



Basi di Dati e Conoscenza  
Progetto A.A. 2020/2021

5

LOGISTICA S.R.L.

0269426

Federico Romersì

10

## Indice

|                                   |   |
|-----------------------------------|---|
| 1. Descrizione del Minimondo..... | 3 |
| 2. Analisi dei Requisiti.....     | 4 |
| 3. Progettazione concettuale..... | 5 |
| 15 4. Progettazione logica.....   | 6 |
| 5. Progettazione fisica.....      | 8 |
| Appendice: Implementazione.....   | 9 |

## 1. Descrizione del Minimondo

1 Sistema informativo di gestione delle flotte

2 L'azienda Logistica S.r.l. gestisce una flotta di autoveicoli per effettuare trasporti a livello

3 europeo. La società fonda il suo business sulla trasparenza verso i propri clienti e per questo

4 motivo vuole realizzare un sistema informativo che consenta loro di conoscere, in tempo

5 reale, lo stato delle consegne.

6 La gestione delle consegne espresse di Logistica S.r.l. permette di trasportare a destinazione

7 i pacchi in al più tre giorni. I pacchi spediti dai clienti vengono recuperati direttamente

8 all'indirizzo fornito dal cliente. Un cliente, pertanto, al momento della registrazione nel

9 sistema, fornisce tutte le sue informazioni anagrafiche, indicando un indirizzo di residenza,

10 un indirizzo di fatturazione, un numero arbitrario di indirizzi da cui è possibile recuperare

11 un pacco in spedizione, un numero arbitrario di recapiti (telefono, cellulare, email)

12 indicandone uno come mezzo di comunicazione preferito, ed inserendo i dati relativi alla

13 sua carta di credito (intestatario, numero, data di scadenza, codice CVV).

14 Logistica S.r.l. ha sul territorio differenti centri operativi. Alcuni di questi centri sono dei

15 centri di prossimità, altri sono dei centri di smistamento nazionale, altri sono dei centri di

16 smistamento internazionale. I centri di prossimità hanno a disposizione una flotta di veicoli

17 di piccola dimensione (ciascuno identificato da una targa e da un codice alfanumerico

18 univoco) che si preoccupano di raggiungere i clienti per il ritiro dei pacchi. I centri di

19 smistamento, invece, hanno a disposizione degli autoarticolati che consentono di

20 raggiungere altri centri di smistamento. Un centro di smistamento nazionale può

21 raggiungere unicamente un altro centro di smistamento nazionale. Un centro di

22 smistamento internazionale, invece, può raggiungere centri di smistamento della stessa

23 nazione di appartenenza e tutti gli altri centri di smistamento internazionale. Ciascun centro

24 operativo è identificato da un codice, un indirizzo, coordinate geografiche (latitudine e

25 longitudine), da un recapito telefonico, da un indirizzo email di segreteria e dal nome di un

26 responsabile.

27 Un cliente può, in qualsiasi momento, richiedere la spedizione di un pacco. Al momento

28 della richiesta, il sistema richiede di inserire le informazioni sul destinatario (nome,

29 indirizzo, stato), le dimensioni del pacco che si vuole inviare ed il suo peso. Gli

30 amministratori del sistema forniscono delle stime di prezzo per categorie di pacchi

31 (modificabili in qualsiasi momento dall'amministrazione). In particolare, data la somma

32 delle dimensioni del pacco (lunghezza + larghezza + profondità), questo viene associato ad

una specifica categoria. All'interno della categoria, il prezzo è calcolato in relazione alla classe di peso cui il pacco appartiene.

Il cliente, pertanto, riceve una stima del costo. Questo costo verrà poi confermato dagli addetti alla spedizione, quando il pacco verrà recapitato ad un centro di smistamento nazionale e verrà convertito in un costo effettivo, che verrà poi scalato dalla carta di credito dell'utente.

Alla ricezione dell'ordine, il cliente riceve un codice di consegna che potrà utilizzare per tracciare il suo pacco, dal recupero presso l'indirizzo di partenza da lui fornito fino alla consegna. Inoltre, il sistema informativo determina, in funzione delle coordinate geografiche del mittente, qual è il centro di prossimità che si occuperà del recupero del pacco. In caso di spedizione nazionale, il sistema determinerà anche (in funzione delle coordinate del destinatario) qual è il centro di smistamento nazionale che si occuperà della parte finale della consegna. In caso di spedizione internazionale, il sistema assegnerà la consegna al centro di smistamento internazionale più vicino.

Automaticamente, il sistema assegna il recupero della spedizione ad uno dei veicoli del centro di prossimità. Ciascun veicolo ha una capienza massima di pacchi, pertanto verrà selezionato un veicolo con ancora spazio sufficiente. Un veicolo ha un valore volumetrico (inteso come somma delle dimensioni dei pacchi) massimo che viene utilizzato per questo controllo e l'assegnazione. Se non è possibile assegnare un veicolo ad un pacco, questo viene messo in stato di "attesa di recupero" dal sistema. Il giorno successivo, tutti i pacchi in attesa di recupero vengono automaticamente assegnati ai veicoli del centro, secondo la stessa i codici di spedizione. Il sistema, automaticamente, assegna il pacco alla fase successiva della spedizione (ad es, da un centro locale al destinatario, da un centro locale al centro di smistamento nazionale, dal centro di smistamento nazionale al centro di smistamento internazionale, ecc.) sempre secondo la stessa logica.

Viene registrata sempre la data e l'ora in cui un pacco entra in una fase successiva (ad esempio, "raggiunto centro di smistamento nazionale XXX", oppure "in attesa di recupero", oppure "in attesa di smistamento presso XXX", oppure "in consegna", oppure "consegnato", oppure "in transito").

Il cliente ha, in ogni momento, la possibilità di visualizzare lo stato del pacco, fornendo il suo codice di vettura. Inoltre, tutti i veicoli di Logistica S.r.l. sono equipaggiati di dispositivo GPS che, ogni 5 secondi, comunica al sistema la propria posizione. Quando un pacco è "in transito", il cliente ha la possibilità di visualizzare la latitudine e la longitudine

del veicolo che sta trasportando il suo pacco.

Gli amministratori della società generano su base giornaliera dei report indicanti quante spedizioni sono state accettate e quante spedizioni sono state consegnate. Inoltre, per tutte le spedizioni consegnate, il report fornisce informazioni sulle entrate totali (a valle dell'aggiornamento del prezzo effettivo da parte degli operatori dei centri dislocati sul territorio).

Per calcolare la distanza tra due punti espressi in coordinate geografiche (latitudine e longitudine) è possibile utilizzare la seguente formula, dove  $r$  è il raggio della terra:

$$d = 2r \arcsin( \sqrt{ \sin^2( (\phi_2 - \phi_1)/2 ) + \cos(\phi_1) \cos(\phi_2) \sin^2( (\lambda_2 - \lambda_1)/2 ) } )$$

sa logica.

## 2. Analisi dei Requisiti

### Identificazione dei termini ambigui e correzioni possibili

| Linea | Termine                  | Nuovo termine   | Motivo correzione   |
|-------|--------------------------|---|---|
| 8     | Indirizzo                | Coordinate recupero pacco (latitudine e longitudine)          | Il termine indirizzo deve essere inteso come coordinate geografiche |
| 24    | Indirizzo                | Indirizzo C.O. (centro operativo)                             | Omonimia  |
| 29    | Indirizzo                | coordinate consegna pacco (latitudine e longitudine)          | Il termine indirizzo deve essere inteso come coordinate geografiche |
| 40    | Indirizzo partenza di    | Indirizzo recupero pacco                                      | Omonimia  |
| 5     | Consegna                 | Spedizione  | Sinonimo di spedizione  |
| 45    | Consegna                 | Spedizione  | Sinonimo di spedizione  |
| 46    | Consegna                 | Spedizione  | Sinonimo di spedizione  |
| 57    | Centro locale            | Centro di prossimità  | Sinonimo di centro di prossimità                                    |
| 29    | Stato                    | Nazione   | Il termine “stato” è ambiguo  |
| 39    | Codice consegna di       | Codice di spedizione di                                       | Sinonimo di codice di spedizione                                    |
| 38    | Utente                   | Cliente   | Sinonimo di cliente   |
| 42    | Mittente                 | Cliente   | Sinonimo di cliente   |
| 9     | informazioni anagrafiche | informazioni anagrafiche (CF, nome, cognome, data di nascita) | specificazione delle informazioni anagrafiche                       |
| 29    | dimensioni               | dimensioni  | specificazione delle informazioni sulle dimensioni                  |

|    |  |    |  |   |
|----|--|----|--|---|
|    |  |    | (lunghezza, larghezza, profondità)   | del pacco   |
| 36 | centro di smistamento nazionale  | di | centro di prossimità   | in caso di una spedizione locale il pacco non raggiungerebbe mai un centro di smistamento nazionale e quindi non verrebbe calcolato il costo effettivo  |
| 65 | codice di vettura  |    | codice di spedizione   | sinonimo di codice di spedizione  |
| 55 | centro di prossimità   | di | centro operativo   | la scansione del codice di spedizione e la successiva assegnazione alla fase successiva deve essere effettuata ad ogni arrivo di un veicolo in un qualsiasi centro operativo e non solamente nel centro di prossimità |
| 16 | veicolo di piccole dimensioni  | di | furgone  | ambiguità nella classificazione dei veicoli   |
| 20 | Un centro di smistamento nazionale può raggiungere unicamente un altro centro di smistamento nazionale | di | L'organizzazione e dei centri di smistamento è gerarchica: il centro di prossimità si occupa del ritiro e delle consegne e può raggiungere unicamente centri di smistamento nazionali, un centro di smistamento nazionale può raggiungere un altro centro di | ambiguità sulla strutturazione logistica dei centri di smistamento  |

|  |  |  |  |
|--|--|--|--|
|  |  | smistamento nazionale o un centro di smistamento internazionale appartenente alla sua stessa nazione |  |
|--|--|--|--|

### Specifica disambiguata

#### Sistema informativo di gestione delle flotte

L'azienda Logistica S.r.l. gestisce una flotta di autoveicoli per effettuare trasporti a livello europeo. La società fonda il suo business sulla trasparenza verso i propri clienti e per questo motivo vuole realizzare un sistema informativo che consenta loro di conoscere, in tempo reale, lo stato delle consegne.

La gestione delle spedizioni espresse di Logistica S.r.l. permette di trasportare a destinazione i pacchi in al più tre giorni. I pacchi spediti dai clienti vengono recuperati direttamente all'indirizzo di recupero pacco fornito dal cliente. Un cliente, pertanto, al momento della registrazione nel sistema, fornisce tutte le sue informazioni anagrafiche, ovvero il CF, il nome, cognome, anno di nascita; indicando un indirizzo di residenza, un indirizzo di fatturazione, un numero arbitrario di coordinate (latitudine e longitudine), da cui è possibile recuperare un pacco in spedizione, un numero arbitrario di recapiti (telefono, cellulare, email) indicandone uno come mezzo di comunicazione preferito, ed inserendo i dati relativi alla sua carta di credito (intestatario, numero, data di scadenza, codice CVV). Logistica S.r.l. ha sul territorio differenti centri operativi. Alcuni di questi centri sono dei centri di prossimità, altri sono dei centri di smistamento nazionale, altri sono dei centri di smistamento internazionale. I centri di prossimità hanno a disposizione una flotta di furgoni (ciascuno identificato da una targa e da un codice alfanumerico univoco) che si preoccupano di raggiungere i clienti per il ritiro dei pacchi. I centri di smistamento, invece, hanno a disposizione degli autoarticolati che consentono di raggiungere altri centri di smistamento. L'organizzazione dei centri di smistamento è gerarchica: il centro di prossimità si occupa del ritiro e delle consegne e può raggiungere unicamente centri di smistamento nazionali, un centro di smistamento nazionale può raggiungere un altro centro di smistamento nazionale o un centro di smistamento internazionale appartenente alla sua stessa nazione. Un centro di smistamento internazionale, invece, può raggiungere centri di smistamento della stessa nazione di appartenenza e tutti gli altri centri di smistamento internazionale. Ciascun

centro operativo è identificato da un codice, un indirizzo C.O (centro operativo), coordinate geografiche (latitudine e longitudine), da un recapito telefonico, da un indirizzo email di segreteria e dal nome di un responsabile.

Un cliente può, in qualsiasi momento, richiedere la spedizione di un pacco. Al momento della richiesta, il sistema richiede di inserire le informazioni sul destinatario (nome, coordinate consegna pacco (latitudine e longitudine), stato), le dimensioni (lunghezza, larghezza, profondità) del pacco che si vuole inviare ed il suo peso. Gli amministratori del sistema forniscono delle stime di prezzo per categorie di pacchi (modificabili in qualsiasi momento dall'amministrazione). In particolare, data la somma delle dimensioni del pacco (lunghezza + larghezza + profondità), questo viene associato ad una specifica categoria. All'interno della categoria, il prezzo è calcolato in relazione alla classe di peso cui il pacco appartiene.

Il cliente, pertanto, riceve una stima del costo. Questo costo verrà poi confermato dagli addetti alla spedizione, quando il pacco verrà recapitato ad un centro di prossimità e verrà convertito in un costo effettivo, che verrà poi scalato dalla carta di credito del cliente.

Alla ricezione dell'ordine, il cliente riceve un codice di spedizione che potrà utilizzare per tracciare il suo pacco, dal recupero presso l'indirizzo di recupero pacco da lui fornito fino alla consegna. Inoltre, il sistema informativo determina, in funzione delle coordinate geografiche del cliente, qual è il centro di prossimità che si occuperà del recupero del pacco. In caso di spedizione nazionale, il sistema determinerà anche (in funzione delle coordinate del destinatario) qual è il centro di smistamento nazionale che si occuperà della parte finale della spedizione. In caso di spedizione internazionale, il sistema assegnerà la spedizione al centro di smistamento internazionale più vicino.

Automaticamente, il sistema assegna il recupero della spedizione ad uno dei veicoli del centro di prossimità. Ciascun veicolo ha una capienza massima di pacchi, pertanto verrà selezionato un veicolo con ancora spazio sufficiente. Un veicolo ha un valore volumetrico (inteso come somma delle dimensioni dei pacchi) massimo che viene utilizzato per questo controllo e l'assegnazione. Se non è possibile assegnare un veicolo ad un pacco, questo viene messo in stato di "attesa di recupero" dal sistema. Il giorno successivo, tutti i pacchi in attesa di recupero vengono automaticamente assegnati ai veicoli del centro, secondo la stessa logica.

Quando un veicolo raggiunge il centro operativo, gli operatori del centro scansionano tutti i codici di spedizione. Il sistema, automaticamente, assegna il pacco alla fase successiva della spedizione (ad es, da un centro locale al destinatario, da un centro di prossimità al centro di smistamento nazionale, dal centro di smistamento nazionale al centro di smistamento internazionale, ecc.) sempre secondo la stessa logica.



Viene registrata sempre la data e l'ora in cui un pacco entra in una fase successiva (ad esempio, "raggiunto centro di smistamento nazionale XXX", oppure "in attesa di recupero", oppure "in attesa di smistamento presso XXX", oppure "in consegna", oppure "consegnato", oppure "in transito").

Il cliente ha, in ogni momento, la possibilità di visualizzare lo stato del pacco, fornendo il suo codice di spedizione. Inoltre, tutti i veicoli di Logistica S.r.l. sono equipaggiati di dispositivo GPS che, ogni 5 secondi, comunica al sistema la propria posizione. Quando un pacco è "in transito", il cliente ha la possibilità di visualizzare la latitudine e la longitudine del veicolo che sta trasportando il suo pacco.

Gli amministratori della società generano su base giornaliera dei report indicanti quante spedizioni sono state accettate e quante spedizioni sono state consegnate. Inoltre, per tutte le spedizioni consegnate, il report fornisce informazioni sulle entrate totali (a valle dell'aggiornamento del prezzo effettivo da parte degli operatori dei centri dislocati sul territorio).

Per calcolare la distanza tra due punti espressi in coordinate geografiche (latitudine e longitudine) è possibile utilizzare la seguente formula, dove  $r$  è il raggio della terra:

$$d = 2r \arcsin(\sqrt{\sin^2((\phi_2 - \phi_1)/2) + \cos(\phi_1) \cos(\phi_2) \sin^2((\lambda_2 - \lambda_1)/2)})$$

## Glossario dei Termini

| Termine          | Descrizione  | Sinonimi    | Collegamenti |
|------------------|--|-------------|--------------|
| Veicolo          | Veicolo utilizzato dall'azienda per il trasporto dei pacchi, può essere di piccole dimensioni (furgone) per il ritiro dei pacchi dai clienti o di grandi dimensioni per il trasporto da un centro di smistamento ad un altro | autoveicolo |              |
| centro operativo | centri operativi dell'azienda presenti sul territorio per  |             |              |

|            |   |          |  |
|------------|---|----------|--|
|            | l'organizzazione e la logistica delle spedizioni, si dividono in centri di prossimità, centri di smistamento nazionale e centri di smistamento internazionale   |          |  |
| cliente    | cliente che richiede la spedizione di un determinato pacco  | mittente |  |
| pacco      | pacco di cui è richiesta la spedizione da parte del cliente. Sono catalogati in base alle dimensioni e al peso  |          |  |
| spedizione | processo di ritiro di un pacco dal cliente e di invio all'indirizzo di destinazione. La spedizione può avvenire in territorio nazionale oppure internazionale nel caso di spedizione di un pacco in una nazione diversa da quella di partenza | consegna |  |

## Raggruppamento dei requisiti in insiemi omogenei

### Frasi relative a veicolo

Ciascun veicolo ha una capienza massima di pacchi, pertanto verrà selezionato un veicolo con ancora spazio sufficiente. Un veicolo ha un valore volumetrico (inteso come somma delle dimensioni dei pacchi) massimo che viene utilizzato per questo controllo e l'assegnazione. tutti i veicoli di Logistica S.r.l. sono equipaggiati di dispositivo GPS che, ogni 5 secondi, comunica al sistema la propria posizione.

5

### Frasi relative a centro operativo

Logistica S.r.l. ha sul territorio differenti centri operativi. Alcuni di questi centri sono dei centri di prossimità, altri sono dei centri di smistamento nazionale, altri sono dei centri di smistamento

internazionale.

L'organizzazione dei centri di smistamento è gerarchica: il centro di prossimità si occupa del ritiro e delle consegne e può raggiungere unicamente centri di smistamento nazionali, un centro di smistamento nazionale può raggiungere un altro centro di smistamento nazionale o un centro di smistamento internazionale appartenente alla sua stessa nazione.

Un centro di smistamento internazionale, invece, può raggiungere centri di smistamento della stessa nazione di appartenenza e tutti gli altri centri di smistamento internazionale. Ciascun centro operativo è identificato da un codice, un indirizzo C.O (centro operativo), coordinate geografiche (latitudine e longitudine), da un recapito telefonico, da un indirizzo email di segreteria e dal nome di un responsabile.

Alla ricezione di un ordine, il sistema informativo determina, in funzione delle coordinate geografiche del cliente, qual è il centro di prossimità che si occuperà del recupero del pacco. In caso di spedizione nazionale, il sistema determinerà anche (in funzione delle coordinate del destinatario) qual è il centro di smistamento nazionale che si occuperà della parte finale della spedizione. In caso di spedizione internazionale, il sistema assegnerà la spedizione al centro di smistamento internazionale più vicino.

#### **Frase relative a cliente**

Un cliente, pertanto, al momento della registrazione nel sistema, fornisce tutte le sue informazioni anagrafiche, ovvero il CF, il nome, cognome, anno di nascita; indicando un indirizzo di residenza, un indirizzo di fatturazione, un numero arbitrario di indirizzi da cui è possibile recuperare un pacco in spedizione, un numero arbitrario di recapiti (telefono, cellulare, email) indicandone uno come mezzo di comunicazione preferito, ed inserendo i dati relativi alla sua carta di credito (intestatario, numero, data di scadenza, codice CVV).

Un cliente può, in qualsiasi momento, richiedere la spedizione di un pacco.

Alla ricezione dell'ordine, il cliente riceve un codice di spedizione che potrà utilizzare per tracciare il suo pacco, dal recupero presso l'indirizzo di recupero pacco da lui fornito fino alla consegna.

5

#### **Frase relative a pacco**

La gestione delle consegne espresse di Logistica S.r.l. permette di trasportare a destinazione i pacchi in al più tre giorni. I pacchi spediti dai clienti vengono recuperati direttamente alle coordinate di recupero del pacco fornite dal cliente.

Gli amministratori del sistema forniscono delle stime di prezzo per categorie di pacchi (modificabili in qualsiasi momento dall'amministrazione). In particolare, data la somma delle dimensioni del pacco (lunghezza + larghezza + profondità), questo viene associato ad una specifica categoria. All'interno della categoria, il prezzo è calcolato in relazione alla classe di peso cui il pacco appartiene.

#### **Frase relative a spedizione**

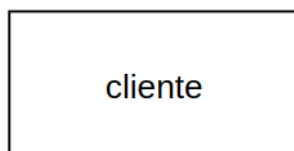
In caso di spedizione nazionale, il sistema determinerà anche (in funzione delle coordinate del destinatario) qual è il centro di smistamento nazionale che si occuperà della parte finale della

consegna. In caso di spedizione internazionale, il sistema assegnerà la consegna al centro di smistamento internazionale più vicino.

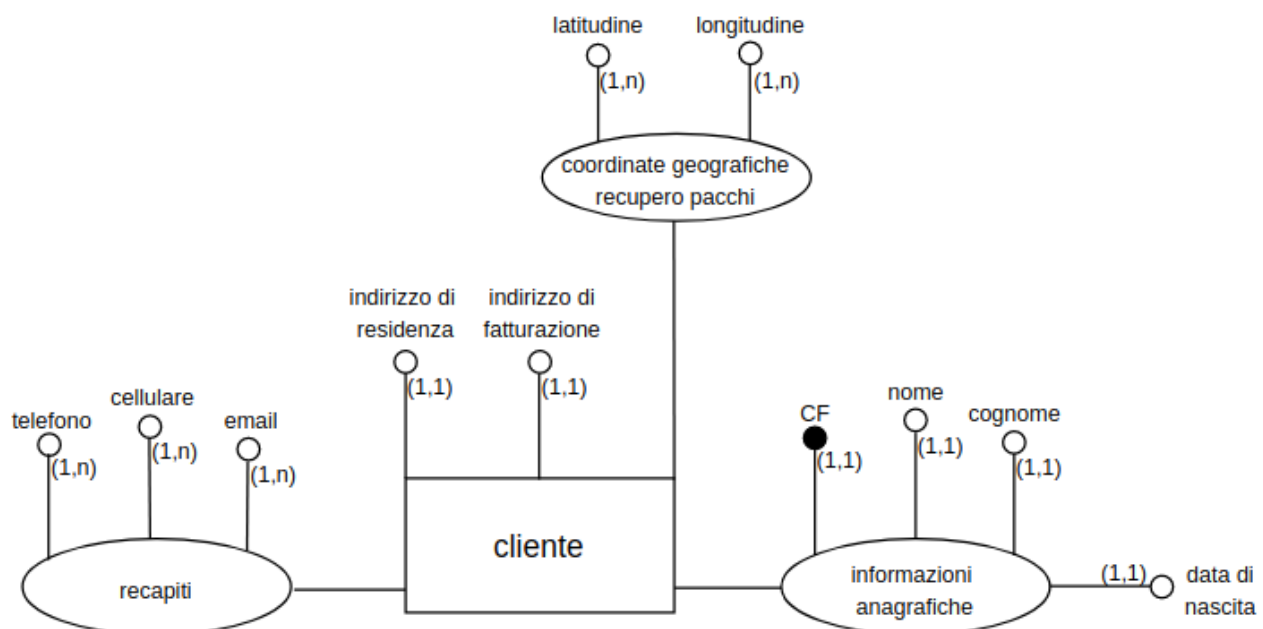
### 3. Progettazione concettuale

#### Costruzione dello schema E-R

5 Parto analizzando l'entità "cliente"



aggiungo i vari attributi dell'entità cliente

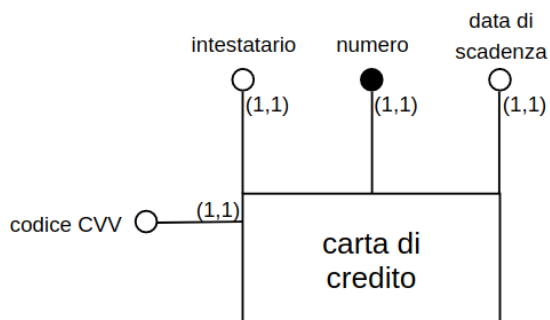


sono presenti 3 attributi composti: uno è l'attributo "recapiti" formato dagli attributi "telefono", "cellulare" ed "email", tutti con cardinalità (1,n) in quanto è possibile specificare un numero arbitrario di recapiti; un altro è l'attributo "informazioni anagrafiche" contenente il nome, il cognome, la data di nascita e il codice fiscale (CF) del cliente, mentre l'ultimo sono le coordinate geografiche dei vari indirizzi in cui recuperare i pacchi.

Il codice fiscale è identificatore di cliente in quanto consideriamo che nel nostro minimondo il codice fiscale non sia soggetto ad omocodia, in questo modo, essendo unico, siamo in grado di identificare ogni cliente tramite il CF.

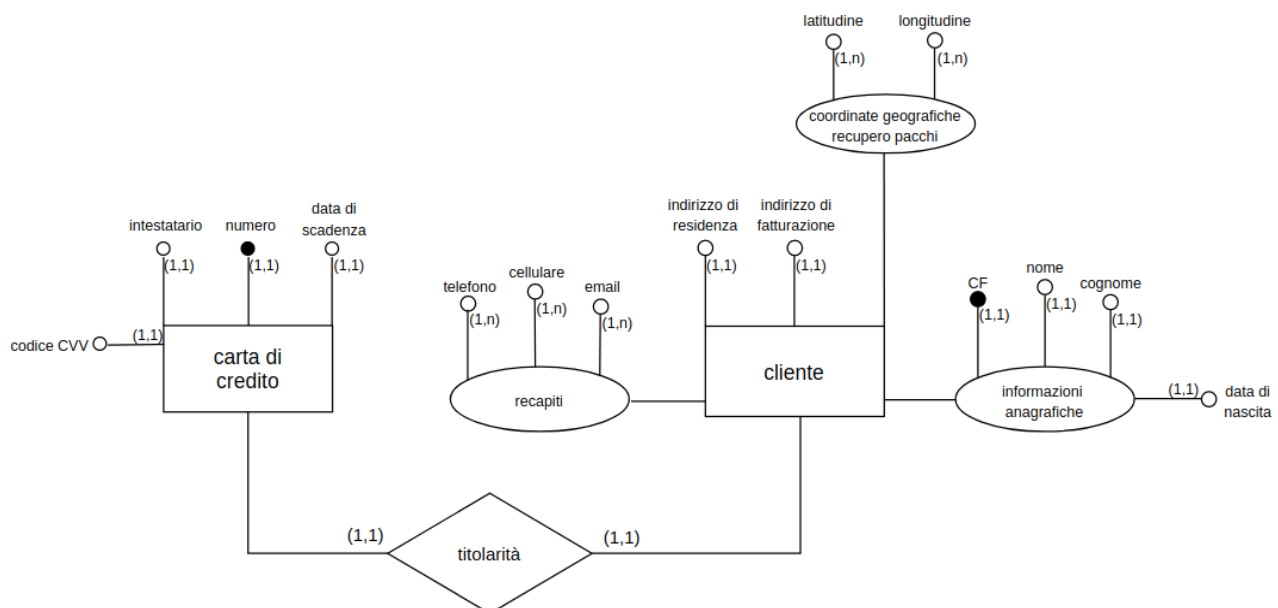
Al momento della registrazione il cliente deve registrare una carta di credito per il pagamento della

spedizione, creiamo quindi l'entità carta di credito.



5 Considero il “numero” della carta di credito come identificatore in quanto è univoco per ogni carta di credito e ci permette quindi di distinguere una carta dall'altra.

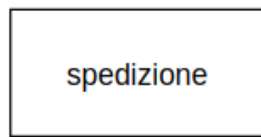
10 A questo punto vado a collegare tramite una relazione l'entità “cliente” e l'entità “carta di credito”.



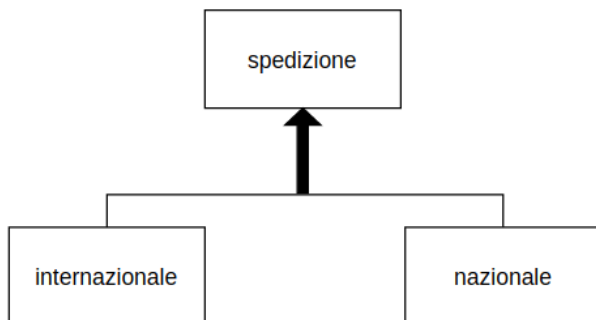
15 La relazione “titolarità” mette in relazione il cliente con la propria carta di credito, la relazione è del tipo uno a uno in quanto ogni cliente ha una sola carta di credito e ogni carta di credito ha un solo proprietario.

20

entità “spedizione”

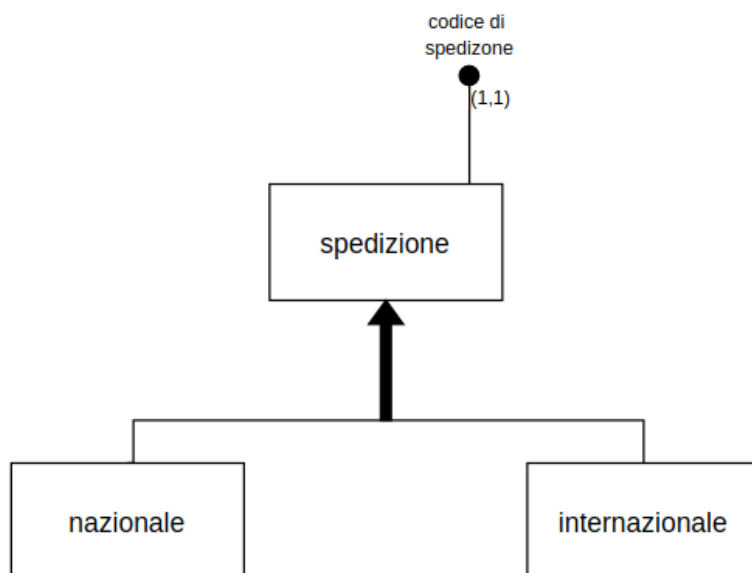


l’entità “spedizione” ha 2 entità figlie, infatti una spedizione può essere nazionale o internazionale.



10

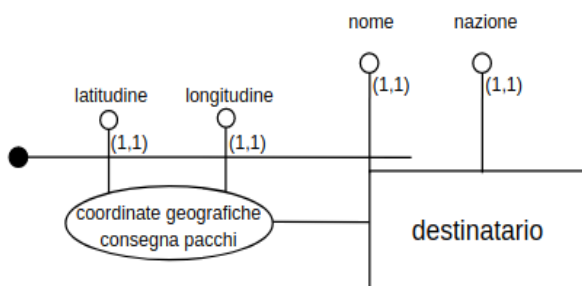
l’attributo di spedizione è “codice di spedizione” che è anche identificatore dato che ogni spedizione ha un proprio codice univoco.



15

Al momento della richiesta inoltre verranno inseriti I dati delle entità “destinatario” e dell’entità “pacco”.

Destinatario:



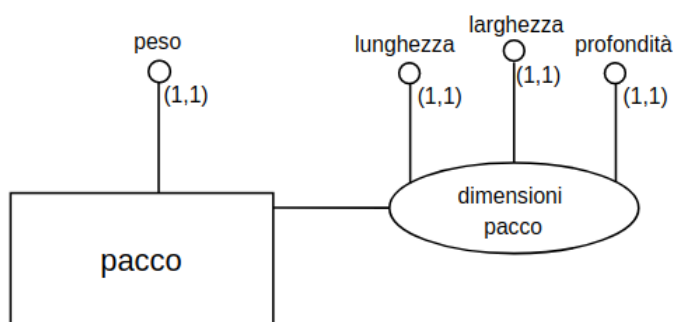
5

destinatario ha come attributo nome, nazione e un attributo composto contenente latitudine e longitudine dell'indirizzo del destinatario del pacco.

Latitudine, longitudine e nome sono un identificatore di destinatario in quanto assumo che ci sia una sola persona di un certo nome che abiti in quelle precise coordinate.

10

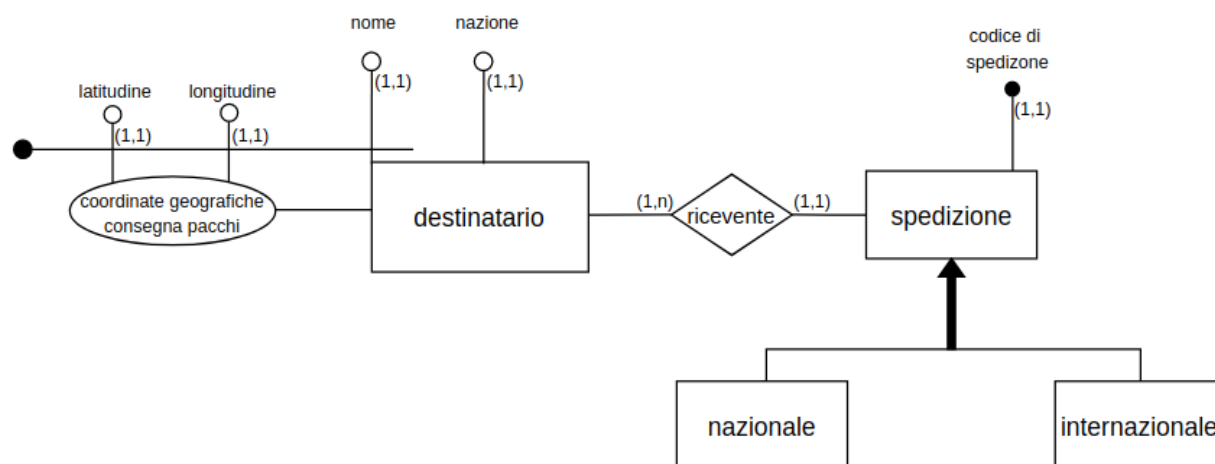
Pacco:



15 l'entità pacco ha come attributo il peso e un attributo composto contenente la lunghezza, larghezza e profondità del pacco.

