

# **Progetto di Basi di dati II**

Studente: Scaraggi Davide, matricola 661635

Anno accademico: 2017/2018

## Sommario

Raccolta dei requisiti .....	4
Specifiche .....	4
Progettazione concettuale .....	4
Scelta del corretto livello di astrazione .....	4
Individuazione di omonimi e sinonimi .....	4
Costruzione del glossario dei termini .....	5
Riorganizzazione delle frasi per concetti.....	5
Progettazione del modello ER .....	5
Entità Pagina e Termine .....	6
Entità Utente, Sessione, Pagina e Termine .....	6
Schema integrato .....	7
Progettazione logica .....	8
Tavola dei volumi .....	8
Analisi delle ridondanze .....	9
Non sono presenti ridondanze. ....	9
Eliminazione delle generalizzazioni .....	9
Tavola degli accessi .....	9
Operazione 1 .....	9
Operazione 2 .....	9
Operazione 3 .....	9
Operazione 4 .....	10
Operazione 5 .....	10
Partizionamento ed accorpamento di concetti.....	10
Scelta degli identificatori principali .....	10
Schema logico.....	11
Tabella amministratore .....	11
Tabella pagina.....	11
Tabella riferimento.....	12
Tabella sessione.....	13
Tabella sessionelettura.....	14
Tabella termine .....	15
Tabella utente .....	15
Progettazione fisica .....	16

Trigger .....	16
Applicazione web .....	18
Grafica dell'applicazione web .....	19
Data Warehouse.....	22
Identificazione dei concetti base per l'analisi dimensionale.....	22
Identificazione delle dimensioni .....	22
Ristrutturazione dello schema ER .....	22

## Raccolta dei requisiti

### Specifiche

Una compagnia che gestisce un piccolo motore di ricerca sul web vuole usare una base di dati per tenere traccia della struttura delle pagine indicizzate, e delle sessioni che gli utenti hanno con il motore di ricerca. Per quanto riguarda la struttura delle pagine indicizzate, interessa conoscere, per ogni pagina, l'URL, il titolo, l'insieme delle pagine puntate dai riferimenti (link) che appaiono nella pagina, l'insieme dei termini che vi appaiono e, per ogni termine, al numero di volte in cui il termine appare nella pagina. Le sessioni sono così strutturate: l'utente presenta un termine di ricerca, il sistema segnala tutte le pagine correlate, e l'utente legge alcune di queste pagine; specifichiamo ora a quali informazioni la compagnia è interessata, per quanto riguarda le sessioni. Per ogni sessione, identificata da un codice, l'organizzazione è interessata al giorno in cui si è svolta. La sessione può essere eseguita da un utente anonimo, sul quale il sistema non ha informazioni, oppure da un utente registrato, del quale il sistema conosce username, password, indirizzo e-mail, e la lista delle pagine effettivamente lette dall'utente.

### Progettazione concettuale

Per poter realizzare un modello ER che sia rispondente alle richieste del committente è necessario effettuare un'analisi delle specifiche tale da poter individuare eventuali ambiguità o incoerenze e cercare di riorganizzarle in modo che risultino più chiare. A tal fine si procede nell'effettuare le seguenti operazioni:

- Scelta del corretto livello di astrazione
- Individuazione di omonimi e sinonimi
- Costruzione del glossario dei termini
- Riorganizzazione delle frasi per concetti

#### Scelta del corretto livello di astrazione

I seguenti termini sono stati modificati in quanto risultavano troppo astratti o troppo specifici:

- Pagina indicizzata ➔ Pagina
- Termine di ricerca ➔ Termine

#### Individuazione di omonimi e sinonimi

Nelle specifiche non sono presenti omonimi e/o sinonimi.

