UNIVERSITÀ DEGLI STUDI DELL'AQUILA

Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

CORSO DI LAUREA IN INFORMATICA

Insegnamento Laboratorio di Basi di Dati

Simplest Note

Membri del team		
Cognome e nome	**Matricola**	**Ruolo**
Pecce Ivo	251924	Analisi, progettazione concettuale e logica, implementazione in SQL e realizzazione dell'interfaccia in PHP/HTML/JavaScript/CSS

A.A. 2019/2020

Sommario

1 – Formalizzazione ed analisi dei requisiti

1.1 – Descrizione dell'applicazione

L'applicazione da realizzare consiste in un blocco note online, con possibilità di condivisione delle note, revisione delle stesse e creazione di collegamenti tra note.
L'applicazione dovrà essere accessibile solo ad un'utenza registrata. Ciascun utente avrà a disposizione un blocco note virtuale, composto da un numero arbitrario di pagine, che di seguito chiameremo semplicemente note, ognuna dotata di titolo e contenuto. Le singole note saranno associate alla data di creazione e a un numero arbitrario di tag, intesi come parole o brevi sequenze di parole (ad esempio "novità", "lavoro" o "corrispondenza privata").
Il contenuto di ciascuna nota sarà composto da uno o più paragrafi sequenziali. Ciascun paragrafo conterrà del testo, la data di ultima modifica e un riferimento all'ultimo utente che ne ha modificato il contenuto. Il sistema provvederà a salvare tutte le revisioni di ciascun paragrafo, in modo da fornire una storia della stesura delle note. In pratica, ad ogni modifica di un paragrafo ne verrà creata una nuova copia, mentre il paragrafo col contenuto precedente rimarrà nella base di dati come "storia".
Sarà possibile condividere una nota con altri utenti del sistema, fornendo loro accesso in lettura ed eventualmente anche in scrittura.
Infine, sarà possibile creare gerarchie di note, definendo alcune note come sotto-note di altre.

1.1.1 – Glossario dei termini

Viene di seguito definito un glossario dei termini, utile per le prossime fasi della progettazione.
In particolare, il glossario conterrà una breve descrizione dei termini individuati ed eventuali sinonimi e collegamenti con altri termini.

Termine	**Descrizione**	**Sinonimi**	**Collegamenti**
Utente	Utente finale che utilizzerà l'applicazione		
Nota	Ciò che l'utente scriverà e che sarà oggetto di diverse operazioni, tra cui modifica e condivisione con altri utenti	Pagina	Titolo, Contenuto, Paragrafo, Tag
Titolo	Titolo della nota		Nota
Contenuto	Contenuto della nota, diviso in paragrafi		Paragrafo
Paragrafo	Suddivisione interna del contenuto di una nota		Nota, Contenuto
Tag	Breve sequenza di parole che permette di catalogare una nota		Nota
Revisione	Revisione di un paragrafo		Paragrafo

1.1.2 – Ristrutturazione dei requisiti

Frasi di carattere generale
L'applicazione da realizzare consiste in un blocco note online, con possibilità di condivisione delle note, revisione delle stesse e creazione di collegamenti tra note.

Frasi relative agli utenti
L'applicazione dovrà essere accessibile solo ad un'utenza registrata. Ciascun utente avrà a disposizione un blocco note virtuale.
Sarà possibile condividere una nota con altri utenti del sistema, fornendo loro accesso in lettura ed eventualmente anche in scrittura.

Frasi relative alle note
Il blocco appunti virtuale sarà composto da un numero arbitrario di pagine, che di seguito chiameremo semplicemente note, ognuna dotata di titolo e contenuto.
Sarà possibile creare gerarchie di note, definendo alcune note come sotto-note di altre.

Frasi relative al contenuto di una nota
Il contenuto di ciascuna nota sarà composto da uno o più paragrafi sequenziali. Ciascun paragrafo conterrà del testo, la data di ultima modifica e un riferimento all'ultimo utente che ne ha modificato il contenuto. Il sistema provvederà a salvare tutte le revisioni di ciascun paragrafo, in modo da fornire una storia della stesura delle note. In pratica, ad ogni modifica di un paragrafo ne verrà creata una nuova copia, mentre il paragrafo col contenuto precedente rimarrà nella base di dati come "storia".