Collegiamo l'entità "spedizione" e "destinatario".

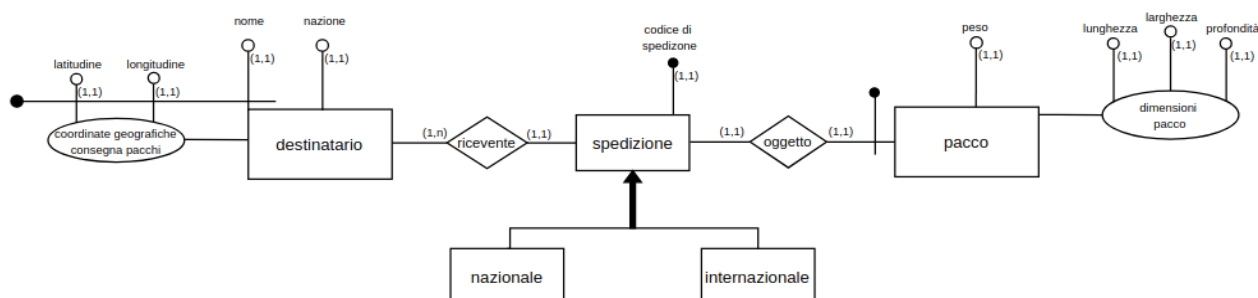




La relazione “ricevente” è di tipo uno a molti in quanto una spedizione ha un solo destinatario mentre un destinatario può avere più spedizioni associate a lui.

5

E poi con l’entità “pacco”:

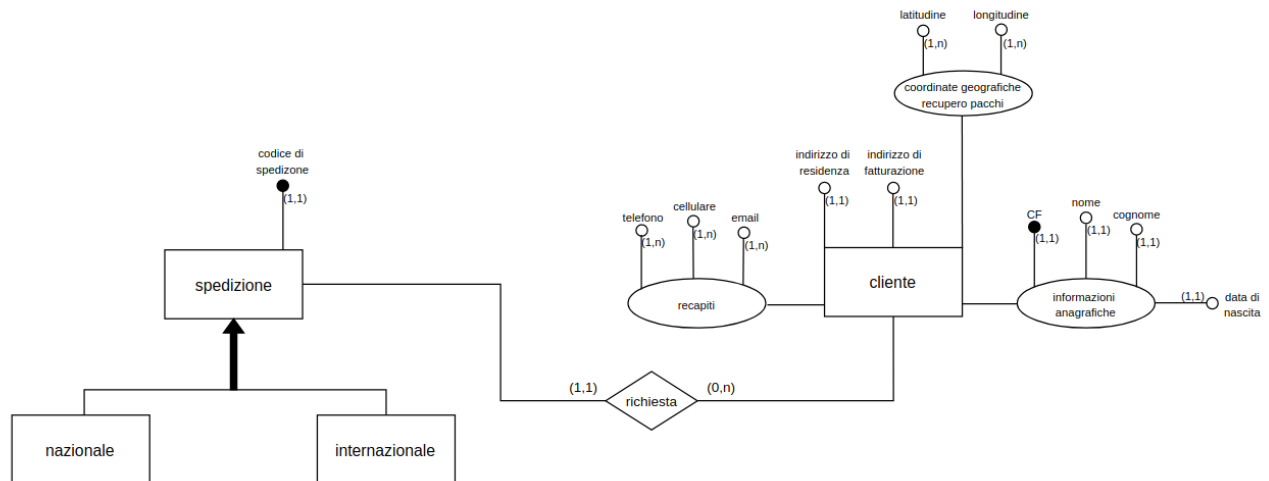


10

l’entità “pacco” è un’identità debole, aggiungiamo pertanto un attributo esterno per identificarla tramite l’entità spedizione, in quanto ogni spedizione e quindi ogni codice di spedizione fanno riferimento ad un solo pacco; inoltre il concetto di “pacco” esiste solo in relazione al concetto di “spedizione”.

15 La relazione “oggetto” è del tipo uno a uno.

20 l’entità “spedizione” è inoltre in relazione all’entità “cliente” tramite la relazione “richiesta” che esprime la possibilità da parte del cliente di effettuare una nuova spedizione.

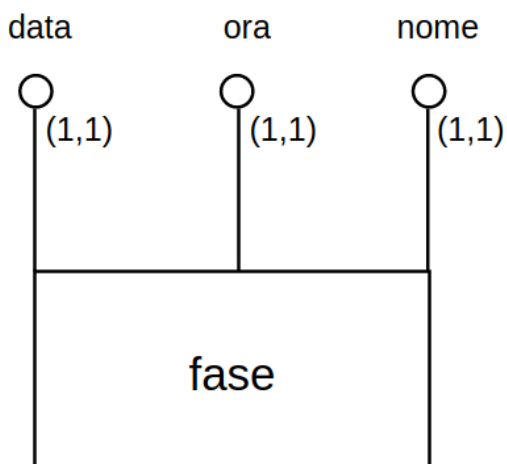


L'associazione è del tipo uno a molti in quanto una spedizione è caratterizzata da un solo cliente mentre un cliente può richiedere un numero arbitrario di spedizioni.

5

10

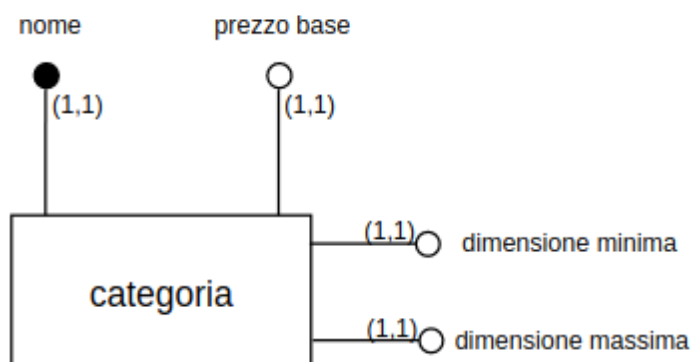
l'entità "pacco" è inoltre associata ad altre 2 entità, la prima è l'entità "fase":



con gli attributi sono "data", "ora" e "nome", tutti con cardinalità (1,1).

20

la seconda è l'entità "categoria":

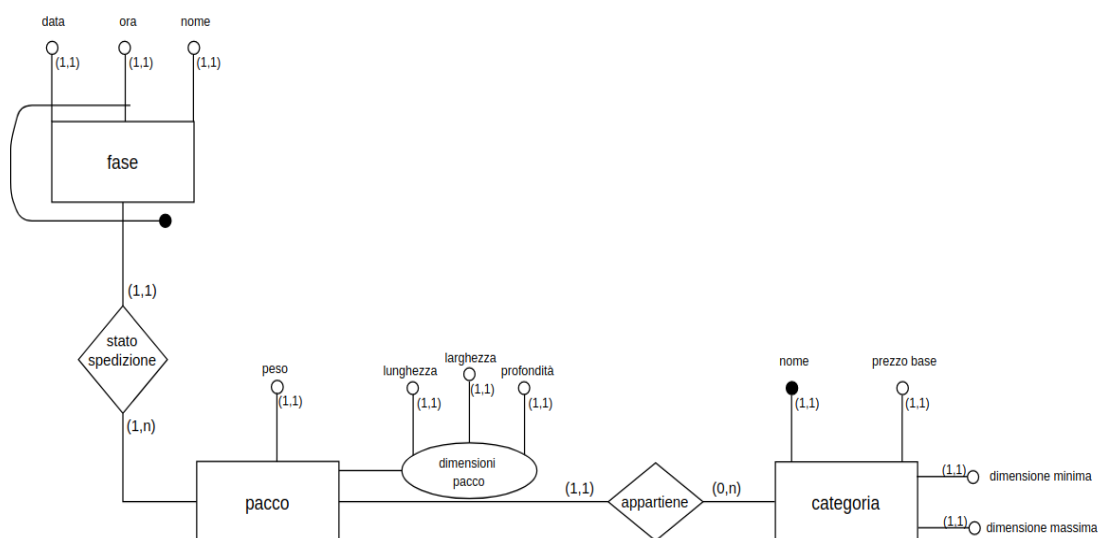


5

con i rispettivi attributi "nome", che è anche identificatore per l'entità e "prezzo base" con cardinalità (1,1).

10

queste 2 entità sono in relazione con l'entità "pacco" rispettivamente tramite l'associazione "stato spedizione" e "appartiene":



15

l'associazione "stato di spedizione" ha una cardinalità uno a molti in quanto un'istanza di fase fa riferimento ad un unico pacco, mentre un pacco nel corso della spedizione attraversa più fasi.

l'entità "fase" è un'entità debole in quanto ha bisogno di un attributo esterno per essere identificata.

La relazione "appartiene" associa ad ogni pacco la categoria a cui esso appartiene, ha quindi anche essa una cardinalità uno a molti in quanto ogni pacco appartiene ad una sola categoria mentre una determinata categoria può da 0 a n pacchi.

20

5

10

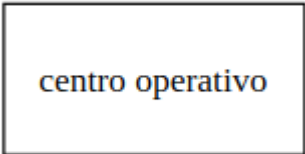
15

20

25

30

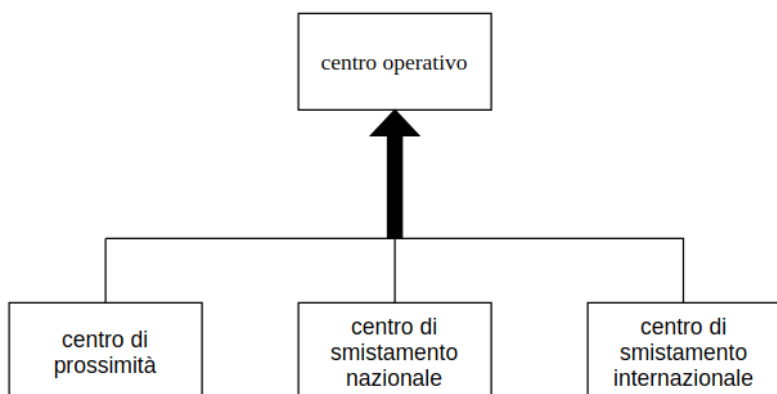
Continuo con l'entità centro operativo.



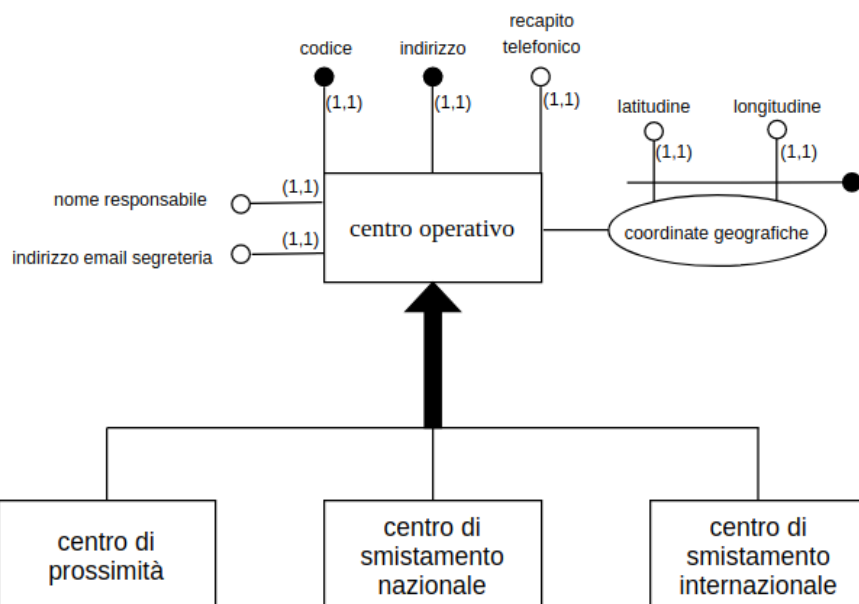
centro operativo

35

I centri operativi sono divisi in centri di prossimità, centri di smistamento nazionale e centri di smistamento internazionale. A centro operativo va quindi applicata una generalizzazione, in questo caso una generalizzazione totale in quanto tutti I centri operativi fanno parte di una di queste categorie ed esclusiva dato che l'entità padre fa parte di una sola entità figlia.



vado adesso a specificare gli attributi del centro operativo



5

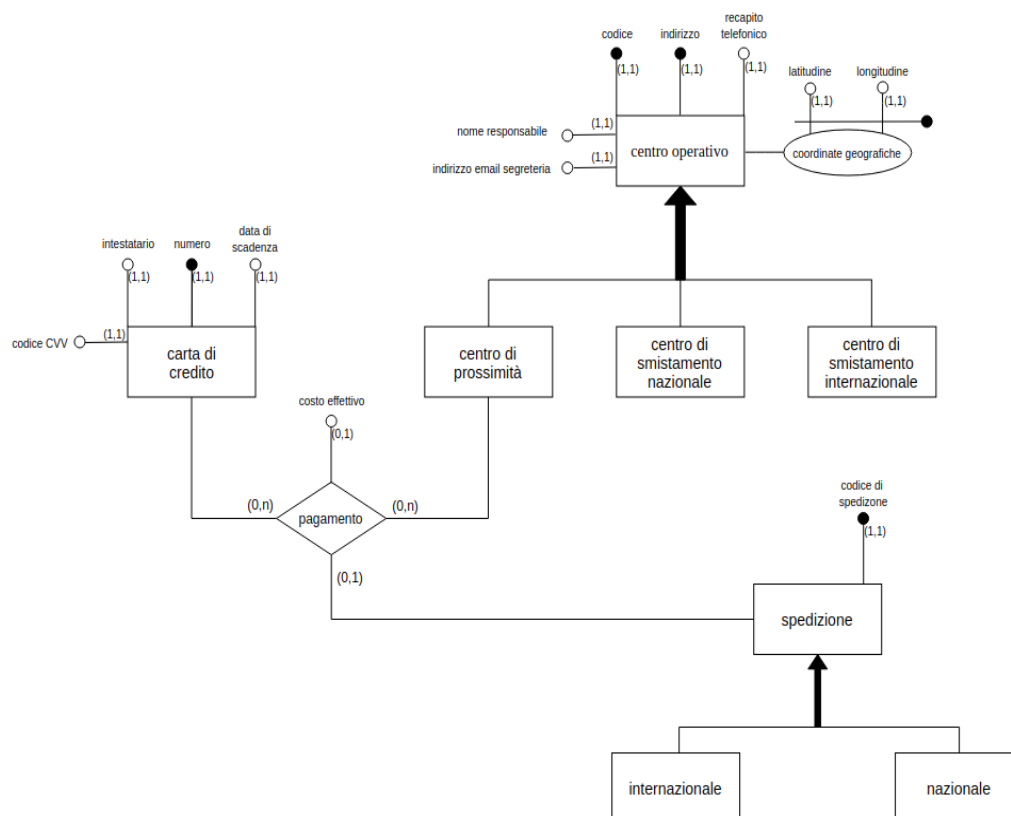
in particolare gli attributi “codice”, “indirizzo” e “latitudine e longitudine” sono identificatori per l’entità “centro operativo”.

l’attributo “coordinate geografiche” è un attributo composto in quanto è composto dai due dati “latitudine” e “longitudine”.

10 Tutti gli attributi sono inoltre caratterizzati da una cardinalità (1,1) in quanto sono unici e devono essere presenti tutti.

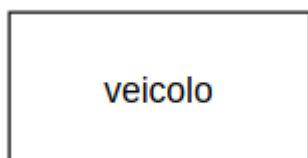
l’entità figlia “centro di prossimità” è in relazione a l’entità “carta” tramite l’associazione “pagamento”:

15

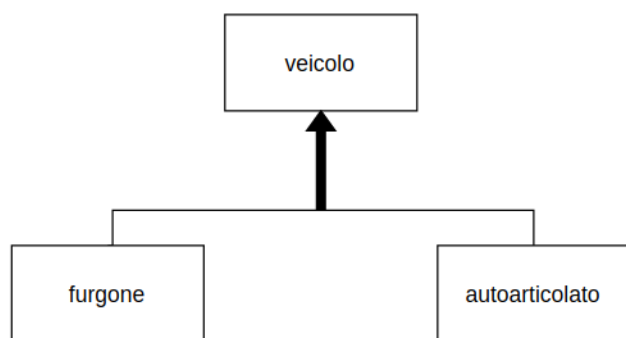


l'associazione è del tipo molti a molti per quanto riguarda carta di credito e centro operativo, in quanto una carta di credito può contribuire a più pagamenti e un centro di prossimità può confermare e successivamente addebitare un numero arbitrario di "costi effettivi". Mentre ha una cardinalità pari a uno per quanto riguarda spedizione in quanto una spedizione può avere associato un solo pagamento.

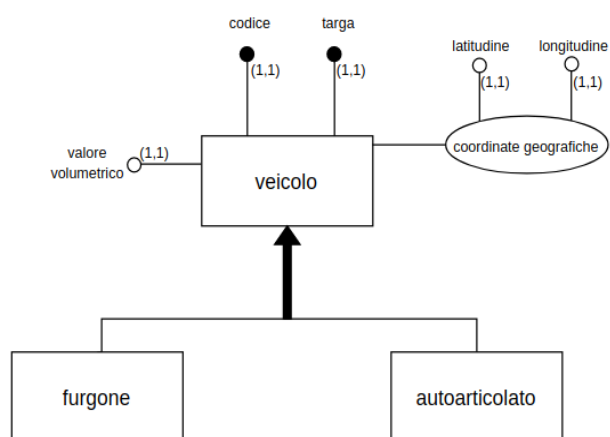
Entità "veicolo":



un veicolo può essere un furgone o un autoarticolato, pertanto applico una generalizzazione all'entità:



aggiungo gli attributi di veicolo:



gli attributi codice e targa sono identificatori dell'entità "veicolo".

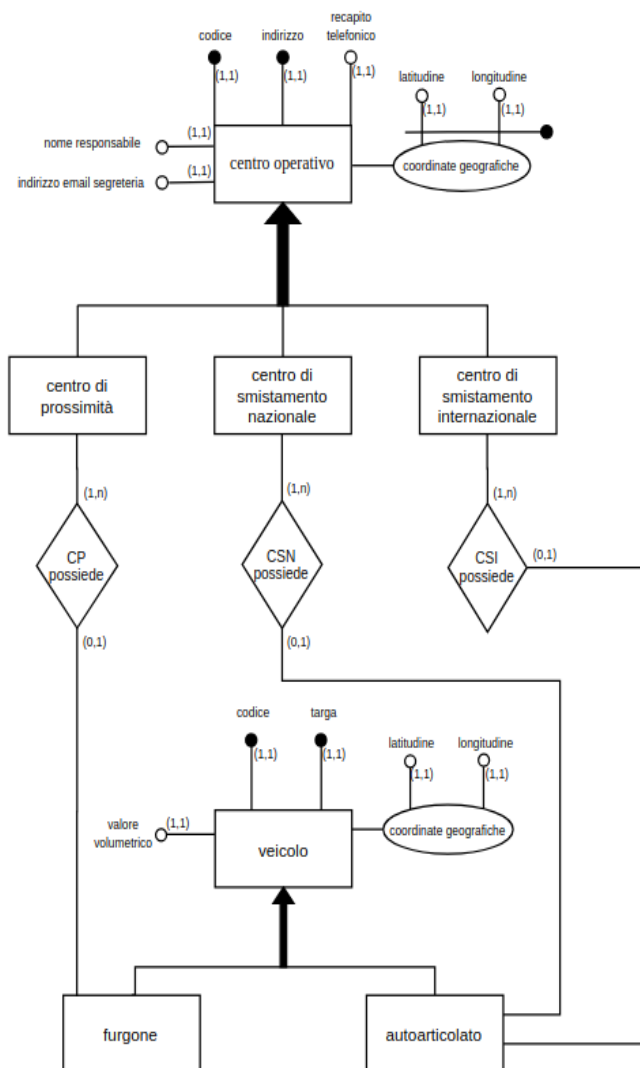
10

15

20

25

l'entità veicolo è in associazione con l'entità centro operativo.



In particolare è presente un'associazione tra ogni tipo di centro operativo e il veicolo da esso posseduto. Ogni relazione è del tipo uno a molti dato che un furgone o un autoarticolato può appartenere ad un solo centro operativo mentre un centro operativo può possedere più furgoni o autoarticolati.

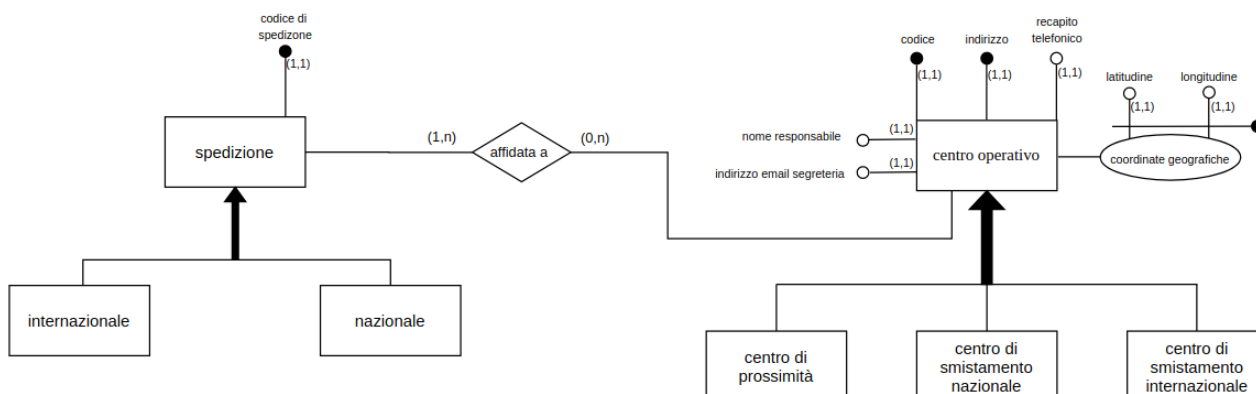
10

15

20



Metto adesso in relazione le entità “centro operativo” e “spedizione” tramite l’associazione “affidata a”:

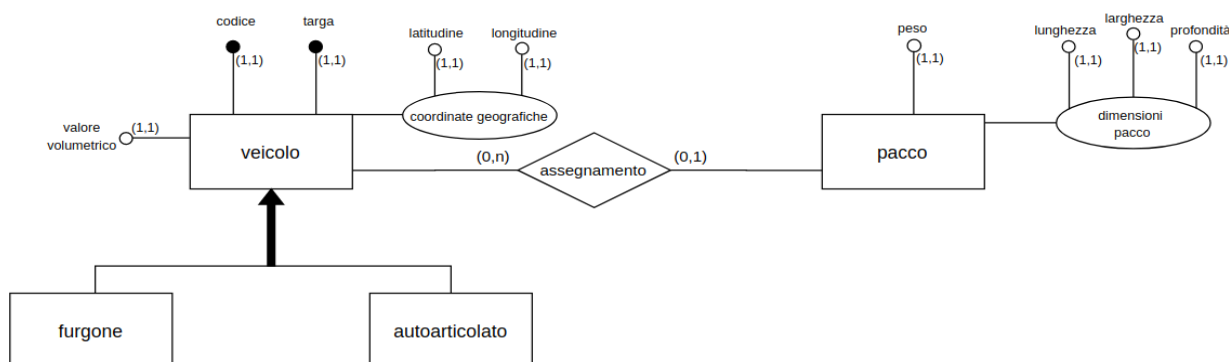


l’associazione “affidata a” è del tipo molti a molti in quanto una spedizione per essere gestita nella sua interezza ha bisogno di più centri operativi, a sua volta un centro operativo gestisce più spedizioni.

La cardinalità minima da parte del centro operativo è 0 in quanto nella fase iniziale un centro operativo potrebbe non aver dover gestire nessuna spedizione.

10

Le entità “veicolo” e “pacco” sono in relazione tramite l’associazione “assegnamento”:



la cardinalità della relazione “assegnamento” è del tipo uno a molti (con cardinalità minima minima 0) in quanto un pacco è presente all’interno della relazione solamente quando è effettivamente associato ad un veicolo. Un veicolo invece può essere assegnato o a nessun pacco e quindi essere inutilizzato, o in generale a più pacchi.

20

25

5

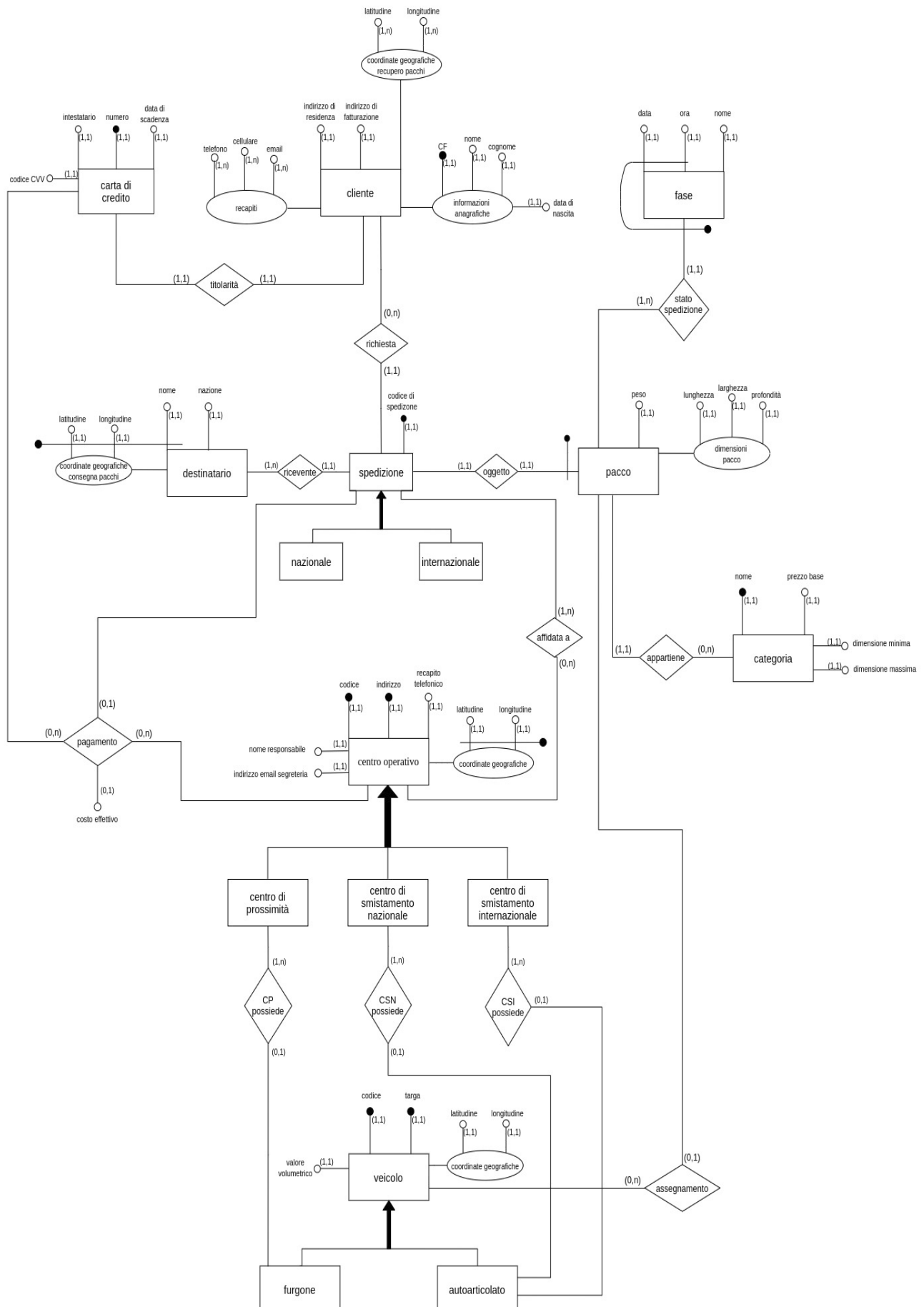
**Integrazione finale**

10 È presente un conflitto sui nomi per quanto riguarda le associazioni tra le specializzazioni di centro operativo e le specializzazioni di veicolo in quanto hanno tutte lo stesso nome “possiede”. Per risolvere questo conflitto verranno ridenominate come: “CP possiede” per identificare l’associazione tra centro di prossimità e furgone, “CSN possiede” per identificare l’associazione tra centro di smistamento nazionale e autoarticolato ed infine “CSI possiede” per identificare l’associazione tra il centro di smistamento internazione e autoarticolato.

15

20

25



## 5 Regole aziendali

- Un centro di prossimità può raggiungere esclusivamente un centro di smistamento nazionale
- un centro di smistamento nazionale può raggiungere un centro di smistamento internazionale solo se appartengono alla stessa nazione
- un centro di smistamento internazionale può raggiungere un centro di smistamento nazionale solo se appartengono alla stessa nazione
- la categoria di appartenenza del pacco è determinata in base alla somma di lunghezza, larghezza e profondità
- il prezzo si ottiene applicando una formula in base al prezzo base della categoria ed al peso del pacco
- il valore volumetrico residuo di un veicolo si ottiene sottraendo al valore volumetrico totale la somma del volume di tutti i pacchi assegnati a quel veicolo
- un pacco che non può essere assegnato a nessun veicolo viene messo in stato di attesa
- I pacchi in attesa devono essere assegnati ad un veicolo il giorno successivo
- ogni 5 secondi il veicolo deve comunicare la propria posizione
- il numero di spedizione accettate in un giorno si ottiene contando quante spedizioni sono in fase “accettata”
- il numero di spedizione consegnate in un giorno si ottiene contando quanti pacchi sono entrati in fase “consegnato” in quel giorno
- le entrate totali in un giorno si ottengono sommando i costi effettivi dei pacchi consegnati in quel giorno

40

## Dizionario dei dati

| Entità           | Descrizione   | Attributi  | Identificatori                |
|------------------|---|--|-------------------------------|
| Cliente          | Persona iscritta al sistema che usufruisce del servizio di spedizione dell'azienda                          | Telefono, cellulare, email, indirizzo di residenza, indirizzo di fatturazione, latitudine, longitudine, CF, nome, cognome, data di nascita | Codice                        |
| carta di credito | carta di credito appartenente ad un cliente sulla quale verranno addebitati I costi di spedizione del pacco | codice CVV, intestatario, numero, data di scadenza   | numero                        |
| spedizione       | istanza della spedizione di un pacco  | codice di spedizione   | codice di spedizione          |
| destinatario     | nome e indirizzo a cui è associata la consegna di un pacco  | nome, nazione, latitudine, longitudine   | latitudine, longitudine, nome |
| pacco            | oggetto che si vuole spedire  | peso , lunghezza, larghezza, profondità  | spedizione                    |
| centro operativo | centro logistico di smistamento dei pacchi  | codice, nome responsabile, indirizzo email segreteria, indirizzo, recapito telefonico, latitudine, longitudine                             | codice                        |
| veicolo          | veicolo per il trasporto dei pacchi   | valore volumetrico, codice, targa, latitudine, longitudine   | codice, targa                 |
| categoria        | categoria a cui appartiene il pacco destinato alla spedizione   | nome, prezzo base, dimensione minima, dimensione   | nome                          |

|      |                               |                            |                          |
|------|-------------------------------|----------------------------|--------------------------|
| fase | fase in cui si trova il pacco | massima<br>data, ora, nome | data, ora,<br>spedizione |
|------|-------------------------------|----------------------------|--------------------------|

## 4. Progettazione logica

### Volume dei dati

- 5 •Stimiamo a regime un numero di clienti di circa 100000 persone e quindi altrettante carte di credito registrare.
- Ipotizzando che ogni cliente effettui in media 5 spedizioni abbiamo 500000 spedizioni.
- 10 •Il destinatario è pari a 200000 in quanto possiamo supporre che presumibilmente alcune spedizioni avranno lo stesso destinatario, già registrato all'interno della base di dati.
- Il numero di pacchi sarà uguale al numero di spedizioni.
- 15 •Supponendo che la maggior parte delle spedizioni sia di tipo nazionale avremo la seguente successione di fasi: “richiesta di conferma” → “in attesa di recupero” → “raggiunto centro di prossimità XXX” → “in transito” → “raggiunto centro di smistamento nazionale XXX” → “in transito” → “raggiunto centro di smistamento nazionale XXX” → “in transito” → “raggiunto centro di prossimità XXX” → “in consegna” → “consegnato”; di 12 fasi, nel caso di spedizione locale saranno 7 mentre nel caso di spedizione internazionale 15. Si sceglie quindi di porre in media 10 fasi
- 20 per ogni spedizione, ovvero 5000000 di fasi.
- Scegliamo una media di 5 categorie per classificare i pacchi.
- Si suppone che l'azienda abbia 500 centri operativi.
- 25 •Ipotizzando che ogni centro operativo abbia in media 4 veicoli si hanno in totale 2000 veicoli.
- Titolarità assume lo stesso numero di cliente e carta di credito.
- 30 •Richiesta assume lo stesso numero di spedizioni.
- Ricevente assume lo stesso numero di spedizioni.
- Oggetto assume lo stesso numero di spedizioni.
- 35 •Stato di spedizioni assume lo stesso numero di fase.
- Appartiene assume lo stesso numero di spedizione.
- 40 •Essendo che una spedizione nazionale coinvolge 4 centri operativi, una locale 1 un centro mentre una internazionale 6 centri calcoliamo in media 3 centri operativi coinvolti per ogni spedizione, ovvero avremo un numero di relazioni “affidata a” pari a 1500000.
- pagamento assume lo stesso numero di spedizione.
- 45 •Possiede assume lo stesso numero di veicolo.

•Sapendo che una spedizione nazionale coinvolge 5 veicoli, una internazionale 7 veicoli ed una locale 2 veicoli ipotizziamo che per ogni spedizioni vengano assegnati in media 4 veicoli per ogni pacco, ovvero 2000000 di veicoli.

