Università di Pisa

Dipartimento di Scienze della Terra
Corso di Laurea Magistrale in Scienze Ambientali

Progetto e realizzazione di un sistema client-server per la gestione di dati ondametrici

Relatore: Dott. Ing. Lorenzo Cappietti

Controrelatore: Dott. Giandomenico Mastroeni

Candidato: Davide Samuele Franchi

Anno Accademico 2014/2015

Indice

1. Introduzione

1.1 Motivazione del lavoro svolto

I dati ondametrici possono derivare da vari fonti, come una rete ondametrica (fig 1.4) o da simulazioni numeriche di hindcasting o forecatsing.

Una rete ondametrica è un sistema composto da boe (fig 1.2) e sensori (fig 1.3)che permettono l'acquisizione di dati sullo stato del mare e delle condizioni meteorologiche a contorno

I modelli di hindcasting sono costruiti sulla base di misure storiche di determinati parametri e servono per ricostruire lo stato del mare nel passato .

I modelli di forecasting si basano sulle previsioni meteorologiche allo scopo di predire lo stato del mare in una determinata area. (fig 1.1).

Tali dati non possono essere lasciati a loro stessi, ma dovranno subire tutta una serie di procedure che, in conclusione, li renderanno disponibili al pubblico per la loro visione e successive elaborazioni.

Qualsiasi sia la fonte dei dati diventa di fondamentale importanza la loro validazione e gestione finalizzata all'utilizzo pratico nel campo delle problematiche ambientali o ingegneristiche progettuali.

In ambito ambientale, lo sviluppo di una procedura di gestione di dati meteo-oceanografici si rende di fondamentale importanza per comprendere la dinamica della costa e operare nella direzione giusta per quanto riguarda la gestione del litorale [1]

Dal punto di vista ingegneristico i dati acquisiti archiviati e elaborati sono utili nella progettazione delle opere di protezione del litorale, per la comprensione dell'efficacia delle opere stesse e per la definizione dello stato locale del mare, allo scopo di rendere più sicura la balneazione e la navigazione.

Normalmente una rete ondametrica o un modella numerico restituiscono dati relativi all'altezza d'onda (H), alla temperatura del mare (T) e alla direzione di propagazione del moto ondoso (D).

L'insieme di queste tre tipologie di dati ci permettono di descrivere dettagliatamente lo stato locale del mare e le conseguenze nella morfodinamica della costa.

I dati che si propone di gestire sono:

- data della registrazione ;
- ora della registrazione;
- Hmo (altezza d'onda significativa);
- Hmax (altezza d'onda massima);
- Tp (periodo d'onda di picco);
- Tm (periodo medio dell'onda) ;
- Dmt (direzione media di propagazione);
- Dmp (direzione media di picco);
- Dmw (direzione media delle windwaves);
- Tmp (temperatura dell'acqua).

La trattazione della definizione e caratterizzazione dei precedenti parametri verrà effettuata nel capitolo 3.

Il sistema completo dovrà permettere ad un operatore di interrogare i dati in tempo reale oppure di accedere ad un archivio storico, con la possibilità di visualizzazione, analisi e memorizzazione.

Le procedure sviluppate potranno avere seguito per l'attività redazionale in fase progettuale di ulteriori opere di difesa in situ o del loro miglioramento.

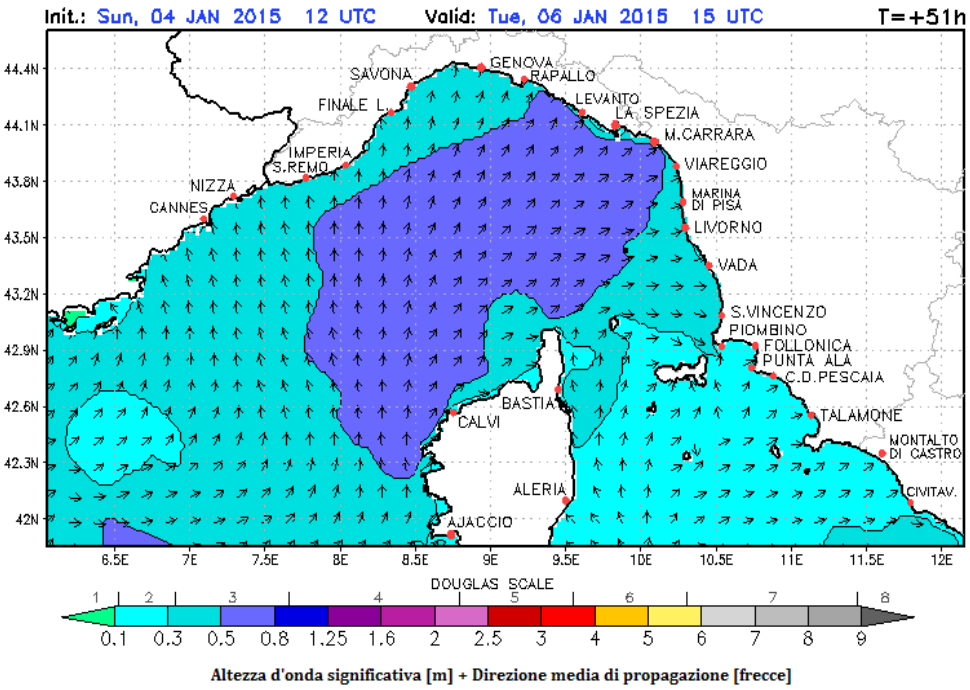

Figura 1.1: Modello di previsione dell'altezza d'onda e direzione media di propagazione nel Mar Ligure relativa alle ore 15 del 06/01/2015

La totalità del lavoro svolto si è inserito all'interno del tirocinio curriculare del corso di laurea di Scienze Ambientali dell'Università di Pisa.

Figura 1.3: Boa ondametrica usata dalla Rete Nazionale Ondametrica

Figura 1.2: Rete Ondametrica Nazionale

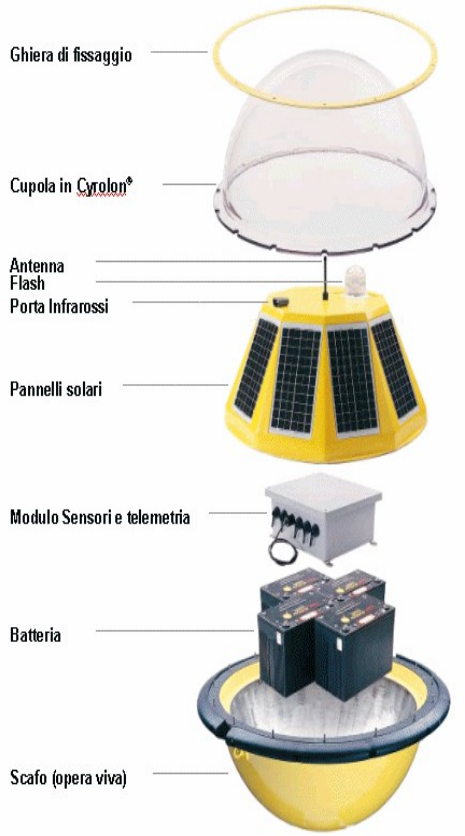

Figura 1.4: Schema di composizione della boa con i relativi sensori

1.2 Obiettivo

L'obiettivo di questa tesi, consiste nello sviluppare una procedura operativa basata su un software con interfaccia web che permette la gestione dei dati derivanti da fonti generiche come boe ondametriche o modelli di simulazione numerica. Lo sviluppo del software permetterà agli utenti di avere a disposizione dati ondametrici via web che saranno organizzati in tabelle e grafici.

2.Definizione e descrizione del moto ondoso

Nell'ambiente costiero, il moto ondoso da vento è il fenomeno dominante per quanto riguarda una scala temporale che va dalle osservazioni giornaliere a dati registrati nell'arco di una ventina di anni.

Descriveremo il fenomeno utilizzando prima un approccio qualitativo e successivamente uno quantitativo che lo caratterizzi attraverso equazioni matematiche.

2.1Analisi qualitativa

Localmente la superficie del mare in quiete è assimilabile a un piano di direzione orizzontale, cioè risulta essere perpendicolare rispetto alla direzione verticale (quella assunta da un filo a piombo), la quale dipende solamente dall'attrazione di gravità.

Nel momento in cui un evento perturba lo stato di quiete, si crea un'oscillazione di masse in cui la gravità tende a riportare verso il basso le particelle liquide che raggiungono un'altezza maggiore.

Tale oscillazione forma le cosiddette gravity waves, che si muovono per effetto della ridistribuzione della pressione.

Si definisce quindi come moto ondoso un movimento fortemente irregolare in cui la superficie del mare si alza e si abbassa periodicamente rispetto al livello di quiete (SWL: Still Water Level).

Considerando un punto P in mare e η_P come posizione del livello della superficie libera nel punto P in funzione del tempo, otteniamo il seguente grafico (fig. 2.1), dove η_P nel tempo è misurata con una determinata frequenza di campionamento.

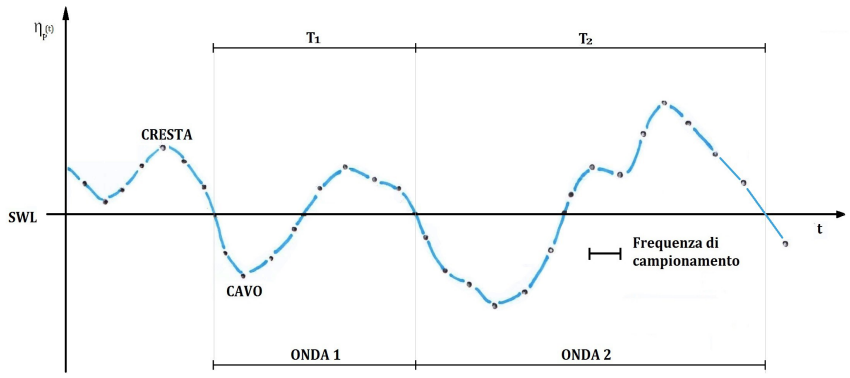

Figura 2.1: Esempio di un moto ondoso misurato su un ondametro

In questo grafico un'onda è la porzione del fenomeno che intercorre tra due attraversamenti dello SWL nella stessa direzione. Si può quindi notare come la funzione η_P sia irregolare nel tempo; il punto di massima altezza di ogni onda è chiamato "cresta", quello più basso è chiamato "cavo".

Il tempo che intercorre tra due attraversamenti di 0 è detto periodo (T) e il suo inverso è la frequenza (f), mentre il dislivello tra una cresta e il cavo di un onda è detta altezza dell'onda (H).

I valori della successione dei periodi e delle altezze costituiscono un processo di tipo stocastico, ovvero non è possibile prevedere il valore successivo; il moto ondoso è quindi irregolare o random, nel senso che la successione degli eventi non è prevedibile.

2.1.1 Cause del moto ondoso

I fenomeni che possono causare il moto ondoso sono vari, tra cui:

- oggetto che cade su un bacino in stato di quiete. Questo genera onde in tutte le direzioni che formano un moto di tipo propagatorio.
- evento sismico. Può causare o lo spostamento del fondale marino, o la caduta di un costone roccioso, assimilabile al punto precedente.
- maree dovute all'attrazione gravitazionale dei corpi celesti vicini alla Terra, sopratutto Luna e Sole. Le maree lunari hanno un periodo di 12,4 ore; in genere la marea raggiungerà il suo massimo nella regione della Terra rivolta verso la Luna, mentre dalla parta opposta avremo un fenomeno di alta marea, ma che raggiunge altezze inferiori.

- variazione della pressione atmosferica: crea le maree bariche, o maree meteorologiche. Se si ha una diminuzione di pressione in un settore di mare, per esempio dovuta all'arrivo di una perturbazione, nell'area di bassa pressione il livello del mare si alza di un'altezza Δh. La forza di gravità e la pressione tendono a equilibrare la spinta che arriva dalla zona a più alta pressione, provocando in questo modo la formazione delle onde.

Per esempio una perturbazione in una determinata area del Mediterraneo porta a una situazione di bassa pressione, che verrà riequilibrata con un innalzamento del livello del mare. In prima approssimazione questo fenomeno può essere schematizzato come in figura 2.2

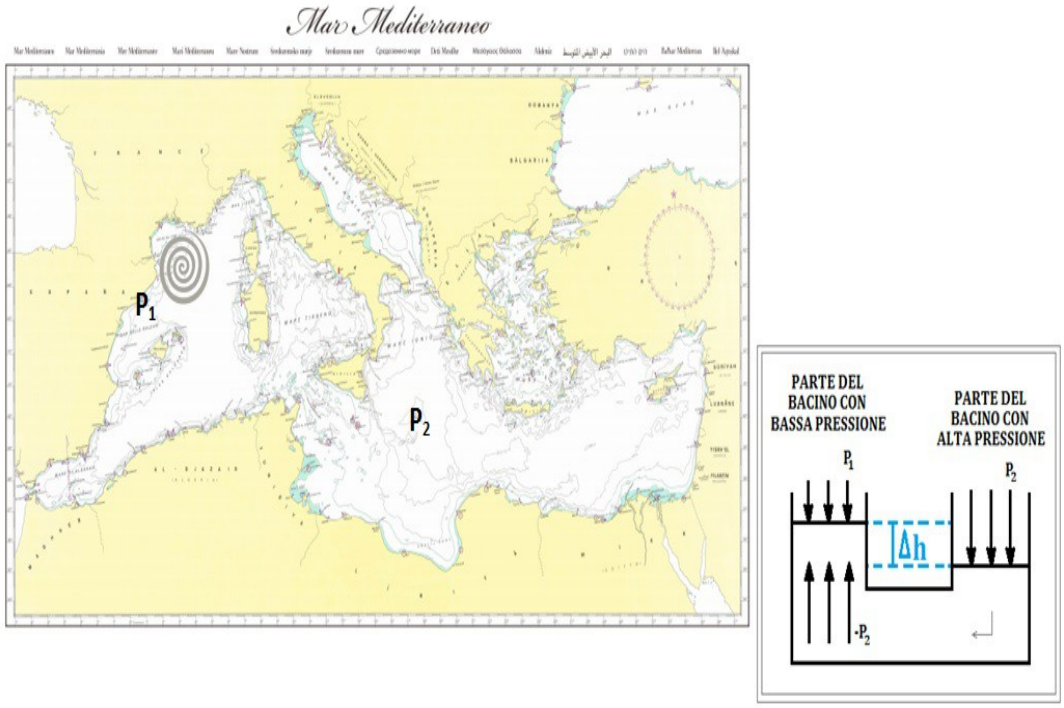

Figura 2.2: Perturbazione nel bacino del Mediterraneo e relativa approssimazione del fenomeno.

$x = \gamma \cdot \Delta h$

$P_1 + \gamma \cdot \Delta h = P_2$

$\Delta h = \frac{P_2 - P_1}{\gamma}$ x è la pressione esercitata da una colonna d'acqua di altezza Δh, che pesa circa 1000 kg/m^3 = γ (peso specifico).

Esempio: se $P_1 - P_2 = 1\,mbar = 10^{-3}\,kg/cm^2$

e considerando che: $\gamma = 1000\,kg/m^3 = 10^{-3}\,kg/cm^3$

$$\Delta h = \frac{10^{-3}\,kg/cm^2}{10^{-3}\,kg/cm^3} = 1\,cm$$

Quindi uno squilibrio barico di 1 mbar provoca un innalzamento del mare di 1 cm.

Frequentemente si osservano minimi barici di una decina di mbar e raramente anche di 50-60 mbar, come per esempio durante l'evento di dicembre 2013.

Nella figura 2.3 si osserva un brusco abbassamento di pressione tra il 25 e il 28 dicembre, correlato con un innalzamento del livello del mare negli stessi giorni (fig. 2.4).

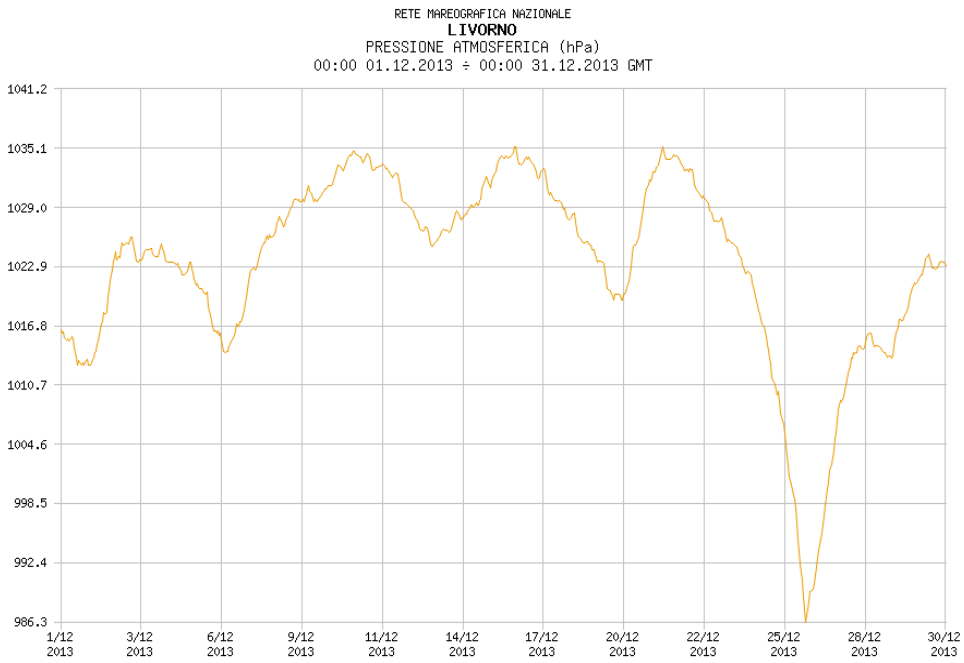

Figura 2.3: variazione della pressione atmosferica nel mese di Dicembre 2013

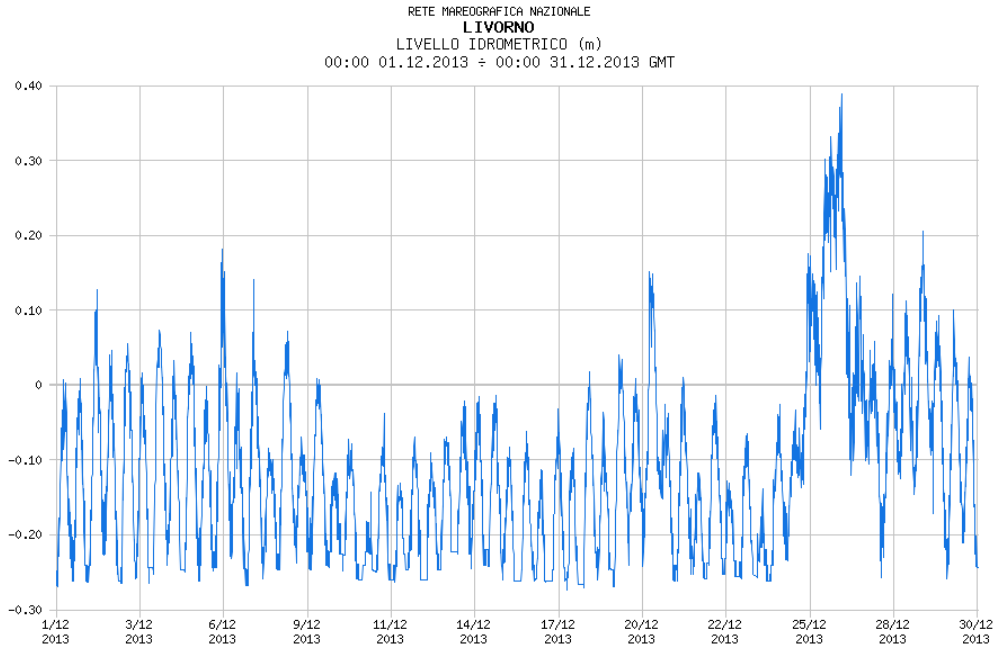

Figura 2.4:variazioni del livello idrometrico nel mese di dicembre 2013.

- vento.: è la causa più importante di formazione delle onde (*sea waves*), le quali nel campo del regime dei litorali hanno maggiore importanza perché sono le più frequenti e riversano sulla costa il maggior quantitativo di energia media all'anno (fig. 2.5).

Date le precedenti considerazioni, prenderemo in esame solamente i moti ondosi formati dall'azione del vento, poiché riversano sulle coste la maggior energia media annua determinando la morfodinamica dei litorali e la sollecitazione delle opere a protezione dei litorali in orizzonti temporali lunghi.

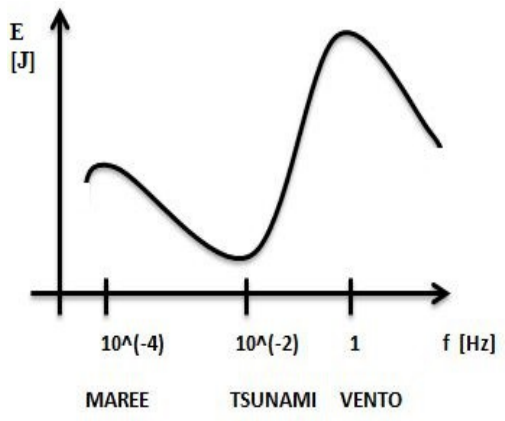

Figura 2.5: Energia riversata dalle diverse tipologie di onde

2.1.2 Formazione del moto ondoso per azione del vento

Il vento esercita un'azione tangenziale alla superficie del mare, compiendo un lavoro sulle particelle della superficie.

La velocità del vento solitamente diminuisce abbassandosi di quota, come da figura 2.6. Si nota un andamento costante a quote più alte, fino al cosiddetto *strato limite dell'atmosfera*, dalla cui quota il vento diminuisce gradualmente la sua intensità.

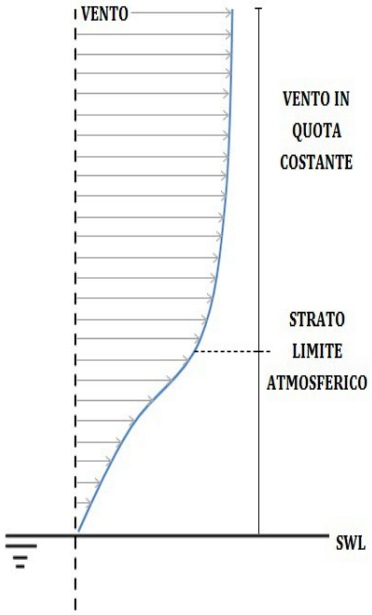

Figura 2.6: Profilo di velocità del vento

Si osservano varie fasi al perdurare dell'azione del vento:

I fase: si nota una prima increspatura della superficie, con formazione di *ripples* (fig. 2.7), che aumentano la superficie a disposizione dell'azione del vento.

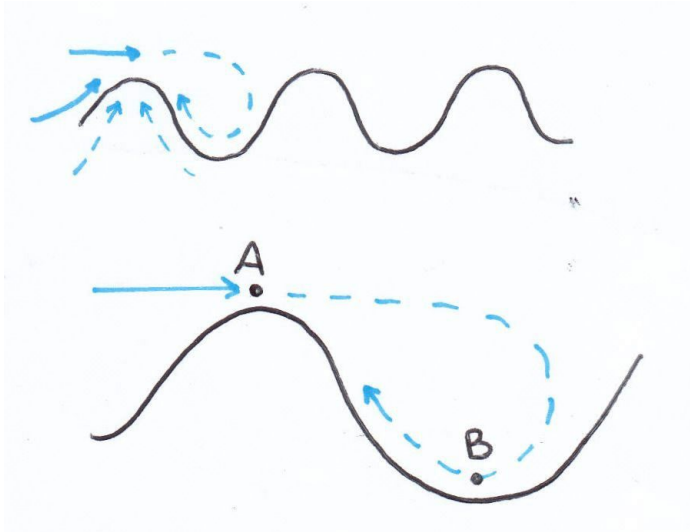

Figura 2.7: fasi dell'ingrossamento delle onde

II fase: sulle ripples osserviamo due fenomeni.

- sottovento nelle ripples il vento soffia in senso contrario a causa di una formazione vorticosa che trascina particelle liquide dai cavi alle creste.
- se indichiamo con A un punto in prossimità della cresta e con B un punto in prossimità del cavo (con A e B sulla stessa linea di corrente), abbiamo che la pressione in A risulta minore della pressione in B; infatti secondo Bernoulli

  il trinomio $z+\frac{v^2}{2g}+\frac{P}{\gamma}$ è costante per due punti.

  Nel nostro caso infatti

$$z_A+\frac{v_A^2}{2g}+\frac{P_A}{\gamma}=z_B+\frac{v_B^2}{2g}+\frac{P_B}{\gamma} \quad ,$$

  da cui:

$$P_A=(zB-zA)\cdot\gamma+\gamma\cdot\left(\frac{v_A^2-v_B^2}{2g}\right)+P_B$$

($z_B - z_A$) è un numero negativo dell'ordine di (-)10^{-2}m; il termine di velocità è un numero positivo dell'ordine di (-) 10^{-1}, quindi P_A è minore di P_B. Questo fenomeno si somma contribuendo così alla formazione di onde più grandi.

III fase: l'onda si muove con velocità *c* (*celerità*), nella stessa direzione del vento che ha velocità *v*; quando c = v, i due fenomeni precedenti smettono di agire, non si ha più vento apparente e lo scambio di energia tra vento e mare scompare. L'intensità del vento è fortemente correlata all'intensità dell'agitazione ondosa e a un vento più intenso corrisponde una maggior agitazione ondosa.

Il mare raggiunge il massimo stato d'agitazione quando il vento soffia abbastanza a lungo, con intensità adeguata e la parte di mare su cui soffia ("fetch") è abbastanza estesa.

In questo caso si parla di mare completamente sviluppato ("fully developed sea"); spesso il vento cessa prima che il mare raggiunga il suo massimo di agitazione e si parla di moto ondoso limitato nel tempo ("time limited").

L'area interessata da una perturbazione può essere idealmente suddivisa in sotto-aree.

Le onde generate in ognuna di queste sotto-aree si propagano e prima o poi si sommano, quindi si genera un moto ondoso più intenso rispetto a quello che si genererebbe se il vento soffiasse in una sola di queste sotto aree.

Si ha quindi una correlazione diretta tra l'intensità dell'agitazione del mare e le dimensioni dell'area di fetch.

Il "fetch geografico" è il fetch che realmente insiste sul paraggio in esame (tratto di costa in esame), è limitato dalle formazioni morfologiche della costa. Il "settore di traversia" è il settore angolare riferito a un paraggio, che contiene tutte le direzioni possibili dalle quali possono provenire, dal largo, le onde (fig. 2.8). Il sotto-settore che comprende le onde più violente è detto "settore di traversia principale", dove solitamente arrivano i venti più intensi e frequenti.

Figura 2.8: Schema di un fetch geografico

Se non si dispone di dati ondametrici o anemometrici si deve stimare l'area di fetch partendo dai modelli. È dimostrato che oltre un valore massimo di estensione dell'area di fetch, un suo ulteriore aumento non incrementa l'agitazione del mare. Quindi se l'area di fetch stimata è maggiore di questo valore massimo, si prende in considerazione il valore massimo. Si parla quindi di "fetch efficace".

2.2Analisi quantitativa

I parametri legati all'agitazione del mare per l'ottenimento di un modello analitico, sono:

- U_{10} = velocità del vento a 10 metri di quota
- f = area di fetch
- t = durata del vento

In funzione di questi tre parametri posso definire l'altezza H: $H \propto (U_{10}, f, t)$.

2.2.1Analisi statistica a breve termine nel dominio del tempo

Poniamo il caso di avere un andamento di η (variazione rispetto allo SWL) nel tempo fornito da un ondametro (fig. 2.9). Volendo definire un parametro sintetico di intensità di questo moto ondoso, deve essere riferito a una qualche media rispetto a un certo orizzonte temporale.

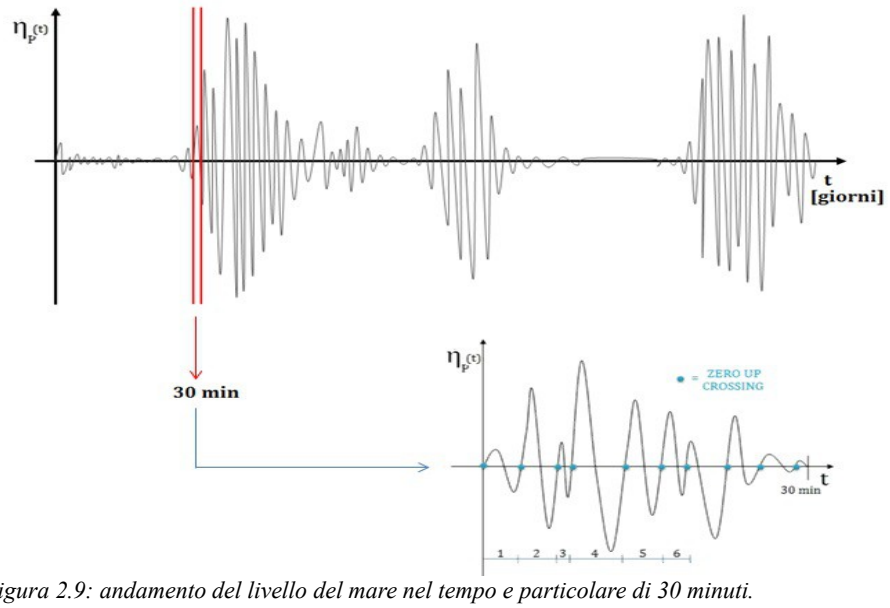

Figura 2.9: andamento del livello del mare nel tempo e particolare di 30 minuti.

Se l'intervallo di tempo fosse troppo lungo non si noterebbe la variazione del fenomeno (per esempio se mediassi stati di moto con stati di quiete); se Δt fosse troppo piccolo non avrei sufficienti dati per un'analisi statistica.

Le reti ondametriche internazionali usano quindi un Δt di circa 30 minuti[39].

Per questo motivo questo tipo di analisi si dice a breve termine.

La singola onda è quella che si trova tra un punto di zero-up crossing e il successivo (fig. 2.9).Ogni 30 minuti[39] quindi ottengo n onde, per ognuna delle quali posso definire l'altezza (H_i) e il periodo (T_i).

#	H	T
1	H1	T1
2	H2	T2
3	H3	T3
4	H4	T4
...	...	...
m	Hm	Tm

Tabella 2.1: numero delle onde e loro caratteristiche

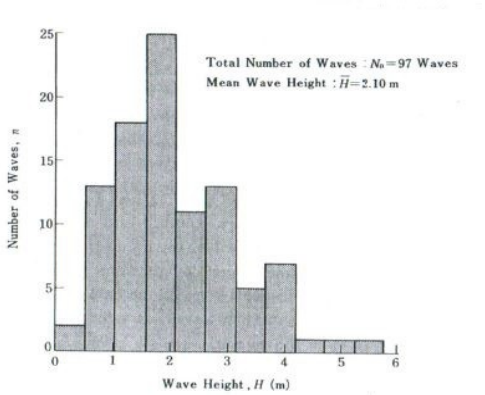

Figura 2.10: distribuzione in classi di altezza delle onde

Sulla base di questi dati possiamo ricavare per esempio l'onda di massima altezza, il valore medio e una distribuzione in classi di altezza delle onde (vedi tabella 2.1 e fig. 2.10).

Sulle ordinate di figura 2.10, n indica il numero di onde la cui altezza ricade in una determinata classe, mentre sulle ascisse, H corrisponde alle diverse classi di altezza, che hanno ampiezza ΔH.

A questo punto, per ottenere la distribuzione della densità di probabilità, si modificano le variabili degli assi, in modo da ottenere, sulle ascisse $\frac{H}{\overline{H}}$ (altezza rispetto alla media) e sulle ordinate (fig. 2.11).

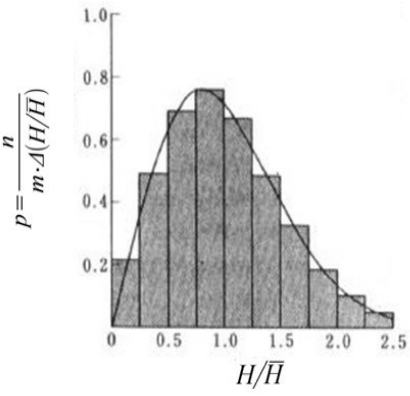

Figura 2.11: distribuzione della densità di probabilità

Questa distribuzione ha sempre la stessa forma per tutti i moti ondosi in tutte le parti del mondo e si descrive con la **Funzione di Rayleigh:**

$$P=\frac{\pi}{2}\cdot\frac{H}{\overline{H}}\cdot e^{-\frac{\pi}{4}\cdot\left(\frac{H}{\overline{H}}\right)^2}$$

Osservando quindi la funzione si nota che per caratterizzarla basta conoscere il valore di $\bar{H}$.

Tradizionalmente si utilizza un parametro, chiamato H_s (altezza significativa).

Prima dell'avvento delle misure strumentali questo parametro derivava da stime a vista fatte dai marinai semplicemente osservando lo stato d'agitazione del mare, nei confronti di una scala graduata.

Quando la misura di H_s è fatta a vista si usa indicarla con H_v. Con lo sviluppo di strumentazioni moderne atte alla misurazione del livello del mare, si è cercato di legare operativamente H_v con l'altezza derivabile dalle misure ondametriche.

Per raggiungere questo scopo, le altezze d'onda H_i vengono ordinate per altezze decrescenti (H_i^*). Si dividono i valori di H in N gruppi, facendo la media della N-esima partizione che contiene le altezze maggiori.

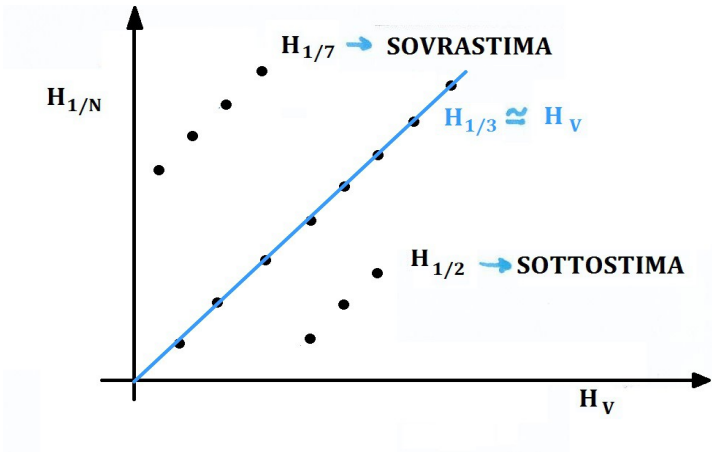

Figura 2.12: correlazione tra Hv e H1/N

Da una correlazione tra H_v e $H_{1/N}$ è stato stimato che le due misure coincidono per N=3 (fig 2.12). Per questo motivo l'altezza significativa ottenuta tramite l'analisi zero-crossing, può essere calcolata come $H_{1/3}$. Questa misura sarà sempre maggiore rispetto a $\bar{H}$, perché si considerano soltanto le onde di altezza maggiore.

C'è una correlazione diretta tra H_s e $\bar{H}$, in quanto: $H_s = 1{,}4\ \bar{H}$

L'altezza massima delle onde, in mari infinitamente profondi, può raggiungere un valore fino a 2 volte l'H_s, secondo la correlazione: $\left(\frac{H_{max}}{H_{1/3}}\right)_{MOD} \approx 0{,}7\sqrt{\ln(m)}$.

N.B. : il $T_{1/3}$, cioè il periodo significativo si calcola dalla media dei periodi associati alle altezze d'onda utilizzate per calcolare $H_{1/3}$.

2.2.2Analisi statistica a breve termine nel dominio della frequenza

L'analisi statistica a breve termine nel dominio della frequenza si basa sullo studio dello spettro in frequenza del moto ondoso, che vedremo essere la distribuzione in frequenza di un valore calcolato sulla base delle ampiezze delle sue componenti armoniche.

Avendo a disposizione una registrazione ondametrica al variare del tempo, la si può scomporre, tramite Trasformata di Fourier, come somma di funzioni sinusoidali, con periodo e ampiezza noti (fig. 2.13).

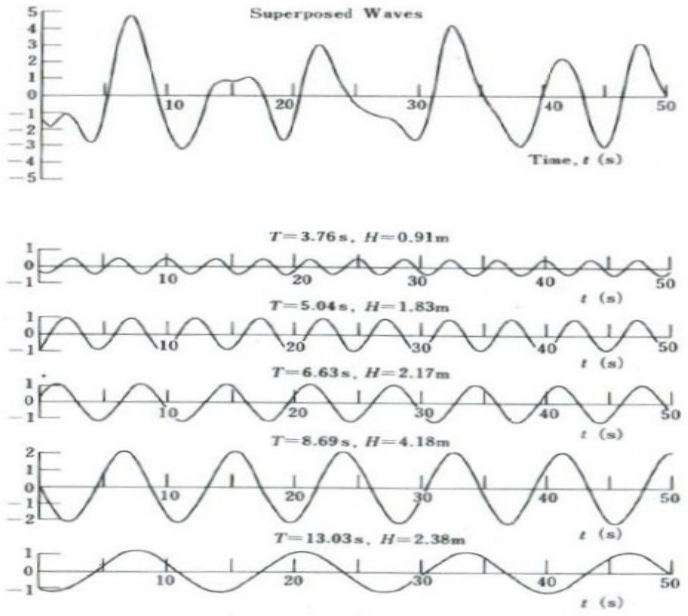

Figura 2.13: onda scomposta nelle sue componenti armoniche

Le varie componenti armoniche sono:

$$\eta_1 = \frac{H_1}{2} \cdot \cos\left(\frac{2\pi}{T_1} \cdot t + \varphi_1\right)$$

$$\eta_2 = \frac{H_2}{2} \cdot \cos\left(\frac{2\pi}{T_2} \cdot t + \varphi_2\right)$$

$$\eta_m = \frac{H_m}{2} \cdot \cos\left(\frac{2\pi}{T_m} \cdot t + \varphi_m\right)$$

La componente armonica con ampiezza maggiore prevale rispetto alle altre componenti, in quanto l'onda risultante dalla sommatoria di tutte le componenti armoniche riprende la forma d'onda della componente prevalente, mentre le altre componenti costituiscono una sorta di disturbo sull'onda.

Per ottenere la funzione che descrive l'onda di partenza, bisogna quindi fare la sommatoria delle funzioni che descrivono le componenti armoniche:

$$\eta_p = \sum_{i=1}^{m} \frac{H_i}{2} \cdot \cos\left(\frac{2\pi}{2} \cdot t + \varphi_i\right)$$

Dove:

- $H_i/2 = a_i$, cioè l'ampiezza dell'onda iesima;
- $1/T_i = f_i$, cioè la frequenza dell'onda iesima;
- $2\pi * f_i = \omega_i$, cioè la frequenza angolare o pulsazione.

Per svolgere questo tipo di analisi si parte dalla registrazione ondametrica e si suddivide in intervalli di 30 minuti[39], come per l'analisi zero-crossing.

Ogni intervallo viene ulteriormente ripartito in sotto intervalli di circa 1 minuto (K sotto intervalli). Ogni sotto intervallo j (j= 1, …, K) si scompone con Trasformata di Fourier, ottenendo in questo modo le varie a_{ij}(fig. 2.14).

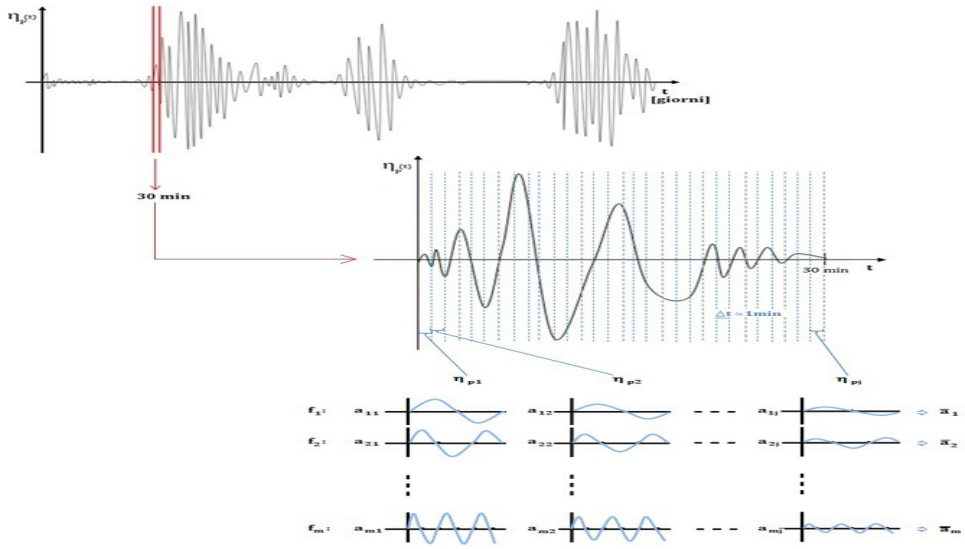

*Figura 2.14:*Scomposizione del moto in intervalli di circa 30 min. e successivamente in intervalli di circa 1 min.. Ogni sotto-intervallo è scomposto nelle componenti armoniche con trasformata di Fourier. Per ogni classe di frequenza si ottiene così l'ampiezza media.

Se si pone sulle ascisse la frequenza e sulle ordinate la media di K valori alla frequenza i-esima, (calcolata come $a_i = \sum_{j=1}^{K} a_{ij}$) si ottiene la distribuzione in frequenza delle ampiezze (fig. 2.15).

La frequenza che corrisponde al picco di questa distribuzione è detta "frequenza di picco" (f_p), mentre il periodo è chiamato "periodo di picco" (o periodo significativo, T_p).

Lo spettro in frequenza è così definito perché è una quantità direttamente proporzionale all'energia meccanica trasportata da quella frequenza.

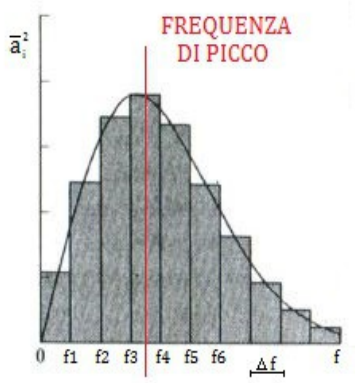

Figura 2.15: spettro dell'ampiezza

Per definire la funzione spettro in frequenza (fig. 2.16) si procede modificando la variabile sulle ordinate, in modo da ottenere:

$$\frac{\bar{a}_i^2}{(2 \cdot \Delta f)}$$

e si fa il passaggio al limite, per Δf che tende a 0.

$$S(f) = \lim_{\Delta f \to 0} \frac{\bar{a}_i^2}{2 \cdot \Delta f}$$

L'integrale della funzione S(f) da 0 a infinito, è detto m_0, momento di ordine 0 della

funzione S(f), da cui posso calcolare H_s, come

$$H_{m_0} = 4 \cdot \sqrt{m_0} \quad .$$

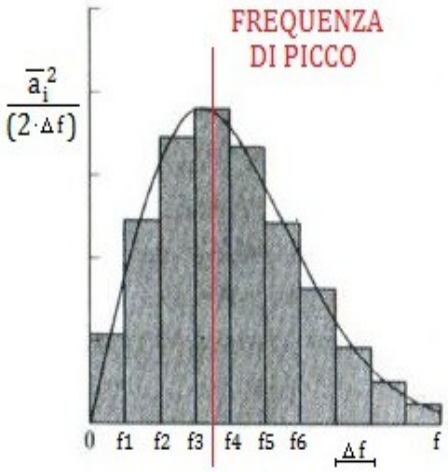

Figura 2.16: spettro in frequenza

In caso di un unico moto ondoso regolare, lo spettro assumerebbe valori diversi da zero solo ad una certa frequenza (fig. 2.17).

