



UIIP Project Work - Sistema Editoriale The Yellow Daily

**Documento di architettura comprensivo di modello
dati e tecnologie scelte**

Gruppo Giallo

Andrea Castaldo
Antonio Maria Fonzo
Ivan Salinaro
Armando Soriano
Cinzia Tito

07/05/2013



Storia del documento

Data	Versione	Descrizione	Autore
11 Aprile 2013	1.0	Prima stesura.	Armando Soriano
15 Aprile 2013	1.1	Seconda stesura: baseline 1.	Armando Soriano
22 Aprile 2013	1.2	Terza stesura: baseline 2.	Armando Soriano
3 Maggio 2013	1.3	Quarta stesura: baseline 3.	Armando Soriano
7 Maggio 2013	1.4	Quinta stesura: baseline 4.	Armando Soriano

Membri del team "Gruppo Giallo"

Nominativo
Andrea Castaldo
Antonio Maria Fonzo
Ivan Salinaro
Armando Soriano
Cinzia Tito

Documenti di riferimento

RIF.	Codice	Data	Titolo
			UIIP Project Work - Sistema Editoriale

Allegati al documento






Fornito da.	Codice documento	Data	Titolo





Abbreviazioni e Acronimi

CRUD	Create, Read, Update, Delete (Creazione, Visualizzazione, Modifica, Cancellazione)
UI	User Interface
BL	Business Logic
EIS	Enterprise Information Service
DTO	Data Transfer Object
DAO	Data Access Object
WS	Web Service
DB	DataBase
DBMS	DataBase Management System
RDBMS	Relational DataBase Management System
MVC	Model View Controller
JNDI	Java Naming and Directory Interface
PL/SQL	Procedural Language/Structured Query Language
E-R	Entity-Relationship (Entità-Relazione)
JSP	JavaServer Pages
JSTL	JavaServer Pages Standard Tag Library

Indice dei contenuti

INDICE DELLE FIGURE	6
1 INTRODUZIONE.....	7
1.1 SCOPO DEL DOCUMENTO	7
1.2 SCOPO DEL PROGETTO	7
2 ARCHITETTURA LOGICA	8
2.1 INTRODUZIONE	8
2.2 ARCHITETTURA LOGICA IN DETTAGLIO	10
2.2.1 <i>Presentation Layer - User Interface</i>	11
2.2.2 <i>Business Logic Layer</i>	11
2.2.3 <i>Resource Management Layer – Enterprise Information Service</i>	11
2.2.4 <i>Data Transfer Object</i>	11
2.2.5 <i>Receiver e Transmitter</i>	12
3 ARCHITETTURA FISICA	13
3.1 INTRODUZIONE	13
3.2 ARCHITETTURA FISICA IN DETTAGLIO	13
3.3 DEPLOYMENT DIAGRAM	14
4 MODELLO DEI DATI.....	16
4.1 INTRODUZIONE	16
4.1.1 <i>Modello ER</i>	16
4.1.2 <i>Modello Logico</i>	17
4.1.3 <i>Modello Fisico</i>	18
4.2 DESCRIZIONE CAMPI E ASSUNZIONI.....	24
4.2.1 <i>Account</i>	24
4.2.1 <i>Appartenenza_Gruppo</i>	25
4.2.2 <i>Funzionalità</i>	25
4.2.3 <i>Gestione_Notizia</i>	26
4.2.4 <i>Gruppo</i>	26
4.2.5 <i>Notizia</i>	27
4.3 SCRIPT SQL.....	28
4.4 TAVOLA DEI VOLUMI	29
5 TECNOLOGIE UTILIZZATE	30
5.1  SPRING MVC	30
5.2  APACHE TOMCAT	31
5.3  APACHE AXIS2	31
5.4  APACHE LOG4J	31
5.5  FRAMEWORK QUARTZ	32



5.6	JAVAMAIL		32
5.7	ORACLE 11G		32



Indice delle figure

Figura 1 - Architettura three-layer	8
Figura 2 - Architettura Logica del sistema.....	10
Figura 3 - Architettura Fisica	13
Figura 4 - Deployment diagram	14
Figura 5 - Modello ER	16
Figura 6 - Screenshot Account.....	18
Figura 7 - Screenshot Appartenenza_Gruppo.....	19
Figura 8 - Screenshot Funzionalita	20
Figura 9 - Screenshot Gestione_Notizia	21
Figura 10 - Screenshot Gruppo	22
Figura 11 - Screenshot Notizia	23



1 Introduzione

1.1 Scopo del documento

Lo scopo del presente documento è quello di evidenziare le scelte progettuali effettuate nell'ambito del project work "Sistema Editoriale" (denominato **The Yellow Daily**) per quanto riguarda l'architettura (logica e fisica), il modello dei dati e le tecnologie utilizzate.

1.2 Scopo del progetto

Lo scopo del progetto è la realizzazione di un sistema editoriale in grado di gestire il ciclo di vita delle notizie per una redazione giornalistica di una agenzia di stampa il cui business primario è la fornitura di news verso clienti esterni (giornali, tv, portali, ecc.).

Il sistema è multiutente ed accessibile via web (http) mediante l'uso di un browser (Internet Explorer, Chrome, Firefox), quindi senza dover installare nessun software aggiuntivo sui PC dei giornalisti.

Il sistema permette l'accesso solo a utenti autorizzati e appartenenti alla redazione. Gli accessi sono controllati e vincolati all'inserimento di login e password.

Il sistema prevede, inizialmente, due tipologie di utenti:

1. Amministratore;
2. Giornalista.

L'utente **Amministratore** ha la facoltà di gestire gli account del sistema, in particolare può effettuare operazioni CRUD (Creazione, Visualizzazione, Modifica, Cancellazione).

L'utente **Giornalista** ha, invece, la facoltà di gestire le notizie (creazione, modifica, versionamento, visualizzazione, ricerca, cancellazione, archiviazione, ecc.).

Il sistema, infine, permette di poter ricevere notizie in modalità automatica (batch) da contributori esterni e poterle successivamente inviare verso i sistemi dei clienti.



2 Architettura Logica

2.1 Introduzione

La presente sezione del documento illustra l'architettura logica del sistema di editoria *The Yellow Daily*.

A livello concettuale il sistema è progettato in termini di tre diversi componenti funzionali (layer):

1. Presentazione (Presentation Layer);
2. Logica dell'applicazione (Application Logic Layer);
3. Gestione delle risorse (Resource Management Layer).

L'architettura così composta è denominata **Three-Layer** (Figura 1 - Architettura three-layer)

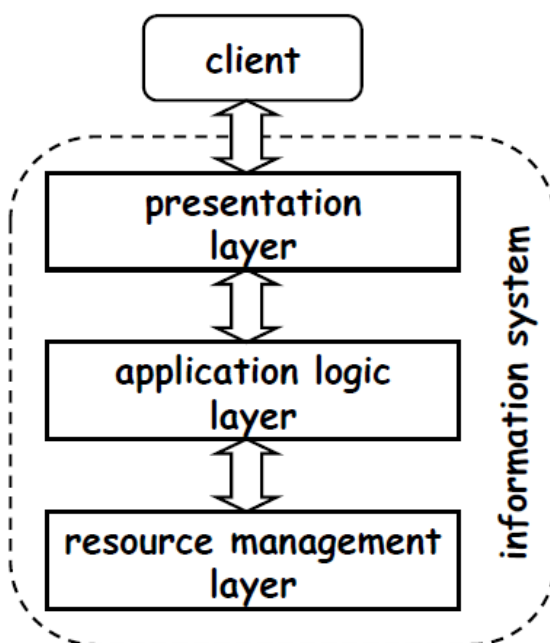


Figura 1 - Architettura three-layer

Un **Client** è un qualsiasi utente che vuole effettuare un'operazione sul sistema. I Client interagiscono con il sistema attraverso il Presentation Layer.

