|  |  |
| --- | --- |
|  |  |

Basi di Dati

Progetto A.A. 2023/2024

Customer Relationship Management

0308520

Valerio Baldazzi

**Indice**

[1. Descrizione del Minimondo 2](#_Toc527814491)

[2. Analisi dei Requisiti 3](#_Toc902236831)

[3. Progettazione concettuale 4](#_Toc55390959)

[4. Progettazione logica 5](#_Toc180212209)

[5. Progettazione fisica 7](#_Toc1804305735)

Tutto il testo su sfondo grigio, all’interno di questo template, deve essere eliminato prima della consegna. Viene utilizzato per fornire informazioni sulla corretta compilazione del report di progetto.

Non modificare il formato del documento:

- Carattere: Times New Roman, 12pt

- Dimensione pagina: A4

- Margini: superiore/inferiore 2,5cm, sinistro/destro: 1,9cm

# Descrizione del Minimondo

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22232425 | Inserire all’interno di questo riquadro la specifica così come è stata fornita. Riportare nella colonna a sinistra la numerazione delle righe. Questi numeri dovranno essere utilizzati per riferirsi al testo nelle sezioni successive.  Un sistema di Customer Relationship Management (o gestione delle relazioni con i clienti) è un sistema informativo che verte sulla fidelizzazione del cliente. Si vuole realizzare un sistema CRM per un’azienda marketing-oriented che intende realizzare relazioni durevoli di breve e lungo periodo con i propri clienti, massimizzando quindi il valore degli stessi. L’azienda contatta periodicamente i suoi clienti, di cui è a conoscenza di tutte le informazioni anagrafiche di interesse e a cui associa anche un certo numero di contatti telefonici e di email.  Inoltre, per ogni cliente, viene memorizzata anche la data di registrazione nel sistema. Gli operatori dell’azienda interagiscono periodicamente con i clienti per proporre nuove offerte commerciali, come sconti su prodotti, saldi o promozioni particolari, per fare alcuni esempi.Queste offerte sono di varia natura e quindi il sistema deve poter permettere alla segreteria di fornire una breve descrizione per ogni offerta.  Quando un operatore contatta un cliente mediante uno dei recapiti telefonici forniti, propone al cliente una delle offerte dell’azienda. Poiché lo stesso cliente può essere contattato da più utenti, il sistema deve prevedere un meccanismo per registrare delle note in cui gli operatori annotano dettagli sul risultato dell'interazione. Gli utenti possono richiamare l'elenco delle note di un cliente che stanno contattando.  Qualora un cliente accettasse l'offerta, questa deve essere registrata nel sistema, indicando quale  utente ha permesso l'accettazione e in quale data.  Un cliente potrebbe decidere di richiedere un appuntamento in una delle sedi dell'azienda. In questo caso, alla nota deve essere allegato un appuntamento, indicando in quale sede e in quale data/ora lo stesso si svolgerà.  La segreteria può, in ogni momento, generare un report che mostri, in un intervallo temporale specificato, per tutti i clienti del sistema, quanti sono stati contattati e quante volte, così come quante offerte sono state accettate da ciascuno. |

# Analisi dei Requisiti

Lo scopo di questa sezione è raffinare la specifica fornita, andando ad effettuare un’operazione preliminare di disambiguazione.

## Identificazione dei termini ambigui e correzioni possibili

Compilare la seguente tabella, facendo riferimento alla specifica del minimondo di riferimento precedentemente indicata. Individuare i termini ambigui nella specifica (indicando la linea in cui essi compaiono), indicare il nuovo termine che si intende adottare nella specifica, ed indicare il motivo del cambiamento che si propone.

|  |  |  |  |
| --- | --- | --- | --- |
| **Linea** | **Termine** | **Nuovo termine** | **Motivo correzione** |
| 15,16, 19 | Utente | Operatore | Gli utenti nel sistema che hanno il compito di contattare i clienti sono gli operatori. |
| 5,9 | Periodicamente | Non più di 30 chiamate l’ora | Ogni operatore può effettuare ragionevolmente al più 30 chiamate l’ora. |
| 6 | Informazioni anagrafiche di interesse | Nome, cognome, data di nascita, codice fiscale, indirizzo di residenza | Specifico le informazioni anagrafiche dei clienti di cui l’azienda ha bisogno. |
| 6 | Certo numero | Numero arbitrario | L’azienda non pone un limite al numero di contatti telefonici ed email forniti dai clienti. |
| 13 | Recapiti | Contatti | Il termine Recapiti può essere confuso con gli indirizzi fisici. |
| 24 | Per tutti i clienti del sistema, quanti sono stati contattati e quante volte | Numero totale di clienti contattati e per ogni cliente quante volte è stato contattato | Specifico in che modo la segreteria deve generare il report sui clienti contattati. |

### Specifica disambiguata

|  |
| --- |
| Riportare in questo riquadro la specifica di progetto corretta, applicando le disambiguazioni proposte.  Un sistema di Customer Relationship Management (o gestione delle relazioni con i clienti) è un sistema informativo che verte sulla fidelizzazione del cliente. Si vuole realizzare un sistema CRM per un’azienda marketing-oriented che intende realizzare relazioni durevoli di breve e lungo periodo con i propri clienti, massimizzando quindi il valore degli stessi.  L’azienda è a conoscenza del nome, cognome, data di nascita, codice fiscale, indirizzo di residenza dei propri clienti a cui associa anche un numero arbitrario di contatti telefonici e di email.  Inoltre, per ogni cliente, viene memorizzata anche la data di registrazione nel sistema. Ogni operatore dell’azienda interagisce con i clienti non più di 30 volte l’ora, per proporre nuove offerte commerciali, come sconti su prodotti, saldi o promozioni particolari, per fare alcuni esempi. Queste offerte sono di varia natura e quindi il sistema deve poter permettere alla segreteria di fornire una breve descrizione per ogni offerta.  Quando un operatore contatta un cliente mediante uno dei contatti telefonici forniti, propone al cliente una delle offerte dell’azienda. Poiché lo stesso cliente può essere contattato da più operatori, il sistema deve prevedere un meccanismo per registrare delle note in cui gli operatori annotano dettagli sul risultato dell'interazione. Gli operatori possono richiamare l'elenco delle note di un cliente che stanno contattando.  Qualora un cliente accettasse l'offerta, questa deve essere registrata nel sistema, indicando quale operatore ha permesso l'accettazione e in quale data.  Un cliente potrebbe decidere di richiedere un appuntamento in una delle sedi dell'azienda. In questo caso, alla nota deve essere allegato un appuntamento, indicando in quale sede e in quale data/ora lo stesso si svolgerà.  La segreteria può, in ogni momento, generare un report che mostri, in un intervallo temporale specificato, il numero totale di clienti contattati e per ogni cliente quante volte è stato contattato, così come quante offerte sono state accettate da ciascuno. |

## Glossario dei Termini

Realizzare un dizionario dei termini, compilando la tabella qui sotto, a partire dalle specifiche precedentemente disambiguate

|  |  |  |  |
| --- | --- | --- | --- |
| **Termine** | **Descrizione** | **Sinonimi** | **Collegamenti** |
| Operatore | Lavoratore che esegue compiti specifici, spesso tecnici o manuali, all'interno di un'azienda o di un'organizzazione. |  | Nota, Offerta Accettata |
| Cliente | Persona o azienda che acquista beni o servizi da un'altra persona o azienda. |  | Nota, Appuntamento, Offerta Accettata |
| Offerta | Proposta commerciale presentata da un venditore/operatore commerciale a un potenziale acquirente che specifica i termini e le condizioni di vendita di beni o servizi. |  | Nota, Offerta Accettata |
| Offerta Accettata | Situazione in cui una proposta di vendita, acquisto, lavoro o altro tipo di offerta è stata ufficialmente accettata dalla parte a cui era destinata, indicando il consenso a procedere secondo i termini e le condizioni proposti. |  | Offerta, Cliente, Operatore |
| Appuntamento | Incontro programmato tra due o più persone, fissato per una data,un luogo e un orario specifici. |  | Cliente, Nota |
| Nota | Breve annotazione scritta utilizzata per ricordare informazioni importanti. |  | Offerta, Cliente, Operatore, Appuntamento |

