Università Politecnica delle Marche

ZEUSPHONE S.p.A.

Progetto di NGDB

Bedetta Alessandro
Caporusso Chiara Amalia
Galeazzi Margherita

2023

Indice

Elenco delle tabelle

Elenco delle figure

Elenco degli snippet di codice

Capitolo 1

Introduzione

La seguente relazione tratterà la progettazione e l'implementazione di una base di dati per un operatore fisico, già operante, sul territorio italiano, nel settore delle telecomunicazioni da diversi anni. Per adattarsi alle nuove tecnologie e migliorare l'assistenza e il supporto dei clienti(con particolare occhio di riguardo nei confronti delle aziende), l'azienda necessita di espandere le proprie infrastrutture, e perciò si è deciso di optare per un sistema NoSQL.
I sistemi NoSQL, offrono diversi vantaggi rispetto ai classici database relazionali (RDBMS), riportiamo i più importanti di seguito:

- **Scalabilità orizzontale**: I sistemi NoSQL sono progettati per scalare orizzontalmente, consentendo di gestire grandi volumi di dati e un aumento del carico di lavoro distribuendo i dati su più nodi. Ciò consente di aumentare le prestazioni e la capacità del sistema in modo rapido ed efficiente, si prestano perciò a lavorare con grandi moli di dati.
- **Flessibilità dello schema**: A differenza dei dei database relazionali, che richiedono uno schema rigido e predefinito, i sistemi NoSQL offrono una maggiore flessibilità dello schema, ciò significa che è possibile aggiungere, rimuovere o modificare campi all'interno dei documenti o delle colonne senza dover apportare modifiche allo schema dell'intero database. Ciò risulta particolarmente utile in scenari in cui la struttura dei dati può variare nel tempo.
- **Alta velocità di lettura/scrittura**: I sistemi NoSQL sono ottimizzati per prestazioni elevate, consentendo di gestire grandi volumi di operazioni di lettura e scrittura in modo rapido.
- **Supporto per dati non strutturati**: I sistemi NoSQL sono in grado di gestire dati non strutturati, come documenti JSON, BSON, file multimediali, dati gerarchici e dati a grafo. Questa flessibilità consente di gestire una vasta gamma di tipi di dati, consentendo una maggiore agilità nello sviluppo di applicazioni.
- **Disponibilità e tolleranza ai guasti**: I sistemi NoSQL sono progettati per essere altamente disponibili e tolleranti ai guasti. Utilizzando tecniche come la replica dei dati su più nodi e la distribuzione dei dati in modo ridondante, i sistemi NoSQL possono continuare a funzionare anche in presenza di guasti hard o soft.
- **Diversi modelli di dati**: I sistemi NoSQL offrono diversi modelli di dati, come document, column-family, key-value e graph.

Bisogna però prestare attenzione al fatto che i sistemi NoSQL non sono adatti a tutti i casi, curando le esigenze specifiche del progetto. È quindi buona norma, prima di scegliere il tipo di database da implementare, condurre un'analisi approfondita dei requisiti.

Capitolo 2

Analisi dei requisiti e contesto di riferimento

Questo capitolo definirà il contesto di riferimento del progetto che andremo poi ad implementare. Al fine di sviluppare un sistema informativo che sia in linea con le necessità dell'azienda, la conoscenza del contesto di riferimento in cui esso opererà è un requisito minimo. Si prevede che la base di dati risultante da questo progetto avrà una durata di circa tre anni, a seguito dei quali potrebbero essere necessarie operazioni di manutenzione o di trasferimento nel sistema di gestione dei dati storici aziendali.

Il contesto in cui opererà il nostro sistema informativo è quello di un azienda operante nel settore delle telecomunicazioni, la ZeusPhone S.p.A., in questo ambiente, i **clienti** contattano l'azienda o per stipulare un nuovo **contratto** oppure per segnalare un **guasto**; a seguito della stipulazione poi vengono generate le **bollette** relative ai vari contratti.
A seguito quindi del rilevamento di un guasto, viene aperta una pratica di guasto e per la risoluzione di questo, al guasto vengono assegnati uno o più **tecnici**.
Il rapporto tra tecnici e azienda è regolamentato mediante un **contratto** di assunzione.

Mediante l'analisi dei requisiti si vogliono identificare i problemi, che il sistema frutto del nostro progetto, dovrà risolvere e le caratteristiche che esso dovrà possedere.

Requisiti funzionali

L'obiettivo che vogliamo perseguire è la creazione di una base di dati che faciliti per l'azienda la gestione dei clienti, dei dipendenti e dei guasti agli impianti di trasmissione.

I requisiti funzionali che il nostro sistema dovrà possedere, sono:

- Gestione delle attività CRUD (Create Read Update) efficiente;
- Semplicità di utilizzo.

Nelle sezioni successive si approfondiranno i requisiti relativi alle varie componenti che fanno parte del contesto.

Requisiti circa i contratti

Per quanto riguarda i contratti stipulati con i clienti, si vogliono conoscere i campi relativi ai dati anagrafici dello stesso, la tipologia del cliente, l'indirizzo di fornitura, il codice di migrazione, il tipo di collegamento della linea fissa, i dati relativi all'offerta scelta, la data di fine vincolo, il metodo di pagamento e la data di stipulazione. Per quanto riguarda i contratti relativi ai dipendenti basterà solamente inserire i dati essenziali.

Requisiti circa gli impianti

Deve essere nota la tipologia dell'impianto, lo stato (ovvero se attivo o inattivo) e il relativo codice.

Requisiti circa i guasti

Relativamente ai guasti dovranno essere conservate anche le informazioni riguardanti le segnalazioni da parte dei clienti sulla base delle quali si andranno ad aprire guasti relativi ai problemi all'impianto interessato.
Tra le varie informazioni che riguardano il guasto, dovranno essere presenti la data e l'ora di apertura e anche quelle di chiusura.

Requisiti circa i tecnici

Ogni tecnico è identificato da un codice che verrà associato al guasto da riparare. Dei tecnici verranno gestiti, oltre ai dati anagrafici, anche i rispettivi status, ovvero se il tecnico è o meno disponibile per essere assegnato ad un guasto aperto. Dei tecnici viene riportata anche la specializzazione.

Requisiti non funzionali

Altri requisiti importanti per il nostro progetto ma non funzionali ad esso, sono:

- Utilizzo di database NoSQL, nello specifico utilizzeremo MongoDB e la sua GUI, Compass;
- Utilizzo di sole tecnologie e librerie Open-Source, per ridurre i costi ed avere una maggiore sicurezza.

Mongo DB

Quando parliamo di MongoDB stiamo descrivendo un database open-source NoSQL che offre soluzioni flessibili, scalabili e ad alte prestazioni per la gestione di grandi quantità di dati. È progettato per affrontare le sfide presentate dalle moderne applicazioni le quali spesso richiedono l'archiviazione e il recupero di dati complessi e non strutturati.

A differenza dei database relazionali tradizionali, MongoDB segue un modello di dati **orientato al documento**. Ciò significa che i dati vengono memorizzati in documenti flessibili, simili a JSON, consentendo una rappresentazione dinamica e senza schema. Ogni documento può avere una struttura unica, che consente agli sviluppatori di memorizzare e recuperare i dati nel modo più adatto alle esigenze della loro applicazione.

L'architettura di MongoDB si basa sulla logica di **database distribuito** (su vari server), fornendo scalabilità orizzontale e alta disponibilità. Utilizza un concetto chiamato **sharding** , che prevede il partizionamento dei dati tra macchine diverse, consentendo un'espansione senza soluzione di continuità man mano che i dati crescono. Questa natura distribuita fa sì che MongoDB sia in grado di gestire grandi volumi di dati e sopportare carichi di traffico elevati in modo efficiente.

Uno dei maggiori punti di forza di MongoDB è la sua flessibilità in fase di interrogazione. Supporta un ricco linguaggio di query che consente agli sviluppatori di eseguire **query complesse** e aggregazioni sui dati, rendendolo adatto a una vasta gamma di casi d'uso. Il linguaggio di interrogazione supporta una varietà di operatori (come le query geo-spaziali o la ricerca di testo) i quali consentono agli sviluppatori di esprimere facilmente complesse manipolazioni dei dati. Un'altra caratteristica degna di nota di MongoDB è la sua capacità di fornire operazioni ad alte prestazioni e bassa latenza. Ciò avviene attraverso diversi meccanismi, tra cui file mappati in memoria, I/O asincrono e cache integrata. Queste ottimizzazioni, combinate con la sua scalabilità orizzontale, consentono a MongoDB di gestire carichi di lavoro di lettura e scrittura pesanti mantenendo la reattività.

Mongo offre anche un set completo di funzionalità per la replica dei dati e la tolleranza di errore. Supporta la replica automatica dei dati su più server, fornendo ridondanza dei dati e assicurando che il sistema rimanga operativo anche in caso di guasti hardware. Oltre a ciò offre varie opzioni per il backup dei dati, il ripristino puntuale e le transazioni distribuite, dando agli sviluppatori gli strumenti per costruire applicazioni robuste e affidabili.

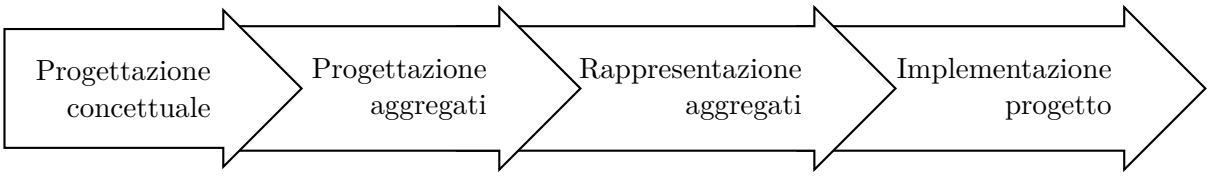
Inoltre, MongoDB ha una comunità vivace e attiva, che ha contribuito a realizzare una ricca varietà di librerie, strumenti e framework che si integrano perfettamente con il database. Questo ampio supporto rende più facile per gli sviluppatori lavorare con MongoDB e accelera il processo di sviluppo.

Riassumendo, MongoDB è un database NoSQL moderno, flessibile e scalabile che offre un modello di dati orientato ai documenti, un'architettura distribuita, potenti capacità di interrogazione, prestazioni elevate e una robusta tolleranza di errore. Ha ottenuto un'adozione diffusa in vari settori e casi d'uso, che vanno dalle applicazioni web su larga scala all'analisi in tempo reale e alle applicazioni Internet of Things (IoT)

Capitolo 3

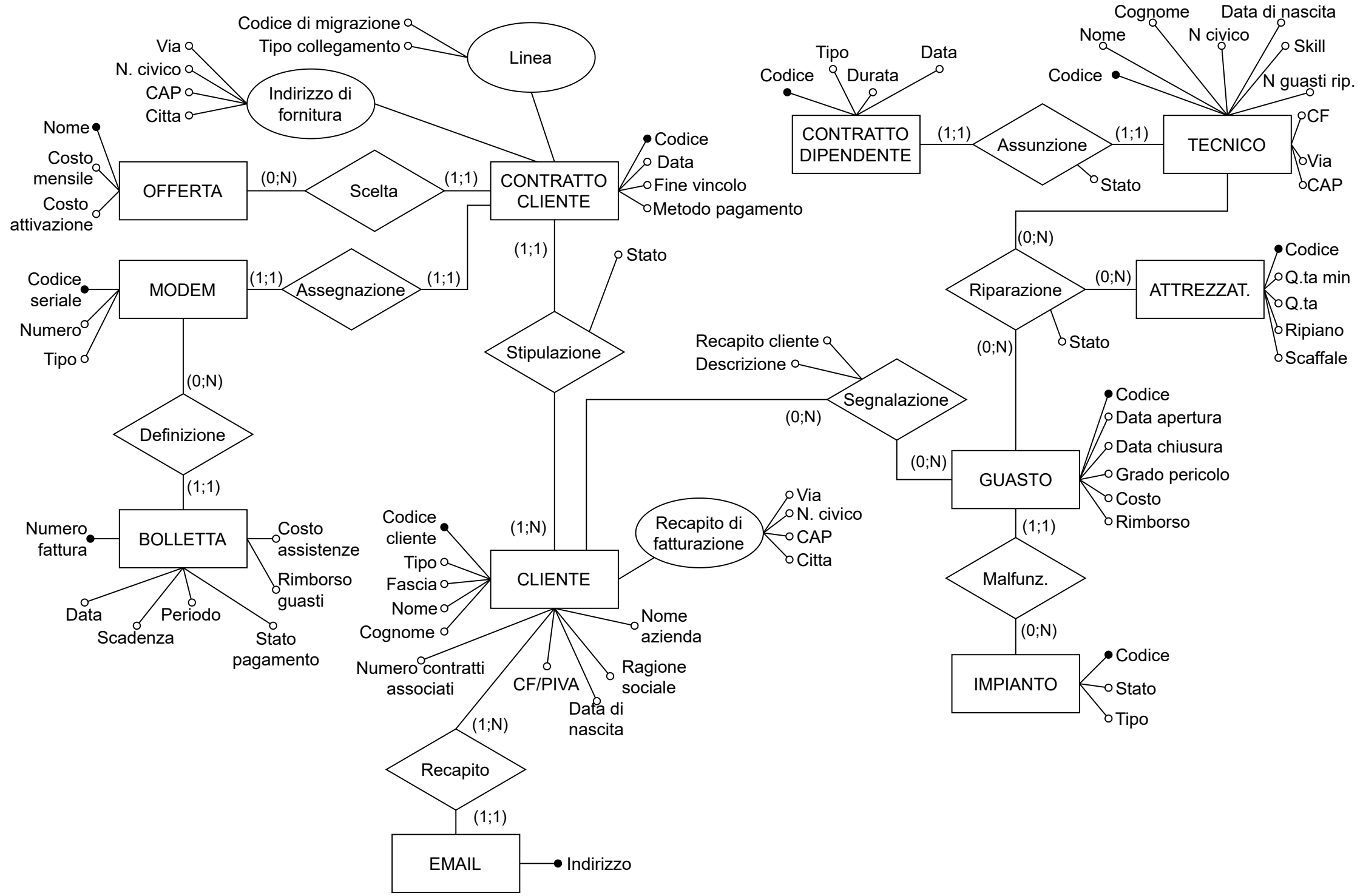
Progettazione

La fase principale del lavoro svolto è appunto la progettazione, permettendoci di implementare il nostro sistema SQL. Tale fase si concentra sul come andremo ad organizzare i dati nel database, facendo quindi riferimento agli stessi obiettivi di progettazione dei database relazionali. Nel diagramma sottorappresentato vengono riportati i passaggi fondamentali della progettazione di un database NoSQL.



Progettazione concettuale

Seguendo il processo di progettazione riportato nel diagramma, il primo passaggio riguarda la progettazione concettuale. Di seguito viene riportato lo schema *entity-relationship* utilizzato.


