Progettazione di basi di dati

Monday, 27 March 2023 22:02

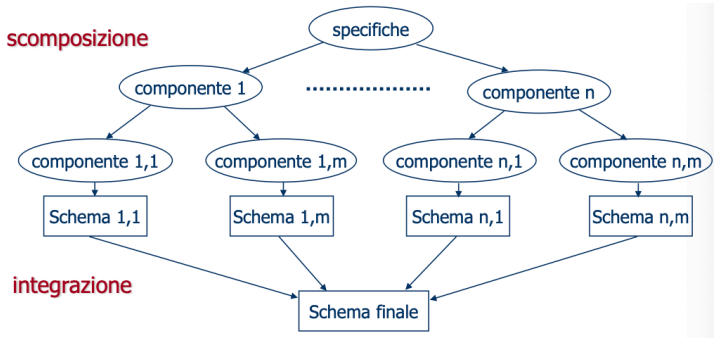
- Si inizia studiando il ciclo della vita:
  - Studio fattibilità
    Noi qui dobbiamo definire costi e priorità
  - Raccolta dei requisiti
    Si studiano le proprietà del sistema
    Questo è diviso in vari step:
    - Raccolta dei requisiti
      - Possiamo prenderlo da
        Utenti, documentazioni o simili
    - Analisi dei requisiti
      - Comprendere attività

      Le attività di sopra si susseguono e raffinano tra di loro
      Ogni raffinamento aggiunge dettagli, oppure corregge/modifica
    - Requisiti puliti
      - No argomenti troppo generici/specifici
      - Standardizzare struttura frasi → Stesso stile
      - Le frasi devono essere semplici
      - Usare posto per gli insegnanti, luogo per studenti
        Qui quindi è consigliato usare un glossario dei termini
        Termine → Descrizione → Sinonimi → Collegati
        Per aiutarsi è possibile costruire un glossario

Termine	Descrizione	Sinonimi	Collegamenti
Partecipante	Persona che partecipa ai corsi	Studente	Corso, Società
Docente	Docente dei corsi. Può essere esterno	Insegnante	Corso
Corso	Corso organizzato dalla società. Può avere più edizioni.	Seminario	Docente
Società	Ente presso cui i partecipanti lavorano o hanno lavorato	Posti	Partecipante

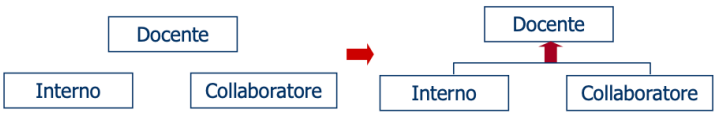
    - Traduzione schema ER
      Per la traduzione usare strategie:
      - Bottom-Up
        - Si sviluppano tanti piccoli semplici schemi parziali che poi uniamo
        - Ripartizione attività

- Integrazione
  Quindi, noi avremo un qualcosa del genere:



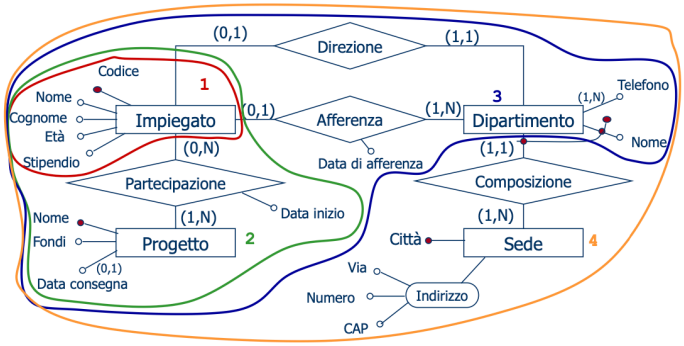
- Step:
  1. Creare una entità relativa a una classe di oggetti con pro
  2. Trovare legame logico fra 2 entità

