**Immagine che contiene tastiera

Descrizione generata automaticamente**

**Edoardo Mariani 1815762 Web-Shop 31/08/2020**

**Indice**

Web-Shop: pag 2;

Progettazione concettuale: pag 4;

Specifica dei requisiti: pag 6;

Glossario dei termini: pag 9;

Diagramma E/R: pag 12;

Dizionario dei dati: pag 13;

Vincoli esterni: pag 17;

Tabella dei volumi: pag 19;

Tabella delle operazioni: pag 20;

Ristrutturazione schema concettuale: pag 21;

Diagramma E/R ristrutturato: pag 26;

Dizionario dei dati ristrutturato: pag 27;

Vincoli esterni Diagramma ristrutturato: pag 33;

Tabella dei volumi ristrutturati: pag 36;

Tabella di accesso: pag 37;

Traduzione diretta al modello relazionale. Pag 40;

Creazione tabelle: Linguaggio SQL: pag 43;

Vincoli Trigger Funzioni: pag 43;

Operazione sulla base di dati QUERY: pag 74;

Soddisfazione dei requisiti 2: pag 77;

**Web-Shop**

Si vuole realizzare un sito web di compravendita di abiti usati in Europa. Il sistema deve permettere l’acquisto e la vendita degli indumenti nonché garantire tutti i dati relativi a tali operazioni. Abbiamo gli utenti di cui vogliamo conoscere il nome, il cognome, la data di nascita, la città di residenza, l’e-mail e il nickname con cui è autenticato nel sistema (quest’ultimo può essere opzionale), la media dei feedback ottenuti, l’indirizzo di residenza, il codice fiscale e il numero/i di telefono/i. Ogni utente può pubblicare, comprare e vendere un annuncio che non è altro che la rappresentazione del capo d’abbigliamento. Esso ha un codice (che è univoco), la relativa data e ora della pubblicazione, della vendita e dell’ acquisto, delle foto (che possono essere delle stringhe), una breve descrizione, il prezzo e il periodo dell’usato (meno di un mese, meno di 6 mesi,più di un anno). Per capire l’affidabilità dei prodotti venduti dagli utenti, il sistema fornisce anche la possibilità di rilasciare feedback ad un annuncio, ma solo ad acquisto avvenuto e per questo si vuole tenere traccia dei feedback generati da un utente e assegnati ad un altro utente. Esso è caratterizzato da un voto il quale valore può assumere un numero compreso da 1 a 10.

Per capire il trend, si vuole fare un’analisi di mercato capendo quale categorie di vestiti sono quelle dominanti. Proprio per questo è importante che gli annunci siano divisi in categorie che hanno come tipologia i maglioni, le felpe, i pantaloni, le scarpe, le t-shirt e i cappelli.

Per una questione di sicurezza, si è pensato di far avere agli utenti un conto ricaricabile sulla piattaforma, senza essere direttamente collegati col conto in banca. Di questo conto, interessa il saldo e i relativi movimenti, con l’importo, la data e l’ora del movimento.

Si vuole tenere traccia anche delle spedizioni relative agli annunci venduti di cui si vuole sapere la tipologia, le date delle spedizioni effettuate e di quelle non ancora effettuate con il loro relativo codice. La tipologia della spedizione può essere autonoma, ovvero se ne possono occupare autonomamente gli utenti, oppure di servizio. Quest’ultima è particolare in quanto a usare questo sito non sono solo gli utenti ma anche i direttori di vari magazzini i quali vengono riforniti da aziende che, avendo stock di abiti datati o che non sono riusciti a vendere nell’anno corrente, collocano i vestiti all’interno di questi depositi. Quindi abbiamo le aziende che riforniscono vari magazzini e i direttori di quest’ultimi si occupano solo di pubblicare e vendere i vestiti sulla piattaforma. Il sistema per riconoscere un direttore di magazzino rispetto ad un normale utente, controlla se è stata caricata durante la fase di registrazione una speciale licenza che, se approvata, attiva la seguente modalità: “Less-Tax”. I guadagni del sito provengono dalla tassa del 5% applicata ad ogni annuncio venduto ma, se il venditore è un direttore di magazzino, dato che normalmente gli annunci pubblicati da quest’ultimo sono numerosi, si applica una tassa minore se vengono incassati più di un tot di soldi al mese: la “Less-Tax” del 1,5%. Per questo è di interesse al sito di tener traccia del loro conto.

Ad ogni modo dei magazzini sono di interesse l’indirizzo, nonché le città in cui si trovano, l’azienda che li rifornisce ,la ragione sociale,la spedizione che, se è di servizio, preleva gli indumenti direttamente nel magazzino corrispondente (di cui si vuole sapere la data e l’ora) e li consegna al cliente. Per ogni azienda che fornisce i magazzini si vogliono conoscere l’indirizzo, la città in cui si trova, la ragione sociale, il nome, i dipendenti che ci lavorano e i direttori che la dirigono. Uno e un solo dipendente può dirigere l’azienda. Dei dipendenti si vogliono conoscere il nome, il cognome, il codice fiscale e la data di nascita. Infine si vuole tenere lo storico degli eventuali dipendenti/direttori che lavoravano in una determinata azienda e che quindi non lavorano più attualmente. Alcuni di questi dipendenti che lavorano tutt’ora (ma non i direttori), si occupano delle spedizioni degli annunci che riguardano solo i magazzini che vengono riforniti dall’azienda dove lavorano. Si vuole tenera traccia di questi dipendenti in modo da capire che provvedimenti prendere quando una consegna a loro carico non viene effettuata entro un certo periodo.

**Progettazione Concettuale**

**Strutturazione dei requisiti**

|  |
| --- |
| **FRASE RELATIVA ALL’UTENTE** |
| Abbiamo gli utenti di cui vogliamo conoscere il nome, il cognome, la data di nascita, la città di residenza, l’e-mail e il nickname con cui è autenticato nel sistema (quest’ultimo può essere opzionale), la media dei feedback ottenuti, l’indirizzo di residenza, il codice fiscale e il numero/i di telefono/i. |

|  |
| --- |
| **FRASE RELATIVA ALL’ANNUNCIO** |
| Esso ha un codice (che è univoco), la relativa data e ora della pubblicazione, della vendita e dell’ acquisto, delle foto (che possono essere delle stringhe), una breve descrizione, il prezzo e il periodo dell’usato (meno di un mese, meno di 6 mesi,più di un anno). |

|  |
| --- |
| **FRASE RELATIVA AL FEEDBACK** |
| Per capire l’affidabilità dei prodotti venduti dagli utenti, il sistema fornisce anche la possibilità di rilasciare feedback ad un annuncio, ma solo ad acquisto avvenuto e per questo si vuole tenere traccia dei feedback generati da un utente e assegnati ad un altro utente. Esso è caratterizzato da un voto il quale valore può assumere un numero compreso da 1 a 10. |

|  |
| --- |
| **FRASE RELATIVA ALLE CATEGORIA** |
| Per capire il trend, si vuole fare un’analisi di mercato capendo quale categorie di vestiti sono quelle dominanti. Proprio per questo è importante che gli annunci siano divisi in categorie che hanno come tipologia i maglioni, le felpe, i pantaloni, le scarpe, le t-shirt e i cappelli. |

|  |
| --- |
| **FRASE RELATIVA AL CONTO** |
| Per una questione di sicurezza, si è pensato di far avere agli utenti un conto ricaricabile sulla piattaforma, senza essere direttamente collegati col conto in banca. Di questo conto, interessa il saldo e i relativi movimenti, con l’importo, la data e l’ora del movimento. |

|  |
| --- |
| **FRASE RELATIVA AL MOVIMENTO** |
| Di questo conto, interessa il saldo e i relativi movimenti con l’importo, la data e l’ora del movimento. |

|  |
| --- |
| **FRASE RELATIVA ALLA SPEDIZIONE** |
| Si vuole tenere traccia anche delle spedizioni relative agli annunci venduti di cui si vuole sapere la tipologia, le date delle spedizioni effettuate e di quelle non ancora effettuate con il loro relativo codice. La tipologia della spedizione può essere autonoma, ovvero se ne possono occupare autonomamente gli utenti, oppure di servizio. |

|  |
| --- |
| **FRASE RELATIVA AL DIRETTORE DI MAGAZZINO** |
| A usare questo sito non sono solo gli utenti ma anche i direttori di vari magazzini i quali vengono riforniti da aziende che, avendo stock di abiti datati o che non sono riusciti a vendere nell’anno corrente, collocano i vestiti all’interno di questi depositi. Quindi abbiamo le aziende che riforniscono vari magazzini e i direttori di quest’ultimi si occupano solo di pubblicare e vendere i vestiti sulla piattaforma. Il sistema per riconoscere un direttore di magazzino rispetto ad un normale utente, controlla se è stata caricata durante la fase di registrazione una speciale licenza. |

|  |
| --- |
| **FRASE RELATIVA AL MAGAZZINO** |
| dei magazzini sono di interesse l’indirizzo, nonché le città in cui si trovano, l’azienda che li rifornisce ,la ragione sociale,la spedizione che, se è di servizio, preleva gli indumenti direttamente nel magazzino corrispondente (di cui si vuole sapere la data e l’ora) e li consegna al cliente. |

|  |
| --- |
| **FRASE RELATIVA ALL’AZIENDA** |
| Per ogni azienda che fornisce i magazzini si vogliono conoscere l’indirizzo, la città in cui si trova, la ragione sociale, il nome, i dipendenti che ci lavorano e i direttori che la dirigono. |

|  |
| --- |
| **FRASE RELATIVA AI DIPENDENTI** |
| Dei dipendenti si vogliono conoscere il nome, il cognome, il codice fiscale e la data di nascita. |

