UNIVERSITÀ DEGLI STUDI DI UDINE
Corso di Informatica e Internet of Things, Big Data and Web

Cognome e Nome	Numero Matricola
Mattia Guiotto	147654
Nicola Zerajic De Giorgio	149299
Andrea Comisso	147984

Descrizione delle informazioni da gestire e dei servizi da prevedere

Compito di Basi di dati e sistemi informativi - II

13 dicembre 2001

Esercizio 5:

Si vuole progettare una base di dati di supporto alle attività di un'azienda alimentare di carni.

L'azienda opera su un certo numero di mercati, che possono essere sia Macrozone (Triveneto, Isole) che singole regioni (Lombardia, Piemonte, etc.). Ogni cliente appartiene ad uno specifico mercato. I clienti sono seguiti da agenti, coordinati da capi-area. Le vendite sono fatte su articoli, che sono suddivisi in opportune linee di prodotto. Ogni cliente che intenda comprare della carne effettua un ordine in cui vengono specificati gli articoli che si vogliono acquistare e il loro prezzo. Gli articoli venduti appartengono a opportune partite di animali aquistate da un certo insieme di fornitori. Le partite possono essere di prima, seconda e terza scelta. Ogni partita contiene articoli appartenenti ad un'unica linea di prodotto ed è identificata univocamente da un opportuno codice. Nell'ambito di ciascuna partita, ogni articolo è identificato univocamente da un codice numerico (primo articolo della partita, secondo articolo della partita, etc.). Ad ogni partita è inoltre associato un costo diretto, suddiviso in costo di acquisto, spedizione e stoccaggio. Il costo del singolo articolo si ottiene dividendo il costo della partita cui appartiene per il numero di articoli in essa contenuti.

Si definisca uno schema Entità-Relazioni che descriva il contenuto informativo del sistema, illustrando con chiarezza le eventuali assunzioni fatte. Lo schema dovrà essere completato con attributi ragionevoli per ciascuna entità (identificando le possibili chiavi) e relazione. Vanno specificati accuratamente i vincoli di cardinalità e partecipazione di ciascuna relazione.

Come stabilito con i professori non abbiamo preso in considerazione le ultime righe del testo che specificano come ottenere il costo dei singoli articoli dalle rispettive partite.
Abbiamo quindi inserito il prezzo dei singoli articoli come attributo della relazione Articolo.

Analisi dei requisiti

L'analisi dei requisiti presuppone un adeguato studio preliminare dell'organizzazione, valutando i seguenti aspetti:

- gli obiettivi del sistema informatico da realizzare
- le attività dell'organizzazione che devono essere supportate dal sistema informatico
- le unità organizzative che utilizzeranno il sistema informatico.

Con questo studio, inoltre, vengono già individuati, ad un massimo livello di astrazione, i bisogni informativi, le procedure di interesse e le loro interazioni con i settori dell'organizzazione.
Il risultato della fase di analisi dei requisiti consiste in una serie di documenti che specificano:

- Le informazioni scambiate o condivise tra i settori, fissando il significato dei termini. Spesso, infatti, passando da un settore ad un'altro, lo stesso termine è usato con significati diversi (omonimi) oppure termini diversi denotano la stessa cosa (sinonimi).
- La conoscenza procedurale, ponendo l'attenzione sui servizi che il sistema deve offrire ai suoi utenti per definire cosa devono fare le operazioni, piuttosto che come lo devono fare.

Le informazioni sono di solito raccolte con l'attività di analisi dei dati.
Ci si dedica inizialmente alla comprensione e alla specifica degli aspetti strutturali, in particolare di quei fatti da memorizzare nella base di dati, cercando di darne una descrizione indipendentemente da come essi vengono utilizzati dalle operazioni.
Successivamente si specificano le operazioni controllando che siano eseguibili con le informazioni disponibili.
La specifica delle operazioni servono a ricavare una descrizione delle informazioni necessarie per eseguirle.

La metodologia di analisi dei requisiti utilizzata è la seguente:

- Si analizza il sistema informativo esistente raccogliendo una prima versione dei requisiti, espressa in linguaggio naturale.
- Si rivedono i requisiti espressi in linguaggio naturale, per eliminare

ambiguità e imprecisioni.
- Si raggruppano le frasi relative a categorie diverse di dati, operazioni.
- Si costruisce un glossario dei termini.
- Si specificano le operazioni degli utenti.
- Si verifica la completezza (tutti gli aspetti dei requisiti sono stati presi in considerazione) e consistenza (tutti i concetti usati nella specifica sono stati definiti, in particolare le operazioni fanno riferimento a dati definiti e i dati definiti sono usati dalle operazioni).

Glossario dei termini

Termine	Descrizione	Sinonimi	Links
Mercato	Area di territorio che può essere una macrozona o una singola regione		Cliente, macrozona, regione
Macrozona	Zona geografica formata da due o più regioni (ad esempio Triveneto)		Mercato, regione
Regione	Singola regione che può essere contenuta in una o più macrozone		Mercato, macrozona
Cliente	Entità che effettua ordini in un singolo mercato		Mercato, agente ordine
Agente	Dipendente dell'azienda che segue zero o più clienti		Cliente, agente
Ordine	Insieme di articoli richiesti dal cliente	Vendite	Articolo , cliente
Linea di prodotto	Categoria specifica di un articolo (ad esempio coscia di pollo, petto di pollo ecc…)		Articolo
Articolo	Concretizzazione della linea di prodotto (ad esempio confezione di petto di pollo con relativo codice, peso, prezzo, data di scadenza)		Ordine, linea di prodotto, partita

Partita	Insieme di articoli appartenenti ad un’unica linea di prodotto		Fornitore, articolo
Fornitore	Entità che fornisce le partite		Partita

Revisione dei requisiti e raggruppamento delle frasi

Frasi di carattere generale:
Si desidera creare una base di dati per un’azienda che si occupa della vendita di carne.

Frasi relative al cliente:
Per ogni cliente si vuole memorizzare la P.IVA che lo identifica, nome, sede e email. Ogni cliente appartiene ad un unico specifico mercato ed può essere seguito da agenti. Ogni cliente essendo interessato a comprare della carne effettua un ordine

Frasi relative all’articolo:
Per ogni articolo si vuole memorizzare la data di scadenza, prezzo, peso e il codice che lo identifica all’interno di una partita. Gli articoli possono far parte di un ordine ed uno specifico articolo appartiene ad un’unica linea di prodotto.

Frasi relative all’ordine:
Per ogni ordine si vuole memorizzare il numero di ordine che lo identifica, la data in cui è stato effettuato, il metodo di pagamento e il costo. Ogni ordine è composto da uno o più articoli. Il suo costo è dato dalla somma dei prezzi degli articoli che lo compongono.