Codice di migrazione
Tipo collegamento
Linea
Via
N. civico
CAP
Citta
Indirizzo di fornitura
Nome
Costo mensile
Costo attivazione
OFFERTA
(0;N)
Scelta
(1;1)
CONTRATTO CLIENTE
Codice
Data
Fine vincolo
Metodo pagamento
Codice seriale
Numero
Tipo
MODEM
(1;1)
Assegnazione
(1;1)
(1;1)
Stato
Stipulazione
(0;N)
Definizione
(1;1)
Numero fattura
BOLLETTA
Costo assistenze
Rimborso guasti
Data
Scadenza
Periodo
Stato pagamento
Codice cliente
Tipo
Fascia
Nome
Cognome
(1;N)
CLIENTE
Recapito di fatturazione
Via
N. civico
CAP
Citta
Nome azienda
Numero contratti associati
CF/PIVA
Ragione sociale
Data di nascita
(1;N)
Recapito
(1;1)
EMAIL
Indirizzo
Recapito cliente
Descrizione
Segnalazione
(0;N)
(0;N)
Tipo
Durata
Data
Codice
CONTRATTO DIPENDENTE
(1;1)
Assunzione
(1;1)
Stato
Cognome
Data di nascita
Nome
N civico
Skill
Codice
N guasti rip.
TECNICO
CF
Via
CAP
(0;N)
Riparazione
(0;N)
ATTREZZAT.
Stato
Codice
Q.ta min
Q.ta
Ripiano
Scaffale
(0;N)
(0;N)
GUASTO
Codice
Data apertura
Data chiusura
Grado pericolo
Costo
Rimborso
(1;1)
Malfunz.
(0;N)
IMPIANTO
Codice
Stato
Tipo

Osservando quindi lo schema E-R riportato alla pagina precedente possiamo vedere che l'entità principale è **CLIENTE**. Tale entità risulta per l'appunto essere collegata ad altre entità fondamentali per lo sviluppo del progetto, quali: Guasto e di conseguenza Impianto e Tecnico, Contratto Cliente, Bolletta e Offerta.

Dallo Schema E-R di partenza abbiamo effettuato delle **semplificazioni**, potendoci quindi concentrare in maniera più dettagliata sulle entità centrali dello schema.

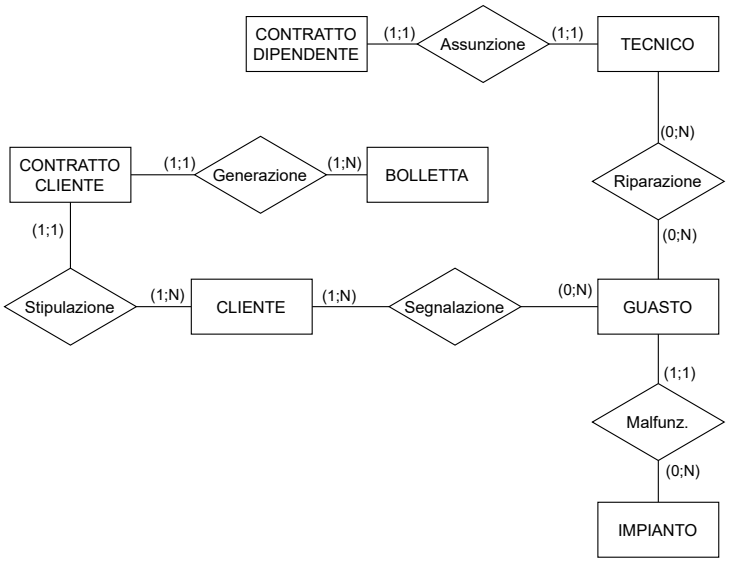


Figura 3.1: Schema E-R semplificato

Come si può vedere, dallo schema E-R allo schema semplificato sono state rimosse alcune entità, come:

- L'entità Attrezzatura che è stata rimossa in quanto considerata come secondaria;
- L'entità E-Mail che è stata rimossa in quanto si è deciso di accorpare tutte le mail appartenenti al cliente all'interno dell'entità Cliente;
- L'entità Modem che è stata rimossa per garantire una maggiore attenzione della parte principale del progetto, accorpando il codice seriale del Modem in Contratto Cliente e collegando l'entità Bolletta direttamente a Contratto Cliente;

Considerazioni

- L'entità **Cliente** deve essere intesa come la persona fisica che decide di sottoscrivere un contratto;

- L'entità **Contratto Cliente** deve essere intesa come riferimento a uno specifico contratto che può sottoscrivere il cliente, che quindi può essere più di uno. Tale scelta è stata ereditata dal progetto precedente in quanto il servizio di fornitura può essere erogato ad indirizzi diversi rispetto all'ubicazione del cliente.
- Sia nell'entità **Bolletta** che nell'entità **Guasto** sono presenti due attributi relativi ad eventuali costi di assistenza e rimborsi che vengono presi in considerazione nella registrazione della bolletta del cliente, a seguito di una segnalazione da parte dello stesso. Tali attributi non verranno conderati al fine di per garantire una maggiore attenzione sullo sviluppo della parte centrale del progetto in esame.

Progettazione degli aggregati

In questa fase di progettazione si sono andati a definire gli aggregati. Definire un aggregato e la sua progettazione significa andare a considerare due aspetti fondamentali, in cui il primo è legato ai concetti di *entità radice* ed *entità annidata*. Andiamo quindi a considerare il concetto di annidamento, in cui si vede l'entità radice come macro-oggetto contenente diversi micro-oggetti, considerati come suoi attributi. Il secondo aspetto è legato al concetto dei **confini** che dovrà avere l'aggregato, ovvero quali delle entità e relazioni devono essere mantenute insieme come elementi dell'entità radice.
È inoltre necessario, per una corretta definizione di questi aspetti, prestare particolare attenzione ai pattern di accesso ai dati, ovvero alla direzione con cui le operazioni di lettura e scrittura vengono effettuate. Nel caso in questione, le entità considerate ai fini della progettazione degli aggregati sono riportate in Figura 3.1.
In questa analisi abbiamo quindi concentrato l'attenzione sulle possibili alternative nella scelta dell'entità radice e delle entità annidate per i vari aggregati.

Analisi

Nei paragrafi precedenti sono state fatte alcune considerazioni, ma non a livello dettagliato. In questa sezione, attraverso un'accurata analisi, andremo a definire gli aggregati che meglio si adattano al progetto in esame e che hanno un *costo* minore per il contesto in considerazione.
Per realizzare tale analisi quantitativamente si impiegherà la **tavola dei volumi** (Tab. 3.1), delle operazioni e le relative frequenze (Tab. 3.2).

Tabella 3.1: Tavola dei volumi

CONCETTO	TIPO	VOLUME
Contratto	E	100.500
Contratto Cliente	E	100.000
Contratto Dipendente	E	400.000
Cliente	E	50.000
Guasto	E	1.000.000
Impianto	E	200.000
Bolletta	E	4.000.000
Tecnico	E	400.000
Stipulazione	R	100.000
Definizione	R	4.000.000

CONCETTO	TIPO	VOLUME
Segnalazione	R	2.000.000
Assunzione	R	400.000
Riparazione	R	2.000.000
Malfunzionamento	R	200.000

Tabella 3.2: Tavola delle operazioni con frequenza

OPERAZIONE	FREQUENZA
1. **Inserimento nuovo contratto cliente**	15 volte al giorno
2. Inserimento nuovo tecnico	10 volte l'anno
3. **Inserimento nuovo guasto**	15 volte al giorno
4. Inserimento nuova bolletta linea fissa	1 volta al mese (per ogni contratto)
5. Consultazione guasti, chiusi nella giornata odierna, visualizzando la durata di riparazione del guasto, ordinati per fascia cliente e durata di riparazione	1 volta al giorno
6. Modifica contratto cliente	4 volte a settimana
7. Modifica contratto tecnico	3 volte l'anno
8. **Chiusura guasto**	13 volte al giorno
9. Cancellazione contratto cliente	1 volta al mese
10. Cancellazione tecnico	1 volta l'anno
11. **Consultazione dati dei clienti**	20 volte al giorno
12. **Consultazione dati dei contratti clienti**	10 volte al giorno
13. **Consultazione dati dei tecnici**	15 volte al giorno
14. **Consultazione dati dei contratti dipendenti**	5 volte al giorno
15. **Consultazione dati della bolletta**	30 volte al giorno
16. Statistica acquisizione clienti	1 volta al mese
17. Statistica contratti per tipo collegamento	1 volta ogni 6 mesi
18. Statistica numero guasti riparati per tecnico	1 volta al mese
19. Statistica numero tecnici per skill	2 volte l'anno
20. Statistica guasti di un dato impianto	1 volta al mese
21. Statistica numero clienti in base alla fascia	2 volte l'anno
22. Statistica numero contratti per cliente	1 volta al mese
23. Statistica ragione sociale azienda	2 volte l'anno
24. Statistica tecnici con contratto a tempo indeterminato	1 volta l'anno
25. Statistica numero guasti per grado di pericolo	1 volta al mese
26. Verifica avvenuto pagamento bolletta da parte dei clienti	1 volta al giorno
27. Visualizzazione bollette con scadenza la settimana successiva alla data di consultazione	1 volta a settimana
28. Aggiornamento pagamento	1 volta al mese (per ogni contratto)
29. **Consultazione tecnici occupati**	15 volte al giorno
30. Cessazione contratto cliente	1 volta al mese
31. Cessazione contratto tecnico	1 volta l'anno

Come si può notare, per la progettazione degli aggregati si sono considerate soltanto le operazioni più frequenti e le relative entità rimanenti contenute nello schema semplificato. Le operazioni principali

sono state evidenziate in grassetto nella precedente tabella. Oltre alle operazioni presenti sono state introdotte ulteriori operazioni e le loro relative frequenze, con l'obiettivo di adattare al meglio il precedente progetto alla progettazione e l'analisi di un sistema NoSQL. Si è quindi deciso di introdurre:

- Visualizzazione guasti per l'impianto X (30 volte al giorno);
- Visualizzazione bollette non pagate da un dato cliente X (40 volte a giorno);
- Visualizzazione segnalazioni fatte dal cliente X (8 volte al giorno);
- Visualizzazione impianti in riparazione (5 volte al giorno);

Un'ulteriore nota da fare è sulle entità Contratto Cliente e Offerta. Essendo le operazioni di modifica dell'offerta spesso eseguite relativamente allo specifico contratto del cliente e non avendo queste operazioni una sufficiente rilevanza fra quelle previste si è scelto di considerare Offerta annidata all'entità radice Contratto Cliente. In fine, per quanto riguarda Tecnico e Contratto Dipendente, essendo legate da una relazione di tipo 1:1 si è scelto di considerare la seconda entità annidata nella prima.
Nelle analisi realizzate di seguito si è considerato che il costo di una scrittura equivale al costo di due letture (1S = 2L).

Aggregato - CONTRATTO CLIENTE & CLIENTE

Nel paragrafo corrente, l'attenzione è rivolta all'aggregato composto dalle entità *"CONTRATTO CLIENTE"* e *"CLIENTE"*. Sono state analizzate le operazioni relative a queste entità al fine di calcolare e determinare l'entità meno costosa per le operazioni e le frequenze considerate, da assegnare al ruolo di entità radice.
Valuteremo i costi di entrambe le alternative, in cui in un caso un'entità assume il ruolo di entità radice e l'altra di entità annidata e viceversa. Nella parte seguente si riportano i calcoli realizzati.

Operazioni:

Consultazione dati dei clienti (20 volte a giorno)
Consultazione dati dei contratti clienti (10 volte al giorno)
Inserimento contratto cliente (15 volte a giorno)

- Numero medio di contratti per clienti: $\frac{100.000}{50.000} = 2$
- Numero medio di clienti per contratto: $\frac{100.000}{100.000} = 1$

Tabella 3.3: Analisi aggregato CONTRATTO CLIENTE & CLIENTE

OPERAZIONE	**CONTRATTO in CLIENTE**	**CLIENTE in CONTRATTO**
Consultazione dati dei clienti (20/g)	$1L \cdot 20 = 20L$	$1L \cdot 20 = 20L$
Consultazione dati dei contratti clienti (10/g)	$1L \cdot 10 = 10L$	$1L \cdot 10 = 10L$
Inserimento contratto cliente (15/g)	$(1L + 1S) \cdot 15 = 45L$	$1S \cdot 15 = 30L$
TOTALE	$75L$	$60L$

In base a quanto è emerso durante la precedente fase di analisi Tabella 3.3 si è deciso di includere l'informazione relativa al CLIENTE nell'entità radice CONTRATTO CLIENTE.

Aggregato - CONTRATTO CLIENTE & BOLLETTA

Nel paragrafo corrente, l'attenzione è rivolta all'aggregato composto dalle entità *"CONTRATTO CLIENTE"* e *"BOLLETTA"*. Sono state analizzate le operazioni relative a queste entità al fine di calcolare e determinare l'entità meno costosa per le operazioni e le frequenze considerate, da assegnare al ruolo di entità radice. Durante la valutazione del numero di letture e scritture, è stato considerato il risultato ottenuto dall'analisi di CONTRATTO CLIENTE & CLIENTE (quindi i dati di CLIENTE in CONTRATTO CLIENTE).
Valuteremo i costi di entrambe le alternative, in cui in un caso un'entità assume il ruolo di entità radice e l'altra di entità annidata e viceversa. Nella parte seguente si riportano i calcoli realizzati.

Operazioni:

Consultazione dati della bolletta (30 volte al giorno)
Visualizzazione bollette non pagate da un dato cliente X (40 volte a giorno);

- Numero medio di bollette per contratto cliente: $\frac{4.000.000}{100.000} = 40$
- Numero medio di contratti cliente per bolletta: $\frac{4.000.000}{4.000.000} = 1$

Tabella 3.4: Analisi aggregato CONTRATTO CLIENTE & BOLLETTA

OPERAZIONE	**CONTRATTO in BOLLETTA**	**BOLLETTA in CONTRATTO**
Consultazione dati della bolletta (30/g)	$1L \cdot 30 = 30L$	$1L \cdot 30 = 30L$
Visualizzazione bollette non pagate da un cliente X (40/g);	$40L \cdot 40 = 1600L$	$1L \cdot 40 = 40L$
TOTALE	$1630L$	$70L$

In base a quanto è emerso dai calcoli riportati nella Tabella 3.4 si è deciso di utilizzare CONTRATTO CLIENTE come entità radice e BOLLETTA come entità annidata.

Aggregato - CLIENTE & GUASTO

Rivolgeremo l'attenzione ora all'aggregato composto dalle entità *"CLIENTE"* e *"GUASTO"*. Sono state analizzate le operazioni relative a queste entità al fine di calcolare e determinare l'entità meno costosa per le frequenze delle operazioni considerate, da assegnare al ruolo di entità radice.
Valuteremo i costi di entrambe le alternative, in cui in un caso un'entità assume il ruolo di entità radice e l'altra di entità annidata e viceversa. Verrà valutata anche una terza alternativa, ovvero il caso in cui il confine dell'aggregato è la relationship. Nella parte seguente si riportano i calcoli realizzati.

Operazioni:

Inserimento nuovo guasto (15 volte al giorno);
Chiusura guasto (13 volte al giorno);
Visualizzazione segnalazioni fatte dal cliente X (8 volte al giorno)

- Numero medio di guasti per cliente: $\frac{2.000.000}{50.000} = 40$
- Numero medio di clienti per guasto: $\frac{2.000.000}{1.000.000} = 2$

Tabella 3.5: Analisi aggregato CLIENTE & GUASTO

OPERAZIONE	**CLIENTE in GUASTO**	**GUASTO in CLIENTE**	**K(CLIENTE) IN GUASTO**
Inserimento nuovo guasto (15 volte al giorno)	$1S \cdot 15 = 30L$	$(1L+1S) \cdot 2 \cdot 15 = 90L$	$1S \cdot 15 = 30L$
Chiusura guasto (13 volte al giorno)	$(1L + 1S) \cdot 13 = 39L$	$(1L+1S) \cdot 2 \cdot 13 = 78L$	$(1L + 1S) \cdot 13 = 39L$
Visualizzazione segnalazioni fatte dal cliente X (8 volte al giorno)	$1L \cdot 40 \cdot 8 = 320L$	$1L \cdot 8 = 8L$	$1L \cdot 8 = 8L$
TOTALE	$389L$	$176L$	$77L$

In base a quanto è emerso dai calcoli riportati nella Tabella 3.5 si è deciso di lasciare CONTRATTO CLIENTE e GUASTO separati.

