Progetto di Basi di Dati A.A. 2023-24

Nome gruppo: "G_ESQL".

Membri:
- Matteo Zaniboni
- Claudio Rocca
- Leonardo Alleruzzo Bellei

Indice

1) Raccolta/Analisi dei requisiti

La raccolta/analisi dei requisiti consiste nella completa individuazione dei problemi che il sistema informativo da realizzare deve risolvere e le caratteristiche che il sistema software deve avere. Si definiscono quindi quali dati devono essere gestiti e quali operazioni devono essere consentite.

1.1 Documento di specifica

Tutti gli utenti della piattaforma possiedono un indirizzo mail, nome, cognome ed eventuale recapito telefonico. Gli utenti sono divisi in due tipologie: docenti e studenti. I docenti possiedono anche: nome del dipartimento di afferenza e nome del corso di cui sono titolari. Gli studenti inseriscono anche l'anno di immatricolazione ed un codice alfanumerico univoco. I docenti possono creare delle tabelle di esercizio SQL, ognuna di esse dispone di nome, data di creazione e numero di righe. Tutte le tabelle di esercizio possiedono un insieme di attributi: ogni attributo dispone di un nome, un tipo, e può essere parte della chiave primaria della tabella di esercizio. Devono poter essere inseriti dai docenti anche i vincoli di integrità tra attributi di diverse tabelle di esercizio. I docenti possono creare dei test: ciascuno contenente un titolo univoco, una data di creazione e volendo una foto. Ogni test può includere diversi quesiti: ognuno di essi dispone di un numero progressivo univoco, un livello di difficoltà, un campo descrizione ed un numero di risposte. Un quesito fa riferimento ad una o più tabelle di esercizio create dal docente. I quesiti possono essere solo di due categorie: quesiti a risposta chiusa o quesiti di codice. I quesiti a risposta chiusa dispongono di una serie di opzioni di risposta: ognuna dispone di una numerazione univoca ed un campo testo. I quesiti di codice hanno una o più soluzioni date da sketch di codice SQL. Ogni test dispone di un campo booleano VisualizzaRisposte che se settato a True, rende le risposte dei quesiti visibili agli studenti, altrimenti restano nascoste. Gli studenti possono svolgere un test, inserendo una o più risposte per ciascun quesito. Viene tracciato il completamento del test, ossia: data di inserimento della prima risposta, data di inserimento dell'ultima risposta e lo stato. Nel caso di quesiti a risposta chiusa, la risposta consiste nell'opzione scelta tra quelle disponibili. Nel caso di quesiti di codice, la risposta consiste in un campo testo. Inoltre, gli studenti possono sottomettere più risposte per lo stesso quesito, in istanti diversi di tempo. Ogni risposta possiede il campo booleano esito, che vale True se la risposta è corretta altrimenti vale False, sia per quesiti di codice che a risposta chiusa. All'interno della piattaforma si possono inviare messaggi: ognuno contenente un titolo, un campo testo, una data di inserimento, e fa riferimento ad uno specifico test. Il messaggio inviato da un docente viene ricevuto da tutti gli studenti, mentre il messaggio inviato da uno studente viene ricevuto da uno specifico docente.

1.2 Decomposizione in gruppi di frasi

Frasi relative a:

- UTENTI

Tutti gli utenti della piattaforma possiedono un indirizzo mail, nome, cognome ed eventuale recapito telefonico. Gli utenti sono divisi in due tipologie: docenti e studenti.

- STUDENTI

Gli studenti dispongono di un campo anno di immatricolazione e di un codice alfanumerico univoco. Gli studenti possono svolgere un test, inserendo una o più risposte per ciascun quesito. Inoltre, gli studenti possono sottomettere più risposte per lo stesso quesito, in istanti diversi di tempo.

- DOCENTI

I docenti dispongono del nome del dipartimento di afferenza e nome del corso di cui sono titolari. I docenti possono creare delle tabelle di esercizio e devono essere inserire anche i vincoli di integrità referenziale tra attributi di diverse tabelle di esercizio. In aggiunta a ciò, i docenti possono creare dei test.

- TABELLE DI ESERCIZIO

Vengono create dai docenti ed ognuna di esse dispone di nome, data di creazione e numero di righe. Inoltre, ogni tabella dispone di un insieme di attributi.

- ATTRIBUTO

Ogni attributo dispone di un nome, un tipo e può essere parte della chiave primaria della tabella di esercizio.

- TEST

Un test può essere creato da un docente ed ogni test dispone di un titolo univoco, una data di creazione e volendo una foto. Inoltre, ogni test può includere diversi quesiti, ed ha un campo booleano VisualizzaRisposte che, se settato a True, rende le risposte dei quesiti visibili agli studenti, altrimenti restano nascoste.

- QUESITO

Ogni quesito dispone di un numero progressivo univoco, un livello di difficoltà, un campo descrizione ed un numero di risposte. Un quesito fa riferimento ad una o più tabelle di esercizio create dal docente. I quesiti possono essere solo di due categorie: quesiti a risposta chiusa o quesiti di codice.

- DOMANDA CHIUSA

I quesiti a risposta chiusa dispongono di una serie di opzioni di risposta. In questi quesiti la risposta consiste nell'opzione scelta tra quelle disponibili. Ogni opzione di risposta dispone di una numerazione univoca ed un campo testo.

- CODICE
I quesiti di codice hanno una o più soluzioni date da sketch di codice SQL. In questi quesiti la risposta consiste in un campo testo.

- RISPOSTA

Nel caso di quesiti a risposta chiusa, la risposta consiste nell'opzione scelta tra quelle disponibili. Nel caso di quesiti di codice, la risposta consiste in un campo testo.
Gli studenti possono sottomettere più risposte per lo stesso quesito, in istanti diversi di tempo. Ogni risposta possiede il campo booleano esito, che vale True se la risposta è corretta altrimenti vale False, sia per quesiti di codice che a risposta chiusa.

- MESSAGGI

Ogni messaggio contiene un titolo, un campo testo, una data di inserimento, e fa riferimento ad uno specifico test. Il messaggio inviato da un docente viene ricevuto da tutti gli studenti, mentre il messaggio inviato da uno studente viene ricevuto da uno specifico docente.

- COMPLETAMENTO

Si vuole tenere traccia del completamento del test, ossia: data di inserimento della prima risposta, data di inserimento dell'ultima risposta, stato.

- APPARTENENZA

Ogni quesito fa riferimento ad una o più tabelle, si vuole pertanto tenere traccia delle tabelle riferire da ogni quesito

- VINCOLI D'INTEGRITA'

Vincoli di integrità inseriti dal docente durante la creazione delle tabelle di esercizio.
Vengono memorizzati in una tabella contenente l'attributo referente e l'attributo riferito.

1.3 Operazioni sui dati.

- OPERAZIONE 1. Registrare un nuovo utente sulla piattaforma.
- OPERAZIONE 2. Autenticare l'accesso di un profilo utente sulla piattaforma.
- OPERAZIONE 3. Inserimento di una nuova tabella di esercizio, con relativi meta-dati.
- OPERAZIONE 4. Inserimento di una riga per una tabella di esercizio definita dal docente.
- OPERAZIONE 5. Creazione di un nuovo test.
- OPERAZIONE 6. Creazione di un nuovo quesito con le relative risposte.
- OPERAZIONE 7. Abilitare / disabilitare la visualizzazione delle risposte per uno specifico test.
- OPERAZIONE 8. Inserimento di un messaggio.
- OPERAZIONE 9. Inserimento di una nuova risposta ad un quesito.

- OPERAZIONE 10. Visualizzazione dell'esito della risposta.
- OPERAZIONE 11. Visualizzazione dei test disponibili.
- OPERAZIONE 12. Visualizzazione dei quesiti presenti all'interno di ciascun test.
- OPERAZIONE 13. Modificare lo stato di un test.
- OPERAZIONE 14. Visualizzare la classifica degli studenti, sulla base del numero di test completati.
- OPERAZIONE 15. Visualizzare la classifica degli studenti, sulla base del numero di risposte corrette inserite rispetto al numero totale di risposte inserite.
- OPERAZIONE 16. Visualizzare la classifica dei quesiti, in base al numero di risposte inserite dagli studenti.

1.4 Tavola media dei volumi.

La tavola dei volumi fornisce una stima del numero di occorrenze di entità/ relazioni presenti nel modello E-R.

