Project Work UIIP - Sistema Editoriale

Documento di architettura comprensivo di modello dati e tecnologie scelte

Gruppo Giallo

Andrea Castaldo
Antonio Maria Fonzo
Ivan Salinaro
Armando Soriano
Cinzia Tito

16/04/2013

Reply
Living network

Revision History

Date	Version	Description	Author
16 Aprile 2013	R2A	E' stata aggiunta la possibilità di effettuare la cancellazione logica di un account.	Armando

INDICE

1 INTRODUZIONE 3

1.1 SCOPO DEL DOCUMENTO 3

2 PROGETTO ARCHITETTURALE 4

3 TECNOLOGIE 7

4 MODELLO DEI DATI 9

4.1 STRUTTURA DEL DATABASE 9

4.1.1 Modello ER 9

4.1.2 Modello Logico 10

4.1.3 Modello Fisico 11

4.2 DESCRIZIONE CAMPI E ASSUNZIONI 17

4.2.1 Account 17

4.2.1 Appartenenza_Gruppo 18

4.2.1 Funzionalita 18

4.2.1 Gestione_Notizia 18

4.2.2 Gruppo 19

4.2.3 Notizia 19

4.3 SCRIPT SQL 21

4.3.1 Crea Utente 21

4.3.2 Creazione tabelle 21

4.3.2.1 Account 21

4.3.2.2 Gruppo 22

4.3.2.3 Appartenenza_Gruppo 22

4.3.2.4 Funzionalita 22

4.3.2.1 Notizia 23

4.3.2.2 Gestione_Notizia 23

4.3.1 Sequence 24

4.3.2 Trigger 24

4.3.3 Popolamento DB 25

4.3.4 Cancellazione record tabelle 27

4.3.5 Cancellazione tabelle 27

4.4 TAVOLA DEI VOLUMI 28

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" per quanto riguarda l'architettura, le tecnologie utilizzate e il modello dei dati.
In particolare, nella prima sezione verrà presentata l'architettura JEE scelta per la realizzazione del progetto, nella seconda sezione invece verranno descritte le tecnologie utilizzate, nell'ultima verrà analizzata la struttura del database.

2 Progetto architetturale

La logica architetturale dell'applicazione è basata su una implementazione di tipo three-layer (Figura 1 - Modello architetturale):

1. Presentation layer;
2. Business-Logic layer;
3. Data layer.

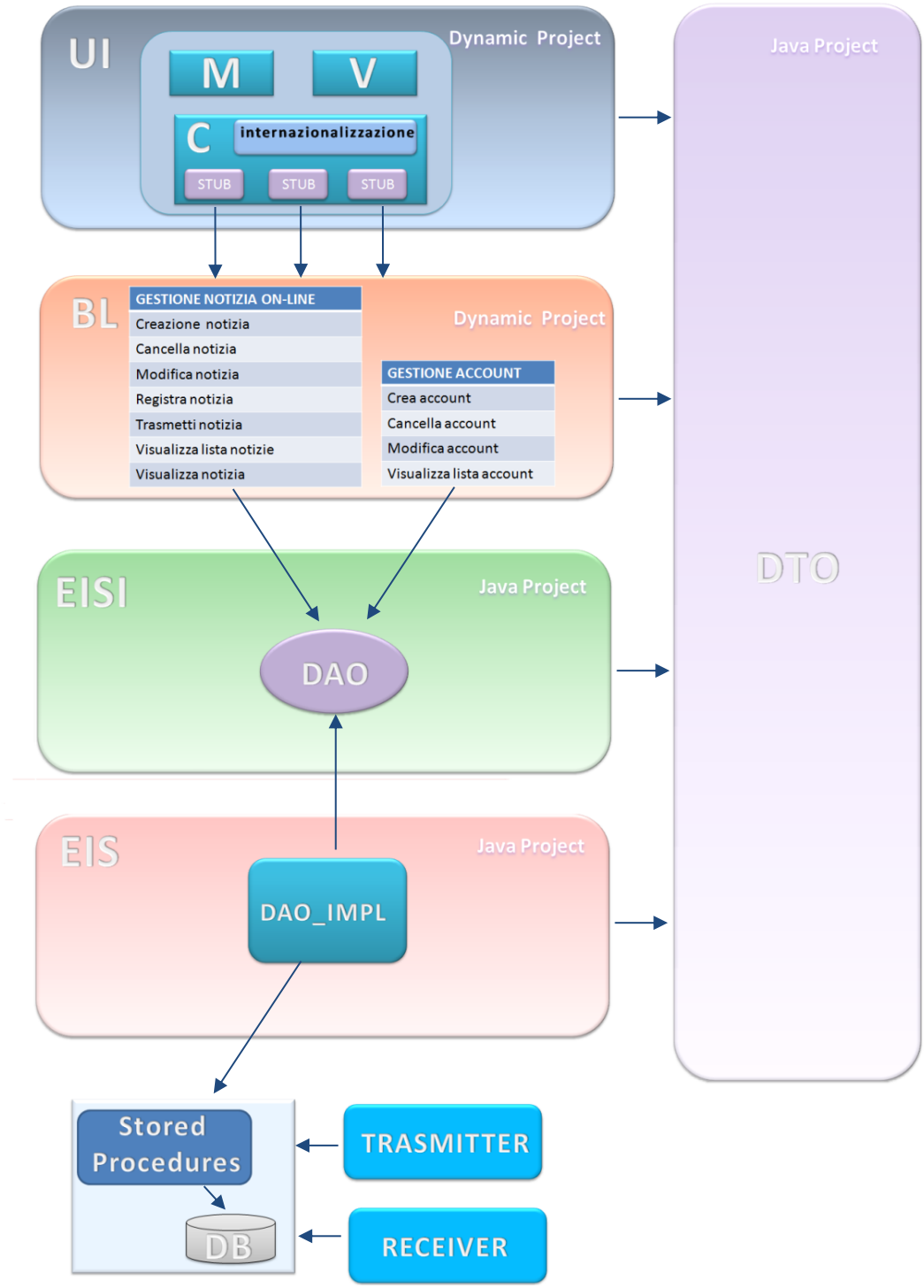

Figura 1 - Modello architetturale

Dal lato fisico, tale architettura verrà ulteriormente divisa in sette layer differenti, ma dipendenti tra loro.