Mano a mano che il moto ondoso diventa irregolare, si aggiungono altre linee nello spettro, che tendono a infittirsi, fino a definire una curva (fig. 2.18).

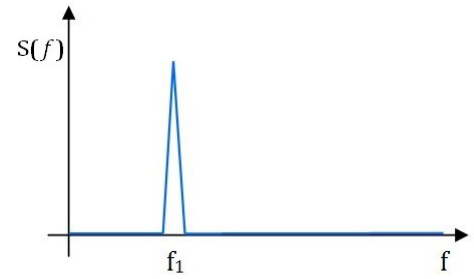

Figura 2.17: spettro in frequenza di un moto ondoso lineare

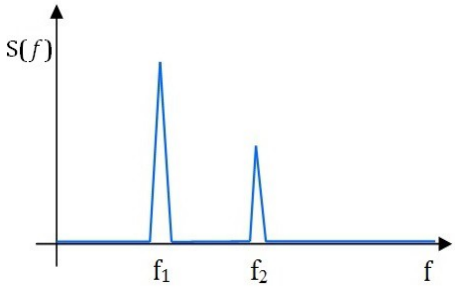

Figura 2.18: Spettro in frequenza di due moti ondosi regolari con ampiezza e periodo diversi.

Con componenti armoniche che hanno frequenze simili, si ottiene uno spettro a banda stretta, tipico di moti ondosi quasi regolari. In natura si osserva questo fenomeno quando le onde si formano lontane (migliaia di km) rispetto alla zona dove viene effettuata la misurazione.

Questo avviene in quanto si ha un processo di selezione fisica delle componenti armoniche. Nel procedere del moto ondoso, le componenti con i periodi più piccoli corrispondono alle componenti a bassa energia, che si perdono; per cui nello spettro non si ha più la coda ad alta frequenza, ma rimangono quelle frequenze che hanno la maggior capacità di compiere lavoro, dunque di spostare le particelle d'acqua.

Si ottiene così uno spettro a banda stretta, dove le frequenze rimanenti sono vicine alla frequenza di picco (fig.2.19).

Questo tipo di spettro caratterizza onde che sono chiamate onde di swell.

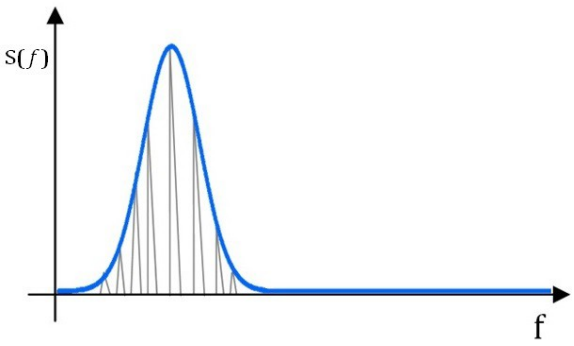

Figura 2.19: spettro a banda stretta tipico di onde di swell.

Se invece lo spettro risulta essere a banda larga, significa che il moto ondoso è fortemente irregolare. Il fenomeno si osserva in zone in cui nel momento della misurazione sta soffiando vento. Questo tipo di spettro caratterizza onde che sono

chiamate “wind wave” oppure “sea” *(fig.2.20).*

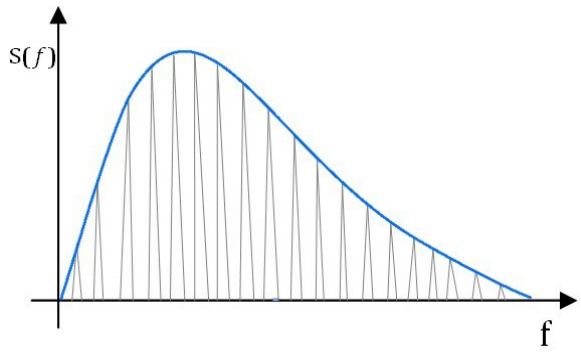

Figura 2.20: spettro a banda larga tipico di onde a mare vivo

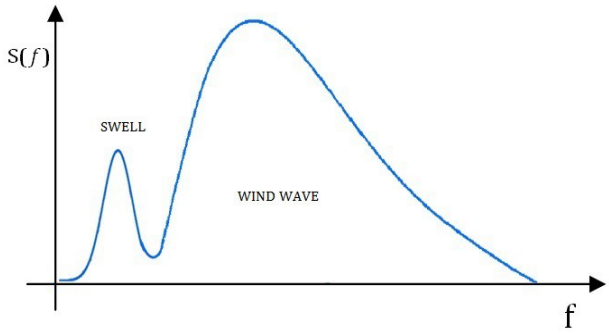

Figura 2.21: spettro a doppio picco

Avendo invece una condizione di sovrapposizione tra lo stato di mare locale, dove tira vento, e una componente di onda di swell, si ottiene lo spettro a doppio picco (fig. 2.21).

2.2.3Modelli di spettri in frequenza

Abbiamo visto che, avendo le misure ondametriche in un punto P nel mare, si può caratterizzare il moto ondoso in quel punto, sia nel dominio del tempo che nel dominio della frequenza, e così ottenere i parametri significativi H_s e T_s.

Al contrario, avendo H_s e T_s, ricavati dal bollettino dello stato del mare , si può ottenere la forma spettrale.

Per l'ottenimento dello spettro si dovranno considerare le caratteristiche del mare in esame, quali l'area di fetch caratteristica, l'estensione del mare, da dove arriva il moto ondoso, ecc. Per tipologie omogenee di mare, le mareggiate hanno andamenti spettrali simili, quindi è possibile proporre funzioni parametriche continue che approssimano gli spettri.

Esistono diversi tipi di queste funzioni, aventi formula generale:

$$S(f, T_s, H_s, \ldots) = f^{\alpha} \cdot H_s^{\beta} \cdot T_s^{\gamma} \cdot e^{-\alpha f}$$

Tra tutte le possibili funzioni analitiche dello spettro in frequenza, si sceglie quella che più si addice al bacino in studio.

Per il Mediterraneo solitamente si usa la Forma J.O.N.S.W.A.P. (acronimo di un programma di ricerca che ha misurato i moti ondosi del Mare del Nord).

$$S_J(f) = \frac{\alpha_J g^2}{(2\pi)^4 f^5} \mathrm{e}^{\frac{-5}{4}\left(\frac{f}{f_p}\right)^{-4}} \gamma^{e^{\alpha}}$$

Questa forma contiene una serie di parametri: α e β sono detti parametri di Philips, mentre γ è detto *fattore di elevazione del picco*. γ solitamente varia tra 1 e 7, quanto più è grande tanto più lo spettro è a banda stretta. Invece con γ = 1, la forma dello spettro viene definita dalla funzione di Pierson – Moskovitz.

È stato stimato un valore ottimale del parametro γ di 3,3, che spesso viene utilizzato anche per il Mar Mediterraneo, mentre in realtà fa riferimento agli studi del moto ondoso nel Mare del Nord.

Nel Mediterraneo si dovrebbe infatti utilizzare una funzione alla J.O.N.S.W.A.P. con γ circa 2, anche se per ragioni di cautela è lecito utilizzare un valore maggiore.

Uno spettro tipico utilizzabile per descrivere la sovrapposizione dello stato di mare locale e della componente di swell è lo **Spettro di Ochi**.

$$S(f)=\sum_{j=1}^{2}\frac{H_{sj}^{2}T_{pj}}{4\Gamma(\lambda_j)}\frac{(\lambda_j+0.25)^{\lambda_j}}{(T_{pj}f)^{(4\lambda_j+1)}}\exp\left\{-\frac{(\lambda_j+0.25)}{(T_{pj}f)^{4}}\right\}$$

2.3 Teoria del moto ondoso lineare

Detta anche Teoria di Airy, o Teoria di Stokes al primo ordine di approssimazione, o Teoria delle onde di ampiezza infinitesima.

2.3.1Introduzione

Questa è stata la prima teoria proposta storicamente (metà 1800), quindi risulta essere una delle più semplici oggi disponibili, ma nonostante questo riesce a fornire risultati tecnicamente accettabili. Si basa sull'ipotesi che le onde siano regolari e di altezza infinitesima. In realtà non possono esistere onde di altezza infinitesima ma le previsioni che questa teoria ci fornisce presentano errori tecnicamente accettabili. Allo stesso modo anche parlare di onde regolari è un'approssimazione consentita alla luce della scomposizione in componenti armoniche, perché se per esempio si considera un effetto del moto ondoso si può dire che questo è uguale alla somma degli effetti di tutti i fenomeni che lo compongono.

Considerando:

η_T = fenomeno totale

η_a e η_b = componenti

$$\eta_T = \eta_a + \eta_b$$

Esempio 1. La legge che descrive il fenomeno è: $P(\eta_T) = 3\eta_T$

$$P(\eta) = 3(\eta_a + \eta_b) = 3\eta_a + 3\eta_b = P(\eta_a) + P(\eta_b)$$

In questo modo si osserva la dimostrazione di quanto affermato precedentemente.

Esempio 2. La legge che descrive il fenomeno è: $P(\eta_T) = 3\eta_T^2$

$$P(\eta) = 3(\eta_a + \eta_b)^2 = 3\eta_a^2 + 3\eta_b^2 + 6\eta_a \cdot \eta_b$$

La teoria è verificata solo nel caso che il fenomeno sia lineare. Dalla formula si osserva la comparsa di un *termine non lineare* ($6 \cdot \eta a \cdot \eta b$) dipendente dal prodotto delle cause e non dalla loro somma. Nell'analisi del moto ondoso normalmente gli effetti del termine non lineare sono trascurabili; diventano importanti nel caso delle onde anomale: se il termine non lineare fosse molto grande, l'analisi non sarebbe valida.

2.3.2 Modello del moto ondoso lineare

Si elencano una serie di risultati ottenuti da Airy senza dimostrarli. Questi dipendono dalle equazioni cardinali della meccanica dei fluidi (*equazioni di Navier-Stokes*).

Il punto di partenza è la seconda legge di Newton: $\vec{F} = m \cdot \vec{a}$, che vale per una particella. Considerando invece un dominio liquido (insieme infinito di particelle materiali) ottengo le equazioni di Navier-Stokes:

$$\frac{D\vec{U}}{Dt} = -\frac{1}{\rho} \cdot \nabla P + \nu \nabla^2 \vec{U} + g\vec{k}$$

Dove a sinistra dell'uguaglianza si nota la variazione di velocità rispetto al tempo, cioè l'accelerazione; a destra ci sono le forze per unità di massa: il primo termine si riferisce alla pressione, il secondo alla viscosità (forze tangenziali dovute al trascinamento), il terzo è la componente della forza di gravità che agisce solo nella direzione z (sapendo che k è il versore della direzione dell'asse z).

Queste equazioni sono sempre valide ma vengono aggiunte delle condizioni al contorno che caratterizzano il sistema in esame e delle ipotesi semplificatrici (ipotesi di linearità, fluido ideale incomprimibile, forze viscose nulle, ampiezza dell'onda infinitesima, fondale piano).

Otteniamo quindi un sistema di equazioni differenziali la cui risoluzione definisce la teoria del moto ondoso lineare. Queste equazioni sono state risolte da Airy considerando un sistema bidimensionale, in cui si osserva il fenomeno lungo un piano individuato dalle rette z e x. Con la risoluzione del set di equazioni del modello lineare, è possibile calcolare una serie di parametri:

- η: $\eta = \frac{H}{2} \cdot \cos(k \cdot x - \omega \cdot t)$
- campo di pressione p: $p = \rho \cdot g \cdot z + \rho \cdot g \cdot \frac{H}{2} \cdot \frac{\cosh[k \cdot (z+d)]}{\cosh(kd)} \cdot \cos(kx) \cdot \cos(\omega \cdot t)$
- la componente orizzontale della velocità u durante la propagazione del moto ondoso: $u = \frac{H}{2} \cdot \frac{\pi \cdot \cosh[k \cdot (z+d)]}{T \cdot senh(kd)} \cdot \cos(kx - \omega \cdot t)$

- la componente verticale della velocità v:

$$u = \frac{H}{2} \cdot \frac{\pi \cdot senh[k \cdot (z+d)]}{T \cdot senh(kd)} \cdot sen(kx - \omega \cdot t)$$

Esaminando le componenti orizzontali e verticali come funzioni della posizione, appaiono sfasate di un angolo di 90°. I valori estremi della velocità orizzontale si ottengono quando (kx – ωt) è uguale a 0, π, 2π, …, in corrispondenza del passaggio dalla cresta o dal cavo. I valori estremi della velocità verticale v compaiono invece a (kx – ωt) = π/2, 3π/2, 5π/2, …, dove lo spostamento della funzione η(t) rispetto allo SWL è 0.

La variazione verticale della componente velocità è meglio visualizzabile partendo dal fondo, cioè dove k(z+d) = 0. In questo punto i termini iperbolici che contengono z, sia per u che per v, hanno il loro minimo, rispettivamente 1 e 0. Spostandosi verso la superficie, il modulo della velocità aumenta gradualmente.

Si vuole quindi calcolare come variano i parametri d'onda, mentre l'onda si muove da largo verso costa.

Sono noti H_0, T_0, α_0: parametri d'onda a largo con fondale $d \to \infty$, ricavabili dalle registrazioni degli ondametri. Si considerano fondali di questo tipo se: $\frac{d}{L} > \frac{1}{2}$

Sapendo che $K = \frac{2\pi}{L}$, si può anche scrivere: $K \cdot d > \pi$.

A questo punto, occorre calcolare i parametri T_d, L_d, α_d e H_d alla profondità definita d.

1) Periodo (T_d).
   Parlando di moto ondoso regolare, il periodo non varia spostandoci verso costa (a meno che non intervengano fattori esterni). Quindi $T_d = T_0$.
2) Lunghezza d'onda (L_d).
   Si usa l'equazione di dispersione:

$$\omega^2 = k \cdot g \cdot tgh(k \cdot d) \quad .$$

Quindi se $k \cdot d > \pi$ (onda in acque profonde) il valore di $tgh(k \cdot d)$ può essere approssimato a 1.

Se invece $k \cdot d > \frac{\pi}{10}$ (onda in acque basse), il valore di $tgh(kd)$ vale circa quanto k·d.

Si può quindi dedurre che, fissati profondità e periodo, ci può essere solo una lunghezza d'onda che soddisfi questa equazione. Su un dato fondale, non possono esistere due onde con lunghezza diversa, se hanno lo stesso periodo.

L'equazione di dispersione può essere riscritta, sapendo che:

$$\omega = \frac{2\pi}{T} \quad \text{e} \quad K = \frac{2\pi}{L} \quad .$$

$$\left(\frac{2\pi}{T}\right)^2 = \frac{2\pi}{L} \cdot g \cdot tgh\left(\frac{2\pi}{L} \cdot d\right)$$

Questa equazione è definita in maniera implicita per quanto riguarda L, che viene ricavata attraverso metodi iterativi. Sapendo che L nel procedere dell'onda verso costa diventa monotonicamente più piccola, si può utilizzare il metodo iterativo per bisezione. Tale metodo prevede che in precedenza si sia calcolata L_0 (lunghezza d'onda in acque infinitamente profonde).

$$k \cdot d > \pi \quad \text{quindi} \quad tgh(k \cdot d) \simeq 1 \quad \text{per cui} \quad \omega^2 = \frac{2\pi}{L_0} \cdot g \qquad L_0 = \frac{g \cdot T_0^2}{2\pi}$$

Ottenuto il valore di L_0, si individua L_d per tentativi successivi, come per esempio in tabella 2.2.

L	K	K·g·tanh(K·d)	ω^2
L_0	K_0	$K_0 \cdot g \cdot \tanh(K_0 \cdot d)$	$2\pi/T_d$
$L_1=L_0/2$	K_1	$2K_0 \cdot g \cdot \tanh(2K_0 \cdot d)$	$2\pi/T_d$
$L_2=L_1+L_1/2$	K_2	$4K_0 \cdot g \cdot \tanh(4K_0 \cdot d)$	$2\pi/T_d$
…	…	…	$2\pi/T_d$
…	…	…	$2\pi/T_d$

Tabella 2.2: metodo iterativo per bisezione

Si ferma l'iterazione per quella lunghezza d'onda che rende il termine ω^2 circa uguale a: $K \cdot g \cdot \tanh(K \cdot d)$.

3) Angolo tra raggio d'onda e la perpendicolare alla costa (α_d). Si utilizza la Legge di Snell (legge della rifrazione):

$$\frac{c}{sen\alpha} = cost$$

Per due punti, avremo quindi: $\frac{c_0}{sen\alpha_0} = \frac{c_d}{sen\alpha_d}$, dove:

$$c_0 = \frac{L_0}{T_0} = \frac{g}{2\pi} \cdot T_0 \quad \text{e} \quad c_d = \frac{L_d}{T_d}$$

Dalla Legge di Snell possiamo ricavare:

$$\alpha_d = arcsen\left(\frac{c_d}{c_0} \cdot sen\alpha_0\right)$$

4) Altezza d'onda (H_d).

Per ricavare l'altezza d'onda a una profondità d, si considera il principio di conservazione dell'energia.

Energia e potenza.

L'energia è la capacità di compiere un lavoro, cioè la capacità di spostare delle masse. L'energia meccanica è data dalla somma dell'energia potenziale, che dipende dalla quota a cui la massa è posta e dall'energia cinetica, che dipende invece dal moto stesso della massa in questione.

$$E_m = E_p + E_c = m \cdot g \cdot z + \frac{1}{2} \cdot m \cdot v^2$$

Considerando il mare come un insieme di particelle si può ricavare l'energia meccanica come somma delle energie di ogni particella. Integrando per le infinite particelle e mediando sulla lunghezza d'onda, si ottiene:

$$E_p = \frac{\gamma \cdot H^2}{16} \qquad E_c = \frac{\gamma \cdot H^2}{16} \quad , \quad \gamma = \rho \cdot g$$ peso specifico dell'acqua.

Di conseguenza:

$$E_m = \frac{\gamma \cdot H^2}{8}$$, per unità di lunghezza d'onda.

L'unità di misura di E_m è $\frac{J}{m^2}$, m^2 in quanto si considera la superficie definita dalla lunghezza d'onda e dalla distanza unitaria fra cresta e cresta. Nota l'energia del sistema, si può ricavare anche la potenza:

$$P = E_m \cdot c_g$$

misurata in $\frac{W}{m}$. P è il lavoro compiuto nell'unita di tempo attraverso una superficie larga 1 m in cresta profonda d.

c_g è la celerità di gruppo, celerità con cui si propaga l'energia dell'onda ed è definita come: $c_g = n \cdot c$ dove c è la celerità di propagazione della forma d'onda,

mentre:

$$n=\frac{1}{2}\cdot\left(1+\frac{2k_d}{senh(2k_d)}\right)$$

In acque profonde, $senh\to\infty$ per cui $n=\frac{1}{2}$. L'energia si propaga quindi con celerità dimezzata rispetto a quella della forma d'onda.

In acque basse invece $senh(2kd)\to 2kd$ per cui $n=1$, quindi la celerità di gruppo è uguale alla celerità del fronte d'onda.

Si considera una linea di riva infinitamente estesa e un fondale avente linee batimetriche rettilinee e parallele. In acque intermedie, prendendo in esame la cresta tra due punti A_1 e B_1, si osserva che questi si trovano a profondità differenti, quindi la cinematica d'onda nei due punti è diversa. Durante il propagarsi dell'onda, la cresta raggiunge il punto A_1', la cui cinematica è uguale a quella del punto A_1, in quanto si trovano sulla stessa linea batimetrica. Chiamando *l* la distanza $\overline{A_1A_1'}$, questa deve restare costante, perché i due punti si propagano in modo identico, quindi seguendo percorsi paralleli (fig. 2.22).

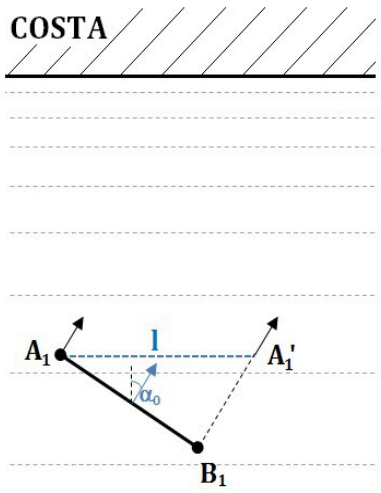

Figura 2.22: caratteristiche di propagazione di una porzione di cresta d'onda

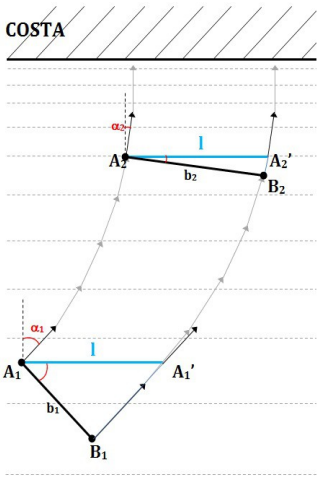

Figura 2.23: schema di propagazione di una porzione di cresta d'onda da acque intermedie ad acque basse.

In acque basse invece la velocità di propagazione dipende solo da *d*, infatti:

$$c=\frac{L}{T}=\sqrt{gd}$$.

Calcolando la velocità di propagazione dell'onda in A_1 e B_1, si osserva che $C_B > C_A$,

perché la celerità di propagazione è proporzionale alla profondità del fondale. È per questo motivo che si ha la rifrazione dei punti d'onda.

In figura 2.23 si osserva che, per il triangolo rettangolo $A_1B_1A_1'$, l'ipotenusa è definita come $l=\frac{\overline{A_1B_1}}{cos\alpha_1}$, ma essendo *l* costante: $\frac{\overline{A_1B_1}}{cos\alpha_1}=\frac{\overline{A_2B_2}}{cos\alpha_2}$, da cui si ricava che $\frac{cos\alpha_1}{cos\alpha_2}=\frac{\overline{A_1B_1}}{\overline{A_2B_2}}$.

Quindi la distanza tra i due punti in cresta tende ad allontanarsi seguendo questo rapporto e adagiandosi infine su *l*.

2.4 Propagazione del moto ondoso

Si analizza lo sviluppo del moto ondoso ipotizzando una genesi in fondali a profondità infinita, cioè profondità alle quali gli effetti del moto ondoso superficiale non arrivano sul fondo. Il moto ondoso si propaga verso la costa passando dalle acque a profondità intermedia, fino alle acque basse (fig. 2.24).

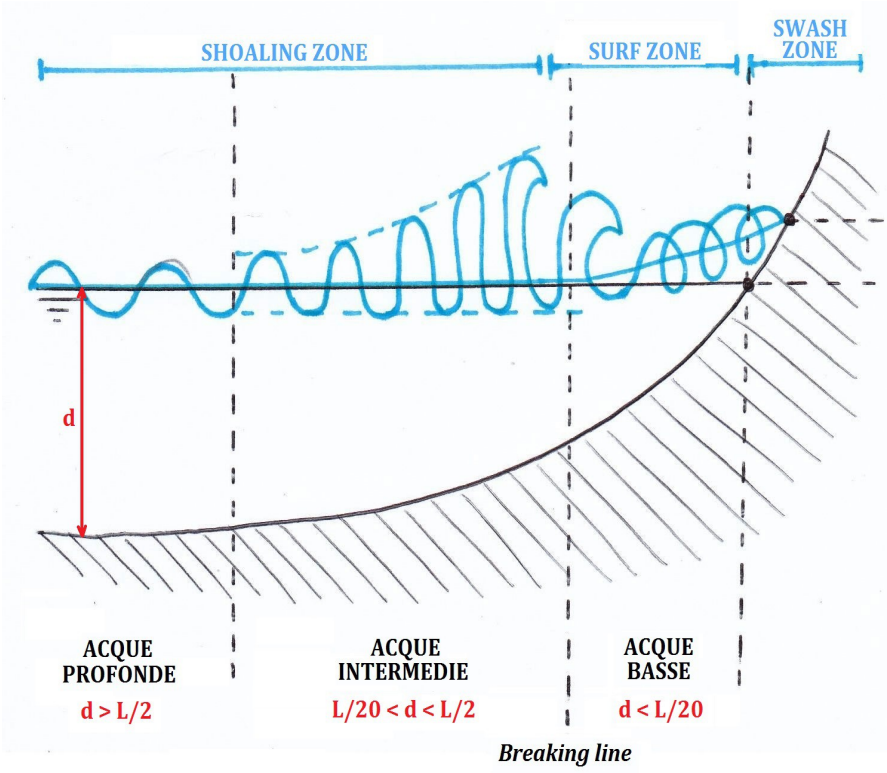

Figura 2.24: schema di propagazione del moto ondoso

- Acque profonde: non c'è interazione con il fondale e i parametri altezza, lunghezza, forma, restano invariati. Non avviene trasporto di massa, per cui le particelle d'acqua hanno un moto di tipo orbitale.
- Acque intermedie: inizia interazione con il fondale. Il moto si modifica: l'altezza si abbassa, la lunghezza tende ad accorciarsi e la forma diventa meno simmetrica, con creste più appuntite e cavo più schiacciato. Inizia un trasporto di massa in cui il moto a orbitale non è chiuso ma si propaga lungo la direzione dell'onda. Con il diminuire della profondità l'inviluppo altezza-cavo si allontana finché l'altezza dell'onda aumenta fortemente; l'aumentare dell'altezza e la diminuzione della lunghezza provocano un aumento della “ripidità dell'onda”, cioè

H/L.

- Fra le acque intermedie e le acque basse la ripidità diventa un fattore limitante per la propagazione dell'onda che quindi “frange”. Si dice “shoaling zone”, la zona prima del frangimento e “surf zone”, la zona successiva al frangimento. La linea di frangimento viene detta “breaking line”.
- Acque basse: forte interazione con il fondale. Si instaura un movimento vorticoso dovuto alla caduta dell'acqua dall'onda che frange e viene trasportata dalla propagazione dell'onda; questo genera un consistente trasporto di massa.
- La zona della battigia dove la lama d'acqua risale lungo la costa si chiama “swash zone”. Nella swash zone il livello medio del mare è più alto a causa del forte trasporto di massa. Il fenomeno di incremento di altezza della linea di riva (Δz) dovuto al moto ondoso, è detto “wave set-up”. Il periodo T dell'onda può essere suddiviso in un periodo di risalita T/2 (“run-up”) e in un periodo di discesa T/2 (“run-down”). Il punto di massima risalita viene detto “up-rush”.

Se l'avanzamento del moto ondoso è ortogonale alla costa, gran parte della massa d'acqua torna indietro passando sul fondo: “corrente di undertow”.

Se invece il moto ondoso incide con un certo angolo rispetto alla costa, si hanno due componenti: la corrente di undertow e la componente litoranea che crea un flusso di massa diretto parallelamente alla costa (fig. 2.25).

La componente litoranea è debole prima della linea dei frangenti ma, avvicinandosi alla costa, genera una corrente più forte (fig. 2.26).

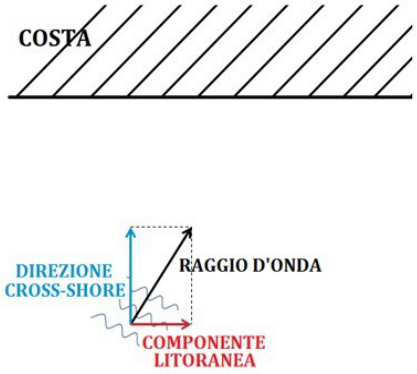

Figura 2.25: scomposizione del raggio d'onda (direzione di propagazione del moto).

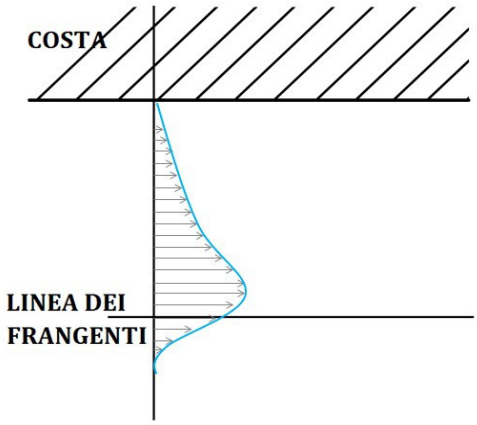

Figura 2.26: profilo di velocità della corrente litoranea.

Spostandosi verso il largo l'intensità della corrente di undertow pian piano diminuisce (fig. 2.27). Se il fondo è erodibile, il trasporto dei sedimenti operato dalla corrente di undertow si annulla nella zona dei frangenti, provocando un accumulo dei sedimenti erosi nella surf zone; tali accumuli prendono il nome di *barre* (fig. 2.28).

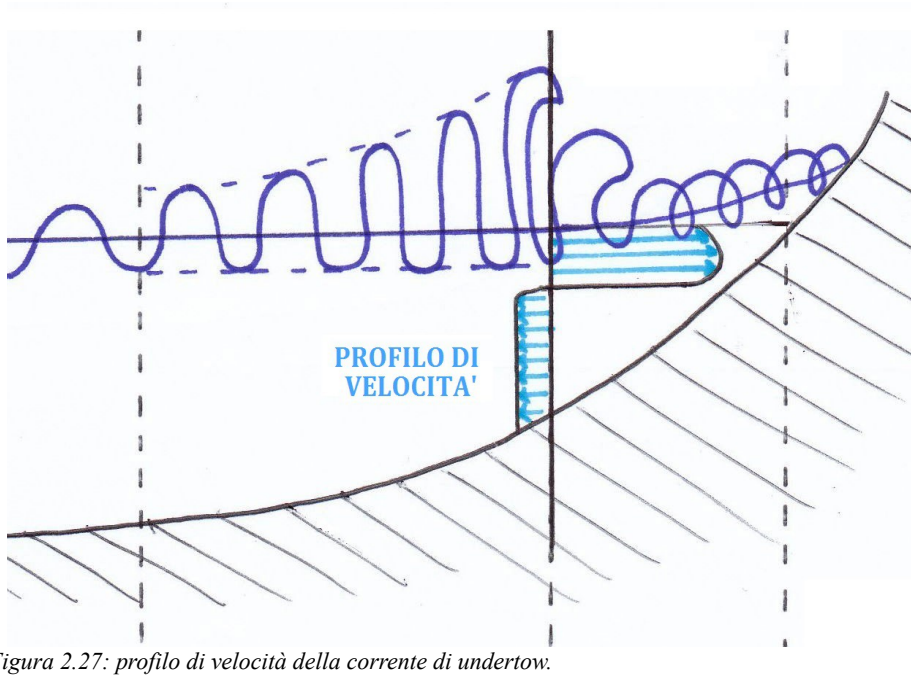

Figura 2.27: profilo di velocità della corrente di undertow.

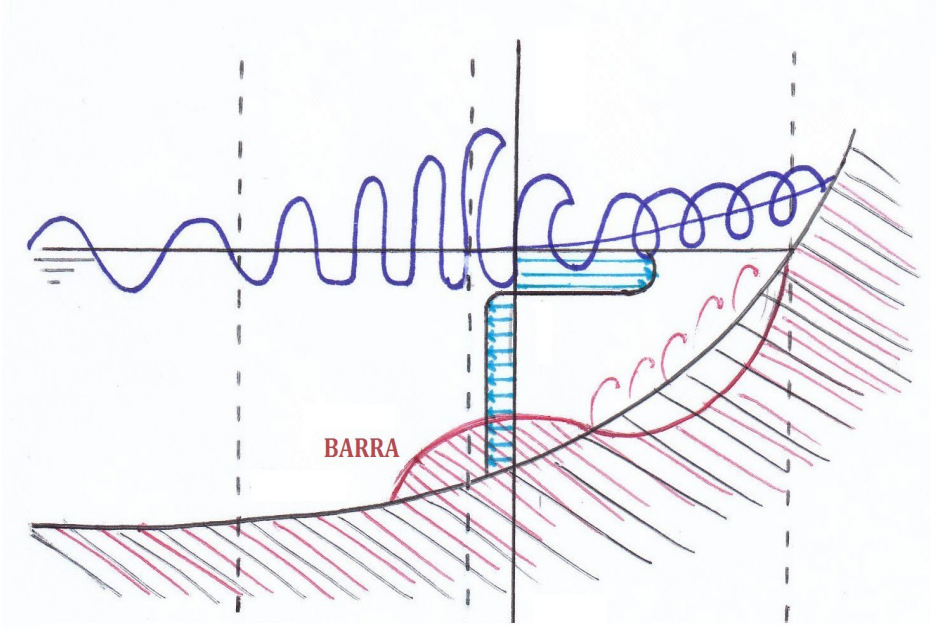

Figura 2.28: formazione delle barre.

3. Definizione dell'architettura del sistema di gestione dei dati

Con questo capitolo, si va a definire la struttura dell'intero sistema di gestione dei dati.

I dati che si propone di gestire, possono arrivare sia da mezzi di rilevazione del fenomeno fisico relativo alle onde marine (attraverso boe e trasduttori di pressione) sia da modelli numerici realizzati tramite software.

La realizzazione del sistema realizzato si suddivide nelle seguenti macrocomponenti:

1. Costruzione delle componenti gestionali: realizzazione del contenitore (il database e le tabelle) dove sono stati inseriti i dati (avvenuta con il software MySQL);

2. Procedura di acquisizione dei dati: quella parte di programmazione necessaria affinché il dato venga prelevato dalla fonte da cui proviene,qualunque essa sia, attraverso una metodologia che possa permettere il trasferimento all'interno del database;

3. Gestione dei dati: tutti quei processi relativi al controllo e alla validazione. Lo scopo di questo passaggio è fondamentale, in quanto un dato non errato e validato potrà essere soggetto a successiva elaborazione;

4. Interrogazione ed elaborazione dei dati: questa macrocomponente è dedicata all'utilizzatore finale, cioè a quell'utente che vuole visualizzare i dati richiesti. Il dato viene richiamato "a schermo" attraverso la richiesta effettuata dall'utente. Tale richiesta viene interpretata, direttamente dal sistema di gestione, come fosse una query fatta al database. L'elaborazione è stata realizzata per la costruzione di tabelle e grafici ed ha richiesto l'uso di un linguaggio di programmazione per la realizzazione e gestione di pagine web dinamiche (php).

Le componenti nel dettaglio vengono specificate all'interno delle singole

macrocomponenti.

Per quanto riguardano le componenti gestionali, in primo luogo è stato strutturato il database.

Nell'immagine (fig. 3.1), si osserva la struttura "fisica" del database realizzato.

Nel caso di questa tesi, la struttura dell'intero sistema di gestione assume delle peculiarità relative al data trattato .

Di fatto, viene creato un database chiamato dati_ondametrici, suddiviso in due tabelle chiamate archivio e dati giornalieri.

La prima tabella contiene tutti i dati che sono stati caricati sul database mentre la seconda serve da contenitore temporaneo di quei dati che vengono prelevati nelle ultime 24 ore.

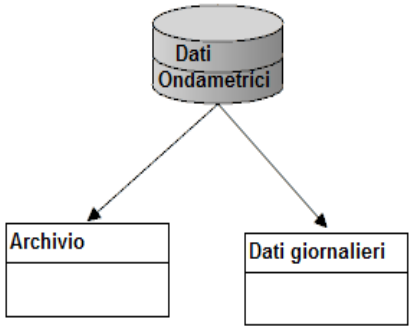

Figura 3.1: struttura "fisica" del database

Costruito il contenitore, si è passati alla realizzazione di un sistema in grado di inserire il contenuto, cioè i dati ondametrici.

La procedura di acquisizione del dato è stata realizzata tramite la creazione di un file php contenente la procedura necessaria al caricamento dei dati, sfruttando sia le query SQL che la funzionalità dei tag con cui si struttura l'HTML.

Il controllo e la validazione dei dati è avvenuta tramite una serie di passaggi che hanno compreso sia l'analisi, cioè la ricerca nel data set di valori anomali e mancanti, che la loro conversione in un dato accettabile.

Sono stati applicati dei valori soglia per l'individuazione degli errori:

10 metri per la *Hm0*, 15 metri per Hmax, 15 secondi per il *Tp* , 10 secondi per il *Tm,* tutti i valori maggiori di 360 per quanto riguardano le direzioni e temperature dell'acqua minori di 0 °C e maggiori di 40°C.

Tali soglie sono state scelte attraverso un'analisi soggettiva delle serie temporali osservate nelle misurazioni effettuate dalla Rete Ondametrica Nazionale prendendo come riferimento la boa al largo di La Spezia **[40]**.

Tale procedura è stata realizzata in maniera tale che si attivi anche quando i dati acquisiti presentano errori relativi ad una formattazione anomala.

Il sistema, riconoscendo tali errori grazie ai comandi realizzabili con delle query, sostituisce il dato errato con un valore NULL.

Per quanto riguarda il dato mancante, questo viene visto dal sistema MySQL come dato nullo, che non ha alcun valore.

L'automazione nella sistematica ricerca di dati errati e mancanti avviene attraverso la definizione di quelli che sono chiamati eventi di sistema che consistono in semplici query inizializzate secondo un certo intervallo di tempo.

Da notare che tale analisi, nella sua complessità, viene svolta esclusivamente nella tabella dati giornalieri perché dalle prove di sistema risulta essere meno impegnata la memoria RAM, quindi il processo avviene più velocemente.

I dati controllati e validati, da dati giornalieri, vengono periodicamente caricati nella tabella archivio ad una certa cadenza temporale.

A questo punto del lavoro si è reso necessario passare alla realizzazione di tutta la parte lato utente, vale a dire il sito web.

Per fare questo è stato utilizzato il linguaggio di programmazione php, che può sfruttare in simultanea le potenzialità del linguaggio HTML e gli stili di pagine CSS.

Grazie a questi linguaggi di programmazione è stato possibile realizzare un sito web dinamico, cioè in grado di modificare il suo contenuto attraverso il collegamento instaurato con il contenuto del database.

Le richieste dell'utente, in questo modo, vengono trasformate in query che permettono la selezione dei dati e inviano a schermo ciò che è stato richiesto.

Nello schema successivo (fig. 3.2) viene visualizzata la struttura completa dell'architettura del sistema.

Con il colore verde viene messa in evidenza la procedure relativa all'acquisizione dei dati, con il colore giallo la parte relativa alla gestione e con il colore rosso la componente relativa all'elaborazione.

Una trattazione pratica di come è stata realizzata la struttura verrà effettuato nel capitolo 5

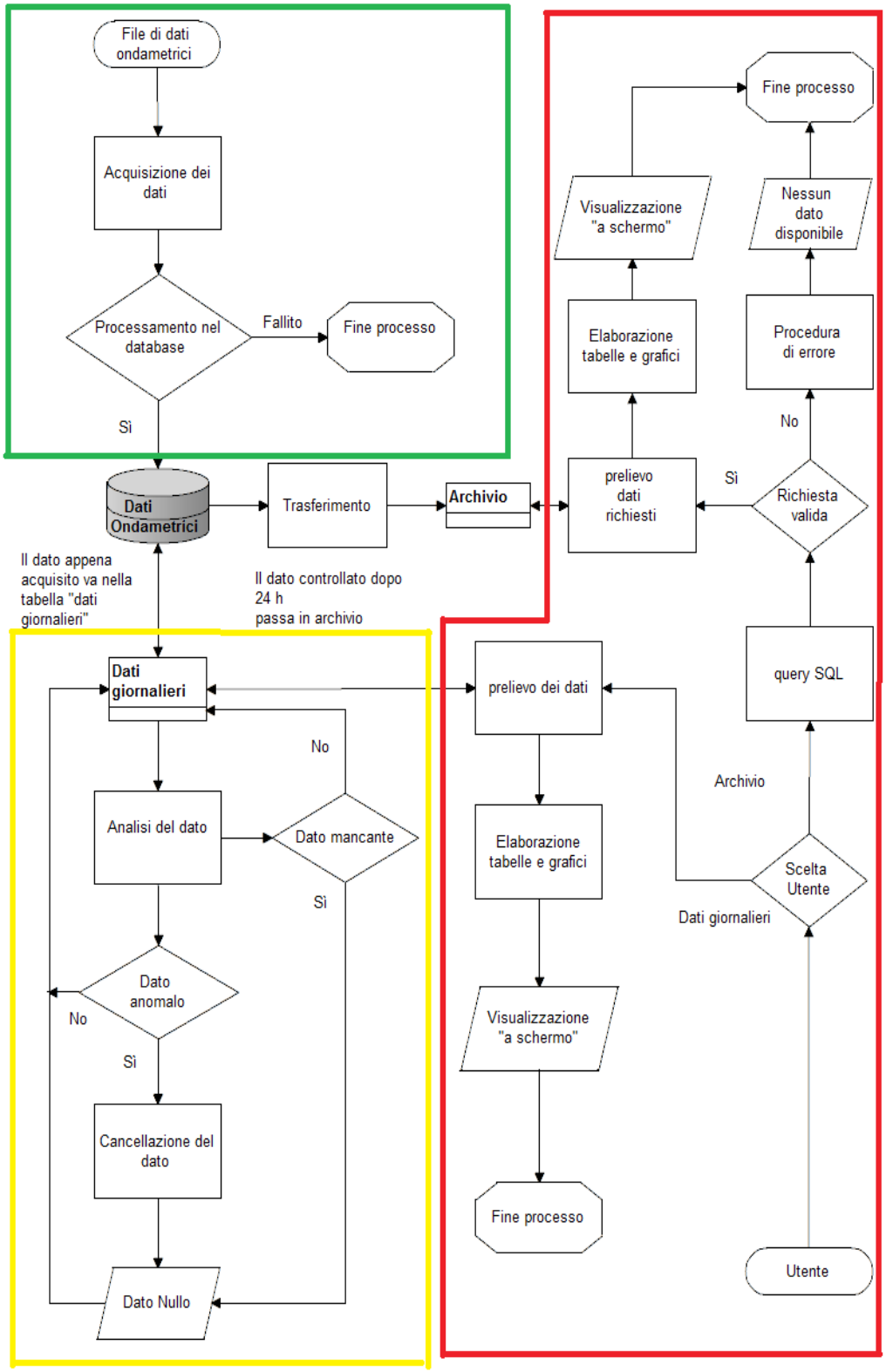

Figura 3.2: architettura dell'intero sistema

4. Definizione e funzionamento degli strumenti software usati

L'acquisizione, la gestione e l'elaborazione dei dati ondametrici avviene tramite un insieme di software.

Tali programmi si devono interfacciare fra di loro riuscendo ad eseguire tutte le operazioni necessarie affinché gli utilizzatori finali possano consultare ed elaborare i dati attraverso un sito web.

4.1 Programmi utilizzati

Nel mondo dell'informatica esistono numerose piattaforme che al loro interno contengono altrettanti numerosi software che possono dare un notevole contributo per raggiungere lo scopo prefissato.

La piattaforma scelta per la costruzione del sistema è XAMPP (versione 5.6.3).

XAMMP viene distribuito gratuitamente con licenza General Pubblic License (GNU) e

al suo interno contiene:

- Il database MySQL;
- il software di gestione del database phpmyAdmin;
- gli strumenti per utilizzare i linguaggi di programmazione php e Perl.
- il software di gestione server Apache;

La scelta di tale piattaforma è dovuta alla sua capacità di far interagire tutti i programmi elencati allo scopo della realizzazione di un sito web dinamico, dove le informazioni mostrate sono continuamente aggiornate e scelte dall'utente.

Una componente di rilevante importanza all'interno di XAMPP è l'application server.

Questo strumento rappresenta l'infrastruttura e il supporto necessario allo sviluppo e alla esecuzione dei linguaggi di programmazione ed è capace di interpretare le pagine web dinamiche realizzate con il php [2].

Questo significa, praticamente, che durante l'installazione di XAMPP viene

creata una cartella (denominata htdocs) all'interno della quale vengono messi i files relativi alla programmazione della restituzione dei dati lato utente.

I documenti inseriti, se richiamati attraverso browser, restituiscono come risultato l'elaborazione nei linguaggi usati.