Aggregato - TECNICO & GUASTO

Nel paragrafo corrente, l'attenzione è rivolta all'aggregato composto dalle entità *"TECNICO"* e *"GUASTO"*. Sono state analizzate le operazioni relative a queste entità al fine di calcolare e determinare l'entità meno costosa per le operazioni e le frequenze considerate, da assegnare al ruolo di entità radice. Valuteremo i costi di entrambe le alternative, in cui in un caso un'entità assume il ruolo di entità radice e l'altra di entità annidata e viceversa. Nella parte seguente si riportano i calcoli realizzati. Verrà valutata anche una terza alternativa, ovvero il caso in cui il confine dell'aggregato è la relationship. Siamo poi andati ad analizzare le entità *Tecnico* e *Guasto*.

Operazioni:

Consultazione dati dei tecnici (15 volte al giorno);
Consultazione tecnici occupati (15 volte al giorno);

- Numero medio di tecnici per guasto: $\frac{2.000.000}{400.000} = 5$
- Numero medio di guasti per tecnico: $\frac{2.000.000}{1.000.000} = 2$

Tabella 3.6: Analisi aggregato TECNICO & GUASTO

OPERAZIONE	**TECNICO in GUASTO**	**GUASTO in TECNICO**	**K(GUASTO) IN TECNICO**
Consultazione dati dei tecnici (15 volte al giorno)	$2L \cdot 15 = 30$	$1L \cdot 15 = 15L$	$1L \cdot 15 = 15L$
Consultazione tecnici occupati (15 volte al giorno)	$2L \cdot 15 = 30$	$5L \cdot 15 = 75L$	$1L \cdot 15 = 15L$
TOTALE	$60L$	$90L$	$30L$

In base a quanto è emerso dai calcoli riportati nella Tabella 3.6 si è deciso di lasciare TECNICO e GUASTO separati.

Aggregato - GUASTO & IMPIANTO

Nel paragrafo corrente, l'attenzione è rivolta all'aggregato composto dalle entità *"GUASTO"* e *"IMPIANTO"*. Sono state analizzate le operazioni relative a queste entità al fine di calcolare e determinare l'entità meno costosa per le operazioni e le frequenze considerate, da assegnare al ruolo di entità radice. Valuteremo i costi di entrambe le alternative, in cui in un caso un'entità assume il ruolo di entità radice e l'altra di entità annidata e viceversa. Nella parte seguente si riportano i calcoli realizzati.

Operazioni:

Visualizzazione guasti per l'impianto X (30 volte al giorno)
Visualizzazione impianti in riparazione (5 volte a giorno)

- Numero medio di guasti per impianto: $\frac{1.000.000}{200.000} = 5$
- Numero medio di impianti per guasto: $\frac{1.000.000}{1.000.000} = 1$

Tabella 3.7: Analisi aggregato GUASTO & IMPIANTO

OPERAZIONE	**GUASTO in IMPIANTO**	**IMPIANTO in GUASTO**
Visualizzazione impianti in riparazione (5/g)	$1L \cdot 5 = 5L$	$1L \cdot 5 = 5L$
Visualizzazione guasti per l'impianto X (30/g)	$1L \cdot 30 = 30L$	$5L \cdot 30 = 150L$
TOTALE	$35L$	$155L$

In base a quanto è emerso dai calcoli riportati nella Tabella 3.7 si è deciso di utilizzare IMPIANTO come entità radice e GUASTO come entità annidata.

Aggregati finali

Ricapitolando, gli aggregati finali ottenuti dalle analisi svolte precedentemente sono:

1° aggregato: costituito da CONTRATTO CLIENTE, che al suo interno contiene CLIENTE, BOLLETTA e OFFERTA;

2° aggregato: costituito da TECNICO, che al suo interno contiene CONTRATTO DIPENDENTE.

3° aggregato: costituito da IMPIANTO, che al suo interno contiene GUASTO;

Inoltre, si sono previsti dei collegamenti sia tra l'aggregato relativo a Contratto Cliente e Impianto e sia tra l'aggregato relativo a Impianto e Tecnico.

I singoli aggregati che sono stati analizzati precedentemente vengono riportati in Figura 3.2.

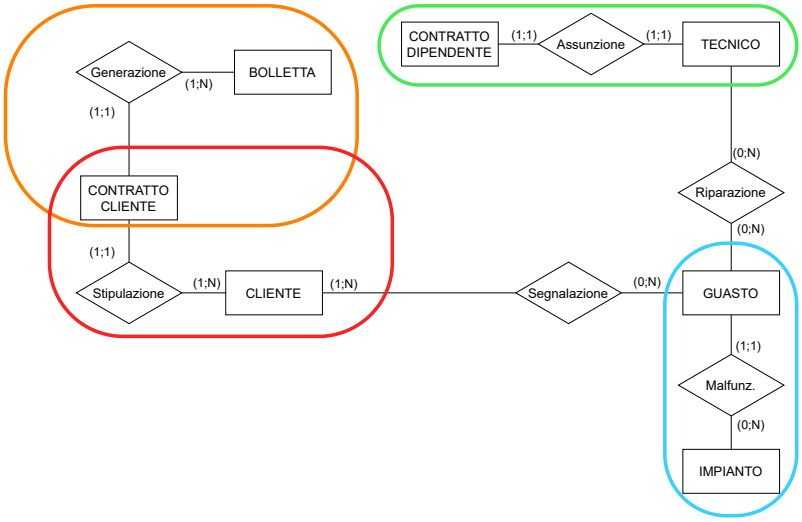


Figura 3.2: Aggregati a livello 1

Nella Figura 3.3 sono invece rappresentati i tre macro-aggregati ottenuti e i relativi collegamenti.

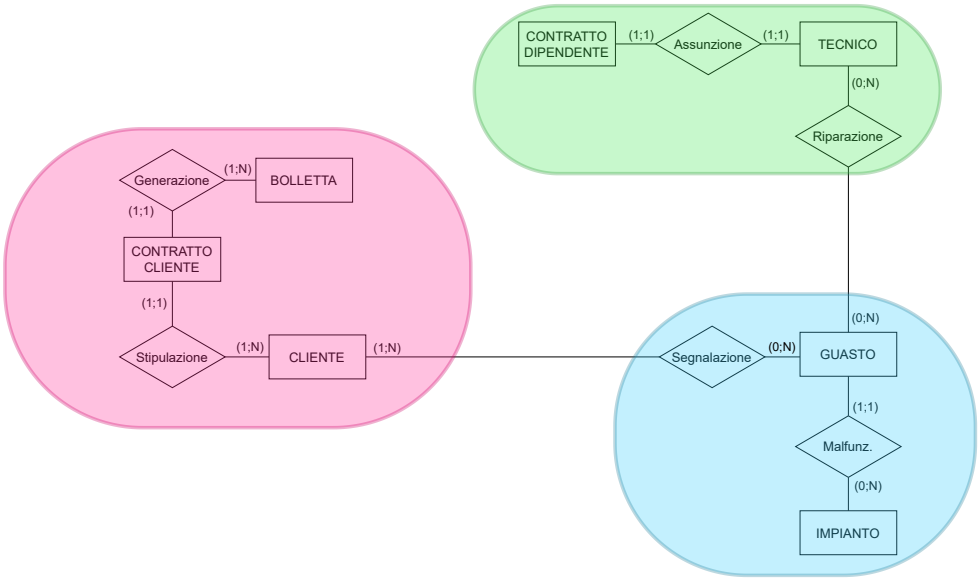


Figura 3.3: Aggregati a livello 2

Rappresentazione degli aggregati

Nei paragrafi precedenti abbiamo definito schematicamente gli aggregati, sinteticamente, a seguito delle analisi svolte antecedentemente, possiamo concludere che la soluzione ottimale si ottiene considerando:

1° aggregato: costituito da CONTRATTO CLIENTE, che al suo interno contiene CLIENTE, BOLLETTA e OFFERTA;

2° aggregato: costituito da TECNICO, che al suo interno contiene CONTRATTO DIPENDENTE.

3° aggregato: costituito da IMPIANTO, che al suo interno contiene GUASTO.

Il modello di rappresentazione che adotteremo per rappresentare i nostri aggregati è il modello NoAM (NoSQL Abstract Data Model), e per la forma delle entries si è optato per la strategia di rappresentazione ETF (Entry Per Top Level Field), arrivando ad avere un'entry per ogni attributo della radice ed un'entry complessa per i vari oggetti annidati. Di seguito vengono riportati i risultati di tale fase di progettazione.

Rappresentazione IMPIANTO-GUASTO

Tabella 3.8: Rappresentazione NoAM dell'aggregato impianto-guasto

Codice: 1	Stato: 'attivo'
	Tipo: 'Fibra Ottica'

	Guasti:[{
	Codice: 4,
	DataApertura: 2018-02-03T15:03:56.000+00:00,
	DataChiusura: 2018-02-03T17:35:46.000+00:00,
	GradoPericolo: 'Alto',
	Segnalazioni: [{
	Cliente: 1,
	RecapitoCl:'3795327112',
	Descrizione: 'Nessun segnale'
	}]
	}]

Codice: 9	Stato: 'attivo'
	Tipo: 'Piastra utente'

	Guasti:[{
	Codice: 9,
	DataApertura: 2020-02-23T09:14:07.000+00:00,
	DataChiusura: 2020-02-23T17:35:40.000+00:00,
	GradoPericolo: 'Alto',
	Segnalazioni: [{
	Cliente: 8,
	RecapitoCl:'3452784554',
	Descrizione: 'Segnale telefonico assente, avvistamento incendio nei dintorni'
	},{
	Cliente: 3,
	RecapitoCl:'3493845271',
	Descrizione: 'Incendio'
	}]
	}]

Codice: 10	Stato: 'attivo'
	Tipo: 'Fibra Ottica'

	Guasti:[{
	Codice: 9,
	DataApertura: 2020-07-01T08:12:45.000+00:00,
	DataChiusura: 2020-07-02T10:13:43.000+00:00,
	GradoPericolo: 'Basso',
	Segnalazioni: [{
	Cliente: 13,
	RecapitoCl:'3229384371',
	Descrizione: 'Segnale internet debole'
	}]
	},{
	Codice: 18,
	DataApertura: 2020-12-30T18:30:35.000+00:00,
	DataChiusura: 2021-01-02T09:05:23.000+00:00,
	GradoPericolo: 'Medio',
	Segnalazioni: [{
	Cliente: 6,
	RecapitoCl:'3795267883',
	Descrizione: 'segnale telefonico assente'
	}]
	}]

Codice: 14	Stato: 'attivo'
	Tipo: 'Linea Rame'
	Guasti: []

Il pattern di accesso per l'aggregato impianto è il seguente:

Impianto → *Guasto*

Rappresentazione TECNICO-CONTRATTO DIPENDENTE

Tabella 3.9: Rappresentazione NoAM dell'aggregato tecnico-contratto dipendente

Codice: 1	CF: 'SSSNLM70C03G273K'
	Nome: 'Anselmo'
	Cognome: 'Sassi'
	DataNasc: 1970-03-30
	Skill: 'base'
	Via: 'Via Leopardi'
	NCivico: 30
	Cap: 23098
	Citta: Rovereto
	NguastiRip: 5
	Contratto: {
	Codice: 'CD00000001',
	Data: 2000-07-13 00:00:00,
	Tipo: 'indeterminato',
	Durata: NULL,
	Stato: 'attivo'
	}
	Riparazioni: [{
	CodGuasto: 3,
	Stato: 'effettuata'
	},
	{
	CodGuasto: 4,
	Stato: 'effettuata'
	},
	{
	CodGuasto: 6,
	Stato: 'effettuata'
	},
	...
	]

Codice: 12	CF: 'DPTNNN89L48G273O' Nome: 'Antonina' Cognome: 'DiPietro' DataNasc: 1989-07-08 Skill: 'centrale telefonica' Via: 'Via Pirandello' NCivico: 65

	Cap: 27398
	Citta: Palermo
	NguastiRip: 0

	Contratto: {
	Codice: 'CD00000012',
	Data: 2020-10-23 00:00:00,
	Tipo: 'determinato',
	Durata: 12,
	Stato: 'attivo'
	}

	Riparazioni: [ ]

I pattern di accesso per l'aggregato tecnico sono i seguenti:
Tecnico → *Contratto dipendente*
Tecnico → *Riparazioni*

Rappresentazione CONTRATTO-CLIENTE-BOLLETTA

Tabella 3.10: Rappresentazione NoAM dell'aggregato contratto-cliente-bolletta

Codice: CC00000001	Data: 2017-12-11 00:00:00
	FineVincolo: 2019-12-11 00:00:00
	MetodoPag: 'c/c'
	TipoCollegamento: 'FFTC'
	CodiceMigrazione: 'ACX9583103948271A'
	ViaForn: 'Via Romani'
	NCivicoForn: 15
	CAPForn: 20019
	CittaForn: Milano
	NomeOff: 'Absolute VDSL'
	CodModem: 'HG3452LKJ12'
	Stato: 'attivo'

	Cliente: {
	Codice: 1,
	Fascia: 'Residenziale',
	ViaFatt: 'Via Romani',
	NCivicoFatt: 15,
	CAPFatt: 20019,
	CittaFatt: 'Milano',
	Tipo: 'privato',
	PIVAoCF: 'RSSMRA80A01F205X',
	Nome: 'Mario',
	Cognome: 'Rossi',
	DataNasc: 1980-01-01,
	RagSoc: NULL,
	NomeAz: NULL,
	NcostrattiAss: 1,
	email: [
	'm.rossi01@outlook.it',
	'm.rossi80@virgilio.it',
	'mario.rossi@gmail.com'
	]
	}

	Bollette: [{
	NFattura: 1,
	Data: 2018-02-12 00:00:00,
	Scadenza: 2018-02-22,
	Periodo: '11/01/2018-11/02/2018',
	StatoPag: 'effettuato',
	},
	{
	NFattura: 2,
	Data: 2018-03-12 00:00:00,
	Scadenza: 2018-03-22,
	Periodo: '11/02/2018-11/03/2018',
	StatoPag: 'effettuato',
	}]

	Offerta: {
	Nome: 'Absolute VDSL',
	CostoMens: 19.90,
	CostoAtt: 25.90
	}

Codice: CC0000015	Data: 2019-02-11 00:00:00
	FineVincolo: 2022-02-11 00:00:00
	MetodoPag: 'c/c'
	TipoCollegamento: 'FFTH'
	CodiceMigrazione: 'DTT9563829154437A'
	ViaForn: 'Via Secondi'
	NCivicoForn: 98
	CAPForn: 67801

CittaForn: Mantova
NomeOff: 'Piu Fibra'
CodModem: 'GT4738AZU47'
Stato: 'attivo'

Cliente: {
Codice: 15,
Fascia: 'Alta',
ViaFatt: 'Via Secondi',
NCivicoFatt: 98,
CAPFatt: 67801,
CittaFatt: 'Mantova',
Tipo: 'azienda',
PIVAoCF: '44324590811',
Nome: NULL,
Cognome: NULL,
DataNasc: NULL,
RagSoc: 's.p.a.',
NomeAz: 'Risparmio Cose s.p.a.',
NcostrattiAss: 1,
email: [
'risparmiocose@risparmio.it',
'risparmiocose@segreteriamministrativa.it'
]
}

Bollette: [{
NFattura: 26,
Data: 2020-12-09 00:00:00,
Scadenza: 2020-09-21,
Periodo: '11/08/2020-11/09/2020',
StatoPag: 'effettuato',
},
{
NFattura: 27,
Data: 2020-12-12 00:00:00,
Scadenza: 2020-12-21,
Periodo: '11/11/2020-11/12/2020',
StatoPag: 'effettuato',
}]

Offerta: {
Nome: 'Piu Fibra',
CostoMens: 24.99,
CostoAtt: 27.90
}

I pattern di accesso per l'aggregato contratto cliente sono i seguenti:
Contratto cliente → *Cliente* → *Email*

Contratto cliente → Bolletta
Contratto cliente → Offerta

EAO o ETF

L'utilizzo della strategia EAO (Entry per Aggregate Object) contrasterebbe con lo scopo di ridurre le computational performance, in quanto per accedere ad un singolo attributo si dovrebbe caricare l'intero blocco che rappresenta l'oggetto e succesivamente estrarre la parte dei dati a noi necessaria. Si riconferma quindi la scelta della strategia ETF.

