UNIVERSITÀ DEGLI STUDI DI MILANO

FACOLTÀ DI SCIENZE E TECNOLOGIE

CORSO DI LAUREA IN INFORMATICA MUSICALE

CREAZIONE COLLABORATIVA DI PLAYLIST VIA WEB

RELATORE:
Luca A. Ludovico

CORRELATORE:
Andrea Frabetti

TESI DI LAUREA DI:
Marcello Radice
Matricola N° 776505

Anno Accademico 2012/2013

Indice

1 Introduzione **1**

1.1 Premessa....1

1.2 Obiettivi....2

1.3 Struttura....3

2 Basi di dati **4**

2.1 DBMS e MySQL....5

2.2 Il modello relazionale....6

2.3 L'algebra relazionale....7

3 Gli strumenti di sviluppo **9**

3.1 SQL.... 9

3.1.1 Tipi di dato....10

3.1.2 Creazione di relazioni e chiavi.... 10

3.1.3 Cancellazione e modifica....11

3.1.4 Interrogazioni....11

3.2 HTML....13

3.2.1 La sintassi.... 13

3.2.2 L'intestazione del documento.... 14

3.2.3 Il corpo del documento.... 15

3.2.4 Le tabelle.... 15

3.2.5 I collegamenti ipertestuali....16
3.2.6 I Form.... 16
3.3 PHP....17
3.3.1 La sintassi.... 18
3.3.2 Le variabili....18
3.3.3 Le strutture di controllo.... 20
3.3.4 Le funzioni....22
4 Applicativo **23**
4.1 Il database....23
4.1.1 phpMyAdmin....25
4.2 Le pagine web.... 27
4.2.1 La classifica e la top 10.... 27
4.2.2 Il player audio....31
4.2.3 La votazione....33
4.2.4 Il lato amministrativo....35
4.2.5 L'inserimento di un nuovo record....38
4.2.6 La modifica di un record....41
4.2.7 L'eliminazione di un record.... 46
5 Conclusioni e sviluppi futuri **48**
Bibliografia **50**

1 Introduzione

1.1 Premessa

R.C.S. Radio Cernusco Stereo nasce nel 1977 in un piccolo spazio preso in prestito all'oratorio Paolo VI sull'onda, è il caso di dirlo, del grande fermento che aveva visto la nascita delle prime radio "libere" in Italia l'anno precedente... Dal 2003 è iniziato un progressivo lavoro di ammodernamento tecnologico che ci ha portato ad automatizzare completamente tutte le emissioni, ad essere presenti su Internet con un sito www.rcs939.it ricco di contenuti dal quale è possibile ascoltarci online da tutto il mondo. [1]

Oggi Radio Cernusco Stereo è sia una normale radio, trasmette infatti in Fm alla frequenza 93.9, sia una Web radio, ovvero un'emittente radiofonica che trasmette in forma digitale il proprio palinsesto attraverso Internet.

La R.C.S. presenta, dal 2012, un programma chiamato RCS LAB che si occupa di trovare ogni settimana un nuovo artista emergente, presentare un suo singolo e introdurlo nella programmazione settimanale della radio. Sul sito della radio è presente una sezione apposita, chiamata appunto RCS LAB suddivisa per anni, che contiene la recensione settimanale di ogni artista. Con il passare del tempo, visto il successo di questo programma e spinti dal desiderio di poter dare un maggiore spazio agli artisti emergenti, sì è pensato di fare una classifica delle canzoni degli artisti emergenti permettendo al pubblico di riascoltare e votare le canzoni preferite. Questo permette da un lato che gli artisti emergenti abbiano un maggior feedback delle loro canzoni sul pubblico, avendo inoltre la possibilità di poter essere riascoltati, dall'altro che la radio pubblicizzi il sito grazie al sistema di votazione. Il fine della classifica inoltre è quello di premiare gli artisti più apprezzati

1.2 Obiettivi

Il seguente elaborato ha lo scopo di approfondire lo studio nell'ambito del web database, per implementare una pagina interattiva con contenuti multimediali fruibili dagli utenti.

Web e database sono due tecnologie che, in modo complementare, permettono l'accesso facile e veloce a grandi quantità di dati e informazioni, facendo interagire dinamicamente pagine Web e database. Su queste tecnologie si basano i siti di vendita come “Amazon” o “e-bay” che permettono l'acquisto e le vendita online aggiornando grandi quantità di dati in tempo reale. Senza il supporto di un database interattivo con le pagine web sarebbe impossibile la gestione della grande mole di dati costringendo a creare e gestire una pagina diversa per ogni articolo in vendita.

Queste tecnologie sono in costante sviluppo e aggiornamento al fine di aggiungere funzioni sempre nuove che permettano sia una stesura più semplice del codice per programmare le pagine internet e per strutturare il database, sia l'aggiunta di nuove applicazioni web.

Con questo elaborato si è voluto portare un esempio concreto di come utilizzare queste tecnologie per strutturare una pagina web, con annesse pagine amministrative per la gestione dei dati, e per creare una pagina interattiva contenente informazioni e dati degli artisti emergenti della Radio Cernusco Stereo.

Per sviluppare questa pagina sono stati usati diversi linguaggi. Il primo è il linguaggio SQL, che si occupa della creazione e strutturazione del database e della gestione dei dati all'interno di esso; il secondo è HTML, che viene utilizzato per creare le pagine internet sulle quali verranno visualizzati i dati contenuti nel database; il terzo è PHP, necessario per collegare e interfacciare la pagina HTML con il database e accedere ai dati in esso contenuti.

L'insieme di questi tre linguaggi dà come risultato effettivo la gestione e la visualizzazione in pagine internet, tramite il proprio browser, delle informazioni contenute nel database.

Sul sito internet della radio si distingueranno poi due parti, una dedicata agli amministratori del sito per inserire, modificare e eliminare i dati relativi agli artisti e alle canzoni, e una dedicata al pubblico.

La pagina pubblica permette agli utenti di:

- vedere le informazioni degli artisti

- collegarsi ai video caricati dagli artisti su altri siti
- ascoltare le canzoni degli artisti
- votare le canzoni preferite per generare una classifica pubblica aggiornata in tempo reale.

1.3 Struttura

L'elaborato finale è strutturato in varie parti: una parte introduttiva legata al concetto di Base di Dati, un capitolo dedicato ai linguaggi fondamentali utilizzati per la realizzazione dell'applicativo, un capitolo che descrive lo svolgimento e la creazione dell'applicativo e un capitolo conclusivo .

Nella prima parte sarà introdotto il concetto di Base di Dati, comprendendo i modelli e gli algoritmi su cui è basata e gli strumenti implementativi odierni per realizzare un database online.

Nella seconda parte saranno analizzate le caratteristiche principali dei tre linguaggi utilizzati per la creazione della Base di Dati e della pagina Web, in modo da poter comprendere appieno le loro funzioni e il modo in cui essi cooperano al fine di realizzare gli applicativi.

Nella terza parte verrà presentata una trattazione sul lavoro svolto, spiegando dettagliatamente punto per punto le parti da cui esso è composto, al fine di mostrare il processo di ideazione e creazione del database e delle pagine internet e la loro funzione all'interno del progetto complessivo.

Verranno inoltre riportati fedelmente frammenti di codice e screenshot delle parti rilevanti e di maggiore valenza del progetto al fine di dimostrare la loro diretta funzionalità all'interno del lavoro stesso.

Nella parte conclusiva si presenteranno i risultati a cui ha portato il lavoro successivamente alla sua pubblicazione, riportando i dati statistici contenenti le informazioni riscontrate dalla radio in seguito all'aggiunta della nuova pagina interattiva e dell'impatto che essa ha avuto sul nuovo modo di vivere la musica sia da parte degli ascoltatori sia degli artisti emergenti.

2 Basi di dati

“L'informazione è lo scambio di conoscenza tra due o più persone, all'interno di una comunità o nella società.” La storia e l'evoluzione umana sono frutto di un' accumulazione di conoscenze tramandate attraverso l'informazione, diventata nel corso del tempo una risorsa vitale per ogni organizzazione. Con l'avvento di internet, lo scambio di informazioni è stato facilitato abbattendo di fatto le barriere geografiche. È nata così automaticamente la necessità di poter gestire, acquisire, elaborare e archiviare le informazioni per renderle consultabili e accessibili agevolmente ad ogni interessato. In questo contesto, vista la necessità, sono nati i sistemi di gestione di dati, ovvero sistemi in grado di minimizzare i tempi ed i costi per la manipolazione delle informazioni. Indipendentemente dal fatto che si stia utilizzando la carta o un programma informatico, nel momento in cui vengono raccolti dei dati e si schedano per uno scopo preciso si crea un database. Con il termine Base di Dati denotiamo, quindi, una collezione di dati tra loro correlati, utilizzati per rappresentare le informazioni di interesse in un sistema informativo. [2] Il sistema informativo è l'apparato di un organizzazione che si occupa di gestire e rendere disponibili le informazioni. La Base di Dati è quindi il nucleo e la parte fondamentale del sistema informativo, in quanto è lo strumento mediante il quale vengono gestite le informazioni. In generale nella gestione dei database vengono usati due modelli: operativi o database analitici. Il primo caso viene utilizzato da molte compagnie, organizzazioni e istituzioni in tutto il mondo, con lo scopo di raccogliere, modificare e mantenere dati su una base aggiornata. I tipi di dati immagazzinati sono dinamici, ovvero cambiano continuamente per riflettere informazioni sempre aggiornate poiché i dati di queste compagnie, come ad esempio negozi al dettaglio, ospedali, compagnie pubblicitarie, subiscono continue fluttuazioni. Un database analitico immagazzina invece dati storici legati a un preciso momento. Questo tipo di database è quindi utile per stabilire tendenze, visualizzare dati statistici o effettuare proiezioni commerciali strategiche. I dati raccolti sono statici, non vengono quasi mai modificati se non raramente. Questi tipi di database vengono quindi utilizzati per esempio da laboratori chimici, compagnie geologiche o compagnie di analisi di mercato.

2.1 DBMS e MySQL

Il DBMS o Sistema di gestione di basi di dati è la componente essenziale nella realizzazione di qualsiasi sistema informatico, questa tecnologia si è sviluppata ed è ormai affermata in seguito all'evoluzione dei sistemi di gestione dati iniziata alla fine degli anni sessanta.

Il DBMS è un sistema software ospitato su un computer che si occupa appunto della manipolazione dei dati, offrendo garanzie di affidabilità, efficienza, consistenza e protezione.

I dati in esso contenuti sono resi agevolmente disponibili in vari ambiti applicativi che possono inoltre condividere i dati stessi evitandone la ridondanza. Si può inoltre avere una visione ad alto livello dei dati introducendo un controllo sugli stessi garantendo e minimizzando i problemi legati alla protezione. I DBMS adottano, dal punto di vista strutturale, un' architettura client server con due moduli distinti. Il lato server si occupa della memorizzazione e gestione dei dati, mentre il lato client, eseguito sull'elaboratore dell'utente, gestisce l'interazione tra utente e DBMS attraverso opportune interfacce. Tutte queste operazioni sono possibili grazie a MySQL, che verrà utilizzato per lo sviluppo dell'elaborato. MySQL è un RDBMS ovvero un sistema di gestione della base di dati e include il software di gestione della base di dati, il database server e vari client. Il database server risiede nella macchina assieme ai dati e riceve le richieste effettuate dai client. In seguito a queste richieste, accede alla base di dati e fornisce in risposta i risultati delle richieste. MySQL riconosce le query, ovvero le richieste dei client espresse nel linguaggio SQL, inoltre le operazioni sui database sono sostanzialmente gestite da PHP grazie ad un insieme di funzioni disponibili per i vari database, tra cui MySQL. Quest'ultimo è disponibile su diverse piattaforme, è relazionale, open source e distribuito gratuitamente sotto piattaforma GNU. PHP e MySQL sono una potente combinazione che rende semplice la creazione di applicazioni Web. [3] Per l'applicativo verrà utilizzato phpMyAdmin, un'applicazione Web, che consente di amministrare un database MySQL tramite un qualsiasi browser.