  3. Cercare di generalizzare



  4. Cercare di aggregare gli attributi in una entità
  5. Cercare di aggregare gli attributi in una relazione

□ Inside up
- Iniziamo dal concetto core, e poi espandiamo questo core (me
- No passi integrazione
- Bisogna sempre controllare per concetti non implementati

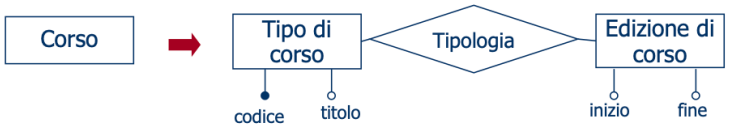


□ Hybrid
- Suddivisione dei requisiti in componenti separati
- Definizione di uno schema scheletro
  - ◇ Si individuano i concetti più importanti
  - ◇ Organizzano in uno schema concettuale
  - ◇ Si concentra sugli aspetti essenziali poi

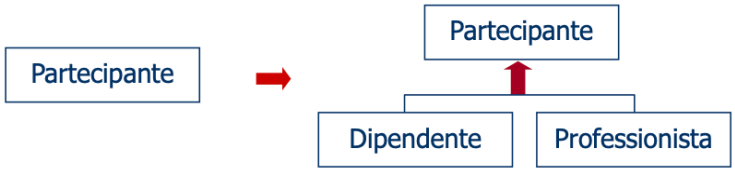
□ Top-Down
- Si parte da una schema iniziale astratto, completo, e lo raffinia
- Bisogna avere sin dall'inizio una visione globale
  Quindi è complesso quando abbiamo problemi complicati

Si seguono i seguenti step:

1. Una entità descrive 2 concetti diversi legati logicamente

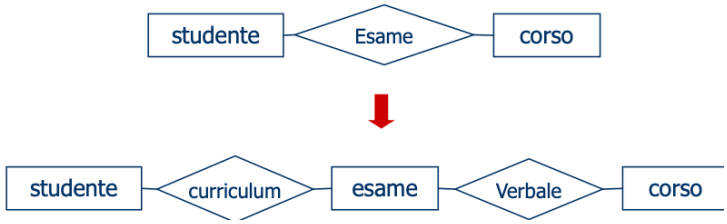


2. Cercare di trovare sotto-entità di entità



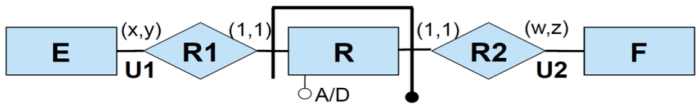
3. Trasformare la relazione in 2 relazioni fra le due entità (

4. Trasformare la relazione in entità

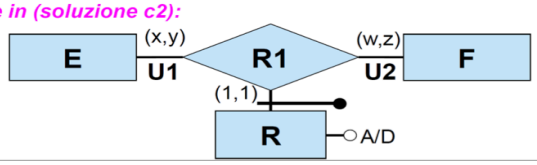


E nota che questo si può trasformare in 2 modi:

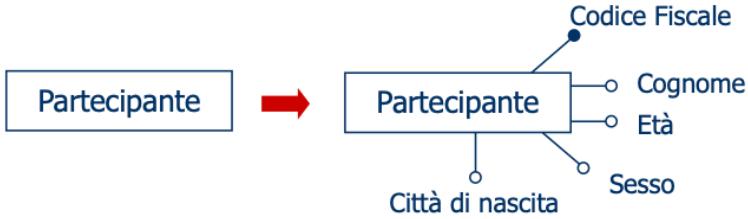
Si trasforma in (soluzione c1):



Oppure in (soluzione c2):

