**RELAZIONE PROGETTO BASI DI DATI 2023/2024**

Progetto realizzato da:

* Luis Popa 914802

**1. Raccolta/Analisi dei requisiti (testo completo delle specifica sui dati, lista delle operazioni, tavola media dei volumi, glossario dei dati):**

**- SPECIFICHE SUI DATI:**

La piattaforma ESQL è progettata per gestire utenti, test, quesiti, tabelle di esercizio, risposte e comunicazioni tra docenti e studenti, tramite un database relazionale chiamato ESQLDB.

**1. Utenti**

Gli utenti della piattaforma sono divisi in due tipologie: docenti e studenti. I dati relativi a ciascun utente sono organizzati come segue:

* Dati comuni a tutti gli utenti:
  + Email: Indirizzo email univoco che funge da identificatore.
  + Nome: Il nome dell'utente.
  + Cognome: Il cognome dell'utente.
  + Recapito telefonico (opzionale): Numero di telefono del singolo utente.
* Dati specifici dei docenti:
  + Nome del dipartimento: Il nome del dipartimento di afferenza del docente.
  + Nome del corso: Il nome del corso di cui il docente è titolare.
* Dati specifici degli studenti:
  + Anno di immatricolazione: L'anno in cui lo studente si è immatricolato.
  + Codice studente: Un codice alfanumerico univoco di 16 caratteri che identifica lo studente.

**2. Tabelle di Esercizio**

I docenti hanno la possibilità di creare delle tabelle di esercizio all'interno della piattaforma. Ogni tabella di esercizio contiene i seguenti campi:

* Nome: Il nome della tabella di esercizio.
* Data di creazione: La data in cui la tabella è stata creata.
* Numero di righe: Il numero di righe che la tabella è progettata per contenere.
* Attributi: Ogni tabella di esercizio ha un insieme di attributi, definiti dai docenti. Ogni attributo contiene:
  + Nome: Il nome dell'attributo.
  + Tipo: Il tipo di dato (es. integer, varchar, ecc.).
  + Parte della chiave primaria (booleano): Indica se l'attributo fa parte della chiave primaria della tabella.
* Vincoli di integrità: I docenti possono impostare vincoli di integrità tra gli attributi di diverse tabelle di esercizio per garantire la consistenza dei dati.

**3. Test**

I docenti possono creare test che gli studenti possono svolgere. Ogni test dispone dei seguenti dati:

* Titolo univoco: Il titolo del test, che deve essere unico all'interno della piattaforma.
* Data di creazione: La data in cui il test è stato creato.
* Foto (opzionale): Una foto o immagine associata al test.

Ogni test può includere una serie di quesiti. I quesiti all'interno di un test sono numerati progressivamente e ciascun quesito ha i seguenti dati:

* Numero progressivo: Numero che identifica univocamente il quesito all'interno di un test specifico.
* Livello di difficoltà: Un campo enumerativo (*enum*) che può assumere i valori "Basso", "Medio" o "Alto".
* Descrizione: Una descrizione testuale del quesito.
* #num\_risposte: Il numero di risposte attese per il quesito (ridondante).
* Tabelle di riferimento: Uno o più riferimenti a tabelle di esercizio create dal docente che sono utilizzate nel quesito.

I quesiti possono essere di due tipi:

1. Quesiti a risposta chiusa:
   * Un quesito di questo tipo include una serie di opzioni di risposta, ciascuna delle quali contiene:
     + Numerazione: Un numero univoco che identifica l’opzione all'interno del quesito.
     + Testo: Il testo dell’opzione di risposta.
2. Quesiti di codice:
   * Un quesito di questo tipo richiede una o più soluzioni sotto forma di sketch di codice SQL, che implementano quanto richiesto dalla descrizione del quesito.

**4. Risposte degli Studenti**

Gli studenti possono rispondere ai quesiti del test e le risposte variano a seconda del tipo di quesito:

* Quesiti a risposta chiusa: Lo studente seleziona una delle opzioni di risposta disponibili.
* Quesiti di codice: Lo studente fornisce una soluzione sotto forma di codice SQL.

Ogni risposta ha un esito che indica se è corretta o meno:

* Esito: Un campo booleano che può assumere i valori:
  + True: La risposta è corretta (nel caso di quesiti a risposta chiusa, coincide con l'opzione giusta; nel caso di quesiti di codice, la soluzione produce lo stesso output di quella del docente).
  + False: La risposta è errata.

Gli studenti possono fornire più risposte per lo stesso quesito, in momenti diversi, ma solo se il test non è stato contrassegnato come Concluso.

**5. Completamento del Test**

La piattaforma tiene traccia dello stato di avanzamento dei test. Per ciascun test, vengono registrate le seguenti informazioni:

* Data di inserimento della prima risposta: Data e ora in cui lo studente inserisce la sua prima risposta nel test.
* Data di inserimento dell'ultima risposta: Data e ora in cui lo studente inserisce la sua ultima risposta nel test.
* Stato: Lo stato del test, che può assumere uno dei seguenti valori (*enum*):
  + Aperto: Il test è stato avviato ma lo studente non ha inserito alcuna risposta.
  + InCompletamento: Il test è stato iniziato ma non ancora concluso.
  + Concluso: Lo studente ha completato il test.

**6. Messaggi**

La piattaforma permette l'invio di messaggi tra docenti e studenti. I dati relativi ai messaggi includono:

* Titolo: Il titolo del messaggio.
* Testo: Il contenuto testuale del messaggio.
* Data di inserimento: La data e ora in cui il messaggio è stato inserito.
* Test di riferimento: Il test a cui si riferisce il messaggio.
* Messaggi inviati dai docenti: Se il messaggio è inviato da un docente, viene ricevuto da tutti gli studenti che partecipano a quel test.
* Messaggi inviati dagli studenti: Se il messaggio è inviato da uno studente, il destinatario è uno specifico docente.

**7. Visualizzazione delle Risposte**

Ogni test dispone di un campo booleano VisualizzaRisposte. Se settato a True, le risposte corrette ai quesiti sono visibili agli studenti. Se impostato a False, le risposte restano nascoste

**- LISTA DELLE OPERAZIONI:**

**Operazioni che riguardano tutti gli utenti (docenti e studenti):**

1. Autenticazione/registrazione sulla piattaforma:
   * Ogni utente può registrarsi o autenticarsi per accedere alle funzionalità della piattaforma.
2. Visualizzazione dei test disponibili:
   * Gli utenti possono visualizzare l’elenco dei test che sono stati creati e resi disponibili dai docenti.
3. Visualizzazione dei quesiti all'interno di ciascun test:
   * Gli utenti possono accedere ai quesiti di ogni test

**Operazioni che riguardano SOLO i docenti:**

1. Inserimento di una nuova tabella di esercizio con i relativi meta-dati:
   * I docenti possono creare nuove tabelle di esercizio, specificando nome, data di creazione, numero di righe e attributi.
2. Inserimento di una riga in una tabella di esercizio:
   * I docenti possono aggiungere nuove righe alle tabelle di esercizio esistenti.
3. Creazione di un nuovo test:
   * I docenti possono creare nuovi test, specificando titolo, data di creazione e foto
4. Creazione di un nuovo quesito con relative risposte:
   * I docenti possono creare nuovi quesiti per i test, specificando il livello di difficoltà, descrizione e risposte (opzioni o soluzioni di codice).
5. Abilitazione/disabilitazione della visualizzazione delle risposte per un test:
   * I docenti possono decidere se rendere visibili o meno le risposte dei quesiti ai propri studenti tramite il campo VisualizzaRisposte.
6. Inserimento di un messaggio:
   * I docenti possono inviare messaggi associati a un test, che vengono ricevuti da tutti gli studenti della piattaforma.