2.2 Il modello relazionale

Esistono vari tipi di modello di database come per esempio quello gerarchico, reticolare o relazionale. Attualmente la maggior parte dei DBMS utilizza il modello relazionale, che è un modello logico di rappresentazione o strutturazione dei dati.

Questo modello non è stato usato dai primi DBMS, ma dalla sua definizione ad opera di Edgard F. Codd nel 1970. Matematico di professione, credeva fermamente di poter applicare precise branche della matematica nella risoluzione di problemi quali la ridondanza, la scarsa integrazione dei dati e la troppa dipendenza della struttura database dalla sua implementazione fisica. [4] Codd, ricercatore della IBM, ha rivoluzionato il mondo dei DBMS così che quello relazionale, ne è diventato il modello più diffuso; infatti per mezzo di esso è possibile rappresentare efficacemente qualsiasi insieme di dati del mondo reale.

Il modello relazionale è basato sulla teoria degli insiemi e la logica dei predicati di primo grado. È strutturato su una singola struttura dati ovvero la relazione, con precise basi matematiche, che permette di creare una rappresentazione consistente e logica dell'informazione permettendo di separare gli aspetti logici della base di dati dagli aspetti implementativi. Ciò rende la progettazione più semplice e efficiente permettendo di poter progettare la base di dati senza doversi preoccupare degli aspetti dell'implementazione fisica di essa.

Secondo questo modello, i dati di un database relazionale sono immagazzinati in relazioni. Ogni relazione è composta da *tuple* ovvero righe o records, e *attributi* o campi. La struttura fondamentale di un database è data da tabelle. Ogni tabella rappresenta un soggetto unico e specifico, e contiene almeno un campo detto *chiave primaria* che identifica univocamente ogni record. Un campo invece rappresenta una delle caratteristiche del soggetto della tabella alla quale appartiene. Nei campi vengono infatti effettivamente immagazzinati i dati. Ogni campo contiene un unico valore e il suo nome identifica il tipo di valore che rappresenta. Un record invece rappresenta un'unica istanza del soggetto della tabella ed è formato dall'intera serie dei campi di una tabella. Ogni record viene quindi identificato in tutto il database attraverso il valore contenuto nel campo della chiave primaria di quel record.

id	Nome	Cognome	Data di nascita
1	Mario	Rossi	1980-12-09
2	Marta	Fumagalli	1977-08-06
3	Luca	Arnaboldi	1960-04-16
4	Luigi	Ceriani	1981-05-21

Fig 1 - Un esempio di tabella con phpMyAdmin.

L'insieme degli schemi delle relazioni costituisce il modello della base di dati, ovvero la descrizione dei dati memorizzati nella stessa. La totalità dei dati presenti nel sistema è detto invece istanza della base di dati. A differenza dello schema della base di dati che, una volta create le relazioni, è statico, l'istanza della base di dati è un insieme dinamico poiché i dati vengono aggiornati a seconda delle necessità. I dati vengono presentati attraverso le *view* ovvero tabelle virtuali composte da campi appartenenti a una o più tabelle nel database. Le view permettono di vedere le informazioni contenute nel database sotto diverse prospettive tali da permettere una grande flessibilità nel lavoro con i dati. Possiamo infatti creare view che raggruppino solo i dati di campi specifici connettendo anche diverse tabelle in cui i dati sono in collegamento fra loro.

2.3 L'algebra relazionale

L'algebra relazionale consente la manipolazione delle relazioni. Essa è composta da cinque operazioni di base:

- *Selezione*

  La selezione su una relazione R, dato un predicato F su R, indicata con $\sigma_F(R)$, genera una relazione che contiene tutte le tuple di R che verificano F.

- *Proiezione*

  La proiezione di una relazione R su un insieme A={ $A_1, A_2, \ldots, A_m$ }$\subseteq$ U_R di nomi di attributi di R, indicata con $\Pi_{A1,\ldots,Am}$ (R), è una relazione di grado *m* le cui tuple hanno come attributi solo gli attributi specificati in A.

- *Prodotto cartesiano*

  Il prodotto cartesiano di due relazioni R ed S, di grado rispettivamente k_1 e k_2, indicato con R x S, è una relazione di grado k_1+k_2 le cui tuple sono tutte le possibili tuple che hanno come prime k_1 componenti tuple di R, e come ultime k_2 componenti tuple di S.

- *Unione*

  L'unione delle relazioni R ed S, indicata con $R \cup S$, è l'insieme delle tuple che sono in R od in S.

- *Differenza*

  *L*a differenza delle relazioni R ed S, indicata con R – S, è l'insieme delle tuple che sono in R ma non in S.

Tutte queste operazioni vengono utilizzate per la manipolazione dei dati attraverso il linguaggio SQL il quale utilizza appunto l'algebra relazionale.

Si potrebbe approfondire maggiormente il discorso sui DBMS, il modello relazionale e l'algebra relazionale, ma non è questo l'obbiettivo dell'elaborato. Rimando quindi ai capitoli 1 e 2 del libro Catania, Ferrari, Guerrini - Sistemi di Gestione dati, Concetti e Architetture e al capitolo 1 del libro Hernandez, Viescas – SQL Query ai quali si è fatto riferimento se si è interessati ad un'ulteriore ricerca.

3 Gli strumenti di sviluppo

Per l' attuazione del presente elaborato sono stati utilizzati diversi linguaggi al fine di realizzare l'applicativo. Ogni linguaggio ha una sua funzione specifica e svolge un diverso compito all'interno del progetto.

In seguito si analizzeranno brevemente i vari linguaggi presentando le loro caratteristiche principali e accennando alle loro funzionalità basilari che saranno poi utilizzate anche nello sviluppo dell'applicativo.

3.1 SQL

L'SQL, abbreviazione di *Structured Query Language,* nasce nel 1974, ad opera di Donald Chamberlin, come strumento per lavorare con database che si fondano sul modello relazionale.

La versione iniziale si chiamava SEQUEL, in seguito ad essa ci furono varie versioni del linguaggio fino a quando nel 1983 l'IBM rilasciò DB2, ovvero il DBMS relazionale diffuso fino ad oggi. Nel 1986 SQL divenne lo standard per i software utilizzanti il modello relazionale, nonostante il linguaggio fosse ancora lontano dall'essere completo. Negli anni successivi vennero realizzate altre versioni, che furono SQL/89, SQL/92 e SQL/2003. Tale processo di standardizzazione mirava alla consolidazione di un linguaggio che funzionasse su tutti i DBMS. Ciò tuttavia non fu del tutto possibile e i vari produttori implementarono diversi linguaggi con numerose variazioni adottando gli standard ad un livello non superiore al minimo, definito dall'ANSI come Entry Level.

SQL è pertanto il linguaggio per definire e manipolare i dati che vengono utilizzati dai DBMS relazionali oggi disponibili. Questo linguaggio è stato il primo con caratteristiche dichiarative progettato appositamente per l'accesso e la manipolazione dei dati rappresentando così una

tappa fondamentale nello sviluppo delle basi di dati. SQL è basato sull'algebra relazionale, già accennata nel capitolo precedente. La differenza con l'algebra relazionale è che, in questo caso, l'utente specifica solo quale deve essere il risultato della sua interrogazione, caratterizzandolo in maniera dichiarativa senza occuparsi, come per l'algebra relazionale, dell'ordine in cui le operazioni devono essere eseguite formulando l'interrogazione come sequenza di operazioni. SQL pertanto permette all'utente di precisare solo quale deve essere il risultato, attraverso costrutti di programmazione denominati *query*, di un' operazione senza dover indicare l'operazione stessa.

3.1.1 Tipi di dato

In questa sezione si tratta brevemente dei tipi più utilizzati di dato SQL, che verranno anche utilizzati nell'applicativo.

Si distinguono tre grandi macro categorie: *tipi numerici, carattere e temporali.*

Fra i dati di tipo numerico esistono gli *integer* che rappresentano i valori di tipo interi, la loro precisione è espressa in numero di bit o cifre a seconda dell'implementazione in SQL.

Fra quelli di tipo carattere ci sono i *character varying* detti *varchar* che rappresentano stringhe di caratteri di una lunghezza massima definita dall'utente.

Fra i dati di tipo temporale esiste *date* che rappresenta le date espresse in formato anno, mese e giorno con rispettivamente 4 cifre, 2 e 2 per ogni campo.

Un altro dato temporale è *timestamp e* rappresenta il dato date unito con il dato time, che rappresenta il tempo in ore, minuti e secondi. Timestamp è quindi rappresentato da anno, mese, giorno, ora, minuto e secondo.

3.1.2 Creazione di relazioni e chiavi

In SQL per creare relazioni è usato il comando *create table* specificandone lo schema. La sintassi di questo comando è la seguente:

create table <nome relazione> (<specifica colonna>)

Il campo specifica colonna comprende il nome della colonna, il dominio della colonna, che deve essere di un tipo di dato SQL e poi altri possibili valori legati a vincoli di obbligatorietà e chiavi. Le chiavi si indicano in SQL con *primary key* e *unique.* Primary key indica la chiave primaria che può essere al massimo una per ogni relazione, mentre unique indica chiavi alternative, che possono essere una o più per ogni relazione. La chiave primaria specifica che per ogni tupla della relazione i valori siano non nulli e diversi da quelli di ogni altra tupla.

3.1.3 Cancellazione e modifica

L'eliminazione di una relazione è effettuata mediante il comando *drop table.* La sintassi per questo comando è:

drop table <nome relazione> {restrict | cascade}

Nome relazione è appunto il nome della relazione da cancellare, mentre restrict e cascade sono due opzioni. Se viene usato restrict la relazione viene eliminata solamente se non si riferisce ad altre relazioni all'interno del database. Se viene invece usato cascade, l'eliminazione della relazione comporta l'eliminazione anche di tutte le altre relazioni ad essa collegate.

Il comando per l'aggiornamento di una relazione è simile a quello per l'eliminazione anche se il risultato è completamente diverso, la sintassi è la seguente:

alter table <nome relazione> <modifica>

In modifica si possono fare diverse operazioni, tra cui aggiunta (*add*) di colonne specificandone il nome; modifica *(alter)* di una colonna specificata, con l'opportunità di modificare anche il valore di default*;* eliminazione (*drop*) di una colonna specificata con la possibile aggiungere i parametri *restrict* e *cascade* che hanno la stessa funzione nel comando *drop table.*

3.1.4 Interrogazioni

In SQL le interrogazioni vengono formulate con il comando *select* seguito da diverse clausole.

La forma è:

select {distinct $R_{i1}.C_1, R_{i2}.C_2, \ldots, R_{i}.Cn$ *|*} from* R_1, R_2, R_k *where F;*

- $R_i(i=1,\ldots,k)$ è una lista di nomi di relazioni. *Q*uesta è la *clausola from* e specifica le relazioni oggetto dell'interrogazione.

- $R_{ij}.C_{ij}(j=1,\ldots,n, i_j \in 1,\ldots,k)$ è una lista di nomi di colonne, la notazione R.C indica la colonna C della relazione R. Inoltre R_{ij} indica una delle relazioni specificate nella clausola from, mentre * indica che tutte le colonne delle relazioni nella clausola from vanno restituite. Questa è la *clausola di proiezione.*

- F qualifica le tuple delle relazioni che devono essere restituite dall'interrogazione, il formato di F è dato da una combinazione degli operatori AND, OR e NOT. Solo le tuple che verificano F soddisfano l'interrogazione, questa è la *clausola di qualificazione* ed è opzionale.

Questa formula rappresenta la forma base delle interrogazioni; ci sono anche molti altri operatori e funzioni che consentono di specificare un vasto insieme di espressioni e predicati, questo elaborato non ha il compito di analizzarli approfonditamente, ma ne vengono citati solo alcuni tra i più comuni, usati anche per l'applicativo.

Operazione di *join,* che permette di correlare dati memorizzati in relazioni diverse attraversando le associazioni esistenti fra i dati.

Clausola *order by* seguita da una lista di nomi di colonne*,* che permette di ordinare il risultato in maniera diversa, scelta dall'utente. Si può inoltre specificare se ordinare il risultato in modo crescente con l'opzione *"asc"* o decrescente con "*desc"*.