I principali vantaggi dell'uso dell'application server di XAMPP possono essere così riassunti [2]:

- *Semplificazione delle attività di sviluppo:* gli application server creano un ambiente nel quale si possono utilizzare gli strumenti di sviluppo più diffusi sul mercato, consentendo di produrre e distribuire rapidamente le applicazioni.
- *Supporto di vari linguaggi, strumenti e piattaforme software*: a seconda dell'application server utilizzato, le applicazioni possono essere scritte nel linguaggio preferito dal programmatore.
- *Riusabilità del codice:* deriva sia dalla programmazione orientata ad oggetti, sia dall'utilizzo dell'approccio a componenti. Una volta sviluppata la logica applicativa, essa può essere condivisa e riutilizzata.
- *Scalabilità:* la capacità del sistema di crescere o diminuire in funzione delle necessità e disponibilità
- *Alte prestazioni.*
- *Estendibilità:* l'architettura modulare degli application server e il supporto per i server e per i moduli applicativi che possono essere caricati dinamicamente, consente di estendere facilmente le funzionalità del sistema e delle relative applicazioni.
- *Robustezza*: i componenti del server e la logica applicativa possono essere riconfigurati, aggiunti o rimossi senza interruzioni nell'erogazione dei servizi agli utenti. Queste caratteristiche sono particolarmente importanti per garantire l'alta disponibilità del sistema.

Per la realizzazione dell'intero sistema di gestione è stato necessario svolgere un lavoro di tirocinio in supporto all'elaborazione della tesi.

Durante il tirocinio, non tutti i programmi contenuti all'interno di XAMPP sono stati utilizzati.

La creazione del database e delle tabelle non ha richiesto nessun supporto da

parte di un server remoto, quindi non sono state usate in pieno le funzionalità relative ad Apache.

I dati ondametrici, il database e le tabelle dove sono contenuti, sono stati memorizzati direttamente sul PC dove si è realizzato l'intero sistema.

In questo caso si dice che si è lavorato in "locale" (con un localhost), cioè si risulta in grado di comunicare con il proprio PC come se fosse una macchina remota, usando le stesse applicazioni utilizzabili proprio come se si lavorasse in un server.

Nella lettura della tesi spesso ci si potrà imbattere nel termine server ed è sempre da ricordare che il comportamento dei programmi, così come verranno definiti, non cambia se vengono utilizzati in applicazioni che si appoggiano a server remoto o a locale

Un altro aspetto è quello legato all'utilizzo del linguaggio di programmazione.

In questo caso la scelta è ricaduta su php, tralasciando Pearl, in quanto la sua potenza è tale da essere in grado di implementare soluzioni avanzate le quali permettono un controllo completo sulle operazioni che possono essere svolte dal server.

A supporto di Php sono stati associati una serie di strumenti molto utili, fra cui il RepositoryPEAR. [26]

Questo strumento contiene decine di classi ben organizzate e documentate per svolgere la maggior parte delle operazioni richieste durante lo sviluppo di applicazioni web.

Tra queste si ricorda il layer di astrazione per l’accesso ai database, le classi per il debugging ed il logging, quelle per la generazione di grafici avanzati e quelle per la gestione delle template

Uno svantaggio nell'uso di php consiste nella distribuzione in due versioni differenti (oggi esistono la 4 e la 5)

Tale fattore provoca una limitazione degli sviluppatori nell'utilizzo delle caratteristiche della nuova versione, dato che la maggior parte dei servizi associati ai server continuano ad aggiornare quella precedente.

Oltretutto il fatto che php funzioni ad estensioni non è sempre un vantaggio, dato che spesso e volentieri i vari servizi server hanno configurazioni differenti.

In compenso, il linguaggio Php è un ottimo linguaggio, uno dei più utilizzati tra

quelli open source per lo sviluppo web.

4.2 Installazione e configurazione dei componenti

Come già detto in precedenza, XAMPP rappresenta il contenitore e contemporaneamente il programma che permette l'interfacciamento di altri software utilizzabili per lo sviluppo di sistemi adatti a trattare i dati.

Con questo paragrafo si vuol dare una spiegazione su come installare e configurare questo fondamentale programma [1].

Il software è una distribuzione dell'azienda Apache, scaricabile dal sito **https://www.apachefriends.org/it/index.html**.

Un semplice ed intuitivo wizard setup permette a qualsiasi utente di installare la piattaforma sul proprio PC.

Finita l'installazione apparirà il seguente pannello di controllo (fig. 4.1).

Il pannello di controllo consente di installare/disinstallare e avviare/fermare i vari servizi.

Si nota in fig 4.1 che i servizi disponibili sono più numerosi di quanti ne sono stati elencati nella definizione del contenuto di XAMPP.

I programmi presenti ma non elencati (come FileZilla,Mercury e Tomcat) non sono da prendere in considerazione in quanto non sono stati utilizzati per lo scopo prefissato.

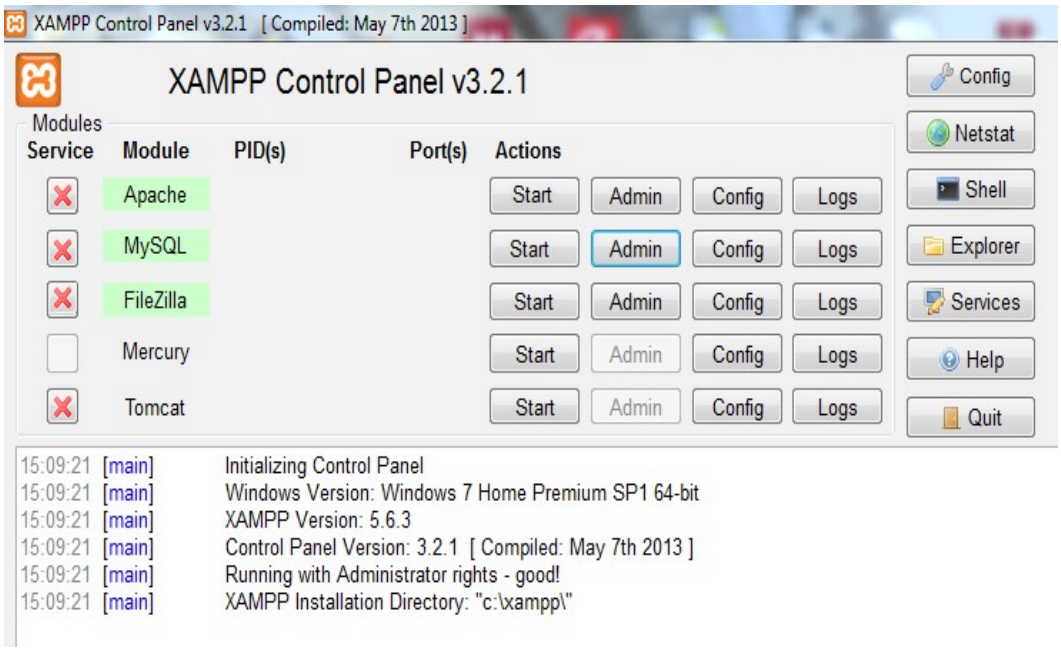

Figura 4.1: pannello di controllo di XAMPP

La parte presa in considerazione è quella relativa all'installazione di MySQL e Apache.

La loro installazione all'interno del PC è avvenuta spuntando la voce corrispondente del menu Modules Service.

Il pannello di controllo apparirà nel seguente modo (fig 4.2):

Figura 4.2: il pannello di controllo dopo l'installazione delle componenti

Si può verificare la corretta installazione dei servizi MySQL e Apache dal pannello di controllo di sistema, tramite la maschera dei servizi (fig 4.3).

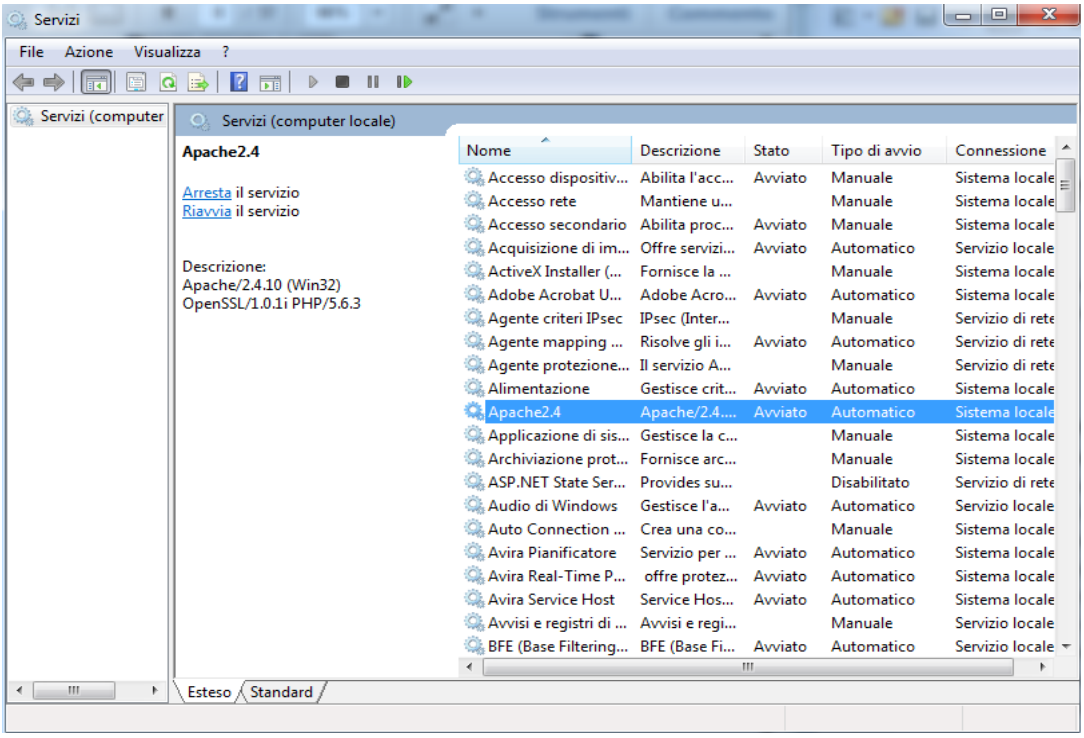

Figura 4.3: pannello dei servizi

Cliccando con il tasto destro del mouse sulle voci corrispondenti ad Apache e MySQL si apre un menu a tendina da cui è possibile selezionare le proprietà del servizio .

Selezionata la casella proprietà, è stato impostato come tipo di avvio "Automatico", in modo tale da rendere già disponibili entrambi i servizi all'accensione del PC.

Alla fine, il pannello di controllo appare come in fig 4.4.

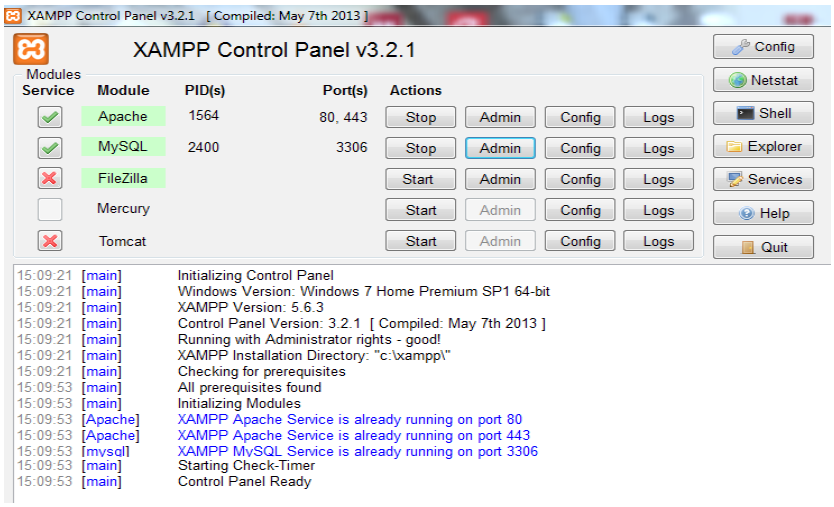

Figura 4.4: attivazione dei servizi tramite il pannello di controllo

Dopo aver concluso con l'installazione dei servizi, si è passati alla loro configurazione.

Per prima cosa si deve cliccare sul pulsante Admin relativo ad Apache.

Viene aperta la seguente pagina (fig 4.5).

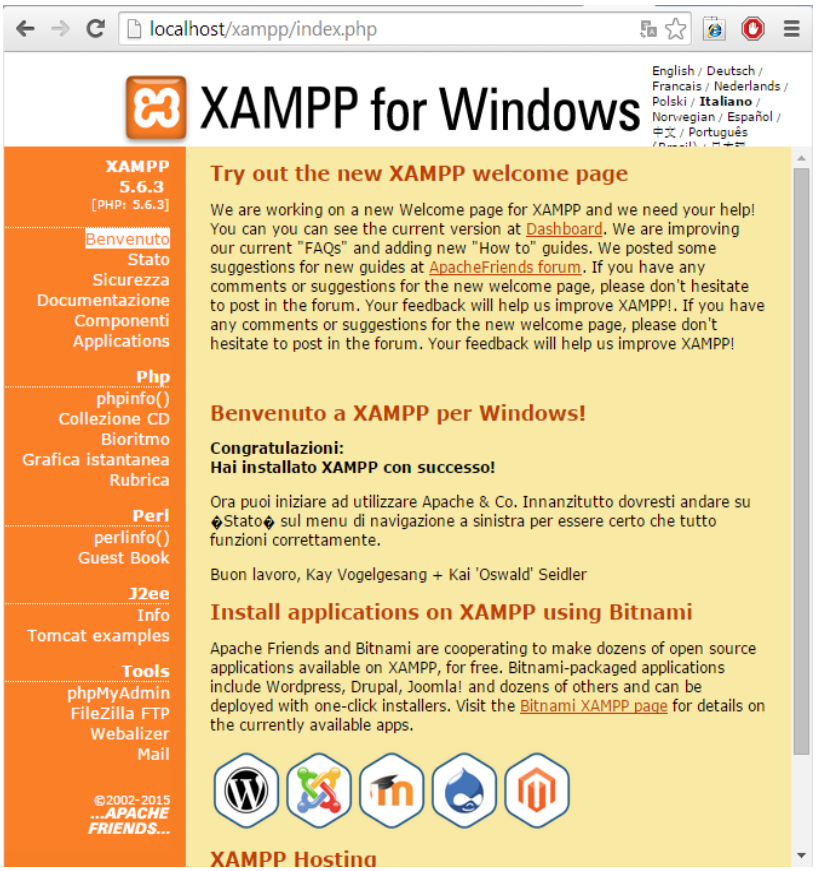

Figura 4.5: prima pagina di XAMPP

Da questa pagina si può controllare lo stato dei servizi installati e soprattutto verificare il livello di sicurezza con cui sono protetti sono il database e l'accesso alla sua gestione con phpmyAdmin.

Se viene cliccato il tasto sicurezza, inizialmente la situazione compare come in fig 4.6.

A questo punto sono state prese delle precauzioni, proteggendo l'intero sistema cliccando su http://localhost/security/xamppsecurity.php.

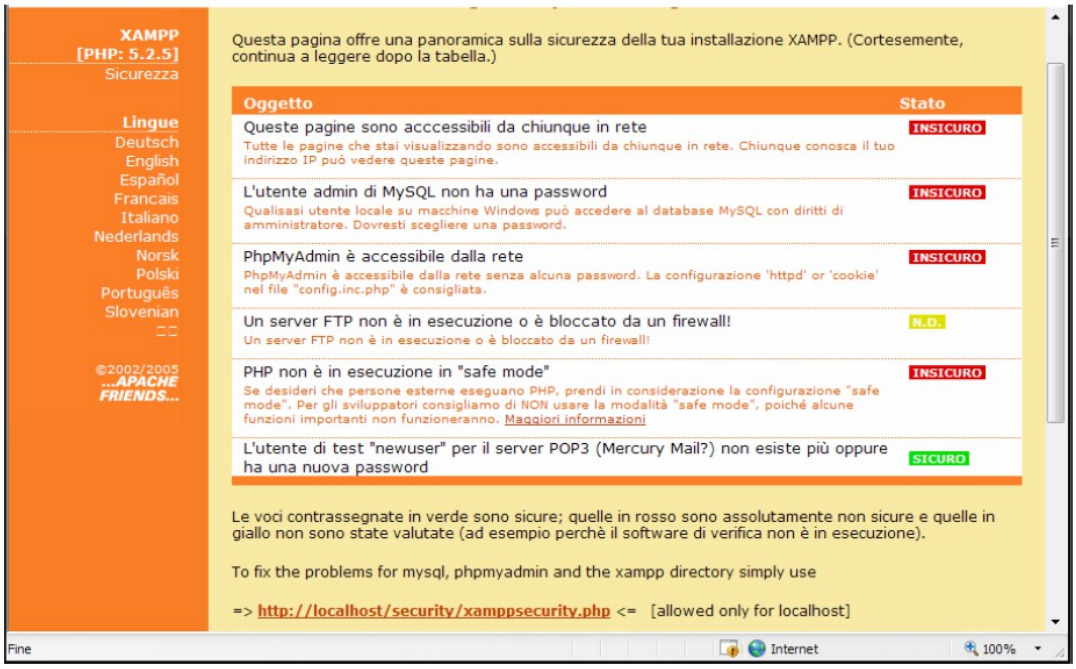

Figura 4.6: impostazioni di sicurezza

Appaiono delle righe dove è stato inserito il nome utente e la password di accesso all'interfaccia di amministrazione di XAMPP (fig 4.7).

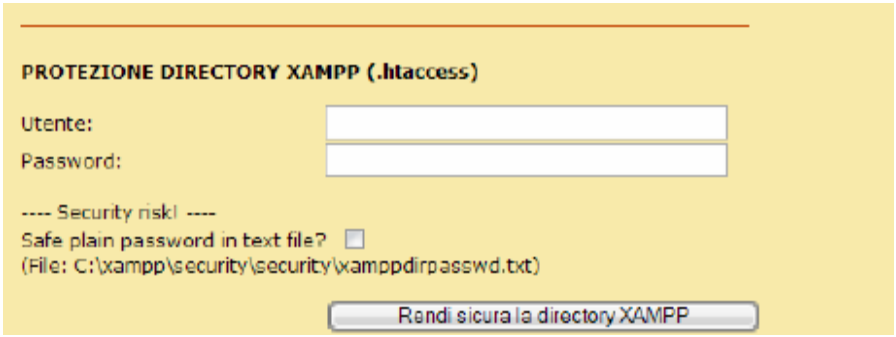

Figura 4.7: impostare la sicurezza della directory di XAMPP

La directory di XAMPP in questo modo è protetta ed appare questo messaggio di conferma (fig 4.8)

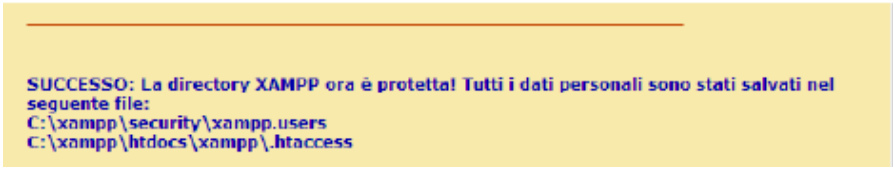

Figura 4.8: operazione eseguita con successo

Dalla stessa pagina utilizzata per inserire la password di accesso all'interfaccia di amministrazione di XAMPP, sono stati immessi l'utente e la password a protezione di MySQL (fig 4.9)

Figura 4.9: la sicurezza di MySQL

Una volta inserita la password, la conferma è avvenuta cliccando sul bottone "cambia la password" che rende la modifica permanente.

Subito dopo averlo premuto, è comparso il seguente messaggio di conferma: "The root password was successfully changed. Please restart MYSQL for changes".

Dopo queste operazioni è stata necessaria la modifica di alcune opzioni nel file di configurazione, (config.inc.php), in modo da consentire l'accesso al database.

Tale file si trova nella cartella c:\xampp\phpMyAdmin.

All'apertura del file, sono state ricercate le righe[22] riportate in fig. 4.10.

```
/* Authentication type and info */
$cfg['Servers'][$i]['auth_type'] = 'config';
$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = '';
$cfg['Servers'][$i]['extension'] = 'mysql';
```

Figura 4.10: Modifica della configurazione

A queste viene attribuito:

- il valore http $cfg['Servers'][$i]['auth_type'] alla variabile
- un valore uguale a quello che è stato precedentemente assegnato alla password di protezione del database MySQL alla variabile $cfg['Servers'][$i]['password'].

Ritornando alla pagina di XAMPP alla voce sicurezza, d'ora in avanti si può notare la situazione ottenuta con le nuove impostazioni di sicurezza (fig 4.11).

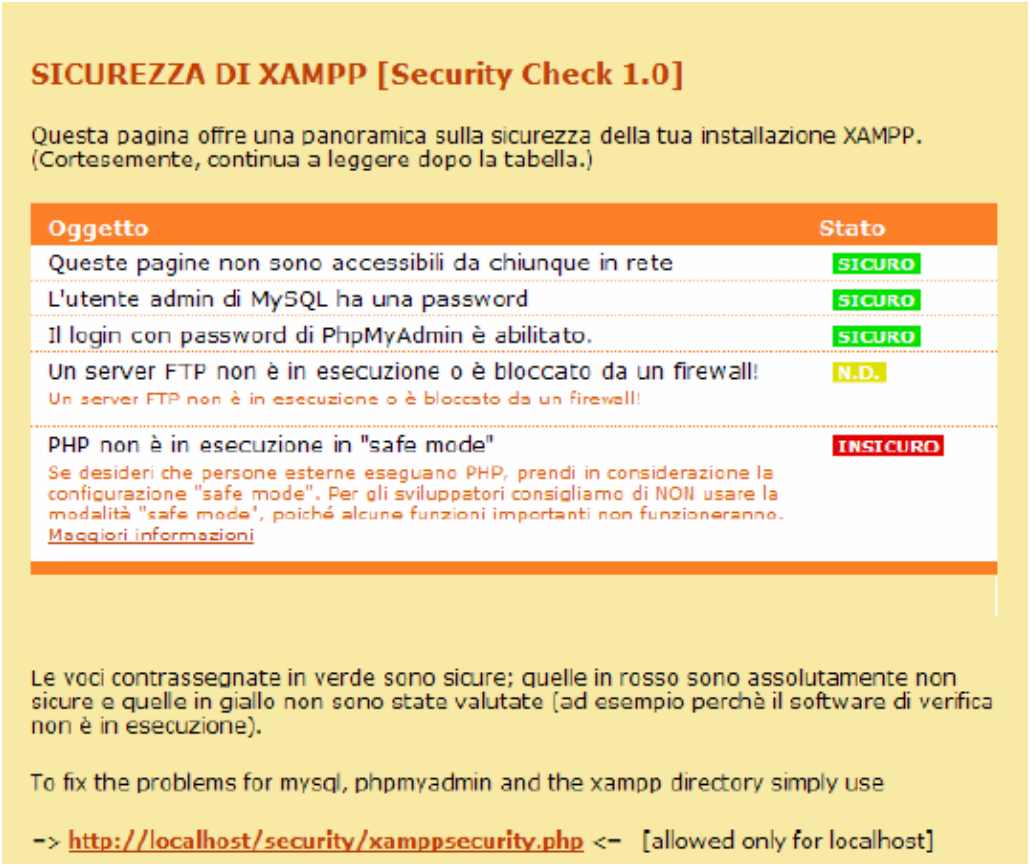

Figura 4.11: nuove impostazioni di sicurezza

4.3 Il database

Un database è una collezione di dati che viene gestita e organizzata da un software specifico, il DBMS (DataBase Management System, ovvero Sistema di Gestione di DataBase) [6].

Un DBMS è sostanzialmente un software che si frappone fra l'utente ed i dati veri e propri.

Grazie a questo strato intermedio l'utente e le applicazioni non accedono ai dati così come sono memorizzati effettivamente, ma ne vedono solamente una rappresentazione logica.

Ciò permette un elevato grado di indipendenza fra le applicazioni e la memorizzazione fisica dei dati.

La caratteristica principale secondo cui i DBMS vengono normalmente classificati consiste nella rappresentazione logica dei dati che essi mostrano ai loro utilizzatori.

Nel corso degli anni sono stati adottati numerosi modelli per i dati, a fronte dei quali esistono quindi vari tipi di database [6].

I tipi più comuni sono:

- ***Database gerarchici***: i dati vengono organizzati in insiemi legati fra loro da relazioni di "possesso", in cui un insieme di dati può possedere altri insiemi di dati, ma un insieme può appartenere solo ad un altro insieme. La struttura risultante e' un albero di insiemi di dati.

- ***Database reticolari***: il modello reticolare e' molto simile a quello gerarchico, ed infatti nasce come estensione di quest'ultimo. Anche in questo modello, gli insiemi di dati sono legati da relazioni di possesso, ma ogni insieme di dati può appartenere a uno o più insiemi. La struttura risultante è una rete di insiemi di dati.

- ***Database relazionali***: i database appartenenti a questa categoria si basano

sul modello relazionale la cui struttura principale è la relazione, cioè una tabella bidimensionale composta da righe e colonne. Ciascuna riga, che in terminologia relazionale viene chiamata tupla, rappresenta un'entità che si vuole memorizzare nel database. Le caratteristiche di ciascuna entità' sono definite invece dalle colonne delle relazioni, che vengono chiamate attributi. Entità con caratteristiche comuni, cioè descritti dallo stesso insieme di attributi, faranno parte della stessa relazione.

- ***Database ad oggetti***: (object-oriented): lo schema di un database ad oggetti è rappresentato da un insieme di classi che definiscono le caratteristiche ed il comportamento degli oggetti che popoleranno il database. La principale differenza con i modelli esaminati finora consiste nella non passività dei dati. Infatti con un database tradizionale (intendendo con questo termine qualunque database non ad oggetti) le operazioni che devono essere effettuate sui dati vengono demandate alle applicazioni che li utilizzano. Con un database object-oriented, al contrario, gli oggetti memorizzati nel database contengono sia i dati che le operazioni possibili su tali dati.

I primi due tipi di database, quelli gerarchici e reticolari appartengono ormai alla storia dell'informatica.

La maggior parte dei database attualmente utilizzati appartiene alla categoria dei database relazionali.

I motivi di questo successo (anche commerciale) vanno ricercati nella rigorosità matematica e nella potenzialità espressa dal modello relazionale su cui si basano, nella sua semplicità di utilizzo e nella disponibilità di un linguaggio di interrogazione standard, l'SQL, che permette di sviluppare applicazioni indipendenti dal particolare DBMS relazionale utilizzato **[6]**.

Indipendentemente dal tipo di database, le funzionalità principali che ci si deve aspettare da un DBMS sono quelle di:

- consentire l'accesso ai dati attraverso uno schema concettuale, invece che attraverso uno schema fisico;
- permettere la condivisione e l'integrazione dei dati fra applicazioni differenti;
- controllare l'accesso concorrente ai dati;
- assicurare la sicurezza e l'integrità dei dati.

Grazie a queste caratteristiche le applicazioni che vengono sviluppate possono contare su una sorgente dati sicura, affidabile e generalmente scalabile.

4.3.1 Il modello relazionale

I database relazionali sono il tipo di database attualmente più diffuso

I motivi di questo successo sono fondamentalmente due:

- forniscono sistemi semplici ed efficienti per rappresentare e manipolare i dati;
- si basano su un modello, quello relazionale, con solide basi teoriche.

Il modello relazionale e' stato proposto originariamente da E.F. Codd in un ormai famoso articolo del 1970 **[5]**

Grazie alla sua coerenza ed usabilità, il modello e' diventato negli anni '80 quello più utilizzato per la produzione di DBMS.

La struttura fondamentale del modello relazionale è appunto la "relazione", cioè una tabella bidimensionale costituita da righe (tuple) e colonne (attributi) **[3].**

Le relazioni rappresentano le entità che si ritiene essere interessanti nel database.

Ogni istanza dell'entità troverà posto in una tupla della relazione, mentre gli attributi della relazione rappresenteranno le proprietà dell'entità

In realtà, volendo essere rigorosi, una relazione è solo la definizione della struttura della tabella, cioè il suo nome e l'elenco degli attributi che la compongono.

Quando essa viene popolata con delle tuple, si parla di "istanza di relazione"

Le tuple in una relazione sono un insieme nel senso matematico del termine, cioè una collezione non ordinata di elementi differenti.

Per distinguere una tupla da un'altra si ricorre al concetto di "chiave primaria", cioè ad un insieme di attributi che permettono di identificare univocamente una tupla in una relazione.

Gli attributi della chiave primaria non possono assumere il valore NULL (che significa un valore non determinato), in quanto non permetterebbero più di identificare una particolare tupla in una relazione.

Questa proprietà delle relazioni e delle loro chiavi primarie va sotto il nome di integrità delle entità.

Ogni attributo di una relazione è caratterizzato da un nome e da un dominio.

Il dominio indica quali valori possono essere assunti da una colonna della relazione. Spesso un dominio viene definito attraverso la dichiarazione di un tipo per l'attributo (ad esempio dicendo che consiste una stringa di venti caratteri).

Caratteristica fondamentale dei domini di un database relazionale è che siano "atomici", cioè che i valori contenuti nelle colonne non possano essere separati in valori di domini più semplici.

Formalmente si dice che non è possibile avere attributi multivalore (multivalued), per questo motivo sono state inserite le foreign key (chiavi esterne).

Una chiave esterna è una combinazione di attributi di una relazione che sono chiave primaria per un'altra relazione.

Una caratteristica fondamentale dei valori presenti in una chiave esterna è che, a meno che non siano NULL, devono corrispondere a valori esistenti nella chiave primaria della relazione a cui si riferiscono.

Uno dei grandi vantaggi del modello relazionale è che esso definisce anche un algebra, chiamata appunto "algebra relazionale".

Tutte le manipolazioni possibili sulle relazioni sono ottenibili grazie alla combinazione di cinque soli operatori[27]: RESTRICT, PROJECT, TIMES, UNION e MINUS.

Per comodità sono stati anche definiti tre operatori addizionali che possono essere ottenuti applicando i soli cinque operatori fondamentali: JOIN, INTERSECT e DIVIDE.

Gli operatori relazionali ricevono come argomento una relazione o un insieme di relazioni e restituiscono una singola relazione come risultato.

In breve, si vedono questi otto operatori:

RESTRICT: restituisce una relazione contenente un sottoinsieme delle tuple della relazione a cui viene applicato. Gli attributi rimangono gli stessi.

PROJECT: restituisce una relazione con un sottoinsieme degli attributi della relazione a cui viene applicato. Le tuple della relazione risultato vengono composte dalle tuple della relazione originale in modo che continuino ad essere un insieme in

senso matematico.

TIME: viene applicato a due relazioni ed effettua il prodotto cartesiano delle tuple. Ogni tupla della prima relazione viene concatenata con ogni tupla della seconda.

JOIN: vengono concatenate le tuple di due relazioni in base al valore di un insieme dei loro attributi.

UNION: applicando questo operatore a due relazioni compatibili, se ne ottiene una contenente le tuple di entrambe le relazioni. Due relazioni sono compatibili se hanno lo stesso numero di attributi e gli attributi corrispondenti nelle due relazioni hanno lo stesso dominio.

MINUS: applicato a due relazioni compatibili, ne restituisce una terza contenente le tuple che si trovano solo nella prima relazione.

INTERSECT: applicato a due relazioni compatibili, restituisce una relazione contenente le tuple che esistono in entrambe le relazioni.

DIVIDE: applicato a due relazioni che abbiano degli attributi comuni, ne restituisce una terza contenente tutte le tuple della prima relazione che possono essere fatte corrispondere a tutti i valori della seconda relazione.

I database relazionali compiono tutte le operazioni sulle tabelle utilizzando l'algebra relazionale, anche se normalmente non permettono all'utente di utilizzarla.

L'utente interagisce con il database attraverso un'interfaccia differente, il linguaggio SQL, un linguaggio dichiarativo che permette di descrivere insiemi di dati.

Le istruzioni SQL vengono scomposte dal DBMS in una serie di operazioni relazionali.

4.3.2 Introduzione a MySQL

MySQL è un relational database management system (RDBMS) formato da una componente detta client che permette l'accesso ai servizi e alle risorse di un server

Consiste in un software libero rilasciato a licenza GNU (General Public License) e viene sviluppato per essere il più possibile conforme agli standard relativi al linguaggio SQL (ANSI SQL e ODBC SQL).

I sistemi e i linguaggi di programmazione che supportano MySQL sono molto numerosi: (ODBC, Java, Mono, .NET, php, Python) anche se in questo caso viene preferito l'utilizzo di php.

Il codice sorgente di MySQL era inizialmente di proprietà della società MySQL AB ma veniva però distribuito con la licenza GNU GPL oltre che con una licenza commerciale.

Fino alla versione 4.0, una buona parte del codice del client era licenziato con la GNU LGPL e poteva dunque essere utilizzato anche per applicazioni proprietarie.

Dalla versione 4.1 in poi, anche il codice del client è distribuito sotto GNU GPL.

Esiste peraltro una clausola estensiva che consente l'utilizzo di MySQL con una vasta gamma di licenze libere.

I suoi principali introiti provenivano dal supporto agli utilizzatori di MySQL tramite il pacchetto Enterprise, dalla vendita delle licenze commerciali e dall'utilizzo da parte di terzi del marchioMySQL.

Il 16 gennaio 2008 Sun Microsystems ha acquistato la società per un miliardo di dollari, stimando il mercato del database in 15 miliardi di dollari.

Il 20 aprile 2009 alla stessa Sun Microsystems è stata proposta l'acquisizione da parte di Oracle per 7,4 miliardi di dollari.

L'accordo, approvato dall'antitrust USA, è poi passato al vaglio degli organi corrispondenti dell'Unione Europea, preoccupati dal conflitto di interessi costituito dai database commerciali Oracle rispetto a MySQL.

Il padre di MySQL, Michael Widenius, ha lanciato una petizione online per opporsi alla fusione.

Nonostante ciò l'Unione Europea ha dato parere favorevole, e l'acquisizione è stata completata il 27 gennaio 2010.

Il software MediaWiki, che gestisce i siti del progetto Wikipedia, è basato su

database MySQL.

La scelta di MySQL è stata presa in considerazione sulla base dei seguenti punti di forza che tale software offre nell'ambito della gestione del database[47]:

- facilità d'uso, in quanto MySQL è un database ad alte prestazioni ma relativamente semplice e molto meno complicato da installare e amministrare rispetto ad altri sistemi di maggiori dimensioni;
- supporto del linguaggio di query;
- capacità, vale a dire il server è in grado di gestire contemporaneamente connessioni da più client, ed è possibile accedervi utilizzando diverse interfacce che permettono di eseguire query e visualizzare i risultati. Altra possibilità consiste nella scelta di una grande quantità di interfacce di programmazione per differenti linguaggi quali C, Perl, Java, Python e php (il linguaggio di programmazione scelto che verrà meglio definito nei prossimo paragrafi);
- connettività e sicurezza, in quanto MySQL è abilitato all'uso della rete e i database possono essere accessibili da dovunque in Internet, in modo tale da poter condividere i dati in esso contenuti. Il server, quando il database è accessibile online, andrà a disporre un meccanismo di controllo degli acessi in modo che chi non ha i diritti necessari per vedere i dati, non può farlo. Per fornire una maggiore sicurezza, il server collegato a MySQL supporta connessioni criptate attraverso specifici protocolli (come il Secure Sockets Layer, SSL);
- dimensioni ridotte, perché il software occupa uno spazio su hard disk molto minore, paragonato ai prodotti che si trovano oggi in commercio;
- disponibilità e costi, essendo un progetto Open Source che risulta scaricabile liberamente nei termini della GNU General Public License (GPL).

Per quanto riguarda il supporto, nelle distribuzioni MySQL è incluso il manuale di riferimento, disponibile anche online al seguente indirizzo (**http://dev.mysql.com/doc/refman/5.6/en/**) che beneficia regolarmente di numerose annotazioni da parte della comunità di utilizzatori.

4.3.3 Amministrazione di MySQL : phpMyAdmin

Esistono diversi tipi di MySQL Manager, ovvero di strumenti per l'amministrazione di MySQL.

Uno dei programmi più popolari per amministrare i database MySQL è phpmyAdmin[42].

PhpmyAdmin è un'applicazione web scritta in php, rilasciata con licenza GPL, che consente di amministrare un database MySQL tramite un qualsiasi browser.

Le potenzialità che hanno spinto all'uso di questo programma sono:

- la facilità di comprendere dei comandi di base del linguaggio SQL attraverso un'interfaccia grafica intuitiva.
- la possibilità di creare una database da zero, sfruttando le maschere e le opzioni di default.

Attraverso questo programma è possibile gestire i permessi, prelevati da MySQL, relativi alle operazioni che gli utenti possono fare sul database e definire gli amministratori.

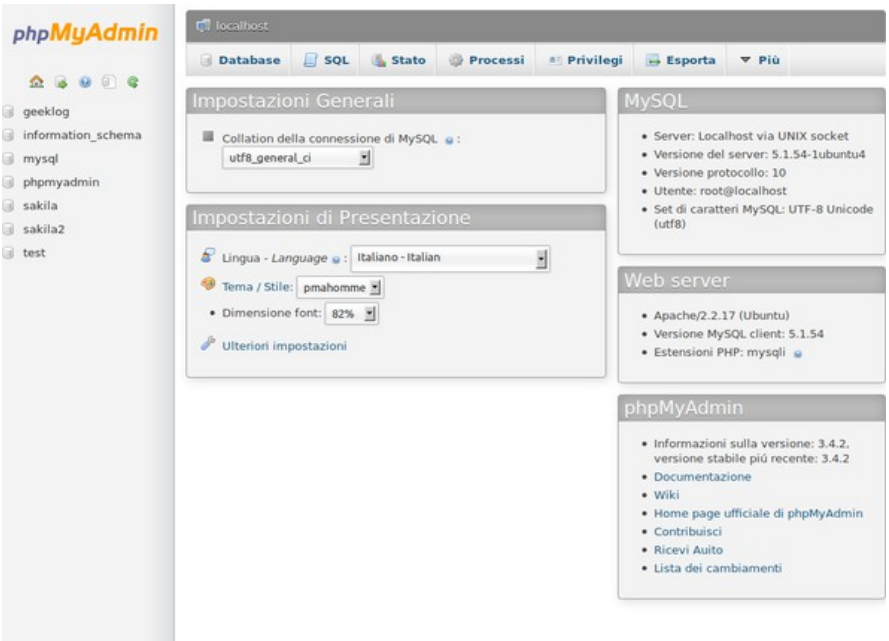

Figura 4.12: Home di phpMyAdmin

Dal punto di vista dell'amministrazione, il programma rende disponibile un'interfaccia grafica per la gestione degli utenti: l'interfaccia permette l'inserimento di un nuovo utente, la modifica della relativa password e la gestione dei permessi che l'utente ha sul database.

Un aspetto interessante consiste nella presenza di un feedback sulla creazione delle tabelle che permette di individuare eventuali errori.

Inoltre esistono tutte quelle funzionalità per l'inserimento dei dati (popolazione del database), per tute le query nel linguaggio SQL e per il backup dei dati.

La home di questo programma si presenta come in fig. 4.12.

4.3.4 Tipi di tabelle MySQL (Storage Engine)

MySQL permette di utilizzare numerosi tipi diversi di tabelle, ovvero diversi "storage engine" (motori di archiviazione) per la memorizzazione dei dati [28]

La distinzione più importante fra i diversi sistemi è quella fra transazionali e non transazionali.

I motori transazionali generalmente sono più sicuri (permettono di recuperare i dati anche in caso di crash di MySQL o di problemi hardware) consentendo di effettuare più modifiche e di convalidarle tutte insieme oppure ripristinare la situazione preesistente se qualcosa dovesse andar male.

Dal canto loro, i motori non transazionali hanno il vantaggio di una maggiore velocità, minore utilizzo di spazio su disco e minor richiesta di memoria per gli update.

Esistono diverse tipologie di tabelle (MyISAM, Merge,InnoDB,CSV ecc.), ma in questo caso è stato scelto lo storage engine InnoDB, che risulta il migliore per lavorare alla realizzazione del database contenente i dati ondametrici .

Le caratteristiche che permettono il suo ottimo uso nella gestione dei dati ondametrici, sono dovuti dalla presenza di un buffer che memorizza nella cache gli aggiornamenti e gli applica in background. Questo può notevolmente accelerare gli inserimenti dei dati, riducendo il numero di scritture fisiche richieste.

Inoltre è possibile utilizzare attributi, come ROW_FORMAT o KEY_BLOCK_SIZE, che durante la creazione delle tabelle consentono la compressione del contenuto (la compressione dei dati) diminuendo lo spazio necessario alla memorizzazione.

InnoDB è uno storage engine transazionale ottimizzato per l'uso concorrente dei dati fra molti utenti e per essere molto performante su grandi quantità di dati.

La configurazione della tabella InnoDB passa attraverso la definizione delle opzioni ma in questo lavoro di tesi sono state utilizzate le opzioni di default già presenti nel sistema.

Sotto si può osservare un esempio di configurazione tipica:

```
innodb_data_home_dir =/ibdata
innodb_data_file_path=ibdata 1:50M;ibdata2:50M:autoextend:max:500M
innodb_buffer_pool_size=70M
innodb_additional_mem_pool_size=10M
```

innodb_log_group_home_dir = /iblogs

innodb_log_files_in_group = 2

innodb_log_file_size=20M

innodb_log_buffer_size=8M

innodb_flush_log_at_trx_commit=1

innodb_lock_wait_timeout=50

skip-external-locking

max_connections=200

read_buffer_size=1M

sort_buffer_size=1M

innodb_data_home_dir indica la directory per il file dei dati.

Se non viene indicata, InnoDB utilizza la directory dati di MySQL e se viene indicata vuota, si deve indicare il percorso completo su innodb_data_file_path.

Si possono effettuare più indicazioni, nell'esempio appena visto si ha un file ibdata1 di 50 megabytes e un file ibdata2 di 50 megabytes che potrà estendersi fino a 500.

L'opzione autoextend può essere indicata solo sull'ultimo file.

innodb_buffer_pool_size dovrebbe essere circa la metà della memoria del PC dove è avvenuta l'installazione di MySQL.

innodb_log_group_home_dir è la directory per i file di log.

Se non viene indicata verrà usata la stessa dei dati.

InnoDB consente l'uso delle transazioni e questo significa che ad ogni connessione al server, MySQL inizia in autocommit mode cioè tutte le istruzioni di aggiornamento vengono rese effettive immediatamente.

Se venisse disattivato l'autocommit con il comando SET AUTOCOMMIT = 0, le modifiche diverranno operative solo quando verrà eseguita l'istruzione COMMIT.

Se al suo posto si esegue ROLLBACK, verranno annullate tutte le modifiche fino alla COMMIT precedente.

InnoDB gestisce il valore AUTO_INCREMENT per una tabella calcolandolo la prima volta dopo l'avvio del server (ad esempio dopo aver inviato il comando INSERT), selezionando il valore massimo esistente sulla tabella e incrementandolo di 1.

A quel punto il valore viene conservato in memoria ma non scritto su disco, per

cui al riavvio successivo sarà ricalcolato.

Questo significa che se venissero cancellati gli ultimi valori della tabella senza fare nuovi inserimenti, al successivo riavvio il server riutilizzerà quei valori.

Con le tabelle InnoDB si possono definire le foreign key, cioè collegare i valori delle colonne che contengono chiavi di altre tabelle alle tabelle stesse.

In questo modo risulta possibile verificare automaticamente quando i valori della tabella madre vengono modificati o eliminati in modo da impedire queste modifiche o, al contrario, modificare di conseguenza anche i valori sulla tabella figlia.

Inoltre non è possibile inserire nella tabella figlia valori che non avranno un corrispondente nella tabella madre.

