**Progetto Basi di Dati**

Cometa S.P.A.

**Abstract**

Cometa S.P.A. e’ uno dei principali rivenditori di componenti elettronici ed elettrodomestici in Veneto, in forte ascesa anche nelle regioni confinanti come la Lombardia e l’Emilia Romagna. La basi di dati dialoga con un front end che permette ad una persona di registrarsi al sito. Ogni cliente ha la possibilità di aggiungere prodotti al proprio carrello ed effettuare ordini, ciascuno dei quali,una volta evaso e arrivato all’indirizzo indicato dal cliente attraverso un corriere di diverse tipologie di velocità di spedizione, genera una fattura associata. A causa di problemi legati alla logistica di più sedi, dovuti alla precedente mancanza di un adeguato sistema di controllo degli ordini, si è vista necessaria la creazione di un sistema che aiuti quindi a gestire le scorte in magazzino, gli acquisti dal sito on-line del gruppo, la lista delle fatture, le spedizioni ed i corrieri associati ai vari ordini.

**Analisi dei requisiti**

**Descrizione Testuale dei Requisiti**

Il sistema permette di gestire in maniera semplice la gestione di ogni sede del gruppo Cometa, e permette di effettuare ordini direttamente dal sito della filiale in maniera semplice e veloce.

Viene permesso agli utenti (clienti) di realizzare ordini, previa registrazione al sito. Un cliente è descritto da:

* Indirizzo e-mail, univoco per ogni cliente
* Password
* Nome e cognome
* Numero di telefono
* Indirizzo di spedizione (composto di città, via, numero civico e CAP)
* Amico

Un ordine, realizzato e salvato da un cliente tramite il sito web della società durante il processo d’acquisto, è caratterizzato da:

* Codice identificativo
* Data
* Tipo di pagamento

Ogni ordine eseguito genera una fattura, a sua volta caratterizzata da:

* Codice identificativo
* Importo totale
* Data

L’ordine è poi associato ad un corriere, che si occupa di consegnare i prodotti ordinati dal cliente all’indirizzo indicato. Un corriere può essere un corriere normale oppure un corriere espresso. Un corriere è caratterizzato da:

* Targa, univoca per ogni corriere

Il corriere si occupa quindi della spedizione legata all’ordine, questa caratterizzata da:

* Data di arrivo

Ogni prodotto venduto dalla società, conservato in un magazzino, è dotato di e contraddistinto da:

* Codice identificativo
* Nome
* Costo
* Produttore
* Categoria

Prima dell’acquisto I prodotti vengono raggruppati dall’utente in un carrello, il quale può essere o il carrello personale del cliente oppure il carrello di un amico. Un carrello è contraddistinto da:

* Codice identificativo
* Costo totale

L’inclusione di un prodotto in un carrello è caratterizzata da:

* Quantità di un prodotto incluso

I prodotti vengono normalmente conservati in magazzini, situati in varie città sul territorio internazionale, ognuno caratterizzato da:

* Codice identificativo
* Città in cui ha sede

La giacenza di un prodotto in un magazzino è contraddistinta da:

* Quantità rimanente di un prodotto

**Glossario**

* **Prodotto:** Una qualsiasi merce ed un qualsiasi oggetto venduto da Cometa S.P.A. e acquistabile tramite il sito web della società
* **Magazzino:** Luogo in cui la merce venduta da Cometa S.P.A. è conservata e da cui è spedita una volta acquistata
* **Carrello:** Collezione di prodotti che un cliente acquista o intende acquistare, scelti dal catalogo mediante il sito web della società
* **Amico:** Cliente associato ad un altro cliente, il quale può scegliere di inviare un ordine all’indirizzo dell’amico invece che al proprio durante l’aggiunta di prodotti al carrello
* **Cliente:** Persona registrata al sito web di Cometa S.P.A. che effettua acquisti tramite esso
* **Corriere:** Dato un ordine, si occupa di trasportare la merce acquistata all’indirizzo del cliente che ha effettuato l’acquisto
* **Importo totale**: importo presente in **Fattura** pari al valore del carrello a cui va, eventualmente, sommato un piccolo supplemento in base al tipo di pagamento deciso, da nessuno fino a 20 euro (0 euro attraverso PAYPAL, 10 scegliendo VISA/MASTERCARD e 20 tramite CONTRASSEGNO)
* **Consegna:** Il processo di trasporto tramite corriere della merce dal magazzino in cui è conservata all’indirizzo del cliente che ha compiuto l’acquisto
* **Spedizione:** Consegna
* **Corriere Espresso:** Un corriere che si occupa specificamente di consegne espresso
* **Consegna Espresso:** Una consegna con un tempo di spedizione garantito al di sotto di cinque giorni lavorativi

**Operazioni Tipiche**

* Ricerca di un prodotto:
  + Lettura
  + 50’000 al giorno
* Aggiunta di un prodotto ad un carrello:
  + Scrittura
  + 20’000 al giorno
* Creazione di un carrello:
  + Scrittura
  + 10’000 al giorno
* Controllo dati di un carrello:
  + Lettura
  + 15’000 al giorno
* Creazione di un ordine:
  + Scrittura
  + 5’000 al giorno
* Consultazione fattura:
  + Lettura
  + 15’000 al giorno
* Verifica della disponibilità di un corriere:
  + Lettura
  + 2’000 al giorno
* Completamento di un ordine, con conseguente generazione fattura:
  + Scrittura
  + 5’000 al giorno
* Registrazione di un nuovo cliente:
  + Scrittura
  + 300 al giorno
* Annullamento di un ordine:
  + Scrittura
  + 100 al giorno
* Creazione di un nuovo prodotto / Eliminazione di un prodotto esistente:
  + Scrittura
  + 5 al giorno

**PROGETTAZIONE CONCETTUALE**

**Lista Entità**

Il Database si compone delle tabelle qui riportate. Gli attributi sottolineati costituiscono gli identificatori delle entità. I campi sono NOT NULL tranne quando specificato dal simbolo ‘\*’.

