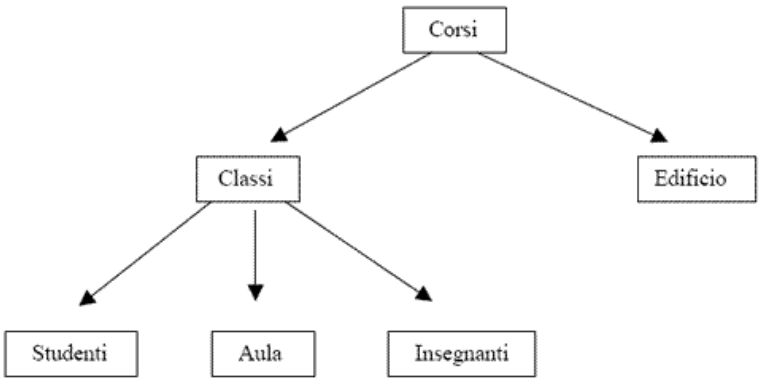
BASI DI DATI - : I modelli di database

DAL 1960 ci si e' orientati verso 3 direzioni:

1 MODELLO GERARCHICO

Se i dati si presentano naturalmente in una struttura ad albero (ES. File System)

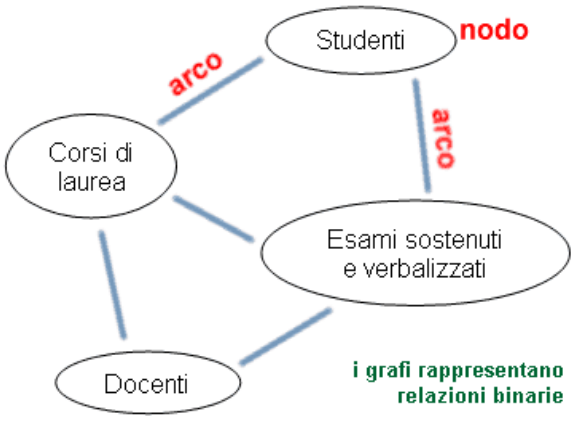
Limiti: rigidità della struttura dati



2 MODELLO RETICOLARE

Se i dati si presentano naturalmente come un grafo orientato – diffuso negli anni 1970

Limiti: complessità e difficoltà nell'implementare la struttura dati



3 MODELLO RELAZIONALE

Proposto nel 1970 dal Edward **F. Codd** (ricercatore IBM) – Quello ormai universalmente diffuso

I DATI SONO UN INSIEME DI RELAZIONI (**RELAZIONE=TABELLA**)

Si basa su concetti fondamentali matematici, utilizza un linguaggio rigoroso, elimina ambiguità nella terminologia e nella simbologia!
Praticamente è il modello quasi universalmente diffuso (gerarchico e reticolare--> superati)

studente

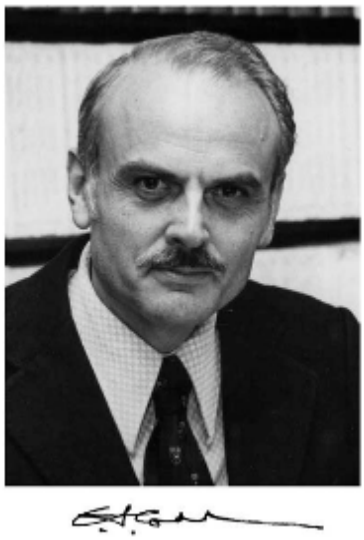
MATR	NOME	CITTA'	C-DIP
123	Carlo	Bologna	Inf
415	Paola	Torino	Inf
702	Antonio	Roma	Log

esame

MATR	COD-CORSO	DATA	VOTO
123	1	7-9-97	30
123	2	8-1-98	28
702	2	7-9-97	20

corso

COD-CORSO	TITOLO	DOCENTE
1	matematica	Barozzi
2	informatica	Meo



Approfondimento (Da wikipedia)
Edgar F. Codd nacque a Portland su un'isola nella contea del Dorset in Inghilterra e studiò matematica e chimicaall'Exeter College dell'Università di Oxford. Nel 1948 si trasferì a New York per lavorare per l'IBM come programmatore matematico. Nel 1953, in dissenso con le politiche portate avanti dal senatore Joseph McCarthy, si trasferì ad Ottawa. Codd ha ricevuto un Turing Award nel 1981.

Edgar F. Codd morì in Florida, all'età di 79 anni, il 18 aprile 2003.

Attività: Negli anni sessanta e settanta, mentre lavorava per l'IBM, creò il modello relazionale per la gestione delle basi di dati, pubblicando "*A Relational Model of Data for Large Shared Data Banks*" ("Un modello relazionale di dati per gestire grandi banche dati condivise")[1] nel 1970.