## Costruzione del glossario dei termini

Nome	Descrizione	Sinonimi	Collegamenti
Pagina	Insieme di tutte le componenti di una pagina web. Per semplicità si suppone che la pagina sia costituita da: URL, titolo, l'insieme di link ad altre pagine, insieme di termini.	/	Termine, Utente, Sessione
Termine	Parola che compare all'interno di una pagina	/	Pagina, Sessione, Utente
Utente	Persona che utilizza il sistema	/	Pagina, Sessione, Termine
Sessione	Arco temporale che comincia nel momento in cui l'utente effettua una ricerca e termina quando ne inizia un'altra.	/	Utente, Termine, Pagina

## Riorganizzazione delle frasi per concetti

Frasi riguardanti le pagine:

Per quanto riguarda la struttura delle pagine indicizzate, interessa conoscere, per ogni pagina, l'URL, il titolo, l'insieme delle pagine puntate dai riferimenti (link) che appaiono nella pagina, l'insieme dei termini che vi appaiono e, per ogni termine, al numero di volte in cui il termine appare nella pagina

Frasi riguardanti le sessioni:

Le sessioni sono così strutturate: l'utente presenta un termine di ricerca, il sistema segnala tutte le pagine correlate\*, e l'utente legge alcune di queste pagine. Per ogni sessione, identificata da un codice, l'organizzazione è interessata al giorno in cui si è svolta. La sessione può essere eseguita da un utente anonimo, oppure da un utente registrato.

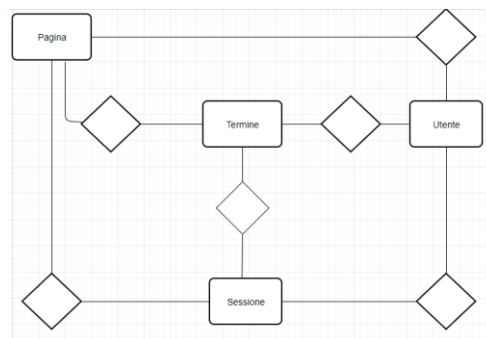
Frasi riguardanti gli utenti:

La sessione può essere eseguita da un utente anonimo, sul quale il sistema non ha informazioni\*<sup>1</sup>, oppure da un utente registrato, del quale il sistema conosce username, password, indirizzo e-mail, e la lista delle pagine effettivamente lette dall'utente.

## Progettazione del modello ER

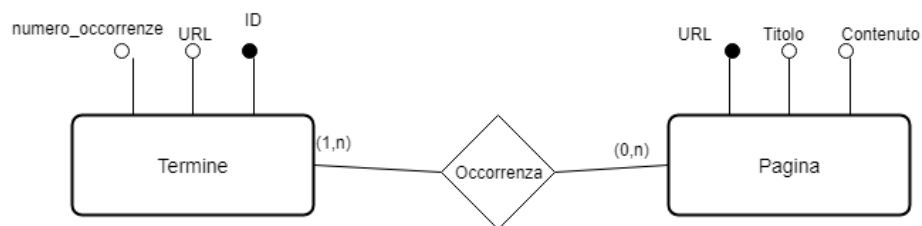
La progettazione concettuale è stata eseguita utilizzando l'approccio ibrido. Quindi, in fase iniziale è stato steso uno schema scheletro in cui sono stati rappresentati i concetti principali. Tali

concetti sono stati successivamente raffinati, realizzando degli schemi specifici. Infine, gli schemi specifici sono stati integrati ottenendo uno schema finale completo.



### Entità Pagina e Termine

Per ogni termine si deve tener conto del numero di occorrenze di ciascun termine all'interno di ciascuna pagina, pertanto si è scelto di inserire un attributo URL all'interno di Termine. Altra soluzione possibile sarebbe stata la creazione di un'opportuna entità debole Occorrenza.