* **Cliente:** rappresenta un cliente della società Cometa
  + Email: varchar(50)
  + Nome: varchar(30)
  + Cognome: varchar(30)
  + Password\_utente: varchar(30)
  + Indirizzo: attributo composto da:
    - Città: varchar(20)
    - Via: varchar(30)
    - Civico: varchar(3)
    - CAP: varchar(5)
  + Cellulare: varchar(9)\*
  + Amico: varchar(50)\*
* **Prodotto:** rappresenta un prodotto venduto tramite il sito web della società
  + Codice\_prodotto: integer
  + Nome: varchar(30)
  + Categoria: varchar(20)
  + Prezzo: numeric(7,2)
  + Produttore: varchar(20)
* **Magazzino:** rappresenta un magazzino nel quale sono conservati i prodotti venduti dalla società
  + Codice\_magazzino: integer
  + Città: varchar(20)
* **Corriere:** rappresenta un corriere che si occupa di effettuare le consegne legate agli ordini
  + Targa: varchar(8)
* **Corriere Normale:** rappresenta un corriere che si dedica a consegne normali
* **Corriere Espresso:** rappresenta un corriere dedito a consegne espresso
* **Ordine:** rappresenta un ordine di merce effettuato da un cliente
  + Codice\_ordine: integer
  + Data: date
  + Tipo\_pagamento: enum, che può avere valore “VISA” o “MASTERCARD” o “CONTRASSEGNO” oppure “PAYPAL”
* **Carrello:** rappresenta il carrello di prodotti selezionati da un cliente per l’acquisto
  + Codice\_carrello: integer
  + Totale: numeric(8,2)
* **Carrello Personale:** rappresenta il carrello personale di un cliente
* **Carrello Amico:** rappresenta il carrello di un amico
* **Fattura:** rappresenta una fattura emessa a seguito di un ordine
  + Codice\_fattura: integer
  + Data: date
  + Totale: numeric(8,2)

**Lista Relazioni**

Prodotto - Magazzino: **RIMANENZA**

* Un magazzino può contenere un prodotto o più di uno, oppure nessuno. **(0,N)**
* Un prodotto può essere conservato in un magazzino o più, o in nessuno. **(0,N)**
* Attributi:
  + Numero\_pezzi: smallint

Prodotto - Carrello: **INCLUSIONE**

* Un prodotto può essere incluso in un carrello o più, oppure in nessuno **(0,N)**
* Un carrello contiene almeno un prodotto al suo interno **(1,N)**
* Attributi:
  + Quantità: smallint

Carrello - Ordine: **INVIO**

* Un carrello si risolve in un solo acquisto e quindi invia un solo ordine **(1,1)**
* Un ordine è associato ad un solo acquisto e quindi ad un solo carrello **(1,1)**

Ordine - Fattura: **EROGAZIONE**

* Un ordine emette una sola fattura una volta portato a termine e consegnato **(0,1)**
* Una fattura viene erogata a seguito di uno ed un solo ordine **(1,1)**

Ordine - Corriere: **SPEDIZIONE**

* Un ordine una volta completato è affidato ad un singolo corriere, se disponibile **(0,1)**
* Un corriere può essere assegnato alla spedizione di un ordine o più, oppure nessuno **(0,N)**
* Attributi:
  + Data\_spedizione: date

Cliente - Ordine: **SALVATAGGIO**

* Un cliente può avere all’attivo un ordine o più di uno, oppure nessuno. **(0,N)**
* Un ordine è salvato da uno e un solo cliente. **(1,1)**

Cliente - Ordine: **ESECUZIONE**

* Un cliente può eseguire un ordine o più alla volta, oppure nessuno **(0,N)**
* Un ordine è associato ad uno ed un solo cliente **(1,1)**

**Altri Vincoli**

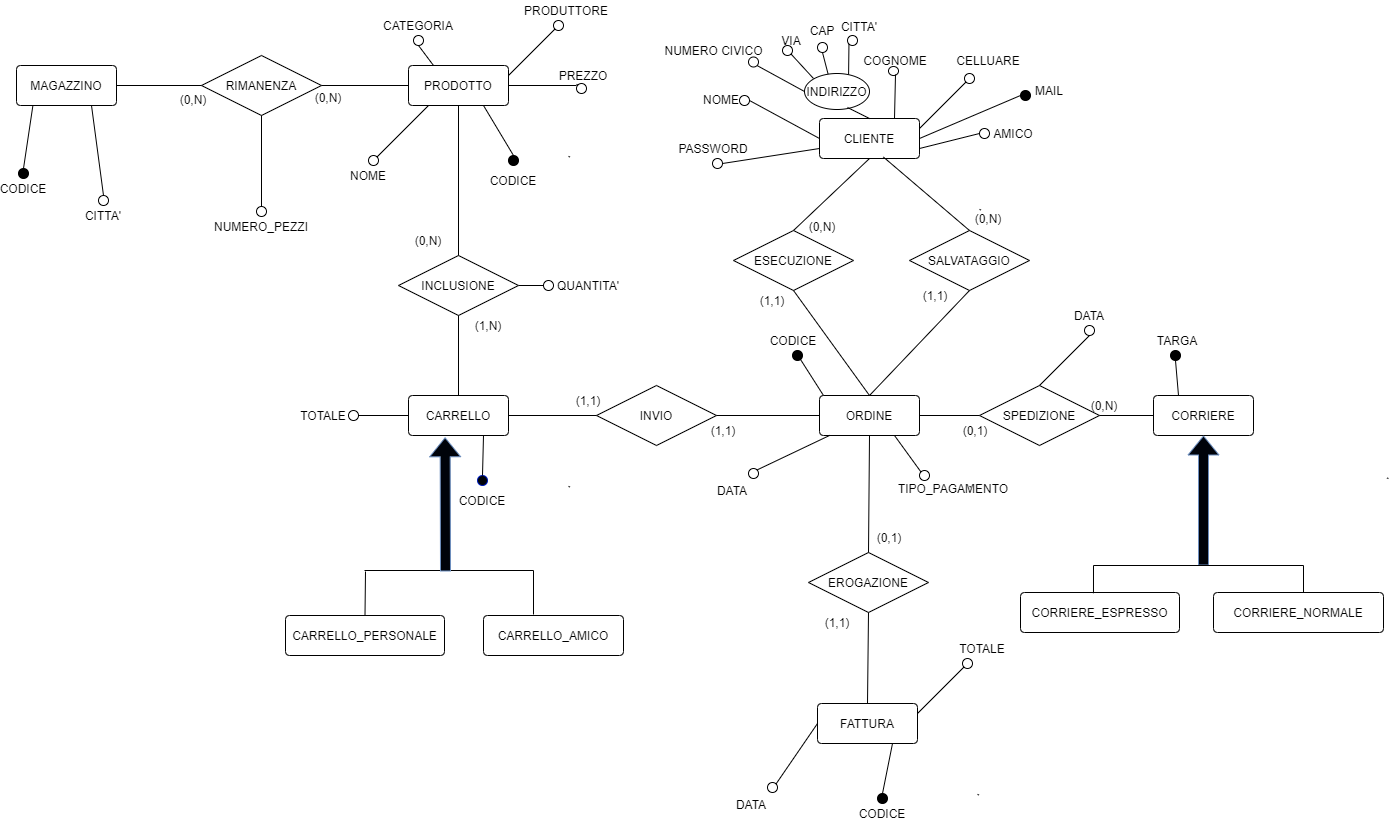
* Un cliente riceve solamente spedizioni che hanno come punto di partenza un magazzino situato nella stessa città indicata all’interno del suo profilo
* Per ricevere una consegna tramite corriere espresso è necessario aver fatto un ordine con metodo di pagamento non PAYPAL