**Operazioni che riguardano SOLO gli studenti:**

1. Inserimento di una risposta a un quesito:
   * Gli studenti possono rispondere ai quesiti all'interno dei test, inserendo una o più risposte (opzioni multiple o soluzioni di codice) fino alla conclusione del test.
2. Visualizzazione dell’esito della risposta:
   * Gli studenti possono visualizzare se la risposta che hanno inserito è corretta (True) o errata (False), in base alla corrispondenza con la soluzione del docente.
3. Inserimento di un messaggio:
   * Gli studenti possono inviare un messaggio a uno specifico docente in riferimento a un test.

**Operazioni di Statistiche (visibili da tutti gli utenti):**

1. Visualizzare la classifica degli studenti in base al numero di test completati:
   * Classifica degli studenti, ordinata in base al numero di test completati, senza mostrare dati sensibili (nome, cognome, email), ma solo il codice alfanumerico dello studente.
2. Visualizzare la classifica degli studenti in base al numero di risposte corrette:
   * Classifica degli studenti basata sul rapporto tra risposte corrette e risposte totali inserite, anche in questo caso senza mostrare dati sensibili, ma solo il codice alfanumerico dello studente.
3. Visualizzare la classifica dei quesiti in base al numero di risposte inserite:
   * Statistiche sui quesiti, ordinati in base al numero di risposte ricevute dagli studenti.

**- TAVOLA MEDIA DEI VOLUMI** :

| **Elemento** | **Descrizione** | **Stima Media** |
| --- | --- | --- |
| Utenti Totali | Numero totale di utenti (Docenti + Studenti) | 20 |
| Docenti | Numero totale di docenti | 4 |
| Studenti | Numero totale di studenti | 16 |
| Test | Numero totale di test creati | 10 |
| Quesiti | Numero totale di quesiti per test | 20 |
| Opzioni di Risposta | Numero totale di opzioni di risposta per quesiti chiusi | 100 |
| Soluzioni di Codice | Numero totale di soluzioni di codice per quesiti di codice | 20 |
| Tabelle di Esercizio | Numero totale di tabelle di esercizio | 5 |
| Attributi delle Tabelle | Numero totale di attributi per tabelle di esercizio | 25 |
| Risposte | Numero totale di risposte inserite dagli studenti | 200 |
| Risposta Chiusa | Numero totale di risposte chiuse | 100 |
| Risposta Aperta | Numero totale di risposte aperte | 100 |
| Messaggi | Numero totale di messaggi inviati | 40 |
| Messaggi per Studenti | Numero totale di messaggi inviati da studenti | 20 |
| Messaggi per Docenti | Numero totale di messaggi inviati da docenti | 20 |
| Completamenti di Test | Numero totale di completamenti di test da parte degli studenti | 80 |
| Soluzioni Risposta Chiusa | Numero totale di soluzioni per quesiti a risposta chiusa | 100 |

**Note**

1. **Utenti Totali**: Il numero totale di utenti è una somma dei docenti e studenti.
2. **Docenti e Studenti**: La distribuzione è stimata come 4 docenti e 16 studenti.
3. **Test**: Ogni docente potrebbe creare mediamente 2-3 test, per un totale di circa 10 test.
4. **Quesiti**: Ogni test contiene mediamente 2 quesiti in media, per un totale di 20 quesiti.
5. **Opzioni di Risposta**: I quesiti a risposta chiusa hanno mediamente 5 opzioni ciascuno.
6. **Soluzioni di Codice**: Ogni quesito di codice ha una o più soluzioni.
7. **Tabelle di Esercizio e Attributi**: Stimiamo che ogni docente crei mediamente 1-2 tabelle di esercizio con circa 5 attributi ciascuna.
8. **Risposte**: Gli studenti inseriscono mediamente 10 risposte ciascuno.
9. **Risposta Chiusa e Aperta**: Stimiamo un numero uguale di risposte chiuse e aperte per semplicità.
10. **Messaggi**: Consideriamo una comunicazione moderata tra docenti e studenti.
11. **Completamenti di Test**: Ogni studente completa mediamente 5 test.
12. **Soluzioni Risposta Chiusa**: Correlati al numero totale di opzioni di risposta.

Questa tavola è pensata per fornire una panoramica completa dei volumi stimati del sistema ESQL.

**-Glossario dei Dati:**

| **Termine** | **Descrizione** | **Collegamenti** |
| --- | --- | --- |
| **Utente** | **Entità principale che rappresenta qualsiasi persona iscritta alla piattaforma.** | **Docente, Studente** |
| **Docente** | **Utente con il ruolo di insegnante nella piattaforma.** | **Utente, Test, Tabella di Esercizio** |
| **Studente** | **Utente con il ruolo di studente nella piattaforma.** | **Utente, Risposta, Completamento Test** |
| **Test** | **Prova creata da un docente che contiene uno o più quesiti.** | **Docente, Quesito, Completamento Test** |
| **Quesito** | **Domanda all'interno di un test.** | **Test, Opzione di Risposta, Soluzione Codice** |
| **Opzione di Risposta** | **Una delle risposte possibili per i quesiti a risposta chiusa.** | **Quesito, Risposta Chiusa** |
| **Soluzione di Codice** | **Codice SQL che risolve un quesito di tipo Codice.** | **Quesito** |
| **Tabella di Esercizio** | **Tabelle SQL create dai docenti per scopi didattici.** | **Docente, Attributo della Tabella** |
| **Attributo della Tabella** | **Colonna di una tabella di esercizio, definita da un nome e un tipo di dato.** | **Tabella di Esercizio** |
| **Risposta** | **Risposta fornita da uno studente a un quesito.** | **Quesito, Risposta Chiusa, Risposta Aperta** |
| **Risposta Chiusa** | **Risposta a quesiti a risposta chiusa, dove lo studente sceglie tra opzioni predefinite.** | **Risposta, Opzione di Risposta** |
| **Risposta Aperta** | **Risposta a quesiti a risposta aperta, dove lo studente inserisce una risposta testuale libera.** | **Risposta** |
| **Messaggio** | **Comunicazione inviata all'interno della piattaforma tra utenti.** | **Test, Messaggio Studente** |
| **Messaggio Studente** | **Messaggi inviati dagli studenti ai docenti.** | **Messaggio, Docente** |
| **Completamento Test** | **Informazioni sullo stato di completamento di un test da parte di uno studente.** | **Studente, Test** |
| **Soluzione Risposta Chiusa** | **Soluzione corretta per quesiti a risposta chiusa, indicata dall'opzione scelta.** | **Quesito, Risposta Chiusa** |

**2) Progettazione Concettuale (diagramma E-R, dizionario delle entità/relazioni, tavola delle business rules).**

**Immagine che contiene testo, diagramma, mappa, Piano

Descrizione generata automaticamente**

\*correzione cardinalità Tabella di esercizio – quesito: (1,1) – (1,n)