Termine	**Tipologia**	**Volume**
Utente	Entità	**205**
Docente	Entità	**5**
Studente	Entità	**200**
Tabella	Entità	**30**
Attributo	Entità	**100**
Test	Entità	**20**
Quesito	Entità	**200**
Q. Risposta Chiusa	Entità	**150**
Q. Codice	Entità	**50**
Opzione di risposta	Entità	**450**
Sketch	Entità	**100**
Messaggi Docente	Entità	**20**
Messaggi Studente	Entità	**400**
Creazione	Relazione	**50**
Composizione	Relazione	**5.000**
Vincoli d'integrità	Relazione	**10.000**
Appartenenza	Relazione	**6.000**
Realizzazione	Relazione	**20**
Completamento	Relazione	**4.000**
Inclusione	Relazione	**4.000**
Risposta	Relazione	**40.000**
Risoluzione	Relazione	**67.500**
Soluzione	Relazione	**5.000**
Riferimento	Relazione	**8.400**
Scambio	Relazione	**20**
Interscambio	Relazione	**400**

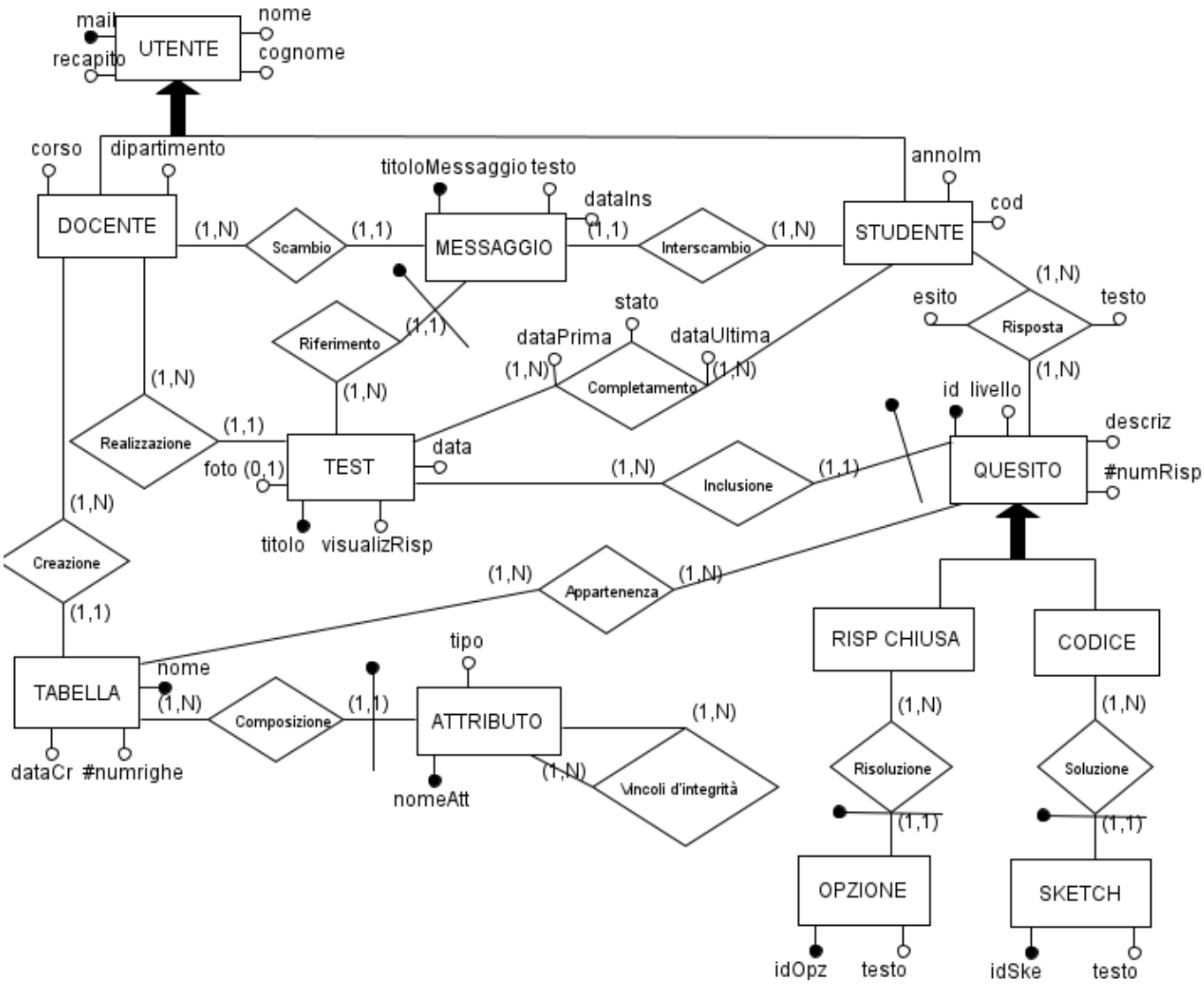
1.5 Glossario dei termini.

Termine	Descrizione	Sinonimi	Collegamenti
Utente	Utilizzatore della piattaforma		Docente, studente
Docente	Utente in grado di create tabelle, test e quesiti		Tabella, test, messaggio
Studente	Utente in grado di inserire risposte ai quesiti		Test, quesito, messaggio
Tabella	Tabella di esercizio creata da un docente		Docente, attributo, quesito
Attributo	Attributo contenuto all'interno di una tabella di esercizio		Tabella
Test	Test creato da un docente		Docente, studente, quesito, tabella, messaggio
Quesito	Quesito creato da un docente all'interno di un test		Studente, test, tabella, quesito risposta chiusa, quesito codice
Quesito Risposta Chiusa	Quesito a risposta chiusa con opzioni create dal docente	Domanda chiusa	Quesito, opzione
Quesito Codice	Quesito che viene risolto con uno sketch di codice SQL	Domanda codice	Quesito, sketch
Opzione di risposta	Possibile soluzione di un quesito a risposta chiusa		Quesito risposta chiusa
Sketch	Possibile soluzione di un quesito codice		Quesito codice
Messaggio	Messaggio scambiato sulla piattaforma		Docente, studente, test
Risposta	Risposta di uno studente ad un quesito		Studente, quesito
Completamento	Stato attuale di un test svolto da uno studente		Studente, test

2) Progettazione Concettuale

Durante questa fase si procede con la modellazione dello schema E-R, adottato per la rappresentazione concettuale dei dati. Inoltre, si vanno a realizzare alcune tabelle per chiarire in maniera descrittiva tutti gli aspetti non riproducibili attraverso lo schema.

2.1 Modello E-R.



2.2 Dizionario delle Entità.

ENTITA'	DESCRIZIONE	ATTRIBUTI	IDENTIDIFICATORE
Utente	Utilizzatore della piattaforma, può registrarsi come docente o studente	mail, nome, cognome, recapito	mail
Docente	Utente in grado di create tabelle, test e quesiti	" + dipartimento, corso	"
Studente	Utente in grado di inserire risposte ai quesiti	" + anno immatricolazione, codice	"
Tabella	Tabella di esercizio creata dal docente	nome, data creazione, numero righe, mail docente	nome
Attributo	Attributo contenuto all'interno di una tabella di esercizio	nome, tipo, nome tabella	Nome + identificatore esterno (nome tabella)
Test	Test creato dal docente	titolo, data, foto, mail docente, visualizza risposte	titolo
Quesito	Quesito creato da un docente all'interno di un test	id, livello, titolo test, descrizione, numero risposte	Id + identificatore esterno (titolo test)
Quesito risposta chiusa	Quesito a risposta chiusa con opzioni create dal docente	"	"
Quesito codice	Quesito che viene risolto con uno sketch di codice SQL	"	"
Opzione	Possibile soluzione di un quesito a risposta chiusa	Id opzione, testo, id quesito, titolo test, corretta	Id opzione + identificatori esterni (titolo test, id quesito)
Sketch	Possibile soluzione di un quesito codice	Id sketch, testo, titolo test, id quesito	IdSketch + Identificatori esterni (titolo test, id quesito)
Messaggio docente	Messaggio inviato da un docente a tutti gli studenti	Titolo messaggio, testo, data inserimento, titolo test, mittente	Titolo messaggio, mittente
Messaggio studente	Messaggio inviato da uno studente ad un docente	Titolo messaggio, testo, data inserimento, titolo test, mittente, destinatario	Titolo messaggio, mittente, destinatario

Per quanto riguarda le Generalizzazioni abbiamo utilizzato " per indicare che le classi figlie ereditano tutti gli attributi della classe padre ed hanno lo stesso identificatore del padre.