Frasi relative ai tag di una nota
Le singole note saranno associate alla data di creazione e a un numero arbitrario di tag, intesi come parole o brevi sequenze di parole (ad esempio "novità", "lavoro" o "corrispondenza privata").

1.2 – Operazioni da realizzare

Le operazioni che l'applicazione dovrà realizzare sono riportate di seguito:
1. Inserimento di un utente.
2. Estrazione della data di ultima modifica di una nota.
3. Estrazione della lista delle note presenti nel blocco di un utente con titolo e data ultima modifica.
4. Creazione di una nuova nota nel blocco di un utente (vuota, fornendo solo il titolo).
5. Visualizzazione del testo di una specifica nota.
6. Accodamento di un paragrafo di testo a una nota.
7. Aggiornamento di un paragrafo di testo di una nota da parte di uno specifico utente.
8. Ricerca di una nota in base al testo presente (sottostringa) nel titolo, nel contenuto o nei tag.
9. Concessione a un utente dei privilegi di lettura (o scrittura) su una specifica nota.

10. Impostazione di una nota come sotto-nota di un’altra
11. Visualizzazione di tutte le sotto-note di una data nota
12. Visualizzazione dello storico delle modifiche per un determinato paragrafo di una nota.

1.3 – Raffinamento della specifica

In fase di creazione, viene registrata la data di creazione della nota, per poterla visualizzare nell’applicazione assieme all’utente che l’ha creata.
Tale funzionalità vale anche per i singoli paragrafi, che conterranno anche la data di ultima modifica, in modo da arricchire lo storico della nota.
Viene data la possibilità di cestinare ed archiviare la nota. Le note presenti nel cestino verranno eliminate automaticamente dopo 7 giorni.
Le note potranno essere definite come sotto-note di altre fino ad una gerarchia di livello 3. Ci saranno dunque note di livello 1, sotto-notte di livello 2 e sotto-sotto-note di livello 3.
I livelli di privilegi degli utenti sulle note saranno 3: proprietario, scrittura e lettura. Gli utenti che hanno privilegi di lettura su una nota potranno visualizzarne il contenuto, ma non modificarlo e/o condividere la nota con altri utenti. Tali permessi saranno infatti caratteristici degli utenti che hanno privilegi di scrittura sulle note. L’utente proprietario della nota è colui che l’ha creata, e detiene i privilegi massimi su di essa. Sarà infatti l’unico a poterla eliminare, archiviare e ripristinare.
Infine, in aggiunta a quanto richiesto dalla specifica, verrà implementata la possibilità di aggiungere allegati alla nota.

Secondo quanto riportato sopra, l’applicazione dovrà dare all’utente la possibilità di effettuare le seguenti operazioni aggiuntive rispetto a quanto richiesto nella specifica:

- Possibilità di cestinare la nota e di poterla ripristinare dal cestino
- Possibilità di spostare la nota in archivio e ripristinarla dallo stesso
- Possibilità di aggiungere e rimuovere allegati

Inoltre, l’applicazione dovrà essere in grado di realizzare le seguenti funzioni, non direttamente visibili all’utente finale:

- Registrazione della data di creazione della nota
- Registrazione della data di creazione e modifica del singolo paragrafo
- Registrazione della data di cestinamento di una determinata nota, con possibilità di cancellazione di tale data al momento di un eventuale ripristino da parte dell’utente
- Cancellazione delle note presenti nel cestino da più di 7 giorni
- All’eliminazione di un utente, verranno cancellate tutte le note di cui è proprietario.