Index

Si sottolinea che MongoDB, sia tramite Shell che tramite GUI Compass, permette la creazione di indici. MongoDB crea di default un indice associato al campo _id, ovvero la chiave di ciascun oggetto, questo campo è creato automaticamente (in quanto previsto di default) quando vengono inseriti nuovi dati. La sua funzione è quella di identificare univocamente gli oggetti.

Nel nostro progetto si è fatto si che il campo _id corrispondesse ai campi che noi avevamo pensato essere gli identificatori univoci dei vari oggetti.

Partitioning

Ovviamente, nel nostro progetto, si è anche valutata l'opzione del partizionamento orizzontale per ottimizzare le prestazioni delle operazioni che accedono frequentemente a un sottoinsieme specifico di dati. Queste operazioni lavorano spesso su una porzione comune dei dati, soprattutto quando si tratta di storicizzazioni, in cui i dati odierni vengono mantenuti insieme a quelli storici. Il partizionamento avrebbe consentito di mantenere separatamente i dati recenti per migliorare l'accesso alle query più frequenti.
Tuttavia, alla luce delle elevate prestazioni raggiunte senza introdurre il partitioning, abbiamo deciso di non procedere con questa operazione nel nostro progetto. Questa scelta è stata motivata anche dalla volontà di mantenere la trattazione del presente elaborato meno complessa.

Traduzione

Infine, in questo capitolo tratteremo l'ultima fase, ovvero quella di traduzione dal modello NoAM al modello NoSQL. La traduzione è risultata alquanto semplice, una volta definiti i modelli NoAM.
Si è optato per il modello document store in cui il blocco radice è suddiviso in un insieme di coppie chiave-valore che rappresentano i singoli elementi, e nello specifico si è deciso di utilizzare MongoDB. I risultati di questa fase sono quindi dei JSON, uno per ogni aggregato.

Codice 3.1: Implementazione in JSON dell'aggregato impianto-guasto

```
1 {
2   "_id": 1,
3   "Codice": 1,
```

```
  "Stato": "attivo",
  "Tipo": "Fibra Ottica",
  "Guasti": [
    {
      "Codice": 4,
      "DataApertura": {
        "$date": "2018-02-03T15:03:56.000Z"
      },
      "DataChiusura": {
        "$date": "2018-02-03T17:35:46.000Z"
      },
      "Segnalazioni": [
        {
          "RecapitoCl": "3795327112",
          "Descrizione": "Nessun segnale",
          "CodCliente": 1
        }
      ],
      "GradoPericolo": "Alto"
    }
  ]
}
```

Codice 3.2: Implementazione in JSON dell'aggregato contratto dipendente - tecnico

```
{
  "_id": 5,
  "Codice": 5,
  "CF": "MRCDZN62C63C351K",
  "Nome": "Domiziana",
  "Cognome": "Marchesi",
  "DataNasc": {
    "$date": {
      "$numberLong": "-245466000000"
    }
  },
  "Skill": "base",
  "Via": "via Orante",
  "Ncivico": 26,
  "Cap": 16537,
  "Citta": "Catania",
  "NguastiRip": 3,
  "Contratto": {
    "Codice": "CD00000005",
    "Data": {
      "$date": "2018-06-10T00:00:00.000Z"
    },
    "Tipo": "determinato",
    "Durata": 30,
    "Stato": "attivo"
  },
  "Riparazioni": [
    {
      "CodGuasto": 5,
      "Stato": "effettuata"
```

```
    },
    {
      "CodGuasto": 8,
      "Stato": "effettuata"
    },
    {
      "CodGuasto": 20,
      "Stato": "effettuata"
    }
  ]
}
```

Codice 3.3: Implementazione in JSON dell'aggregato cotratto cliente- cliente - bolletta - offerta

```
{
  "_id": "CC00000005",
  "Codice": "CC00000005",
  "Data": {
    "$date": "2020-11-22T00:00:00.000Z"
  },
  "FineVincolo": {
    "$date": "2022-11-22T00:00:00.000Z"
  },
  "MetodoPag": "c/c",
  "TipoCollegamento": "FTTH",
  "CodiceMigrazione": "JHY8675935183921E",
  "ViaForn": "via Massimi",
  "NcivForn": 20,
  "CAPForn": 60024,
  "CittaForn": "Milano Marittima",
  "NomeOff": "Ultrainternet Fibra",
  "CodModem": "KJ5032WSA18",
  "Stato": "attivo",
  "Cliente": {
    "Codice": 4,
    "Fascia": "Residenziale",
    "ViaFatt": "via Leotti",
    "NcivicoFatt": 21,
    "CAPFatt": 98754,
    "CittaFatt": "Ancona",
    "Tipo": "privato",
    "PIVAoCF": "NGLGLN93S46A271Q",
    "Nome": "Giuliana",
    "Cognome": "Angeli",
    "DataNasc": {
      "$date": "1993-11-05T23:00:00.000Z"
    },
    "RagSoc": null,
    "NomeAz": null,
    "NcontrattiAss": 2
  },
  "Bollette": [
    {
      "Nfattura": 10,
      "Data": {
```

```
        "$date": "2020-12-23T00:00:00.000Z"
      },
      "Scadenza": {
        "$date": "2021-01-01T23:00:00.000Z"
      },
      "Periodo": "22/11/2020-22/12/2020",
      "StatoPag": "effettuato"
    }
  ],
  "Offerta": {
    "Nome": "Ultrainternet Fibra",
    "CostoMens": {
      "$numberDecimal": "27.99"
    },
    "CostoAtt": {
      "$numberDecimal": "25.90"
    }
  }
}
```

Capitolo 4

Implementazione

A seguito della fase di progettazione, si è proceduto nell'implementazione del database NoSQL.

Migrazione dei dati

Essendo il nostro progetto basato su un database già esistente, ma in formato SQL, anche i dati erano già presenti, quindi grazie all'utilizzo del tool **Studio3T**, nello specifico della sua funzione *'Migration from SQL'* (Figura 4.1), siamo riusciti ad effettuare una corretta migrazione dei dati in delle collection che riflettessero gli aggregati frutto della fase di progettazione.
La migrazione è stata possibile in quanto questo tool, oltre a importare i record di una tabella principale, in una collection, permette anche di importare dati da altre tabelle, nell'aggregato radice, purché venga definito in che modo ciascun record della tabella principale è correlato ai record dell'aggregato radice (Figura 4.2).

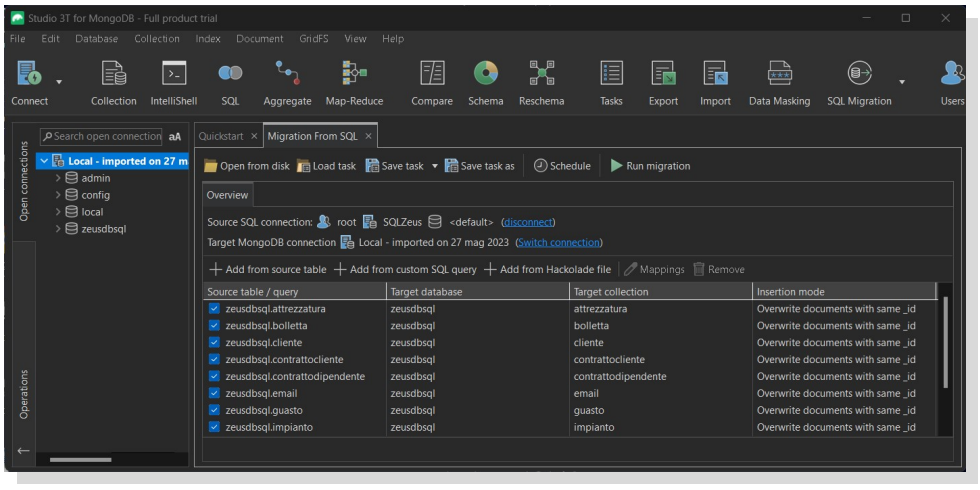


Figura 4.1: Migrazione dei dati

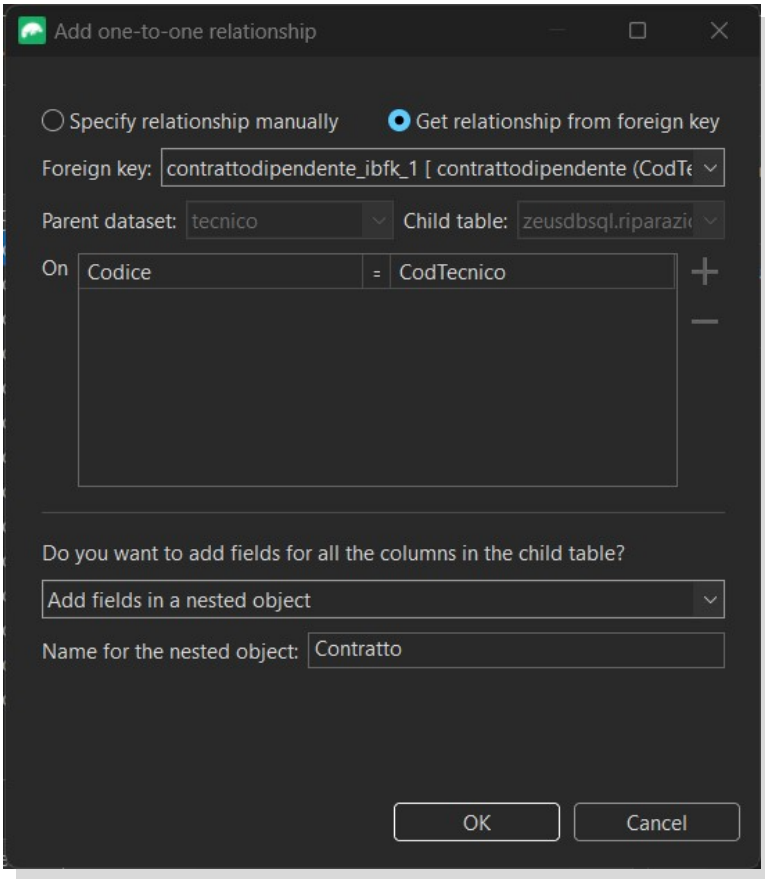


Figura 4.2: Esempio di aggregato, ottenuto importando i dati di Contratto in Tecnico

Query

Abbiamo utilizzato la shell di Mongo inclusa nella GUI Mongo Compass per eseguire tutte le query che riportiamo di seguito.

1. Inserimento nuovo contratto cliente

In questa prima query vediamo l'inserimento di un nuovo contratto.

```
db.contrattocliente.insertOne({"_id": "CC00000016",
"Codice": "CC00000016",
"Data": { "$date": "2023-05-28T00:00:00.000Z" },
"FineVincolo": { "$date": "2026-05-28T00:00:00.000Z" },
"MetodoPag": "c/c",
"TipoCollegamento": "ADSL",
"CodiceMigrazione":  "JYY9560029191466W",
"ViaForn": "via Salocchi",
"NcivForn": 2,
"CAPForn": 63965,
"CittaForn": "Genova",
"NomeOff": "Absolute ADSL",
"CodModem": "WW4744AKK44",
"Stato": "attivo",
"Cliente": {
  "Codice": 15,
  "Fascia": "Alta",
  "ViaFatt": "via Duodeno",
  "NcivicoFatt": 98,
  "CAPFatt": 67801,
  "CittaFatt": "Mantova",
  "Tipo": "azienda",
  "PIVAoCF": "44000590811",
  "Nome": null,
  "Cognome": null,
  "DataNasc": null,
  "RagSoc": "s.p.a.",
  "NomeAz": "Sapori Buoni s.p.a.",
  "NcontrattiAss": 1},
"Bollette": [],
"Offerta": {
  "Nome": "Absolute ADSL",
  "CostoMens": {"$numberDecimal": "29.90"},
  "CostoAtt": {"$numberDecimal": "20.90"}
  }
})

{
  acknowledged: true,
```

```
40    insertedId: 'CC00000016'
41 }
```

Codice 4.1: Query 1 - Inserimento nuovo contratto cliente

Come si può vedere dalla riga 39 della shell riportata sopra, l'inserimento del nuovo contratto è andato a buon fine, per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica, che riportiamo di seguito, insieme al suo risultato.

```
1 db.contrattocliente.find({"_id":"CC00000016"});
```

Codice 4.2: Query 1 - Query di verifica avvenuto inserimento

```
1  {
2    _id: 'CC00000016',
3    Codice: 'CC00000016',
4    Data: 2023-05-28T00:00:00.000Z,
5    FineVincolo: 2026-05-28T00:00:00.000Z,
6    MetodoPag: 'c/c',
7    TipoCollegamento: 'ADSL',
8    CodiceMigrazione: 'JYY9560029191466W',
9    ViaForn: 'via Salocchi',
10   NcivForn: 2,
11   CAPForn: 63965,
12   CittaForn: 'Genova',
13   NomeOff: 'Absolute ADSL',
14   CodModem: 'WW4744AKK44',
15   Stato: 'attivo',
16   Cliente: {
17     Codice: 15,
18     Fascia: 'Alta',
19     ViaFatt: 'via Duodeno',
20     NcivicoFatt: 98,
21     CAPFatt: 67801,
22     CittaFatt: 'Mantova',
23     Tipo: 'azienda',
24     PIVAoCF: '44000590811',
25     Nome: null,
26     Cognome: null,
27     DataNasc: null,
28     RagSoc: 's.p.a.',
29     NomeAz: 'Sapori Buoni s.p.a.',
30     NcontrattiAss: 1
31   },
32   Bollette: [],
33   Offerta: {
34     Nome: 'Absolute ADSL',
35     CostoMens: Decimal128("29.90"),
36     CostoAtt: Decimal128("20.90")
37   }
38 }
```

