Database Exercise System

By Stefano Borzì & Alessandro Maggio

Sommario

Come usare il *Database Exercise System*

Lato Utente

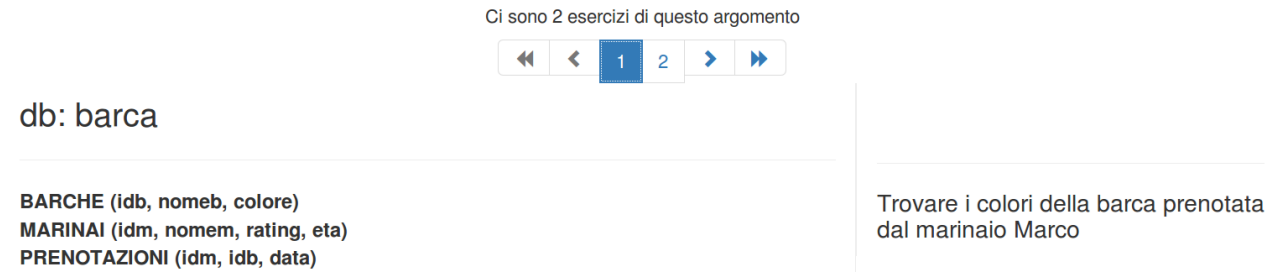
Database Exercise System *è stato creato per gli studenti e per facilitare lo studio della materia di database, pertanto l'interfaccia utente si presenta in maniera semplice ed intuitiva.*

La prima cosa da fare è scegliere la tipologia di esercitazione da affrontare, ovvero *SQL* o *Algebra Relazionale*, ciò è possibile tramite i due pulsanti posti nell'header del sito.

Una volta effettuata la scelta il sistema di esercitazione mostrerà una lista degli argomenti disponibili.

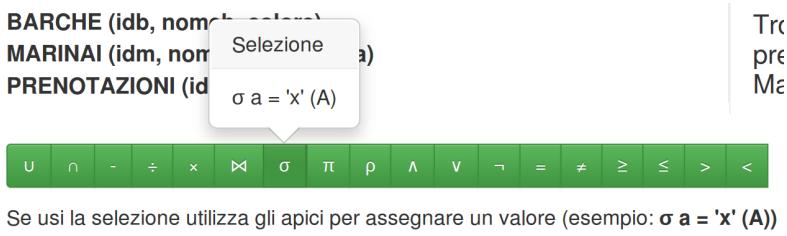
Dopo aver selezionato l'argomento sulla quale esercitarsi verrà mostrato il numero di domande presenti e un selettore che permetterà di selezionare quale esercizio svolgere.

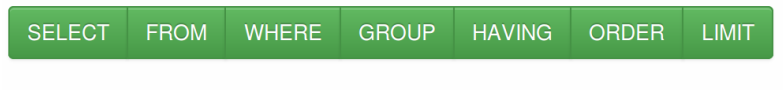
A seguire nella parte sinistra della pagina vi sono le informazioni relative al database e lo schema relazionale delle varie tabelle. Nella parte destra invece il testo della domanda.



In base alla tipologia di esercizio scelto verrà mostrata una toolbar caratteristica che aiuterà l'utente nella creazione della risposta. Ogni pulsate è associato a un pop-up che spiegherà come utilizzare al meglio quel determinato simbolo e/o parola chiave.