## Raggruppamento dei requisiti in insiemi omogenei

Per ciascun elemento “più importante” della specifica (riportata anche nel glossario precedente), estrapolare dalla specifica disambiguata le frasi ad esso associate. Compilare una tabella separata per ciascun elemento individuato.

|  |
| --- |
| **Frasi di carattere generale** |
| Un sistema di Customer Relationship Management (o gestione delle relazioni con i clienti) è un sistema informativo che verte sulla fidelizzazione del cliente. Si vuole realizzare un sistema CRM per un’azienda marketing-oriented che intende realizzare relazioni durevoli di breve e lungo periodo con i propri clienti, massimizzando quindi il valore degli stessi. |

|  |
| --- |
| **Frasi relative agli operatori** |
| Ogni operatore dell’azienda interagisce con i clienti non più di 30 volte l’ora, per proporre nuove offerte commerciali[…]. Quando un operatore contatta un cliente mediante uno dei contatti telefonici forniti, propone al cliente una delle offerte dell’azienda. Poiché lo stesso cliente può essere contattato da più operatori, il sistema deve prevedere un meccanismo per registrare delle note in cui gli operatori annotano dettagli sul risultato dell'interazione. Gli operatori possono richiamare l'elenco delle note di un cliente che stanno contattando. |

|  |
| --- |
| **Frasi relative ai clienti** |
| Qualora un cliente accettasse l'offerta, questa deve essere registrata nel sistema, indicando quale operatore ha permesso l'accettazione e in quale data. Un cliente potrebbe decidere di richiedere un appuntamento in una delle sedi dell'azienda. La segreteria può, in ogni momento, generare un report che mostri, in un intervallo temporale specificato, il numero totale di clienti contattati e per ogni cliente quante volte è stato contattato, così come quante offerte sono state accettate da ciascuno. |

|  |
| --- |
| **Frasi relative alle note** |
| Il sistema deve prevedere un meccanismo per registrare delle note in cui gli operatori annotano dettagli sul risultato dell'interazione. Un cliente potrebbe decidere di richiedere un appuntamento in una delle sedi dell'azienda. In questo caso, alla nota deve essere allegato un appuntamento, indicando in quale sede e in quale data/ora lo stesso si svolgerà. |

|  |
| --- |
| **Frasi relative alle offerte** |
| Ogni operatore dell’azienda[…], per proporre nuove offerte commerciali[…].Queste offerte sono di varia natura e quindi il sistema deve poter permettere alla segreteria di fornire una breve descrizione per ogni offerta. |

|  |
| --- |
| **Frasi relative alle offerte accettate** |
| Qualora un cliente accettasse l'offerta, questa deve essere registrata nel sistema, indicando quale operatore ha permesso l'accettazione e in quale data. |

|  |
| --- |
| **Frasi relative agli appuntamenti** |
| Un cliente potrebbe decidere di richiedere un appuntamento in una delle sedi dell'azienda. In questo caso, alla nota deve essere allegato un appuntamento, indicando in quale sede e in quale data/ora lo stesso si svolgerà. |

# Progettazione concettuale

## Costruzione dello schema E-R

In questa sezione è necessario riportare tutti passi seguiti per la costruzione dello schema E-R finale, a partire dalle specifiche raccolte ed organizzate nel capitolo precedente. Non è richiesto un procedimento specifico: si può adottare una strategia top-down, bottom-up, a macchia d’olio o mista. L’importante è descrivere e commentare tutti i passi della costruzione, andando anche ad inserire “schemi parziali” utilizzati nel processo.

Nello sviluppo dello schema concettuale, è stata adottata la strategia *mista* , la quale permette di combinare i vantaggi della strategia top-down con quelli della stategia bottom-up. Questa strategia consiste nel suddividere i requisiti in componenti separate ma allo stesso tempo bisogna definire uno schema scheletro, contenente i concetti principali dell’applicazione. A seguito di un’attenta analisi dei requisiti dei paragrafi precedenti, si evince che i concetti fondamentali, rappresentati come entità, su cui definire uno schema scheletro sono: Cliente, Offerta, Nota.

Immagine che contiene testo, software, Icona del computer, Software multimediale

Descrizione generata automaticamente

L’entità cliente viene identificata univocamente dal proprio Codice Fiscale, a cui vengono aggiunti gli attributi:Nome, Cognome, Data Di Nascita, Data Di Registrazione. Il cliente ha anche un indirizzo di residenza, modellato da un attributo composto, caratterizzato da: Via, N.Civico, Cap, Città, Provincia e Paese. Al cliente vengono associati un numero arbitrario di contatti telefonici ed email, quindi vengono assegnati 2 attributi multivalore: Telefono ed Email.

Immagine che contiene testo, software, Icona del computer, Software multimediale

Descrizione generata automaticamente

Un cliente può accettare delle offerte e quando questo avviene devono essere salvate nel sistema, quindi viene inserita l’entità Offerta Accettata associata a Cliente. Quando un cliente accetta un’offerta viene memorizzata la data e l’operatore che ha permesso l’accettazione. Quindi viene inserito l’attributo Data Di Accettazione relativo all’entità Offerta Accettata ed aggiunta l’entità Operatore, identificata univocamente da un Codice Operatore, a cui sono aggiunti gli attributi: Nome e Cognome.

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

Ora si passa allo sviluppo della porzione dello scheletro relativa all’entità Nota. Ogni nota viene scritta da un operatore ed associata a un cliente per descrivere l’esito di una proposta di una determinata offerta. Per ogni nota è utile essere a conoscenza della data in cui è stata scritta/modificata , quindi vengono inseriti gli attributi: Esito e Data Di Modifica. L’entità Nota viene associata all’entità Operatore precedentemente introdotta.

Immagine che contiene testo, software, Icona del computer, Software multimediale

Descrizione generata automaticamente

Ad ogni nota può essere allegato un appuntamento, qualora il cliente l’avesse richiesto. Quindi viene introdotta l’entità Appuntamento con i seguenti attributi: Sede, Data, Orario e associata all’entità Nota. L’appuntamento viene identificato univocamente dalla Nota a cui viene allegato.

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

L’ultima porzione dello scheletro da sviluppare è quella relativa all’entità Offerta. L’entità Offerta viene identificata univocamente da un Codice Offerta, a cui vengono aggiunti gli attributi: Nome Offerta , Descrizione e Data Di Scadenza. L’attributo Descrizione è inserito per permettere alla segreteria di descrivere i dettagli dell’offerta stessa. L’attributo Data Di Scadenza è utile per mantenere il sistema aggiornato , garantendo che le offerte obsolete non rimangano attive e non causino confusione.

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

Come analizzato precedentemente, un’offerta qualora venisse accettata da un cliente, deve essere registrata nel sistema, quindi viene legata l’entità Offerta con l’entità Offerta Accettata, introdotta precedentemente, attraverso l’associazione Accettata.