Frasi relative alla partita:
Per ogni partita si vuole memorizzare il codice che lo identifica, il peso, il numero di prodotti, la qualità della carne (può essere di prima,secondo o terza scelta) e il costo diretto che è dato dalla somma tra spedizione, stoccaggio e costo d’acquisto. Ogni partita può contenere articoli appartenenti ad un’unica linea di prodotto.

Frasi relative al fornitore:
Per ogni fornitore si vuole memorizzare la P.IVA che lo identifica, nome, sede, numero di telefono e email.

Frasi relative alla linea di prodotto:
Per ogni linea di prodotto si vuole memorizzare il tipo di carne che lo identifica.

Specifica delle operazioni

	Descrizione
Operazione 1	Inserimento di un articolo
Operazione 2	Inserimento di un ordine
Operazione 3	Legare un articolo ad un ordine già presente nella BDD
Operazione 4	Inserimento di un cliente
Operazione 5	Legare un ordine ad un cliente già presente nella BDD
Operazione 6	Mostrare un ordine con il relativo costo

La progettazione concettuale

Lo scopo della progettazione concettuale è tradurre il risultato dell'analisi dei requisiti in una descrizione degli aspetti strutturale del sistema informatico studiato. Mentre l'analisi dei requisiti analizza cosa si aspettano i singoli settori dal sistema, in questa fase l'attenzione è su come disegnare una base di dati che garantisca per tutti i settori le funzionalità desiderate. Il risultato della progettazione concettuale è lo schema concettuale, indipendente dal DBMS che verrà usato nella realizzazione, con costrutti ad alto livello, adatti a descrivere il significato di ciò che si sta modellando.

Le principali proprietà che caratterizzano un buon progetto concettuale sono le seguenti:

-Completezza concettuale: vanno specificati tutti gli aspetti rilevanti delle applicazioni, in modo dettagliato, per consentirne una realizzazione corretta

-Indipendenza dalle applicazioni: lo schema concettuale non deve essere

fatto in funzione dell'efficienza di particolari applicazioni, ma con l'obiettivo di una efficace modellazione.

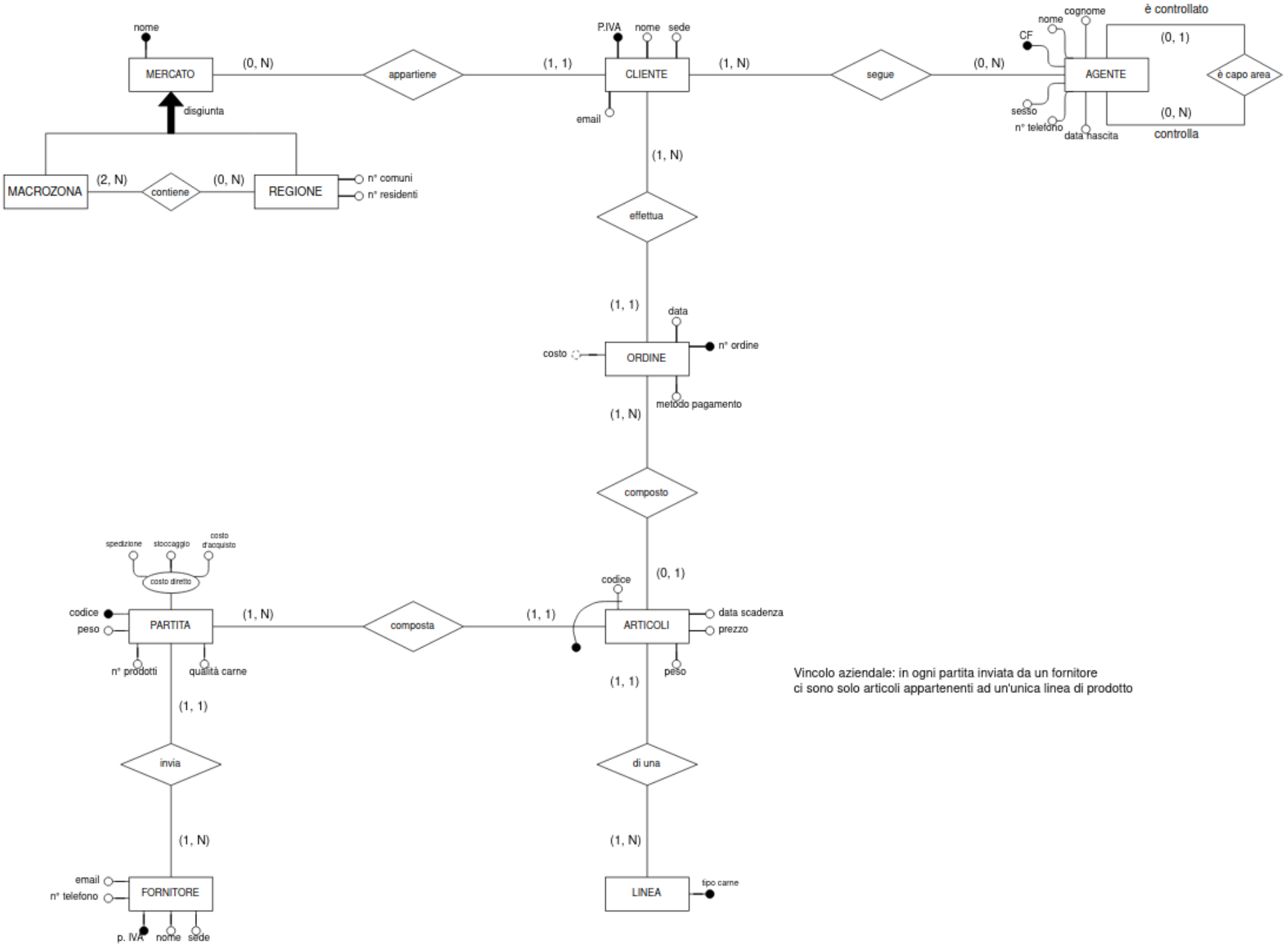
-Indipendenza dal DBMS: nello schema concettuale si prescinde da aspetti specifici del DBMS impiegato nella realizzazione

Per descrivere gli aspetti strutturali di un progetto concettuale abbiamo adottato la soluzione basata sul modello entità-relazione.
Per la realizzazione dello schema ER abbiamo adottato una strategia TOP-DOWN. Il processo comincia con una alta astrazione dei concetti e poco dettaglio. Nel corso del processo i singoli concetti sono approfonditi e arricchiti con dettagli. Tale strategia ci ha dato il vantaggio di avere una visione globale del problema partendo da uno scheletro che toccava tutti gli elementi fondamentali del dominio.