2.3 Dizionario delle Relazioni.

RELAZIONI	DESCRIZIONE	COMPONENTI	ATTRIBUTI
Creazione	Creazione di una tabella da parte di un docente	Docente, Tabella	
Realizzazione	Creazione di un test da parte di un docente	Docente, Test	
Riferimento	Messaggio associato ad uno specifico test	Messaggio, Test	
Completamento	Completamento di un test da parte di uno studente	Test, Studente	dataPrima, dataUltima, stato
Inclusione	Insieme di quesiti inclusi in un test	Test, Quesito	
Appartenenza	Riferimento di un quesito ad una tabella di esercizio	Tabella, Quesito	
Composizione	Attributi che caratterizzano una tabella	Tabella, Attributo	
Risposta	Risposta di uno studente ad un quesito	Studente, Quesito	Esito, testo
Risoluzione	Possibile soluzione di un quesito a risposta chiusa	Risp.Chiusa, Opzione di Risposta	
Soluzione	Possibile soluzione di un quesito codice	Codice, Sketch	
Scambio	Invio di un messaggio da parte di un docente	Messaggio, Docente	
Interscambio	Invio di un messaggio da parte di uno studente	Messaggio, Studente	

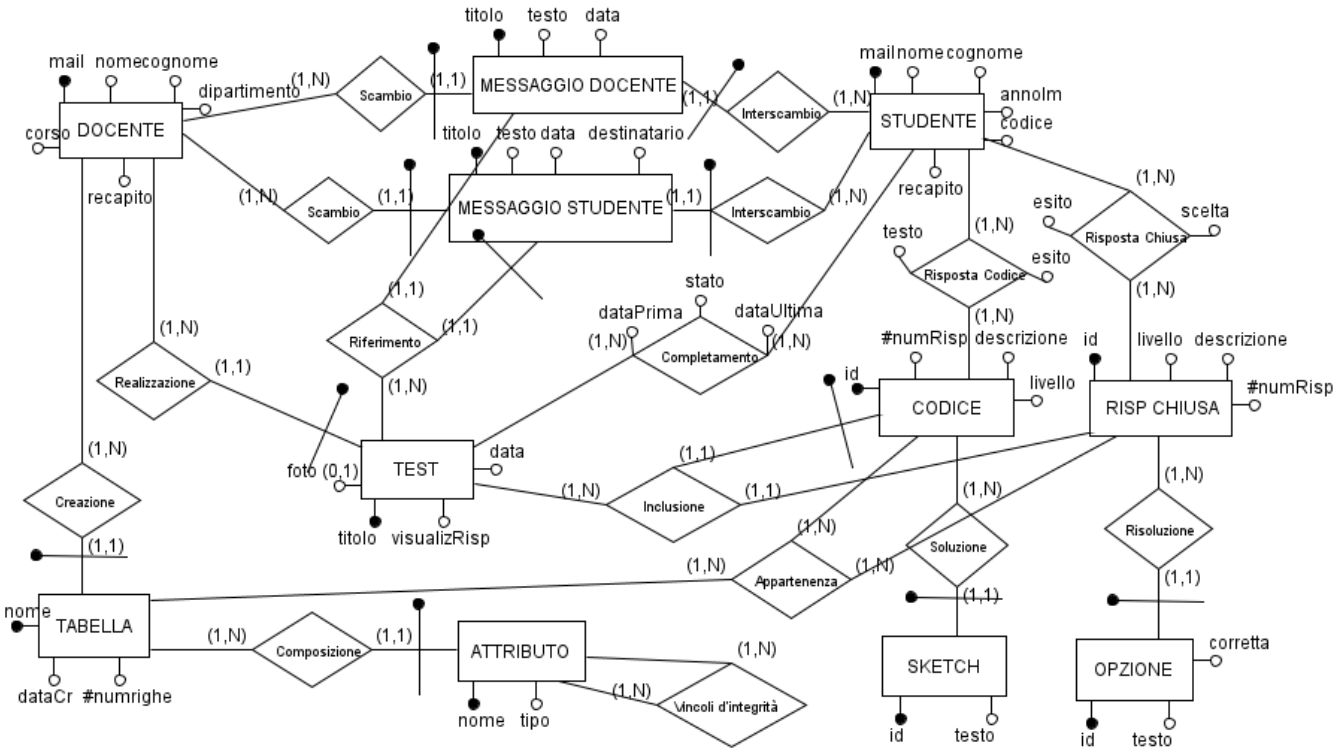
2.4 Tavola delle Business Rules.

REGOLE DI VINCOLO
Il codice di uno studente deve essere lungo 16 caratteri
Ogni quesito è identificato da un numero progressivo, univoco solo all'interno di uno specifico test.
Uno studente può sottomettere più risposte per lo stesso quesito, ma solo se il test non è stato chiuso dal docente
Le opzioni di risposta per i quesiti a risposta chiusa e gli sketch per i quesiti codice sono univoci, ma solo all'interno di uno specifico quesito.
Un messaggio inviato da un docente è recapitato da tutti gli studenti del corso, invece un messaggio comunicato da uno studente è ricevuto dallo specifico docente.
REGOLE DI DERIVAZIONE
Il campo #numRisposte è dato dalla somma di tutte le risposte inserite dagli studenti.
Ogni test può assumere un livello di difficoltà pari a: Basso, Medio, Difficile.
Lo stato di completamento di un test può essere: Aperto, In_Completamento, Concluso.

3) Progettazione Logica

L'obiettivo della progettazione logica è la realizzazione del modello logico (es. relazionale) a partire dalle informazioni del modello E-R. Per fare ciò si mettono in atto alcune pratiche per ottimizzare la traduzione in modello logico.

3.1 Ristrutturazione schema concettuale.



3.2 Analisi delle ridondanze.

Analisi delle ridondanze sui dati, ossia informazioni significative ma derivabili da altre già presenti nel modello E-R. Si procede quindi al calcolo di costo ed occupazione di memoria dello schema con ridondanza e di quello senza ridondanza. Si confrontano i risultati ottenuti e si valuta se mantenere o meno la ridondanza sui dati.

Nel nostro caso specifico si valuta se la seguente ridondanza: campo **#numrisposte** relativo ad un quesito debba essere tenuta o eliminata, sulla base delle seguenti operazioni:

1. Aggiungere una nuova risposta ad un quesito esistente (10 volte/mese, interattiva).
2. Rimuovere un quesito e tutte le risposte ottenute (2 volte/mese, batch).
3. Visualizzare tutti gli utenti presenti nella piattaforma (1 volte/mese, batch).
4. Contare il numero di risposte per ciascun quesito presente nella piattaforma (2 volte/mese, interattiva).

Coefficienti analisi: α (peso operazioni scrittura) = 2, WI (peso operazioni Interattive) = 1, WB (peso operazioni Batch) = 0.5

Tabella dei volumi: 10 risposte per quesito, 20 quesiti, 50 utenti.

Formula: $c(OT) = f(OT) \cdot WT \cdot (\alpha \cdot NCwrite + NCread)$

Caso **CON ridondanza**:

- $c(Op1) = 10 \cdot 1 \cdot (2 \cdot 2 + 0) = 40$ -> il 2 nelle op di scrittura è dato da 1 accesso a Risposta + 1 accesso a Quesito.
- $c(Op2) = 2 \cdot 0{,}5 \cdot (2 \cdot 12 + 0) = 24$ -> il 12 nelle op di scrittura è dato da 10 accessi a Risposta + 1 accesso a Quesito + 1 accesso a RispChiusa o Codice.
- $c(Op3) = 1 \cdot 0{,}5 \cdot (2 \cdot 0 + 100) = 50$ -> il 100 nelle op di lettura è dato da 50 accessi a Utente + 50 accessi a Docente o Studente.
- $c(Op4) = 2 \cdot 1 \cdot (2 \cdot 0 + 40) = 80$ -> il 40 nelle op di scrittura è dato da 20 accessi a Quesito + 20 accessi a RispChiusa o Codice.

Costo totale con ridondanza delle due operazioni in cui è coinvolta la ridonda è:
$c(Srid) = 40 + 80 = 120$

Caso **SENZA ridondanza**:

- $c(Op1) = 10 \cdot 1 \cdot (2 \cdot 1 + 0) = 20$ -> il 1 nelle op di scrittura è dato da 1 accesso a Risposta.
- $c(Op4) = 2 \cdot 1 \cdot (2 \cdot 0 + 240) = 480$ -> il 240 nelle op di scrittura è dato da 20 accessi a Quesito + 20 accessi a RispChiusa o Codice + 200 (20x10) accessi a Risposta.

Costo totale senza ridondanza è:
$c(S) = 20 + 480 = 500$

Per quanto riguarda l'occupazione di memoria:
m(S) = X (byte)
m(Srid) = X + 20 * 4 = X + 80 (Byte) -> dove 20 rappresenta il numero di Quesiti e 4 sono i Byte.

In sintesi, la presenza della ridondanza:
Introduce un overhead di memoria di 80 Byte e migliora lo speedup delle operazioni di circa: 500/120 ~ 4,16.
In questo caso, **è conveniente mantenere l'attributo #numrisposte** visto quanta poca memoria viene aggiunta.

3.3 Lista delle tabelle con vincoli di chiave.

Insieme di tabelle complete di attributi e chiavi primarie (sottolineate) che descrivono il modello logico.