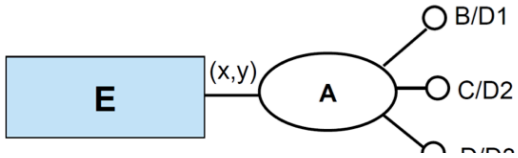


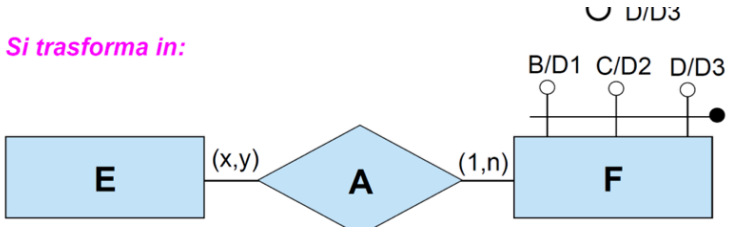
5. Aggiungere gli attributi alle entità



6. Aggiungere attributi alle relazioni

7. Trasformare attributo composto in entità chiave compo

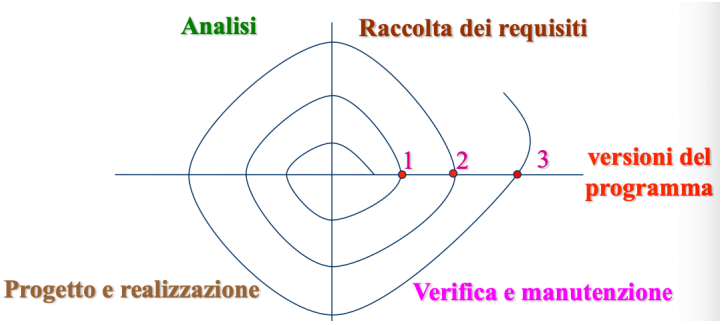




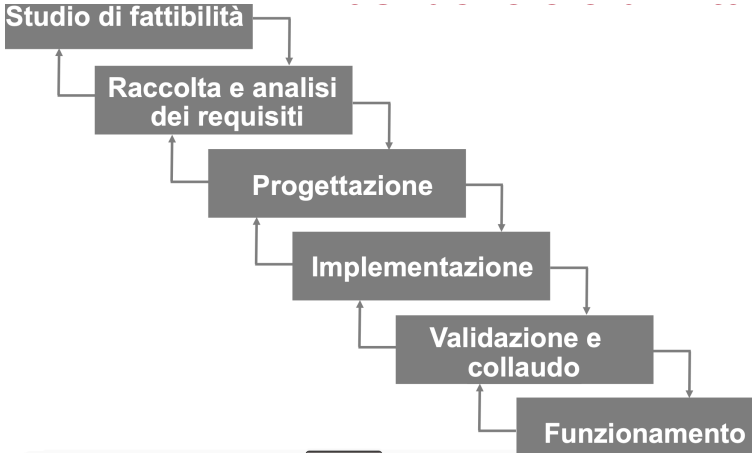
C'hai capito qualcosa? Nemmeno io

- Progettazione di dati e funzioni
  Si divide in:
  - Progettazione dei dati, organizzazione struttura della base di dati
    Si fa prima la progettazione dei dati, dopo il software
  - Progettazione delle applicazioni, si progetta software
- Implementazione
  Inizia la realizzazione
- Validazione e collaudo
  Testing
- Funzionamento
  Public release

Essi si possono eseguire attraverso un modello a spirale



Oppure in maniera sequenziale ciclica



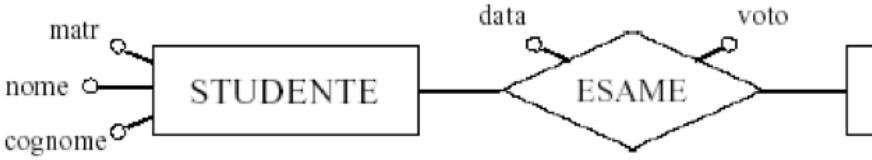
- Ogni progetto deve avere una metodologia, aka una serie di passi che guida un attività
  insieme di strumenti deve permettere:
  - Suddividere progettazioni in fasi
  - Fornire strategie da seguire
  - Fornire linguaggi per descrivere