3.2 HTML

HTML (HyperText Markup Language) è un linguaggio di markup utilizzato attualmente per il Web.

È stato sviluppato verso la fine degli anni ottanta da Tim Berners Lee al CERN di Ginvra assieme al protocollo HTTP. Attorno al 1994 ha avuto una forte diffusione in seguito ai primi utilizzi commerciali del web. Nel corso del tempo ha subito molte revisioni, aggiornamenti e cambiamenti per adattarsi allo sviluppo di Internet. Attualmente la versione standard è quella di HTML 4 introdotta nel 1999; ultimamente si sta lavorando per introdurre una nuova versione di HTML, ovvero l'HTML 5.

In generale un linguaggio di markup è costituito da un insieme di regole che definiscono le modalità di rappresentazione di un testo utilizzando convenzioni standardizzate. Questi linguaggi sono caratterizzati dall'uso di marcatori, in questo caso detti *tag* e la loro funzione è quella di indicare come disporre gli elementi all'interno di una pagina. HTML è quindi utilizzato per istruire i browser a visualizzare le pagine Web. Un documento HTML è quindi un file in formato di tipo testo, che contiene alcune parole chiave che indicano al browser le caratteristiche degli oggetti che lo compongono, quali il colore, la dimensione del testo, lo stile delle linee, delle tabelle ecc. Il linguaggio HTML non possiede variabili, cicli, strutture dati particolari, non è neppure in grado di gestire input da parte dell'utente, ma è in grado solo di descrivere i dati da visualizzare. [5] L'unico modo per interagire con l'utente è attraverso i Form, che verranno approfonditi successivamente.

3.2.1 La sintassi

Un documento HTML è dunque un file di testo contenente informazioni da pubblicare e istruzioni incorporate, dette elementi, che indicano come il browser dovrebbe strutturare o presentare il documento. [6] Un elemento è dunque formato da un tag iniziale formato da parentesi angolari aperte e chiuse contenenti il nome del tag ad esempio: <title>, il contenuto riguardante l'elemento per esempio in questo caso: “titolo” e infine il tag finale corrispondente, simile al tag iniziale ma con l'aggiunta di “ / ” prima del nome del tag, quindi seguendo l'esempio: </title>.

Va inoltre precisato che non tutti gli elementi sono strutturati in questo modo, ne esistono alcuni come per esempio
, che inserisce un' interruzione di linea, che sono detti tag vuoti perché non presentano contenuti e non necessitano del tag di chiusura corrispondente.

Il tag iniziale inoltre può contenere attributi che ne modificano il significato; per esempio un attributo di un' immagine può essere: *width*=”100”, che indica che l'immagine avrà una larghezza di 100 pixel. La maggior parte degli attributi è quindi indicata da un nome dell'attributo seguito dal segno uguale e, infine, il valore dell'attributo compreso tra le virgolette.

3.2.2 L'intestazione del documento

La prima riga di un documento HTML è sempre costituita da <!DOCTYPE> che indica la versione di HTML in uso nella pagina corrente. In seguito, il documento è delimitato dai tag <html> . . . </html>, all'interno di esso è formato da una struttura di tipo gerarchico con due parti principali, ovvero l'intestazione e il corpo della pagina. L'intestazione è racchiusa tra i tag <head> . . . </head> e rappresenta tutte le caratteristiche proprie della pagina, come il titolo o le compatibilità della pagina, mentre il corpo del documento, racchiuso dai tag <body> . . . </body> presenta il contenuto della pagina, ovvero quello che verrà visualizzato nel browser.

Le informazioni presenti nell'elemento <head> sono molto importanti poiché vengono utilizzate per descrivere il contenuto del documento stesso, l'intestazione di un documento HTML equivale al titolo o alla copertina di un documento cartaceo [7]. Tuttavia l'elemento <head> può contenere anche altri tipi di informazioni, detti meta-informazioni, queste sono le parole chiave utilizzate dai motori di ricerca per indicizzare le pagine Web. Oltre alle meta-informazioni l'elemento <head> può contenere altre indicazioni, come già accennato precedentemente, come ad esempio il titolo della pagina compreso fra i tag <title> . . . </title>.

L'elemento <title> permette la visualizzazione del titolo nella barra in cima alla finestra del browser; il titolo è obbligatorio e, se non viene inserito dall'utente, i browser tendono ad adottare l'URL del documento come titolo della pagina. Esso è molto importante per una pagina HTML poiché permette ai browser l'indicizzazione della pagina stessa, è, quindi, necessario che sia conciso e significativo.

Nell' <head> possono anche essere inseriti altri elementi, come <script> che permette di incorporare nella pagina Web programmi realizzati in Javascript; oppure l'elemento <style> che racchiude specifiche legate al tipo di carattere, colori o altri aspetti della presentazione del contenuto.

3.2.3 Il corpo del documento

Come accennato nel paragrafo precedente, il corpo del documento, delimitato da <body> e </body>, racchiude il contenuto del documento HTML.

In questa sezione possono essere inseriti una grande varietà di elementi, per esempio quelli che definiscono i blocchi del contenuto strutturale come <p> per creare un nuovo paragrafo o <hn> per i titoli. In questo caso n è compreso fra 1 e 6 e indica la dimensione. All'interno dei blocchi si trovano gli elementi di linea, come <b> per rendere in grassetto una scritta, <u> per sottolinearlo oppure per evidenziare. Ci sono anche altri elementi come <img> per inserire le immagini. Ciascuno di questi elementi può essere modificato da particolari attributi. Per esempio i paragrafi possono essere giustificati con l'attributo *align* seguito da un opportuno valore, le immagini posso essere ridimensionate a piacere con *width* e *height* e così via. All'interno del corpo degli elementi si digita il testo che si vuole far visualizzare dal browser.

3.2.4 Le tabelle

Uno strumento fondamentale in HTML è l'elemento <table> che permette la realizzazione di tabelle per la visualizzazione dei dati. Per la loro realizzazione si utilizzano più elementi, il primo è appunto <table> che viene aperto all'inizio della tabella e chiuso alla fine con </table>. La tabella è formata da righe e colonne, le righe si delimitano con il tag <tr> (table row) che va aperto per ogni nuova riga, al suo interno vanno inseriti i tag <td> (table data), uno per ogni cella. Per evidenziare l'intestazione si usa il tag <th> (table header). Per la strutturazione della tabella esistono vari parametri gestiti dall'attributo align con valori che possono essere: left, center e right, che permettono lo spostamento della tabella nella pagina. Il parametro per lo spessore del bordo è gestito dall'attributo *border* seguito da un valore, se

questo è 0, il bordo è invisibile. I parametri di altezza e larghezza sia per la tabella in generale sia per le righe o le celle sono gestiti dagli attributi *width* e *height* sempre seguiti da un valore. Altri possibili attributi per modificare una tabella sono *cellspacing*, che gestisce la spaziatura tra le celle, e *cellspadding*, che regola la spaziatura tra il bordo della cella e il suo contenuto, entrambi seguiti da un opportuno valore. Per unire orizzontalmente fra loro delle celle è usato l'attributo *colspan*, mentre per unirle fra loro verticalmente si usa *rowspan*. Alla tabella possiamo cambiare altri parametri come il colore di fondo della stessa con l'attributo *bgcolor* o l'immagine dello sfondo con l'attributo *background*.

3.2.5 I collegamenti ipertestuali

Un altro elemento molto importante in HTML è l'ancoraggio <a> che permette di creare un collegamento ipertestuale (link). Questo elemento è formato da alcuni attributi. Quello principale ed essenziale è *href*, ovvero *Hypertest Reference* che ha la funzione di definire la destinazione del collegamento ipertestuale. Il link può essere di una pagina HTML, di un' immagine, di un video, di un file audio o qualsiasi oggetto che deve essere caricato cliccando sul collegamento ipertestuale. All'attributo *href* seguito dall'uguale, va indicato successivamente il percorso del file da aprire, o l'URL se la pagina si trova già sul Web. Dopo *href*, sempre all'interno dei tag <a>, va inserito il testo cliccabile che verrà visualizzato dal browser per aprire il link, il testo può anche essere sostituito da un' immagine usando il tag <img> e inserendo il nome del file; ad esempio .jpg o .png, da aprire.

3.2.6 I Form

I moduli form realizzano l'interazione tra client e server in una pagina HTML permettendo all'utente che visualizza la stessa d' inviare informazioni al server. I moduli sono costituiti da campi che vengono compilati dagli utenti e in seguito inviati a un server per essere elaborati.

I form si inseriscono in un documento HTML con il tag <form> . . . </form>. All'interno del tag ci sono alcuni parametri tra cui *method* e *action, method* specifica come le informazioni contenute nel modulo *form* debbano essere passate al server; esistono fondamentalmente due opzioni differenti: *get* o *post*. La *get* indica che i valori inseriti dall'utente debbano essere

inviati al server separandoli con un punto di domanda dal nome della pagina di destinazione. La post permette di rendere invisibile all'utente la stringa inviata al server, includendola nel body del messaggio. L'attributo *action* invece indica la destinazione del *form*, specificandone l'azione invocata in seguito alla pressione del bottone di invio del *form*. Il "bottone" è costituito mediante l'elemento <input> seguito dall'attributo *type* con diverse specifiche che definiscono in che modo sarà fatto il *form*. *type* può essere di tipo:

- *Text*, in questo modo si permette all'utente di immettere dati dalla tastiera in formato alfanumerico.
- *Radio*, in questo modo si permette all'utente la scelta di una fra diverse alternative.
- *Checkbox*, con questo metodo, invece, si permette all'utente la selezione di più alternative diverse.
- *Submit* è il "pulsante" che invia i dati del form all'indirizzo specificato secondo il metodo indicato.
- *Reset* è il "pulsante" che annulla gli inserimenti e le scelte fatte nel form, ripristinando di fatto i valori di default.

Seguito da *type* c'è l'attributo *name* che assegna il nome al dato inserito dall'utente e successivamente può esserci anche *value*. Quest'ultimo assume valori diversi a seconda del metodo scelto, per esempio per *submit* indica il testo che verrà visualizzato sul pulsante.

3.3 PHP

Il linguaggio PHP (*Hypertext Preprocessor*) risale al 1995, quando Rasmus Lerdof cominciò a scriverne la prima versione, basandosi sulla sintassi del linguaggio Perl. Questo era un noto linguaggio di programmazione ad alto livello, dinamico e procedurale, già introdotto nel 1987 da Larry Wall. Nel 1998 grazie a Andi Gutmans e Zeev Suraski venne introdotta la versione 3.0 di php nella quale comparivano le prime direttive per l'utilizzo di un database. Nel 2000 avvenne il cambiamento radicale con una nuova versione, la 4.0, che introdusse migliorie nella compatibilità con i server web, vennero aggiunte la gestione di sessioni HTML, e divenne un motore in grado di utilizzare molti più costrutti di programmazione. Dal 2008 la

versione 5 diventa l'unica stabile e in continuo sviluppo sostituendo la versione 4.

PHP è dunque un linguaggio di programmazione interpretato, concepito per la programmazione Web e la realizzazione di pagine dinamiche, integrandosi perfettamente con il linguaggio HTML; viene infatti utilizzato, anche nel caso di questo elaborato, per interfacciarsi con il database sul server. A differenza di HTML, in cui il browser Web utilizza i tag e il markup per generare un documento, il codice PHP viene invece eseguito sul server Web, arricchendo e modificando l'output HTML originale. [8] Questo linguaggio è quindi usato sostanzialmente per *server side scripting*, ovvero lo sviluppo di applicazioni Web lato server, anche se offre altre funzioni come lo *shell scripting* ovvero scripting a riga di comando. PHP può essere anche impiegato per la creazione di *applicazioni desktop,* anche se questo ambito di utilizzo non è il più diffuso.