**Specifica dei requisiti**

**1** Di ogni **SOGGETTO** si vogliono conoscere:

* 1. il nome;
  2. il cognome;
  3. la data di nascita;
  4. la **CITTA’** di residenza (vedi **2**);
  5. l’email;
  6. se è un **UTENTE** (vedi **3**) o un **DIRETTORE DI MAGAZZINO** (vedi **13**);
  7. l’indirizzo che a sua volta è formato da:
     1. la via;
     2. il civico;
     3. il CAP;
  8. il codice fiscale;
  9. i/il numero/i di telefono/i;
  10. il **CONTO** (vedi **5**);
  11. l’**ANNUNCIO** (vedi **6**) che ha acquistato/venduto/pubblicato;

1. Di ogni **CITTA’** si vogliono conoscere:
   1. il nome;
   2. la **NAZIONE** (vedi **3**) di cui fa parte;
2. Di ogni **NAZIONE** si vogliono conoscere:
   1. il nome;
3. Di ogni **FEEDBACK** si vogliono conoscere:
   1. il voto;
   2. l’**ANNUNCIO** (vedi **6**) a cui è stato generato;
4. Di ogni **CONTO** si vogliono conoscere:
   1. il saldo;
   2. il **MOVIMENTO** (vedi **11**);
   3. il **SOGGETTO** (vedi **1**) proprietario;
5. Di ogni **ANNUNCIO** si vogliono conoscere:
   1. una descrizione;
   2. il codice;
   3. il periodo dell’usato;
   4. il prezzo;
   5. delle foto (che possono essere stringhe);
   6. la data e l’orario di:
      1. annuncio pubblicato;
      2. annuncio vendutose è stato venduto di cui si vuole sapere:
         1. la **SPEDIZIONE** (vedi **10**);
      3. annuncio acquistato se è stato acquistato di cui si vogliono sapere:
         1. il **FEEDBACK** (vedi **4**) (se presente);
   7. il **MAGAZZINO** (vedi **9**);
   8. il **SOGGETTO** (vedi **1**) che l’ha pubblicato/venduto/acquistato;
   9. la **CATEGORIA** di cui fa parte (vedi **7**);
6. Di ogni **CATEGORIA** si vogliono conoscere:
   1. il tipo di categoria (maglioni, felpe, pantaloni, scarpe, t-shirt, cappelli);
7. Di ogni **AZIENDA** si vogliono conoscere:
   1. il nome;
   2. la ragione sociale;
   3. la **CITTA’** (vedi **2**) dove risiede;
   4. l’indirizzo che a sua volta è formato da:
      1. la via;
      2. il civico;
      3. il CAP;
   5. i **DIPENDENTI** (vedi **12**);
8. Di ogni **MAGAZZINO** si vogliono conoscere:
   1. l’indirizzo che a sua volta è formato da:
      1. la via;
      2. il civico;
      3. il CAP;
   2. la **CITTA’**  (vedi **3**) in cui si trova;
   3. la ragione sociale;
   4. l’**AZIENDA** (vedi **8**) fornitrice;
   5. il **DIRETTORE DI MAGAZZINO** (vedi **13**) proprietario;
9. Di ogni **SPEDIZIONE** si vogliono conoscere:
   1. la tipologia di spedizione che può essere:
      1. autonoma;
      2. di servizio di cui si vuole sapere da quale **MAGAZZINO** (vedi **9**) viene prelevato l’ordine;
   2. si può trovare nei seguenti stati:
      1. effettuata;
      2. non effettuata;
   3. un codice;
   4. la data;
   5. l’**UTENTE** (vedi **1**) a cui spedirlo;
10. Di ogni **MOVIMENTO** si vogliono conoscere:
    1. la data e l’ora;
    2. importo;
    3. il **CONTO** (vedi **5**) di cui fa parte;
11. Di ogni **DIPENDENTE** si vogliono conoscere:
    1. il nome;
    2. il cognome;
    3. la data di nascita;
    4. il codice fiscale;
    5. l’**AZIENDA** (vedi **8**) in cui lavora e/o se la dirige;
    6. la **SPEDIZIONE** (vedi **10**) che ha fatto;
12. Di ogni **DIRETTORE DI MAGAZZINO** si vogliono conoscere oltre ai dati di **SOGGETTO** (vedi **1**):
    1. la licenza (booleano);
13. Di ogni **UTENTE si** vogliono conoscere oltre ai dati di **SOGGETTO** (vedi **1**):
    1. Il nickname (opzionale);
    2. La media\_feedback;

**Glossario dei termini**

|  |  |  |  |
| --- | --- | --- | --- |
| **TERMINE** | **DESCRIZIONE** | **SINONIMI** | **COLLEGAMENTI** |
| Soggetto | Rappresenta l’attore che interagisce col sito (compra/ vende/ pubblica annunci); | Attore | Annuncio, Conto, Città |
| Utente | E’ colui che compra/ vende/ pubblica vestiti nel sistema; | Cliente | Annuncio, Conto, Città, Feedback, Spedizione |
| Città | Rappresenta le città in una nazione di cui fa parte; |  | Nazione,Soggetto, Magazzino, Azienda |
| Nazione | Rappresenta le nazioni; |  | Città |
| Feedback | Rappresenta la valutazione di un prodotto da parte di un utente; |  | Utente,Annuncio Acquistato |
| Azienda | Azienda fornitrice di vestiti datati per un magazzino; |  | Dipendente,Magazzino,Città |
| Conto | E’ il conto corrente o di utente o di un magazzino; | Banca | Soggetto,Movimento |
| Annuncio | Rappresenta l’indumento sul sito; | Prodotto, Ordine,  Indumento,  vestiti,  merce | Soggetto, Categoria |
| Annuncio Venduto | Rappresenta l’annuncio venduto; | Annuncio | Soggetto, Categoria, Spedizione |
| Annuncio Acquistato | Rappresenta l’annuncio acquistato; | Annuncio | Categoria, Soggetto, Feedback |
| Categoria | Rappresenta la tipologia di un indumento; |  | Annuncio |
| Magazzino | E’ il posto in cui vengono lasciati stock di vestiti da un’azienda; |  | Servizio, Azienda, Direttore magazzino, Città |
| Spedizione | Rappresenta la merce che deve essere inviata o da un utente o da un magazzino; |  | Annuncio venduto, Utente |
| Servizio | Rappresenta la tipologia di spedizione; | Spedizione | Utente, Annuncio venduto, Dipendente, Magazzino |
| Autonoma | Rappresenta la tipologia di spedizione; | Spedizione | Utente, Annuncio venduto, Dipendente |
| Effettuata | Rappresenta la spedizione effettuata; | Spedizione | Utente, Annuncio venduto, Dipendente, Magazzino |
| Non Effettuata | Rappresenta la spedizione non effettuata; | Spedizione | Utente, Annuncio venduto, Dipendente, Magazzino |
| Movimento | Rappresenta i  pagamenti/  guadagni effettuati da parte di un utente o un magazzino; |  | Conto |
| Dipendente | E’ colui che lavora in una azienda e/o effettua spedizioni; |  | Azienda, Servizio |
| Direttore di magazzino | E’ colui che gestisce i magazzini di sua proprietà e lui può vendere e pubblicare annunci |  | Magazzino, Annuncio, Conto, Città |

**Diagramma E/R**

Immagine che contiene testo, mappa

Descrizione generata automaticamente

**Dizionario dei dati**

|  |  |  |  |
| --- | --- | --- | --- |
| **ENTITA’** | **DESCRIZIONE** | **ATTRIBUTI** | **IDENTIFICATORE** |
| Soggetto | Un Soggetto deve assumere uno dei due valori: Utente, Direttore di magazzino | nome, cognome, data\_nascita,  e-mail, CF, indirizzo | CF |
| Utente | Un utente che pubblica/compra/vende annunci | IS-A da Soggetto, media\_feedback, nickname\* |  |
| Città | La città di una nazione dove si trovano gli utenti, i magazzini, le aziende | nome | nome (identificatore esterno) |
| Nazione | Una nazione nel mondo | nome | nome |
| Feedback | Un feedback dato da un utente | voto |  |
| Conto | Un conto di un utente o di un magazzino | saldo |  |
| Annuncio | Un annuncio che viene pubblicato/ venduto/ acquistato da un Soggetto | codice, DataOra, foto, descrizione, prezzo, periodo\_usato | codice |
| Annuncio Venduto | Un annuncio che viene venduto ad un Soggetto | IS-A da Annuncio, DataOra |  |
| Annuncio Acquistato | Un annuncio che viene acquistato da un Utente | IS-A da Annuncio, DataOra |  |
| Categoria | Una categoria di un annuncio | nome |  |
| Azienda | Un’azienda che rifornisce un magazzino | indirizzo, nome, ragione\_sociale |  |
| Magazzino | Un magazzino dove all’interno ci sono i vestiti e quindi gli annunci; | indirizzo, nome, ragione\_sociale | indirizzo |
| Spedizione | Una spedizione di un annuncio | codice, data | codice |
| Effettuato | Una spedizione di annuncio venduto che è stata effettuata | IS-A da Spedizione, Data |  |
| Non Effettuato | Una spedizione di annuncio venduto ancora in corso | IS-A da Spedizione, Data |  |
| Autonoma | Una spedizione che viene gestita tra gli utenti; | IS-A da Spedizione |  |
| Servizio | Una spedizione che viene gestita da un magazzino | IS-A da Spedizione |  |
| Movimento | Un prelievo/versamento in un conto | importo, DataOra | DataOra (identificatore esterno) |
| Dipendente | Un dipendente di un’azienda | nome, cognome, Data\_nascita, CF | CF |
| Direttore Magazzino | Il direttore di un magazzino | IS-A da Soggetto |  |
| ContoSogg | Associa il Conto ad un Soggetto | Soggetto,Conto |  |
| ContMov | Associa un conto a dei movimenti | Conto,Movimento |  |
| ordine | Associa un Soggetto ad un Annuncio:  rappresenta quale Soggetto ha pubblicato l’Annuncio e a quale Utente è stato nel caso venduto e qual è il Soggetto che l’ha acquistato | Soggetto,Annuncio |  |
| generato | Associa un Utente ad uno o più Feedback: rappresenta quale Utente ha pubblicato il Feeback | Utente,Feedback |  |
| possiede | Associa un Feedback ad un Annuncio: rappresenta a quale Annuncio Acquistato appartiene il Feedback | Feedback,Annuncio Acquistato |  |
| tipo | Associa un Annuncio ad una Categoria | Annuncio, Categoria |  |
| compra | Associa un Annuncio Acquistato ad un Soggetto: rappresenta gli acquisti effettuati da un Soggetto |  |  |
| consegna | Associa un Annuncio Venduto ad una Spedizione: rappresenta quale annuncio verrà spedito | Annuncio Venduto, Spedizione |  |
| prelevazione | Associa una Spedizione ad un Magazzino: rappresenta dove la spedizione deve andare a prelevare l’ordine nel caso tale ordine sia stato venduto da un magazzino | Servizio, Magazzino, DataOra |  |
| proprietario | Associa un Direttore di magazzino ad un Magazzino: rappresenta il proprietario di un magazzino | Direttore di magazzino, Magazzino |  |
| CitMag | Associa un Magazzino ad una Città: rappresenta la città dove risiede un magazzino | Magazzino, Città |  |
| abita | Associa un Utente ad una Città: rappresenta in quale città un utente risiede | Soggetto,Città |  |
| CitNaz | Associa una Città ad una Nazione: rappresenta in quale nazione una città fa parte | Nazione, Città |  |
| fornitore | Associa un Magazzino ad un’Azienda: rappresenta quale azienda rifornisce un magazzino | Magazzino, Azienda |  |
| impiegato corrente | Associa un Dipendente ad un’Azienda: rappresenta l’impiegato che attualmente lavora in un’azienda | Azienda, Dipendente, inizio |  |
| direzione | Associa un Dipendente ad un’Azienda: rappresenta quale dipendente dirige l’azienda | Azienda, Dipendente  (IS-A da impiegato corrente) |  |
| impiegato passato | Associa un Dipendente ad un’Azienda: rappresenta quale dipendente lavorava in un’azienda | Azienda, Dipendente, fine |  |
| locazAz | Associa un’Azienda ad una Città: rappresenta la città dove risiede l’azienda | Azienda, Città |  |
| appartiene | Associa una Spedizione ad un Utente: rappresenta a quale utente deve essere effettuata la spedizione | Spedizione, Utente |  |
| fornito | Associa un Dipendente ad una Spedizione: rappresenta quale Dipendente ha effettuato la Spedizione | Servizio, Dipendente |  |