**Lista Generalizzazioni**

**Carrello** è una generalizzazione totale di **Carrello Personale** e **Carrello Amico**

**Corriere** è una generalizzazione totale di **Corriere Normale** e **Corriere Espresso**

**Schema Concettuale**

****

**PROGETTAZIONE LOGICA**

**Ristrutturazione dello schema**

**Analisi delle ridondanze**

**1)** Osservando lo schema concettuale si nota come l’attributo *Totale*, associato all’entità **Carrello**, potrebbe essere calcolato tramite accesso alla relazione **Inclusione** e all’entità **Prodotto**. *Totale* rappresenta infatti la somma di tutti gli attributi *Prezzo* delle entità **Prodotto** incluse in un **Carrello**. Si analizzano quindi le operazioni riguardanti questo attributo per capire se vada eliminato o meno.

* **Operazione 1:** Aggiunta di un prodotto ad un carrello:
  + Scrittura
  + 20’000 al giorno
* **Operazione 2:** Controllo dati di un carrello:
  + Lettura
  + 15’000 al giorno
* **Operazione 3:** Controllo dell’importo richiesto a seguito della generazione di un ordine:
  + Lettura
  + 5’000 al giorno

È noto inoltre inoltre che un **Carrello** è in media in relazione con 2 entità **Prodotto** (20’000 aggiunte di un **Prodotto** ad un **Carrello** al giorno / 10’000 entità **Carrello** create ogni giorno, si noti che la creazione di un **Carrello** implica l’aggiunta di un **Prodotto** ad esso). Questo si giustifica nel fatto che una considerevole parte degli acquisti vengono eseguiti su un singolo **Prodotto**.

**Con ridondanza:**

**Operazione 1:** L’aggiunta di un **Prodotto** ad un **Carrello** genera una variazione su *Totale*, quindi 1 accesso in scrittura su **Carrello**, e un accesso in lettura su **Prodotto** (non si considerano gli accessi in lettura necessari a stabilire la relazione tra le entità e gli accessi in scrittura sulla tabella della loro relazione, perchè indipendenti dalla presenza o meno dell'attributo). 1 x 20’000 + 1 x 2 x 20’000 = 60’000

**Operazione 2:** Il controllo dei dati di un **Carrello** causa su di esso 1 accesso in lettura. 1 x 15’000 = 15’000

**Operazione 3:** Il controllo dell’importo causa 1 accesso in lettura su **Carrello**. 1 x 5’000 = 5’000

**Costo giornaliero complessivo:** 60’000 + 5’000 + 15’000 = 80’000

**Senza ridondanza:**

**Operazione 1:** In assenza di un campo *Totale*, l’aggiunta di un **Prodotto** non genera su **Carrello** accessi in scrittura e non genera accessi in lettura su **Prodotto** (non si considerano gli accessi in lettura necessari a stabilire la relazione tra le entità e gli accessi in scrittura sulla tabella della loro relazione, perchè indipendenti dalla presenza o meno dell'attributo). Il costo è quindi 0.

**Operazione 2:** Il controllo dei dati di un **Carrello** causa 1 accesso in lettura su di esso, e mediamente 2 accessi in lettura su **Inclusione** e 2 accessi in lettura su **Prodotto**. (1 + 2 + 2) x 15’000 = 75’000

**Operazione 3:** Il controllo dell’importo causa 1 accesso in lettura su **Carrello**, e mediamente 2 accessi in lettura su **Inclusione** e 2 accessi in lettura su **Prodotto**. (1 + 2 + 2) x 5’000 = 25’000

**Costo giornaliero complessivo:** 25’000 + 75’000 + 0 = 100’000

**Conclusioni:**

Il costo complessivo richiesto da dover accedere singolarmente ai prodotti per calcolare *Totale* quando richiesto risulta maggiore del costo complessivo per mantenere aggiornato l’attributo (100’000 > 80’000). Si sceglie quindi di mantenere l’attributo *Totale* nell’entità **Carrello** e di conseguenza la ridondanza risultante.

**2)** Osservando lo schema si nota inoltre che l’attributo *Data*, associato all’entità **Fattura**, potrebbe essere ricavato tramite accesso alla relazione **Erogazione** collegata all’entità **Ordine** e alla relazione **Spedizione**. *Data* coincide infatti con l’attributo omonimo *Data* della relazione **Spedizione** collegata a **Ordine**. Si analizzano quindi le operazioni riguardanti questo attributo per capire se sia opportuna la sua eliminazione.

* **Operazione 1:** Erogazione di una fattura:
  + Scrittura
  + 5’000 al giorno
* **Operazione 2:** Consultazione dei dati di una fattura:
  + Lettura
  + 15’000 al giorno

È noto che ogni **Fattura** è in relazione con un solo **Ordine**, associato ad una sola **Spedizione**.

**Con ridondanza:**

**Operazione 1:** L’erogazione di una **Fattura** genera per la creazione dell’attributo *Data* 1 accesso in lettura su **Ordine**, 1 accesso in lettura su **Spedizione** e 1 accesso in scrittura su **Fattura** (si ignorano accessi in lettura e scrittura indipendenti dall’esistenza dell’attributo). (1 + 1) x 5’000 + 1 x 2 x 5’000 = 20’000

**Operazione 2:** Il controllo dei dati della **Fattura** causa 1 accesso in lettura. 1 x 15’000 = 15’000

**Costo giornaliero complessivo:** 20’000 + 15’000 = 35’000

**Senza ridondanza:**

**Operazione 1:** In assenza di un campo *Data*, l’erogazione di una **Fattura** non genera ulteriori accessi né in lettura né in scrittura (si ignorano accessi in lettura e scrittura indipendenti dall’esistenza dell’attributo). Il costo è quindi 0.

**Operazione 2:** Il controllo dei dati della **Fattura** causa 1 accesso in lettura su **Fattura**, 1 accesso in lettura su **Erogazione** e 1 accesso in lettura su **Spedizione** (non sono necessari accessi a **Ordine**, è sufficiente confrontare i codici identificativi tra le relazioni). (1 + 1 + 1) x 15’000 = 45’000

**Costo giornaliero complessivo:** 45’000 + 0 = 45’000

**Conclusioni:**

Il costo complessivo richiesto se si desidera calcolare *Data* quando richiesto risulta maggiore del costo complessivo per mantenere aggiornato l’attributo (45’000 > 35’000). Si sceglie quindi di mantenere la ridondanza e quindi l’attributo *Data* nell’entità **Fattura**.