Studente(MAIL, NOME, COGNOME, PASSWORD, RECAPITO, ANNO_IMMATRICOLAZIONE, CODICE)
Docente(MAIL, NOME, COGNOME, PASSWORD, RECAPITO, DIPARTIMENTO, CORSO)
Tabella(NOME, DATA_CREAZIONE, NUM_RIGHE, MAIL_DOCENTE)
Attributo(NOME, NOME_TABELLA, TIPO)
Vincolo_integrità(TABELLA_REFERENTE, ATTRIBUTO_REFERENTE, TABELLA_RIFERITA, ATTRIBUTO_RIFERITO)
Test(TITOLO, FOTO, DATA_CREAZIONE, VISUALIZZA_RISPOSTA, MAIL_DOCENTE)
Quesito_Risposta_Chiusa(ID, TITOLO_TEST, LIVELLO, DESCRIZIONE, NUM_RISPOSTE)
Quesito_Codice(ID, TITOLO_TEST, LIVELLO, DESCRIZIONE, NUM_RISPOSTE)
Opzione (ID_OPZIONE, TITOLO_TEST, ID_QUESITO, TESTO, CORRETTA)
Sketch (ID_SKETCH, TITOLO_TEST, ID_QUESITO, TESTO)
Appartenenza_Quesito_Chiuso(ID_QUESITO, TITOLO_TEST, NOME_TABELLA)
Appartenenza_Quesito_Codice(ID_QUESITO, TITOLO_TEST, NOME_TABELLA)
Completamento(TITOLO_TEST, MAIL_STUDENTE, STATO, DATA_ULTIMA_RISPOSTA, DATA_PRIMA_RISPOSTA)
Risposta Chiusa(MAIL_STUDENTE, ID_QUESITO, TITOLO_TEST, ESITO, SCELTA)
Risposta Codice(MAIL_STUDENTE, ID_QUESITO, TITOLO_TEST, TESTO, ESITO)
Messaggio_Docente(TITOLO, MITTENTE, TESTO, DATA_INSERIMENTO, TITOLO_TEST)
Messaggio_Studente(TITOLO, MITTENTE , TESTO, DATA_INSERIMENTO, DESTINATARIO, MITTENTE, TITOLO_TEST)

3.4 Lista dei vincoli inter-relazionali.
Elenco dei vincoli inter-relazionali che intercorrono tra le differenti tabelle.

Test.MAIL_DOCENTE > Docente.MAIL
Tabella.MAIL_DOCENTE > Docente.MAIL
Attributo.NOME_TABELLA > Tabella.NOME
Vincolo_Integrità.ATTRIBUTO_RIFERITO > Attributo.NOME
Vincolo_Integrità.TABELLA_RIFERITA > Tabella.NOME
Quesito_Risposta_Chiusa.TITOLO_TEST > Test.TITOLO
Quesito_Codice.TITOLO_TEST > Test.TITOLO
Appartenenza_Quesito_Chiuso.ID_QUESITO > Quesito_Risposta_Chiusa.ID
Appartenenza_Quesito_Chiuso.NOME_TABELLA > Tabella.NOME
Appartenenza_Quesito_Chiuso.TITOLO_TEST > Quesito_Risposta_Chiusa.TITOLO_TEST
Appartenenza_Quesito_Codice.ID_QUESITO > Quesito_Codice.ID
Appartenenza_Quesito_Codice.NOME_TABELLA > Tabella.NOME
Appartenenza_Quesito_Codice.TITOLO_TEST > Quesito_Codice TITOLO_TEST
Completamento.MAIL_STUDENTE > Studente.MAIL

Completamento.TITOLO_TEST > Test.TITOLO
Risposta_Quesito_Chiuso.MAIL_STUDENTE > Studente.MAIL
Risposta_Quesito_Chiuso.SCELTA > Opzione.ID_OPZIONE
Risposta_Quesito_Chiuso.TITOLO_TEST > Quesito_Risposta_Chiusa.TITOLO_TEST
Risposta_Quesito_Chiuso.ID_QUESITO > Quesito_Risposta_Chiusa.ID
Risposta_Quesito_Codice.MAIL_STUDENTE > Studente.MAIL
Risposta_Quesito_Codice.TITOLO_TEST > Quesito_Codice.TITOLO_TEST
Risposta_Quesito_Codice.ID_QUESITO > Quesito_Codice.ID
Messaggio_Docente.MITTENTE > Docente.MAIL
Messaggio_Docente.TITOLO_TEST > Test.TITOLO
Messaggio_Studente.MITTENTE > Studente.MAIL
Messaggio_Studente.TITOLO_TEST > Test.TITOLO
Messaggio_Studente.DESTINATARIO > Docente.MAIL

4) Descrizione delle funzionalità dell'applicazione Web.

Registrazione

Studenti e docenti devono registrarsi per poter utilizzare la piattaforma. A secondo del ruolo che ricoprono gli saranno richiesti dati differenti.

Accesso/Logout

Una volta che si sono registrati sia i docenti che gli studenti, possono accedere alla piattaforma con le credenziali create in fase di registrazione.
Inoltre, una volta terminato l'utilizzo della piattaforma è possibile per un utente effettuare il logout e quindi disconnettersi dalla piattaforma.

Lato Docente

Durante l'utilizzo della piattaforma ad un docente è permesso: creare nuovi test, nuove tabelle d'esercizio, nuovi quesiti e nuovi vincoli. È consentito inoltre popolare tabelle esistenti aggiungendo delle righe, visualizzare i test ed i relativi quesiti, visualizzare statistiche e gestire l'amministrazione di Log Eventi.

Lato Studente

Per quanto riguarda gli studenti invece, gli è consentito di: completare i test rispondendo ai quesiti, visualizzare i test ed i relativi quesiti, visualizzare statistiche e visualizzare l'esito di una risposta.

Messaggi

Sia ai docenti che agli studenti è consentito inviare messaggi e riceverli. Uno studente invierà il messaggio ad un docente specifico mentre un docente invierà il messaggio a tutti gli studenti. Inoltre, entrambi i tipi di utente dispongono di una sezione con i propri messaggi ricevuti.