4.3.5 Tipi di dati

Le colonne che possono essere definite in una tabella MySQL sono di diversi tipi.

Tali tipi si possono suddividere in dati numerici, dati relativi a date e tempo, stringhe e dati geometrici.[29].

Si deve ricordare che tutti i tipi di colonne possono contenere (se dichiarato nella loro definizione) il valore NULL, previsto dallo standard SQL per indicare un "non valore".

Si osservano i tipi di dato (N.B. Le indicazioni comprese fra parentesi quadre sono opzionali.)

Per primi vengono presi in considerazione i dati numerici:

BIT[(M)]
TINYINT[(M)] [UNSIGNED] [ZEROFILL]
SMALLINT[(M)] [UNSIGNED] [ZEROFILL]
MEDIUMINT[(M)] [UNSIGNED] [ZEROFILL]
INT[(M)] [UNSIGNED] [ZEROFILL]
BIGINT[(M)] [UNSIGNED] [ZEROFILL]
FLOAT[(M,D)] [UNSIGNED] [ZEROFILL]
DOUBLE[(M,D)] [UNSIGNED] [ZEROFILL]
DECIMAL[(M[,D])] [UNSIGNED] [ZEROFILL]

Tutti i dati numerici escluso il BIT possono avere le opzioni UNSIGNED e ZEROFILL.

Con la prima si specifica che il numero è senza segno, per cui non saranno consentiti valori negativi mentre con la seconda si indica al server di memorizzare i numeri con degli zeri davanti nel caso in cui la lunghezza sia inferiore a quella massima definita o prevista.

Con l'uso di ZEROFILL, MySQL aggiunge automaticamente UNSIGNED.

Il dato di tipo BIT contiene il numero di bit specificato con M (1 per default), che può andare da 1 a 64.

I dati di tipo TINYINT, SMALLINT, MEDIUMINT, INT e BIGINT rappresentano numeri interi composti rispettivamente da 1, 2, 3, 4 e 8 bytes.

Il TINYINT può contenere 256 valori, che vanno da -128 a +127 oppure da 0 a 255 nel caso di UNSIGNED.

Allo stesso modo, SMALLINT può contenere 65536 valori, MEDIUMINT 16.777.216, INT oltre 4 miliardi, BIGINT circa 18 miliardi di miliardi.

In tutti i casi i valori massimi assoluti vanno dimezzati se non venisse usato UNSIGNED.

FLOAT e DOUBLE rappresentano i numeri in virgola mobile.

M rappresenta il numero totale di cifre rappresentate e D il numero di cifre decimali.

FLOAT è a "precisione singola": i suoi limiti teorici vanno da -3.402823466E+38 a -1.175494351E-38 e da 1.175494351E-38 a 3.402823466E+38, oltre allo zero.

I valori DOUBLE sono invece a "precisione doppia": i limiti teorici sono da -1.7976931348623157E+308 a -2.2250738585072014E-308 e da 2.2250738585072014E-308 a 1.7976931348623157E+308, oltre allo zero.

Per entrambi i tipi di dato i limiti reali dipendono dall'hardware e dal sistema operativo.

Se M e D non vengono indicati come valori possono essere memorizzati fino ai limiti effettivi.

Per questi dati l'uso di UNSIGNED disabilita i valori negativi, ma non ha effetto sui valori massimi positivi memorizzabili.

La precisione dei numeri in virgola mobile risulta affidabile fino (circa) alla settima cifra decimale per i FLOAT e alla quindicesima per i DOUBLE.

I dati DECIMAL rappresentano infine numeri "esatti", con M cifre totali di cui D decimali dove i valori di default sono 10 per M e 0 per D.

Il massimo di cifre consentite è 65 per M e 30 per D.

Per quanto riguardano i formati di date e tempo:

DATE

DATETIME

TIMESTAMP[(M)]

TIME

YEAR[(2|4)]

Una colonna DATE può contenere date da ’1000-01-01’ (1° gennaio 1000) a ’9999-12-31’ (31 dicembre 9999).

Una colonna DATETIME contiene una data e un’ora nel formato ‘AAAA-MM-GG HH:MM:SS’.

Nel caso delle date esistono dei comandi che permettono di effettuare controlli sugli eventuali errori che possono essere fatti durante la scrittura delle query necessarie al loro inserimento e/o errori sul formato.

Tali comandi sono i seguenti:

- ALLOW_INVALID_DATES, necessario per consentire date non valide;
- NO_ZERO_DATE non verranno accettate date a 0 (’0000-00-00’);
- NO_ZERO_IN_DATE non accetta valori 0 per giorno e mese.

In un TIMESTAMP possono essere memorizzati i valori corrispondenti al timestamp Unix, che vanno dalla mezzanotte del 1° gennaio 1970 ad un momento imprecisato dell’anno 2037.

Questo tipo di dato risulta utile per memorizzare automaticamente il momento dell’aggiornamento di una riga di tabella, infatti MySQL può impostare in automatico una colonna TIMESTAMP di una tabella nel momento in cui viene effettuata una INSERT o un UPDATE.

Una colonna TIME contiene un valore di tempo (ore, minuti e secondi) che va da ‘-838:59:59’ a ’838:59:59’.

Infine la colonna YEAR rappresenta, su quattro cifre, un anno compreso fra 1901 e 2155, oppure 0000 mentre su due cifre invece i valori vanno da 70 (1970) a 69 (2069).

I valori relativi al tempo possono essere inseriti sia come stringhe che come numeri e MySQL consente di utilizzare, nel caso delle stringhe, molti caratteri diversi come separatori ma l’importante è che l’ordine dei valori sia sempre anno-mese-giorno-ore-minuti-secondi.

Nel caso della scrittura di stringhe si può dire che:

le colonne di tipo stringa possono avere un attributo CHARACTER SET che indica l’insieme di caratteri utilizzato per la colonna, e un attributo COLLATE che indica la collation relativa (la collation è quell'insieme di informazioni relativo all'alfabeto

scelto).

Un esempio:

```
CREATE TABLE tabella
(
  c1 CHAR(20) CHARACTER SET utf8,
  c2 CHAR(20) CHARACTER SET latin1 COLLATE latin1_bin
);
```

In questa tabella si avrà la colonna c1 definita col set di caratteri utf8 e la relativa collation di default mentre la colonna c2 col set di caratteri latin1 e la relativa collation binaria.

La lunghezza specificata è relativa al numero di caratteri (il numero di byte infatti può variare in base ai set di caratteri usati e al contenuto della colonna).

I tipi di campi previsti:

[NATIONAL] CHAR(M) [BINARY | ASCII | UNICODE]
[NATIONAL] VARCHAR(M) [BINARY]
BINARY(M)
VARBINARY(M)
TINYBLOB
TINYTEXT
BLOB[(M)]
TEXT[(M)]
MEDIUMBLOB
MEDIUMTEXT
LONGBLOB
LONGTEXT
ENUM('valore1','valore2',...)
SET('valore1','valore2',...)

CHAR consiste in una stringa di lunghezza fissa (*M*) riempita con spazi a destra al momento della memorizzazione, che vengono eliminati in fase di lettura.

La lunghezza prevista va da 0 a 255 caratteri.

L’opzione NATIONAL indica che la stringa deve usare il set di caratteri di default. L’attributo BINARY indica che deve essere usata la collation binaria del set di caratteri utilizzato.

CHAR BYTE equivale a CHAR BINARY ed è da notare come se una riga ha lunghezza variabile (cioè se almeno una colonna è definita a lunghezza variabile) qualsiasi campo CHAR di lunghezza superiore a 3 caratteri viene convertito in VARCHAR.

VARCHAR consiste in una stringa a lunghezza variabile dove la lunghezza massima dichiarabile equivale a 65535 caratteri.

Gli attributi NATIONAL e BINARY hanno lo stesso significato visto in CHAR.

BINARY e VARBINARY corrispondono a CHAR e VARCHAR ma memorizzano stringhe di byte invece che di caratteri, infatti non hanno character set.

I formati di tipo BLOB e TEXT sono utilizzati rispettivamente per valori binari e di testo e la loro lunghezza massima varia sulla base di come vengono dichiarati.

Per TINYBLOB e TINYTEXT è di 255 caratteri, 65535 per BLOB e TEXT, 16.777.215 per MEDIUMBLOB e MEDIUMTEXT, 4 gigabyte per LONGBLOB e LONGTEXT.

L'ultimo caso preso in considerazione, riguarda i dati geometrici:

i dati geometrici si basano sulle specifiche dell’Open GIS.

Vengono utilizzati nelle applicazioni dove il database MySQL viene usato come sistema di archiviazione dei dati nei sistemi informatici territoriali.

La potenzialità di questo utilizzo viene messa in risalto dalla possibilità di attribuire ad una coppia di coordinate geografiche che indicano un punto su una carta georeferenziata, una serie di dati specifici.

Sotto vediamo quali sono i tipi di dati geometrici previsti da MySQL:

GEOMETRY
POINT
LINESTRING
POLYGON
MULTIPOINT
MULTILINESTRING
MULTIPOLYGON
GEOMETRYCOLLECTION

Il significato di ogni tipo di dato è così riassunto:

- GEOMETRY può contenere un valore geometrico generico;
- POINT contiene un punto;
- LINESTRING una linea;
- POLYGON un poligono;
- GEOMETRYCOLLECTION rappresenta un insieme di dati geometrici di qualsiasi tipo.

4.4 Il linguaggio di programmazione PHP

La caratteristica significativa del php consiste nel suo essere definito "linguaggio di programmazione lato server."

Questo significa che quando il server riceve una richiesta da parte di un utente per una pagina php, la fa analizzare dall'interprete del linguaggio e restituisce un file contenente solo il codice che deve essere inviato al browser (in linea di massima HTML, ma potrebbe esserci anche codice JavaScript, fogli di stile CSS o qualunque altro contenuto fruibile da un browser, come immagini e documenti Pdf).

Questa è la base per un sito web dinamico.

La funzione principale, quindi, è quella di creare il codice della pagina che viene spedita dall'utente per cui questo linguaggio di programmazione permette l'implementazione di soluzioni avanzate per un controllo completo sulle operazioni svolte dal server.

4.4.1 La produzione del codice

Una componente fondamentale della programmazione in php consiste nella capacità dell'interprete di riuscire a comprendere quale sia il codice da elaborare rispetto a quello da restituire all'utente[30].

Questa fase di riconoscimento permette a php di essere incluso all'interno di un normale codice HTML in modo da rendere dinamica la creazione di un'applicazione web (per esempio, una pagina contenente dati aggiornati secondo un certo intervallo di tempo)[31].

Il codice php deve essere compreso fra appositi tag di apertura e di chiusura, che sono i seguenti:

<?php //tag di apertura

?> //tag di chiusura

Tutto ciò che è contenuto fra questi tag deve corrispondere alle regole sintattiche del php, ed è il codice che sarà eseguito dall'interprete .

Da notare il simbolo // che sta a indicare un commento.

I commenti svolgono un ruolo fondamentale in fase di costruzione del codice, in quanto facilitano la comprensione di passaggi apparentemente oscuri per chi non è a conoscenza delle finalità del programma.

Il loro utilizzo non appesantisce l'esecuzione dello script,in quanto l'interprete php salta tutte le parti che riconosce come commenti, ne il trasferimento della pagina al browser.

In figura 4,13 si può vedere vede un semplice esempio, composto da codice HTML e codice php[31].

Il codice HTML svolge la funzione di "cornice" rispetto al codice php, in quanto realizza la struttura della pagine web nel quale deve essere visualizzato il contenuto dell'elaborazione svolta del codice php.

In questo caso, il codice HTML elabora la struttura attorno al comando php "echo" che ha la funzione di restituire a schermo ciò che viene dichiarato (Buona Giornata!).

Il file può essere richiamato dall'utente attraverso un browser e il server, che ha il compito di interpretare il codice, restituisce una pagina web contenente la riga "Buona Giornata!"

```
<html>
<head>
  <title>
    <?php
      echo "Pagina di prova PHP";
    ?>
  </title>
</head>
<body>
  <?php
    echo "Buona giornata!";
  ?>
</body>
</html>
```

Figura 4.13: esempio di codice PHP con HTML

Da notare bene che il dato inviato al browser che segue il comando echo deve essere racchiuso tra virgolette .

Inoltre, possono essere presenti contemporaneamente più stringhe sulla stessa riga (questo è il nome che viene dato ad una ripetizione di qualunque carattere compreso tra due apici singoli (' ') o doppi (" ")).

Quando l'interprete del php trova questa combinazione di caratteri "<br>", li trasforma in un carattere di ritorno a capo.

L'esempio appena visto è rappresentativo di una pagina di controllo dell'impaginazione del codice HTML attraverso il codice php

In assenza del tag (nome con cui vengono definiti i comandi che modificano in HTML) "<br >", il browser allinea tutto il testo proseguendo sulla singola linea

anche se ciò che viene scritto ha un ritorno a capo.

Il concetto è reso più chiaro attraverso i seguenti esempi:

```
<?php
echo "prima riga";
echo "seconda riga"; <br >
echo "terza riga";
?>
```

Questo codice php insieme ad HTML viene visto dall'utente nel seguente modo:

prima riga seconda riga

terza riga

Aspetto fondamentale nella scrittura del codice php è l'uso del punto e virgola (;) alla fine di ogni istruzione in quanto la sintassi prevede che questa punteggiatura debba obbligatoriamente chiudere le istruzioni stesse.

4.4.2 Variabili del linguaggio php

Una variabile può essere immaginata come una specie di contenitore all'interno del quale viene conservato un valore.

Tale valore può essere cambiato attraverso l'assegnazione, vale a dire ponendo la variabile uguale ad un altro valore[32].

In php si può scegliere il nome delle variabili usando lettere, numeri ed il trattino di sottolineatura [underscore (_)] ma il primo carattere del nome deve essere obbligatoriamente una lettera o un underscore (non un numero).

Altro aspetto di cui tener conto è la sensibilità del nome all'uso delle maiuscole e minuscole.

Se viene scritto due volte un nome di variabile usando la stessa lettera ma in maniera differente, php comprende che esistono due variabili distinte.

A livello di sintassi del linguaggio di programmazione, per dichiarare una variabile si dovrà precedere il nome con cui verrà individuata dal simbolo $.

Php ha una caratteristica che lo rende molto più flessibile rispetto ad altri linguaggi di programmazione che consiste nella non dichiarazione a priori delle variabile, cioè è possibile fare riferimento inizializzando solamente il valore assegnato.

L'uso della variabile diventa fondamentale nel momento in cui è possibile utilizzarla all'interno di espressioni matematiche o logiche.

4.4.3 Tipi di dato

Una variabile può contenere diversi tipi di valori, ognuno dei quali ha un comportamento ed un'utilità differente[33].

Di seguito, vengono analizzati brevemente i tipi di dato che php permette di utilizzare all'interno del proprio codice premettendo che questo linguaggio di programmazione, a differenza di altri linguaggi, associa il tipo di dato al valore e non alla variabile (ad esempio si può assegnare alla stessa variabile una stringa e poi un numero senza incorrere in alcun errore) ed effettua conversioni automatiche dei valori nel momento in cui siano richiesti tipi di dato differenti (ad esempio in un'espressione)[23].

La prima tipologia presa in considerazione sono i **valori booleani**[35]:

Il tipo di dato booleano serve a rappresentare i valori vero o falso all'interno di espressioni logiche.

```
<?php
$vero = true;
$falso = false;
?>
```

Altra tipologia è la **virgola mobile**[23] che consiste in un numero decimale (a volte citato come "double" o "real").

Da notare che per indicare i decimali, non verrà usata la virgola ma il punto mentre la precisione relativa al numero che si potrà attribuire alla variabile sarà di 14 cifre decimali.

Si possono utilizzare le seguenti sintassi:

```
<?php
$vm1 = 4.153; // 4,153
$vm2 = 3.2e5; // 3,2 * 10^5, cioè 320.000
$vm3 = 4E-8;  // 4 * 10^-8, cioè 4/100.000.000 = 0,00000004
?>
```

Come visto in precedenza, una **stringa** è un qualsiasi insieme di caratteri, senza

limitazione, normalmente contenuto all'interno di una coppia di apici doppi o apici singoli.

Le stringhe delimitate da apici sono la forma più semplice e viene consigliata quando all'interno della stringa non vi sono variabili di cui vogliamo ricavare il valore:

```
<?php
$frase = 'Anna disse: "Ciao a tutti!" ma nessuno rispose';
echo $frase;
?>
```

Questo codice stampa la seguente frase:

Anna disse: "Ciao a tutti!" ma nessuno rispose.

Gli apici doppi permettono di usare le stringhe in una maniera più precisa in quanto, se all'interno della stringa delimitata da virgolette php riconosce un nome di variabile, lo sostituisce con il valore della variabile stessa.

```
<?php
$nome = 'Anna';
echo "$nome è simpatica... a pochi"; // stampa: Anna è simpatica... a pochi
echo '$nome è simpatica... a pochi'; // stampa: $nome è simpatica... a pochi
echo "{$nome} è simpatica a pochi";  // è una sintassi alternativa, con lo stesso effetto della prima
?>
```

Esistono un paio di regole molto importanti quando vengono usate le stringhe delimitate da apici o virgolette in quanto può capitare che una stringa debba contenere a sua volta un apice o un paio di virgolette e si ha la necessità di un sistema per far capire a php che quel carattere fa parte della stringa e non è il suo delimitatore.

n questo caso viene usato il cosiddetto 'carattere di escape', cioè la barra rovesciata (backslash: '\'):

```
<?php
echo 'Torniamo un\'altra volta'; // stampa: Torniamo un'altra volta
echo "Torniamo un'altra volta";  // stampa: Torniamo un'altra volta
```

```
echo "Torniamo un\'altra volta"; // stampa: Torniamo un\'altra volta
echo 'Torniamo un'altra volta';  // causa un errore, perché l'apostrofo viene scambiato per
                                 l'apice di chiusura
echo 'Anna disse "Ciao" e se ne andò';   // stampa: Anna disse "Ciao" e se ne andò
echo "Anna disse \"Ciao\" e se ne andò"; // stampa: Anna disse "Ciao" e se ne andò
echo 'Anna disse \"Ciao\" e se ne andò'; // stampa: Anna disse \"Ciao\" e se ne andò
echo "Anna disse "Ciao" e se ne andò";   // errore
?>
```

Da questi esempi si capisce che il backslash dovrà essere utilizzato come carattere di escape quando si vorrà includere nella stringa lo stesso tipo di carattere che la delimita.

Una tipologia di dato ampiamente utilizzato, soprattutto per questo lavoro di tesi, è **l'array** [36]

Si può considerare un array come una variabile complessa, contenente una serie di valori, ciascuno dei quali caratterizzato da una chiave o indice che lo identifica univocamente.

Viene definito un array composto da cinque valori.

```
$colori = array('bianco', 'nero', 'giallo', 'verde', 'rosso');
```

A questo punto ciascuno dei cinque colori è caratterizzato da un indice numerico, che php assegna automaticamente a partire da 0.

Per recuperare un determinato valore dalla variabile che contiene l'array si andrà a specificare il suo indice all'interno di parentesi quadre dietro al nome della variabile:

```
echo $colori[1]; // stampa 'nero'
echo $colori[4]; // stampa 'rosso'
```

4.4.4 Espressioni e operatori

Gli operatori sono un altro degli elementi base di qualsiasi linguaggio di programmazione, in quanto consentono non solo di effettuare le tradizionali operazioni aritmetiche, ma più in generale di manipolare il contenuto delle variabili[34]

Il primo operatore preso in considerazione è **l'operatore di assegnazione**, di cui si vede un esempio:[34].

```
$nome = 'Giorgio' ;
```

Il simbolo '=' serve ad assegnare alla variabile $nome il valore'Giorgio'.

In generale, l'operatore di assegnazione prende ciò che sta alla destra del segno di ugualianza ('=') ed assegna lo stesso valore a ciò che sta a sinistra.

Con l'operatore di assegnazione si può anche usare una variabile per effettuare un calcolo il cui risultato deve essere assegnato alla variabile stessa.

Ad esempio,di avere una variabile di cui si vuole aumentare il valore:

```
$a = $a + 10; // il valore di $a aumenta di 10
```

Con questa istruzione, viene eseguito il calcolo che sta alla destra del segno '=' ed il risultato viene memorizzato nella variabile indicata a sinistra.

Risulta chiaro che il valore della variabile $a prima dell'istruzione viene utilizzato per il calcolo, ma dopo che l'istruzione è stata eseguita viene utilizzata per memorizzare il risultato, quindi il suo valore cambia.

Un risultato di questo tipo si può ottenere anche con gli operatori di assegnazione combinati, che ci permettono di rendere il codice più compatto.

Altri operatori sono quelli che permettono di effettuare operazioni aritmetiche sui dati come l'addizione, la sottrazione, la divisione, la moltiplicazione e il modulo. Operatori fondamentali per il calcolo sono quelli relativi a o**perazioni aritmetiche**[34]

```
$a = 3 + 7; // addizione
$b = 5 - 2; // sottrazione
$c = 9 * 6; // moltiplicazione
$d = 8 / 2; // divisione
$e = 7 % 4; // modulo
                    // (il modulo è il resto della divisione intera, in questo caso 3)

$x += 4;  // incrementa $x di 4 (equivale a $x = $x + 4)
$x -= 3;  // decrementa $x di 3 (equivale a $x = $x – 3)
$x .= $a; // il valore della stringa $a viene concatenato a $x (equivale a $x = $x . $a)
$x /= 5;  // equivale a $x = $x / 5
$x *= 4;  // equivale a $x = $x * 4
$x %= 2;  // equivale a $x = $x % 2
```

In questo modo si dichiara a php l'intenzione di volere assegnare alla variabile specificata a sinistra il risultato dell'operazione che si trova prima del simbolo uguale applicandola alla variabile stessa ed al valore specificato a destra.

Gli operatori utilizzati nei cicli, come si vedrà più avanti, sono **l'incremento e decremento**

```
$a++; ++$a; // incrementa di 1
$a--; --$a; // decrementa di 1
```

Questi funzionano nel caso in cui si voglia incrementare o decrementare una variabile di un una sola unità.

4.4.5 Gli operatori logici e le espressioni booleane

Gli operatori logici sono fondamentali in quanto permettono di far svolgere al programma determinate operazioni invece di altre attraverso il confronto di valori.

Dopo aver valutato un'espressione di questo tipo, php arriva a valutare se essa è vera o falsa per cui il risultato sarà di tipo booleano (true o false).[35]

Di seguito viene visualizzata un tabella (tab. 4.1) con tutte le operazioni che permettono il confronto:

Operatore	Descrizione
==	uguale
!=	diverso
===	identico (cioè uguale e dello stesso tipo: ad esempio per due variabili di tipo intero)
>	maggiore
>=	maggiore o uguale
<	minore
<=	minore o uguale

Tabella 4.1: gli operatori

Esempi:

```
$a = 7; $b = 7.0; $c = 4; //assegnazione valori a tre variabili
$a == $b;  // vero
$a == $c;  // falso
$a === $b; // falso, perché $a è intero mentre $b è float
$a > $c;   // vero
$c >= $a;  // falso, $c è minore di $a
$a < $b;   // falso, hanno lo stesso valore
$c <= $b;  // vero
```

Gli stessi confronti si possono fare anche con altri tipi di variabili, ed in particolare con le stringhe.

In questo caso il confronto viene realizzato sulla base dell'ordine alfabetico dei

caratteri, vale a dire che vengono considerati ‘minori’ i caratteri che ‘vengono prima’ nell’ordine alfabetico.

Quindi ‘a’ è minore di ‘b’, ‘b’ è minore di ‘c’, eccetera.

Inoltre tutte le lettere minuscole sono ‘maggiori’ delle lettere maiuscole, e tutte, maiuscole e minuscole, sono ‘maggiori’ delle cifre da 0 a 9:

```
10 > 8 And 7 < 6; // falso, perché la prima condizione è vera ma la seconda è falsa
10 > 8 Or 7 < 6;  // vero
9 > 5 And 5 == 5; // vero, entrambe le condizioni sono vere
9 > 5 Xor 5 == 5; // falso, solo una delle due deve essere vera perché si verifichi lo 'Xor'
4 < 3 || 7 > 9;   // falso, nessuna delle due condizioni è vera
6 == 6 && 1 > 4;  // falso, solo la prima condizione è vera
```

Per quanto riguarda gli operatori ‘and’ e ‘or’, le due diverse notazioni differiscono per il livello di precedenza in caso di espressioni complesse.

Infatti risulta possibile combinare molti operatori in espressioni anche assai complicate e viene necessario conoscere quale sia l'ordine con cui php valuta i diversi operatori.

Questo ordine ricalca le regole algebriche in base alle quali moltiplicazioni e divisioni hanno la precedenza su addizioni e sottrazioni ma nell'insieme saranno più complesse in quanto dovranno considerare anche gli altri operatori.

Adesso si osserva l’ordine di priorità dei diversi operatori, iniziando da quelli che hanno la priorità maggiore:

- Operatori di incremento e decremento (++ --);
- Moltiplicazione, divisione, modulo (* / %);
- Addizione e sottrazione (+ -);
- Operatori di confronto per minore e maggiore (< <= => >);
- Operatori di confronto per uguaglianza e disuguaglianza (== === !=);
- Operatore logico ‘and’, nella notazione col simbolo (&&);
- Operatore logico ‘or’, nella notazione col simbolo (||);
- Operatori di assegnazione, compresi quelli ‘sintetici’ (= += -= /= *= %= .=);
- Operatore logico ‘and’, nella notazione letterale (And);

- Operatore logico ‘xor’ (Xor);
- Operatore logico ‘or’, nella notazione letterale (Or).

Come avviene nell'algebra, l'uso delle parentesi permette di porre gli operatori dove si vuole, sulla base delle necessità di programmazione.

4.4.6 Strutture di controllo e cicli

Il primo comando che si va ad illustrare è **"if"**[24].

Le strutture di “controllo di flusso” rappresentano un aspetto fondamentale della programmazione in quanto da la possibilità al programmatore di realizzare programmi capaci di eseguire operazioni diverse, ed eventualmente di eseguirle più volte, in base alla valutazione di determinate condizioni.

L’istruzione if risulta la più rappresentativa tra le strutture di controllo del flusso perché permette di eseguire o non eseguire certe porzioni di codice.

Si vede la sintassi più elementare:

```
if ( < condizione > ) <istruzione>
```

Dopo l’"if", deve essere indicata fra parentesi un’espressione (condizione). che viene valutata in senso booleano, cioè il suo valore è considerato vero o falso per cui se una condizione risulta vera l'istruzione successiva viene eseguita oppure nel caso contrario ignorata.

Spesso, al verificarsi della condizione, si necessita di eseguire non una sola ma più istruzioni.

Questo è perfettamente possibile attraverso l'utilizzo dei blocchi di istruzioni compresi fra parentesi graffe, ad esempio:

```
if ($nome == 'Luca') {
        echo "ciao Luca!<br>";
        echo "dove sono i tuoi amici?<br>";
}
    echo "ciao a tutti voi";
```

Se la variabile "$nome" ha effettivamente il valore 'Luca' le istruzioni successive, comprese tra parentesi graffe, saranno eseguite.

Dopo aver valutato la condizione ed eventualmente eseguito le due istruzioni previste, lo script prosegue con ciò che sta fuori dalle parentesi graffe, quindi come da esempio viene prodotta la frase “ciao a tutti voi” in ogni caso.

Si nota che nel construtto mancano il punto e virgola in "if" e nella condizione espressa in quanto non devono averli mentre continuano ad esistere nel blocco di codice condizionato.

Dall'istruzione i"if" si può ottenere un valore booleano anche quando tale espressione non restituisce un valore del genere grazie al fatto che php converte tale valore in booleano utilizzando determinate regole.

```
If (5) {
print "Ciao Luca!";
}
```

Dal punto di vista logico il codice non ha alcun senso ma permette la comprensione di come questo linguaggio di programmazione interpreta l'espressione.

In questo caso la stringa "Ciao Luca!" verrà stampata, in quanto il valore 5, così come qualsiasi valore diverso da 0, è considerato vero.
In sostanza, php considera come falsi:

- il valore numerico 0, nonché una stringa che contiene '0';
- una stringa vuota;
- un array con zero elementi;
- un valore NULL, cioè una variabile che non è stata definita o che è stata eliminata con unset(valore), oppure a cui è stato assegnato il valore NULL esplicitamente; Qualsiasi altro valore, per PHP è un valore vero.

Un’ulteriore possibilità che viene fornita dall’istruzione if in php è quella di utilizzare la parola chiave elseif.

Attraverso "else" si può indicare una seconda condizione, da valutare solo nel caso in cui quella precedente risulti falsa.

Viene indicato di seguito il codice da eseguire nel caso in cui la condizione sia vera, ed eventualmente, con else, il codice previsto per il caso in cui anche la seconda condizione sia falsa.

```
If ($nome == 'Luca') {
        print "bentornato Luca!";
} elseif ($cognome == 'Verdi') {
        print "Buongiorno, signor Verdi";
} else {
        print "ciao $nome!";
}
```

In questo construtto si ha un'istruzione da eseguire quando $nome vale 'Luca' e nel caso in cui ciò non sia vero è prevista una seconda istruzione che si suddivide nel seguente modo: se $cognome sarà 'Verdi' allora verrà stampato "Buongiorno, signor Verdi" ma se nemmeno questo fosse vero, allora verrà eseguita la terza istruzione.

Altri comandi che sono risultati utili nella realizzazione del sistema di gestione dei dati sono **i cicli for, while [25].**

i cicli sono un altro degli elementi fondamentali di qualsiasi linguaggio di programmazione, in quanto permettono di eseguire determinate operazioni in maniera ripetitiva.

Il pericolo nell'uso dei cicli consiste nella realizzazione di una situazione dove non si raggiunga mai una via d'uscita (il cosi detto "infinite loop").

Nel caso del ciclo "for" si suppone di voler mostrare i multipli da 1 a 10 di un numero, ad esempio 5.

```
for ($mul = 1; $mul <= 10; ++$mul) {
$ris = 5 * $mul;
echo "5 * $mul = $ris <br/>";
}
```

Questo costrutto utilizza la parola chiave for seguita fra parentesi dalle istruzioni per definire il ciclo e di seguito si racchiudono fra parentesi graffe tutte le istruzioni che devono essere eseguite ripetutamente.

Le tre istruzioni inserite fra le parentesi tonde e separate da punto e virgola vengono trattate nel seguente questo modo:

•la prima viene eseguita una sola volta, all'inizio del ciclo;

•la terza viene eseguita alla fine di ogni iterazione del ciclo;

•la seconda deve essere una condizione, e viene valutata prima di ogni iterazione del ciclo.

Quando risulta falsa, l'esecuzione del ciclo viene interrotta, ed il controllo passa alle istruzioni dopo le parentesi graffe, invece quando è vera, le istruzioni fra parentesi graffe vengono eseguite.

Ovviamente risulta possibile che tale condizione risulti falsa fin dal primo test per cui in questo caso le istruzioni contenute fra le parentesi graffe non saranno eseguite nemmeno una volta.

Il ciclo realizzato si può definire un "contatore": con la prima istruzione lo si inizializza, con la seconda lo si confronta con un valore limite oltre il quale il ciclo deve terminare, con la terza lo si incrementa dopo ogni esecuzione.

Il ciclo "while" si può considerare come una specie di "if" ripetuto dove si prevede che alla parola chiave "while" segua, fra parentesi, la condizione da valutare e fra parentesi graffe, il codice da rieseguire fino a quando tale condizione rimane vera.

Vediamo con un esempio come ottenere lo stesso risultato dell'esempio relativo al "for" :

```
$mul = 1;
while ($mul <= 10) {
$ris = 5 * $mul;
print("5 * $mul = $ris<br>");
$mul++;
}
```

Il ciclo "while", rispetto al "for", non mette a disposizione le istruzioni per inizializzare e per incrementare il contatore.

Quindi si dovranno inserire queste istruzioni nel flusso generale del codice e ci sarà l'inizializzazione prima del ciclo con l'incremento all'interno del ciclo stesso nella parte finale.

Per uscire da un ciclo, si possono utilizzare due distinti comandi, **break** e **continue.**

Si è visto che php termina l'esecuzione di un ciclo quando la condizione a cui è sottoposto non è più verificata, ma esistono altri strumenti per modificare il comportamento dello script all'interno del ciclo.

Si può dire al programma di non completare la presente iterazione e passare alla successiva (con "continue") oppure di interrompere definitivamente l'esecuzione del ciclo (con "break").

Un esempio:

```
for ($ind = 1; $ind < 500; $ind++) {
if ($ind % 100 == 0) {
break;
} elseif ($ind % 25 == 0) {
continue;
}
echo "valore: $ind <br/>";
}
```

Questo codice imposta un ciclo per essere eseguito 500 volte, con valori di "$ind" che vanno da 1 a 500.

In realtà, le istruzioni poste al suo interno fanno sì che la stampa del valore di $ind non venga eseguita ogni volta che "$ind" corrisponde ad un multiplo di 25 e che il ciclo si interrompa del tutto quando "$ind" raggiunge il valore 100.

4.4.7 Gli Array

Precedentemente si è accennato al tipo di dato Array con il seguente esempio:

```
$colori = array ('bianco' , 'nero' , 'giallo' , 'verde' , 'rosso' );
```

Usando questo tipo di definizione, php associa a ciascuno dei valori elencati un indice numerico a partire da 0 (per esempio, 'bianco' assume l'indice 0, 'nero' l'indice 1, e così via fino a 'rosso' che ha indice 4.)

Per far riferimento ad un solo elemento dell'array si indica il nome dell'array seguito dall'indice contenuto fra parentesi quadre:

```
echo $colori [2]; //stampa 'giallo'
```

Risulta possibile aggiungere un valore all'array anche se questo è stato già definito:

```
$colori[ ] = 'blu'
```

Con questo codice verrà creato un nuovo elemento nell'array $colori, che avrà l'indice 5.

Se l'array $colori non fosse stato ancora definito, questa istruzione lo creerebbe[36] .

Naturalmente risulta possibile indicare direttamente l'indice, anche in modo non consecutivo:

```
$colori [3] = 'arancio';
$colori[7] = 'viola';
```

Dopo questa istruzione, l'elemento con indice 3, che prima valeva 'verde', ha il valore cambiato in 'arancio' e inoltre si ha un nuovo elemento con indice 7 avente valore 'viola'.

Adesso l'array presenta un "buco" in quanto dal codice 5 si salta direttamente al

codice 7.

Successivamente, se verrà usata di nuovo l'istruzione di "incremento" con le parentesi quadre vuote, il nuovo elemento prenderà l'indice 8 perché php va a cercare l'elemento con l'indice più alto e lo aumenta di 1 per creare quello nuovo.

L'argomento degli array può essere anche una stringa:

```
$persona ['nome'] =  Mario
```

Con questa istruzione è stato definito un array di nome $persona, creando un elemento la cui chiave è 'nome' ed il cui valore è 'Mario'

Da notare che quando vengono usate le chiavi associative, queste sono racchiuse fra apici che permettono il mantenimento della pulizia del codice realizzato.

Si vede qualche esempio di creazione e stampa dei valori di un array:

```
$persona['nome'] = 'Mario'; //corretto
$persona[cognome] = 'Rossi'; /*funziona, ma genera un errore 'notice'*/
echo $persona['cognome']; //stampa 'Rossi': corretto
echo "ciao $persona[nome]"; /*stampa 'ciao Mario': corretto (niente apici fra
virgolette)*/
echo "ciao $persona['nome']"; //NON FUNZIONA, GENERA ERRORE
echo "ciao {$persona['nome']}"; /*corretto: per usare gli apici fra virgolette dobbiamo
comprendere il tutto fra parentesi graffe*/
echo "ciao " . $persona['nome']; /*corretto: come alternativa, usiamo il punto per
concatenare  */
```

Si possono creare strutture complesse di dati, attraverso gli array a più dimensioni.

Un array a più dimensioni è composto da uno o più elementi che a loro volta sono degli array.

Supponiamo di voler raccogliere in un array i dati anagrafici di più persone: per ogni persona si va a registrare nome, cognome, data di nascita e città di residenza.

```
persone = array( array('nome' => 'Mario', 'cognome' => 'Rossi', 'data_nascita' =>
'1973/06/15', 'residenza' => 'Roma'), array('nome' => 'Paolo', 'cognome' => 'Bianchi',
'data_nascita' => '1968/04/05', 'residenza' => 'Torino'), array('nome' => 'Luca',
'cognome' => 'Verdi', 'data_nascita' => '1964/11/26', 'residenza' => 'Napoli'));
print $persone[0]['cognome']; // stampa 'Rossi'
print $persone[1]['residenza']; // stampa 'Torino'
print $persone[2]['nome']; // stampa 'Luca'
```

Questo codice va a definire un array formato a sua volta da tre array, elencati separati da virgole, per cui ciascuno di essi ha ricevuto l'indice numerico a partire da 0.

Da notare che, sebbene in questo caso ciascuno dei tre array 'interni' abbia la stessa struttura, in realtà è possibile dare a ciascun array una struttura autonoma.

```
$persone = array( 1 => array('nome' => 'Mario Rossi', 'residenza' => 'Roma', 'ruolo' =>
'impiegato'), 2 => array('nome' => 'Paolo Bianchi', 'data_nascita' => '1968/04/05',
'residenza' => 'Torino'), 'totale_elementi' => 2);
print $persone[1]['residenza']; // stampa 'Roma'
print $persone['totale_elementi']; // stampa '2'
```

In questo caso l'array è formato a sua volta da due array, ai quali vengono assegnati gli indici 1 e 2 e da un terzo elemento, che non è un array ma una variabile intera, con chiave associativa 'totale_elementi'.

I due array che costituiscono i primi due elementi hanno una struttura diversa in quanto mentre il primo è formato dagli elementi 'nome', 'residenza' e 'ruolo', il secondo è formato da 'nome', 'data_nascita' e 'residenza'.

Php permette la gestione degli array tramite delle funzione che vengono elencate di seguito[11]:

- count(array): conta il numero di elementi dell'array. Restituisce un intero;
- array_reverse(array [, booleano]): inverte l'ordine degli elementi dell'array. Se si vuole mantenere le chiavi dell'array di input, il secondo parametro dovrà avere valore TRUE. Restituisce l'array di input con gli elementi invertiti;
- sort(array):ordina gli elementi dell'array. Bisogna fare attenzione, perché

questa funzione, contrariamente a molte altre, modifica direttamente l'array che le viene passato in input e che quindi andrà perso nella sua composizione originale. I valori vengono disposti in ordine crescente;

- rsort(array):ordina gli elementi dell'array in ordine decrescente. Anche questa funzione modifica direttamente l'array passato in input e riassegna le chiavi numeriche a partire da 0;
- asort(array): funziona come sort(), con la differenza che vengono mantenute le chiavi originarie degli elementi;
- arsort(array):come rsort(), ordina in modo decrescente mantenendo però le chiavi originarie;
- in_array(valore, array): cerca il valore all'interno dell'array. Restituisce un valore booleano: vero o falso a seconda che il valore cercato sia presente o meno nell'array;
- array_key_exists(valore, array): cerca il valore fra le chiavi (e non fra i valori) dell'array. Restituisce un valore booleano;
- array_search(valore, array): cerca il valore nell'array e ne indica la chiave. Restituisce la chiave del valore trovato oppure se la ricerca non va a buon fine, il valore FALSE;
- array_merge(array, array [, array...]): fonde gli elementi di due o più array. Gli elementi con chiavi numeriche vengono accodati l'uno all'altro e le chiavi rinumerate. Le chiavi associative invece vengono mantenute, e nel caso vi siano più elementi nei diversi array con le stesse chiavi associative, l'ultimo sovrascrive i precedenti. Restituisce l'array risultante dalla fusione;
- array_pop(array): estrae l'ultimo elemento dell'array, che viene 'accorciato'. Restituisce l'elemento in fondo all'array e, contemporaneamente, modifica l'array in input togliendogli lo stesso elemento;
- array_push(array, valore ,[valore...]):accoda i valori indicati all'array. Equivale all'uso dell'istruzione di accodamento $array[ ]=$valore , con il vantaggio che ci permette di accodare più valori tutti in una volta. Restituisce il numero degli elementi dell'array dopo l'accodamento;

- array_shift(array): estrae un elemento come array_pop(), ma in questo caso si tratta del primo. Anche in questo caso l'array viene 'accorciato' e inoltre gli indici numerici vengono rinumerati facendo rimanere invariati quelli associativi. Restituisce l'elemento estratto dall'array;
- array_unshift(array, valore [,valore...]): inserisce i valori indicati in testa all'array.

Restituisce il numero degli elementi dell'array dopo l'inserimento;

- implode(stringa, array): funzione opposta di explode() che serve a riunire in un'unica stringa i valori dell'array.

4.4.8 Le variabili GET e POST

La principale peculiarità del web dinamico consiste nella possibilità di variare i contenuti delle pagine in base alle richieste degli utenti [37].

Questa possibilità si materializza attraverso i meccanismi che permettono agli utenti, oltre alla richiesta di una pagina ad un web server, di specificare determinati parametri che sono utilizzati dallo script php per determinare quali contenuti la pagina deve mostrare.

In questo caso lo scopo è quello di visualizzare una o più pagine contenenti dati ondametrici prelevati da un database MySQL.

Nel momento in cui si richiama la pagina, si devono specificare i parametri che si vuole visualizzare a schermo, per consentire allo script di estrarre dal database i dati richiesti.

Esistono due modi per passare dati ad uno script: il metodo GET e il metodo POST[37].

- Il metodo GET: consiste nell'accodare i dati all'indirizzo della pagina richiesta, facendo seguire il nome della pagina da un punto interrogativo e dalle coppie nome/valore dei dati che ci interessano. Nome e valore sono separati da un segno di uguale mentre diverse coppie nome/valore sono separate dal segno '&'. In questo caso si avrà, per esempio, la pagina dati_ondametrici.php che mostra i parametri ondametrici sulla base di ciò che ha scelto l'utente. Se si vuole visualizzare i dati relativi a Hmo in una certa data:

  <a href = "dati_ondametrici.php?Hm0 = valore & data = valore">

La precedente riga permette di creare un collegamento fra la pagina web e i comandi contenuti in un file php. Nel caso che stiamo analizzando, all'interno della pagina dati_ondametrici.php vengono richiamate le variabili $_GET ['Hm0 '] (con un determinato valore) e $_GET [' data '] (con un determinato valore). I valori contenuti nella query string vengono memorizzati nell'array $_GET che a sua volta procede a richiamare tutte le procedura contenute nel file .php per richiamare i dati richiesti.

- Il metodo POST: quando una pagina HTML contiene un tag <form>, uno dei suoi attributi è method, che può valere GET o POST. Se il metodo è GET, i dati vengono passati nella query string, come abbiamo visto precedentemente. Se il metodo è POST, i dati vengono invece inviati in maniera da non essere direttamente visibili per l'utente, attraverso la richiesta HTTP che il browser invia al server. I dati che vengono passati attraverso il metodo POST sono memorizzati nell'array $_POST. Anche questo array, come $_GET, è un array superglobale. Quindi, per fare un esempio attraverso un piccolo modulo:

```
<form action="elabora.php" method="post">
<input type="text" name="nome">
<input type="checkbox" name="nuovo" value="si">
<input type="submit" name="submit" value="invia">
</form>
```

Questo modulo contiene una casella di testo che si chiama 'nome' e una checkbox che si chiama 'nuovo', il cui valore è definito come 'sì'. Importante è la presenza del tasto che invia i dati, attraverso il metodo POST, alla pagina elabora.php. In questa pagina si trova a disposizione la variabile $_POST['nome'], contenente il valore che l'utente ha digitato nel campo di testo; inoltre, se è stata selezionata la checkbox, riceverà la variabile $_POST['nuovo'] con valore 'si'. Attenzione però: se la checkbox non viene selezionata dall'utente, la variabile corrispondente risulta non definita.