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

### Integrazione finale

Nell’integrazione finale delle varie parti dello schema E-R è possibile che si evidenzino dei conflitti sui nomi utilizzati e dei conflitti struttuali. Prima di riportare lo schema E-R finale, descrivere quali passi sono stati adottati per risolvere tali conflitti.

Tutte le porzioni dello scheletro sono state sviluppate. Ora vengono unite in un unico schema concettuale , ponendo attenzione nella scelta degli identificatori delle entità che ancora non sono state definite completamente. Per l’entità Nota vengono usati 2 identificatori esterni: il codice fiscale del cliente e il codice dell’offerta che gli viene proposta dall’operatore. Questa scelta risulta la migliore per identificare univocamente una nota, perchè ad un cliente di norma non viene proposta la stessa offerta più volte, da operatori diversi. Per l’entità Offerta Accettata vengono utilizzati gli stessi identificatori esterni dell’entità Nota, quindi il codice fiscale del cliente che accetta l’offerta e il codice dell’offerta che viene accettata. Anche in questo caso l’operatore non è stato incluso tra gli identificatori, perchè risulta ovvio che una stessa offerta non può essere accettata dallo stesso cliente più volte , per mezzo di operatori differenti. L’entità Appuntamento precedentemente introdotta utilizza come identificatore esterno gli identificatori della nota, quindi il codice fiscale del cliente che l’ha richiesto e il codice dell’offerta per cui sta richiedendo un appuntamento. Ora sono state definite completamente tutte le entità , quindi questo è lo schema finale proposto , dove tutte le porzioni dello scheletro generale sono sviluppate e coese.

Immagine che contiene testo, schermata, software, diagramma

Descrizione generata automaticamente

## Regole aziendali

Laddove la specifica non sia catturata in maniera completa dallo schema E-R, corredare lo stesso in questo paragrafo con l’insieme delle regole aziendali necessarie a completare la progettazione concettuale.

L’orario di appuntamento di un cliente che ha più appuntamenti nella stessa data deve essere distanziato di almeno due ore tra un appuntamento e l'altro.

La data di accettazione di un’offerta deve essere antecedente alla data di scadenza dell’offerta stessa.

La data di modifica di una nota relativa ad un’offerta deve essere antecedente alla data di scadenza dell’offerta stessa. (Non vengono proposte offerte scadute ai clienti)

## Dizionario dei dati

Completare la progettazione concettuale riportando nella tabella seguente il dizionario dei dati

|  |  |  |  |
| --- | --- | --- | --- |
| **Entità** | **Descrizione** | **Attributi** | **Identificatori** |
| Cliente | Le informazioni di un cliente che viene contattato | Codice Fiscale, Nome, Cognome, Indirizzo Di Residenza[Via, Numero Civico, Cap, Città, Provincia, Paese], Telefono, Email, Data Di Registrazione | Codice Fiscale |
| Nota | I dettagli dell’interazione tra operatore e cliente in merito a un’offerta proposta. | Esito, Data Di Modifica | Cliente, Offerta |
| Offerta | Le informazioni delle offerte presenti nel sistema | Codice Offerta, Nome Offerta, Descrizione, Data Di Scadenza | Codice Offerta |
| Appuntamento | I dettagli di una prenotazione di un appuntamento effettuata da un cliente | Sede, Data, Orario | Cliente, Sede, Data |
| Operatore | Le informazioni degli operatori nell’azienda | Codice Operatore, Nome, Cognome | Codice Operatore |
| Offerta Accettata | I dettagli delle offerte accettate dai clienti | Data Di Accettazione | Offerta, Cliente |

# Progettazione logica

## Volume dei dati

Questa sezione serve ad illustrare qual è il carico che la base di dati dovrà sopportare. A tal fine, è necessario prevedere un volume di dati attesi. Compilare la tabella sottostante, per ciascun concetto identificato nello schema E-R. I volumi devono essere stimati dallo studente in maniera ragionevole rispetto all’operatività presunta dell’applicativo.

Nell’analisi si è assunto che l’azienda CRM sia di grandi dimensioni, con un numero molto elevato di clienti . E’ possibile che variazioni della dimensione dell’azienda possano influenzare l’analisi dei costi. Si assume che le note associate ai clienti e l’accettazione delle offerte da parte di essi siano distribute uniformemente durante l’anno.

|  |  |  |
| --- | --- | --- |
| **Concetto nello schema** | **Tipo[[1]](#footnote-2)** | **Volume atteso** |
| Cliente | E | 160.000 |
| Offerta | E | 480.000 |
| Nota | E | 100.000.000 |
| Offerta Accettata | E | 20.000.000 |
| Operatore | E | 1000 |
| Appuntamento | E | 6.000.000 |
| Riferita | R | 100.000.000 |
| Relativa | R | 100.000.000 |
| Allegato | R | 6.000.000 |
| Prenotato | R | 6.000.000 |
| Firmata | R | 20.000.000 |
| Accettata | R | 20.000.000 |
| Validata | R | 20.000.000 |
| Scritta | R | 100.000.000 |

Nell’analisi si assume che il sistema mantenga i dati relativi alle note e alle offerte accettate dai clienti per un periodo non superiore ai 5 anni. I dati degli appuntamenti persistono nella base di dati per l’anno in cui sono stati prenotati. Si ipotizza che l’azienda disponga di 1000 operatori e che mediamente ogni operatore può effettuare 20 chiamate/ora. Ogni giornata lavorativa si ipotizza composta da 8 ore e che i giorni lavorativi annuali siano 250. Di conseguenza un operatore mediamente riesce ad effettuare 20 x 8 x 250= 40.000 chiamate/anno. Moltiplicato per il numero di operatori si hanno 40.000 x 1000= 40.000.000 chiamate/anno totali. Da quanto richiesto dall’azienda , ogni cliente nel caso pessimo, deve essere contattato almeno 1 volta al giorno, di conseguenza sapendo che mediamente ogni operatore effettua 20 chiamate, allora con 1000 operatori si hanno 8 x 20 x 1000= 160.000 chiamate/giorno , che corrisponde al numero di clienti che l’azienda riesce a coprire secondo questa specifica. L’azienda per permettere una grande varietà di scelta ai propri clienti, si suppone che metta a disposizione un numero di offerte pari a 3 volte il numero di clienti 3 x 160.000= 480.000 offerte. Si stima che i clienti rispondano alle chiamate effettuate dagli operatori con un tasso del 50% , di conseguenza si hanno 40.000.000 x 0.5= 20.000.000 interazioni/anno. Sapendo che dopo ogni interazione l’operatore deve scrivere una nota, allora si hanno mediamente 20.000.000 di note all’anno. Ipotizzando che l’azienda effettui un buon marketing delle offerte mirato ai propri clienti, si suppone che il 30% di essi richieda un appuntamento e il 20% accetti l’offerta proposta dall’operatore. Quindi si hanno mediamente 20.000.000 x 0.3 = 6.000.000 appuntamenti/anno e 20.000.000 x 0.2= 4.000.000 offerte accettate/anno.

## Tavola delle operazioni

Rappresentare nella tabella sottostante tutte le operazioni *non banali* sulla base di dati che devono essere supportate dall’applicazione, con la frequenza attesa. Le operazioni da supportare devono essere desunte dalle specifiche raccolte.

|  |  |  |
| --- | --- | --- |
| **Cod.** | **Descrizione** | **Frequenza attesa** |
| LN | Lista note | 160.000/giorno |
| SN | Scrivi nota | 80.000/giorno |
| IO | Inserisci offerta | 1920/giorno |
| IOA | Inserisci offerta accettata | 384/giorno |
| RCC | Report clienti contattati | 200/mese |
| RC | Registra Cliente | 640/giorno |
| L1 | Login | 1200/giorno |

Si assume che la segreteria dell’azienda sia composta da 200 persone e che ognuna di esse effettui l’operazione ‘RCC’ 1 volta al mese.