  E deve garantire:
  - Generalità

- Generalità
- Qualità in termini di correttezza, completezza ed efficienza
- Facilità d'uso

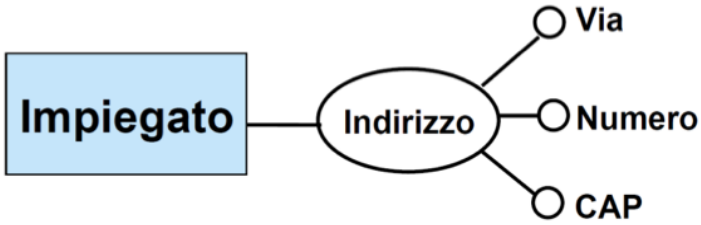
(Nota che, dopo, ognuna di queste fasi ha un proprio schema)
Nella basi di dati noi ne abbiamo una che divide:

- Cosa rappresentare
  - Progettazione concettuale -> Schema concettuale (Modello E-R)
    Traduce requisiti del sistema in una descrizione delle esigenze aziendali
    - Linguaggio grafico semi-formale per la rappresentazione di schemi c
    - Abbiamo diversi costrutti:
      - Entità
        - Classi di oggetti dell'appliciazione
          Es. Impiegato, città, studente, conto corrente
        - Si rappresenta con un quadrato
          (Sotto sotto in "Attributo semplice", i quadrati sono dell
        - Ogni entità ha un nome univoco che deve essere singola
        - Citta="Milano" => istanza, quindi istanza è un insieme d
        - Creare entità se le istanze sono concettualmente indipe
          Aka descrive classi di oggetti con esistenza autonoma
        - Se è importante per l'applicazione/Proprietà significativ
      - Relazione
        - Fatto che descrive un'azione/situazione che crea legami
        - Esistono legami con più entità
        - Quando nei requisiti compare un concetto che associa 2
        - Rappresentati con un rombo (Sotto, ESAME) è una relaz
        - Usare singolare e sostantivi anziché verbi
        - Le entità possono essere collegate tra di loro da più rela
        - Le entità possono essere in relazione con se stesse = Ass
      - Attributo semplice
        - Descrizione di un'entità
        - Anch'esso ha un nome
        - Possono essere messi sia in entità che relazioni e rappre



        - Ogni attributo può avere 1 ed 1 solo valore
        - Scegliere se non ha senso considerare una sua istanza in
        - Se serve solo a rappresentare una proprietà locale di un
      - Attributo composto

- E' un raggruppamento di attributi
- Rappresentato con un cerchio obeso, no aspetta! Non o



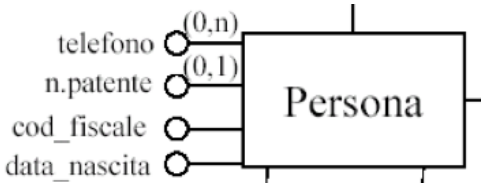
- Cardinalità
  - Associa a ogni entità che partecipa a una relazione
  - Specificano min e massimo di occorrenze delle relazioni
  - Abbiamo 3 simboli:
    - 0-1 minima (0 opzionale, 1 obbligatorio)
    - 1, N per massimo (N non può avere limite) [Si può

    E da questo nascono:
    - Uno a uno (1, 1) (0, 1)
    - Uno a molto (1, N) (0, N)
    - Molto a molti (N, M)

- min-card(Automobile,Proprietario) = 0: esistono automobili non possedute da alcuna persona
- min-card(Persona,Proprietario) = 0: esistono persone che non posseggono alcuna automobile
- max-card(Persona,Proprietario) = n: ogni persona può essere proprietaria di un numero arbitrario di automobili
- max-card(Automobile,Proprietario) = 1: ogni automobile può avere al più un proprietario