5

| Concetto nello schema | Tipo <sup>1</sup> | Volume atteso |
|-----------------------|-------------------|---------------|
| Cliente               | E                 | 100000        |
| carta di credito      | E                 | 100000        |
| spedizione            | E                 | 500000        |
| destinatario          | E                 | 200000        |
| pacco                 | E                 | 500000        |
| fase                  | E                 | 5000000       |
| categoria             | E                 | 5             |
| centro operativo      | E                 | 500           |
| veicolo               | E                 | 2000          |
| titolarità            | R                 | 100000        |
| richiesta             | R                 | 500000        |
| ricevente             | R                 | 500000        |
| oggetto               | R                 | 500000        |
| stato spedizione      | R                 | 5000000       |
| appartiene            | R                 | 500000        |
| affidata a            | R                 | 1500000       |
| pagamento             | R                 | 500000        |
| possiede              | R                 | 10000         |
| assegnamento          | R                 | 2000000       |

---

1 Indicare con E le entità, con R le relazioni



## Tavola delle operazioni

- Assumiamo che ogni giorno si registrino circa 50 clienti, con questa stima avremo che la base di dati dal momento dell'istanziamento raggiungerà il volume atteso di dati in circa 5/6 anni.
- 5 •La frequenza delle registrazioni per le carte di credito sarà la stessa di quella dei clienti.
- Si assume che ogni giorni vengano richieste 200 nuove spedizioni.
- La frequenza di registrazione del destinatario è leggermente minore della frequenza delle nuove spedizioni in quanto si assume che alcuni destinatari siano già registrati all'interno della base di dati.
- Per la registrazione del pacco avremo la stessa frequenza della richiesta di spedizioni.
- 10 •L'aggiornamento della fase di spedizioni viene calcolato assunto che ogni giorno un pacco cambi il suo stato di spedizioni circa 3 volte.
- l'assegnamento del pacco ad una categoria viene fatto ogni volta che si registra un nuovo pacco.
- Dato il numero medio di centri operativi assegnati per ogni pacco assumiamo una frequenza di affidamento della spedizione ad un centro operativo di poco superiore alla frequenza di nuove spedizioni.
- 15 •La frequenza del pagamento di spedizione sarà pari alla frequenza di nuove spedizioni.
- Assumendo che ogni pacchi cambi circa 2 volte al giorno veicolo abbiamo che una frequenza di assegnamento doppia rispetto a quella di nuove spedizioni.
- Ipotizzando che ogni cliente voglia verificare la posizioni del pacco in media una volta al giorno
- 20 avremo una frequenza uguale a quella delle nuove spedizioni.
- Ipotizziamo invece la frequenza di visualizzazione dello stato del pacco leggermente superiore alla frequenza di nuove spedizioni.
- La frequenza di comunicazione della posizione del veicolo, sapendo che ogni veicolo comunica la propria posizione ogni 5 secondi ed assunto che un veicolo comunichi la propria posizione anche
- 25 nel caso in cui non sia in transito è di  $24 \text{ ore} * 60 \text{ minuti} * 60 \text{ secondi} / 5 \text{ secondi} * 2000 \text{ veicoli}$ .
- La frequenza dei report per le spedizioni accettate è di 1 report al giorno.
- La frequenza dei report per le spedizioni consegnate è di 1 report al giorno.
- La frequenza dei report per le spedizioni totali è di 1 report al giorno.

30

| Cod. | Descrizione                       | Frequenza attesa |
|------|-----------------------------------|------------------|
| 1    | Registrazione di un nuovo cliente | 50/giorno        |

|    |  |                                   |
|----|--|-----------------------------------|
| 2  | registrazione carta di credito                                     | 50/giorno                         |
| 3  | richiesta di una nuova spedizione (registrazione del destinatario) | 200/giorno                        |
|    | registrazione del pacco )  | 150/giorno                        |
| 4  | aggiornamento della fase di spedizione                             | 200/giorno                        |
| 5  | assegnamento del pacco alla categoria di appartenenza              | 600/giorno                        |
| 6  | affidamento della spedizione al centro operativo                   | 200/giorno                        |
| 7  | pagamento della spedizione   | 300/giorno                        |
| 8  | assegnamento di un veicolo per il pacco                            | 200/giorno                        |
| 9  | richiesta di tracciamento posizione del pacco                      | 400/giorno                        |
| 10 | richiesta visualizzazione stato del pacco                          | 200/giorno                        |
| 11 | comunicazione posizione veicolo                                    | 300/giorno                        |
| 12 | report spedizione accettate  | 34560000/giorno (24*60*60*2000/5) |
| 13 | report spedizione consegnate                                       | 1/giorno                          |
| 14 | report entrate totali  | 1/giorno                          |

## Costo delle operazioni

### 1) Registrazione di un nuovo cliente

| Concetto | Costrutto | Accessi | Tipo |
|----------|-----------|---------|------|
| Cliente  | entità    | 1       | S    |

- 5 avendo un solo accesso in scrittura con una frequenza di 50 operazioni al giorno il costo sarà di 100 accessi al giorno

### 2) registrazione carta di credito

| Concetto | Costrutto | Accessi | Tipo |
|----------|-----------|---------|------|
| Credito  | Entità    | 1       | L    |

|                  |           |   |   |
|------------------|-----------|---|---|
| titolarità       | relazione | 1 | L |
| carta di credito | entità    | 1 | S |

avendo 2 accessi in lettura e 1 accesso in scrittura il costo della singola operazione è pari a 4, per 50 operazioni al giorno fa 200 accessi al giorno.

5 3) richiesta di una nuova spedizione

| Concetto  | Costrutto | Accessi | Tipo |
|-----------|-----------|---------|------|
| Cliente   | Entità    | 1       | L    |
| richiesta | relazione | 1       | L    |

4) richiesta di una nuova spedizione

| Concetto    | Costrutto | Accessi | Tipo |
|-------------|-----------|---------|------|
| Spedizione  | Entità    | 1       | L    |
| oggetto     | relazione | 1       | L    |
| pacco       | entità    | 1       | L    |
| stato della | relazione | 1       | L    |
| spedizione  |           |         |      |
| fase        | entità    | 1       | S    |

10 avendo 4 accessi in lettura e 1 accesso in scrittura il costo totale per la singola operazione è di 6 accessi, con una frequenza di 600 operazioni al giorno si hanno 3600 accessi al giorno.

5) assegnamento del pacco alla categoria di appartenenza

| Concetto   | Costrutto | Accessi | Tipo |
|------------|-----------|---------|------|
| Pacco      | Entità    | 1       | L    |
| appartiene | relazione | 1       | S    |

15 avendo un accesso in lettura e uno scrittura il costo totale per la singola operazione è di 3 accessi, con una frequenza di 200 operazioni al giorno si hanno in totale 600 accessi al giorno.

6) affidamento della spedizione al centro operativo

| Concetto   | Costrutto | Accessi | Tipo |
|------------|-----------|---------|------|
| Spedizione | Entità    | 1       | L    |
| affidata a | relazione | 1       | S    |

20 avendo un accesso in lettura e uno in scrittura il costo della singola operazione è di 3 accessi, con una frequenza di 300 operazione al giorno si hanno in totale 900 accessi al giorno.

## 7) pagamento della spedizione

| Concetto         | Costrutto | Accessi | Tipo |
|------------------|-----------|---------|------|
| Centro operativo | Entità    | 1       | L    |
| spedizione       | entità    | 1       | L    |
| pagamento        | relazione | 1       | S    |
| carta di credito | entità    | 1       | L    |

5 avendo 3 accessi in lettura ed 1 in scrittura il costo della singola operazione è pari a 5 accessi, con una frequenza di 200 operazioni al giorno si hanno in totale 1000 accessi al giorno.

## 8) assegnamento di un veicolo per il pacco

| Concetto     | Costrutto | Accessi | Tipo |
|--------------|-----------|---------|------|
| Veicolo      | Entità    | 1       | L    |
| assegnamento | relazione | 1       | S    |

10 avendo 1 accesso in lettura ed uno in scrittura il costo della singola operazione è pari a 3 accessi, con una frequenza di 400 operazioni al giorno si hanno in totale 1200 accessi al giorno.

## 9) richiesta di tracciamento posizione del pacco

| Concetto         | Costrutto | Accessi | Tipo |
|------------------|-----------|---------|------|
| Spedizione       | Entità    | 1       | L    |
| oggetto          | relazione | 1       | L    |
| pacco            | entità    | 1       | L    |
| assegnamento     | relazione | 1       | L    |
| veicolo          | entità    | 1       | L    |
| stato spedizione | relazione | 1       | L    |
| fase             | entità    | 1       | L    |

15 avendo 7 accessi in lettura il costo della singola spedizione è pari a 7 accessi, con una frequenza di 200 operazioni al giorno si hanno in toale 1400 accessi al giorno.

20

## 10) richiesta visualizzazione stato del pacco

| Concetto   | Costrutto | Accessi | Tipo |
|------------|-----------|---------|------|
| Spedizione | Entità    | 1       | L    |
| oggetto    | relazione | 1       | L    |
| pacco      | entità    | 1       | L    |

|                  |           |   |   |
|------------------|-----------|---|---|
| stato spedizione | relazione | 1 | L |
| fase             | entità    | 1 | L |

avendo 5 accessi in lettura il costo della singola operazione è pari a 5 accessi, con una frequenza di 300 operazioni al giorno si hanno in totale 1500 accessi al giorno.

5

11) comunicazione posizione veicolo

| Concetto | Costrutto | Accessi | Tipo |
|----------|-----------|---------|------|
| veicolo  | entità    | 1       | L    |

avendo solo 1 accesso in lettura il costo della singola operazione è pari a 1 un accesso, con una frequenza di 34560000 operazioni al giorno si hanno in totale 34560000 accessi al giorno.

10

12) report spedizione accettate

| Concetto         | Costrutto | Accessi | Tipo |
|------------------|-----------|---------|------|
| Spedizione       | Entità    | 200     | L    |
| oggetto          | relazione | 200     | L    |
| pacco            | entità    | 200     | L    |
| stato spedizione | relazione | 200     | L    |
| fase             | entità    | 200     | L    |

avendo 1000 accessi in lettura, con una frequenza di 1 operazione al giorno si hanno in totale 1000 accessi al giorno.

15

13) report spedizione consegnate

| Concetto         | Costrutto | Accessi | Tipo |
|------------------|-----------|---------|------|
| Spedizione       | Entità    | 200     | L    |
| oggetto          | relazione | 200     | L    |
| pacco            | entità    | 200     | L    |
| stato spedizione | relazione | 200     | L    |
| fase             | entità    | 200     | L    |

avendo 1000 accessi in lettura, con una frequenza di 1 operazione al giorno si hanno in totale 1000 accessi al giorno.

20

14) report entrate totali

| Concetto   | Costrutto | Accessi | Tipo |
|------------|-----------|---------|------|
| Spedizione | Entità    | 200     | L    |
| oggetto    | relazione | 200     | L    |

|                  |           |     |   |
|------------------|-----------|-----|---|
| pacco            | entità    | 200 | L |
| stato spedizione | relazione | 200 | L |
| fase             | entità    | 200 | L |
| affidata a       | relazione | 200 | L |
| centro operativo | entità    | 200 | L |
| pagamento        | relazione | 200 | L |

avendo 1600 accessi in lettura, con una frequenza di 1 operazione al giorno si hanno in totale 1600 accessi al giorno.

5

### Ristrutturazione dello schema E-R

10 •Analisi ridonanze:

le uniche ridondanze presenti nello schema sono le 2 associazioni derivabile “pagamento” e “assegnamento”. Si decide di tenere entrambe le ridondanze in quanto portando un beneficio dal punto di vista del numero degli accessi alla base di dati.

Mettiamo a confronto l’operazione di pagamento della spedizione in presenza e in assegna della  
15 ridonanza:

con l’associazione “pagamento”

| Concetto         | Costrutto | Accessi | Tipo |
|------------------|-----------|---------|------|
| Centro operativo | Entità    | 1       | L    |
| spedizione       | entità    | 1       | L    |
| pagamento        | relazione | 1       | S    |
| carta di credito | entità    | 1       | L    |

Si hanno 3 accessi in lettura ed 1 in scrittura il costo della singola operazione è pari a 5 accessi, moltiplicando per la frequenza di 200 operazioni al giorno si hanno in totale 1000 accessi al giorno.

20

Senza l’associazione “pagamento”

| Concetto         | Costrutto | Accessi | Tipo |
|------------------|-----------|---------|------|
| Centro operativo | Entità    | 1       | L    |
| affidata a       | relazione | 1       | L    |
| spedizione       | entità    | 1       | L    |
| richiesta        | relazione | 1       | L    |
| cliente          | entità    | 1       | L    |
| titolarità       | relazione | 1       | L    |
| carta di credito | entità    | 1       | L    |

si hanno 7 accessi in lettura, moltiplicando per la frequenza di 200 operazioni al giorno si hanno in totale 1400 accessi al giorno.

5

Analizzando invece l'operazioni di assegnamento dei veicoli ai pacchi:

in presenza dell'associazione "assegnamento"

| Concetto                | Costrutto           | Accessi | Tipo |
|-------------------------|---------------------|---------|------|
| Veicolo<br>assegnamento | Entità<br>relazione | 1       | L    |
|                         |                     | 1       | S    |

- 10 si ha un accesso in lettura ed uno in scrittura, il costo della singola operazione è pari a 3 accessi. Moltiplicando per la frequenza di 400 operazioni al giorno si hanno in totale 1200 accessi al giorno.

senza l'associazione "assegnamento"

| Concetto  | Costrutto | Accessi | Tipo |
|---|-----------|---------|------|
| Veicolo<br>possiede<br>centro operativo<br>affidata a<br>spedizione<br>oggetto<br>pacco | Entità    | 1       | L    |
|   | relazione | 1       | L    |
|   | entità    | 1       | L    |
|   | relazione | 1       | L    |
|   | entità    | 1       | L    |
|   | relazione | 1       | L    |
|   | entità    | 1       | L    |

- 15 si hanno 7 accessi in lettura, moltiplicando per la frequenza di 400 operazioni al giorno si hanno in totale 2800 accessi al giorno, ovvero più del doppio che nel caso in cui sia presente l'associazione "assegnamento".

- 20 • Eliminazione delle generalizzazioni:

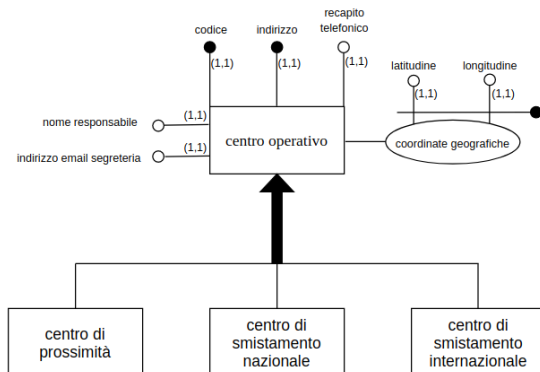
le generalizzazioni da eliminare sono 3, quella su "spedizione", quella su "centro operativo" e quella su "veicolo".

Per tutte e 3 le generalizzazioni si sceglie di accorpate le entità figlie nelle entità padre, per fare questo viene aggiunto un attributo all'entità padre che servirà a distinguere il tipo di occorrenza a cui

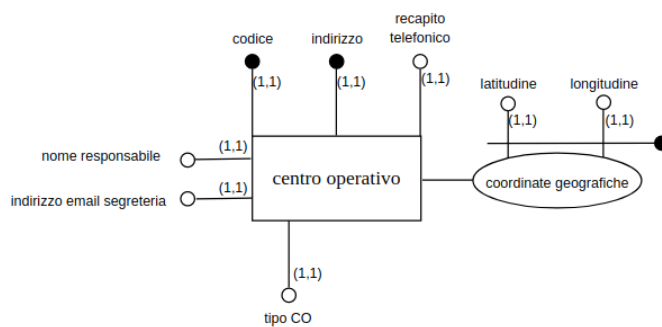
- 25 si fa riferimento. Nel caso di "spedizione" l'attributo aggiuntivo servirà ad identificare una spedizione nazionale da una internazionale, nel caso di "centro operativo" l'attributo servirà a distinguere il 3 tipi di centro operativo: centro di prossimità, centro di smistamento nazionale e centro

di smistamento internazionale. Infine nell'entità veicolo l'attributo aggiuntivo verrà usato per distinguere il tipo di veicolo a cui si fa riferimento, furgone o autoarticolato.

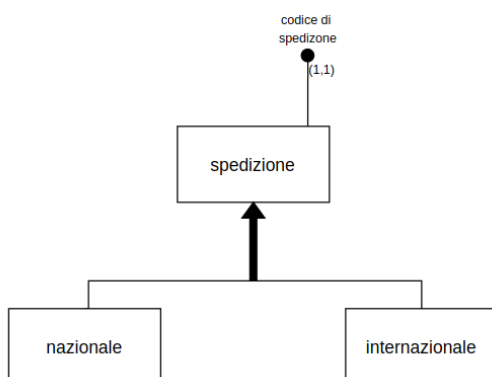
Centro operativo:



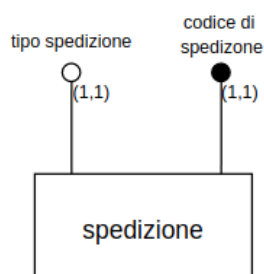
5 diventa



spedizione:

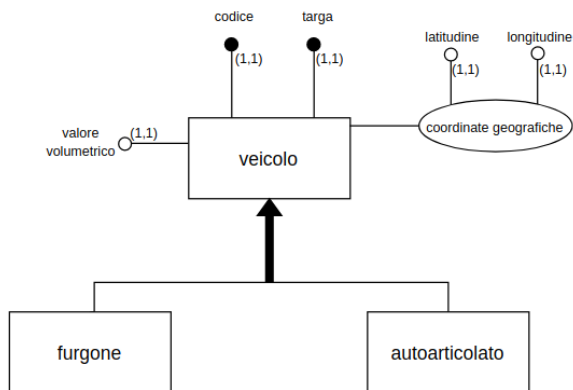


diventa

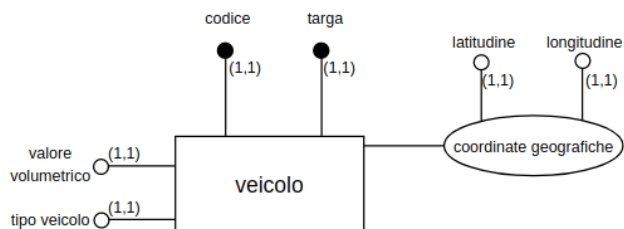




veicolo:



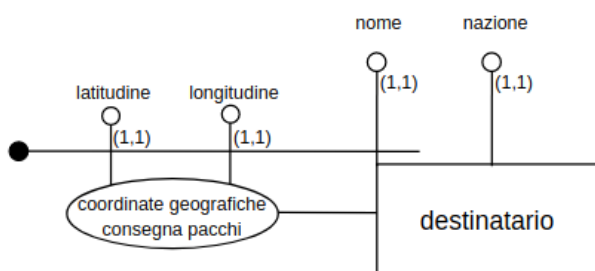
diventa



5

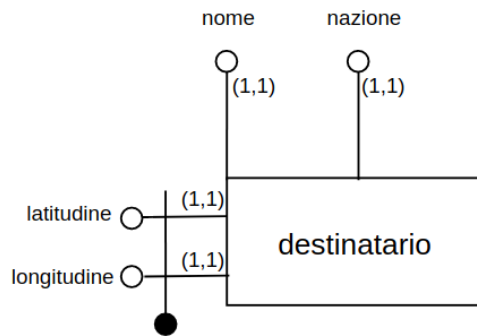
- eliminazione attributi composti e attributi multivalore:

l'entità "destinatario" ha come attributo composto "coordinate geografiche consegna pacchi"

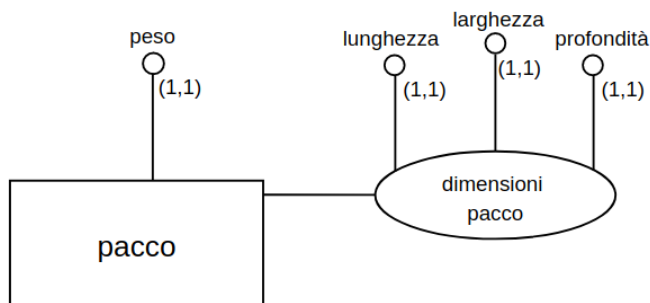


per l'eliminazione dell'attributo composto si sceglie di associare gli attributi componenti direttamente all'associazione "destinatario", che diventa quindi:

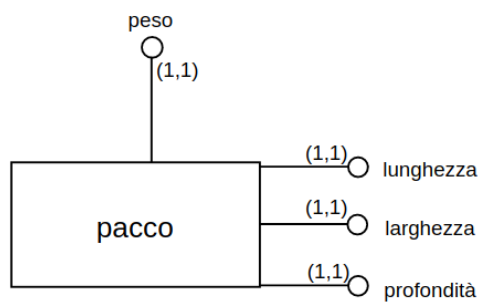
10



l'entità pacco ha come attributo composto "dimensioni pacco"

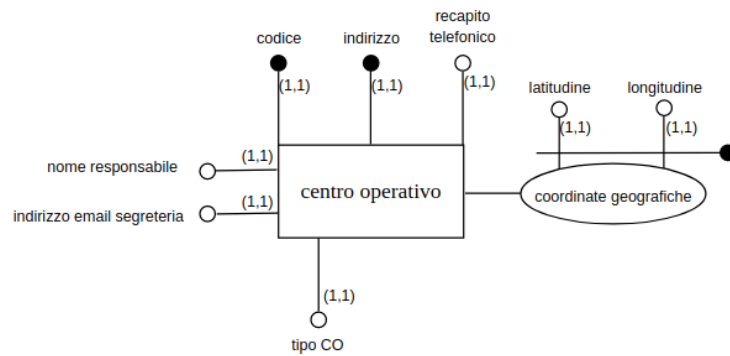


- 5 come per il caso precedente si sceglie di eliminare l'attributo composto e associare direttamente gli attributi all'entità "pacco".

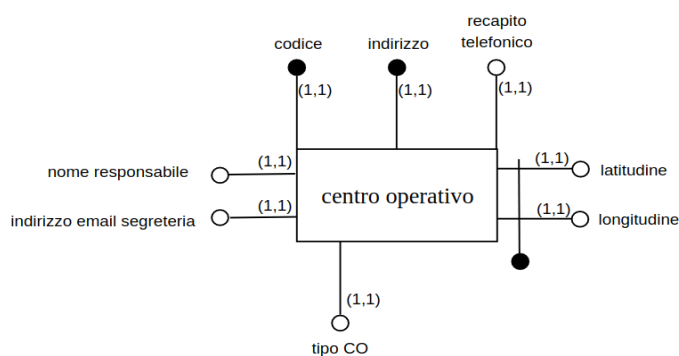


10

l'entità centro operativo ha come attributo composto "coordinate geografiche"

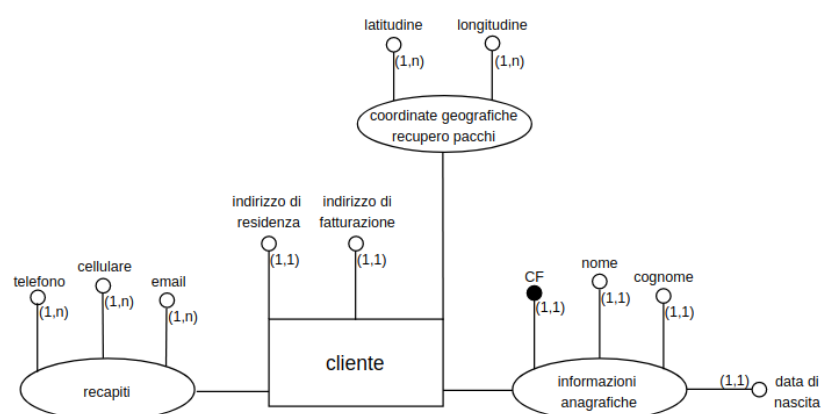


si sceglie anche in questo caso di eliminare l'attributo composto "coordinate geografiche" e associare gli attributi direttamente all'entità "centro operativo".

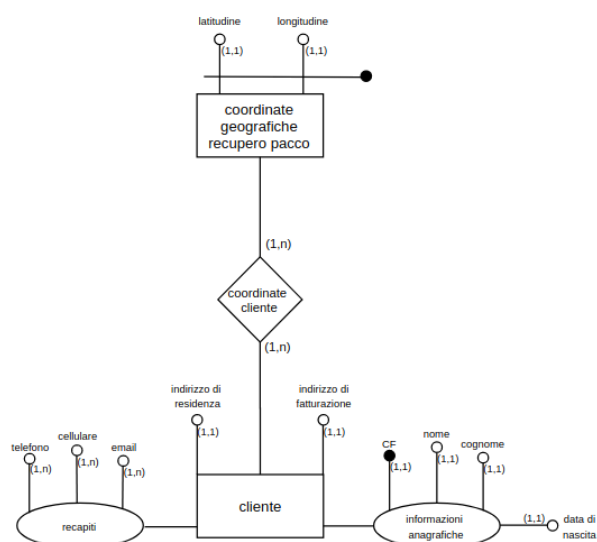


5

Per l'entità "cliente" si ha l'attributo composto "coordinate geografiche recupero pacchi" in cui gli attributi hanno cardinalità (1,n):



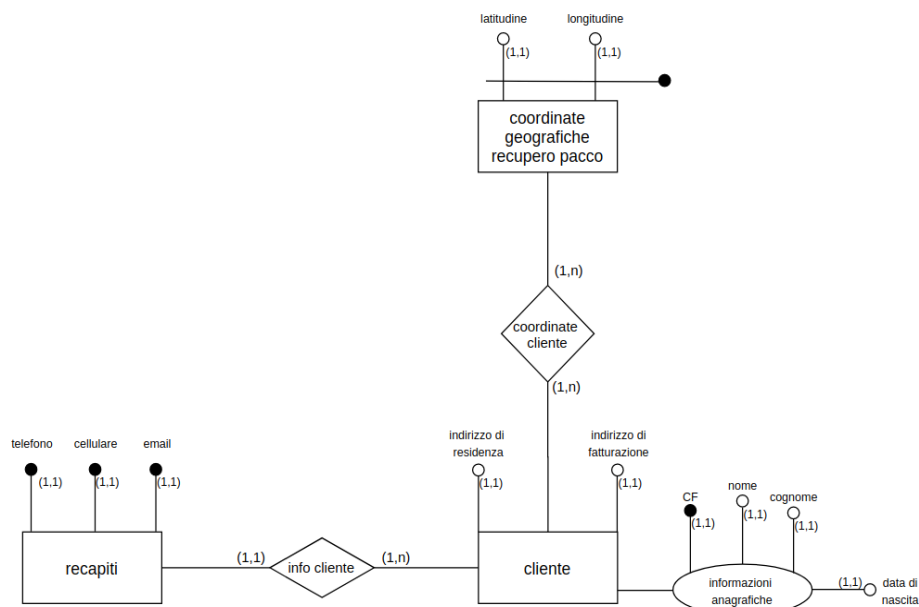
in questo caso si decide di creare una nuova entità “coordinate geografiche recupero pacchi” in cui verranno associati I 2 attributi “latitudine” e “longitudine”. A sua volta l’entità “coordinate geografiche recupero pacchi” sarà associata all’entità “cliente” tramite l’associazione “coordinate cliente”



l’associazione “coordinate cliente” è del tipo molti a molti in quanti un cliente può avere un numero arbitrario di indirizzi di recupero pacchi. Si presuppone inoltre che una coordinata geografica possa essere utilizzata da più di un cliente.

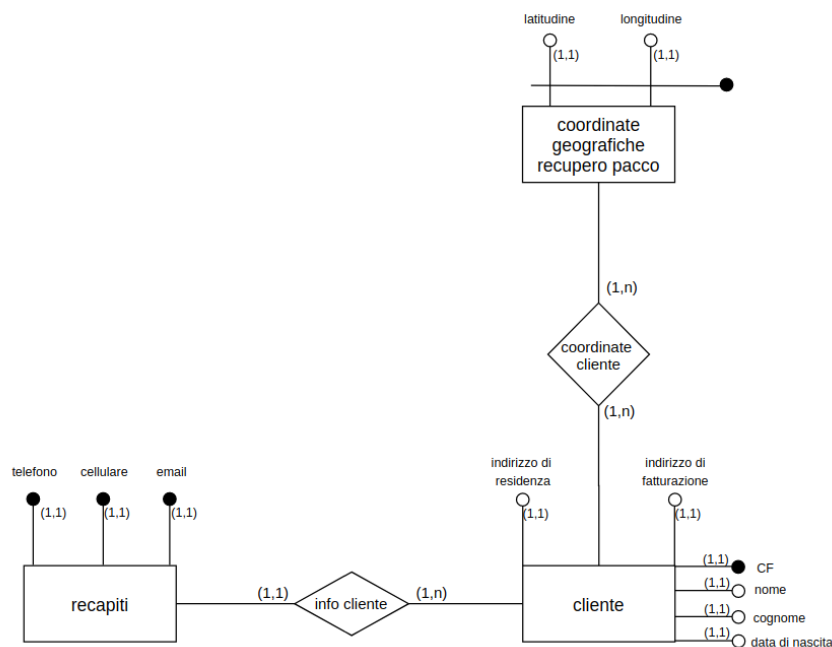
10

Si elimina poi l’attributo “recapiti”. Anche in questo caso gli attributi hanno cardinalità massima n, per cui si aggiunge un’entità “recapiti” a cui sono associati gli attributi “telefono”, “cellulare” e “email”. Le 2 entità sono poi messe in relazione tramite l’associazione “info cliente”:



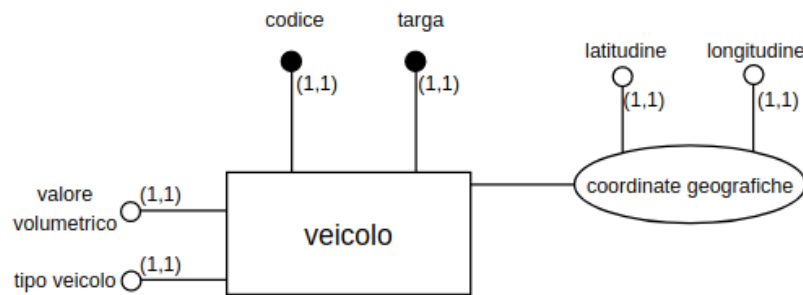
l'associazione "info cliente" è del tipo uno a molti in quanto un cliente può avere più di un recapito mentre un recapito può appartenere ad un solo cliente.

- 5 Si elimina infine l'attributo composto "informazioni anagrafiche", in questo caso si sceglie di associare direttamente gli attributi all'entità cliente.

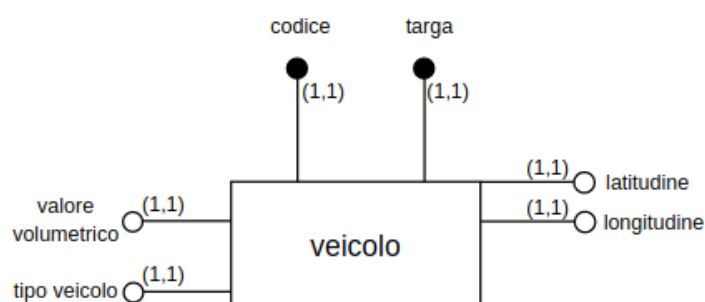


l'entità "veicolo" ha come

20 attributo composto "coordinate geografiche":



si sceglie anche in questo caso di associare i 2 attributi direttamente all'entità "veicolo":



5

- scelta identificatori primari

l'entità veicolo ha come identificatori codice e targa. Tra i due viene scelto come identificatore primario "codice". Questo perché memorizzare una targa richiede 6 byte mentre memorizzare il codice, sapendo che l'entità veicolo ha circa 2000 occorrenze, richiede solo 2 byte.

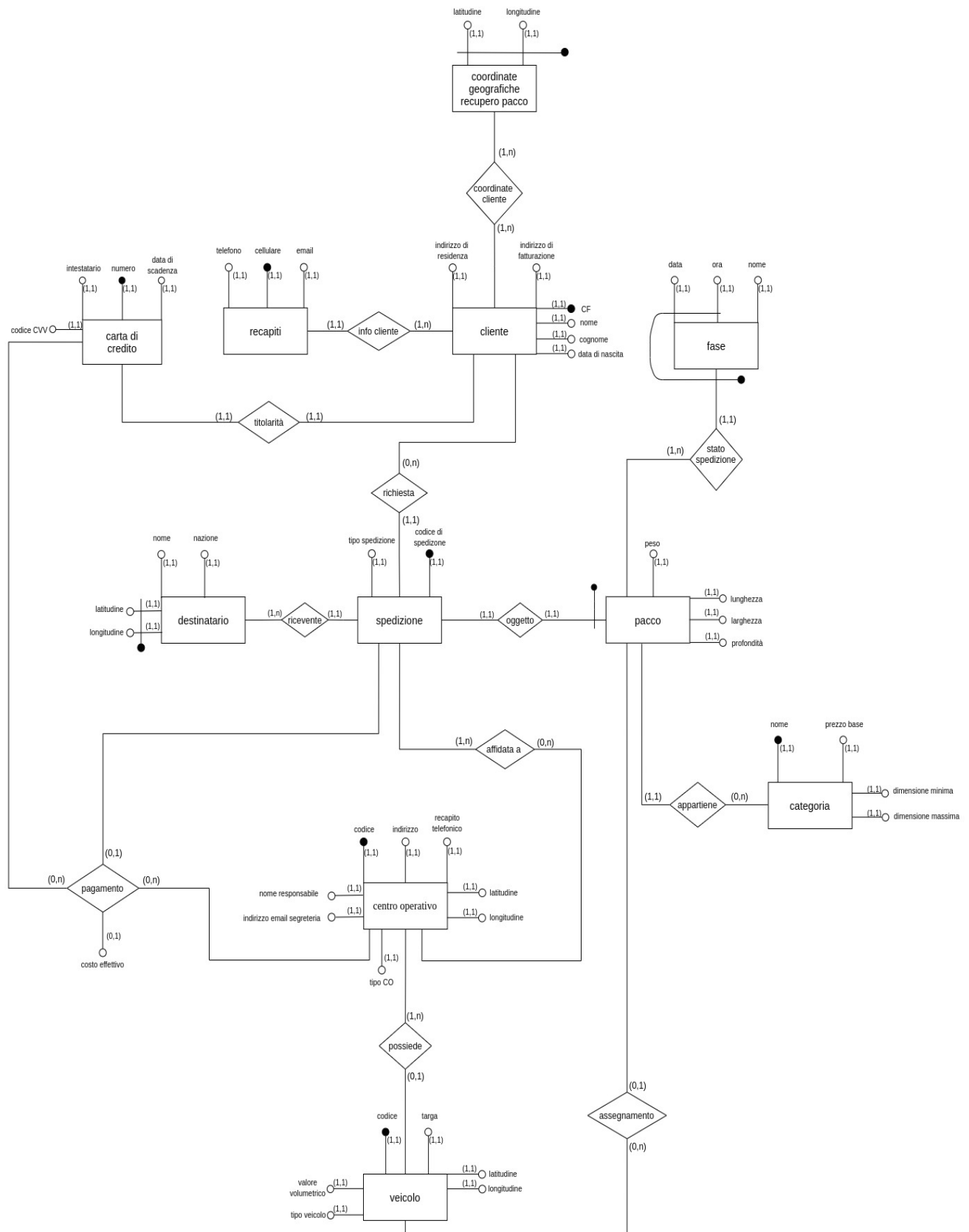
10

l'entità centro operativo ha come identificatori "codice", "indirizzo" e "latitudine e longitudine". Per lo stesso motivo dell'entità veicolo si sceglie come identificatore primario "codice".