**DIZIONARIO DELLE ENTITA’:**

| ENTITÀ | DESCRIZIONE | ATTRIBUTI | IDENTIFICATORI |
| --- | --- | --- | --- |
| Utente | Rappresenta una persona che utilizza la piattaforma, può essere uno studente o un docente. | email password nome cognome recapito\_telefonico tipo\_utente (Docente, Studente) | email (PK) |
| Docente | Specializzazione di Utente, rappresenta un docente. | email nome\_dipartimento nome\_corso | email (PK) |
| Studente | Specializzazione di Utente, rappresenta uno studente. | email anno\_immatricolazione codice\_alfanumerico | email (PK) |
| Test | Insieme di quesiti creato da un docente. | id\_test titolo data\_creazione foto email\_docente VisualizzaRisposte | id\_test (PK) |
| Quesito | Domanda associata a un test, con un numero progressivo, livello di difficoltà, descrizione e categoria. | id\_quesito id\_test numero\_progressivo livello\_difficolta descrizione num\_risposte categoria | id\_quesito (PK) |
| OpzioneRisposta | Opzione di risposta per i quesiti a risposta chiusa. | id\_opzione id\_quesito numerazione testo | id\_opzione (PK) |
| SoluzioneCodice | Sketch di codice SQL che risponde a un quesito di codice. | id\_soluzione id\_quesito sketch\_codice | id\_soluzione (PK) |
| TabellaDiEsercizio | Tabella SQL creata da un docente per esercizi. | id\_tabella nome data\_creazione num\_righe email\_docente id\_quesito | id\_tabella (PK) |
| AttributoTabella | Colonna di una tabella di esercizio, con il tipo e indicazione se è parte della chiave primaria. | id\_attributo id\_tabella nome\_attributo tipo chiave\_primaria | id\_attributo (PK) |
| Risposta | Risposta data da uno studente a un quesito. | id\_risposta id\_quesito email\_studente data\_risposta esito | id\_risposta (PK) |
| RispostaChiusa | Risposta che rappresenta la selezione di un'opzione in un quesito a risposta chiusa. | id\_risposta id\_opzione | id\_risposta (PK) |
| RispostaAperta | Risposta testuale per un quesito di codice. | id\_risposta testo\_risposta | id\_risposta (PK) |
| Messaggio | Comunicazione nella piattaforma associata a un test, inviata da un docente o studente. | id\_messaggio titolo testo data\_inserimento id\_test tipo\_mittente | id\_messaggio (PK) |
| Messaggio Docente | Messaggio inviato da docente a tutti gli studenti inerente ad un test | id\_messaggio | Id\_messaggio(PK) |
| Messaggio Studente | Messaggio inerente a test inviato da studente ad un docente | Id\_messaggio  Email\_docente\_destinatario | Id\_messaggio |
| CompletamentoTest | Stato di completamento di un test da parte di uno studente, con date e stato. | email\_studente id\_test data\_inserimento\_prima\_risposta data\_inserimento\_ultima\_risposta stato\_del\_completamento | email\_studente (PK), id\_test (PK) |
| SoluzioneRispostaChiusa | Soluzione corretta per un quesito a risposta chiusa. | id\_quesito numerazioneOpzione | id\_quesito (PK), numerazioneOpzione (PK) |

**-DIZIONARIO DELLE RELAZIONI** :

| **RELAZIONE** | **DESCRIZIONE** | **COMPONENTI** |
| --- | --- | --- |
| Docente-Crea-Test | Un docente può creare più test. Ogni test è creato da un solo docente. | Docente, Test |
| Test-Contiene-Quesito | Un test può contenere più quesiti. Ogni quesito è associato a un solo test. | Test, Quesito |
| Quesito-Ha-OpzioneRisposta | Un quesito a risposta chiusa può avere più opzioni di risposta. Ogni opzione appartiene a un solo quesito. | Quesito, OpzioneRisposta |
| Quesito-Ha-SoluzioneCodice | Un quesito di codice può avere più soluzioni. Ogni soluzione appartiene a un solo quesito. | Quesito, SoluzioneCodice |
| Docente-Crea-TabellaDiEsercizio | Un docente può creare più tabelle di esercizio. Ogni tabella di esercizio è creata da un solo docente. | Docente, TabellaDiEsercizio |
| TabellaDiEsercizio-Ha-AttributoTabella | Una tabella di esercizio può avere più attributi. Ogni attributo appartiene a una sola tabella di esercizio. | TabellaDiEsercizio, AttributoTabella |
| Quesito-Riferito-A-TabellaDiEsercizio | Un quesito può riferirsi a una o più tabelle di esercizio. Ogni tabella è assocciata ad un solo quesito | Quesito, TabellaDiEsercizio |
| Studente-Dà-Risposta | Uno studente può dare più risposte. Ogni risposta è data da un solo studente. | Studente, Risposta |
| Quesito-Ha-Risposta | Un quesito può avere più risposte dagli studenti. Ogni risposta è associata a un solo quesito. | Quesito, Risposta |
| Risposta-Ha-RispostaChiusa | Una risposta chiusa è un'estensione di una risposta generale. Ogni risposta chiusa è una sola risposta. | Risposta, RispostaChiusa |
| Risposta-Ha-RispostaAperta | Una risposta aperta è un'estensione di una risposta generale. Ogni risposta aperta è una sola risposta. | Risposta, RispostaAperta |
| Test-Ha-Messaggio | Un test può avere più messaggi associati. Ogni messaggio è associato a un solo test. | Test, Messaggio |
| Studente-Completa-Test | Uno studente può completare più test. Ogni completamento è associato a uno studente e un test specifico. | Studente, CompletamentoTest |
| Test-Completa-Studente | Un test può essere completato da più studenti. Ogni completamento è associato a un solo test. | Test, CompletamentoTest |
| Quesito-Ha-SoluzioneRispostaChiusa | Un quesito a risposta chiusa può avere una soluzione corretta specifica per ogni opzione. Ogni opzione è legata a un quesito specifico. | Quesito, SoluzioneRispostaChiusa |
| AttributoTabella può contenere Vincolo Integrità | Un attributo può essere vincolo di integrità tra più di una tabella. | AttributoTabella, AttributoTabella |

**-BUSINESS RULES** :

REGOLE DI VINCOLO SUI DATI DELL’APPLICAZIONE :

1. **Utenti**
   * **Email**: Deve essere unica per ogni utente nella tabella Utente.
   * **Tipo Utente**: Ogni utente deve avere un valore per il campo tipo\_utente, che può essere 'Docente' o 'Studente'.
2. **Docenti**
   * **Email**: Deve corrispondere a una email esistente nella tabella Utente e deve essere unica nella tabella Docente.
   * **Nome Dipartimento**: Non può essere nullo.
   * **Nome Corso**: Non può essere nullo.
3. **Studenti**
   * **Email**: Deve corrispondere a una email esistente nella tabella Utente e deve essere unica nella tabella Studente.
   * **Anno Immatricolazione**: Deve essere una data valida (anno).
   * **Codice Alfanumerico**: Deve essere un codice di esattamente 16 caratteri.
4. **Tabelle di Esercizio**
   * **Nome**: Deve essere unico all’interno della stessa tabella.
   * **Data Creazione**: Deve essere una data valida.
   * **Num Righe**: Deve essere un numero intero non negativo.
5. **Attributi delle Tabelle di Esercizio**
   * **Nome Attributo**: Deve essere unico all'interno della stessa tabella.
   * **Tipo**: Deve essere un tipo valido per gli attributi.
   * **Chiave Primaria**: Deve essere un valore booleano che indica se l'attributo è parte della chiave primaria.
6. **Vincoli di Integrità tra Tabelle di Esercizio**
   * **Vincoli**: Devono essere definiti per garantire la coerenza referenziale tra attributi di tabelle diverse.
7. **Test**
   * **Titolo**: Deve essere unico per ogni test nella tabella Test.
   * **Data Creazione**: Deve essere una data valida.
   * **Foto**: Il percorso del file o URL può essere nullo.
   * **VisualizzaRisposte**: Campo booleano che determina se le risposte devono essere visibili agli studenti.
8. **Quesiti**
   * **Numero Progressivo**: Deve essere unico all'interno di ogni test.
   * **idQuesito**: E’ un intero autoincrementale
   * **Livello Difficoltà**: Deve essere uno dei valori predefiniti ('Basso', 'Medio', 'Alto').
   * **Descrizione**: Deve essere un testo non nullo.
   * **Num Risposte**: Deve indicare il numero di risposte possibili; può essere ridondante.
9. **Opzioni di Risposta**
   * **Numerazione**: Deve essere unica all'interno dello stesso quesito.
   * **Testo**: Deve essere un testo non nullo.
10. **Soluzioni di Codice**
    * **Sketch Codice**: Deve essere un testo non nullo che contiene il codice SQL.
11. **Risposte**
    * **Esito**: Deve essere un valore booleano che indica se la risposta è corretta.
12. **Risposte Chiuse**
    * **Id Opzione**: Deve essere una chiave esterna valida che fa riferimento alla tabella OpzioneRisposta.
13. **Risposte Aperte**
    * **Testo Risposta**: Deve essere un testo non nullo.
14. **Messaggi**
    * **Titolo**: Deve essere un campo non nullo.
    * **Testo**: Deve essere un campo non nullo.
    * **Data Inserimento**: Deve essere una data e ora valida.
    * **Tipo Mittente**: Deve essere 'Docente' o 'Studente'.
    * **Id Test**: Deve essere una chiave esterna valida che fa riferimento alla tabella Test.
15. **Completamento Test**
    * **Email Studente**: Deve essere una chiave esterna valida che fa riferimento alla tabella Studente.
    * **Id Test**: Deve essere una chiave esterna valida che fa riferimento alla tabella Test.
    * **Data Inserimento Prima Risposta**: Deve essere una data e ora valida.
    * **Data Inserimento Ultima Risposta**: Deve essere una data e ora valida.
    * **Stato**: Deve essere uno dei valori predefiniti ('Aperto', 'InCompletamento', 'Concluso').