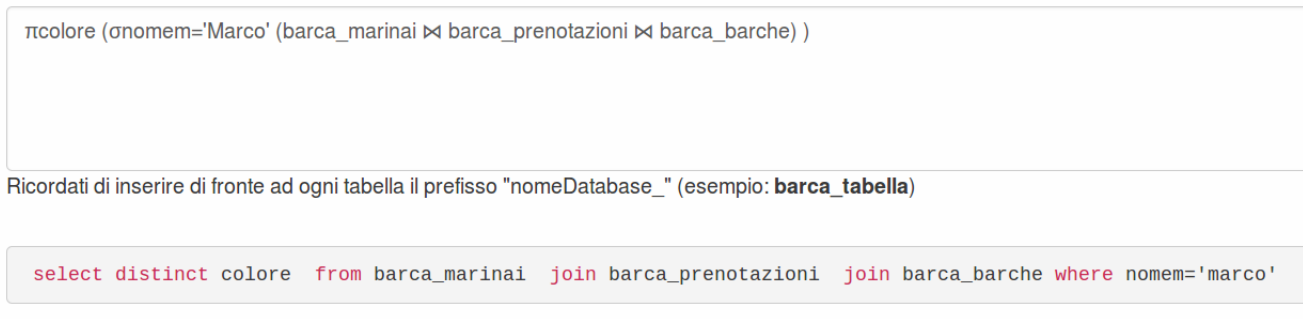
La toolbar dell'algebra relazionale contiene ad esempio tutti gli operatori più comuni.





Discorso analogo vale per l'SQL.

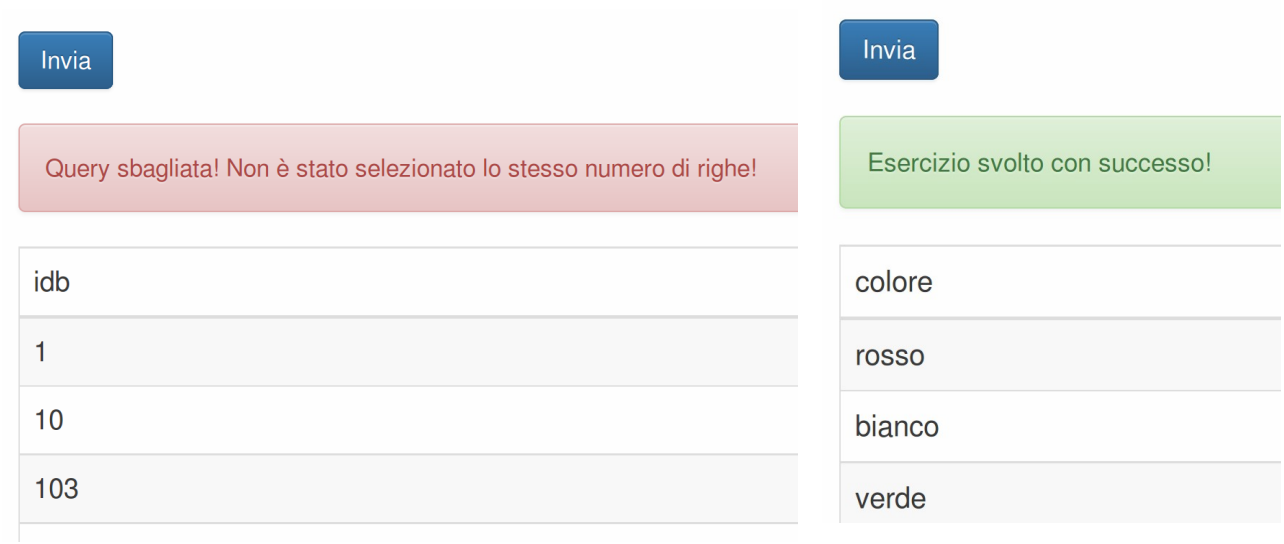
L'input-box posto al di sotto delle varie toolbar è dedicato alla trascrizione della risposta. L'utente deve far attenzione a inserire di fronte ad ogni tabella il prefisso "nomeDatabase_" (esempio: *banca_tabella*). Nel caso dell'algebra relazionale inoltre è necessario l'utilizzo degli apici qualora si debba assegnare un valore come ad esempio durante la selezione *σ a = 'x' (A)*.



Altra caratteristica del sistema di esercitazione dedicato all'algebra relazionale è la

possibilità di visualizzare nel riquadro sottostante la query scritta convertita in SQL.

Una volta che lo studente è convinto della soluzione inserita può testarla tramite il pulsante *Invia* che provvederà a eseguire la query e mostrarne il risultato.