Codice 4.3: Query 1 - Risultato query di verifica avvenuto inserimento

2. Inserimento nuovo tecnico

In questa query vediamo l'inserimento di un nuovo tecnico.

```
db.tecnico.insertOne(
{
  "_id": 16,
  "Codice": 16,
  "CF": "KTSSHL10S41F842Y",
  "Nome": "Shiuli",
  "Cognome": "Kateshi",
  "DataNasc": {
    "$date": "1997-11-14T22:00:00.000Z"
  },
  "Skill": "base",
  "Via": "via Caroci",
  "Ncivico": 78,
  "Cap": 73048,
  "Citta": "Nardo",
  "NguastiRip": 0,
  "Contratto": {
    "Codice": "CD00000016",
    "Data": {
      "$date": "2023-05-20T00:00:00.000Z"
    },
    "Tipo": "indeterminato",
    "Durata": null,
    "Stato": "attivo"
  },
  "Riparazioni": []
})

{
  acknowledged: true,
  insertedId: 16
}
```

Codice 4.4: Query 2 - Inserimento nuovo tecnico

Come si può vedere dalla riga 30 della shell riportata sopra, l'inserimento del tecnico è andato a buon fine, per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica, che riportiamo di seguito, insieme al suo risultato.

```
db.tecnico.find({"_id":16})
```

Codice 4.5: Query 2 - Query di verifica avvenuto inserimento

```
{
  _id: 16,
  Codice: 16,
  CF: 'KTSSHL10S41F842Y',
```

```
  Nome: 'Shiuli',
  Cognome: 'Kateshi',
  DataNasc: 1997-11-14T22:00:00.000Z,
  Skill: 'base',
  Via: 'via Caroci',
  Ncivico: 78,
  Cap: 73048,
  Citta: 'Nardo',
  NguastiRip: 0,
  Contratto: {
    Codice: 'CD00000016',
    Data: 2023-05-20T00:00:00.000Z,
    Tipo: 'indeterminato',
    Durata: null,
    Stato: 'attivo'
  },
  Riparazioni: []
}
```

Codice 4.6: Query 2 - Risultato query di verifica avvenuto inserimento

3. Inserimento nuovo guasto

In questa query vediamo l'inserimento di un nuovo guasto che è stato segnalato da due clienti differenti.

```
db.impianto.updateOne(
    { "_id": 2 },
    {
        "$set": {
            "Stato": "malfunzionante"
        }
    }
)

db.impianto.updateOne(
    { "_id": 2 },
    {
        "$push": {
            "Guasti": {
                "Codice" : 22,
                "DataApertura" : new Date("2023-06-03T12:01:15.000Z"),
                "DataChiusura" : null,
                "GragoPericolo" : "Medio",
                "Segnalazioni":[{
                    "CodCliente" : 4,
                    "RecapitoCl" : "3549328123",
                    "Descrizione" : "segnale internet assente"
                },
                {
```

```
                    "CodCliente" : 7,
                    "RecapitoCl" : "3754938211",
                    "Descrizione" : "segnale telefonico assente"
                }]
            }
        }
    }
)

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}

db.tecnico.updateMany(
    { "_id": { "$in": [5, 6] } },
    {
        "$push": {
            "Riparazioni":{
                "CodGuasto" : 22,
                "Stato" : "in corso"
            }
        }
    }
)

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
```

Codice 4.7: Query 3 - Inserimento nuovo guasto

Il risultato prodotto da questa query ci dice che l'inserimento del nuovo guasto nella subcollection è avvenuto correttamente. Oltre a questo è stato anche aggiornato lo stato dell'impianto a "malfunzionante" e sono state aggiunte le relative voci nelle subcollection "Riparazioni" della collection "tecnico".

```
db.impianto.aggregate([
  {
    $match: {
      "Guasti.Codice": 22
    }
  },
  {
    $project: {
      Codice: 1,
      Stato: 1,
      Tipo: 1,
```

```
      Guasti: {
        $filter: {
          input: "$Guasti",
          as: "guasto",
          cond: { $eq: ["$$guasto.Codice", 22] }
        }
      }
    }
  }
])

db.tecnico.find({"Riparazioni.CodGuasto": 22})
```

Codice 4.8: Query 3 - Query di verifica avvenuto inserimento

```
{
  _id: 2,
  Codice: 2,
  Stato: 'malfunzionante',
  Tipo: 'Fibra Ottica',
  Guasti: [
    {
      Codice: 22,
      DataApertura: 2023-06-03T12:01:15.000Z,
      DataChiusura: null,
      GragoPericolo: 'Medio',
      Segnalazioni: [
        {
          CodCliente: 4,
          RecapitoCl: '3549328123',
          Descrizione: 'segnale internet assente'
        },
        {
          CodCliente: 7,
          RecapitoCl: '3754938211',
          Descrizione: 'segnale telefonico assente'
        }
      ]
    }
  ]
}
```

Codice 4.9: Query 3 - Verifica avvenuto inserimento guasto

```
{
  _id: 5,
  Codice: 5,
  CF: 'MRCDZN62C63C351K',
  Nome: 'Domiziana',
  Cognome: 'Marchesi',
  DataNasc: 1962-03-22T23:00:00.000Z,
  Skill: 'base',
  Via: 'via Orante',
```

```
  Ncivico: 26,
  Cap: 16537,
  Citta: 'Catania',
  NguastiRip: 3,
  Contratto: {
    Codice: 'CD00000005',
    Data: 2018-06-10T00:00:00.000Z,
    Tipo: 'determinato',
    Durata: 30,
    Stato: 'attivo'
  },
  Riparazioni: [
    {
      CodGuasto: 5,
      Stato: 'effettuata'
    },
    {
      CodGuasto: 8,
      Stato: 'effettuata'
    },
    {
      CodGuasto: 20,
      Stato: 'effettuata'
    },
    {
      CodGuasto: 22,
      Stato: 'in corso'
    }
  ]
}
```

Codice 4.10: Query 3 - Verifica avvenuto inserimento riparazione

```
{
  _id: 6,
  Codice: 6,
  CF: 'DNILCN78D25L219W',
  Nome: 'Luciano',
  Cognome: 'Diana',
  DataNasc: 1978-04-24T23:00:00.000Z,
  Skill: 'base',
  Via: 'via Onesti',
  Ncivico: 57,
  Cap: 62850,
  Citta: 'Torino',
  NguastiRip: 4,
  Contratto: {
    Codice: 'CD00000006',
    Data: 2002-09-11T00:00:00.000Z,
    Tipo: 'indeterminato',
    Durata: null,
    Stato: 'attivo'
  },
  Riparazioni: [
```

```
    {
      CodGuasto: 3,
      Stato: 'effettuata'
    },
    {
      CodGuasto: 4,
      Stato: 'effettuata'
    },
    {
      CodGuasto: 13,
      Stato: 'effettuata'
    },
    {
      CodGuasto: 15,
      Stato: 'effettuata'
    },
    {
      CodGuasto: 20,
      Stato: 'effettuata'
    },
    {
      CodGuasto: 22,
      Stato: 'in corso'
    }
  ]
}
```

Codice 4.11: Query 3 - Verifica avvenuto inserimento riparazione

Se un nuovo cliente effettua una segnalazione ma il guasto è già presente all'interno del database, verrà solamente aggiunta la nuova segnalazione.

```

db.impianto.updateOne(
  { "Codice": 2, "Guasti.Codice": 22 },
  { $push: { "Guasti.$.Segnalazioni": {
    "CodCliente": 9,
    "RecapitoCl": "3329845362",
    "Descrizione": "Non funziona internet"
  }}}
)

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Codice 4.12: Query 3 - Inserimento eventuale nuova segnalazione dello stesso guasto

```
{
  _id: 2,
  Codice: 2,
  Stato: 'malfunzionante',
  Tipo: 'Fibra Ottica',
  Guasti: [
    {
      Codice: 22,
      DataApertura: 2023-06-03T12:01:15.000Z,
      DataChiusura: null,
      GragoPericolo: 'Medio',
      Segnalazioni: [
        {
          CodCliente: 4,
          RecapitoCl: '3549328123',
          Descrizione: 'segnale internet assente'
        },
        {
          CodCliente: 7,
          RecapitoCl: '3754938211',
          Descrizione: 'segnale telefonico assente'
        },
        {
          CodCliente: 9,
          RecapitoCl: '3329845362',
          Descrizione: 'Non funziona internet'
        }
      ]
    }
  ]
}
```

Codice 4.13: Query 3 - Verifica inserimento nuova segnalazione

4. Inserimento nuova bolletta linea fissa

In questa query vediamo l'inserimento di una nuova bolletta relativa alla linea fissa, legata al contratto CC00000006.

```
db.contrattocliente.updateOne(
  {
    "_id": "CC00000006",
    "Stato": "attivo"
  },
  {
    "$push": {
      "Bollette": {
        "Nfattura": 59,
        "Data": new Date("2023-05-10T00:00:00.000Z"),
```

```
11          "Scadenza": new Date("2023-05-19T23:00:00.000Z"),
12          "Periodo": "10/04/2023-10/05/2023",
13          "StatoPag": "effettuato"
14        }
15      }
16    }
17  )
18
19  {
20    acknowledged: true,
21    insertedId: null,
22    matchedCount: 1,
23    modifiedCount: 1,
24    upsertedCount: 0
25  }
```

Codice 4.14: Query 4 - Inserimento nuova bolletta linea fissa

Come si può vedere dalla riga 20 della shell riportata sopra, l'inserimento della nuova bolletta è andato a buon fine. Inoltre come si vede nella riga 4, è stata aggiunta una condizione che permette di inserire una nuova bolletta solo se lo stato del contratto è attivo, in quanto nel caso fosse cessato non avrebbe senso dare la possibilità di aggiungere una nuova bolletta relativa ad esso. Nello snippet di Codice sottostante, viene mostrato il caso in cui si prova ad aggiungere una nuova bolletta per un contratto cessato.

```
1   db.contrattocliente.updateOne(
2     {
3       "_id": "CC00000011",
4       "Stato": "attivo"
5     },
6     {
7       "$push": {
8         "Bollette": {
9           "Nfattura": 123,
10          "Data": new Date("2023-06-10T00:00:00.000Z"),
11          "Scadenza": new Date("2023-06-19T23:00:00.000Z"),
12          "Periodo": "10/05/2023-10/06/2023",
13          "StatoPag": "non effettuato"
14        }
15      }
16    }
17  )
18
19  {
20    acknowledged: true,
21    insertedId: null,
22    matchedCount: 0,
23    modifiedCount: 0,
24    upsertedCount: 0
25  }
```

Codice 4.15: Query 4 - Inserimento nuova bolletta linea fissa, su contratto cessato

Come si può vedere dalla riga 20 della shell sopra riportata, l'inserimento non è avvenuto in quanto il contratto cliente è cessato (non rispetta quindi la seconda condizione *"Stato": "attivo"*). Successivamente,

per aver un maggior grado di sicurezza abbiamo eseguito due query di verifica per vedere se l'inserimento corretto era avvenuto e quello errato no. Il risultato è il seguente:

```
db.contrattocliente.find(
  {_id: 'CC00000006'},
  {
    "Bollette": {
      "$elemMatch": {
        "Nfattura": 59
      }
    }
  }
)
```

Codice 4.16: Query 4 - Query di verifica avvenuto inserimento della bolletta

```
{
  _id: 'CC00000006',
  Bollette: [
    {
      Nfattura: 59,
      Data: 2023-05-10T00:00:00.000Z,
      Scadenza: 2023-05-19T23:00:00.000Z,
      Periodo: '10/04/2023-10/05/2023',
      StatoPag: 'effettuato'
    }
  ]
}
```

Codice 4.17: Query 4 - Risultato query di verifica avvenuto inserimento della bolletta

```
db.contrattocliente.find(
  {_id: 'CC00000011'},
  {
    "Bollette": {
      "$elemMatch": {
        "Nfattura": 123
      }
    }
  }
)
```

Codice 4.18: Query 4 - Query di verifica non avvenuto inserimento della bolletta

```
{
  _id: 'CC00000011'
}
```

Codice 4.19: Query 4 - Risultato query di verifica non avvenuto inserimento della bolletta

5. Consultazione guasti, chiusi nella giornata odierna, visualizzando la durata di riparazione del guasto, ordinati per fascia cliente e data di riparazione

In questa query vediamo la lista di tutti i guasti che sono stati chiusi nella giornata odierna, visualizzando la durata di riparazione del guasto, ordinandoli in base alla fascia del cliente e alla data di riparazione.

```
db.impianto.aggregate([
  {
    $unwind: "$Guasti"
  },
  {
    $unwind: "$Guasti.Segnalazioni"
  },
  {
    $lookup: {
      from: "contrattocliente",
      localField: "Guasti.Segnalazioni.CodCliente",
      foreignField: "Cliente.Codice",
      as: "cliente"
    }
  },
  {
    $unwind: "$cliente"
  },
  {
    $match: {
      $expr: {
        $eq: [
            {$dateToString: { format: "Y-m-d", date:"$Guasti.DataChiusura"}},
               {$dateToString:  { format:  "Y-m-d",date: new Date()}}
        ]
      }
    }
  },
  {
    $addFields: {
      DurataRiparazioneOre: {
        $round: [
          {
            $divide: [
              {
              $subtract: ["$Guasti.DataChiusura", "$Guasti.DataApertura"]
              },
              3600000 //Fattore di conversione da millisecondi a ore
            ]
          },
          0 //Arrotondamento con 0 decimali
        ]
      }
```

```
    }
  },
  {
    $addFields: {
      DurataRiparazioneGiorni: {
        $round: [
          {
            $divide: [
              { $subtract: ["$Guasti.DataChiusura",
                    "$Guasti.DataApertura"] },
               86400000//Fattore di conversione da millisecondi a giorni
            ]
          },
          0 //Arrotondamento con 0 decimali
        ]
      }
    }
  },

  {
    $sort: {
      "cliente.Cliente.Fascia": 1,
      DurataRiparazioneOre: 1
    }
  },
  {
    $group: {
      _id: "$Guasti.Segnalazioni.CodCliente",
      DurataRiparazioneOre: { $first: "$DurataRiparazioneOre" },
      DurataRiparazioneGiorni: { $first: "$DurataRiparazioneGiorni" },
      cliente: { $first: "$cliente" },
      Guasti: { $first: "$Guasti" }
    }
  },
  {
    $project: {
       _id: 0,
      CodiceGuasto: "$Guasti.Codice",
      DurataRiparazioneOre: 1,
      DurataRiparazioneGiorni: 1,
      FasciaCliente:"$cliente.Cliente.Fascia",
      CodiceCliente:"$Guasti.Segnalazioni.CodCliente",
      DataChiusura:"$Guasti.DataChiusura",
      DataApertura:"$Guasti.DataApertura",
    }
  }
])
```

Codice 4.20: Query 5 - Consultazione guasti

Il risultato della query sopra riportata e mostrato di seguito:

```
{
  DurataRiparazioneOre: 64,
  DurataRiparazioneGiorni: 3,
```

```
4     CodiceGuasto: 21,
5     FasciaCliente: 'Residenziale',
6     CodiceCliente: 6,
7     DataChiusura: 2023-06-03T00:00:00.000Z,
8     DataApertura: 2023-05-31T08:15:31.000Z
9   }
```

Codice 4.21: Query 5 - Risultato della query di consultazione guasti

6. Modifica contratto cliente

In questa query vediamo, la modifica del contratto di un cliente. Le modifiche possibili riguardano il metodo di pagamento, l'indirizzo di fornitura o l'offerta sottoscritta.