**Eliminazione delle generalizzazioni**

Lo schema E-R contiene 2 generalizzazioni totali ed esclusive, **Carrello** e **Corriere**. Data l’impossibilità di rappresentare propriamente una generalizzazione tramite un sistema tradizionale, entrambe dovrebbero essere trasformate in entità e relazioni. Viene deciso per entrambe di accorpare le classi figlie nell'entità padre.

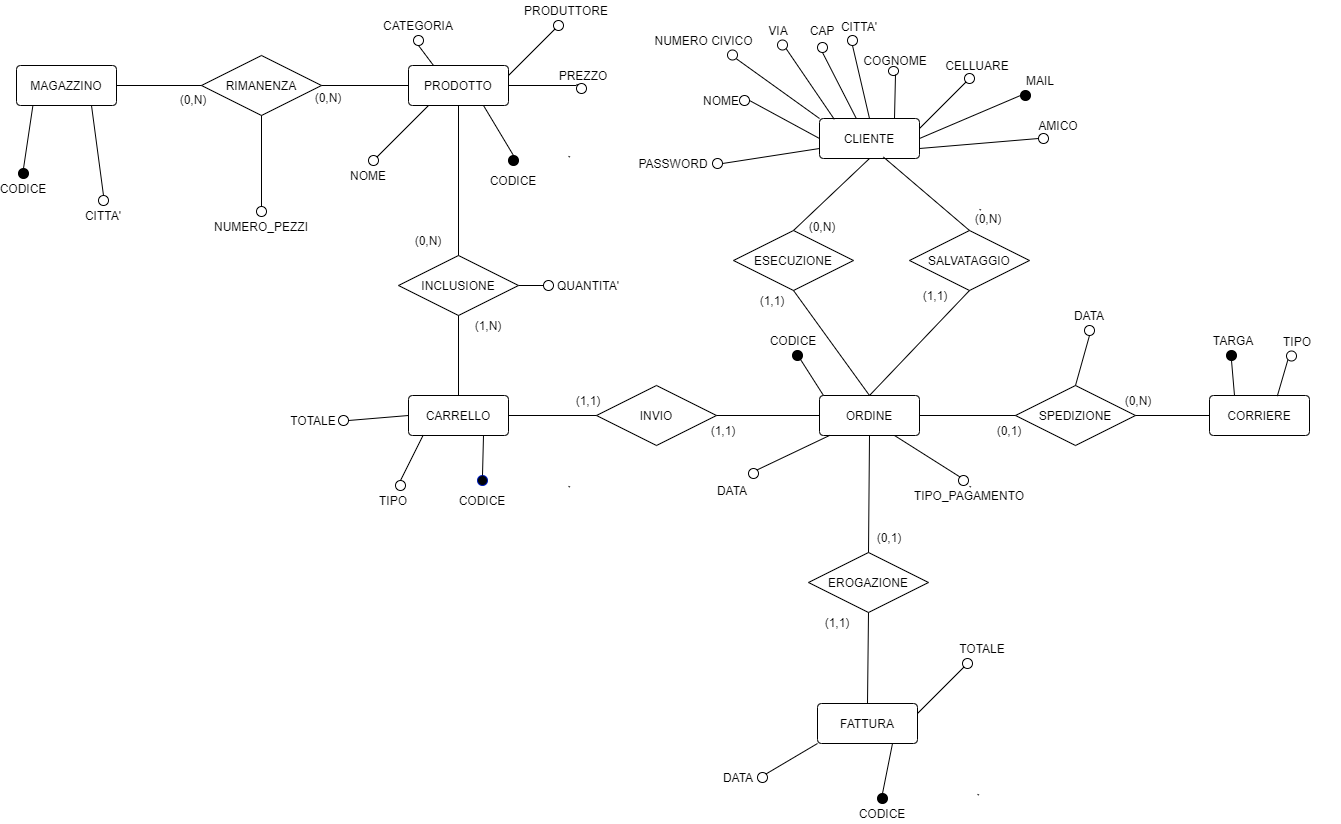
**Carrello:** Si sceglie di accorpare le entità figlie **Carrello personale** e **Carrello amico** all’interno dell’entità padre **Carrello**. Questo perché le operazioni sulle due entità sono le stesse, le entità hanno gli stessi attributi, lo stesso identificativo, e le loro differenze sono limitate all’essere un tipo di carrello o l’altro. Viene aggiunto un attributo *Tipo* all’entità **Carrello** per differenziare i due tipi di entità ora entrambi parte di **Carrello**.

**Corriere:** Si sceglie di accorpare le entità figlie **Corriere normale** e **Corriere espresso** all’interno dell’entità padre **Corriere**. Questo perché le operazioni sulle due entità sono le medesime, entrambe hanno lo stesso identificativo, e sono differenti solo nell’essere un tipo di corriere o l’altro. Viene aggiunto un attributo *Tipo* nell’entità **Corriere** per differenziare le due entità ora entrambe parte di **Corriere**.

**Eliminazione degli attributi composti**

L’entità cliente al suo interno ha l’attributo composto*Indirizzo*che non è rappresentabile in una basi di dati, quindi viene scomposta e vengono creati, rispettivamente, gli attributi: *Città*, *Via*, *Numero civico*e*Cap*.

**Diagramma schema ristrutturato**

****

**Schema relazionale**

Descrizione schema relazionale ed eventuali vincoli di integrità referenziale

**N.B.: il simbolo ”**\*\***” rappresenta la possibilità per un attributo di avere valore nullo**

**CLIENTE**(**email**, password, nome, cognome, cellulare\*\*, città, via, civico, cap, amico\*\*)

- L'attributo amico ha un vincolo di integrità referenziale con l’attributo email

**MAGAZZINO**(**codice\_magazzino**, città)

**CORRIERE**(**targa**, Tipo)

**PRODOTTO**(**codice\_prodotto**, nome, categoria, prezzo, produttore)

**CARRELLO**(**codice\_carrello**, totale, tipo\_carrello)

**ORDINE**(**codice\_ordine**, codice\_carrello\_ordine, codice\_cliente\_ordine, data\_ordine, tipo\_pagamento)

- L'attributo codice\_carrello\_ordine ha un vincolo di integrità referenziale con l’attributo codice\_carrello della relazione Carrello

- L'attributo codice\_cliente\_ordine ha un vincolo di integrità referenziale con l’attributo email della relazione Cliente

**INCLUSIONE**(**Codice\_carrello\_inclusione**, **codice\_prodotto\_inclusione**, quantità)

- L'attributo Codice\_carrello\_inclusione ha un vincolo di integrità referenziale con l’attributo codice\_carrello della relazione Carrello

- L'attributo codice\_prodotto\_inclusione ha un vincolo di integrità referenziale con l’attributo codice\_prodotto della relazione Prodotto