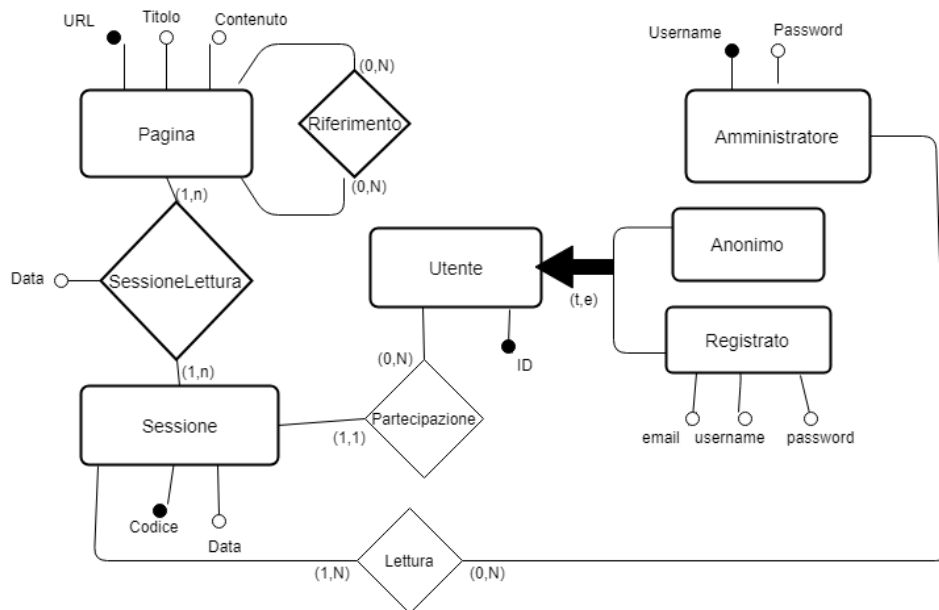


### Entità Utente, Sessione, Pagina e Termine

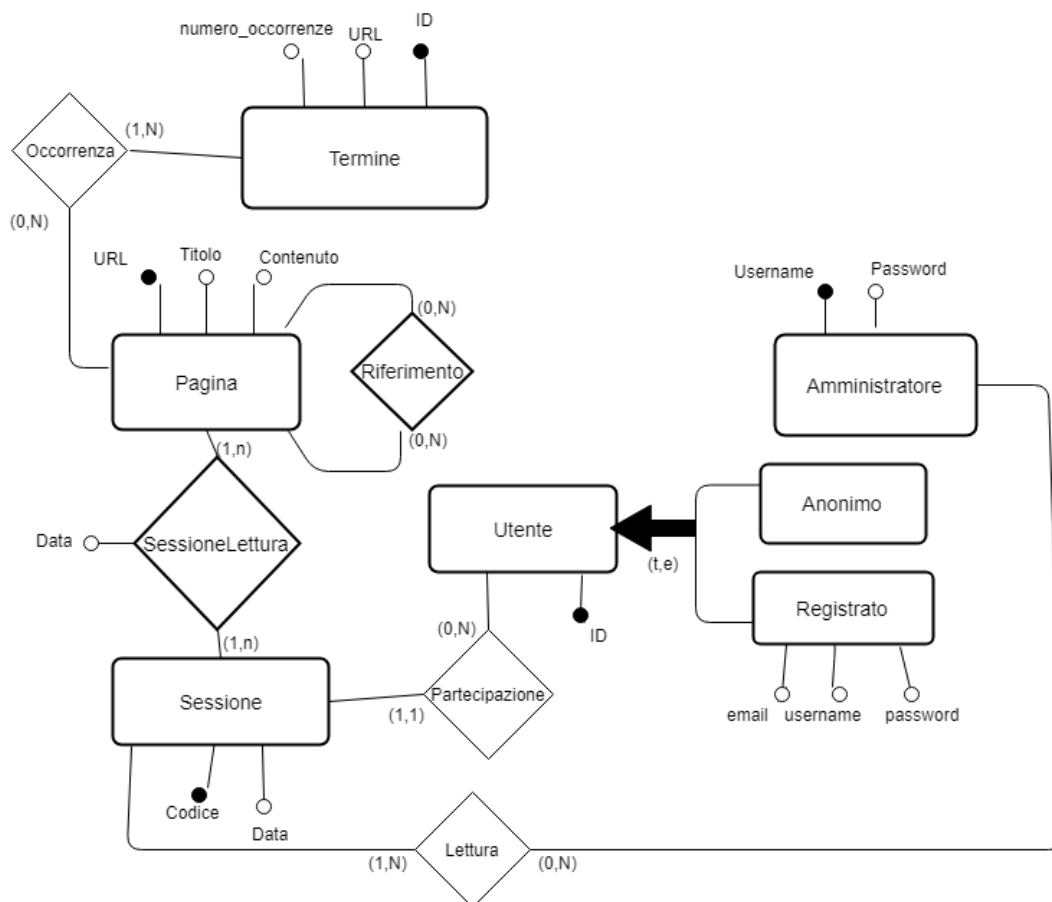
L'utente si distingue tra utente registrato e utente anonimo, dunque si è ritenuto opportuno utilizzare una generalizzazione per distinguere le tipologie di utente. Si è scelto di non inserire una relazione diretta tra utente e pagina, poiché si vuole rappresentare il fatto che una lettura può avvenire sempre e solo nell'ambito di una sessione. Stesso discorso può essere fatto per il termine ricercato da un utente, che verrà contestualizzato sempre nell'ambito di una sessione. Si noti che la data afferente una sessione e quella afferente una lettura sono concetti diversi, questo perché nell'ambito di una sessione possono essere lette più pagine.

