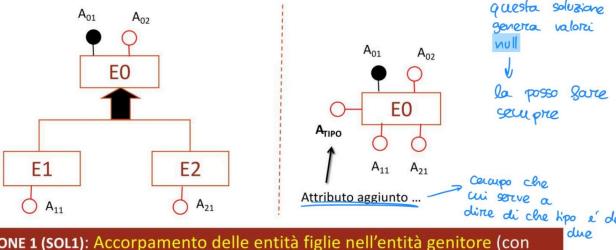
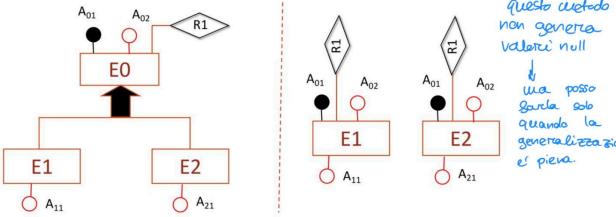


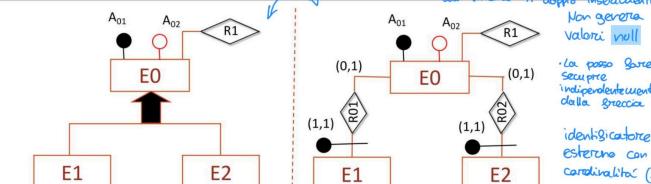
## Progettazione logica



## Progettazione logica



## Progettazione logica



00:03 12\_Progettazione\_logica (1)Mod.pdf IT-IT Basi do Dati

15/55

**Progettazione logica**

Gli attributi multivaleure non sono presenti nel modello logico, ma possono essere modellati anche con una relazione uno-a-molti ...

BASI DI DATI

PROF. MARCO DI FELICE – CORSO DI LAUREA IN INFORMATICA PER IL MANAGEMENT

00:03 12\_Progettazione\_logica (1)Mod.pdf IT-IT Basi do Dati

19/55

**Progettazione logica**

Gli accorpamenti di entità riguardano relazioni uno-ad-uno...

BASI DI DATI

PROF. MARCO DI FELICE – CORSO DI LAUREA IN INFORMATICA PER IL MANAGEMENT

## Formula Costo Operazione

$$c(O_T) = f(O_T) \cdot w_T \cdot (\alpha \cdot NC_{\text{write}} + NC_{\text{read}})$$

utilizzo per confrontare

Parte statica

$f(O_T) \rightarrow$  Frequenza operazione

$w_T \rightarrow$  Peso operazione

↳  $w_i \rightarrow$  Peso operazione Interattiva

↳  $w_B \rightarrow$  Peso operazione Batch

$\alpha \rightarrow$  coefficiente moltiplicativo

Parte statica

$NC_{\text{write}}$  e  $NC_{\text{read}}$  → numero operazioni scrittura/lettura

rispetto al  
diagramma E-R

93. Contare 10 Docenti di una Scuola → 10 letture

95. Ricucovene un ordine del giorno e i suoi 5 argomenti →  $1 + 5 = 6$  scritture

95. Creare un nuovo argomento ed aggiungerlo ad un ordine del giorno (già esistente) →  $1 + 1 = 2$  scritture

↑  
Verifica  
Ordine del  
Giorno

↓  
Scrittura  
nuovo Argomento

### Operazioni

- Creare
  - Verifica
  - Ricucovene
  - Creare e Aggiungere
  - Aggiungere
  - Contare → Op. lettura
- ] Op. Scrittura

Nota → La tabella dei volumi  
influisce su  $NC_w, NC_r$   
se non specificato  
altrimenti

## Analisi memoria

↳ Varia in base alla tabella dei volumi

↳ L'oggetto di un campo di tipo INT → 4 Byte + elemento della tabella

20 Programmi di Dottorato aggiungono un campo #nudocenti (int)

$$20 \cdot 4 \text{ Byte} = 80 \text{ Byte}$$

~~80~~

In generale sotto: 100MB non  
è un problema

## Dipendenze Funzionali (DF)

↳ In termini semplici: una DF stabilisce un vincolo tra due insiemi di attributi in una tabella.