2 – Progettazione concettuale

2.1– Modello Entità-Relazioni

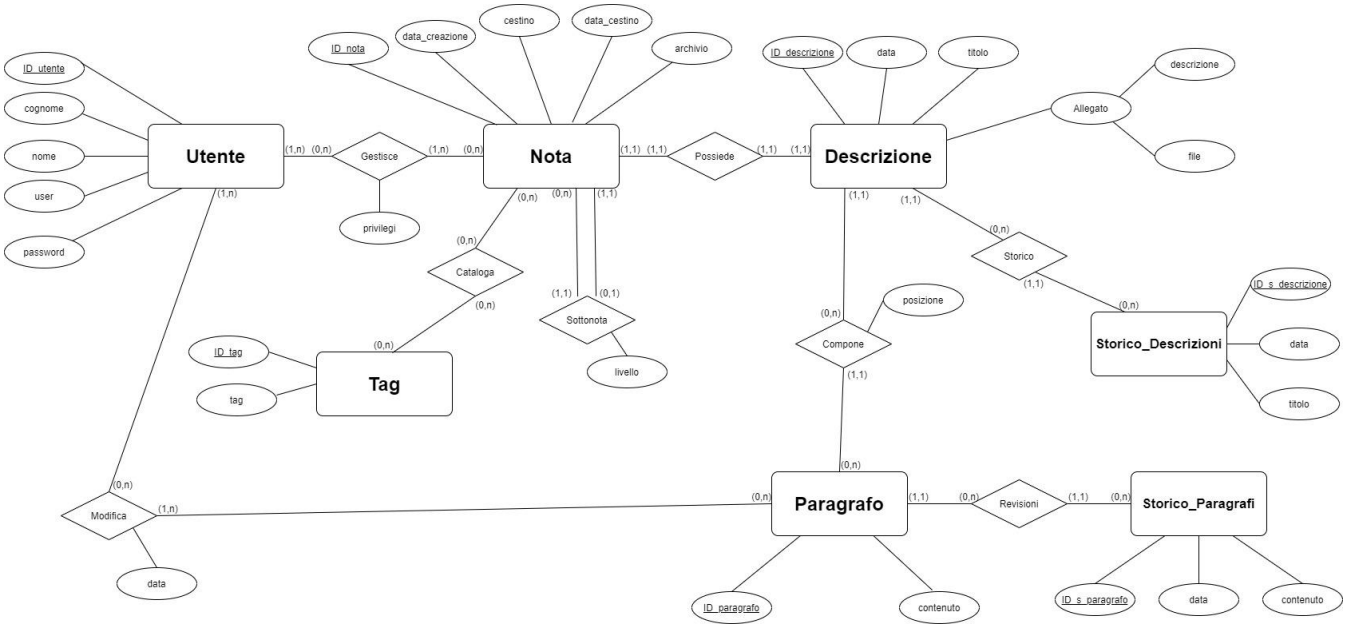
Le entità che risaltano immediatamente all'occhio nella lettura dell'analisi dei requisiti sono:

- Utente: entità che rappresenta gli utenti registrati, composta dai seguenti attributi:
  - ID_utente
  - Cognome
  - Nome
  - User
  - password
- Nota: Entità che rappresenta le note, così composta:
  - ID_nota
  - Titolo
  - Data_creazione
  - Cestino
  - Data_cestino
  - Archivio
  - Allegato, attributo multivalore composto dai seguenti sotto-attributi:
    - Descrizione
    - File
- Paragrafo: Entità che rappresenta i paragrafi, così composta:
  - ID_paragrafo
  - Contenuto
- Tag: Entità rappresentante i tag e così composta:
  - ID_tag
  - tag

Viene inoltre definita un'ulteriore entità, di nome Storico_paragrafi e rappresentante lo storico dei singoli paragrafi. L'entità sarà una copia dell'entità Paragrafo, e i suoi attributi saranno dunque ID_s_paragrafo, data (data di ultima modifica) e contenuto.
Si noti come in tal modo l'entità Nota risulti carica di attributi. Inoltre si vuole fornire la possibilità di poter modificare il titolo della nota e di salvare, come per i paragrafi, lo storico di ogni modifica. Verrà quindi creata una nuova entità chiamata Descrizione, legata all'entità Nota con associazione uno a uno, e avente come attributi ID_descrizione, titolo e allegato.
L'entità Nota diventerà quindi così composta:

- ID_nota
- Data_creazione
- Cestino
- Data_cestino
- Archivio

Il modello E/R risultante sarà il seguente:



(Lo schema in formato JPG e drawio viene allegato alla documentazione)

Un utente può non gestire o gestire più note. Una nota viene gestita da almeno un utente, con determinati privilegi su di essa.
Una nota può essere catalogata o meno in più tag. Un tag può catalogare o meno più note.
Una nota può avere più sotto-note, ognuna ad un determinato livello. Una sotto-nota ha solo una nota superiore.
Una nota possiede una descrizione. Una descrizione è relativa ad una sola nota.
Una descrizione può avere più revisioni in uno storico. Una revisione in uno storico descrizioni è relativa ad una descrizione.
Una descrizione può essere composta o meno da più paragrafi, situati in una determinata posizione nella pagina. Un paragrafo compone una sola descrizione.
Analogamente per lo storico descrizioni, un paragrafo può avere più revisioni in uno storico. Una revisione in uno storico paragrafi è relativa ad un paragrafo.
Un utente può modificare in una data uno o più paragrafi. Un paragrafo è modificato da almeno un utente.

