I.T.E.S. Polo Commerciale "PITAGORA"

Via Pupino 10/A – 74121 Taranto)

TEORIA

sulle

BASI DI DATI

A cura del Prof. Enea Ferri

Cos'è un DATA BASE

E' un insieme di archivi legati tra loro da relazioni. Vengono memorizzati su memorie di massa come un unico insieme, e possono essere manipolati da programmi diversi.

Differenze tra Archivi tradizionali e Data Base

Negli archivi tradizionali la definizione del tracciato record viene effettuata all'interno dei programmi che li utilizzano; i dati possono essere utilizzati solo dai programmi che li hanno generati.
Una modifica del tracciato record comporta la modifica dei programmi che utilizzano l'archivio. I programmi sono quindi strettamente legati ai dati che utilizzano, sia per quanto riguarda la loro definizione, sia per quanto riguarda la tecnica di memorizzazione utilizzata.

I problemi principali di un archivio sono:

- la ridondanza dei dati, cioè gli stessi dati compaiono più volte all'interno dell'archivio;
- l'incongruenza (portata dalla ridondanza), nel caso in cui lo stesso dato sia aggiornato in un archivio e non in un altro;
- l'inconsistenza (portata dall'incongruenza), cioè i dati a disposizione non sono più affidabili, perché non si sa in modo certo quale dei diversi valori sia quello corretto.

Nei data base i programmi si svincolano da questo legame. La definizione dei dati e come questi sono organizzati su memoria di massa fanno parte integrante della struttura del Data base. Pertanto gli stessi dati potranno essere gestiti da programmi scritti in linguaggi di programmazione differenti, purché dotati di opportune "interfacce" che consentano loro di accedere e modificare i dati.
Ogni linguaggio di programmazione avrà un kit di istruzioni per poter utilizzare il data base.

Le caratteristiche fondamentali di una Base di Dati sono:

- *indipendenza dalla struttura fisica dei dati*, i programmi applicativi sono indipendenti dai dati fisici;

Indipendenza dalla struttura fisica dei dati: non esiste un legame tra un file ed un altro se non c'è una chiave che li lega.

- *indipendenza dalla struttura logica dei dati*, i programmi applicativi sono indipendenti dalla struttura logica con cui i dati sono organizzati negli archivi;

Indipendenza dalla struttura logica dei dati: se si modifica un'informazione non c'è bisogno di modificare il programma che utilizza il data base perché viene modificato solo lo schema.

- *utilizzo da parte di più utenti*, i dati organizzati in un unico Data Base possono essere utilizzati da più utenti;
- *eliminazione della ridondanza*, o quantomeno riduzione, gli stessi dati non compaiono più volte in archivi diversi;
- *eliminazione della inconsistenza e quindi della incongruenza*, il Data Base non può presentare campi uguali con valori diversi in archivi diversi;

- *facilità di accesso*, il ritrovamento dei dati è facile e veloce;
- *aderenza ad un preciso **modello** di dati; Il modello dei dati è solitamente uno schema che ci fornisce informazioni su com'è organizzato il nostro data base.*
- *integrità dei dati*, vengono effettuati controlli per evitare anomalie causate dai programmi degli utenti;
- *sicurezza dei dati*, sono previste procedure di controllo per impedire accessi non autorizzati al Data Base;
- *uso dei linguaggi per la gestione del Data Base*, il Data Base viene gestito attraverso comandi per la manipolazione (inserimento, cancellazione, modifica) dei dati, e comandi per effettuare interrogazioni, al fine di ottenere le informazioni desiderate.

Chi gestisce il Data Base

La gestione della struttura del Data base, della sua definizione, organizzazione, manutenzione e consultazione è affidata ad un software dedicato, chiamato DBMS (Data Base Management System)
Il data base è il contenitore all'interno del quale si trovano le informazioni raggruppate in archivi. Il DBMS provvede alla definizione degli archivi, alla loro creazione e alla impostazione delle relazioni che legano tra loro le informazioni in essi contenuti. I DBMS utilizzano il modello RELAZIONALE per la organizzazione dei dati, termine che deriva dalle RELAZIONI che tale modello definisce tra gli archivi che costituiscono il Data Base.