\* Per semplicità, per pagine correlate si intenderanno tutte e sole le pagine contenenti il termine ricercato.

\*<sup>1</sup> Al fine di garantire la tracciabilità degli utenti, si ritiene opportuno assegnare un ID anche agli utenti anonimi



## Schema integrato



## Progettazione logica

Con la seguente fase si procede con l'eventuale ristrutturazione del modello ER realizzato nella fase precedente effettuando delle considerazioni che si basano sulla tavola dei volumi e sulla tavola degli accessi delle seguenti operazioni:

1. Registrare un nuovo utente (50 volte a settimana)
2. Aggiungere una nuova pagina letta (100 volte al minuto)
3. Stampa delle pagine correlate ad un termine (30 volte al minuto)
4. Stampa del numero di occorrenze di un termine in una pagina (100 volte a settimana)
5. Stampa delle pagine lette da un utente (100 volte a settimana)

### Tavola dei volumi

Per poter realizzare una tavola dei volumi vengono fatte delle assunzioni:

- Il sistema è stato usato da 100.000 utenti, di cui una metà sono registrati e l'altra metà non lo sono
- Facendo una media, ogni utente ha partecipato a 100 sessioni di utilizzo del motore di ricerca
- Ogni termine occorre in una pagina la media di 5 volte
- Vi sono 1 milione di pagine e che in ogni pagina vi sono 100 termini diversi fra loro, quindi si hanno 100.000.000 occorrenze. Escludendo i termini ripetuti in pagine diverse, si può assumere che vi siano un totale di 100.000 termini
- Ogni utente in una sessione legge mediamente 4 pagine, quindi considerando 4 pagine per 100.000 utenti per 100 sessioni, in totale vi sono 40.000.000 letture da parte di utenti
- Ogni pagina contiene in media 5 link, quindi vi sono 5.000.000 di riferimenti

Concetto	Tipo	Volume
Utente	E	100.000
Amministratore	E	50
Sessione	E	10.000.000
Registrato	E	50.000
Anonimo	E	50.000
Termine	E	30.000.000
Pagina	E	1.000.000
Occorrenza	E	100.000.000
Partecipazione	R	10.000.000
SessioneLettura	R	400.000
Inserimento	R	10.000.000
Riferimento	R	5.000.000



## Analisi delle ridondanze

Non sono presenti ridondanze.