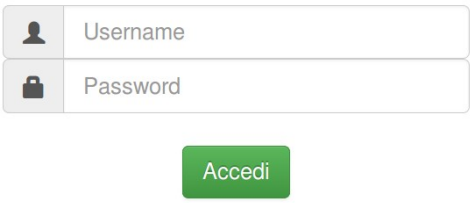


Qualora lo studente non riesca a trovare una risposta alla domanda selezionata ed è curioso di conoscerla può farlo tramite il pulsante *Mostra soluzione*.

Lato Admin

La parte amministrativa di questo sistema d'esercitazione è strutturata principalmente in due pagine raggiungibili dai pulsanti rossi presenti nell'header.

Entrambe le pagine sono protette da *user-name* e *password* per evitare che utenti non autorizzati apportino modifiche ai database presenti e/o alle domande, categorie etc.



Il pannello di login contiene semplicemente un pulsante di conferma e due *input-box* che richiedono i dati per l'autenticazione.

La prima pagina che esamineremo sarà quella chiamata "*dbmanager*" la quale permetterà all'amministratore di creare dei nuovi database fittizi.

Le prime informazioni richieste sono il *nome del database* e il *numero di tabelle* presenti in esso.

La modifica del secondo *input-box* (con valore maggiore e/o uguale a uno) mostrerà automaticamente lo schema di creazione delle singole tabelle, esso conterrà nella sua parte sinistra tutte le proprietà modificabili della tabella in questione e nella parte destra visualizzerà dinamicamente la tabella stessa.

In particolare le proprietà richieste per ogni singola tabella sono:

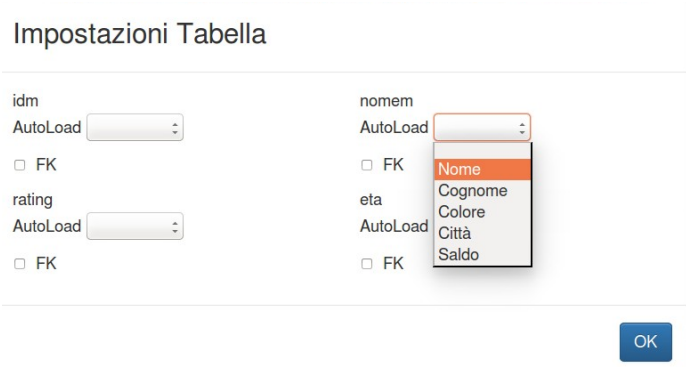
- *Nome tabella.*
- *Numero di attributi (colonne).*
- *Nomi dei singoli attributi.*
- *Numero di righe.*

Prendendo come esempio la tabella "*Marinaio*" e popolando tutti i campi vedremo a

video l'esempio sottostante.

Per ogni tabella inoltre sono presenti delle piccole impostazione raggiungibili dal pulsante (rappresentato dalla chiave inglese) posto prima della scritta "*Tabella numero n*".

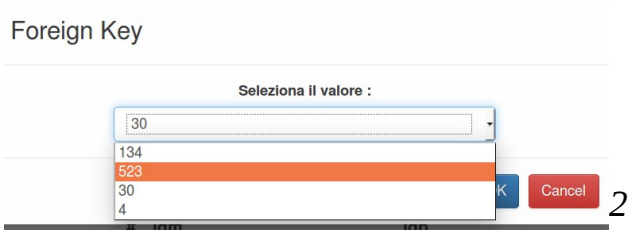
Questo mini-pannello permette "*l'autoload*" e l'abilitazione di *chiave esterne*.



L'*autoload* non fa altro che aiutare l'amministratore nel popolare automaticamente i vari campi utilizzando valori predefiniti contenuti nelle varie categorie selezionabili.

Se la chiave esterna è stata abilitata dal pannello di impostazioni vedremo che tutti gli *input-box* del campo in questione si disabiliteranno e diventeranno attivi i vari pulsanti *FK* che permetteranno di definire le varie chiavi esterne.