Quanti DBMS ci sono

Non esiste un unico DBMS. Ogni produttore di questo software caratterizza la gestione secondo proprie preferenze. Così ACCESS, MYSQL, ORACLE, FOXPRO hanno ciascuno i propri DBMS. In comune hanno il linguaggio proprietario per la gestione e interrogazione dei dati, ovvero il linguaggio SQL standard (Structured query language). Ognuno però fornisce ai diversi linguaggi di programmazione, le interfacce che saranno utilizzate dai programmi per poter utilizzare le informazioni registrate nel data base.
Quindi lo stesso data base potrà essere utilizzato da programmi scritti in linguaggi di programmazione diversi.

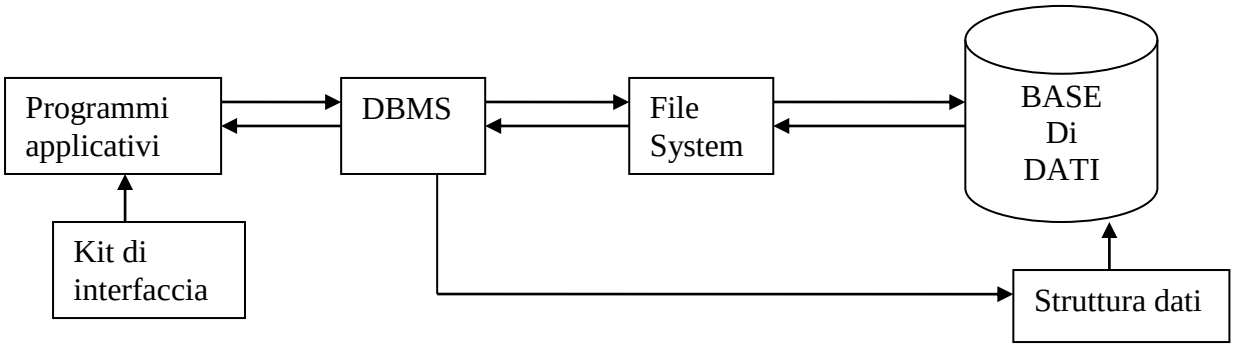
Funzioni di un DBMS

- Gestire grandi quantità di dati, prestando attenzione soprattutto alla efficienza, garantendo un accesso veloce;
- Consentire l'accesso alla Base di dati a più utenti (condivisione dei dati), coordinando gli accessi per evitare di fornire agli utenti dati errati o non aggiornati;
- Garantire l'affidabilità dei dati, controllando gli accessi mediante password, per evitare che utenti non autorizzati possano alterare o danneggiare i dati

DBMS e File System

Il DBMS per poter accedere al Data base ha bisogno del File System. Infatti è il File System il modulo del Sistema Operativo che si occupa della organizzazione e gestione dei file su memoria di massa (creazione, lettura, scrittura, cancellazione, ecc.). Schematizzando:

Utente → Programma applicativo → DBMS → File System → DATA BASE



In questo caso dunque, la maggiore 'distanza' del DBMS dall'hardware, rispetto al file system, permette una maggiore libertà di azione agli utenti. Questo significa che l'utente (programmatore, amministratore di sistema, operatore, ecc.) non dovrà più avere a che fare con record e file, bensì con entità astratte che rappresentano la realtà.

Progettazione di un Data Base

La progettazione di un Data Base passa attraverso le seguenti fasi:

1. **Analisi del problema**
2. **Progettazione concettuale del data base**
3. **Progettazione logica del data base**
4. **Implementazione**

1. Analisi del problema

E' la fase in cui vengono evidenziati i fabbisogni che il Data Base deve soddisfare e le attività aziendali coinvolte nella gestione, per ciascuna delle quali si individuano le funzioni da inserire nella progettazione, con le rispettive priorità.

2. Progettazione concettuale

In questa fase la realtà da studiare e da gestire con il Data Base si esamina attentamente, individuando le "ENTITA'" che la rappresentano. Sono un esempio le entità PRODOTTO, FORNITORE, CLIENTE, VENDITE, DIPENDENTE ecc. Per ogni entità si individuano le proprietà (attributi) per poter distinguere un elemento (istanza) dagli altri. Ad esempio per l'entità PRODOTTO (Codice, Descrizione, Prezzo, Giacenza…). Si costruisce cioè il "Modello dei dati", una rappresentazione astratta degli oggetti che costituiscono il Data Base e delle regole che governano le operazioni tra i dati.
Il più usato è il Modello E-R (entità-Relationship)

2.1 Il Modello E-R

Il modello ENTITA'-ASSOCIAZIONE (E-R) è un modello concettuale di dati che fornisce una serie di strutture, dette *costrutti*, per potere descrivere la realtà che si vuole studiare in una maniera facile da comprendere e senza fare riferimenti ai criteri con cui i dati sono organizzati negli elaboratori. Si basa sui concetti di Entità, Attributi, Associazioni tra Entità.