4.4.9 Le funzioni php utilizzate per la procedura di acquisizione

Php è un linguaggio di programmazione autonomo derivante dal C, ed è nato per soddisfare alcune esigenze che avevano i programmatori nell'elaborare le pagine web

Uno degli aspetti fondamentali legati alla programmazione consiste nel poter accedere a fonti di dati esterne al fine di recuperare o salvare delle informazioni utili da riportare nel sistema di gestione.

Le operazioni principali sui file sono essenzialmente di lettura, scrittura e posizionamento (php fornisce moltissime operazioni ausiliarie per gestire completamente i file, come le operazioni per l'eliminazione e la prova d'esistenza, che saranno analizzate in modo più approfondito in seguito)[38].

Si vedono gli esempi (gli esempi di questo paragrafo sono pure consultabili nella seguente pagina web : http://www.allwebfree.it/php_txt.php):

```
<?php
$fp = fopen( "prova.txt", "w+" );          //Apertura del file prova.txt in lettura,
    fwrite( $fp, ciao a tutti, come va? );  //Scrittura di una stringa sul file
fclose( $fp );                              //Chiusura del file aperto precedentemente
?>
```

Il programma precedente crea un file prova.txt e vi scrive dentro "ciao a tutti,come va?

Oltre ad essere utilizzato per le operazioni da svolgere nel database MySQL, mediante php si possono leggere, modificare, creare e salvare i file di testo che sono stati utilizzati come file per il salvataggio momentaneo dei dati appena generati da una fonte generica.

L'operazione di apertura di un file avviene attraverso la funzione "fopen" che accetta, nella sua forma base, due parametri.

```
<?
$var=fopen("nome_file.txt","tipo");
?>
```

Il primo rappresenta il percorso (path) del file sul quale si opera mentre il

secondo è una stringa che indica alla funzione quali sono le operazioni che si desiderano svolgere sul file.

Di seguito,viene presentata una lista di queste operazioni:

- 'r', apre il file in sola lettura;
- 'r+', apre il file in lettura e scrittura;
- 'w', apre il file in sola scrittura e se il file non esiste viene creato;
- 'w+' apre il file in lettura e scrittura. I dati già scritti andranno persi e se il file non esiste sarà creato;
- 'a', apre in sola scrittura e i dati sono aggiunti in coda a quanto già scritto. Se il file non esiste, viene creato;
- 'a+' apre in lettura e scrittura. I dati sono aggiunti in coda e se il file non esiste, viene creato.

Altra funzione utilizzata nello script relativo all'acquisizione dei dati è "file_exists("testo.txt")".

Tale funzione ha come unico attributo il percorso del file di cui si vuole conoscere l'esistenza.

Restituisce un valore booleano TRUE quando viene individuato il file oppure FALSE quando, nel percorso indicato, il file non esiste.

Per quanto riguarda questo lavoro di tesi, la funzione è stata utilizzata per verificare la presenza del file contenente i dati da trasferire al database.

Di seguito, si può osservare la sua struttura:

```
<?
 $var= file_exists("nome_file.txt");                    // Si verifica l'esistenza nel file nel percorso
                                                              specificato
      if ($var==true) echo"true<br>";                  // Se viene individuato, si ha valore TRUE
      if ($var==false) echo"false<br>";                // Se non esiste, si ha valore FALSE
?>
```

La successiva funzione che viene presa in esame è risultata fondamentale nel prelievo del contenuto del file di testo.

La funzione chiamata in causa consiste in "file ("nome_file.txt")" il cui compito è quello di restituire un array con gli elementi uguali ad ogni riga del file di testo.

Si comprende che questo comando, unito ad altre funzioni array, permette di effettuare una rielaborazione completa del documento preso in analisi.

Si vedrà più avanti come questa "unione" sia stata utilizzata per selezionare prima una riga e successivamente i dati contenuti all'interno della riga stessa.

Una funzione che si collega a ciò che è stato spiegato prima è "explode"[13].

La funzione "explode" ha il compito di suddividere una riga nei dati che la compongono, andando a definire quale elemento separa proprio i singoli dati:

```
explode( "tipo di separatore", "stringa");
```

Un esempio di utilizzo:

```
$testo = "Max Luca Claudio Paolo";
$arr = explode(" ", $testo);
echo $arr[0] . "<br/>"; // Max
echo $arr[1] . "<br/>"; // Luca
echo $arr[2] . "<br/>"; // Claudio
echo $arr[3] . "<br/>"; // Paolo
```

Nell'esempio si vede che come separatore è stato utilizzato un semplice spazio come separatore (definito come attribuito " " di "explode").

Nel caso in cui non venga definito il separatore corrispondente a quello che divide i singoli elementi della riga, si ha come risultato FALSE

Il risultato dell'esempio è un array che contiene quattro elementi corrispondenti ai nomi presenti nella stringa originaria.

4.5 HyperText Markup Language

4.5.1. Introduzione

HyperText Markup Language ("Linguaggio di contrassegno per gli Ipertesti"), vale a dire HTML, non consiste in un vero e proprio linguaggio di programmazione ma in un linguaggio di contrassegno che permette di indicare come disporre gli elementi all'interno di una pagina [43].

Le indicazione vengono date attraverso degli appositi marcatori, detti tag.

Questo significa che l'HTML non ha meccanismi che consentono di prendere delle decisioni, vale a dire non è in grado di compiere iterazioni ne altri costrutti propri della programmazione.

Il mezzo tramite il quale il codice HTML si esprime è il browser.

Il browser è quel programma che viene utilizzato per navigare nel web e svolge principalmente due compiti[50]:

•scarica i vari file che si trovano su un computer remoto, che fanno riferimento a un certo indirizzo;

•legge i documenti scritti in html e, a seconda delle indicazioni ivi contenute, visualizza la pagina in un modo piuttosto che in un altro; inoltre i vari files associati a quel documento (ad esempio le immagini, o i filmati in linguaggio flash) vengono disposti secondo le indicazioni del codice HTML.

La visualizzazione di un file HTML da parte del browser prende il nome di rendering della pagina, per cui il motore di rendering è quella sezione del browser che si occupa di mostrare a schermo la pagina [43].

Il compito del linguaggio HTML è dunque quello di spiegare al browser come i vari files relativi al documento in esame devono essere disposti all'interno della pagina che stiamo visualizzando.

L'organizzazione che si occupa di standardizzare la sintassi del linguaggio HTML (il W3C:World Wide Web Consortium) ha rilasciato diverse versioni di questo linguaggio (HTML 2.0, HTML 3.2, HTML 4.0) e a partire dall'ultima versione

rilasciata, HTML si è evoluto in XHTML (si tratta dell'HTML riformulato come linguaggio XML).

Anche se è stato detto che l'HTML si è evoluto in XHTML ci sono ottime ragioni per lavorare con HTML invece che con XHTML[15]:

- HTML nonostante l'ultima versione rilasciata risalga a HTML 4.01 del 24/12/1999 viene tutt'oggi utilizzato in larga parte dalla maggioranza dei web master;
- alcuni concetti dell'XHTML richiedono già una certa comprensione dei problemi che si acquisisce solo con l'esperienza. L'HTML è più immediato e consente di incominciare subito a produrre documenti web.

All'interno di ogni pagina HTML è presente una serie di marcatori (i tag) a cui viene affidata la visualizzazione del contenuto che si vuol rappresentare [52].

Hanno nomi differenti a seconda della loro funzione e il loro inserimento all'interno del file .html vanno inseriti fra parentesi angolari (<tag>) mentre la loro chiusura viene indicata con "/" (slash).

Si scrive la struttura appena definita:

<tag attributi> contenuto </tag>

Un esempio, con una sintassi che serve a disporre un testo giustificato alla destra della pagina:

** testo **

La sua composizione si può definire così:

<tag attributo_1 = "valore_1". Attributo_2 = "valore2"> contenuto </tag>

Alcuni tag rappresentano un'eccezione in quanto non avendo un contenuto scritto perché indicano la posizione di alcuni elementi (per esempio il tag relativo alle immagini si costruisce in questo modo), non possiedono la chiusura.

Ecco un esempio per l'inserimento di un'immagine:

<img widht ="20" height = "20" src = "immagine.gif" alt = "alt">

Esistono tre tipologie di tag [9] (tab. 4.2):

Elementi di blocco	**Elementi che costituiscono un blocco attorno a sé, come i paragrafi <p>, le tabelle <tab>, i form e soprattutto i **
Elementi inline	**Sono gli elementi che possono essere integrati nel testo come i collegamenti ipertestuali o le immagini**
Liste	**Comprende le liste, numerate e non**

Tabella 4.2 : tipologie dei tag che si trovano in un foglio HTML

Una caratteristica importante del codice HTML consiste nel fatto che i tag possono essere annidati l'uno dentro l'altro.

Ad esempio:

```
<tag1 attributi>
    contenuto 1
        <tag2> contenuto 2 </tag>
</tag1>
```

L'annidamento permette di attribuire formattazioni successive al testo che è stato inserito.

Come si può vedere già nell'esempio è una buona norma utilizzare dei caratteri di tabulazione per far rientrare il testo ogni volta che ci si trova in presenza di un annidamento e man mano che il documento si fa più complesso.

In pratica apertura e chiusura del tag si trovano allo stesso livello, mentre il contenuto viene spostato verso destra di un tab.

Questo aspetto non si tratta soltanto di un fattore visivo ma l'allineamento di

apertura e chiusura tag viene mantenuto anche se il documento viene scorso in verticale con il cursore

La procedura appena definita si chiama indentazione e grazie ad essa il codice HTML risulta più leggibile.

Si confronti ad esempio:

```
<div align= "right"> testo 1 <p align = "left"> testo 2 </p> </div>
```

con

```
<div align= "right">
    testo 1
        <p align = "left">
            testo 2
        </p>
</div>
```

per il browser i due esempio sono equivalenti ma per l'utente è evidente che la differenza sta nel miglioramento della leggibilità.

In fase di realizzazione del codice, una strategia importante è quella di inserire dei “commenti” nei punti più significativi.

Un commento da indicazioni significative per il webmaster ma invisibili al browser e alla lettura dell'utente.

La sintassi è la seguente:

```
<!-- questo è un commento -->
```

L'HTML non è "case sensitive" cioè risulta del tutto indipendente alla forma maiuscola o minuscola con cui possono essere scritti i tag (l'interpretazione del loro significato non cambia).

4.5.2 Struttura della pagina: le componenti fondamentali

La struttura della pagina rappresenta l'organizzazione degli elementi che si vogliono far visualizzare tramite la pagina web.

La struttura della pagina segue uno standard rigido nella disposizione di alcuni tag e righe che obbligatoriamente devono essere presenti per far capire al browser che ciò che gli viene inviato è un documento HTML contenente delle informazione che devono essere visualizzate sulla pagina web con determinati elementi e secondo un determinato stile.

Per primo si deve inserire la riga che indica l'utilizzo delle specifiche del World Wide Web Consortium che riguardano il codice HTML [18]:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

In particolare, tale specifica indica l'utilizzo di HTML 4.01 che risulta un'applicazione conforme alla norma

HTML	Il tipo di linguaggio utilizzato è l'HTML
PUBBLIC	Il documento è pubblico
W3C	Il documento fa riferimento alle specifiche rilasciate dal W3C
-	Indica che le specifiche non sono registrate all'ISO (organizzazione di standardizzazione internazionale). Se lo fossero state, ci sarebbe un segno +.
DTD HTML 4.01 Transizionale	Il documento fa riferimento a una DTD ("Document Type Definition" cioè "Definizione del tipo di documento"); la versione di HTML è la 4.10 "transitional".
EN	La lingua con cui è scritta la DTD è l'inglese.

Tabella 4.3: informazioni fornite dalla prima riga che viene inserita nel documento HTML

Per quel che riguarda l'HTML le indicazioni possibili sono tre:

•Strict: è una DTD particolarmente rigorosa in quanto esclude ogni elemento che riguarda il layout (la cui formattazione è affidata all'utilizzo dei CSS) e non è consentito l'uso degli elementi deprecati:

<!DOCTYPE HTML PUBBLIC "-//W3C//DTD HTML 4.01 Strict//EN http://www.w3.org/TR/html4/strict.dtd>

•Transitional: è una versione temporanea, per consentire il passaggio da una specifica all'altra. Nella DTD transitional i tag deprecati sono ammessi.

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

•Frameset consiste nella DTD che riguarda i frames:

<!DOCTYPE HTML PUBBLIC "-//W3C//DTD HTML 4.01 Frameset//EN http://www.w3.org/TR/html4/frameset.dtd>

La composizione del documento prosegue con l'inserimento dei tag obbligatori che vanno a delimitare le sezioni della pagina (tab 4.4).

Testa (<head>	Contiene informazioni che riguardano come il modo in cui il documento deve essere letto e interpretato. Questo viene definito come il luogo dove porre i riferimenti ad altri documenti, scritti con linguaggi di programmazione php, javascript, fogli di stile css ecc...
Corpo (<body>)	All'interno è racchiuso il contenuto vero e proprio del documento

Tabella 4.4: i tag che devono essere sempre presenti in un documento HTML

Per quanto riguarda il tag <head>, il contenuto di questa porzione di pagina non è visibile agli utenti [44]

4.5.3 La struttura della pagina: le diverse tipologie di tag.

Vengono di seguito elencate le diverse tipologie di tag che sono stati utilizzati nella realizzazione di quei programmi che permettono la gestione e restituzione all'utente, dei dati ondametrici.
Il primo tag preso in considerazione è il tag <title>:

<title>Titolo della pagina</title>.

Definisce il titolo della pagina web costruita.

Come già anticipato questo tag è molto importate soprattutto dal punto di vista dell'ottimizzazione dei contenuti per i motori di ricerca.

Altro tag utilizzato consiste nel tag <link>:

<link rel="stylesheet" type="text/css" href="foglio_di_stile.css">

Questo tag definisce una relazione tra il documento HTML ed una risorsa esterna; normalmente è utilizzato per includere nella pagina un foglio di stile CSS.

Il tag <script> è stato utilizzato per l'inserimento di altri programmi all'interno di un unico programma che ne usa le funzionalità:

<script type="text/javascript" src="script.js"></script>
<script type="text/javascript">
</script>

Con il tag script è possibile eseguire all'interno della pagina degli script lato client (normalmente si tratta di codici di programmi scritti in Javascript).

I tag <meta> (o meta-tag)[46]sono dei tag HTML volti a fornire informazioni aggiuntive (meta-informazioni) su un documento ipertestuale.

A questa funzione gestionale-informativa, si affianca l'importante ruolo che questi tag assumono per l'indicizzazione del sito e all'interno dei motori di ricerca.

L’elemento Meta si colloca nel documento HTML tra i tag <head>...</head> e non ha un tag di chiusura.

Si costituisce di una coppia nome/valore e ha tre attributi principali:

•http-equiv dove le informazioni in esso contenute informano il browser che si tratta di valori di intestazione della connessione http (HTTP Header);

•name che raccoglie informazioni ininfluenti per i browser, ma utili agli utenti per conoscere, per esempio, l’autore del documento;

•content determina il valore da attribuire alla proprietà che lo precede.

Nello specifico, si è utilizzato l'attributo http-equiv che viene descritto di seguito

Tale attributo contiene informazioni utilizzate nella comunicazione tra server (in questo caso localhost) e browser e possiede le seguenti variabili:

•Reply-to va ad indicare l’indirizzo di posta elettronica dell’autore del documento. La sua visualizzazione avviene solo dal lato localhost , ma solitamente non è utilizzato.

<META HTTP-EQUIV=reply-to CONTENT=”webmaster@posta.it”>

•Expires indica al browser la data, in formato GMT, di scadenza del documento. Un documento scaduto non dovrebbe essere memorizzato nella cache del browser e una nuova copia dovrebbe essere richiesta. In realtà molti programmi di gestione della cache ignorano questa variabile.

<META HTTP-EQUIV=expires CONTENT=”Sun, 31 october 1999 12.33.17 GMT “>

•Refresh permette di sfruttare il cosiddetto “client-pull”, cioè il ricaricamento automatico del documento dopo un certo numero di secondi. Per fare questo è stato sufficiente impostare il seguente tag:

<META HTTP-EQUIV=refresh content=”1800″>

In questo esempio, che corrisponde allo stesso meta-tag utilizzato nella programmazione del file di acquisizione, il documento viene automaticamente

ricaricato ogni 1800 secondi (30 minuti).[39]

Per caricare una pagina differente da quella che si sta visualizzando è necessario impostare un URL completo all'interno di CONTENT:

<META HTTP-EQUIV=refresh CONTENT="5;URL=http://www.esempio.it**">**

L'URL deve essere completo e non un semplice nome di percorso.

Questa tecnica è risultata utile per visualizzare ricaricare lo script dove si attiva la procedura di caricamento del dato, dalla fonte al database, con soluzione di continuità evitando qualsiasi intervento manuale.

Altro tag necessario alla costruzione della struttura della pagina è <body>.

Il tag HTML <body> definisce il corpo del documento HTML.

Al suo interno devono essere inseriti tutti i contenuti della pagina web: testi, immagini, link, ecc.

Per quanto riguarda la sintassi, il tag <body> viene aperto subito dopo la chiusura del tag <head> come nel seguente esempio:

```
<html>
    <head>
    <title>Titolo della pagina</title>
    </head>
        <body>
        <div id="container">
        Contenuto della pagina web
        </div>
        <div id="sidebar">
        Contenuto menù laterale
        </div>
        </body>
</html>
```

All'interno del tag <body> possono essere definiti i seguenti tag standard (tab 4.5): (DTD indica in quali doctype il tag è consentito: S=Strict, T=Transitional e F=Frameset).

Attributo standard	Valore	Descrizione	DTD
class	Nome, classe	Specifica il nome di una classe per un elemento.	STF
dir	ltr rtl	Specifica la direzione del testo all'interno di un elemento.	STF
id	id	Specifica un id univoco per un elemento.	STF
lang	codice lingua	Può essere utilizzato per indicare il codice della lingua con cui è scritta l'abbreviazione.	STF
style	definizione stile	Specifica uno stile in linea per un elemento.	STF
title	testo	Può essere utilizzato per fornire la forma completa o espansa dell'abbreviazione.	STF
xml:lang	linguaggio codice	Può essere utilizzato per indicare il codice della lingua con cui è scritta l'abbreviazione in un documento HTML	STF

Tabella 4.5: Attributi standard

Nel corpo del file HTML c'è la possibilità di inserire numerosi tag che possono andare a modificare l'organizzazione della pagine e la stessa visualizzazione attraverso il web.

Realizzare un elenco di tutti questi tag non sarebbe consono allo scopo di questo lavoro di tesi, in quanto non rientra nello scopo prefissato.

4.5.4 La struttura della pagina: il tag

Per la comprensione dei mezzi utilizzati per il raggiungimento dell'obiettivo definito nel par 1.2 è necessario andare a caratterizzare i tag più comuni che sono stati utilizzati nella costruzione dei file relativi alla restituzione dei dati lato utente.

Il tag [52]è uno dei tag più importanti del linguaggio HTML in quanto può essere definito come un elemento "neutro" grazie al quale è possibile assolvere diversi compiti tra cui la definizione di diverse porzioni all'intero di una pagina web.

Il termine "div" nasce come abbreviazione di "divide" (separatore), infatti la funzione di questo tag consiste proprio nel creare delle sezioni all'interno di una pagina web al fine di separare le diverse aree della pagina.

Di seguito si osserva l'uso del tag all'interno del documento HTML.

```
<!DOCTYPE html>
<html>
    <body>
        <div id="container">
        <div id="header">...</div>
        <div id="menu">...</div>
        <div id="content">...</div>
        <div id="footer">...</div>
        </div>
    </body>
</html>
```

Come detto il tag è "neutro" nel senso che, se non viene opportunamente stilizzato, non offre alcun risultato visibile all'interno della pagina web se non un semplice ritorno a capo

Quindi:

```
<div>Questa è una linea...</div>
<div>Questa è un'altra linea...</div>
```

Ne consegue che il tag è "invisibile" perché appartiene unicamente alla

struttura della pagina.

Scopo dei , quindi, è quello di fungere da semplici contenitori il cui aspetto, dimensione, posizione e funzione dovono essere definiti, di volta in volta, dal designer della pagina web.

Normalmente a ciascun div è assegnata una classe o un id oppure, più raramente, una stilizzazione inline.

4.5.5 La struttura della pagina: il tag <table>

Altro elemento fortemente utilizzato nella realizzazione del sito web consiste nel tag <table>[52].

Tale tag permette di costruire le tabelle,cioè quegli elementi nati per incolonnare e tabulare dati e anche per impaginare tutti i vari elementi che compongono una pagina web.

Oggi l'impaginazione è lasciata ai box formati da elementi che hanno di fatto restituito alle tabelle il loro compito originale.

In questo caso, oltre ad esserci gli attributi, all'interno del tag <table> compaiono ulteriori elementi, i quali completano la costruzione della tabella attribuendogli maggiori dettagli.

L'elemento <table> definisce una tabella e necessita di almeno altri due elementi <tr> e <td> che servono per definire righe e colonne.

Questo tag necessita del relativo tag di chiusura </table>

Dal punto di vista degli attributi, si elencano di seguito quelli più comunemente usati in quanto compatibili con tutti i browser:

- **align:** l'attributo align consente di specificare l'allineamento della tabella rispetto al documento con determinate possibilità (center, left,right) che indicano rispettivamente centrato, a sinistra, a destra. Risulta possibile applicarlo oltre che all'elemento <table> anche ai singoli elementi che costituiscono la struttura della tabella stessa, quali: <td> e <th>, descritti più avanti;

- **background**: l'attributo background permette di avere un'immagine come sfondo della tabella, è possibile applicarlo oltre che all'elemento <table> anche agli elementi che costituiscono la struttura della tabella stessa quali <td> e <th>.

Tale attributo può essere utilizzato per caricare delle immagini in formato .jpg o .gif
Nell'utilizzo bisogna prendere una precauzione, in quanto l'attributo background è proprietario di alcuni browser non riconosciuto dal W3C ma approvato se inserito in un CSS.

Inoltre, proprio dal W3C, viene definito come deprecato, quindi è risultato migliore il suo utilizzo attraverso una dichiarazione di stile in un foglio CSS.

```
table{
    background-image:url(immagine_di_sfondo.jpg);
    }
```

Gli esempi presi in considerazione si possono trovare nella seguente pagina web: http://www.web-link.it/html/4_forma.htm e corrispondono a ciò che si può trovare all'interno dei file relativi alla costruzione della componente di restituzione dei dati lato utente.

L'attributo **border** permette di avere un bordo perimetrale che contorna tutte le celle facenti parte della tabella ed è possibile impostare anche lo spessore del bordo(Fig. 4.15). Alcuni browser in mancanza di questo attributo lasciano un piccolo spazio pur non facendo vedere alcun bordo e per recuperare quello spazio è necessario dichiarare border impostandolo a zero.

```
<table border="1">
 <table border="5">
 <table border="10">
```

Figura 4.14: bordi associati alla cella

L'attributo **cellpadding** specifica la quantità di spazio vuoto da lasciare tra i bordi delle celle della tabella e il dato vero e proprio in esse contenuto (Fig 4.15).

Il valore di default è 2 , quindi per tabelle più compresse si deve impostare il cellpadding uguale a zero.

```
<table cellpadding="0">
<table cellpadding="10">
```

In questo caso, così come è stato già osservato, risulta preferibile usare gli attributi all'interno dei fogli di stile CSS dove cellpadding è sostituito dalla proprietà padding.

```
Elemento {
    padding: 10px;
        }
```

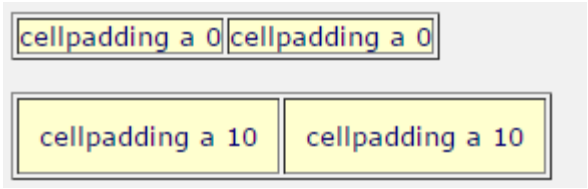

Figura 4.15: cellpadding

L'attributo **cellspacing** specifica la quantità di spazio vuoto da lasciare tra le singole celle di dati della tabella (Fig. 4.16) Il valore di default è 2, per tabelle più compresse si deve impostare cellspacing uguale a zero.

```
<table cellspacing="0">
<table cellspacing="10">
```

Nel caso dell'utilizzo di questo attributo nel foglio di stile CSS, così come è stato fatto nella realizzazione dello "stile" per le pagine web costruite in questo lavoro di tesi, cellspacing è sostituito dalla proprietà border-spacing.

```
Elemento    {
    border-spacing: 10px;
        }
```

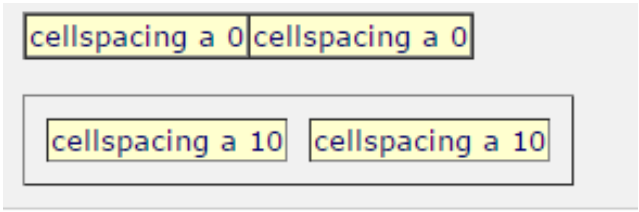

Figura 4.16: immagine del cellspacing

Il browser IE non supporta la proprietà border-spacing.

La coppia di elementi **<tr>...</tr>** inseriti all'interno di <table> definiscono l'inizio di una riga della tabella e questo significa che il numero di righe di una tabella è pari al numero di elementi <tr> in essa contenuti.

Questo tag necessita del relativo tag di chiusura </tr>.

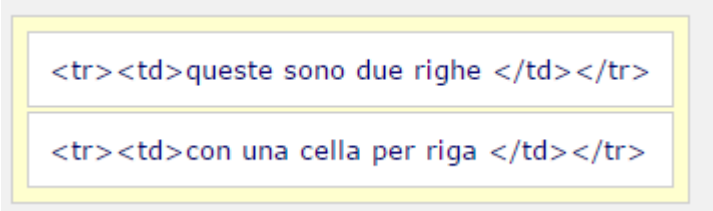

Fig. 4.17: definizione delle righe delle tabelle

<tr><td>questa è una riga con una cella </td></tr>

La coppia di elementi **<th>...</th>** inseriti all'interno di <table> e di <tr> definisce la cella d'intestazione della tabella (Fig. 4.18).

Necessita di essere racchiuso fra i tags <tr> e </tr> esattamente come <td>.

Questo tag necessita del relativo tag di chiusura </th>.

```
<table>
        <tr><th> questa la cella th </th></tr>
        <tr><td> questa la cella td </td></tr>
</table>
```

Figura 4.18: definizione delle celle di intestazione

Gli elementi **<td>...</td>** inseriti all'interno di <table> e di <tr> definiscono la cella vera e propria della tabella in quanto <td> supporta tutti gli attributi di <table>, per cui è possibile, oltre agli allineamenti, avere anche colori e/o immagini di sfondo diverse e indipendenti tra le singole celle (Fig.4.19).

Questo tag è di norma preceduto dal tag **<tr>** che definisce l'inizio della riga e del relativo tag di chiusura</td>.

<tr><td>questa è una cella </td></tr>

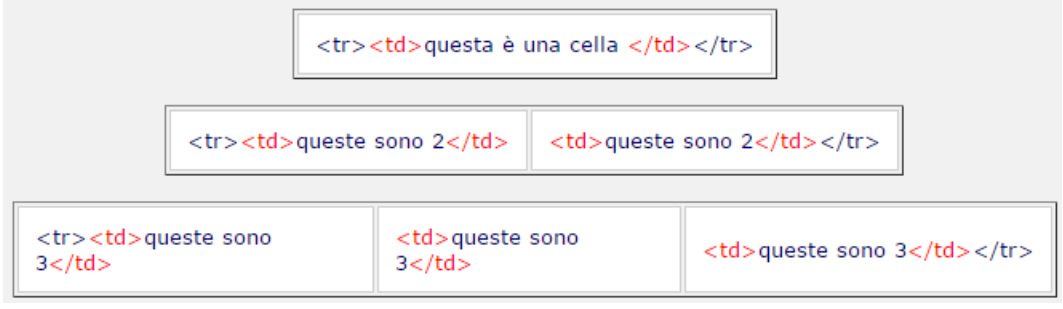

Fig. 4.19:definizione delle celle vere e proprie

Questa coppia di elementi **<thead>...</thead>** inseriti all'interno di <table> servono per definire l'intestazione della tabella (Fig. 4.20)

Viene adoperato e dichiarato prima di **<tbody>**

Tale tag è risultato utile quando si costruiscono tabelle lunghe dove, anche in fase di stampa su ogni pagina, sarebbe ripetuta l'intestazione.

Per evidenziare l'intestazione è possibile fare uso di **<th>** ed è necessario del relativo tag di chiusura **</thead>**.

Nella prossima pagina viene preso in considerazione un esempio

```
<table>
    <thead><tr><th>titolo head sinistra</th>
    <th>titolo head destra</th></tr></thead>
    <tbody>
            <tr><td> cella alto sinistra </td><td> cella alto destra
        </td></tr>
            <tr><td> cella basso sinistra </td><td> cella basso destra
    </td></tr>
    </tbody>
</table>
```

La figura 4.20 mostra il risultato:

titolo head sinistra	titolo head destra
cella body alto sinistra	cella body alto destra
cella body basso sinistra	cella body basso destra

Figura 4.20: Intestazione della tabella

L'elemento tbody definito dalla coppia di elementi <tbody>...</tbody> inseriti all'interno di <table>, servono per specificare la sezione del corpo della tabella influenzando direttamente la visualizzazione della tabella stessa.

Si è adoperato in presenza di <thead>.

Questo tag necessita del relativo tag di chiusura </tbody>

```
<table>
    <thead><tr><th>titolo head sinistra</th><th>titolo head destra</th></tr></thead>
    <tbody>
        <tr><td> cella alto sinistra </td><td> cella alto destra </td></tr>
        <tr><td> cella basso sinistra </td><td> cella basso destra </td></tr>
    </tbody>
</table>
```

4.5.6 La struttura della pagina: gli elenchi puntati e numerati

All'interno del corpo del documento HTML si è reso necessario inserire più elenchi di termini, utilizzando le seguenti tipologie di elenchi[46]:

- **elenchi ordinati;**
- **elenchi non ordinati;**
- **elenchi di definizioni.**

Gli **elenchi ordinati** sono contraddistinti dall'enumerazione degli elementi che compongono la lista.

Sono composti da una serie progressiva ordinata che viene individuata da lettere o numeri.

Il tag utilizzato per aprire un elenco ordinato è **<ol>** ("ordered list") e gli elementi sono individuati dal tag ("list item") (Fig 4.21)

Codice	Resa

`Testo che precede la lista`
`<ol>`
`<li>primo elemento</li>`
`<li>secondo elemento</li>`
`<li>terzo elemento</li>`
`</ol>`
`Testo che segue la lista`
	Testo che precede la lista
	1. primo elemento
	2. secondo elemento
	3. terzo elemento
	Testo che segue la lista

Fig 4.21: descrizione dell'elenco numerato

Lo stile di enumerazione di default del browser è quello numerico, ma è possibile indicare uno stile differente specificandolo per mezzo dell'attributo type.

Ad esempio:

```
<ol type="a">
    <li>primo elemento
    <li>secondo elemento
```

<li>terzo elemento

**
**

Nella tabella che segue (Tab 4.6) si possono vedere le diverse tipologie di elenchi realizzabili con questo metodo.

Descrizione	Codice	Resa
Numeri arabi	**<ol type=”1″> - primo - secondo - terzo **	**1. primo 2.secondo 3.terzo**
Alfabeto minuscolo	**<ol type=”a”> - primo - secondo - terzo **	**a.primo b.secondo c.terzo**
Alfabeto maiuscolo	**<ol type=”A”> - primo - secondo - terzo **	**A. primo B. secondo C. terzo**
Numeri romani minuscoli	**<ol type=”i”> - primo - secondo - terzo **	**i. primo ii.secondo iii.terzo**
Numeri romani maiuscoli	**<ol type=”I”> - primo - secondo - terzo **	**I. primo II.secondo III. terzo**

Tabella 4.6:le diverse tipologie di elenchi numerati realizzabili

Gli **elenchi non ordinati** sono individuati dal tag **<ul>** ("unordered list"), e gli elementi sono contraddistinti anch'essi dal tag <li> (in buona sostanza si tratta di quello che i programmi di videoscrittura chiamano elenchi puntati).

```
<ul>
    <li>primo elemento
    <li>secondo elemento
    <li>terzo elemento
</ul>
```

il tipo di segno grafico utilizzato per individuare gli elementi dell'elenco di default dipende dal browser, ma di solito è un "pallino pieno".

È possibile comunque scegliere un altro tipo di segno(Fig 4.22)

Valore dell'attributo type	Descrizione	Codice	Resa

`type="disc"` (è così di default)	visualizza un **"pallino" pieno**. È la visualizzazione di default	`<ul type="disc">`
		`<li>primo</li>`
		`<li>secondo</li>`
		`<li>terzo</li>`
		`</ul>`
			• primo
			• secondo
			• terzo

`type="circle"`	visualizza un **cerchio vuoto** al proprio interno	`<ul type="circle">`
		`<li>primo</li>`
		`<li>secondo</li>`
		`<li>terzo</li>`
		`</ul>`
			○ primo
			○ secondo
			○ terzo

`type="square"`	Visualizza un **quadrato pieno** al proprio interno	`<ul type="square">`
		`<li>primo`
		`<li>secondo`
		`<li>terzo`
		`</ul>`
			▪ primo
			▪ secondo
			▪ terzo

IFigura 4.22: Tipologie di rappresentazione delle liste non ordinate

Da notare inoltre che il tipo di segno grafico, varia in automatico al variare dell'annidamento della lista, come si osserva nella figura seguente (Fig 4.23)

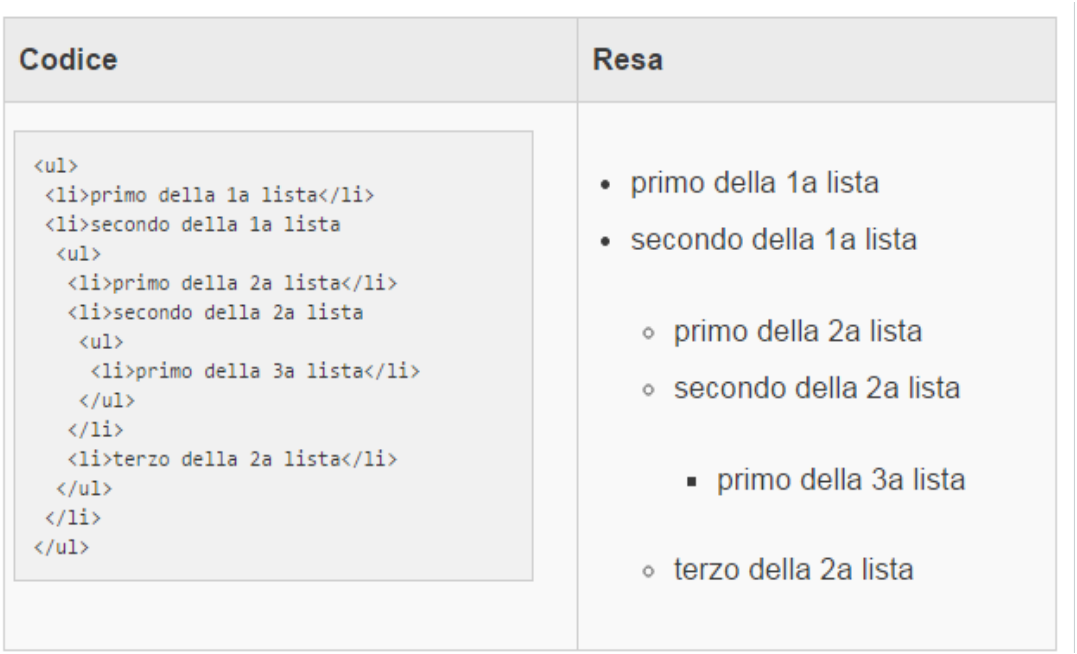

Figura 4.23: conseguenze dell'annidamento sulla struttura dell'elenco

Le **liste di definizione** sono individuate dal tag **<dl>.**

Gli elementi dell'elenco (a differenza delle liste ordinate, e delle liste non ordinate) questa volta sono formati da due parti:

- **<dt>** :definition term, indica il termine da definire e a differenza dell'elemento in questo caso non c'è rientro.
- **<dd>**:definition description è la definizione vera e propria del termine e in genere questo elemento è reso con un rientro

Nella pagina che segue, si osservi l'immagine di esempio (Fig. 4.24)

Codice	Resa
Ecco i principali tag per delimitare il testo: <dl> <dt> </dt> <dd>individua l'apertura di un nuovo paragrafo</dd> <dt> </dt> <dd>individua l'apertura di un nuovo blocco di testo</dd> <dt></dt> <dd>individua l'apertura di un elemento inline, cui attribuire una formattazione atraverso gli stili</dd> </dl> ci sono poi altri tag che…	Ecco i principali tag per delimitare il testo: individua l'apertura di un nuovo paragrafo individua l'apertura di un nuovo blocco di testo individua l'apertura di un elemento inline, cui attribuire una formattazione atraverso gli stili ci sono poi altri tag che…

Figura 4.24: esempi relativi alla lista di definizione

Si deve far notare che nel complesso la scelta del tipo di elenco attraverso l'attributo type è deprecato dal W3C, perché si tratta di una caratteristica che riguarda la formattazione, e dunque andrebbe effettuata utilizzando i CSS.

Sono state seguite le linee guida dettate dalla bibliografia utilizzata per realizzare i documenti HTML, in particolare il manuale HTML redatto da L. Lemay (McGraw-Hill Italy, 1995) che ha rappresentato una guida utile e efficace per la comprensione di HTML.

Con i fogli di stile c'è anche la possibilità di scegliere un'immagine (ad esempio una GIF) come segno distintivo per l'elenco puntato.

4.5.6 La struttura della pagina: i link

All'interno dei documenti HTML realizzati, sono presenti dei link che permettono il passaggio da una pagina all'altra dell'intero sito web[51].

I link costituiscono quel "ponte" che consente di passare da un testo all'altro e la composizione si divide in due parti:

- il contenuto: consiste nella parte visibile del link e proprio per questo l'utente deve essere sempre in grado di capire quali sono i collegamenti da cliccare all'interno della pagina;
- la risorsa: si tratta di un'altra pagina, oppure è un collegamento interno a un punto della pagina stessa.

Esistono link che puntano ad altri documenti, i quali utilizzano un riferimento a un sito web, come si osserva nell'esempio seguente:

Le risorse sono su Miosito.it

che da come risultato:"Le risorse sono su Miosito.it.

Da notare che è indifferente la destinazione di questo "ponte", infatti il meccanismo dei link funziona allo stesso modo indipendentemente dal tipo di risorsa mentre è il browser che si comporta in modo differente a seconda proprio della risorsa.

Nella figura successiva (Fig 4.27) si osservano i diversi comportamenti:

Immagine .gif, .jpg, .png	Viene visualizzata nel browser
Documento .html, .pdf, .doc	La pagina è visualizzata nel browser. Nel caso dei documenti **.doc** e **.pdf** l'utente deve avere installato sul proprio pc l'apposito plugin (nella maggior parte dei casi è sufficiente che abbia installato rispettivamente Microsoft Word e Adobe Acrobat Reader). Se non è installato il plugin il sistema chiederà all'utente se salvare il file.
File .zip, file .exe	Viene chiesto all'utente di scaricare il file

Figura 4.27: varibilità del comportamento degli oggetti linkati

4.5.6 La struttura della pagina: il tag form

Il tag **form** è il tag di fondamentale importanza per quanto riguarda la componente lato utente, in quanto rappresenta senza dubbio il mezzo con il quale le persone possono interagire con le pagine web che sono state costruite[51].

Per organizzare questo genere di interazione è necessario che l'utente sia in grado di effettuare delle scelte sulla base di ciò che gli viene proposto di ricercare.

Per rendere questo possibile, sono stati utilizzati i moduli (cioè i form)

L'invio della richiesta è stato organizzato in due parti.

- **una pagina principale** che contiene i vari campi dei form, consentendo all'utente di effettuare delle scelte;
- **una pagina secondaria** che viene richiamata dalla principale e che effettua il processamento della richiesta. Questa pagina corrisponde ai file php con all'interno i comandi realizzati per richiamare i dati direttamente dal database.

Per quanto riguarda la costruzione delle pagine in HTML, in questo paragrafo ci si occupa solamente della pagina principale, in quanto quella secondaria è già stata definita nell'ambito della descrizione del linguaggio di programmazione php.

Per creare una pagina con dei moduli è stato utilizzato l'apposito tag<form>.

```
<form name="datiUtenti"action="paginaRisposta.php">
</form>
```

"name" serve a indicare il nome del form, **"action"**indica l'URL del programma o della pagina di risposta che processa i dati.

La richiesta effettuata dall'utente, prima di passare alla pagina di risposta, viene codificata dal browser in modo da non poter dare adito ad errori di formattazione della domanda (ad esempio gli spazi vengono convertiti in "+").

Normalmente non è necessario specificare come si vuole effettuare la codifica dei dati perché è sottinteso l'invio di semplice testo.

Per quel che riguarda le componenti dei form, il tag più utilizzato è <input>, il quale permette la formazione di un campo di testo oppure di un bottone:

<input type= "text">

<input type="button">

I vari **<input>** sono dotati di attributi che consentono di indicare il tipo di campo come il nome (necessario per interagire con la pagina secondaria in php) e il valore (utilizzabile per definire il nome del bottone o una stringa di testo di default scritta nell'area di testo).

Vediamo di seguito un esempio:

<input type="text" name="tuoTesto "value="qui il tuo testo">

che da come risultato:

Fig. 4.24: area di testo realizzata con il metodo form

Dopo <input> è necessario inviare la richiesta tramite l'apposito bottone di invio.

La sintassi tradizionale per realizzare un bottone di invio è:

<input type="submit" value="invia i dati">

Un altro bottone utile nella realizzazione del form consiste nel tasto reset il quale, una volta premuto, consente di riportare il form al suo stato originario cancellando ogni cosa scritta dall'utente.

```
<form action= azione>
        <inputtype=”text”><br>
        <input type=”reset” value=”cancella”>
</form>
```

All'utente, viene data la possibilità di poter scegliere fra diverse opzioni attraverso l'impostazione di un menu di opzioni, chiamato "select".

Grazie al tag **<select>** è stato possibile costruire il menu di opzione, dove ciascuna voce è stata compresa all'interno del tag **<option>** e il valore viene specificato attraverso l'attributo "value".

```
<form>
<select name="siti">
    <option value: "http://www.sito_uno.it> sito uno</option>
    <option value: "http://www.sito_due.it> sito due </option>
</select>
</form>
```

In questo modo è possibile inviare un richiesta alla pagina secondaria attraverso la selezione del parametro scelto.

5. Attività svolte

5.1 L'acquisizione dei dati, dalla fonte generica al database

La realizzazione della procedura di acquisizione dei dati, rappresenta quella componente necessaria a collegare una fonte generica di dati con il database.

I dati, appena sono generati dal sistema di modellazione/rilevamento, vengono automaticamente salvati all'interno di un documento al cui interno non avviene alcun tipo di elaborazione.

In questo lavoro di tesi non ci sarà modo di approfondire quale siano le tipologie di file maggiormente idonei a contenere i dati appena generati.

Come contenitore di dati, è stato scelto di usare il file.txt.

Il file di testo (avente estensione .txt) è un file per computer che contiene solo caratteri di scrittura semplici, senza informazioni sul loro formato (dimensione, colore, ecc).

Di solito rappresenta un testo leggibile direttamente dagli utenti senza bisogno di installare programmi appositi.

Il termine si usa in contrapposizione a file binario, che è invece un file contenente dati generici non direttamente leggibili dall'utente

Lo scopo per cui sono nati è semplicemente la lettura e la scrittura.