REGOLE DI DERIVAZIONE:

1. **Aggiornamento di Num\_righe**
   * La colonna num\_righe viene modificata automaticamente tramite un trigger.
2. **Aggiornamento di num\_risposte**
   * La procedura associata aggiorna il campo num\_risposte ogni volta che viene aggiunta una nuova risposta.
3. **Impostazione di Data\_risposta**
   * La data e ora della risposta (data\_risposta) vengono automaticamente registrate al momento dell’inserimento di una nuova risposta.
4. **Registrazione di Data\_inserimento**
   * La data e ora del messaggio (data\_inserimento) vengono automaticamente impostate al momento della creazione del messaggio.
5. **Determinazione dello Stato del Completamento**
   * Il campo Stato\_del\_completamento viene aggiornato a Concluso quando tutte le risposte date dallo studente sono corrette.
6. **Impostazione iniziale dello Stato del Completamento**
   * Quando lo studente fornisce la prima risposta, il campo Stato\_del\_completamento viene automaticamente impostato su InCompletamento.
7. **Conclusione del Test**
   * Un test viene considerato concluso per tutti gli studenti quando il docente cambia il valore del campo VisualizzaRisposte a True.
8. **Creazione della Tabella di Esercizio**
   * La tabella di esercizio definita dal docente viene materializzata nel database, includendo tutti gli attributi e i vincoli definiti.
9. **Verifica delle Risposte Chiuse**
   * Un trigger denominato verificaRispostaCorrettaChiusa viene eseguito per le risposte chiuse per assicurarsi che l'opzione scelta dallo studente corrisponda a quella prevista e aggiorna il campo esito con True o False.

**3. Progettazione Logica (ristrutturazione dello schema concettuale, analisi delle ridondanze, lista delle tabelle con vincoli di chiavi, lista dei vincoli inter-relazionali).**

**- RISTRUTTURAZIONE DELLO SCHEMA CONCETTUALE:**

Prima di tradurre il modello E-R, è necessario ristrutturarlo per motivi di correttezza/efficienza, in primis eliminando le generalizzazioni, si sceglierà la soluzione che minimizza i valori NULL ovvero SOSTITUENDO LE GENERALIZZAZIONI CON RELAZIONI TRA GENITORI – FIGLI

-GENERALIZZAZIONE DI UTENTE: Immagine che contiene diagramma, schizzo, Piano, Disegno tecnico

Descrizione generata automaticamente

-GENERALIZZAZIONE DI RISPOSTA: Immagine che contiene diagramma, schizzo, Piano, Disegno tecnico

Descrizione generata automaticamente

-GENERALIZZAZIONE DI MESSAGGIO: Immagine che contiene testo, schizzo, calligrafia, disegno

Descrizione generata automaticamente

**-ANALISI DELLE RIDONDANZE** : Nel modello E-R, potrebbero essere presenti ridondanze sui dati, ossia informazioni significative ma derivabili da altre già presenti nel modello E-R.

(Eventuali) vantaggi delle ridondanze: Operazioni sui dati più efficienti o

Svantaggi delle ridondanze: Maggiore occupazione di memoria o Maggiore complessità degli aggiornamenti.

Calcolo dei costi delle 4 operazioni descritte nella traccia ( con e senza ridondanza):

|  |  |  |
| --- | --- | --- |
| **OPERAZIONE** | **CON RIDONDANZA** | **SENZA RIDONDANZA** |
| Aggiungere una nuova risposta ad un quesito esistente (10 volte/mese, interattiva) | 10 \* 1 [ ( 1 + 1 ) \* 2 + 0 ] = **40** | 10 \* 1 ( 1 \* 2 + 0 ) = **20** |
| Rimuovere un quesito e tutte le risposte ottenute (2 volte/mese, batch) | 2 \* 0,5 [ ( 1 + 10 ) \* 2 + 0 ] = **22** | 2 \* 0,5 [ ( 1 + 10 ) \* 2 + 0 ] = **22** |
| Visualizzare tutti gli utenti presenti nella piattaforma (1 volte/mese, batch) | 1 \* 0,5 ( 0 \* 2 + 50 ) = **25** | 1 \* 0,5 ( 0 \* 2 + 50 ) = **25** |
| Contare il numero di risposte per ciascun quesito presente nella piattaforma (2 volte/mese, interattiva) | 2 \* 1 ( 0 \* 2 + 1 ) = **2** | 2 \* 1 ( 0 \* 2 + 200 ) = **400** |
| **TOTALE** | **89** | **467** |

Calcolo speedup= 467/89 = **5.25 🡪 Questo significa che le operazioni sono circa 5.25 volte più veloci con la ridondanza rispetto alla situazione senza ridondanza.**

Calcolo occupazione memoria: M(S)=X M(Srid)=X + 4 \* Numero di Quesiti = X + 4 \* 20 = **X + 80 Byte 🡪 La ridondanza aggiunge solo 80 byte all'occupazione totale della memoria, che è una quantità relativamente piccola rispetto al miglioramento delle prestazioni.**

**Conclusione**: La ridondanza è giustificata perché migliora significativamente le prestazioni (speedup di 5.25 volte) con un aumento minimo dell'occupazione della memoria (80 byte). Pertanto, si è deciso di mantenere la ridondanza.

**-LISTA DELLE TABELLE CON VINCOLI DI CHIAVI** :

Utente ( email, password, nome, cognome, recapito\_telefonico, tipo\_utente)

Docente (email, nome\_dipartimento, nome\_corso)

Studente (email, anno\_immatricolazione, codice\_alfanumerico)

Test (id\_test, email\_docente, titolo, data\_creazione, foto, , VisualizzaRisposte)

Quesito (id\_quesito, id\_test, numero\_progressivo, livello\_difficolta, descrizione, num\_risposte, categoria)

OpzioneRisposta (id\_opzione, id\_quesito, numerazione, testo)

SoluzioneCodice (id\_soluzione, id\_quesito, sketch\_codice)

TabellaDiEsercizio (id\_tabella, email\_docente, id\_quesito , nome, data\_creazione, num\_righe, )

AttributoTabella (id\_attributo, id\_tabella, nome\_attributo, tipo, chiave\_primaria)

VincoloIntegrita(id\_attributo\_origine, id\_attributo\_destinazione)

Risposta (id\_risposta, id\_quesito, email\_studente, data\_risposta, esito)

RispostaChiusa (id\_risposta, id\_opzione)

RispostaAperta (id\_risposta, testo\_risposta)

Messaggio (id\_messaggio, titolo, testo, data\_inserimento, id\_test, tipo\_mittente)

MessaggioDocente(id\_messaggio,id\_test)

MessaggioStudente (id\_messaggio, email\_docente\_destinatario)

CompletamentoTest (email\_studente, id\_test, data\_inserimento\_prima\_risposta, data\_inserimento\_ultima\_risposta, stato\_del\_completamento)

SoluzioneRispostaChiusa (id\_quesito, numerazioneOpzione)