Progetto UI, **User Interface**, in questo progetto è implementato il pattern MVC lato Web che conterrà il modello, le viste e i controller. Nei controller di tale pattern vi sarà un modulo per la gestione dell'internazionalizzazione delle lingue, e in più saranno implementati gli Stub dei Web Service. Il controller che si occuperà della funzionalità *crea utente* conterrà al suo interno un componente per l'invio delle eMail ai clienti appena registrati, tale eMail conterrà i dati relativi all'accesso al sistema. Dentro questo componente vi si trova il modulo per la generazione delle password casuali, ai fini di coprire eventuali problemi di sicurezza. I Web Service (WS) implementano le funzionalità che il sistema richiede e faranno parte del progetto BL, **Business Logic**. Tale progetto conterrà esclusivamente WS. Per disaccoppiare la logica di business dal livello dei dati è stato introdotto un ulteriore livello, EIS **Enterprise Information Service**, nel quale è stato implementano il pattern DAO.

Ognuno di questi componenti ha, nel progetto EISI, **EIS-Interface**, una propria interfaccia verso il livello di logica di business . Il Business Layer comunica quindi con lo strato sottostante utilizzando le interfacce presenti nel EISI.

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

Infine, sono state progettate due componenti, **Receiver** e **Trasmitter**, responsabili dello scheduling delle notizie provenienti da e verso il sistema di editoria. Il Trasmitter si occupa di trasmettere ai clienti le notizie, il Receiver riceve dai fornitori esterni le notizie e le memorizza nel database.

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

3 Tecnologie

Nella realizzazione del sistema editoriale è stato previsto l'utilizzo delle seguenti tecnologie:

- Pattern architetturale MVC lato web → SpringMVC 3.2.2 (http://springsource.org);

- Viste del MVC → pagine JSP e tag JSTL;
- Internazionalizzazione → librerie i18n (già presente nella libreria JDK);
- Web Services → i servizi della logica di business sono implementati utilizzando la tecnologia Apache Axis 2 (http://axis.apache.org/ axis2/java/core);

- Invio eMail → librerie JavaMail (http://java.net/projects/javamail/downloads/download/javax.mail.jar);
- Tool di gestione dei log →Apache Log4J (http://logging.apache.org/log4j/1.2/);
- Specifica JNDI → utilizzata per effettuare il lookup di risorse dal contesto dell'applicazione, risorse quali la sessione JavaMail e DataSource. L'utilizzo di tale specifica consentirà di migliorare le prestazioni complessive del sistema;
- Receiver/Trasmitter → sono processi java implementati con il framework Quartz (http://quartz-scheduler.org) e vengono schedulati allo startup;

- DBMS (DataBase Management System) → Oracle 11g (http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html);

- Web Server → Apache Tomcat 6 (http://tomcat.apache.org)

4 Modello dei dati

4.1 Struttura del database

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 2 è mostrato il modello ER.

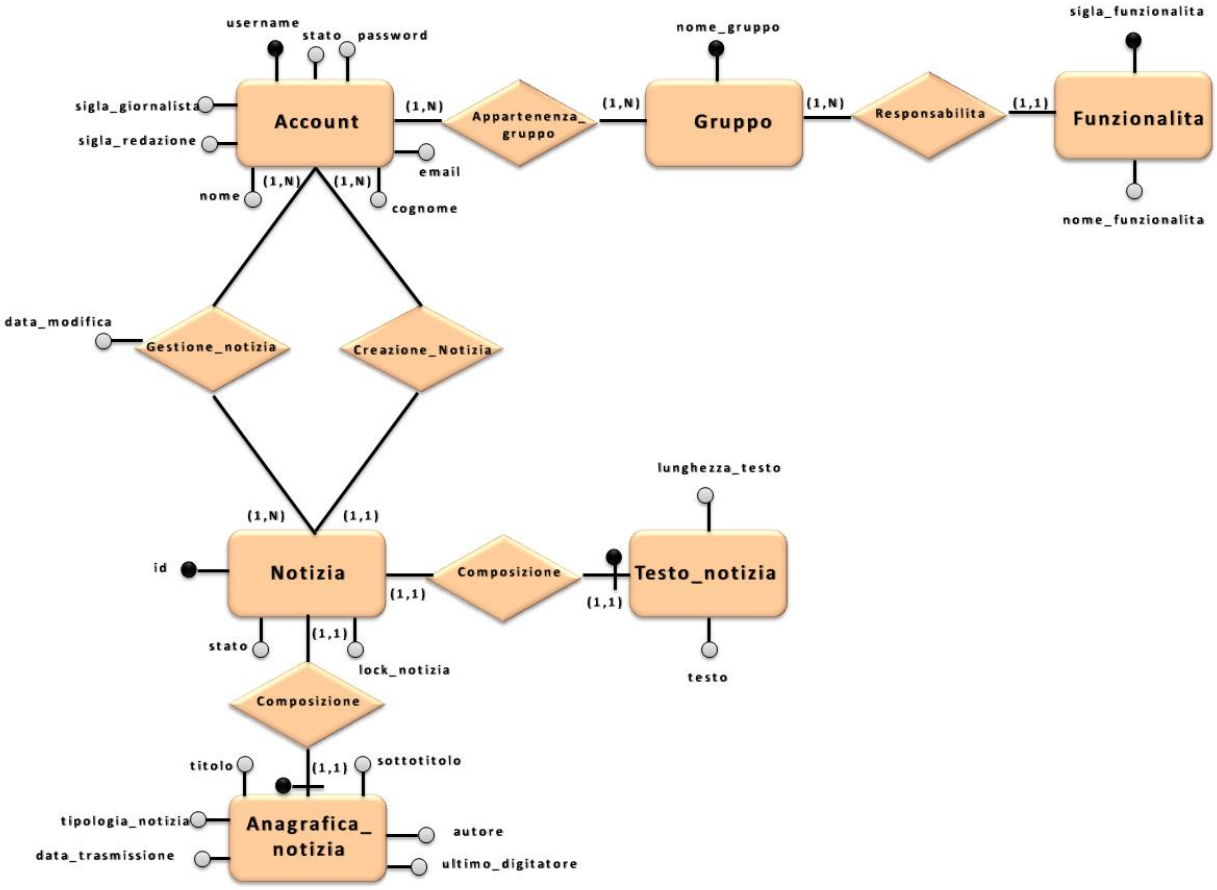

Figura 2 - Modello ER

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

Appartenenza_Gruppo(id, username, nome_gruppo)

Funzionalita(id, sigla_funzionalita, nome_funzionalita, nome_gruppo)

Gestione_Notizia(id, username, notizia, data_modifica)

Gruppo(id, 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.