Con suo grande disappunto, l'IBM tardò a sfruttare i suoi suggerimenti fino a quando la concorrenza cominciò ad implementarli. Ad esempio, Larry Ellison costruì il database Oracle sulla base delle idee di Codd.

Fu autore, quindi, di considerevoli contributi all'informatica, ma il modello relazionale, una teoria generale della gestione dei dati che ebbe un enorme seguito, rimane il suo conseguimento più memorabile.

Database orientati agli oggetti (OODB)
Di recente diffusione, avvicinano i dati alle applicazioni / concetti tipici della programmazione ad oggetti (classi / sottoclassi / metodi / ereditarietà / etc.)
Diffusi solo in applicazioni particolari : multimedialità / medicina / gestione di documenti / ingegneria meccanica.

In una base di dati relazionale:

```
CREATE TABLE Customers  (
    Id          CHAR(12)    NOT NULL PRIMARY KEY,
    Surname     VARCHAR(32) NOT NULL,
    FirstName   VARCHAR(32) NOT NULL,
    DOB         DATE        NOT NULL
);
SELECT InitCap(Surname) || ', ' || InitCap(FirstName)
  FROM Customers
 WHERE Month(DOB) = Month(getdate())
   AND Day(DOB) = Day(getdate())
```

In una base di dati relazionale ad oggetti:

```
CREATE TABLE Customers (
  Id            Cust_Id     NOT NULL  PRIMARY KEY,
  Name          PersonName  NOT NULL,
  DOB           DATE        NOT NULL
);
SELECT Formal( C.Id )
  FROM Customers C
 WHERE BirthDay ( C.DOB ) = TODAY;
```

Cosa si intende con DATABASE:

DATABASE

Con il termine basi di dati (**database)** si indicano in informatica gli archivi di dati, organizzati attraverso tecniche di modellazione dei dati e gestiti sulle memorie di massa dei computer attraverso appositi software, con l'obiettivo di raggiungere una grande efficienza nel trattamento e nel ritrovamento dei dati.

È la base di lavoro per tutti gli utenti diversi con programmi diversi.

Essi hanno le seguenti caratteristiche:

- *efficienza e produttività*, cioè la possibilità di ritrovare facilmente le informazioni desiderate;

- *consistenza*, cioè i dati in essi contenuti devono essere significativi ed essere effettivamente utilizzabili nelle applicazioni dell'azienda;

- *sicurezza*, cioè impedire che i database vengano danneggiati da interventi accidentali o non autorizzati;

- *integrità*, cioè garantire che le operazioni effettuate non provochino una perdita di consistenza ai dati.

I prodotti software per la gestione di database vengono indicati con il termine **DBMS** (DataBase Management System).

I *difetti* che si hanno nella vecchia organizzazione in archivi sono:

- *ridondanza*, cioè i dati compaiono in maniera duplicata

- *incongruenza*, cioè i dati vengono aggiornati in un archivio e non in un altro

- *inconsistenza*, cioè i dati a disposizione non sono più affidabili

Di conseguenza, vengono individuati i *limiti* dell'organizzazione in archivi:
- i programmi sono legati agli archivi che gestiscono, ogni cambiamento muta entrambi
- ridondanza dei dati
- l'organizzazione vincola il programmatore nell'uso del pc

Con la teoria dei Database, invece, sono stati superati questi limiti attraverso:
- *l'indipendenza della struttura fisica dei dati*, cioè modificare i supporti senza modifiche alle applicazioni

- *l'indipendenza della struttura logica dei dati,* cioè modificare le strutture dei database senza modificare il software applicativo

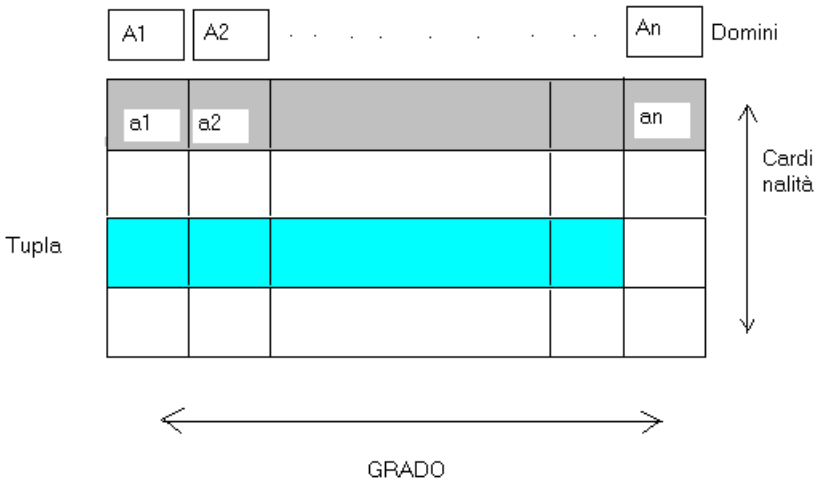
- *utilizzo da parte di più utenti*
- *eliminazione della ridondanza e inconsistenza*
- *facilità di accesso*