15

l'entità "recapiti" ha come identificatori "telefono", "cellulare", "email". Sempre per motivi di occupazione di memoria converrebbe scegliere come identificatore primario "telefono" o "cellulare", in questo caso si sceglie "cellulare".

Schema E-R ristrutturato:



## Trasformazione di attributi e identificatori

Per facilitare la traduzione verso il modello relazionale si decide di fondere l'entità "coordinateGeograficheRecuperoPacco" con la relazione "coordinateCliente" in quanto si avrebbero 2 relazione in cui però la relazione "coordinateGeograficheRecuperoPacco" non aggiungerebbe  
5 informazioni rispetto a "coordinateCliente".

## Traduzione di entità e associazioni

l'entità "cliente" è in associazione molti a molti con l'entità "coordinate geografiche recupero pacco", e in associazione uno a molti con l'entità "recapiti".

Si decide quindi di tradurre l'entità cliente come:

- 10 • Cliente ( CF, nome, cognome, dataDiNascita, indirizzoDiFatturazione, IndirizzoDiResidenza)  
la relazione "info cliente" ha come chiave l'identificatore di "recapiti", si decide quindi di fonderle insieme in quanto esiste una corrispondenza biunivoca tra le occorrenze:

• recapiti ( cellulare, telefono, email, CFcliente)

esiste il vincolo di integrità referenziale tra CFcliente dell'entità "recapiti" e CF dell'entità "cliente".

- 15 La relazione "coordinate cliente" viene tradotta come:

•coordinateCliente (CFcliente, latitudineCGR, longitudineCGR)

con I vincoli di integrità referenziale tra "CFcliente" della relazione "coordinateCliente" e "CF" dell'entità "cliente".

- 20 Per l'associazione uno a uno "titolarità" tra "cliente" e "carta di credito" si decide di tradurre "carta di credito" nel seguente modo:

• cartaDiCredito (numero, intestatario, dataDiScadenza, codice CVV, clienteTitolare)

con il vincolo di integrità referenziale tra "clienteTitolare" dell'entità "carta di credito" e "CF" dell'entità "cliente".

25

Per quanto riguarda la relazione "richiesta", del tipo uno a molti, si decide di rappresentare l'associazione "richiesta" tramite l'entità spedizione.

La stessa cosa viene fatta per la relazione "ricevente".

• Spedizione (codiceSpedizione, tipoSpedizione, CFcliente, latitudineDes, longitudineDes)

- 30 esistono quindi I vincoli di integrità referenziale tra l'attributo "CFcliente" dell'entità spedizione e "CF" dell'entità "cliente", e "latitudineDes" e "longitudineDes" con "latitudine" e "longitudine" dell'entità "destinatario".



- Destinatario (latitudine, longitudine, nome, nazione)

La relazione “oggetto” è del tipo uno a uno, inoltre l’entità “pacco” ha un identificato esterno.

5 In questo caso si procede fondendo la relazione “oggetto” nell’entità “pacco” e includendo l’identificatore dell’entità “spedizione” all’interno di “pacco”.

La relazione “appartiene” è del tipo uno a molti, si decide quindi di unire la relazione con l’entità “pacco”.

- Pacco ( codiceSP, peso, lunghezza, larghezza, profondità, nomeCategoria)

10 con il vincolo di integrità referenziale tra l’attributo “codicePS” di “pacco” e l’attributo “codiceSpedizione” di spedizione e tra l’attributo “nomeCategoria” e l’attributo “nome” dell’entità categoria.

- Categoria (nome, prezzoBase)

15 La relazione “stato spedizione” è del tipo uno a molti, in particolare ha quindi una relazione biunivoca con l’entità fase. Si decide quindi di fondere la relazione all’interno dell’entità.

- Fase ( data, ora, codiceSP, nome)

I vincoli di integrità referenziale sono tra l’attributo “codiceSP” e l’attributo “codiceSpedizione” dell’entità “spedizione”.

20 La relazione “affidata a” è del tipo molti a molti, si mantiene quindi la relazione aggiungendo come attributi gli identificatori delle entità “spedizione” e “centro operativo”.

- AffidataA (codiceSP, codiceCO)

I vincoli di integrità sono tra l’attributo “codiceSP” e l’attributo “codiceSpedizione” dell’entità “spedizione” e tra “codiceCO” e l’attributo “codiceCO” dell’entità “centroOperativo”.

25

l’entità centro operativo viene tradotta come:

- centroOperativo ( codiceCO, indirizzo, recapitoTelefonico, latitudine, longitudine, nomeResponsabile, EmailSegreteria, tipoCO)

30 la relazione “pagamento” è del tipo molti a molti, viene quindi tradotta come:

- pagamento ( codiceSP, codiceCO, numeroCarta, costoEffettivo)

con I vincoli di integrità referenziale tra “codiceCO” e l’attributo “codiceCO” dell’entità “centroOperativo”, tra “numeroCarta” e l’attributo “numero” dell’entità “cartaDiCredito” e tra

“codiceSP” e l’attributo “codice\_spedizione” dell’entità “spedizione”.

La relazione “possiede” è del tipo uno a molti, si decide di non fondere la relazione nell’entità  
5 “veicolo” in modo da evitare la presenza di valori nulli derivati dal fatto che un veicolo può non essere posseduto da nessun centro operativo. Si traduce quindi come:

- possiede ( codiceCO, codiceV)

I vincoli di integrità sono tra “codiceCO” e l’attributo “codiceCO” dell’entità “centroOperativo” e tra  
“codiceV” e l’attributo “codiceV” dell’entità “veicolo”.

10

l’entità “veicolo” viene tradotta come:

- veicolo ( codiceV, targa, valoreVolumetrico, TipoVeicolo, latitudine, longitudine)

la relazione “assegnamento” è del tipo molti a molti, si traduce quindi come:

- 15 • assegnamento (codiceV, codiceSP)

con I vincoli di integrità referenziale tra “codiceV” e l’attributo “codiceV” dell’entità “veicolo” e tra  
“codiceSP” e l’attributo “codiceSpedizione” dell’entità “spedizione”.

20 Riassumendo:

- Cliente ( CF, nome, cognome, dataDiNascita, indirizzoDiFatturazione, IndirizzoDiResidenza)
- recapiti ( cellulare, telefono, email, CFcliente)
- coordinateCliente (CFcliente, latitudineCGR, longitudineCGR)
- coordinateGeoRecPacchi ( latitudine, longitudine)
- 25 • cartaDiCredito (numero, intestatario, dataDiScadenza, codice CVV, clienteTitolare)
- Spedizione (codiceSpedizione, tipoSpedizione, CFcliente, latitudineDes, longitudineDes)
- Destinatario (latitudine, longitudine, nome, nazione)
- Pacco ( codiceSP, peso, lunghezza, larghezza, profondità, nomeCategoria)
- Categoria (nome, prezzoBase)

30 • Fase ( data, ora, codiceSP, nome)

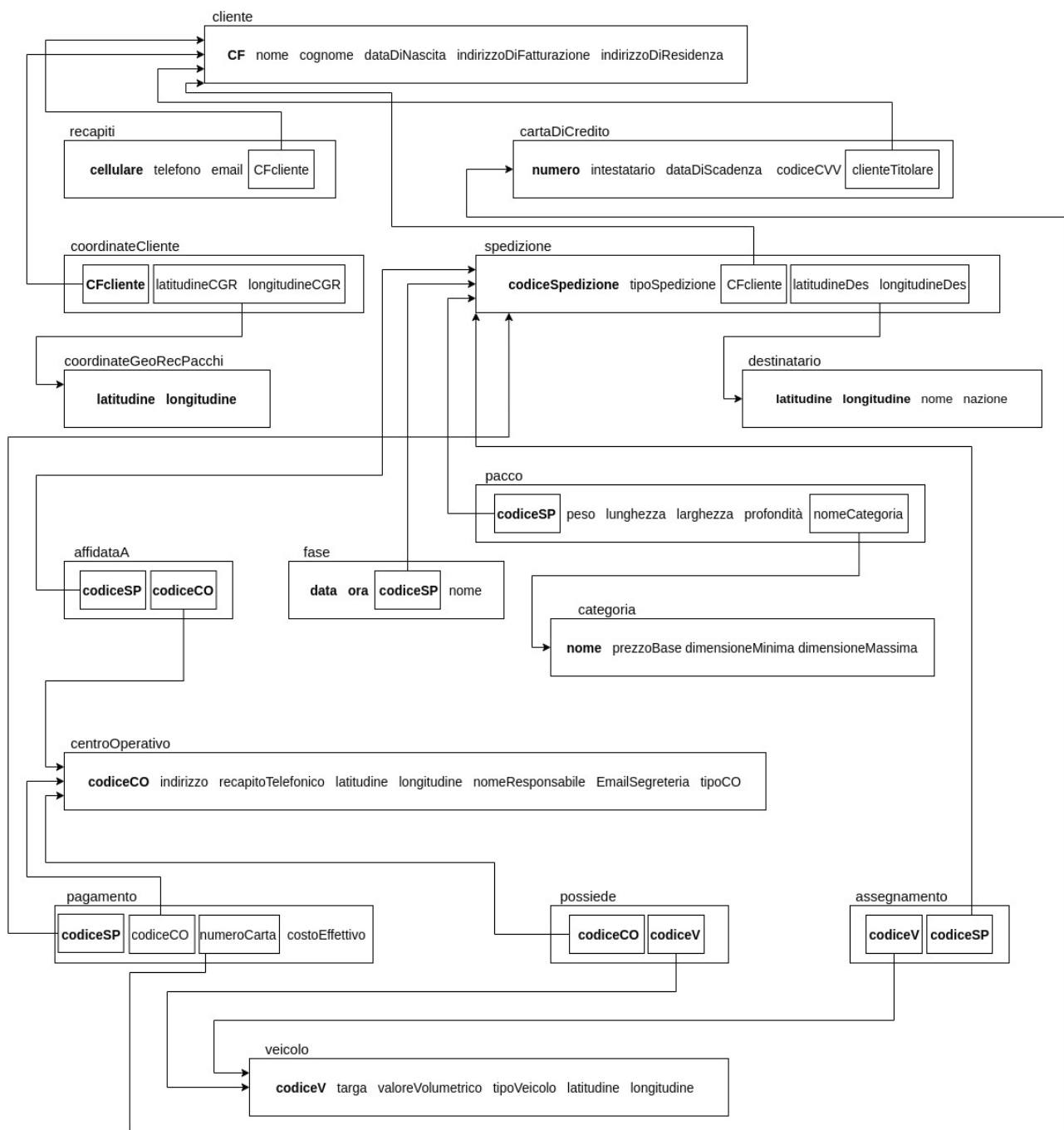
- AffidataA (codiceSP, codiceCO)

• centroOperativo ( codiceCO, indirizzo, recapitoTelefonico, latitudine, longitudine,  
nomeResponsabile, EmailSegreteria, tipoCO)

- pagamento ( codiceSP, codiceCO, numeroCarta, costoEffettivo)
- possiede ( codiceCO, codiceV)
- veicolo ( codiceV, targa, valoreVolumetrico, TipoVeicolo, latitudine, longitudine)
- assegnamento (codiceV, codiceSP)

5

Rappresentazione grafica dello schema:



5

## Normalizzazione del modello relazionale

La relazione cartaDiCredito non rispetta la prima forma normale in quanto l'attributo "intestatario" è  
10 un attributo composto da "nome" e "cognome" dell'intestatario. La relazione diventa quindi:  
cartaDiCredito (numero, nome, cognome, dataDiScadenza, codiceCVV, clienteTitolare)

la relazione cliente non rispetta la prima forma normale in quanto gli attributi  
"indirizzoDiFatturazione" e "indirizzoDiResidenza" sono composti da "via", "città", "cap" e  
15 "provincia". La relazione diventa quindi:  
cliente (CE, nome, cognome, dataDiNascita, viaResidenza, cittàResidenza, capResidenza,  
provinciaResidenza, viaFatturazione, cittàFatturazione, capFatturazione, provinciaFatturazione)

la relazione centroOperativo non è in prima forma normale in quanto l'attributo "indirizzo" è  
20 composto da "via", "città", "cap" e "provincia". La relazione diventa quindi:  
centroOperativo (codiceCO, via, città, cap, provincia, recapitoTelefonico, latitudine, longitudine,  
nomeResponsabile, emailSegreteria, tipoCO)

la relazione destinatario non è in prima forma normale in quanto l'attributo "nome" è composto da  
25 "nome" e "cognome". La relazione diventa quindi:  
destinatario (latitudine, longitudine, nome, cognome, nazione)

adesso lo schema rispetta interamente la prima forma normale, infatti non sono presenti tabelle con  
attributi composti, non sono presenti tabelle con colonne ripetute ed è presente una chiave primaria  
30 per ogni tabella.

Lo schema rispetta interamente la seconda forma normale in quanto è in prima forma normale e tutti  
gli attributi di ogni tabella sono funzionalmente dipendenti dall'intera chiave primaria e non solo da

una parte di essa.

Lo schema rispetta interamente la terza forma normale in quanto è in seconda forma normale e gli attributi di tutte le tabelle che non sono chiavi dipendono esclusivamente dalla chiave stessa. In altre

5 parole nessun attributo presenta una dipendenza transitiva rispetto alla chiave.

## 5. Progettazione fisica

### Utenti e privilegi

all'interno dell'applicazione sono stati previsti 4 tipi di utenti, il cliente, l'amministratore, il centro e il veicolo.

- 5 L'utente "amministratore" ha accesso alle procedure "report\_giornaliero" e a "modifica\_categoria", di conseguenza ha accesso indiretto alla tabella "categoria" per poter andare ad aggiungere o aggiornare le categoria relative ad I pacchi. Ha poi accesso tramite la procedura "report giornaliero", che utilizza le 2 view "spedizioni\_consegnate\_24H" e "spedizioni\_accettate\_24H", di conseguenza ha accesso indiretto in lettura sulla tabella "fase" e su "costo\_spedizione" per riuscire a fornire I
- 10 report giornalieri delle spedizioni consegnate, spedizioni accettate ed entrate.

L'utente "veicolo" è un utente che ha l'unico scopo di connettersi al DB e aggiornare la posizione del veicolo corrispondente all'utente che ha effettuato l'accesso. Ha quindi solo accesso alla procedura "aggiorna\_posizione" che andrà a fare un update sulla tabella "veicolo".

15

L'utente "centro" ha accesso alle procedure "inserimento\_giacenza\_pacchi", "check\_tipo\_spedizione" e "conferma\_addebita". Tramite la procedura "check\_tipo\_co" l'utente centro connesso dal client sarà in grado di ricevere informazioni a riguardo del proprio tipo di centro, andando quindi ad interrogare la tabella "centro\_operativo". La procedura

20 "inserimento\_giacenza\_pacchi" permette al centro di inserire un nuovo pacco all'interno della tabella giacenza\_pacchi in modo da segnalare l'arrivo di un nuovo pacco.

Tramite "conferma\_addebita" può confermare il prezzo della spedizione inserire l'istanza di pagamento all'interno della tabella "pagamento".

- 25 L'utente "login" ha come scopo quello di gestire il login tramite il client e quello di permettere di registrare nuovi clienti, ha quindi accesso alla procedura "login" che andrà ad interrogare la tabella "utenti" e alla procedura "registra\_nuovo\_cliente" per inserire una nuova istanza di "cliente" e quindi accedere alle tabelle "cliente", "carta\_di\_credito", "coordinate\_cliente".

- 30 L'utente cliente può creare un nuovo ordine tramite la procedura "crea\_nuovo\_ordine" che andrà ad inserire una nuova istanza all'interno delle tabelle "pacco", "spedizione", "destinatario".

Tramite “aggiungi\_recapiti” può inserire i propri recapiti dentro al tabella “recapiti”.

La procedura “visualizza\_posizione\_pacco” permette poi di visualizzare la posizione del pacco accedendo quindi alle tabelle “veicolo”, “spedizione”.

Tramite la procedura “stato\_spedizione” può controllare lo stato del pacco, accedendo quindi alle tabelle “fase”, “spedizione”.

Tramite la procedura “aggiorna\_coordinate\_recupero\_pacco” il cliente può aggiornare le coordinate per il prossimo ritiro, facendo quindi un update sulla tabella “coordinante\_cliente”.

Infine la procedura “return\_cf” ha lo scopo di fornire al client connesso come “cliente” il proprio codice fiscale, accedendo quindi alla tabella “cliente”.

10

## Strutture di memorizzazione

| cliente                |              |                        |
|------------------------|--------------|------------------------|
| Attributo              | Tipo di dato | Attributi <sup>2</sup> |
| CF                     | CHAR(16)     | PK, NN                 |
| Nome                   | VARCHAR(45)  | NN                     |
| cognome                | VARCHAR(45)  | NN                     |
| data_di_nascita        | DATE         | NN, UN                 |
| via_residenza          | VARCHAR(90)  | NN                     |
| città_residenza        | VARCHAR(45)  | NN                     |
| cap_residenza          | INT          | NN, UN                 |
| provincia_residenza    | VARCHAR(45)  | NN                     |
| via_fatturazione       | VARCHAR(90)  | NN                     |
| città_fatturazione     | VARCHAR(45)  | NN                     |
| cap_fatturazione       | INT          | NN, UN                 |
| provincia_fatturazione | VARCHAR(45)  | NN                     |

<sup>2</sup> PK = primary key, NN = not null, UQ = unique, UN = unsigned, AI = auto increment. È ovviamente possibile specificare più di un attributo per ciascuna colonna.

| <b>carta_di_credito</b> |                     |                  |
|-------------------------|---------------------|------------------|
| <b>Attributo</b>        | <b>Tipo di dato</b> | <b>Attributi</b> |
| Numero                  | BIGINT              | PK, UN           |
| nome                    | VARCHAR(45)         | NN               |
| cognome                 | VARCHAR(45)         | NN               |
| data_di_scadenza        | DATE                | NN               |
| codice_cvv              | SMALLINT            | NN, UN           |
| cliente_titolare        | CHAR(16)            | NN               |

| <b>recapiti</b>  |                     |                  |
|------------------|---------------------|------------------|
| <b>Attributo</b> | <b>Tipo di dato</b> | <b>Attributi</b> |
| Cellulare        | INT                 | PK, UN           |
| telefono         | INT                 | NN, UN, UQ       |
| email            | VARCHAR(60)         | NN, UQ           |
| cf_cliente       | CHAR(16)            | NN               |

| <b>coordinate_cliente</b> |                     |                  |
|---------------------------|---------------------|------------------|
| <b>Attributo</b>          | <b>Tipo di dato</b> | <b>Attributi</b> |
| cf_cliente                | CHAR(16)            | PK               |
| latitudine_cgr            | FLOAT               | NN               |
| longitudine_cgr           | FLOAT               | NN               |

5

| <b>coordinate_geo_rec_pacchi</b> |                     |                  |
|----------------------------------|---------------------|------------------|
| <b>Attributo</b>                 | <b>Tipo di dato</b> | <b>Attributi</b> |
| Latitudine                       | FLOAT               | PK               |



|             |       |    |
|-------------|-------|----|
| longitudine | FLOAT | PK |
|-------------|-------|----|

| <b>spedizione</b> |                     |                  |
|-------------------|---------------------|------------------|
| <b>Attributo</b>  | <b>Tipo di dato</b> | <b>Attributi</b> |
| codice_spedizione | INT                 | PK, AI, UN       |
| tipo_spedizione   | VARCHAR(45)         | NN               |
| cf_cliente        | CHAR(16)            | NN               |
| latitudine_des    | FLOAT               | NN               |
| longitudine_des   | FLOAT               | NN               |

| <b>destinatario</b> |                     |                  |
|---------------------|---------------------|------------------|
| <b>Attributo</b>    | <b>Tipo di dato</b> | <b>Attributi</b> |
| Latitudine          | FLOAT               | PK               |
| longitudine         | FLOAT               | PK               |
| nome                | VARCHAR(45)         | NN               |
| cognome             | VARCHAR(45)         | NN               |
| nazione             | VARCHAR(45)         | NN               |

5

| <b>pacco</b>     |                     |                  |
|------------------|---------------------|------------------|
| <b>Attributo</b> | <b>Tipo di dato</b> | <b>Attributi</b> |
| codice_sp        | INT                 | PK, UN           |
| peso             | SMALLINT            | NN, UN           |
| lunghezza        | SMALLINT            | NN, UN           |
| larghezza        | SMALLINT            | NN, UN           |
| profondità       | SMALLINT            | NN, UN           |

|                |             |    |
|----------------|-------------|----|
| nome_categoria | VARCHAR(45) | NN |
|----------------|-------------|----|

| categoria          |              |           |
|--------------------|--------------|-----------|
| Attributo          | Tipo di dato | Attributi |
| Nome               | VARCHAR(45)  | PK        |
| prezzo_base        | FLOAT        | NN, UN    |
| dimensione_minima  | FLOAT        | NN        |
| dimensione_massima | FLOAT        | NN        |

5

| fase      |              |            |
|-----------|--------------|------------|
| Attributo | Tipo di dato | Attributi  |
| Data      | TIMESTAMP    | PK         |
| ora       | TIME         | PK, NN     |
| codice_sp | INT          | PK, NN, UN |
| nome      | VARCHAR(45)  | NN         |

| affidata_a |              |           |
|------------|--------------|-----------|
| Attributo  | Tipo di dato | Attributi |
| codice_sp  | INT          | PK, UN    |
| codice_co  | INT          | PK, UN    |

| centro_operativo |              |                |
|------------------|--------------|----------------|
| Attributo        | Tipo di dato | Attributi      |
| Codice_co        | INT          | PK, NN, AI, UN |
| via              | VARCHAR(90)  | NN             |

|                     |             |            |
|---------------------|-------------|------------|
| città               | VARCHAR(45) | NN         |
| cap                 | INT         | NN, UN     |
| provincia           | VARCHAR(45) | NN         |
| recapito_telefonico | INT         | NN, UN, UQ |
| latitudine          | FLOAT       | NN         |
| longitudine         | FLOAT       | NN         |
| nome_responsabile   | VARCHAR(45) | NN         |
| email_segreteria    | VARCHAR(60) | NN, UQ     |
| tipo_co             | VARCHAR(45) | NN         |

5

| pagamento       |              |            |
|-----------------|--------------|------------|
| Attributo       | Tipo di dato | Attributi  |
| codice_sp       | INT          | PK, NN, UN |
| codice_co       | INT          | NN, UN     |
| numero_carta    | BIGINT       | NN, UN     |
| costo_effettivo | FLOAT        | NN, UN     |

| possiede  |              |            |
|-----------|--------------|------------|
| Attributo | Tipo di dato | Attributi  |
| codice_co | INT          | PK, NN, UN |
| codice_v  | INT          | PK, NN, UN |

| assegnamento |              |            |
|--------------|--------------|------------|
| Attributo    | Tipo di dato | Attributi  |
| codice_v     | INT          | PK, NN, UN |
| codice_sp    | INT          | PK, NN, UN |

| veicolo            |              |                |
|--------------------|--------------|----------------|
| Attributo          | Tipo di dato | Attributi      |
| Codice_v           | INT          | PK, NN, UN, AI |
| targa              | CHAR(7)      | NN, UQ         |
| valore_volumetrico | FLOAT        | NN             |
| tipo_veicolo       | VARCHAR(45)  | NN             |
| latitudine         | FLOAT        | NN             |
| longitudine        | FLOAT        | NN             |

5

## Indici

Compilare la seguente tabella, per ciascuna tabella del database in cui sono presenti degli indici. Descrivere le motivazioni che hanno portato alla creazione di un indice.

| Tabella <cliente> |                     |
|-------------------|---------------------|
| Indice <PRIMARY>  | Tipo <sup>3</sup> : |
| cf                | <PR>                |
| Indice <username> | Tipo:               |
| username          | <UQ>                |

10

| Tabella <affidata_a> |       |
|----------------------|-------|
| Indice <PRIMARY>     | Tipo: |
| codice_co            | <PR>  |

<sup>3</sup> IDX = index, UQ = unique, FT = full text, PR = primary.

| Tabella <assegnamento> |       |
|------------------------|-------|
| Indice <PRIMARY>       | Tipo: |
| codice_sp              | <PR>  |

| Tabella <carta_di_credito> |       |
|----------------------------|-------|
| Indice <PRIMARY>           | Tipo: |
| numero                     | <PR>  |
| Indice <cliente_titolare>  | Tipo: |
| cliente_titolare           | <UQ>  |

| Tabella <categoria> |       |
|---------------------|-------|
| Indice <PRIMARY>    | Tipo: |
| nome                | <PR>  |

5

| Tabella <centro_operativo>   |       |
|------------------------------|-------|
| Indice <PRIMARY>             | Tipo: |
| codice_co                    | <PR>  |
| Indice <recapito_telefonico> | Tipo: |
| recapito_telefonico          | <UQ>  |
| Indice <email_segreteria>    | Tipo: |
| email_segreteria             | <UQ>  |

| Tabella <coordinate_cliente> |       |
|------------------------------|-------|
| Indice <PRIMARY>             | Tipo: |
| cf_cliente                   | <PR>  |

| Tabella <destinatario>                       |       |
|--|-------|
| Indice <PRIMARY>                             | Tipo: |
| Latitudine<br>longitudine<br>nome<br>cognome | <PR>  |

| Tabella <fase>                |       |
|-------------------------------|-------|
| Indice <PRIMARY>              | Tipo: |
| data_ora<br>codice_sp<br>nome | <PR>  |

| Tabella <giacenza_pacchi_co> |       |
|------------------------------|-------|
| Indice <PRIMARY>             | Tipo: |
| codice_co<br>codice_sp       | <PR>  |

5

| Tabella <pacco>  |       |
|------------------|-------|
| Indice <PRIMARY> | Tipo: |
| codice_sp        | <PR>  |

| Tabella <pagamento>   |       |
|-----------------------|-------|
| Indice <PRIMARY>      | Tipo: |
| codice_sp             | <PR>  |
| Indice <codice_co>    | Tipo: |
| codice_co             | <IDX> |
| Indice <numero_carta> | Tipo: |
| numero_carta          | <IDX> |

| Tabella <possiede>    |       |
|-----------------------|-------|
| Indice <PRIMARY>      | Tipo: |
| codice_co<br>codice_v | <PR>  |

| Tabella <recapiti> |       |
|--------------------|-------|
| Indice <PRIMARY>   | Tipo: |
| cellulare          | <PR>  |
| Indice <telefono>  | Tipo: |

|                                  |              |
|----------------------------------|--------------|
| codice_co                        | <UQ>         |
| <b>Indice &lt;email&gt;</b>      | <b>Tipo:</b> |
| email                            | <UQ>         |
| <b>Indice &lt;cf_cliente&gt;</b> | <b>Tipo:</b> |
| cf_cliente                       | <UQ>         |

|   |              |
|---|--------------|
| <b>Tabella &lt;riassegnazione_veicolo&gt;</b> |              |
| <b>Indice &lt;PRIMARY&gt;</b>                 | <b>Tipo:</b> |
| codice_sp<br>codice_co                        | <PR>         |

|                                      |              |
|--------------------------------------|--------------|
| <b>Tabella &lt;spedizione&gt;</b>    |              |
| <b>Indice &lt;PRIMARY&gt;</b>        | <b>Tipo:</b> |
| codice_spedizione                    | <PR>         |
| <b>Indice &lt;latitudine_des&gt;</b> | <b>Tipo:</b> |
| latitudine_des<br>longitudine_des    | <IDX>        |

|                               |              |
|-------------------------------|--------------|
| <b>Tabella &lt;utenti&gt;</b> |              |
| <b>Indice &lt;PRIMARY&gt;</b> | <b>Tipo:</b> |
| username                      | <PR>         |

|                                      |              |
|--------------------------------------|--------------|
| <b>Tabella &lt;veicolo&gt;</b>       |              |
| <b>Indice &lt;PRIMARY&gt;</b>        | <b>Tipo:</b> |
| codice_v                             | <PR>         |
| <b>Indice &lt;latitudine_des&gt;</b> | <b>Tipo:</b> |
| targa                                | <UQ>         |

## Trigger

- imposta\_categoria\_pacco

5 Trigger per aggiornare automaticamente il campo “nome\_categoria” di “pacco” in modo che all’inserimento di una nuova tupla “pacco” il trigger va a scorrere le varie categorie presenti nel sistema, preleva per ogni categoria il range di dimensione da rispettare per appartenere a quella determinata categoria e li confronta con la somma delle dimensioni del pacco appena inserito. Una volta trovato la categoria a cui appartiene il pacco il sistema aggiorna la tupla che si vuole inserire  
10 settando il valore “nome\_categoria” pari al nome della categoria trovata.

DELIMITER !

```
CREATE TRIGGER imposta_categoria_pacco
    BEFORE INSERT ON pacco FOR EACH ROW
    BEGIN
15    DECLARE done int DEFAULT FALSE;
    DECLARE var_nome VARCHAR(45);
    DECLARE dim_min FLOAT;
    DECLARE dim_max FLOAT;
    DECLARE somma FLOAT;
20    DECLARE cursore1 CURSOR FOR SELECT nome FROM categoria;
    DECLARE cursore2 CURSOR FOR SELECT dimensione_minima FROM categoria;
    DECLARE cursore3 CURSOR FOR SELECT dimensione_massima FROM categoria;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
    SET somma = new.larghezza + new.lunghezza + new.profondità;
25    open cursore1;
    open cursore2;
    open cursore3;
    read_loop: loop
        fetch cursore1 into var_nome;
30        fetch cursore2 into dim_min;
        fetch cursore3 into dim_max;
        if (somma > dim_min AND somma < dim_max) then
```



```

        SET new.nome_categoria = var_nome;
        leave read_loop;
    end if;
end loop;
5   close cursore1;
    close cursore2;
    close cursore3;
end !
DELIMITER ;

```

10

#### • assegnamento\_veicolo

trigger per verificare che un pacco non sia assegnato a più di un veicolo, l'assegnamento infatti è limitato al periodo in cui il pacco è nel veicolo, nel momento in cui il pacco verrà assegnato ad un nuovo veicolo il precedente assegnamento dovrà essere eliminato.

15

Se invece il pacco è in giacenza in qualche centro operativo non dovrà essere assegnato a nessun veicolo e quindi non dovrà essere presente all'interno della relazione "assegnamento".

```

DELIMITER //
20  CREATE TRIGGER assegnamento_veicolo
    BEFORE INSERT ON assegnamento FOR EACH ROW
    BEGIN
        DECLARE num INT;
        SELECT count(*) FROM assegnamento WHERE codice_sp = new.codice_sp INTO num;
25  IF (num >= 1) THEN
            SIGNAL SQLSTATE "45000";
        END IF;
    END //
DELIMITER ;

```

30

#### • scadenza\_carta\_di\_credito

trigger per controllare se all'inserimento della carta di credito questa sia effettivamente ancora in

corso di validità, ovvero che la data di scadenza della carta di credito sia posteriore alla data di registrazione della carta di credito.

DELIMITER //

```
5 CREATE TRIGGER scadenza_carta_di_credito
    BEFORE INSERT ON carta_di_credito FOR EACH ROW
    BEGIN
    IF ( CURRENT_TIMESTAMP > new.data_di_scadenza) THEN
        SIGNAL SQLSTATE "45000";
10     END IF;
    END //
```

DELIMITER ;

15 • indirizzo\_di\_fatturazione

trigger per compilare in modo automaticamente I campi corrispondenti all'indirizzo di fatturazione. In caso di mancanza dell'indirizzo di fatturazione infatti si assume quest'ultimo come uguale all'indirizzo di residenza.

```
20 DELIMITER //
CREATE TRIGGER inserimento_indirizzo_fatturazione
    BEFORE INSERT ON cliente FOR EACH ROW
    BEGIN
    IF (new.via_fatturazione IS NULL and new.città_fatturazione IS NULL and
25 new.cap_fatturazione IS NULL
        and new.provincia_fatturazione IS NULL) THEN
        SET new.via_fatturazione = new.via_residenza;
        SET new.città_fatturazione = new.città_residenza;
        SET new.cap_fatturazione = new.cap_residenza;
30     SET new.provincia_fatturazione = new.provincia_residenza;
    END IF;
    END //
```

DELIMITER ;

- verifica\_tipo\_spedizione

- 5 trigger che al momento dell’inserimento di una nuova spedizione va a interrogare l’attributo “nazione” del destinatario per stabilire se la spedizione è di tipo nazionale o internazionale. Una volta stabilito viene settato il campo “tipo\_spedizione”.

```
DELIMITER //
```

10

```
CREATE TRIGGER verifica_tipo_spedizione
```

```
BEFORE INSERT ON spedizione FOR EACH ROW
```

```
BEGIN
```

```
    DECLARE nazione_des VARCHAR(45);
```

15

```
    SELECT nazione
```

```
    FROM destinatario
```

```
    WHERE new.latitudine_des = latitudine AND new.longitudine_des = longitudine INTO  
nazione_des;
```

20

```
    IF (nazione_des = "italia") THEN
```

```
        SET new.tipo_spedizione = "nazionale";
```

```
    ELSE
```

```
        SET new.tipo_spedizione = "internazionale";
```

```
    END IF;
```

25

```
END //
```

```
DELIMITER ;
```

30

- controllo\_cf

trigger per andare a fare il check sul codice fiscale inserito e controllare se la sintassi corrisponda a

quella standard per un codice fiscale.