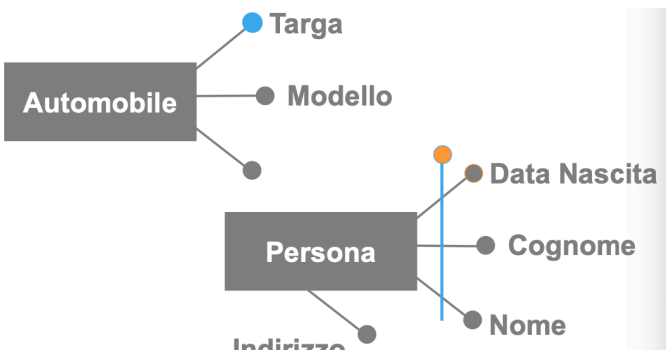
- Cardinalità di un attributo

  Anche gli attributi possono avere cardinalità!

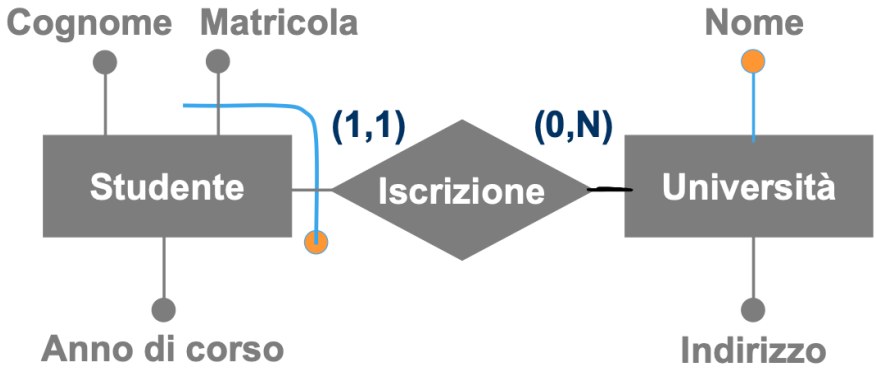


- Identificatore interno
  - Identificare univocamente occorrenza di una identità, se
  - Possiamo avere raggruppamenti di attributi per fare 1 id

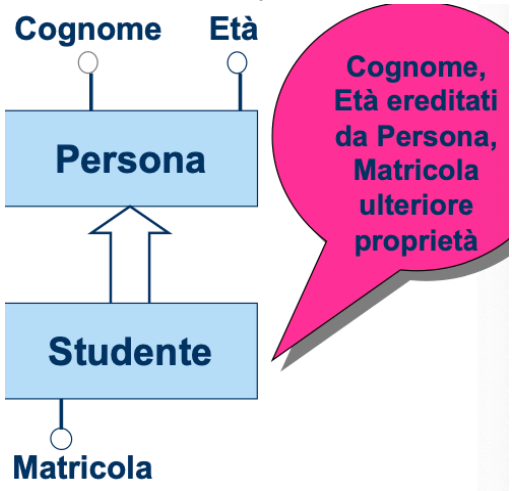
    In questo caso si identifica con una linea



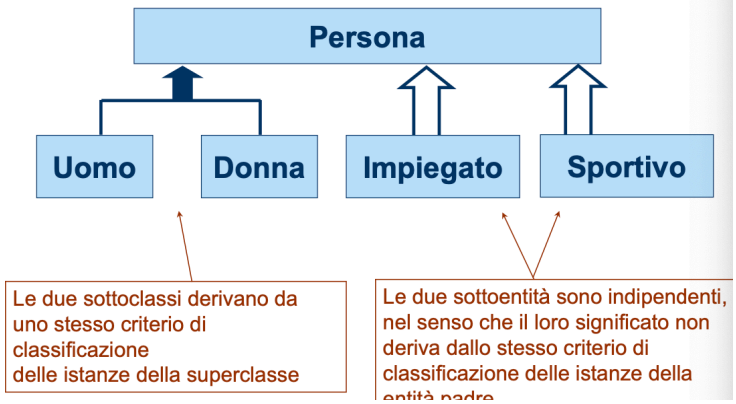
- Identificatore esterno
  - Quando ci serve un'entità esterna per identificare la nos
    Questo significa che abbiamo un grado di dipendenza