Il **Presentation Layer** è il livello del sistema che gestisce la comunicazione con le entità esterne al sistema stesso (client). Comprende le componenti che si occupano di presentare l'informazione verso i client, e che consentono ai client di interagire con il sistema per sottomettere operazioni ed



ottenere risultati. I client possono essere completamente esterni ed indipendenti dal presentation layer. Nel nostro caso il presentation layer è costituito dai moduli del web server che concorrono a creare i documenti HTML (ad es. Java Servlet), mentre il browser è il client.

L'**Application Logic Layer** è il livello del sistema che si occupa del processamento dei dati necessario per produrre i risultati da inoltrare al livello di presentazione. Il livello della logica applicativa è anche chiamato processo di business, insieme delle regole di business, o semplicemente server (in questi casi il sottostante livello è rispettivamente chiamato persistence storage, business objects, o database).

Il **Resource Management Layer** è il livello che gestisce i dati che sono necessari al funzionamento dell'intero sistema. I dati possono risiedere su una base dati, un file system, o altri contenitori di informazioni. In linea di principio, il livello di gestione delle risorse gestisce le differenti sorgenti di dati che fanno parte del sistema informativo, indipendentemente dalla loro natura. Nel caso in cui esso è implementato tramite un DBMS, è detto semplicemente data layer.

E' stata adottata l'architettura three-tier in quanto fornisce diversi vantaggi:

- Consente l'integrazione di sistemi differenti all'interno di una LAN;
- Tutta la logica dell'applicazione risiede nello strato intermedio, garantendo per l'intero sistema:
 - maggiore portabilità;
 - manutenzione più semplice;
 - aggiornamento di uno qualsiasi dei tier più semplice;
 - maggiore flessibilità;
 - maggiore scalabilità;
 - maggiore sicurezza.
- Il livello della logica dell'applicazione può essere distribuito su diversi server.

2.2 Architettura Logica in dettaglio

L'architettura Three-Layer del sistema sviluppato è mostrata nella Figura 2.

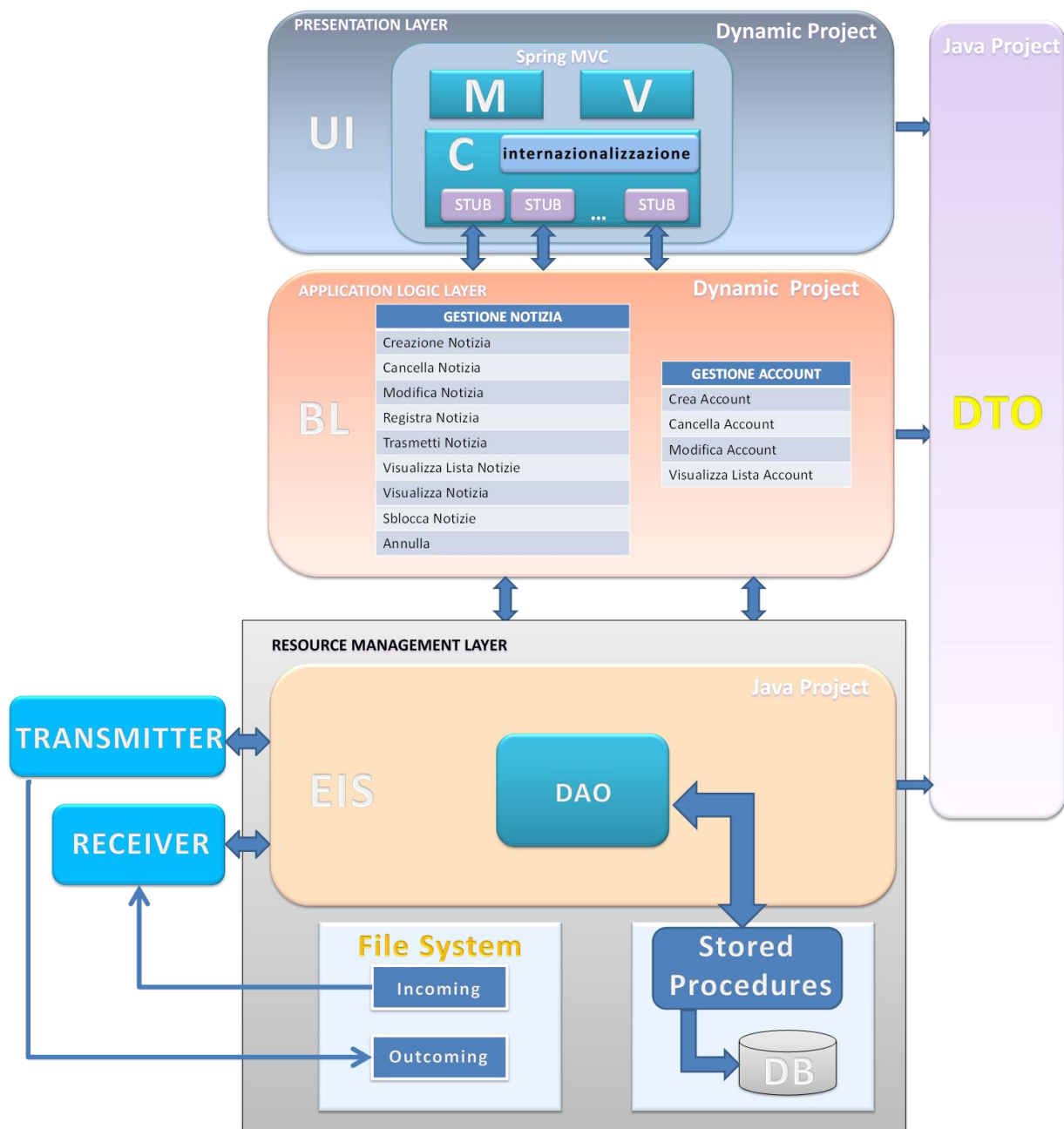


Figura 2 - Architettura Logica del sistema



2.2.1 Presentation Layer - User Interface

Il Presentation Layer è realizzato nel progetto dinamico **User Interface** (UI).

Questo progetto implementa il **Design Pattern MVC**, con l'ausilio del framework *Spring MVC*, che gestisce il modello, le viste e i controller.

Nel controller sono implementati gli Stub dei Web Service e sono presenti i moduli per la gestione dell'internazionalizzazione, la validazione delle form e l'invio delle e-mail.

Il modulo per l'invio delle e-mail permette alla funzionalità "*CreaUtente*" di generare ed inviare le e-mail (contenente i dati di accesso al sistema) agli utenti appena registrati. Questo modulo contiene un componente per la generazione pseudocasuale delle password, ai fini di coprire eventuali problemi di sicurezza.

2.2.2 Business Logic Layer

Il Business Logic Layer è realizzato nel progetto dinamico **Business Logic** (BL) e si occupa della logica dell'applicazione.

Il compito principale di questo layer è quello di accogliere le richieste provenienti dal *Presentation Layer* e di fornire le funzionalità richieste chiamando il livello sottostante per l'accesso ai dati o ai servizi necessari.

Il Business Logic Layer realizzato è composto dai *Web Services* (WS) che implementano le funzionalità che il sistema richiede. E' stato realizzato un web service per ogni funzionalità.

2.2.3 Resource Management Layer – Enterprise Information Service

Per disaccoppiare la logica di business dal livello dei dati è stato introdotto un ulteriore livello, **Enterprise Information Service** (EIS), nel quale è stato implementato il pattern DAO (Data Access Object).

La persistenza dei dati è mantenuta su un database Oracle e la manutenzione e l'accesso fisico ai dati è realizzata tramite **Stored Procedures**.

2.2.4 Data Transfer Object

Nel progetto DTO, **Data Transfer Object**, si trovano i Java Bean che mappano le tabelle del database.

2.2.5 Receiver e Transmitter

Infine, sono state progettate due componenti, **Receiver** e **Transmitter**, responsabili dello scheduling delle notizie provenienti da e verso il sistema di editoria. Il Transmitter si occupa di trasmettere ai clienti le notizie, il Receiver riceve dai fornitori esterni le notizie e le memorizza nel database. Le notizie ricevute sono memorizzate nella cartella “incoming” mentre le notizie da trasmettere sono memorizzate nella cartella “outcoming”.



3 Architettura Fisica

3.1 Introduzione

La presente sezione del documento illustra l'architettura fisica del sistema di editoria *The Yellow Daily*.