DELIMITER //

CREATE TRIGGER controllo\_cf

5        BEFORE INSERT ON cliente FOR EACH ROW

        BEGIN

         IF NOT new.cf regexp '^[A-Z]{6,6}[0-9]{2,2}[A-Z][0-9]{2,2}[A-Z][0-9]{3,3}[A-Z]

\$' THEN

             SIGNAL SQLSTATE '45000';

10        END IF;

        END //

DELIMITER ;

15    • controllo\_targa

trigger per andare a controllare la targa inserita per I veicolo dei centri operativi.

DELIMITER //

CREATE TRIGGER controllo\_targa

        BEFORE INSERT ON veicolo FOR EACH ROW

20        BEGIN

         IF NOT new.targa regexp '^[A-Za-z]{2}[0-9]{3}[A-Za-z]{2}\$' THEN

             SIGNAL SQLSTATE '45000';

         END IF;

        END //

25    DELIMITER ;

• controllo\_email\_centro

trigger per il controllo sull'inserimento dell'email per I centri operativi.

30    DELIMITER //

CREATE TRIGGER controllo\_email\_centro

        BEFORE INSERT ON centro\_operativo FOR EACH ROW

        BEGIN

```

        IF NOT new.email_segreteria regexp '[a-zA-Z0-9][a-zA-Z0-9._-]*[a-zA-Z0-9._-]@[a-zA-Z0-9][a-zA-Z0-9._-]*[a-zA-Z0-9]\.[a-zA-Z]{2,63}$' THEN
            SIGNAL SQLSTATE '45000';
        END IF;
5      END //
DELIMITER ;

```

• controllo\_email\_recapiti

10 trigger per il controllo sull'inserimento dell'email per I clienti.

```
DELIMITER //
```

```
CREATE TRIGGER controllo_email_recapiti
```

```
    BEFORE INSERT ON recapiti FOR EACH ROW
```

```
    BEGIN
```

```
15      IF NOT new.email regexp '[a-zA-Z0-9][a-zA-Z0-9._-]*[a-zA-Z0-9._-]@[a-zA-Z0-9][a-zA-Z0-9._-]*[a-zA-Z0-9]\.[a-zA-Z]{2,63}$' THEN
```

```
          SIGNAL SQLSTATE '45000';
```

```
        END IF;
```

```
    END //
```

```
20 DELIMITER ;
```

## Eventi

• report

25 Evento per la generazione dei report giornalieri contenente informazioni a riguardo delle spedizioni accettate, delle spedizioni consegnate e delle entrate totali delle spedizioni consegnate a valle dell'aggiornamento del prezzo da parte del centro operativo. L'evento resta attivo durante tutta la vita del DB.

```
SET GLOBAL EVENT_SCHEDULER = ON;
```

30

```
CREATE EVENT report
```

```
ON SCHEDULE EVERY 1 day
ON COMPLETION PRESERVE
DO
```

```
5      SELECT sp_accettate, sp_consegnate, entrate_totali
      FROM spedizioni_accettate_24H, spedizioni_consegnate_24H;
```

• riassegnazione

10 Evento per andare a riassegnare I pacchi che precedentemente sono stati messi “in attesa di smistamento” a causa dell’assenza di un veicolo libero per la loro spedizione. L’evento resta attivo durante tutta la vita del DB.

```
SET GLOBAL EVENT_SCHEDULER = ON;
```

```
15 DELIMITER //
CREATE EVENT riassegnazione
ON SCHEDULE EVERY 1 day
ON COMPLETION PRESERVE
DO
```

```
20     BEGIN
        DECLARE done INT DEFAULT FALSE;
        DECLARE cod_sp INT;
        DECLARE cod_co INT;
        DECLARE type VARCHAR(45);
```

```
25         DECLARE cursore1 CURSOR FOR SELECT codice_sp FROM
riassegnazione_veicolo;
```

```
        DECLARE cursore2 CURSOR FOR SELECT codice_co FROM
riassegnazione_veicolo;
```

```
30         DECLARE cursore3 CURSOR FOR SELECT tipo FROM riassegnazione_veicolo;
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
```

```
        open cursore1;
```

```
open cursore2;
open cursore3;

read_loop: loop
5      fetch cursore1 INTO cod_sp;
      fetch cursore2 INTO cod_co;
      fetch cursore3 INTO type;

      IF done THEN
10          LEAVE read_loop;
      END IF;

      CALL riassegnazione_veicolo_pro(cod_sp, cod_co, type);
      DELETE FROM riassegnazione_veicolo WHERE codice_sp = cod_sp AND
15  codice_co = cod_co;
      END loop;
      close cursore1;
      close cursore2;
      close cursore3;
20  END //
DELIMITER ;
```

## Viste

- posizione\_pacco

Vista che crea una tabella contenente le informazioni associate alla posizione (latitudine e  
25 longitudine) di ogni pacco che si trovi in transito in quel momento.

Se un pacco non è presente nella seguente lista significa che in quel preciso momento non è associato a nessun veicolo (la relazione “assegnamento” non contiene il codice\_sp associato a quel pacco), ovvero il pacco non è nella fase “in transito”. Un pacco infatti è presente nella relazione “assegnamento” solo se associato ad un veicolo.

```
30 CREATE VIEW posizione_pacco (codice_spedizione, latitudine, longitudine) AS
SELECT pacco.codice_sp, latitudine, longitudine
```

```
FROM pacco JOIN assegnamento on assegnamento.codice_sp = pacco.codice_sp
      JOIN veicolo ON veicolo.codice_v = assegnamento.codice_v
WHERE pacco.codice_sp = assegnamento.codice_sp;
```

5

- costo\_spedizione

vista che crea una tabella in cui viene associato ad ogni pacco il costo stimato di spedizione calcolato in base al prezzo base della categoria di appartenenza del pacco e poi modificato in base al peso del pacco e il costo effettivo che verrà aggiornato dal centro di prossimità.

10 In questo caso la variazione associata al peso viene fatto semplicemente moltiplicando il prezzo base della categoria per il valore del peso.

Viene effettuato il left join per la tabella “pagamento” in quanto questa non avrà la entry associata alla spedizione fino all’arrivo del pacco presso il centro di prossimità che provvederà ad inserire il costo effettivo della spedizione.

15

```
CREATE VIEW costo_spedizione (codice_spedizione, stima_costo, costo_effettivo) AS
SELECT pacco.codice_sp, prezzo_base*peso, costo_effettivo
FROM pacco JOIN categoria ON nome = nome_categoria
      LEFT JOIN pagamento ON pagamento.codice_sp = pacco.codice_sp;
```

20

- spedizioni\_accettate\_24H

vista che riporta il numero di spedizioni accettate nell’arco delle ultime 24 ore, viene usata in particolare per il calcolo del report giornaliero.

25

Viene usata la clausola where “data\_ora > (NOW() - interval 1 day)” per il calcolo di tutti I pacchi in cui la data e l’ora di accettazione è maggiore della data e ora in cui viene effettuato il controllo meno 1 giorno.

30 CREATE VIEW spedizioni\_accettate\_24H (sp\_accettate) AS

```
SELECT count(*)
```

```
FROM fase
```

```
where nome = "accettata" AND data_ora > (NOW() - interval 1 day);
```



- spedizioni\_consegnate\_24H

5 vista che riporta il numero di spedizioni consegnate nell'arco delle ultime 24 ore con le entrate totali derivate dalla somma del costo effettivo (quindi del costo a valle dell'aggiornamento da parte del centro di prossimità) delle spedizioni consegnate nelle ultime 24 ore.

```
CREATE VIEW spedizioni_consegnate_24H (sp_consegnate, entrate_totali) AS
```

```
SELECT count(*), sum(costo_effettivo)
```

```
10 FROM fase JOIN costo_spedizione ON codice_sp = codice_spedizione
```

```
WHERE nome = "consegnato" AND data_ora > (NOW() - interval 1 day);
```

- spazio\_disponibile\_veicolo

15 vista che riporta per ogni veicolo lo spazio disponibile per caricare altri pacchi, lo spazio disponibile viene calcolato prelevando il valore volumetrico del veicolo e sottraendo tutti I pacchi assegnati ad esso.

```
CREATE VIEW spazio_disponibile_veicolo (codice_veicolo, spazio_disponibile) AS
```

```
SELECT                                veicolo.codice_v,                                valore_volumetrico-
```

```
20 COALESCE(sum(lunghezza*larghezza*profondità),0)
```

```
FROM veicolo LEFT join assegnamento on veicolo.codice_v = assegnamento.codice_v
```

```
LEFT join pacco on assegnamento.codice_sp = pacco.codice_sp
```

```
GROUP BY veicolo.codice_v;
```

25 • coordinata\_ritiro\_pacco\_cliente

vista che riporta le coordinate in cui dovrà essere recuperato il pacco del cliente

```
CREATE VIEW coordinata_ritiro_pacco_cliente (cf_cl, latitudine_cl, longitudine_cl) AS
```

```
SELECT cf, latitudine_cgr, longitudine_cgr
```

```
FROM cliente JOIN coordinate_cliente as cc ON cc.cf_cliente = cf
```

30

## Stored Procedures e transazioni

- inserimento\_giacenza\_pacchi\_co

procedura per gestire l'arrivo di un veicolo contenente I pacchi nel centro operativo.

In particolare all'arrivo di un nuovo veicolo vengono scansionati I codici di spedizione, una volta scansionati lo specifico centro operativo tramite il client invoca la procedura e inserisce I pacchi all'interno della relazione "inserimento\_giacenza\_pacco\_co" e vengono quindi eliminate le relazioni di assegnamento tra I pacchi e il veicolo.

Viene quindi aggiornata la fase di spedizione del pacco inserendo che il pacco è arrivato al centro operativo XXX.

Il sistema verifica poi se il centro operativo dovrà inviare il pacco ad un nuovo centro operativo oppure al destinatario finale della spedizione.

Nel primo caso assegna un nuovo veicolo per il trasporto del pacco, nel caso in cui non ci siano veicoli con abbastanza spazio disponibile ad ospitare il pacco il sistema aggiorna la fase di spedizione del pacco mettendo il pacco come "in attesa di smistamento".

Nel secondo caso invece assegna sempre un veicolo per la consegna del pacco e aggiorna lo stato della spedizione mettendo "in consegna". Se invece non è disponibile un veicolo per la consegna aggiorna lo stato in "in attesa di smistamento".

Viene usata una transazione con livello di isolamento repeatable read in modo da garantire che se un veicolo viene riconosciuto come disponibile questo lo rimanga per tutta la transazione, permette inoltre di eseguire in modo atomico gli inserimenti.

È stato inoltre introdotto un ritardo artificiale tramite la sleep per simulare un ritardo tra le varie fasi di spedizione in modo da avere una struttura più realistica.

DELIMITER //

```
CREATE PROCEDURE inserimento_giacenza_pacchi_co (IN var_codice_co INT, IN
var_codice_sp INT)
```

```
BEGIN
```

```
    DECLARE done INT DEFAULT FALSE;
```

```
    DECLARE cod_v INT;
```

```
    DECLARE spaz_disp FLOAT;
```

```
    DECLARE num_ord INT;
```

```
    DECLARE count INT;
```

```
    -- cursore che scorre tutti i furgoni appartenenti ad un centro operativo
```

```
    DECLARE cursore1 CURSOR FOR SELECT v.codice_v FROM possiede AS p JOIN
```

```
veicolo as v ON p.codice_v = v.codice_v
                                WHERE codice_co = var_codice_co and
tipo_veicolo = "furgone";
    -- cursore che scorre tutti i veicoli appartenenti ad un centro operativo
5      DECLARE cursore2 CURSOR FOR SELECT v.codice_v FROM possiede AS p JOIN
veicolo as v ON p.codice_v = v.codice_v
                                WHERE codice_co = var_codice_co;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

10
    DECLARE exit HANDLER FOR SQLEXCEPTION
    BEGIN
        ROLLBACK;
        RESIGNAL;
15    END;

    SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
    START TRANSACTION;

20    -- inserimento del pacco all'interno di giacenza_pacchi_co
    INSERT INTO giacenza_pacchi_co VALUES (var_codice_co, var_codice_sp);

    -- quando un pacco arriva in un nuovo centro operativo viene scansionato il codice di
    spedizione
25    -- e inserito all'interno di "giacenza_pacchi_co"
    -- all'arrivo del nuovo veicolo viene eliminato l'assegnamento tra il pacco consegnato al
    nuovo
    -- centro operativo ed il veicolo che lo trasportava
    DELETE FROM assegnamento WHERE codice_sp = var_codice_sp;
30    -- viene aggiornata la fase della spedizione
    INSERT INTO fase VALUES (CURRENT_TIMESTAMP, var_codice_sp,
    CONCAT("raggiunto centro operativo ", var_codice_co));
    SELECT ordine_spedizione FROM affidata_a WHERE codice_co = var_codice_co AND
    codice_sp = var_codice_sp INTO num_ord;
```

```
SET num_ord = num_ord + 1;
SELECT count(*) FROM affidata_a WHERE codice_sp = var_codice_sp AND
ordine_spedizione = num_ord INTO count;
-- controllo se il centro operativo è l'ultimo nella lista degli affidamenti (non c'è nessun centro
5 operativo dopo
-- di lui). in questo caso il centro operativo è addetto alla consegna del pacco al destinatario
IF ( count = 0) THEN
-- verifica se esiste un veicolo appartenente al centro operativo con abbastanza spazio
disponibile per il pacco
10 open cursore1;
read_loop2: loop
fetch cursore1 INTO cod_v;
(SELECT spazio_disponibile FROM spazio_disponibile_veicolo WHERE
codice_veicolo = cod_v) INTO spaz_disp;
15 IF (spaz_disp > (SELECT lunghezza*larghezza*profondità FROM pacco
WHERE codice_sp = var_codice_sp)) THEN
-- se esiste un veicolo con spazio disponibile inserisce il seguente
assegnamento
INSERT INTO assegnamento VALUES (cod_v, var_codice_sp);
20 DO SLEEP(1);
INSERT INTO fase VALUES (CURRENT_TIMESTAMP,
var_codice_sp, "in consegna");
DO SLEEP(1);
INSERT INTO fase VALUES (CURRENT_TIMESTAMP,
25 var_codice_sp, "consegnato");
DELETE FROM assegnamento WHERE codice_sp = var_codice_sp;
LEAVE read_loop2;
ELSEIF done THEN
-- se non esiste il veicolo aggiorna la fase del pacco in "in attesa di
30 recupero"
DO SLEEP(1);
INSERT INTO fase VALUES (CURRENT_TIMESTAMP,
var_codice_sp, CONCAT("in attesa di smistamento presso ", var_codice_co));
INSERT INTO riassegnazione_veicolo VALUES (var_codice_co,
```

```
var_codice_sp, "smistamento");
        LEAVE read_loop2;
    END IF;
END loop;
5      close cursore1;

-- altrimenti viene assegnato un nuovo veicolo per la spedizione verso un altro centro
ELSE
    open cursore2;
10     read_loop3: loop
        fetch cursore2 INTO cod_v;
        (SELECT spazio_disponibile FROM spazio_disponibile_veicolo WHERE
codice_veicolo = cod_v) INTO spaz_disp;
        IF (spaz_disp > (SELECT lunghezza*larghezza*profondità FROM pacco
15  WHERE codice_sp = var_codice_sp)) THEN
            INSERT INTO assegnamento VALUES (cod_v, var_codice_sp);
            DO SLEEP(1);
            INSERT INTO fase VALUES (CURRENT_TIMESTAMP,
var_codice_sp, "in transito");
20             LEAVE read_loop3;
        ELSEIF done THEN
            -- se non esiste il veicolo aggiorna la fase del pacco in "attesa di
smistamento presso XXX"
            INSERT INTO fase VALUES (CURRENT_TIMESTAMP,
25  var_codice_sp, CONCAT("in attesa di smistamento presso ", var_codice_co));
            INSERT INTO riassegnazione_veicolo VALUES (var_codice_co,
var_codice_sp, "smistamento");
            LEAVE read_loop3;
        END IF;
30     END loop;
    close cursore2;

END IF;
```

```
COMMIT;  
END //  
DELIMITER ;
```

5

- start\_ritiro\_pacco

procedura chiamata all'interno della procedura "inserimento\_nuovo\_pacco", viene chiamata in seguito all'assegnazione dei centri operativi per una determinata spedizione per inizializzare la procedura di spedizione. Viene come prima cosa verificato se in quel momento esiste un veicolo appartenente al centro operativo con abbastanza spazio per ospitare il pacco, se il veicolo esiste viene creata l'assegnazione veicolo-pacco, altrimenti il pacco viene messo "in attesa di recupero" e viene inserito all'interno della relazione "riassegnazione\_veicolo" che conterrà tutti i veicoli che il giorno seguente dovranno essere riprocessati.

10

15 Non viene impostata nessuna transazione in quanto verrà eseguita all'interno della transazione della procedura "crea\_nuovo\_ordine".

```
DELIMITER //
```

```
CREATE PROCEDURE start_ritiro_pacco(IN cod_sp INT, IN cod_co INT)
```

20 

```
BEGIN
```

```
    DECLARE done INT DEFAULT FALSE;
```

```
    DECLARE cod_v INT;
```

```
    DECLARE spaz_disp FLOAT;
```

```
    -- cursore che scorre tutti i furgoni appartenenti ad un centro operativo
```

25 

```
    DECLARE cursore1 CURSOR FOR SELECT v.codice_v FROM possiede AS p JOIN  
    veicolo as v ON p.codice_v = v.codice_v
```

```
                                WHERE codice_co = cod_co and
```

```
    tipo_veicolo = "furgone";
```

```
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
```

30

```
    -- verifica se esiste un veicolo appartenente al centro operativo con abbastanza spazio  
    disponibile per il pacco
```

```
    open cursore1;
```

```
    read_loop2: loop
```

```

        fetch cursore1 INTO cod_v;
        (SELECT spazio_disponibile FROM spazio_disponibile_veicolo WHERE
codice_veicolo = cod_v) INTO spaz_disp;
        IF (spaz_disp > (SELECT lunghezza*larghezza*profondità FROM pacco
5  WHERE codice_sp = cod_sp)) THEN
            -- se esiste un veicolo con spazio disponibile inserisce il seguente
            assegnamento
            INSERT INTO assegnamento VALUES (cod_v, cod_sp);
            LEAVE read_loop2;
10        ELSEIF done THEN
            -- se non esiste il veicolo aggiorna la fase del pacco in "in attesa di
            recupero"
            INSERT INTO fase VALUES (CURRENT_TIMESTAMP, cod_sp,
            "in attesa di recupero");
15        INSERT INTO riassegnazione_veicolo VALUES (cod_sp, cod_co,
            "ritiro");
            LEAVE read_loop2;
            END IF;
        END loop;
20    close cursore1;

END //
DELIMITER ;

```

25

• riassegnazione\_veicolo\_pro

procedura chiamata dall'evento "riassegnazione" per gestire la riassegnazione dei veicoli che il giorno precedente erano stati messi nello stato "in attesa di recupero" o in "attesa di smistamento", se  
30 il pacco era in attesa di recupero viene chiamata la procedura "start\_ritiro\_pacco". Se invece il pacco era in attesa di smistamento, viene avviata una transazione con livello di isolamento read uncommitted, viene eliminato il pacco dalla relazione "giacenza\_pacchi\_co" e poi viene chiamata la procedura "inserimento\_giacenza\_pacchi". La transazione è stata impostata al livello di isolamento più basso in quanto ha il solo scopo di eseguire in modo atomico le due azioni di delete e di call.

```

DELIMITER //
CREATE PROCEDURE riassegnazione_veicolo_pro (IN var_codice_sp INT, IN var_codice_co
INT, IN var_tipo VARCHAR(45))
BEGIN
5      DECLARE exit HANDLER FOR SQLEXCEPTION
      BEGIN
          ROLLBACK;
          RESIGNAL;
      END;
10
      IF (var_tipo = "ritiro") THEN
          CALL start_ritiro_pacco (var_codice_sp, var_codice_co);
      ELSE
          SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
15      START TRANSACTION;
          DELETE FROM giacenza_pacchi_co WHERE codice_sp =
var_codice_sp AND codice_co = var_codice_co;
          CALL inserimento_giacenza_pacchi_co (var_codice_co,
var_codice_sp);
20      COMMIT;
      END IF;
END //

DELIMITER ;
25

```

- inserimento\_nuovo\_pacco

procedura chiamata all'interno della procedura "crea\_nuovo\_ordine", va a verificare quali dovranno essere I centri operativi coinvolti durante l'intero processo di spedizione.

Oltre a determinare I centri operativi il sistema assegnare un "numero di ordine" che stabilisce in che ordine I centri operativi dovranno occuparsi del pacco, questo permette di registrare l'ordine in cui il pacco passa da un centro all'altro e permette ad ogni centro operativi di andare a trovare il prossimo centro destinatario del pacco andando semplicemente a verificare il centro a cui corrisponde un



numero di ordine pari al proprio numero d'ordine + 1;

In particolare il sistema verificherà come prima cosa se è possibile completare la spedizione tramite l'utilizzo di un unico centro di prossimità, questa condizione è verificata tramite "IF ( dist > 40) THEN". In questo caso si è supposto che il limite massimo per cui una spedizione può essere  
 5 completata da un unico centro operativo è che la distanza tra il cliente (indirizzo di recupero) e il destinatario sia inferiore a 40 km. (la distanza viene misurata richiamando la funzione "distanza" che prende in input la latitudine e la longitudine dei 2 punti e restituisce in output la loro distanza).

Se la spedizione non può essere portata a termine da un unico centro il sistema si occuperà di determinare a quale centro di smistamento nazionale spetti il secondo passaggio. Il centro di  
 10 smistamento nazionale viene scelto in modo da minimizzare la distanza tra il primo centro di prossimità ed il centro di smistamento nazionale che si sta per scegliere.

Viene poi effettuato un controllo per stabilire se la spedizione è di tipo nazionale o internazionale.

Se la spedizione è internazionale vengono scelti quindi anche I 2 centri di smistamento internazionale, il primo viene scelto in modo da essere il più vicino al centro di smistamento  
 15 nazionale che si è occupato della fase precedente della spedizione, il secondo viene scelto in modo da essere il più vicino al destinatario.

A prescindere dal tipo di spedizione vengono quindi poi scelti in ordine il centro di smistamento nazionale e il centro di prossimità più vicini al destinatario.

Infine viene chiamata la procedura "start\_ritiro\_pacco".

20 Non è stato impostata nessuna transazione in quanto la procedura sarà eseguita all'interno della transazione della procedura "crea\_nuovo\_ordine".

DELIMITER //

CREATE PROCEDURE inserimento\_nuovo\_pacco(IN var\_codice\_sp INT, IN var\_peso  
 25 SMALLINT, IN var\_lunghezza SMALLINT,

IN var\_larghezza SMALLINT, IN var\_profondità SMALLINT)

BEGIN

DECLARE dist FLOAT;

DECLARE ind INT;

30 DECLARE co1 INT;

DECLARE co2 INT;

DECLARE co3 INT;

DECLARE co4 INT;

DECLARE co5 INT;

```
DECLARE co6 INT;
```

```
INSERT INTO pacco (codice_sp, peso, lunghezza, larghezza, profondità)  
VALUES (var_codice_sp, var_peso, var_lunghezza, var_larghezza, var_profondità);
```

5

```
SET ind = 1;
```

```
-- calcolo centro prossimità più vicino all'indirizzo di recupero
```

```
SELECT codice_co
```

```
FROM spedizione JOIN coordinata_ritiro_pacco_cliente on cf_cliente = cf_cl,
```

10

```
centro_operativo
```

```
WHERE codice_spedizione = var_codice_sp AND
```

```
tipo_co = "centro prossimità" AND
```

```
distanza(latitudine, longitudine,
```

```
latitudine_cl, longitudine_cl) = (SELECT min(distanza(latitudine, longitudine,
```

15

```
latitudine_cl, longitudine_cl))
```

```
FROM spedizione
```

```
JOIN coordinata_ritiro_pacco_cliente on cf_cliente = cf_cl,
```

```
centro_operativo
```

```
WHERE
```

```
codice_spedizione = var_codice_sp
```

```
AND
```

```
tipo_co = "centro prossimità") INTO co1;
```

25

```
-- inserimento centro di prossimità addetto al ritiro del pacco
```

```
INSERT INTO affidata_a VALUES (var_codice_sp, co1, ind);
```

```
SET ind = ind + 1;
```

30

```
-- calcolo distanza cliente-destinazione
```

```
SELECT distanza(latitudine_des, longitudine_des, latitudine_cl, longitudine_cl)
```

```
FROM spedizione JOIN coordinata_ritiro_pacco_cliente ON cf_cliente = cf_cl
```

```
WHERE codice_spedizione = var_codice_sp INTO dist;
```

```
-- se la distanza è minore di 40 sarà lo stesso centro di prossimità che ha ritirato
-- il pacco ad occuparsi della consegna
IF ( dist > 40) THEN
5      -- cerco il centro di smistamento nazionale più vicino al centro di prossimità
      -- che ha ritirato il pacco
      SELECT c2.codice_co
      FROM centro_operativo AS c1, centro_operativo AS c2
      WHERE c1.codice_co = co1 AND c2.codice_co != co1 AND c2.tipo_co = "centro
10  smistamento nazionale"
          AND distanza(c1.latitudine, c1.longitudine,
          c2.latitudine,  c2.longitudine) = (SELECT  min(distanza(c1.latitudine,
c1.longitudine,
          c2.latitudine, c2.longitudine))
                                                    FROM
centro_operativo AS c1, centro_operativo AS c2
                                                    WHERE
c1.codice_co = co1 AND c2.codice_co != co1
20                                                    AND
c2.tipo_co = "centro smistamento nazionale") INTO co2;

      INSERT INTO affidata_a VALUES (var_codice_sp, co2, ind);
      SET ind = ind + 1 ;
25

      -- verifico se la spedizione è del tipo internazionale
      IF ( (SELECT tipo_spedizione FROM spedizione WHERE codice_spedizione =
var_codice_sp) = "internazionale") THEN
30      -- cerco il centro di smistamento internazionale più vicino al centro di
smistamento
          -- nazionale precedentemente scelto
          SELECT c2.codice_co
          FROM centro_operativo AS c1, centro_operativo AS c2
```

```
WHERE c1.codice_co = co2 AND c2.codice_co != co2 AND c2.tipo_co =
"centro smistamento internazionale"
AND distanza(c1.latitudine, c1.longitudine,
c2.latitudine, c2.longitudine) = (SELECT min(distanza(c1.latitudine,
5 c1.longitudine,
c2.latitudine, c2.longitudine))
FROM
centro_operativo AS c1, centro_operativo AS c2
10 WHERE
c1.codice_co = co2 AND c2.codice_co != co2
AND c2.tipo_co = "centro smistamento internazionale") INTO co3;
15 INSERT INTO affidata_a VALUES (var_codice_sp, co3, ind);
SET ind = ind + 1;
20 -- cerco il centro di smistamento internazionale più vicino al destinatario
SELECT codice_co
FROM spedizione, centro_operativo
WHERE spedizione.codice_spedizione = var_codice_sp AND
codice_co != co3 AND
25 tipo_co = "centro smistamento internazionale" AND
distanza(latitudine, longitudine, latitudine_des, longitudine_des) =
(SELECT min(distanza(latitudine, longitudine, latitudine_des, longitudine_des))
FROM spedizione, centro_operativo
WHERE spedizione.codice_spedizione
30 = var_codice_sp AND
codice_co != co3 AND
```

```

                                tipo_co      =      "centro
smistamento internazionale") INTO co4;

5
                                INSERT INTO affidata_a VALUES (var_codice_sp, co4, ind);
                                SET ind = ind + 1;

10
                                END IF;

                                -- cerco il centro di smistamento nazionale più vicino al destinatario
                                SELECT codice_co
                                FROM spedizione, centro_operativo
15
                                WHERE spedizione.codice_spedizione = var_codice_sp AND
                                    codice_co != co2 AND
                                    tipo_co = "centro smistamento nazionale" AND
                                    distanza (latitudine, longitudine, latitudine_des, longitudine_des) = (SELECT
                                min(distanza (latitudine, longitudine, latitudine_des, longitudine_des))

20
                                FROM spedizione, centro_operativo

                                WHERE  spedizione.codice_spedizione =

var_codice_sp AND

25
                                codice_co != co2 AND

                                tipo_co = "centro smistamento
nazionale") INTO co5;

30
                                INSERT INTO affidata_a VALUES (var_codice_sp, co5, ind);
                                SET ind = ind + 1;
```

```
-- cerco il centro di prossimità più vicino al destinatario
SELECT codice_co
FROM spedizione, centro_operativo
WHERE spedizione.codice_spedizione = var_codice_sp AND
5      codice_co != co1 AND
      tipo_co = "centro prossimità" AND
      distanza (latitudine, longitudine, latitudine_des, longitudine_des) = (SELECT
min(distanza (latitudine, longitudine, latitudine_des, longitudine_des))

FROM spedizione, centro_operativo

WHERE  spedizione.codice_spedizione =

var_codice_sp AND

      codice_co != co1 AND

      tipo_co = "centro prossimità")
INTO co6;

20      INSERT INTO affidata_a VALUES (var_codice_sp, co6, ind);

END IF;

-- inserisce il pacco all'interno di giacenza_pacchi_co per "avvertire" il centro di prossimità
25 -- che dovrà occuparsi del ritiro del pacco
CALL start_ritiro_pacco(var_codice_sp, co1);

END//

30 DELIMITER ;
```

- inserimento\_nuova\_spedizione

procedura chiamata dalla procedura “crea\_nuovo\_ordine” per l’inserimento di un nuovo pacco all’interno della relazione “spedizione”.

Non viene impostata nessuna relazione perchè la procedura vive all'interno della transazione della procedura "crea\_nuovo\_ordine".

DELIMITER //

CREATE PROCEDURE inserimento\_nuova\_spedizione(OUT codice\_sp INT,

5        IN var\_cf\_cliente CHAR(16), IN var\_latitudine\_des FLOAT, IN var\_longitudine\_des  
FLOAT)

BEGIN

    DECLARE retval INT;

        INSERT INTO spedizione(cf\_cliente, latitudine\_des, longitudine\_des)

10        VALUES (var\_cf\_cliente, var\_latitudine\_des, var\_longitudine\_des);

        SELECT max(codice\_spedizione) FROM spedizione INTO codice\_sp;

END //

DELIMITER ;

15

#### • crea\_nuovo\_ordine

procedura chiamata dal client in seguito alla creazione di un nuovo ordine da parte di un cliente. Viene come prima cosa inserita una nuova istanza di destinatario in base all'input fornito dal cliente, se esiste già un'istanza di quel destinatario non ne viene creata una nuova.

20

Vengono poi chiamate le procedure "inserimento\_nuova\_spedizione" e "inserimento\_nuovo\_pacco". Viene poi fatto un controllo se le misure del pacco sono compatibili con le "modalità" di spedizione fornite dall'azienda. In particolare si verifica se il volume del pacco è minore del volume del minimo veicolo di cui l'azienda è a disposizione.

25

In questo modo si è certi che qualunque veicolo di qualunque centro operativo sarà in grado di trasportare il pacco.

Il confronto tra il volume del veicolo e il volume del pacco si basa su una semplificazione dovuta alla non conoscenza della "forma" del volume disponibile del veicolo. Infatti il prodotto larghezza\*lunghezza\*profondità fornisce un valore puramente numerico del volume del pacco.

30

Viene poi aggiornata la fase di spedizione del pacco in "accettata" e vengono ritornati Il costo di spedizione e il numero della spedizione appena creata.

Viene impostata una transazione con livello di isolamento serializable, ovvero con il livello di isolamento più alto, a causa della procedura "inserimento\_nuova\_spedizione" per eliminare la possibilità di inserimenti fantasma, in quanto potrebbe verificarsi che vengano create 2 nuove

spedizioni nello stesso momento e quindi all'interno di "inserimento\_nuova\_spedizione" venga fatta la insert della spedizione, ma poi la "select max(codice\_spedizione)" restituisca il valore dell'altra spedizione creata concorrentemente.

La transazione ha anche lo scopo di eseguire in modo atomico l'intero processo di creazione di una nuova spedizione.