L’operazione ‘RC’ si stima che venga svolta mediamente 160.000/ 250= 640 volte al giorno.

L’operazione ‘LN’ si stima che venga svolta mediamente 20 x 8 x 1000= 160.000 volte al giorno.

L’operazione ‘SN’ si stima che venga svolta mediamente 160.000 x 0.5= 80.000 volte al giorno.

L’operazione ‘IO’ si stima che venga svolta mediamente 480.000/250= 1920 volte al giorno.

L’operazione ‘IOO’ si stima che venga svolta mediamente 1920 x 0.2= 384/giorno dato dal tasso di accettazione delle offerte dei clienti dell’azienda.

## Costo delle operazioni

In riferimento a tutte le operazioni precedentemente indicate, calcolarne il costo supponendo, per questa fase del progetto, che il costo in scrittura di un dato sia doppio rispetto a quello in lettura.

Si suppone che un accesso in scrittura abbia un costo doppio rispetto a un accesso in lettura.

**LN – Lista note**

* 625 di accessi in lettura all’entità *Nota*
* 625 di accessi in lettura alla relazione *Relativa*
* 625 di accessi in lettura all’entità *Offerta*
* 625 di accessi in lettura alla relazione *Scritta*
* 625 di accessi in lettura all’entità *Operatore*
* 38 di accessi in lettura alla relazione *Allegato*
* 38 di accessi in lettura all’entità *Appuntamento*

Costo totale: 3201

Accessi/giorno: 3201 x 160.000= 512.160.000

**SN – Scrivi nota**

* 1 accesso in scrittura all’entità *Nota*
* 1 accesso in scrittura alla relazione *Riferita*
* 1 accesso in scrittura alla relazione *Relativa*
* 1 accesso in scrittura alla relazione *Scritta*
* 1 accesso in scrittura all’entità *Appuntamento* (30% dei casi)
* 1 accesso in scrittura alla relazione *Allegato* (30% dei casi)

Costo totale: 6 x 2= 12

Accessi/giorno: 80.000 x 12= 960.000

**IO – Inserisci offerta**

* 1 accesso in scrittura all’entità *Offerta*

Costo totale: 1 x 2= 2

Accessi/giorno: 1920 x 2= 3840

**IOA – Inserisci offerta accettata**

* 1 accesso in scrittura all’entità *Offerta Accettata*
* 1 accesso in scrittura alla relazione *Accettata*
* 1 accesso in scrittura alla relazione *Firmata*
* 1 accesso in scrittura alla relazione *Validata*

Costo totale: 4 x 2= 8

Accessi/giorno: 384 x 8= 3072

Assumendo che mediamente l’intervallo temporale scelto non sia superiore ad 1 mese

**RCC – Report clienti contattati**

* 2.400.000 di accessi in lettura all’entità *Nota*
* 2.400.000 di accessi in lettura alla relazione *Riferita*
* 160.000 di accessi in lettura all’entità *Cliente*
* 480.000 di accessi in lettura alla relazione *Firmata*
* 480.000 di accessi in lettura all’entità *Offerta Accettata*

Costo totale: 5.920.000

Accessi/mese: 5.920.000 x 200 = 1.184.000.000

**RC– Registra cliente**

* 1 accesso in scrittura all’entità *Cliente*

Costo totale: 1 x 2= 2

Accessi/giorno: 2 x 640 = 1280

Dall’analisi si evince che le operazioni ‘LN’ e ‘RCC’ sono il “collo di bottiglia” del sistema. Rimuovendo l’entità *Operatore* e aggiungendo alle entità associate precedentemente ad essa l’attributo “Codice Operatore”, ignorando “Nome Operatore” e “Cognome Operatore” superflui in questo caso, è possibile ottenere un buon risparmio in termini prestazionali(ovviamente questa scelta va contro la pratica di una buona progettazione concettuale, ma avvicinandosi verso la costruzione del database risulta utile) . Riguardo l’operazione ‘RCC’ non è possibile in questo punto della progettazione fare nulla; l’ipotesi dell’inserimento di un dato ridondante come ad esempio ‘NumeroOffertaAccettate’ all’entità Cliente in questo caso non sarebbe valido, poichè il report è relativo ad un intervallo temporale specificato dalla segreteria . Non è stata presa in considerazione l’opzione di accorpare le entità *Appuntamento* e *Nota*, poichè questa scelta avrebbe portato alla presenza di tanti valori NULL (~70% delle note)all’interno della base di dati.

**LN – Lista note**

Accessi/giorno: 1951 x 160.000 = 312.160.000

Risparmio: 200.000.000

Con le scelte prese, le operazioni ‘SN’ e ‘IOA’ risparmiano un accesso in scrittura( alla relazione *Scritta* per ‘SN’ e alla relazione *Validata* per ‘IOA’).

L'operazione ‘L1’ non è soggetta a una stima del costo perchè l'operazione di login è parte integrante del sistema e necessaria per l'accesso degli utenti al sistema stesso. Dato che il suo costo è principalmente associato alla progettazione fisica e all'implementazione del sistema, non è possibile effettuare una stima del costo in questa fase di analisi.

## Ristrutturazione dello schema E-R

Descrivere (laddove necessario fornendo anche degli schemi) quali passi vengono adottati per ristrutturare lo schema E-R, ad esempio in termini di:

Analisi delle ridondanze

Eliminazione delle generalizzazioni

Scelta degli identificatori primari

Si noti che in questa fase è possibile fare riferimento al costo delle operazioni precedentemente realizzato per guidare le scelte. Ad esempio, un leggero spreco di memoria legato alla non rimozione di ridondanze può essere facilmente giustificato da un guadagno in termini di prestazioni.

L’entità *Cliente* subisce una ristrutturazione perchè possiede 2 attributi multivalore che non sono rappresentabili nel modello relazionale : Telefono ed Email. I 2 attributi vengono trasformati in entità autonome, che vengono associate all’entità di appartenenza, ovvero *Cliente*. L’attributo composto ‘Indirizzo Di Residenza’ viene scomposto negli attributi ‘Indirizzo’ (che conterrà : via , numero civico e cap), ‘Città’, ‘Provincia’, ‘Paese’ . In questo modo la registrazione di un nuovo cliente risulta più dispendiosa , ma dato il costo iniziale notevolmente piccolo dell’operazione di registrazione di un cliente, è possibile trascurare questo incremento.

Immagine che contiene testo, schermata, software, diagramma

Descrizione generata automaticamente

In base alle analisi sui dati effettuate precedentemente, è utile rimuovere l’entità *Operatore* e aggiungere l’attributo ‘Cod. Operatore’ alle entità associate ad essa, per aumentare le prestazioni del nostro sistema.

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

In base a quanto discusso, si ottiene il seguente schema ristrutturato.

Immagine che contiene testo, schermata, software, diagramma

Descrizione generata automaticamente

## Trasformazione di attributi e identificatori

Qualora siano presenti, in questa fase della progettazione, attributi ripetuti o identificatori esterni, descrivere quali trasformazioni vengono realizzate sul modello per facilitare la traduzione nello schema relazionale.