3.2 Architettura Fisica in dettaglio

L'architettura fisica adottata è di tipo three-tier (a tre strati) e prevede la suddivisione del sistema in tre diversi moduli dedicati rispettivamente alla presentazione, alla logica funzionale (business logic) e alla gestione dei dati persistenti.

Tali moduli sono intesi interagire fra loro secondo le linee generali del paradigma client-server (l'interfaccia è cliente della business logic, e questa è cliente del modulo di gestione dei dati persistenti) e utilizzando interfacce ben definite.

Una soluzione three-tier tipica prevede, per esempio, un PC dedicato all'interfaccia utente grafica (presentation), una workstation o un web server per la business logic e un database server o un mainframe per la gestione dei dati. Il Thin Client è il cliente-utilizzatore vero e proprio dell'applicazione e accede al sistema tramite browser web. La Figura 3 mostra l'architettura fisica del sistema.

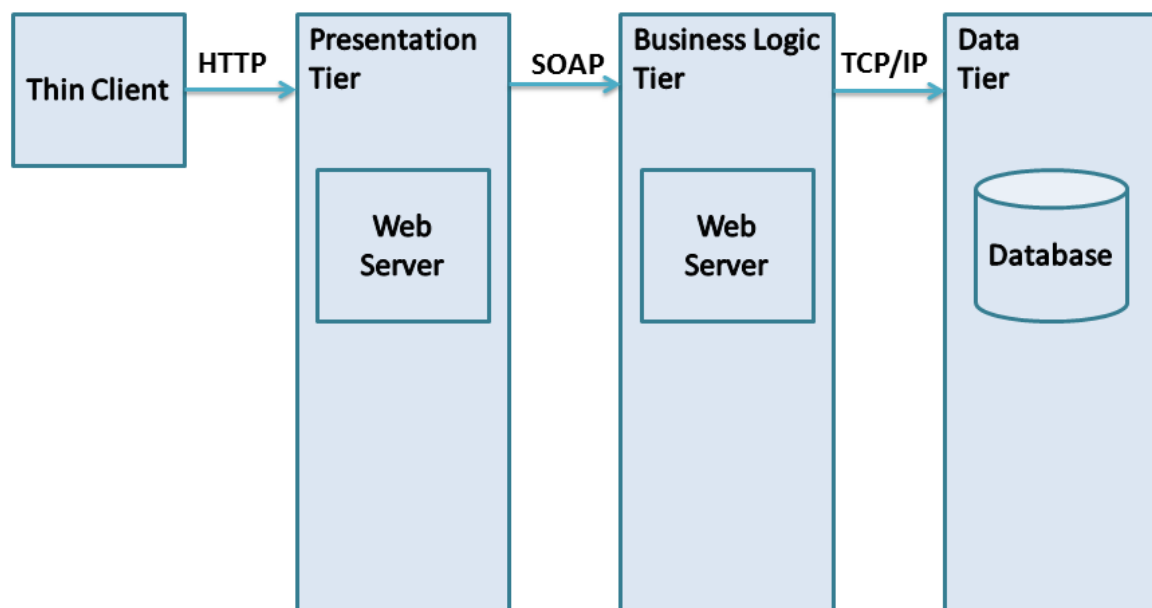


Figura 3 - Architettura Fisica

Un'architettura così composta permette a ciascuno dei tre moduli di essere modificato o sostituito indipendentemente dagli altri conferendo scalabilità e manutenibilità all'applicazione. Permette inoltre che i diversi moduli siano distribuiti su diversi nodi di una rete anche eterogenea.

In particolare l'applicazione "The Yellow Daily" può essere distribuita su 5 macchine diverse:

1. Presentation Tier;
2. Business Tier;
3. Data Tier;
4. Receiver;
5. Transmitter.

3.3 Deployment Diagram

In Figura 4 è riportato il deployment diagram del sistema.

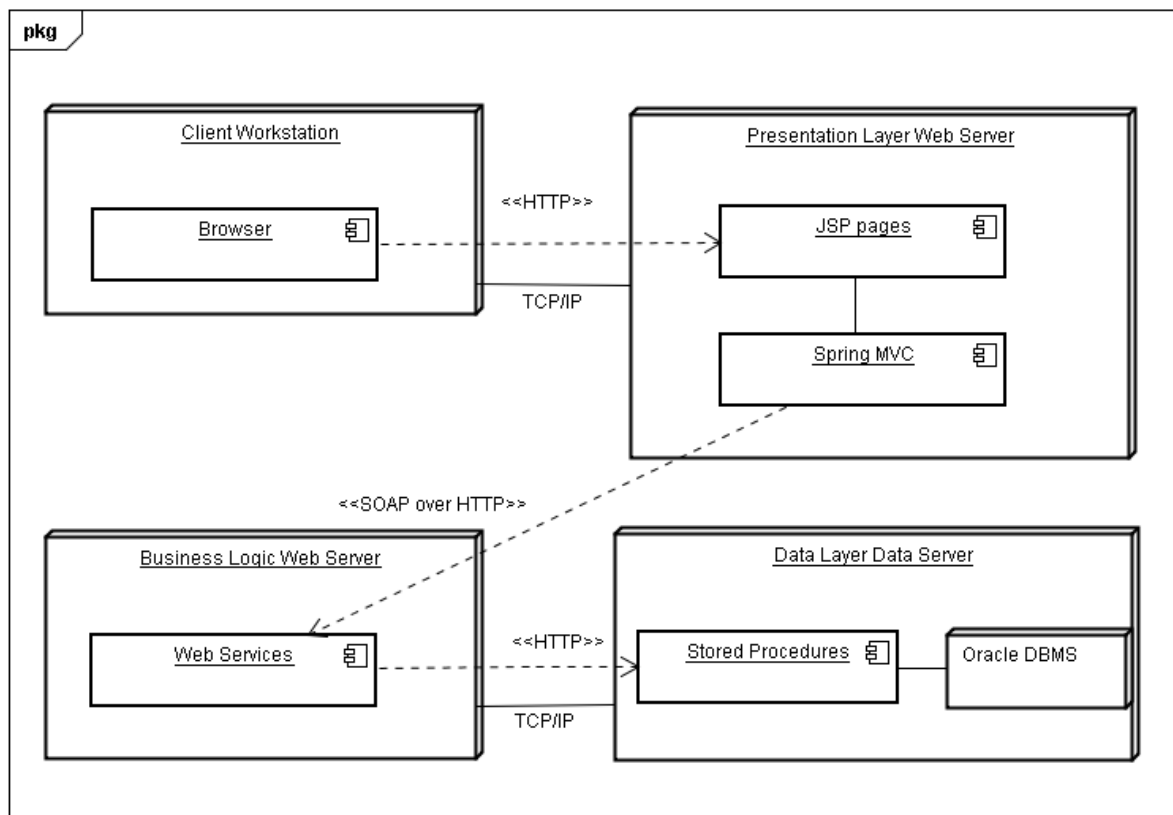


Figura 4 - Deployment diagram

Il Web Server utilizzato, secondo le richieste espresse nel documento dei requisiti, è Apache Tomcat mentre il database relazionale è Oracle 11g.

4 Modello dei dati

4.1 Introduzione

In questo paragrafo verranno analizzati il modello E-R (Entity-Relationship), il modello logico e il modello fisico dei dati scelto per realizzare la persistenza dei dati del sistema editoriale.

4.1.1 Modello ER

In Figura 5 è mostrato il modello ER.

