RELAZIONE TECNICA

Alessio Tuscano

A.S. 2025–2026

Contents

Progetto Applicazione Web

Introduzione del Progetto

Il progetto prevede la realizzazione di un'applicazione web completa, sviluppata progressivamente attraverso diverse fasi. Ogni fase si concentra su un aspetto specifico dello sviluppo, costruendo gradualmente un sistema funzionante e professionale.

La ***prima fase*** comprende:

1. **Analisi** dei requisiti
2. **Diagramma E-R** realizzato con Mermaid
3. **Schema logico**
4. **DDL**
5. **Dizionario** dei dati
6. **Cocnlusioni**
7. **Relazione tecnica** finale del progetto realizzata in markdown

Analisi dei Requisiti

Il sistema prevede un database per la gestione di un centro polispecialistico. Fornirà la digitalizzazione dei dati dei pazienti (anagrafici e clinici), la gestione dei medici con le rispettive specializzazioni e gli orari di lavoro, le prenotazioni e la conservazione storica degli esami diagnostici. Sono inoltre previste funzionalità per l'assegnazione delle sale ambulatoriali nei diversi reparti e per la gestione dei pagamenti e della fatturazione.

Diagramma Entità-Relazione (E-R)

```
erDiagram
    FATTURA o|--|| PAGAMENTO : riferisce
    PAGAMENTO ||--o{ PAZIENTE : paga
    PRENOTAZIONE ||--|{ PAZIENTE : prenota
    STORICO ||--o{ PAZIENTE : prenota
    STORICO |{--o{ PRENOTAZIONE : prenota
    MEDICO o{--|| PRENOTAZIONE : chiamato
    MEDICO ||--|{ REPARTO : lavora
    REPARTO |{--o| AMBULATORIO : sta
    ESAME ||--o{ AMBULATORIO : svolto
    ESAME ||--o{ MEDICO : effettua
    MEDICO |{--|{ ORARIOLAVORO : turno
    MEDICO o{--|| SPECIALIZZAZIONE : possiede
    MEDICO ||--o{ MEDICO_ORARIOLAVORO : ausiliaria
    MEDICO_ORARIOLAVORO }o--|| ORARIOLAVORO : ausiliaria

    FATTURA {
        int codiceFattura PK
        int FKPagamento FK
    }

    PAGAMENTO {
        int codicePagamento PK
        string codiceFiscale FK
        date data
        int ora
        int minuti
        double somma
        enum metodo "card / cash"
    }

    PAZIENTE {
        string codiceFiscale PK
        string nome
```

```
    string cognome
    date dataNascita
    string anamnesi
    string ind_cap
    string ind_citta
    string ind_via
    string ind_civico
}

STORICO {
    int codiceEsame PK "FK"
    date data PK
    int oraInizio PK
    string codiceFiscale FK
    int oraFine "NULL"
    string diagnosi
    string prescrizione
}

PRENOTAZIONE {
    int codicePrenotazione PK
    string codiceFiscale FK
    string codiceMedico FK
    date data
    int oraInizio
    int oraFine "NULL"
    string tipoVisita
}

MEDICO {
    string codiceMedico PK
    int codiceReparto FK
    string codiceFiscale "UNIQUE"
    string nome
    string cognome
    binary primario "NULL"
}

REPARTO {
    int codiceReparto PK
    string nomeReparto
}

AMBULATORIO {
    int codiceAmbulatorio PK
    int codiceReparto FK
```

```
    int piano
}

ESAME {
    int codiceEsame PK
    int codiceAmbulatorio FK
    string codiceMedico FK
    string diagnosi
    string referto "NULL"
}

ORARIOLAVORO {
    enum giorno PK "L / M / Me / G / V / S / D"
    int oraInizio PK
    int oraFine
}

SPECIALIZZAZIONE {
    string codiceSpecializzazione PK
    string codiceMedico FK
    enum tipo "Medica / Chirurgia / Clinica"
    string titolo
    date dataConseguimento
    int votoConseguimento
}

MEDICO_ORARIOLAVORO {
    string codiceMedico PK
    enum giorno PK "L / M / Me / G / V / S / D"
    int oraInizio PK
}
```

Schema Logico Relazionale

PAGAMENTO (codicePagamento, codiceFiscale, data, ora, minuti, somma, metodo)

FATTURA (codiceFattura, codicePagamento)

PAZIENTE (codiceFiscale, nome, cognome, dataNascita, anamnesi, ind_cap, ind_citta, ind_via, ind_civico)