## Eliminazione delle generalizzazioni

La generalizzazione riguardante le entità Utente, Utente\_Registrato e Utente\_Anonimo può essere trasformata accorpendo le entità figlie nell'entità padre. Si è scelto per questo tipo di trasformazione, seppure potrebbe produrre molti valori nulli, poiché l'unica operazione in cui si differenziano le due entità è quella di registrazione. Gli attributi Username, Email e Password saranno accorpati in Utente.

## Tavola degli accessi

### Operazione 1

Nome	Concetto	Accessi	Tipo
Utente	E	1	S

Costo dell'operazione: considerando doppio il costo di scrittura rispetto a quello di lettura e frequenza 50 (a settimana), il costo è  $50 * 2 = 100$  accessi a settimana.

### Operazione 2

Costo dell'operazione: considerando la frequenza di 100 volte al minuto e un costo della scrittura doppio rispetto a quello per la lettura, avremo:  $(3 + 2 * 3) * 100 = 900$  accessi al minuto.

### Tavola degli accessi in assenza di ridondanza

Nome	Concetto	Accessi	Tipo
Utente	E	1	L
Partecipazione	R	1	L
Sessione	E	1	L
SessioneLettura	R	1	S
Pagina	E	1	S

Costo dell'operazione: considerando la frequenza di 100 volte al minuto e un costo della scrittura doppio rispetto a quello per la lettura, avremo:  $(3 + 2 * 2) * 100 = 700$  accessi al minuto.

### Operazione 3

Nome	Concetto	Accessi	Tipo
Termine	E	1	L
Occorrenza	E	1	L
Pagina	E	1	L

Costo dell'operazione: considerata la frequenza di 30 volte al minuto, avremo 90 accessi al minuto.

#### Operazione 4

Nome	Concetto	Accessi	Tipo
Termine	E	1	L
Occorrenza	E	1	L
Pagina	E	1	L

Costo dell'operazione: considerata la frequenza di 100 volte a settimana, avremo 300 accessi a settimana.

#### Operazione 5

##### Tavola degli accessi in presenza di ridondanza

Nome	Concetto	Accessi	Tipo
Utente	E	1	L
Partecipazione	R	1	L
Sessione	E	1	L
SessioneLettura	R	1	L
Pagina	E	1	L

Costo dell'operazione: considerata la frequenza di 100 accessi a settimana, avremo 300 accessi a settimana

#### Partizionamento ed accorpamento di concetti

Nessun partizionamento/accorpamento necessario.

#### Scelta degli identificatori principali

Ogni entità è stata già progettata con dei propri identificatori.

## Schema logico

### Tabella amministratore

```
CREATE TABLE public.amministratore
(
    username character varying(30) NOT NULL,
    password character varying(30),
    CONSTRAINT amministratore_pkey PRIMARY KEY (username)
)
WITH (
    OIDS=FALSE
);
ALTER TABLE public.amministratore
    OWNER TO postgres;
```

### Tabella pagina

```
CREATE TABLE public.pagina
(
    url text NOT NULL,
    titolo text,
    contenuto text,
    CONSTRAINT pagina_pk PRIMARY KEY (url)
)
WITH (
    OIDS=FALSE
);
ALTER TABLE public.pagina
    OWNER TO postgres;
```

```

-- Trigger: added_page on public.pagina

-- DROP TRIGGER added_page ON public.pagina;

CREATE TRIGGER added_page
  AFTER INSERT
  ON public.pagina
  FOR EACH ROW
  EXECUTE PROCEDURE public.paginaaggiunta();

```

## Tabella riferimento

```

CREATE TABLE public.riferimento
(
  pagina text,
  pagina_puntata text,
  id integer NOT NULL DEFAULT nextval('riferimento_id_seq'::regclass),
  CONSTRAINT riferimento_pkey PRIMARY KEY (id),
  CONSTRAINT riferimento_fk FOREIGN KEY (pagina)
    REFERENCES public.pagina (url) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION
)
WITH (
  OIDS=FALSE
);
ALTER TABLE public.riferimento
  OWNER TO postgres;

-- Index: public.fki_ddd

-- DROP INDEX public.fki_ddd;