5) Codice SQL completo dello schema della base di dati.

```
CREATE DATABASE IF NOT EXISTS ESQLDB;

USE ESQLDB;
SET SQL_SAFE_UPDATES=0;

CREATE TABLE DOCENTE(
        MAIL VARCHAR(60),
        NOME VARCHAR(20),
        COGNOME VARCHAR(20),
        RECAPITO VARCHAR(20),
   DIPARTIMENTO VARCHAR(20) NOT NULL,
   CORSO VARCHAR(20) NOT NULL,
   PASSWORD VARCHAR(20) NOT NULL,

   PRIMARY KEY (MAIL)
)
ENGINE=INNODB;

CREATE TABLE STUDENTE(
        MAIL VARCHAR(60),
        NOME VARCHAR(20),
```

```
        COGNOME VARCHAR(20),
        RECAPITO VARCHAR(20),
    ANNO_IMMATRICOLAZIONE CHAR(4) NOT NULL,
    CODICE CHAR(16) NOT NULL,
    PASSWORD VARCHAR(60) NOT NULL,

        PRIMARY KEY (MAIL)
)
ENGINE=INNODB;

CREATE TABLE TEST (
    TITOLO VARCHAR(20) NOT NULL,
    DATA_CREAZIONE DATE,
    FOTO LONGBLOB,
    MAIL_DOCENTE VARCHAR(20) NOT NULL,
    VISUALIZZA_RISPOSTE BOOLEAN DEFAULT 0,

    PRIMARY KEY (TITOLO),
    FOREIGN KEY (MAIL_DOCENTE) REFERENCES DOCENTE(MAIL)
)
ENGINE=INNODB;


CREATE TABLE MESSAGGIO_DOCENTE(
    TITOLO_MESSAGGIO VARCHAR(20),
    TESTO VARCHAR(255),
    DATA_INSERIMENTO DATE,
    TITOLO_TEST VARCHAR(20),
    MITTENTE VARCHAR(20),
```

```
    PRIMARY KEY (TITOLO_MESSAGGIO, MITTENTE),
    FOREIGN KEY (TITOLO_TEST) REFERENCES TEST(TITOLO),
    FOREIGN KEY (MITTENTE) REFERENCES DOCENTE(MAIL)
  )
ENGINE=INNODB;

CREATE TABLE MESSAGGIO_STUDENTE(
    TITOLO_MESSAGGIO VARCHAR(20),
    TESTO VARCHAR(255),
    DATA_INSERIMENTO DATE,
    TITOLO_TEST VARCHAR(20),
    MITTENTE VARCHAR(20),
    DESTINATARIO VARCHAR(20),

    PRIMARY KEY (TITOLO_MESSAGGIO, MITTENTE, DESTINATARIO),
    FOREIGN KEY (TITOLO_TEST) REFERENCES TEST(TITOLO),
    FOREIGN KEY (MITTENTE) REFERENCES STUDENTE(MAIL),
    FOREIGN KEY (DESTINATARIO) REFERENCES DOCENTE(MAIL)
  )
ENGINE=INNODB;


CREATE TABLE IF NOT EXISTS SEQUENZA_ID_QUESITI (
            TITOLO_TEST VARCHAR(20),
            ID_QUESITO INT NOT NULL,

    PRIMARY KEY(TITOLO_TEST)
)
```

```
ENGINE = INNODB;

CREATE INDEX idx_id_quesito ON sequenza_id_quesiti (ID_QUESITO);



CREATE TABLE QUESITO_RISPOSTA_CHIUSA (
   ID INT NOT NULL,
   LIVELLO ENUM('BASSO', 'MEDIO', 'ALTO') NOT NULL,
   TITOLO_TEST VARCHAR(60) NOT NULL,
   DESCRIZIONE VARCHAR(60),
   NUM_RISPOSTE INT DEFAULT 0,

   FOREIGN KEY (TITOLO_TEST) REFERENCES TEST(TITOLO),
   PRIMARY KEY(ID, TITOLO_TEST)
)
ENGINE=INNODB;

CREATE TABLE QUESITO_CODICE (
   ID INT NOT NULL,
   LIVELLO ENUM('BASSO', 'MEDIO', 'ALTO') NOT NULL,
   TITOLO_TEST VARCHAR(60) NOT NULL,
   DESCRIZIONE VARCHAR(60),
   NUM_RISPOSTE INT DEFAULT 0,

   FOREIGN KEY (TITOLO_TEST) REFERENCES TEST(TITOLO),
   PRIMARY KEY(ID, TITOLO_TEST)
)
ENGINE=INNODB;
```

```
CREATE TABLE TABELLA(
   NOME VARCHAR(60),
   DATA_CREAZIONE DATE,
   NUMRIGHE INT,
   MAIL_DOCENTE VARCHAR(60),

   PRIMARY KEY (NOME),
   FOREIGN KEY (MAIL_DOCENTE) REFERENCES DOCENTE(MAIL)
)
ENGINE=INNODB;

CREATE TABLE ATTRIBUTO (
   NOME VARCHAR(20),
   TIPO VARCHAR(255) NOT NULL CHECK (TIPO LIKE 'varchar%' OR TIPO     IN ('INT',
'BOOLEAN', 'DATE', 'BLOB')),
   NOME_TABELLA VARCHAR(60),
   FOREIGN KEY (NOME_TABELLA) REFERENCES TABELLA(NOME),

   PRIMARY KEY (NOME, NOME_TABELLA)
)
ENGINE=INNODB;

CREATE TABLE IF NOT EXISTS SEQUENZA_ID_OPZIONI (
         ID INT NOT NULL AUTO_INCREMENT,
         TITOLO_TEST VARCHAR(20),
         ID_QUESITO INT NOT NULL,
   ID_OPZIONE INT NOT NULL,

   PRIMARY KEY(ID)
)
```

```
ENGINE = INNODB;

CREATE INDEX idx_id_opzione ON sequenza_id_opzioni (ID_OPZIONE);

CREATE TABLE OPZIONE(
        ID_OPZIONE INT,
   TESTO VARCHAR(50) NOT NULL,
   ID_QUESITO INT REFERENCES QUESITO_RISPOSTA_CHIUSA.ID,
   TITOLO_TEST VARCHAR(20) REFERENCES QUESITO_RISPOSTA_CHIUSA.TITOLO_TEST,
   CORRETTA BOOLEAN NOT NULL DEFAULT FALSE,

   PRIMARY KEY(ID_OPZIONE, ID_QUESITO, TITOLO_TEST)
   )
ENGINE=INNODB;

CREATE TABLE IF NOT EXISTS SEQUENZA_ID_SKETCH (
        ID INT NOT NULL AUTO_INCREMENT,
        TITOLO_TEST VARCHAR(20),
        ID_QUESITO INT NOT NULL,
   ID_SKETCH INT NOT NULL,

   PRIMARY KEY(ID)
)
ENGINE = INNODB;

CREATE INDEX idx_id_sketch ON sequenza_id_sketch (ID_SKETCH);

CREATE TABLE SKETCH(
        ID_SKETCH INT,
```

```
    TESTO VARCHAR(255) NOT NULL,
    TITOLO_TEST VARCHAR(60) REFERENCES QUESITO_CODICE.TITOLO_TEST,
    ID_QUESITO INT REFERENCES QUESITO_CODICE.ID,

    PRIMARY KEY(ID_SKETCH, TITOLO_TEST, ID_QUESITO)
)
ENGINE=INNODB;



CREATE TABLE RISPOSTA_QUESITO_CHIUSO (
        MAIL_STUDENTE VARCHAR(20),
    ESITO BOOLEAN,
    SCELTA INT,
    ID_QUESITO INT,
    TITOLO_TEST VARCHAR(20),

    PRIMARY KEY (MAIL_STUDENTE, ID_QUESITO, TITOLO_TEST),
    FOREIGN KEY (MAIL_STUDENTE) REFERENCES STUDENTE(MAIL),
    FOREIGN KEY(SCELTA) REFERENCES OPZIONE(ID_OPZIONE),
    FOREIGN KEY (ID_QUESITO) REFERENCES QUESITO_RISPOSTA_CHIUSA(ID),
    FOREIGN KEY(TITOLO_TEST) REFERENCES QUESITO_RISPOSTA_CHIUSA(TITOLO_TEST)

)
ENGINE=INNODB;

CREATE TABLE RISPOSTA_QUESITO_CODICE(
        MAIL_STUDENTE VARCHAR(20),
    TESTO VARCHAR(255),
    ESITO BOOLEAN,
    ID_QUESITO INT,
```

```
    TITOLO_TEST VARCHAR(20),

    PRIMARY KEY (MAIL_STUDENTE, ID_QUESITO, TITOLO_TEST),
    FOREIGN KEY (ID_QUESITO) REFERENCES QUESITO_CODICE(ID),
    FOREIGN KEY (MAIL_STUDENTE) REFERENCES STUDENTE(MAIL),
    FOREIGN KEY (ID_QUESITO) REFERENCES QUESITO_CODICE(ID)
)
ENGINE=INNODB;



CREATE TABLE COMPLETAMENTO (
    DATA_PRIMA_RISPOSTA DATE,
    DATA_ULTIMA_RISPOSTA DATE,
    STATO ENUM('APERTO', 'IN_COMPLETAMENTO', 'CONCLUSO'),
    MAIL_STUDENTE VARCHAR(20),
        TITOLO_TEST VARCHAR(20) NOT NULL,

    PRIMARY KEY (MAIL_STUDENTE, TITOLO_TEST),
    FOREIGN KEY (MAIL_STUDENTE) REFERENCES STUDENTE(MAIL),
    FOREIGN KEY (TITOLO_TEST) REFERENCES TEST(TITOLO)
)
ENGINE=INNODB;

CREATE TABLE VINCOLO_INTEGRITA (
        TABELLA_REFERENTE VARCHAR(20),