La mancanza di formattazione li rende poveri dal punto di vista estetico ma in compenso non occorrono particolari programmi per leggerli e possono essere trasferiti direttamente da un sistema operativo all'altro.

La scelta dei file di testo deriva fondamentalmente dal loro utilizzo nei software proprio per l'immagazzinamento dei dati.[50]

Generalmente, i dati possono essere utilizzati facilmente da altri programmi e sistemi, con una conoscenza minima della loro struttura: si è meno vincolati alla dimensione dei campi e all'ordine dei byte .

Ad esempio, sebbene ogni foglio elettronico abbia un suo formato di memorizzazione binaria, tutti possono lavorare con file di testo dove la matrice di dati è semplicemente un elenco di valori separati da caratteri standard (per esempio un TAB "	").

Altro vantaggio consiste nel fatto che l'eventuale corruzione dei dati causa,in genere, meno danni e solo localmente.

Perdere un byte in un file di testo, di solito vuol dire perdere un dato mentre perderlo, per esempio, in un file binario può voler dire sfasare e rovinare l'intero file.

All'interno del file txt vengono immagazzinati una certa quantità di dati sulla base della mole del lavoro di rilevazione/modellazione svolto a monte.

La totalità dei dati è organizzata in righe e una riga contiene a sua volta un set di dati che vengono divisi da un carattere separatore.

Il caricamento dei dati all'interno del database, avviene secondo un certo intervallo di tempo che dipende dalla cadenza con cui ogni riga viene aggiunta all'interno del file.

Ogni riga che si trova all'interno del file txt è ordinata secondo la data e l'ora in cui è stata generata, dalla più recente (che occupa la prima posizione) alla più vecchia (che occupa l'ultima posizione).

Inizialmente, quando il sistema di rilevamento/modellazione non è ancora stato attivato, il file di testo risulta completamente vuoto in quanto non è stato generato alcun dato.

Dopo l'attivazione, il primo set di dati generati vengono trasferiti nel file txt ed occupano la prima riga.

A questo punto si attiva la procedura di acquisizione che ha il compito di prelevare la riga e di trasferirla nella tabella prescelta del database.

Quando viene generato il secondo set di dati, questi vanno ad occupare la prima riga provocando la translazione del primo set di dati alla seconda riga.

Il sistema di acquisizione, quindi, è stato realizzato in maniera tale da prelevare sempre ciò che individua nella prima riga, trasferendolo poi alla tabella del database.

Il tutto viene poi eseguito automaticamente, secondo un intervallo temporale pari a quello in cui avviene la rilevazione di una misurazione ondametrica da parte dello strumento (tipo boa).

Questo intervallo di tempo, come già descritto nel capitolo 2, è pari a 30 minuti.

Di seguito, vengono messe in evidenze le componenti più significative del file php e HTML che gestisce l'acquisizione dei dati.

Nel dettaglio, la procedura di acquisizione dei dati da file txt alla tabella scelta (dati_giornalieri) nel database dati ondametrici è stata costruita attraverso un file con estensione .php la cui struttura è stata definita usando il linguaggio HTML.

La pagina risultante si chiama acquisizione.php e si può consultare l'intero file nell'appendice A.

La prima caratteristica di cui si deve tener conto, consiste nel fatto che la procedura di acquisizione funziona solamente se il file php viene richiamato (in locale) da un browser.

Questo significa che l'avvio e il mantenimento di tale procedura si ha tenendo aperta una scheda del browser del PC nel quale si trova il file dove vengono immagazzinati temporaneamente i dati generati.

L'aggiornamento (il refresh) della pagina avvia la procedura di acquisizione vera e propria.

Per questo scopo, è stato utilizzato il meta tag HTML "http-equiv" inizializzato con un refresh, il quale va ad effettuare l'aggiornamento automatico della pagina sulla base del valore assegnato all'attributo content (valore che viene espresso in secondi).

Sotto si vede il comando utilizzato:

```
<head>
    <title> Caricamento dati </title>
    <meta http-equiv="refresh" content="1800">
</head>
```

Si nota che il meta tag è stato inserito fra i tag <head>...</head> (è obbligatorio farlo) e che risulta impostato un aggiornamento ogni 1800 secondi (30 minuti)[39].

Tale impostazione segue l'intervallo di tempo definito già nel capitolo 2 per quanto riguarda l'acquisizione dei dati dalla boa.

Detto questo si passa a definire il file operativo, costruito interamente con il php.

Il primo passo nella costruzione della procedura, è consistito nel definire una variabile che ha il compito di controllare la presenza del file, contenente i dati generati, nel percorso specificato.

```
$prova = file_exists("dati_giornalieri.txt");
```

La non specificazione del percorso indica che sia il file di acquisizione che il file di dati si trovano nella stessa cartella (directory).

Da qui inizia un ciclo if sulla base del risultato della funzione "file_exists".

Se la funzione restituisce un valore TRUE, cioè il file risulta presente, allora il programma passa all'elaborazione del file.

Per prima cosa, il file viene aperto:

```
$var=fopen("dati_giornalieri.txt", "r");
```

La sua apertura viene effettuata in sola lettura e ciò si capisce dall'uso della costante r come secondo attribuito della funzione "fopen".

Adesso, il contenuto del file che è stato inizializzato dalla variabile $var, viene fatto passare dalla funzione "file".

```
$var2=file("dati_giornalieri.txt");
```

Questa funzione estrae le singole righe del file di testo, formando un array che risulta inizializzare la variabile "$var2".

Chiuso il file di testo originario, tramite la funzione fclose, viene realizzato un ciclo "for" che si trova all'interno di "if":

```
$i=0;
for($a=0; $a<=$i;$a++)      {

                                    $ris= $var2[$a];
                                    echo "La riga di dati inviata al database<br>";
                                    echo $ris;
                            }
```

Il ciclo "for" è servito per prelevare dati dall'array, in quanto per la variabile "$a"=0 viene fatto scorrere un puntatore fra gli elementi dello stesso array (vale a dire lungo le righe) fino a quando il valore della stessa è minore o uguale al valore della variabile "$i" (che è 0).

Si comprende che il ciclo si interrompe immediatamente all'elemento 0 che

corrisponde alla prima riga del file di testo.

$a, che vale 0, rappresenta la posizione del puntatore sull'array "$var2".

Ora l'elemento 0 va ad inizializzare la variabile "$ris":

```
$ris= $var2[$a];
```

Si potrebbe dire che il ciclo "for" risulti inutile, in quanto bastava porre subito la variabile "$a" uguale a 0 per estrarre il primo elemento dell'array "$var2", ma è stato deciso di utilizzare un ciclo in modo tale da facilitare il possibile recupero di righe perse per il mancato avvio della procedura di aggiornamento.

Per esempio se si volesse estrarre le prime tre righe, basta modificare il valore attribuito alla variabile "$i", da 0 a 2.

Terminato il ciclo "for", il programma rientra nel ciclo "if" dove è stata utilizzata la funzione "explode".

Tale funzione viene definita con gli attributi TAB, che indica il tipo di separatore che divide i singoli parametri e la variabile "$ris" nella quale è stata salvata la prima riga.

```
$cmp=explode ("	", $ris);
```

"explode" prende i singoli parametri della riga e li divide fra di loro all'interno di un nuovo array.

A questo punto è bastato inizializzare per ogni elemento dell'array (quindi per ogni parametro) una singola variabile:

```
$i=  $cmp[0];
$b=  $cmp[1];
ecc...;
```

Organizzati i singoli parametri si è passati all'apertura del database con i comandi php appropriati (si ricorda che siamo ancora all'interno del ciclo "if")[7], come si nota nella pagina seguente

```
$connessione = mysql_connect("localhost", "root", "********")
    or die("Connessione non riuscita: " . mysql_erro() );
```

```
mysql_select_db ("dati_ondametrici") or die('Nessun database presente');
```

Altra parte importante è stata la strutturazione della query necessaria da porre al database per il trasferimento dei dati:

```
$sql="INSERT INTO dati_giornalieri (data_reg,ora_reg,Hmo,Hmax,Tp,Tm,Dmt,Dmp,Dmw,Tmp) VALUES ('$i','$b','$c','$d','$e','$f','$g','$h','$l','$m')";
```

La query permette l'inserimento nella tabella dati_giornalieri, in particolare all'interno delle colonne specificate fra parentesi i valori dei dati che inizializzano le variabili fra parentesi.

Solo adesso si conclude il ciclo iniziale, in quanto tale procedura si avvia soltanto se il file "sorgente" si trova collocato nella cartella come è stato indicato dal percorso nella funzione "file_exists".

Se non viene individuato, il programma passa direttamente alla parte seguente al comando "else" e viene inviato a schermo il seguente messaggio:

```
else {
echo "il file dati sorgente non è presente nella directory";
    }
```

5.2 Struttura del database

Come detto nel capitolo 3 precedenti, la realizzazione del database rappresenta il cuore dell'intero lavoro svolto ed è fondamentale capire quali e quanti sono stati i passaggi necessari nella procedura della sua costruzione.

Questo paragrafo comprende tutti i comandi SQL necessari alla realizzazione del database e delle tabelle necessarie all'immagazzinamento dei dati ondametrici.

La costruzione, necessita di un passaggio preliminare legato all'uso del software di gestione PhpMyadmin.

Quest'ultimo deve essere aperto tramite l'accesso ad un qualsiasi browser fra Mozilla Firefox, Google Chrome, Internet Exploerer e Opera.

La su apertura avviene digitando nella barra di navigazione la seguente riga: localhost/phpmyadmin.

Figura 5.1: accesso a phpMyAdmin

Al primo accesso è necessario che venga specificato il nome utente (root) e la password nell'apposito form (fig 5.1).

Cliccare su esegui porta alla comparsa della home page del programma di gestione in questione (fig 5.2).

Cliccando sul tasto SQL, appare una schermata con al centro un foglio elettronico completamente bianco e un cursore.

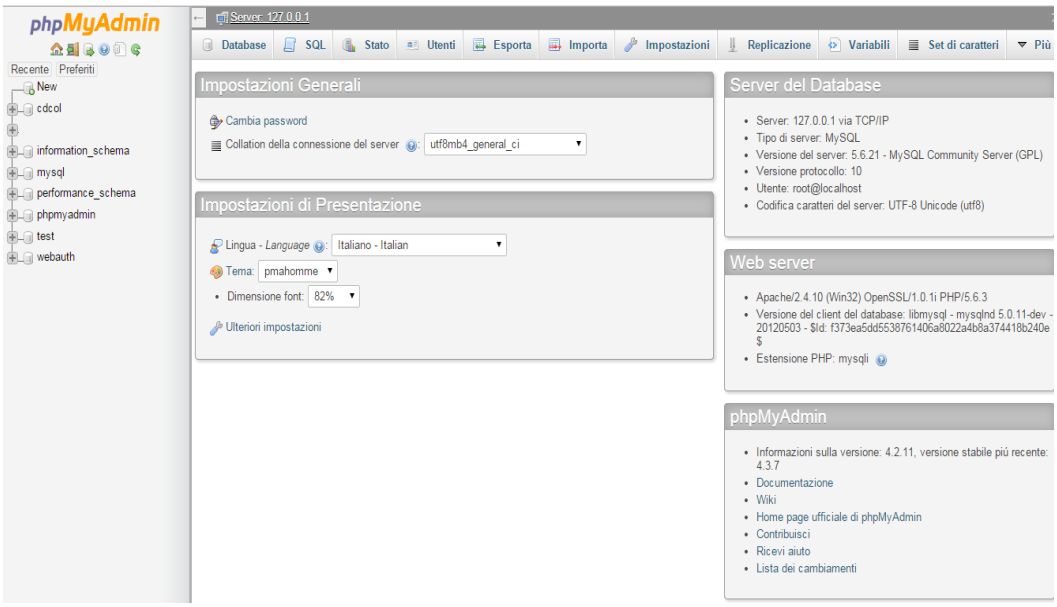

Figura 5.2: home page di phpMyAdmin

Questo spazio serve per poter scrivere i comandi per la costruzione del database.

Vengono mostrate di seguito tutte le istruzioni realizzate che hanno portato alla strutturazione del database e delle relative tabelle[14]:

CREATE database dati_ondametrici;
USE dati_ondametrici;

CREATE table dati_giornalieri (data_reg DATE, ora_reg TIME, Hmo DECIMAL (3,2), Hmax DECIMAL (3,2),Tp DECIMAL (3,2), Tm DECIMAL (3,2), Dmt DECIMAL (5,2), Dmp DECIMAL (5,2), Dmw DECIMAL (5,2), Tmp DECIMAL (4,2));

CREATE table archivio (data_reg DATE, ora_reg TIME, Hmo DECIMAL (3,2), Hmax DECIMAL (3,2), Tp DECIMAL (3,2), Tm DECIMAL (3,2), Dmt DECIMAL (5,2), Dmp DECIMAL (5,2), Dmw DECIMAL (5,2), Tmp DECIMAL (4,2));

Il comando **CREATE** è stato usato per creare il database dati_ondametrici, le tabelle dati_giornalieri e archivio (specificando per ognuna il tipo di dati appartenenti al campo).

Il comando **USE** serve a far comprendere al sistema MySQL che il database di riferimento per effettuare e salvare le modifiche risulti dati_ondametrici.

Al momento dell'esecuzione dei comandi, nella rappresentazione schematica delle componenti di **PhpmyAdmin** (colonna di sinistra) appaiono le icone raffiguranti il database dati_ondametrici e le tabelle archivio e dati_giornalieri

A questo punto si può dire che è stato realizzato il contenitore.

Prima dell'inserimento del contenuto, durante lo svolgimento della fase di realizzazione, è stato deciso di passare alla parte dell'analisi dei dati.

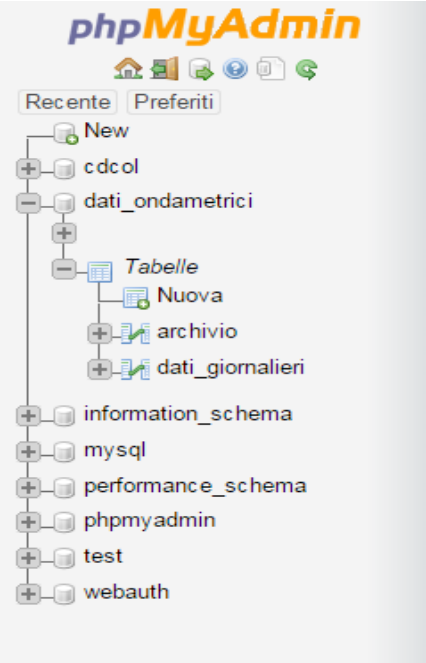

Figura 5.3: rappresentazione schematica delle tabelle

Una scelta del genere si giustifica tenendo conto del fatto che l'inserimento dei dati, senza un adeguato sistema di controllo, avrebbe potuto provocare dei problemi a livello della piattaforma (XAMPP) e quindi relative perdite di tempo nel risolvere proprio questi problemi.

Tale componente acquista, in special modo con questa tesi, un ruolo veramente importante perché la correttezza dei dati può permettere sia la miglior comprensione del fenomeno fisico studiato che una valida elaborazione.

In questo caso, più che di comandi, si può parlare dell'utilizzo di vere e proprie query il cui compito consiste nella ricerca di quei dati che, sulla base del parametro rappresentante, non ricadono all'interno degli intervalli stabiliti.

I valori “fuori scala” sono sostituiti con un valore impostato, in particolare è stato scelto il valore **NULL**

Dall'architettura del sistema, si capisce che la procedura di controllo viene attribuita alla tabella dati_giornalieri.

I dati all'interno di tale tabella sono controllati dalle seguenti query:

UPDATE dati_giornalieri SET Hmax = NULL WHERE Hmax >= 15;

UPDATE dati_giornalieri SET Hmo = NULL WHERE Hmo >= 10;

UPDATE dati_giornalieri SET Tp = NULL WHERE Tp >= 20;

UPDATE dati_giornalieri SET Tm = NULL WHERE Tm >=15;

UPDATE dati_giornalieri SET Tmp = NULL WHERE Tmp >= 40 OR Tmp <= 0;

UPDATE dati_giornalieri SET Dmt = NULL WHERE Dmt <= 0 OR Dmt >= 360;

UPDATE dati_giornalieri SET Dmp = NULL WHERE Dmp <= 0 OR Dmp >= 360;

UPDATE dati_giornalieri SET Dmw = NULL WHERE Dmw <= 0 OR Dmw >= 360;

Il significato delle query appena elencate:

avviene la sostituzione **(UPDATE)** del valore determinato dal comando **SET** (quindi **NULL**) dove **(WHERE)** il valore rientra nell'intervallo definito.

La modalità del controllo dei dati è automatica.

Questo compito è stato assegnato al programma di gestione del database e al comando **CREATE EVENT**[14].

Di seguito sono elencate le serie di query utilizzate per realizzare i comandi di analisi sulla tabella dati_giornalieri

.

CREATE EVENT controllo_0 ON SCHEDULE EVERY 30 MINUTE STARTS'AAAA-MM-GG hh-mm-ss ' ON COMPLETION NOTPRESERVE DO UPDATE dati_giornalieri SET Hmax = NULL WHERE Hmax >= 15;

CREATE EVENT controllo_1 ON SCHEDULE EVERY 30 MINUTE STARTS'AAAA-MM-GG hh-mm-ss ' ON COMPLETION NOTPRESERVE DO UPDATE dati_giornalieri SET Hmo = NULL WHERE Hmo >= 10;

CREATE EVENT controllo_2 ON SCHEDULE EVERY 30 MINUTE STARTS 'AAAA-MM-GG hh-mm-ss ' ON COMPLETION NOTPRESERVE DO UPDATE dati_giornalieri SET Tp = NULL WHERE Tp >= 20;

CREATE EVENT controllo_3 ON SCHEDULE EVERY 30 MINUTE STARTS'AAAA-MM-GG hh-mm-ss ' ON COMPLETION NOTPRESERVE DO UPDATE dati_giornalieri SET Tm = NULL WHERE Tm >=15;

CREATE EVENT controllo_4 ON SCHEDULE EVERY 30 MINUTE STARTS'AAAA-MM-GG hh-mm-ss ' ON COMPLETION NOTPRESERVE DO UPDATE dati_giornalieri SET Tmp = NULLWHERE Tmp >= 40 OR Tmp <= 0;

CREATE EVENT controllo_5 ON SCHEDULE EVERY 30 MINUTE STARTS'AAAA-MM-GG hh-mm-ss ' ON COMPLETION NOTPRESERVE DO UPDATE dati_giornalieri SET Dmt = NULL WHERE Dmt <= 0 OR Dmt >= 360;

CREATE EVENT controllo_6 ON SCHEDULE EVERY 30 MINUTE STARTS'AAAA-MM-GG hh-mm-

ss ' ON COMPLETION NOTPRESERVE DO UPDATE dati_giornalieri SET Dmp = NULL WHERE Dmp <= 0 OR Dmp >= 360;

CREATE EVENT controllo_7 ON SCHEDULE EVERY 30 MINUTE STARTS 'AAAA-MM-GG' ON COMPLETION NOTPRESERVE DO UPDATE dati_giornalieri SET Dmw = NULL WHERE Dmw <= 0 OR Dmw >= 360;

Per ogni singolo comando, avviene la creazione di un evento con **CREATE EVENT**, al quale viene assegnato un nome (controllo più numero in questo caso) (fig 5.4).

Come nell'inserimento dei comandi SQL, si ha la comparsa schematica degli eventi nella colonna di sinistra.

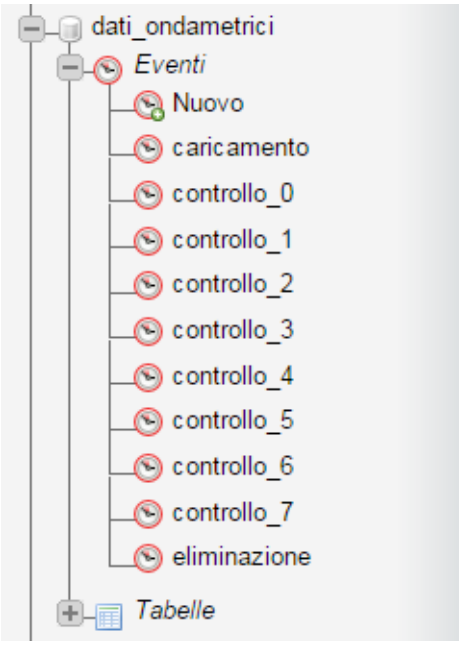

Figura 5.4: rappresentazione schematica degli eventi

Gli eventi sono dei compiti eseguiti in base ad una pianificazione, realizzabile tramite uno speciale tipo di file denominato **EVENT SCHEDULER**.

Tale file contiene tutta la procedura necessaria affinché una serie di eventi vengano eseguiti secondo un certo intervallo temporale e secondo un determinato ordine definito dall'amministratore.

Per far sì che funzioni, l'event scheduler è stato attivato con il seguente comando:

SET GLOBAL event scheduler = ON;

"Settando" event scheduler su ON ogni evento realizzato funziona come definito.

Il valore che precede **MINUTE**, corrispondente a 30, consiste nell'intervallo di tempo trascorso tra il caricamento di una riga di dati (per esempio da una rilevazione della boa) ad un altro caricamento (cioè ad una successiva rilevazione).

Il termine **STARTS** pone l'anno **(AAAA),** il mese **(MM)**, il giorno **(GG)**, l'ora **(hh)**, i minuti **(mm)** e i secondi **(ss)** in cui si ritiene necessario far partire il comando.

Le query sono state realizzate tenendo conto di un tempo di inizio, cioè del momento in cui il comando è operativo.

Quello che viene a mancare in tutte le query è il comando di termine.

Questo è dovuto dal fatto che tali controlli andranno effettuati per tutto il tempo di funzionamento del database.

MySQL, senza il comando che indica la data di terminazione, va a far "scadere" (cioè a non rendere più valido) l'evento dopo il suo primo utilizzo nonostante venga definito un intervallo di tempo nel quale deve ripetersi.

Per ovviare a questo problema, si è utilizzato il comando specifico **ON COMPLETION NOT PRESERVE** che serve a mantenere il problema suddetto.

Avvenuto il controllo dei dati caricati nella tabella dati_giornalieri, a questo punto sono state prese in considerazione le azioni per il trasferimento dei dati nella tabella archivio.

La procedura necessaria è stata realizzata considerando la struttura della tabella dati_giornalieri.

Questo ha implicato la progettazione di un sistema capace di andare a copiare nella tabella archivio, quella riga di dati (appartenente alla tabella dati_giornalieri) la cui acquisizione è risultata risalire al giorno precedente.

Il sistema di caricamento nella tabella archivio avviene con una cadenza temporale pari a quella che si ha nell'acquisizione dei dati nella tabella dati_giornalieri, vale a dire ogni 30 minuti[39].

Allo stesso tempo la riga di dati copiata viene cancellata dalla tabella dati_giornalieri.

Il processo appena descritto avviene utilizzando le seguenti query:

INSERT into dati_ondametrici.archivio (SELECT * FROM dati_giornalieri` ORDER BY ora_reg DESC LIMIT 0, 1);

Si ha il caricamento **(INSERT)** all'interno della tabella archivio appartenente al database dati_ondametrici, avvenuto attraverso la selezione **(SELECT)** dalla **(FROM)** tabella dati giornalieri, di quel dato il cui orario di caricamento viene individuato come l'ultimo acquisito in assoluto.

Nella costruzione della query appare **ORDER BY** che permette l'ordinamento delle righe secondo (in questo caso) gli orari di caricamento dal più remoto al più recente.

Con il comando **DESC** si inverte l'ordine che passa ad essere dal più recente al

più remoto e con **LIMIT** si seleziona il primo dato.

Viene garantita, anche in questo caso, l'automazione della procedura tramite il comando **CREATE EVENT:**

CREATE EVENT caricamento ON SCHEDULE EVERY 30 MINUTE STARTS 'AAAA-MM-GG hh-mm-ss' ON COMPLETION NOT PRESERVE DO INSERT into dati_ondametrici.archivio (SELECT * FROM dati_giornalieri ORDER BY ora_reg DESC LIMIT 0,1);

Nel frattempo, viene cancellata la riga che è stata appena copiata:

DELETE FROM dati_giornalieri WHERE ora_reg <= DATE_SUB(NOW( ), INTERVAL 1 DAY);

L'operazione viene realizzata andando a cancellare **(DELETE)** dalla tabella dati_giornalieri quelle righe (in questo caso una, ma la potenzialità del comando è tale da funzionare anche per tutte quelle righe caricate più di 24 ore fa e non ancora cancellate) la cui ora (ora_reg) è inferiore o uguale al valore definito dalla funzione D**ATE SUB.**

Tale funzione ha il compito di andare a sottrarre due istanti tempo definiti.

L'operazione realizzata in questo caso consiste nella sottrazione dei parametri giorno, ora , minuti e secondi in cui si attiva il comando **(NOW ( ))** con lo stesso istante di tempo ma del giorno precedente **(INTERVAL 1 DAY).**

Anche in questo caso è necessario rendere automatico il comando attraverso la definizione di un evento.

CREATE EVENT eliminazione ON SCHEDULE EVERY 30 MINUTE STARTS 'AAAA-MM-GG hh-mm-ss' ON COMPLETION NOT PRESERVE DELETE FROM dati_giornalieri WHERE ora_reg <= DATE_SUB(NOW(), INTERVAL 1 DAY);

5.3 Costruzione delle componenti di restituzione del dato lato utente

5.3.1 Introduzione

Una componente importante del sistema di gestione dei dati che ancora non è stata presa in considerazione, consiste nella restituzione "a schermo" dei dati controllati e acquisiti.

La necessità di gestire e rendere disponibili i dati ondametrici è stata delineata anche dalla normativa vigente con riferimento all'art. 2 comma 7 del decreto legislativo n. 180 del 1998 (convertito nella legge 267/98) dove si vanno a configurare le linee guida per il potenziamento di tutti i piani di monitoraggio e di conseguenza delle reti di misurazione dove avviene l'acquisizione del dato fisico.

Inoltre, con la Direttiva del Presidente del Consiglio dei Ministri del 27 febbraio 2004 dal titolo "indirizzi operativi per la gestione organizzativa e funzionale del sistema di allertamento nazionale e regionale per il rischio idrogeologico ed idraulico ai fini di protezione civile" viene esaltata l'importanza dei sistemi di gestione di dati contenenti osservazioni qualitative e quantitative, dirette e strumentali dell’evento meteo-marino in atto, insieme alla previsione a breve termine dei relativi effetti attraverso misure raccolte in tempo reale rilevate da differenti reti di rilevamento.

Tali reti registrano dati che dopo lo storage e il controllo all'interno del database vengono messi a disposizione attraverso un sistema di richiesta lato utente, realizzato tramite un struttura tipo pagina web dove i soggetti interessati possono decidere cosa visualizzare attraverso delle selezioni.

La selezione del dato comporta da parte dell'intero sistema di gestione, l'elaborazione dei dati in esso contenuti.

Nell'architettura di sistema presentata nel capitolo 3 si mette in risalto la possibilità di far elaborare il dato attraverso due mezzi differenti, vale a dire la tabella e il grafico.

La tabella, posta all'interno del database relazionale, è un insieme di elementi (valori) organizzati in colonne verticali (ognuna identificata con il proprio nome) e righe orizzontali.

Viene definita cella, quell'unità in cui la riga e la colonna si intersecano.

Una tabella del database ha un numero specificato di colonne ma può avere un numero qualsiasi di righe.

Per quanto riguarda la parte grafica, è stato deciso di realizzare grafici a linee.

Tale decisione è stata presa in quanto il grafico a linee si presta meglio ad una rappresentazione dell'andamento del fenomeno ondoso rispetto al procedere del tempo ed è la tipologia di grafico che viene utilizzata da tutti gli strumenti scientifici per dare una rapida e comprensiva rappresentazione dei dati ondametrici.

Come detto nel capitolo precedente, l'elaborazione è legata all'uso di linguaggi di come php, html e css.

Tali linguaggi, insieme, concorrono alla realizzazione di pagine web nelle quali i dati possono essere richiamati e visualizzati nei due formati citati precedentemente.

Sulla base della varietà di dati che si ha a disposizione è stato deciso di suddividere la parte relativa alla restituzione all'utente in due distinte parti che corrispondono a due differenti pagine web.

Queste pagine web sono realizzate per rendere maggiormente fruibili i dati anche da utenti non esperti.

Le pagine realizzate si suddividono nel seguente modo:

- home, che consiste nella pagine introduttiva al sistema di restituzione dati lato utente, nella quale si trovano i riferimenti alle pagine relative all'elaborazione dei dati presi in considerazione;

- dati giornalieri, dove si possono visualizzare i dati riferiti alle ultime 24 ore. Al suo interno, i dati sono organizzati in una tabella e in 3 grafici distinti. La tabella permette la visualizzazione di tutti i campi contenuti in dati_giornalieri (quelli specificati nella definizione di questa componente nel paragrafo 5.1) mentre dal punto di vista grafico è stato deciso di mostrare l'andamento dei parametri, Hmo (altezza significativa), Tp (periodo di picco) e Dmt (direzione media di propagazione rispetto al tempo;

- archivio, dove si possono visualizzare tutti quei dati appartenenti alla tabella della componente del database denominata archivio. Rispetto alla pagina dati giornalieri non si ha l'immediata restituzione del dato ma si da all'utente la possibilità di scegliere il parametro (Hmo, Tp e Dmt) e l'intervallo di tempo da osservare. L'elaborazione avviene in tre modalità differenti, sotto forma di tabella, di grafico e rosa delle altezza d'onda. L'intervallo di tempo selezionabile non può andare oltre ad una settimana.

La motivazione principale per cui è stato realizzato un sistema di restituzione dei dati lato utente del genere si trova nella definizione della struttura dei principali siti istituzionali che, per quanto riguarda il lavoro svolto nella realizzazione di questa tesi, hanno significato un esempio da seguire.

Siti di gestione di dati ondametrici, che registrano e gestiscono dati direttamente dalla strumentazione posta in mare, sono la Rete Ondametrica Nazionale (RON) e idromare (servizio mareografico legato all'ISPRA).

Una suddivisione delle pagine web, così come è stata realizzata, può sicuramente migliorare la fruibilità dei dati e rendere più comprensivo il quadro dello stato locale del mare nella zona dove è avvenuta la misurazione oppure rendere maggiormente comprensibile i dati provenienti da un modello numerico.

5.3.2La struttura della componente “Home”

La home page è la prima pagina relativa al sistema di restituzione dati lato utente e si tratta di una schermata che permette la visione e l'utilizzo della componente gestionale dei dati.

Il suo scopo è quello di introdurre qualsiasi tipo di utente alla visualizzazione e all'uso dei dati ondametrici attraverso un'interfaccia che si divide in:

- una parte testuale, la quale descrive brevemente il contenuto del sito realizzato;
- una parte che richiama a siti esterni;
- una mappa con segnaposto.

La struttura dell'intera pagina viene definita dal codice HTML e CSS, il quale viene richiamato attraverso la seguente riga:

```
<link type ="text/css" rel= "stylesheet" href="css/homestyle.css" media="all">
```

La componente relativa al collegamento ai siti esterni è stata realizzata definendo un'apposita sezione (definita dal comando HTML div) denominata “link”.

Al suo interno sono state impostate delle immagini che funzionano da collegamenti ipertestuali a pagine web indicate dalla rappresentazione dell'immagine stessa.

Viene riportato un esempio di codice:

```
<div id="link">
<a href="http://www.unipi.it/"><img src="unipi.png" width="100" height="100"/></a>
</div>
```

Cliccando sull'immagine che riporta il logo dell'Università di Pisa, si apre la pagina web relativa.

La parte testuale, insieme alla mappa, è contenuta nella parte centrale della pagina (il corpo), definita dal linguaggio HTML tramite una sezione definita “elaborazione”

Per rendere più leggibile il sito, tale sezione è stata suddivisa in due ulteriori sotto sezioni chiamate “testo”, contenente solamente la parte testuale e

“mappa”,contente la carta.

Di seguito viene riportata la struttura con la quale è stato impostato il codice relativo alle componenti suddette.

```
<div id="elaborazione">
    <div id="testo">
        "testo introduttivo al sistema di gestione dei dati"
    </div>
    <div id="mappa">
        <center>
            <?php include 'mappa.php';?>
        </center>
    </div>
</div>
```

La mappa con segnaposto è stata realizzata con una pagina di programmazione a parte denominata "mappa.php".

L'intero file viene riportato in appendice A, mentre le componenti più importanti per la comprensione del suo funzionamento sono riportate di seguito.

All'interno del head, in una struttura HTML, viene richiamato uno script con riferimento alla mappa utilizzata (in questo caso la mappa di "Google Maps").

```
<script src="https://maps.googleapis.com/maps/api/js?v=3.exp&signed_in=true">
</script>
```

Richiamata la mappa, è stata definita una funzione denominata "initialize" dove vengono inizializzate delle variabili che servono a stabilire le coordinate del punto preso in considerazione (dove si trova la boa presa in analisi), l'altezza di visualizzazione (zoom) e il punto dove viene centrata.

```
function initialize() {
    var myLatlng = new google.maps.LatLng(43.928331,9.826669); // corrisponde alla posizione della boa ondametrica di La Spezia che è stata presa come riferimento.
```

```
var mapOptions = {
                    zoom: 10,
                    center: myLatlng
                };
```

In questo caso si nota che il punto messo in evidenza corrisponde alle coordinate dove viene centrata la mappa.

A questo punto, la mappa così costruita non presenta alcun segnaposto.

L'aggiunta del segnaposto e della sua casella di testo è avvenuta tramite l'inserimento delle "opzioni" alla mappa.

Tali "opzioni" vengono attivate tramite l'inizializzazione della variabile chiamata "map":

```
var map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);
```

Viene definito il testo che è stato inserito nella casella di testo del segnaposto (da notare che riprende la struttura HTML):

```
var contentString = '<div id="content">
    '<div id="siteNotice">'+
    '</div>'+
        '<h1 id="firstHeading" class="firstHeading"></h1>'+
    '<div id="bodyContent">'+
        '       <p><a href=dati_giornalieri.php>'+
                '<b>Boa La Spezia</b> </a> '+
                '</p>'+
                '<p>' +
                'LAT: 43.928331 N <br>'+
                'LONG: 9.826669 E'+
                '</p>'+
    '</div>'+
'</div>';
```

Dopo aver specificato il testo e il collegamento alla pagina relativa alla gestione dei dati per la boa presa in considerazione, avvenuto tramite il comando HTML "href", sono state realizzate la casella di testo e il segnaposto.

```
var infowindow = new google.maps.InfoWindow     ({
```

```
                content: contentString          // casella di testo
                });

var marker = new google.maps.Marker({
                position: myLatlng,     //segnaposto
                map: map,
                });
```

5.3.2La struttura della componente "Dati Giornalieri"

La pagina web denominata dati giornalieri (che fa riferimento allo script dati_giornalieri.php) è una componente del sistema di restituzione dei dati lato utente realizzata mediante php, HTML e i fogli di stile CSS.

Il richiamo al foglio di stile avviene attraverso le seguente riga contenuta all'interno del head di pagina:

```
<link type ="text/css" rel= "stylesheet" href="css/style.css" media="all">
```

La funzione principale della pagina è quella di richiamare i dati delle ultime 24 ore secondo due specifiche modalità di elaborazione, vale a dire la visualizzazione dei dati mediante tabella e grafico.

Il richiamo di queste due tipologie di elaborazioni avviene attraverso il "click" sui relativi nomi "tabella" e "grafici" i quali rappresentano delle ancore (dei riferimenti, link) alle due pagine php che permettono la costruzione della tabella (tabella_giornaliera.php) e dei grafici (grafici_giornalieri.php).

I due riferimenti sono riconoscibili attraverso l'utilizzo dello specifico tag HTML denominato **<href>**

```
<a href = "tabella_giornaliera.php"> Tabella </a>
<a href = "grafici_giornalieri.php"> Grafici </a>
```

Altri riferimenti si trovano sulla colonna sinistra del corpo, dove compare un menu non ordinato**(<ul>)** con un'ancora che porta alla apertura della pagina archivio.

```
<ul id="menu">
    <li>
    <a href = "archivio.php"> Archivio </a>
    </li>
</ul>
```

Tutta la struttura della pagina viene descritta in appendice A.

Come detto prima, cliccando su tabella, comparirà la tabella contenente i dati

giornalieri.

La pagina che appare all'utente ha la stessa struttura di quella precedente (dati_giornalieri.php) in quanto il foglio di stile CSS di riferimento è lo stesso (in generale è lo stesso per tutte le pagine, quindi questo aspetto non viene più ripetuto in seguito).

La parte che interessa l'utente, la tabella, viene elaborata attraverso il seguente codice (Fig, 5.5 ; 5.6; 5.7; 5.8; 5.9).

```php
<?php

$connessione = mysql_connect("localhost", "root", "********")
        or die("Connessione non riuscita: " . mysql_error());
        mysql_select_db ("dati_ondametrici");

$query="SELECT * FROM dati_giornalieri";
$risultati=mysql_query($query);
$num=mysql_numrows($risultati);




?>
```

Figura 5.5 : Costruzione della tabella - serie di comandi relativi all'aperutra del database

L'elaborazione della tabella, richiede l'apertura del database nel quale sono immagazzinati i dati (Fig. 5.5).

Ciò avviene con lo specifico comando **"mysql_connect"**, il quale permette l'accesso al database attraverso la specificazione di tre parametri relativi al server, all'username e alla password con la quale si proteggono i dati contenuti nel database.

Altro passo è stato quello di selezionare il database dati_ondametrici con il comando **"mysql_select_db"**.

La tabella deve riportare tutti i dati contenuti all'interno di dati_giornalieri, per cui si è scelto di realizzare un query valida per questo tipo di operazione, vale a dire una semplice selezione **(SELECT)** di tutto il contenuto di ogni colonna **(*)** della tabella dati_giornalieri.

La variabile "$query", inizializzata con la query appena definita, viene richiamata all'interno della funzione mysql_query che trasforma una riga scritta in php in una richiesta per il database gestito con MySQL.

Il risultato della query, contenuto nella variabile "$risultati", viene inserito

all'interno della funzione **"mysql_num_rows"** la quale restituisce il numero di righe prelevate che vanno a inizializzare la variabile "$num".

Dopo l'invio al database della richiesta sotto forma di query, la procedura prevede l'inizio della realizzazione della struttura della tabella(Fig 5.6)[43]

```
<table id="dati" border="0" cellspacing="2" cellpadding="2">

 <tr>
 <th><font face="Arial, Helvetica, sans-serif">Data</font></th>
 <th><font face="Arial, Helvetica, sans-serif">Ora</font></th>
 <th><font face="Arial, Helvetica, sans-serif">Hmo</font></th>
  <th><font face="Arial, Helvetica, sans-serif">Hmax</font></th>
 <th><font face="Arial, Helvetica, sans-serif">Tp</font></th>
 <th><font face="Arial, Helvetica, sans-serif">Tm</font></th>
 <th><font face="Arial, Helvetica, sans-serif">Dmt</font></th>
 <th><font face="Arial, Helvetica, sans-serif">Dmp</font></th>
 <th><font face="Arial, Helvetica, sans-serif">Dmw</font></th>
 <th><font face="Arial, Helvetica, sans-serif">Tmp</font></th>
 </tr>
```

Figura 5.6: Costruzione della tabella - Intestazione tabella

La figura 5.6, permette la costruzione dell'intestazione della tabella che viene realizzata sfruttando gli specifici tag già definiti nel capitolo 4 per quanto riguarda l'uso di HTML.

```
<?php
     $i=0;
     while ($i < $num) {

          $data=mysql_result($risultati,$i,"data_reg");
          $ora=mysql_result($risultati,$i,"ora_reg");
          $hmo=mysql_result($risultati,$i,"Hmo");
          $hmax=mysql_result($risultati,$i,"Hmax");
          $tp=mysql_result($risultati,$i,"Tp");
          $tm=mysql_result($risultati,$i,"Tm");
          $dmt=mysql_result($risultati,$i,"Dmt");
          $dmp=mysql_result($risultati,$i,"Dmp");
          $dmw=mysql_result($risultati,$i,"Dmw");
          $tmp=mysql_result($risultati,$i,"Tmp");
?>
```

Figura 5.7: Costruzione della tabella - Raccolta dati

Realizzata l'intestazione si è utilizzato l'operatore while predisponendo il loop (Fig 5.7):

```
$i=0;
     while ($i < $num)        {
```

//CODICE PER LA STAMPA DEI DATI

$i++

}

che eseguirà il codice il numero definito di volte.

Ogni volta $i viene incrementato di uno dichiarando allo scritp quale riga deve essere letta.

Visto che l'indice della prima linea è 0, il loop funzionerà correttamente fino a quando ci saranno righe da leggere.

Il codice per la stampa dei dati è stato realizzato tramite l'inizializzazione delle variabili che contengono i dati relativi alla colonna che viene presa in considerazione.

Il tutto viene eseguito attraverso la funzione mysql-result che si occupa di assegnare ogni informazione presente nell'array alla sua variabile.

```
<tr>

<td><font face="Arial, Helvetica, sans-serif"><?php echo $data;?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><?php echo $ora;?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><?php echo $hmo;?></font></td>
 <td><font face="Arial, Helvetica, sans-serif"><?php echo $max;?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><?php echo $tp;?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><?php echo $tm;?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><?php echo $dmt;?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><?php echo $dmp;?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><?php echo $dmw;?></font></td>
<td><font face="Arial, Helvetica, sans-serif"><?php echo $tmp;?></font></td>


</tr>
```

Figura 5.8: Costruzione della tabella - Visualizzazione a schermo

In figura 5.8 si possono osservare tutti i comandi HTML che permettono di visualizzare a schermo una tabella con tutte le relative celle contenenti i dati che vengono richiamati dal loop definito precedentemente.

Cliccando su "grafici", vengono visualizzati i grafici relativi all'andamento dell'altezza significativa, del periodo di picco e della direzione media di propagazione in funzione delle ultime 24 ore.

Tali grafici, vengono inseriti nella pagine tramite il richiamo realizzato nel seguente modo:

```
<div id="elaborazione">
    <img src="grafico_Hmo.php" />
    <img src="grafico_Tp.php" />
    <img src="grafico_Dmt.php"/>
</div>
```

In questo modo vengono richiamate le immagini che fanno riferimento alle tre pagine php dove è avvenuta la vera e propria elaborazione.

Nella seguente immagine (Fig 5.9) si può vedere la procedura utilizzata per la costruzione del grafico relativo alla direzione media di propagazione (Dmt).

La stessa procedura è stata utilizzata per costruire quelli relativo a Hmo e Tp.

```
include("phpgraphlib.php");
$graph = new PHPGraphLib(500,400);

$connessione = mysql_connect("localhost", "root", "*******")
        or die("Connessione non riuscita: " . mysql_error());
        mysql_select_db ("dati_ondametrici") or die('Nessun database presente');;

        $dataArray=array();


$sql="SELECT * FROM dati_giornalieri";
$result=mysql_query($sql) or die('query fallita: ' . mysql_error());


if ($result){

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

    $ora_reg = $row["ora_reg"];
    $Dmp = $row["Dmp"];

    $dataArray[$ora_reg]=$Dmp;

  }
}
```

Figura 5.9: Procedura per la costruzione dei grafici.

L'elaborazione è avvenuta attraverso l'inclusione di una libreria appositamente scaricata dal web che si chiama phpgraphlib.php[19] [20].

Tale strumento permette che una variabile sia inizializzata con una funzione che stabilisce la realizzazione di un nuovo grafico (new PHPGraphLib) avente una certa dimensione orizzontale e verticale.

Come è stato già visto in precedenza è risultato necessario aprire una connessione stabile al database, così da poter effettuare il prelievo dei dati.