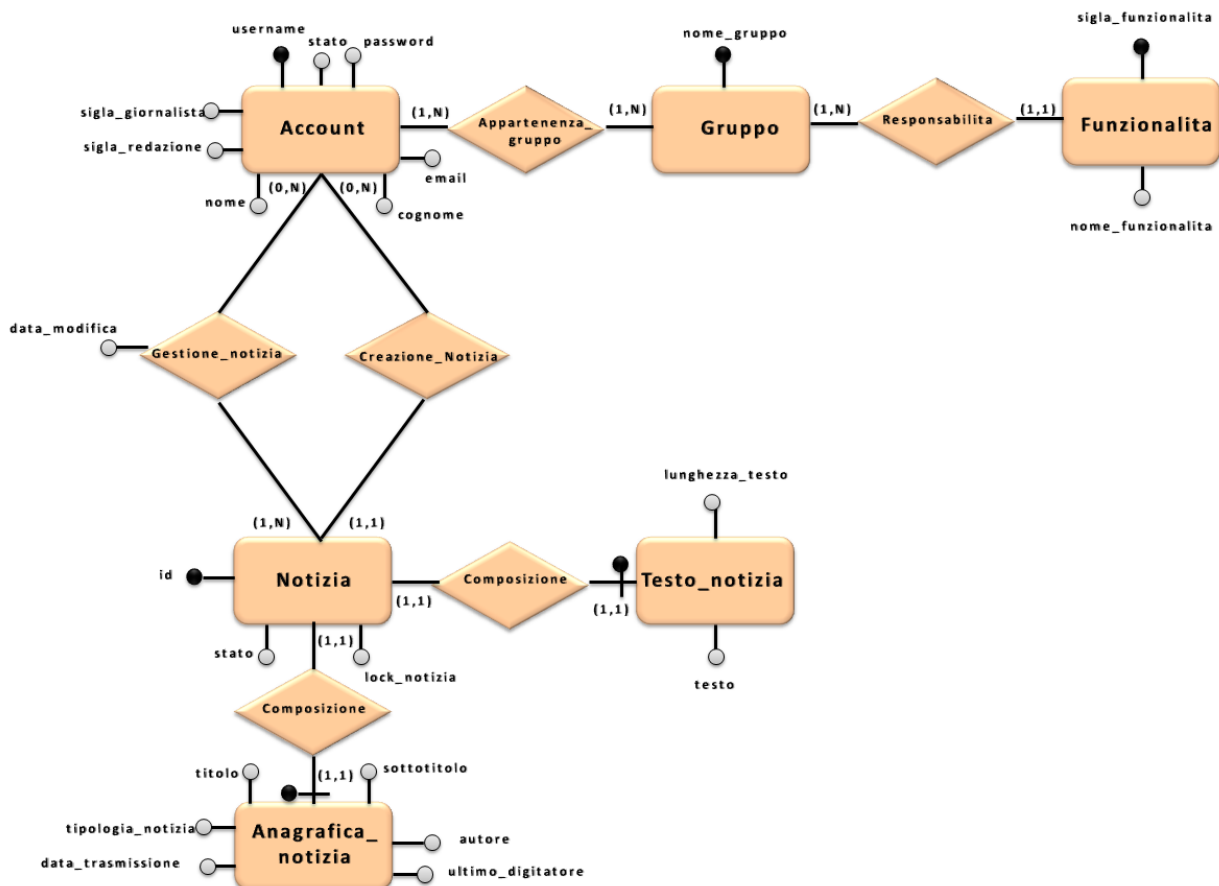


Figura 5 - Modello ER

La scelta del modello ER è stata fatta in base alle specifiche richieste dall'utente in particolare sono state riportate le seguenti assunzioni:



- ACCOUNT e GRUPPO sono legate da una molteplicità di tipo *molti a molti* in quanto ciascun account può essere associato a uno o più gruppi, questo in accordo con i requisiti utente che richiedevano che un amministratore all'interno del sistema editoriale potesse assumere anche il ruolo di giornalista.
- GRUPPI e FUNZIONALITÀ sono caratterizzati da una molteplicità di tipo *molti a uno* in quanto in base alle specifiche espresse nei requisiti, a ciascun gruppo è possibile associare solo determinate funzionalità.
- ACCOUNT e NOTIZIA: si è scelto di collegare queste due entità attraverso due relazioni, la prima riferita alla funzionalità *crea notizia* e la seconda *gestione notizia*. Questa soluzione è stata scelta in previsione di una futura funzionalità che permetterebbe di tenere traccia di tutte le modifiche apportate dai giornalisti su una notizia. A tale scopo è stato aggiunto l'attributo *data Modifica* alla relazione *gestione notizia*.
- TESTO_NOTIZIA e ANAGRAFICA_NOTIZIA sono legate all'entità NOTIZIA con una molteplicità di tipo *uno a uno* e sono legate a quest'ultima attraverso una chiave esterna.

4.1.2 Modello Logico

A seguito della ristrutturazione del modello ER appena descritto si è giunto al seguente modello logico:

Account(username, password, nome, cognome, email, sigla_redazione, sigla_giornalista, stato)

Appartenenza_Gruppo(username, nome_gruppo)

Funzionalità(sigla_funzionalità, nome_funzionalità, nome_gruppo)

Gestione_Notizia(username, notizia, data_modifica)

Gruppo(nome_gruppo)

Notizia(id, stato, lock_notizia, titolo, sottotitolo, tipologia_notizia, autore, ultimo_digitatore, data_creazione, data_trasmissione, testo, lunghezza_testo)

Oltre agli identificatori naturali per ogni tabella presente nel DB è stato aggiunto un identificativo logico ID.

4.1.3 Modello Fisico

Di seguito vengono riportati gli screenshot relativi alle tabelle realizzate seguendo il modello logico mostrato sopra.