### Definizione Formale

↳ Si dice che esiste una dipendenza funzionale tra un insieme di attributi  $Y$  e un insieme  $Z$  (scritta come  $Y \rightarrow Z$ ) se, in ogni possibile istanza della tabella, a uguali valori di  $Y$  corrispondono necessariamente uguali valori di  $Z$ .

es.

In una tabella degli impiegati, se ogni persona ha un unico stipendio, dicono che l'attributo Impiegato determina funzionalmente lo Stipendio ( $\text{Impiegato} \rightarrow \text{Stipendio}$ )

Allo stesso modo, se ogni progetto ha un'unica sede, esiste la dipendenza Progetto  $\rightarrow$  Sede

### Caratteristiche Fondamentali:

↳ Livello di schema → Le DF devono essere definite a livello di schema (basandosi sulle regole del mondo reale) e non semplicemente osservando un'istanza temporanea di dati, poiché una coincidenza in una tabella specifica potrebbe non valere in generale.

↳ Direzionalità → Le DF hanno sempre un ordine specifico, ad esempio, il Corso può Determinare il Docente, ma il Docente non determina necessariamente un unico Corso se ne insegna più di uno.

↳ Relazione con le Chiavi → Il concetto di DF è una generalizzazione di quello di superchiave. Infatti, se un insieme di attributi  $K$  è una superchiave, allora  $K$  determina funzionalmente tutti gli attributi della tabella.

### Dipendenze "Buone" vs "Cattive"

↳ Dipendenze "Buone" → Sono quelle in cui la parte sinistra della dipendenza è una superchiave. Queste non causano ridondanze.

↳ Dipendenze "Cattive" → Si verificano quando la parte sinistra non contiene una superchiave. Queste dipendenze generano ridondanze fisiche e anomalie di aggiornamento e cancellazione.

### Ruolo della Normalizzazione

↳ L'identificazione delle dipendenze funzionali è il passo necessario per la normalizzazione

↳ Se una tabella non rispetta la Forma Normale di Boyce-Codd (FNBC) (ovvero presenta dipendenze dove la parte sinistra non è superchiave), deve essere decomposta in tabelle più piccole per eliminare le anomalie.

↳ Esistono algoritmi specifici, come il calcolo della chiusura di un insieme di attributi ( $X^+$ ), per verificare quali attributi sono determinati da altri e per individuare le chiavi del sistema.

## Superchiave

Nel modello relazionale una superchiave è definita come un insieme di attributi di una relazione che consente di identificare in univoca maniera univoca le sue tuple (ovvero le righe della tabella).

↳ Definizione formale: Un sottoinsieme di attributi  $K$  è una superchiave se la relazione non contiene due tuple distinte  $t_1$  e  $t_2$  che abbiano gli stessi valori per quegli attributi ( $t_1[K] = t_2[K]$ )

↳ Relazione con le dipendenze funzionali:

↳ Un insieme di attributi  $K$  è una superchiave se determina funzionalmente tutti gli attributi della relazione. Attraverso l'algoritmo della chiusura,  $K$  è una superchiave se  $K^+$  contiene l'intero insieme degli attributi dello schema ( $K \rightarrow U$ )

↳ Differenza tra chiave e Superchiave

↳ Mentre la superchiave garantisce l'univocità, la chiave è una chiave minicuale. Ciò significa che se si sottrae anche solo un attributo da una chiave, questa perde la capacità di identificare univocamente le righe, al contrario, una superchiave può contenere attributi "in più" non necessari alla distinzione univoca.