**Vincoli Esterni**

-**V.MagazzinoAzienda**: Un magazzino che risiede nella stessa città della sua azienda fornitrice non può avere lo stesso indirizzo di quest’ultima e viceversa:

* Per ogni M appartenente a MAGAZZINO e per ogni A appartenente a AZIENDA e per ogni C appartenente a CITTA’ e quindi per le istanze (M,C) di “locaAz” e (A,C) di “CitMag”, sia J l’indirizzo di M e J’ l’indirizzo di A, J != J’;

-**V.DirettoreMagazzino**: Un direttore di un magazzino non può acquistare annunci;

* Per ogni istanza D di DIRETTORE MAGAZZINO isa SOGGETTO non deve esistere l’istanza (D,AC) di “ordine” con AC appartenente a ANNUNCIO ACQUISTATO isa ANNUNCIO;

-**V.DipendenteAzienda**: Se un dipendente non lavora più nell’azienda dove lavorava, la data e l’ora di fine deve essere strettamente maggiore della data e l’ora dell’inizio:

* Per ogni istanza (D,A) di “impiegato corrente” con D appartenente a DIPENDENTE e A appartenente a AZIENDA, sia i l’attributo inizio di “impiegato corrente”, se (D,A) è anche un’istanza di “impiegato passato”, sia j l’attributo fine di “impiegato passato”, deve valere che j>i;

-**V.SpedizioneServizio**: Se un direttore di magazzino ha pubblicato un annuncio allora la sua eventuale spedizione deve essere per forza di servizio:

* Per ogni istanza (D,AV) di “ordine” con D appartenente a DIRETTORE MAGAZZINO isa SOGGETTO e AV appartenente a ANNUNCIO, se AV appartiene a ANNUNCIO VENDUTO isa ANNUNCIO, sia (AV,SP) con SP appartenente a SPEDIZIONE, SP deve per forza assumere il valore SER appartenente a SERVIZIO e non a AUTONOMA;

-**V.SpedizioneAutonoma**: Se un cliente ha pubblicato un annuncio allora la sua eventuale spedizione deve essere per forza autonoma:

* Per ogni istanza (U,AV) di “ordine” con U appartenente a UTENTE isa SOGGETTO e AV appartenente a ANNUNCIO, se AV appartiene a ANNUNCIO VENDUTO isa ANNUNCIO, sia (AV,SP) con SP appartenente a SPEDIZIONE, SP deve per forza assumere il valore AUT appartenente a AUTONOMA e non a SERVIZIO;

-**V.Feedback**: Se esiste un Feedback da parte di un utente deve per forza appartenere ad un annuncio acquistato:

* Se esiste F appartenente a FEEDBACK, allora deve esistere l’istanza (F,AC) di “possiede” dove AC appartiene a ANNUNCIO ACQUISTATO isa ANNUNCIO;

**Tabella dei volumi**

|  |  |  |
| --- | --- | --- |
| **CONCETTO** | **TIPO** | **VOLUME** |
| Soggetto | E | 1.500.000 |
| Utente | E | 1.000.000 |
| Direttore Magazzino | E | 500.000 |
| Città | E | 100 |
| Nazione | E | 30 |
| Feedback | E | 1.000.000 |
| Conto | E | 1.500.000 |
| Annuncio | E | 3.000.000 |
| Annuncio Venduto | E | 1.000.000 |
| Annuncio Acquistato | E | 1.000.000 |
| Categoria | E | 3.000.000 |
| Azienda | E | 1.000 |
| Magazzino | E | 500.000 |
| Spedizione | E | 1.000.000 |
| Effettuato | E | 500.000 |
| Non Effetuato | E | 500.000 |
| Autonoma | E | 500.000 |
| Servizio | E | 500.000 |
| Movimento | E | 3.000.000 |
| Dipendente | E | 50.000 |
| ContoSogg | R | 1.500.000 |
| ContMov | R | 3.000.000 |
| ordine | R | 1.500.000 |
| generato | R | 1.000.000 |
| possiede | R | 1.000.000 |
| compra | R | 1.000.000 |
| tipo | R | 3.000.000 |
| consegna | R | 1.000.000 |
| prelevazione | R | 500.000 |
| CitMag | R | 500.000 |
| CitNaz | R | 100 |
| fornitore | R | 500.000 |
| impiegato corrente | R | 40.000 |
| direzione | R | 1.000 |
| impiegato passato | R | 9.000 |
| locazAz | R | 1.000 |
| appartiene | R | 1.000.000 |
| abita | R | 1.500.000 |
| proprietario | R | 500.000 |
| fornito | R | 500.000 |

**Tabella delle operazioni**

|  |  |  |  |
| --- | --- | --- | --- |
| **OPERAZIONI** | **DESCRIZIONE** | **FREQUENZA** | **TIPO** |
|  | Visualizza il numero di spedizioni per ogni città; | 1/anno | B |
|  | Visualizza annunci che non sono stati venduti; | 1/anno | B |
|  | Visualizzare in quale città avvengono più spedizioni | 1/anno | B |
|  | Visualizzare quali dipendenti ancora non hanno consegnato le loro spedizioni | 1/giorno | B |
|  | Calcola la media dei feedback ottenuti da un utente; | 10/giorno | B |
|  | Visualizzare qual è la categoria dominante; | 1/mese | B |
|  | Visualizzare i dipendenti che lavorano in un’ azienda situata nella città in cui sono stati venduti più annunci; | 1/mese | B |
|  | Visualizzare le città in cui avvengono le spedizioni dove non risiedono dei magazzini; | 1/mese | B |
|  | Utenti che hanno acquistato annunci da soggetti concittadini; (stessa città); | 10/mese | B |
|  | Inserimento nuovo utente | 100/giorno | I |
|  | Inserimento direttore mgazzino | 1/mese | I |
|  | Acquisto annuncio | 100/giorno | I |
|  | Inserisci feeback | 1/giorno | I |
|  | Visualizza Annunci disponibili | 3/giorno | B |

**Ristrutturazione schema concettuale**

**Eliminazione attributo/i multivalore**

E’ stato eliminato l’attributo multivalore “tel” di “Soggetto” ed è diventata una nuova entità “Telefono”:

Immagine che contiene schermo, testo, lavagnabianca, orologio

Descrizione generata automaticamente

E’ stato eliminato l’attributo multivalore “foto” di “Annuncio” ed è diventata una nuova entità “Foto”:

Immagine che contiene testo, mappa

Descrizione generata automaticamente

**Eliminazione attributo/i composti**

E’ stato eliminato l’attributo composto “indirizzo” di tutte quelle entità che lo implementavano e sono stati aggiunti gli attributi “via”,”CAP”,”civico” alle entità corrispondenti:

Immagine che contiene bianco

Descrizione generata automaticamente

Per questo motivo sono state collegate tutte le entità che avevano come attributo “indirizzo” ad un’unica entità “indirizzo”;

**Eliminazione di ISA tra entità**

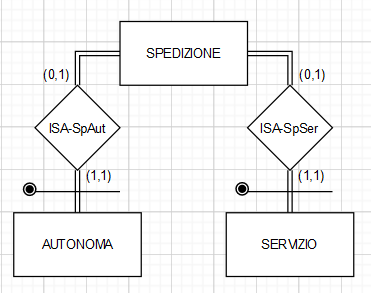
Sono state eliminate le generalizzazioni complete, aggiungendo relazioni ad esse, legate a loro volta da nuovi vincoli:

**Entità Spedizione**

Immagine che contiene testo, mappa

Descrizione generata automaticamente

**V.SpedizioneISAEffNonEff**: ogni istanza di Spedizione deve partecipare o alla relazione ISA-SpEff o alla relazione ISA-SpN ma non ad entrambe;



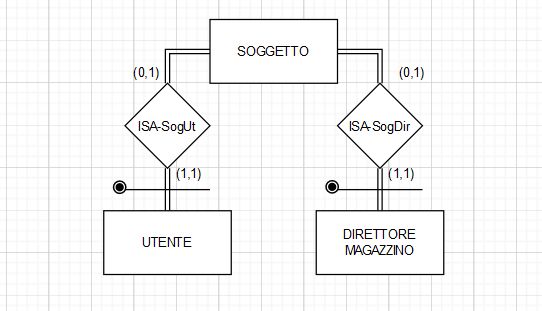
**V.SpedizioneISAEffNonEff**: ogni istanza di Spedizione deve partecipare o alla relazione ISA-SpAut o alla relazione ISA-SpSer ma non ad entrambe;

**Entità Annuncio**

**Immagine che contiene testo, mappa

Descrizione generata automaticamente**

**Entità Soggetto**

****

**V.SoggettoISASogUtISASogDir**: ogni istanza di Soggetto deve partecipare o alla relazione ISA-SogUt o alla relazione ISA-SogDir ma non ad entrambe;

Sono state eliminate le ISA tra relazioni, aggiungendo opportuni vincoli esterni e opportune relazioni/entità:

**Relazione Direzione**

E’ stata aggiunta l’entità DIRETTORE che estende DIPENDENTE e quindi è stata aggiunta anche la relazione “ISA-Dip”:

**Immagine che contiene testo, mappa

Descrizione generata automaticamente**

**V.Direttore**: Per ogni AZIENDA deve esistere uno ed un solo DIPENDENTE che è anche un DIRETTORE e che è anche un impiegato corrente;

**Scelta degli identificatori**

Sono stati scelti gli identificatori principali per ogni entità e per quelle per cui non è stato possibile sceglierne uno, sono stati aggiunto degli “id”. Tali entità sono Conto, Feedback, Magazzino e tutte le entità Indirizzo.

**Diagramma E/R: schema concettuale ristrutturato**

In base alle decisioni fatte durante la ristrutturazione è stato prodotto il seguente schema concettuale ristrutturato:

Immagine che contiene testo

Descrizione generata automaticamente

**Dizionario dei dati ristrutturato**

|  |  |  |  |
| --- | --- | --- | --- |
| **ENTITA’** | **DESCRIZIONE** | **ATTRIBUTI** | **IDENTIFICATORE** |
| Soggetto | Un Soggetto deve assumere uno dei due valori: Utente, Direttore di magazzino | nome, cognome, data\_nascita,  e-mail, CF, indirizzo | CF |
| Utente | Un utente che pubblica/compra/vende annunci | media\_feedback, nickname\* | Soggetto + Utente (identificatore esterno) |
| Città | La città di una nazione dove si trovano gli utenti, i magazzini, le aziende | nome | nome (identificatore esterno) |
| Nazione | Una nazione nel mondo | nome | nome |
| Feedback | Un feedback dato da un utente | voto,id | id |
| Conto | Un conto di un utente o di un magazzino | saldo | Saldo + Soggetto (identificatore esterno) |
| Foto | Le foto di un annuncio che è stato pubblicato | percorso |  |
| Annuncio | Un annuncio che viene pubblicato/ venduto/ acquistato da un Soggetto | codice, DataOra, descrizione, prezzo, periodo\_usato | codice |
| Annuncio Venduto | Un annuncio che viene venduto ad un Soggetto | DataOra | Annuncio Venduto + Annuncio (identificatore esterno) |
| Annuncio Acquistato | Un annuncio che viene acquistato da un Utente | DataOra | Annuncio Acquistato + Annuncio (identificatore esterno) |
| Categoria | Una categoria di un annuncio | nome | nome |
| Azienda | Un’azienda che rifornisce un magazzino | indirizzo, nome, ragione\_sociale, id | id |
| Magazzino | Un magazzino dove all’interno ci sono i vestiti e quindi gli annunci; | indirizzo, nome, ragione\_sociale, id | id |
| Spedizione | Una spedizione di un annuncio | codice, data | codice + Utente (identificatore esterno) |
| Effettuata | Una spedizione di annuncio venduto che è stata effettuata | Data | Spedizione + Utente (identificatore esterno) |
| Non Effettuata | Una spedizione di annuncio venduto ancora in corso | Data | Spedizione + Utente (identificatore esterno) |
| Autonoma | Una spedizione che viene gestita tra gli utenti; |  | Autonoma + Spedizione (identificatore esterno) |
| Servizio | Una spedizione che viene gestita da un magazzino |  | Servizio + Spedizione (identificatore esterno) |
| Movimento | Un prelievo/versamento in un conto | importo, DataOra | DataOra + Conto (identificatore esterno) |
| Dipendente | Un dipendente di un’azienda | nome, cognome, Data\_nascita, CF | CF |
| Direttore Magazzino | Il direttore di un magazzino | licenza | Direttore Magazzino + Soggetto (identificatore esterno) |
| Indirizzo | L’indirizzo di un Soggetto | Via, Cap, Civico, id | Via,CAP,Civico |
| Telefono | Il numero di telefono di un Soggetto | num | num |
| Direttore | Il direttore di un’AZIENDA |  | Direttore + Dipendente (identificatore esterno) |
| ISA-SpEff | Associa una Spedizione a Effettuata: rappresenta quando una spedizione è stata effettuata |  |  |
| ISA-SpN | Associa una Spedizione a Non Effettuata: rappresenta quando una spedizione non è stata effettuata |  |  |
| ISA-SpSer | Associa una Spedizione a Servizio: rappresenta la modalità della spedizione |  |  |
| ISA-Dip | Associa un Direttore ad un Dipendente: un Dipendente può essere anche un Direttore |  |  |
| ISA-SpAut | Associa una Spedizione a Autonoma: rappresenta la modalità della spedizione |  |  |
| ISA-AnnAc | Associa un Annuncio ad un Annuncio Acquistato: rappresenta quando un Annuncio è stato Acquistatob |  |  |
| IndSog | Associa un Soggetto ad un Indirizzo: rappresenta l’indirizzo di un soggetto |  |  |
| ISA-AnnVen | Associa un Annuncio ad Annuncio Venduto: rappresenta quando un annuncio che in precedenza è stato pubblicato, poi viene venduto |  |  |
| ISA-SogUt | Associa un Soggetto ad un Utente: rappresenta quando un Soggetto è un Utente e quindi non un Direttore di Magazzino |  |  |
| ISA-SogDir | Associa un Soggetto ad un Direttore Magazzino: rappresenta quando un Soggetto è un Direttore Magazzino e quindi non è un utente |  |  |
| compra | Associa un Annuncio Acquistato ad un Soggetto: rappresenta gli acquisti effettuati da un Soggetto |  |  |
| Ind\_Az | Associa un’Azienda ad un Indirizzo: rappresenta l’indirizzo di un’azienda |  |  |
| Ind\_Mag | Associa un Magazzino ad un Indirizzo: rappresenta l’indirizzo di un magazzino |  |  |
| ContoSogg | Associa il Conto ad un Soggetto | Soggetto,Conto |  |
| ContMov | Associa un conto a dei movimenti | Conto,Movimento |  |
| ordine | Associa un Soggetto ad un Annuncio:  rappresenta quale Soggetto ha pubblicato l’Annuncio e a quale Utente è stato nel caso venduto e qual è il Soggetto che l’ha acquistato | Soggetto,Annuncio |  |
| generato | Associa un Utente ad uno o più Feedback: rappresenta quale Utente ha pubblicato il Feeback | Utente,Feedback |  |
| possiede | Associa un Feedback ad un Annuncio: rappresenta a quale Annuncio Acquistato appartiene il Feedback | Feedback,Annuncio Acquistato |  |
| tipo | Associa un Annuncio ad una Categoria | Annuncio, Categoria |  |
| HaFoto | Associa un Annuncio ad una o più sue foto | percorso |  |
| consegna | Associa un Annuncio Venduto ad una Spedizione: rappresenta quale annuncio verrà spedito | Annuncio Venduto, Spedizione |  |
| prelevazione | Associa una Spedizione ad un Magazzino: rappresenta dove la spedizione deve andare a prelevare l’ordine nel caso tale ordine sia stato venduto da un magazzino | Servizio, Magazzino, DataOra |  |
| HaTelefono | Associa un Soggetto ad un Telefono: rappresenta i/il numero/i di telefono di un Soggetto | num |  |
| proprietario | Associa un Direttore di magazzino ad un Magazzino: rappresenta il proprietario di un magazzino | Direttore di magazzino, Magazzino |  |
| CitMag | Associa un Magazzino ad una Città: rappresenta la città dove risiede un magazzino | Magazzino, Città |  |
| abita | Associa un Utente ad una Città: rappresenta in quale città un utente risiede | Soggetto,Città |  |
| CitNaz | Associa una Città ad una Nazione: rappresenta in quale nazione una città fa parte | Nazione, Città |  |
| fornitore | Associa un Magazzino ad un’Azienda: rappresenta quale azienda rifornisce un magazzino | Magazzino, Azienda |  |
| impiegato corrente | Associa un Dipendente ad un’Azienda: rappresenta l’impiegato che attualmente lavora in un’azienda | Azienda, Dipendente, inizio |  |
| impiegato passato | Associa un Dipendente ad un’Azienda: rappresenta quale dipendente lavorava in un’azienda | Azienda, Dipendente, fine |  |
| locazAz | Associa un’Azienda ad una Città: rappresenta la città dove risiede l’azienda | Azienda, Città |  |
| appartiene | Associa una Spedizione ad un Utente: rappresenta a quale utente deve essere effettuata la spedizione | Spedizione, Utente |  |
| fornito | Associa un Dipendente ad una Spedizione: rappresenta quale Dipendente ha effettuato la Spedizione | Servizio, Dipendente |  |

**Vincoli esterni Diagramma ristrutturato**

Quelli sottolineati sono i vincoli che sono stati aggiunti a seguito della ristrutturazione.