4.1.3.1 Account

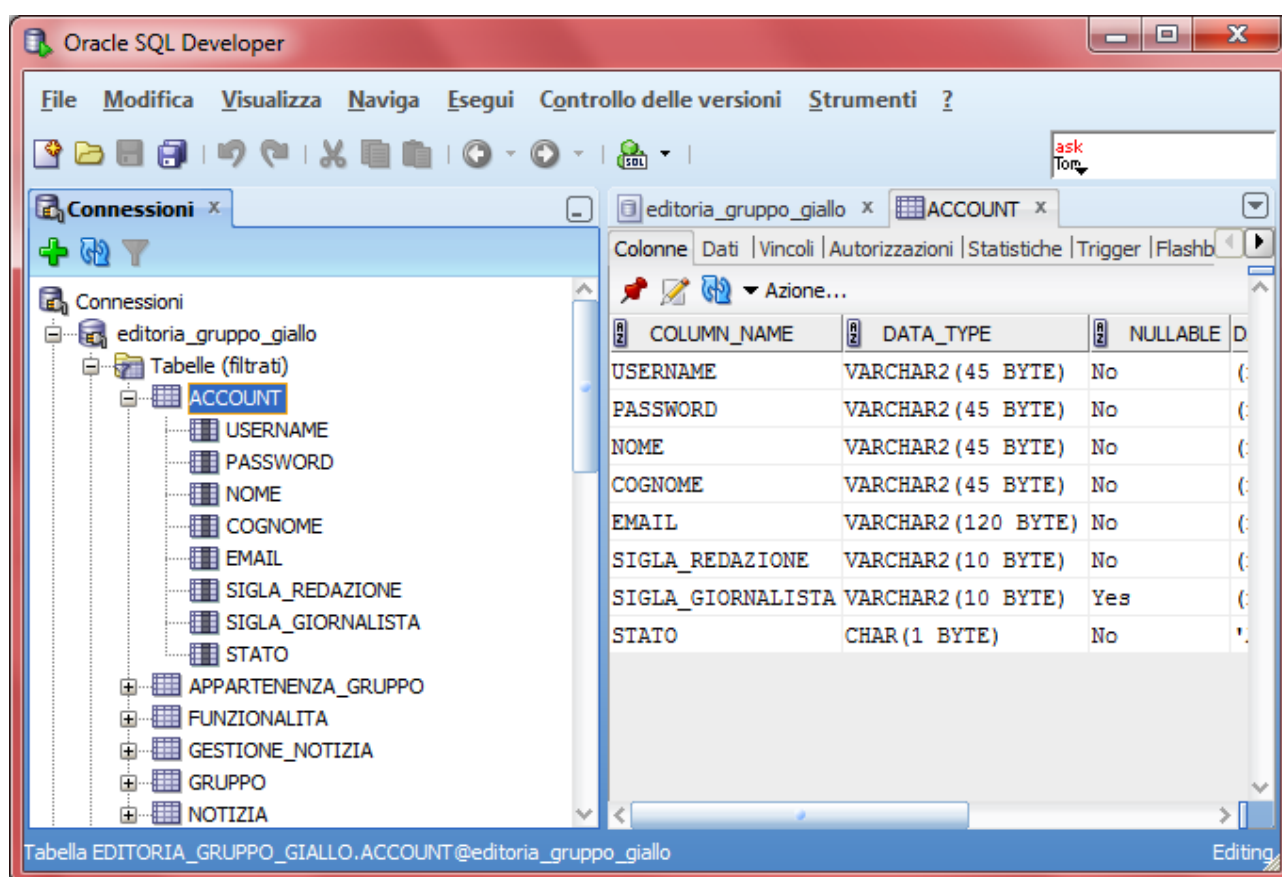


Figura 6 - Screenshot Account

4.1.3.2 Appartenenza_Gruppo

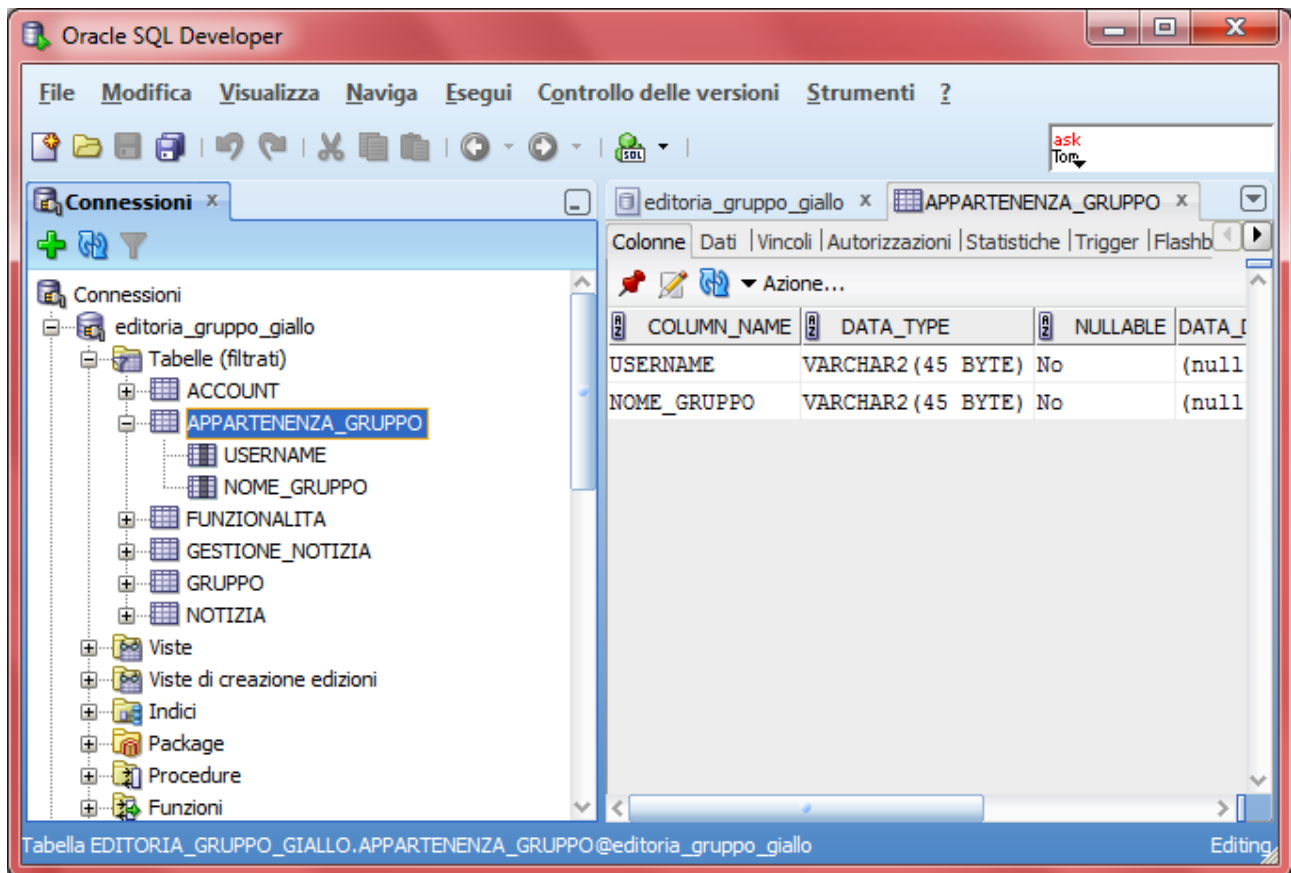


Figura 7 - Screenshot Appartenenza_Gruppo

4.1.3.3 Funzionalità

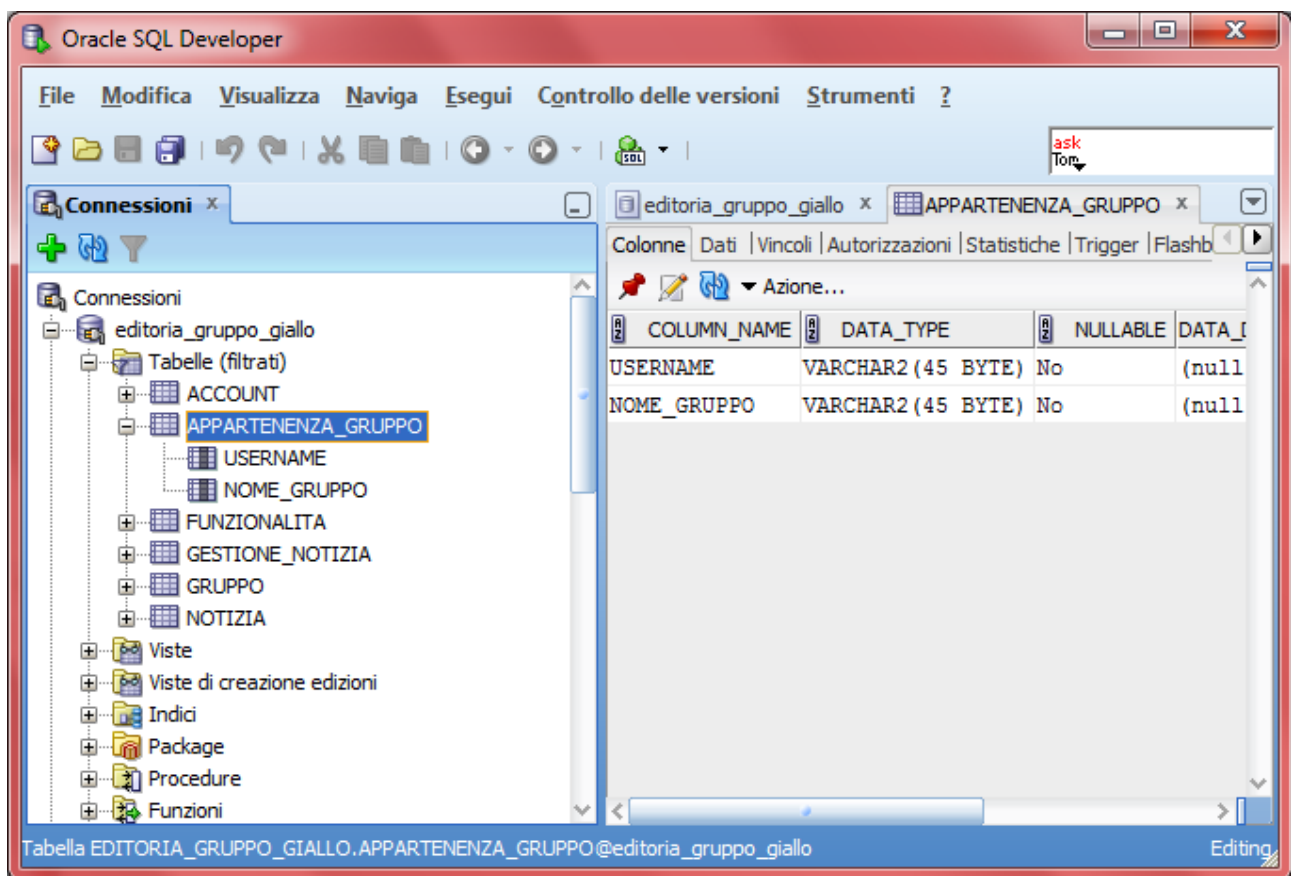