```
1  var newOfferta = "Ultra ADSL"; //Necessaria solo per la modifica dell'offerta
2  db.contrattocliente.updateOne(
3    { "_id": "CC00000013" },
4    { "$set": {
5        //Parte per la modifica del metodo di pagamento
6        "MetodoPag": "c/c",
7        //Parte per la modifica dell'indirizzo di fornitura
8        "ViaForn": "via verga",
9        "NcivForn": 789,
10       "CAPForn": 00042,
11       "CittaForn": "Roma",
12       //Parte per la modifica dell'offerta
13       "Offerta": {
14         "Nome": newOfferta,
15         "CostoMens": { "$numberDecimal": "15.99" },
16         "CostoAtt": { "$numberDecimal": "18.90" }
17       },
18       "NomeOff": newOfferta,
19     }
20   }
21 )
22
23 {
24   acknowledged: true,
25   insertedId: null,
26   matchedCount: 1,
27   modifiedCount: 1,
28   upsertedCount: 0
29 }
```

Codice 4.22: Query 6 - Modifica contratto cliente

Come si può vedere dalla riga 24 della shell riportata sopra, la modifica del contratto del cliente è andata a buon fine. Nella query sopra riportata si sono effettuate tutte e tre le modifiche possibili contemporaneamente, ma è possibile anche effettuare solo alcune delle modifiche rimuovendo le parti

relative ai campi che non si vogliono modificare. Per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica, per vedere che i campi fossero appunto aggiornati, riportiamo di seguito sia la query che il suo risultato.

```
db.contrattocliente.find(
  { "_id": "CC00000013" },
  {
    "Cliente": 0,
    "Bollette": 0
  }
)
```

Codice 4.23: Query 6 - Query di verifica avvenuta modifica del contratto cliente

Come si può vedere nella shell riportata sopra, nella riga 4 e 5 sono stati messi a 0 sia *"Cliente"* che *"Bollette"*, in quanto per verificare che siano stati modificati i dati relativi al contratto non ci interessano i campi relativi al cliente e alle bollette.

```
{
  _id: 'CC00000013',
  Codice: 'CC00000013',
  Data: 2020-04-15T00:00:00.000Z,
  FineVincolo: 2020-04-15T00:00:00.000Z,
  MetodoPag: 'c/c',
  TipoCollegamento: 'ADSL',
  CodiceMigrazione: 'EFR9483617302947K',
  ViaForn: 'via verga',
  NcivForn: 789,
  CAPForn: 34,
  CittaForn: 'Roma',
  NomeOff: 'Ultra ADSL',
  CodModem: 'PT5743QIU02',
  Stato: 'attivo',
  Offerta: {
    Nome: 'Ultra ADSL',
    CostoMens: Decimal128("15.99"),
    CostoAtt: Decimal128("18.90")
  }
}
```

Codice 4.24: Query 6 - Risultato query di verifica avvenuta modifica del contratto cliente

7. Modifica contratto tecnico

In questa query vediamo, la modifica del contratto di un tecnico. Le modifiche possibili riguardano il tipo di contratto e la durata.

```
1  var tipoContr = "indeterminato";
2  var durataContr; //Va specificata solo per i contratti a tempo determinato
3
4  db.tecnico.updateOne(
5    { "Contratto.Codice": "CD00000004" },
6    {
7      "$set": {
8        "Tipo": tipoContr,
9        "Durata": tipoContr === "indeterminato" ? null : durataContr //La
             durata viene settata a null se il contratto è a tempo indeterminato altrimenti
             gli si da il valore specificato nella variabile sopra
10     }
11   }
12 )
13
14 {
15   acknowledged: true,
16   insertedId: null,
17   matchedCount: 1,
18   modifiedCount: 1,
19   upsertedCount: 0
20 }
```

Codice 4.25: Query 7 - Modifica contratto tecnico

Come si può vedere dalla riga 15 della shell riportata sopra, la modifica del contratto del tecnico è andata a buon fine. Nella query sopra riportata si è fatto in modo che quando il contratto è a tempo indeterminato, se anche per errore viene passato un valore alla variabile riguardante la durata del contratto, essa viene ignorata. Per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica, per vedere che i campi fossero appunto aggiornati, riportiamo di seguito sia la query che il suo risultato.

```
1  db.tecnico.find(
2    { "Contratto.Codice": "CD00000004" },
3    { "Riparazioni": 0}
4  )
```

Codice 4.26: Query 7 - Query di verifica avvenuta modifica del contratto tecnico

Come si può vedere nella shell riportata sopra, nella riga 3 è stato messo a 0 il campo *"Riparazioni"*, in quanto per verificare che siano stati modificati i dati relativi al contratto del tecnico non ci interessano le riparazioni da esso effettuate.

```
1  {
2    _id: 4,
3    Codice: 4,
```

```
  CF: 'CSTLBN85A28A944T',
  Nome: 'Albano',
  Cognome: 'Castellani',
  DataNasc: 1985-01-27T23:00:00.000Z,
  Skill: 'base',
  Via: 'via Giusti',
  Ncivico: 56,
  Cap: 51235,
  Citta: 'Torino',
  NguastiRip: 1,
  Contratto: {
    Codice: 'CD00000004',
    Data: 2020-05-27T00:00:00.000Z,
    Tipo: 'indeterminato',
    Durata: null,
    Stato: 'attivo'
  }
}
```

Codice 4.27: Query 7 - Risultato query di verifica avvenuta modifica del contratto tecnico

8. Chiusura guasto

La query seguente mostra la chiusura di un guasto a seguito della sua riparazione. Si procederà quindi ad aggiornare la riparazione facendola passare allo stato di "effettuata", si riporterà lo stato dell'impianto ad attivo, verrà aggiornato il numero dei guasti riparati associati ai tecnici ed infine verrà impostata la data di chiusura del guasto.

```
db.impianto.updateOne(
  { Codice: 17 },
  { $set: { Stato: 'attivo' } }
)

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Codice 4.28: Query 8 - Chiusura di un guasto, aggiornamento stato impianto

```
db.impianto.updateOne(
  {
    "Codice": 17,
    "Guasti": {
      "$elemMatch": {
```

```
        "Codice": 21
      }
    }
  },
  {
    $set: { "Guasti.$.DataChiusura": new Date("2023-06-03T00:00:00.000Z") }
  }
)

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Codice 4.29: Query 8 - Chiusura di un guasto, aggiornamento data chiusura

```
db.tecnico.updateMany(
  {
    "Riparazioni": {
      "$elemMatch": {
        "CodGuasto": 21
      }
    }
  },
  {$set: { "Riparazioni.$.Stato": "effettuata" } })

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Codice 4.30: Query 8 - Chiusura di un guasto, aggiornamento stato riparazione

```
db.tecnico.find({"Riparazioni": {
      "$elemMatch": {
        "CodGuasto": 21}
      }}).forEach(function(doc) {
  var count = 0;
  doc.Riparazioni.forEach(function(riparazione) {
    if (riparazione.Stato === "effettuata") {
      count++;}
});
  db.tecnico.updateOne({ "_id": doc._id }, { "$set": { "NguastiRip": count
      } });
});
```

Codice 4.31: Query 8 - Chiusura di un guasto aggiornamento guasti riparati da un tecnico

Come si può vedere dalle shell riportate sopra, la chiusura del guasto è andata a buon fine, per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica, che riportiamo di seguito, insieme al suo risultato.

```
db.tecnico.find(
  {
    "Riparazioni": {
      "$elemMatch": {
        "CodGuasto": 21
      }
    }
  })
```

Codice 4.32: Query 8 - Query di verifica avvenuto aggiornamento delle riparazioni del tecnico e dello stato della riparazione

```
{
  _id: 9,
  Codice: 9,
  CF: 'RCCBTN85A26D969U',
  Nome: 'Ubertino',
  Cognome: 'Ricci',
  DataNasc: 1985-01-25T23:00:00.000Z,
  Skill: 'giuntista',
  Via: 'via Ungaretti',
  Ncivico: 34,
  Cap: 86243,
  Citta: 'Genova',
  NguastiRip: 1,
  Contratto: {
    Codice: 'CD00000009',
    Data: 2020-09-03T00:00:00.000Z,
    Tipo: 'determinato',
    Durata: 9,
    Stato: 'attivo'
  },
  Riparazioni: [
    {
      CodGuasto: 21,
      Stato: 'effettuata'
    }
  ]
}
```

Codice 4.33: Query 8 - Risultato query di verifica avvenuto aggiornamento delle riparazioni del tecnico e dello stato della riparazione

```
db.impianto.find(
  {
    "Codice": 17,
    "Guasti": {
```

```
      "$elemMatch": {
        "Codice": 21
      }
    }
  })
```

Codice 4.34: Query 8 - Query di verifica avvenuto aggiornamento dello stato dell'impianto e della data di chiusura del guasto

```
{
  _id: 17,
  Codice: 17,
  Stato: 'attivo',
  Tipo: 'Fibra Ottica',
  Guasti: [
    {
      Codice: 21,
      DataApertura: 2023-05-31T08:15:31.000Z,
      DataChiusura: 2023-06-03T00:00:00.000Z,
      GradoPericolo: 'medio',
      Segnalazioni: [
        {
          CodCliente: 6,
          RecapitoCl: '3472819284',
          Descrizione: 'Il modem ha una luce rossa da ieri'
        }
      ]
    }
  ]
}
```

Codice 4.35: Query 8 - Risultato query di verifica avvenuto aggiornamento dello stato dell'impianto e della data di chiusura del guasto

9. Cancellazione contratto cliente

Con questa query andremo ad effettuare la cancellazione, dal database, del contratto di un cliente, in base al suo codice.

```
var query = { "Cliente.Codice": 6 };

db.contrattocliente.updateMany(
  query,
  { "$inc": { "Cliente.NcontrattiAss": -1 } }
)

{
  acknowledged: true,
```

```
10    insertedId: null,
11    matchedCount: 2,
12    modifiedCount: 2,
13    upsertedCount: 0
14  }
15
16  db.contrattocliente.deleteOne(
17    {Codice:"CC00000007"}
18    )
19
20    {
21    acknowledged: true,
22    deletedCount: 1
23  }
```

Codice 4.36: Query 9 - Cancellazione contratto cliente

Come si può vedere dalla riga 9 della shell riportata sopra, la cancellazione del contratto cliente è andata a buon fine, per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica dell'avvenuta cancellazione, che riportiamo di seguito.

```
1  db.contrattocliente.find(
2    {Codice:"CC00000007"}
3    )
```

Codice 4.37: Query 9 - Query di verifica avvenuta cancellazione del contratto cliente

Come ci si aspettava la query non produce un risultato in quanto appunto non vi è più nel nella collection contrattocliente un documento con il codice "CC00000007".

10. Cancellazione tecnico

Con questa query andremo ad effettuare la cancellazione di un tecnico, in base al suo codice, e del suo relativo contratto dal database.

```
1  db.tecnico.deleteOne(
2    {Codice:10}
3    )
4
5    {
6    acknowledged: true,
7    deletedCount: 1
8  }
```

Codice 4.38: Query 10 - Cancellazione tecnico

Come si può vedere dalla riga 6 della shell riportata sopra, la cancellazione del tecnico è andata a buon fine, per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica dell'avvenuta cancellazione, che riportiamo di seguito.

```
db.tecnico.find(
  {Codice:10}
  )
```

Codice 4.39: Query 10 - Query di verifica avvenuta cancellazione del tecnico

Come ci si aspettava la query non produce un risultato in quanto appunto non vi è più nel nella collection tecnico un documento con il codice 10.

11. Consultazione dati dei clienti

La query seguente ci permetterà di visualizzare tutti i dati di tutti i clienti e il numero del contratto ad essi associato.

```
db.contrattocliente.aggregate([
  {
    $group: {
      _id: "$Cliente.Codice",
      contratti: { $addToSet: "$Codice" },
      datiCliente: { $first: "$Cliente" }
    }
  },
  {
    $project: {
      contratti: 1,
      datiCliente: 1
    }
  },
  {
    $sort: {
      "_id": 1
    }
  }
])
```

Codice 4.40: Query 11 - Consultazione dati dei clienti

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  _id: 1,
  contratti: [
    'CC00000001'
  ],
  datiCliente: {
    Codice: 1,
    Fascia: 'Residenziale',
    ViaFatt: 'via Romani',
    NcivicoFatt: 15,
```

```
11      CAPFatt: 20019,
12      CittaFatt: 'Milano',
13      Tipo: 'privato',
14      PIVAoCF: 'RSSMRA80A01F205X',
15      Nome: 'Mario',
16      Cognome: 'Rossi',
17      DataNasc: 1979-12-31T23:00:00.000Z,
18      RagSoc: null,
19      NomeAz: null,
20      NcontrattiAss: 1
21    }
22  }
23  {
24    _id: 2,
25    contratti: [
26      'CC00000002',
27      'CC00000003'
28    ],
29    datiCliente: {
30      Codice: 2,
31      Fascia: 'Residenziale',
32      ViaFatt: 'via Battisti',
33      NcivicoFatt: 25,
34      CAPFatt: 61121,
35      CittaFatt: 'Pesaro',
36      Tipo: 'privato',
37      PIVAoCF: 'BNCCSN96R48G479Q',
38      Nome: 'Cassandra',
39      Cognome: 'Bianchi',
40      DataNasc: 1996-10-07T22:00:00.000Z,
41      RagSoc: null,
42      NomeAz: null,
43      NcontrattiAss: 2
44    }
45  }
46  ...
```

Codice 4.41: Query 11 - Risultato della query di consultazione dati dei clienti

Nella riga 46 si vedono tre puntini di sospensione perchè si è deciso di non riportare tutto il risultato per non allungare troppo questa relazione.

12. Consultazione dati dei contratti clienti

La query seguente ci permetterà di visualizzare tutti i dati relativi ai contratti di tutti i clienti.

```
1  db.contrattocliente.aggregate([
2    {
3      $project: {
4        "CodiceCliente": "$Cliente.Codice",
```

```
      "_id": 0,
      "Codice": 1,
      "Data": 1,
      "FineVincolo": 1,
      "MetodoPag": 1,
      "TipoCollegamento": 1,
      "CodiceMigrazione": 1,
      "ViaForn": 1,
      "NcivForn": 1,
      "CAPForn": 1,
      "CittaForn": 1,
      "NomeOff": 1,
      "CodModem": 1,
      "Stato": 1
    }
  }
])
```

Codice 4.42: Query 12 - Consultazione dati dei contratti clienti

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  Codice: 'CC00000001',
  Data: 2017-12-11T00:00:00.000Z,
  FineVincolo: 2019-12-11T00:00:00.000Z,
  MetodoPag: 'c/c',
  TipoCollegamento: 'FTTC',
  CodiceMigrazione: 'ACX9583103948271A',
  ViaForn: 'via Romani',
  NcivForn: 15,
  CAPForn: 20019,
  CittaForn: 'Milano',
  NomeOff: 'Absolute VDSL',
  CodModem: 'HG3452LKJ12',
  Stato: 'attivo',
  CodiceCliente: 1
}
{
  Codice: 'CC00000002',
  Data: 2016-12-11T00:00:00.000Z,
  FineVincolo: 2018-12-11T00:00:00.000Z,
  MetodoPag: 'bollettino',
  TipoCollegamento: 'ADSL',
  CodiceMigrazione: 'XJE9128374650193Q',
  ViaForn: 'via Battisti',
  NcivForn: 25,
  CAPForn: 61121,
  CittaForn: 'Pesaro',
  NomeOff: 'Ultra ADSL',
  CodModem: 'AK2098DLK89',
  Stato: 'cessato',
  CodiceCliente: 2
}
...
```

Codice 4.43: Query 12 - Risultato della query di consultazione dati dei contratti clienti

Nella riga 33 si vedono tre puntini di sospensione perchè si è deciso di non riportare tutto il risultato per non allungare troppo questa relazione.