-**V.Indirizzo**: Un SOGGETTO, un MAGAZZINO e un’AZIENDA non possono avere lo stesso indirizzo di residenza:

* Per ogni istanza (S,Ind) di “IndSog” con S appartenente a SOGGETTO e Ind appartenente a INDIRIZZO e per ogni istanza (Az,Ind’) con Az appartenente a AZIENDA e Ind’ appartenente a INDIRIZZO e per ogni istanza (Mag,Ind’’) con Mag appartenente a MAGAZZINO e Ind’’ appartenente a INDIRIZZO deve valere la regola: Ind != Ind’ != Ind’’;

-**V.Feedback**: Se esiste un Feedback da parte di un utente deve per forza appartenere ad un annuncio acquistato:

* Se esiste F appartenente a FEEDBACK, allora deve esistere l’istanza (F,AC) di “possiede” dove AC appartiene a ANNUNCIO ACQUISTATO isa ANNUNCIO;

-**V.DataOra**: La data e l’ora di annuncio acquistato deve essere maggiore alla data e l’ora del medesimo annuncio appena pubblicato:

* Per ogni istanza (AN,AP) di “ISA-AnnAc” con AN appartenente a ANNUNCIO e AP appartenente a ANNUNCIO ACQUISTATO, sia D l’attributo “DataOra” di AN e D’ l’attributo “DataOra” di AC deve valere la regola: D < D’;

**-V.SpedizioneData**: La data di una Spedizione effettuata non può essere minore della data della sua Spedizione non effettuata:

* Per ogni istanza (S,E) di “ISA-SpEff” con S appartenente a SPEDIZIONE e E appartenente a EFFETTUATA e per ogni istanza (S,N) con N appartenente a NON EFFETTUATA, sia D l’attributo data di N e D’ l’attributo data di E, deve valere la regola D <= D’;

**-V.SpedizioneISAEffNonEff**: ogni istanza di Spedizione deve partecipare o alla relazione ISA-SpEff o alla relazione ISA-SpN ma non ad entrambe:

* Per ogni SP appartenente a SPEDIZIONE, SP deve fare parte dell’istanza (SP,EFF) di “ISA-SpEff” con EFF appartenente a EFFETTUATA, oppure deve fare parte dell’istanza (SP,NEFF) di “ISA-SpN” con NEFF appartenente a NON EFFETTUATA, ma non può farne parte ad entrambe;

**-V.SpedizioneISASpSerISASpAut**: ogni istanza di Spedizione deve partecipare o alla relazione ISA-SpAut o alla relazione ISA-SpSer ma non ad entrambe:

* Per ogni SP appartenente a SPEDIZIONE, SP deve fare parte dell’istanza (SP,EFF) di “ISA-SpSer” con EFF appartenente a SERVIZIO, oppure deve fare parte dell’istanza (SP,NEFF) di “ISA-SpAut” con NEFF appartenente a AUTONOMA, ma non può farne parte ad entrambe;

**-V.SoggettoISASogUtISASogDir**: ogni istanza di Soggetto deve partecipare o alla relazione ISA-SogUt o alla relazione ISA-SogDir ma non ad entrambe:

* Per ogni SP appartenente a SOGGETTO, SP deve fare parte dell’istanza (SP,EFF) di “ISA-SogUt” con EFF appartenente a UTENTE, oppure deve fare parte dell’istanza (SP,NEFF) di “ISA-SogDir” con NEFF appartenente a DIRETTORE MAGAZZINO, ma non può farne parte ad entrambe;

**-V.Direttore**: Per ogni AZIENDA deve esistere uno ed un solo DIPENDENTE che è anche un DIRETTORE e che è un impiegato corrente:

* Per ogni istanza (Az,Dip) di “impiegato corrente” con Az appartenente a AZIENDA e Dip appartenente a DIPENDENTE, deve esistere l’istanza (Dip,Dir) di “ISA-Dip” con Dip appartenente a DIRETTORE il cui numero di tale istanza deve essere >0 e <2;

-**V.CompraVendiUtente**: Un utente non può comprare un annuncio che egli stesso ha pubblicato:

* Per ogni (U,AP) di “ordine” con U appartenente a UTENTE isa SOGGETTO e AP appartenente a ANNUNCIO, non deve esistere un’istanza (U,AC) con AC appartenente a ANNUNCIO ACQUISTATO dove AC=AP;

-**V.DirettoreMagazzino**: Un direttore di un magazzino non può acquistare annunci;

* Per ogni istanza D di DIRETTORE MAGAZZINO isa SOGGETTO non deve esistere l’istanza (D,AC) di “ordine” con AC appartenente a ANNUNCIO ACQUISTATO isa ANNUNCIO;

-**V.DipendenteAzienda**: Se un dipendente non lavora più nell’azienda dove lavorava, la data e l’ora di fine deve essere strettamente maggiore della data e l’ora dell’inizio:

* Per ogni istanza (D,A) di “impiegato corrente” con D appartenente a DIPENDENTE e A appartenente a AZIENDA, sia i l’attributo inizio di “impiegato corrente”, se (D,A) è anche un’istanza di “impiegato passato”, sia j l’attributo fine di “impiegato passato”, deve valere che j>i;

-**V.SpedizioneServizio**: Se un direttore di magazzino ha pubblicato un annuncio allora la sua eventuale spedizione deve essere per forza di servizio:

* Per ogni istanza (D,AV) di “ordine” con D appartenente a DIRETTORE MAGAZZINO isa SOGGETTO e AV appartenente a ANNUNCIO, se AV appartiene a ANNUNCIO VENDUTO isa ANNUNCIO, sia (AV,SP) con SP appartenente a SPEDIZIONE, SP deve per forza assumere il valore SER appartenente a SERVIZIO e non a AUTONOMA;

-**V.SpedizioneAutonoma**: Se un cliente ha pubblicato un annuncio allora la sua eventuale spedizione deve essere per forza autonoma:

* Per ogni istanza (U,AV) di “ordine” con U appartenente a UTENTE isa SOGGETTO e AV appartenente a ANNUNCIO, se AV appartiene a ANNUNCIO VENDUTO isa ANNUNCIO, sia (AV,SP) con SP appartenente a SPEDIZIONE, SP deve per forza assumere il valore AUT appartenente a AUTONOMA e non a SERVIZIO;

**Tabella dei volumi ristrutturati**

|  |  |  |
| --- | --- | --- |
| **CONCETTO** | **TIPO** | **VOLUME** |
| Soggetto | E | 1.500.000 |
| Utente | E | 1.000.000 |
| Direttore Magazzino | E | 500.000 |
| Città | E | 100 |
| Nazione | E | 30 |
| Feedback | E | 1.000.000 |
| Conto | E | 1.500.000 |
| Annuncio | E | 3.000.000 |
| Telefono | E | 1.500.000 |
| Annuncio Venduto | E | 1.000.000 |
| Annuncio Acquistato | E | 1.000.000 |
| Categoria | E | 3.000.000 |
| Azienda | E | 1.000 |
| Indirizzo | E | 2.100.000 |
| Magazzino | E | 500.000 |
| Spedizione | E | 1.000.000 |
| Effettuato | E | 500.000 |
| Non Effetuato | E | 500.000 |
| Autonoma | E | 500.000 |
| Servizio | E | 500.000 |
| Movimento | E | 3.000.000 |
| Dipendente | E | 50.000 |
| Direttore | E | 1.000 |
| ContoSogg | R | 1.500.000 |
| ContMov | R | 3.000.000 |
| ordine | R | 1.500.000 |
| generato | R | 1.000.000 |
| possiede | R | 1.000.000 |
| tipo | R | 3.000.000 |
| consegna | R | 1.000.000 |
| prelevazione | R | 500.000 |
| CitMag | R | 500.000 |
| CitNaz | R | 100 |
| fornitore | R | 500.000 |
| impiegato corrente | R | 40.000 |
| impiegato passato | R | 9.000 |
| locazAz | R | 1.000 |
| appartiene | R | 1.000.000 |
| abita | R | 1.500.000 |
| HaTelefono | R | 1.500.000 |
| proprietario | R | 500.000 |
| IndSog | R | 1.500.000 |
| Ind\_Az | R | 1.000 |
| Ind\_Mag | R | 500.000 |
| ISA-SpEff | R | 500.000 |
| ISA-SpN | R | 500.000 |
| ISA-Dip | R | 1.000 |
| ISA-SpAut | R | 500.000 |
| ISA-AnnAc | R | 1.000.000 |
| ISA-SpSer | R | 500.000 |
| ISA-SogUt | R | 1.000.000 |
| ISA-SogDir | R | 500.000 |
| ISA-AnnVen | R | 1.000.000 |
| fornito | R | 500.000 |

**Tabelle di accesso**

Sono state calcolate solo le funzioni più complicate;

**Op.1: visualizza il numero di spedizioni per ogni città;**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONCETTO** | **COSTRUTTO** | **ACCESSI** | **TIPO** |
| Spedizione | Entità | 1000 | L |
| Città | Entità | 10 | L |
| appartiene | Relazione | 1000 | L |
| Utente | Entità | 1000 | L |
| Soggetto | Entità | 1000 | L |
| ISA-SogUt | Relazione | 1000 | L |
| abita | Relazione | 1000 | L |

**Op.2: Visualizza annunci che non sono stati venduti;**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONCETTO** | **COSTRUTTO** | **ACCESSI** | **TIPO** |
| Annuncio | Entità | 100.000 | L |
| AnnuncioVenduto | Entità | 10.000 | L |
| Isa-AnnVen | Entità | 100.000 | L |