- *sicurezza e integrità dei dati*, cioè procedure di controllo previste per impedire accessi non autorizzati
- *uso di linguaggi per la gestione del database*, attraverso comandi per la manipolazione dei dati contenuti in esso e comandi per effettuare interrogazioni (SQL)

DATABASE:

MODELLO RELAZIONALE

Si chiama RELAZIONALE perchè è basato sul concetto matematico di *RELAZIONE tra INSIEMI di oggetti.*



RELAZIONE (definizione rigorosa):

Dal punto di vista matematico, dati n insiemi A1, A2, ..An, ***si dice relazione*** *un sottoinsieme dell'insieme di tutte le n-uple a1, a2,.. an che si possono costruire prendendo nell'ordine un elemento a1 dal primo insieme A1, a2 dal secondo insieme A2, e cosi' via.*

Definizioni di ***Grado, Cardinalità, domini, tuple:***

- ***Grado della relazione: e' il numero n (colonne della tabella)***
- ***Cardinalità della relazione: il numero delle n-uple (n-upla = riga della tabella)***
- ***Domini della relazione: Gli insiemi A1, A2, .. An***
- ***Tupla: riga della relazione (riga della tabella)***

I nomi A1, A2, An delle colonne sono i nomi dei ***CAMPI*** *della tabella.*

I valori che compaiono in una colonna sono ***OMOGENEI*** *fra loro, cioe' appartengono a uno stesso DOMINIO.*

La ***relazione*** *rappresenta una ENTITA' (del modello E/R).*

Ogni ***n-upla*** *(o tupla) rappresenta una ISTANZA dell'entità.*

Il ***DOMINIO*** *e' l'insieme dei valori che possono essere assunti da un attributo (campo).*

La ***CHIAVE*** *della relazione e' un attributo o una combinazione di attributi che identificano UNIVOCAMENTE le n-uple (righe della tabella).*

IMPORTANTISSIMO : (requisiti fondamentali del modello relazionale):

1) tutte le RIGHE contengono lo stesso numero di colonne
2) gli attributi rappresentano le informazioni elementari NON scomponibili ulteriormente: un solo valore
3) i valori assunti da un CAMPO sono OMOGENEI (dello stesso tipo)
4) IN UNA RELAZIONE OGNI RIGA E' DIVERSA da tutte l altre (non ci sono righe doppie: la chiave primaria distingue una riga dall'altra)
5) LE n-uple (righe) compaiono nella tabella secondo un ordine NON prefissato

La relazione come tabella

Acquisto

NumFattura	Fornitore	CodProd	Reparto	Quantità

Ogni riga rappresenta una n-tupla della relazione R

L'ordine delle righe non è importante

Tutte le righe sono diverse tra loro

L'ordine delle colonne è significativo e corrisponde all'ordine dei domini A_1, A_2, ..., A_n sui quali la relazione R è definita

Ogni colonna è rappresentata con un titolo che è il nome del dominio corrispondente.

[Codd, 1970 informatico britannico, fondatore della teoria delle basi di dati relazionali.]

REGOLA DI INTEGRITA' DEL MODELLO RELAZIONALE: ***non ci sono NULL nel campo CHIAVE!***

RISCRITTURA REGOLE DEL MODELLO E/R (per il modello relazionale):

1) OGNI ENTITA' DIVENTA UNA RELAZIONE
2) OGNI ATTRIBUTO DI UNA ENTITA' DIVENTA UN ATTRIBUTO DELLA RELAZIONE (TABELLA) CIOE' IL NOME DI UNA COLONNA
3) OGNI ATTRIBUTO DELLA RELAZIONE EREDITA' LE CARATTERISTICHE DELL'ATTRIBUTO DELL'ENTITA' DA CUI DERIVA (ovviamente!)
4) L'IDENTIFICATORE UNIVOCO DI UNA ENTITA' DIVENTA LA CHIAVE PRIMARIA DELLA RELAZIONE
5) L'ASSOCIAZIONE 1 a 1 DIVENTA UN UNICA RELAZIONE
6) *L'IDENTIFICATORE UNIVOCO DELL'ENTITA' DI PARTENZA (ASSOCIAZIONI 1 a N) DIVENTA LA CHIAVE ESTERNA DELL'ENTITA' DI ARRIVO ASSOCIATA (quella dalla parte N)*
7) L'ASSOCIAZIONE N a N DIVENTA UNA NUOVA RELAZIONE COMPOSTA ALMENO DAGLI IDENTIFICATORI UNIVOCI DELLE DUE ENTITA' DI PARTENZA.