Costruzione dello schema ER

Per prima cosa abbiamo analizzato e approfondito ogni termine del testo dei requisiti che ci sembrava importante, categorizzandolo in entità oppure relazione.
Per quanto riguarda l'entità mercato abbiamo deciso di specializzarla in due entità: macrozona e regione. Nel testo dei requisiti infatti è spiegato esplicitamente che il mercato (entità in cui opera l'azienda) è costituito da una macrozona (insieme di due o più regioni) o da singole regioni. La specializzazione è totale e disgiunta, infatti un mercato può coprire o una macrozona o una singola regione.
Per quanto riguarda l'entità cliente abbiamo deciso di identificarlo nella figura di un rivenditore al dettaglio di carne o un supermercato, quindi caratterizzato con nome, indirizzo della sede, email (telefono) e partita IVA che è anche chiave primaria.
La relazione che lega mercato a cliente è "appartiene" e abbiamo scelto di dare zero come minima cardinalità sul lato di mercato in quanto potrebbero essere presenti nella base di dati dei mercati che non hanno ancora clienti. Ogni cliente è seguito da almeno un agente (persona vera e propria) e un agente può seguire più clienti, oppure, se è appena stato inserito nella base di dati, potrebbe non essergli stato assegnato ancora nessun cliente.
Abbiamo interpretato il capo area come un agente.
Abbiamo quindi creato la relazione ricorsiva "è capo area": un agente può non controllare nessun altro agente (quindi non è capo area) oppure ne può

controllare molti, ogni agente è controllato da al più un altro agente.
Il cliente per essere tale deve aver effettuato almeno un ordine e un ordine specifico è effettuato da uno e un solo cliente. L'entità ordine ha come attributi un codice univoco, la data in cui è stato effettuato, il metodo di pagamento e il costo complessivo. Quest'ultimo attributo è derivato, in quanto può essere ricavato dalla somma dei prezzi dei singoli articoli contenuti nell'ordine. La ridondanza di questo attributo verrà studiata in seguito.
Un articolo è una confezione di un taglio di carne (preso da una linea di prodotto), con un suo peso, prezzo, data di scadenza e codice che, seguendo le istruzioni del testo, è un numero progressivo che identifica un articolo all'interno di una partita specifica. Articolo è quindi una entità debole e la sua chiave primaria è formata dal codice dell'articolo e dal codice della partita in cui è contenuto.
Il fornitore, entità che produce effettivamente la materia prima cioè la carne, invia più partite di carne, caratterizzate da un codice univoco, peso, numero di articoli, qualità della carne e dal costo diretto, un attributo composto.
Ogni partita, secondo le richieste del testo, contiene solo articoli appartenenti a una sola linea di prodotto. Questo aspetto non può essere gestito direttamente dai vincoli di cardinalità, infatti leggendo lo schema così com'è potremmo dire che un fornitore invia una partita nella quale ci sono più articoli di più linee di prodotto. è quindi necessario specificare esplicitamente il seguente vincolo aziendale: ogni partita contiene solo articoli appartenenti a un'unica linea di prodotto.



Vincolo aziendale: in ogni partita inviata da un fornitore
ci sono solo articoli appartenenti ad un'unica linea di prodotto

La progettazione logica

L'obiettivo della progettazione logica è tradurre lo schema ER prodotto nella fase di progettazione concettuale in uno schema logico in grado di descrivere gli stessi dati in maniera corretta ed efficiente.
Non si tratta di una pura e semplice traduzione per 2 motivi:

- alcuni aspetti non sono direttamente o univocamente rappresentabili.
- è necessario prestare attenzione alle prestazioni.

La progettazione logica si articola in 2 fasi:

- ristrutturazione dello schema ER: è una fase indipendente dal modello logico scelto e si basa su criteri di ottimizzazione dello schema e di semplificazione della fase successiva. In questa fase vengono eliminati tutti i costrutti che non possono essere direttamente rappresentati nel modello logico.
- traduzione verso il modello logico: si fa riferimento a uno specifico modello logico (modello relazionale). Si opera traducendo entità e associazioni.

Ristrutturazione dello schema ER

Per prima cosa vengono analizzate le prestazioni della base di dati valutando:

- *costo di un'operazione*: valutato secondo il numero di occorrenze di entità e associazioni che mediamente vanno visitate per rispondere ad un'operazione sulla base di dati.
- *occupazione di memoria*: valutato in base allo spazio di memoria in byte utilizzato per memorizzare i dati descritti dallo schema

Per analizzare questi parametri bisogna conoscere le seguenti informazioni:

- *volume dei dati*: numero di occorrenze di ogni entità e associazioni dello schema
- *caratteristiche delle operazioni*:
    - frequenza (numero medio di esecuzioni in un certo intervallo di tempo)
    - dati coinvolti (entità e relazioni).

Tabella delle operazioni

	Frequenza
operazione 1	300°000 al giorno
operazione 2	56°000 a settimana
operazione 3	240°000 al giorno
operazione 4	2500 all’anno
operazione 5	56°000 a settimana
operazione 6	62°000 al mese

Tabella dei volumi

Concetto	Tipo	Volume
cliente	E	25000
ordine	E	29°760°000
articolo	E	1°000°000°000
effettua	R	29°760°000
composto	R	892°000°000

Abbiamo ipotizzato che la nostra base di dati abbia una vita di 10 anni. In media vengono inseriti 2500 clienti all’anno, in 10 anni vengono resi disponibili un miliardo di articoli e vengono effettuati 8000 ordini al giorno con 30 articoli per ordine.
Avendo queste informazioni è possibile fare una stima del costo di un’operazione sulla base di dati, contando il numero di accessi alle occorrenze di entità e relazioni necessarie per eseguire l’operazione.
La stima del costo è rappresentata da una tavola degli accessi che contiene informazioni sul tipo di accesso (Lettura e Scrittura) e il costo dell’accesso.
Le operazioni di scrittura sono più onerose delle operazioni di lettura.

La ristrutturazione dello schema ER si suddivide quindi nelle seguenti fasi:

- *analisi delle ridondanze*: fase nella quale, in base all'analisi sulle prestazioni appena descritta, si decide se mantenere o eliminare eventuali ridondanze dello schema.
- *eliminazione di generalizzazioni*: se nello schema sono presenti delle generalizzazioni, queste vengono analizzate e sostituite da adeguati costrutti.
- *scelta degli identificatori*: vengono scelte le chiavi primarie candidate per ciascuna entità.

Analisi delle ridondanze

La ridondanza in uno schema concettuale è la presenza di un dato che può essere derivato da altri. La presenza di ridondanza ha sia dei vantaggi che svantaggi, semplifica le interrogazioni ma allo stesso tempo appesantisce gli aggiornamenti. La decisione se mantenere o eliminare una ridondanza va presa confrontando il costo di esecuzione delle operazioni che coinvolgono il dato ridondante.
Nella nostra base di dati, il dato ridondante è l'attributo 'costo' dell'entità 'ordine'. In quanto il costo di un ordine può essere ricavato contando i prezzi delle istanze degli articoli legati a quel specifico ordine, presenti sulla relazione 'composto'. A quel punto per ottenere il costo basta sommare il prezzo di ogni singolo articolo.