**RIMANENZA**(**codice\_prodotto\_rimanenza, codice\_magazzino\_rimanenza**, numero\_pezzi)

- L'attributo codice\_prodotto\_rimanenza ha un vincolo di integrità referenziale con l’attributo codice\_prodotto della relazione Prodotto

- L'attributo codice\_magazzino\_rimanenza ha un vincolo di integrità referenziale con l’attributo codice\_magazzino della relazione Magazzino

**FATTURA**(**codice\_fattura**, codice\_ordine\_fattura, data\_emissione, totale)

- L'attributo codice\_ordine\_fattura ha un vincolo di integrità referenziale con l’attributo codice\_ordine della relazione Ordine

- L'attributo data\_emissione ha un vincolo di integrità referenziale con l’attributo data\_avvenuta\_spedizone della relazione Spedizione

**SPEDIZIONE**(**codice\_ordine\_spedizione**, targa\_spedizione, data\_avvenuta\_spedizone\*\*)

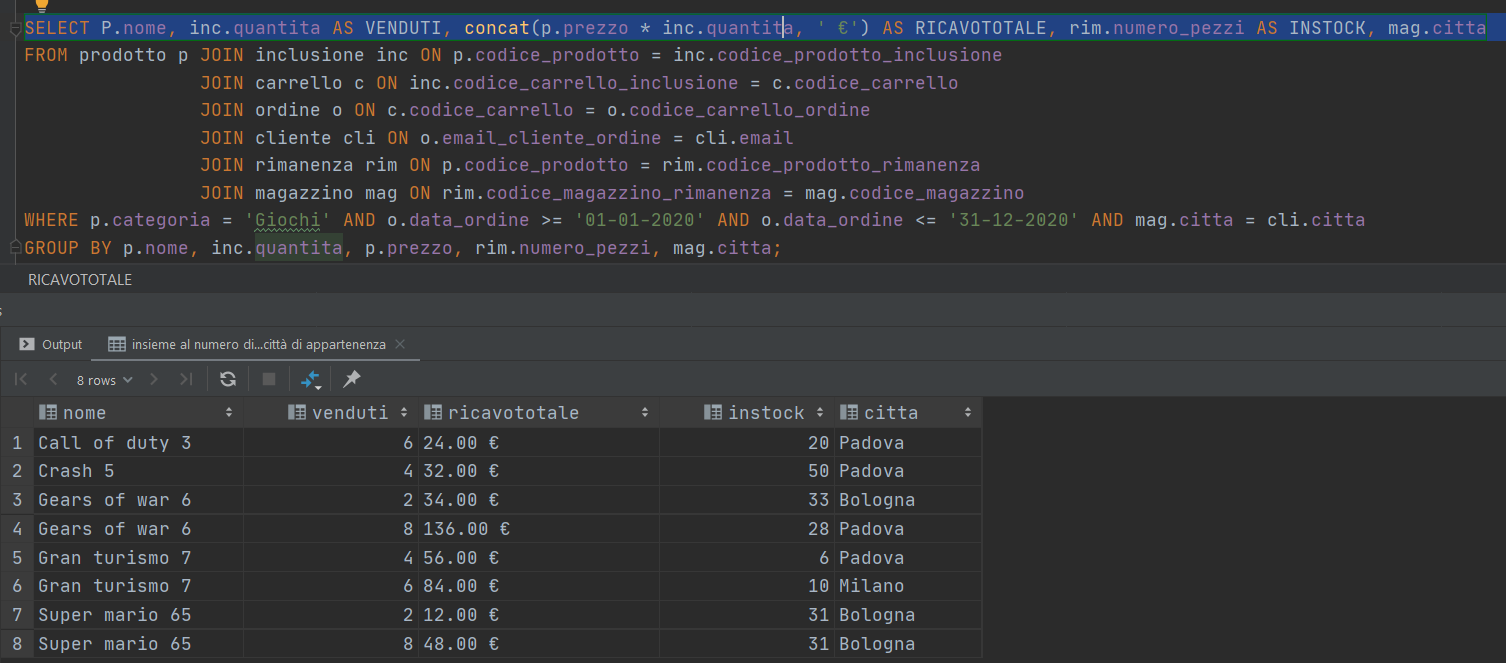
- L'attributo codice\_ordine\_spedizione ha un vincolo di integrità referenziale con l’attributo codice\_ordine della relazione Ordine

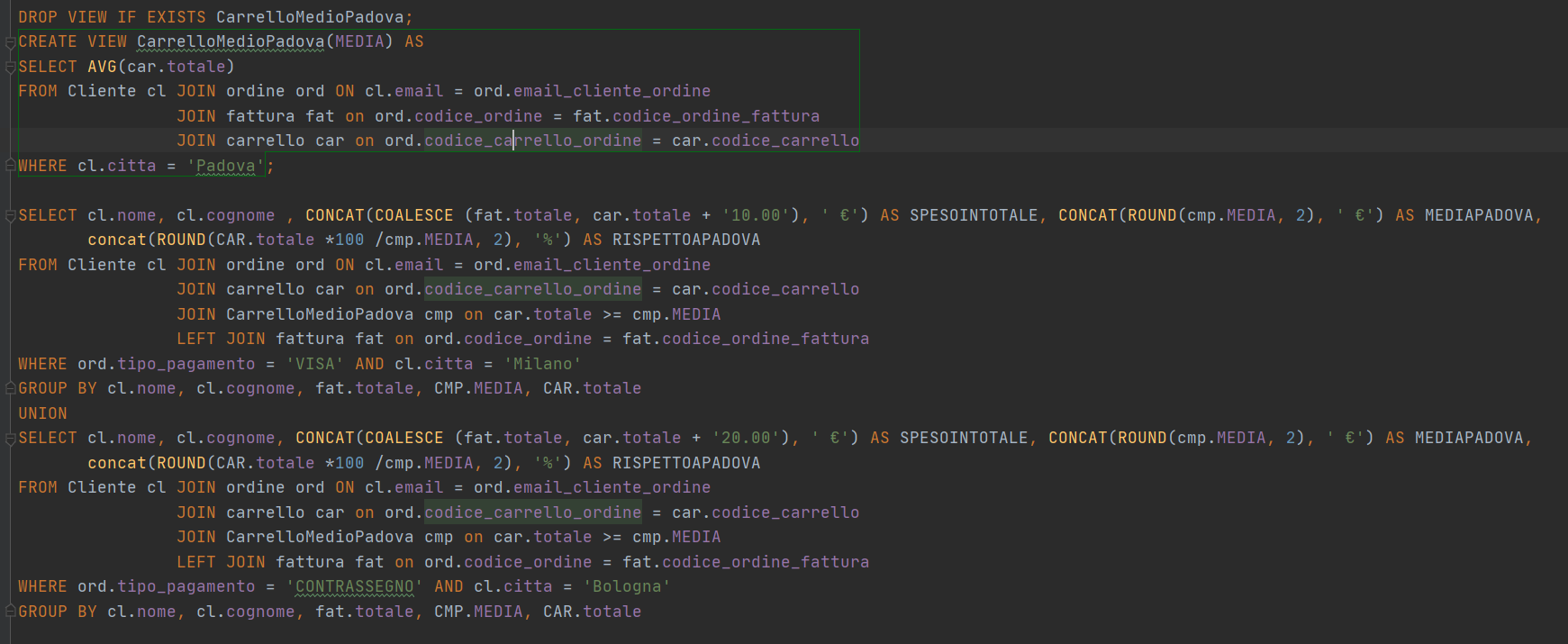
- L'attributo targa\_spedizione ha un vincolo di integrità referenziale con l’attributo targa della relazione Corriere