2.2– Formalizzazione dei vincoli

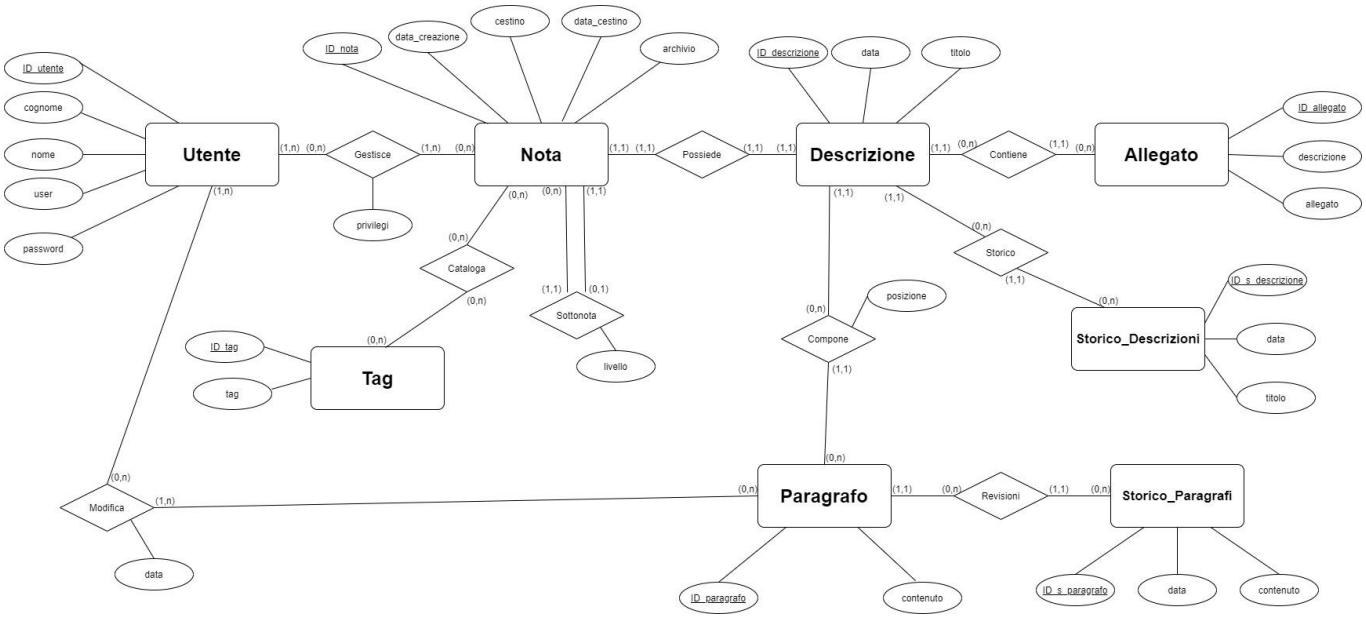
In questo paragrafo si formalizzano alcuni vincoli sulla base di dati, non esprimibili nel modello E/R. Alcuni di questi vincoli sono già stati formalizzati nell'analisi dei requisiti, ma vengono di seguito ripetuti per averne una trattazione completa.
Primo di tutti, un utente può modificare solo note su cui è proprietario o ha permessi di scrittura. Utenti con permessi di lettura possono solamente visualizzare le note. Inoltre, solo gli utenti proprietari di una nota possono archiviarla e/o cestinarla.
Una volta creata, la nota può non avere paragrafi, ma deve necessariamente avere un titolo.

Gli allegati non vengono salvati nello storico, per non appesantire la base di dati nel corso del tempo. Inoltre, gli allegati possono essere solamente inseriti o eliminati, ma non modificati.
Come già espresso in precedenza, le note possono avere sotto-note di livello massimo pari a 3.
All'eliminazione di una nota, verrà eliminata la descrizione e gli eventuali paragrafi con lo storico relativo, oltre che gli eventuali allegati. Analogo vincolo viene imposto per i paragrafi.
All'eliminazione di un utente, vengono eliminate anche le note di cui è proprietario.