```

```
CREATE INDEX fki_ddd
ON public.riferimento
USING btree
(pagina COLLATE pg_catalog."default");
```

## Tabella sessione

```
CREATE TABLE public.sessione
(
    codice integer NOT NULL DEFAULT nextval('sessione_codice_seq'::regclass),
    utente integer NOT NULL DEFAULT nextval('sessione_utente_seq'::regclass),
    data timestamp with time zone,
    termine character varying(20),
    CONSTRAINT sessione_pk PRIMARY KEY (codice),
    CONSTRAINT sessione_fk FOREIGN KEY (utente)
        REFERENCES public.utente (id) MATCH SIMPLE
        ON UPDATE NO ACTION ON DELETE NO ACTION
)
WITH (
    OIDS=FALSE
);
ALTER TABLE public.sessione
    OWNER TO postgres;

-- Index: public.fki_sessione_fk

-- DROP INDEX public.fki_sessione_fk;

CREATE INDEX fki_sessione_fk
ON public.sessione
USING btree
(utente);
```

## Tabella sessionelettura

```
CREATE TABLE public.sessionelettura
(
  sessione integer NOT NULL,
  url text NOT NULL,
  data timestamp with time zone NOT NULL,
  CONSTRAINT sessionelettura_pkey PRIMARY KEY (sessione, url, data),
  CONSTRAINT sessione_fk FOREIGN KEY (sessione)
    REFERENCES public.sessione (codice) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT url_fk FOREIGN KEY (url)
    REFERENCES public.pagina (url) MATCH SIMPLE
    ON UPDATE NO ACTION ON DELETE NO ACTION
)
WITH (
  OIDS=FALSE
);
ALTER TABLE public.sessionelettura
  OWNER TO postgres;

-- Index: public.fki_url_fk

-- DROP INDEX public.fki_url_fk;

CREATE INDEX fki_url_fk
  ON public.sessionelettura
  USING btree
  (url COLLATE pg_catalog."default");
```

## Tabella termine

```
CREATE TABLE public.termine
(
  numero_occorrenze integer,
  url text NOT NULL,
  id character varying(20) NOT NULL,
  CONSTRAINT termine_pk PRIMARY KEY (url, id)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE public.termine
  OWNER TO postgres;
```

## Tabella utente

```
CREATE TABLE public.utente
(
  id integer NOT NULL DEFAULT nextval('utente_id_seq'::regclass),
  email character varying(50),
  username character varying(50),
  password character varying(50),
  CONSTRAINT utete_pk PRIMARY KEY (id),
  CONSTRAINT unique_username UNIQUE (username)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE public.utente
```

## Progettazione fisica

In PostgreSQL la struttura hash messa a disposizione dalla chiave primaria garantisce un accesso puntuale efficiente. Questa caratteristica viene fortemente sfruttata in quanto gli accessi puntuali vengono effettuati su campi chiave.

## Trigger

Nella realizzazione del database non sono stati previsti molti trigger, poiché si è ritenuto di poter proteggere la base di dati da operazioni potenzialmente dannose attraverso l'utilizzo delle chiavi e nel caso dell'username definendolo come UNIQUE. Anche per quanto riguarda le funzioni per il popolamento, queste non si sono dimostrate necessarie, in quanto il popolamento è avvenuto attraverso dei file csv caricati con l'uso di PGAdmin. L'unico caso in cui si è ritenuto opportuno definire un trigger ed una funzione è stato quello della memorizzazione dei termini, infatti era necessario estrarre singolarmente le parole dai contenuti in cui erano presenti e quindi inserirle nell'apposita tabella.

```
CREATE OR REPLACE FUNCTION public.urlinesistente()  
    RETURNS trigger AS  
$BODY$  
DECLARE  
i INTEGER := 1;  
  
BEGIN  
  
    IF(SELECT  
        COUNT(*) = 0  
        FROM pagina  
        WHERE pagina.url=NEW.url)  
  
    THEN  
        RAISE EXCEPTION 'Non esiste una pagina con questo url';
```



```

END IF;

RETURN NEW;

END;

$BODY$

LANGUAGE plpgsql VOLATILE

COST 100;

ALTER FUNCTION public.urlinesistente()

OWNER TO postgres;

