valore di chiave (primaria).
- Ciò spiega perché si suole spesso dire che il modello relazionale è un *modello basato sui valori*: l'informazione distribuita su relazioni differenti, si ricava ricercando sulla base di dati la presenza di un valore *comune*

Integrità referenziale: sintassi

- Al fine di evidenziare i vincoli di integrità referenziale, si aggiunge, accanto all'elenco degli attributi che fungono da chiavi esterne, il nome della relazione riferita, cioè:

⟨ ChiaveEsterna ⟩ ::=
⟨ ElencoAttributi ⟩ **:** ⟨ NOME_TABELLARIFERITA ⟩

Definizioni

Notazione relazionale dell'esempio

- Con riferimento all'esempio precedente, lo schema relazionale della BD viene così espresso:

STUDENTI (Matricola, Nome, Cognome, Indirizzo)
CORSI(Codice, Corso)
CARRIERE(MatStudente: STUDENTI, CodiceCorso: CORSI,
Data, Voto)

- **Per completare lo schema occorre, per ogni relazione, esplicitare gli eventuali altri vincoli intra-relazionali.**

Esempio di Base di Dati Relazionale

GIOCATORI

CodTessera	Nome	Cognome	Ruolo	Età	Squadra

SQUADRE

Nome	ColoriSociali	AnnoDiFondazione	Stadio

Vincoli intra-relazionali

- IC_1(GIOCATORI) $\leftarrow$ Ruolo $\in$ { attaccante, difensore, portiere, centrocampista }
- IC_2(GIOCATORI) $\leftarrow$ Età $\leq 100 \wedge$ Età ≥ 0

- IC_1(SQUADRE) $\leftarrow$ Nome $\in$ {'Atalanta', 'Bologna', ... 'Udinese' }
- IC_2(SQUADRE) $\leftarrow$ AnnoDiFondazione $\leq 2005 \wedge$ AnnoDiFondazione ≥ 1800

Definizioni

Esempio di Base di Dati Relazionale

- Seconda Tupla NON VALIDA (valore di ruolo)

CodTessera	Nome	Cognome	Ruolo	Età	Squadra
00001	Victor	Osimhen	attaccante	24	Napoli
00002	Vincenzo	Moscato	docente	25	Napoli

- Seconda Tupla NON VALIDA (valore di età)

CodTessera	Nome	Cognome	Ruolo	Età	Squadra
00001	Victor	Osimhen	attaccante	24	Napoli
00002	Antonio	Picariello	portiere	101	Napoli

Esempio di Base di Dati Relazionale

- Seconda Tupla NON VALIDA (valore di NOME)

Nome	ColoriSociali	AnnoDiFondazione	Stadio
Napoli	Bianco-Azzurro	1926	Maradona
Südtirol	Bianco-Rosso	1974	Druso

- Seconda Tupla NON VALIDA (valore di Anno di Fondazione)

Nome	ColoriSociali	AnnoDiFondazione	Stadio
Napoli	Bianco-Azzurro	1926	Maradona
Milan	Rosso-Nero	1200	San Siro

Scelta delle chiavi

GIOCATORI

CodTessera	Nome	Cognome	Ruolo	Età	Squadra
00001	Victor	Osimhen	attaccante	24	Napoli
00002	Theo	Hernàndez	difensore	25	Milan
00003	Andrea	Petagna	attaccante	27	Monza
00004	Kim	Min-jae	difensore	26	Napoli

SQUADRE

Nome	ColoriSociali	AnnoDiFondazione	Stadio
Napoli	Bianco-Azzurro	1926	Maradona
Monza	Bianco-Rosso	1912	Brianteo
Inter	Nero-Azzzurro	1908	San Siro
Milan	Rosso-Nero	1899	San Siro

Definizioni

- **CHIAVI di GIOCATORI:**
  - **CodTessera** (chiave primaria)
  - **Nome,Cognome**
- **CHIAVE ESTERNA DI GIOCATORI**
  - **Squadra** della relazione **GIOCATORI** che si riferisce alla chiave primaria **Nome** della relazione **SQUADRE**.

Definizione dei dati in SQL

- SQL è l'acronimo di Structured Query Language
- Versioni:
  - SQL-86
  - SQL2
  - SQL-92
  - SQL3
- SQL è un linguaggio di tipo **dichiarativo**

SQL comprende sia istruzioni per la definizioni di dati (**DDL**) che per la loro manipolazione (**DML**).