        ATTRIBUTO_REFERENTE VARCHAR(20),
    TABELLA_RIFERITA VARCHAR(20),
        ATTRIBUTO_RIFERITO VARCHAR(20),
```

```
   PRIMARY KEY(TABELLA_REFERENTE, ATTRIBUTO_REFERENTE, TABELLA_RIFERITA,
ATTRIBUTO_RIFERITO),
   FOREIGN KEY(ATTRIBUTO_RIFERITO) REFERENCES ATTRIBUTO(NOME),
   FOREIGN KEY(TABELLA_RIFERITA) REFERENCES TABELLA(NOME)
)
ENGINE=INNODB;

CREATE TABLE APPARTENENZA_QUESITO_CHIUSO(
        NOME_TABELLA VARCHAR(20),
   TITOLO_TEST VARCHAR(20) NOT NULL,
   ID_QUESITO INT NOT NULL,

   PRIMARY KEY(NOME_TABELLA, TITOLO_TEST, ID_QUESITO),
   FOREIGN KEY (NOME_TABELLA) REFERENCES TABELLA(NOME),
   FOREIGN KEY(TITOLO_TEST) REFERENCES QUESITO_RISPOSTA_CHIUSA(TITOLO_TEST),
   FOREIGN KEY(ID_QUESITO) REFERENCES QUESITO_RISPOSTA_CHIUSA(ID)
)
ENGINE = INNODB;



CREATE TABLE APPARTENENZA_QUESITO_CODICE(
        NOME_TABELLA VARCHAR(20),
   TITOLO_TEST VARCHAR(20) NOT NULL,
   ID_QUESITO INT NOT NULL,

   PRIMARY KEY(NOME_TABELLA, TITOLO_TEST, ID_QUESITO),
   FOREIGN KEY(TITOLO_TEST) REFERENCES QUESITO_CODICE(TITOLO_TEST),
   FOREIGN KEY (NOME_TABELLA) REFERENCES TABELLA(NOME),
   FOREIGN KEY(ID_QUESITO) REFERENCES QUESITO_CODICE(ID)
```

```
)
ENGINE = INNODB;



-- ------ VISTE ------------------------------
DELIMITER $
CREATE VIEW CLASSIFICA_STUDENTI_TEST_COMPLETATI AS
SELECT STUDENTE.CODICE, COUNT(STATO) AS NUMERO_TEST_COMPLETATI
FROM STUDENTE
JOIN COMPLETAMENTO ON MAIL = MAIL_STUDENTE
WHERE STATO = 'CONCLUSO'
GROUP BY CODICE
ORDER BY NUMERO_TEST_COMPLETATI DESC;
$

DELIMITER $
CREATE VIEW VISTA_RISPOSTE AS
SELECT RQ.ID_QUESITO, RQ.TITOLO_TEST, S.CODICE, RQ.ESITO
FROM RISPOSTA_QUESITO_CHIUSO AS RQ, STUDENTE AS S
WHERE S.MAIL = RQ.MAIL_STUDENTE
UNION ALL
SELECT RC.ID_QUESITO, RC.TITOLO_TEST, S.CODICE, RC.ESITO
FROM RISPOSTA_QUESITO_CODICE AS RC, STUDENTE AS S
WHERE S.MAIL= RC.MAIL_STUDENTE;
$

DELIMITER $
CREATE VIEW CLASSIFICA_STUDENTI_QUESITI_CORRETTI AS
SELECT CODICE, (100*SUM(CASE WHEN ESITO=TRUE THEN 1 ELSE 0 END)/ COUNT(*)) AS
RISPOSTE_CORRETTE
```

```
FROM VISTA_RISPOSTE
GROUP BY CODICE
ORDER BY RISPOSTE_CORRETTE DESC;
$

CREATE VIEW QUESITI_COMPLETATI_MAGGIORMENTE AS
SELECT ID_QUESITO, TITOLO_TEST, SUM(NUMERO_RISPOSTE) AS NUMERO_RISPOSTE
FROM (
   SELECT ID_QUESITO, TITOLO_TEST, COUNT(ID_QUESITO) AS NUMERO_RISPOSTE
   FROM RISPOSTA_QUESITO_CODICE
   GROUP BY ID_QUESITO, TITOLO_TEST

   UNION ALL

   SELECT ID_QUESITO, TITOLO_TEST, COUNT(ID_QUESITO) AS NUMERO_RISPOSTE
   FROM RISPOSTA_QUESITO_CHIUSO
   GROUP BY ID_QUESITO, TITOLO_TEST
) AS Subquery

GROUP BY ID_QUESITO, TITOLO_TEST
ORDER BY NUMERO_RISPOSTE DESC;
$
-- ------ PROCEDURE ------------------------------
DELIMITER $
CREATE PROCEDURE VISUALIZZAZIONE_TEST_DISPONIBILI()
        SELECT TITOLO, DATA_CREAZIONE, VISUALIZZA_RISPOSTE FROM TEST;
$

DELIMITER $
```

```
CREATE PROCEDURE VISUALIZZAZIONE_QUESITI(IN TITOLO VARCHAR(20))
        SELECT ID, LIVELLO, DESCRIZIONE
   FROM QUESITO_RISPOSTA_CHIUSA
        WHERE TITOLO = TITOLO_TEST
   UNION
   SELECT ID, LIVELLO, DESCRIZIONE
   FROM QUESITO_CODICE
   WHERE TITOLO = TITOLO_TEST
   ORDER BY ID ASC;
$


DELIMITER $
CREATE PROCEDURE INSERIMENTO_NUOVO_TEST(IN TITOLO VARCHAR(20), IN
DATA_CREAZIONE DATE, IN FOTO BLOB, IN MAIL_DOCENTE VARCHAR(20))
BEGIN
   INSERT INTO TEST(TITOLO, DATA_CREAZIONE, FOTO, MAIL_DOCENTE) VALUES(TITOLO,
DATA_CREAZIONE, FOTO, MAIL_DOCENTE);
END $


DELIMITER $
CREATE PROCEDURE INSERIMENTO_QUESITO_CODICE(IN TITOLO_TEST VARCHAR(20), IN
LIVELLO VARCHAR(20), IN DESCRIZIONE VARCHAR(60))
BEGIN
        INSERT INTO QUESITO_CODICE (TITOLO_TEST, LIVELLO, ID, DESCRIZIONE)
        VALUES (TITOLO_TEST, LIVELLO, 0, DESCRIZIONE);


        UPDATE QUESITO_CODICE
        SET ID = (SELECT ID_QUESITO FROM SEQUENZA_ID_QUESITI WHERE
SEQUENZA_ID_QUESITI.TITOLO_TEST = TITOLO_TEST)
   WHERE QUESITO_CODICE.TITOLO_TEST = TITOLO_TEST AND ID = 0;
END $
```

```
DELIMITER ;


DELIMITER $

CREATE PROCEDURE INSERIMENTO_QUESITO_RISPOSTA_CHIUSA(IN TITOLO_TEST
VARCHAR(20), IN LIVELLO VARCHAR(20), IN DESCRIZIONE VARCHAR(60))

BEGIN

        INSERT INTO QUESITO_RISPOSTA_CHIUSA (TITOLO_TEST, LIVELLO, ID, DESCRIZIONE)

        VALUES (TITOLO_TEST, LIVELLO, 0, DESCRIZIONE);


        UPDATE QUESITO_RISPOSTA_CHIUSA

        SET ID = (SELECT ID_QUESITO FROM SEQUENZA_ID_QUESITI WHERE
SEQUENZA_ID_QUESITI.TITOLO_TEST = TITOLO_TEST)

   WHERE QUESITO_RISPOSTA_CHIUSA.TITOLO_TEST = TITOLO_TEST AND ID = 0;

END $

DELIMITER ;


DELIMITER $

CREATE PROCEDURE INSERIMENTO_OPZIONE(IN TITOLO_TEST VARCHAR(20), IN ID_QUESITO
INT, TESTO VARCHAR(50), CORRETTA BOOLEAN)

BEGIN

        INSERT INTO OPZIONE(ID_OPZIONE, TESTO, ID_QUESITO, TITOLO_TEST, CORRETTA)
VALUES(0, TESTO, ID_QUESITO, TITOLO_TEST, CORRETTA);


        UPDATE OPZIONE

   SET ID_OPZIONE = (SELECT ID_OPZIONE

                        FROM SEQUENZA_ID_OPZIONI

         WHERE SEQUENZA_ID_OPZIONI.TITOLO_TEST = TITOLO_TEST AND
SEQUENZA_ID_OPZIONI.ID_QUESITO = ID_QUESITO)

        WHERE OPZIONE.TITOLO_TEST = TITOLO_TEST AND OPZIONE.ID_QUESITO =
ID_QUESITO AND ID_OPZIONE = 0;

END $
```

```
DELIMITER ;



DELIMITER $

CREATE PROCEDURE INSERIMENTO_SKETCH(TESTO VARCHAR(50), IN TITOLO_TEST
VARCHAR(20), IN ID_QUESITO INT)

BEGIN

	INSERT INTO SKETCH(ID_SKETCH, TESTO, ID_QUESITO, TITOLO_TEST) VALUES(0,
TESTO, ID_QUESITO, TITOLO_TEST);


	UPDATE SKETCH

   SET ID_SKETCH = (SELECT ID_SKETCH

				FROM SEQUENZA_ID_SKETCH

	 WHERE SEQUENZA_ID_SKETCH.TITOLO_TEST = TITOLO_TEST AND
SEQUENZA_ID_SKETCH.ID_QUESITO = ID_QUESITO)

	WHERE SKETCH.TITOLO_TEST = TITOLO_TEST AND SKETCH.ID_QUESITO =
ID_QUESITO AND ID_SKETCH = 0;

END $

DELIMITER ;



DELIMITER $

CREATE PROCEDURE INVIO_MESSAGGIO_DOCENTE(IN TITOLO_MESSAGGIO VARCHAR(20),
TESTO VARCHAR(255), DATA_INSERIMENTO DATE, TITOLO_TEST VARCHAR(20), MITTENTE
VARCHAR(20))