3.3.1 La sintassi

Il linguaggio PHP è caratterizzato da un interprete di righe *(parser)* che esamina il codice PHP all'interno di una pagina. L'inizio dello script è sempre delimitato da un particolare *tag*. Quando il parser lo incontra esegue tutto il codice trovato sino a quando non incontrerà i tag di chiusura, che indicano al parser di tornare alla modalità di visualizzazione. Questo meccanismo permette di inserire codice php all'interno di codice HTML: tutto ciò che si trova all'esterno dei tag php viene lasciato inalterato, mentre tutto ciò che si trova all'interno viene eseguito come codice PHP. [9] Dunque i tag di apertura e chiusura del codice PHP sono i seguenti <?php e ?>. Quando viene richiesta una pagina contenente PHP, è prima elaborato il codice PHP ed eventuali output vengono inviati successivamente al browser richiedente la pagina. Un accorgimento nell'uso del PHP, è che quando si crea una pagina contenente sia codice HTML che PHP l'estensione della pagina dovrà essere .php o il codice PHP non verrà eseguito.

3.3.2 Le variabili

In PHP le variabili sono costituite da un nome che inizia obbligatoriamente con il simbolo del Dollaro $. Il valore della variabile viene assegnato dall'operatore = seguito dal valore della stessa. Per il tipo di variabile non dobbiamo preoccuparci quando la si inizializza, perché essa

assume il tipo del valore assegnatole. Se per esempio scriviamo così $var= “pippo” la nostra $var sarà infatti di tipo string.

In PHP esistono otto tipi di dati, dei quali quattro scalari, due composti e due speciali. I dati di tipo *scalari* sono:

- *boolean* - questi sono variabili booleane e possono assumere solamente due valori, ovvero true o false;
- *integer* - sono le variabili di tipo intero e possono assumere qualsiasi valore dell'insieme numerico Z;
- *float* - le variabili di tipo float possono contenere i numeri floating-point, ovvero a virgola mobile e possono essere dichiarati in forma comune o in notazione esponenziale;
- *string* – una variabile stringa in PHP può contenere un insieme di caratteri a 8bit. La stringa va inserita dopo l'uguale, preceduto dalla variabile, compreso fra singoli (' . . .') o doppi apici (“ . . . “).

I tipi di dati compositi sono:

- *array* – ovvero strutture dati che permettono di raggruppare in un'unica struttura, avente un unico identificativo, un insieme di variabili e di associare inoltre a ogni elemento della struttura un indice per reperire in modo diretto le variabili all'interno dell'array;
- *object* - *q*uesti sono i tipi oggetto utilizzati per la programmazione ad oggetti.

Anche i dati di tipo speciale sono due, ovvero:

- *resource* - questo tipo è un puntatore a una risorsa esterna a PHP;
- *NULL* - Null indica che una variabile contiene un puntatore di valore nullo.

3.3.3 Le strutture di controllo

Secondo il teorema di Jacopini-Bohm un programma strutturato può essere costruito mediante le tre strutture di controllo fondamentali: selezione, iterazione (o loop), istruzione di sequenza (un'assegnazione, una chiamata di funzione, ecc.).[10]

I costrutti del linguaggio PHP derivano gran parte della loro sintassi sostanzialmente dal linguaggio C. Come in C, infatti, la sintassi vuole che le istruzioni terminino con un punto e virgola e inoltre i blocchi di istruzione siano racchiusi fra parentesi graffe. Vengono di seguito analizzati alcuni dei costrutti fondamentali di questo linguaggio.

Il primo è il costrutto di selezione, che permette di effettuare una scelta in base alla quale verranno eseguiti successivamente diversi blocchi di codice. La sintassi del costrutto di selezione (*if*) di PHP è molto simile a quella del C, ovvero è costituita da un *if* seguito da una condizione, se la condizione è verificata esegue una parte di codice, se non lo è, un'altra:

```
if (condizione){
      se la condizione è vera
}
else{
      se la condizione è falsa
}
```

Un altro costrutto fondamentale è il ciclo *while* che si comporta esattamente come nel linguaggio C, ovvero istruisce l'interprete PHP affinché, se la condizione è verificata, esegua ripetutamente le istruzioni in esso racchiuse. Il valore dell'espressione viene verificato ogni volta che il ciclo si ripete così, se il valore cambia durante l'esecuzione dell'istruzione, il ciclo termina solamente alla successiva iterazione. Quindi, quando la clausola non è più verificata, il ciclo termina e non viene più eseguita l'istruzione; ovviamente se la clausola non è verificata fin dall'inizio, l'istruzione non viene eseguita nemmeno una volta.

```
While (condizione){
      istruzioni
}
```

Una variante del ciclo *while* è il ciclo *do ... while*, simile al ciclo while, ma il controllo della condizione viene messo in coda, dopo le istruzioni, in modo da eseguirle almeno una volta e

controllare la condizione solo al termine dell'esecuzione delle stesse.

```
do {
      istruzioni
} while (condizione)
```

L'ultimo ciclo disponibile in PHP è il ciclo *for*. Il comportamento è identico a quello del linguaggio C.

La sintassi del ciclo for è formata dall'inizializzazione di un contatore, da una condizione di uscita del ciclo che controlla il contatore e da un'operazione che agisce sul contatore modificandolo ad ogni ciclo. Anche in questo caso, la condizione controlla il contatore. Se quest'ultimo la verifica, si eseguono le istruzioni, altrimenti si interrompe il ciclo. Alla fine di ogni ciclo, viene eseguita l'operazione sul contatore, modificandolo a seconda dell'operazione impostata. La sintassi è dunque la seguente:

```
for(inizializzazione; condizione; conteggio){
      istruzioni
}
```

Altre strutture molto utili per il PHP sono *switch* e *break.* Switch permette di associare azioni diverse a seconda dei diversi valori assunti da una variabile e quindi eseguire diverse istruzioni a seconda del valore della variabile. Il break serve per interrompere l'esecuzione dello switch e far sì che, terminato un blocco di istruzioni, non passi al blocco successivo. La sintassi è la seguente:

```
switch ($var){
      case "val_1":{
            istruzioni
            break;
      }
      case "val_n":{
            istruzioni
            break;
      }
}
```

3.3.4 Le funzioni

Le funzioni sono uno strumento che permette di ripetere un'operazione quante volte si vuole, scomponendo il programma in sottoprogrammi. In questo modo viene anche facilitata la scrittura del codice rendendolo più "snello" senza dover ripetere parti di codice. Nel linguaggio PHP ci sono fondamentalmente tre tipologie di funzioni ovvero: definite dall'utente, funzioni variabili e funzioni predefinite.

Le funzioni definite dall'utente sono appunto funzioni create dal programmatore all'interno della pagina PHP. Per definire una funzione le si assegna un nome in seguito al parametro *function,* non ci possono essere più funzioni con lo stesso nome all'interno di un programma. Dopo il nome, compresi fra parentesi tonde, ci sono i nomi dei parametri formali separati dalla virgola. Dopo i parametri formali, comprese fra parentesi graffe, ci sono le istruzioni che compirà la funzione. La funzione così definita potrà essere richiamata quante volte si vuole all'interno del programma semplicemente scrivendo il nome della funzione e associandole per esempio il valore di ritorno a una variabile. La sintassi è la seguente:

```
function name (parametri formali separati da virgole){
      istruzioni
      return val_di_ritorno;
}
```

4 Applicativo

In questa sezione si tratterà l'analisi dell'applicativo web, analizzando punto per punto le fasi che ne hanno permesso l'ideazione e la creazione. Per comprendere l'utilità dell'applicativo è necessario fare una breve premessa che illustri le necessità di Radio Cernusco Stereo. L'idea della pagina web che si andrà ad analizzare è nata dal fatto che a RCS è presente un programma chiamato RCS LAB, che si occupa di intervistare ogni settimana un diverso artista emergente e di inserire nella programmazione settimanale un suo singolo. Oltre a questo servizio, già esistente da due anni, si è pensato di raccogliere tutti i brani degli artisti emergenti che sono stati trasmessi nel 2012 e nel 2013, visualizzandoli in una pagina e permettendo così agli ascoltatori di riascoltare quando vogliono tutte le canzoni degli artisti emergenti precedentemente trasmesse. Inoltre si è pensato di introdurre un sistema di votazione, in modo da permettere agli utenti di votare le canzoni preferite, generando così una sorta di playlist ordinata in base alla classifica, con le canzoni degli artisti emergenti più apprezzati. In base a questa classifica, i singoli di maggiore successo verranno premiati dalla Radio, con premi che saranno decisi e gestiti da RCS. In questo modo si offre agli artisti emergenti sia una maggiore visibilità (permettendo di ascoltare nuovamente i loro brani), sia di avere un riscontro sull'apprezzamento del pubblico riguardo le loro canzoni (permettendo agli utenti di poter interagire direttamente con gli artisti esprimendo le loro preferenze). Questa è l'idea di partenza che ha promosso la realizzazione dell'applicativo.

4.1 Il database

Per generare e gestire una classifica contenente informazioni riguardanti gli artisti e le loro canzoni, è stato necessario realizzare un database che contenesse le stesse. Per prima cosa è stato ideato il database sia analizzando le necessità della Radio sia riflettendo sui servizi che si volevano offrire capendo così in che modo organizzare e suddividere i dati. É stato generato

così uno schema logico, presentato in figura 2.

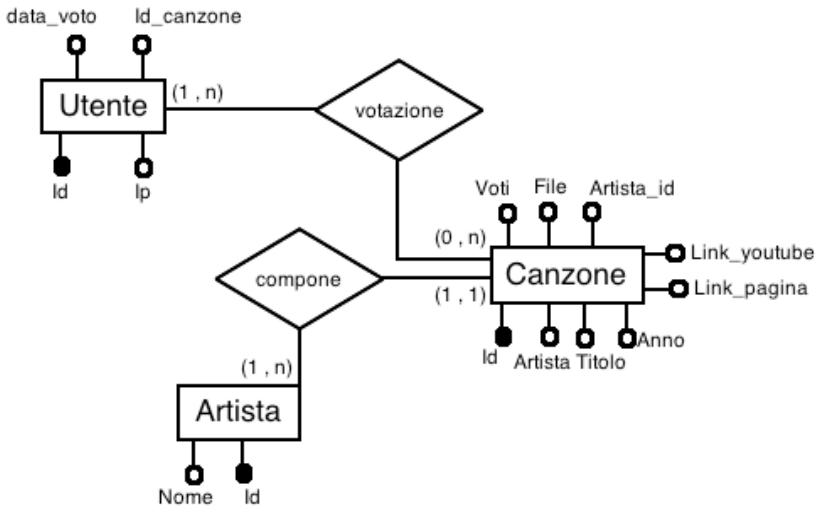


Fig 2 – Schema logico database rcs939_rcslab

Lo schema della base di dati chiamata “rcs939_rcslab” è molto semplice, ci sono tre relazioni: Artista, Canzone e Utente legati fra loro con due associazioni: Votazione: Utente (1,n) ↔ Canzone (0,n) e Compone: Artista (1,n) ↔ Canzone (1,1). L'associazione “Votazione” implica che un utente esiste quando vota da 1 a n canzoni, mentre una canzone può essere votata da 0 fino a n utenti. L'associazione “Compone” implica invece che per ogni artista deve esistere una o più canzoni e ogni canzone che esiste ha un solo artista. Riguardo gli attributi di ogni relazione sono indicati secondo il seguente schema entità relazione:

- Artista (Id, Nome)
- Canzone (Id, Artista, Titolo, Anno, Link_pagina, Link_youtube, Voti, File, Artista_id^{Artista})
- Utente (Id, Ip, Id_canzone^{Canzone}, data_voto)

Gli attributi sottolineati sono le chiavi di ogni relazione. Ogni tupla di tutte le relazioni è quindi identificata da un campo Id, mentre le parole scritte in apice indicano la relazione a cui si riferisce la parola scritta prima della parola in apice. Quindi in questo caso, nella relazione Canzone l'attributo Artista_id si riferisce agli attributi Id della relazione artista, Artista_id è

quindi la chiave esterna di Canzone; la stessa cosa vale per Id_canzone[Canzone].

4.1.1 phpMyAdmin

Il database così ideato è stato successivamente trascritto utilizzando il linguaggio SQL e il programma phpMyAdmin, caricato opportunamente sul server di produzione in hosting che ospita il sito della radio, per permettere successivamente alle pagine del sito di accedere ai dati contenuti nel database.