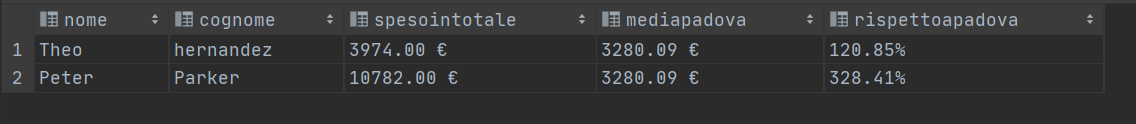
**QUERY & INDICI**

**Query**

1. **Mostrare i giochi acquistati nel 2020 e il loro ricavo totale, insieme al numero di pezzi rimasti dal magazzino acquistato e le loro relativa città di appartenenza**

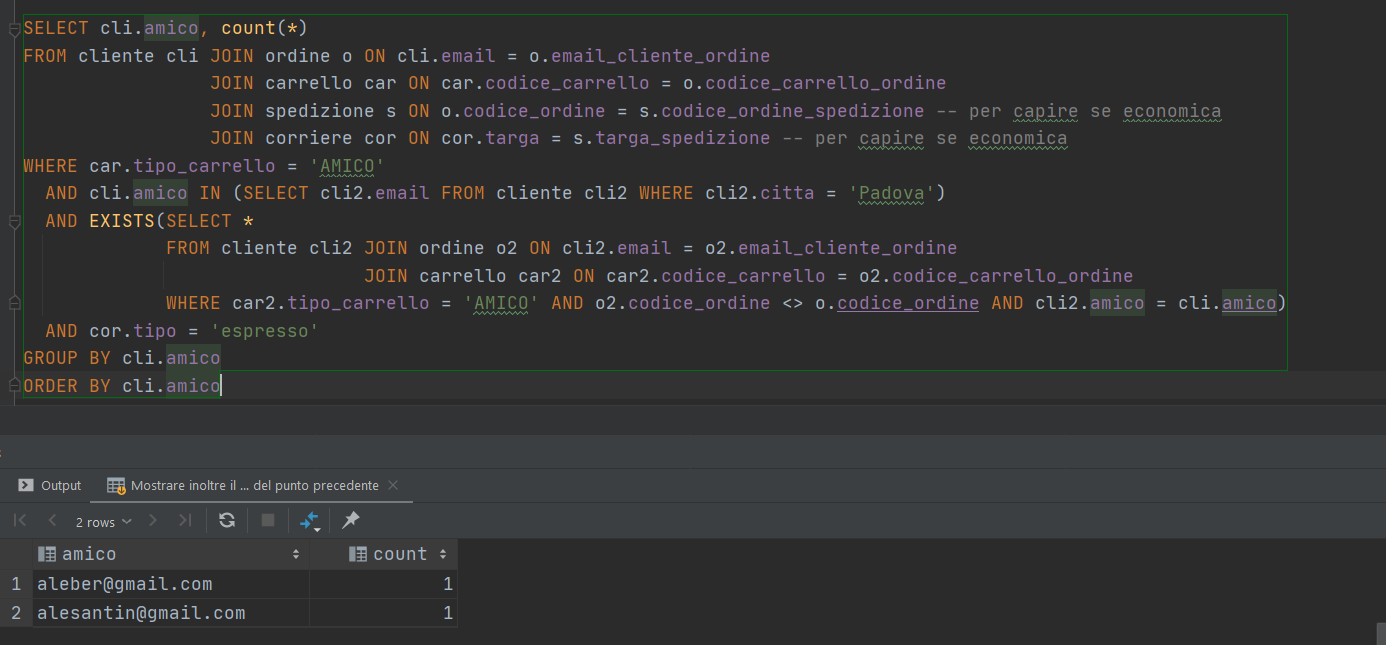


2. **Mostrare il nome, il cognome e il totale speso in fattura (se non ancora emessa, indicare il valore in carrello calcolando però il costo della spedizione) dei clienti residenti a Milano e Bologna che hanno rispettivamente effettuato almeno un ordine con pagamento tramite VISA e CONTRASSEGNO, il cui costo sia maggiore del prezzo medio degli ordini dei clienti di Padova. Infine mostrare la loro percentuale di spesa in carrello rispetto alla media padovana**