```

```

CREATE OR REPLACE FUNCTION public.paginaaggiunta()

RETURNS trigger AS

$BODY$

DECLARE

i INTEGER := 1;

termini CHARACTER VARYING[];

BEGIN

termini = string_to_array(NEW.contenuto, ' ');

WHILE(i<array_length(termini, 1))

LOOP

IF(SELECT

COUNT(*) > 0

FROM termine

WHERE termine.url=NEW.url AND termini[i]=termine.id)

THEN

UPDATE termine SET numero_occorrenze= numero_occorrenze+1

WHERE termini[i] = termine.id AND termine.url=NEW.url;

i = i+1;

```

```
ELSE

INSERT INTO termine(id,url,numero_occorrenze)
VALUES(termini[i],NEW.url,1);

END IF;

END LOOP;


RETURN NEW;

END;
$body$

LANGUAGE plpgsql VOLATILE

COST 100;

ALTER FUNCTION public.paginaaggiunta()

OWNER TO postgres;
```

## Applicazione web

L'applicazione web è stata realizzata in modo da poter garantire la riusabilità di varie componenti e facilitarne la manutenibilità. La web app si poggia su una struttura che consente di disaccoppiare le componenti, distribuendo l'applicazione su 3 livelli (MVC). La richiesta viene inviata dal browser, attraverso una JSP specifica, alla Servlet responsabile dell'entità su cui si sta lavorando. La Servlet prende in carico la richiesta all'interno di un handler (doGet, doPost) e a seconda dei parametri ricevuti chiama il metodo specifico. Il metodo chiamato racchiude la logica di business e per realizzare i propri compiti si interfaccia con la base di dati PostgreSQL utilizzando il Data Access Object.

Ciascuna entità del contesto applicativo è stata implementata con una classe apposita (Entity). Le classi Entity vengono utilizzate per poter realizzare il passaggio dei dati attraverso i 3 livelli implementati. Le classi Entity realizzate sono le seguenti:

- Pagina
- Sessione
- SessioneLettura
- Termine
- Utente

Le Servlet sono state implementate cercando di racchiudere in ciascuna di esse le operazioni relative ad una specifica entità del contesto applicativo. Le Servlet implementate sono le seguenti:

- AmministratoreServlet: si occupa della gestione delle operazioni legate all'attività dell'amministratore.
- PaginaServlet: si occupa della gestione delle pagine.
- TermineServlet: si occupa della gestione delle operazioni svolte sui termini, come estrazione dati o ricerca.
- UtenteServlet: si occupa di gestire le attività degli utenti dal login, fino alle varie operazioni che possono svolgere nel contesto del sistema.

#### Grafica dell'applicazione web

L'interfaccia grafica è stata implementata utilizzando il framework Bootstrap, realizzando delle JSP quanto più possibile semplici ed intuitive. È stata riposta molta attenzione nella strutturazione dell'interfaccia, realizzando header, footer e main in modo che fossero riusabili.

Le JSP implementate sono le seguenti:

- Cronologia.jsp
- Footer.jsp
- Header.jsp
- HomeAdmin.jsp
- HomeAnonimo.jsp
- HomeUtente.jsp
- Index.jsp
- LetturaPagina.jsp
- Login.jsp
- LoginAmministratore.jsp
- MenuAdmin.jsp
- MenuAnonimo.jsp
- MenuUtente.jsp
- PresentazioneRisultati.jsp
- Registrazione.jsp
- VisualizzaDati.jsp

Di seguito vengono riportati alcuni screenshot dell'interfaccia.