Entità:

È un oggetto esistente nel mondo reale che si vuole rappresentare nel modello concettuale. Un'entità può essere una cosa o una persona, un luogo oppure un concetto. Ad esempio *Libro* e *Utente* sono esempi di entità di una applicazione per la gestione di una biblioteca. Il Sig. Mario Rossi è un esempio di una occorrenza (o ISTANZA) dell'entità *Utente*. "SQL, il linguaggio della base di dati" è un esempio di istanza dell'entità Libro

Prodotto, Fornitore e Cliente sono esempi di entità nella gestione delle vendite e degli acquisti di prodotti di magazzino.

Alunno, Materia e Voto sono entità nella gestione delle valutazioni degli alunni nelle varie materie.

Scrittore e Argomento sono entità nella gestione degli autori di libri o riviste che trattano argomenti specifici.

Attributi:

Sono le proprietà necessarie per caratterizzare un elemento di una entità. Come visto prima, l'entità Prodotto è caratterizzato dagli attributi:Codice, Descrizione, Prezzo, Giacenza, ecc.

L'entità Utente è caratterizzata dagli attributi: Cognome, Nome, Indirizzo, CodiceFiscale, ecc.

Tra gli attributi di una entità si definisce CHIAVE l'attributo che identifica univocamente una istanza dell'entità. Si definiscono DESCRITTORI gli altri attributi, i cui valori possono ripetersi per diverse istanze dell'entità, ad esempio il comune di residenza, l'indirizzo, il nome.

Associazioni tra entità:

Sono relazioni definite tra due entità.

L'entità da cui parte l'associazione si chiama entità padre; quella a cui si arriva si chiama entità figlio.

La direzione di una associazione può essere obbligatoria o facoltativa. Si indica con una linea continua quando è obbligatoria, con la linea tratteggiata quando è facoltativa.

Invece di dire Associazione tra due Entità di solito si parla di RELAZIONE tra due entità. Indicano entrambe un collegamento tra le istanze di due insiemi.

Di solito le relazioni vengono descritte mediante un verbo. Ad esempio tra Alunni e Materie esiste la relazione:"Un alunno può essere verificato in più materie". Tra Materia e Alunno esiste la relazione:" Una Materia può essere oggetto di verifica per più alunni".

Relazione 1:1 (si legge 1 a 1)

Si ottiene quando una istanza della prima entità è associata ad una sola istanza della seconda entità. Es: Persona e Codice_Fiscale. Ad una persona corrisponde un solo codice fiscale. Ad un codice fiscale corrisponde una sola persona.

Relazione 1:N (si legge uno a molti)

Si ottiene quando una istanza della prima entità è associata a molte istanze della seconda entità.

Es. Comune, Persona. Ad un comune sono associate molte persone che vi risiedono. Ad una persona è associato un solo comune in cui risiede.

Relazione M:N (si legge Molti a Molti)

Si ha quando una istanza della prima entità è associata a molte istanze della seconda entità e viceversa, ad una istanza della seconda entità sono associate molte istanze della prima entità. Es. Studente e Materia. Uno studente può essere verificato in più materie; una materia può essere oggetto di verifica per più studenti.

2.2 Regole da seguire per una corretta modellazione dei dati.

- Individuare le Entità necessarie per descrivere la realtà che si vuole studiare.
- Per ciascuna entità definire gli attributi necessari per poter distinguere tra loro le diverse istanze dell'entità, evitando di ripetere attributi uguali in entità diverse.
- Stabilire le relazioni esistenti tra le entità:
    - se tra due entità si individua una relazione 1:1 fondere le due entità in un'unica entità
    - se tra due entità esiste una relazione M:N è necessario trasformarla in due relazioni 1:N introducendo una ulteriore entità, detta Entità associativa o entità cuscinetto, perché altrimenti non potrebbero essere rappresentate nel modello relazionale.
- Trasformare le relazioni complesse, ovvero quelle che coinvolgono più entità in relazioni binarie, che coinvolgono solo due entità
- Eliminare le relazioni ridondanti: Es. Persona, Paese e Provincia. Un paese di trova in una Provincia (1:N tra Provincia e Paese). Una Persona risiede in un Paese (1:N Tra Persona e Paese); Una Persona Abita in Provincia di (1:N Tra Persona e Provincia). La relazione tra Persona e Provincia è ridondante.