**-LISTA DEI VINCOLI INTER-RELAZIONALI** :

Utente.email → Docente.email

Utente.email → Studente.email

Docente.email → Test.email\_docente

Test.id\_test → Quesito.id\_test

Quesito.id\_quesito → OpzioneRisposta.id\_quesito

Quesito.id\_quesito → SoluzioneCodice.id\_quesito

Quesito.id\_quesito → TabellaDiEsercizio.id\_quesito

TabellaDiEsercizio.id\_tabella → AttributoTabella.id\_tabella

Quesito.id\_quesito → Risposta.id\_quesito

Studente.email → Risposta.email\_studente

Risposta.id\_risposta → RispostaChiusa.id\_risposta

Risposta.id\_risposta → RispostaAperta.id\_risposta

Test.id\_test → Messaggio.id\_test

Messaggio.id\_messaggio → MessaggioStudente.id\_messaggio

Docente.email → MessaggioStudente.email\_docente\_destinatario

Studente.email → CompletamentoTest.email\_studente

Test.id\_test → CompletamentoTest.id\_test

Quesito.id\_quesito → SoluzioneRispostaChiusa.id\_quesito

AttributoTabella.id\_attributo → VincoloIntegrita.id\_attributo\_origine

AttributoTabella.id\_attributo → VincoloIntegrita.id\_attributo\_destinazione

**NORMALIZZAZIONE** :

La **normalizzazione** è un processo di progettazione dei database che mira a organizzare i dati in modo tale da ridurre la ridondanza e migliorare l'integrità dei dati. Questo processo implica la decomposizione di una tabella in tabelle più piccole e la definizione di relazioni tra di esse. La normalizzazione si basa su una serie di **forme normali** (normal forms), ciascuna delle quali affronta aspetti specifici della ridondanza e della dipendenza dei dati:

**Prima Forma Normale (1NF)**: Garantisce che ogni colonna contenga valori atomici e che ogni riga sia unica.

**Seconda Forma Normale (2NF)**: Assicura che tutti gli attributi non chiave dipendano interamente dalla chiave primaria.

**Terza Forma Normale (3NF)**: Garantisce che tutti gli attributi non chiave dipendano solo dalla chiave primaria, eliminando dipendenze transitive.

**Forma Normale di Boyce-Codd (BCNF)**: Una versione più rigorosa della 3NF, che affronta alcune anomalie che non sono gestite dalla 3NF.

**Quarta Forma Normale (4NF)**: Gestisce le dipendenze multi-valore, garantendo che ogni tabella rappresenti un'unica relazione.

**Quinta Forma Normale (5NF)**: Si occupa delle dipendenze di join, garantendo che le tabelle possano essere ricostruite senza perdita di dati.

**1NF**: Non ci sono valori non atomici.

**2NF**: Non ci sono dipendenze parziali.

**3NF**: Non ci sono dipendenze transitive.

Anche **BCNF è verificata**, perché tutte le tabelle soddisfano il vincolo che ogni attributo non chiave dipende dalla chiave primaria o da una superchiave, e non ci sono dipendenze funzionali che violano questa regola.

**4NF**: Non ci sono dipendenze multi-valore.

**5NF**: Non ci sono dipendenze complesse join multiple

Si può affermare che nello schema:

* **Ridondanze Minime**: La struttura attuale sembra avere una ridondanza minima e una buona organizzazione dei dati. Le tabelle sono progettate per rappresentare entità distinte e le loro relazioni.
* **Performance**: Se le query sono semplici e non ci sono problemi di performance, una normalizzazione eccessiva potrebbe non essere necessaria. I dati sono ben organizzati e la ridondanza è già limitata.
* **Semplicità**: E’ abbastanza chiaro e non sembra eccessivamente complesso. La normalizzazione al massimo livello potrebbe introdurre complessità non necessaria.

**-Descrizione -ad alto livello- delle funzionalità dell’applicazione Web. :**

**ESQL** è una piattaforma per supportare la didattica del corso di dati, questa consente l’interazione tra STUDENTI e DOCENTI del corso, i docenti possono creare test e relativi quesiti, che possono essere a risposta chiusa o aperta, in modo tale che gli studenti possano visualizzarli nell’apposita sezione e completarli, ed eventualmente vedere delle classifiche inerenti a punteggio e risposte. E’ prevista la funzionalità di inviare messaggi per entrambi gli utenti.

**Di seguito le funzionalità approfondite con relative immagini:**

Pagina Iniziale: Immagine che contiene testo, schermata, Carattere, design

Descrizione generata automaticamente

Nel caso l’utente non si sia mai registrato dovrà registrarsi o come studente o come docente e compilare Il form a cui verrà reindirizzato a seconda del tipo di registrazione, in seguito dovrà fare il LOGIN inserendo email e password.

Login effettuato come STUDENTE:  
Immagine che contiene testo, schermata, Carattere, design

Descrizione generata automaticamente

In questa homepage lo studente potrà visualizzare i test disponibili, visualizzare gli esiti, visualizzare i messaggi ricevuti dal docente inerenti ai test, inoltre potrà visualizzare le classifiche Studenti per test, Classifiche studenti per risposte e le classifiche dei quesiti.

Visualizzazione Test Disponibili: Immagine che contiene testo, schermata, Carattere, design

Descrizione generata automaticamente

Cliccando “Partecipa al Test” verrà aperto l’esercizio a cui lo studente dovrà rispondere Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamenteCliccando su “Invia Messaggio” lo studente potrà inviare un messaggio al docente che ha creato il test chiedendo chiarimenti / informazioni

Completato il test lo studente potrà visualizzare l’esito solo quando il DOCENTE avrà corretto tutti i test e settato il booleano “Visualizza Risposte a TRUE” Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Inoltre sarà possibile visualizzare le seguenti classifiche Immagine che contiene testo, schermata, Rettangolo, Carattere

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Login effettuato come DOCENTE: Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

In questa homepage il DOCENTE potrà visualizzare i test esistenti creare dei nuovi test, potrà vedere i messaggi ricevuti dagli studenti e potrà guardare le stesse classifiche degli studenti.

Docente Visualizza Test: Immagine che contiene testo, schermata, software, Pagina Web

Descrizione generata automaticamente

Da questa pagina il docente potrà creare un nuovo test oppure modificarne uno esistente, inoltre se il campo visualizzaz risposte verrà settato e salvato, gli studenti potranno visualizzare i loro esiti e di conseguenza anche le classifiche si aggiorneranno.

Docente Crea Test: Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Riempiti tutti i campi, il test verrà visualizzato nella lista precedente e da essa cliccando su “titolo” sarà possibile aggiungere quesiti al test:

Nella pagina sottostante, il docente visualizzerà i dettagli del suo test e le rispettive domande che potranno essere a scelta multipla oppure domande aperte con quesiti di codice, inoltre potrà inviare un messaggio che sarà destinato a tutti gli studenti dove potrà chiarire o dare eventuali specifiche per il testImmagine che contiene testo, schermata, schermo, Carattere

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, design

Descrizione generata automaticamente

Docente crea quesiti chiusi: Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Tramite questo form il docente potrà creare il quesito a scelta multipla con il testo e 3 opzioni di risposte, di cui solo 1 sarà corretta, inoltre potrà scegliere il livello di difficoltà facile, medio, alto.