	INSERT INTO MESSAGGIO_DOCENTE(TITOLO_MESSAGGIO, TESTO,
DATA_INSERIMENTO, TITOLO_TEST, MITTENTE) VALUES(TITOLO_MESSAGGIO, TESTO,
DATA_INSERIMENTO, TITOLO_TEST, MITTENTE)

$


DELIMITER $
```

```
CREATE PROCEDURE INVIO_MESSAGGIO_STUDENTE(IN TITOLO_MESSAGGIO VARCHAR(20),
TESTO VARCHAR(255), DATA_INSERIMENTO DATE, TITOLO_TEST VARCHAR(20), MITTENTE
VARCHAR(20), DESTINATARIO VARCHAR(20))


        INSERT INTO MESSAGGIO_STUDENTE(TITOLO_MESSAGGIO, TESTO,
DATA_INSERIMENTO, TITOLO_TEST, MITTENTE, DESTINATARIO) VALUES(TITOLO_MESSAGGIO,
TESTO, DATA_INSERIMENTO, TITOLO_TEST, MITTENTE, DESTINATARIO)
$


DELIMITER $
CREATE PROCEDURE CAMBIAMENTO_VISUALIZZAZIONE_RISPOSTE_TEST(IN TITOLO_TEST
VARCHAR(20), MAIL_DOCENTE VARCHAR(20), OPZIONE_SCELTA BOOLEAN)


        UPDATE TEST
   SET VISUALIZZA_RISPOSTE = OPZIONE_SCELTA
   WHERE TEST.TITOLO_TEST = TITOLO_TEST AND TEST.MAIL_DOCENTE = MAIL_DOCENTE
$


DELIMITER $
CREATE PROCEDURE INSERIMENTO_NUOVO_STUDENTE(IN MAIL VARCHAR(20),IN NOME
VARCHAR(20),IN COGNOME VARCHAR(20),IN RECAPITO VARCHAR(20),IN
ANNO_IMMATRICOLAZIONE CHAR(4),
IN CODICE CHAR(16), IN PAZZWORD VARCHAR(60) )


        INSERT INTO STUDENTE(MAIL, NOME, COGNOME, RECAPITO,
ANNO_IMMATRICOLAZIONE, CODICE, PAZZWORD) VALUES(MAIL, NOME, COGNOME,
RECAPITO, ANNO_IMMATRICOLAZIONE, CODICE, PAZZWORD)
$


DELIMITER $
CREATE PROCEDURE CAMBIO_STATO_TEST (IN MAIL VARCHAR(60), IN TITOLO_TEST
VARCHAR(60))
BEGIN
```

```
   IF ((SELECT COUNT(*) FROM COMPLETAMENTO WHERE MAIL_STUDENTE = MAIL AND
COMPLETAMENTO.TITOLO_TEST = TITOLO_TEST) = 0) THEN

      INSERT INTO COMPLETAMENTO (DATA_PRIMA_RISPOSTA, DATA_ULTIMA_RISPOSTA,
STATO, MAIL_STUDENTE, TITOLO_TEST)

      VALUES (null, null, 'APERTO', MAIL, TITOLO_TEST);

   END IF;

END $

DELIMITER ;



DELIMITER $$

CREATE PROCEDURE inserisci_risposta_quesito_chiuso(

   IN mail_studente VARCHAR(20),

   IN id_quesito INT,

   IN titolo_test VARCHAR(20),

   IN scelta INT

)

BEGIN

DECLARE C int;

SELECT COUNT(*)  INTO C FROM RISPOSTA_QUESITO_CHIUSO WHERE MAIL_STUDENTE =
RISPOSTA_QUESITO_CHIUSO.MAIL_STUDENTE

                  AND ID_QUESITO = RISPOSTA_QUESITO_CHIUSO.ID_QUESITO AND
TITOLO_TEST = RISPOSTA_QUESITO_CHIUSO.TITOLO_TEST;

         IF(C = 0)

      THEN

                  INSERT INTO RISPOSTA_QUESITO_CHIUSO (MAIL_STUDENTE, ID_QUESITO,
TITOLO_TEST, scelta, ESITO)

                  VALUES (mail_studente, id_quesito, titolo_test, scelta, esito);

   END IF;


   IF(C <> 0)
```

```
      THEN

    UPDATE RISPOSTA_QUESITO_CHIUSO SET RISPOSTA_QUESITO_CHIUSO.SCELTA = SCELTA
WHERE MAIL_STUDENTE = RISPOSTA_QUESITO_CHIUSO.MAIL_STUDENTE

                AND ID_QUESITO = RISPOSTA_QUESITO_CHIUSO.ID_QUESITO AND
TITOLO_TEST = RISPOSTA_QUESITO_CHIUSO.TITOLO_TEST;

        END IF;


   IF (SCELTA IN (SELECT ID_OPZIONE

                                FROM OPZIONE

                                WHERE CORRETTA = 1 AND ID_QUESITO =
OPZIONE.ID_QUESITO AND TITOLO_TEST = OPZIONE.TITOLO_TEST)) THEN

                UPDATE RISPOSTA_QUESITO_CHIUSO

                SET ESITO = 1 WHERE ID_QUESITO =
RISPOSTA_QUESITO_CHIUSO.ID_QUESITO AND TITOLO_TEST =
RISPOSTA_QUESITO_CHIUSO.TITOLO_TEST AND MAIL_STUDENTE =
RISPOSTA_QUESITO_CHIUSO.MAIL_STUDENTE;


        END IF;


   IF (SCELTA NOT IN (SELECT ID_OPZIONE

                                FROM OPZIONE

                                WHERE CORRETTA = 1 AND ID_QUESITO =
OPZIONE.ID_QUESITO AND TITOLO_TEST = OPZIONE.TITOLO_TEST)) THEN

                UPDATE RISPOSTA_QUESITO_CHIUSO

                SET ESITO = 0 WHERE ID_QUESITO =
RISPOSTA_QUESITO_CHIUSO.ID_QUESITO AND TITOLO_TEST =
RISPOSTA_QUESITO_CHIUSO.TITOLO_TEST AND MAIL_STUDENTE =
RISPOSTA_QUESITO_CHIUSO.MAIL_STUDENTE;

        END IF;


END;$$

DELIMITER $
```

```
CREATE PROCEDURE inserisci_risposta_quesito_codice(
   IN mail_studente VARCHAR(20),
   IN id_quesito INT,
   IN titolo_test VARCHAR(20),
   IN testo_risposta VARCHAR(255)
)
BEGIN
DECLARE C int;
SELECT COUNT(*)  INTO C FROM RISPOSTA_QUESITO_CODICE WHERE MAIL_STUDENTE = RISPOSTA_QUESITO_CODICE.MAIL_STUDENTE
                AND ID_QUESITO = RISPOSTA_QUESITO_CODICE.ID_QUESITO AND TITOLO_TEST = RISPOSTA_QUESITO_CODICE.TITOLO_TEST;
        IF(C = 0)
      THEN
                INSERT INTO RISPOSTA_QUESITO_CODICE (MAIL_STUDENTE, ID_QUESITO, TITOLO_TEST, TESTO, ESITO)
                VALUES (mail_studente, id_quesito, titolo_test, testo_risposta, esito);
   END IF;

   IF(C <> 0)
      THEN
    UPDATE RISPOSTA_QUESITO_CODICE SET TESTO = TESTO_RISPOSTA WHERE MAIL_STUDENTE = RISPOSTA_QUESITO_CODICE.MAIL_STUDENTE
                AND ID_QUESITO = RISPOSTA_QUESITO_CODICE.ID_QUESITO AND TITOLO_TEST = RISPOSTA_QUESITO_CODICE.TITOLO_TEST;
        END IF;

        IF(TESTO_RISPOSTA IN(SELECT TESTO FROM SKETCH WHERE ID_QUESITO = SKETCH.ID_QUESITO AND TITOLO_TEST = SKETCH.TITOLO_TEST)) THEN
          UPDATE RISPOSTA_QUESITO_CODICE SET ESITO = 1 WHERE MAIL_STUDENTE = RISPOSTA_QUESITO_CODICE.MAIL_STUDENTE
```

```
                    AND TITOLO_TEST = RISPOSTA_QUESITO_CODICE.TITOLO_TEST AND
ID_QUESITO = RISPOSTA_QUESITO_CODICE.ID_QUESITO;
   END IF;
   IF(TESTO_RISPOSTA NOT IN(SELECT TESTO FROM SKETCH WHERE ID_QUESITO =
SKETCH.ID_QUESITO AND TITOLO_TEST = SKETCH.TITOLO_TEST)) THEN
                    UPDATE RISPOSTA_QUESITO_CODICE SET ESITO = 0 WHERE MAIL_STUDENTE
= RISPOSTA_QUESITO_CODICE.MAIL_STUDENTE
      AND TITOLO_TEST = RISPOSTA_QUESITO_CODICE.TITOLO_TEST AND ID_QUESITO =
RISPOSTA_QUESITO_CODICE.ID_QUESITO;
          END IF;



END;$
DELIMITER ;


-- ------ TRIGGER ------------------------------


DELIMITER //
CREATE TRIGGER CAMBIO_STATO_COMPLETAMENTO_QUESITO_CODICE
AFTER INSERT ON RISPOSTA_QUESITO_CODICE
FOR EACH ROW
BEGIN
   DECLARE num_risposte INT;
   DECLARE DATA date;
   SELECT CURDATE() INTO DATA;