Tavola degli accessi in presenza di ridondanza

Operazione 3 (legare un articolo ad un ordine già presente nella BDD):

concetto	costrutto	accesso	tipo
ordine	E	1	S
composto	R	1	S
ordine	E	1	L
articolo	E	1	L

Operazione 6 (mostrare un ordine con il relativo costo):

concetto	costrutto	accesso	tipo
ordine	E	1	L

Tavola degli accessi in assenza di ridondanza

Operazione 3 (legare un articolo ad un ordine già presente nella BDD):

concetto	costrutto	accesso	tipo
composto	R	1	S

Operazione 6 (mostrare un ordine con il relativo costo):

concetto	costrutto	accesso	tipo
ordine	E	1	L
composto	R	336	L
articolo	E	336	L

336 è il numero medio di articoli in un ordine ottenuto da
articolo(1°000°000°000) / ordine(29°760°000)

Assunzioni:

- assumo che un accesso in scrittura ha un costo doppio rispetto ad un accesso in lettura.

Calcolo degli accessi in presenza di ridondanza:

numero di accessi:
(2+2+1+1)*240000 + (1)*2000 = 1°442°000

Calcolo degli accessi in assenza di ridondanza:

numero di accessi:
(2)*240000 + (673)*2000 = 1°826°000

In conclusione, se la frequenza delle operazioni è quella indicata nella tavola delle operazioni allora conviene mantenere la ridondanza.

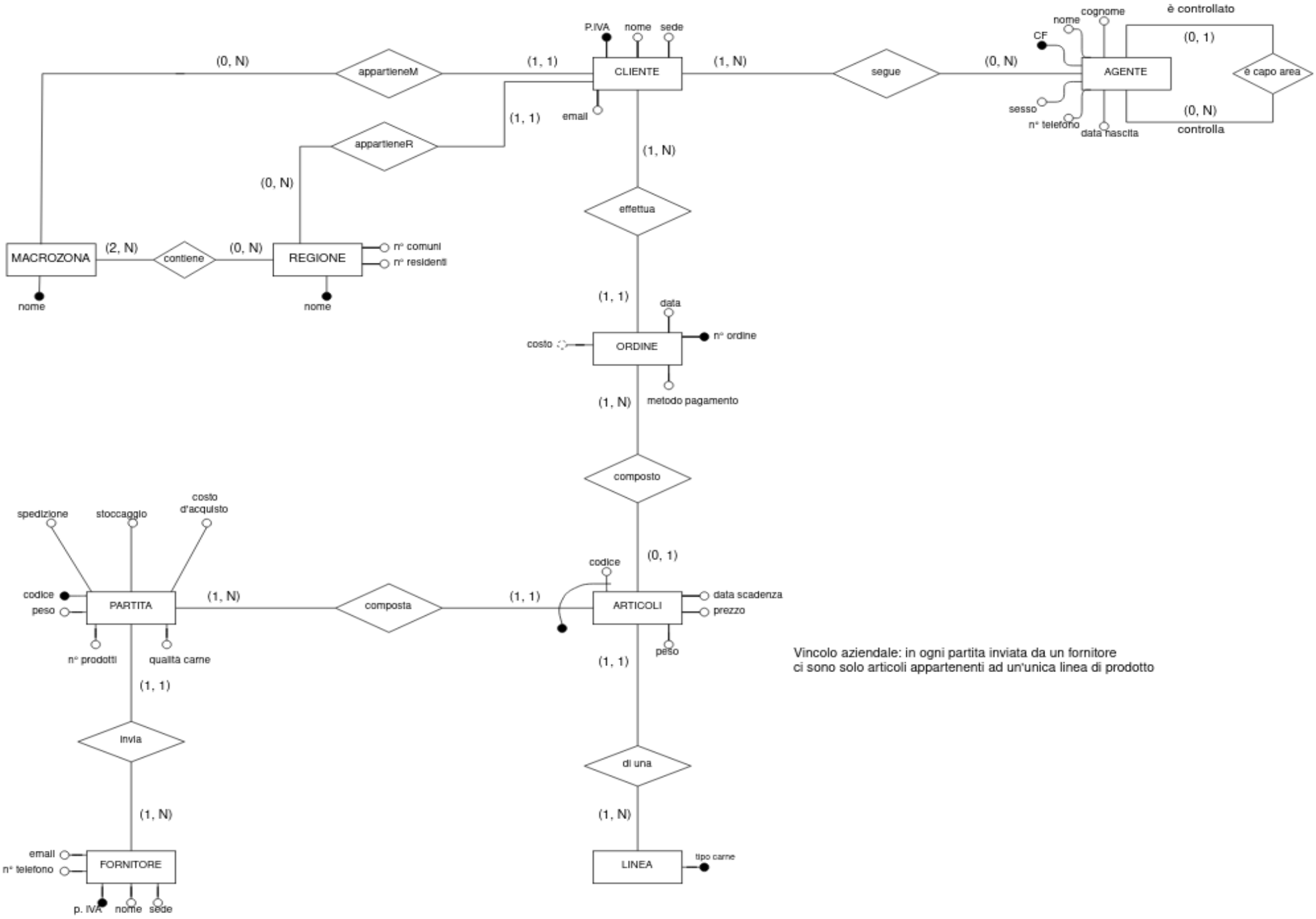
Eliminazione delle generalizzazioni

Nel nostro schema ER è presente una generalizzazione. Il requisito che un mercato possa essere o una macrozona o una regione è stato gestito infatti specificando l'entità mercato in due entità figlie, tra loro legate da una relazione.

Esistono più modi per eliminare una generalizzazione:

- accorpando le entità figlie nel genitore
- unendo il genitore alle entità figlie
- sostituendo la generalizzazione con relazioni uno a uno tra le entità figlie e il genitore

Dopo un'analisi delle relazioni fra le entità che compongono la nostra generalizzazione abbiamo scelto la seconda alternativa.
Le entità Macrozona e Regione hanno quindi ereditato gli attributi di Mercato mentre la relazione Appartiene viene duplicata (una relativa a Macrozona e una a Regione), mantenendo la stessa cardinalità con l'entità Cliente.
La scelta di utilizzare il secondo modello di eliminazione è conveniente in quanto nella base di dati ci sono operazioni che riguardano solo istanze di Macrozona oppure istanze di Regione. Un mercato infatti è identificato o in una macrozona o in una regione.
Rispetto alla prima alternativa abbiamo il vantaggio di non avere valori nulli negli attributi (tutti gli attributi delle entità figlie verrebbero uniti a quelli del genitore e ad esempio un'istanza di macrozona avrebbe gli attributi relativi alla regione nulli), risparmiando spazio. Rispetto alla terza scelta, per accedere alle istanze di Macrozona e Regione non dobbiamo visitare l'entità Mercato, quindi abbiamo meno operazioni di accesso alla base di dati.
Con l'eliminazione della generalizzazione presente nello schema ER sono stati introdotti 2 vincoli aziendali:
-un cliente deve appartenere o ad una regione o ad una macrozona (non entrambe).
- un cliente non può non essere in relazione con nessuna delle 2 entità



Vincolo aziendale: in ogni partita inviata da un fornitore
ci sono solo articoli appartenenti ad un'unica linea di prodotto

Scelta degli identificatori

La chiave primaria serve a identificare univocamente una istanza di un'entità. Un'entità può avere più chiavi primarie candidate ma in generale è meglio sceglierne una in quanto quest'ultima verrà usata per la costruzione degli indici, che verranno descritti in seguito.
Una chiave primaria ovviamente non deve essere costituita da un attributo che ammette valori nulli, è da preferire se è composta dal minor numero di attributi ed è meglio se gli attributi sono interni all'entità senza coinvolgere troppe entità.