Docente Crea quesito di CODICE: Immagine che contiene testo, elettronica, schermata, software

Descrizione generata automaticamente

Tramite questo form il docente potrà creare quesiti di codice inerenti a query SQL, nel testo della domanda potrà scrivere ciò che vuole ottenere dalla query, esempio “seleziona tutti i camion con targa = 555) , e nel codice dovrà inserire la query necessaria per risolvere la domanda. Inserito il quesito, il docente dovrà creare la tabella necessaria per lo svolgimento dell’esercizio dunque verrà reindirizzato alla pagina di

Creazione delle tabella: Immagine che contiene testo, schermata, software, numero

Descrizione generata automaticamente

Dunque dovrà inserire il nome della tabella, la data, il numero di righe ( che potranno essere modificate in seguito) , il nome della chiave primaria e il tipo, fatto ciò la tabella verrà visualizzata nell’elenco e cliccando sul nome di essa potrà effettuare aggiunte di attributi e vincoliImmagine che contiene testo, schermata, numero, linea

Descrizione generata automaticamente

Docente aggiunge ATTRIBUTI e VINCOLI per collegarla ad altre tabelle:Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Aggiunto l’attributo il docente potrà popolarlo ovvero inserire dei valori cliccando su “vai agli inserimenti”: Immagine che contiene schermata, linea, testo

Descrizione generata automaticamente

Docente Visualizza Messaggio:  
Immagine che contiene testo, Carattere, schermata

Descrizione generata automaticamente

Cliccando su “seleziona messaggi” dalla homepageDocente si potranno visualizzare i messaggi ricevuti inerenti ad un determinato test

**-CODICE SQL** :

CREATE DATABASE esqldb\_ultimaVersione;

USE esqldb\_ultimaVersione;

CREATE TABLE Utente (

    email VARCHAR(255) PRIMARY KEY,

    password VARCHAR(255) NOT NULL,

    nome VARCHAR(100),

    cognome VARCHAR(100),

    recapito\_telefonico VARCHAR(20),

    tipo\_utente ENUM('Docente', 'Studente') NOT NULL

);

CREATE TABLE Docente (

    email VARCHAR(255) PRIMARY KEY,

    nome\_dipartimento VARCHAR(255),

    nome\_corso VARCHAR(255),

    FOREIGN KEY (email) REFERENCES Utente(email)

);

CREATE TABLE Studente (

    email VARCHAR(255) PRIMARY KEY,

    anno\_immatricolazione YEAR,

    codice\_alfanumerico CHAR(16),

    FOREIGN KEY (email) REFERENCES Utente(email)

);

CREATE TABLE Test (

    id\_test INT AUTO\_INCREMENT PRIMARY KEY,

    titolo VARCHAR(255) UNIQUE NOT NULL,

    data\_creazione DATE NOT NULL,

    foto VARCHAR(255), -- Percorso del file o URL dell'immagine

    email\_docente VARCHAR(255),

    VisualizzaRisposte BOOLEAN NOT NULL,

    FOREIGN KEY (email\_docente) REFERENCES Docente(email)

);

CREATE TABLE Quesito (

    id\_quesito INT AUTO\_INCREMENT PRIMARY KEY,

    id\_test INT,

    numero\_progressivo INT,

    livello\_difficolta ENUM('Basso', 'Medio', 'Alto') NOT NULL,

    descrizione TEXT NOT NULL,

    num\_risposte INT NOT NULL, -- Ridondanza concettuale, indica il numero di risposte per quesito

    categoria ENUM('Risposta Chiusa', 'Codice') NOT NULL,

    FOREIGN KEY (id\_test) REFERENCES Test(id\_test),

    UNIQUE (id\_test, numero\_progressivo)

);

CREATE TABLE OpzioneRisposta (

    id\_opzione INT AUTO\_INCREMENT PRIMARY KEY,

    id\_quesito INT,

    numerazione INT,

    testo VARCHAR(255) NOT NULL,

    FOREIGN KEY (id\_quesito) REFERENCES Quesito(id\_quesito),

    UNIQUE (id\_quesito, numerazione)

);

CREATE TABLE SoluzioneCodice (

    id\_soluzione INT AUTO\_INCREMENT PRIMARY KEY,

    id\_quesito INT,

    sketch\_codice TEXT NOT NULL,

    FOREIGN KEY (id\_quesito) REFERENCES Quesito(id\_quesito)

);

CREATE TABLE TabellaDiEsercizio (

    id\_tabella INT AUTO\_INCREMENT PRIMARY KEY,

    nome VARCHAR(255) NOT NULL,

    data\_creazione DATE NOT NULL,

    num\_righe INT,

    email\_docente VARCHAR(255),

    id\_quesito INT,

    FOREIGN KEY (email\_docente) REFERENCES Docente(email),

    FOREIGN KEY (id\_quesito) REFERENCES Quesito(id\_quesito)

);

CREATE TABLE AttributoTabella (

    id\_attributo INT AUTO\_INCREMENT PRIMARY KEY,

    id\_tabella INT,

    nome\_attributo VARCHAR(255) NOT NULL,

    tipo VARCHAR(100) NOT NULL,

    chiave\_primaria BOOLEAN,

    FOREIGN KEY (id\_tabella) REFERENCES TabellaDiEsercizio(id\_tabella)

);

CREATE TABLE Risposta (

    id\_risposta INT AUTO\_INCREMENT PRIMARY KEY,

    id\_quesito INT,

    email\_studente VARCHAR(255),

    data\_risposta TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

    esito BOOLEAN, -- True se la risposta è corretta, False altrimenti

    FOREIGN KEY (id\_quesito) REFERENCES Quesito(id\_quesito),

    FOREIGN KEY (email\_studente) REFERENCES Studente(email)

);

CREATE TABLE RispostaChiusa (

    id\_risposta INT PRIMARY KEY,

    id\_opzione INT NOT NULL,

    FOREIGN KEY (id\_risposta) REFERENCES Risposta(id\_risposta),

    FOREIGN KEY (id\_opzione) REFERENCES OpzioneRisposta(id\_opzione)

);

CREATE TABLE RispostaAperta (

    id\_risposta INT PRIMARY KEY,

    testo\_risposta TEXT NOT NULL,

    FOREIGN KEY (id\_risposta) REFERENCES Risposta(id\_risposta)

);

CREATE TABLE Messaggio (

    id\_messaggio INT AUTO\_INCREMENT PRIMARY KEY,

    titolo VARCHAR(255) NOT NULL,

    testo TEXT NOT NULL,

    data\_inserimento TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

    id\_test INT,

    tipo\_mittente ENUM('Docente', 'Studente') NOT NULL,

    FOREIGN KEY (id\_test) REFERENCES Test(id\_test)

);

CREATE TABLE MessaggioStudente (

    id\_messaggio INT PRIMARY KEY,

    email\_docente\_destinatario VARCHAR(255),

    FOREIGN KEY (id\_messaggio) REFERENCES Messaggio(id\_messaggio),

    FOREIGN KEY (email\_docente\_destinatario) REFERENCES Docente(email)

);

CREATE TABLE CompletamentoTest (

    email\_studente VARCHAR(255),

    id\_test INT,

    data\_inserimento\_prima\_risposta TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

    data\_inserimento\_ultima\_risposta TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

    stato\_del\_completamento ENUM('Aperto', 'InCompletamento', 'Concluso') NOT NULL,

    PRIMARY KEY (email\_studente, id\_test),

    FOREIGN KEY (email\_studente) REFERENCES Studente(email)

        ON DELETE CASCADE

        ON UPDATE CASCADE,

    FOREIGN KEY (id\_test) REFERENCES Test(id\_test)

        ON DELETE CASCADE

        ON UPDATE CASCADE

);

CREATE TABLE SoluzioneRispostaChiusa (

    id\_quesito INT,

    numerazioneOpzione INT,

    FOREIGN KEY (id\_quesito) REFERENCES Quesito(id\_quesito),

    PRIMARY KEY (id\_quesito, numerazioneOpzione)

);

DELIMITER //

CREATE TRIGGER TriggerInCompletamento

AFTER INSERT ON Risposta

FOR EACH ROW

BEGIN

        UPDATE CompletamentoTest

        SET stato\_del\_completamento = 'InCompletamento',

            data\_inserimento\_ultima\_risposta = CURRENT\_TIMESTAMP

        WHERE email\_studente = NEW.email\_studente AND id\_test = (SELECT id\_test FROM Quesito WHERE id\_quesito = NEW.id\_quesito);

END; //

DELIMITER ;

DELIMITER //

CREATE TRIGGER TriggerConclusoPerTutti

AFTER UPDATE ON Test

FOR EACH ROW

BEGIN

    IF NEW.VisualizzaRisposte = TRUE THEN

        UPDATE CompletamentoTest

        SET stato\_del\_completamento = 'Concluso',

            data\_inserimento\_ultima\_risposta = CURRENT\_TIMESTAMP

        WHERE id\_test = NEW.id\_test;

    END IF;

END; //

DELIMITER ;