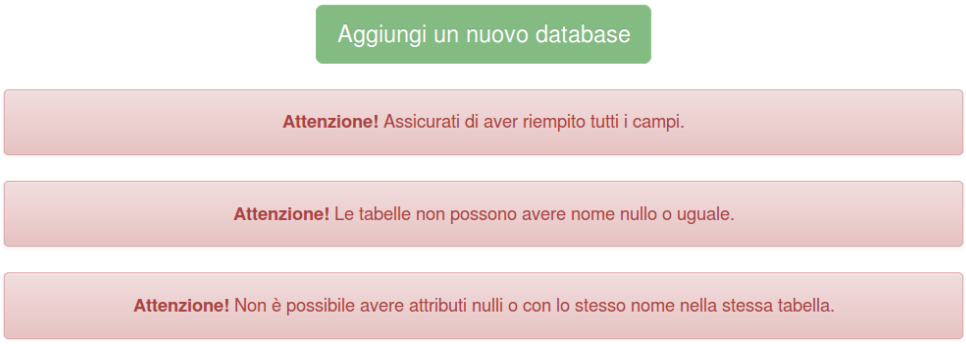
Se la chiave esterna non è mai definita alla pressione del tasto *FK* si aprirà un pannello che permetterà di selezionare la *tabella*, l'*attributo* e il *valore* da collegare.



2

Se invece il campo ha già un attributo e una tabella di riferimento verrà chiesto solo il valore.

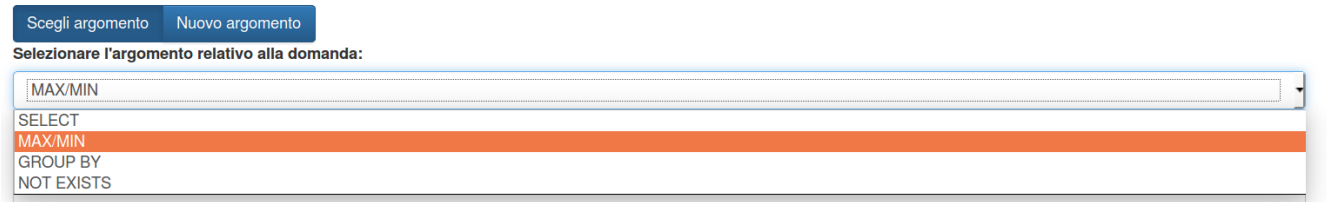
L'utente amministratore inoltre sarà aiutato dagli *alert* che controlleranno la corretta creazione del nuovo database.
Per confermare il tutto è necessario cliccare sul tasto "*Aggiungi un nuovo database*".



La seconda pagina di amministrazione invece permette di inserire nuove domande e/o aggiungere nuovi argomenti e soluzioni.

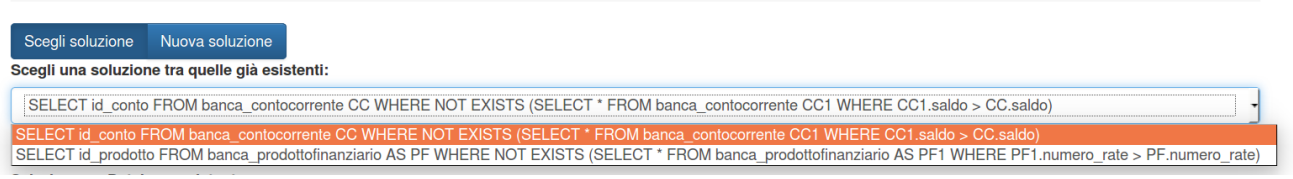
La prima informazione richiesta è il tipo di domanda (SQL o Algebra Relazionale) selezionabile tramite i radio-button.

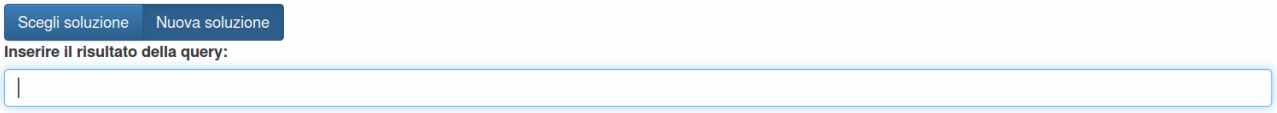
Successivamente l'amministratore dovrà decidere se la nuova domanda apparterà a un argomento già esistente (selezionabile dal menù a tendina) oppure se creare una nuova categoria di argomenti.