3 – Progettazione logica

3.1– Ristrutturazione del modello Entità-Relazioni

Procediamo ora con la progettazione logica del database.
La prima fase consiste nella ristrutturazione del modello E/R illustrato in precedenza.
L'unica ristrutturazione che si evince riguarda l'attributo *allegato* dell'entità Descrizione. Tale attributo, essendo composto, diventerà un'entità associata da relazione uno a molti con l'entità Descrizione. In questo modo viene data anche la possibilità di aggiungere più di un allegato ad una nota.
Lo schema E/R ristrutturato e pronto per essere trasformato nel relativo modello relazionale è il seguente:



(Lo schema in formato JPG e drawio viene allegato alla documentazione)

3.2– Progettazione del modello relazionale

Procediamo ora con la traduzione del modello E/R illustrato sopra nel relativo modello relazionale.

Tabelle derivate da entità:
Utente (<u>ID_utente</u>, cognome, nome, user, password)
Nota (<u>ID_nota</u>, data_creazione, cestino, archivio, data_cestino, ID_descrizione, ID_superiore, livello)
Descrizione (<u>ID_descrizione</u>, data, titolo)
Storico_Descrizioni (<u>ID_s_descrizione</u>, data, titolo, ID_descrizione)
Allegato (<u>ID_allegato</u>, descrizione, allegato, ID_descrizione)
Paragrafo (<u>ID_paragrafo</u>, contenuto, ID_descrizione, posizione)
Storico_Paragrafi (<u>ID_s_paragrafo</u>, data, contenuto, ID_descrizione, posizione, ID_paragrafo)
Tag (<u>ID_tag</u>, tag)

Tabelle derivate da relazioni:
Gestisce (ID_gestisce, ID_utente, ID_nota, privilegi)
Modifica (ID_modifica, ID_utente, ID_paragrafo, data)
Cataloga (ID_cataloga, ID_nota, ID_tag)

Nel modello relazionale sono già state inserite le chiavi esterne. Si faccia attenzione alla chiave esterna ID_superiore della relazione Nota, che fa riferimento alla chiave primaria della nota superiore, in caso sia stata impostata come sotto-nota.
Si noti inoltre che, pur essendoci nel modello E/R un'associazione uno a uno tra le entità Nota e Descrizione, queste ultime sono state lasciate separate nel modello relazionale, per i motivi riportati nel paragrafo 2.1. Viene quindi inserita una chiave esterna ID_descrizione nella relazione Nota, facente riferimento alla descrizione e per la quale verranno stabiliti nell'implementazione tramite SQL dei vincoli di integrità referenziale. Inoltre su questa chiave esterna verranno applicati dei vincoli di unicità, in modo da non permettere che la stessa descrizione faccia riferimento a due note diverse.
Tra tutte le chiavi esterne vengono imposti dei vincoli di integrità referenziale con le relazioni a cui fanno riferimento.
Per finire, è stato aggiunto un attributo di nome *data* alla relazione Storico_paragrafi, facente riferimento alla data di modifica del paragrafo presente della relazione Modifica. Il contenuto di tale attributo verrà copiato al momento della modifica di un paragrafo, quando ne verrà salvato lo storico.

4 – Implementazione tramite SQL

4.1– Allegati alla documentazione

In allegato alla presente documentazione, è possibile trovare l'implementazione del modello relazionale sopra descritto nel linguaggio SQL. In particolare, è stato utilizzato il DBMS MySQL, con il quale è stato effettuato il dump del database, disponibile sia in una versione integrale, comprendente tabelle, dati, trigger, eventi, procedure e funzioni, sia scomposto in più file separati, ognuno contenente una parte realizzata.
Le query richieste dalla specifica sono allegate alla presente sotto forma di script SQL, ognuno identificato con il numero di query richiesta dalla specifica. In tali script sarà presente solamente una chiamata alla procedura o funzione a cui fa riferimento, riportando dei dati come esempio.
Ulteriori funzioni realizzate sono state salvate solamente come funzioni o procedure, senza riportare la query chiamante di esempio.