DELIMITER //

CREATE PROCEDURE crea\_nuovo\_ordine (IN var\_latitudine\_des FLOAT, IN var\_longitudine\_des FLOAT, IN var\_nome VARCHAR(45)

, IN var\_cognome VARCHAR(45), IN var\_nazione VARCHAR(45), IN var\_cf\_cliente CHAR(16), IN var\_peso SMALLINT,

IN var\_lunghezza SMALLINT, IN var\_larghezza SMALLINT, IN var\_profondità SMALLINT, OUT costo FLOAT, OUT codice INT)

BEGIN

DECLARE somma FLOAT;

DECLARE volume\_massimo FLOAT;

DECLARE exit HANDLER FOR SQLEXCEPTION

BEGIN

ROLLBACK;

RESIGNAL;

END;

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

START TRANSACTION;

insert into destinatario (latitudine, longitudine, nome, cognome, nazione)

values (var\_latitudine\_des, var\_longitudine\_des, var\_nome, var\_cognome, var\_nazione) on duplicate key update nome=nome;

CALL inserimento\_nuova\_spedizione(@codice\_sp, var\_cf\_cliente, var\_latitudine\_des, var\_longitudine\_des);

CALL inserimento\_nuovo\_pacco(@codice\_sp, var\_peso, var\_lunghezza, var\_larghezza, var\_profondità);



```
SET somma = var_larghezza * var_lunghezza * var_profondità;
SELECT min(valore_volumetrico) FROM veicolo INTO volume_massimo;
IF (somma > volume_massimo) THEN
5      SIGNAL SQLSTATE '45001' SET MESSAGE_TEXT = "non è possibile
spedire il pacco, dimensione troppo grande";
      ELSE
          INSERT INTO fase VALUES (CURRENT_TIMESTAMP, @codice_sp,
"accettata");
10      END IF;

      SELECT stima_costo FROM costo_spedizione WHERE codice_spedizione =
@codice_sp INTO costo;
      SET codice = @codice_sp;
15      COMMIT;

END //
20 DELIMITER ;

25 • login
procedura chiamata dal client per gestire la procedura di login e stabilire con che ruolo il client
intende eseguire.
DELIMITER //
CREATE PROCEDURE login (IN var_username VARCHAR(45), IN var_pass VARCHAR(45),
30 OUT var_role INT)
BEGIN
    DECLARE var_user_role ENUM("amministratore", "cliente", "centro", "veicolo");

    SELECT ruolo
```

```
FROM utenti
WHERE username = var_username AND password = md5(var_pass) INTO var_user_role;
```

```

IF var_user_role = "amministratore" THEN
5      SET var_role = 1;
ELSEIF var_user_role = "cliente" THEN
      SET var_role = 2;
ELSEIF var_user_role = "centro" THEN
      SET var_role = 3;
10  ELSEIF var_user_role = "veicolo" THEN
      SET var_role = 4;
ELSE
      SET var_role = 5;
END IF;
15  END //
DELIMITER ;
```

```

20  • crea_utente
procedura per la creazione di un nuovo utente.
DELIMITER //
CREATE PROCEDURE crea_utente (IN username varchar(45), IN pass varchar(45), IN ruolo
varchar(45))
25  BEGIN
      INSERT INTO utenti VALUES (username, md5(pass), ruolo);
END //
DELIMITER ;
```

30

• registra\_nuovo\_cliente

procedura chiamata dal client per la creazione di un nuovo cliente, viene usata una transazione con il

livello di isolamento impostato a read uncommitted, in modo da garantire l'atomicità per l'inserimento delle nuove istanze di "cliente", "carta\_di\_credito" e "coordinate\_cliente".

DELIMITER //

```
CREATE PROCEDURE registra_nuovo_cliente (IN cf CHAR(16), IN nome_cliente
5  VARCHAR(45), IN cognome_cliente VARCHAR(45),
      IN data_di_nascita DATE, IN via_residenza VARCHAR(90), IN
città_residenza VARCHAR(45),
      IN cap_residenza INT UNSIGNED, IN provincia_residenza VARCHAR(45),
IN via_fatturazione VARCHAR(90),
10      IN città_fatturazione VARCHAR(45), IN cap_fatturazione INT UNSIGNED,
      IN provincia_fatturazione VARCHAR(45), IN numero BIGINT UNSIGNED,
      IN nome_intestatario VARCHAR(45), IN cognome_intestatario
VARCHAR(45), IN data_di_scadenza DATE,
      IN codice_cvv SMALLINT UNSIGNED, IN cliente_titolare CHAR(16), IN
15  latitudine FLOAT, IN longitudine FLOAT,
      IN username VARCHAR(45), IN password VARCHAR(45))
BEGIN

    DECLARE EXIT HANDLER FOR SQLEXCEPTION
20  BEGIN
        ROLLBACK;
        RESIGNAL;
    END;

    SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
25  START TRANSACTION;

    CALL crea_utente(username, password, "cliente");

30  INSERT INTO cliente VALUES (cf, nome_cliente, cognome_cliente,
data_di_nascita, via_residenza, città_residenza,
      cap_residenza, provincia_residenza, via_fatturazione, città_fatturazione,
cap_fatturazione,
      provincia_fatturazione, username);
```

```
INSERT INTO carta_di_credito VALUES (numero, nome_intestatario,  
cognome_intestatario, data_di_scadenza, codice_cvv, cliente_titolare);
```

```
5      INSERT INTO coordinate_cliente VALUES (cf, latitudine, longitudine);
```

```
      COMMIT;
```

```
END //
```

```
DELIMITER ;
```

10

• aggiungi\_recapiti

procedura chiamata dal client per l’inserimento di un nuovo recapito da parte del cliente

```
15 DELIMITER //
```

```
CREATE PROCEDURE aggiungi_recapiti (IN cellulare INT UNSIGNED, IN telefono INT  
UNSIGNED, IN email VARCHAR(60), IN cf CHAR(16))
```

```
BEGIN
```

```
      INSERT INTO recapiti VALUES (cellulare, telefono, email, cf);
```

```
20 END//
```

```
DELIMITER ;
```

25 • visualizza\_posizione\_pacco

procedura chiamata dal client per la visualizzazione della posizione del pacco in seguito alla richiesta del cliente.

Viene impostata una transazione a livello serializable per evitare di avere un inserimento fantasma e quindi un incongruenza in “data\_ora = select max(data\_ora)...”

```
30 DELIMITER //
```

```
CREATE PROCEDURE visualizza_posizione_pacco(IN codice_sp INT UNSIGNED, IN cf  
CHAR(16), OUT lat FLOAT, OUT lon FLOAT)
```

```
BEGIN
```

```
SET TRANSACTION READ ONLY;
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SELECT latitudine, longitudine
FROM posizione_pacco JOIN fase ON posizione_pacco.codice_spedizione =
5 fase.codice_sp
JOIN spedizione ON spedizione.codice_spedizione = fase.codice_sp
WHERE nome = "in transito" AND codice_sp = posizione_pacco.codice_spedizione
AND cf_cliente = cf
AND data_ora = (SELECT max(data_ora)
10 FROM fase
WHERE codice_sp = fase.codice_sp) INTO lat, lon;

COMMIT;
END //
DELIMITER ;
15
```

- check\_tipo\_co

```
20 procedura chiamata in automatico da client in fase di login da parte di un centro operativo per
determinare il tipo di centro operativo. Viene usata la transazione con livello read committed in
modo da leggere solo dati coerenti.
DELIMITER //
CREATE PROCEDURE check_tipo_co(IN cod_co INT UNSIGNED, OUT tipo VARCHAR(45))
25 BEGIN
SET TRANSACTION READ ONLY;
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
SELECT tipo_co
FROM centro_operativo
30 WHERE cod_co = codice_co INTO tipo;

COMMIT;
END //
DELIMITER ;
```

- stato\_spedizione

procedura chiamata dal client per verificare lo stato della spedizione da parte del cliente.

- 5 Viene usata una procedura con livello di isolamento read committed in modo da leggere solo dati coerenti.

DELIMITER //

CREATE PROCEDURE stato\_spedizione(IN cod\_sp INT UNSIGNED, IN cf CHAR(16))

BEGIN

10 SET TRANSACTION READ ONLY;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

SELECT nome, data\_ora

FROM fase JOIN spedizione ON codice\_sp = codice\_spedizione

WHERE cf = cf\_cliente AND codice\_sp = cod\_sp;

15 COMMIT;

END //

DELIMITER ;

20

- return\_cf

procedura chiamata in automatico in fase di login da parte di un cliente per andare a salvare il codice fiscale appartenente a quel cliente.

Viene usata un transazione con livello di isolamento read committed in modo da andare a leggere solo dati coerenti.

25

DELIMITER //

CREATE PROCEDURE return\_cf(IN username\_cliente VARCHAR(45), OUT cf\_cliente CHAR(16))

BEGIN

30 SET TRANSACTION READ ONLY;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

SELECT cf

FROM cliente

```
        WHERE username_cliente = username INTO cf_cliente;
    COMMIT;
END //
DELIMITER ;
```

5

- conferma\_addebita

10 procedura chiamata dal centro di prossimità per confermare il costo effettivo di spedizione e procedere all'addebito sulla carta di credito del cliente.

Viene usata una transazione con livello di isolamento read committed in modo da leggere dati coerenti.

```
DELIMITER //
```

```
15 CREATE PROCEDURE conferma_addebita(IN codice_sp INT UNSIGNED, IN codice_co INT
    UNSIGNED, IN costo_effettivo FLOAT UNSIGNED)
```

```
BEGIN
```

```
    DECLARE num_carta BIGINT UNSIGNED;
```

```
20    SET TRANSACTION READ ONLY;
```

```
    SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
```

```
    SELECT numero
```

```
    FROM cliente JOIN spedizione ON cf_cliente = cf
```

```
    JOIN carta_di_credito ON cf = cliente_titolare
```

```
25    WHERE codice_spedizione = codice_sp INTO num_carta;
```

```
    INSERT INTO pagamento VALUES (codice_sp, codice_co, num_carta, costo_effettivo);
```

```
    COMMIT;
```

```
END //
```

```
30 DELIMITER ;
```

- report\_giornaliero

procedura chiamata dall'amministratore per andare a leggere il report delle ultime 24 ore per quanto riguarda le spedizioni consegnate, accettate e i ricavi.

Viene usata una procedura con livello di isolamento read committed in modo da andare a leggere dati coerenti.

```
DELIMITER //
```

```
CREATE PROCEDURE report_giornaliero(OUT num_sp_accettate INT, OUT num_sp_consegnate  
INT, OUT entrate_tot FLOAT)
```

```
BEGIN
```

```
10      SET TRANSACTION READ ONLY;
```

```
      SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
```

```
          SELECT sp_accettate, sp_consegnate, entrate_totali
```

```
          FROM spedizioni_consegnate_24H, spedizioni_accettate_24H;
```

```
      COMMIT;
```

```
15  END //
```

```
DELIMITER ;
```

20

- modifica\_categoria

procedura chiamata dall'amministratore per andare ad aggiornare o aggiungere una categoria all'interno delle categorie dei pacchi.

```
DELIMITER //
```

```
25  CREATE PROCEDURE modifica_categoria(IN nome_cat VARCHAR(45), IN prezzo FLOAT  
UNSIGNED, IN dim_minima FLOAT, IN dim_massima FLOAT)
```

```
BEGIN
```

```
      INSERT INTO categoria (nome, prezzo_base, dimensione_minima, dimensione_massima)
```

```
      VALUES (nome_cat, prezzo, dim_minima, dim_massima)
```

```
30  ON DUPLICATE KEY UPDATE prezzo_base = VALUES(prezzo_base),
```

```
      dimensione_minima = VALUES(dimensione_minima), dimensione_massima =
```

```
VALUES(dimensione_massima);
```

```
END //
```



DELIMITER ;

5

- aggiorna\_posizione

procedura chiamata dal veicolo per inviare la propria posizione

DELIMITER //

```
10 CREATE PROCEDURE aggiorna_posizione(IN lat FLOAT, IN lon FLOAT, IN targa CHAR(7))  
BEGIN
```

```
    UPDATE veicolo
```

```
    SET latitudine = lat, longitudine = lon
```

```
    WHERE veicolo.targa = targa;
```

```
15 END //
```

DELIMITER ;

```
20 • aggiorna_coordinate_recupero_pacco
```

procedura chiamata dal cliente per andare a cambiare le coordinate per il recupero dei pacchi

DELIMITER //

```
CREATE PROCEDURE aggiorna_coordinate_recupero_pacco(IN cf CHAR(16), IN lat FLOAT, IN  
lon FLOAT)
```

```
25 BEGIN
```

```
    UPDATE coordinate_cliente
```

```
    SET latitudine_cgr = lat, longitudine_cgr = lon
```

```
    WHERE cf = cf_cliente;
```

```
END //
```

```
30 DELIMITER ;
```

## Appendice: Implementazione

### Codice SQL per istanziare il database

```
DROP SCHEMA IF EXISTS logisticasrl;
```

```
CREATE SCHEMA logisticasrl;
```

```
5  USE logisticasrl;
```

```
CREATE TABLE utenti
```

```
    (  
10      username VARCHAR(45) PRIMARY KEY,  
      password CHAR(32) NOT NULL,  
      ruolo ENUM("amministratore", "cliente", "centro", "veicolo") NOT NULL  
    );
```

```
15
```

```
CREATE TABLE cliente
```

```
    (  
      cf CHAR(16) PRIMARY KEY,  
      nome VARCHAR(45) NOT NULL,  
20      cognome VARCHAR(45) NOT NULL,  
      data_di_nascita DATE NOT NULL,  
      via_residenza VARCHAR(90) NOT NULL,  
      città_residenza VARCHAR(45) NOT NULL,  
      cap_residenza INT UNSIGNED NOT NULL,  
25      provincia_residenza VARCHAR(45) NOT NULL,  
      via_fatturazione VARCHAR(90) NOT NULL,  
      città_fatturazione VARCHAR(45) NOT NULL,  
      cap_fatturazione INT UNSIGNED NOT NULL,  
      provincia_fatturazione VARCHAR(45) NOT NULL,  
30      username VARCHAR(45) UNIQUE NOT NULL,  
      FOREIGN KEY (username) REFERENCES utenti(username)
```

);

CREATE TABLE carta\_di\_credito

5           (  
            numero BIGINT UNSIGNED PRIMARY KEY,  
            nome VARCHAR(45) NOT NULL,  
            cognome VARCHAR(45) NOT NULL,  
            data\_di\_scadenza DATE NOT NULL,  
10           codice\_cvv SMALLINT UNSIGNED NOT NULL,  
            cliente\_titolare CHAR(16) UNIQUE NOT NULL,  
            FOREIGN KEY (cliente\_titolare) REFERENCES cliente(cf)  
          );

15

CREATE TABLE recapiti

          (  
            cellulare INT UNSIGNED PRIMARY KEY,  
20           telefono INT UNSIGNED NOT NULL UNIQUE,  
            email VARCHAR(60) NOT NULL UNIQUE,  
            cf\_cliente CHAR(16) UNIQUE NOT NULL,  
            FOREIGN KEY (cf\_cliente) REFERENCES cliente(cf)  
          );

25

CREATE TABLE coordinate\_cliente

          (  
            cf\_cliente CHAR(16) PRIMARY KEY,  
30           latitudine\_cgr FLOAT NOT NULL,  
            longitudine\_cgr FLOAT NOT NULL,  
            FOREIGN KEY (cf\_cliente) REFERENCES cliente(cf)  
          );

CREATE TABLE destinatario

```
(  
    latitudine FLOAT,  
5    longitudine FLOAT,  
    nome VARCHAR(45) NOT NULL,  
    cognome VARCHAR(45) NOT NULL,  
    nazione VARCHAR(45) NOT NULL,  
    PRIMARY KEY (latitudine, longitudine, nome, cognome)  
10 );
```

CREATE TABLE spedizione

```
15    (  
        codice_spedizione INT UNSIGNED PRIMARY KEY AUTO_INCREMENT,  
        tipo_spedizione VARCHAR(45) DEFAULT NULL,  
        cf_cliente CHAR(16) NOT NULL,  
        latitudine_des FLOAT NOT NULL,  
20    longitudine_des FLOAT NOT NULL,  
        FOREIGN KEY (latitudine_des, longitudine_des) REFERENCES  
destinatario(latitudine, longitudine),  
        FOREIGN KEY (cf_cliente) REFERENCES cliente(cf)  
    );  
25
```

CREATE TABLE pacco

```
(  
    codice_sp INT UNSIGNED PRIMARY KEY,  
30    peso SMALLINT unsigned NOT NULL,  
    lunghezza SMALLINT unsigned NOT NULL,  
    larghezza SMALLINT unsigned NOT NULL,  
    profondit  SMALLINT unsigned NOT NULL,  
    nome_categoria VARCHAR(45) DEFAULT NULL,
```

```
FOREIGN KEY (codice_sp) REFERENCES spedizione(codice_spedizione)
);
```

```
5 CREATE TABLE categoria
```

```
    (
        nome VARCHAR(45) PRIMARY KEY,
        prezzo_base FLOAT UNSIGNED NOT NULL,
        dimensione_minima FLOAT NOT NULL,
10     dimensione_massima FLOAT NOT NULL
    );
```

```
CREATE TABLE fase
```

```
15     (
        data_ora TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
        codice_sp INT UNSIGNED,
        nome VARCHAR(45) NOT NULL,
        PRIMARY KEY (data_ora, codice_sp, nome),
20     FOREIGN KEY (codice_sp) REFERENCES spedizione(codice_spedizione)
    );
```

```
CREATE TABLE centro_operativo
```

```
25     (
        codice_co INT UNSIGNED PRIMARY KEY AUTO_INCREMENT,
        via VARCHAR(90) NOT NULL,
        città VARCHAR(45) NOT NULL,
        cap INT UNSIGNED NOT NULL,
30     provincia VARCHAR(45) NOT NULL,
        recapito_telefonico INT UNSIGNED NOT NULL UNIQUE,
        latitudine FLOAT NOT NULL,
        longitudine FLOAT NOT NULL,
        nome_responsabile VARCHAR(45) NOT NULL,
```

```
email_segreteria VARCHAR(60) NOT NULL UNIQUE,  
tipo_co VARCHAR(45) NOT NULL
```

```
);
```

5

```
CREATE TABLE affidata_a
```

```
(
```

```
    codice_sp INT UNSIGNED,
```

10

```
    codice_co INT UNSIGNED,
```

```
    ordine_spedizione INT UNSIGNED,
```

```
    PRIMARY KEY (codice_sp, codice_co),
```

```
    FOREIGN KEY (codice_sp) REFERENCES spedizione(codice_spedizione),
```

```
    FOREIGN KEY (codice_co) REFERENCES centro_operativo(codice_co)
```

15

```
);
```

```
CREATE TABLE pagamento
```

```
(
```

20

```
    codice_sp INT UNSIGNED PRIMARY KEY,
```

```
    codice_co INT UNSIGNED,
```

```
    numero_carta BIGINT UNSIGNED,
```

```
    costo_effettivo FLOAT UNSIGNED NOT NULL,
```

```
    FOREIGN KEY (codice_co) REFERENCES centro_operativo(codice_co),
```

25

```
    FOREIGN KEY (numero_carta) REFERENCES carta_di_credito(numero),
```

```
    FOREIGN KEY (codice_sp) REFERENCES spedizione(codice_spedizione)
```

```
);
```

30

```
CREATE TABLE veicolo
```

```
(
```

```
    codice_v INT UNSIGNED PRIMARY KEY AUTO_INCREMENT,
```

```
    targa CHAR(7) NOT NULL UNIQUE,
```

```
    valore_volumetrico FLOAT NOT NULL,  
    tipo_veicolo VARCHAR(45) NOT NULL,  
    latitudine FLOAT NOT NULL,  
    longitudine FLOAT NOT NULL
```

```
5      );
```

```
CREATE TABLE possiede
```

```
10      (  
        codice_co INT UNSIGNED,  
        codice_v INT UNSIGNED,  
        PRIMARY KEY (codice_co, codice_v),  
        FOREIGN KEY (codice_co) REFERENCES centro_operativo(codice_co),  
        FOREIGN KEY (codice_v) REFERENCES veicolo(codice_v)  
15      );
```

```
CREATE TABLE assegnamento
```

```
20      (  
        codice_v INT UNSIGNED,  
        codice_sp INT UNSIGNED,  
        PRIMARY KEY (codice_sp),  
        FOREIGN KEY (codice_v) REFERENCES veicolo(codice_v),  
25      FOREIGN KEY (codice_sp) REFERENCES spedizione(codice_spedizione)  
        );
```

```
CREATE TABLE giacenza_pacchi_co
```

```
30      (  
        codice_co INT UNSIGNED,  
        codice_sp INT UNSIGNED,  
        PRIMARY KEY (codice_co, codice_sp),  
        FOREIGN KEY (codice_co) REFERENCES centro_operativo(codice_co),
```

```
FOREIGN KEY (codice_sp) REFERENCES spedizione(codice_spedizione)
);
```

```
5 CREATE TABLE riassegnazione_veicolo
  (
    codice_sp INT,
    codice_co INT,
    tipo VARCHAR(45),
10 PRIMARY KEY (codice_sp, codice_co)
  );
```

```
15
insert into categoria
values ("piccola", 10, 0, 100),
("media", 40, 100, 500),
("grande", 80, 500, 2000);
```

20

```
insert into veicolo
values (1, "aa111aa", 2000, "furgone", 41, 41),
25 (2, "aa222aa", 2000, "furgone", 89, 23),
(3, "aa333aa", 2000, "furgone", 89, 23),
(4, "aa444aa", 2000, "furgone", 89, 23),
(5, "aa555aa", 2000, "furgone", 89, 23),
(6, "aa666aa", 2000, "furgone", 89, 23),
30 (7, "aa777aa", 10000, "autoarticolato", 89, 23),
(8, "aa888aa", 10000, "autoarticolato", 89, 23),
(9, "aa999aa", 10000, "autoarticolato", 89, 23),
(10, "aa000bb", 10000, "autoarticolato", 89, 23),
(11, "aa111bb", 2000, "furgone", 89, 23),
```



```

(12, "aa222bb", 2000, "furgone", 89, 23),
(13, "aa333bb", 2000, "furgone", 89, 23),
(14, "aa444bb", 2000, "furgone", 89, 23),
(15, "aa555bb", 10000, "autoarticolato", 89, 23),
5 (16, "aa666bb", 10000, "autoarticolato", 89, 23),
(17, "aa777bb", 2000, "furgone", 89, 23),
(18, "aa888bb", 2000, "furgone", 89, 23),
(19, "aa999bb", 10000, "autoarticolato", 89, 23),
(20, "aa000cc", 10000, "autoarticolato", 89, 23),
10 (21, "aa111cc", 10000, "autoarticolato", 89, 23),
(22, "aa222cc", 10000, "autoarticolato", 89, 23);

15 (via, latitudine e longitudine sono state ricavate tramite google maps)
INSERT INTO centro_operativo
VALUES (1, "via di panico 7", "roma", 00186, "roma", 069812846, 41.899992, 12.467946, "russo",
"centroroma@gmail.com", "centro prossimità"),
(2, "via paolo orsi 15", "roma", 00178, "roma", 067323182, 41.821876, 12.584744, "verdi",
20 "centroroma2@gmail.com", "centro prossimità"),
(3, "via camilla ravera 5", "roma", 00135, "roma", 068473192, 41.958950, 12.395798, "rossi",
"centroroma3@gmail.com", "centro prossimità"),
(4, "via edimburgo 83", "roma", 00144, "roma", 06783256, 41.819247, 12.453754, "bianchi",
"centroroma4@gmail.com", "centro smistamento nazionale"),
25 (5, "via pal grande 10", "fiumicino", 00054, "roma", 06532324, 41.771333, 12.264104, "romano",
"centrofiumicino@gmail.com", "centro smistamento internazionale"),
(6, "via dei serragli 32", "firenze", 50125, "firenze", 05432654, 43.767032, 11.245384, "ricci",
"centrofirenze@gmail.com", "centro prossimità"),
(7, "via amati 7", "pistoia", 51100, "pistoia", 05636734, 43.931099, 10.918457, "gatti",
30 "centropistoia@gmail.com", "centro prossimità"),
(8, "via del pino 15", "firenze", 50137, "firenze", 05358796, 43.779575, 11.295022, "testa",
"centrofirenze2@gmail.com", "centro smistamento nazionale"),
(9, "rue de rungis 14", "parigi", 75013, "parigi", 57389354, 48.821856, 2.344542, "fontana",
"centroparigi@gmail.com", "centro prossimità"),

```

```
(10, "passage panel 8", "parigi", 75018, "parigi", 324235112, 48.896086, 2.342369, "ferrari",
"centroparigi2@gmail.com", "centro smistamento nazionale"),
(11, "rue delagarde 41", "montfermeil", 93370, "parigi", 32434234, 48.902070, 2.568547, "costa",
"centromontfermeil@gmail.com", "centro smistamento internazionale");
```

5

```
insert into possiede
```

```
values (1,1),
```

```
10 (1,2),
(2,3),
(2,4),
(3,5),
(3,6),
15 (4,7),
(4,8),
(5,9),
(5,10),
(6,11),
20 (6,12),
(7,13),
(7,14),
(8,15),
(8,16),
25 (9,17),
(9,18),
(10,19),
(10,20),
(11,21),
30 (11,22);
```

```
CALL crea_utente ("1", "uno", "centro");
```

```
CALL crea_utente ("2", "due", "centro");
```

CALL crea\_utente ("3", "tre", "centro");  
CALL crea\_utente ("4", "quattro", "centro");  
CALL crea\_utente ("5", "cinque", "centro");  
CALL crea\_utente ("6", "sei", "centro");  
5 CALL crea\_utente ("7", "sette", "centro");  
CALL crea\_utente ("8", "otto", "centro");  
CALL crea\_utente ("9", "nove", "centro");  
CALL crea\_utente ("10", "dieci", "centro");  
CALL crea\_utente ("11", "undici", "centro");  
10  
CALL crea\_utente ("admin", "ciao", "amministratore");  
  
CALL crea\_utente ("aa111aa", "ciao", "veicolo");  
CALL crea\_utente ("aa222aa", "ciao", "veicolo");  
15 CALL crea\_utente ("aa333aa", "ciao", "veicolo");  
CALL crea\_utente ("aa444aa", "ciao", "veicolo");  
CALL crea\_utente ("aa555aa", "ciao", "veicolo");  
CALL crea\_utente ("aa666aa", "ciao", "veicolo");  
CALL crea\_utente ("aa777aa", "ciao", "veicolo");  
20 CALL crea\_utente ("aa888aa", "ciao", "veicolo");  
CALL crea\_utente ("aa999aa", "ciao", "veicolo");  
CALL crea\_utente ("aa000bb", "ciao", "veicolo");  
CALL crea\_utente ("aa111bb", "ciao", "veicolo");  
CALL crea\_utente ("aa222bb", "ciao", "veicolo");  
25 CALL crea\_utente ("aa333bb", "ciao", "veicolo");  
CALL crea\_utente ("aa444bb", "ciao", "veicolo");  
CALL crea\_utente ("aa555bb", "ciao", "veicolo");  
CALL crea\_utente ("aa666bb", "ciao", "veicolo");  
CALL crea\_utente ("aa777bb", "ciao", "veicolo");  
30 CALL crea\_utente ("aa888bb", "ciao", "veicolo");  
CALL crea\_utente ("aa999bb", "ciao", "veicolo");  
CALL crea\_utente ("aa000cc", "ciao", "veicolo");  
CALL crea\_utente ("aa111cc", "ciao", "veicolo");  
CALL crea\_utente ("aa222cc", "ciao", "veicolo");

## Codice del Front-End

- main

```
#include <stdio.h>
```

```
5 #include <stdlib.h>
```

```
#include <string.h>
```

```
#include <mysql.h>
```

```
#include "defines.h"
```

```
10
```

```
typedef enum{
```

```
    AMMINISTRATORE = 1,
```

```
    CLIENTE,
```

```
15
```

```
    CENTRO,
```

```
    VEICOLO,
```

```
    FAILED_LOGIN
```

```
}role_t;
```

```
20
```

```
struct configuration conf;
```

```
static MYSQL *conn;
```

```
25
```

```
static role_t attempt_login(MYSQL *conn, char *username, char *password){
```

```
    MYSQL_STMT *login_procedure; //preparation statement per la procedura di login
```

```
30
```

```
    MYSQL_BIND param[3]; //parametri da collegare al preparation statement
```

```
    int role = 0;
```

```
if (!setup_prepared_stmt(&login_procedure, "call login(?, ?, ?)", conn)){
    print_stmt_error(login_procedure, "errore inizializzazione login statement\n");
    goto err2;
5      }

//preparazione parametri
memset(param, 0, sizeof(param));

10      param[0].buffer_type = MYSQL_TYPE_VAR_STRING; //IN
      param[0].buffer = username;
      param[0].buffer_length = strlen(username);

      param[1].buffer_type = MYSQL_TYPE_VAR_STRING; //IN
15      param[1].buffer = password;
      param[1].buffer_length = strlen(password);

      param[2].buffer_type = MYSQL_TYPE_LONG; //OUT
      param[2].buffer = &role;
20      param[2].buffer_length = sizeof(role);

      if (mysql_stmt_bind_param(login_procedure, param) != 0) {
          print_stmt_error(login_procedure, "impossibile associare i parametri per il login\n");
          goto err;
25      }

//run procedure
//invia al dbms le informazioni
      if (mysql_stmt_execute(login_procedure) != 0) {
30          print_stmt_error(login_procedure, "impossibile eseguire la procedura di login\n");
          goto err;
      }

//preparo i parametri di output
```

```
memset(param, 0, sizeof(param));
param[0].buffer_type = MYSQL_TYPE_LONG;
param[0].buffer = &role;
param[0].buffer_length = sizeof(role);

5
if (mysql_stmt_bind_result(login_procedure, param)){
    print_stmt_error(login_procedure, "impossibile recuperare il parametro di output\n");
    goto err;
}

10
//recupero il parametro di output
if (mysql_stmt_fetch(login_procedure)){
    print_stmt_error(login_procedure, "impossibile bufferizzare i risultati\n");
    goto err;
}

15
mysql_stmt_close(login_procedure);

return role;

20
err:
    mysql_stmt_close(login_procedure);
err2:
    return FAILED_LOGIN;

25
}

30
```

```
static void registra_nuovo_cliente(){
```

```
    MYSQL_STMT *prepared_stmt;    //preparation statement per la procedura di login
```

MYSQL\_BIND param[22]; //parametri da collegare al preparement statement

```
5      char cf[17], nome_cliente[46], cognome_cliente[46], data_di_nascita[11], via_residenza[91],
citta_residenza[46];
      char      provincia_residenza[46],      via_fatturazione[91],      citta_fatturazione[46],
provincia_fatturazione[46];
      char nome_intestatario[46], cognome_intestatario[46], data_di_scadenza[11];
10     char cap_residenza[6], cap_fatturazione[6], numero[17], codice_cvv[4];
      char longitude[46], latitudine[46], username[46], password[46];
      int cap_residenza_int, cap_fatturazione_int;
      float longitude_float, latitudine_float;
      long long int numero_int;
15     short int codice_cvv_int;
      MYSQL_TIME data_di_nascita_date, data_di_scadenza_date;

      //richiesta informazioni
      printf("\033[2J\033[H");
20     printf("inserisci un username: ");
      getInput(46, username, false);
      printf("inserisci una password: ");
      getInput(46, password, true);
      printf("\ninserisci i tuoi dati: \n");
25     printf("codice fiscale: ");
      getInput(17, cf, false);
      printf("nome: ");
      getInput(46, nome_cliente, false);
      printf("cognome: ");
30     getInput(46, cognome_cliente, false);
      printf("data di nascita (yyyy-mm-dd): ");
      getInput(11, data_di_nascita, false);
      printf("via di residenza: ");
      getInput(91, via_residenza, false);
```

```
printf("città di residenza: ");
getInput(17, citta_residenza, false);
printf("cap della città di residenza: ");
getInput(6, cap_residenza, false);
5 printf("provincia di residenza: ");
getInput(46, provincia_residenza, false);
printf("\ninserisci i dati di fatturazione (lasciando vuoto saranno impostati pari a quelli di
residenza): \n");
printf("via di fatturazione: ");
10 getInput(91, via_fatturazione, false);
printf("città di fatturazione: ");
getInput(46, citta_fatturazione, false);
printf("cap di fatturazione: ");
getInput(6, cap_fatturazione, false);
15 printf("provincia_fatturazione: ");
getInput(46, provincia_fatturazione, false);
printf("\ninserisci i dati della carta di credito:\n");
printf("numero carta di credito: ");
getInput(17, numero, false);
20 printf("nome dell'intestatario della carta: ");
getInput(46, nome_intestatario, false);
printf("cognome dell'intestatario della carta: ");
getInput(46, cognome_intestatario, false);
printf("data di scadenza della carta (yyyy-mm-dd): ");
25 getInput(11, data_di_scadenza, false);
printf("codice_cvv della carta: ");
getInput(4, codice_cvv, false);
printf("\ninserisci i dati per il recupero dei pacchi:\n");
printf("inserisci la longitudine per il recupero delle spedizioni: ");
30 getInput(46, longitudine, false);
printf("inserisci la latitudine per il recupero delle spedizioni: ");
getInput(46, latitudine, false);
```



```

//conversione dei valori
numero_int = atoi(numero);
codice_cvv_int = atoi(codice_cvv);
5   cap_residenza_int = atoi(cap_residenza);
    cap_fatturazione_int = atoi(cap_fatturazione);
    longitudine_float = atof(longitudine);
    latitudine_float = atof(latitudine);

10   const char s[2] = "-";

    data_di_scadenza_date.year = atoi(strtok(data_di_scadenza, s));
    data_di_scadenza_date.month = atoi(strtok(NULL, s));
    data_di_scadenza_date.day = atoi(strtok(NULL, s));

15   data_di_nascita_date.year = atoi(strtok(data_di_nascita, s));
    data_di_nascita_date.month = atoi(strtok(NULL, s));
    data_di_nascita_date.day = atoi(strtok(NULL, s));

20

25   if (!setup_prepared_stmt(&prepared_stmt, "call
registra_nuovo_cliente(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)", conn)){
        print_stmt_error(prepared_stmt, "errore inizializzazione login statement\n");
        exit(EXIT_FAILURE);
    }

30   //preparazione parametri
    memset(param, 0, sizeof(param));

    param[0].buffer_type = MYSQL_TYPE_STRING; //IN