L’attributo ‘CodiceOperatore’ è ripetuto nelle entità *Nota* e *Offerta Accettata*, ma in funzione della stima dei costi svolta precedentemente e delle scelte prese, è necessaria questa ripetizione. Gli identificatori esterni introdotti nel modello concettuale sono necessari.

## Traduzione di entità e associazioni

Riportare in questa sezione la traduzione di entità ed associazioni nello schema relazionale.

Fornire una rappresentazione grafica del modello relazionale completo.

Alcuni attributi sono stati ridenominati per migliorare la comprensione della base di dati. Il modello relazionale è il seguente:

Cliente(CodiceFiscale, Nome, Cognome, DataDiNascita, DataDiRegistrazione, Indirizzo, Città, Provincia, Paese)

Email(NomeEmail, CodiceFiscaleCliente)

Telefono(NumeroDiTelefono, CodiceFiscaleCliente)

Nota(CodiceFiscaleCliente, CodiceOfferta, Esito, Operatore, DataDiModifica)

Offerta(CodiceOfferta, NomeOfferta, Descrizione, DataDiScadenza)

OffertaAccettata(CodiceOfferta, CodiceFiscaleCliente, DataDiAccettazione, Operatore)

Appuntamento(CodiceFiscaleCliente, CodiceOfferta, Sede, Data, Orario)

Con i vincoli:

Email(CodiceFiscaleCliente) ⊆ Cliente(CodiceFiscale)

Telefono(CodiceFiscaleCliente) ⊆ Cliente(CodiceFiscale)

Nota(CodiceFiscaleCliente) ⊆ Cliente(CodiceFiscale)

Nota(CodiceOfferta) ⊆ Offerta(CodiceOfferta)

OffertaAccettata(CodiceOfferta) ⊆ Offerta(CodiceOfferta)

OffertaAccettata(CodiceFiscaleCliente) ⊆ Cliente(CodiceFiscale)

Appuntamento(CodiceFiscaleCliente, CodiceOfferta) ⊆ Nota(CodiceFiscaleCliente, CodiceOfferta)

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

## Normalizzazione del modello relazionale

Mostrare se il modello relazionale precedentemente descritto è in forma normale. Se non lo è, descrivere quali trasformazioni si effettuano per normalizzarlo. Se si sceglie di non normalizzarlo, giustificare le motivazioni da un punto di vista prestazionale.

Il modello relazionale precedentemente proposto non risulta essere normalizzato. Sono presenti le dipendenze funzionali in *Cliente*: Citta 🡪 Provincia, Paese e Provincia🡪Paese. Quindi la relazione soffre di anomalie di: aggiornamento, inserimento e cancellazione. Dato che è un evento raro che una città cambi provincia e rarissimo che una città e/o provincia cambino paese, la relazione non necessita di essere modificata . In questo modo viene resa anche più efficiente la registrazione di un nuovo cliente.

# Progettazione fisica

## Utenti e privilegi

Descrivere, all’interno dell’applicazione, quali utenti sono stati previsti con quali privilegi di accesso su quali tabelle, giustificando le scelte progettuali.

Si prevedono tre ruoli, per implementare il Principle of Least Privilege:

* Login
  + Grant in esecuzione sull’operazione ‘L1’.
* Operatore
  + Grant in esecuzione sulle operazioni ‘LN’, ‘SN’, ’IOA’.
* Segreteria
  + Grant in esecuzione sulle operazioni ‘IO’, ‘RCC’ e ‘RC’.

Per identificare gli utenti si introduce una tabella *Utenti* per mantenere le credenziali.

Immagine che contiene testo, schermata, software, Icona del computer

Descrizione generata automaticamente

Per la creazione dei ruoli è stata adottata la tecnica RBAC:

In questo modo vengono disaccoppiati i privilegi dagli utenti effettivi. Infatti i permessi sono assegnati ai ruoli e gli utenti per acquisire i relativi privilegi devono attivare il ruolo corrispondente.

* Creazione degli utenti:

CREATE USER 'login'@'localhost' IDENTIFIED BY 'login';

CREATE USER 'operatore'@'localhost' IDENTIFIED BY 'operatore';

CREATE USER 'segreteria'@'localhost' IDENTIFIED BY 'segreteria';

* Creazione dei ruoli:

CREATE ROLE loginRole;

CREATE ROLE operatoreRole;

CREATE ROLE segreteriaRole;

* Assegnazione dei ruoli:

GRANT loginRole TO 'login'@'localhost';

GRANT operatoreRole TO 'operatore'@'localhost';

GRANT segreteriaRole TO 'segreteria'@'localhost';

* Impostazione dei ruoli predefiniti:

SET DEFAULT ROLE loginRole FOR 'login'@'localhost';

SET DEFAULT ROLE operatoreRole FOR 'operatore'@'localhost';

SET DEFAULT ROLE segreteriaRole FOR 'segreteria'@'localhost';

* Assegnazioni dei privilegi per ogni ruolo:

GRANT EXECUTE ON PROCEDURE crm.login TO 'loginRole';

GRANT SELECT ON crm.utenti TO 'loginRole';

GRANT EXECUTE ON PROCEDURE crm.inserisci\_offerta TO 'segreteriaRole';

GRANT EXECUTE ON PROCEDURE crm.elimina\_offerte TO 'segreteriaRole';

GRANT EXECUTE ON PROCEDURE crm.mostra\_offerte\_scadute TO 'segreteriaRole';

GRANT EXECUTE ON PROCEDURE crm.registra\_cliente TO 'segreteriaRole';

GRANT EXECUTE ON PROCEDURE crm.report\_clienti TO 'segreteriaRole';

GRANT EXECUTE ON PROCEDURE crm.inserisci\_offerta\_accettata TO 'operatoreRole';

GRANT EXECUTE ON PROCEDURE crm.lista\_note TO 'operatoreRole';

GRANT EXECUTE ON PROCEDURE crm.mostra\_clienti TO 'operatoreRole';

GRANT EXECUTE ON PROCEDURE crm.mostra\_offerte TO 'operatoreRole';

GRANT EXECUTE ON PROCEDURE crm.scrivi\_nota TO 'operatoreRole';

GRANT EXECUTE ON PROCEDURE crm.mostra\_email TO 'operatoreRole';

GRANT EXECUTE ON PROCEDURE crm.mostra\_telefoni TO 'operatoreRole';

## Strutture di memorizzazione

Compilare la tabella seguente indicando quali tipi di dato vengono utilizzati per memorizzare le informazioni di interesse nelle tabelle, per ciascuna tabella.

|  |  |  |
| --- | --- | --- |
| **Tabella <Utenti>** | | |
| **Colonna** | **Tipo di dato** | **Attributi[[2]](#footnote-3)** |
| Username | VARCHAR(30) | PK,NN |
| Password | VARCHAR(32) | NN |
| Ruolo | ENUM('operatore', 'segreteria') | NN |

|  |  |  |
| --- | --- | --- |
| **Tabella <Cliente>** | | |
| **Colonna** | **Tipo di dato** | **Attributi[[3]](#footnote-4)** |
| CodiceFiscale | CHAR(16) | PK,NN |
| Nome | VARCHAR(30) | NN |
| Cognome | VARCHAR(30) | NN |
| Indirizzo | VARCHAR(50) | NN |
| Città | VARCHAR(30) | NN |
| Provincia | CHAR(2) | NN |
| Paese | VARCHAR(30) | NN |
| DataDiNascita | DATE | NN |
| DataDiRegistrazione | DATE | NN |

|  |  |  |
| --- | --- | --- |
| **Tabella <Email>** | | |
| **Colonna** | **Tipo di dato** | **Attributi[[4]](#footnote-5)** |
| NomeEmail | VARCHAR(255) | PK |
| CodiceFiscaleCliente | CHAR(16) | NN |