STORICO (codiceEsame, data, oraInizio, codiceFiscale, oraFine*, diagnosi, prescrizione)

PRENOTAZIONE (codicePrenotazione, codiceFiscale, codiceMedico, data, oraInizio, oraFine*, tipoVisita)

MEDICO (codiceMedico, codiceReparto, orario, codiceFiscale (UNIQUE), nome, cognome, primario*)

REPARTO (codiceReparto, nomeReparto)

AMBULATORIO (codiceAmbulatorio, codiceReparto, piano)

ESAME (codiceEsame, codiceAmbulatorio, codiceMedico, diagnosi, referto*)

ORARIOLAVORO (giorno, oraInizio, oraFine)

SPECIALIZZAZIONE (codiceSpecializzazione, codiceMedico, tipo, titolo, dataConseguimento, votoConseguimento)

MEDICO_ORARIOLAVORO (codiceMedico, giorno, oraInizio)

DDL (Data Definition Language)

Il **DDL** è l'insieme di comandi SQL (Structured Query Language) utilizzati per definire, creare e modificare la struttura di un Database. Grazie ad esso è possibile creare tabelle, record, attributi e relazioni tra entità specificandone le Primary Key e le Foreign Key per garantire l'integrità dei dati. Inoltre, permette di applicare vincoli Intra-Relazionali e Inter-Relazionali in modo da mantenere il Database coerente e affidabile.

Di seguito le porzioni di codice salienti con le relative spiegazioni.

Ruolo delle Chiavi Primarie

`Come detto in precedenza, le Chiavi Primarie (o Primary Key), serovno per mantenere l'integ`

Esempio:

```
CREATE TABLE PAGAMENTO (
    codicePagamento INT PRIMARY KEY,
    ...
)
```

In questo caso, i record all'interno della tabella "PAGAMENTO" saranno identificati dal "codicePagamento".

Ruolo delle Chiavi Esterne

`Al contrario delle chiavi primarie, le Chiavi Esterne (o Foreign Key) hanno la funzione di c`

Esempio:

```
CREATE TABLE PAGAMENTO (
    CREATE TABLE PAGAMENTO (
        ...
        codiceFiscale VARCHAR(16),
        FOREIGN KEY (codiceFiscale) REFERENCES PAZIENTE(codiceFiscale),
        ...
)
```

Nell'esempio riportato, l'attributo "codiceFiscale" è una chiave esterna che fa riferimento alla chiave primaria della tabella "PAZIENTE", consentendo di associare ogni record di "PAGAMENTO" a un paziente esistente e garantendo così il **vincolo di integrità referenziale**.

Gestione dei Vincoli nel Database

Un **vincolo** in un Databse è una regola applicata ai dati che ha lo scopo di garantire la correttezza, la coerenza e l'affidabilità delle informazioni contenute nella base di dati. Queste regole limitano i valori che possono essere inseriti o modificati all'interno delle tabelle, impedendo quelle operazioni che violerebbero le funzionalità o la struttura del Database.

Vi sono varie tipologie di vincoli, di seguito vengono elencati i più importanti.

Vincolo di Dominio

`Il vincolo di dominio permette di stabilire dei limiti per quanto riguarda un certo attribut`

Esempio:

```
...
CONSTRAINT check_ora CHECK(ora BETWEEN 0 AND 23)
...
```

Questa riga di codice limita il valore dell'attributo "ora" nell'intervallo tra 0 e 23.

Vincolo di Integrità Referenziale

`Il vincolo di integrità referenziale permette di verificare l'esistenza di una refernza di u`

Esempio:

```
...
    CONSTRAINT check_FKPaziente FOREIGN KEY(codiceFiscale)
        REFERENCES PAZIENTE(codiceFiscale)
        ON DELETE CASCADE
```

```
        ON UPDATE CASCADE
    ...
```

Il vincolo di integrità in questione stabilisce se vi è una vera relazione tra le FK dei record della tabella "PAGAMENTO" e le PK dei record nella tabella "PAZIENTI". Nel caso in cui il vincolo non sia rispettato, si può gestire l'errore in due metodi differenti: 1. Utilizzo de **CASCADE** -> propagazione automatica dell'eliminazione o modifica delle tabelle referenziate / referenzianti garantendone la coerenza e l'integrità 2. Utilizzo de **RESTRICT** -> impedimento dell'eliminazione o della modifica di un record nella tabella referenziante nel caso in cui esistano dei record correlati nelle tabelle referenziate, preservandone l'integrità