Account

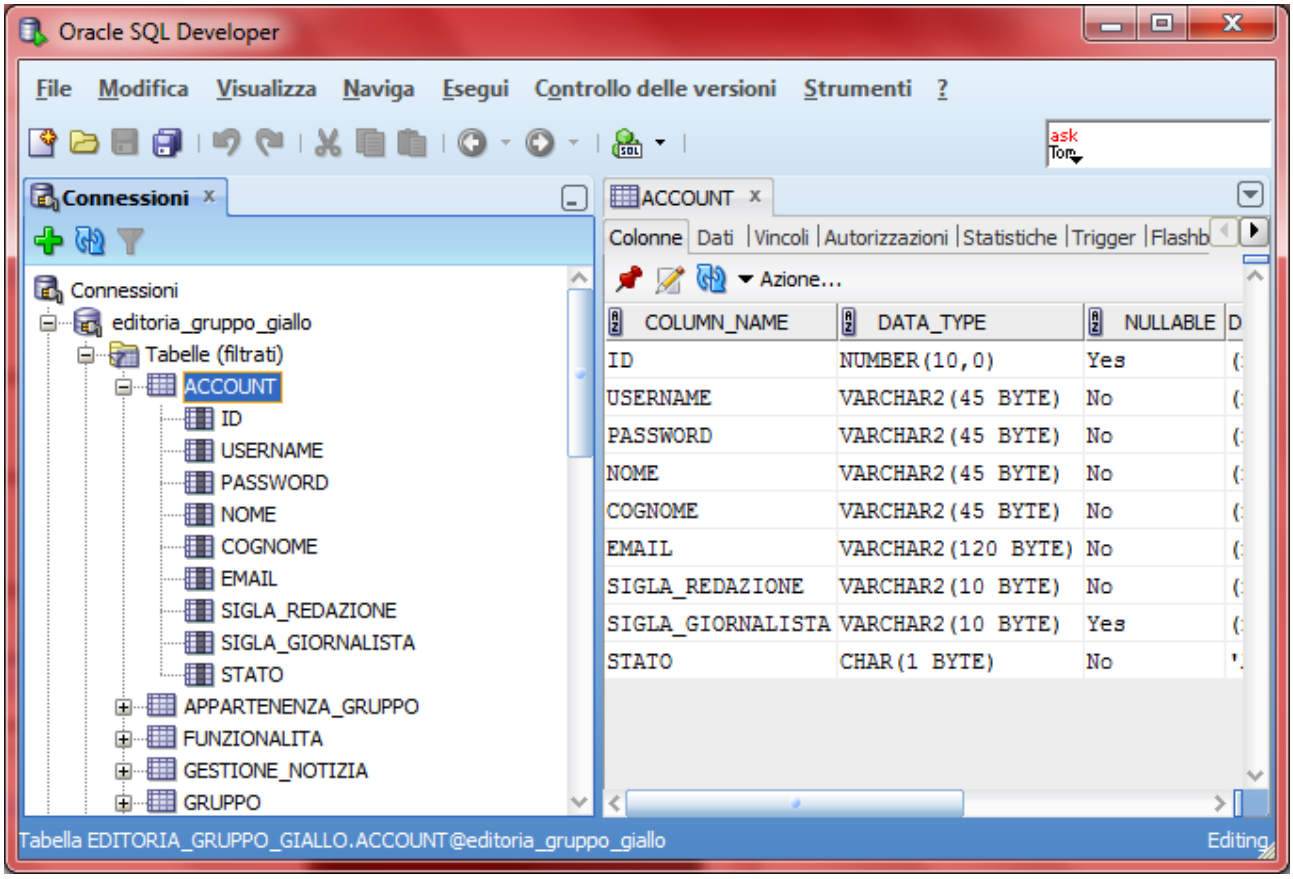

Figura 3 - Screenshot Account

Appartenenza_Gruppo

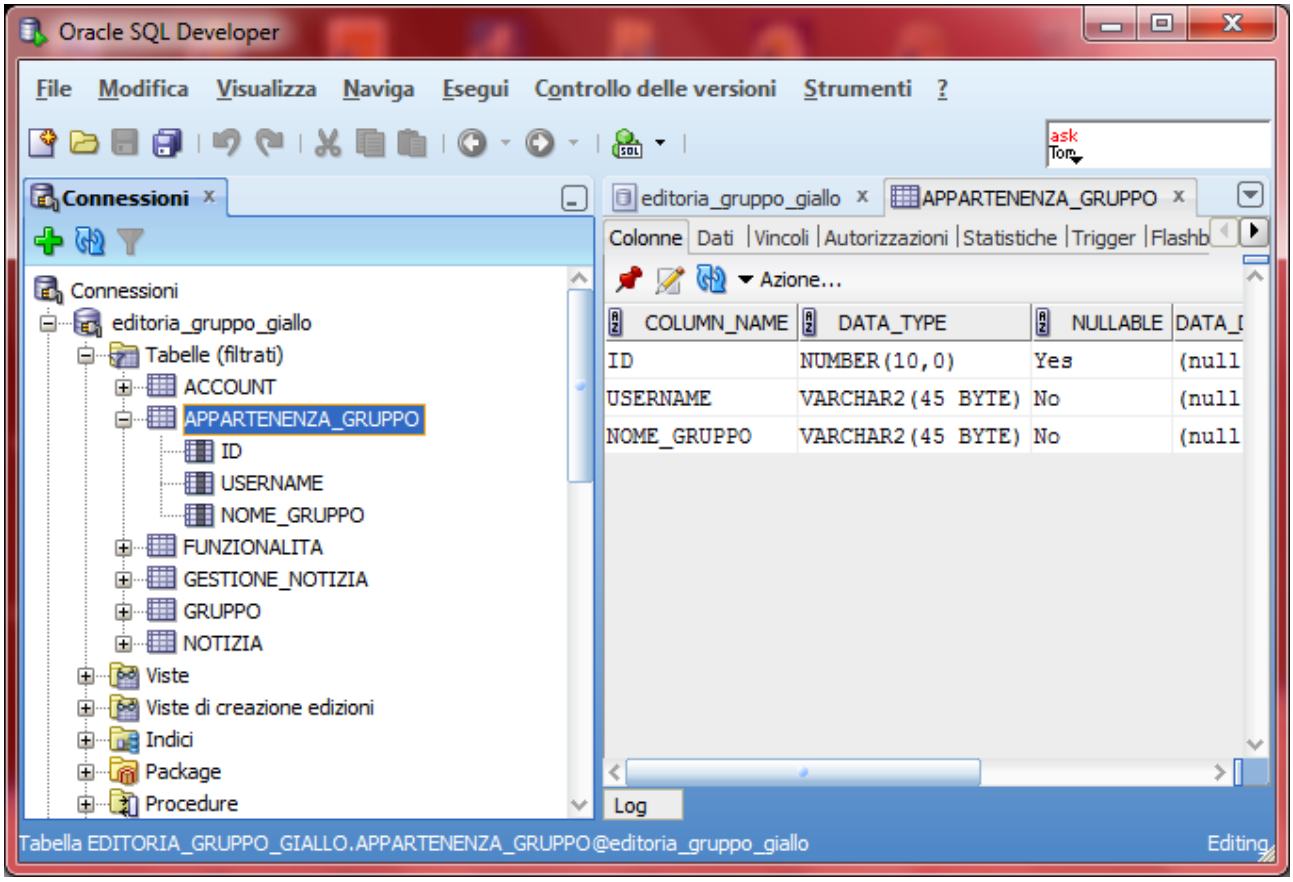

Figura 4 - Screenshot Appartenenza_Gruppo

Funzionalita

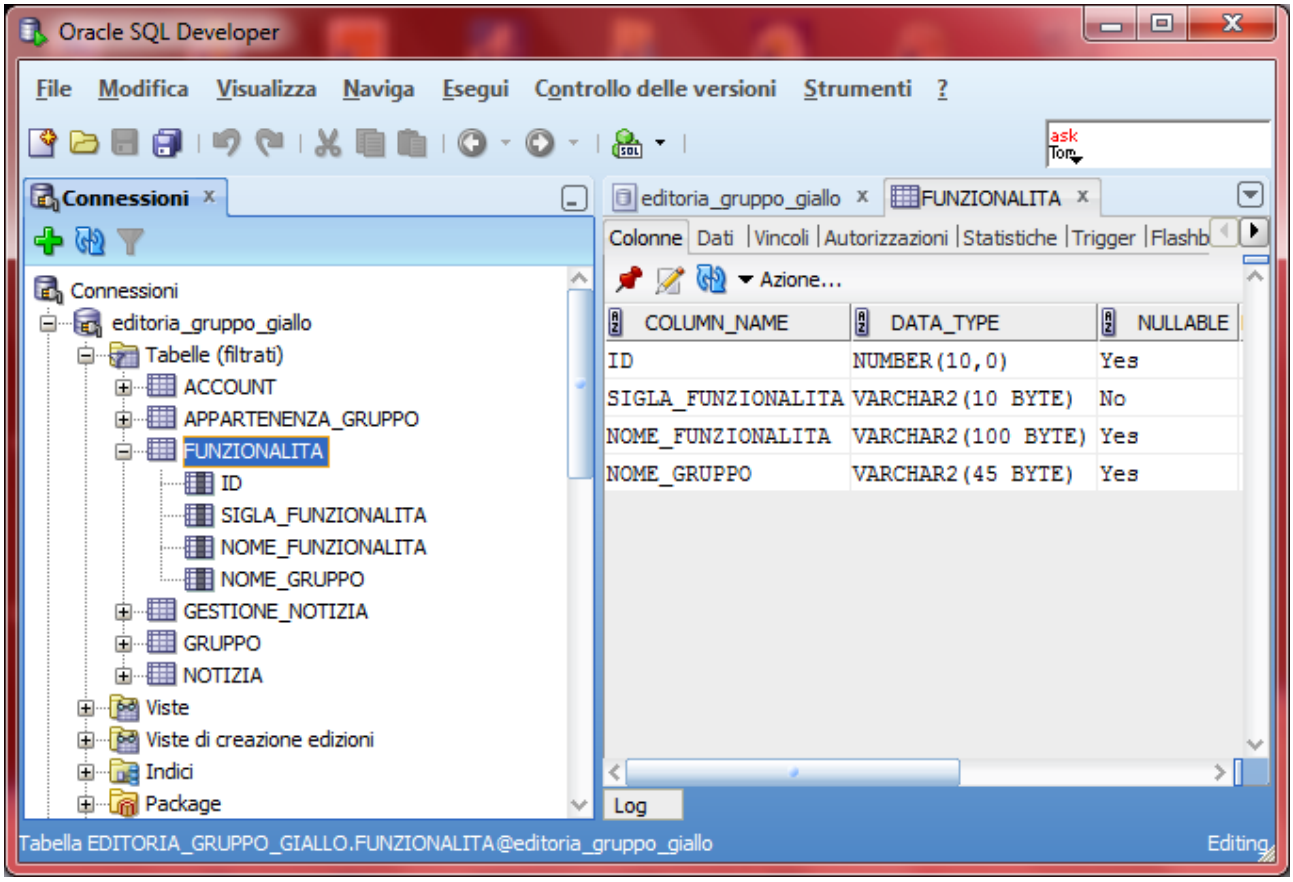

Figura 5 - Screenshot Funzionalita

Gestione_Notizia

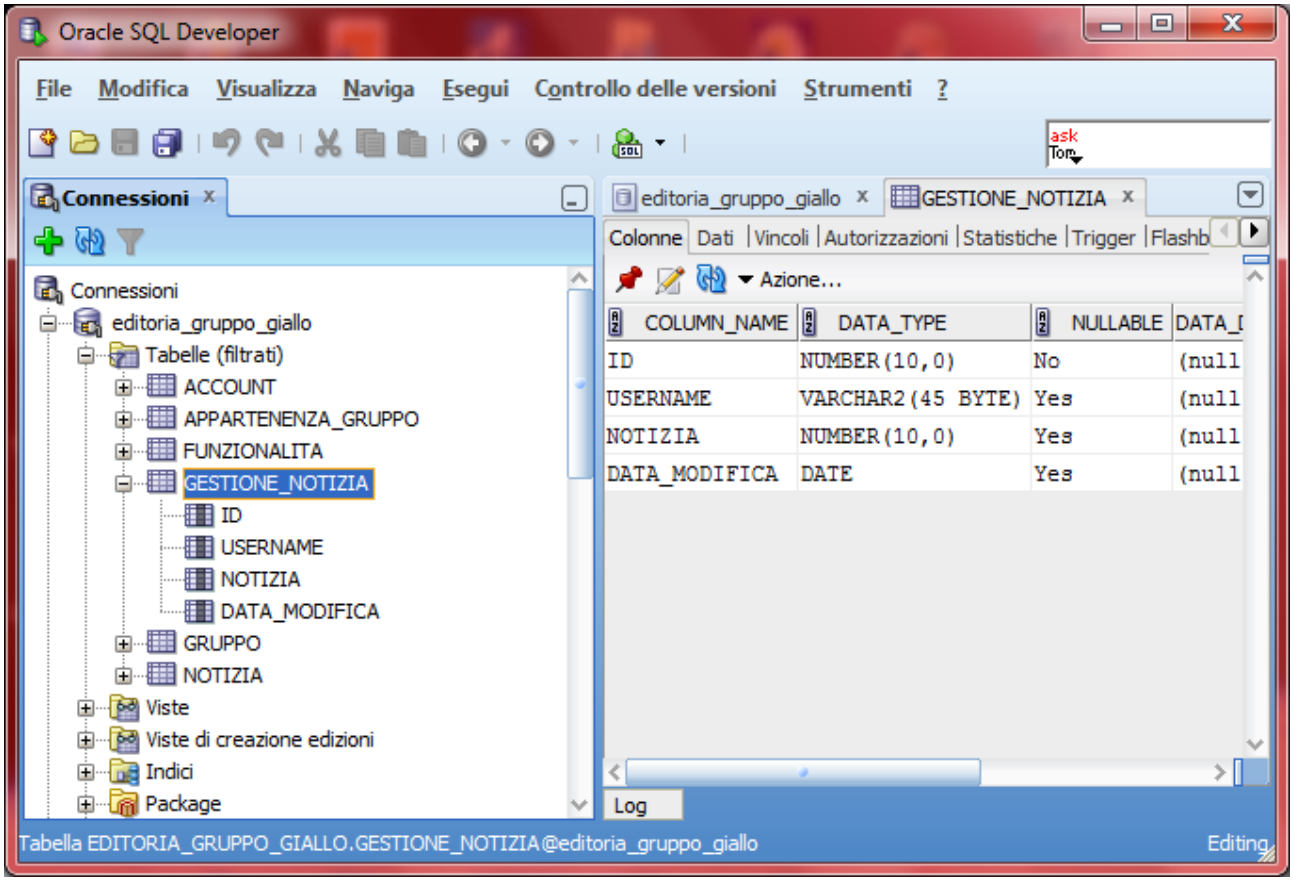

Figura 6 - Screenshot Gestione_Notizia

Gruppo

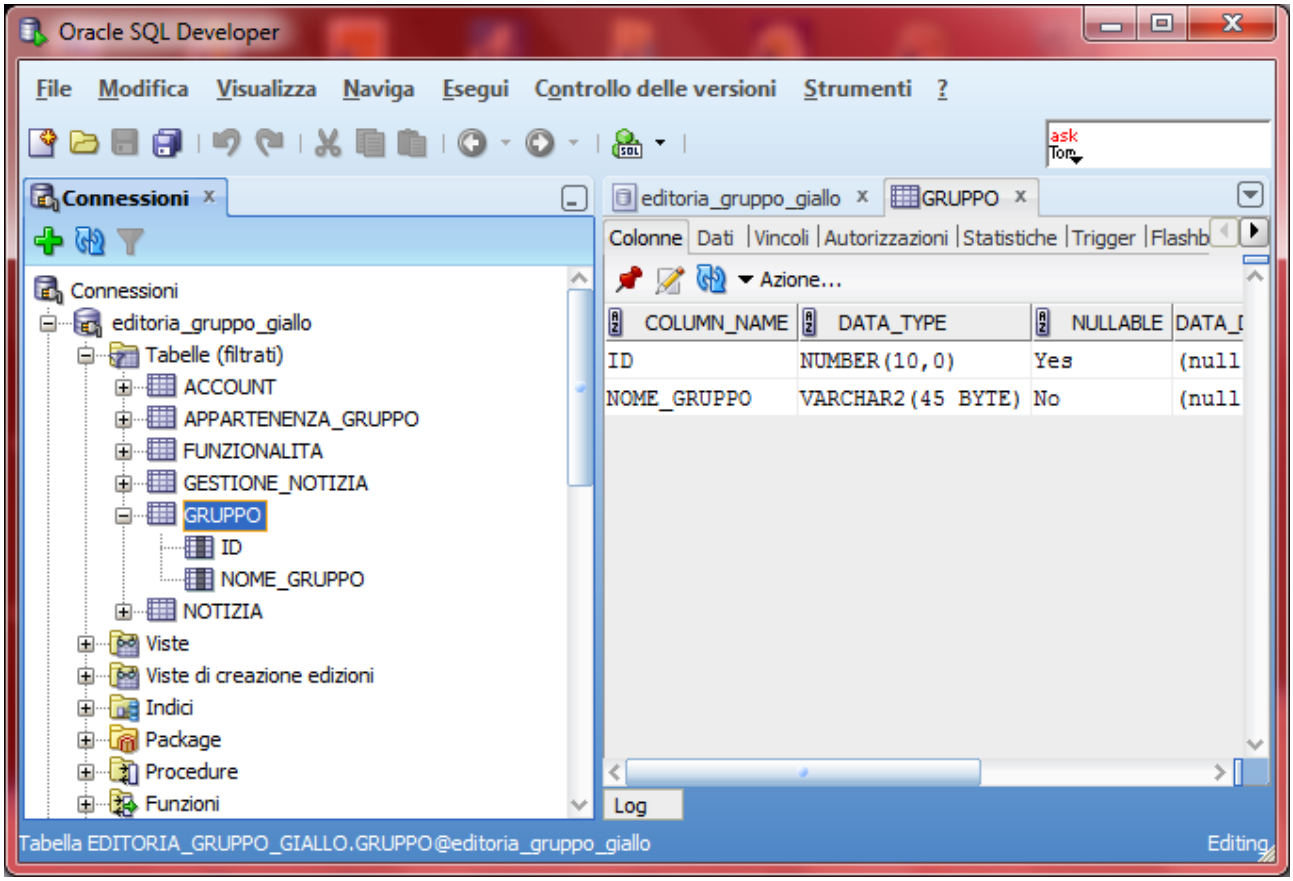

Figura 7 - Screenshot Gruppo

Notizia

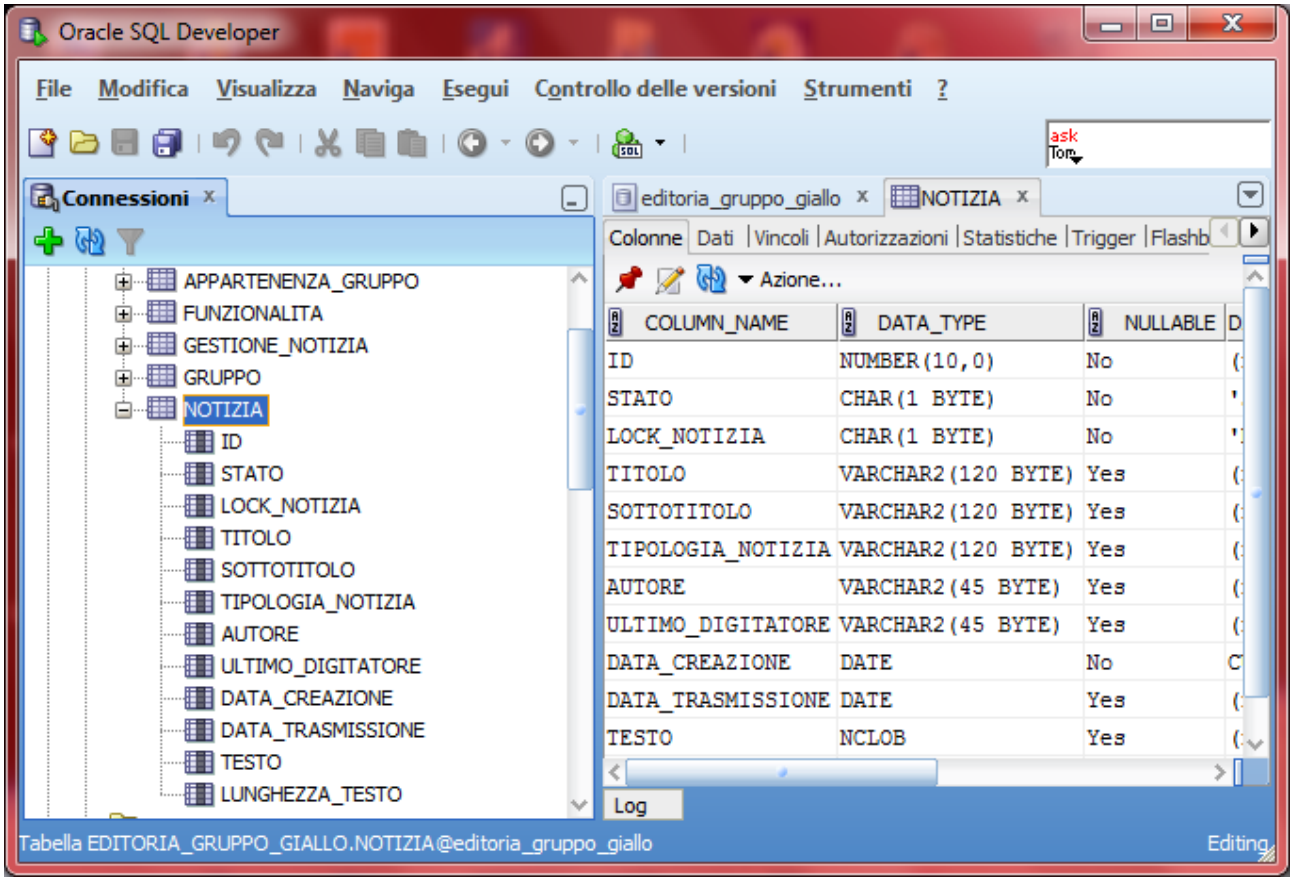

Figura 8 - Screenshot Notizia

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

id	Identificatore
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 inserire manualmente tale campo all'amministratore per permettere la registrazione di due giornalisti con lo stesso nome e cognome al sistema.

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

id	Identificatore