```

```
param[0].buffer = cf;
param[0].buffer_length = strlen(cf);

param[1].buffer_type = MYSQL_TYPE_VAR_STRING; //IN
5 param[1].buffer = nome_cliente;
param[1].buffer_length = strlen(nome_cliente);

param[2].buffer_type = MYSQL_TYPE_VAR_STRING; //IN
10 param[2].buffer = &cognome_cliente;
param[2].buffer_length = strlen(cognome_cliente);

param[3].buffer_type = MYSQL_TYPE_DATE;
param[3].buffer = &data_di_nascita_date;
param[3].buffer_length = sizeof(data_di_nascita_date);
15

param[4].buffer_type = MYSQL_TYPE_VAR_STRING;
param[4].buffer = &via_residenza;
param[4].buffer_length = strlen(via_residenza);

20 param[5].buffer_type = MYSQL_TYPE_VAR_STRING;
param[5].buffer = &citta_residenza;
param[5].buffer_length = strlen(citta_residenza);

param[6].buffer_type = MYSQL_TYPE_LONG;
25 param[6].buffer = &cap_residenza_int;
param[6].buffer_length = sizeof(cap_residenza_int);
param[6].is_unsigned = true;

param[7].buffer_type = MYSQL_TYPE_VAR_STRING;
30 param[7].buffer = &provincia_residenza;
param[7].buffer_length = strlen(provincia_residenza);

param[8].buffer_type = MYSQL_TYPE_VAR_STRING;
param[8].buffer = &via_fatturazione;
```

```
param[8].buffer_length = strlen(via_fatturazione);
```

```
param[9].buffer_type = MYSQL_TYPE_VAR_STRING;
```

```
param[9].buffer = &citta_fatturazione;
```

```
5 param[9].buffer_length = strlen(citta_fatturazione);
```

```
param[10].buffer_type = MYSQL_TYPE_LONG;
```

```
param[10].buffer = &cap_fatturazione_int;
```

```
param[10].buffer_length = sizeof(cap_fatturazione_int);
```

```
10 param[10].is_unsigned = true;
```

```
param[11].buffer_type = MYSQL_TYPE_VAR_STRING;
```

```
param[11].buffer = &provincia_fatturazione;
```

```
param[11].buffer_length = strlen(provincia_fatturazione);
```

```
15
```

```
param[12].buffer_type = MYSQL_TYPE_LONGLONG;
```

```
param[12].buffer = &numero_int;
```

```
param[12].buffer_length = sizeof(numero_int);
```

```
param[12].is_unsigned = true;
```

```
20
```

```
param[13].buffer_type = MYSQL_TYPE_VAR_STRING;
```

```
param[13].buffer = &nome_intestatario;
```

```
param[13].buffer_length = strlen(nome_intestatario);
```

```
25 param[14].buffer_type = MYSQL_TYPE_VAR_STRING;
```

```
param[14].buffer = &cognome_intestatario;
```

```
param[14].buffer_length = strlen(cognome_intestatario);
```

```
param[15].buffer_type = MYSQL_TYPE_DATE;
```

```
param[15].buffer = &data_di_scadenza_date;
```

```
30 param[15].buffer_length = sizeof(data_di_scadenza_date);
```

```
param[16].buffer_type = MYSQL_TYPE_SHORT;
```

```
param[16].buffer = &codice_cvv_int;
```

```
param[16].buffer_length = sizeof(codice_cvv_int);

param[17].buffer_type = MYSQL_TYPE_STRING;
param[17].buffer = &cf;
5 param[17].buffer_length = strlen(cf);

param[18].buffer_type = MYSQL_TYPE_FLOAT;
param[18].buffer = &latitudine_float;
10 param[18].buffer_length = sizeof(latitudine_float);

param[19].buffer_type = MYSQL_TYPE_FLOAT;
param[19].buffer = &longitudine_float;
param[19].buffer_length = sizeof(longitudine_float);

15 param[20].buffer_type = MYSQL_TYPE_VAR_STRING;
param[20].buffer = &username;
param[20].buffer_length = strlen(username);

param[21].buffer_type = MYSQL_TYPE_VAR_STRING;
20 param[21].buffer = &password;
param[21].buffer_length = strlen(password);

if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
25     finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per la creazione dell'ordine\n", true);
}

//run procedure
30 if (mysql_stmt_execute(prepared_stmt) != 0){
    print_stmt_error(prepared_stmt, "errore durante la creazione del nuovo ordine\n");
}

else{
```

```
printf("\nregistrazione cliente avvenuta con successo!\n");

printf("premi invio per continuare: ");

5      }

mysql_stmt_close(prepared_stmt);

}

10

int main(){

15      role_t role;
      char op;
      char options[2] = {'1', '2'};

      if (!parse_config("users/login.json", &conf)){
20          fprintf(stderr, "errore nel caricamento della configurazione del login\n");
          exit(EXIT_FAILURE);
      }

      conn = mysql_init (NULL);
25      if (conn == NULL){
          fprintf(stderr, "mysql_init() failed\n");
          mysql_close(conn);
          exit(EXIT_FAILURE);
      }

30      if (mysql_real_connect(conn, conf.host, conf.db_username, conf.db_password, conf.database,
conf.port, NULL, CLIENT_MULTI_STATEMENTS | CLIENT_MULTI_RESULTS) == NULL){
          fprintf(stderr, "mysql_real_connect() failed\n");
          mysql_close(conn);
      }
  }
```

```
        exit(EXIT_FAILURE);
    }

    ripeti:

5      printf("\033[2J\033[H");
      printf("### che cosa vuoi fare? ###\n\n");
      printf("1) login\n");
      printf("2) registra nuovo cliente\n");

10

      op = multiChoice("seleziona un'opzione", options, 2);

      if (op == '1') {
15          printf("\033[2J\033[H");
          printf("inserisci le tue credenziali:\n");
          printf("username: ");
          getInput(128, conf.username, false);
          printf("password: ");
20          getInput(128, conf.password, true);

          role = attempt_login(conn, conf.username, conf.password);

          switch(role){
25              case AMMINISTRATORE:
                  run_as_administrator (conn);
                  break;

              case CLIENTE:
30                  run_as_client (conn);
                  break;

              case CENTRO:
                  run_as_centro (conn);
```

```
break;
```

```
case VEICOLO:
```

```
run_as_veicolo (conn);
```

```
break;
```

```
case FAILED_LOGIN:
```

```
fprintf(stderr, "credenziali non valide\n");
```

```
exit(EXIT_FAILURE);
```

```
break;
```

```
default:
```

```
fprintf(stderr, "Invalid condition at %s:%d\n", __FILE__, __LINE__);
```

```
abort();
```

```
}
```

```
printf("ciao!\n");
```

```
mysql_close(conn);
```

```
}
```

```
else if (op == '2') {
```

```
registra_nuovo_cliente();
```

```
goto ripeti;
```

```
}
```

```
return 0;
```

```
}
```

```
• cliente
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
#include "defines.h"
```

```
char cf[17];
```

5

```
static void crea_nuovo_ordine(MYSQL *conn){
```

```
    MYSQL_STMT *prepared_stmt;
```

10

```
    MYSQL_BIND param[12];
```

```
    char latitudine[46], longitudine[46], nome[46], cognome[46], nazione[46];
```

```
    char peso[5], lunghezza[5], larghezza[5], profondita[5];
```

```
    float latitudine_float, longitudine_float, costo_stimato = 0;
```

15

```
    short int peso_int, lunghezza_int, larghezza_int, profondita_int;
```

```
    int codice_sp;
```

```
    //richiesta informazioni
```

```
    printf("\nlatitudine del destinatario: ");
```

```
    getInput(46, latitudine, false);
```

20

```
    printf("longitudine del destinatario: ");
```

```
    getInput(46, longitudine, false);
```

```
    printf("nome del destinatario: ");
```

```
    getInput(46, nome, false);
```

```
    printf("cognome del destinatario: ");
```

25

```
    getInput(46, cognome, false);
```

```
    printf("nazione del destinatario: ");
```

```
    getInput(46, nazione, false);
```

```
    printf("peso del pacco: ");
```

```
    getInput(5, peso, false);
```

30

```
    printf("lunghezza del pacco: ");
```

```
    getInput(5, lunghezza, false);
```

```
    printf("larghezza del pacco: ");
```

```
    getInput(5, larghezza, false);
```

```
    printf("profondita del pacco: ");
```



```
    getInput(5, profondita, false);

    //conversione dei valori
5    peso_int = atoi(peso);
    lunghezza_int = atoi(lunghezza);
    larghezza_int = atoi(larghezza);
    profondita_int = atoi(profondita);
    latitudine_float = atof(latitudine);
10    longitudine_float = atof(longitudine);

    if (!setup_prepared_stmt(&prepared_stmt, "call crea_nuovo_ordine(?,?,?,?,?,?,?,?,?,?)",
conn)){
15        finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo statement
crea_nuovo_ordine\n", false);
    }

    //preparazione parametri
20    memset(param, 0, sizeof(param));

    param[0].buffer_type = MYSQL_TYPE_FLOAT;
    param[0].buffer = &latitudine_float;
    param[0].buffer_length = sizeof(latitudine_float);
25

    param[1].buffer_type = MYSQL_TYPE_FLOAT;
    param[1].buffer = &longitudine_float;
    param[1].buffer_length = sizeof(longitudine_float);

    param[2].buffer_type = MYSQL_TYPE_VAR_STRING;
30    param[2].buffer = &nome;
    param[2].buffer_length = strlen(nome);

    param[3].buffer_type = MYSQL_TYPE_VAR_STRING;
```

```
param[3].buffer = &cognome;
param[3].buffer_length = strlen(cognome);

param[4].buffer_type = MYSQL_TYPE_VAR_STRING;
5 param[4].buffer = &nazione;
param[4].buffer_length = strlen(nazione);

param[5].buffer_type = MYSQL_TYPE_STRING;
param[5].buffer = &cf;
10 param[5].buffer_length = strlen(cf);

param[6].buffer_type = MYSQL_TYPE_SHORT;
param[6].buffer = &peso_int;
param[6].buffer_length = sizeof(peso_int);
15

param[7].buffer_type = MYSQL_TYPE_SHORT;
param[7].buffer = &lunghezza_int;
param[7].buffer_length = sizeof(lunghezza_int);

param[8].buffer_type = MYSQL_TYPE_SHORT;
20 param[8].buffer = &larghezza_int;
param[8].buffer_length = sizeof(larghezza_int);

param[9].buffer_type = MYSQL_TYPE_SHORT;
25 param[9].buffer = &profondita_int;
param[9].buffer_length = sizeof(profondita_int);

param[10].buffer_type = MYSQL_TYPE_FLOAT; //OUT
param[10].buffer = &costo_stimato;
30 param[10].buffer_length = sizeof(costo_stimato);

param[11].buffer_type = MYSQL_TYPE_LONG; //OUT
param[11].buffer = &codice_sp;
param[11].buffer_length = sizeof(codice_sp);
```

```
    if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
        finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per la creazione dell'ordine\n", true);
5      }

    //run procedure
    if (mysql_stmt_execute(prepared_stmt) != 0){
        print_stmt_error(prepared_stmt, "errore durante la creazione del nuovo ordine\n");
10    }
    else{
        //preparo i parametri di output
        memset(param, 0, sizeof(param));
        param[0].buffer_type = MYSQL_TYPE_FLOAT;
15        param[0].buffer = &costo_stimato;
        param[0].buffer_length = sizeof(costo_stimato);

        param[1].buffer_type = MYSQL_TYPE_LONG;    //OUT
        param[1].buffer = &codice_sp;
20        param[1].buffer_length = sizeof(codice_sp);

        if (mysql_stmt_bind_result(prepared_stmt, param)){
            print_stmt_error(prepared_stmt, "impossibile recuperare il parametro di
output\n");
25            exit(EXIT_FAILURE);
        }

        //recupero il parametro di output
        if (mysql_stmt_fetch(prepared_stmt)){
30            print_stmt_error(prepared_stmt, "impossibile bufferizzare i risultati\n");
            exit(EXIT_FAILURE);
        }

        printf("\nregistrazione del nuovo ordine avvenuta con successo!\n");
        printf("codice di spedizione: %d\n", codice_sp);
```

```
        printf("prezzo stimato per la spedizione: %.2f €\n", costo_stimato);
    }

    mysql_stmt_close(prepared_stmt);
5   }

void aggiungi_recapiti(MYSQL *conn){
10
    MYSQL_STMT *prepared_stmt;
    MYSQL_BIND param[4];

    char cellulare[11], telefono[11];
15    long int cellulare_int, telefono_int;
    char email[61];

    printf("\ninserisci i tuoi dati: \n");
    printf("numero di cellulare: ");
20    getInput(11, cellulare, false);
    printf("numero di telefono: ");
    getInput(11, telefono, false);
    printf("email: ");
    getInput(61, email, false);
25

    //conversione parametri
    cellulare_int = atoi(cellulare);
    telefono_int = atoi(telefono);

30    if (!setup_prepared_stmt(&prepared_stmt, "call aggiungi_recapiti(?, ?, ?, ?)", conn)){
        print_stmt_error(prepared_stmt, "errore inizializzazione login statement\n");
        exit(EXIT_FAILURE);
    }
}
```

```
//preparazione parametri
memset(param, 0, sizeof(param));

param[0].buffer_type = MYSQL_TYPE_LONG;
5 param[0].buffer = &cellulare_int;
  param[0].buffer_length = sizeof(cellulare_int);
  param[0].is_unsigned = true;

param[1].buffer_type = MYSQL_TYPE_LONG;
10 param[1].buffer = &telefono_int;
  param[1].buffer_length = sizeof(telefono_int);
  param[1].is_unsigned = true;

param[2].buffer_type = MYSQL_TYPE_VAR_STRING;
15 param[2].buffer = &email;
  param[2].buffer_length = strlen(email);

param[3].buffer_type = MYSQL_TYPE_STRING;
20 param[3].buffer = &cf;
  param[3].buffer_length = strlen(cf);

if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
    finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per la creazione dell'ordine\n", true);
25 }

//run procedure
if (mysql_stmt_execute(prepared_stmt) != 0){
    print_stmt_error(prepared_stmt, "errore durante la creazione del nuovo ordine\n");
30 }
else {
    printf("recapiti aggiunti con successo!\n");
}
```

```
mysql_stmt_close(prepared_stmt);

}

5
void visualizza_posizione_pacco (MYSQL *conn){

    MYSQL_STMT *prepared_stmt;
    MYSQL_BIND param[4];

10
    char codice_sp[10];
    long int codice_sp_int;
    float latitudine = 0, longitudine = 0;

15
    printf("\ninserisci il codice di spedizione: ");
    getInput(10, codice_sp, false);

    //conversioni parametri

20
    codice_sp_int = atoi(codice_sp);

    if (!setup_prepared_stmt(&prepared_stmt, "call visualizza_posizione_pacco(?, ?, ?, ?)",
conn)){
25
        print_stmt_error(prepared_stmt, "errore inizializzazione login statement\n");
        exit(EXIT_FAILURE);
    }

    //preparazione parametri

30
    memset(param, 0, sizeof(param));

    param[0].buffer_type = MYSQL_TYPE_LONG;
    param[0].buffer = &codice_sp_int;
    param[0].buffer_length = sizeof(codice_sp_int);
```

```
param[0].is_unsigned = true;

param[1].buffer_type = MYSQL_TYPE_STRING;
param[1].buffer = &cf;
5 param[1].buffer_length = strlen(cf);

param[2].buffer_type = MYSQL_TYPE_FLOAT;
param[2].buffer = &latitudine;
10 param[2].buffer_length = sizeof(latitudine);

param[3].buffer_type = MYSQL_TYPE_FLOAT;
param[3].buffer = &longitudine;
param[3].buffer_length = sizeof(longitudine);

15 if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
    finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per la visualizzazione delle coordinate del pacco\n", true);
    }

20 //run procedure
if (mysql_stmt_execute(prepared_stmt) != 0){
    print_stmt_error(prepared_stmt, "errore durante visualizzazione delle coordinate del
pacco\n");
25 }
else {

    //preparo i parametri di output
    memset(param, 0, sizeof(param));

30 param[0].buffer_type = MYSQL_TYPE_FLOAT;
param[0].buffer = &latitudine;
param[0].buffer_length = sizeof(latitudine);
```

```
param[1].buffer_type = MYSQL_TYPE_FLOAT; //OUT
param[1].buffer = &longitudine;
param[1].buffer_length = sizeof(longitudine);

5      if (mysql_stmt_bind_result(prepared_stmt, param)){
        print_stmt_error(prepared_stmt, "impossibile recuperare il parametro di
output\n");
        exit(EXIT_FAILURE);
      }

10     //recupero il parametro di output
      if (mysql_stmt_fetch(prepared_stmt)){
        print_stmt_error(prepared_stmt, "impossibile bufferizzare i risultati\n");
        exit(EXIT_FAILURE);

15     }

      if (latitudine == 0){
        printf("non puoi visualizzare la posizione per questa spedizione\n");
        goto exit2;

20     }

      printf("il pacco si trova in posizione: \n");
      printf("latitudine: %f\n", latitudine);
      printf("longitudine: %f\n", longitudine);

25     }

exit2:
      mysql_stmt_close(prepared_stmt);

30  }
```

```
void stato_spedizione(MYSQL *conn){
```



```
MySQL_STMT *prepared_stmt;
MySQL_BIND param[2];

5      int status;
      long int codice_sp_int;
      char codice_sp[8];
      char nome[46];
      MYSQL_TIME data_ora;

10

      //richiesta informazioni
      printf("\ncodice di spedizione: ");
      getInput(8, codice_sp, false);

15

      //conversione dei valori
      codice_sp_int = atoi(codice_sp);

20      if (!setup_prepared_stmt(&prepared_stmt, "call stato_spedizione(?, ?)", conn)){
          finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo statement
stato_spedizione\n", false);
      }

25

      //preparazione parametri
      memset(param, 0, sizeof(param));

      param[0].buffer_type = MYSQL_TYPE_LONG;
      param[0].buffer = &codice_sp_int;
      param[0].buffer_length = sizeof(codice_sp_int);
      param[0].is_unsigned = true;

30

      param[1].buffer_type = MYSQL_TYPE_STRING;
```

```
    param[1].buffer = &cf;
    param[1].buffer_length = strlen(cf);

/*
5    param[2].buffer_type = MYSQL_TYPE_VAR_STRING;
    param[2].buffer = &nome;
    param[2].buffer_length = sizeof(nome);

    param[3].buffer_type = MYSQL_TYPE_TIMESTAMP;
10    param[3].buffer = &data_ora;
    param[3].buffer_length = sizeof(data_ora);
*/

    if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
15        finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per la visualizzazione dello stato della spedizione\n", true);
    }

    //run procedure
20    if (mysql_stmt_execute(prepared_stmt) != 0){
        print_stmt_error(prepared_stmt, "errore durante la visualizzazione dello stato della
spedizione\n");
    }

25    else{

        if (mysql_stmt_store_result(prepared_stmt)) {
            fprintf(stderr, " mysql_stmt_execute(), 1 failed\n");
            fprintf(stderr, " %s\n", mysql_stmt_error(prepared_stmt));
30            exit(0);
        }

        if (mysql_stmt_num_rows(prepared_stmt) < 1){
            printf("non puoi visualizzare lo stato di questa spedizione\n");
        }
    }
}
```

```
        goto exit;
    }

5      //preparazione parametri
      memset(param, 0, sizeof(param));

      param[0].buffer_type = MYSQL_TYPE_VAR_STRING;
      param[0].buffer = &nome;
10     param[0].buffer_length = sizeof(nome);

      param[1].buffer_type = MYSQL_TYPE_TIMESTAMP;
      param[1].buffer = &data_ora;
      param[1].buffer_length = sizeof(data_ora);

15     if(mysql_stmt_bind_result(prepared_stmt, param)) {
        finish_with_stmt_error(conn, prepared_stmt, "Unable to bind column parameters\n",
true);
    }

20     /* assemble course general information */
    while (true) {
        status = mysql_stmt_fetch(prepared_stmt);

25         if (status == 1 || status == MYSQL_NO_DATA)
            break;

        printf("-->      %-40s      %d/%d/%d      %d:%d:%d\n",      nome,
data_ora.day,data_ora.month,data_ora.year,data_ora.hour,data_ora.minute,data_ora.second);

30     }

    exit:

    status = mysql_stmt_next_result(prepared_stmt);
    if (status > 0)
```

```
        finish_with_stmt_error(conn, prepared_stmt, "Unexpected condition", true);

    }

5      mysql_stmt_close(prepared_stmt);

    }

10 char *return_cf(MYSQL *conn){

        MYSQL_STMT *prepared_stmt;
        MYSQL_BIND param[2];

15      //richiesta codice fiscale

        if (!setup_prepared_stmt(&prepared_stmt, "call return_cf(?,?)", conn)){
            finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo statement
20 return_cf\n", false);
        }

        //preparazione parametri
        memset(param, 0, sizeof(param));

25      param[0].buffer_type = MYSQL_TYPE_STRING; //IN
        param[0].buffer = &conf.username;
        param[0].buffer_length = strlen(conf.username);

30      param[1].buffer_type = MYSQL_TYPE_STRING; //OUT
        param[1].buffer = &cf;
        param[1].buffer_length = sizeof(cf);
```

```
    if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
        finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per la richiesta del codice fiscale\n", true);
    }
5
    //run procedure
    if (mysql_stmt_execute(prepared_stmt) != 0){
        print_stmt_error(prepared_stmt, "errore durante la richiesta del codice fiscale\n");
    }
10
    else {
        //preparo i parametri di output
        memset(param, 0, sizeof(param));

        param[0].buffer_type = MYSQL_TYPE_STRING; //OUT
15
        param[0].buffer = &cf;
        param[0].buffer_length = sizeof(cf);

        if (mysql_stmt_bind_result(prepared_stmt, param)){
            print_stmt_error(prepared_stmt, "impossibile recuperare il parametro di
20 output\n");

            exit(EXIT_FAILURE);
        }

        //recupero il parametro di output
25
        if (mysql_stmt_fetch(prepared_stmt)){
            print_stmt_error(prepared_stmt, "impossibile bufferizzare i risultati\n");
            exit(EXIT_FAILURE);
        }
    }
30

    mysql_stmt_close(prepared_stmt);

    return cf;
```

```
}
```

```
5 void aggiorna_coordinate_recupero_pacco(MYSQL *conn){

    MYSQL_STMT *prepared_stmt;
    MYSQL_BIND param[3];

10 char latitudine[10], longitudine[10];
    float latitudine_float, longitudine_float;

    //richiesta informazioni
15 printf("\ninserisci le nuove coordinate per il recupero dei pacchi:\n");
    printf("latitudine: ");
    getInput(10, latitudine, false);
    printf("longitudine: ");
    getInput(10, longitudine, false);

20

    //conversione dei valori
    latitudine_float = atof(latitudine);
    longitudine_float = atof(longitudine);

25

    //richiesta codice fiscale

    if (!setup_prepared_stmt(&prepared_stmt, "call aggiorna_coordinate_recupero_pacco(?,?,?)",
conn)){
30         finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo statement
aggiorna_coordinate_recupero_pacco\n", false);
    }

    //preparazione parametri
```

```
memset(param, 0, sizeof(param));

param[0].buffer_type = MYSQL_TYPE_STRING; //OUT
param[0].buffer = &cf;
5 param[0].buffer_length = strlen(cf);

param[1].buffer_type = MYSQL_TYPE_FLOAT; //OUT
param[1].buffer = &latitudine_float;
10 param[1].buffer_length = sizeof(latitudine_float);

param[2].buffer_type = MYSQL_TYPE_FLOAT; //OUT
param[2].buffer = &longitudine_float;
15 param[2].buffer_length = sizeof(longitudine_float);

if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
    finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per l'aggiornamento delle coordinate\n", true);
    }
20

//run procedure
if (mysql_stmt_execute(prepared_stmt) != 0){
    print_stmt_error(prepared_stmt, "errore durante l'aggiornamento delle coordinate\n");
    }
25 else {
    printf("coordinate aggiornate con successo");
    }

mysql_stmt_close(prepared_stmt);
30
}
```

```
5 void run_as_client(MYSQL *conn){

    char op;
    char options[6] = {'1', '2', '3', '4', '5', '6'};

10 printf("cambio del ruolo in cliente\n");

    if(!parse_config("users/cliente.json", &conf)){
        fprintf(stderr, "impossibile caricare il ruolo di cliente\n");
        exit(EXIT_FAILURE);
15    }

    //funzione di libreria per cambiare il ruolo
    if(mysql_change_user(conn, conf.db_username, conf.db_password, conf.database)){
        fprintf(stderr, "mysql_change_user() failed\n");
20        exit(EXIT_FAILURE);
    }

    strcpy(cf, return_cf(conn));

25 while(true){
        printf("\033[2J\033[H");
        printf("### che cosa vuoi fare? ###\n\n");
        printf("1) crea un nuovo ordine\n");
        printf("2) aggiungi recapiti\n");
        printf("3) visualizza coordinate correnti del pacco\n");
30        printf("4) visualizza stato spedizione\n");
        printf("5) aggiorna le coordinate per il recupero dei pacchi\n");
        printf("6) esci\n");
```



```
op = multiChoice("seleziona un'opzione", options, 6);
```

```
switch(op){
```

```
    case '1':
```

```
        crea_nuovo_ordine(conn);
```

```
        break;
```

```
    case '2':
```

```
        aggiungi_recapiti(conn);
```

```
        break;
```

```
    case '3':
```

```
        visualizza_posizione_pacco(conn);
```

```
        break;
```

```
    case '4':
```

```
        stato_spedizione(conn);
```

```
        break;
```

```
    case '5':
```

```
        aggiorna_coordinate_recupero_pacco(conn);
```

```
        break;
```

```
    case '6':
```

```
        return;
```

```
    default:
```

```
        fprintf(stderr, "Invalid condition at %s:%d\n", __FILE__, __LINE__);
```

```
        abort();
```

```
}
```

```
getchar();
```

```
    }
}
```

5

```
• centro
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

10

```
#include "defines.h"
```

15

```
static void aggiungi_pacco(MYSQL *conn){
```

```
    MYSQL_STMT *prepared_stmt;
    MYSQL_BIND param[3];
```

20

```
    char codice_sp[8];
    int codice_sp_int;
    int num_centro;
```

25

```
//richiesta informazioni
printf("\ncodice del pacco: ");
getInput(8, codice_sp, false);
```

30

```
//conversione dei valori
codice_sp_int = atoi(codice_sp);
num_centro = atoi(conf.username);
```

```
    if (!setup_prepared_stmt(&prepared_stmt, "call inserimento_giacenza_pacchi_co(?,?)",
conn)){
```

```
        finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo statement
inserimento_giacenza_pacchi_co\n", false);
    }

5    //preparazione parametri
    memset(param, 0, sizeof(param));

    param[0].buffer_type = MYSQL_TYPE_LONG;
    param[0].buffer = &num_centro;
10    param[0].buffer_length = sizeof(num_centro);

    param[1].buffer_type = MYSQL_TYPE_LONG;
    param[1].buffer = &codice_sp_int;
    param[1].buffer_length = sizeof(codice_sp_int);

15    if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
        finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per l'inserimento del pacco in giacenza\n", true);
    }

20    //run procedure
    if (mysql_stmt_execute(prepared_stmt) != 0){
        print_stmt_error(prepared_stmt, "errore durante l'inserimento del pacco in giacenza\
n");
25    }
    else{
        printf("pacco con codice: %d aggiunto in giacenza\n", codice_sp_int);
    }

30    mysql_stmt_close(prepared_stmt);

}
```

```
void conferma_addebita(MYSQL *conn){

    MYSQL_STMT *prepared_stmt;

5    MYSQL_BIND param[3];

    char codice_sp[8];
    int codice_sp_int;
    int num_centro;
10    char costo_effettivo[8];
    float costo_effettivo_float;

    //richiesta informazioni
    printf("\ncodice del pacco: ");
15    getInput(8, codice_sp, false);
    printf("inserisci il costo di spedizione effettivo: ");
    getInput(8, costo_effettivo, false);

    //conversione dei valori
20    codice_sp_int = atoi(codice_sp);
    num_centro = atoi(conf.username);
    costo_effettivo_float = atof(costo_effettivo);

    if (!setup_prepared_stmt(&prepared_stmt, "call conferma_addebita(?,?,?)", conn)){
25        finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo statement
        conferma_addebita\n", false);
    }

    //preparazione parametri
30    memset(param, 0, sizeof(param));

    param[0].buffer_type = MYSQL_TYPE_LONG;
    param[0].buffer = &codice_sp_int;
    param[0].buffer_length = sizeof(codice_sp_int);
```

```
param[0].is_unsigned = true;

param[1].buffer_type = MYSQL_TYPE_LONG;
param[1].buffer = &num_centro;
5 param[1].buffer_length = sizeof(num_centro);
param[1].is_unsigned = true;

param[2].buffer_type = MYSQL_TYPE_FLOAT;
param[2].buffer = &costo_effettivo_float;
10 param[2].buffer_length = sizeof(costo_effettivo_float);
param[2].is_unsigned = true;

if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
15     finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per l'inserimento del costo effettivo\n", true);
}

//run procedure
20 if (mysql_stmt_execute(prepared_stmt) != 0){
    print_stmt_error(prepared_stmt, "errore durante l'inserimento del costo effettivo\n");
}
else{
    printf("conferma del costo effettivo avvenuta con successo\n");
25 }

mysql_stmt_close(prepared_stmt);
}

30

void run_as_centro(MYSQL *conn){
```

```
MySQL_STMT *prepared_stmt;
MySQL_BIND param[2];

5      char op;
      char tipo[46];
      long int codice_co;

      printf("cambio del ruolo in centro operativo\n");

10     if(!parse_config("users/centro.json", &conf)){
        fprintf(stderr, "impossibile caricare il ruolo di cliente\n");
        exit(EXIT_FAILURE);
    }

15     //funzione di libreria per cambiare il ruolo
    if(mysql_change_user(conn, conf.db_username, conf.db_password, conf.database)){
        fprintf(stderr, "mysql_change_user() failed\n");
        exit(EXIT_FAILURE);

20     }

25     //verifica il tipo di centro

    if (!setup_prepared_stmt(&prepared_stmt, "call check_tipo_co(?,?)", conn)){
        finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo statement
check_tipo_co\n", false);

30     }

    //preparazione parametri
    codice_co = atoi(conf.username);
    memset(param, 0, sizeof(param));
```

```
param[0].buffer_type = MYSQL_TYPE_LONG; //IN
param[0].buffer = &codice_co;
param[0].buffer_length = sizeof(codice_co);
5 param[0].is_unsigned = true;

param[1].buffer_type = MYSQL_TYPE_STRING; //OUT
param[1].buffer = &tipo;
param[1].buffer_length = strlen(tipo);
10

if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
    finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per la visualizzazione delle coordinate del pacco\n", true);
15 }

//run procedure
if (mysql_stmt_execute(prepared_stmt) != 0){
    print_stmt_error(prepared_stmt, "errore durante visualizzazione delle coordinate del
20 pacco\n");
}
else {
    //preparo i parametri di output
    memset(param, 0, sizeof(param));
25

    param[0].buffer_type = MYSQL_TYPE_STRING; //OUT
    param[0].buffer = &tipo;
    param[0].buffer_length = sizeof(tipo);

    if (mysql_stmt_bind_result(prepared_stmt, param)){
30         print_stmt_error(prepared_stmt, "impossibile recuperare il parametro di
output\n");
        exit(EXIT_FAILURE);
    }
}
```

```

//recupero il parametro di output
if (mysql_stmt_fetch(prepared_stmt)){
    print_stmt_error(prepared_stmt, "impossibile bufferizzare i risultati\n");
5      exit(EXIT_FAILURE);
    }
}

mysql_stmt_close(prepared_stmt);
10

if (strcmp(tipo, "centro prossimità") == 0){
    char options[3] = {'1', '2', '3'};

15      while(true){
        printf("\033[2J\033[H");
        printf("### che cosa vuoi fare? ###\n\n");
        printf("1) aggiungi pacco in giacenza\n");
        printf("2) conferma costo e procedi all'addebito\n");
20      printf("3) esci\n");

        op = multiChoice("seleziona un'opzione", options, 3);

        switch(op){
25          case '1':
            aggiungi_pacco(conn);
            break;

            case '2':
30          conferma_addebita(conn);
            break;

            case '3':
                return;
        }
    }
}
```



```
        default:
            fprintf(stderr, "Invalid condition at %s:%d\n", __FILE__, __LINE__);
            abort();

5
    }

    getchar();

10
}

else {
    char options[2] = {'1', '2'};

15

    while(true){
        printf("\033[2J\033[H");
        printf("### che cosa vuoi fare? ###\n\n");
        printf("1) aggiungi pacco in giacenza\n");
        printf("2) esci\n");

20

        op = multiChoice("seleziona un'opzione", options, 2);

        switch(op){

25
            case '1':
                aggiungi_pacco(conn);
                break;

            case '2':

30
                return;

        default:
            fprintf(stderr, "Invalid condition at %s:%d\n", __FILE__, __LINE__);
            abort();
    }
```

```
        }

        getchar();

5      }

    }

10 }

• amministratore

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

20 #include "defines.h"

void report_giornaliero(MYSQL *conn){

25     MYSQL_STMT *prepared_stmt;
    MYSQL_BIND param[3];

    int num_sp_accettate = 0, num_sp_consegnate = 0;

30    float entrate_tot = 0;

    if (!setup_prepared_stmt(&prepared_stmt, "call report_giornaliero(?, ?, ?)", conn)){
        finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo statement
```

```
report_giornaliero\n", false);
    }

    //preparazione parametri
5    memset(param, 0, sizeof(param));

    param[0].buffer_type = MYSQL_TYPE_LONG; //OUT
    param[0].buffer = &num_sp_accettate;
    param[0].buffer_length = sizeof(num_sp_accettate);
10
    param[1].buffer_type = MYSQL_TYPE_LONG; //OUT
    param[1].buffer = &num_sp_consegnate;
    param[1].buffer_length = sizeof(num_sp_consegnate);

    param[2].buffer_type = MYSQL_TYPE_FLOAT; //OUT
    param[2].buffer = &entrata_tot;
    param[2].buffer_length = sizeof(entrata_tot);

15
    if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
        finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per il report giornaliero\n", true);
    }

25
    //run procedure
    if (mysql_stmt_execute(prepared_stmt) != 0){
        print_stmt_error(prepared_stmt, "errore durante la richiesta del report giornaliero\n");
    }
    else {
30
        //preparo i parametri di output
        memset(param, 0, sizeof(param));

        param[0].buffer_type = MYSQL_TYPE_LONG; //OUT
        param[0].buffer = &num_sp_accettate;
```

```
param[0].buffer_length = sizeof(num_sp_accettate);

param[1].buffer_type = MYSQL_TYPE_LONG; //OUT
param[1].buffer = &num_sp_consegnate;
5 param[1].buffer_length = sizeof(num_sp_consegnate);

param[2].buffer_type = MYSQL_TYPE_FLOAT; //OUT
param[2].buffer = &entrata_tot;
10 param[2].buffer_length = sizeof(entrata_tot);

if (mysql_stmt_bind_result(prepared_stmt, param)){
    print_stmt_error(prepared_stmt, "impossibile recuperare il parametro di
output\n");
    exit(EXIT_FAILURE);
15 }