↳ Esistenza e Multiplicità

↳ Ogni relazione possiede sempre almeno una superchiave (nel caso estremo è costituita dall'insieme di tutti i suoi attributi) e ne possono esistere molte più all'interno della stessa tabella.

↳ Livello di schema

↳ La definizione di una superchiave deve essere fatta a livello di schema (basandosi sulle regole del dominio e della realtà di interesse) e non semplicemente osservando un'istanza di dati, poiché un'univocità rilevata casualmente in una tabella specifica potrebbe non valere in futuro.

## Normalizzazione

Processo volto ad eliminare le ridondanze fisiche e a prevenire anomalie di aggiornamento, inserimento e cancellazione.

Il problema: le anomalie dei dati

↳ Quando le informazioni non sono ben distribuite tra le tabelle, si verificano tre problemi:

↳ Ridondanza fisica

↳ Lo stesso dato (es. lo stipendio di un docente o l'indirizzo di un dipartimento) viene ripetuto in decine di righe ≠.

↳ Anomalia di aggiornamento

↳ Se un dato cambia (es. il docente riceve un aumento), bisogna modificare manualmente ogni singola riga in cui compare, con il rischio di lasciare dati inconsistenti.

↳ Anomalia di cancellazione → se eliminiamo l'unica riga che contiene una certa informazione (es. unico corso di un docente), rischiamo di perdere definitivamente anche i dati anagrafici del docente stesso.

Le dipendenze funzionali

↳ La normalizzazione si basa sullo studio delle dipendenze funzionali. Si dice che un insieme di attributi  $Y$  determina  $Z$  ( $Y \rightarrow Z$ ) se, & volta che il valore di  $Y$  si ripete in due righe, deve necessariamente ripetersi anche il valore di  $Z$ .

↳ Dipendenze "Buone"

↳ La parte sinistra della freccia è una superchiave. In questo caso non ci sono ridondanze perché ogni valore della chiave identifica una riga univoca.

↳ Dipendenze "Cattive"

↳ La parte sx non è una superchiave questo genera duplicati inutili e anomalie.

Le forme Normali

↳ Esistono diversi livelli di qualità di una tabella chiamati Forme Normali:

↳ Prima forma Normale (PFN): impone che ogni colonna contenga solo valori atomici (niente liste o gruppi di valori in una sola cella)

↳ Seconda forma Normale (SFN) → non devono esserci dipendenze parziali, avendo attributi che dipendono solo da una parte di una chiave composta

↳ Terna forma Normale (TFN) → è il compimento più usato. Garantisce che non ci siano dipendenze transitive e che ogni "freccia" logica parta da una superchiave o arrivi a un attributo che fa parte di una chiave.

↳ Forma Normale di Boyce-Codd (FNBC) → è la più restrittiva. Ogni dipendenza funzionale deve avere obbligatoria come una superchiave a sinistra. Tuttavia, non è sempre possibile ottenerla preservando tutti i vincoli, motivo per cui spesso ci si gira alla 2FN.

### Algoritmo di Normalizzazione (in 2FN)

↳ Per normalizzare una tabella che presenta problemi, si segue una procedura formale:

- ① Copertura ridotta → si seppelliscono le dipendenze eliminando attributi e regole superflue
- ② Raggruppamento → si uniscono le dipendenze che hanno la stessa parte sinistra.
- ③ Creazione tabelle → ogni gruppo diventa una tabella separata, dove la parte sx della dipendenza serve da chiave primaria.
- ④ Verifica delle chiavi → ci si assicura che almeno una delle nuove tabelle contenga la chiave della tabella originale per poter ricostruire i dati tramite join.

### Esempi

PFN → Una tabella è in PFN se tutti i suoi attributi sono definiti su domini atomici, ovvero se ogni cella contiene un solo valore elementare e non liste o gruppi di dati.

ES.

Una Tabella così con colonna Ingo Docente che contiene il valore "M. Di Felice, Professore, Codice: 1234"

### Soluzione

↳ Dividere le informazioni in colonne separate: Docente, Ruolo, CodiceDocente. Solo così la relazione è normalizzata in 1FN.