|  |  |  |
| --- | --- | --- |
| **Tabella <Telefono>** | | |
| **Colonna** | **Tipo di dato** | **Attributi[[5]](#footnote-6)** |
| NumeroDiTelefono | VARCHAR(30) | PK |
| CodiceFiscaleCliente | CHAR(16) | NN |

|  |  |  |
| --- | --- | --- |
| **Tabella <Nota>** | | |
| **Colonna** | **Tipo di dato** | **Attributi[[6]](#footnote-7)** |
| CodiceFiscaleCliente | CHAR(16) | PK,NN |
| CodiceOfferta | CHAR(6) | PK,NN |
| Esito | TEXT | NN |
| Operatore | VARCHAR(30) | NN |
| DataDiModifica | DATE | NN |

|  |  |  |
| --- | --- | --- |
| **Tabella <Offerta>** | | |
| **Colonna** | **Tipo di dato** | **Attributi[[7]](#footnote-8)** |
| CodiceOfferta | CHAR(6) | PK,NN |
| NomeOfferta | VARCHAR(50) | NN |
| Descrizione | TEXT | NN |
| DataDiScadenza | DATE | NN |

|  |  |  |
| --- | --- | --- |
| **Tabella <OffertaAccettata>** | | |
| **Colonna** | **Tipo di dato** | **Attributi[[8]](#footnote-9)** |
| CodiceOfferta | CHAR(6) | PK,NN |
| CodiceFiscaleCliente | CHAR(16) | PK,NN |
| DataDiAccettazione | DATE | NN |
| Operatore | VARCHAR(30) | NN |

|  |  |  |
| --- | --- | --- |
| **Tabella <Appuntamento>** | | |
| **Colonna** | **Tipo di dato** | **Attributi[[9]](#footnote-10)** |
| CodiceFiscaleCliente | CHAR(16) | PK,NN |
| CodiceOfferta | CHAR(6) | PK,NN |
| Sede | VARCHAR(30) | NN |
| Data | DATE | NN |
| Orario | TIME | NN |

## Indici

Compilare la seguente tabella, per ciascuna tabella del database in cui sono presenti degli indici. Descrivere le motivazioni che hanno portato alla creazione di un indice, facendo riferimento al costo delle operazioni individuate nella sezione precedente.

Attenzione: non è necessario riportare gli indici autogenerati in fase di definizione dello schema (ad esempio, per la gestione della chiave primaria), ma *soltanto* quelli introdotti per motivi prestazionali.

Non si considerano gli indici autogenerati dal DBMS. Si è deciso di aggiungere quattro indici sulle colonne ‘DataDiModifica’ di *Nota*, ‘DataDiAccettazione’ di *OffertaAccettata*, ‘DataDiScadenza’ di *Offerta* e ‘Data’ di *Appuntamento*. Questi indici permettono il miglioramento delle prestazioni dell’operazione report clienti e delle operazioni di cancellazione delle offerte scadute, delle note e appuntamenti datati. Infatti il report viene svolto specificando un intervallo temporale. L’operazione di lista delle note è gia ottimizzata dall’indice autogenerato sulla chiave primaria, poiché la ricerca viene effettuata attraverso il codice fiscale.

|  |  |
| --- | --- |
| **Tabella <Nota>** | |
| **Indice <DataDiModifica\_nota\_idx>** | **Tipo[[10]](#footnote-11):** |
| DataDiModifica | <IDX> |

|  |  |
| --- | --- |
| **Tabella <OffertaAccettata>** | |
| **Indice <DataDiAccettazione \_idx>** | **Tipo[[11]](#footnote-12):** |
| DataDiAccettazione | <IDX> |

|  |  |
| --- | --- |
| **Tabella <Offerta>** | |
| **Indice <DataDiScadenza \_idx>** | **Tipo[[12]](#footnote-13):** |
| DataDiScadenza | <IDX> |

|  |  |
| --- | --- |
| **Tabella <Appuntamento>** | |
| **Indice <DataAppuntamento \_idx>** | **Tipo[[13]](#footnote-14):** |
| Data | <IDX> |

## Trigger

Descrivere quali trigger sono stati implementati, mostrando il codice SQL per la loro instanziazione. Si faccia riferimento al fatto che il DBMS di riferimento richiede di utilizzare trigger anche per realizzare vincoli di check ed asserzioni.

É stato creato un trigger che permette di verificare che durante l’inserimento di un’offerta accettata da parte di un cliente, la data di accettazione sia precedente alla data di scadenza dell’offerta a cui sta facendo riferimento. In questo modo si evita che un cliente possa avere associate offerte che durante l’accettazione non erano più valide. Il secondo trigger, simile al primo controlla se effettivamente il momento in cui viene inserita la nota relativa a un cliente ed una offerta, quest’ultima non sia già scaduta. In questo modo si evita che vengono registrate interazione con i clienti in merito ad offerte non più valide. Il terzo trigger verifica che prima dell’inserimento di un appuntamento , non ci siano altri appuntamenti dello stesso cliente nella stessa data, ravvicinati a meno di 2 ore. In questo modo si evita il sovrapporsi di appuntamenti prenotati da un cliente nella stessa data.

Trigger ‘offertaaccettata\_BEFORE\_INSERT’

BEGIN

DECLARE scadenza DATE;

SELECT DataDiScadenza INTO scadenza

FROM offerta

WHERE offerta.CodiceOfferta= NEW.CodiceOfferta;

IF NEW.DataDiAccettazione > scadenza THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Errore: offerta già scaduta!';

END IF;

END

Trigger ‘nota\_BEFORE\_INSERT’

BEGIN

DECLARE scadenza DATE;

SELECT DataDiScadenza INTO scadenza

FROM offerta

WHERE offerta.CodiceOfferta= NEW.CodiceOfferta;

IF NEW.DataDiModifica > scadenza THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Errore: offerta già scaduta!';

END IF;

END

Trigger ‘appuntamento\_BEFORE\_INSERT’

BEGIN

DECLARE counter INT DEFAULT 0;

DECLARE precedente TIME;

DECLARE successivo TIME;

SELECT COUNT(\*) INTO counter

FROM appuntamento

WHERE appuntamento.CodiceFiscaleCliente= NEW.CodiceFiscaleCliente AND

appuntamento.`Data` = NEW.`Data`;

IF counter > 0 THEN

SELECT MAX(Orario) INTO precedente

FROM appuntamento

WHERE appuntamento.Orario< NEW.Orario;

SELECT MIN(Orario) INTO successivo

FROM appuntamento

WHERE appuntamento.Orario> NEW.Orario;

IF precedente IS NOT NULL THEN

IF TIMEDIFF(NEW.Orario, precedente) < '02:00:00' THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Errore: già

presente un appuntamento precedente per il cliente a meno di due ore!';

END IF;

END IF;

IF successivo IS NOT NULL THEN

IF TIMEDIFF(successivo, NEW.Orario) < '02:00:00' THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Errore: già

presente un appuntamento successivo per il cliente a meno di due ore!';

END IF;

END IF;

END IF;

END

## Eventi

Descrivere quali eventi sono stati implementati, mostrando il codice SQL per la loro instanziazione. Si descriva anche se gli eventi sono istanziati soltanto in fase di configurazione del sistema, o se alcuni eventi specifici vengono istanziati in maniera effimera durante l’esecuzione di alcune procedure.