4.2– Realizzazione dei vincoli

I vincoli formalizzati in precedenza sono stati inseriti nell'implementazione in SQL.
In particolare, il vincolo riguardante i permessi su una nota viene implementato specificando nelle chiamate di procedure e funzioni anche l'ID_utente, in modo che si possa verificare i privilegi dell'utente su una nota (tramite costrutti IF-THEN-ELSE) ed impedire eventuali azioni non concesse. Ad esempio, nella chiamata della procedura *cestina_nota*, viene richiesto, oltre l'ID della nota, anche quello dell'utente in modo da verificare se ha i permessi per cestinare una nota.
Sulle chiavi esterne sono stati impostati dei vincoli di integrità referenziale tramite delle COSTRAINT. Inoltre è stata specificata su ognuna la clausola ON UPDATE CASCADE ON DELETE CASCADE.
All'eliminazione di un utente vengono eliminate tutte le note di cui è proprietario tramite un'apposita procedura, che utilizza un cursore che scorre tutte le note in cui è proprietario e richiama per ognuna la procedura di eliminazione della nota.

4.3– Ulteriori note sull'implementazione in SQL

Il salvataggio dello storico di una descrizione o di un paragrafo è stato implementato per mezzo di appositi trigger.
È stato inoltre impostato un trigger in fase di aggiornamento della relazione Nota, che verifica se l'attributo booleano *cestino* è stato posto ad 1. In caso affermativo aggiorna l'attributo

data_cestino, contenente la data in cui la nota è stata spostata nel cestino. Se invece l'attributo *cestino* è posto a 0, l'attributo *data_cestino* viene posto a NULL.
È stato creato sul database uno script con eseguito ricorrenza giornaliera che verifica se la nota è presente nel cestino da oltre sette giorni. In caso affermativo lo script provvede ad eliminare la nota chiamando la procedura apposita.

5 – Realizzazione dell'interfaccia grafica

Al progetto è stata aggiunta un'interfaccia grafica realizzata tramite pagina web, sviluppata nei linguaggi PHP, HTML, JavaScript e CSS.
Il codice sorgente è presente tra gli allegati alla documentazione.

5.1– Funzionalità realizzate in PHP

Le funzionalità realizzate in PHP comprendono:

- Registrazione ed eliminazione di un utente
- Aggiunta ed eliminazione di una nota
- Aggiunta dei paragrafi in coda ad una nota
- Modifica dei singoli paragrafi
- Eliminazione di un paragrafo (momentaneamente realizzata lasciando vuoto il campo del paragrafo. In caso di eliminazione del testo e riscrittura sullo stesso campo l'applicativo interpreterà tale azione come modifica del paragrafo)
- Lista delle note con i relativi privilegi dell'utente, data di ultima modifica e testo della nota
- L'interfaccia grafica impedisce modifiche su note sulle quali l'utente non ha privilegi di scrittura.

5.2– Funzionalità non ancora realizzate e sviluppi futuri

Al momento sono presenti alcune funzionalità richieste dalla specifica e non realizzate in PHP (ma già realizzate in SQL), che saranno perciò aggiunte in versioni future dell'applicazione.

Tra le funzionalità ancora da realizzare vi sono:

- Possibilità di condivisione di una nota con altri utenti
- Possibilità di impostare una nota come sotto-nota di un'altra
- Ricerca di una nota in base al contenuto, titolo o tag
- Possibilità di catalogare una nota tramite dei tag
- Possibilità di aggiungere ed eliminare allegati
- Visualizzazione dello storico delle modifiche di una nota
- Gestione del cestino e dell'archivio delle note
- Visualizzazione delle sotto-note di una determinata nota
- Possibilità di spostare i paragrafi all'interno di una nota

Inoltre dovrà essere effettuata un’operazione di pulizia e ristrutturazione, assieme ad un restyling dell’interfaccia grafica.

6 – Scheda sintetica

Nome del progetto: Simplest Note versione 1.0

Autori:

Nome	Cognome	Matricola	Ruolo nello sviluppo del progetto
Ivo	Pecce	251924	Analisi, progettazione concettuale e logica, implementazione in SQL e realizzazione dell'interfaccia in PHP/HTML/JavaScript/CSS

Data di consegna del progetto: 15/06/2020