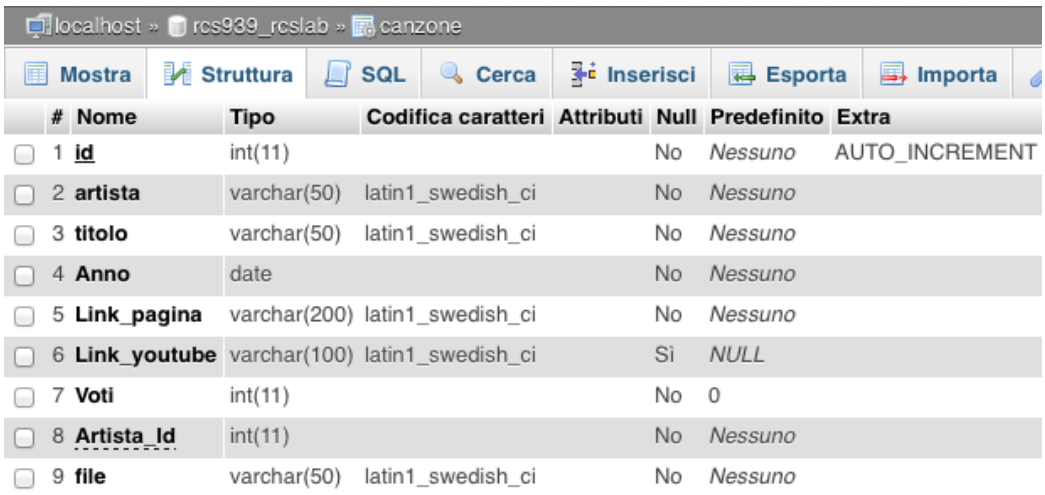
Viene riportato qui di seguito il codice SQL per la creazione delle relazioni.

```
CREATE TABLE IF NOT EXISTS `artista` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `nome` varchar(32) NOT NULL,
  PRIMARY KEY (`id`)
)
CREATE TABLE IF NOT EXISTS `canzone` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `artista` varchar(50) NOT NULL,
  `titolo` varchar(50) NOT NULL,
  `Anno` date NOT NULL,
  `Link_pagina` varchar(200) NOT NULL,
  `Link_youtube` varchar(100) DEFAULT NULL,
  `Voti` int(11) NOT NULL DEFAULT '0',
  `Artista_Id` int(11) NOT NULL COMMENT 'Fkey to Artista',
  `file` varchar(50) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `Id_UNIQUE` (`id`),
  KEY `id_idx` (`Artista_Id`)
)

CREATE TABLE IF NOT EXISTS `Utente` (
  `Id` int(11) NOT NULL AUTO_INCREMENT,
  `Ip` varchar(15) NOT NULL,
  `data_voto` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `Id_canzone` int(11) NOT NULL,
```

```
  PRIMARY KEY (`Id`)
)
```

Con il seguente codice in phpMyAdmin verranno create delle tabelle, come quella mostrata in figura 3, permettendo così successivamente di poter popolare il database inserendo i dati suddivisi nei vari campi.



#	Nome	Tipo	Codifica caratteri	Attributi	Null	Predefinito	Extra
1	id	int(11)			No	*Nessuno*	AUTO_INCREMENT
2	artista	varchar(50)	latin1_swedish_ci		No	*Nessuno*	
3	titolo	varchar(50)	latin1_swedish_ci		No	*Nessuno*	
4	Anno	date			No	*Nessuno*	
5	Link_pagina	varchar(200)	latin1_swedish_ci		No	*Nessuno*	
6	Link_youtube	varchar(100)	latin1_swedish_ci		Sì	*NULL*	
7	Voti	int(11)			No	0	
8	Artista_Id	int(11)			No	*Nessuno*	
9	file	varchar(50)	latin1_swedish_ci		No	*Nessuno*	

Fig 3 – Tabella Canzone

Gli attributi id di ogni relazione sono interi che vengono assegnati automaticamente al momento della creazione di una nuova tupla, sono interi che si incrementano in modo automatico, in questo modo non bisogna preoccuparsi di calcolare un id ma questo viene calcolato e associato automaticamente. Riguardo alle relazioni Canzone e Artista, il contenuto è molto semplice: vengono inserite informazioni utili alla loro catalogazione. Per la relazione canzone, vengono inserite informazioni come il titolo, il nome dell'artista e la data in cui sono state trasmesse dalla radio. Ci sono inoltre due campi: link_pagina e link_youtube in cui vengono inseriti degli URL che permetteranno successivamente di realizzare, sulla pagina web, pulsanti cliccabili che rimandano alle pagine corrispondenti su altri siti. Il campo file contiene invece il nome del file .mp3 che servirà successivamente per costituire il percorso del file e aprire un media player per riprodurre la canzone dal browser degli utenti. Il campo voti viene invece aggiornato automaticamente ogni volta che la canzone riceve un voto, incrementandolo e permettendo così di generare la classifica. La relazione Utente è stata

invece pensata per tenere traccia del voto degli utenti e per immagazzinare i dati in modo da impedire che una persona voti più volte continuamente uno stesso artista. Si è pensato infatti di permettere agli utenti di poter votare una volta al giorno tutte le canzoni che vogliono, ma se vogliono rivotare la stessa canzone devono aspettare il giorno seguente. Per realizzare questa idea, quando un utente vota, si memorizza il suo codice Ip nell'apposito campo Ip, in Id_canzone viene memorizzato l'id della canzone che ha votato, mentre in data_voto la data in formato timestamp, quindi contenente anche l'ora, di quando è stato effettuato il voto. In questo modo grazie ad un controllo, fatto in PHP, che analizzeremo di seguito, si può tenere traccia delle votazioni e permettere di rivotare solo il giorno seguente.

4.2 Le pagine web

In seguito alla creazione del database si è passato alla progettazione delle pagine web.

Per prima cosa sì è deciso di distinguere sul sito della radio una parte pubblica dedicata agli utenti per la votazione degli artisti, la visualizzazione della classifica e l'ascolto delle canzoni e una parte riservata agli amministratori per la gestione dei dati contenuti nel database permettendo funzioni di inserimento, modifica ed eliminazione.

Radio Cernusco Stereo adotta per la gestione del proprio sito internet il CMS (Content Management System) Joomla, che si basa su PHP e MySQL. Per la creazione delle pagine web si sono quindi creati dei moduli scritti in HTML e PHP da integrare successivamente all'interno del sito. Tutto il codice sorgente è stato validato sul sito validator.w3.org.

4.2.1 La classifica e la top10

Per la parte pubblica sono stati pensati, quindi, due moduli simili da integrare in una pagina del sito precedentemente creata. I due moduli servono per visualizzare separatamente la top 10 e la classifica generale contenente tutti gli artisti fra gli anni 2012, 2013 e 2014. Entrambi funzionano allo stesso modo integrando HTML con PHP per visualizzare una tabella dinamica, che si aggiorna con la votazione degli utenti invertendo le righe. Di fianco ad ogni

artista sono inoltre presenti dei pulsanti cliccabili che permettono di aprire il media player della canzone desiderata per ascoltarla, aprire vari link inerenti alla canzone e votare le canzoni stesse.

Qui di seguito viene presentato il codice per il modulo che crea la top 10.

(prima di ogni parte del codice ci sarà un opportuno commento per spiegarne il contenuto.)

// tag di apertura della pagina HTML, della sezione head e la creazione del titolo della pagina.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
      <head>
            <title>TOP10 RCS LAB</title>
            <meta HTTP-EQUIV="Content-Type" CONTENT="text/html;
            charset=windows-1252">
```

// Script per inizializzare la funzione popup che servirà per aprire il media player della canzone desiderata.

```
            <script type="text/javascript">
                  function popUp(URL) {
                        day = new Date();
                        id = day.getTime();
                        eval("page" + id + " = window.open(URL, '" + id +
                        "', '  top=100, left=400,
                        width=480,height=70,');");
                  }
            </script>
```

// chiusura dell'elemento head e apertura della sezione body e inizializzazione della tabella visualizzando i titoli di ogni campo e impostande i colori di sfondo e del font.

```
      <body>

      <table align="center"   cellspacing='0'>
            <tr><th bgcolor='#FF0000' colspan='8'><font color="#FF0000">Top
10</font></th></tr>
            <tr>
                  <th bgcolor='#FFFF00'>Voti</th>
                  <th bgcolor='#FFFF00'>Artista</th>
```

```
                <th bgcolor='#FFFF00'>Titolo</th>
                <th bgcolor='#FFFF00'>Data Pubblicazione  </th>
                <th bgcolor='#FFFF00'><font
                color="#FFFF00">Playerca</font></th>
                <th bgcolor='#FFFF00'><font
                color="#FFFF00">Playerca</font></th>
                <th bgcolor='#FFFF00'><font
                color="#FFFF00">Playerca</font></th>
                <th bgcolor='#FFFF00'></th>
            </tr>
```

// apertura della sezione php e connessione al database rcs939_rcslab.

```
<?php
      $server = 'localhost';
      $user = 'rcs939_rcslab';
      $pass = 'QUgzk13C';
      $db = 'rcs939_rcslab';
      $connection = mysql_connect($server, $user, $pass)
      or die ("Could not connect to server ... \n" . mysql_error ());
      mysql_select_db($db)
      or die ("Could not connect to database ... \n" . mysql_error ());
```

// query SQL per raccogliere le informazioni da visualizzare all'interno della tabella, associandole ad una variabile.

```
      $result = mysql_query("SELECT voti, artista, titolo, anno,
      link_pagina, link_youtube, id, file FROM canzone order by voti desc,
artista limit 10")or die(mysql_error());
```

// loop tra i risultati della query del database, visualizzandoli in tabella.

```
      while($row = mysql_fetch_array( $result )) {
```

// emissione del contenuto di ogni riga nella tabella richiamando ai campi opportuni la funzione di pop up, i link alle pagine desiderate e l'apertura del file voto.php per permettere la funzione di votazione.

```
            echo "<tr>";
            echo '<td> <font color=FF0000>'.$row['voti'].'</font></td>';
            echo '<td> <font color=006699>'.$row['artista'].'</font></td>';
            echo '<td> <font color=006699>'.$row['titolo'].'</font></td>';
```

```
            echo '<td> <font color=006699>'.$row['anno'].'</font></td>';
```

// visualizza il pulsante per aprire la popup, associandogli il percorso del file e aprendolo nella finestra audio.php richiamata nella popup. (audio.php verrà approfondito in seguito.) Tutte i file delle canzoni sono caricati sul server della radio in una apposita cartella.

```
            echo '<td> <a href="#"  onclick="popUp(\'audio.php?id='.
            $row['file'].'\');"><img alt="play" src="play.png"></a></td>';
```

// visualizza un pulsante con il link alla pagina del sito della radio con la recensione della canzone. Il secondo echo visualizza un altro pulsante con il link al video della canzone su Youtube. Grazie a un controllo con un if, se per una canzone non è presente il video, il pulsante sarà grigio e non cliccabile.

```
            echo '<td NOWRAP ><a href="' . $row['link_pagina'] . '"
            target=" _blank" ><img alt="rcs" src="rcs.png"></a></td>';
            if ($row['link_youtube'] == 'errore.html' ){
                  echo '<td NOWRAP>  <img alt="tube"
                  src="BWyoutube.png"></td>';
            }
            else{
                  echo '<td NOWRAP> <a href="' . $row['link_youtube'] . '"
                  target="_blank"> <img alt="tube"
                  src="youtube.png"></a></td>';
            }
```

// visualizza il pulsante che permette la votazione. Il pulsante richiama un altro file: voto.php passandogli una variabile contenente l'id della canzone che l'utente vuole votare. (voto.php verrà approfondito in seguito.)

```
            echo '<td bgcolor=006699 ><a href="votot.php?id=' .
            $row['id'] . '"><button type=submit>VOTA</button></a></td>';
            echo "</tr>";
      }
```

// chiusura della tabella, della sezione PHP, della sezione body e della pagina HTML.