Oppure con linguaggio leggermente diverso:

- **ogni *entità* diventa una tabella**
- **ogni *attributo* di un'entità diventa un attributo della relazione (nome di una colonna della tabella)**
- **ogni colonna della relazione eredita le caratteristiche dell'attributo dell'entità da cui deriva**
- **l'identificatore univoco di un'entità diventa la *chiave primaria* della relazione derivata**

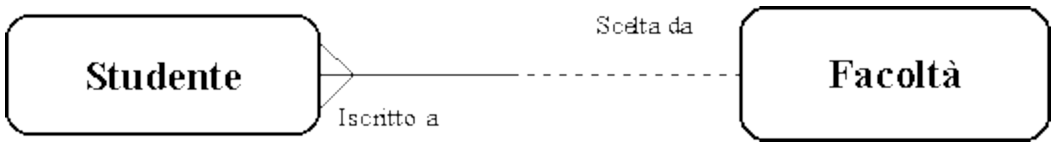
Associazioni

- l’associazione *uno a uno* diventa un’unica relazione che contiene gli attributi della prima e della seconda entità
- l'identificatore univoco dell’entità di partenza nell’associazione *uno a molti* diventa **chiave esterna** (*foreign key*) dell'entità di arrivo associata, cioè i suoi attributi - identificatori univoci - diventano colonne della seconda relazione
- l'associazione *molti a molti* diventa una nuova relazione (in aggiunta alle relazioni derivate dalle entità) composta dagli identificatori univoci delle due entità e dagli eventuali attributi dell’associazione.

ESEMPI :

Regole di derivazione: esempio 2

Facoltà e Studenti (*associazione uno a molti*)



Ogni Studente *deve essere* iscritto a *una sola* Facoltà.

Ogni Facoltà *può essere* scelta da *uno o più* Studenti.

Facoltà

CodFac	Descrizione	Città

Studente

Matricola	Cognome	Nome	Indirizzo	*CodFac*

Facoltà (CodFac, Descrizione, Città)

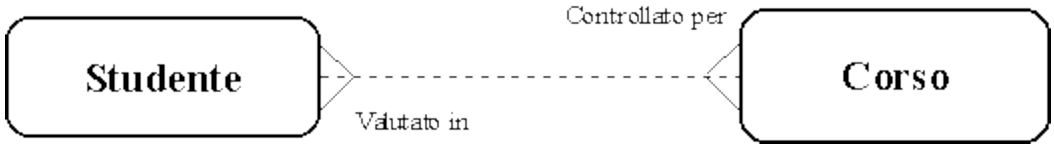
Studente (Matricola, Cognome, Nome, Indirizzo, *CodFac*)

Nota:

In generale dal modello concettuale vengono derivate le tabelle che rappresentano le entità; l'associazione *uno a molti* viene tradotta aggiungendo agli attributi dell'entità a molti la chiave dell'entità a uno (**chiave esterna**).

Regole di derivazione: esempio 3

Studenti e Corsi (*associazione molti a molti*)



Ogni Studente *può essere* valutato in *uno o più* Corsi.

Ogni Corso *può essere* controllato per *uno o più* Studenti.

Studente

Matricola	Cognome	Nome	Indirizzo

Corso

CodCorso	Descrizione	Anno

Prova d'esame

Matricola	CodCorso	Data	Voto

Studente (Matricola, Cognome, Nome, Indirizzo)

Corso(CodCorso, Descrizione, Anno)

Prova d'esame (*Matricola, CodCorso*, Data, Voto)

In generale, vengono derivate le relazioni corrispondenti alle entità; l'associazione *molti e molti* viene tradotta con una terza relazione contenente le chiavi delle due entità e gli eventuali attributi dell'associazione.

NB: ***(sottolineato = chiave primaria, corsivo = chiave esterna)***

Esercizio: Associazione sportiva con iscritti, discipline sportive

Dopo aver costruito il modello E/R, definire le tabelle utilizzando le regole di derivazione

SOLUZIONE:

- **Entità** :

Disciplina, Iscritto

- **Associazione** :

uno a molti

- **Tabelle** :

Disciplina (CodDiscipl, Descrizione)

Iscritto (CodIscritto, Cognome, Nome, Indirizzo, *CodDiscipl*)

Esercizio: Catalogo di prodotti informatici con modelli, produttori, rivenditori

Dopo aver costruito il modello E/R, definire le tabelle utilizzando le regole di derivazione

SOLUZIONE:

- **Entità** :

Produttore, Rivenditore, Modello

- **Associazioni** :

uno a molti tra Produttore e Rivenditore

uno a molti tra Produttore e Modello

- **Tabelle** :

Produttore (CodProd, NomeProd, IndirProd)

Rivenditore (CodRiven, RagioneSociale, Telefono, *CodProd*)

Modello (CodMod, Descrizione, *CodProd)*