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

4.2.1 Funzionalita

id	Identificatore
sigla_funzionalita	Sigla relativa alla funzionalità, chiave primaria
nome_funzionalita	Nome della funzionalità
nome_gruppo	Attributo riferito al campo nome_gruppo della tabella Gruppo

Tabella 3 - Tabella Funzionalita

4.2.1 Gestione_Notizia

id	Identificatore
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

4.2.2 Gruppo

id	Identificatore
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

4.2.3 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 CLOB 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

In questo paragrafo vengono elencati gli script SQL che permettono la creazione del database, in particolare:

- *creazione utente*: permette di creare un utente per la connessione al database, a cui vengono assegnati tutti i privilegi;
- *creazione sequenza*: consente la creazione delle sequenze che incrementano il campo *id* di ciascuna tabella in maniera sequenziale;
- *trigger:* permette l'incremento della sequenza creata al passo precedente nel momento in cui si genera un nuovo evento di inserimento;
- *creazione tabelle*: permettono la creazione di tutte le tabelle del database;
- *popolamento tabella:* permettono di inserire, in alcune delle tabelle create, dei record predefiniti;
- *cancellazione record tabelle:* consente di cancellare i record all'interno delle tabelle
- *cancellazione tabelle:* consente di cancellare tutte le tabelle dal database.

4.3.1 Crea Utente

```
CREATE USER EDITORIA_GRUPPO_GIALLO IDENTIFIED BY GIALLO;
GRANT ALL PRIVILEGES TO EDITORIA_GRUPPO_GIALLO;
DISCONNECT;
CONNECT EDITORIA_GRUPPO_GIALLO;
```

4.3.2 Creazione tabelle

4.3.2.1 Account

```
CREATE TABLE "EDITORIA_GRUPPO_GIALLO"."ACCOUNT"
  (
    "ID"                NUMBER(10,0),