3. Progettazione logica

Nel modello relazionale un Database è un insieme di tabelle.
Una tabella è paragonabile ad un archivio dove le colonne sono i campi del record
Le righe rappresentano le istanze della entità rappresentata dalla tabella (i record dell'archivio)
Dominio: insieme dei valori che possono essere presenti in una colonna
Grado di una tabella: Numero delle colonne
Cardinalità di una tabella: Numero delle righe

Regole di derivazione

Vengono attuate nel passaggio dal livello concettuale al livello logico:

1. ogni entità diventa una tabella
2. ogni istanza di una entità diventa una riga della tabella
3. ogni attributo della entità diventa una colonna della tabella
4. la chiave primaria della entità diventa identificatore univoco delle righe della tabella

Chiave primaria

Una chiave primaria è un attributo (chiave semplice) o un insieme di attributi (chiave composta) di una entità che identificano in modo univoco una istanza. Deve avere le seguenti proprietà:

- il valore deve essere specificato per ogni istanza
- il valore deve essere unico per ogni istanza
- il valore non deve cambiare o diventare nullo

Chiave artificiale

E' formata da un attributo privo di un significato proprio, che viene aggiunto agli altri in modo artificiale per ottenere un codice univoco per ogni istanza. Di solito la chiave artificiale è composta da un contatore che si incrementa in modo automatico quando si aggiunge una nuova istanza all'entità.

Chiave esterna

E' un attributo aggiunto in una entità, necessario per potere identificare l'entità padre ad essa associata mediante una relazione.

Aggiungere attributi al modello

In genere, quando tra due entità si identifica una relazione molti a molti, con la introduzione della entità associativa, si rende necessario aggiungere ulteriori attributi per poter descrivere correttamente le due relazioni uno a molti generate. Ad esempio, tra Studenti e Materie esiste una relazione molti a molti. Informazioni come la data della verifica, il tipo della verifica e il voto riportato non possono essere attributi di nessuna delle due entità. Diventano invece attributi della entità associativa Verifica, introdotta per poter trasformare una relazione molti a molti in due relazioni uno a molti; essa conterrà gli attributi detti precedentemente oltre alle due chiavi esterne necessarie per poter identificare lo studente e la materia.

4. Implementazione

E' la fase in cui utilizzando opportuni linguaggi messi a disposizione dal DBMS in uso, si realizza concretamente il Data Base sul computer.

Il DDL (Data Definition Language) ha lo scopo di creare la struttura delle tabelle e di impostare le relazioni tra chiavi primarie e chiavi esterne. In questo momento si fissano le regole che dovranno essere seguite durante le fasi di inserimento, modifica e cancellazione di record o di intere tabelle.

Aggiungere le regole di integrità dei dati

L'integrità dei dati nel modello relazionale attesta che i dati contenuti nel data base siano corretti e *consistenti*, ovvero coerenti, affidabili, conformi all'uso che se ne deve fare e con contraddittori.

Integrità dell'entità: per ogni istanza di una entità deve esistere il valore della chiave primaria, essere unico e non NULL[1].

Integrità referenziale: per ogni valore della chiave esterna deve esistere un valore di chiave primaria nella entità associata. Questo significa che con questo vincolo non si potrà inserire una istanza nella entità figlio se non esiste l'istanza da associare nella entità padre, e non si potrà cancellare una istanza della entità padre se vi sono istanze collegate da una relazione nell'entità figlio.

Regole di inserzione

- Inserimento dipendente: consente l'inserimento di una istanza nella entità figlio solo se esiste già la chiave primaria nella entità padre

[1] Con il termine NULL si intende valore non assegnato, che è diverso dal dire valore 0 per un numero o spazio per una stringa

- Inserzione automatica: permette l'inserimento di una istanza nella entità figlio. Se l'istanza nella entità padre non esiste, viene creata
- Inserzione nulla: consente l'inserimento di una istanza nella entità figlio anche se nella entità padre non esiste l'istanza associata. La chiave esterna viene impostata a NULL.
- Inserzione di default: viene inserita l'istanza nella entità figlio anche se non esiste nella entità padre l'istanza associata; la chiave esterna viene impostata ad un valore di default.
- Nessun effetto: l'inserimento della istanza nella entità figlio viene effettuata comunque, senza effettuare alcun controllo di consistenza.