Figura 8 - Screenshot Funzionalità

4.1.3.4 Gestione_Notizia

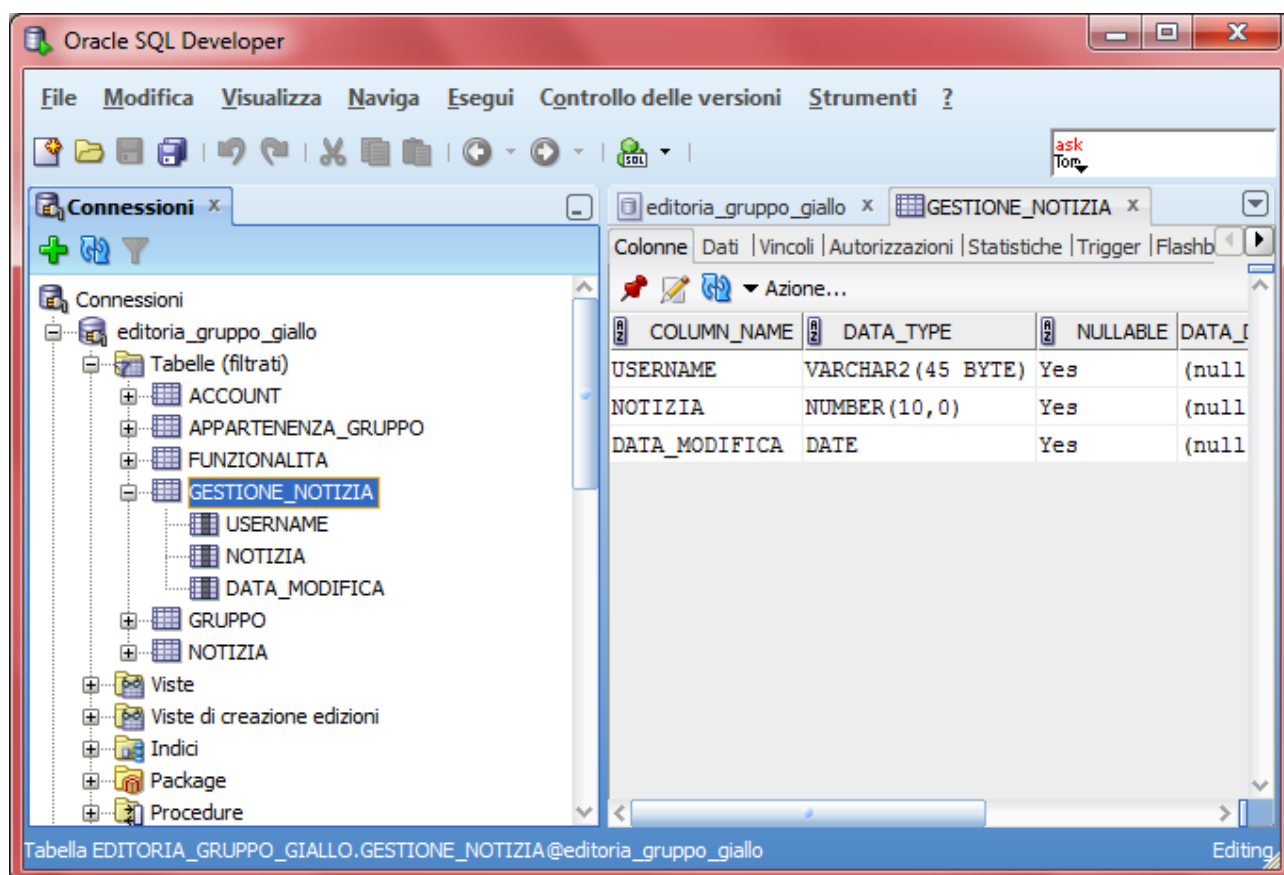


Figura 9 - Screenshot Gestione_Notizia

4.1.3.5 Gruppo

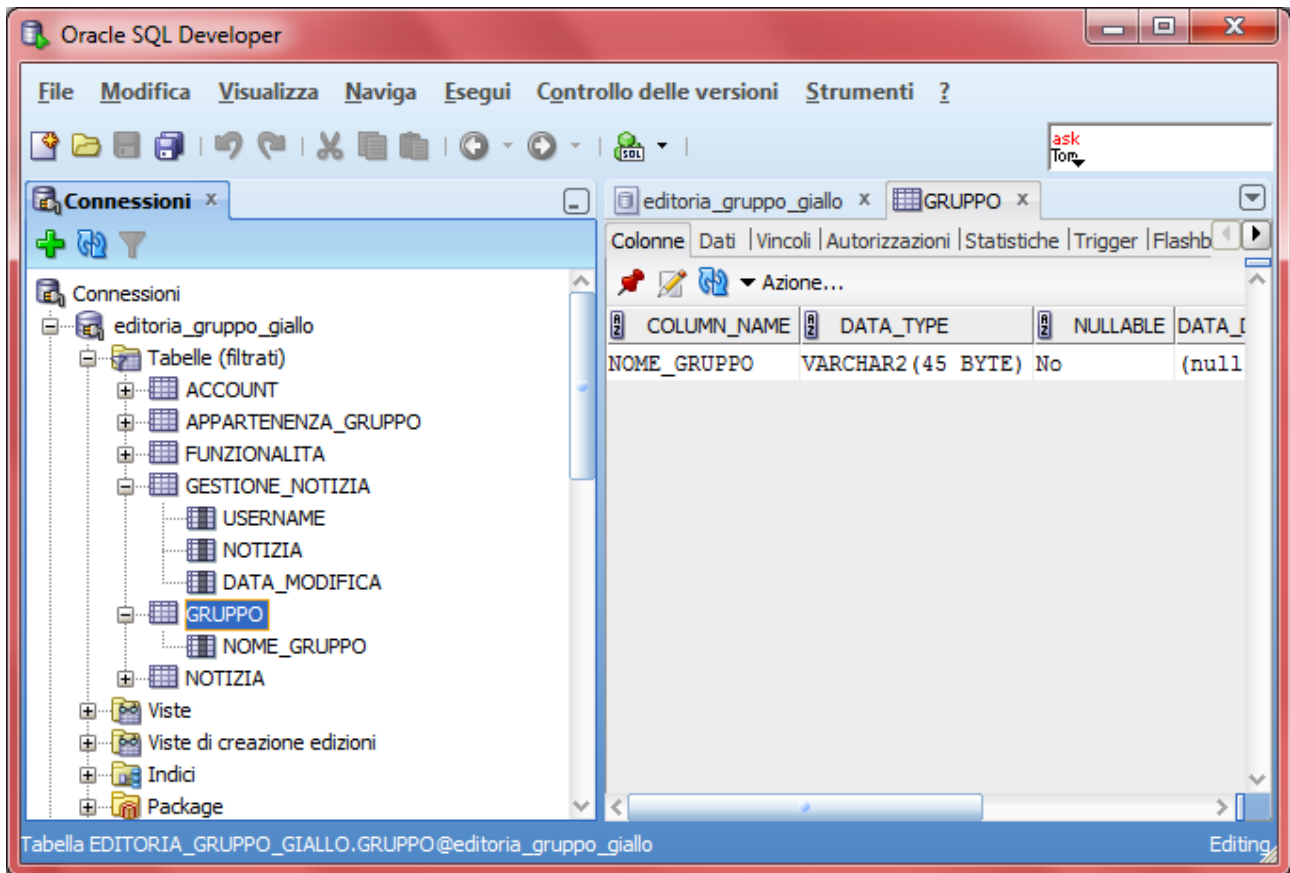
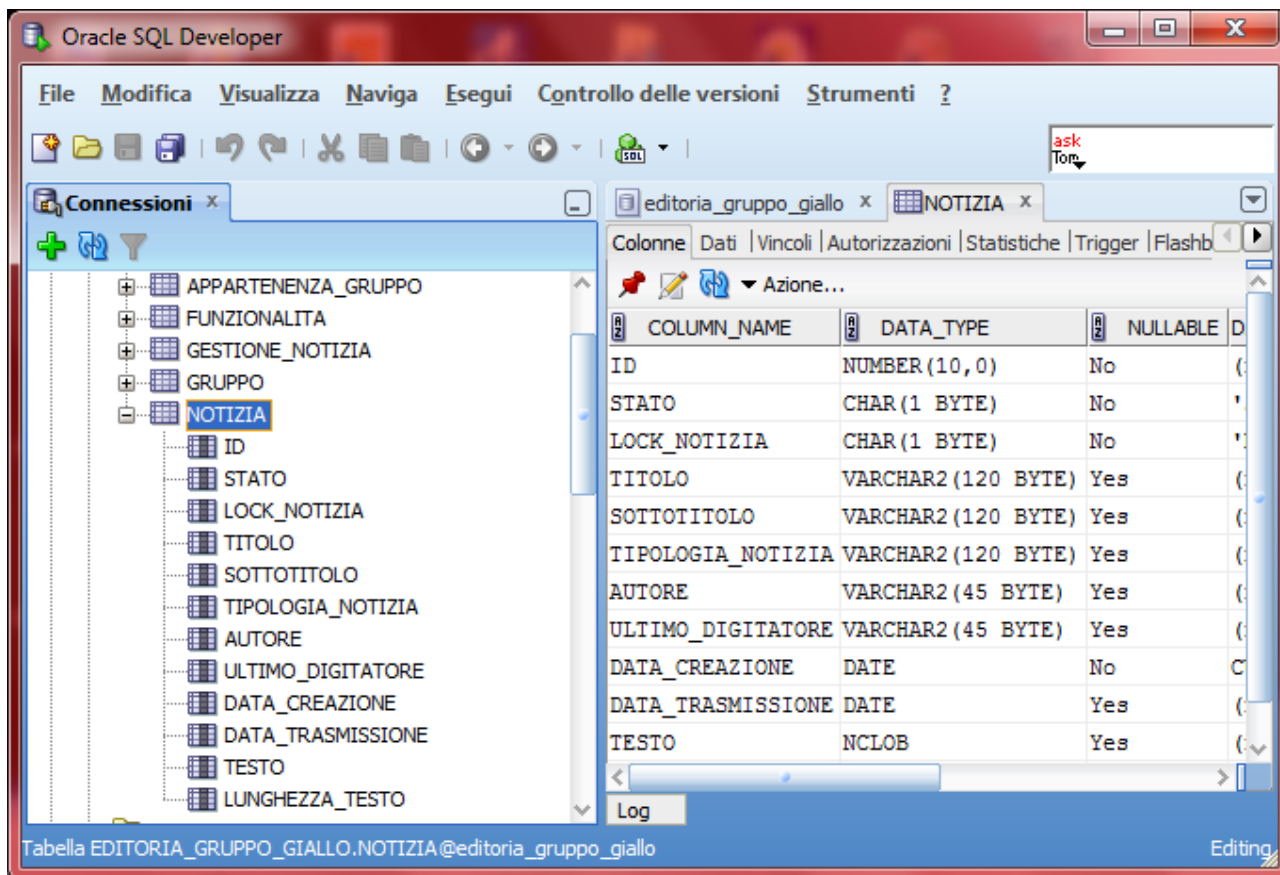