    "USERNAME"          VARCHAR2(45 BYTE) NOT NULL ENABLE,
    "PASSWORD"          VARCHAR2(45 BYTE) NOT NULL ENABLE,
    "NOME"              VARCHAR2(45 BYTE) NOT NULL ENABLE,
    "COGNOME"           VARCHAR2(45 BYTE) NOT NULL ENABLE,
    "EMAIL"             VARCHAR2(120 BYTE) NOT NULL ENABLE,
    "SIGLA_REDAZIONE"   VARCHAR2(10 BYTE) NOT NULL ENABLE,
    "SIGLA_GIORNALISTA" VARCHAR2(10 BYTE),
     "STATO"    CHAR(1 BYTE) DEFAULT 'A' NOT NULL ENABLE,
     CHECK (stato       ='A'
  OR stato             ='D') ENABLE,
```

```
    CONSTRAINT "ACCOUNT_PK" PRIMARY KEY ("USERNAME"),
    CONSTRAINT "ACCOUNT_UK1" UNIQUE ("ID")
  );
```

4.3.2.2 Gruppo

```
CREATE TABLE "EDITORIA_GRUPPO_GIALLO"."GRUPPO"
  (
    "ID"          NUMBER(10,0),
    "NOME_GRUPPO" VARCHAR2(45 BYTE),
    CONSTRAINT "GRUPPO_PK" PRIMARY KEY ("NOME_GRUPPO"),
    CONSTRAINT "ACCOUNT_UK" UNIQUE ("ID")
  );
```