//recupero il parametro di output
if (mysql_stmt_fetch(prepared_stmt)){
    print_stmt_error(prepared_stmt, "impossibile bufferizzare i risultati\n");
20 exit(EXIT_FAILURE);
}

printf("\nreport delle ultime 24 ore:\n");
printf("numero di spedizioni accettate: %d\n", num_sp_accettate);
25 printf("numero di spedizioni consegnate: %d\n", num_sp_consegnate);
printf("entrate totali: %.2f €", entrata_tot);
}

mysql_stmt_close(prepared_stmt);
30
}
```

```
void modifica_categoria(MYSQL *conn){

    MYSQL_STMT *prepared_stmt;
    MYSQL_BIND param[4];

5    char nome[46];
    char dim_min[8], dim_max[8];
    float dim_min_float, dim_max_float = 0;
    char prezzo_base[8];
10    float prezzo_base_float;

    printf("\ninserisci i parametri: \n");
    printf("nome della categoria: ");
    getInput(46, nome, false);
15    printf("prezzo base per la spedizione: ");
    getInput(8, prezzo_base, false);
    printf("dimensione minima del pacco: ");
    getInput(8, dim_min, false);
    printf("dimensione massima del pacco: ");
20    getInput(8, dim_max, false);

    //conversione parametri
    prezzo_base_float = atof(prezzo_base);
    dim_min_float = atof(dim_min);
25    dim_max_float = atof(dim_max);

    if (!setup_prepared_stmt(&prepared_stmt, "call modifica_categoria(?, ?, ?, ?)", conn)){
        finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo statement
30 modifica_categoria\n", false);
    }

    //preparazione parametri
    memset(param, 0, sizeof(param));
```

```
param[0].buffer_type = MYSQL_TYPE_VAR_STRING; //IN
param[0].buffer = &nome;
param[0].buffer_length = strlen(nome);

5
param[1].buffer_type = MYSQL_TYPE_FLOAT; //IN
param[1].buffer = &prezzo_base_float;
param[1].buffer_length = sizeof(prezzo_base_float);

10
param[2].buffer_type = MYSQL_TYPE_FLOAT; //IN
param[2].buffer = &dim_min_float;
param[2].buffer_length = sizeof(dim_min_float);

param[3].buffer_type = MYSQL_TYPE_FLOAT; //IN
15
param[3].buffer = &dim_max_float;
param[3].buffer_length = sizeof(dim_max_float);

if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
20
    finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei parametri
per la modifica della categoria\n", true);
}

//run procedure
25
if (mysql_stmt_execute(prepared_stmt) != 0){
    print_stmt_error(prepared_stmt, "errore durante la richiesta della modifica della
categoria\n");
}
else {
30
    printf("categoria modificata con successo!");
}

mysql_stmt_close(prepared_stmt);
```

```
}
```

5

```
• veicolo
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

10 

```
#include <mysql.h>
```

```
#include <unistd.h>
```

```
#include "defines.h"
```

15

```
void run_as_veicolo(MYSQL *conn){
```

```
    MYSQL_STMT *prepared_stmt;
```

20 

```
    MYSQL_BIND param[3];
```

```
    char targa[8];
```

```
    float latitudine = 45, longitudine = 13;
```

25 

```
    printf("cambio del ruolo in veicolo\n");
```

```
    if(!parse_config("users/veicolo.json", &conf)){
```

```
        fprintf(stderr, "impossibile caricare il ruolo di veicolo\n");
```

```
        exit(EXIT_FAILURE);
```

30 

```
    }
```

```
    //funzione di libreria per cambiare il ruolo
```

```
    if(mysql_change_user(conn, conf.db_username, conf.db_password, conf.database)){
```

```
        fprintf(stderr, "mysql_change_user() failed\n");
```

```
        exit(EXIT_FAILURE);
    }

    strcpy(targa, conf.username);

5

    printf("\033[2J\033[H");

10    while(true){

        if (!setup_prepared_stmt(&prepared_stmt, "call aggiorna_posizione(?, ?, ?)", conn)){
            finish_with_stmt_error(conn, prepared_stmt, "impossibile inizializzare lo
statement aggiorna_posizione\n", false);
15        }

        //preparazione parametri
        memset(param, 0, sizeof(param));

20        param[0].buffer_type = MYSQL_TYPE_FLOAT; //IN
        param[0].buffer = &latitudine;
        param[0].buffer_length = sizeof(latitudine);

        param[1].buffer_type = MYSQL_TYPE_FLOAT; //IN
25        param[1].buffer = &longitudine;
        param[1].buffer_length = sizeof(longitudine);

        param[2].buffer_type = MYSQL_TYPE_STRING; //IN
        param[2].buffer = &targa;
30        param[2].buffer_length = strlen(targa);

        if (mysql_stmt_bind_param(prepared_stmt, param) != 0){
            finish_with_stmt_error(conn, prepared_stmt, "impossibile fare il bind dei
```



```
parametri per l'aggiornamento della posizione\n", true);
```

```
}
```

```
//run procedure
```

5

```
if (mysql_stmt_execute(prepared_stmt) != 0){
```

```
    print_stmt_error(prepared_stmt, "errore durante la richiesta per
```

```
l'aggiornamento della posizione\n");
```

```
}
```

```
else {
```

10

```
    printf("aggiornamento posizione del veicolo... \n");
```

```
    printf("latitudine: %f\n", latitudine);
```

```
    printf("longitudine: %f\n\n", longitudine);
```

```
}
```

15

```
mysql_stmt_close(prepared_stmt);
```

```
sleep(5);
```

```
}
```

```
}
```

20

25

```
• inout
```

```
#include <unistd.h>
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

30

```
#include <ctype.h>
```

```
#include <termios.h>
```

```
#include <sys/ioctl.h>
```

```
#include <pthread.h>
```

```
#include <signal.h>
```

```
#include <stdbool.h>

#include "defines.h"

5 // Per la gestione dei segnali
static volatile sig_atomic_t signo;
typedef struct sigaction sigaction_t;
static void handler(int s);

10 char *getInput(unsigned int lung, char *stringa, bool hide)
{
    char c;
    unsigned int i;

15 // Dichiarare le variabili necessarie ad un possibile mascheramento dell'input
sigaction_t sa, savealrm, saveint, savehup, savequit, saveterm;
sigaction_t savetstp, savettin, savettou;
struct termios term, oterm;

20 if(hide) {
    // Svuota il buffer
    (void) fflush(stdout);

    // Cattura i segnali che altrimenti potrebbero far terminare il programma, lasciando
25 l'utente senza output sulla shell
sigemptyset(&sa.sa_mask);
sa.sa_flags = SA_INTERRUPT; // Per non resettare le system call
sa.sa_handler = handler;
(void) sigaction(SIGALRM, &sa, &savealrm);
30 (void) sigaction(SIGINT, &sa, &saveint);
(void) sigaction(SIGHUP, &sa, &savehup);
(void) sigaction(SIGQUIT, &sa, &savequit);
(void) sigaction(SIGTERM, &sa, &saveterm);
(void) sigaction(SIGTSTP, &sa, &savetstp);
```

```
(void) sigaction(SIGTTIN, &sa, &savettin);
(void) sigaction(SIGTTOU, &sa, &savettou);

// Disattiva l'output su schermo
5   if (tcgetattr(fileno(stdin), &oterm) == 0) {
        (void) memcpy(&term, &oterm, sizeof(struct termios));
        term.c_lflag &= ~(ECHO|ECHONL);
        (void) tcsetattr(fileno(stdin), TCSAFLUSH, &term);
    } else {
10      (void) memset(&term, 0, sizeof(struct termios));
        (void) memset(&oterm, 0, sizeof(struct termios));
    }
}

15 // Acquisisce da tastiera al più lung - 1 caratteri
for(i = 0; i < lung; i++) {
    (void) fread(&c, sizeof(char), 1, stdin);
    if(c == '\n') {
        stringa[i] = '\0';
20      break;
    } else
        stringa[i] = c;

    // Gestisce gli asterischi
25   if(hide) {
        if(c == '\b') // Backspace
            (void) write(fileno(stdout), &c, sizeof(char));
        else
            (void) write(fileno(stdout), "*", sizeof(char));
30   }
}

// Controlla che il terminatore di stringa sia stato inserito
if(i == lung - 1)
```

```
        stringa[i] = '\0';

        // Se sono stati digitati più caratteri, svuota il buffer della tastiera
        if(strlen(stringa) >= lung) {
5            // Svuota il buffer della tastiera
            do {
                c = getchar();
            } while (c != '\n');
        }
10
        if(hide) {
            //L'a capo dopo l'input
            (void) write(fileno(stdout), "\n", 1);

15            // Ripristina le impostazioni precedenti dello schermo
            (void) tcsetattr(fileno(stdin), TCSAFLUSH, &oterm);

            // Ripristina la gestione dei segnali
            (void) sigaction(SIGALRM, &savealrm, NULL);
20            (void) sigaction(SIGINT, &saveint, NULL);
            (void) sigaction(SIGHUP, &savehup, NULL);
            (void) sigaction(SIGQUIT, &savequit, NULL);
            (void) sigaction(SIGTERM, &saveterm, NULL);
            (void) sigaction(SIGTSTP, &savetstp, NULL);
25            (void) sigaction(SIGTTIN, &savettin, NULL);
            (void) sigaction(SIGTTOU, &savettou, NULL);

            // Se era stato ricevuto un segnale viene rilanciato al processo stesso
            if(signo)
30                (void) raise(signo);
        }

        return stringa;
    }
}
```

```
// Per la gestione dei segnali
static void handler(int s) {
    signo = s;
5   }

bool yesOrNo(char *domanda, char yes, char no, bool predef, bool insensitive)
{
10
    // I caratteri 'yes' e 'no' devono essere minuscoli
    yes = tolower(yes);
    no = tolower(no);

15    // Decide quale delle due lettere mostrare come predefinite
    char s, n;
    if(predef) {
        s = toupper(yes);
        n = no;
20    } else {
        s = yes;
        n = toupper(no);
    }

25    // Richiesta della risposta
    while(true) {
        // Mostra la domanda
        printf("%s [%c/%c]: ", domanda, s, n);

30        char c;
        getInput(1, &c, false);

        // Controlla quale risposta è stata data
        if(c == '\0') { // getInput() non può restituire '\n'!
```

```
        return predef;
    } else if(c == yes) {
        return true;
    } else if(c == no) {
5         return false;
    } else if(c == toupper(yes)) {
        if(predef || insensitive) return true;
    } else if(c == toupper(yes)) {
        if(!predef || insensitive) return false;
10    }
    }
}
```

```
char multiChoice(char *domanda, char choices[], int num)
15 {

    // Genera la stringa delle possibilità
    char *possib = malloc(2 * num * sizeof(char));
    int i, j = 0;
20    for(i = 0; i < num; i++) {
        possib[j++] = choices[i];
        possib[j++] = '/';
    }
    possib[j-1] = '\0'; // Per eliminare l'ultima '/'
25

    // Chiede la risposta
    while(true) {
        // Mostra la domanda
        printf("%s [%s]: ", domanda, possib);
30

        char c;
        getInput(1, &c, false);

        // Controlla se è un carattere valido
```

```

        for(i = 0; i < num; i++) {
            if(c == choices[i])
                return c;
        }
5      }
    }

```

10

- utils

```

#include <stdio.h>
15 #include <stdlib.h>
#include <string.h>

#include "defines.h"

20 void print_stmt_error (MYSQL_STMT *stmt, char *message)
{
    fprintf (stderr, "%s\n", message);
    if (stmt != NULL) {
        fprintf (stderr, "Error %u (%s): %s\n",
25         mysql_stmt_errno (stmt),
        mysql_stmt_sqlstate(stmt),
        mysql_stmt_error (stmt));
    }
}

30

void print_error(MYSQL *conn, char *message)
{
    fprintf (stderr, "%s\n", message);

```

```
    if (conn != NULL) {
        #if MYSQL_VERSION_ID >= 40101
        fprintf (stderr, "Error %u (%s): %s\n",
mysql_errno (conn), mysql_sqlstate(conn), mysql_error (conn));
5         #else
        fprintf (stderr, "Error %u: %s\n",
mysql_errno (conn), mysql_error (conn));
        #endif
    }
10 }

bool setup_prepared_stmt(MYSQL_STMT **stmt, char *statement, MYSQL *conn)
{
    bool update_length = true;
15
    *stmt = mysql_stmt_init(conn);
    if (*stmt == NULL)
    {
        print_error(conn, "Could not initialize statement handler");
20         return false;
    }

    if (mysql_stmt_prepare (*stmt, statement, strlen(statement)) != 0) {
        print_stmt_error(*stmt, "Could not prepare statement");
25         return false;
    }

    mysql_stmt_attr_set(*stmt, STMT_ATTR_UPDATE_MAX_LENGTH, &update_length);

30     return true;
}

void finish_with_error(MYSQL *conn, char *message)
{
```



```
        print_error(conn, message);
        mysql_close(conn);
        exit(EXIT_FAILURE);
    }
5
void finish_with_stmt_error(MYSQL *conn, MYSQL_STMT *stmt, char *message, bool
close_stmt)
{
    print_stmt_error(stmt, message);
10    if(close_stmt) mysql_stmt_close(stmt);
    mysql_close(conn);
    exit(EXIT_FAILURE);
}

15 static void print_dashes(MYSQL_RES *res_set)
{
    MYSQL_FIELD *field;
    unsigned int i, j;

20    mysql_field_seek(res_set, 0);
    putchar('+');
    for (i = 0; i < mysql_num_fields(res_set); i++) {
        field = mysql_fetch_field(res_set);
        for (j = 0; j < field->max_length + 2; j++)
25            putchar('-');
        putchar('+');
    }
    putchar('\n');
}

30 static void dump_result_set_header(MYSQL_RES *res_set)
{
    MYSQL_FIELD *field;
    unsigned long col_len;
```

```
unsigned int i;

/* determine column display widths -- requires result set to be */
/* generated with mysql_store_result(), not mysql_use_result() */
5
mysql_field_seek (res_set, 0);

for (i = 0; i < mysql_num_fields (res_set); i++) {
    field = mysql_fetch_field (res_set);
10    col_len = strlen(field->name);

    if (col_len < field->max_length)
        col_len = field->max_length;
    if (col_len < 4 && !IS_NOT_NULL(field->flags))
15        col_len = 4; /* 4 = length of the word "NULL" */
    field->max_length = col_len; /* reset column info */
}

print_dashes(res_set);
20 putchar('|');
mysql_field_seek (res_set, 0);
for (i = 0; i < mysql_num_fields(res_set); i++) {
    field = mysql_fetch_field(res_set);
    printf(" %-*s |", (int)field->max_length, field->name);
25 }
putchar('\n');

print_dashes(res_set);
}
30
void dump_result_set(MYSQL *conn, MYSQL_STMT *stmt, char *title)
{
    int i;
    int status;
```

```

int num_fields;    /* number of columns in result */
MYSQL_FIELD *fields; /* for result set metadata */
MYSQL_BIND *rs_bind; /* for output buffers */
MYSQL_RES *rs_metadata;
5  MYSQL_TIME *date;
   size_t attr_size;

/* Prefetch the whole result set. This in conjunction with
   * STMT_ATTR_UPDATE_MAX_LENGTH set in `setup_prepared_stmt`
10  * updates the result set metadata which are fetched in this
   * function, to allow to compute the actual max length of
   * the columns.
   */
if (mysql_stmt_store_result(stmt)) {
15     fprintf(stderr, " mysql_stmt_execute(), 1 failed\n");
     fprintf(stderr, " %s\n", mysql_stmt_error(stmt));
     exit(0);
}

20  /* the column count is > 0 if there is a result set */
   /* 0 if the result is only the final status packet */
   num_fields = mysql_stmt_field_count(stmt);

if (num_fields > 0) {
25     /* there is a result set to fetch */
     printf("%s\n", title);

     if((rs_metadata = mysql_stmt_result_metadata(stmt)) == NULL) {
         finish_with_stmt_error(conn, stmt, "Unable to retrieve result metadata\n",
30  true);
     }

     dump_result_set_header(rs_metadata);

```

```
fields = mysql_fetch_fields(rs_metadata);

rs_bind = (MYSQL_BIND *)malloc(sizeof (MYSQL_BIND) * num_fields);
if (!rs_bind) {
5      finish_with_stmt_error(conn, stmt, "Cannot allocate output buffers\n", true);
}
memset(rs_bind, 0, sizeof (MYSQL_BIND) * num_fields);

/* set up and bind result set output buffers */
10 for (i = 0; i < num_fields; ++i) {

    // Properly size the parameter buffer
    switch(fields[i].type) {
        case MYSQL_TYPE_DATE:
        case MYSQL_TYPE_TIMESTAMP:
15      case MYSQL_TYPE_DATETIME:
        case MYSQL_TYPE_TIME:
            attr_size = sizeof(MYSQL_TIME);
            break;
20      case MYSQL_TYPE_FLOAT:
            attr_size = sizeof(float);
            break;
        case MYSQL_TYPE_DOUBLE:
            attr_size = sizeof(double);
25      break;
        case MYSQL_TYPE_TINY:
            attr_size = sizeof(signed char);
            break;
        case MYSQL_TYPE_SHORT:
        case MYSQL_TYPE_YEAR:
30      attr_size = sizeof(short int);
            break;
        case MYSQL_TYPE_LONG:
        case MYSQL_TYPE_INT24:
```

```
        attr_size = sizeof(int);
        break;
    case MYSQL_TYPE_LONGLONG:
        attr_size = sizeof(int);
5         break;
    default:
        attr_size = fields[i].max_length;
        break;
    }

10

    // Setup the binding for the current parameter
    rs_bind[i].buffer_type = fields[i].type;
    rs_bind[i].buffer = malloc(attr_size + 1);
    rs_bind[i].buffer_length = attr_size + 1;

15

    if(rs_bind[i].buffer == NULL) {
        finish_with_stmt_error(conn, stmt, "Cannot allocate output buffers\n",
true);
    }

20 }

    if(mysql_stmt_bind_result(stmt, rs_bind)) {
        finish_with_stmt_error(conn, stmt, "Unable to bind output parameters\n",
true);
25 }

    /* fetch and display result set rows */
    while (true) {
        status = mysql_stmt_fetch(stmt);

30

        if (status == 1 || status == MYSQL_NO_DATA)
            break;

        putchar('|');
```

```
for (i = 0; i < num_fields; i++) {  
  
    if (rs_bind[i].is_null_value) {  
5         printf (" %-*s |", (int)fields[i].max_length, "NULL");  
        continue;  
    }  
  
    switch (rs_bind[i].buffer_type) {  
10  
        case MYSQL_TYPE_VAR_STRING:  
        case MYSQL_TYPE_DATETIME:  
            printf("    %-*s    |", (int)fields[i].max_length,  
15            (char*)rs_bind[i].buffer);  
            break;  
  
        case MYSQL_TYPE_DATE:  
        case MYSQL_TYPE_TIMESTAMP:  
            date = (MYSQL_TIME *)rs_bind[i].buffer;  
20            printf(" %d-%02d-%02d |", date->year, date->month,  
            date->day);  
            break;  
  
        case MYSQL_TYPE_STRING:  
25            printf(" %-*s |", (int)fields[i].max_length, (char  
            *)rs_bind[i].buffer);  
            break;  
  
        case MYSQL_TYPE_FLOAT:  
30        case MYSQL_TYPE_DOUBLE:  
            printf(" %.02f |", *(float *)rs_bind[i].buffer);  
            break;  
  
        case MYSQL_TYPE_LONG:
```

```

case MYSQL_TYPE_SHORT:
case MYSQL_TYPE_TINY:
    printf("  %-*d  |", (int)fields[i].max_length, *(int
*)rs_bind[i].buffer);
5          break;

case MYSQL_TYPE_NEWDECIMAL:
    printf(" %-*02lf |", (int)fields[i].max_length, *(float*)
rs_bind[i].buffer);
10          break;

default:
    printf("ERROR:  Unhandled  type  (%d)\n",
rs_bind[i].buffer_type);
15          abort();
        }
    }
    putchar('\n');
    print_dashes(rs_metadata);
20  }

mysql_free_result(rs_metadata); /* free metadata */

/* free output buffers */
25  for (i = 0; i < num_fields; i++) {
        free(rs_bind[i].buffer);
    }
    free(rs_bind);
    }
30  }

```

- parse

```
#include <stddef.h>
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
5 #include <string.h>
```

```
#include "defines.h"
```

```
#define BUFF_SIZE 4096
```

```
10
```

```
// The final config struct will point into this
```

```
static char config[BUFF_SIZE];
```

```
/**
```

```
15 * JSON type identifier. Basic types are:
```

```
*     o Object
```

```
*     o Array
```

```
*     o String
```

```
*     o Other primitive: number, boolean (true/false) or null
```

```
20 */
```

```
typedef enum {
```

```
    JSMN_UNDEFINED = 0,
```

```
    JSMN_OBJECT = 1,
```

```
    JSMN_ARRAY = 2,
```

```
25    JSMN_STRING = 3,
```

```
    JSMN_PRIMITIVE = 4
```

```
} jsmntype_t;
```

```
enum jsmnerr {
```

```
30     /* Not enough tokens were provided */
```

```
    JSMN_ERROR_NOMEM = -1,
```

```
    /* Invalid character inside JSON string */
```

```
    JSMN_ERROR_INVALID = -2,
```

```
    /* The string is not a full JSON packet, more bytes expected */
```



```
JSMN_ERROR_PART = -3

};

/**
5  * JSON token description.
   * type          type (object, array, string etc.)
   * start          start position in JSON data string
   * end           end position in JSON data string
   */
10 typedef struct {
    jsmntype_t type;
    int start;
    int end;
    int size;
15 #ifdef JSMN_PARENT_LINKS
    int parent;
#endif
    } jsmntok_t;

20 /**
   * JSON parser. Contains an array of token blocks available. Also stores
   * the string being parsed now and current position in that string
   */
typedef struct {
25     unsigned int pos; /* offset in the JSON string */
    unsigned int toknext; /* next token to allocate */
    int toksuper; /* superior token node, e.g parent object or array */
    } jsmn_parser;

30 /**
   * Allocates a fresh unused token from the token pool.
   */
static jsmntok_t *jsmn_alloc_token(jsmn_parser *parser, jsmntok_t *tokens, size_t num_tokens) {
    jsmntok_t *tok;
```

```
        if (parser->toknext >= num_tokens) {
            return NULL;
        }
        tok = &tokens[parser->toknext++];
5      tok->start = tok->end = -1;
        tok->size = 0;
#ifdef JSMN_PARENT_LINKS
        tok->parent = -1;
#endif
10     return tok;
    }

/**
 * Fills token type and boundaries.
15 */
static void jsmn_fill_token(jsmntok_t *token, jsmntype_t type,
                           int start, int end) {
    token->type = type;
    token->start = start;
20    token->end = end;
    token->size = 0;
}

/**
25 * Fills next available token with JSON primitive.
 */
static int jsmn_parse_primitive(jsmn_parser *parser, const char *js,
                               size_t len, jsmntok_t *tokens, size_t num_tokens) {
    jsmntok_t *token;
30    int start;

    start = parser->pos;

    for (; parser->pos < len && js[parser->pos] != '\0'; parser->pos++) {
```

```
        switch (js[parser->pos]) {
#ifdef JSMN_STRICT
            /* In strict mode primitive must be followed by "," or "]" or "]" */
            case ':':

5      #endif

            case '\t' : case '\r' : case '\n' : case ' ' :
            case ',' : case ']' : case '}' :
                goto found;
        }
10      if (js[parser->pos] < 32 || js[parser->pos] >= 127) {
            parser->pos = start;
            return JSMN_ERROR_INVALID;
        }
    }
15 #ifdef JSMN_STRICT
        /* In strict mode primitive must be followed by a comma/object/array */
        parser->pos = start;
        return JSMN_ERROR_PART;
    #endif
20 found:
    if (tokens == NULL) {
        parser->pos--;
        return 0;
25    }
    token = jsmn_alloc_token(parser, tokens, num_tokens);
    if (token == NULL) {
        parser->pos = start;
        return JSMN_ERROR_NOMEM;
30    }
    jsmn_fill_token(token, JSMN_PRIMITIVE, start, parser->pos);
#ifdef JSMN_PARENT_LINKS
        token->parent = parser->toksuper;
#endif
#endif
```

```
    parser->pos--;
    return 0;
}

5  /**
   * Fills next token with JSON string.
   */
static int jsmn_parse_string(jsmn_parser *parser, const char *js,
    size_t len, jsmntok_t *tokens, size_t num_tokens) {
10     jsmntok_t *token;

    int start = parser->pos;

    parser->pos++;

15     /* Skip starting quote */
    for (; parser->pos < len && js[parser->pos] != '\0'; parser->pos++) {
        char c = js[parser->pos];

20         /* Quote: end of string */
        if (c == '"') {
            if (tokens == NULL) {
                return 0;
            }

25             token = jsmn_alloc_token(parser, tokens, num_tokens);
            if (token == NULL) {
                parser->pos = start;
                return JSMN_ERROR_NOMEM;
            }

30             jsmn_fill_token(token, JSMN_STRING, start+1, parser->pos);
#ifdef JSMN_PARENT_LINKS
                token->parent = parser->toksuper;
#endif
            return 0;
        }
    }
```

```

    }

    /* Backslash: Quoted symbol expected */
    if (c == '\\' && parser->pos + 1 < len) {
5         int i;
        parser->pos++;
        switch (js[parser->pos]) {
            /* Allowed escaped symbols */
            case '\\': case '/' : case '\\' : case 'b' :
10         case 'f' : case 'r' : case 'n' : case 't' :
                break;
            /* Allows escaped symbol \uXXXX */
            case 'u':
                parser->pos++;
15         for(i = 0; i < 4 && parser->pos < len && js[parser->pos] != '\
0'; i++) {

                /* If it isn't a hex character we have an error */
                if(!((js[parser->pos] >= 48 && js[parser->pos] <= 57)
20         || /* 0-9 */
                (js[parser->pos] >= 65 &&
js[parser->pos] <= 70) || /* A-F */
                (js[parser->pos] >= 97 &&
js[parser->pos] <= 102)))) { /* a-f */

                    parser->pos = start;
25         return JSMN_ERROR_INVALID;
                }
                parser->pos++;
            }
            parser->pos--;
30         break;
        /* Unexpected symbol */
        default:
            parser->pos = start;
            return JSMN_ERROR_INVALID;
    }

```

```
        }
    }
}
parser->pos = start;
5   return JSMN_ERROR_PART;
}

/**
 * Parse JSON string and fill tokens.
10 */
static int jsmn_parse(jsmn_parser *parser, const char *js, size_t len, jsmntok_t *tokens, unsigned int
num_tokens) {
    int r;
    int i;
15   jsmntok_t *token;
    int count = parser->toknext;

    for (; parser->pos < len && js[parser->pos] != '\0'; parser->pos++) {
        char c;
20         jsmntype_t type;

        c = js[parser->pos];
        switch (c) {
            case '{': case '[':
25                 count++;
                if (tokens == NULL) {
                    break;
                }
                token = jsmn_alloc_token(parser, tokens, num_tokens);
30                 if (token == NULL)
                    return JSMN_ERROR_NOMEM;
                if (parser->toksuper != -1) {
                    tokens[parser->toksuper].size++;
                }
            }
        }
    }
    return JSMN_ERROR_PART;
}

#ifdef JSMN_PARENT_LINKS
```

```
token->parent = parser->tksuper;

#endif

    }
    token->type = (c == '{' ? JSMN_OBJECT : JSMN_ARRAY);
5    token->start = parser->pos;
    parser->tksuper = parser->toknext - 1;
    break;
case '}': case ']':
    if (tokens == NULL)
10        break;
    type = (c == '}' ? JSMN_OBJECT : JSMN_ARRAY);
#ifdef JSMN_PARENT_LINKS
    if (parser->toknext < 1) {
        return JSMN_ERROR_INVALID;
15    }
    token = &tokens[parser->toknext - 1];
    for (;;) {
        if (token->start != -1 && token->end == -1) {
            if (token->type != type) {
20                return JSMN_ERROR_INVALID;
            }
            token->end = parser->pos + 1;
            parser->tksuper = token->parent;
            break;
25        }
        if (token->parent == -1) {
            if (token->type != type || parser->tksuper == -1) {
                return JSMN_ERROR_INVALID;
            }
30            break;
        }
        token = &tokens[token->parent];
    }
#else
```

```

for (i = parser->toknext - 1; i >= 0; i--) {
    token = &tokens[i];
    if (token->start != -1 && token->end == -1) {
        if (token->type != type) {
5             return JSMN_ERROR_INVALID;
        }
        parser->toksuper = -1;
        token->end = parser->pos + 1;
        break;
10     }
}
/* Error if unmatched closing bracket */
if (i == -1) return JSMN_ERROR_INVALID;
for (; i >= 0; i--) {
15     token = &tokens[i];
    if (token->start != -1 && token->end == -1) {
        parser->toksuper = i;
        break;
    }
20 }
#endif

break;
case '\":
    r = jsmn_parse_string(parser, js, len, tokens, num_tokens);
25     if (r < 0) return r;
    count++;
    if (parser->toksuper != -1 && tokens != NULL)
        tokens[parser->toksuper].size++;
    break;
30 case '\t' : case '\r' : case '\n' : case ' ':
    break;
case ' ':
    parser->toksuper = parser->toknext - 1;
    break;

```



```

        case ',':
            if (tokens != NULL && parser->toksuper != -1 &&
                tokens[parser->toksuper].type != JSMN_ARRAY &&
                tokens[parser->toksuper].type != JSMN_OBJECT) {
5   #ifdef JSMN_PARENT_LINKS
            parser->toksuper = tokens[parser->toksuper].parent;

        #else

            for (i = parser->toknext - 1; i >= 0; i--) {
                if (tokens[i].type == JSMN_ARRAY || tokens[i].type
10  == JSMN_OBJECT) {
                    if (tokens[i].start != -1 && tokens[i].end == -1)
                        {
                            parser->toksuper = i;
                            break;
15  }
                }
            }

        #endif
    }

20  break;

    #ifdef JSMN_STRICT
        /* In strict mode primitives are: numbers and booleans */
        case '-': case '0': case '1': case '2': case '3': case '4':
        case '5': case '6': case '7': case '8': case '9':
25  case 't': case 'f': case 'n':
            /* And they must not be keys of the object */
            if (tokens != NULL && parser->toksuper != -1) {
                jsmntok_t *t = &tokens[parser->toksuper];
                if (t->type == JSMN_OBJECT ||
30  (t->type == JSMN_STRING && t->size != 0)) {
                    return JSMN_ERROR_INVALID;
                }
            }
        }
    #else

```

```
/* In non-strict mode every unquoted value is a primitive */
default:

#endif

    r = jsmn_parse_primitive(parser, js, len, tokens, num_tokens);
5    if (r < 0) return r;
    count++;
    if (parser->toksuper != -1 && tokens != NULL)
        tokens[parser->toksuper].size++;
    break;
10

#ifdef JSMN_STRICT
    /* Unexpected char in strict mode */
    default:
        return JSMN_ERROR_INVALID;
15 #endif
    }
}

    if (tokens != NULL) {
20     for (i = parser->toknext - 1; i >= 0; i--) {
        /* Unmatched opened object or array */
        if (tokens[i].start != -1 && tokens[i].end == -1) {
            return JSMN_ERROR_PART;
        }
25     }
    }

    return count;
}

30

/**
 * Creates a new parser based over a given buffer with an array of tokens
 * available.
 */
```

```
static void jsmn_init(jsmn_parser *parser) {
    parser->pos = 0;
    parser->toknext = 0;
    parser->toksuper = -1;
5   }

static int jsoneq(const char *json, jsmntok_t *tok, const char *s)
{
    if (tok->type == JSMN_STRING
10     && (int) strlen(s) == tok->end - tok->start
        && strncmp(json + tok->start, s, tok->end - tok->start) == 0) {
        return 0;
    }
    return -1;
15 }

static size_t load_file(char *filename)
{
    FILE *f = fopen(filename, "rb");
20   if(f == NULL) {
        fprintf(stderr, "Unable to open file %s\n", filename);
        exit(1);
    }

25   fseek(f, 0, SEEK_END);
    size_t fsize = ftell(f);
    fseek(f, 0, SEEK_SET); //same as rewind(f);

    if(fsize >= BUFF_SIZE) {
30     fprintf(stderr, "Configuration file too large\n");
        abort();
    }

    fread(config, fsize, 1, f);
```

```
        fclose(f);

        config[fsize] = 0;
        return fsize;
5    }

int parse_config(char *path, struct configuration *conf)
{
    int i;
10    int r;
    jsmn_parser p;
    jsmntok_t t[128]; /* We expect no more than 128 tokens */

    load_file(path);
15

    jsmn_init(&p);
    r = jsmn_parse(&p, config, strlen(config), t, sizeof(t)/sizeof(t[0]));
    if (r < 0) {
        printf("Failed to parse JSON: %d\n", r);
20        return 0;
    }

    /* Assume the top-level element is an object */
    if (r < 1 || t[0].type != JSMN_OBJECT) {
25        printf("Object expected\n");
        return 0;
    }

    /* Loop over all keys of the root object */
30    for (i = 1; i < r; i++) {
        if (jsoneq(config, &t[i], "host") == 0) {
            /* We may use strdup() to fetch string value */
            conf->host = strdup(config + t[i+1].start, t[i+1].end-t[i+1].start);
            i++;
        }
    }
}
```

```
    } else if (jsoneq(config, &t[i], "username") == 0) {  
        conf->db_username = strdup(config + t[i+1].start, t[i+1].end-t[i+1].start);  
        i++;  
    } else if (jsoneq(config, &t[i], "password") == 0) {  
5        conf->db_password = strdup(config + t[i+1].start, t[i+1].end-t[i+1].start);  
        i++;  
    } else if (jsoneq(config, &t[i], "port") == 0) {  
        conf->port = strtol(config + t[i+1].start, NULL, 10);  
        i++;  
10    } else if (jsoneq(config, &t[i], "database") == 0) {  
        conf->database = strdup(config + t[i+1].start, t[i+1].end-t[i+1].start);  
        i++;  
    } else {  
        printf("Unexpected key: %.*s\n", t[i].end-t[i].start, config + t[i].start);  
15    }  
    }  
    return 1;  
}
```

20

25