Le chiavi primarie delle entità del nostro schema ER sono le seguenti:

- nome per l'entità Macrozona: supponendo che non esistano due macrozone con stesso nome
- nome per Regione: supponendo che non esistano due regioni con stesso nome
- partita IVA per Cliente: il cliente è un negozio identificato dalla sua partita IVA
- codice fiscale per Agente: l'agente è una persona
- numero ordine per Ordine: univoco fra tutti gli ordini di tutti i clienti
- codice per Partita
- partita IVA per Fornitore
- coppia (codice, codicePartita) per Articolo: articolo è un'entità debole quindi oltre al codice che identifica un articolo all'interno di una partita serve anche il codice della partita dove è contenuto
- tipo carne per Linea

Traduzione verso il modello logico

In questa fase ci si occupa di convertire lo schema ER ristrutturato in uno schema relazionale equivalente, cioè un modello basato sulle relazioni, che possono corrispondere alle entità o alle relazioni dello schema ER.
Una relazione R o tabella si rappresenta così: R(A), dove A è l'insieme degli attributi della relazione.
Le righe di una tabella (tuple) rappresentano le singole istanze della relazione e sono identificate dalla chiave primaria. è possibile inoltre specificare delle chiavi esterne che fungono da identificatori di tuple appartenenti a relazioni diverse. In questo modo si possono mettere in relazione più tabelle.
La chiave esterna che si riferisce ad una certa tabella R assume valori in un sottoinsieme dei valori della chiave primaria di R.

I metodi di traduzione dello schema ER in schema relazionale cambiano a seconda del tipo e della cardinalità delle associazioni tra le entità.
Noi abbiamo proceduto nel seguente modo:

Macrozona (E) -> macrozona(nome)

Regione (E) -> regione(nome, numeroComuni, numeroResidenti)

Contiene (R) -> contiene(macrozona, regione):
CE: macrozona → Macrozona(nome)
regione → regione(nome)

Cliente (E) -> cliente(P.IVA, nome, sede, email, macrozona, regione):
CE: macrozona → Macrozona(nome)
regione → regione(nome)

Agente (E) -> agente(CF, nome, sesso, email, capoArea):
CE: capoArea → Agente(CF)

Segue (R) -> segue(cliente, agente):
CE: cliente → cliente(P.IVA)
agente → agente(CF)

Ordine (E) -> ordine(numero, costo, data, metodoPagamento, cod_cliente):
CE: cod_cliente → cliente(P.IVA)
VNN: {cod_cliente}

Linea (E) -> linea(tipoCarne)

Fornitore (E) -> fornitore(P.IVA, telefono, email, nome, sede)

Partita (E) -> partita(codice, peso, numeroProdotti, spedizione,
stoccaggio, costo_d'acquisto, Piva_fornitore):
CE: Piva_fornitore → fornitore(P.IVA)
VNN: {Piva_fornitore}

Articolo (E) -> articolo(codice, partita, data, scadenza, prezzo, peso,
tipoCarne, cod_ordine):
CE: partita → partita(codice)
tipoCarne → linea(tipoCarne)
cod_ordine → ordine(numero)
VNN: {tipoCarne}

Vincoli referenziali e gestione delle associazioni

Nello schema ER le entità Macrozona e Regione sono legate da Contiene, una associazione molti a molti. Questo si traduce nello schema relazionale con l'introduzione di due chiavi esterne nella relazione Contiene che insieme costituiscono la sua chiave primaria. Il vincolo su questa relazione è che ogni istanza di macrozona deve essere contenuta in almeno due tuple di contiene, infatti una macrozona è costituita da almeno due regioni.
Le associazioni uno a molti si traducono semplicemente inserendo tra gli attributi della relazione che può avere più tuple per una stessa istanza di un'altra relazione l'identificatore di quest'ultima relazione, cioè la sua chiave esterna. Non c'è quindi bisogno di convertire l'associazione in una relazione. Ad esempio l'associazione Effettua dello schema ER non viene convertita in una relazione nello schema relazionale, in quanto basta aggiungere la chiave esterna cod_cliente, che punta alla relazione cliente, nella relazione ordine.

A questo punto, lo schema va annotato con i vincoli d'integrità che non siamo riusciti a rappresentare in modo esplicito e con le regole di derivazione degli attributi derivati. Quando ho in un verso dell'associazione la cardinalità (1,N), la cardinalità minima (1) non è codificabile direttamente nello schema relazionale attraverso vincoli di primary key, key secondarie, NOT NULL, UNIQUE.
Questi sono i vincoli che si sono venuti a creare:

-ogni tupla di macrozona deve essere contenuta in almeno 2 tuple della relazione contiene.
-ogni tupla di linea deve essere presente in almeno una tupla della relazione articolo
-ogni tupla di partita deve occorrere in almeno una tupla della relazione articolo
-ogni tupla di fornitore deve occorrere in almeno una tupla della relazione partita
-ogni tupla di cliente deve occorrere in almeno una tupla della relazione ordine
-ogni tupla di cliente deve occorrere in almeno una tupla della relazione segue

Regola di derivazione:
il valore dell'attributo costo di ogni istanza della relazione ordine può essere ricavato andando a sommare i valori presenti nell'attributo prezzo nelle istanze della relazione articolo

La progettazione fisica

L'obiettivo della progettazione fisica è rappresentare lo schema relazionale sul DBMS mediante SQL, popolare la base di dati, implementare tecniche di controllo di inconsistenze e realizzare interrogazioni per operare sui dati.