Nella fi 5.9 si vede che è stato stabilito un comando "if" al cui interno contiene uno "while".

Il risultato della query ("$result") viene inserito nella funzione "mysql_fetch_array", la quale carica le righe di ciò che è stato prelevato all'interno di un'array associativo.

Le righe ("$row") corrisponderanno ai dati da inserire in ascissa ("ora_reg") e quelli da inserire in ordinata (in questo caso Dmt).

L'inserimento avviene all'interno dell'array, inizializzato prima del ciclo "$dataArray".

La costruzione definitiva del grafico si ha con la definizione del seguente comando, che permette l'invio della richiesta direttamente alla libreria.

```
$graph ->addData($dataArray);
```

Di seguito sono elencati i parametri che possono modificare lo stile della rappresentazione grafica.

```
$graph->setTitle("titolo");          // definisce il titolo
$graph->setBars(false);              // grafico a barre
$graph->setLine(true);               // grafico a linee
$graph->setDataPoints(true);         // definisce punti per l'incrocio dei valori
$graph->setDataPointColor("red");    // colore dei punti
$graph->setDataValues(true);         // stampa i valori sugli assi
$graph->setDataValueColor("black");  // colore dei valori
```

Alla fine si è specificato il comando che avvia la creazione del grafico.

```
$graph->createGraph( );
```

5.3.3 La struttura della componente "Archivio"

La pagina web denominata archivio (che fa riferimento allo script archivio.php) è una componente del sistema di restituzione dei dati lato utente realizzata mediante php, HTML e i fogli di stile CSS.

Nella parte relativa la head, oltre al richiamo del foglio di stile CSS unico per tutte le pagine realizzate, è stato inserito un riferimento alla pagina scritta in javascript che costruisce il calendario consentendo la selezione dell'intervallo di tempo:

```
<script language="JavaScript" src="ts_picker.js">
</script>
```

Lo scopo della pagina archivio è quello di permettere la visualizzazione a schermo di dati che vengono selezionati sulla base di un parametro (Hmo,Tp,e Dmt) e di un intervallo di tempo che non superi 1 settimana nel caso della visualizzazione in tabella e in grafico.

Nel caso del grafico a rosa delle altezza d'onda, si ha la visualizzazione dei dati relativi alla Hmax distribuiti sulla base della direzione di propagazione dell'onda stessa (sono stati realizzati dei settori il cui arco è definito dagli intervalli di tabella 5.1).

Nome settore	Intervallo Angoli in °	
N	337.5	22.5
NE	22.5	67.5
E	67.5	112.5
SE	112.5	157.5
S	157.5	202.5
SW	202.5	247.5
W	247.5	292.5
NW	292.5	337.5

Tabella 5.1: Divisione dei singoli settori

Per ogni settore (Nord, N-Ovest, Ovest, S-Ovest, Sud, S-Est, Est e N-Est) viene contato il numero di rilevazioni ondametriche corrispondenti.

Ad ogni rilevazione, oltre che alla direzione di propagazione, si ha una misura di altezza massima d'onda.

Quindi il singolo settore viene a sua suddiviso sulla base del valore di altezza.

Nella figura 5.10 si vedono le divisioni realizzate sulla base dell'intensità del fenomeno ondoso.

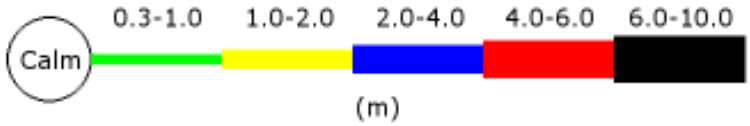

Figura 5.10: intervalli di altezza d'onda per settore di propagazione

L'intervallo di tempo selezionabile viene lasciato libero ma si consiglia, per rendere maggiormente visibile e comprensibile l'elaborazione grafica, di selezionare intervalli di tempo superiori a 1 mese.

La "rosa" è stata realizzata grazie ad una libreria php presa dalla serie di librerie JpGraph (http://jpgraph.net/).

Il motivo per cui è stato scelto questo software consiste nella sua praticità ed intuitività nell'uso e nella versatilità dei programmi già pre-compilati.

La pagina archivio.php si differenzia dalla pagina dati_ondametrici.php in quanto contiene quello che nel linguaggio HTML viene chiamato form.

Il form permette di inviare delle richieste lato utente che vengono tradotte in comandi sotto forma di query lato database.

```
<p> Selezionare il parametro
<select name="par" class="combobox1">
<option value></option>
<option value="Hmo"> Hmo </option>
<option value="Tp"> Tp   </option>
<option value="Dmp"> Dmp </option>
</select> </p>
"
    dal
    "
<input type="Text" name="data_reg_i" value="">
<a href= "javascript:show_calendar('document.selezione.data_reg_i', document.selezione.data_reg_i.value);">
<img src="immagini/cal.gif" width="16" height="16" border="0" alt="Seleziona la data"></a>
"
    al
    "
<input type="Text" name="data_reg_f" value="">

<a href= "javascript:show_calendar('document.selezione.data_reg_f', document.selezione.data_reg_f.value);">
<img src="immagini/cal.gif" width="16" height="16" border="0" alt="Seleziona la data"></a>

<input type="submit" value="INVIA">
```

Figura 5.11 Contenuto , con i relativi tag, del form relativo alla richiesta

L'immagine (Fig. 5.11) si riferisce al contenuto del form.

Si osserva come attraverso il tag HTML <select> si è realizzato il menu di opzione, dove ciascun parametro risulta compreso all'interno del tag <option> .

Inoltre, il tag <select> ha come attributi name (il cui valore "par" verrà riscritto nella variabile che si trova nello script dove avviene l'elaborazione, praticamente è ciò che consente il collegamento fra il file che contiene il form e il file d'elaborazione.) e class che ne definisce lo stile di visualizzazione (legato proprio al foglio di stile CSS).

Il valore che nel tag <option> viene assegnato a "value" permette di collegare la richiesta lato utente con il contenuto di una colonna (Hmo, Tp e Dmt) che si trova all'interno della tabella archivio nel database.

Per quanto riguarda la selezione da calendario, si è utilizzato il tag input che ha il compito di costruire sulla pagina web la casella dove l'utente può selezionare la data richiesta.

Il tag <input> si presenta con l'attributo name che, come già scritto in precedenza, serve per collegare questa parte di form al foglio di elaborazione e un value non espresso.

In questo caso, viene data l'opportunità di scegliere la data direttamente da un calendario, che appare premendo sull'immagine a destra della casella di testo.

Questo calendario è stato direttamente scaricato da http://www.javascriptkit.com/script/script2/timestamp.shtml e modificato per renderlo utilizzabile in ambiente MySQL.

La modifica si è resa necessaria in quanto il database memorizza la data nel formato AAAA-MM-GG mentre il calendario risultava impostato per la restituzione di una data nel formata GG-MM-AAAA.

Per meglio approfondire l'argomento, si rimanda all'appendice A dove sono riportati tutti i codici realizzati per la restituzione lato utente dei dati.

Il risultato ottenuto è mostrato in fig. 5.12:

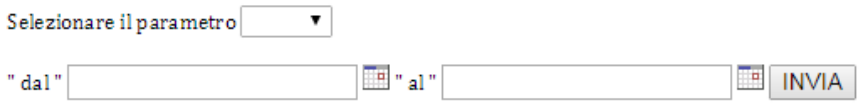

Figura 5.12: Visualizzazione del form a schermo

Dall'immagine si osservano le caselle di testo dove poter effettuare la scelta sia del parametro che dell'intervallo di tempo.

Per quanto riguarda l'elaborazione della tabella e del grafico, le modalità con cui vengono costruite sono simili a quelle già viste nel caso dei dati giornalieri.

Le differenze a cui si può accennare consistono nella comparsa delle variabili che definiscono il collegamento fra il form sulla pagina web (quindi la richiesta) e la tabella archivio situata nel database dati ondametrici e la diversa definizione della query, in quanto il set di dati da prelevare nel database è differente.

Nel primo caso vengono dichiarate le seguenti variabili:

```
$par=$_POST['par'];
$data_reg_i=$_POST['data_reg_i'];
$data_reg_f= $_POST ['data_reg_f'];
```

Le variabili sono inizializzate utilizzando "$_POST" che consiste in un array che provvede a recuperare ciò che è stato richiesto attraverso il form, sotto forma di coppie chiave-valore (http://www.phpnews.it/corsi/corso-php-base/i-form-e-larray-$_post/)

Nel secondo caso, la query utilizzata è la seguente:

$sql="SELECT data_reg,ora_reg,$par FROM archivio WHERE data_reg BETWEEN '$data_reg_i' AND '$data_reg_f' ";

Viene inizializzata la variabile $sql con un query che va a selezionare (SELECT) la data, l'ora e il valore che viene assegnato alla variabile $par, da (FROM) archivio dove (WHERE) i parametri prelevati rientrano all'interno dell'intervallo di tempo definito da una data iniziale (data_reg_i) e una data finale (data_reg_f).

Altro aspetto, da tenere fortemente in considerazione nell'ambito della visualizzazione dei dati, è la rosa delle altezza d'onda.

6.Risultati

Il lavoro di tesi e il precedente tirocinio, hanno portato alla realizzazione dell'intero sistema di gestione dei dati.

Per comprenderne a fondo il funzionamento in tutte le sue componenti, dall'acquisizione fino alla restituzione lato utente, si è voluto effettuare una prova con un set di dati ondametrici reali.

Tali dati sono stati direttamente scaricati dalla pagina web idromare che gestisce le rilevazioni della Rete Ondametrica Nazionale (RON) (http://www.idromare.it/analisi_grafici_ron_intro.php).

Questo sistema di misurazione dei parametri del moto ondoso è presente dal 1989. Nel complesso, alle ultime rilevazioni disponibili che risalgono a fine anno 2014, la rete può contare su 15 boe dislocate lungo le coste italiane. Per adesso ha avuto un rendimento, in termini di rilevazione e archiviazione dati, prossimo al 92% riuscendo a restituire una grande quantità di dati validi per qualsiasi tipologia di elaborazione scientifica ed ingegneristica. A partire dal 1 Gennaio 2015 il servizio di rilevamento di dati ondametrici è stato sospeso. Il set di dati utilizzati per il test di prova del sistema, sono corrispondenti alle misurazioni effettuate dal 1989 al 2005 per un totale 84567 di dati misurati.

Osservando attentamente l'intera serie di dati scaricati, si è da subito resa necessaria una pre-elaborazione degli stessi per renderli compatibili con le caratteristiche del database. L'elaborazione è stata effettuata con un foglio di calcolo ed ha consistito nella variazione della formattazione. Questo significa che si è dovuto modificare la formattazione della data (da GG/MM/AAAA a AAAA-MM-GG) e cambiare il separatore delle cifre decimali (dalla virgola al punto).

La mole di dati pre-elaborata è stata copiata dal foglio di calcolo ad un file di testo e poi successivamente caricata direttamente nella tabella "archivio" del database "dati ondametrici" tramite la seguente query:

LOAD DATA LOCAL INFILE 'percorso del file.txt all'interno del PC' INTO TABLE archivio;

Per verificare la funzionalità delle procedura di acquisizione, quindi il popolamento della tabella "dati giornalieri" è stato realizzato un file di testo contenente una grande quantità di dati ondametrici (circa 10^4 dati) già correttamente formattati, il quale funziona da "sorgente".

A questa "sorgente" è stato legato lo specifico programma di acquisizione dati (si veda l'appendice A per approfondimento) permettendo il trasferimento di una riga di dati , con un intervallo di tempo pari a 30 minuti, all'interno della tabella dati giornalieri.

Il vero e proprio test di funzionamento consiste nella verifica del funzionamento dei vari automatismi che permettono il passaggio dei dati dal file sorgente, alla loro restituzione lato utente tramite la visualizzazione a pagina web.

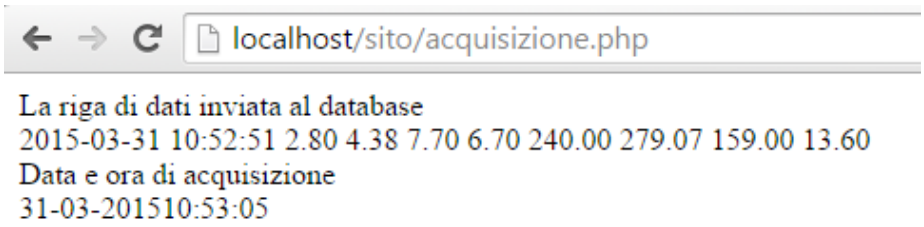

Figura 6.2: schermata di acquisizione dal file sorgente alla tabella dati giornalieri del database

L'avviamento della procedura di acquisizione, avviene aprendo il browser alla pagina sito/acquisizione.php, come si vede in figura 6.2.
Dalla Fig. 6.2 si osserva che la riga di dati inviata al database ha un orario leggermente differente (dell'ordine dei secondi) rispetto alla sua ora di acquisizione.

Questa piccola discrepanza è dovuta dall'intervallo di tempo necessario per l'avviamento manuale della procedura di acquisizione.

Appena la riga di dati è caricata nella tabella "dati giornalieri", questa viene soggetta ad una serie di eventi (come visto nel cap. 5) che permettono l'eliminazione di quei dati che non corrispondono alla realtà fisica del modo ondoso.

Nella Fig. 6.3 appare la finestra relativa alla descrizione di un evento (in questo caso controllo0).

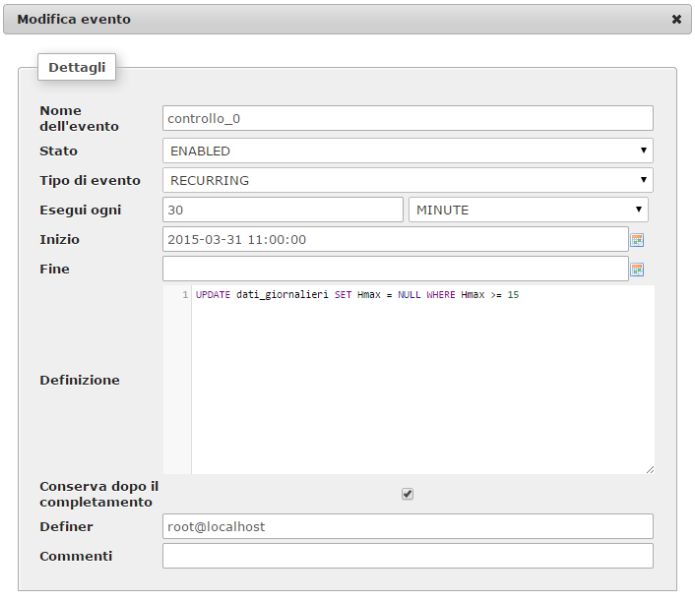

Figura. 6.3: finestra che mostra come è strutturato un evento

In questo esempio, con l'evento si è andati a definire quella query che permette la correzione dei dati appartenenti al parametro Hmax (altezza massima).

UPDATE dati_giornalieri SET Hmax=NULL WHERE Hmax >= 15;

Qualsiasi valore di altezza massima che superi o sia uguale a 15 m viene automaticamente sostituito con il valore NULL.

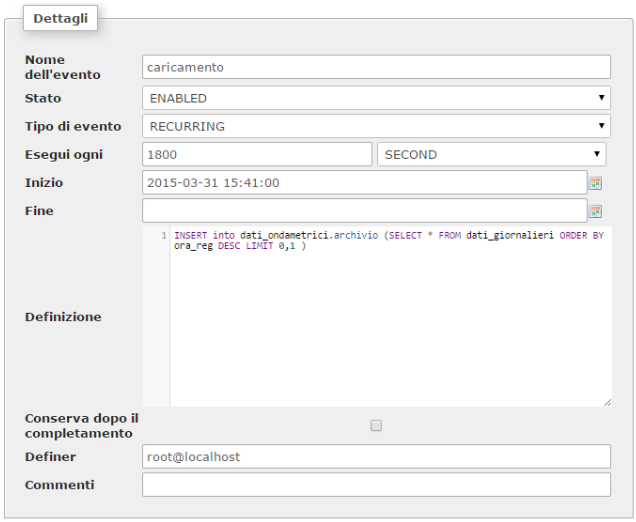

Figura 6.4: Caricamento delle righe di dati da dati_giornalieri a archivio

Dopo la correzione delle singole righe di dati, queste devono essere copiate nella

tabella archivio (Fig. 6.4), dove verranno "immagazzinate" fino a quando il database rimane operativo.

Con l'attivazione dell'evento relativo al passaggio della riga di dati dalla tabella dati giornalieri alla tabella archivio, si completa la procedura gestionale.

Per quanto riguarda la restituzione dei dati lato utente, come già specificato nel capitolo 5, è stata realizzata una "home", la quale introduce all'uso e alla visualizzazione dei dati ondametrici.

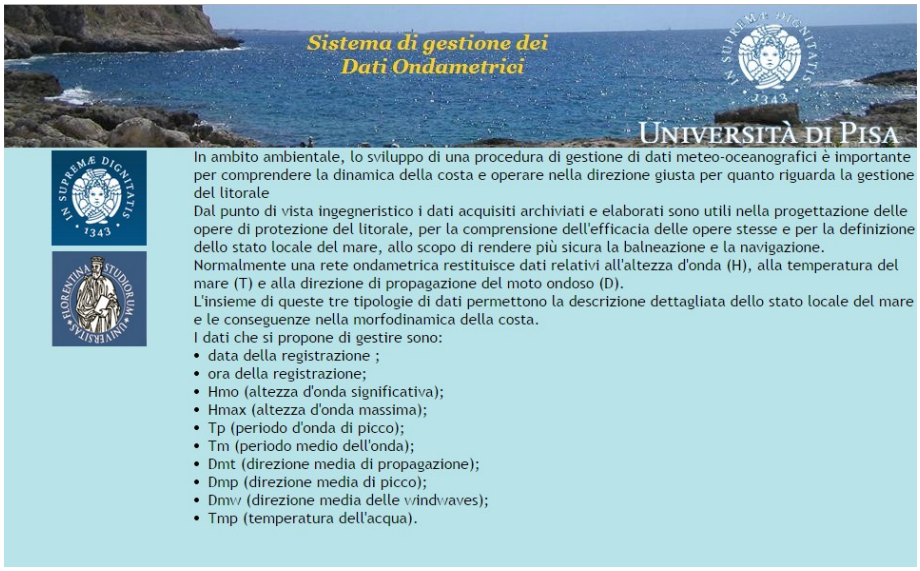

Fig. 6.5: la home page del sito, contenente l'introduzione ai dati ondametrici e i collegamenti alla pagine istituzionali.

La pagina presenta una componente introduttiva dove vengono caratterizzati i dati gestiti (Fig. 6.5) e una mappa con dei segnaposto indicanti la posizione di sensori e boe, che possono usufruire della piattaforma appena realizzata. La porzione di mappa presa in considerazione (Fig. 6.6) corrisponde al settore di costa antistante Marina di Pisa.

La scelta è ricaduta su questo luogo in quanto sono presenti numerose opere di ingegneria marittima (in particolar modo barriere e pennelli) la cui sollecitazione da parte del moto ondoso, deve essere controllata sia per motivi di sicurezza che per comprendere la reale funzionalità delle opere stesse.

Quindi, il sistema realizzato, può fare da supporto a tali strumenti per evitare la perdita di dati utili e il loro successivo utilizzo per future analisi ed ulteriori elaborazioni.

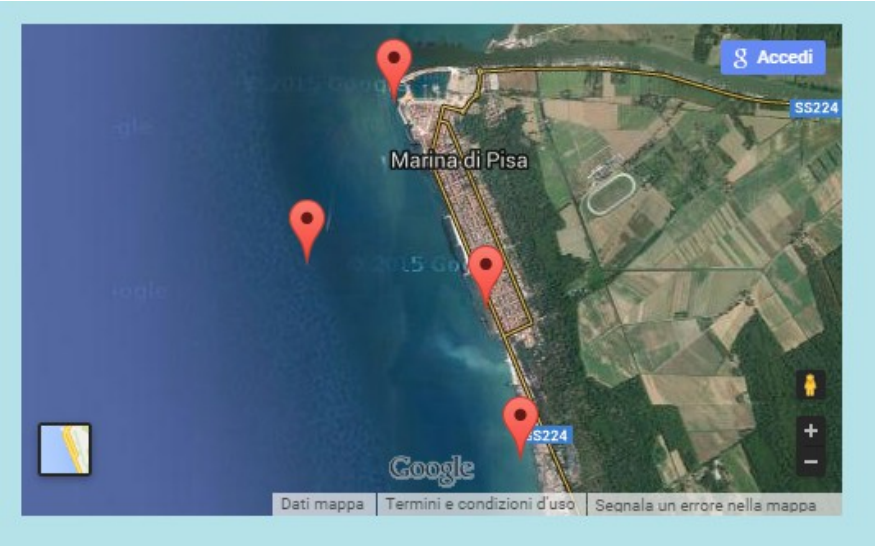

Fig. 6.6: Mappa della costa di Marina di Pisa, dove sono stati individuati dei punti che rappresentano possibili sensori e boe per la rilevazione dei parametri d'onda.

L'accesso ai dati, che come detto nei capitoli precedenti si dividono a livello temporale fra dati_giornalieri e archivio, avviene cliccando sul segnaposto posizionato in mare.

Sopra questo, si apre una casella di testo il cui contenuto è a sua volta cliccabile (Fig. 6.7).

Figura 6.7: Modalità di accesso ai dati ondametrici attraverso casella di testo messa insieme al segnaposto.

A questo punto si apre la pagina dati giornalieri, il cui contenuto consiste nei dati che lo strumento associato ha rilevato nelle ultime 24 ore.

Nella figura 6.8 si può notare la schermata principale.

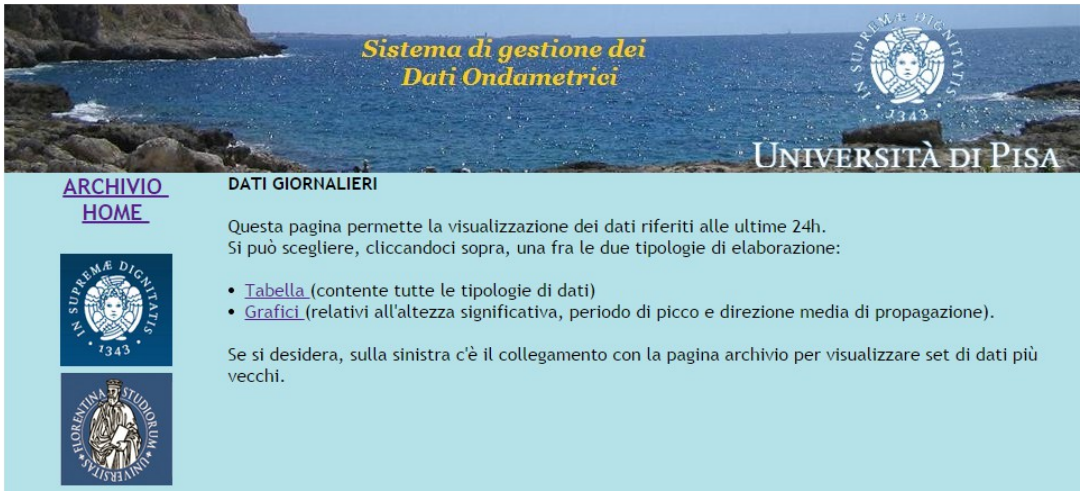

Figura 6.8: paina principale per la selezione dei dati giornalieri

Da questa pagina si possono effettuare due scelte che riguardano il tipo di elaborazione richiesta.

Nel caso della scelta della tabella si ottiene il seguente risultato (Fig. 6.9):

Data	Ora	Hmo	Hmax	Tp	Tm	Dmt	Dmp	Dmw	Tmp
2015-05-13	14:36:39	1.10	1.59	6.70	5.60	279.00	296.81	181.00	13.70
2015-05-13	14:38:19	0.80	1.19	7.10	5.30	281.00	302.15	209.00	12.10
2015-05-13	14:39:59	0.80	1.31	7.70	5.90	266.00	312.94	194.00	12.00
2015-05-13	14:41:39	0.80	1.18	5.60	5.40	282.00	327.91	174.00	11.50
2015-05-13	14:43:19		0.00						
2015-05-13	14:44:59		0.00						
2015-05-13	14:46:39		0.00						
2015-05-13	14:48:19		0.00						

Figura 6.9: Elaborazione della tabella all'interno della pagina dati giornalieri

Nella tabella di figura 6.9 rimangono vuote alcune celle in quanto il dato non è stato rilevato oppure è stato annullato dal sistema di controllo della validità.

Per quanto riguarda la scelta dell'elaborazione dei dati giornalieri sotto forma di grafico si ottengono i risultati come nelle figure 6.10, 6.11, 6.12

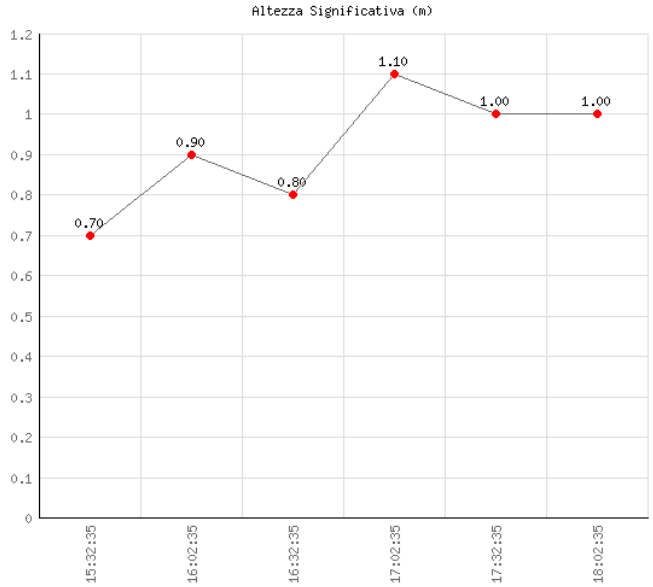

Figura 6.10: elaborazione grafica dei dati giornalieri relativi all'altezza significativa

:

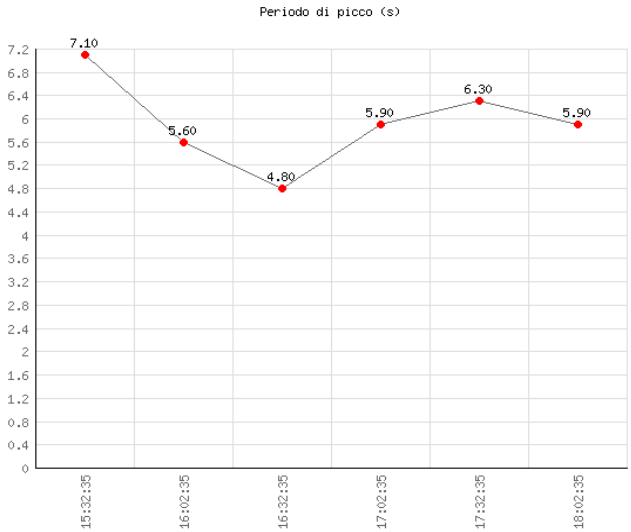

Figura 6.11 :elaborazione grafica dei dati giornalieri relativi al periodo di picco

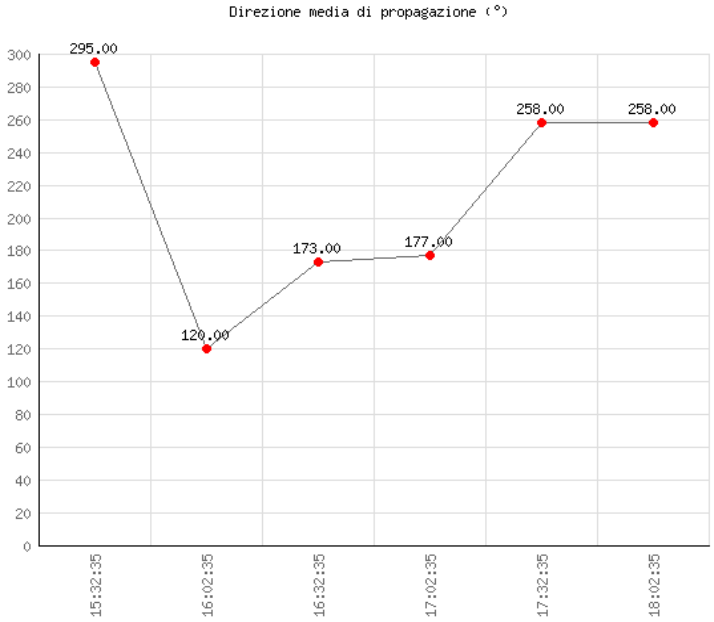

Figura 6.12 :elaborazione grafica dei dati giornalieri relativi alla direzione media di propagazione

La pagina principale relativa all'archivio si presenta come in Figura 6.11:

Figura 6.11: pagina principale dell'archivio da dove si possono le tipologie di elaborazioni possibili

Dalla figura 6.11 si nota come all'utente viene data la possibilità di scegliere fra: l'elaborazione tipo tabella, un'elaborazione tipo grafico e un'elaborazione dalla rosa delle altezza d'onda.

Nel caso delle prime due tipologie di elaborazione, la pagina che compare è quella di figura 6.12, dove viene data la possibilità di selezionare un parametro (fra Hmo, Tp e Dmt) e un intervallo di tempo che non superi 1 settimana.

Figura 6.12: Elaborazione dei dati sotto forma di tabella, dove l'utente può scegliere parametro e intervallo di tempo

Nel caso dell'elaborazione della rosa delle altezze d'onda, si da la possibilità di scegliere l'intervallo di tempo che desidera (Fig. 6.13).

Figura 6.13:pagina relativa all'elaborazione della rosa delle altezze d'onda dove l'utente può scegliere l'intervallo di tempo desiderato

Selezionato il tipo di elaborazione il sistema di gestione dei dati restituisce tabelle e grafici che si presentano con la solita formattazione vista nell'elaborazione dei dati giornalieri.

Nel caso dell'elaborazione tipo tabella, l'intervallo di tempo consentito è pari ad una settimana, ma cambia la quantità di dati visualizzabili-

Questo avviene in quanto l'archivio contiene dati che sono stati rilevati con due differenti cadenze temporali, vale dire che per date prima del 2001-12-29 le rilevazioni sono state realizzate ogni tre ore mentre successivamente ogni mezz'ora.

Nel primo caso la seleziona porta ad avere 56 osservazioni (8 per ogni giorno) mentre nel secondo caso ma i dati osservabili sono 192 (48 per ogni giorno).

Per quanto riguarda l'elaborazione grafica, la quantità massima di dati visualizzabili è pari a 50 perché si vuol mantenere il grafico leggibile e facilmente comprensibile.

Se viene selezionato un intervallo di tempo troppo ampio rispetto a quello consentito oppure che contiene troppi dati, si attiva un sistema di errore che porta al reindirizzamento verso la pagina di selezione del parametro e dell'intervallo di tempo dell'archivio, come si vede in figura 6.14 (l'esempio, in questo caso, si riferisce alla pagina relativa all'elaborazione dei grafici).

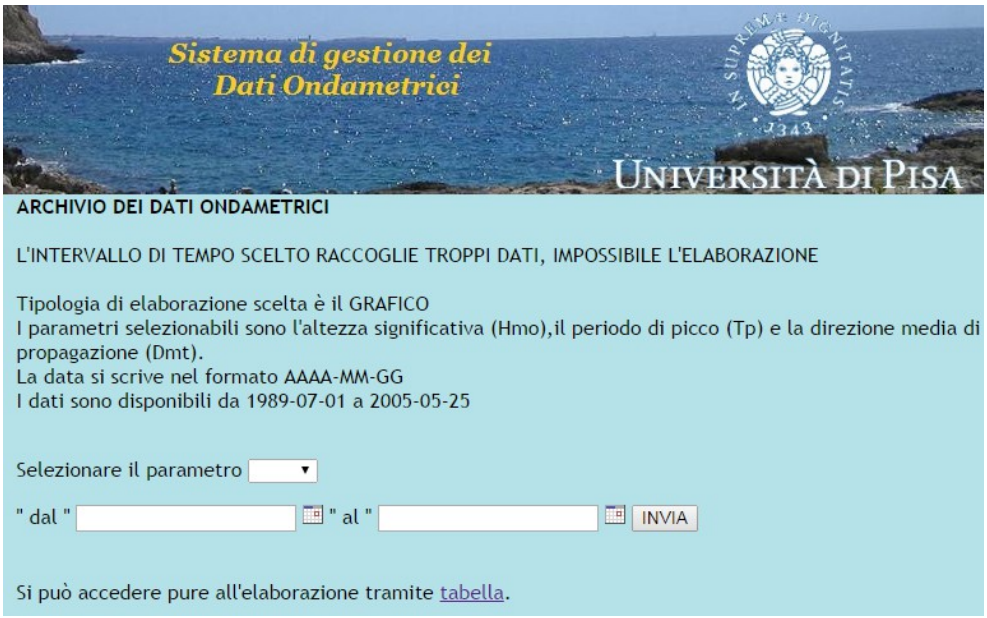

Figura 6.14: pagina che appare quando si sceglie un intervallo di tempo troppo ampio nell'elaborazione.

La rosa delle altezza d'onda, dopo aver selezionato un intervallo di tempo, viene visualizzata a schermo come in figura 6.15.

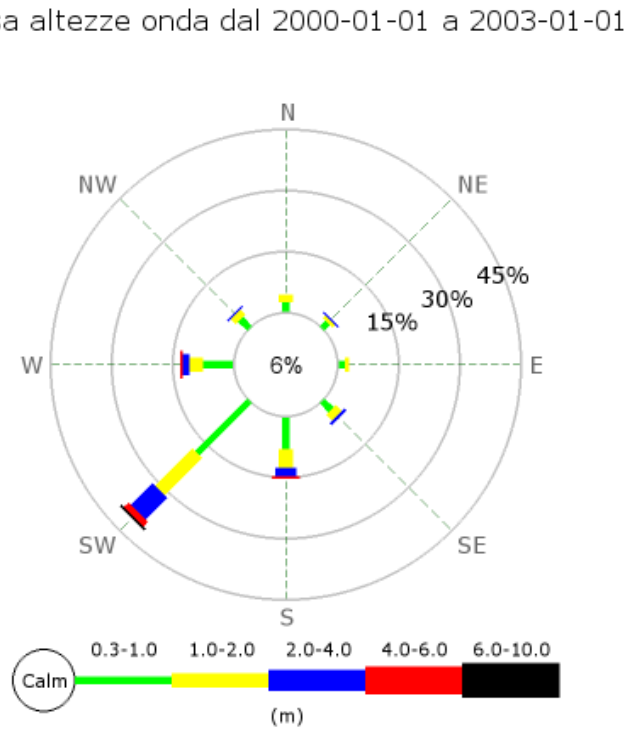

Figura 6.15: rosa delle altezza d'onda con i dati di Hmax e Dmt dal 2000-01-01 al 2003-01-01

7. Conclusioni

Il sistema realizzato permette la gestione di dati derivanti da una fonte generica, come boe ondametriche o modelli di simulazione numerica.

I software e i linguaggi di programmazioni gestiti di concerto dalla piattaforma XAMPP, hanno sopperito alla necessità di realizzare un sistema di gestione dei dati collegato ai dati derivanti dallo studio del fenomeno fisico del moto ondoso.

MySQL si è adattato all'organizzazione dei dati ondametrici in termini sia strutturali (database e tabelle con gli specifici tipi di dati), sia per quanto riguarda la costruzione delle query necessarie per la correzione di eventuali errori nella formattazione e nella validità del dato.

In generale, il linguaggio di programmazione php ha consentito di svolgere tutte le operazioni necessarie sia all'acquisizione che alla restituzione lato utente attraverso un'elaborazione grafica sviluppata in tabelle e grafici.

La sua semplicità d'uso e la sua struttura molto simile ad altri linguaggi di programmazione largamente utilizzati su scala mondiale (C e C++ prevalentemente) hanno permesso una rapida costruzione di tutte le procedure realizzate in questo lavoro di tesi.

L'insieme dei risultati ottenuti possono essere ripresi in più modalità, secondo le necessità degli utilizzatori finali.

Un ulteriore passo in avanti, rispetto a quello che già è stato fatto, consiste nell'approfondimento, nell'ambito dell'elaborazione, in termini statistici.

L'operatività del sistema di gestione dei dati può essere così lunga che, con i dovuti aggiornamenti, si possono avere dei dati all'interno dell'archivio tali da permettere elaborazioni statistiche con obiettivo previsionale anche per analisi degli eventi estremi.

Questo vuol dire che se il sistema viene affiancato ad un sistema di rilevamento (tipo boe ondametriche) si può avere un quadro completo della situazione del mare locale e di come le condizioni marine possono fluttuare lungo gli anni di rilevamento.

Inoltre, il sistema di restituzione lato utente, può essere correlato di una pagina web dove attraverso una mappa si possono individuare gli strumenti di

rilevamento georeferenziati e avere un quadro dello stato del mare in tempo reale.

La struttura dell'intero sistema di gestione però, oltre a fornire un servizio di immagazzinamento, elaborazione e restituzione di dati ondametrici, può essere modificata per far sì che siano gestiti dati di diversa natura.

La potenzialità del sistema consiste proprio in questo: tramite la modifica di determinati parametri è possibile occuparsi di anche di dati differenti da quelli di moto ondoso.

I parametri da modificare sono quelli che indicano al database il tipo di dato da immagazzinare e di conseguenza si dovranno modificare le caratteristiche delle colonne che racchiudono i dati stessi, come già è stato spiegato nei capitoli precedenti.

Per concludere, il punto di forza dell'intero lavoro di tesi consiste nell'aver costruito uno strumento estremamente versatile che può essere utilizzato non solo nel campo dello studio nel moto ondoso e che può dare un forte contributo nella raccolta e elaborazione di dati di qualsiasi genere.

8.Bibliografia

[1] A. Atzeni, M. Vallini Introduzione a XAMPP (http://security.polito.it/~lioy/01nbe/xampp.pdf).

[2] A. Carri.2006. Guida all'Application Server (http://www.html.it/guide/guida-application-server/) 11 Luglio 2006.

[3]Codd E. F.. 1970. A relational model of data for large shared data banks *Communications of the ACM* volume 13, numero 6 Giugno 1970.

[4]---. 1985. Is your DBMS Really Relational!? *ComputerWorld* 14 Ottobre 1985.

[5]---. 1985. Does your DBMS Run by the Rules *ComputerWorld* 21 Ottobre 1985.

[6] C. Grau. 2010. Basi di dati e modello relazionale (http://www.html.it/pag/16421/basi-di-dati-e-modello-relazionale/) *Guida pratica php e MySQL 15 Gennaio 2010* paragrafo 2.

[7] ---. 2010. La connessione a MySQL (http://www.html.it/pag/16422/la-connessione-a-mysql/) *Guida pratica php e MySQL 15 Gennaio 2010* paragrafo 3.

[8] ---. 2010. La chiusura della connessione a MySQL(http://www.html.it/pag/16422/la-connessione-a-mysql/) *Guida pratica php e MySQL 15 Gennaio 2010* paragrafo 4.

[9] C. Lamanna. 2013.Classificazione degli elementi HTML (http://www.html.it/pag/14210/classificazione-di-elementi-e-tag-xhtml/) *Guida CSS di base 24 Giugno 2010* paragrafo 3.

[10] D. Andrioli, S. Cortesi, M.Cucinato, P. Olson, L. Perugini. 1997-2015. Funzioni MySQL.(http://php.net/manual/it/ref.mysql.php) *Manuale di PHP* ©1997-2015 Gruppo di Documentazione PHP

[11]---. 1997-2015. Struttura del linguaggio e Tipi di dati. (http://php.net/manual/it/language.types.php) *Manuale di PHP* . ©1997-2015 Gruppo di Documentazione PHP

[12] ---. 1997-2015. Le strutture di controllo. (http://php.net/manual/it/language.control-structures.php). *Manuale di PHP* ©1997-2015 Gruppo di Documentazione PHP

[13] ---. 1997-2015. La funzione explode.(http://php.net/manual/it/function.explode.php). *Manuale di PHP* ©1997-2015 Gruppo di Documentazione PHP

[14] D. Axmark, M. Widenius. 2015. CREATE EVENT Syntax. (http://dev.mysql.com/doc/refman/5.6/en/create-event.html) *MySQL 5.6 Reference Manual* . 05 Marzo 2015 (revisione n° 41916) paragrafo 13.2.6.

[15] ---. 2015. LOAD DATA INFILE Syntax (http://dev.mysql.com/doc/refman/5.6/en/load-data.html) *MySQL 5.6 Reference Manual* . 05 Marzo 2015 (revisione n° 41916) paragrafo 13.1.11.

[16] Decreto Legge 11 giugno 1998, n. 180. "Misure urgenti per la prevenzione del rischio idrogeologico ed a favore delle zone colpite da disastri franosi nella regione Campania".*(G.U. n.134 del 11-6-1998)*

[17] Direttiva del Presidente del Consiglio dei Ministri del 27 febbraio 2004. "Indirizzi operativi per la gestione organizzativa e funzionale del sistema di allertamento nazionale e regionale per il rischio idrogeologico ed idraulico ai fini di protezione civile" *(G. U. n. 59 del 11 marzo 2004).*

[18] D. Ragget, A. Le Hors, I. Jacobs. 1999 *HTML 4.10 Specificatin W3C Recommendation* (http://www.w3.org/TR/html401/) 24 Dicembre 1999.

[19] E. Brueggeman, 2013. PHPGraphLib Graphing Library (http://www.ebrueggeman.com/phpgraphlib) *The website of Elliot Brueggeman*, 2013.

[20] ---. 2013. MySQL and PHPGraphLib (http://www.ebrueggeman.com/phpgraphlib/documentation/tutorial-mysql-and-phpgraphlib) *The website of Elliot Brueggeman*, 2013.

[21] F. Stutto. 2006. Rendere php disponibile nel sistema (http://www.html.it/pag/16577/rendere-php-disponibile-nel-sistema/) *Guida PHP su Windows* 17 Mag. 2006 paragrafo 5.

[22] --- 2006. Personalizzazione del file php.ini (http://www.html.it/pag/16579/personalizzazione-del-file-phpini/) *Guida PHP su Windows* 17 Mag. 2006 paragrafo 8.

[23] G. Farina. **2006.** I tipi di dato (http://www.html.it/pag/16389/i-tipi-di-dato-i/) *Guida PHP teorica* 29 Maggio 2006 paragrafi 4-5.

[24] ---. **2006.** Le strutture di controllo: if, else e else if (http://www.html.it/pag/16393/strutture-di-controllo-if-else-e-else-if/) *Guida PHP teorica* 29 Maggio 2006 paragrafo 8.

[25] ---. **2006.** Le strutture di controllo: while, for, switch (http://www.html.it/pag/16394/strutture-di-controllo-while-for-switch/) *Guida PHP teorica* 29 Maggio 2006 paragrafo 9.

[26] G. Beaver, M. Jansen, M. Wieseman . *Pear documentation* (http://pear.php.net/manual/) the Pear website.

[27] G. Gillini. 2006. Introduzione ai RDBMS (http://www.html.it/pag/32137/introduzione-ai-rdbms/) *Guida MySQL* 18 Aprile 2006 paragrafo 1.

[28]---. **2006.** I tipi di tabelle MySQL (http://www.html.it/pag/32144/tipi-di-tabelle/) *Guida MySQL* 18 Aprile 2006 paragrafo 8.

[29]---. **2006.** I tipi di dato MySQL (http://www.html.it/pag/32145/tipi-di-dati/) *Guida MySQL* 18 Aprile 2006 paragrafo 9.

[30] ---. **2006.** Caratteristiche e vantaggi di PHP (http://www.html.it/pag/16675/caratteristiche-e-vantaggi-di-php/) *Guida PHP di base* 10 Luglio 2006 paragrafo 3.