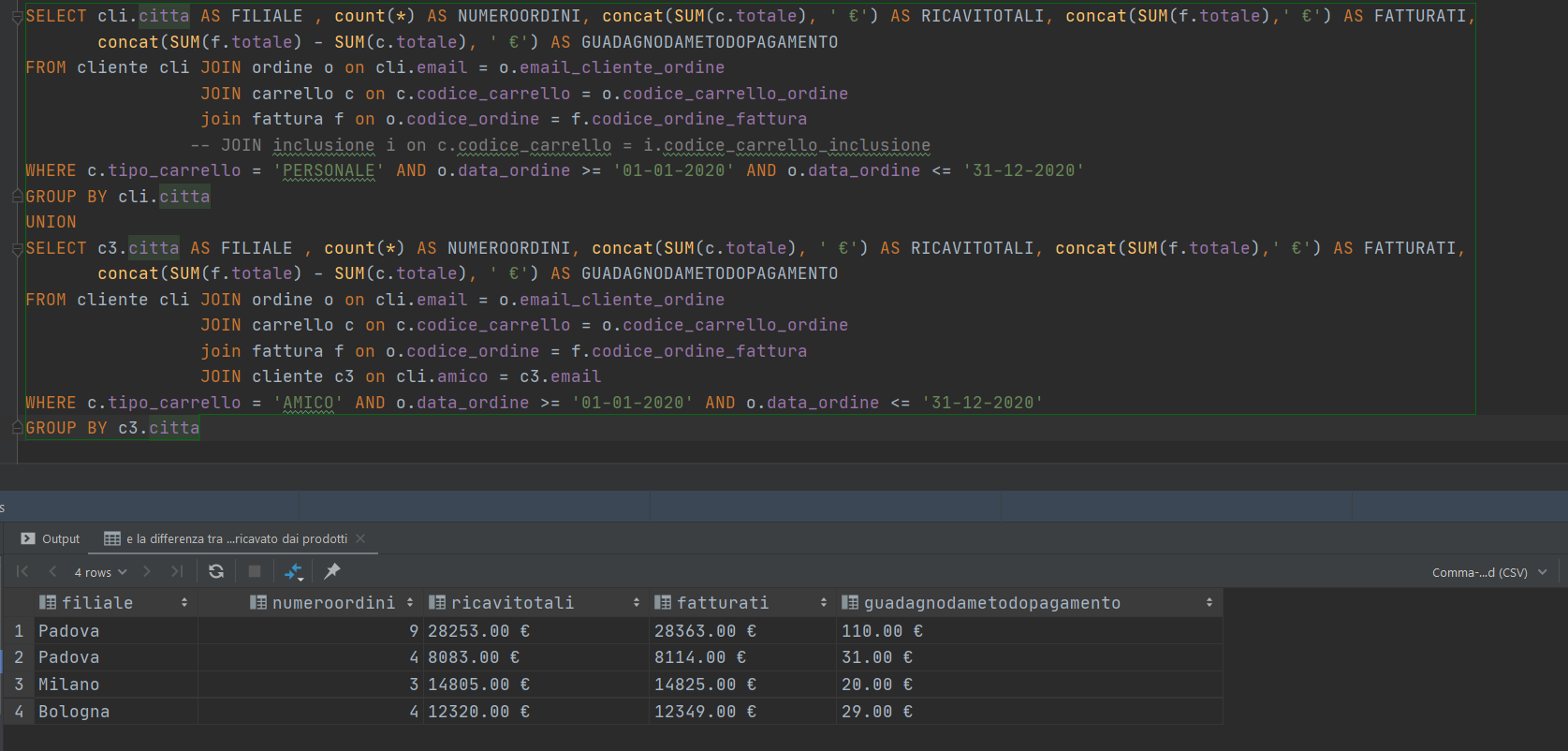


3. **Mostrare l'e-mail di tutti i clienti padovani, in ordine alfabetico, solo se hanno ricevuto almeno 2 regali e almeno uno di essi tramite corriere espresso.**

**Inoltre, mostrare il numero di regali tramite corriere espresso ricevuti dagli utenti del punto precedente**

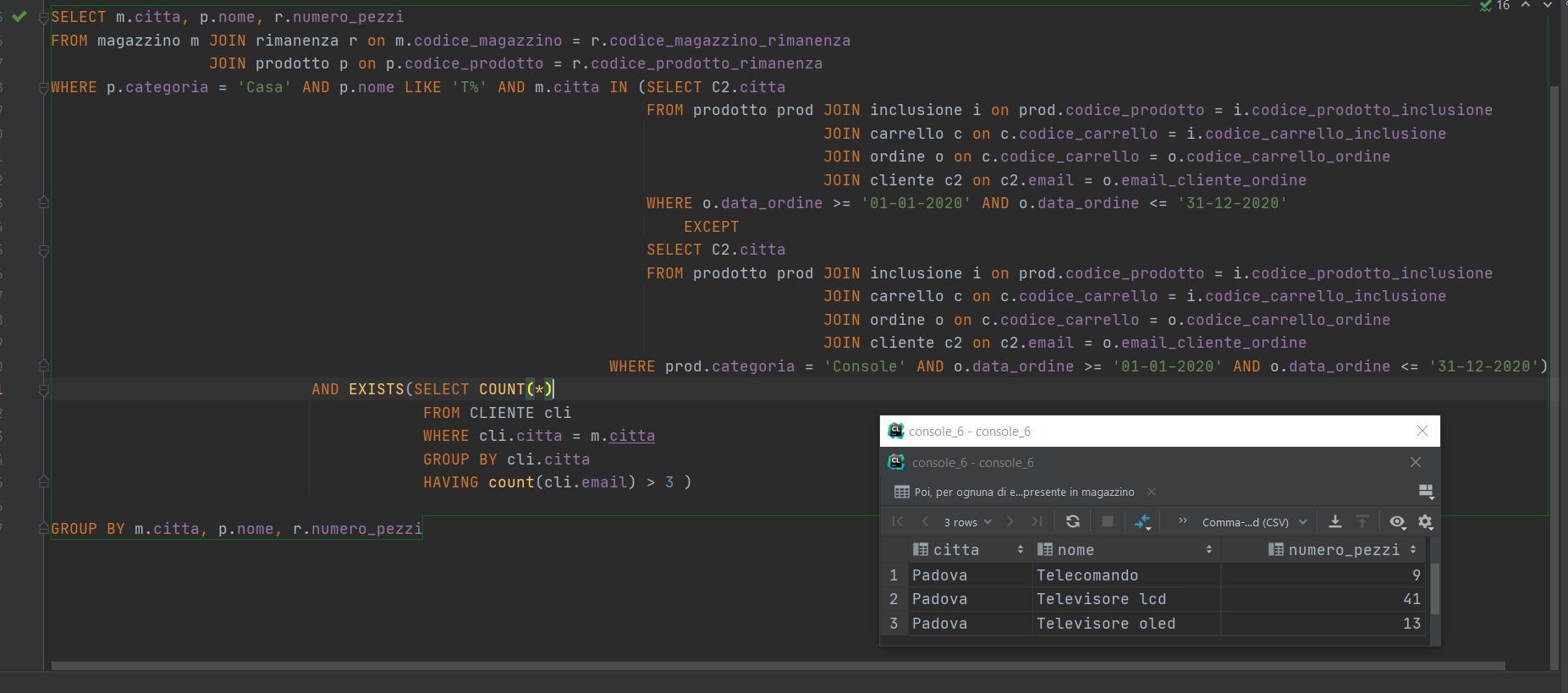


4. **Mostrare per ogni filiale il numero di ordini ricevuti, il ricavo totale, quanto hanno fatturato nel 2020 e la differenza tra fatturato e ricavato dai prodotti**