Strutture indici nei DBMS relazionali

La costruzione di indici in una relazione mette a disposizione strutture ausiliarie per localizzare velocemente i dati di interesse e mantenere performante la struttura a fronte di una modifica del file dei dati.
Le operazioni sui file possono essere suddivise in 2 categorie:

- operazioni di recupero dei dati (ricerca)
- operazioni di aggiornamento

I dati memorizzati su disco sono organizzati in file di record.
I record possono essere memorizzati in modo sequenziale o ordinato.
Sequenziale significa che i record vengono inseriti alla fine del file.
Ordinato significa che è possibile ordinare fisicamente i record sulla base dei valori di uno dei loro campi.
Vantaggi e svantaggi:

-Avere un file dei dati ordinato permette di effettuare una ricerca efficiente (ricerca binaria con complessità O(logn) dove n rappresenta il numero di record del file) ma è poco efficiente a fronte di un'operazione di aggiornamento O(nlogn).

- Un file non ordinato non permette di effettuare una ricerca efficiente. Infatti il suo costo computazionale è O(n). Mentre l'inserimento di un file è efficiente (theta di 1).

Quindi, se nella mia base di dati, l'operazione di inserimento è molto frequente, ma l'operazione di ricerca è poco frequente allora il file non ordinato è una buona soluzione. Se al contrario, l'operazione di inserimento è poco frequente, ma l'operazione di ricerca è molto frequente, allora il sorted file è la soluzione più adatta.

Se a questo file aggiungiamo una struttura ausiliaria indice,
quest'ultima ci consente di avere una maggiore efficienza in 2 casi:

- quando le operazioni di inserimento e aggiornamento sono molto frequenti.

- quando le operazioni di ricerca sono molto più frequenti delle operazioni di aggiornamento.

Nel caso in cui le operazioni di ricerca sono poco frequenti e quelle di inserimento sono molto più numerose, la struttura ad indice non porta alcun vantaggio.In quanto l'allocazione della struttura ha un costo in termini di spazio di memoria e mantenerla performante nel caso di inserimenti ha un costo computazionale. In questo caso è più conveniente usare un file non ordinato.
Per la nostra base di dati, risulta vantaggioso creare degli indici per le tabelle 'ordine' e 'articolo'. In quanto in entrambe le tabelle le operazioni ricerca e aggiornamento sono molto frequenti. l'indice più adatto è il multilivello dinamico (B-tree o B+-tree)

```
CREATE INDEX indice_n_ordine ON ordine (n_ordine);



CREATE INDEX indice_codice_articolo ON articolo (cod_art);
```

Abbiamo scelto come campo di indicizzazione le chiavi primarie delle 2 tabelle.

Trigger

L'obiettivo dei trigger è quello di gestire le eccezioni nell'inserimento, nell'aggiornamento e nell'eliminazione di tuple dalle tabelle per mantenere i vincoli d'integrità. Sono meccanismi che si attivano automaticamente alla modifica delle tabelle che gestiscono.
Bisognerebbe implementare i trigger per ogni vincolo aziendale, noi ne abbiamo implementati alcuni.
Abbiamo implementato due trigger, il primo trigger per la gestione delle linee di produzione nelle partite, perché i singoli articoli nelle partite devono appartenere alla stessa linea di produzione.
Il secondo trigger l'abbiamo creato per la gestione delle aree di cui i clienti appartengono, perché un cliente non può appartenere contemporaneamente ad una regione e ad un macroarea e ogni cliente deve appartenere ad almeno una delle due quindi non può avere i campi doc_regione e cod_macrozona entrambi nulli.

Inoltre abbiamo implementato tre query.
La prima per calcolare gli introiti medi di tutti i clienti in base all'agente che li ha seguiti.
La seconda query per il totale degli introiti dovute alle vendite effettuate in Friuli.
La terza query per calcolare il totale delle spese che il cliente con la p.iva=345678 ha sostenuto.

```
-------INSERIMENTO DI AGENTI
insert into agente(cf, nome, email, sesso, capo_area) values
('lcrpn99t211493q', 'Lucrezia', 'lucrezia99@gmail.com','F',
null);
insert into agente(cf, nome, email, sesso, capo_area) values
('jrdut87t21r487q', 'Jordan', 'jordan87@gmail.com','M',
'lcrpn99t211493q');
insert into agente(cf, nome, email, sesso, capo_area) values
('dnsgt67t98q234t', 'Denis', 'denis67@gmail.com','M', null);
insert into agente(cf, nome, email, sesso, capo_area) values
('gltrg89t211345a', 'Giulia', 'giulia89@gmail.com','F',
'dnsgt67t98q234t');


-------INSERIMENTO MACROZONA
insert into macrozona(nome) values ('Triveneto');
insert into macrozona(nome) values ('Sud-Italia');
insert into macrozona(nome) values ('Centro-Italia');
insert into macrozona(nome) values ('Nord-Italia');
insert into macrozona(nome) values ('Isole');
```

```
-------INSERIMENTO DI CLIENTI
insert into cliente(p_iva, nome,sede,  email, cod_macrozona,
cod_regione) values (345678, 'palmarket','via dei tigli 56'
,'palmarketCatania@gmail.com','Isole',null);
insert into cliente(p_iva, nome,sede,  email, cod_macrozona,
cod_regione) values (378958, 'friulmarket','via dei preti 1'
,'friulmarketudine@gmail.com',null,'Friuli');
insert into cliente(p_iva, nome,sede,  email, cod_macrozona,
cod_regione) values (498769, 'carrefour','via giorgio 89'
,'carrefourBari@gmail.com',null,'Puglia');




-------INSERIMENTO ORDINE
insert into ordine(n_ordine, costo_ordine,data_ordine,
metodo_pagamento, cod_cliente) values (1,45642.56,
'02/02/2022', 'paypal', 345678);
insert into ordine(n_ordine, costo_ordine,data_ordine,
metodo_pagamento, cod_cliente) values (2,86642.56,
'02/02/2022', 'bonifico', 378958);
insert into ordine(n_ordine, costo_ordine,data_ordine,
metodo_pagamento, cod_cliente) values (3,85642.56,
'02/02/2022', 'bonifico', 498769);







create or replace function valida_partita()
returns trigger language plpgsql as
$$
declare
    codicearticolo numeric;
begin
    select  cod_art into codicearticolo
    from articolo A
    where  exists (select * from Articolo A1 where
```

```
A.cod_part=A1.cod_part and A.tipocarne<>A1.tipocarne);

    if found then
      raise notice 'la partita deve avere tutti gli articoli
della stessa linea';
      return null; -- annulla l'operazione
    end if;

    IF (TG_OP = 'UPDATE' or TG_OP = 'INSERT') THEN
     RETURN NEW;
     END IF;

end;
$$;


create trigger controllo_update_linea_partita
before update on articolo
for each row when (new.tipocarne<> old.tipocarne)
execute procedure valida_partita();



create trigger controllo_insert_linea_partita
before insert on articolo
for each row
execute procedure valida_partita();




-------FUNZIONE DEI CLIENTI E DELLE SUE ZONE
create or replace function validazione_cliente()
returns trigger language plpgsql as
$$
declare
piva character(11);
begin

    select c.p_iva into piva
```

```
    from cliente c
    where (c.cod_macrozona = null and c.cod_regione= null) or
(c.cod_macrozona <> null and c.cod_regione<> null) ;


    if found  then
      raise notice 'Non può esistere un cliente che
appartiene ad una regione e ad una macrozona
contemporaneamente oppure che non appartenga a nessuna delle
due';
      return null; -- annulla l'operazione
    end if;

    IF (TG_OP = 'UPDATE' or TG_OP = 'INSERT') THEN
     RETURN NEW;
     END IF;

end;
$$;


------- TRIGGER PER GESTIRE I CLIENTI  E DELLE SUE ZONE

create trigger controllo_aggiornamento_cliente
before  update on cliente
for each row when (new.cod_macrozona<>old.cod_macrozona  OR
new.cod_regione<> old.cod_regione)
execute procedure validazione_cliente();

create  trigger controllo_inserimento_cliente
before insert on cliente
for each row
execute procedure validazione_cliente();


-------INTERROGAZIONI

---SOMMA DELLE SPESE CHE OGNI CLIENTE HA EFFETTUATO IN BASE
ALL'AGENTE CHE L'HA SEGUITO

CREATE view segue_cliente AS
```

```
select p_iva, costo_ordine
from ordine, cliente
where cod_cliente = p_iva;



select agenteid, sum(costo_ordine)
from segue_cliente, segue
where p_iva = clienteid
group by agenteid;


---TOTALE DEGLI INTROITI DOVUTE ALLE VENDITE EFFETTUATE IN
FRIULI




select c.p_iva,sum(a.prezzo)
from articolo a, ordine o,cliente c
where c.p_iva=o.cod_cliente and o.n_ordine=a.ordine and
c.cod_regione='Friuli'
group by p_iva;




---IL TOTALE DELLE SPESE CHE IL CLIENTE CON LA P.IVA=345678
HA SOSTENUTO


select sum(costo_spedizione+costo_stoccaggio+costo_acquisto)
from partita
where p_ivaforn=345678
```