**Si abilita l’event\_scheduler che è disabilitato di default:**

SET GLOBAL event\_scheduler = ON;

**Evento che elimina gli appuntamenti che hanno data più vecchia di 1 anno dalla data corrente:**

CREATE EVENT IF NOT EXISTS ` AppuntamentiCleanup`

ON SCHEDULE

EVERY 1 DAY

STARTS CURRENT\_DATE

ON COMPLETION PRESERVE

DO

DELETE FROM `appuntamento`

WHERE `Data` < DATE\_SUB(CURRENT\_DATE, INTERVAL 1 YEAR));

**Evento che elimina le Note più vecchie di 5 anni dalla data corrente:**

CREATE EVENT IF NOT EXISTS `Note\_Cleanup`

ON SCHEDULE

EVERY 1 DAY

STARTS CURRENT\_DATE

ON COMPLETION PRESERVE

DO

DELETE FROM `nota`

WHERE `DataDiModifica` < DATE\_SUB(CURRENT\_DATE, INTERVAL

5 YEAR);

**Evento che elimina le offerte accettate dai clienti più vecchie di 5 anni dalla data corrente:**

CREATE EVENT IF NOT EXISTS `OffertaAccettata\_Cleanup`

ON SCHEDULE

EVERY 1 DAY

STARTS CURRENT\_DATE

ON COMPLETION PRESERVE

DO

DELETE FROM `offertaaccettata`

WHERE `DataDiAccettazione` < DATE\_SUB(CURRENT\_DATE, INTERVAL

5 YEAR);

## Viste

Mostrare e commentare il codice SQL necessario a creare tutte le viste necessarie per l’implementazione dell’applicazione.

Viene creata questa View per permettere alla segreteria di accedere alla lista delle offerte che sono scadute, che nessuno dei clienti ha accettato e che non hanno note associate, per decidere quali eliminare. Le note sono importanti nel sistema, quindi finchè persiste la nota associata ad un (cliente,offerta) allora l’offerta relativa deve rimanere intatta nella base di dati.

CREATE VIEW Offerte\_Scadute\_Non\_Accettate AS (

SELECT \*

FROM offerta

WHERE CodiceOfferta NOT IN (

SELECT DISTINCT CodiceOfferta

FROM offertaaccettata)

AND CURRENT\_DATE > DataDiScadenza)

AND CodiceOfferta NOT IN(

SELECT DISTINCT CodiceOfferta

FROM nota)

Questa View permette agli operatori di visualizzare i clienti da contattare, in ordine dal meno recentemente contattato. In questo modo gli operatori vedranno i clienti che non sono contattati da più tempo nei primi risultati(per contatto si intende l’interazione con il cliente in questo caso, ovvero il cliente ha risposto).

CREATE VIEW `clienti\_in\_ordine\_di\_contatto` AS

SELECT

`cliente`.`CodiceFiscale` AS `CodiceFiscale`,

`cliente`.`Nome` AS `Nome`,

`cliente`.`Cognome` AS `Cognome`,

`cliente`.`Indirizzo` AS `Indirizzo`,

`cliente`.`Città` AS Città,

`cliente`.`Provincia` AS Provincia,

`cliente`.`Paese` AS Paese,

`cliente`.`DataDiNascita` AS DataDiNascita

`cliente`.`DataDiRegistrazione` AS DataDiRegistrazione

FROM

(cliente LEFT JOIN (SELECT nota.CodiceFiscaleCliente AS CodiceFiscaleCliente,

MAX(nota.DataDiModifica) AS DataRecente

FROM nota

GROUP BY nota.CodiceFiscaleCliente) AS note

ON ((cliente.CodiceFiscale = note.CodiceFiscaleCliente)))

ORDER BY note.DataRecente IS NULL DESC, note.DataRecente

Questa View permette agli operatori di visualizzare le offerte ancora valide per proporle ai clienti.

CREATE VIEW `offerte\_valide` AS

SELECT

`offerta`.`CodiceOfferta` AS `CodiceOfferta`,

`offerta`.`NomeOfferta` AS `NomeOfferta`,

`offerta`.`Descrizione` AS `Descrizione`,

`offerta`.`DataDiScadenza` AS `DataDiScadenza`

FROM

`offerta`

WHERE

(CURRENT\_DATE() <= `offerta`.`DataDiScadenza`)

## Funzioni

Questa funzione permette di simulare un input con variabili variadiche all’interno delle procedure. Viene passata una stringa di valori separati da ‘;’ e questa funzione estrae il campo corretto in base alla posizione specificata. E’ stato scelto come output un varchar(255) per utilizzare la stessa funzione sia per la registrazioni dei numeri di telefono sia delle email dei clienti.

Funzione ‘input\_variadico’:

CREATE FUNCTION `input\_variadico`(valori text, posizione int) RETURNS varchar(255)

DETERMINISTIC

BEGIN

DECLARE campo VARCHAR(255);

DECLARE delimitatore\_max INT;

SET delimitatore\_max = LENGTH(valori) - LENGTH(REPLACE(valori, ';', '')) + 1;

IF posizione <= delimitatore\_max THEN

SET campo = REPLACE(SUBSTRING(SUBSTRING\_INDEX(valori,';', posizione),

LENGTH(SUBSTRING\_INDEX(valori,';', posizione-1)) + 1),

';','');

SET campo= REPLACE(campo, ' ', '');

ELSE

SET campo= NULL;

END IF;

RETURN campo;

END

## Stored Procedures e transazioni

Mostrare e commentare le stored procedure che sono state realizzate per implementare la logica applicativa delle operazioni sui dati, evidenziando quando (e perché) sono state realizzate operazioni transazionali complesse.

Operazione ‘LN’:

CREATE PROCEDURE `lista\_note`(in cf char(16))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

SET TRANSACTION READ ONLY;

START TRANSACTION;

SELECT \*

FROM `nota` LEFT JOIN `appuntamento`

ON (`nota`.CodiceFiscaleCliente,`nota`.CodiceOfferta)=

(`appuntamento`.CodiceFiscaleCliente,`appuntamento`.CodiceOfferta)

WHERE `nota`.CodiceFiscaleCliente=cf;

COMMIT;

END

Operazione ‘SN’:

CREATE PROCEDURE `scrivi\_nota`(in cf char(16), in codiceOfferta char(6), in esito text,

in codiceOperatore char(4), in dataDiModifica date,

in sede varchar(30), in dataAppuntamento date,

in orarioAppuntamento time)

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

START TRANSACTION;

INSERT INTO `nota`(CodiceFiscaleCliente,CodiceOfferta,Esito,

Operatore,DataDiModifica)

VALUES(cf,codiceOfferta,esito,codiceOperatore,dataDiModifica);

IF (sede IS NOT NULL AND dataAppuntamento IS NOT NULL

AND orarioAppuntamento IS NOT NULL) THEN

INSERT INTO `appuntamento`(CodiceFiscaleCliente, CodiceOfferta,

Sede, `Data`, Orario)

VALUES(cf,codiceOfferta,sede,dataAppuntamento,orarioAppuntamento);

END IF;

COMMIT;

END

Operazione ‘IO’

CREATE PROCEDURE `inserisci\_offerta`(in codiceOfferta char(6), in nomeOfferta varchar(30),

in descrizione text, in dataScadenza date)

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

START TRANSACTION;

INSERT INTO `offerta`(CodiceOfferta,NomeOfferta,Descrizione,DataDiScadenza)

VALUES(codiceOfferta,nomeOfferta,descrizione,dataScadenza);

COMMIT;

END

Operazione ‘IOA’