Regole di cancellazione

- Cancellazione con restrizione: viene cancellata una istanza nella entità padre solo se non ci sono istanze correlate nella entità figlio
- Cancellazione a cascata: viene sempre cancellata una istanza padre e vengono cancellate tutte le istanze nella entità figlio
- Cancellazione nulla: consente sempre una istanza nella entità padre. Tutte le istanze della entità figlio restano e la chiave esterna viene impostata a NULL.
- Cancellazione di default: cancella sempre una istanza della entità padre; se esiste qualche istanza nella entità figlio ad essa correlata, la chiave esterna viene impostata ad un valore di default.
- Nessun effetto: viene sempre cancellata l'istanza della entità padre senza effettuare alcun controllo di consistenza

Scelta delle regole

- Evitare l'uso di inserzione e cancellazioni nulle (vincolo di esistenza della istanza nella entità padre)
- Usare la regola di inserzione automatica o dipendente (dati coerenti)
- Usare la regola di cancellazione a cascata (conservata coerenza per le chiavi esterne)

Manipolazione dei dati

Le seguenti operazioni sono possibili solo tra tabelle aventi la stessa struttura dei dati

Unione

Vengono aggiunte in coda ad una tabella (Append) le righe di una seconda tabella per produrne una terza.

Differenza

La differenza tra due tabelle è una terza tabella contenente le righe della prima che non sono presenti nella seconda.

Intersezione

L'intersezione tra due tabelle crea una terza tabella contenente le righe comuni ad entrambe.

Prodotto

Il prodotto tra due tabelle (detto anche prodotto cartesiano) ne genera una terza, avente grado uguale alla somma dei gradi delle tabelle date, e in cui ciascuna riga si ottiene concatenando ogni riga della prima tabella con ciascuna riga della seconda.

Le seguenti operazioni sono possibili su una singola tabella e su due o più tabelle anche con struttura dei dati diversa

Proiezione

La proiezione di una tabella ne genera un'altra contenente solo alcune colonne. Il grado della tabella risultante sarà minore del grado della tabella di partenza

Selezione

La selezione si applica ad una tabella e ne genera un'altra contenente solo un gruppo di righe che soddisfano una certa condizione. Il grado è uguale a quello della tabella di partenza, la cardinalità sarà minore

Congiunzione (Join)

L'operazione di congiunzione viene effettuata tra due tabelle (padre e figlio) collegate mediante una associazione. Viene generata una terza tabella cove si combinano le righe della prima con le righe della seconda.

1. Equi Join: si ha quando è stato effettuato l'inserimento dipendente. Pertanto verranno considerate tutte le righe della prima tabella e tutte le righe della seconda tabelle. La tabella ottenuta avrà cardinalità uguale a quella delle due tabelle; Il grado della tabella risultante sarà uguale alla somma dei gradi delle due tabelle date –1 (chiave esterna e chiave primaria sono uguali, per cui viene preso una sola volta il suo valore).
2. Left Join: si ha quando vi sono istanze della prima tabella che non hanno istanze collegate nella seconda tabella. La tabella risultante avrà quindi tutte le righe della prima tabella e solo le righe della seconda tabella che hanno chiave esterna impostata. La cardinalità è uguale a quella della prima tabella, il grado è sempre uguale alla somma dei gradi –1.
3. Right Join: si ha quando vi sono istanze della seconda tabella che non hanno un collegamento con istanze della prima. La tabella risultante avrà tutte le righe della seconda tabella e solo le righe della prima che hanno chiave esterna impostata. La cardinalità è uguale a quella della seconda tabella, il grado è sempre uguale alla somma dei gradi –1.

La Normalizzazione

E' un procedimento che tende a eliminare la ripetizione dei dati dalle tabelle, dividendole in tabelle più piccole attraverso operazioni di proiezione. La teoria della normalizzazione è basata sul concetto di forma normale. Una tabella relazionale è in una particolare forma normale se soddisfa un insieme di vincoli.