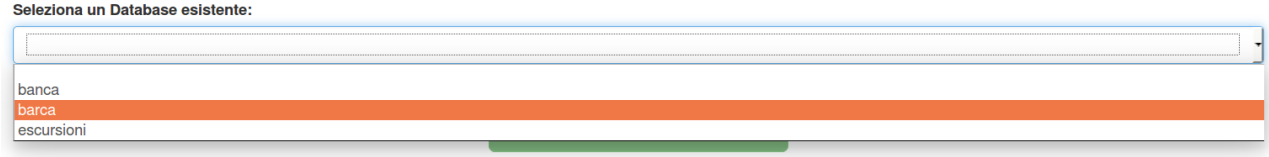
L'informazione consecutiva che viene richiesta è il *testo della domanda*.

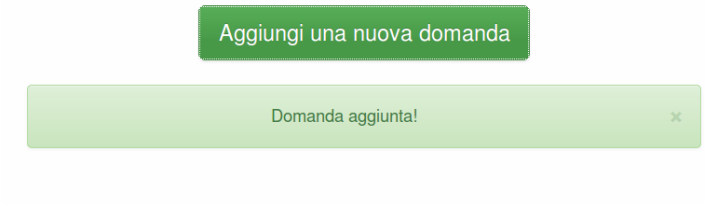
A seguire l'admin si ritroverà d'inanzi a un'altra scelta ovvero quella della soluzione, anche qui proprio come per gli argomenti potrà sceglierne se inserirne una nuova (scritta in SQL per entrambe le tipologie) oppure selezionarne una tra quelle esistenti.





Infine prima di inoltrare la nuova domanda sarà necessario selezionare un database tra quelli creati prima.





Anche qui come per il *“dbmanager”* il tasto di conferma è situato in fondo alla pagina nella quale sono presenti anche alert relativi a errori e/o operazioni completate con successo.

Come funziona

Il *Database Exercise System* è una **single-page-app** ovvero un'applicazione web che una volta visitata la pagina web essa carica una sola volta tutte le librerie poiché quando apparentemente si cambia pagina (es. si passa da “SQL” a “Algebra Relazionale”) viene caricata solo la struttura HTML di quella pagina (viene caricato un file HTML nella pagina) e non tutte le librerie CSS/JS.

Per rendere il *Database Exercise System* una **single-page-app** è stato utilizzato il seguente framework javascript **AngularJS** mentre per la grafica è stato utilizzato il framework CSS **Bootstrap.**

Core del progetto

Nel progetto si possono distinguere 3 file principali che compongono il “core” del progetto, ovvero:

- **app.js & routes.js**

- **API.php & APIadmin.php**

app.js & route.js

Essi contengono tutto il codice Javascript responsabile della gestione dei dati degli esercizi e dell'interazione con l'utente.

App.js gestisce (la parte **frontend**):

- la conversione dell'algebra relazionale in SQL
- invio della query SQL/ALG creata dall'utente alla API
- autenticazione (tramite API)
- DataBase Manager
- Inserimento di nuovi esercizi e soluzioni

API.php & APIadmin.php

API & APIadmin sono responsabili della parte **backend** del sito, ovvero dell'interazione con il database quindi:

- confronto della query dell'utente con la query soluzione di un esercizio
- creazione di nuovi esercizi e soluzioni
- creazione di nuovi “database” relativi agli esercizi

- autenticazione e gestione della sessione dell'utente-admin che ha accesso al pannello di controllo

Partials

Tutti i files all'interno di partials costituiscono tutte le strutture HTML delle varie pagine all'interno del sistema di esercitazione che vengono man mano caricati tramite Javascript.