Dizionario dei Dati

FATTURA + codiceFattura INT - Identificativo univoco della fattura + codicePagamento INT - Pagamento a cui la fattura si riferisce

PAGAMENTO + codicePagamento INT - Identificatore univoco del pagamento + codiceFiscale VARCHAR(16) - Paziente che effettua il pagamento + data DATE - Data del pagamento + ora INT - Ora del pagamento + minuti INT - Minuti del pagamento + somma DOUBLE - Importo del pagamento + metodo VARCHAR(20) - Metodo di pagamento

PAZIENTE + codiceFiscale VARCHAR(16) - Identificativo univoco del paziente + nome VARCHAR(50) - Nome del paziente + cognome VARCHAR(50) - Cognome del paziente + dataNascita DATE - Data di nascita del paziente + anamnesi TEXT - Informazioni cliniche pregresse del paziente + ind_cap VARCHAR(10) - CAP dell'indirizzo di residenza + ind_citta VARCHAR(50) - Città di residenza + ind_via VARCHAR(50) - Via di residenza + ind_civico VARCHAR(10) - Numero civico

STORICO + codiceEsame INT - Esame svolto + data DATE - Data dell'esame + oraInizio INT - Ora di inizio + codiceFiscale VARCHAR(16) - Paziente coinvolto + oraFine INT - Ora di fine (opzionale) + diagnosi TEXT - Diagnosi medica + prescrizione TEXT - Prescrizione medica

PRENOTAZIONE + codicePrenotazione INT - Identificativo univoco della prenotazione + codiceFiscale VARCHAR(16) - Paziente che prenota + codiceMedico VARCHAR(10) - Medico assegnato + data DATE - Data della prenotazione + oraInizio INT - Ora di inizio + oraFine INT - Ora di fine (opzionale) + tipoVisita VARCHAR(50) - Tipologia di visita

MEDICO + codiceMedico VARCHAR(10) - Identificativo univoco del medico + codiceReparto INT - Reparto di appartenenza + codiceFiscale VARCHAR(16) - Codice fiscale del medico + nome VARCHAR(50) - Nome del medico + cognome VARCHAR(50) - Cognome del medico + primario BINARY - Indica se il

medico è primario

REPARTO + codiceReparto INT - Identificativo univoco del reparto + nomeReparto VARCHAR(50) - Nome del reparto

AMBULATORIO + codiceAmbulatorio INT - Identificativo univoco dell'ambulatorio + codiceReparto INT - Reparto di appartenenza + piano INT - Piano dell'edificio

ESAME + codiceEsame INT - Identificativo univoco dell'esame + codiceAmbulatorio INT - Ambulatorio in cui si svolge + codiceMedico VARCHAR(10) - Medico che effettua l'esame + diagnosi TEXT - Diagnosi dell'esame + referto TEXT - Referto medico (opzionale)

ORARIOLAVORO + giorno VARCHAR(2) - Giorno della settimana + oraInizio INT - Ora di inizio turno + oraFine INT - Ora di fine turno

SPECIALIZZAZIONE + codiceSpecializzazione VARCHAR(20) - Identificativo della specializzazione + codiceMedico VARCHAR(10) - Medico specializzato + tipo VARCHAR(30) - Tipo di specializzazione + titolo VARCHAR(50) - Titolo conseguito + dataConseguimento DATE - Data di conseguimento + votoConseguimento INT - Voto finale

MEDICO_ORARIOLAVORO + codiceMedico VARCHAR(10) - Medico assegnato al turno + giorno VARCHAR(2) - Giorno del turno + oraInizio INT - Ora di inizio turno

---

Conclusione

Ho sviluppato il progetto in maniera più dettagliata possibile, definendo l'**architettura di un Database** per la gestione di un centro polispecialistico. Ho cominicato con l'analisi dei requisiti, dalla quale ho sviluppato uno **schema concettuale** e di conseguenza anche quello **logico**. Infine ho creato il **DDL** del sistema con il relativo **dizionario dei dati** che fornisce le informazioni fondamentali per l'interpretazione della progettazione.

Le mie scelte progettuali sono mirate all'integrità e alla coerenza dei dati, favorendo inoltre il fattore di **scalabilità** per limplementazione futura. Infatti, il modello relazionale che ho proposto consente la gestione delle varie entità, facilitandone il controllo e i vincoli all'interno del Database.

Come **sviluppi successivi** si propongono l'effettivo applicativo web e l'implementazione di un sistema di query adatte ad un carico reale di dati.