DELIMITER //

CREATE TRIGGER TriggerIncrementoRighe

AFTER INSERT ON AttributoTabella

FOR EACH ROW

BEGIN

    UPDATE TabellaDiEsercizio SET num\_righe = num\_righe + 1 WHERE id\_tabella = NEW.id\_tabella;

END; //

DELIMITER ;

DELIMITER //

CREATE PROCEDURE VerificaCredenzialiStudente(IN emailParam VARCHAR(255), IN passwordParam VARCHAR(255))

BEGIN

    SELECT COUNT(\*) AS counter

    FROM Utente

    WHERE email = emailParam AND password = passwordParam and tipo\_utente = "studente";

END //

DELIMITER ;

DELIMITER //

CREATE PROCEDURE VerificaCredenzialiDocente(IN emailParam VARCHAR(255), IN passwordParam VARCHAR(255))

BEGIN

    SELECT email AS email\_docente, COUNT(\*) AS counter

    FROM Utente

    WHERE email = emailParam AND password = passwordParam AND tipo\_utente = 'Docente'

    GROUP BY email; -- Se ci sono più docenti con la stessa email e password, restituirà solo uno

END //

DELIMITER ;

DELIMITER //

CREATE PROCEDURE InserisciUtente(IN p\_email VARCHAR(255), IN p\_password VARCHAR(255), IN p\_nome VARCHAR(100), IN p\_cognome VARCHAR(100), IN p\_telefono VARCHAR(20), IN p\_tipo\_utente ENUM('Docente', 'Studente'))

BEGIN

    INSERT INTO Utente (email, password, nome, cognome, recapito\_telefonico, tipo\_utente) VALUES (p\_email, p\_password, p\_nome, p\_cognome, p\_telefono, p\_tipo\_utente);

    SELECT LAST\_INSERT\_ID() AS email;

END //

DELIMITER ;

DELIMITER //

CREATE PROCEDURE InserisciDocente(IN p\_email\_docente VARCHAR(255), IN p\_nome\_dipartimento VARCHAR(255), IN p\_nome\_corso VARCHAR(255))

BEGIN

    INSERT INTO Docente (email, nome\_dipartimento, nome\_corso) VALUES (p\_email\_docente, p\_nome\_dipartimento, p\_nome\_corso);

END //

DELIMITER ;

DELIMITER //

CREATE PROCEDURE RegistraStudente(

    IN p\_email VARCHAR(255),

    IN p\_password VARCHAR(255),

    IN p\_nome VARCHAR(100),

    IN p\_cognome VARCHAR(100),

    IN p\_telefono VARCHAR(20),

    IN p\_anno\_immatricolazione YEAR

)

BEGIN

    INSERT INTO Utente (email, password, nome, cognome, recapito\_telefonico, tipo\_utente)

    VALUES (p\_email, p\_password, p\_nome, p\_cognome, p\_telefono, 'Studente');

    INSERT INTO Studente (email, anno\_immatricolazione, codice\_alfanumerico)

    VALUES (p\_email, p\_anno\_immatricolazione, SUBSTRING(MD5(RAND()), 1, 16));

END //

DELIMITER ;

DELIMITER //

CREATE PROCEDURE GetMessaggiDocente()

BEGIN

    SELECT id\_messaggio, titolo, testo, data\_inserimento, id\_test, tipo\_mittente

    FROM Messaggio

    WHERE tipo\_mittente="docente"

    ORDER BY data\_inserimento DESC;

END //

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE GetMessaggiStudente(

    IN email\_docente VARCHAR(255)

)

BEGIN

    SELECT m.id\_messaggio, m.titolo, m.testo, m.data\_inserimento, m.id\_test, m.tipo\_mittente, ms.email\_docente\_destinatario

    FROM Messaggio m

    JOIN MessaggioStudente ms ON m.id\_messaggio = ms.id\_messaggio

    WHERE m.tipo\_mittente = 'Studente'

      AND ms.email\_docente\_destinatario = email\_docente

    ORDER BY m.data\_inserimento DESC;

END$$

DELIMITER ;

DELIMITER //

CREATE PROCEDURE InserisciMessaggio(

    IN titoloMessaggio VARCHAR(255),

    IN testoMessaggio TEXT,

    IN idTest INT

)

BEGIN

    INSERT INTO Messaggio (titolo, testo, tipo\_mittente, id\_test)

    VALUES (titoloMessaggio, testoMessaggio, 'Docente', idTest);

END //

DELIMITER ;

DELIMITER //

CREATE PROCEDURE InserisciMessaggioStudente(

    IN titoloMessaggio VARCHAR(255),

    IN testoMessaggio TEXT,

    IN idTest INT,

    IN emailDocenteDestinario VARCHAR(255)

)

BEGIN

    -- Inserisce il nuovo messaggio nella tabella Messaggi,

    -- impostando il tipo\_mittente su 'Studente'

    INSERT INTO Messaggio (titolo, testo, id\_test, tipo\_mittente)

    VALUES (titoloMessaggio, testoMessaggio, idTest, 'Studente');

    -- Ottiene l'ID del messaggio appena inserito

    SET @ultimoIdMessaggio = LAST\_INSERT\_ID();

    -- Inserisce un record nella tabella MessaggiStudenti

    -- con l'ID del messaggio e l'ID del docente destinatario

    INSERT INTO MessaggioStudente (id\_messaggio, email\_docente\_destinatario)

    VALUES (@ultimoIdMessaggio, emailDocenteDestinario);

END //

-- Recupero dell'ID dell'ultimo messaggio inserito

SET @id\_messaggio = LAST\_INSERT\_ID();

DELIMITER ;

DELIMITER //

CREATE PROCEDURE InserisciTest(

    IN p\_titolo VARCHAR(255),

    IN p\_data\_creazione DATE,

    IN p\_foto VARCHAR(255),

    IN p\_email\_docente VARCHAR(255),

    IN p\_VisualizzaRisposte BOOLEAN

)

BEGIN

    DECLARE error\_message TEXT;

    -- Gestione degli errori

    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION

    BEGIN

        GET DIAGNOSTICS CONDITION 1 error\_message = MESSAGE\_TEXT;

        SELECT CONCAT('Error: ', error\_message) AS error;

    END;

    INSERT INTO Test (titolo, data\_creazione, foto, email\_docente, VisualizzaRisposte)

    VALUES (p\_titolo, p\_data\_creazione, p\_foto, p\_email\_docente, p\_VisualizzaRisposte);

END //

DELIMITER ;

DELIMITER //

CREATE PROCEDURE GetTestsByDocenteId(IN docente\_email VARCHAR(255))

BEGIN

    SELECT \* FROM Test WHERE email\_docente = docente\_email;

END//

DELIMITER ;

DELIMITER //

CREATE PROCEDURE UpdateTestVisualizzaRisposte(IN test\_id INT, IN visualizza\_risposte INT)

BEGIN

    UPDATE test SET VisualizzaRisposte = visualizza\_risposte WHERE id\_test = test\_id;

END //

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE GetTestCompletati()

BEGIN

    SELECT S.codice\_alfanumerico AS codice\_studente,

           COUNT(C.id\_test) AS numero\_test\_completati

    FROM Studente S

    LEFT JOIN CompletamentoTest C ON S.email = C.email\_studente AND C.stato\_del\_completamento = 'Concluso'

    GROUP BY S.email

    ORDER BY numero\_test\_completati DESC;

END$$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE InserisciOpzioneRisposta(

    IN \_id\_quesito INT,

    IN \_numerazione INT,

    IN \_testo TEXT

)

BEGIN

    INSERT INTO OpzioneRisposta (id\_quesito, numerazione, testo) VALUES (\_id\_quesito, \_numerazione, \_testo);

END$$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE InserisciSoluzioneCodice(

    IN idQuesito INT,

    IN sketchCodice TEXT

)

BEGIN

    INSERT INTO SoluzioneCodice (id\_quesito, sketch\_codice) VALUES (idQuesito, sketchCodice);

END$$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE `OttieniRisposteStudente`(

    IN \_email\_studente VARCHAR(255)

)