   SELECT COUNT(*) INTO num_risposte FROM RISPOSTA_QUESITO_CODICE WHERE
ID_QUESITO = NEW.ID_QUESITO AND MAIL_STUDENTE = NEW.MAIL_STUDENTE AND
TITOLO_TEST = NEW.TITOLO_TEST;


   -- Se è la prima risposta, aggiorna lo stato del test
```

```
   IF num_risposte = 1 THEN
      UPDATE COMPLETAMENTO SET STATO = 'IN_COMPLETAMENTO',
DATA_PRIMA_RISPOSTA = DATA
      WHERE MAIL_STUDENTE = NEW.MAIL_STUDENTE AND TITOLO_TEST =
NEW.TITOLO_TEST;
   END IF;
END; //


 DELIMITER //
CREATE TRIGGER CAMBIO_STATO_COMPLETAMENTO_QUESITO_CHIUSO
AFTER INSERT ON RISPOSTA_QUESITO_CHIUSO
FOR EACH ROW
BEGIN
   DECLARE num_risposte INT;
   DECLARE DATA DATE;

   SELECT curdate() INTO DATA;
   SELECT COUNT(*) INTO num_risposte FROM RISPOSTA_QUESITO_CHIUSO WHERE
ID_QUESITO = NEW.ID_QUESITO AND MAIL_STUDENTE = NEW.MAIL_STUDENTE AND
TITOLO_TEST = NEW.TITOLO_TEST;

   IF num_risposte = 1 THEN
      UPDATE COMPLETAMENTO SET STATO = 'IN_COMPLETAMENTO',
DATA_PRIMA_RISPOSTA = DATA
      WHERE MAIL_STUDENTE = NEW.MAIL_STUDENTE AND TITOLO_TEST =
NEW.TITOLO_TEST;
   END IF;

END;//

DELIMITER //
```

```
CREATE TRIGGER AGGIUNTA_SEQUENZA_QUESITI
AFTER INSERT ON TEST
FOR EACH ROW
BEGIN
   INSERT INTO SEQUENZA_ID_QUESITI (TITOLO_TEST, ID_QUESITO) VALUES (NEW.TITOLO,
0);
END;
//

DELIMITER //
CREATE TRIGGER AGGIUNTA_SEQUENZA_OPZIONI
AFTER INSERT ON OPZIONE
FOR EACH ROW
BEGIN
   DECLARE riga_esistente INT;

   SELECT COUNT(*) INTO riga_esistente
   FROM SEQUENZA_ID_OPZIONI
   WHERE TITOLO_TEST = NEW.TITOLO_TEST AND ID_QUESITO = NEW.ID_QUESITO;

   IF riga_esistente = 0 THEN
      INSERT INTO SEQUENZA_ID_OPZIONI (TITOLO_TEST, ID_QUESITO, ID_OPZIONE)
      VALUES (NEW.TITOLO_TEST, NEW.ID_QUESITO, 0);
   END IF;
END;//
DELIMITER ;

DELIMITER //
CREATE TRIGGER AGGIUNTA_SEQUENZA_SKETCH
AFTER INSERT ON SKETCH
```

```
FOR EACH ROW
BEGIN
   DECLARE riga_esistente INT;

   SELECT COUNT(*) INTO riga_esistente
   FROM SEQUENZA_ID_SKETCH
   WHERE TITOLO_TEST = NEW.TITOLO_TEST AND ID_QUESITO = NEW.ID_QUESITO;

   IF riga_esistente = 0 THEN
      INSERT INTO SEQUENZA_ID_SKETCH (TITOLO_TEST, ID_QUESITO, ID_SKETCH)
      VALUES (NEW.TITOLO_TEST, NEW.ID_QUESITO, 0);
   END IF;
END;//
DELIMITER ;

DELIMITER //
CREATE TRIGGER AGGIORNAMENTO_SEQUENZA_QUESITI_RISPOSTA_CHIUSA
AFTER INSERT ON QUESITO_RISPOSTA_CHIUSA
FOR EACH ROW
BEGIN
   UPDATE SEQUENZA_ID_QUESITI
   SET ID_QUESITO = ID_QUESITO + 1
   WHERE TITOLO_TEST = NEW.TITOLO_TEST;

END;
//

DELIMITER //
CREATE TRIGGER AGGIORNAMENTO_SEQUENZA_OPZIONI
```

```
AFTER INSERT ON OPZIONE
FOR EACH ROW
BEGIN
   UPDATE SEQUENZA_ID_OPZIONI
   SET ID_OPZIONE= ID_OPZIONE + 1
   WHERE TITOLO_TEST = NEW.TITOLO_TEST AND ID_QUESITO = NEW.ID_QUESITO;

END;
//

DELIMITER //
CREATE TRIGGER AGGIORNAMENTO_SEQUENZA_SKETCH
AFTER INSERT ON SKETCH
FOR EACH ROW
BEGIN
   UPDATE SEQUENZA_ID_SKETCH
   SET ID_SKETCH= ID_SKETCH + 1
   WHERE TITOLO_TEST = NEW.TITOLO_TEST AND ID_QUESITO = NEW.ID_QUESITO;

END;
//

DELIMITER //

CREATE TRIGGER AGGIORNAMENTO_SEQUENZA_QUESITI_CHIUSI
AFTER INSERT ON QUESITO_CODICE
FOR EACH ROW
BEGIN
   UPDATE SEQUENZA_ID_QUESITI
```

```
   SET ID_QUESITO = ID_QUESITO + 1
   WHERE TITOLO_TEST = NEW.TITOLO_TEST;
END;
//



DELIMITER //
CREATE TRIGGER CAMBIO_STATO_CONCLUSO
AFTER INSERT ON RISPOSTA_QUESITO_CODICE
FOR EACH ROW
BEGIN
DECLARE num_quesiti INT;
DECLARE num_risposte_chiuse_corrette INT;
DECLARE num_risposte_codice_corrette INT;

SELECT COUNT(*) INTO num_quesiti FROM QUESITO_RISPOSTA_CHIUSA, QUESITO_CODICE
WHERE QUESITO_RISPOSTA_CHIUSA.TITOLO_TEST = QUESITO_CODICE.TITOLO_TEST AND
QUESITO_RISPOSTA_CHIUSA.TITOLO_TEST = NEW.TITOLO_TEST;
SELECT COUNT(*) INTO num_risposte_chiuse_corrette FROM RISPOSTA_QUESITO_CHIUSO
WHERE MAIL_STUDENTE = NEW.MAIL_STUDENTE AND TITOLO_TEST = NEW.TITOLO_TEST
AND ESITO = 1;
SELECT COUNT(*) INTO num_risposte_codice_corrette FROM RISPOSTA_QUESITO_CODICE
WHERE MAIL_STUDENTE = NEW.MAIL_STUDENTE AND TITOLO_TEST = NEW.TITOLO_TEST
AND ESITO = 1;

IF(num_quesiti = num_risposte_chiuse_corrette + num_risposte_codice_corrette) THEN
        UPDATE COMPLETAMENTO SET STATO = 'CONCLUSO'
      WHERE MAIL_STUDENTE = NEW.MAIL_STUDENTE AND TITOLO_TEST =
NEW.TITOLO_TEST;
END IF;
END //
DELIMITER ;
```

```
DELIMITER //
CREATE TRIGGER CAMBIO_STATO_TUTTICONCLUSO
AFTER UPDATE ON TEST
FOR EACH ROW
BEGIN

   IF OLD.VISUALIZZA_RISPOSTE = 0 AND NEW.VISUALIZZA_RISPOSTE = 1 THEN
      UPDATE COMPLETAMENTO SET STATO = 'CONCLUSO' WHERE TITOLO_TEST =
NEW.TITOLO;
   END IF;
END //
DELIMITER ;

DELIMITER //
CREATE TRIGGER CAMBIA_DATA_ULTIMA_RISPOSTA_CHIUSA
AFTER INSERT ON RISPOSTA_QUESITO_CHIUSO
FOR EACH ROW
BEGIN
        DECLARE DATA VARCHAR(20);
        SET DATA = CURDATE();
   UPDATE COMPLETAMENTO SET DATA_ULTIMA_RISPOSTA = DATA WHERE
COMPLETAMENTO.MAIL_STUDENTE = NEW.MAIL_STUDENTE AND
COMPLETAMENTO.TITOLO_TEST = NEW.TITOLO_TEST;
END//
DELIMITER ;

DELIMITER //
CREATE TRIGGER CAMBIA_DATA_ULTIMA_RISPOSTA_CODICE
```

```
   AFTER INSERT ON RISPOSTA_QUESITO_CODICE
   FOR EACH ROW
BEGIN
   DECLARE DATA VARCHAR(20);
        SET DATA = CURDATE();
   UPDATE COMPLETAMENTO SET DATA_ULTIMA_RISPOSTA = DATA WHERE
COMPLETAMENTO.MAIL_STUDENTE = NEW.MAIL_STUDENTE AND
COMPLETAMENTO.TITOLO_TEST = NEW.TITOLO_TEST;
END//
DELIMITER ;



DELIMITER //
CREATE TRIGGER AGGIORNAMENTO_NUM_RISPOSTE_QUESITO_CHIUSO
AFTER INSERT ON RISPOSTA_QUESITO_CHIUSO
FOR EACH ROW
BEGIN
        UPDATE QUESITO_RISPOSTA_CHIUSA SET NUM_RISPOSTE = NUM_RISPOSTE + 1
WHERE TITOLO_TEST = NEW.TITOLO_TEST AND ID = NEW.ID_QUESITO;
END//


DELIMITER //
CREATE TRIGGER AGGIORNAMENTO_NUM_RISPOSTE_QUESITO_CODICE
AFTER INSERT ON RISPOSTA_QUESITO_CODICE
FOR EACH ROW
BEGIN
        UPDATE QUESITO_CODICE SET NUM_RISPOSTE = NUM_RISPOSTE + 1 WHERE
TITOLO_TEST = NEW.TITOLO_TEST AND ID = NEW.ID_QUESITO;
END//
```