[31] ---. **2006.** PHP e HTML (http://www.html.it/pag/16676/php-e-lhtml/) *Guida PHP di base* 10 Luglio 2006 paragrafo 4.

[32] ---. **2006.** Le variabili (http://www.html.it/pag/16679/le-variabili3/) *Guida PHP di base* 10 Luglio 2006 paragrafo 7.

[33] ---. **2006.** I tipi di dato (http://www.html.it/pag/16680/i-tipi-di-dato/ *Guida PHP di base* 10 Luglio 2006 paragrafo 8.

[34] ---. **2006.** Espressioni e operatori aritmetici di PHP (http://www.html.it/pag/16681/espressioni-e-operatori-aritmetici-di-php/) *Guida PHP di base* 10 Luglio 2006 paragrafo 9.

[35] ---. **2006.** Gli operatori logici e le espressioni booleane in PHP (http://www.html.it/pag/16682/gli-operatori-logici-e-le-espressioni-booleane-in-php/) *Guida PHP di base* 10 Luglio 2006 paragrafo 10.

[36] ---. **2006.** Gli Array (http://www.html.it/pag/16688/gli-array2/) *Guida PHP di base* 10 Luglio 2006 paragrafo 16.

[37] ---. **2006.** Le variabili GET e POST (http://www.html.it/pag/16695/le-variabili-get-e-post/) *Guida PHP di base* 10 Luglio 2006 paragrafo 23.

[38] ---. **2006.** Accedere ai file (http://www.html.it/pag/16698/accedere-ai-file/) *Guida PHP di base* 10 Luglio 2006 paragrafo 26.

[39] H.Shih. 2003 *Triaxys Directional Wave Buoy for Nearshore Wave Measurements* - Test and Evaluation Plan, NOAA Technical Report, Silver Spring, Maryland, Gennaio 2003.

[40] Istituto Superiore per la Protezione e la Ricerca Ambientale, portale Idromare. *Rete Ondametrica Nazionale – Tabella – Selezione dei dati* (http://www.idromare.it/analisi_tabelle_ron_intro.php)

[41] Legge 3 agosto 1998, n. 267. "Conversione in legge, con modificazioni, del decreto-legge 11 giugno 1998, n. 180, recante misure urgenti per la prevenzione del rischio idrogeologico ed a favore delle zone colpite da disastri franosi nella regione Campania". *(G. U. n. 183 del 7 agosto 1998)*

[42] L. Forti. 2006. Perché usare PhpMyAdmin (http://www.html.it/pag/32011/introduzione84/) *Guida PhpMyadmin* 30 Marzo 2006 paragrafo 1.

[43] L. Lemay. 1995. *Il manuale HTML- un'introduzione.* McGraw-Hill Italy ISBN 88 386 0353-7 Novembre 1995 capitoli 3,4,5,6,12.

[44] L.Ruggiero. 2005. *Utilizzo del tag di base* (http://www.mrwebmaster.it/html/utilizzo-tag-base_6742.html) 4 Maggio 2005.

[45] L. H. Holthuijsen. **2010**. *Waves in Oceanic and Coastal Waters* Febbraio 2010 ISBN 978-113-9462-52-5 , cap 3,4.

[46] M. Valente. 2006 I *Meta tag come scriverli correttamente* (http://www.html.it/articoli/i-meta-tag-come-scriverli-correttamente-1/) 20/03/2006.

[47] P. Du Bois. 2004. *MySQL*. Pearson Educational Italia ISBN 88-7192-196-8, tavola dei contenuti da pg xvi a xvii

[48] R. G. Dean, R. A. Dalrymple. 1991 *Water Wave Mechanics for Engineers and Scientists,* Gennaio 1991 ISBN: 978-981-02-0420-4, cap.2,3,4.

[49] J. Lewis, N.B. Dale. 2004 *Computer Science Illuminated* (2° edizione) *P*aperback, 699 Pages, Published by Jones & Bartlett Publishers ISBN-13: 978-0-7637-0799-6, ISBN: 0-7637-0799-6

[50] W. Cecchin. 2006. Come funziona un browser(http://www.html.it/pag/16027/come-funziona-un-browser/) *Guida HTML* 7 Marzo 2006 paragrafo 2.

[51] ---. 2006. Lo standard HTML (http://www.html.it/pag/16028/prima-di-cominciare-davvero-lo-standard-html/) *Guida HTML* 7 Marzo 2006 paragrafo 3.

[52] --- 2006. I tag del HTML (http://www.html.it/pag/16030/i-tag-dellhtml-come-scriverli/) *Guida HTML* 7 Marzo 2006 paragrafo 59.

[53] --- 2006. Il Doctype DTD (http://www.html.it/pag/16084/il-doctype-dtd/) *Guida HTML* 7 Marzo 2006 paragrafo 5.

[54] W. Kamphuis, *Introduction to Coastal Engineering and Management* Maggio 2010, ISBN: 978-981-283-484-3, cap 3,5.

[55] Y. Goda *Random Seas and Design of Maritime Structures* Giugno 2010, ISBN: 978-981-4282-39-0, cap.1,2.

9.APPENDICE A

Programma di acquisizione

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional/dtd"_>


<!--
##########################################################################
#####

        All'interno di questa pagina ci sono tutti i comandi necessari affinchè
        avvenga il caricamento automatico dei dati all'interno della tabella
        dati_giornalieri.

##########################################################################
####### !-->

<html xmlns = "http://www.w3.org/1999/xhtml"_>

<head>

<title> Caricamento dati </title>
<meta http-equiv="refresh" content="1800">

</head>

<?php


$prova = file_exists("dati_giornalieri.txt");

if ($prova==true){

        $var=fopen("dati_giornalieri.txt", "r");
        $var2=file("dati_giornalieri.txt");
        fclose($var);

        $i=0;

                for($a=0; $a<=$i;$a++){

                        $ris= $var2[$a];
                        echo "La riga di dati inviata al database<br>";
                        echo $ris;

                }

        $cmp=explode ("          ", $ris);

                $i=$cmp[0];
                $b=$cmp[1];
                $c=$cmp[2];
                $d=$cmp[3];
                $e=$cmp[4];
                $f=$cmp[5];
```

```
                $g=$cmp[6];
                $h=$cmp[7];
                $l=$cmp[8];
                $m=$cmp[9];


        $connessione = mysql_connect("localhost", "root", "dbondametrico")
        or die("Connessione non riuscita: " . mysql_error());
        mysql_select_db ("dati_ondametrici") or die('Nessun database presente');


                $sql="INSERT INTO copia
                (data_reg,ora_reg,Hmo,Hmax,Tp,Tm,Dmt,Dmp,Dmw,Tmp) VALUES
        ('$i','$b','$c','$d','$e','$f','$g','$h','$l','$m')";
                $result=mysql_query($sql) or die('query fallita: ' . mysql_error());

} else {

        echo "il file dati sorgente non è presente nella directory";

        }
?>

</html>
```

Pagina di visualizzazione della home

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional/dtd"_>

<html xmlns = "http://www.w3.org/1999/xhtml"_>

<!--
##########################################################################
#

        Questa pagina contiene tutto il codice relativo alla realizzazione della homa page
        al cui interno c'è una descrizione del lavoro relativo all'elaborazione dei dati (tabelle
        e grafici)


##########################################################################
##!-->

<head>

<title> D.O. : Home</title>
<!-- Riferimento al foglio di stile !-->

<link type ="text/css" rel= "stylesheet" href="css/homestyle.css" media="all">

</head>


<body>

<!-- Questo tag va a definire il contenitore, dove tutto è compreso !-->

        <div id = "contenitore">

                <div id="head">

        <!-- Immagine dell'intestazione !-->

        <img src="porto.jpg" width="960" height="150" border="0" alt="">

                </div>


                <div id="pagina">

                        <div id="link"style="height:800px">

                <!-- Organizzazione di tutti i link che si trovano nella parte sinistra della pagina !-->
                <center> <a href="http://www.unipi.it/" target="_blank"><img src="unipi.png"
                width="100" height="100"  border="" vspace="3" hspace="3"  /></a> </center>
                <center> <a href="http://www.unifi.it/" target="_blank"><img src="unifi.png"
                width="100" height="100"  border="" vspace="3" hspace="3"  /></a> </center>
                        </div>
                        <div id="elaborazione">
```

```
<!-- Il corpo della pagina !-->

                                        <div id="testo">
Lo sviluppo di un sistema di gestione dei dati ondametrici è utile dal punto di vista ambientale
in quanto è un valido supporto per comprendere la dinamica di costa e operare nella direzione giusta per quanto riguarda la gestione del litorale.<br>
Dal punto di vista ingegneristico, questo sistema è utile nella progettazione delle opere di protezione del litorale, per la comprensione dell'efficacia delle opere stesse e per la definizione dello stato locale del mare,
allo scopo di rendere più sicura la balneazione e la navigazione. <br>
Una rete ondametrica restituisce dati relativi all'altezza d'onda (H), alla temperatura del mare (T) e
alla direzione di propagazione del moto ondoso (D).<br>
L'insieme di queste tre tipologie di dati permettono la descrizione dettagliata dello stato locale del mare e
le conseguenze nella morfodinamica della costa.<br>
<br>
I dati che si propone di gestire sono:<br>
<br>
<li>data della registrazione;</li>
<li>ora della registrazione;</li>
<li>Hmo (altezza d'onda significativa);</li>
<li>Hmax (altezza d'onda massima);</li>
<li>Tp (periodo d'onda di picco);</li>
<li>Tm (periodo medio dell'onda);</li>
<li>Dmt (direzione media di propagazione);</li>
<li>Dmp (direzione media di picco);</li>
<li>Dmw (direzione media delle windwaves);</li>
<li>Tmp (temperatura dell'acqua).</li> <br>
<br>
Nella carta si osservano dei segnaposti di esempio che indicano la posizione della strumentazione.<br>
Un segnaposto, se cliccato, apre una casella di testo la cui scritta rimanda alla pagina di gestione dei dati per quello strumento

                                        </div>
                                        <div id="mappa"style="height:400px">
                                <br>
                                <center>
        <!-- comando php che permette l'inclusione della  mappa   all'interno del corpo della pagina !-->
                                <?php include 'mappa.php';?>
                                </center>
                                        </div>
                                </div>
                        </div>

<div id="footer">

Info sull'attività <br> Sito di prova relativo alla gestione di dati ondametric

                        </div>
                </div>

</body>
</html>
```

Pagina relativa alla visualizzazione dei dati giornalieri

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional/dtd"_>

<html xmlns = "http://www.w3.org/1999/xhtml"_>

<!--
#########################################################################

        Questa pagina contiene una componente del sito, quella relativa ai dati_giornalieri.
        All'interno la pagina sarà suddivisa in due parti, quella relativa alla visualizzazione
        della tabella e quella relativa ai grafici.
        Nella tabella si va a visualizzare tutti i tipi di dati che sono stati trattati nel database mentre
        i grafici fanno riferimento al periodo di picco, all'altezza significativa e alla direzione media
di      propagazione.


#########################################################################
##!-->

<head>

<title> D.O. : dati giornalieri </title>

        <!-- Riferimento al foglio di stile !-->
        <link type ="text/css" rel= "stylesheet" href="css/homestyle.css" media="all">

</head>


<body>

<!-- Questo tag va a definire il contenitore, dove tutto è compreso !-->
<div id = "contenitore">

        <div id = "head">

        <!-- Immagine dell'intestazione !-->
        <img src="porto.jpg" width="960" height="150" border="0" alt="">

        </div>

                <div id = "pagina">

                        <div id = "link"style="height:800px">


                <center> <li> <a href = "archivio.php"> ARCHIVIO </a><br> </center>
                <center> <li> <a href = "home.php"> HOME        </a> <br> </center><br>

                <center> <a href="http://www.unipi.it/" target="_blank"><img src="unipi.png"
                width="100" height="100"  border="" vspace="3" hspace="3"  /></a> </center>
                <center> <a href="http://www.unifi.it/" target="_blank"><img src="unifi.png"
                width="100" height="100"  border="" vspace="3" hspace="3"  /></a> </center>
                        </div>
```

```
<!-- Il corpo della pagina !-->
                        <div id="elaborazione">


                                <div id = "testo">

                <b> DATI GIORNALIERI</b><br><br>
                Questa pagina permette la visualizzazione dei dati riferiti alle ultime 24h.
<br>
                Si può scegliere, cliccandoci sopra, una fra le due tipologie di elaborazione:
<br>
                <br>

                <li> <a href = "tabella_giornaliera.php"> Tabella </a> (contente tutte le
tipologie di             dati)<br>
                <li> <a href = "grafici_giornalieri.php"> Grafici </a> (relativi all'altezza
significativa,           periodo di picco e direzione media di propagazione).
                <br>
                <br>
                Se si desidera, sulla sinistra c'è il collegamento con la pagina archivio per
                visualizzare set di dati più vecchi.

        <br>
        <br>
        <br>
        <br>
        <br>
        <br>
                <img src="boa.jpg" width="750" height="500" border="0" alt="">


                                </div>


                        </div>


                </div>


        <div id="footer">

Info sull'attività <br>
Sito di prova relativo alla gestione di dati ondametrici realizzato d

        </div>


                <!-- Questo tag racchiude tutti gli elementi della pagina al di sotto del header !-->
</div>

</body>

</html>
```

Costruzione della tabella contenente i dati giornalieri

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional/dtd"_>

<html xmlns = "http://www.w3.org/1999/xhtml"_>

<!--
##########################################################################
#####

        Questa pagina contiene le funzioni php necessario all'accesso al database e alla tabella
        dove sono immagazzinati i dati. I dati vengono prelevati tramite una query e organizzati in
una tabella        inviata a schermo. I dati fanno riferimento alle ultime 24 h.

##########################################################################
######## !-->

<head>

<title> D.O. : tabella giornaliera </title>
<link type ="text/css" rel= "stylesheet" href="css/style.css" media="all">

</head>

<body>

        <div id = "contenitore">

        <div id = "header">

                <div id = "top_logo">
                <img src="header.png" width="255" height="86" border="0" alt="">
                </div>

                <div id = "top_logo_sx">
                <img src=" " width="705" height="86" border="0" alt="" usemap="#Sir">
                </div>

        </div>

        <div id ="divisore">
        <img src="divisore.jpeg" width="960" height="34" border="0" id="tale_center" alt="">
        </div>

                <div id = "pagina">
                        <table border="0" cellspacing="0" cellpadding="0">
                                <tbody>
                                <tr>
                                        <td valign="top" width="180">

                                        <div id="colsx">

                                                <ul id="menu">
                                                <li>
                                                        <a href = "home.php"> Home </a> <br>
                                                </li>
                                                <li>
```

```
                                        <a href = "archivio.php"> Archivio </a>
                                    </li>
                                </div>

                                </td>

                                <td valign="top" width="720">
                                <div id="titolo"> TABELLA DATI GIORNALIERI </div>

                                <div id ="testo">

                                    <div id = "riga">

                                    <a href = "tabella_giornaliera.php"> Tabella
</a>
                                    <a href = "grafici_giornalieri.php"> Grafici </a>

                                    </div>

<?php

        $connessione = mysql_connect("localhost", "root", "dbondametrico")
        or die("Connessione non riuscita: " . mysql_error());

            mysql_select_db ("dati_ondametrici");

                $query="SELECT * FROM dati_giornalieri";
                $risultati=mysql_query($query);
                $num=mysql_numrows($risultati);

?>

                                    <table id="dati" border="0" cellspacing="2"
cellpadding="2">
                                    <tr>

        <th><font face="Arial, Helvetica, sans-   serif">Data</font></th>
        <th><font face="Arial, Helvetica, sans-serif">Ora</font></th>
        <th><font face="Arial, Helvetica, sans-serif">Hmo</font></th>
        <th><font face="Arial, Helvetica, sans-serif">Hmax</font></th>
        <th><font face="Arial, Helvetica, sans-serif">Tp</font></th>
        <th><font face="Arial, Helvetica, sans-serif">Tm</font></th>
        <th><font face="Arial, Helvetica, sans-serif">Dmt</font></th>
        <th><font face="Arial, Helvetica, sans-serif">Dmp</font></th>
        <th><font face="Arial, Helvetica, sans-serif">Dmw</font></th>
        <th><font face="Arial, Helvetica, sans-serif">Tmp</font></th>

                                    </tr>

<?php
                $i=0;
                    while ($i < $num) {
                        $data=mysql_result($risultati,$i,"data_reg");
                        $ora=mysql_result($risultati,$i,"ora_reg");
                        $hmo=mysql_result($risultati,$i,"Hmo");
                        $hmax=mysql_result($risultati,$i,"Hmax");
                        $tp=mysql_result($risultati,$i,"Tp");
                        $tm=mysql_result($risultati,$i,"Tm");
                        $dmt=mysql_result($risultati,$i,"Dmt");
```

```
                                        $dmp=mysql_result($risultati,$i,"Dmp");
                                        $dmw=mysql_result($risultati,$i,"Dmw");
                                        $tmp=mysql_result($risultati,$i,"Tmp");
?>

                                    <tr>

                <td><font face="Arial, Helvetica, sans-serif"><?php echo $data;?></font></td>
                <td><font face="Arial, Helvetica, sans-serif"><?php echo $ora;?></font></td>
                <td><font face="Arial, Helvetica, sans-serif"><?php echo $hmo;?></font></td>
                <td><font face="Arial, Helvetica, sans-serif"><?php echo $hmax;?></font></td>
                <td><font face="Arial, Helvetica, sans-serif"><?php echo $tp;?></font></td>
                <td><font face="Arial, Helvetica, sans-serif"><?php echo $tm;?></font></td>
                <td><font face="Arial, Helvetica, sans-serif"><?php echo $dmt;?></font></td>
                <td><font face="Arial, Helvetica, sans-serif"><?php echo $dmp;?></font></td>
                <td><font face="Arial, Helvetica, sans-serif"><?php echo $dmw;?></font></td>
                <td><font face="Arial, Helvetica, sans-serif"><?php echo $tmp;?></font></td>

                                    </tr>

<?php

                                    $i++;
                                        }
            mysql_close();
?>

                                    </table>

                                    </div>

                                    </td>

                            </tr>
                            </tbody>

                        </table>

            </div>

<div id ="footer"> Info sull'attività

<ADDRESS> Davide Samuele Franchi
<a href = "mail to: davide.samuele.franchi@virgilio.it"> davide.samuele.franchi@virgilio.it
</a>

</div>

        </div>
</body>
</html>
```

Richiamo alla pagina va a costruire i grafici contenenti i dati giornalieri

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional/dtd"_>

<html xmlns = "http://www.w3.org/1999/xhtml"_>

<!--
#########################################################################
###

        Questa pagina contiene le funzioni php necessarie all'accesso al database e alla tabella
        dove sono immagazzinati i dati. I dati vengono prelevati tramite una query e organizzati in
tre grafici        differenti,contententi i valori di Hmo, Tp, e Dmp tutti in funzione del tempo. I dati
fanno riferimento alle     ultime 24 h.

#########################################################################
###### !-->

<head>

<title> D.O. : grafici giornalieri </title>
<link type ="text/css" rel= "stylesheet" href="css/style.css" media="all">

</head>

<body>

<div id = "contenitore">

        <div id = "header">

        <       div id = "top_logo">
                        <img src="header.png" width="255" height="86" border="0" alt="">
                </div>

                <div id = "top_logo_sx">
                        <img src=" " width="705" height="86" border="0" alt=""
usemap="#Sir">
                </div>

        </div>

        <div id ="divisore">
                <img src="" width="960" height="34" border="0" id="tale_center" alt="">
        </div>

                <div id = "pagina">
                        <table border="0" cellspacing="0" cellpadding="0">
                                <tbody>
                                <tr>

                                        <td valign="top" width="180">

                                                <div id="colsx">

                                                        <ul id="menu">
                                                        <li>
```

```
                                        <a href = "home.php"> Home
</a> <br>
                                    </li>
                                    <li>
                                        <a href = "archivio.php">
Archivio </a>
                                    </li>

                                </div>

                            </td>

                            <td valign="top" width="720">
                                <div id="titolo"> GRAFICI GIORNALIERI </div>

                                <div id ="testo">

                                    <div id = "riga">

                                    <a href = "tabella_giornaliera.php">
Tabella </a>
                                    <a href = "grafici_giornalieri.php">
Grafici </a>

                                    </div>

                                    <div id="elaborazione">
                                        <img src="grafico_Hmo.php" />
                                        <img src="grafico_Tp.php" />
                                        <img src="grafico_Dmp.php"/>

                                    </div>

                                </div>

                            </td>

                        </tr>
                        </tbody>

                    </table>

                </div>

<div id ="footer"> Info sull'attività

<ADDRESS> Davide Samuele Franchi
<a href = "mail to: davide.samuele.franchi@virgilio.it"> davide.samuele.franchi@virgilio.it
</a>

</div>
            </div>
</body>
</html>
```

Pagina che permetta la costruzione dei grafici con i dati giornalieri (esempio)

```php
<?php

//#####################################################################
//
// Questa pagina contiene la struttura che permette la realizzazione del grafico
// relativo ai dati giornalieri con riferimento al parametro del periodo di picco (Tp).
//
//#####################################################################


include("phpgraphlib.php");
	$graph = new PHPGraphLib(500,400);
		$connessione = mysql_connect("localhost", "root", "dbondametrico")
		or die("Connessione non riuscita: " . mysql_error());
			mysql_select_db ("dati_ondametrici") or die('Nessun database presente');

	$dataArray=array();
	$sql="SELECT * FROM dati_giornalieri";
	$result=mysql_query($sql) or die('query fallita: ' . mysql_error());

		if ($result){
			while($row = mysql_fetch_assoc($result)){

	$ora_reg = $row["ora_reg"];
	$Dmp = $row["Dmp"];
	$dataArray[$ora_reg]=$Dmp;}

}

		$graph ->addData($dataArray);
		$graph->setTitle("Direzione media di propagazione");
		$graph->setBars(false);
		$graph->setLine(true);
		$graph->setDataPoints(true);
		$graph->setDataPointColor("red");
		$graph->setDataValues(true);
		$graph->setDataValueColor("black");
		$graph->createGraph();

		mysql_close();

?>
```

Pagina contenente la realizzazione del form relativo alla richiesta lato utente

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional/dtd"_>

<html xmlns = "http://www.w3.org/1999/xhtml"_>

<!--
##########################################################################
#####

        Questa pagina contiene i comandi necessari affinchè l'utente possa selezionare,
        tramite l'apposito form, un intervallo di tempo (che non deve superare la settimana)
        e un parametro fra periodo di picco (Tp), altezza significativa (Hmo) e Direzione media
        di propagazione (Dmp).
        Tutto viene elaborato sotto forma di tabella


##########################################################################
######## !-->

<head>

        <title> D.O.: archivio </title>
        <link type ="text/css" rel= "stylesheet" href="css/style.css" media="all"/>
        <script language="JavaScript" src="ts_picker.js">

</head>

<body>

        <div id = "contenitore">

        <div id = "header">

                <div id = "top_logo">
                <img src="header.png" width="255" height="86" border="0" alt="">
                </div>

                <div id = "top_logo_sx">
                <img src=" " width="705" height="86" border="0" alt="" usemap="#Sir">
                </div>

        </div>

        <div id ="divisore">
        <img src="" width="960" height="34" border="0" id="tale_center" alt="">
        </div>

        <div id = "pagina">
                <table border="0" cellspacing="0" cellpadding="0">
                        <tbody>
                        <tr>
                                <td valign="top" width="180">

                                <div id="colsx">
                                        <ul id="menu">
```

```
                                        <li>
                                                <a href = "home.php"> Home </a> <br>
                                        </li>
                                        <li>
                                                <a href = "dati_giornalieri.php"> Dati
giornalieri </a>
                                        </li>
                                </div>

                                </td>

                                <td valign="top" width="720">
                                        <div id="titolo"> ARCHIVIO </div>

                                        <div id ="testo">

                        <p> ELABORAZIONE DELLLA TABELLA <p>


                                        div id= "form">

                        <form method="post" name="selezione"
action="archivio_tabella.php">

                        <p> Selezionare il parametro
                        <select name="par" class="combobox1">
                                <option value></option>
                                <option value="Hmo"> Hmo </option>
                                <option value="Tp"> Tp   </option>
                                <option value="Dmp"> Dmp </option>
                        </select> </p>
"
                                dal

        <input type="Text" name="data_reg_i" value="">
                <a href= "javascript:show_calendar('document.selezione.data_reg_i',
                        document.selezione.data_reg_i.value);">
                        <img src="immagini/cal.gif" width="16" height="16" border="0"
                        alt="Seleziona la  data"></a>
"
        al
                "
        <input type="Text" name="data_reg_f" value="">
                <a href= "javascript:show_calendar('document.selezione.data_reg_f',
                        document.selezione.data_reg_f.value);">
                        <img src="immagini/cal.gif" width="16" height="16" border="0"
                        alt="Seleziona la data"></a>

        <input type="submit" value="INVIA">

                        </form>

                        </div>

                </div>


                </div>
```

```
<p> L'intervallo temporale massimo selezionabile è pari ad una settimana </p>

                    </td>

            </div>
<div id ="footer"> Info sull'attività

<ADDRESS> Davide Samuele Franchi
<a href = "mail to: davide.samuele.franchi@virgilio.it"> davide.samuele.franchi@virgilio.it
</a>
</div>

            </div>

</body>


    </html>
```

Pagine di elaborazione della rosa delle altezza d'onda

```
<?php

//Avviene la connessione al database con i relativi parametri.

$connessione = mysql_connect("localhost", "root", "dbondametrico")
OR die("Connessione non riuscita: " . mysql_errAND());

        // Viene selezionato il database da cui prelevare i dati.
        mysql_select_db ("dati_ondametrici");


$data_reg_i=$_POST['data_reg_i'];
$data_reg_f= $_POST ['data_reg_f'];


//      Una variabile viene inizializzata con una query.

        $query="SELECT `Hmax`, `Dmt` FROM archivio WHERE `Dmt`>= 0 AND `Dmt`<= 360 AND
        (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";

//      Invia una query Mysql.

        $risultati=mysql_query($query);

//      Il VALORE $num sarà il numero delle righe memANDizzate in $risultati e
//      verrà utilizzato nel loop che si occuperà di recuperare i dati e
//      visualizzarli nel browser.

        $num=mysql_numrows($risultati);


//SELEZIONE E DIVISIONE PER CLASSI DI Hmax CONSIDERANDO VALORI PER Dmt per N

//Query che permette la selezione

$sql_0= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0 AND `Hmax`<0.3 ) AND
(`Dmt`>=337.5 OR `Dmt`< 22.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";

        $ris_0=mysql_query ($sql_0);//Interpretazione della query da parte di php
        $num_0=mysql_numrows($ris_0);//Conto le righe selezionate
        $per_0= ($num_0*100)/$num;    //Calcolo in che percentuale corrispondono al numero
totale di                                       misurazioni effettuate
        $ar_0 = round ($per_0,1);          //Arrotondo il VALORE percentuale al primo VALORE
decimale


$sql_1= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0.3 AND `Hmax`<=1 ) AND
(`Dmt`>=337.5 OR `Dmt`< 22.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_1=mysql_query ($sql_1);
        $num_1=mysql_numrows($ris_1);
        $per_1= ($num_1*100)/$num;
        $ar_1 = round ($per_1,1);

$sql_2= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>1 AND `Hmax`<=2 ) AND
(`Dmt`>=337.5 OR `Dmt`< 22.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_2=mysql_query ($sql_2);
```

```
        $num_2=mysql_numrows($ris_2);
        $per_2= ($num_2*100)/$num;
        $ar_2 = round ($per_2,1);

$sql_3= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>2 AND `Hmax`<=4 ) AND
(`Dmt`>=337.5 OR `Dmt`< 22.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_3=mysql_query ($sql_3);
        $num_3=mysql_numrows($ris_3);
        $per_3= ($num_3*100)/$num;
        $ar_3 = round ($per_3,1);

$sql_4= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>4 AND `Hmax`<=6 ) AND
(`Dmt`>=337.5 OR `Dmt`< 22.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_4=mysql_query ($sql_4);
        $num_4=mysql_numrows($ris_4);
        $per_4= ($num_4*100)/$num;
        $ar_4 = round ($per_4,1);

$sql_5= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>6) AND (`Dmt`>=337.5 OR `Dmt`<
22.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_5=mysql_query ($sql_5);
        $num_5=mysql_numrows($ris_5);
        $per_5= ($num_5*100)/$num;
        $ar_5= round ($per_5,1);

//SELEZIONE E DIVISIONE PER CLASSI DI Hmax CONSIDERANDO VALORI PER Dmt per NE



$sql_6= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0 AND `Hmax`<0.3 ) AND
(`Dmt`>=22.5 AND `Dmt`< 67.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_6=mysql_query ($sql_6);
        $num_6=mysql_numrows($ris_6);
        $per_6= ($num_6*100)/$num;
        $ar_6= round ($per_6,1);

$sql_7= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0.3 AND `Hmax`<=1 ) AND
(`Dmt`>=22.5 AND `Dmt`< 67.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_7=mysql_query ($sql_7);
        $num_7=mysql_numrows($ris_7);
        $per_7= ($num_7*100)/$num;
        $ar_7= round ($per_7,1);

$sql_8= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>1 AND `Hmax`<=2 ) AND
(`Dmt`>=22.5 AND `Dmt`< 67.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_8=mysql_query ($sql_8);
        $num_8=mysql_numrows($ris_8);
        $per_8= ($num_8*100)/$num;
        $ar_8 = round ($per_8,1);

$sql_9= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>2 AND `Hmax`<=4 ) AND
(`Dmt`>=22.5 AND `Dmt`< 67.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_9=mysql_query ($sql_9);
        $num_9=mysql_numrows($ris_9);
        $per_9= ($num_9*100)/$num;
        $ar_9= round ($per_9,1);


$sql_10= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>4 AND `Hmax`<=6 ) AND
```

```
(`Dmt`>=22.5 AND `Dmt`< 67.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_10=mysql_query ($sql_10);
        $num_10=mysql_numrows($ris_10);
        $per_10= ($num_10*100)/$num;
        $ar_10= round ($per_10,1);

$sql_11= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>6) AND (`Dmt`>=22.5 AND
`Dmt`< 67.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
$ris_11=mysql_query ($sql_11);
$num_11=mysql_numrows($ris_11);
$per_11= ($num_11*100)/$num;
$ar_11= round ($per_11,1);


//SELEZIONE E DIVISIONE PER CLASSI DI Hmax CONSIDERANDO VALORI PER Dmt per E

$sql_12= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0 AND `Hmax`<0.3 ) AND
(`Dmt`>=67.5 AND `Dmt`< 112.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_12=mysql_query ($sql_12);
        $num_12=mysql_numrows($ris_12);
        $per_12= ($num_12*100)/$num;
        $ar_12= round ($per_12,1);

$sql_13= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0.3 AND `Hmax`<=1 ) AND
(`Dmt`>=67.5 AND `Dmt`< 112.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_13=mysql_query ($sql_13);
        $num_13=mysql_numrows($ris_13);
        $per_13= ($num_13*100)/$num;
        $ar_13= round ($per_13,1);


$sql_14= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>1 AND `Hmax`<=2 ) AND
(`Dmt`>=67.5 AND `Dmt`< 112.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_14=mysql_query ($sql_14);
        $num_14=mysql_numrows($ris_14);
        $per_14= ($num_14*100)/$num;
        $ar_14 = round ($per_14,1);


$sql_15= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>2 AND `Hmax`<=4 ) AND
(`Dmt`>=67.5 AND `Dmt`< 112.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_15=mysql_query ($sql_15);
        $num_15=mysql_numrows($ris_15);
        $per_15= ($num_15*100)/$num;
        $ar_15= round ($per_15,1);


$sql_16= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>4 AND `Hmax`<=6 ) AND
(`Dmt`>=67.5 AND `Dmt`< 112.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_16=mysql_query ($sql_16);
        $num_16=mysql_numrows($ris_16);
        $per_16= ($num_16*100)/$num;
        $ar_16= round ($per_16,1);

$sql_17= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>6) AND (`Dmt`>=67.5 AND
`Dmt`< 112.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_17=mysql_query ($sql_17);
        $num_17=mysql_numrows($ris_17);
        $per_17= ($num_17*100)/$num;
```

```
        $ar_17= round ($per_17,1);
//SELEZIONE E DIVISIONE PER CLASSI DI Hmax CONSIDERANDO VALORI PER Dmt per SE

$sql_18= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0 AND `Hmax`<0.3 ) AND
(`Dmt`>=112.5 AND `Dmt`< 157.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
$ris_18=mysql_query ($sql_18);
$num_18=mysql_numrows($ris_18);
$per_18= ($num_18*100)/$num;
$ar_18= round ($per_18,1);

$sql_19= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0.3 AND `Hmax`<=1 ) AND
(`Dmt`>=112.5 AND `Dmt`< 157.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_19=mysql_query ($sql_19);
        $num_19=mysql_numrows($ris_19);
        $per_19= ($num_19*100)/$num;
        $ar_19= round ($per_19,1);


$sql_20= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>1 AND `Hmax`<=2 ) AND
(`Dmt`>=112.5 AND `Dmt`< 157.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_20=mysql_query ($sql_20);
        $num_20=mysql_numrows($ris_20);
        $per_20= ($num_20*100)/$num;
        $ar_20 = round ($per_20,1);


$sql_21= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>2 AND `Hmax`<=4 ) AND
(`Dmt`>=112.5 AND `Dmt`< 157.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_21=mysql_query ($sql_21);
        $num_21=mysql_numrows($ris_21);
        $per_21= ($num_21*100)/$num;
        $ar_21= round ($per_21,1);


$sql_22= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>4 AND `Hmax`<=6 ) AND
(`Dmt`>=112.5 AND `Dmt`< 157.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_22=mysql_query ($sql_22);
        $num_22=mysql_numrows($ris_22);
        $per_22= ($num_22*100)/$num;
        $ar_22= round ($per_22,1);

$sql_23= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>6) AND (`Dmt`>=112.5 AND
`Dmt`< 157.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_23=mysql_query ($sql_23);
        $num_23=mysql_numrows($ris_23);
        $per_23= ($num_23*100)/$num;
        $ar_23= round ($per_23,1);

//SELEZIONE E DIVISIONE PER CLASSI DI Hmax CONSIDERANDO VALORI PER Dmt per S

$sql_24= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0 AND `Hmax`<0.3 ) AND
(`Dmt`>=157.5 AND `Dmt`< 202.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_24=mysql_query ($sql_24);
        $num_24=mysql_numrows($ris_24);
        $per_24= ($num_24*100)/$num;
        $ar_24= round ($per_24,1);

$sql_25= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0.3 AND `Hmax`<=1 ) AND
(`Dmt`>=157.5 AND `Dmt`< 202.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
```

```
        $ris_25=mysql_query ($sql_25);
        $num_25=mysql_numrows($ris_25);
        $per_25= ($num_25*100)/$num;
        $ar_25= round ($per_25,1);


$sql_26= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>1 AND `Hmax`<=2 ) AND
(`Dmt`>=157.5 AND `Dmt`< 202.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_26=mysql_query ($sql_26);
        $num_26=mysql_numrows($ris_26);
        $per_26= ($num_26*100)/$num;
        $ar_26 = round ($per_26,1);


$sql_27= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>2 AND `Hmax`<=4 ) AND
(`Dmt`>=157.5 AND `Dmt`< 202.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_27=mysql_query ($sql_27);
        $num_27=mysql_numrows($ris_27);
        $per_27= ($num_27*100)/$num;
        $ar_27= round ($per_27,1);


$sql_28= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>4 AND `Hmax`<=6 ) AND
(`Dmt`>=157.5 AND `Dmt`< 202.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_28=mysql_query ($sql_28);
        $num_28=mysql_numrows($ris_28);
        $per_28= ($num_28*100)/$num;
        $ar_28= round ($per_28,1);

$sql_29= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>6) AND (`Dmt`>=157.5 AND
`Dmt`< 202.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_29=mysql_query ($sql_29);
        $num_29=mysql_numrows($ris_29);
        $per_29= ($num_29*100)/$num;
        $ar_29= round ($per_29,1);

//SELEZIONE E DIVISIONE PER CLASSI DI Hmax CONSIDERANDO VALORI PER Dmt per SW

$sql_30= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0 AND `Hmax`<0.3 ) AND
(`Dmt`>=202.5 AND `Dmt`< 247.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_30=mysql_query ($sql_30);
        $num_30=mysql_numrows($ris_30);
        $per_30= ($num_30*100)/$num;
        $ar_30= round ($per_30,1);

$sql_31= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0.3 AND `Hmax`<=1 ) AND
(`Dmt`>=202.5 AND `Dmt`< 247.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_31=mysql_query ($sql_31);
        $num_31=mysql_numrows($ris_31);
        $per_31= ($num_31*100)/$num;
        $ar_31= round ($per_31,1);


$sql_32= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>1 AND `Hmax`<=2 ) AND
(`Dmt`>=202.5 AND `Dmt`< 247.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_32=mysql_query ($sql_32);
        $num_32=mysql_numrows($ris_32);
        $per_32= ($num_32*100)/$num;
        $ar_32 = round ($per_32,1);
```

```
$sql_33= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>2 AND `Hmax`<=4 ) AND
(`Dmt`>=202.5 AND `Dmt`< 247.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_33=mysql_query ($sql_33);
        $num_33=mysql_numrows($ris_33);
        $per_33= ($num_33*100)/$num;
        $ar_33= round ($per_33,1);


$sql_34= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>4 AND `Hmax`<=6 ) AND
(`Dmt`>=202.5 AND `Dmt`< 247.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_34=mysql_query ($sql_34);
        $num_34=mysql_numrows($ris_34);
        $per_34= ($num_34*100)/$num;
        $ar_34= round ($per_34,1);

$sql_35= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>6) AND (`Dmt`>=202.5 AND
`Dmt`< 247.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_35=mysql_query ($sql_35);
        $num_35=mysql_numrows($ris_35);
        $per_35= ($num_35*100)/$num;
        $ar_35= round ($per_35,1);


//SELEZIONE E DIVISIONE PER CLASSI DI Hmax CONSIDERANDO VALORI PER Dmt per W

$sql_36= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0 AND `Hmax`<0.3 ) AND
(`Dmt`>=247.5 AND `Dmt`< 292.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_36=mysql_query ($sql_36);
        $num_36=mysql_numrows($ris_36);
        $per_36= ($num_36*100)/$num;
        $ar_36= round ($per_36,1);

$sql_37= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0.3 AND `Hmax`<=1 ) AND
(`Dmt`>=247.5 AND `Dmt`< 292.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_37=mysql_query ($sql_37);
        $num_37=mysql_numrows($ris_37);
        $per_37= ($num_37*100)/$num;
        $ar_37= round ($per_37,1);


$sql_38= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>1 AND `Hmax`<=2 ) AND
(`Dmt`>=247.5 AND `Dmt`< 292.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_38=mysql_query ($sql_38);
        $num_38=mysql_numrows($ris_38);
        $per_38= ($num_38*100)/$num;
        $ar_38 = round ($per_38,1);


$sql_39= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>2 AND `Hmax`<=4 ) AND
(`Dmt`>=247.5 AND `Dmt`< 292.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_39=mysql_query ($sql_39);
        $num_39=mysql_numrows($ris_39);
        $per_39= ($num_39*100)/$num;
        $ar_39= round ($per_39,1);
```

```
$sql_40= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>4 AND `Hmax`<=6 ) AND
(`Dmt`>=247.5 AND `Dmt`< 292.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_40=mysql_query ($sql_40);
        $num_40=mysql_numrows($ris_40);
        $per_40= ($num_40*100)/$num;
        $ar_40= round ($per_40,1);

$sql_41= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>6) AND (`Dmt`>=247.5 AND
`Dmt`< 292.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_41=mysql_query ($sql_41);
        $num_41=mysql_numrows($ris_41);
        $per_41= ($num_41*100)/$num;
        $ar_41= round ($per_41,1);

//SELEZIONE E DIVISIONE PER CLASSI DI Hmax CONSIDERANDO VALORI PER Dmt per NW

$sql_42= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0 AND `Hmax`<0.3 ) AND
(`Dmt`>=292.5 AND `Dmt`< 337.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_42=mysql_query ($sql_42);
        $num_42=mysql_numrows($ris_42);
        $per_42= ($num_42*100)/$num;
        $ar_42= round ($per_42,1);

$sql_43= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>=0.3 AND `Hmax`<=1 ) AND
(`Dmt`>=292.5 AND `Dmt`< 337.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_43=mysql_query ($sql_43);
        $num_43=mysql_numrows($ris_43);
        $per_43= ($num_43*100)/$num;
        $ar_43= round ($per_43,1);


$sql_44= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>1 AND `Hmax`<=2 ) AND
(`Dmt`>=292.5 AND `Dmt`< 337.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_44=mysql_query ($sql_44);
        $num_44=mysql_numrows($ris_44);
        $per_44= ($num_44*100)/$num;
        $ar_44 = round ($per_44,1);


$sql_45= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>2 AND `Hmax`<=4 ) AND
(`Dmt`>=292.5 AND `Dmt`< 337.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_45=mysql_query ($sql_45);
        $num_45=mysql_numrows($ris_45);
        $per_45= ($num_45*100)/$num;
        $ar_45= round ($per_45,1);


$sql_46= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>4 AND `Hmax`<=6 ) AND
(`Dmt`>=292.5 AND `Dmt`< 337.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_46=mysql_query ($sql_46);
        $num_46=mysql_numrows($ris_46);
        $per_46= ($num_46*100)/$num;
        $ar_46= round ($per_46,1);
```

```
$sql_47= "SELECT `Hmax`, `Dmt` FROM `archivio` WHERE (`Hmax`>6) AND (`Dmt`>=292.5 AND
`Dmt`< 337.5) AND (`data_reg` BETWEEN '$data_reg_i' AND '$data_reg_f')";
        $ris_47=mysql_query ($sql_47);
        $num_47=mysql_numrows($ris_47);
        $per_47= ($num_47*100)/$num;
        $ar_47= round ($per_47,1);



//ELABORAZIONE DEL GRAFICO

require_once ('jpgraph.php');
require_once ('jpgraph_windrose.php');



// Data can be specified using both ordinal index of the axis
// as well as the direction label
$data = array
        (
// lo 0 rappresenta l'EST e la numerazione è ordinata in senso anti-orario

  'N' => array($ar_0,$ar_1,$ar_2,$ar_3,$ar_4,$ar_5),
  'NE'=> array ($ar_6,$ar_7,$ar_8,$ar_9,$ar_10,$ar_11),
        'E'=> array ($ar_12,$ar_13,$ar_14,$ar_15,$ar_16,$ar_17),
        'SE' => array ($ar_18,$ar_19,$ar_20,$ar_21,$ar_22,$ar_23),
        'S' => array ($ar_24,$ar_25,$ar_26,$ar_27,$ar_28,$ar_29),
        'SW' => array ($ar_30,$ar_31,$ar_32,$ar_33,$ar_34,$ar_35),
        'W' => array ($ar_36,$ar_37,$ar_38,$ar_39,$ar_40,$ar_41),
        'NW' => array ($ar_42,$ar_43,$ar_44,$ar_45,$ar_46,$ar_47)
        );

//Viene elaborato il grafico
$graph = new WindroseGraph(500,500);
$graph->title->Set("Rosa altezze onda dal $data_reg_i a $data_reg_f");

//i dati vengono plottati
$wp = new WindrosePlot($data);
$wp->SetType(WINDROSE_TYPE8);        //Tipologia di grafico a 8 punti cardinali

//Manda a schermo la legenda

$legendtxt = ('(m)');
 $wp ->legend->SetText($legendtxt);

// Invio al browser
$graph->Add($wp);
$graph->Stroke();

 mysql_close();

?>
```