## Registrazione

Inserisci i tuoi dati

**E-mail**

**Username**

**Password**

**Inserisci nuovamente la password**

Benvenuto!

Non sei ancora registrato? [Registrati](#)

oppure

[Procedi senza login](#)

[Accedi come amministratore](#)

Benvenuto nella modalità amministratore!







[Torna indietro](#)


[Home](#)[Logout](#)

## Home

Sessioni



Show  entriesSearch: 

Codice sessione 	Utente 	Data sessione 	Termine cercato 	Pagina letta 	Data lettura pagina 
20	1002	10.09.2018 23.28	ac	Nessuna pagina letta.	/
21	1002	10.09.2018 23.35	et	Nessuna pagina letta.	/
22	1002	11.09.2018 14.35	ac	Nessuna pagina letta.	/
23	1002	11.09.2018 15.07	ac	Nessuna pagina letta.	/

[Home](#)[Cronologia ricerche](#) [Visualizza](#)[I miei dati](#) [Logout](#)

## Cronologia

Show  entriesSearch: 

Uri 	Data 
http://bbc.co.uk/suspendisse/omare.json	12.09.2018 16.59
http://ox.ac.uk/pede/ullamcorper/augue/a/suscipit/nulla/elit.png	12.09.2018 10.18
https://dailymotion.com/sem.aspx	12.09.2018 09.19
https://dailymotion.com/sem.aspx	12.09.2018 09.27
https://dailymotion.com/sem.aspx	12.09.2018 09.29
https://dailymotion.com/sem.aspx	12.09.2018 09.31

[Home](#)[Vai al login](#)

## Home

Inserisci un termine da ricercare

## Data Warehouse

Purtroppo, a causa delle tempistiche ridotte, si è potuto procedere solo ad una progettazione del Data Warehouse, piuttosto che alla vera e propria implementazione. I risultati dell'analisi condotta vengono riportati di seguito.

### Identificazione dei concetti base per l'analisi dimensionale

Il fatto può essere identificato nell'entità Termine in quanto un possibile obiettivo di analisi potrebbe essere quello di capire la diffusione di un certo termine. Questo potrebbe avvenire per diverse ragioni, a partire da studi di linguistica fino ad usi commerciali nell'ambito del marketing. La misura del fatto che sembra logico considerare è il numero di occorrenze.

### Identificazione delle dimensioni

La fase successiva è quella di individuazione delle dimensioni d'analisi, navigando nello schema ed individuando concetti che consentono di raggruppare istanze di fatti. Le dimensioni individuate sono le seguenti:

- Tempo: questa dimensione è fondamentale per poter raggruppare l'uso dei termini in degli archi di tempo. In tal modo. Si potrà dunque capire, ad esempio, quante volte una parola è stata utilizzata in un determinato periodo.
- Utente: anche grazie a quando memorizzato nella base di dati, ma non sfruttato attualmente dall'applicazione, sarà possibile capire quali sono i trend di ricerca dei termini da parte degli utenti.
- Pagine: con questa terza dimensione i fatti possono esser raggruppati in base a dove vengono utilizzati.

### Ristrutturazione dello schema ER

Innanzitutto, va detto che vi sono diverse alternative di ristrutturazione, ma per gli scopi trattati precedentemente l'ideale sarebbe necessario poter collegare l'entità *Pagina* ad una *Tipologia*. In questo modo sarebbe possibile effettuare della analisi più sensate a riguardo dei termini, potendoli collocare in base ai diversi contesti d'uso. Un secondo intervento di ristrutturazione utile è l'inserimento dell'entità *Fascia d'età*, in modo da poter raggruppare le istanze di fatti in base alla fascia di età a cui appartengono gli utenti del motore di ricerca, tuttavia, per far ciò sarebbe necessario richiede la data di nascita agli utenti, in fase di registrazione. Il risultato di queste ristrutturazioni è uno schema a fiocco di neve riportato di seguito.