```
            echo "</table></br>";
?>
      </body>
</html>
```

L'output di questo codice sarà la tabella visualizzata in figura 4, ovviamente i dati inseriti sono solo d'esempio poiché il contenuto viene aggiornato continuamente seguendo la votazione del pubblico.

Voti	Artista	Titolo	Data Pubblicazione				
17	Alberto Molon	Ti troverò ancora	2013-06-10		rcs	YouTube	VOTA
8	Virgo	Ti ho visto venire dal mare	2013-10-14		rcs	YouTube	VOTA
5	Alessandro Bosco	Nel mondo che vorrei	2013-03-04		rcs	YouTube	VOTA
5	Andrea Fabiano	Butterfly	2013-06-03		rcs	YouTube	VOTA
5	Andrea La Greca	Saprò	2013-05-06		rcs	YouTube	VOTA
5	Senso Inverso	Disarmata	2013-07-01		rcs	YouTube	VOTA
4	Cheap Wine	Waiting on the door	2013-02-11		rcs	YouTube	VOTA
3	Amelie	Un'Altra Vita	2013-10-07		rcs	YouTube	VOTA
3	Down To Ground	Early In The Morning	2013-05-20		rcs	YouTube	VOTA
3	Kutso	Mara	2013-04-22		rcs	YouTube	VOTA

Fig 4 – tabella che visualizza la top 10

4.2.2 Il player audio

I marcatori <audio> e <video> permettono di inserire risorse multimediali nelle nostre pagine web che possiamo controllare mediante JavaScript. [11]

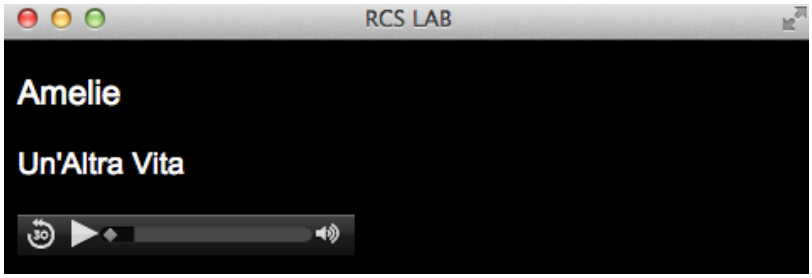


Fig 5 – esempio del player audio

Il seguente codice implementa la pagina che apre il player audio utilizzato per riprodurre le canzoni degli artisti della RCS LSB. Il Player audio è quello predefinito di HTML 5: premendo sul pulsante apposito nella pagina di visualizzazione della classifica, viene passato tramite URL il percorso della canzone corrispondente che si desidera eseguire. Esso viene passato al player che apre la canzone caricandola e permettendone la riproduzione. Viene

inoltre visualizzato anche il nome dell'artista e il titolo della canzone. Qui di seguito ne viene analizzato il codice:

// inizializzazione della pagina HTML indicandone il titolo e apertura del tag body.

```
<!DOCTYPE html>
<html>
	<head>
		<meta HTTP-EQUIV="Content-Type" CONTENT="text/html;
		charset=windows-1252">
		<title>RCS LAB</title>
		<script type="text/javascript">
			defaultStatus = "RCS LAB"
		</script>
	</head>
	<body style="background-color:black;">
```

// apertura del tag PHP, riceve la variabile tramite GET, la salva nella variabile $file e si connette al database

```
<?php
	$file= $_GET['id'];

	$server = 'localhost';
	$user = 'rcs939_rcslab';
	$pass = 'QUgzk13C';
	$db = 'rcs939_rcslab';
	$connection = mysql_connect($server, $user, $pass)
	or die ("Could not connect to server ... \n" . mysql_error ());
	mysql_select_db($db)
	or die ("Could not connect to database ... \n" . mysql_error ());
```

// ottiene i risultati dal database per visualizzare il nome dell'artista e il titolo della canzone che verrà rirprodotta

```
	$result = mysql_query("SELECT artista, titolo FROM canzone where
	file='$file'")or die(mysql_error());
	while($row = mysql_fetch_array( $result )) {
		echo '<p style="font-family:arial;color:white;font-
		size:20px;">' . $row['artista'] . '</p>';
		echo '<p style="font-family:arial;color:white;font-
		size:18px;">' . $row['titolo'] . '</p>';
```

```
        }
?>
```

// corpo della pagina html, visualizzazione del player audio con opportuni parametri per visualizzare il pannello di comandi (controls), impostare l'esecuzione automatica del file audio (autoplay), precaricandolo grazie all'attributo preload con valore “auto” . Viene inoltre associato al player il percorso del file audio. Infine chiusura dei tag della pagina HTML.

```
            <audio src="<?php echo $file; ?>" preload="auto" controls
            autoplay>
            <p style="font-family:arial;color:white;font-size:18px;">Your
            browser does not support the audio element. </p>
        </audio>


        </body>
</html>
```

4.2.3 La votazione

Il seguente codice implementa la funzione di votazione. L'idea di base di questo codice era quella di realizzare una funzione semplice, specialmente per l'utente, per permettergli di votare. Si è deciso infatti di visualizzare un pulsante di fianco ad ogni artista in modo da rendere l'azione del voto legata solamente alla pressione di un pulsante senza dover digitare nessuna informazione come il titolo della canzone o il nome dell'artista. Un altro requisito del sistema di votazione era quello di non permettere all'utente di votare una stessa canzone più volte consecutivamente, rischiando di falsare l'esito della classifica. Si è avuta così la necessità di memorizzare delle informazioni sull'utente per fare un controllo su ogni votazione e impedire un'altro voto dell'utente alla stessa canzone nell'arco della stessa giornata. Si è deciso dunque, per non dover costringere l'utente a rilasciare dati o a registrarsi inserendo per esempio la mail, di memorizzare l'indirizzo ip. Grazie dunque a un controllo incrociato sull'ip dell'utente, sull'ora e giorno di votazione e sull'id della canzone votata, è possibile permettere ad ogni utente di votare tutte le canzoni che vuole, ma fornire per ogni canzone al massimo un voto al giorno. Se un utente cerca di votare senza aver aspettato il giorno seguente, comparirà un messaggio di errore, altrimenti la votazione avverrà con successo. Il seguente codice PHP implementa la seguente funzione:

// inizializzazione del documento HTML, funzione che permette di tornare automaticamente alla pagina della classifica, dopo 3 secondi che si ha votato e inizializzazione della parte PHP con connessione al database.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
      <head>
            <title> VOTO </title>
            <meta HTTP-EQUIV="Content-Type" CONTENT="text/html;
            charset=windows-1252">
            <meta http-equiv="refresh" content="3;
            url=http://newsite.rcs939.it/rcslab/classifica.php">
      </head>
      <body>
<?php
      $server = 'localhost';
      $user = 'rcs939_rcslab';
      $pass = 'QUgzk13C';
      $db = 'rcs939_rcslab';
      $connection = mysql_connect($server, $user, $pass)
      or die ("Could not connect to server ... \n" . mysql_error ());
      mysql_select_db($db)
      or die ("Could not connect to database ... \n" . mysql_error ());
```

// il file voto.php riceve, come precedentemente affermato, una variabile contenente l'id della canzone da votare, questo if controlla se la variabile è presente nell'URL e controlla che sia valida

```
      if (isset($_GET['id']) && is_numeric($_GET['id'])){
```

// ottiene il valore di id, di ip grazie ad una apposita funzione e la data corrente associandola alla variabile $newdata.

```
            $id = $_GET['id'];
            $ip = $_SERVER['REMOTE_ADDR'];
            $a=time();
            $newdata=date('Y-m-d', $a);
```

// fa una query di selezione cercando nella tabella utenti se c'è già una votazione di un certo

utente, controllando l'ip, riguardo una certa canzone, controllando l'id, se esiste ne seleziona la data di votazione e la salva in una variabile $data_voto.

```
$result=mysql_query("select data_voto from Utente where ip='$ip' and
id_canzone='$id'");
$data_voto=mysql_fetch_assoc($result);
```

// fa un controllo grazie ad un if, se la nuova data è successiva alla data della precedente votazione allora aggiorna la tupla riguardante la votazione corrente con la nuova data e stampa a video un messaggio di votazione avvenuta correttamente. Se la nuova data non è maggiore della data corrente visualizza un messaggio di errore.

```
if($newdata > $data_voto['data_voto']){
      mysql_query("UPDATE canzone SET voti=voti+1 WHERE id=$id");
      mysql_query("DELETE FROM Utente WHERE ip='$ip' and
      id_canzone='$id'");
      mysql_query("INSERT INTO Utente(ip, id_canzone) VALUES
      ('$ip','$id')");
      echo "Votazione avvenuta con successo! Grazie per aver votato!
      <br></br>";
}
else{
      echo "Puoi votare al massimo una volta al giorno ogni artista,
      ritorna domani! <br></br>";
}
}
?>

      </body>
</html>
```

4.2.4 Il lato amministrativo

Per il lato amministrativo la necessità era quella di avere delle pagine che permettessero agevolmente la completa gestione dei dati, attraverso funzioni di inserimento, modifica e eliminazione senza dover accedere direttamente al database su phpMyAdmin.

Per prima cosa sono stati creati due moduli distinti, sempre sotto forma di tabella, che mostrassero l'elenco di tutti gli artisti e l'elenco di tutte le canzoni con tutte le informazioni

correlate. Di fianco a ogni campo di ogni tabella sono presenti dei pulsanti per permettere la modifica delle informazioni nei vari campi o l'eliminazione di un intero record. Prima della tabella è anche presente un pulsante per consentire di aggiungere un nuovo record.

Il seguente codice permette la creazione della pagina che gestisce la tabella contenente le canzoni, il codice della pagina degli artisti è molto simile. Anche il procedimento per la visualizzazione della tabella è similare a quello visto precedentemente per il lato pubblico, cambiano solamente alcuni parametri grafici della tabella.

// apertura dei tag della pagina html e head, titolo della pagina e script per permettere l'inserimento di un file css, per abbellire la grafica e permettere ad alcuni pulsanti di cambiare colore al passaggio del mouse; chiusura del tag head e apertura del tag body.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
      <head>
            <meta HTTP-EQUIV="Content-Type" CONTENT="text/html;
            charset=windows-1252">
            <title>AMMINISTRATORI RCS LAB</title>
            <script type="text/javascript" src="slideout.js"></script>
            <link rel="stylesheet" type="text/css" href="slideout.css">
      </head>
      <body>
```

// tag <div . . . > per associare il css ai pulsanti indicati fra i tag <a>...

```
            <div id="dhtmlgoodies_menu">
                  <ul>
                        <li><a href="view_artista.php">Vai a
                        Artisti</a></li>
                        <li><a href="new_canzone.php">Aggiungi una nuova
                        Canzone</a><br></li>
                  </ul>
            </div>
```

// inizializzazione della tabella che mostrerà l'elenco delle canzoni, inserendo i titoli di ogni campo.

```
            <table border  cellspacing='0'>
                  <tr><th bgcolor=FF0000 colspan='12'>Elenco Canzoni RCS
                        Lab</tr>
                  <tr>
                        <th bgcolor=FFFF00>Id</th>
                        <th bgcolor=FFFF00>Artista</th>
                        <th bgcolor=FFFF00>Titolo</th>
                        <th bgcolor=FFFF00>Data Pubblicazione</th>
                        <th bgcolor=FFFF00>File</th>
                        <th bgcolor=FFFF00></th>
                        <th bgcolor=FFFF00></th>
```

```
                </tr>
```

// inizio del codice PHP e connessione al database.

```
<?php
      $server = 'localhost';
      $user = 'rcs939_rcslab';
      $pass = 'QUgzk13C';
      $db = 'rcs939_rcslab';
      $connection = mysql_connect($server, $user, $pass)
      or die ("Could not connect to server ... \n" . mysql_error ());
      mysql_select_db($db)
      or die ("Could not connect to database ... \n" . mysql_error ());
```

// query al database che seleziona i campi desiderati associandoli ad una variabile; loop fra i risultati della query e emissione del contenuto di ogni riga nella tabella e infine chiusura della tabella, dei tag PHP, del body e della pagina HTML.