13. Consultazione dati dei tecnici

La query seguente ci permetterà di visualizzare tutti i dati di tutti i tecnici.

```
1 db.tecnico.find(
2   {},
3   {
4     "Contratto": 0,
5     "Riparazioni": 0
6   }
7 )
```

Codice 4.44: Query 13 - Consultazione dati dei tecnici

Il risultato della query sopra riportata è mostrato di seguito:

```
1  {
2    _id: 1,
3    Codice: 1,
4    CF: 'SSSNLM70C03G273K',
5    Nome: 'Anselmo',
6    Cognome: 'Sassi',
7    DataNasc: 1970-03-29T23:00:00.000Z,
8    Skill: 'base',
9    Via: 'via Leopardi',
10   Ncivico: 30,
11   Cap: 23098,
12   Citta: 'Rovereto',
13   NguastiRip: 5
14 }
15 {
16   _id: 2,
17   Codice: 2,
18   CF: 'BTTHLN77C47D612H',
19   Nome: 'Helen',
20   Cognome: 'Bettini',
21   DataNasc: 1977-03-06T23:00:00.000Z,
22   Skill: 'base',
23   Via: 'via Asiaghi',
24   Ncivico: 23,
25   Cap: 94387,
26   Citta: 'Firenze',
27   NguastiRip: 2
28 }
29 ...
```

Codice 4.45: Query 13 - Risultato della query di consultazione dati dei tecnici

Nella riga 29 si vedono tre puntini di sospensione perchè si è deciso di non riportare tutto il risultato per non allungare troppo questa relazione.

14. Consultazione dati dei contratti dipendenti

La query seguente ci permetterà di visualizzare tutti i dati relativi ai contratti di tutti i dipendenti.

```
1 db.tecnico.aggregate([
2   {
3     $project: {
4         _id:0,
5       "CodiceTecnico": "$Codice",
6       "Contratto": 1
7     }
8   }
9 ])
```

Codice 4.46: Query 14 - Consultazione dati dei contratti dipendenti

Il risultato della query sopra riportata è mostrato di seguito:

```
1  {
2    Contratto: {
3      Codice: 'CD00000001',
4      Data: 2000-07-13T00:00:00.000Z,
5      Tipo: 'indeterminato',
6      Durata: null,
7      Stato: 'attivo'
8    },
9    CodiceTecnico: 1
10 }
11 {
12   Contratto: {
13     Codice: 'CD00000002',
14     Data: 2016-05-27T00:00:00.000Z,
15     Tipo: 'determinato',
16     Durata: 20,
17     Stato: 'cessato'
18   },
19   CodiceTecnico: 2
20 }
21 ...
```

Codice 4.47: Query 14 - Risultato della query di consultazione dati dei contratti dipendenti

Nella riga 21 si vedono tre puntini di sospensione perchè si è deciso di non riportare tutto il risultato per non allungare troppo questa relazione.

15. Consultazione dati della bolletta

La query seguente ci permetterà di visualizzare tutti i dati relativi alla bolletta con il valore di Nfattura pari a quello indicato nella query.

```
db.contrattocliente.find(
  { "Bollette": { $elemMatch: { "Nfattura": 6 } } },
  { "Bollette.$": 1 }
)
```

Codice 4.48: Query 15 - Consultazione dati della bolletta

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  _id: 'CC00000003',
  Bollette: [
    {
      Nfattura: 6,
      Data: 2020-07-03T00:00:00.000Z,
      Scadenza: 2020-07-12T22:00:00.000Z,
      Periodo: '02/06/2020-02/07/2020',
      StatoPag: 'effettuato'
    }
  ]
}
```

Codice 4.49: Query 15 - Risultato della query di consultazione dati della bolletta

16. Statistica acquisizione clienti

La query seguente ci permetterà di visualizzare il numero di clienti acquisiti nel mese specificato dell'anno specificato.

```
var year = 2016;
var month = 12;

db.contrattocliente.aggregate([
  {
    $match: {
      Data: {
        $gte: new Date(year, month - 1, 1),
        $lt: new Date(year, month, 1)
      }
    }
  },
```

```
  {
    $group: {
      _id: {
        Mese: { $month: "$Data" },
        Anno: { $year: "$Data" }
      },
      NumeroClienti: { $sum: 1 }
    }
  },
  {
    $project: {
      Mese: "$_id.Mese",
      Anno: "$_id.Anno",
      NumeroClienti: 1,
      _id: 0
    }
  }
])
```

Codice 4.50: Query 16 - Statistica acquisizione clienti

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  NumeroClienti: 2,
  Mese: 12,
  Anno: 2016
}
```

Codice 4.51: Query 16 - Risultato della query di statistica acquisizione clienti

17. Statistica contratti per tipo collegamento

La query seguente ci permetterà di visualizzare il numero di contratti suddivisi per il tipo di collegamento.

```
db.contrattocliente.aggregate([
  {
    $group: {
      _id: "$TipoCollegamento",
      Numero_Contratti_Affiliati: { $sum: 1 }
    }
  },
  {
    $project: {
      TipoCollegamento: "$_id",
      Numero_Contratti_Affiliati: 1,
      _id: 0
    }
  }
```

```
15 ])
```

Codice 4.52: Query 17 - Statistica contratti per tipo collegamento

Il risultato della query sopra riportata è mostrato di seguito:

```
1  {
2    Numero_Contratti_Affiliati: 5,
3    TipoCollegamento: 'FTTC'
4  }
5  {
6    Numero_Contratti_Affiliati: 3,
7    TipoCollegamento: 'FTTH'
8  }
9  {
10   Numero_Contratti_Affiliati: 8,
11   TipoCollegamento: 'ADSL'
12 }
```

Codice 4.53: Query 17 - Risultato della query di statistica acquisizione clienti

18. Statistica numero guasti riparati per tecnico

La query seguente ci permetterà di visualizzare il numero di guasti riparati da ogni tecnico.

```
1 db.tecnico.find({},
2     { Codice: 1,
3       NguastiRip: 1,
4       _id:0})
```

Codice 4.54: Query 18 - Statistica numero guasti riparati per tecnico

Il risultato della query sopra riportata è mostrato di seguito:

```
1  {
2    Codice: 1,
3    NguastiRip: 5
4  }
5  {
6    Codice: 2,
7    NguastiRip: 2
8  }
9  {
10   Codice: 3,
11   NguastiRip: 3
12 }
13 ...
```

Codice 4.55: Query 18 - Risultato della query di statistica numero guasti riparati per tecnico

Nella riga 13 si vedono tre puntini di sospensione perchè si è deciso di non riportare tutto il risultato per non allungare troppo questa relazione.

19. Statistica numero tecnici per skill

La query seguente ci permetterà di visualizzare il numero di tecnici suddivisi per il tipo di skill possedute.

```
db.tecnico.aggregate([
  {
    $group: {
      _id: "$Skill",
      NTecnici: { $sum: 1 }
    }
  },
  {
    $project: {
      Skill: "$_id",
      NTecnici: 1,
      _id: 0
    }
  }
])
```

Codice 4.56: Query 19 - Statistica numero tecnici per skill

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  NTecnici: 9,
  Skill: 'base'
}
{
  NTecnici: 2,
  Skill: 'giuntista'
}
{
  NTecnici: 4,
  Skill: 'centrale telefonica'
}
```

Codice 4.57: Query 19 - Risultato della query di statistica numero tecnici per skill

20. Statistica guasti di un dato impianto

La query seguente ci permetterà di visualizzare il numero di guasti dell'impianto di cui si passa il codice.

```
db.impianto.aggregate([
  {
    $match: {
      Codice: 10
    }
  },
  {
    $project: {
      Codice: 1,
      Stato: 1,
      Tipo: 1,
      NumeroGuasti: { $size: "$Guasti" }
    }
  }
])
```

Codice 4.58: Query 20 - Statistica guasti di un dato impianto

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  _id: 10,
  Codice: 10,
  Stato: 'attivo',
  Tipo: 'Fibra Ottica',
  NumeroGuasti: 2
}
```

Codice 4.59: Query 20 - Risultato della query di statistica guasti di un dato impianto

21. Statistica numero clienti in base alla fascia

La query seguente ci permetterà di visualizzare il numero di clienti in base alla fascia di appartenenza.

```
db.contrattocliente.aggregate([
  {
    $group: {
      _id: "$Cliente.Fascia",
      NClienti: { $sum: 1 }
    }
  }
```

```
])
```

Codice 4.60: Query 21 - Statistica numero clienti in base alla fascia

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  _id: 'Residenziale',
  NClienti: 7
}
{
  _id: 'Affari',
  NClienti: 5
}
{
  _id: 'Alta',
  NClienti: 3
}
```

Codice 4.61: Query 21 - Risultato della query di statistica numero clienti in base alla fascia

22. Statistica numero contratti per cliente

La query seguente ci permetterà di visualizzare il numero di contratti associati ad ogni cliente, ordinati in ordine crescente in base al codice del cliente.

```
db.contrattocliente.aggregate([
  {
    $group: {
      _id: "$Cliente.Codice",
      ClienteCodice: { $first: "$Cliente.Codice" },
      NcontrattiAss: { $first: "$Cliente.NcontrattiAss" }
    }
  },
  {
    $project: {
      _id: 0,
      ClienteCodice: 1,
      NcontrattiAss: 1
    }
  },
  {
    $sort: {
      ClienteCodice: 1
    }
  }
])
```

Codice 4.62: Query 22 - Statistica numero contratti per cliente

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  ClienteCodice: 1,
  NcontrattiAss: 1
}
{
  ClienteCodice: 2,
  NcontrattiAss: 2
}
{
  ClienteCodice: 4,
  NcontrattiAss: 2
}
...
```

Codice 4.63: Query 22 - Risultato della query di statistica numero contratti per cliente

Nella riga 13 si vedono tre puntini di sospensione perchè si è deciso di non riportare tutto il risultato per non allungare troppo questa relazione.

23. Statistica ragione sociale azienda

La query seguente ci permetterà di visualizzare il numero di aziende suddivise in base alla loro ragione sociale, ovvero di vedere quante delle aziende clienti sono delle s.r.l., delle s.n.c., delle s.a.s. o delle s.p.a. .

```
db.contrattocliente.aggregate([
  {
    $match: {
      "Cliente.RagSoc": { $ne: null }
    }
  },
  {
    $group: {
      _id: "$Cliente.RagSoc",
      NumeroAziende: { $sum: 1 }
    }
  },
  {
    $project: {
      _id: 0,
      RagSoc: "$_id",
      NumeroAziende: 1
    }
  }
])
```

Codice 4.64: Query 23 - Statistica ragione sociale azienda

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  NumeroAziende: 3,
  RagSoc: 's.n.c.'
}
{
  NumeroAziende: 1,
  RagSoc: 's.a.s.'
}
{
  NumeroAziende: 3,
  RagSoc: 's.p.a.'
}
{
  NumeroAziende: 1,
  RagSoc: 's.r.l.'
}
```

Codice 4.65: Query 23 - Risultato della query di statistica ragione sociale azienda

24. Statistica tecnici con contratto a tempo indeterminato

La query seguente ci permetterà di visualizzare il numero di tecnici il cui contratto risulta essere a tempo indeterminato.

```
db.tecnico.aggregate([
   {
    $match: { "Contratto.Tipo":"indeterminato"}
  },
  {
    $group: {
      _id: "$Contratto.Tipo",
      NTecnici: { $sum: 1 }
    }
  },
  {
    $project: {
      TipoContratto: "$_id",
      NTecnici: 1,
      _id: 0
    }
  }
])
```

Codice 4.66: Query 24 - Statistica tecnici con contratto a tempo indeterminato

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  NTecnici: 6,
  TipoContratto: 'indeterminato'
}
```

Codice 4.67: Query 24 - Risultato della query di statistica tecnici con contratto a tempo indeterminato

25. Statistica numero guasti per grado di pericolo

La query seguente ci permetterà di visualizzare il numero dei guasti suddivisi in base al grado di pericolosità.

```
db.impianto.aggregate([
  {
    $unwind: {
      path: "$Guasti",
    },
  },
  {
    $group: {
        _id: "$Guasti.GradoPericolo",
        Nguasti: {
          $sum: 1,
        },
      },
  },
])
```

Codice 4.68: Query 25 - Statistica numero guasti per grado di pericolo

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  _id: 'Alto',
  Nguasti: 3
}
{
  _id: 'Basso',
  Nguasti: 9
}
{
  _id: 'Medio',
  Nguasti: 9
}
```

Codice 4.69: Query 25 - Risultato della query di statistica numero guasti per grado di pericolo

26. Verifica avvenuto pagamento bolletta da parte dei clienti

La query seguente ci permetterà di verificare se il cliente ha effettuato o meno il pagamento della bolletta, restituendo la lista delle bollette pagate con i dati relativi al cliente.