BEGIN

    SELECT Test.id\_test,

           Quesito.numero\_progressivo,

           Quesito.descrizione AS descrizione\_quesito,

           Risposta.esito,

           Risposta.data\_risposta

    FROM Risposta

    JOIN Quesito ON Risposta.id\_quesito = Quesito.id\_quesito

    JOIN Test ON Quesito.id\_test = Test.id\_test

    WHERE Risposta.email\_studente = \_email\_studente AND Test.VisualizzaRisposte = 1

    ORDER BY Test.id\_test, Quesito.numero\_progressivo;

END$$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE InserisciNellaTabellaDiEsercizio(

    IN \_nome VARCHAR(255),

    IN \_data\_creazione DATE,

    IN \_num\_righe INT,

    IN \_email\_docente VARCHAR(255),

    IN \_id\_quesito INT

)

BEGIN

    -- Inserisci il record nella tabella

    INSERT INTO TabellaDiEsercizio (nome, data\_creazione, num\_righe, email\_docente, id\_quesito)

    VALUES (\_nome, \_data\_creazione, \_num\_righe, \_email\_docente, \_id\_quesito);

    -- Restituisci l'ID dell'ultima riga inserita

    SELECT LAST\_INSERT\_ID() AS id\_tabella;

END$$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE AggiungiAttributoTabella(

    IN \_id\_tabella INT,

    IN \_nome\_attributo VARCHAR(255),

    IN \_tipo VARCHAR(255)

)

BEGIN

    INSERT INTO AttributoTabella (id\_tabella, nome\_attributo, tipo, chiave\_primaria)

    VALUES (\_id\_tabella, \_nome\_attributo, \_tipo, 1);

END$$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE GetDettagliTabellaEAttributo()

BEGIN

    SELECT

        t.id\_tabella,

        t.nome AS nome\_tabella,

        t.data\_creazione,

        t.num\_righe,

        t.email\_docente,

        t.id\_quesito,

        a.id\_attributo,

        a.nome\_attributo,

        a.tipo,

        a.chiave\_primaria

    FROM

        TabellaDiEsercizio t, AttributoTabella a

    WHERE

        t.id\_tabella = a.id\_tabella;

END$$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE AggiungiAttributo(

    IN `\_nome\_attributo` VARCHAR(255),

    IN `\_tipo` VARCHAR(255),

    IN `\_id\_tabella` INT

)

BEGIN

    INSERT INTO `attributotabella` (`nome\_attributo`, `tipo`, `chiave\_primaria`, `id\_tabella`)

    VALUES (\_nome\_attributo, \_tipo, 0, \_id\_tabella);

END$$

DELIMITER ;

CREATE VIEW VistaTestCompletati AS

SELECT S.email AS codice\_studente,

       COUNT(C.id\_test) AS numero\_test\_completati

FROM Studente S

LEFT JOIN CompletamentoTest C ON S.email = C.email\_studente AND C.stato\_del\_completamento = 'Concluso'

GROUP BY S.email;

CREATE VIEW ClassificaRisposteEsatte AS

SELECT

    S.codice\_alfanumerico AS codice\_studente,

    COUNT(CASE WHEN R.esito = TRUE THEN 1 END) AS risposte\_corrette,

    COUNT(R.id\_risposta) AS totale\_risposte,

    (COUNT(CASE WHEN R.esito = TRUE THEN 1 END) \* 100.0 / COUNT(R.id\_risposta)) AS percentuale\_corrette

FROM

    Studente S

JOIN

    Risposta R ON S.email = R.email\_studente

GROUP BY

    S.codice\_alfanumerico

HAVING

    COUNT(R.id\_risposta) > 0

ORDER BY

    percentuale\_corrette DESC, totale\_risposte DESC;

    CREATE VIEW ClassificaQuesiti AS

SELECT

    Q.id\_quesito,

    COUNT(R.id\_risposta) AS numero\_risposte

FROM

    Quesito Q

LEFT JOIN

    Risposta R ON Q.id\_quesito = R.id\_quesito

GROUP BY

    Q.id\_quesito

ORDER BY

    numero\_risposte DESC;

    DELIMITER //

DELIMITER $$

CREATE PROCEDURE GetTestByStudenteId(

    IN email\_studente\_param VARCHAR(255)

)

BEGIN

    SELECT Test.\*, Docente.email, CompletamentoTest.stato\_del\_completamento

    FROM Test

    JOIN Docente ON Test.email\_docente = Docente.email

    LEFT JOIN CompletamentoTest ON Test.id\_test = CompletamentoTest.id\_test

        AND CompletamentoTest.email\_studente = email\_studente\_param

    ORDER BY Test.email\_docente, Test.data\_creazione ASC;

END$$

DELIMITER ;

DELIMITER $$

CREATE PROCEDURE AggiornaNumRigheTabellaEsercizio(nome\_tabella VARCHAR(255))

BEGIN

    -- Variabile per costruire la query dinamica

    SET @queryDinamica = CONCAT('SELECT COUNT(\*) INTO @numRighe FROM ', nome\_tabella);

    -- Esegue la query per ottenere il conteggio delle righe

    PREPARE stmt FROM @queryDinamica;

    EXECUTE stmt;

    DEALLOCATE PREPARE stmt;

    -- Aggiorna il campo num\_righe in TabellaDiEsercizio

    SET @queryAggiornamento = CONCAT('UPDATE TabellaDiEsercizio SET num\_righe = @numRighe WHERE nome = ''', nome\_tabella, '''');

    PREPARE stmtAggiornamento FROM @queryAggiornamento;

    EXECUTE stmtAggiornamento;

    DEALLOCATE PREPARE stmtAggiornamento;

END$$

DELIMITER ;

DELIMITER $$

CREATE TRIGGER verificaRispostaCorrettaChiusa

AFTER INSERT ON RispostaChiusa

FOR EACH ROW

BEGIN

    DECLARE soluzioneCorretta INT;

    -- Verifica se l'id\_opzione inserito corrisponde alla soluzione corretta per l'id\_quesito associato

    SELECT COUNT(\*) INTO soluzioneCorretta

    FROM SoluzioneRispostaChiusa

    WHERE id\_quesito = (SELECT id\_quesito FROM Risposta WHERE id\_risposta = NEW.id\_risposta)

          AND numerazioneOpzione = (SELECT numerazione FROM OpzioneRisposta WHERE id\_opzione = NEW.id\_opzione);

    -- Se soluzioneCorretta > 0, allora la risposta è corretta (esito = TRUE), altrimenti è incorretta (esito = FALSE)

    UPDATE Risposta

    SET esito = IF(soluzioneCorretta > 0, TRUE, FALSE)

    WHERE id\_risposta = NEW.id\_risposta;

END$$

DELIMITER ;

DELIMITER //

CREATE TRIGGER TriggerConclusoStudente

AFTER INSERT ON Risposta

FOR EACH ROW

BEGIN

    -- Controlla se il numero di quesiti unici nel test è uguale al numero di quesiti a cui lo studente ha risposto correttamente almeno una volta.

    IF (SELECT COUNT(DISTINCT id\_quesito) FROM Quesito WHERE id\_test = (SELECT id\_test FROM Quesito WHERE id\_quesito = NEW.id\_quesito)) =

       (SELECT COUNT(DISTINCT id\_quesito) FROM Risposta WHERE email\_studente = NEW.email\_studente AND esito = TRUE AND id\_quesito IN (SELECT id\_quesito FROM Quesito WHERE id\_test = (SELECT id\_test FROM Quesito WHERE id\_quesito = NEW.id\_quesito)))

    THEN

        -- Aggiorna lo stato del completamento del test a 'Concluso'.

        UPDATE CompletamentoTest

        SET stato\_del\_completamento = 'Concluso',

            data\_inserimento\_ultima\_risposta = CURRENT\_TIMESTAMP

        WHERE email\_studente = NEW.email\_studente AND id\_test = (SELECT id\_test FROM Quesito WHERE id\_quesito = NEW.id\_quesito);

    END IF;

END; //

DELIMITER ;