```
      $result = mysql_query("SELECT id, artista, titolo, anno, file FROM
      canzone order by anno desc")or die(mysql_error());

      while($row = mysql_fetch_array( $result )) {

            echo "<tr>";
            echo '<td>' . $row['id'] . '</td>';
            echo '<td>' . $row['artista'] . '</td>';
            echo '<td>' . $row['titolo'] . '</td>';
            echo '<td>' . $row['anno'] . '</td>';
            echo '<td>' . $row['file'] . '</td>';
            echo '<td bgcolor=006699 ><a href="edit_canzone.php?id=' .
            $row['id'] . '"><button type=submit>EDIT</button></a></td>';
            echo '<td bgcolor=006699 ><a href="delete_canzone.php?id=' .
            $row['id'] . '"><button type=submit>DELETE</button></a></td>';
            echo "</tr>";
      }
      echo "</table>";
?>

      </body>
</html>
```

L'output di questo codice permette la visualizzazione della pagina in figura 6:

Vai a Artisti
Aggiungi una nuova Canzone

Elenco Canzoni RCS Lab						
Id	**Artista**	**Titolo**	**Data Pubblicazione**	**File**		
214	Davide Papasidero	Adesso Oppure Mai	2014-02-03	songs/Davide_Papasidero_Adesso_Oppure_Mai.mp3	EDIT	DELETE
215	Vanessa Marchi	Mare Mosso	2014-01-27	songs/Vanessa_Marchi_Mare_Mosso.mp3	EDIT	DELETE
217	Io Drama	Vergani Marelli 1	2014-01-20	songs/Io_Drama_Vergani_Marelli_1.mp3	EDIT	DELETE
212	Lorenzo Malvezzi	Test Psicoattitudinale	2014-01-13	songs/Lorenzo_Malvezzi_Test_Psicoattitudinale.mp3	EDIT	DELETE
213	Monia Emme	Come il Vento	2014-01-06	songs/Monia_Emme_Come_Il_Vento.mp3	EDIT	DELETE
210	Camera 133	La Distrazione	2013-12-30	songs/Camera_133_La Distrazione.mp3	EDIT	DELETE
172	Silvio Brambilla	Magnifico Messere	2013-11-18	songs/Silvio_Brambilla_Magnifico_Messere.mp3	EDIT	DELETE
209	Adriano Tarullo	Come Volevi Tu	2013-11-04	songs/Adriano_Tarullo_Come_Volevi_Tu.mp3	EDIT	DELETE
211	Vena	La Grinta	2013-10-28	songs/Vena_La_Grinta.mp3	EDIT	DELETE
147	Virgo	Ti ho visto venire dal mare	2013-10-14	songs/Virgo_Ti_Ho_Visto.mp3	EDIT	DELETE
151	Amelie	Un'Altra Vita	2013-10-07	songs/Amelie_Un_Altra_Vita.mp3	EDIT	DELETE
93	Matrioska	Come mi vuoi	2013-09-30	songs/Matrioska_Come_Mi_Vuoi.mp3	EDIT	DELETE
108	Emil	Il Concetto di Concetta	2013-09-23	songs/Emil_Il_Concetto_di_Concetta.mp3	EDIT	DELETE

Fig 6 – Lato amministrativo, visualizzazione canzoni

In questo codice vengono richiamati altri tre diversi file che svolgono diverse funzioni: new_canzone.php, edit_canzone.php e delete_canzone.php. Nella pagina di visualizzazione artisti si richiamano tre file simili: new_artista.php, edit_artista.php, delete_artista.php.

Questi tre file svolgono diverse funzioni e verranno analizzati in seguito.

4.2.5 L'inserimento di un nuovo record

Per inserire un nuovo record è stato creato un file chiamato new_canzone.php per le canzoni e new_artista.php per gli artisti. Si analizzerà ora il codice di new_canzone.php:

// inizializzazione della funzione renderForm che serve per visualizzare il form di inserimento dei dati. apertura dei tag html, sezione head, inserimento del titolo, chiusura dell'head e apertura della sezione body.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<?php

      function renderForm($artista, $titolo, $anno, $link_pagina,
      $link_youtube, $artista_id, $file){
?>
<html>
      <head>
            <title>Nuova Canzone</title>
            <meta HTTP-EQUIV="Content-Type" CONTENT="text/html;
charset=windows-1252">
            <script type="text/javascript" src="slideout.js"></script>
            <link rel="stylesheet" type="text/css" href="slideout.css" >
```

```
</head>
<body>
```

// attraverso un form con metodo post vengono visualizzati a schermo dei campi vuoti, correlati da titolo, che permettono all'utente di inserire i nuovi dati per la creazione di un nuovo record. Per confermare di voler inserire i dati è presente un pulsante di invio che salva i dati all'interno di variabili corrispondenti per ogni campo.

```
<div id="dhtmlgoodies_menu">
      <ul>
            <li><a href="view_canzone.php">Torna a
            Canzoni</a><br></li>
      </ul>
</div>
<br>
<font>Aggiungi una nuova Canzone compilando tutti i
campi</font>
<br><br>
<form method="post" action="new_canzone.php">
      <b>Artista: </b> <input size="40" type="text"
      name="artista" value="<?php echo $artista; ?>" ><br>
      <b>Titolo: </b> <input size="40" type="text"
      name="titolo" value="<?php echo $titolo; ?>" ><br>
      <b>Data Pubblicazione (aaaa-mm-gg): </b> <input size="10"
      type="text" name="anno" value="<?php echo $anno; ?>"><br>
      <b>Link RCS LAB: </b> <input size="80" type="text"
      name="link_pagina" value="<?php echo $link_pagina;>"><br>
      <b>Link Youtube: </b> <input size="80" type="text"
      name="link_youtube"value="<?php echo$link_youtube;>"><br>
      <b>Artista ID: </b> <input size="10" type="text"
      name="artista_id" value="<?php echo $artista_id;?>" ><br>
      <b>File: </b> <input size="50" type="text" name="file"
      value="songs/<?php echo $file; ?>"><br>
      <input type="submit" name="Invio">
      <input type="reset" value="Annulla">
</form>
```

// subito dopo l'apertura del tag PHP si chiude la funzione renderForm e poi c'è un controllo, che controlla se il pulsante di invio è stato premuto. Dopodiché si connette al database.

```
<?php
      }
      if (isset ($_POST['Invio'])) {
            $artista=$_POST['artista'];
            $titolo=$_POST['titolo'];
            $anno=$_POST['anno'];
            $link_pagina=$_POST['link_pagina'];
            $link_youtube=$_POST['link_youtube'];
            $artista_id=$_POST['artista_id'];
            $file=$_POST['file'];

            $server = 'localhost';
```

```
            $user = 'rcs939_rcslab';
            $pass = 'QUgzk13C';
            $db = 'rcs939_rcslab';
            $connection = mysql_connect($server, $user, $pass)
            or die ("Could not connect to server ... \n" . mysql_error ());
            mysql_select_db($db)
            or die ("Could not connect to database ... \n".mysql_error());
```

// controllo che verifica che tutti i campi siano stati compilati, se è così recupera i valori inseriti in ogni variabile, e crea una nuova tupla con i dati inseriti dall'utente, altrimenti visualizza il modulo di inserimento dati, senza cancellare quelli già compilati e visualizza un messaggio di errore chiedendo di inserire i dati mancanti. Se l'inserimento è avvenuto con successo viene infine visualizzata un nuovo modulo vuoto per l'inserimento e una frase di inserimento avventuto con successo.

```
            If ($artista == '' || $titolo == '' || $anno == '' ||
                  $link_pagina == '' || $link_youtube == '' || $artista_id
                  == '' || $file == '')

                  renderForm($artista, $titolo, $anno, $link_pagina,
                  $link_youtube, $artista_id);

                  echo '<div style="padding:4px; border:1px solid red;
                  color:red;"> ERRORE: Per favore, compila tutti i campi
                  presenti nel modulo! </div>';
            else{
                  $result=mysql_query("INSERT INTO canzone (`id`,
                  `artista`, `titolo`, `anno`, `link_pagina`,
                  `link_youtube`, `voti`,`artista_id`, `file` ) VALUES
                  (NULL, '$artista', '$titolo', '$anno', '$link_pagina',
                  '$link_youtube', '0', '$artista_id', '$file') ");

                  renderForm();

                  echo '<div style="padding:4px; border:1px solid green;
                  color:green;">Inserimento avvenuto con successo!</div>';
            }
      }
```

// Questo else è la condizione iniziale della pagina che visualizza un form vuoto con i campi da compilare.

```
      else{
            renderForm($artista, $titolo, $anno, $link_pagina,
            $link_youtube, $artista_id, $file);
      }
?>
      </body>
</html>
```

(In seguito ci sarebbe dell'altro codice che visualizza l'elenco di tutti gli artisti per recuperarne l'id, ma è molto simile ai codici precedentemente illustrati per la visualizzazione delle tabelle.)

L'output di questo codice è mostrato in figura 7 alla pagina seguente.

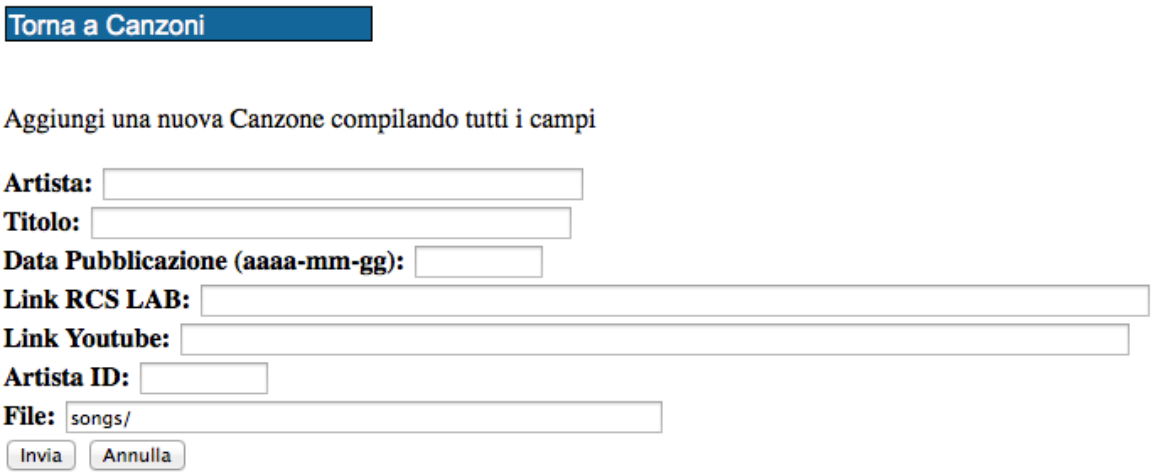


Fig 7 – Modulo per l'inserimento di un nuovo record

4.2.6 La modifica di un record

Anche per la modifica di un record è stato creato un file chiamato edit_canzone.php per le canzoni e edit_artista.php per gli artisti. La differenza è che quando queste pagine vengono richiamate tramite il link, gli viene passato l'id della canzone o artista da eliminare. Si analizzerà ora il codice di edit_canzone.php:

// dopo l'apertura del tag PHP inizializza una funzione che comprenderà il form html per modificare i dati, questo è stato fatto perchè il form viene richiamato più volte. In seguito vine chiuso il tag PHP.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<?php
      function renderForm($id, $artista, $titolo, $anno, $link_pagina,
      $link_youtube, $artista_id, $file, $error){
?>
```

// apertura del codice HTML con la parte dell'head e body con pulsante per tornare alle canzoni.

```
<html>
	<head>
		<title>Modifica Canzone</title>
		<meta HTTP-EQUIV="Content-Type" CONTENT="text/html;
		charset=windows-1252">
		<script type="text/javascript" src="slideout.js"></script>
		<link rel="stylesheet" type="text/css" href="slideout.css" >
	</head>
	<body>

		<div id="dhtmlgoodies_menu">
			<ul>
				<li><a href="view_canzone.php">Torna a
				Canzoni</a><br></li>
			</ul>
		</div>
```

// inizializzazione del form per la modifica dei dati. Il form è simile a quello per l'inserimento di un nuovo record, la differenza è che questo verrà visualizzato precompilato con i dati già inseriti da modificare.