```
db.contrattocliente.aggregate([
      {
        $unwind: {
          path: "$Bollette",
        },
      },
      {
        $match: {
          "Bollette.StatoPag": "effettuato",
        },
      },
      {
        $project: {
          CodiceCliente: "$Cliente.Codice",
          CodiceModem: "$CodModem",
          Nfattura: "$Bollette.Nfattura",
          DataEmissioneFattura: "$Bollette.Data",
          Scadenza: "$Bollette.Scadenza",
          Periodo: "$Bollette.Periodo",
          Importo: "$Offerta.CostoMens",
        },
      },
    ]
)
```

Codice 4.70: Query 26 - Verifica avvenuto pagamento bolletta da parte dei clienti

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  _id: 'CC00000001',
  CodiceCliente: 1,
  CodiceModem: 'HG3452LKJ12',
  Nfattura: 1,
  DataEmissioneFattura: 2018-02-12T00:00:00.000Z,
  Scadenza: 2018-02-21T23:00:00.000Z,
  Periodo: '11/01/2018-11/02/2018',
  Importo: Decimal128("19.90")
}
{
  _id: 'CC00000001',
  CodiceCliente: 1,
  CodiceModem: 'HG3452LKJ12',
  Nfattura: 2,
  DataEmissioneFattura: 2020-12-12T00:00:00.000Z,
```

```
  Scadenza: 2020-12-21T23:00:00.000Z,
  Periodo: '11/11/2020-11/12/2020',
  Importo: Decimal128("19.90")
}
...
{
  _id: 'CC00000003',
  CodiceCliente: 2,
  CodiceModem: 'AZ6057KOJ23',
  Nfattura: 5,
  DataEmissioneFattura: 2020-01-03T00:00:00.000Z,
  Scadenza: 2020-01-12T23:00:00.000Z,
  Periodo: '02/12/2019-02/01/2020',
  Importo: Decimal128("27.99")
}
...
```

Codice 4.71: Query 26 - Risultato della query di verifica avvenuto pagamento bolletta da parte dei clienti

27. Visualizzazione bollette con scadenza la settimana successiva alla data di consultazione

La query seguente ci permetterà di visualizzare tutte le bollette con scadenza la settimana successiva alla data di consultazione.

```
db.contrattocliente.aggregate([
   {
    $unwind: "$Bollette",
  },
  {
    $match: {
      $expr: {
        $and: [
          {
            $gte: [
              "$Bollette.Scadenza",
              new Date(),
            ],
          },
          {
            $lte: [
              "$Bollette.Scadenza",
              {
                $add: [new Date(), 7 * 24 * 60 * 60 * 1000, ],
              },
            ],
          },
```

```
        ],
      },
    },
  },
  {
    $project: {
      _id: 0,
      CodiceCliente: "$Cliente.Codice",
      CodiceContratto: "$_id",
      CodModem: "$CodModem",
      Nfattura: "$Bollette.Nfattura",
      DataEmissione: "$Bollette.Data",
      Scadenza: "$Bollette.Scadenza",
      Periodo: "$Bollette.Periodo",
      Importo: "$Offerta.CostoMens",
    },
  }
])
```

Codice 4.72: Query 27 - Visualizzazione bollette con scadenza la settimana successiva alla data di consultazione

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  CodiceCliente: 2,
  CodiceContratto: 'CC00000003',
  CodModem: 'AZ6057KOJ23',
  Nfattura: 155,
  DataEmissione: 2023-05-31T00:00:00.000Z,
  Scadenza: 2023-06-10T23:00:00.000Z,
  Periodo: '31/04/2023-31/05/2023',
  Importo: Decimal128("27.99")
}
{
  CodiceCliente: 5,
  CodiceContratto: 'CC00000006',
  CodModem: 'LE6784WQO20',
  Nfattura: 196,
  DataEmissione: 2023-05-31T00:00:00.000Z,
  Scadenza: 2023-06-10T23:00:00.000Z,
  Periodo: '31/04/2023-31/05/2023',
  Importo: Decimal128("15.99")
}
```

Codice 4.73: Query 27 - Risultato della query di visualizzazione bollette con scadenza la settimana successiva alla data di consultazione

28. Aggiornamento pagamento

Con questa query andremo ad effettuare l'aggiornamento dello stato di un pagamento, in base al numero di fattura.

```
db.contrattocliente.updateOne(
  {
    "Bollette": {
      "$elemMatch": {
        "Nfattura": 7
      }
    }
  },
  {
    "$set": { "Bollette.$.StatoPag": "effettuato" }
  }
)

{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Codice 4.74: Query 28 - Aggiornamento pagamento

Come si può vedere dalla riga 15 della shell riportata sopra, l'aggiornamento dello stato del pagamento è andato a buon fine, per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica dell'avvenuto aggiornamento, che riportiamo di seguito.

```
db.contrattocliente.find(
  {
    "Bollette": {
      "$elemMatch": {
        "Nfattura": 7
      }
    }
  },
  {
      "Offerta":0,
      "Cliente":0
  }
)
```

Codice 4.75: Query 28 - Query di verifica avvenuto aggiornamento del pagamento

```
{
  _id: 'CC00000003',
```

```
  Codice: 'CC00000003',
  Data: 2019-02-02T00:00:00.000Z,
  FineVincolo: 2019-02-02T00:00:00.000Z,
  MetodoPag: 'c/c',
  TipoCollegamento: 'FTTC',
  CodiceMigrazione: 'AKO0294817503294P',
  ViaForn: 'via Battisti',
  NcivForn: 25,
  CAPForn: 61121,
  CittaForn: 'Pesaro',
  NomeOff: 'Ultrainternet Fibra',
  CodModem: 'AZ6057KOJ23',
  Stato: 'attivo',
  Bollette: [
    {
      Nfattura: 5,
      Data: 2020-01-03T00:00:00.000Z,
      Scadenza: 2020-01-12T23:00:00.000Z,
      Periodo: '02/12/2019-02/01/2020',
      StatoPag: 'effettuato'
    },
    {
      Nfattura: 6,
      Data: 2020-07-03T00:00:00.000Z,
      Scadenza: 2020-07-12T22:00:00.000Z,
      Periodo: '02/06/2020-02/07/2020',
      StatoPag: 'effettuato'
    },
    {
      Nfattura: 7,
      Data: 2021-01-03T00:00:00.000Z,
      Scadenza: 2021-01-12T23:00:00.000Z,
      Periodo: '02/12/2020-02/01/2021',
      StatoPag: 'effettuato'
    }
  ]
}
```

Codice 4.76: Query 28 - Risultato della query di verifica avvenuto aggiornamento del pagamento

29. Consultazione tecnici occupati

La query seguente ci permetterà di visualizzare la lista dei tecnici che al momento sono occupati nella risoluzione del guasto.

```
db.tecnico.aggregate([
  {$unwind: {
    path: "$Riparazioni"
  }},
```

```
  {
    $match: {
      "Riparazioni.Stato": "in corso"
    }
  },
  {
    $group: {
      _id: '$Codice',
      Nome: { $first: '$Nome' },
      Cognome: { $first: '$Cognome' },
      Skill: { $first: '$Skill' },
      quantita: { $sum: 1 }
    }
  }
])
```

Codice 4.77: Query 29 - Consultazione tecnici occupati

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  _id: 5,
  Nome: 'Domiziana',
  Cognome: 'Marchesi',
  Skill: 'base',
  quantita: 1
}
{
  _id: 6,
  Nome: 'Luciano',
  Cognome: 'Diana',
  Skill: 'base',
  quantita: 2
}
{
  _id: 4,
  Nome: 'Albano',
  Cognome: 'Castellani',
  Skill: 'base',
  quantita: 1
}
```

Codice 4.78: Query 29 - Risultato della query di consultazione tecnici occupati

30. Cessazione contratto cliente

Con questa query effettueremo la cessazione di un contratto cliente.

```
1
2  db.contrattocliente.updateOne(
3    { "Codice": "CC00000005" },
4    {
5      "$set": {
6        "Stato": "cessato",
7      }
8    }
9  )
10
11 {
12   acknowledged: true,
13   insertedId: null,
14   matchedCount: 1,
15   modifiedCount: 1,
16   upsertedCount: 0
17 }
```

Codice 4.79: Query 30 - Cessazione contratto cliente

Come si può vedere dalla riga 12 della shell riportata sopra, la cessazione del contratto cliente è andata a buon fine, per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica, che riportiamo di seguito, insieme al suo risultato.

```
1  db.contrattocliente.find(
2    {"Codice": "CC00000005"},
3    {
4      Cliente:0,
5      Bollette:0,
6      Offerta:0
7    })
```

Codice 4.80: Query 30 - Query di verifica avvenuto inserimento

```
1  {
2    _id: 'CC00000005',
3    Codice: 'CC00000005',
4    Data: 2020-11-22T00:00:00.000Z,
5    FineVincolo: 2022-11-22T00:00:00.000Z,
6    MetodoPag: 'c/c',
7    TipoCollegamento: 'FTTH',
8    CodiceMigrazione: 'JHY8675935183921E',
9    ViaForn: 'via Massimi',
10   NcivForn: 20,
11   CAPForn: 60024,
12   CittaForn: 'Milano Marittima',
13   NomeOff: 'Ultrainternet Fibra',
```

```
14    CodModem: 'KJ5032WSA18',
15    Stato: 'cessato'
16  }
```

Codice 4.81: Query 30 - Risultato query di verifica avvenuta cessazione del contratto cliente

31. Cessazione contratto tecnico

Con questa query effettueremo la cessazione di un contratto tecnico.

```
1  db.tecnico.updateOne(
2    { "Contratto.Codice": "CD00000001" },
3    {
4      "$set": {
5        "Contratto.Stato": "cessato",
6      }
7    }
8  )
9
10 {
11   acknowledged: true,
12   insertedId: null,
13   matchedCount: 1,
14   modifiedCount: 1,
15   upsertedCount: 0
16 }
```

Codice 4.82: Query 31 - Cessazione contratto tecnico

Come si può vedere dalla riga 11 della shell riportata sopra, la cessazione del contratto tecnico è andata a buon fine, per aver un maggior grado di sicurezza successivamente abbiamo eseguito una query di verifica, che riportiamo di seguito, insieme al suo risultato.

```
1  db.tecnico.aggregate([
2      {
3      $match: { "Contratto.Codice": "CD00000001"}
4      },
5      {
6      $project: {
7        _id: 0,
8        CodiceContratto: "$Contratto.Codice",
9        StatoContratto: "$Contratto.Stato",
10       }
11   }
12 ])
```

Codice 4.83: Query 31 - Query di verifica avvenuto inserimento

```
{
  CodiceContratto: 'CD00000001',
  StatoContratto: 'cessato'
}
```

Codice 4.84: Query 31 - Risultato query di verifica avvenuta cessazione del contratto tecnico

32. Visualizzazione guasti per l'impianto X

La query seguente ci permetterà di visualizzare la lista di guasti dell'impianto di cui si passa il codice.

```
db.impianto.aggregate([
  {
    $match: {
      Codice: 10
    }
  },
  {
    $project: {
      _id: 0,
      CodiceImpianto: "$Codice",
      Stato: "$Stato",
      Guasti: {
        $map: {
          input: "$Guasti",
          as: "guasto",
          in: {
            CodiceGuasto: "$$guasto.Codice",
            DataApertura: "$$guasto.DataApertura",
            DataChiusura: "$$guasto.DataChiusura",
            GradoPericolo: "$$guasto.GradoPericolo",
          }
        }
      }
    }
  }
])
```

Codice 4.85: Query 32 - Visualizzazione guasti per l'impianto X

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  CodiceImpianto: 10,
  Stato: 'attivo',
  Guasti: [
    {
      CodiceGuasto: 10,
      DataApertura: 2020-07-01T08:12:45.000Z,
```

```
      DataChiusura: 2020-07-02T10:13:43.000Z,
      GradoPericolo: 'Basso'
    },
    {
      CodiceGuasto: 18,
      DataApertura: 2020-12-30T18:30:35.000Z,
      DataChiusura: 2021-01-02T09:05:23.000Z,
      GradoPericolo: 'Medio'
    }
  ]
}
```

Codice 4.86: Query 32 - Risultato della query di visualizzazione guasti per l'impianto X

33. Visualizzazione bollette non pagate da un dato cliente X

Con questa query andremo a visualizzare le bollette non pagate da un dato cliente.

```
db.contrattocliente.aggregate([
    {
    $unwind: {
      path: "$Bollette",
    }
  },
  {
    $match: {
      "Codice": "CC00000009",
      "Bollette.StatoPag": "non effettuato"
    }
  },
  {
    $project: {
      Codice: 1,
      Bollette:1
    }
  }
])
```

Codice 4.87: Query 33 - Visualizzazione bollette non pagate da un dato cliente X

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  _id: 'CC00000009',
  Codice: 'CC00000009',
  Bollette: {
    Nfattura: 15,
```

```
    Data: 2021-01-05T00:00:00.000Z,
    Scadenza: 2021-01-14T23:00:00.000Z,
    Periodo: '04/12/2020-04/01/2021',
    StatoPag: 'non effettuato'
  }
}
```

Codice 4.88: Query 33 - Query che riporta la bolletta non pagata dal cliente X

34. Visualizzazione segnalazioni fatte dal cliente X

La query seguente ci permetterà di visualizzare la lista delle segnalazioni relative ai guasti, effettuate da un cliente di cui si passa il codice.

```
db.impianto.aggregate([
  {
    $unwind: {
      path: "$Guasti",
    }
  },
  {
    $unwind: {
      path: "$Guasti.Segnalazioni",
    }
  },
  {
    $match:{ "Guasti.Segnalazioni.CodCliente": 7}
  },
  {
    $project:
      {
        _id: 0,
        CodiceImpianto: "$Codice",
        CodiceGuasto: "$Guasti.Codice",
        Descrizione: "$Guasti.Segnalazioni.Descrizione",
        GradoPericoloGuasto: "$Guasti.GradoPericolo"
      }
  }
])
```

Codice 4.89: Query 34 - Visualizzazione segnalazioni fatte dal cliente X

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  CodiceImpianto: 2,
  CodiceGuasto: 22,
  Descrizione: 'segnale telefonico assente',
  GradoPericoloGuasto: 'Medio'
```

```
}
{
  CodiceImpianto: 13,
  CodiceGuasto: 11,
  Descrizione: 'Ho problemi con la connessione',
  GradoPericoloGuasto: 'Medio'
}
```

Codice 4.90: Query 34 - Risultato della query di visualizzazione segnalazioni fatte dal cliente X

35. Visualizzazione impianti in riparazione

La query seguente ci permetterà di visualizzare la lista degli impianti che al momento sono in riparazione.

```
db.impianto.find(
  {
    Stato:"malfunzionante"
  },
  {
    _id:0,
    Guasti:0
  })
```

Codice 4.91: Query 35 - Visualizzazione impianti in riparazione

Il risultato della query sopra riportata è mostrato di seguito:

```
{
  Codice: 2,
  Stato: 'malfunzionante',
  Tipo: 'Fibra Ottica'
}
{
  Codice: 12,
  Stato: 'malfunzionante',
  Tipo: 'Fibra Ottica'
}
{
  Codice: 17,
  Stato: 'malfunzionante',
  Tipo: 'Fibra Ottica'
}
```

Codice 4.92: Query 35 - Risultato della query di visualizzazione impianti in riparazione

Capitolo 5

Conclusioni e sviluppi futuri

In conclusione, il lavoro di progettazione e sviluppo di un database NoSQL con MongoDB a partire da un database relazionale ha raggiunto con successo i suoi obiettivi principali. Nello svolgimento del progetto, è stata effettuata una migrazione dei dati dal database relazionale al database NoSQL, implementando le necessarie modifiche al modello di dati per adattarlo alle caratteristiche di MongoDB. Sono state anche sviluppate le funzionalità di lettura, scrittura e interrogazione dei dati utilizzando le potenti funzionalità offerte da MongoDB.
Il principale vantaggio ottenuto dall'utilizzo di un sistema NoSQL è stato la flessibilità offerta dal database. MongoDB ha consentito di modellare i dati in modo più flessibile rispetto al modello relazionale, consentendo di gestire dati non strutturati o semi-strutturati in modo più efficiente, e consentendo un livello di annidamento che nei database relazionali non è possibile avere.
Nonostante il buon livello raggiunto dal progetto, ci sono ancora opportunità per ulteriori sviluppi e miglioramenti. Alcune delle possibili aree di sviluppo futuro includono:

- **Ottimizzazione delle prestazioni**: È possibile continuare a ottimizzare le prestazioni del database NoSQL attraverso l'ottimizzazione delle query, l'uso di indici appropriati e la configurazione dei parametri di MongoDB;
- **Integrazione con altri tool**: MongoDB può essere integrato con altre tecnologie e tool per fornire funzionalità aggiuntive, quali ad esempio, l'integrazione con strumenti di analisi dei dati la quale potrebbe consentire l'estrazione di informazioni significative dai dati archiviati nel database;
- **Altre soluzioni NoSQL**: Oltre a MongoDB, esistono altre soluzioni NoSQL che potrebbero essere esplorate in futuro, si potrebbero valutare altre basi di dati NoSQL come Cassandra, Couchbase o Redis per determinare quale sia la soluzione più adatta alle esigenze specifiche del progetto.