Analisi dei requisiti

Dominio del problema

Nel dipartimento di Matematica e Informatica (DMI) gli studenti che seguono la materia Database non utilizzano e/o non hanno uno strumento adatto per esercitarsi con l' Algebra Relazionale e l' SQL.
A tal proposito diversi studenti come progetto della materia hanno intenzione di creare uno strumento di esercitazione che colmi questa mancanza.
Questo sistema di esercitazione dovrà però, fornire oltre che ad un'interfaccia per gli studenti dove poter eseguire le proprie query e visualizzare il risultato e la soluzione dell'esercizio (ed il risultato della soluzione-query dell'esercizio) anche un pannello di amministrazione in modo da poter far aggiornare al docente gli esercizi del sistema di esercitazione.

Operazioni

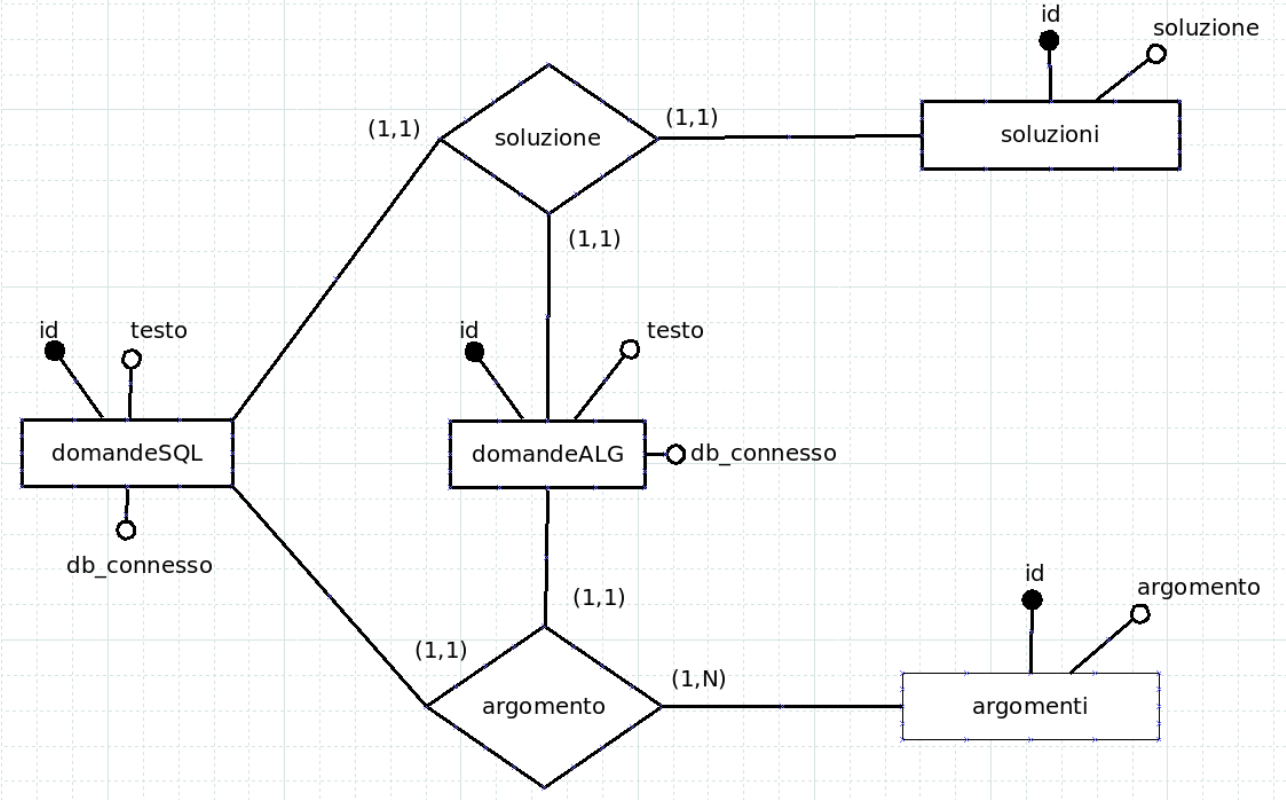
1. Carico di nuovi esercizi e nuovi database relativi agli esercizi (6-10 volte all'anno)
2. Query “SELECT” degli esercizi (frequenza arbitraria a seconda del numero degli studenti che utilizzano il sistema)