Prima forma normale

Una tabella è in 1FN se vi sono solo campi elementari e viene ridotta se non eliminata la ridondanza dei dati.
ogni riga di ciascuna tabella deve poter essere identificata in modo univoco tramite un gruppo di dati in essa contenuti. In altre parole, in una tabella del tipo:

Nome	Età	Professione
Alberto	30	Impiegato
Gianni	24	Studente
Alberto	30	Impiegato
Giulia	50	Insegnante

non è possibile distinguere il dato inserito nella prima riga da quello inserito nella terza: le due righe sono infatti identiche. L'Alberto della prima riga, di 30 anni impiegato, non è infatti distiunguibile dall'Alberto, 30 anni, impiegato, della terza riga.

Il problema potrebbe essere risolto inserendo un altro campo nella tabella, con valore diverso per ogni riga, ad esempio il codice fiscale. A questo punto il database sarebbe in **prima forma normale**.

Il campo o l'insieme di campi diversi per ciascuna riga e sufficienti ad identificarla sono detti **chiave primaria** della tabella (in questo caso il codice fiscale).

Codice Fiscale	Nome	Età	Professione
LBRRSS79Y12T344A	Alberto	30	Impiegato
GNNBNCT84A11L611B	Gianni	24	Studente
LBRMNN79E64A112A	Alberto	30	Impiegato
GLSTMT59U66P109B	Giulia	50	Insegnante

La tabella vista poco sopra è in 1NF; per chiarezza facciamo un esempio di una tabella che, seppur munita di una chiave primaria, non può essere considerata in forma normale:

Codice Fiscale	Nome	Dettagli
LBRRSS79Y12T344A	Alberto	età: 30; professione: Impiegato
GNNBNCT84A11L611B	Gianni	età: 24; professione: Studente

La tabella qui sopra NON è in 1NF in quanto, pur avendo una chiave primaria, presenta un campo (dettagli) che non contiene dati in forma atomica.

Seconda forma normale

Una tabella con chiave primaria composta è in 2FN quando è in 1FN e tutti i campi non chiave dipendono interamente dalla chiave primaria e non solo da una sua parte.

Per fare un esempio si supponga di avere a che fare con il database di una scuola con una **chiave primaria composta** dai campi "Codice Matricola" e "Codice Esame":

Codice Matricola	Codice Esame	Nome Matricola	Voto Esame
1234	**M01**	Rossi Alberto	6
1234	**L02**	Rossi Alberto	7
1235	**L02**	Verdi Mario	8

Il database qui sopra si trova in 1NF ma non in 2NF in quanto il campo "Nome Matricola" non dipende dall'intera chiave ma solo da una parte di essa ("Codice Matricola").
Per rendere il nostro database 2NF dovremo scomporlo in due tabelle:

Codice Matricola	Codice Esame	Voto Esame
1234	**M01**	6
1234	**L02**	7
1235	**L02**	8

e

Codice Matricola	Nome Matricola
1234	Rossi Alberto
1235	Verdi Mario

Nella prima tabella il campo "Voto" dipende correttamente dalla chiave primaria composta da "Codice Matricola" e "Codice Esame", nella seconda tabella il campo "Nome Matricola" dipende correttamente dalla sola chiave primaria presente ("Codice Matricola").

Ora il nostro database è normalizzato in seconda forma normale.

Terza forma normale

Una tabella è in 3FN quando è in 2FN e tutti i campi non chiave dipendono solo dalla chiave primaria.
Per fare un esempio torniamo all'ipotetico database della palestra; supponiamo di avere una base dati che associ il codice fiscale dell'iscritto al corso frequentato ed all'insegnante di riferimento. Si supponga che il nostro DB abbia un'unica chiave primaria ("Codice Fiscale") e sia così strutturato:

Codice Fiscale	Codice Corso	Insegnante
LBRRSS79Y12T344A	BB01	Marco
GNNBNCT84A11L611B	BB01	Marco
LBRMNN79E64A112A	BB01	Marco
GLSTMT59U66P109B	AE02	Federica

Il nostro database non è certamente 3NF in quanto il campo "insegnante" non dipende dalla chiave primaria ma dal campo "Codice Corso" (che non è chiave).

Per normalizzare il nostro DB in 3NF dovremo scomporlo in due tabelle:

Codice Fiscale	Codice Corso
LBRRSS79Y12T344A	BB01
GNNBNCT84A11L611B	BB01
LBRMNN79E64A112A	BB01
GLSTMT59U66P109B	AE02

e

Codice Corso	Insegnante
BB01	Marco
AE02	Federica

Il nostro database è ora in terza forma normale.