CREATE PROCEDURE `inserisci\_offerta\_accettata`(in codiceOfferta char(6), in cf char(16),

in dataAccettazione date,

in codiceOperatore char(4))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

START TRANSACTION;

INSERT INTO `offertaaccettata`(CodiceOfferta,CodiceFiscaleCliente,

DataDiAccettazione)

VALUES(codiceOfferta,cf,dataAccettazione,codiceOperatore);

COMMIT;

END

Operazione ‘RCC’:

CREATE PROCEDURE `report\_clienti`(in dataInizio date, in dataFine date)

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

SET TRANSACTION READ ONLY;

START TRANSACTION;

SELECT COUNT(DISTINCT CodiceFiscaleCliente) AS Totale

FROM `nota`

WHERE (`nota`.`DataDiModifica`>dataInizio AND `nota`.`DataDiModifica`<dataFine);

SELECT CodiceFiscale,Nome,Cognome, COUNT(DataDiModifica) AS Contattato,

COUNT(DataDiAccettazione) AS Accettato

FROM `nota` JOIN `cliente` ON `nota`.`CodiceFiscaleCliente` = `cliente`.`CodiceFiscale`

LEFT JOIN `offertaaccettata` ON

(`nota`.`CodiceFiscaleCliente`,`nota`.`CodiceOfferta`) =

(`offertaaccettata`.`CodiceFiscaleCliente`,`offertaaccettata`.`CodiceOfferta`)

WHERE (`nota`.`DataDiModifica`>dataInizio AND `nota`.`DataDiModifica`<dataFine)

AND ( (`offertaaccettata`.`DataDiAccettazione`>dataInizio AND

`offertaaccettata`.`DataDiAccettazione`<dataFine) OR

(`offertaaccettata`.`DataDiAccettazione` IS NULL) )

GROUP BY `CodiceFiscale`,`Nome`,`Cognome`;

COMMIT;

END

Operazione ‘RC’:

CREATE PROCEDURE `registra\_cliente`(in cf char(16),in nome varchar(30),

in cognome varchar(30),in indirizzo varchar(50),

in città varchar(30), in provincia char(2),

in paese varchar(30), in dataNascita date,

in dataRegistrazione date, in telefoni text, in email text)

BEGIN

DECLARE posizione INT DEFAULT 1;

DECLARE campo VARCHAR(255);

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

START TRANSACTION;

INSERT INTO `cliente`(CodiceFiscale,Nome,Cognome,Indirizzo,Città,Provincia,Paese,

DataDiNascita,DataDiRegistrazione)

VALUES(cf,nome,cognome,indirizzo,città,provincia,paese,dataNascita,dataRegistrazione);

WHILE input\_variadico(telefoni, posizione) IS NOT NULL DO

SET campo = input\_variadico(telefoni, posizione);

INSERT INTO `telefono`(NumeroDiTelefono,CodiceFiscaleCliente)

VALUES(campo,cf);

SET posizione = posizione + 1;

END WHILE;

SET posizione=1;

WHILE input\_variadico(email, posizione) IS NOT NULL DO

SET campo = input\_variadico(email, posizione);

INSERT INTO `email`(nomeEmail,CodiceFiscaleCliente) VALUES(campo,cf);

SET posizione = posizione + 1;

END WHILE;

COMMIT;

END

Operazione ‘L1’:

CREATE PROCEDURE `login` (in var\_username varchar(30), in var\_pass varchar(30),

out var\_role INT)

BEGIN

DECLARE var\_user\_role ENUM('operatore', 'segreteria');

SELECT `Ruolo`

FROM `utenti`

WHERE `Username` = var\_username AND `Password` = MD5(var\_pass)

INTO var\_user\_role;

IF var\_user\_role = 'operatore' THEN SET var\_role = 1;

ELSEIF var\_user\_role = 'segreteria' THEN SET var\_role = 2;

ELSE SET var\_role = 3;

END IF;

END

Altre procedure utili per la base di dati:

L’operazione per mostrare le offerte scadute è utile per la segreteria per avere i dettagli delle offerte candidate ad essere eliminate dalla base di dati. L’eliminazione è gestita attraverso l’operazione ‘elimina\_offerte’.

Operazione ‘Mostra\_offerte\_scadute’:

CREATE PROCEDURE `mostra\_offerte\_scadute`()

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

SET TRANSACTION READ ONLY;

START TRANSACTION;

SELECT \*

FROM `offerte\_scadute\_non\_accettate`;

COMMIT;

END

Operazione ‘Elimina\_offerte’:

CREATE PROCEDURE `elimina\_offerte`(in codiceOfferta char(6))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

START TRANSACTION;

IF (SELECT COUNT(\*) FROM `offerte\_scadute\_non\_accettate`

WHERE `CodiceOfferta`= codiceOfferta)>0 THEN

DELETE FROM `offerta` WHERE `CodiceOfferta`= codiceOfferta;

ELSE

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Offerta non scaduta o

ancora attiva dai clienti.';

END IF;

COMMIT;

END

Le procedure che seguono sono utili per l’operatore quando deve contattare un cliente. L’operazione ‘mostra\_clienti’ attraverso l’ausilio di una View mostra i clienti in ordine da favorire le chiamate verso quelli che non sono contatatti da più tempo (per contatto si intende che il cliente risponda). L’operazione ‘mostra\_offerte’ ,anch’essa attraverso l’uso di una View,mostra agli operatori le offerte che ancora sono valide, per proporle ai clienti. Seguono le operazioni per mostrare i recapiti telefoni dei clienti e le rispettive email.

Operazione ‘Mostra\_clienti’:

CREATE PROCEDURE `mostra\_clienti`()

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

SET TRANSACTION READ ONLY;

START TRANSACTION;

SELECT \*

FROM `clienti\_in\_ordine\_di\_contatto`;

COMMIT;

END

Operazione ‘Mostra\_offerte’:

CREATE PROCEDURE `mostra\_offerte`()

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

SET TRANSACTION READ ONLY;

START TRANSACTION;

SELECT \*

FROM `offerte\_valide`;

COMMIT;

END

Operazione ‘Mostra\_Telefoni’:

CREATE PROCEDURE `mostra\_telefoni`(in cf char(16))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

SET TRANSACTION READ ONLY;

START TRANSACTION;

SELECT NumeroDiTelefono

FROM `telefono`

WHERE `telefono`.CodiceFiscaleCliente=cf;

COMMIT;

END

Operazione ‘Mostra\_Email’:

CREATE PROCEDURE `mostra\_email`(in cf char(16))

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

SET TRANSACTION READ ONLY;

START TRANSACTION;

SELECT NomeEmail

FROM `email`

WHERE `email`.CodiceFiscaleCliente=cf;

COMMIT;

END

1. Indicare con E le entità, con R le relazioni [↑](#footnote-ref-2)
2. PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna. [↑](#footnote-ref-3)
3. PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna. [↑](#footnote-ref-4)
4. PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna. [↑](#footnote-ref-5)
5. PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna. [↑](#footnote-ref-6)
6. PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna. [↑](#footnote-ref-7)
7. PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna. [↑](#footnote-ref-8)
8. PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna. [↑](#footnote-ref-9)
9. PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna. [↑](#footnote-ref-10)
10. IDX = index, UQ = unique, FT = full text, PR = primary. [↑](#footnote-ref-11)
11. IDX = index, UQ = unique, FT = full text, PR = primary. [↑](#footnote-ref-12)
12. IDX = index, UQ = unique, FT = full text, PR = primary. [↑](#footnote-ref-13)
13. IDX = index, UQ = unique, FT = full text, PR = primary. [↑](#footnote-ref-14)