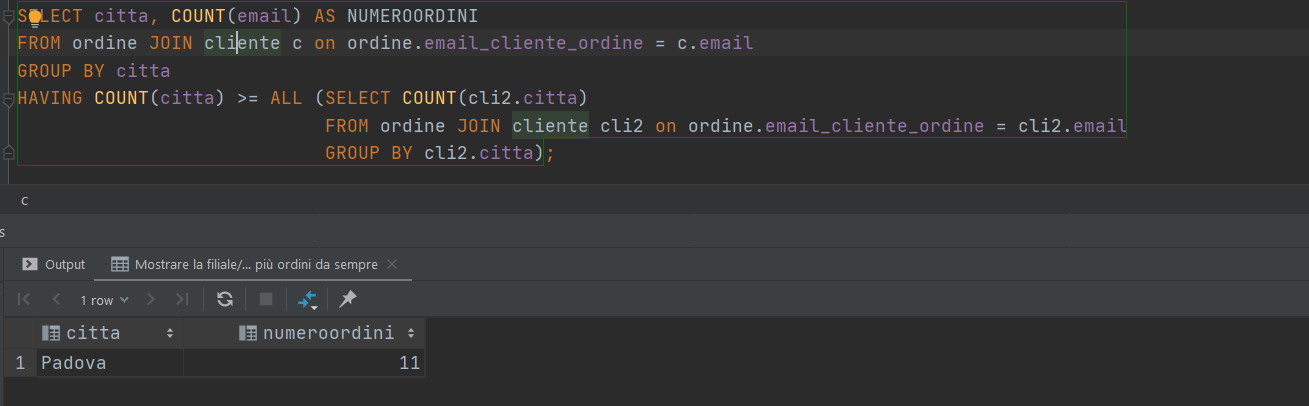
****

5**. Mostrare le filiale che hanno venduto almeno una console nel 2020 e hanno più di 3 utenti associati.**

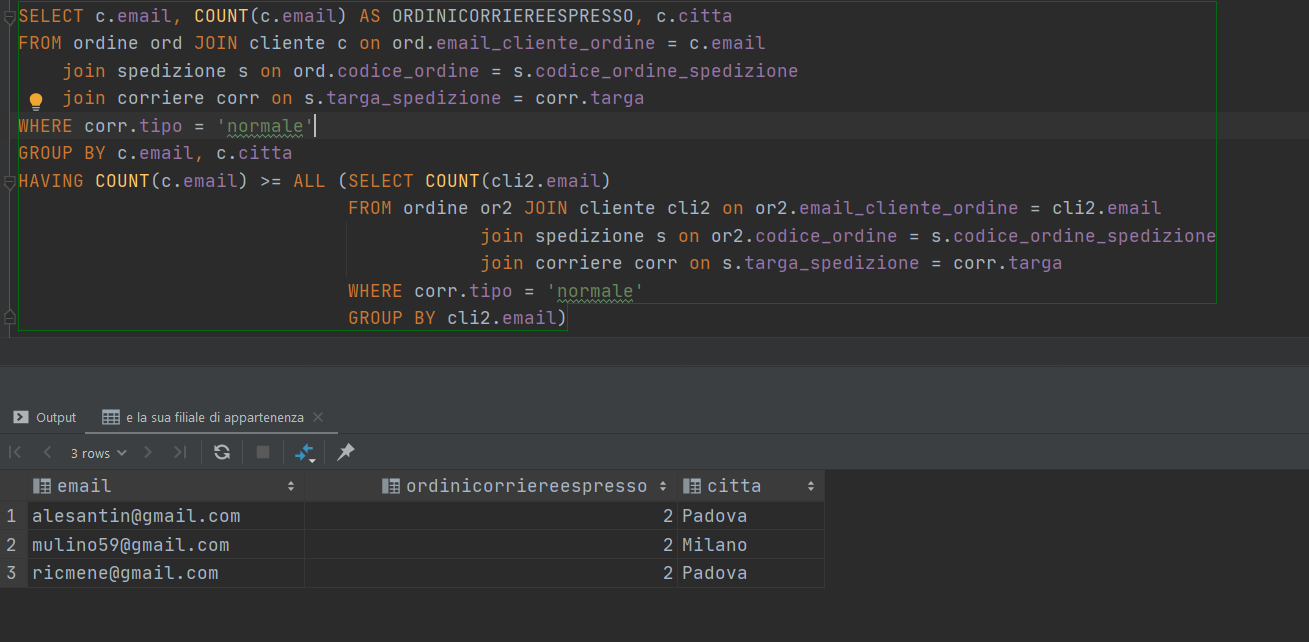
**Poi, per ognuna di esse, elencare i prodotti per la casa cui nomi iniziano con la lettera “T” e la loro quantità presente in magazzino**



6. **Mostrare la filiale/i (i in caso di numero di ordini uguali) che ha eseguito più ordini “life to date”**



7. **Mostrare il cliente/i (i in caso di numero di ordini uguali) e la sua filiale di appartenenza che ha eseguito più ordini da sempre usando il corriere espresso**



**Indici**

**Precisazione necessaria: Il nostro DB viene gestito da Postgre, un DBMS che crea un indice sulla chiave primaria di ogni tabella in maniera automatica**

Si noti che la tabella Prodotto è formata da relativamente poche righe, per cui la velocità in lettura sarà sempre reattiva. Il suo impiego è però massiccio solo per operazioni in lettura, mentre le operazioni in scrittura sono molto rare a causa dei lunghi cicli di vita sul mercato dei prodotti elettronici. Per questo motivo è ritenuto opportuno creare un indice e viene scelta la coppia di attributi (**categoria, prezzo)** perché il primo è il campo maggiormente usato dal cliente per selezionare la famiglia di prodotti da cercare, mentre il secondo si tara a seconda delle proprie possibilità di spesa.

L’operazione per crearlo è la seguente:

**CREATE INDEX idx\_prodotto ON Prodotto(categoria, prezzo);**

Per lo store manager del punto vendita, la tabella Ordine è molto interessante in caso di reclami o problemi perchè contiene tutti gli ordini effettuati e si interfaccia con la tabella carrello. E’ formata dal proprio codice univoco e tramite la mail del cliente, si possono vedere tutti gli ordini da lui effettuati. La tabella dispone anche di un campo data\_ordine con cui si possono isolare i singoli ordini problematici e facilitare la ricerca (tranne che in rari casi di ordini multipli nelle stesse ore, casistica praticamente inesistente).

Per questo motivo è ritenuto opportuno creare un indice, questa volta di tipo hash sull’attributo (**email\_cliente\_ordine)**

L’operazione per crearlo è la seguente:

**CREATE INDEX idx\_ordine ON Ordine USING hash (email\_cliente\_ordine);**

**File BasiDiDatiMain.cpp**

La consegna contiene un singolo file dal nome BasiDiDatiMain.cpp che contiene l’invocazione sequenziale alle 7 Query mostrate precedentemente. Cambiando i parametri di accesso al database (host, user, password, porta) con i più opportuni alla propria configurazione, si può procedere alla compilazione del file e la creazione dell’eseguibile. In fondo al file è presente il codice, da cui togliere il commento, per bloccare la repentina chiusura dell’applicazione dopo aver eseguito il codice. Le query non sono parametriche, quindi non ci sarà nessuna interazione con l’utente che lancia l’eseguibile.