CREATE TABLE

È usato per creare una nuova relazione: in esso si specificano:

- **il nome della relazione**
- **il nome ed il tipo dei suoi attributi**
- **i vincoli intra e inter-relazionali**

Definizioni

Tipi di Dato

- **Numeric**
  - numeri interi (integer, int, smallint)
  - numeri reali a precisione differente in virgola fissa e in virgola mobile, (real, float, double precision).
- **Stringhe**
  - di caratteri di lunghezza fissa (char(n))
  - di caratteri a lunghezza variabile (varchar(n))
  - di bit a
    - lunghezza fissa (bit(n))
    - variabile (bitvarying(n)).
- **Data e Ora** permette di esprimere data e ora.
  - Ha dieci posizioni aventi per componenti *YEAR*, *MONTH* e *DAY* in vari formati
  - Il tipo time ha otto posizioni con i componenti *HOUR*, *MINUTE* e *SECOND*.
  - Il tipo interval permette, invece, di stabilire un valore temporale relativo.

Vincoli

- **NOT NULL** specifica il vincolo che il valore dell'attributo deve essere diverso da NULL.
- **UNIQUE** specifica il vincolo che il valore (o i valori) dell'attributo (o degli attributi) specificato in una tupla deve essere unico (vincolo generico di chiave).
- **primary key** specifica che uno o più attributi sono chiave primaria di una relazione:
  - per default, è not NULL e unique.
- **foreign key** permette di specificare un vincolo di integrità referenziale.
  - La specifica [opzionale] delle politiche di violazione del vincolo di chiave esterna avviene attraverso le opzioni on delete set null, on delete set default, on delete cascade, on delete no action, on update cascade.

Sintassi

```
Create table nomeTabella
(
   nomeAttributo Dominio [Default][Vincoli]
   {,nomeAttributo Dominio [Default][Vincoli]}
   [,altriVincoli]
)
```

Esempio BD carriere studenti

- Notazione relazionale dello schema:
  (a meno di ulteriori vincoli intra-relazionali)

STUDENTI (Matricola, Nome, Cognome, Indirizzo)

CORSI(Codice, Corso)

CARRIERE(MatStudente: STUDENTI, CodiceCorso: CORSI, Data, Voto)

Esempio BD carriere studenti

```
Create table CORSI
(
 Codice varchar(10),
 Corso varchar(100),
 primary key (codice)
)

Create table STUDENTI
(
 Matricola integer,
 Nome varchar(50),
 Cognome varchar(50),
 Indirizzo varchar(150),
 primary key(Matricola)
)
```


Esempio BD carriere studenti

```
Create table CARRIERE
(
 MatStudente integer,
 CodiceCorso varchar(10),
 data date,
 voto integer,
 primary key(MatStudente,CodiceCorso),
 foreign key(MatStudente) references
 STUDENTI(Matricola)
 on delete CASCADE,
 foreign key(CodiceCorso) references
 CORSI(Codice)
 on delete CASCADE
)
```

Esempio BD Campionato calcio

- Notazione **completa** dello schema relazionale :

SQUADRE(Nome, ColoriSociali, AnnoDiFondazione, Stadio)

Nome $\in$ { Atalanta, ..., Udinese }

AnnoDiFondazione $\leq$ 2005 $\wedge$ AnnoDiFondazione$\geq$ 1800

GIOCATORI(CodTessera, Nome, Cognome, Ruolo, Età, Squadra:SQUADRE)

Nome $\neq$ NULL

Cognome $\neq$ NULL

Ruolo $\in$ {attaccante, difensore, portiere, centrocampista}

Età $\leq$ 100 $\wedge$ Età $\geq$ 0

Definizioni

Esempio BD Campionato calcio

```
Create table SQUADRE
(
 Nome varchar(50)
 Check(Nome='Atalanta'OR ...OR
 Nome='Udinese') ,
 ColoriSociali varchar(30),
 AnnoDiFondazione integer
 Check( AnnoDiFondazione >= 1800 AND
        AnnoDiFondazione <= 2005),
 Stadio varchar(100)
 primary key (Nome)
)
```

Esempio BD Campionato calcio

```
Create table GIOCATORI
(
 CodTessera varchar(10)
 Nome varchar(50) NOT NULL,
 Cognome varchar(50) NOT NULL,
 Ruolo varchar(15)CHECK(Ruolo='Portiere' or ...
 or Ruolo='Attaccante'),
 Eta integer  Check( Eta>0 AND Eta <= 100),
 Squadra varchar(50),
 primary key (CodTessera),
 foreign key(Squadra) references SQUADRE(Nome)
 on delete SET NULL
)
```

Gestione della sicurezza dei dati

- Uno dei compiti più importanti di un amministratore di base di dati, o *Data Base Administrator* (DBA), consiste proprio nel definire ed implementare opportune politiche di sicurezza e controllo degli accessi ed il **linguaggio SQL supporta** con apposite istruzioni dedicate l'attuazione di tali politiche.
- ad ogni utente siano associate delle apposite credenziali di accesso espresse usualmente in termini di ***username*** e ***password***. Si noti che il DBMS ha sempre predefinito almeno un utente amministratore, attraverso il quale è possibile configurare il sistema di basi di dati.