```
	<form action="" method="post">
			<input type="hidden" name="id" value="<?php echo $id; ?
>">
			<div>
				<p><strong>ID:</strong> <?php echo $id; ?></p>

				<strong>Artista: </strong> <input size="40"
				type="text" name="artista" value="<?php echo
				$artista; ?>"><br>

				<strong>Titolo: </strong> <input size="40"
				type="text" name="titolo" value="<?php echo
				$titolo; ?>"><br>

				<strong>Data Pubblicazione (aaaa-mm-gg): </strong>
				<input size="10" type="text" name="anno" value="<?
				php echo $anno; ?>"><br>

				<strong>Link RCS LAB: </strong> <input size="80"
				type="text" name="link_pagina" value="<?php echo
				$link_pagina; ?>"><br>

				<strong>Link Youtube: </strong> <input size="80"
				type="text" name="link_youtube" value="<?php echo
				$link_youtube; ?>"><br>

				<strong>Artista ID: </strong> <input size="10"
				type="text" name="artista_id" value="<?php echo
				$artista_id; ?>"><br>

				<strong>File: </strong> <input size="50"
				type="text" name="file" value="<?php echo $file; ?
				>"><br>
```

```
                        <input type="submit" name="submit" value="Invio">
                        <input type="reset" value="Annulla">
                    </div>
            </form>
      </body>
</html>
```

// inizializzazione di un modulo PHP che visualizza un messaggio di errore quando viene richiamato.

```
<?php
      if ($error != ''){
            echo '<div style="padding:4px; border:1px solid red;
color:red;">'.$error.'</div>';
      }
?>
```

// chiusura della funzione inizializzata precedentemente subito dopo l'apertura del tag PHP, connessione al database e verifica se il modulo è stato inviato. Se lo è, inizia a elaborare il modulo e lo salva nel database

```
<?php
      }

      $server = 'localhost';
      $user = 'rcs939_rcslab';
      $pass = 'QUgzk13C';
      $db = 'rcs939_rcslab';

      $connection = mysql_connect($server, $user, $pass)
      or die ("Could not connect to server ... \n" . mysql_error ());
      mysql_select_db($db)
      or die ("Could not connect to database ... \n" . mysql_error ());

      if (isset($_POST['submit'])){
```

// verificare che il valore di 'id' sia un intero valido prima di ottenere i dati del modulo.

```
            if (is_numeric($_POST['id'])){
```

// ottiene i dati del modulo e verifica che siano validi.

```
                  $id = $_POST['id'];
                  $artista=mysql_real_escape_string
                  (htmlspecialchars($_POST['artista']));
                  $titolo=mysql_real_escape_string
                  (htmlspecialchars($_POST['titolo']));
                  $anno=mysql_real_escape_string
                  (htmlspecialchars($_POST['anno']));
                  $link_pagina=mysql_real_escape_string
                  (htmlspecialchars($_POST['link_pagina']));
                  $link_youtube=mysql_real_escape_string
                  (htmlspecialchars($_POST['link_youtube']));
                  $artista_id=mysql_real_escape_string
                  (htmlspecialchars($_POST['artista_id']));
                  $file=mysql_real_escape_string
                  (htmlspecialchars($_POST['file']));
```

// controlla che tutti i campi siano stati compilati.

```
if ($artista == '' || $titolo == '' || $anno == '' ||
$link_pagina == '' || $link_youtube == '' || $artista_id
== '' || $file == '' ){
```

// se non lo sono genera un messaggio di errore e visualizza un nuovo modulo.

```
$error = 'ERRORE: Per favore, compila tutti i campi
presenti nel modulo!';
renderForm($id, $artista, $titolo, $anno,
$link_pagina, $link_youtube, $artista_id, $error);

}
```

// se non c'è errore salva i dati nel database e visualizzazione del messaggio di corretta modifica.

```
else{
      mysql_query("UPDATE canzone SET artista='$artista',
      titolo='$titolo', anno='$anno',
      link_pagina='$link_pagina',
      link_youtube='$link_youtube',
      artista_id='$artista_id',
      file='$file' WHERE id=$id ")or die(mysql_error());
      header("Location: view_canzone.php");

      renderForm($id, $artista, $titolo, $anno,
      $link_pagina, $link_youtube, $artista_id, $file,
      $error);

      echo '<div style="padding:4px; border:1px solid
      green; color:green;">
      <font color=green>Modifica avvenuta con
      successo.</font></div>';
}
}
```

// altrimenti se l' 'id' non è valido, viene visualizzato un altro errore. Altrimenti se il modulo non è stato inviato, ottengo i dati dal db e visualizza nuovamente il modulo.

```
else{

      echo 'Error!';
}
}
else{
```

// ottiene il valore 'id' dall'URL (se esiste), assicurandosi che sia valido (controlla che sia numerico/maggiore di 0), esegue una query e associa i risultati a una variabile.

```
if (isset($_GET['id']) && is_numeric($_GET['id']) &&
$_GET['id'] > 0){
      $id = $_GET['id'];
```

```
                $result = mysql_query("SELECT artista, titolo, anno,
                link_pagina, link_youtube, artista_id, file FROM canzone
                WHERE id=$id ")
                or die(mysql_error());
                $row = mysql_fetch_array($result);
```

// verifica che l' 'id' corrisponda a una riga nel database e ottiene i dati dal db.

```
                if($row){
                      $artista = $row['artista'];
                      $titolo = $row['titolo'];
                      $anno = $row['anno'];
                      $link_pagina = $row['link_pagina'];
                      $link_youtube = $row['link_youtube'];
                      $artista_id = $row['artista_id'];
                      $file = $row['file'];
```

// visualizza il modulo.

```
                      renderForm($id, $artista, $titolo, $anno,
                      $link_pagina, $link_youtube, $artista_id, $file,
                      '');
                }
                else{
```

// se non corrisponde visualizza: Nessun risultato.

```
                      echo "Nessun risultato!";
                }
          }
          else{
```

// se l' id nell'URL non è valido, o se non vi è alcun valore di id, visualizza un errore. Infine chiude il tag PHP.

```
                echo 'Errore!';
          }
    }
?>
```

L'output del seguente codice è visualizzabile in figura 8.

Fig 8 – Modulo per la modifica di un record

4.2.7 L'eliminazione di un record

L'ultimo modulo è quello di eliminazione, anche in questo caso abbiamo un file delete_canzone.php e delete_artista.php. Anche per queste pagine, quando vengono richiamate viene loro passato l'id del record da eliminare. Si analizza ora il codice di delete_canzone.php:

// Inizializzazione file HTML con head e visualizzazione del bottone per tornare alle canzoni.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>
      <head>
            <title>Elimina Canzone</title>
            <meta HTTP-EQUIV="Content-Type" CONTENT="text/html;
            charset=windows-1252">
            <script type="text/javascript" src="slideout.js"></script>
            <link rel="stylesheet" type="text/css" href="slideout.css" >
      </head>
      <body>

            <div id="dhtmlgoodies_menu">
                  <ul>
                        <li><a href="view_canzone.php">Torna a
                        Canzoni</a><br></li>
                  </ul>
            </div>
```

// apertura del tag PHP e connessione al database

```
<?php
```

```
$server = 'localhost';
$user = 'rcs939_rcslab';
$pass = 'QUgzk13C';
$db = 'rcs939_rcslab';
$connection = mysql_connect($server, $user, $pass)
or die ("Could not connect to server ... \n" . mysql_error ());
mysql_select_db($db)
or die ("Could not connect to database ... \n" . mysql_error ());
```

// controlla che la variabile id sia impostata nell'URL e che sia valida.

```
if (isset($_GET['id']) && is_numeric($_GET['id'])){
```

// ottiene il valore id.

```
$id = $_GET['id'];
```

// chiede se si è sicuri di voler eliminare il record e chiusura del codice PHP.

```
echo "Sei sicuro di voler eliminare il record?<br>";
?>
```

// Form con pulsante per confermare l'eliminazione del record.

```
<form action="" method="post">
        <input type="submit" name="submit" value="YES">
</form>


</body>
</html>
```

// se si è sicuri di voler eliminare il file e si è quindi premuto il pulsante, controllando l'operazione con un if, si elimina il record e viene visualizzato un messaggio di eliminazione avvenuta correttamente.

```
<?php
if (isset($_POST['submit'])){


$result = mysql_query("DELETE FROM canzone WHERE id=$id")
or die(mysql_error());


      echo '<div style="padding:4px; border:1px solid green;
      color:green;"><font color=green>Eliminazione avvenuta con
      successo.</font></div>';

}
}
?>
```

5 Conclusioni e sviluppi futuri

Attraverso gli argomenti analizzati in questo scritto, è stato possibile evidenziare le singole funzionalità dei diversi linguaggi, e come esse cooperino per la realizzazione di un qualsiasi applicativo. Questa è solo una delle molteplici funzionalità dei linguaggi, infatti esse sono molto più estese e permettono di offrire svariate funzioni che non sono state prese in considerazione in questo elaborato. I linguaggi utilizzati, inoltre, sono in continua evoluzione, mantenendosi alla pari con lo sviluppo del Web e garantendo nuove implementazioni per adeguarsi alle nuove tecnologie e agli standard futuri. L'applicativo sviluppato per la radio potrà così venir successivamente aggiornato, adeguandosi alle esigenze della stessa. Con la pubblicazione online, Radio Cernusco ha deciso di premiare gli artisti più votati inserendoli nelle playlist settimanali dedicate alla musica emergente dando loro una possibilità per essere riascoltati. Al momento della pubblicazione, la pagina ha avuto successo, con una buona partecipazione degli ascoltatori della radio, grazie anche alla condivisione della notizia sulla pagina Facebook e Twitter della radio. Sì pensa che questo progetto possa essere molto utile sia per la RCS, per ottenere maggiori visualizzazioni rispetto al sito grazie a nuovi possibili ascoltatori (invitati a votare dagli stessi artisti emergenti trasmessi dalla Radio), ma soprattutto per gli artisti emergenti per continuare ad essere trasmessi e diffondere la propria musica. Grazie anche all'aiuto e al supporto dei social network si spera che questa pagina possa ottenere un grande numero di visualizzazioni in modo da ricambiare l'impegno e il lavoro che svolgono ogni giorno i volontari della RCS. Pensando al futuro della Radio Cernusco Stereo, un' ulteriore implementazione potrebbe essere, ad esempio, quella di estendere il sistema di votazione, non solo agli artisti emergenti, ma a tutte le canzoni mandate in onda, includendo così anche gli artisti famosi. Un'altra possibilità sarebbe quella di creare un podcast, permettendo agli utenti di riascoltare un intero programma. L'obiettivo futuro è quindi quello di offrire un servizio che permetta agli ascoltatori, oltre che di seguire la diretta dal sito, di poter riascoltare qualsiasi trasmissione e canzone inserita nella programmazione; in questo modo la radio, attraverso il sito stesso, diviene interattiva con gli utenti. Al giorno d'oggi, infatti, l'offerta musicale è maggiore rispetto agli anni in cui la radio

era uno dei pochi mezzi che consentivano l'ascolto della musica. Oggi, quest'ultima è diventata “portatile” ed esistono numerosi servizi che permettono di ascoltare la musica desiderata al momento desiderato. Le radio, per restare al passo con i tempi, devono quindi offrire programmi di interazione con gli ascoltatori e adeguarsi alle tecnologie odierne.

Bibliografia

[1] rcs939.it: *Storia* www.rcs939.it/vita-rcs/storia.html

[2] Barbara Catania, Elena Ferrari, Giovanna Guerrini: *Sistemi di Gestione Dati Concetti e Architetture* De Agostini Scuola 2006

[3] [8] Michele E. Davis, John A. Phillips: *Programmare in PHP e MySQL* Tecniche Nuove, Milano 2008

[4] Michael J. Hernandez, John L. Viescas: *SQL Query* Mondadori Informatica 2001

[5] Paolo Camagni, Riccardo Nikolassy: *I linguaggi del Web, HTML, CSS, Javascript, VBScript, ASP, PHP* Hopeli Editore Milano 2009

[6] [7] Thomas A. Powell: *HTML La Reference* McGraw Hill 2001

[9] [10] Paolo Camagni, Riccardo Nikolassy: *PHP Dall'HTML allo sviluppo di siti web dinamici* Hopeli Editore Milano 2005

[11] Gabriele Gigliotti: *HTML 5 sviluppare oggi il web di domani* Apogeo s.r.l. 2012