Utilizzo di R per il popolamento della base di dati

Abbiamo utilizzato ‘R’ per popolare le tabelle ‘CLIENTE’,’ORDINE’,’SEGUE’ con dati casuali.
Per la tabella ‘CLIENTE’ abbiamo generato 10.000 tuple, i cui valori degli attributi sono stati presi in modo casuale da elementi presenti in vettori precedentemente creati.
Per le tabelle ‘ordine’ e ‘segue’ abbiamo generato rispettivamente 100.000 e 8000 tuple.
Ad esempio, nella relazione cliente, l’attributo P.IVA è chiave primaria. Di conseguenza non possono esistere 2 tuple con lo stesso valore in quell’attributo. Per garantire questo vincolo, abbiamo creato un vettore di 10000 elementi diversi presi da un range di interi compresi 111111 e 999999.
Una volta inserite le tuple nella tabella cliente è stato possibile popolare le tabelle ‘ordine’ e ‘segue’. Dato che l’attributo ‘cod_cliente’ della relazione ‘ordine’ è chiave esterna rispetto alla tabella cliente e tale valore non può essere NULL abbiamo popolato per prima la relazione ‘cliente’. Stessa cosa per la relazione ‘segue’, l’attributo ‘clienteid’ è chiave esterna rispetto alla tabella cliente e tale valore non può essere NULL dato che fa parte della chiave primaria.

Questo è il codice scritto in R per popolare tali tabelle:

```
#POPOLAMENTO CLIENTE:

zona <- dbGetQuery(con,"select * from
progettobasi.macrozona")
v_zona <- zona$nome
```

```
 df_cliente <- data.frame(p_iva =
sample(111111:999999,10000,replace = F),nome =
sample(NA,10000,replace = T) ,cod_macrozona =
sample(v_zona,10000,replace = T),
cod_regione=sample(NA,10000,replace = T)  )

dbWriteTable(con, name=c("progettobasi","cliente"), value =
df_cliente, append=T, row.names=F)


#POPOLAMENTO ORDINE:

v_pivacliente <- dbGetQuery(con,"select * from
progettobasi.cliente")
v_pivacliente1 <- v_pivacliente$p_iva
v_data <- readLines("data.txt")
v_nordine <- readLines("Nordine.txt")

df_ordine <-
data.frame(n_ordine=sample(v_nordine,100000,replace =
F),costo_ordine=sample(100:10000,100000,replace =
T),data_ordine=sample(v_data,100000,replace = T),
metodo_pagamento=sample(NA,100000,replace =
T),cod_cliente=sample(v_pivacliente1,100000,replace = T))




#POPOLAMENTO SEGUE:

agente <- dbGetQuery(con,"select * from progettobasi.agente")
 v_agente <- agente$cf
df_segue <-
data.frame(clienteid=sample(v_pivacliente1,8000,replace =
F),agenteid=sample(v_agente,8000,replace = T))
dbWriteTable(con,name=c("progettobasi","segue"),value=df_segu
e,append=T,row.names=F)
```

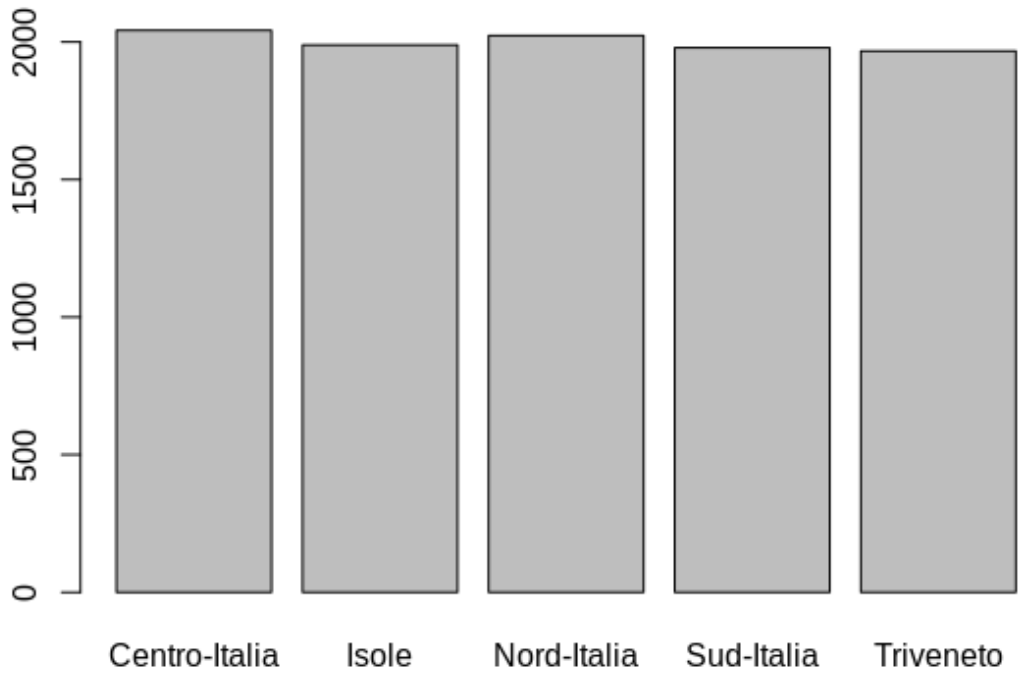
Esempio tuple delle tabelle cliente, ordine, segue generate in modo casuale:

p_iva	nome	sede	email	cod_macrozona	cod_regione
184581				Isole	
935801				Nord-Italia	
177029				Nord-Italia	
498963				Isole	
227786				Triveneto	
752821				Nord-Italia	
819511				Triveneto	
528741				Triveneto	
625145				Sud-Italia	
602802				Centro-Italia	
650956				Isole	
540674				Nord-Italia	
236247				Triveneto	
235172				Triveneto	
612222				Nord-Italia	
814883				Isole	
381926				Triveneto	
629003				Triveneto	
348333				Sud-Italia	
227264				Centro-Italia	
726171				Sud-Italia	
350358				Triveneto	
578271				Nord-Italia	
575357				Centro-Italia	
282693				Triveneto	
595961				Nord-Italia	
126472				Nord-Italia	
299550				Triveneto	
775483				Centro-Italia	
675099				Triveneto	
534945				Nord-Italia	
568042				Isole	
338877				Triveneto	
373451				Nord-Italia	
736031				Triveneto	

n_ordine	costo_ordine	data_ordine	metodo_pagamento	cod_cliente
10997	6783	27/10/2019		956501
65715	3643	18/06/2012		566888
64347	1317	28/10/2014		243782
15689	9698	16/05/2010		709887
32357	9278	25/12/2013		135356
65926	9547	17/06/2014		459909
79726	466	10/06/2011		268546
92018	9723	10/03/2013		591455
35700	2700	15/04/2019		428313
14164	7705	08/06/2015		750613
70431	1525	29/10/2017		635562
21977	5905	17/12/2011		300133
61805	7856	11/04/2016		454136
13373	5040	19/08/2014		254082
84204	5883	25/10/2012		317609
14617	9631	18/01/2013		370429
54352	9709	11/08/2011		597115
6451	7373	02/06/2015		720788
20589	4192	27/01/2010		186167
55102	6603	29/09/2012		924281
72911	8658	09/12/2019		501793
83867	6278	25/08/2013		279335
25676	5466	14/11/2011		294715
44754	5765	05/01/2010		632049
22981	6399	01/06/2013		419923
39838	5698	19/03/2017		566701
49761	3374	20/05/2016		883476
44904	9549	25/12/2013		538792
33876	4209	04/08/2018		710273
23222	359	02/08/2013		677677
10513	3364	19/08/2018		723879
13697	8559	01/03/2018		396359
34114	3418	03/07/2016		731477
45630	6561	06/10/2011		505562
7912	483	06/09/2016		455095

clienteid	agenteid
767244	lcrpn99t21l493q
334718	gltrg89t21l345a
987826	gltrg89t21l345a
305672	lcrpn99t21l493q
986332	dnsgt67t98q234t
500535	lcrpn99t21l493q
948333	jrdut87t21r487q
878187	dnsgt67t98q234t
966275	dnsgt67t98q234t
341626	jrdut87t21r487q
208027	dnsgt67t98q234t
756988	gltrg89t21l345a
435507	gltrg89t21l345a
899446	dnsgt67t98q234t
699110	dnsgt67t98q234t
646015	gltrg89t21l345a
305852	lcrpn99t21l493q
393808	gltrg89t21l345a
788202	dnsgt67t98q234t
829202	lcrpn99t21l493q
372513	jrdut87t21r487q
197878	jrdut87t21r487q
239339	jrdut87t21r487q
644982	dnsgt67t98q234t
586771	gltrg89t21l345a
246976	jrdut87t21r487q
727953	jrdut87t21r487q
852727	lcrpn99t21l493q
744898	lcrpn99t21l493q
446248	jrdut87t21r487q
654577	gltrg89t21l345a
749158	dnsgt67t98q234t
502778	lcrpn99t21l493q
771574	gltrg89t21l345a
483542	gltrg89t21l345a

Infine, abbiamo effettuato un’analisi statistica con le tuple appena generate. Abbiamo creato un grafico che mostra la distribuzione dei clienti nelle macrozone in cui risiedono.



Dal grafico possiamo vedere che i clienti sono abbastanza uniformemente distribuiti nelle macrozone

Questo è il codice R per la generazione del grafico:

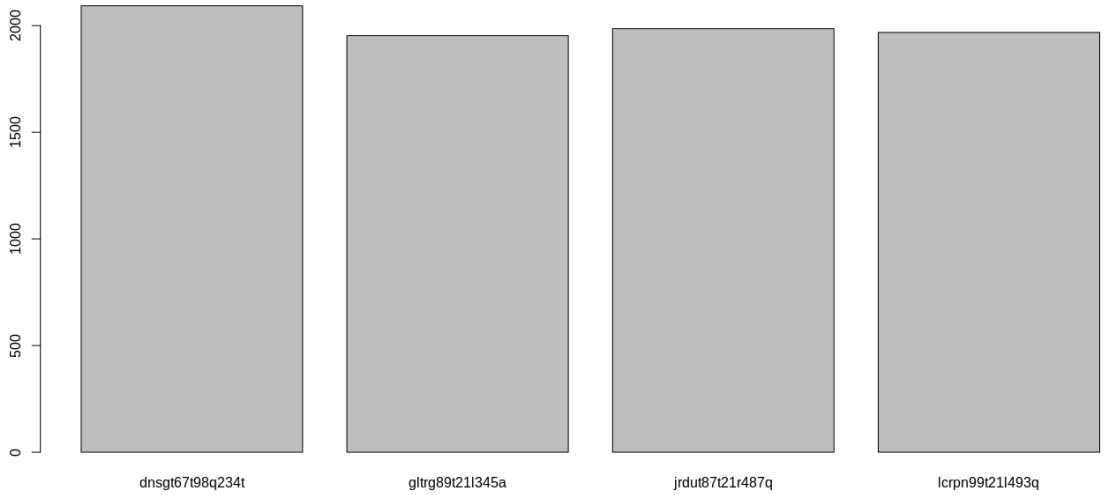
```
df <- dbGetQuery(con, "select cod_macrozona, p_iva
                       from cliente")


barplot(table(df$cod_macrozona))
```

Un secondo grafico mostra la distribuzione dei clienti seguiti dai rispettivi agenti.

```
df <- dbGetQuery(con, "select *
                       from segue")


barplot(table(df$agenteid))
```


0
500
1000
1500
2000
dnsgt67t98q234t
gltrg89t21l345a
jrdut87t21r487q
lcrpn99t21l493q