- Generalizzazione
  - Si chiama relazione ISA, e questo quando
    Una entità è un istanza di un'altra
    Aka extends, e qui si eredita



  - Si ereditano anche le relazioni
  - La freccia è nera se la generalizzazione è completa
    E' bianca se non lo è
  - Non esiste ereditarietà multipla



  - A sinistra abbiamo is-a, a destra generalizzazione
  - Uno o più concetti risultano un caso particolare di un alt
    Es. Collaboratore, Interno → Docente
    Professionista, dipendente → Partecipante

Professionista, dipendente → Partecipante

- Sottoinsieme

□ Esistono le documentazioni

Che non sono altro che tabelle di essi

Entità:

Dizionario dei dati: entità

Entità	Descrizione	Attributi	Identificatori
Impiegato	Dipendente dell'azienda	Codice Cognome Stipendio Anzianità	{ Codice }
Progetto	Progetti aziendali	Nome Budget	{ Nome }
Dipartimento	Struttura aziendale	Nome Telefono	{ Nome, Sede }
Sede	Sede dell'azienda	Città Indirizzo (Via, CAP)	{ Città, Indirizzo }

Relazioni:

Relazione	Descrizione	Componenti	Attributi
Direzione	Direzione di un dipartimento	Impiegato, Dipartimento	
Afferenza	Afferenza ad un dipartimento	Impiegato, Dipartimento	Data
Partecipazione	Partecipazione ad un progetto	Impiegato, Progetto	
Composizione	Composizione dell'Azienda	Dipartimento, Sede	

Attributi:

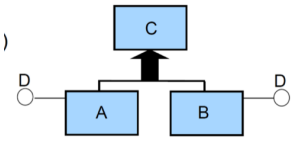
Attributo	Entità/Relazione	Dominio	Descrizione
Codice	Impiegato	Intero	Codice identificativo di impiegati
Cognome	Impiegato	Stringa	Cognome di impiegato
Stipendio	Impiegato	Reale	Stipendio di impiegato
Nome	Progetto	Stringa	Nome del progetto
...	...	...	...

Vicoli esterni:

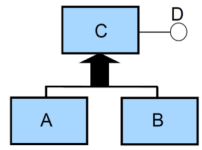
Vincoli di integrità esterni
(1) Il direttore di un dipartimento deve afferire a tale dipartimento da almeno 5 anni
(2) Un impiegato non deve avere uno stipendio maggiore del direttore del dipartimento al quale afferisce
(3) Un dipartimento con sede a Roma deve essere diretto da un impiegato con più di dieci anni di anzianità
(4) Un impiegato non può partecipare ad un numero di progetti maggiore di due volte il numero di dipartimenti ai quali afferisce

- Come farlo
  - Progettazione logica -> Modello logico (Modello relazionale)
    Tradurre lo schema concettuale in un modello dei dati del DBMS
    -Giuro che un giorno scriverò DBSM anziché DBMS senza accorgermene-
    E questa si articola in 2 fasi:
    - Ristrutturazione schema concettuale
    - Traduzione verso modello logico

    In più questo permette una descrizione dei dati indipendenti
    Ed è utile per la documentazione
  - Progettazione fisica
    Completa lo schema logico con le specifiche hw/sw
- Schema concettuale deve essere:
  - Corretto
    Deve avere una semantica e sintassi come cristo (il prof) comanda
  - Completo
    Tutti i dati devono essere presenti
    E da ogni dato è possibile arriva da qualcun altro
  - Leggibile
    Deve essere auto esplicativo
  - Minimale
    No ridondanze, e per ridondanze si intendono anche
    Concetti che è possibile reperire da altre parti
    Es.



⇒



.