Figura 10 - Screenshot Gruppo

4.1.3.6 Notizia



The screenshot shows the Oracle SQL Developer interface. On the left, the 'Conessioni' pane displays a tree view of the database schema, with the 'NOTIZIA' table selected under the 'EDITORIA_GRUPPO_GIALLO' schema. The main pane shows the 'NOTIZIA' table structure with columns and their data types.

COLUMN_NAME	DATA_TYPE	NULLABLE	D
ID	NUMBER (10, 0)	No	(
STATO	CHAR (1 BYTE)	No	,
LOCK_NOTIZIA	CHAR (1 BYTE)	No	,
TITOLO	VARCHAR2 (120 BYTE)	Yes	(
SOTTOTITOLO	VARCHAR2 (120 BYTE)	Yes	(
TIPOLOGIA_NOTIZIA	VARCHAR2 (120 BYTE)	Yes	(
AUTORE	VARCHAR2 (45 BYTE)	Yes	(
ULTIMO_DIGITATORE	VARCHAR2 (45 BYTE)	Yes	(
DATA_CREAZIONE	DATE	No	C
DATA TRASMISSIONE	DATE	Yes	(
TESTO	NCLOB	Yes	(

Tabella EDITORIA_GRUPPO_GIALLO.NOTIZIA@editoria_gruppo_giallo

Figura 11 - Screenshot Notizia

4.2 Descrizione campi e assunzioni

In questa sezione verranno descritti i campi presenti in ciascuna tabella del database con le relative assunzioni.

4.2.1 Account

username	Username associato all'account, chiave primaria
password	Password associata all'account
nome	Nome dell'utente
cognome	Cognome dell'utente
email	E-mail dell'utente
sigla_redazione	Sigla relativa al nome della Redazione
sigla_giornalista	Sigla relativa al giornalista
stato	Indica lo stato dell'account

Tabella 1- Tabella Account

A valle di un ragionamento sul processo di accesso al sistema si è scelto di generare il valore del campo **password** in modo automatico dal sistema per motivi di sicurezza.

Tale valore insieme all'username, inserito dall'amministratore, verranno inviate via e-mail al relativo giornalista. A tal scopo è stato aggiunto il campo **email** alla tabella Account.

Nel documento dei requisiti era stato specificato che l'username fosse generato automaticamente in funzione del nome e del cognome dell'utente. Nell'applicazione si è preferito far generare in modo automatico tale campo per permettere la registrazione di due account aventi lo stesso nome e cognome.

Per rendere la password resistente agli attacchi è applicata la *crittografia tramite algoritmo MD5*. Durante la registrazione di un utente, la password generata in modo automatico durante il processo verrà codificata tramite MD5 e la sua firma digitale verrà memorizzata nel database. Successivamente, durante il login la password immessa dall'utente subirà lo stesso trattamento e verrà confrontata con la copia in possesso del server, per avere la certezza dell'autenticità del login.

Il campo *stato*, rispettando quelli che sono i requisiti utente, è stato settato in modo tale da poter assumere solo i seguenti valori:

- A (attivo)
- D (delete - cancellato)

4.2.1 Appartenenza_Gruppo

username	Chiave esterna riferita al campo username della tabella Account
nome_gruppo	Chiave esterna riferita al campo nome_gruppo della tabella Gruppo

Tabella 2 - Tabella Appartenenza_Gruppo

4.2.2 Funzionalità

sigla_funzionalità	Sigla relativa alla funzionalità, chiave primaria
nome_funzionalità	Nome della funzionalità
nome_gruppo	Attributo riferito al campo nome_gruppo della tabella Gruppo

Tabella 3 - Tabella Funzionalità

4.2.3 Gestione_Notizia

username	Chiave esterna riferita al campo username della tabella Account
notizia	Chiave esterna riferita al campo id della tabella Notizia
data_modifica	Data in cui la notizia è modificata

Tabella 4 - Tabella Gestione_Notizia

4.2.4 Gruppo

nome_gruppo	Nome relativo al gruppo, chiave primaria

Tabella 5 - Tabella Gruppo

I gruppi possibili predefiniti su Database in accordo alle specifiche fornite sono:

- Amministratore
- Giornalista

4.2.5 Notizia

id	Chiave primaria della notizia
stato	Indica lo stato della notizia
lock_notizia	Indica se la notizia è stata già presa in carico (bloccata) da un giornalista
titolo	Titolo della notizia
sottotitolo	Sottotitolo della notizia
tipologia_notizia	Indica il genere della notizia ad esempio notizia sportiva, notizia di cronaca.
autore	Sigla del giornalista che ha creato la notizia
ultimo_digitatore	Sigla dell'ultimo giornalista che ha modificato la notizia o sigla dell'ultimo giornalista che ha impostato il lock o ha trasmesso la notizia.
data_creazione	Data di creazione della notizia
data_trasmissione	Data in cui la notizia viene trasmessa
testo	Testo della notizia
lunghezza_testo	Lunghezza del testo della Notizia

Tabella 6 - Tabella Notizia

Il campo **stato**, rispettando quelli che sono i requisiti utente, è stato settato in modo tale da poter assumere solo i seguenti valori:

- S (editabile)
- Q (in trasmissione)
- T (trasmessa)
- C (cancellata)

Allo stesso modo, il campo **lock_notizia** è stato settato con i seguenti valori:

- Y (bloccata da un altro utente)
- N (disponibile)

La chiave primaria **id** viene auto incrementata mediante l'uso delle sequenze di Oracle.