4.3.2.3 Appartenenza_Gruppo

```
CREATE TABLE "EDITORIA_GRUPPO_GIALLO"."APPARTENENZA_GRUPPO"
  (
    "ID"          NUMBER(10,0),
    "USERNAME"    VARCHAR2(45 BYTE),
    "NOME_GRUPPO" VARCHAR2(45 BYTE),
    CONSTRAINT "APPARTENENZA_GRUPPO_PK" PRIMARY KEY ("USERNAME",
"NOME_GRUPPO"),
    CONSTRAINT "APPARTENENZA_GRUPPO_UK" UNIQUE ("ID"),
    CONSTRAINT "APPARTENENZA_GRUPPO_FK1" FOREIGN KEY ("USERNAME")
REFERENCES "EDITORIA_GRUPPO_GIALLO"."ACCOUNT" ("USERNAME") ON
  DELETE CASCADE ENABLE,
    CONSTRAINT "APPARTENENZA_GRUPPO_FK2" FOREIGN KEY ("NOME_GRUPPO")
REFERENCES "EDITORIA_GRUPPO_GIALLO"."GRUPPO" ("NOME_GRUPPO") ON
  DELETE CASCADE ENABLE
  );
```

4.3.2.4 Funzionalita

```
CREATE TABLE "EDITORIA_GRUPPO_GIALLO"."FUNZIONALITA"
  (
    "ID"                 NUMBER(10,0),
    "SIGLA_FUNZIONALITA" VARCHAR2(10 BYTE),
    "NOME_FUNZIONALITA"  VARCHAR2(100 BYTE),
    "NOME_GRUPPO"        VARCHAR2(45 BYTE),
    CONSTRAINT "FUNZIONALITA_PK" PRIMARY KEY ("SIGLA_FUNZIONALITA"),
    CONSTRAINT "EDITORIA_GRUPPO_GIALLO_UK" UNIQUE ("ID"),
    CONSTRAINT "FUNZIONALITA_FK1" FOREIGN KEY ("NOME_GRUPPO") REFERENCES
"EDITORIA_GRUPPO_GIALLO"."GRUPPO" ("NOME_GRUPPO") ON DELETE CASCADE
ENABLE
```

```
  );
```

4.3.2.1 Notizia

```
CREATE TABLE "EDITORIA_GRUPPO_GIALLO"."NOTIZIA"
  (
    "ID"                NUMBER(10,0),
    "STATO"             CHAR(1 BYTE) DEFAULT 'S' NOT NULL ENABLE,
    "LOCK_NOTIZIA"      CHAR(1 BYTE) DEFAULT 'N' NOT NULL ENABLE,
    "TITOLO"            VARCHAR2(120 BYTE),
    "SOTTOTITOLO"       VARCHAR2(120 BYTE),
    "TIPOLOGIA_NOTIZIA" VARCHAR2(120 BYTE),
    "AUTORE"            VARCHAR2(45 BYTE),
    "ULTIMO_DIGITATORE" VARCHAR2(45 BYTE),
    "DATA_CREAZIONE" DATE DEFAULT CURRENT_TIMESTAMP NOT NULL ENABLE,
    "DATA_TRASMISSIONE" DATE,
    "TESTO" NCLOB,
    "LUNGHEZZA_TESTO" NUMBER(4,0),
    CHECK (stato       ='S'
  OR stato             ='Q'
  OR stato             ='T'
  OR stato             ='C') ENABLE,
    CHECK (lock_notizia='Y'
  OR lock_notizia      ='N') ENABLE,
    CONSTRAINT "NOTIZIA_PK" PRIMARY KEY ("ID"),
    CONSTRAINT "NOTIZIA_FK" FOREIGN KEY ("ULTIMO_DIGITATORE") REFERENCES
"EDITORIA_GRUPPO_GIALLO"."ACCOUNT" ("USERNAME") ENABLE
  );
```

4.3.2.2 Gestione_Notizia

```
CREATE TABLE "EDITORIA_GRUPPO_GIALLO"."GESTIONE_NOTIZIA"
  (
    "ID"                  NUMBER(10,0),
    "USERNAME"    VARCHAR2(45 BYTE),
    "NOTIZIA"                 NUMBER(10,0),
    "DATA_MODIFICA" DATE,
    CONSTRAINT "GESTIONE_NOTIZIA_PK" PRIMARY KEY ("ID"),
    CONSTRAINT "GESTIONE_NOTIZIA_FK1" FOREIGN KEY ("USERNAME") REFERENCES
"EDITORIA_GRUPPO_GIALLO"."ACCOUNT" ("USERNAME") ON DELETE CASCADE ENABLE,
     CONSTRAINT "GESTIONE_NOTIZIA_FK2" FOREIGN KEY ("NOTIZIA")
REFERENCES "EDITORIA_GRUPPO_GIALLO"."NOTIZIA" ("ID") ON DELETE CASCADE
ENABLE
  );
```

4.3.1 Sequence

```
CREATE SEQUENCE "EDITORIA_GRUPPO_GIALLO"."SEQ_ID_ACCOUNT" MINVALUE 0
MAXVALUE 9999999999999999999999999999 INCREMENT BY 1 START WITH 20 CACHE
20 NOORDER NOCYCLE ;

CREATE SEQUENCE "EDITORIA_GRUPPO_GIALLO"."SEQ_ID_GRUPPO" MINVALUE 0
MAXVALUE 9999999999999999999999999999 INCREMENT BY 1 START WITH 0 CACHE
20 NOORDER NOCYCLE ;

CREATE SEQUENCE "EDITORIA_GRUPPO_GIALLO"."SEQ_ID_APPARTENENZA_GRUPPO"
MINVALUE 0 MAXVALUE 9999999999999999999999999999 INCREMENT BY 1 START
WITH 0 CACHE 20 NOORDER NOCYCLE ;

CREATE SEQUENCE "EDITORIA_GRUPPO_GIALLO"."SEQ_ID_FUNZIONALITA" MINVALUE 0
MAXVALUE 9999999999999999999999999999 INCREMENT BY 1 START WITH 0 CACHE
20 NOORDER NOCYCLE ;

CREATE SEQUENCE "EDITORIA_GRUPPO_GIALLO"."SEQ_ID_NOTIZIA" MINVALUE 0
MAXVALUE 9999999999999999999999999999 INCREMENT BY 1 START WITH 0 CACHE
20 NOORDER NOCYCLE ;

CREATE SEQUENCE "EDITORIA_GRUPPO_GIALLO"."SEQ_ID_GESTIONE_NOTIZIA"
MINVALUE 0 MAXVALUE 9999999999999999999999999999 INCREMENT BY 1 START
WITH 0 CACHE 20 NOORDER NOCYCLE ;
```