Definizioni

Creazione di un utente

- **CREATE USER** username **IDENTIFIED BY** password

- Una volta definite le credenziali di accesso, bisogna stabilire i *privilegi* che ogni utente deve avere sulle risorse associate allo schema della base di dati (in altri termini quali azioni ogni utente è autorizzato a compiere), dove per *risorsa* si intende una qualsiasi classe di oggetti (tabella, vista, schema, ecc.) che costituisce la base di dati stessa

Definizioni

Privilegi

- $P = \langle R, U_1, U_2, A, T \rangle$
- dove:
  - **R** rappresenta la risorsa su cui è concesso il privilegio;
  - **U_1** rappresenta l'utente che concede il privilegio;
  - **U_2** rappresenta l'utente che ottiene il privilegio;
  - **A** rappresenta l'insieme delle azioni che sono permesse sulla risorsa;
  - **T** rappresenta l'autorizzazione concessa all'utente che riceve il privilegio di trasmettere lo stesso privilegio ad altri utenti.

Definizioni

Principali privilegi (1)

- create: permette di definire nuove istanze per una data risorsa;
- drop: permette di rimuovere istanze di una data risorsa;
- update: permette di modificare lo stato di un oggetto o istanza di una data risorsa (nel caso di tabelle, ne permette la modifica dei valori di una o più tuple);
- insert: è applicabile solo a risorse di tipo tabella e vista e permette di modificarne lo stato aggiungendo nuove tuple al loro interno;
- delete: è applicabile solo a risorse di tipo tabelle e vista e permette di modificarne lo stato rimuovendo tuple da esse;

Principali Privilegi (2)

- select: permette di leggere lo stato di un oggetto o istanza di una data risorsa
- resource: permette di creare, modificare e rimuovere risorse da uno schema; implica quindi, anche i permessi discussi precedentemente;
- connect: premette ad un dato utente di connettersi al DBMS per potere accedere alle varie risorse;
- all privileges: permette ad un utente di ottenere tutti i privilegi possibili sulle risorse di una base di dati.

Definizioni

GRANT e REVOKE

- **GRANT** privilegio1,...,privilegioN **ON** NomeRisorsa **TO** username
- **[WITH GRANT OPTION]**

- **REVOKE** privilegio1,...,privilegioN **ON** NomeRisorsa **FROM** username [⟨ **RESTRICT** | **CASCADE** ⟩]

Definizioni

Definizioni

Ruoli

- Spesso quando più utenti devono condividere gli stessi privilegi, è possibile creare un apposito *ruolo* o *profilo*. Dopo avere associato a quest'ultimo un insieme di privilegi, tutti gli utenti con quel ruolo erediteranno in maniera automatica i relativi singoli privilegi connessi.

- **CREATE ROLE** nomeruolo
- **GRANT** privilegio1,...,privilegioN **ON** NomeRisorsa1 **TO** nomeruolo [**WITH GRANT OPTION**];
  ....,
  GRANT privilegio1,...,privilegioN **ON** NomeRisorsaN **TO** nomeruolo [**WITH GRANT OPTION**];
- **GRANT** nomeruolo **TO** username
-

Esempio

- **CREATE ROLE** giornalista;
  GRANT connect, update **ON** CAMPIONATO.* **TO** giornalista;

- **CREATE ROLE** tifoso;
  GRANT connect, select **ON** CAMPIONATO.GIOCATORI **TO** tifoso;

- **CREATE USER** paola **IDENTIFIED BY** delgeni;
- **CREATE USER** carla **IDENTIFIED BY** aldino; **CREATE USER** vinni **IDENTIFIED BY** moscato; **CREATE USER** antonio **IDENTIFIED BY** picariello;
- **GRANT** giornalista **TO** paola; **GRANT** gornalista **to** carla; **GRANT** tifoso **TO** vinni; **GRANT** tifoso **TO** antonio