Per quanto concerne il campo **testo** si è preferito superare il limite tecnico di un massimo di 4000 caratteri utilizzando un NCLOB rispetto ad un Varchar2.

Rispetto ai requisiti richiesti è stato aggiunto un ulteriore attributo **tipologia_notizia** che potrà essere utilizzata in futuro qualora si volesse implementare una funzione di ricerca per genere.

4.3 Script SQL

Per la gestione del database Oracle sono stati generati alcuni script SQL, in particolare:

- *creazione utente*: permette di creare un utente per la connessione al database, a cui vengono assegnati tutti i privilegi (1.createUser.sql);
- *creazione tabelle*: permettono la creazione di tutte le tabelle del database (2.createTables.sql);
- *creazione sequenza*: consente la creazione delle sequenze che incrementano il campo *id* di ciascuna tabella in maniera sequenziale (3.createSequence.sql);
- *creazione trigger*: permette l'incremento della sequenza creata al passo precedente nel momento in cui si genera un nuovo evento di inserimento (4.createTrigger.sql);
- *creazione stored procedure*: consente la creazione delle stored procedure (5.createProcedures.sql);
- *popolamento database*: permettono di inserire, in alcune delle tabelle create, dei record predefiniti (6.populateDB.sql);
- *cancellazione record tabelle*: consente di cancellare i record all'interno delle tabelle (7.cleanAll.sql);
- *cancellazione contenuto database*: consente di cancellare tutte le tabelle, le sequence, i trigger e le stored procedures dal database (8.dropContent.sql).

4.4 Tavola dei volumi

Di seguito viene mostrata una tabella dei volumi per esprimere i dati del sistema in un intervallo di tempo pari a due mesi. Si tenga conto che tale sistema tende ad aumentare il volume dei dati nel tempo.

Si presuppone che il sistema di editoria abbia 100 utenti, di cui 5 amministratori e 95 giornalisti. Inoltre si presuppone che ogni utente pubblica mediamente 50 notizie ogni due mesi e che ogni notizia venga mediamente modificata tre volte.

Concetto	Costrutto	Volume
Account	Entità	100
Appartenenza_Gruppo	Relazione	105
Gruppo	Entità	2
Funzionalità	Entità	12
Notizia	Entità	5000
Gestione_Notizia	Relazione	15000

Tabella 7 - Tavola dei volumi



5 Tecnologie utilizzate

Nella realizzazione del sistema editoriale **The Yellow Daily** sono state utilizzate le tecnologie descritte di seguito.

5.1 Spring MVC

Spring MVC (<http://springsource.org>) è il sotto-framework di Spring che permette di servirsi del Framework Spring per realizzare applicazioni web. Permette quindi di avvantaggiarci delle peculiarità di Spring (IoC, DI, ecc...) anche nello sviluppo di web application.

Spring MVC ovviamente usa il pattern MVC:

- **Model:** qui spring non fa nulla e segue la sua filosofia di base. Il modello è rappresentato dai POJO, classi Java "nude e crude" (JavaBean).
- **Controller:** il grosso di Spring MVC consiste nel fornire classi di supporto per la parte controller, ossia per ricevere e processare request http provenienti da un client. Il controller deve anche reindirizzare il client sulla vista giusta.
- **View:** la parte controller crea viste da mostrare al client, ma la generazione della vista vera e propria è demandata al framework di visualizzazione che si è scelto di usare. Spring supporta diversi framework di "vista" per il web: JSP, Velocity, JSF, ecc...

Spring quindi non fornisce una tecnologia di vista propria, ma invita a scegliere quella che si preferisce. Anche qui Spring segue la sua filosofia di base di non invasività. Il framework usato (JSP, Velocity, JSF, ecc...) non è consapevole in alcun modo di essere usato da Spring (nessuna dipendenza).

In particolare nello sviluppo dell'applicazione **The Yellow Daily** è stato utilizzato il framework Spring MVC 3.2.2 abbinata alla tecnologia di "vista" *Servlet-JSP* e ai tag *JSTL*.

L'utilizzo del framework ha permesso inoltre di fornire:

- **Validazione delle form:** consente di validare le form di inserimento dei dati in base ai requisiti;
- **Internazionalizzazione:** possibilità di visualizzare l'interfaccia utente in lingue differenti.



5.2 Apache Tomcat

Apache Tomcat (<http://tomcat.apache.org>) è un contenitore servlet open source sviluppato dalla Apache Software Foundation.

Implementa le specifiche di JavaServer Pages (JSP) e Servlet di Sun Microsystems, fornendo così una piattaforma software per l'esecuzione di applicazioni Web sviluppate in linguaggio Java.

Tomcat offre molteplici vantaggi:

- Semplice da utilizzare;
- Stabile
- Affidabile

Tomcat è utilizzato per effettuare il deploy del Presentation Tier e del Business Logic Tier.



5.3 Apache Axis2

Apache Axis2 (<http://axis.apache.org/axis2/java/core>) è un'infrastruttura di Apache Software Foundation per creare, pubblicare e utilizzare Web Service in Java.

Axis 2 offre molteplici vantaggi:

- Supporto per Web Service stateful ed asincroni;
- Ricevere ed elaborare messaggi SOAP;
- Maggiore scalabilità;
- Deployment/undeployment di Web Service a runtime;
- Velocità;
- Stabilità.

E' stato utilizzato Axis 2 per implementare i servizi della logica di business.



5.4 Apache Log4J

Apache Log4J (<http://logging.apache.org/log4j/1.2/>) è una libreria java sviluppata da Apache Software Foundation che permette di mettere appunto un ottimo sistema di logging per tenere sotto controllo il comportamento di una applicazione, sia in fase di sviluppo che in fase di test e messa in opera del prodotto finale.



5.5 Framework Quartz

Quartz (<http://quartz-scheduler.org/>) è un framework open source che permette di schedulare i processi da una qualsiasi applicazione java, sia desktop sia web.

Quartz fornisce molteplici vantaggi:

- Configurazione semplice dei processi;
- Persistenza dei processi mediante memorizzazione su banche dati;
- Rischiazione automatica dei processi in fase di start up;
- Facile monitoraggio del flusso di esecuzione dei processi.

E' stato utilizzato Quartz per schedulare allo startup i processi Receiver e Transmitter.



5.6 JavaMail

JavaMail (<http://www.oracle.com/technetwork/java/javamail/index.html>) è una libreria che fornisce le classi necessarie per la gestione della posta elettronica in linguaggio Java.

Supporta tutti i protocolli di posta elettronica POP, SMTP, IMAP anche nelle diverse versioni (per esempio POP3). È già incluso nella Java EE, mentre nella J2SE è necessario scaricarlo dal sito della Sun. Per il suo funzionamento occorre anche il package JAF sempre scaricabile dal medesimo sito.

E' stata utilizzata tale libreria per l'invio delle e-mail (contenenti i dati di accesso al sistema) agli utenti appena registrati.



5.7 Oracle 11g

Oracle (<http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html>) è uno tra i più famosi database management system (DBMS), cioè sistema di gestione di basi di dati.

Oracle fa parte dei cosiddetti RDBMS (Relational DataBase Management System) ovvero di sistemi di database basati sul Modello relazionale che si è affermato come lo standard dei database dell'ultimo decennio.