4.3.2 Trigger

```
create or replace
TRIGGER editoria_gruppo_giallo.trigger_id_account
BEFORE INSERT ON account
FOR EACH ROW
BEGIN
  SELECT seq_id_account.nextval INTO :new.id FROM dual;
END trigger_id_account;
/
create or replace
TRIGGER editoria_gruppo_giallo.trigger_id_gruppo
BEFORE INSERT ON gruppo
FOR EACH ROW
BEGIN
     SELECT seq_id_gruppo.nextval INTO :new.id FROM dual;
END trigger_id_gruppo;
/
```

```
create or replace
TRIGGER editoria_gruppo_giallo.trigger_id_appartenenza_gruppo
BEFORE INSERT ON appartenenza_gruppo
FOR EACH ROW
BEGIN
      SELECT seq_id_appartenenza_gruppo.nextval INTO :new.id FROM dual;
END trigger_id_appartenenza_gruppo;
/
create or replace
TRIGGER editoria_gruppo_giallo.trigger_id_funzionalita
BEFORE INSERT ON funzionalita
FOR EACH ROW
BEGIN
      SELECT seq_id_funzionalita.nextval
      INTO :new.id FROM dual;
END trigger_id_funzionalita;
/
create or replace
TRIGGER editoria_gruppo_giallo.trigger_id_notizia
BEFORE INSERT ON notizia
FOR EACH ROW
BEGIN
      SELECT seq_id_notizia.nextval
      INTO :new.id FROM dual;
END trigger_id_notizia;
/
create or replace
TRIGGER editoria_gruppo_giallo.trigger_id_gestione_notizia
BEFORE INSERT ON gestione_notizia
FOR EACH ROW
BEGIN
      SELECT seq_id_gestione_notizia.nextval
      INTO :new.id FROM dual;
END trigger_id_gestione_notizia;
/
```

4.3.3 Popolamento DB

```
INSERT INTO
editoria_gruppo_giallo.account(id,username,password,nome,cognome,email,si
gla_redazione,sigla_giornalista,stato)
VALUES(seq_id_account.nextval,'admin','admin','Mario','Rossi','admin@edit
oriagruppogiallo.it','admin',NULL,'A');
INSERT INTO
editoria_gruppo_giallo.account(id,username,password,nome,cognome,email,si
```

```
gla_redazione,sigla_giornalista,stato)
VALUES(seq_id_account.nextval,'admin1','admin1','Luca','Verdi','admin1@ed
itoriagruppogiallo.it','admin1',NULL,'A');

INSERT INTO editoria_gruppo_giallo.gruppo (nome_gruppo) VALUES
('AMMINISTRATORE');
INSERT INTO editoria_gruppo_giallo.gruppo (nome_gruppo) VALUES
('GIORNALISTA');

INSERT INTO editoria_gruppo_giallo.appartenenza_gruppo
(id,username,nome_gruppo) VALUES
(seq_id_appartenenza_gruppo.nextval,'admin','AMMINISTRATORE');
INSERT INTO editoria_gruppo_giallo.appartenenza_gruppo
(id,username,nome_gruppo) VALUES
(seq_id_appartenenza_gruppo.nextval,'admin','GIORNALISTA');
INSERT INTO editoria_gruppo_giallo.appartenenza_gruppo
(id,username,nome_gruppo) VALUES
(seq_id_appartenenza_gruppo.nextval,'admin1','AMMINISTRATORE');

INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'CreaAcc','CreaAccount','AMMINISTRATORE');
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'CancAcc','CancellaAccount','AMMINISTRATORE'
);
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'ModAcc','ModificaAccount','AMMINISTRATORE')
;
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'ListaAcc','ListaAccount','AMMINISTRATORE');

INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'CreaNot','CreazioneNotizia','GIORNALISTA');
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'ModNot','ModificaNotizia','GIORNALISTA');
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'RegNot','RegistraNotizia','GIORNALISTA');
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'CancNot','CancellazioneNotizia','GIORNALIST
A');
```

```
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'TrasmNot','TrasmettiNotizia','GIORNALISTA')
;
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'VisNot','VisualizzaNotizia','GIORNALISTA');
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'ListaNot','ListaNotizia','GIORNALISTA');
INSERT INTO editoria_gruppo_giallo.funzionalita
(id,sigla_funzionalita,nome_funzionalita,nome_gruppo) VALUES
(seq_id_funzionalita.nextval,'Ann','Annulla','GIORNALISTA');

commit;
```

4.3.4 Cancellazione record tabelle

```
DELETE FROM editoria_gruppo_giallo.GESTIONE_NOTIZIA;
DELETE FROM editoria_gruppo_giallo.NOTIZIA;
DELETE FROM editoria_gruppo_giallo.FUNZIONALITA;
DELETE FROM editoria_gruppo_giallo.APPARTENENZA_GRUPPO;
DELETE FROM editoria_gruppo_giallo.GRUPPO;
DELETE FROM editoria_gruppo_giallo.ACCOUNT;
```

4.3.5 Cancellazione tabelle

```
DROP TABLE editoria_gruppo_giallo.appartenenza_gruppo;
DROP TABLE editoria_gruppo_giallo.funzionalita;
DROP TABLE editoria_gruppo_giallo.gestione_notizia;
DROP TABLE editoria_gruppo_giallo.notizia;
DROP TABLE editoria_gruppo_giallo.account;
DROP TABLE editoria_gruppo_giallo.gruppo;

DROP SEQUENCE editoria_gruppo_giallo.seq_id_account;
DROP SEQUENCE editoria_gruppo_giallo.seq_id_gruppo;
DROP SEQUENCE editoria_gruppo_giallo.seq_id_appartenenza_gruppo;
DROP SEQUENCE editoria_gruppo_giallo.seq_id_funzionalita;
DROP SEQUENCE editoria_gruppo_giallo.seq_id_notizia;
DROP SEQUENCE editoria_gruppo_giallo.seq_id_gestione_notizia;
```

4.4 Tavola dei volumi

Di seguito viene mostra 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
Funzionalita	Entità	12
Notizia	Entità	5000
Gestione_Notizia	Relazione	15000

Tabella 7 - Tavola dei volumi