### 2FN

↳ Una tabella è in 2FN se è in 1FN e se tutti gli attributi non-chiave dipendono dall'intera chiave primaria e non solo una parte di essa (assenza dipendenze parziali).

### Esempio violazione

↳ Consideriamo lo schema IMPIEGATO (Impiegato, Stipendio, Progetto, Budget)

PK { Impiegato, Progetto }

$d_{f_1}$ : Impiegato → Stipendio

$d_{f_2}$ : Progetto → Budget

Soluzione → Decomporre in più tabelle in cui ogni attributo dipenda solo dalla sua chiave specifica (es. una tabella per gli stipendi degli impiegati e una per i budget dei progetti)

# TFN

Una tabella è in TFN se e' in ZFN e se ogni attributo non chiave dipende dalla chiave primaria in modo diretto, ovvero non tramite altri attributi (assenza di dipendenze transitive)

## Esempio di violazione

IMPIEGATO (Impiegato, Categoria, Stipendio)

PK  $\rightarrow$  Impiegato

df<sub>1</sub>: Impiegato  $\rightarrow$  Categoria

df<sub>2</sub>: Categoria  $\rightarrow$  Stipendio

Lo stipendio dipende quindi dall'Impiegato solo attraverso la Categoria. Questa è una dipendenza transitiva che causa ridondanze (se molti impiegati hanno la stessa categoria, ripeteranno lo stipendio molte volte)

## Soluzione

↳ Creare una tabella separata per le categorie: CATEGORIE (Categoria, Stipendio)

Forma normale di Boyce e Codd (FNBC)

↳ È la forma più restrittiva. Una tabella è in FNBC se,  $\forall$  dipendenza funzionale  $X \rightarrow A$ , X è una superchiave.

## Esempio di violazione

NEGOZIO (Località, Stato, Prefisso)

PK  $\rightarrow$  {Località, Stato}

df: Stato  $\rightarrow$  Prefisso

Notiamo che lo Stato da solo non è una super-chiave, perché sotto uno Stato ci sono più Località)

## Esempio di compromesso

↳ In una tabella con {Località, Stato, Abitanti}

Se la df è Località, Stato  $\rightarrow$  Abitanti

e la coppia {Località, Stato} è la chiave, allora FNBC è rispettata.

## Forca Normale di Boyce e Codd (FNBC)

↳ Una sferzata di relazione  $R(x)$  si dice in Forca Normale di Boyce e Codd (FNBC) se, per ogni dipendenza funzionale non banale del tipo  $Y \rightarrow Z$  definita su di essa, l'insieme di attributi  $Y$  è una superchiave della relazione. In termini più semplici, ciò significa che ogni volta che un insieme di dati ne determina un altro in modo univoco, quell'insieme deve essere in grado di identificare l'intera riga della tabella.

### Scopo della FNBC

↳ Se una tabella rispetta questa forca normale, è garantito che non presenti ridondanze fisiche né anomalie di aggiornamento o cancellazione. Una tabella in FNBC è considerata "ben progettata" dal punto di vista logico.

↳ Dipendenze non banali → Il vincolo si applica solo alle dipendenze non banali, ovvero quelle in cui l'attributo di destra non è già contenuto nell'insieme di sx.

(es.)

### Caso Conforme

EVENTO (località, stato, abitanti)

Se esiste la df e la coppia (località, stato) è una superchiave  
↳ località, stato  $\rightarrow$  abitanti ]  $\rightarrow$  FNBC è rispettata

non genera duplicati

### Caso non conforme

EVENTO (località, stato, prefisso)

Se esiste la df Stato  $\rightarrow$  prefisso con lo stato non è una superchiave, la FNBC è violata perché si generano duplicati inutili del prefisso per ogni località dello stesso stato.