Glossario

Termine	Descrizione	Collegamenti
Docente	Utente che utilizzerà il pannello di amministrazione per poter aggiungere esercizi e “database” relativi agli esercizi	DomandeSQL/ALG, soluzioni, argomenti, *possibilità di inserire nuovi “database”*
Studente	Utente che svolgerà gli esercizi caricati nel database	Database relativi agli esercizi

Progettazione concettuale

Schema ER



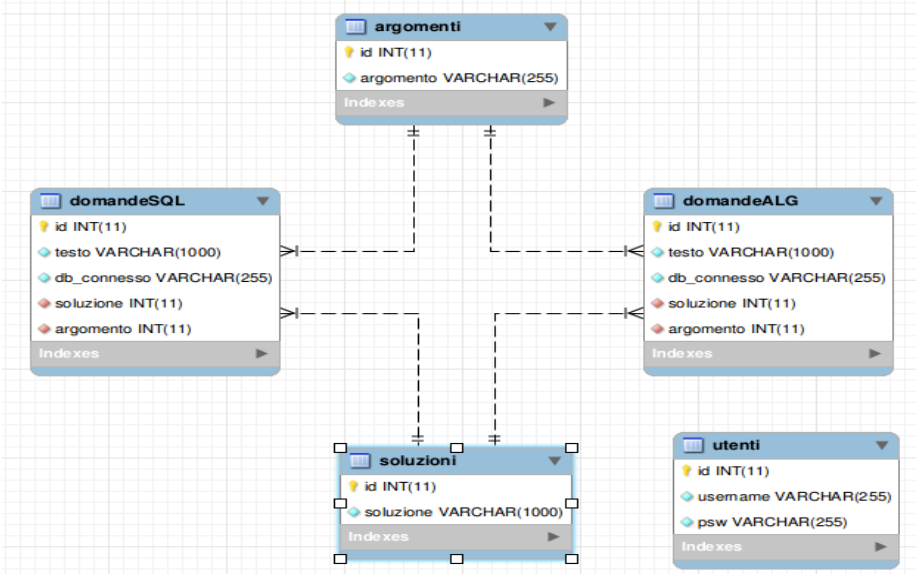
Progettazione logica

Schema relazionale

soluzioni (id, soluzione)
argomenti(id, argomento)
domandeSQL (id, testo, db_connesso, soluzione, argomento)
domandeALG (id, testo, db_connesso, soluzione, argomento)

utenti(id, username, psw)

Modello relazionale



Progettazione Fisica

```
-- Create 'argomenti' table
CREATE TABLE IF NOT EXISTS argomenti(
    id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    argomento varchar(255) NOT NULL
);

-- Create 'soluzioni' table
CREATE TABLE IF NOT EXISTS soluzioni(
    id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    soluzione varchar(1000) NOT NULL
);

-- Create 'domandeALG' table
CREATE TABLE IF NOT EXISTS domandeALG(
    id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    testo varchar(1000) NOT NULL,
    db_connesso varchar(255) NOT NULL,
    soluzione INT NOT NULL,
    argomento INT NOT NULL,
    FOREIGN KEY (soluzione) REFERENCES soluzioni(id),
    FOREIGN KEY (argomento) REFERENCES argomenti(id)
);
-- Create 'domandeSQL' table
CREATE TABLE IF NOT EXISTS domandeSQL(
    id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    testo varchar(1000) NOT NULL,
    db_connesso varchar(255) NOT NULL,
    soluzione INT NOT NULL,
    argomento INT NOT NULL,
    FOREIGN KEY (soluzione) REFERENCES soluzioni(id),
    FOREIGN KEY (argomento) REFERENCES argomenti(id)
);

CREATE TABLE IF NOT EXISTS utenti(
    id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
    username varchar(255) NOT NULL,
    psw varchar(255) NOT NULL
);
```