**Op.3: visualizzare in quale città avvengono più spedizioni;**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONCETTO** | **COSTRUTTO** | **ACCESSI** | **TIPO** |
| Spedizione | Entità | 10.000 | L |
| Città | Entità | 10 | L |
| appartiene | Relazione | 1000 | L |
| Utente | Entità | 1000 | L |
| ISA-SogUt | Relazione | 1000 | L |
| Soggetto | Entità | 1000 | L |
| abita | Relazione | 100 | L |
| Città | Entità | 10 | L |

**Op.4: visualizzare quali Dipendenti ancora non hanno consegnato le loro Spedizioni;**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONCETTO** | **COSTRUTTO** | **ACCESSI** | **TIPO** |
| Dipendente | Entità | 100 | L |
| Servizio | Entità | 100 | L |
| Spedizione | Entità | 50 | L |
| Effettuata | Entità | 50 | L |
| fornito | Relazione | 100 | L |
| ISA-SpSer | Relazione | 100 | L |
| ISA-SpEff | Relazione | 50 | L |

**Op.5: calcola la media dei feedback ottenuti da un utente;**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONCETTO** | **COSTRUTTO** | **ACCESSI** | **TIPO** |
| Annuncio Venduto | Entità | 100.000 | L |
| possiede | Relazione | 50.000 | L |
| Feedback | Entità | 50.000 | L |
| ISA-AnnAc | Relazione | 100.000 | L |
| Annuncio | Entità | 100.000 | L |
| ordine | Relazione | 100.000 | L |
| Soggetto | Entità | 100.000 | L |

**Op.6: visualizzare quali sono le categorie dominanti;**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONCETTO** | **COSTRUTTO** | **ACCESSI** | **TIPO** |
| Annuncio | Entità | 100 | L |
| Tipo | Relazione | 100 | L |
| Categoria | Entità | 1 | L |

**Op.7: Visualizzare i dipendenti che lavorano in un’ azienda situata nella città in cui sono stati venduti più annunci;**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONCETTO** | **COSTRUTTO** | **ACCESSI** | **TIPO** |
| Dipendente | Entità | 100 | L |
| impiegato corrente | Relazione | 100 | L |
| Azienda | Entità | 10 | L |
| Città | Entità | 10 | L |
| LocazAz | Relazione | 10 | L |
| Soggetto | Entità | 1000 | L |
| ordine | Relazione | 10.000 | L |
| Annuncio | Entità | 10.000 | L |
| Annuncio Venduto | Entità | 5.000 | L |
| ISA-AnnVen | Entità | 5.000 | L |

**Op.8: Visualizzare le città in cui avvengono le spedizioni dove non risiedono dei Magazzini;**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONCETTO** | **COSTRUTTO** | **ACCESSI** | **TIPO** |
| Magazzino | Entità | 100 | L |
| Spedizione | Entità | 10.000 | L |
| Città | Entità | 10 | L |
| Utente | Entità | 100.000 | L |
| Soggetto | Entità | 1.000.000 | L |
| CitMag | Relazione | 100 | L |
| appartiene | Relazione | 10.000 | L |
| Isa-SogUt | Relazione | 100.000 | L |

**Op.9: Utenti che hanno acquistato annunci da soggetti concittadini; (stessa città);**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONCETTO** | **COSTRUTTO** | **ACCESSI** | **TIPO** |
| Annuncio | Entità | 100.000 | L |
| AnnuncioAcquistato | Entità | 25.000 | L |
| AnnuncioVenduto | Entità | 25.000 | L |
| Isa-AnnAc | Relazione | 50.000 | L |
| ISA-AnnVen | Relazione | 50.000 | L |
| ordine | Relazione | 25.000 | L |
| Città | Entità | 10 | L |
| Utente | Entità | 100.000 | L |
| abita | Relazione | 100.000 | L |
| Soggetto | Entità | 1.000.000 | L |

**Traduzione diretta al modello relazionale**

**SPEDIZIONE**(codice, Data, Utente) è stata accorpata la relazione “appartiene”:

* **foreign key**: SPEDIZIONE[Utente] ⊆ UTENTE[Soggetto]
* **inclusione**: SPEDIZIONE[codice] ⊆ ANNUNCIO VENDUTO[Spedizione]

**AUTONOMA**(Spedizione) è stata accorpata la relazione “ISA-SpAut”:

* **foreign key**: AUTONOMA[Spedizione] ⊆ SPEDIZIONE[codice]

**SERVIZIO**(Spedizione, Dipendente, Magazzino) sono state accorpate “ISA-SpSer”, “prelevazione”, “fornito”:

* **foreign key**: SERVIZIO[Spedizione] ⊆ SPEDIZIONE[codice]
* **foreign key**: SERVIZIO[Dipendente] ⊆ DIPENDENTE[CF]
* **foreign key**: SERVIZIO[Magazzino] ⊆ MAGAZZINO[id]

**EFFETTUATA**(Spedizione, Data) è stata accorpata la relazione “ISA-SpEff”:

* **foreign key**: EFFETTUATA[Spedizione] ⊆ SPEDIZIONE[codice]

**NON EFFETTUATA**(Spedizione, Data) è stata accorpata la relazione “ISA-SpN”:

* **foreign key**: NON EFFETTUATA[Spedizione] ⊆ SPEDIZIONE[codice]

**UTENTE**(nickname\*,media\_feedback, Soggetto) è stata accorpata la relazione “ISA-SogUt”:

* **foreign key**: UTENTE[Soggetto] ⊆ SOGGETTO[CF]

**media\_feedbakc:** deve essere calcolata in base ai feedback ottenuti;

**SOGGETTO**(CF, DataNascita, email, nome, cognome, Conto, Via, civico, CAP, Città) sono state accorpate le relazione “ContoSogg”, “IndSog”, “abita”:

* **foreign key**: SOGGETTO[Conto] ⊆ CONTO[id]
* **foreign key**: SOGGETTO[Via,CAP,Civico] ⊆ INDIRIZZO[Via,CAP,Civico]
* **foreign key**: SOGGETTO[Città] ⊆ CITTA[nome]
* **inclusione**: SOGGETTO[CF] ⊆ TELEFONO[Soggetto]

**TELEFONO**(num, Soggetto) è stata accorpata la relazione “HaTelefono”:

* **foreign key**: TELEFONO[Soggetto] ⊆ SOGGETTO[CF]

**CONTO**(id, saldo, Soggetto):

* **inclusione**: CONTO[id] ⊆ SOGGETTO[Conto]
* **foreign key**: CONTO[Soggetto] ⊆ SOGGETTO[CF]

**MOVIMENTO**(importo, DataOra, Conto) è stata accorpata la relazione “ContMov”:

* **foreign key**: MOVIMENTO[Conto] ⊆ CONTO[id]

**DIRETTORE MAGAZZINO**(licenza, Soggetto) è stata accorpata la relazione “ISA-SogDir”:

* **foreign key**: DIRETTORE[Soggetto] ⊆ SOGGETTO[CF]
* **inclusione**: DIRETTORE[Soggetto] ⊆ MAGAZZINO[DirettoreMagazzino]

**INDIRIZZO**(Via, CAP, civico)

**CITTA**(nome, Nazione) è stata accorpata la relazione “CitNaz”:

* **foreign key**: CITTA[Nazione] ⊆ NAZIONE[nome]

**NAZIONE**(nome):

* **inclusione**: NAZIONE[nome] ⊆ CITTA[Nazione]

**ANNUNCIO**(codice, DataOra, periodo\_usato, prezzo, descrizione, Soggetto, Categoria) sono state accorpate le relazioni “ordine”, “tipo”:

* **foreign key**: ANNUNCIO[Soggetto] ⊆ SOGGETTO[CF]
* **foreign key**: ANNUNCIO[Categoria] ⊆ CATEGORIA[nome]

**FOTO**(percorso, Annuncio) è stata accorpata la relazione “HaFoto”:

* **foreign key**: FOTO[Annuncio] ⊆ ANNUNCIO[codice]

**ANNUNCIO ACQUISTATO**(Annuncio, DataOra,Soggetto) è stata accorpata la relazione “ISA-AnnAc” e “compra”:

* **foreign key**: ANNUNCIO ACQUISTATO[Annuncio] ⊆ ANNUNCIO[codice]
* **foreign key**: ANNUNCIO ACQUISTATO[Soggetto] ⊆ SOGGETTO[CF]

**FEEDBACK**(id, voto, AnnuncioAcquistato, Utente) sono state accorpate le relazioni “generato” e “possiede”:

* **foreign key**: FEEDBACK[AnnuncioAcquisto] ⊆ ANNUNCIO ACQUISTATO[Annuncio]
* **foreign key**: FEEDBACK[Utente] ⊆ UTENTE[Soggetto]

**ANNUNCIO VENDUTO**(Annuncio, Spedizione, DataOra) sono state accorpate le relazioni “ISA-AnnVen” e “consegna”:

* **foreign key**: ANNUNCIO VENDUTO[Annuncio] ⊆ ANNUNCIO[codice]
* **foreign key**: ANNUNCIO VENDUTO[Spedizione] ⊆ SPEDIZIONE[codice]

**CATEGORIA**(nome)

**AZIENDA**(ragione\_sociale, id, nome, Città, Via, Civico, CAP) sono state accorpate le relazioni “Ind\_Az”, “locaz\_Az”:

* **foreign key**: AZIENDA[Via,CAP,Civico] ⊆ INDIRIZZO [Via,CAP,civico]
* **foreign key**: AZIENDA[Città] ⊆ CITTA[nome]
* **inclusione**: AZIENDA[id] ⊆ impiegato corrente[Azienda]
* **inclusione**: AZIENDA[id] ⊆ fornitore[Azienda]

**DIPENDENTE**(nome, cognome, DataNascita, CF)

**MAGAZZINO**(id, ragione\_sociale, Via, Civico, CAP, DirettoreMagazzino, Città) sono state accorpate le relazioni “proprietario”, “Ind\_Mag”, “CitMag”:

* **foreign key**: MAGAZZINO[DirettoreMagazzino] ⊆ DIRETTORE MAGAZZINO[Soggetto]
* **foreign key**: MAGAZZINO[Via,CAP,Civico] ⊆ INDIRIZZO [Via,CAP,Civico]
* **foreign key**: MAGAZZINO[Città] ⊆ CITTA[nome]
* **inclusione**: MAGAZZINO[id] ⊆ fornitore[Magazzino]

**impiegato passato**(fine, Azienda, Dipendente):

* **foreign key**: impiegato passato[Azienda] ⊆ AZIENDA[id]
* **foreign key**: impiegato passato[Dipendente] ⊆ DIPENDENTE[CF]

**impiegato corrente**(inizio, Azienda, Dipendente):

* **foreign key**: impiegato corrente[Azienda] ⊆ AZIENDA[id]
* **foreign key**: impiegato corrente[Dipendente] ⊆ DIPENDENTE[CF]

**fornitore**(Azienda, Magazzino):

* **foreign key**: fornitore[Azienda] ⊆ AZIENDA[id]
* **foreign key**: fornitore[Magazzino] ⊆ MAGAZZINO[id]

**DIRETTORE**(Dipendente):

* **foreign key**: DIRETTORE[Dipendente] ⊆ DIPENDENTE[CF]

**Creazione tabelle: Linguaggio SQL**

create table SPEDIZIONE(

codice int not null auto\_increment primary key,

Data date not null,

Utente varchar(15) not null,

AnnuncioVenduto int not null,

foreign key (Utente)

references UTENTE(Soggetto)

on delete cascade

on update cascade,

foreign key(AnnuncioVenduto)

references annuncioacquistato(Annuncio)

on delete cascade

on update cascade,

);

create table CONTO(

id int not null auto\_increment primary key,

saldo int not null,

Soggetto varchar(15) not null,

foreign key(Soggetto)

references SOGGETTO(CF)

on delete cascade

on update cascade

);

create table NAZIONE(

nome varchar(50) not null primary key

);

create table CITTA(

nome varchar(50) not null primary key,

Nazione varchar(50) not null,

foreign key (Nazione)

references NAZIONE(nome)

on delete cascade

on update cascade

);

create table SOGGETTO(

CF varchar(15) not null primary key,

DataNascita date not null,

email varchar(50) not null,

nome varchar(50) not null,

cognome varchar(50) not null,

Via varchar(50) not null,

civico int not null,

CAP int(5) not null,

Città varchar(50) not null,

foreign key (Via,civico,CAP)

references INDIRIZZO (Via,civico,CAP)

on delete cascade

on update cascade,

foreign key (Città)

references CITTA(nome)

on delete cascade

on update cascade

);

create table UTENTE(

nickname varchar(50),

media\_feedback enum('0','1','2','3',4','5','6','7','8','9','10') not null default '0',

Soggetto varchar(15) not null,

foreign key (Soggetto)

references SOGGETTO(CF)

on delete cascade

on update cascade

);

create table AUTONOMA(

Spedizione int not null,

foreign key (Spedizione)

references SPEDIZIONE(codice)

on delete cascade

on update cascade

);

create table SERVIZIO(

Spedizione int not null,

Dipendente varchar(15) not null,

Magazzino int not null,

foreign key (Spedizione)

references SPEDIZIONE(codice)

on delete cascade

on update cascade,

foreign key (Dipendente)

references dipendente(CF)

on delete cascade

on update cascade,

foreign key (Magazzino)

references magazzino(id)

on delete cascade

on update cascade

);

create table EFFETTUATA(

Data date not null,

Spedizione int not null,

foreign key (Spedizione)

references SPEDIZIONE(codice)

on delete cascade

on update cascade

);

create table NONEFFETTUATA(

Data date not null,

Spedizione int not null,

foreign key (Spedizione)

references SPEDIZIONE(codice)

on delete cascade

on update cascade

);

create table TELEFONO(

num varchar(10) not null primary key,

Soggetto varchar(15) not null,

foreign key (Soggetto)

references SOGGETTO(CF)

on delete cascade

on update cascade

);

create table MOVIMENTO(

importo bigint not null default 0,

DataOra timestamp not null primary key,

Conto int not null,

foreign key (Conto)

references CONTO(id)

on delete cascade

on update cascade

);

create table DIRETTOREMAGAZZINO(

licenza boolean not null,

Soggetto varchar(15) not null,

foreign key (Soggetto)

references SOGGETTO(CF)

on delete cascade

on update cascade

);

create table ANNUNCIO(

codice int auto\_increment not null primary key,

DataOra timestamp not null,

periodo\_usato enum('meno di 1 mese','meno di 6 mesi','meno di 1 anno','più di 1 anno','nuovo') not null default 'nuovo',

prezzo int not null,

descrizione varchar(100) not null,

Soggetto varchar(15) not null,

Categoria varchar(15) not null,

foreign key (Soggetto)

references SOGGETTO(CF)

on delete cascade

on update cascade,

foreign key (Categoria)

references CATEGORIA(nome)

on delete cascade

on update cascade

);

create table CATEGORIA(

nome varchar(15) not null primary key

);

create table FOTO(

percorso varchar(50) not null primary key,

Annuncio int not null,

foreign key (Annuncio)

references ANNUNCIO (codice)

on delete cascade

on update cascade

);

create table ANNUNCIOACQUISTATO(

DataOra timestamp not null,

Annuncio int not null,

Soggetto varchar(15) not null,

foreign key (Annuncio)

references ANNUNCIO (codice)

on delete cascade

on update cascade,

foreign key (Soggetto)

references Soggetto (CF)

on delete cascade

on update cascade

);

create table ANNUNCIOVENDUTO(

DataOra timestamp not null,

Annuncio int not null,

Utente varchar(15) not null,

foreign key (Annuncio)

references ANNUNCIO(codice)

on delete cascade

on update cascade,

foreign key (Utente)

references ANNUNCIO(Soggetto)

on delete cascade

on update cascade

);

create table FEEDBACK(

id int auto\_increment not null primary key,

voto enum('0','1','2','3',4','5','6','7','8','9','10') not null default '0',

AnnuncioAcquistato int not null,

Utente varchar(15) not null,

foreign key (AnnuncioAcquistato)

references AnnuncioAcquistato(Annuncio)

on delete cascade

on update cascade,

foreign key (Utente)

references UTENTE(Soggetto)

on delete cascade

on update cascade

);

create table DIPENDENTE(

nome varchar(15) not null,

cognome varchar(15) not null,

DataNascita date not null,

CF varchar(15) not null primary key

);

create table AZIENDA(

ragione\_sociale varchar(200) not null,

id int auto\_increment not null primary key,

nome varchar(50) not null,

Città varchar(50) not null,

Via varchar(50) not null,

civico int not null,

CAP int(5) not null,

foreign key (Via,civico,CAP)

references INDIRIZZO (Via,civico,CAP)

on delete cascade

on update cascade,

foreign key (Città)

references CITTA(nome)

on delete cascade

on update cascade

);

create table MAGAZZINO(

id int auto\_increment not null primary key,

ragione\_sociale varchar(200) not null,

Via varchar(50) not null,

civico int not null,

CAP int(5) not null,

DirettoreMagazzino varchar(15) not null,

Città varchar(50) not null,

foreign key (Via,civico,CAP)

references INDIRIZZO (Via,civico,CAP)

on delete cascade

on update cascade,

foreign key (Città)

references CITTA(nome)

on delete cascade

on update cascade,

foreign key (DirettoreMagazzino)

references DIRETTOREMAGAZZINO(Soggetto)

on delete cascade

on update cascade

);

create table DIRETTORE(

Dipendente varchar(15) not null,

foreign key (Dipendente)

references DIPENDENTE(CF)

on delete cascade

on update cascade

);

create table impiegatopassato(

fine timestamp not null,

Azienda int not null,

Dipendente varchar(15) not null,

foreign key (Azienda)

references AZIENDA(id)

on delete cascade

on update cascade,

foreign key (Dipendente)

references DIPENDENTE(CF)

on delete cascade

on update cascade

);

create table impiegatocorrente(

inizio timestamp not null,

Azienda int not null,

Dipendente varchar(15) not null,

foreign key (Azienda)

references AZIENDA(id)

on delete cascade

on update cascade,

foreign key (Dipendente)

references DIPENDENTE(CF)

on delete cascade

on update cascade

);

create table compra(

Annuncio int unique not null,

Soggetto varchar(15) unique not null,

foreign key (Annuncio)

references ANNUNCIOACQUISTATO(Annuncio)

on delete cascade

on update cascade,

foreign key (Soggetto)

references SOGGETTO(CF)

on delete cascade

on update cascade

);

create table fornitore(

Azienda int not null,

Magazzino int not null,

foreign key (Azienda)

references AZIENDA(id)

on delete cascade

on update cascade,

foreign key (Magazzino)

references MAGAZZINO(id)

on delete cascade

on update cascade

);

create table INDIRIZZO(

Via varchar(50) not null,

civico int not null,

CAP int(5) not null,

primary key(Via,civico,CAP)

);

**Vincoli: TRIGGER, FUNZIONI**

-**V.Indirizzo**: Un SOGGETTO, un MAGAZZINO e un’AZIENDA non possono avere lo stesso indirizzo di residenza:

* Per ogni istanza (S,Ind) di “IndSog” con S appartenente a SOGGETTO e Ind appartenente a INDIRIZZO e per ogni istanza (Az,Ind’) con Az appartenente a AZIENDA e Ind’ appartenente a INDIRIZZO e per ogni istanza (Mag,Ind’’) con Mag appartenente a MAGAZZINO e Ind’’ appartenente a INDIRIZZO deve valere la regola: Ind != Ind’ != Ind’’;
* **SQL:**
* **Funzione**
* delimiter $$
* create function ControlloIndirizzi(Via varchar(50), civico int, CAP int) returns int DETERMINISTIC begin
* select count(\*) into @var2 from magazzino
* where magazzino.via=Via and
* magazzino.civico=civico and
* magazzino.CAP=CAP;
* select count(\*) into @var3 from azienda
* where azienda.via=Via and
* azienda.civico=civico and
* azienda.CAP=CAP;
* if(@var2+@var3=0) then return 1;
* end if;
* return 0;
* end $$
* delimiter ;
* **Trigger**
* delimiter $$
* create trigger V\_indirizzo\_soggetto
* before insert on soggetto
* for each row
* begin
* if(ControlloIndirizzi(new.Via,new.civico,new.CAP)=0) then
* signal sqlstate '45000'
* set message\_text='Un SOGGETTO non può avere lo stesso indirizzo di un MAGAZZINO o un\' AZIENDA';
* end if;
* end $$
* delimiter ;
* delimiter $$
* create trigger V\_indirizzo\_magazzino
* before insert on magazzino
* for each row
* begin
* if(ControlloIndirizzi(new.Via,new.civico,new.CAP)=0) then
* signal sqlstate '45000'
* set message\_text='Un MAGAZZINO non può avere lo stesso indirizzo di un SOGGETTO o un\' AZIENDA';
* end if;
* end $$
* delimiter ;
* delimiter $$
* create trigger V\_indirizzo\_azienda
* before insert on azienda
* for each row
* begin
* if(ControlloIndirizzi(new.Via,new.civico,new.CAP)=0) then
* signal sqlstate '45000'
* set message\_text='Un\'AZIENDA non può avere lo stesso indirizzo di un SOGGETTO o un MAGAZZINO';
* end if;
* end $$
* delimiter ;

-**V.DataOra**: La data e l’ora di annuncio acquistato deve essere maggiore alla data e l’ora del medesimo annuncio appena pubblicato:

* Per ogni istanza (AN,AP) di “ISA-AnnAc” con AN appartenente a ANNUNCIO e AP appartenente a ANNUNCIO ACQUISTATO, sia D l’attributo “DataOra” di AN e D’ l’attributo “DataOra” di AC deve valere la regola: D < D’;
* **SQL**:
* **Funzione**
* /\* INIZIO: V.DataOra; \*/
* delimiter $$
* create function ControlloOra(asdf timestamp, codice int) returns int deterministic begin
* set @var1 = (select DataOra from annuncio
* where annuncio.codice=codice);
* if(@var1>asdf) then return 0;
* end if;
* return 1;
* end $$
* delimiter ;
* **Trigger**
* delimiter $$
* create trigger V\_DataOraAcquisto
* before insert on annuncioacquistato
* for each row
* begin
* if(ControlloOra(new.DataOra,new.Annuncio)=0) then
* signal sqlstate '45000'
* set message\_text='La DataOra dell\'annuncio acquistato deve essere maggiore della DataOra dell\'annuncio pubblicato';
* end if;
* end $$
* delimiter ;
* delimiter $$
* create trigger V\_DataOraVenduto
* before insert on annunciovenduto
* for each row
* begin
* if(ControlloOra(new.DataOra,new.Annuncio)=0) then
* signal sqlstate '45000'
* set message\_text='La DataOra dell\'annuncio venduto deve essere maggiore della DataOra dell\'annuncio pubblicato';
* end if;
* end $$
* delimiter ;
* /\* FINE: V.DataOra; \*/

**-V.SpedizioneData**: La data di una Spedizione effettuata non può essere minore della data della sua Spedizione non effettuata:

* Per ogni istanza (S,E) di “ISA-SpEff” con S appartenente a SPEDIZIONE e E appartenente a EFFETTUATA e per ogni istanza (S,N) con N appartenente a NON EFFETTUATA, sia D l’attributo data di N e D’ l’attributo data di E, deve valere la regola D <= D’;
* **SQL**
* **FUNZIONI**
* /\* INIZIO: Controllo della data \*/
* delimiter $$
* create function ControlloData(x date,Spedizione int) returns int deterministic begin
* select Data into @var1 from noneffettuata
* where noneffettuata.Spedizione=Spedizione;
* if(@var1>x) then return 0; end if;
* return 1;
* end $$
* delimiter ;
* /\* FINE: Controllo della data \*/
* **TRIGGER**
* /\* INIZIO: trigger per l'ISA disgiunta di SPEDIZIONE (EFFETTUATA) \*/
* delimiter $$
* create trigger V\_DataOra
* before insert on effettuata
* for each row
* begin
* if(ControlloData(new.Data,new.Spedizione)=0) then
* signal sqlstate '45000'
* set message\_text='La data deve essere maggiore o uguale alla Data della sua Spedizione non effettuata';
* end if;
* end $$
* delimiter ;
* /\* FINE: trigger per l'ISA disgiunta di SPEDIZIONE (EFFETTUATA) \*/

**(forse eliminare)-V.SpedizioneISAEffNonEff**: ogni istanza di Spedizione deve partecipare o alla relazione ISA-SpEff o alla relazione ISA-SpN ma non ad entrambe:

* Per ogni SP appartenente a SPEDIZIONE, SP deve fare parte dell’istanza (SP,EFF) di “ISA-SpEff” con EFF appartenente a EFFETTUATA, oppure deve fare parte dell’istanza (SP,NEFF) di “ISA-SpN” con NEFF appartenente a NON EFFETTUATA, ma non può farne parte ad entrambe;
* **FUNZIONI**
* /\* Si è pensato a livello applicazione che quando viene comprato un annuncio viene inserito direttamente in “non effettuato”;
* /\* INIZIO: funzione per controllare se il record che sto per inserire in noneffettuata non sia già in effettuata \*/
* delimiter $$
* create function ControlloISASpedizione(Spedizione int) returns int deterministic begin
* select count(\*) into @var1 from effettuata
* where effettuata.Spedizione=Spedizione;
* if(@var1>0) then return 0; end if;
* return 1;
* end $$
* delimiter ;
* /\* FINE: funzione per controllare se il record che sto per inserire in noneffettuata non sia già in effettuata \*/
* **TRIGGER**
* /\* INIZIO: trigger per l'ISA disgiunta di SPEDIZIONE (NONEFFETTUATA) \*/
* delimiter $$
* create trigger V\_SpedizioneISAEffNonEff\_2
* before insert on noneffettuata
* for each row
* begin
* if(ControlloISASpedizione(new.Spedizione)=0) then
* signal sqlstate '45000'
* set message\_text='Questa spedizione è stata già effettuata';
* end if;
* end $$
* delimiter ;
* /\* FINE: trigger per l'ISA disgiunta di SPEDIZIONE (NONEFFETTUATA) \*/

**-V.SpedizioneISASpSerISASpAut**: ogni istanza di Spedizione deve partecipare o alla relazione ISA-SpAut o alla relazione ISA-SpSer ma non ad entrambe:

* Per ogni SP appartenente a SPEDIZIONE, SP deve fare parte dell’istanza (SP,EFF) di “ISA-SpSer” con EFF appartenente a SERVIZIO, oppure deve fare parte dell’istanza (SP,NEFF) di “ISA-SpAut” con NEFF appartenente a AUTONOMA, ma non può farne parte ad entrambe;
* **SQL**
* **FUNZIONI**
* /\* INIZIO: controllo che solo un'istanza tra SERVIZIO/AUTONOMA ci può essere \*/
* delimiter $$
* create function ControlloDisgiunzioneServizio(x int) returns int deterministic begin
* select count(\*) into @var1 from servizio
* where servizio.Spedizione=x;
* if(@var1>0) then return 0; end if;
* return 1;
* end $$
* delimiter ;
* /\* FINE: controllo che solo un'istanza tra SERVIZIO/AUTONOMA ci può essere \*/
* /\* INIZIO: controllo che solo un'istanza tra SERVIZIO/AUTONOMA ci può essere \*/
* delimiter $$
* create function ControlloDisgiunzioneAutonoma(x int) returns int deterministic begin
* select count(\*) into @var1 from autonoma
* where autonoma.Spedizione=x;
* if(@var1>0) then return 0; end if;
* return 1;
* end $$
* delimiter ;
* **TRIGGER**
* /\* INIZIO: disgiunzione AUTONOMA da SERVIZIO \*/
* delimiter $$
* create trigger V\_SpedizioneISASpSerISASpAut\_1
* before insert on autonoma
* for each row
* begin
* if(ControlloDisgiunzioneServizio(new.Spedizione)=0) then
* signal sqlstate '45000'
* set message\_text='Questa spedizione è impostata su SERVIZIO';
* end if;
* end $$
* delimiter ;
* /\* FINE: disgiunzione AUTONOMA da SERVIZIO \*/
* /\* INIZIO: disgiunzione SERVIZIO da AUTONOMA \*/
* delimiter $$
* create trigger V\_SpedizioneISASpSerISASpAut\_2
* before insert on servizio
* for each row
* begin
* if(ControlloDisgiunzioneAutonoma(new.Spedizione)=0) then
* signal sqlstate '45000'
* set message\_text='Questa spedizione è impostata su AUTONOMA';
* end if;
* end $$
* delimiter ;
* /\* FINE: disgiunzione SERVIZIO da AUTONOMA \*/

**-V.SoggettoISASogUtISASogDir**: ogni istanza di Soggetto deve partecipare o alla relazione ISA-SogUt o alla relazione ISA-SogDir ma non ad entrambe:

* Per ogni SP appartenente a SOGGETTO, SP deve fare parte dell’istanza (SP,EFF) di “ISA-SogUt” con EFF appartenente a UTENTE, oppure deve fare parte dell’istanza (SP,NEFF) di “ISA-SogDir” con NEFF appartenente a DIRETTORE MAGAZZINO, ma non può farne parte ad entrambe;
* **SQL**
* **FUNZIONI**
* /\* INIZIO: funzione per il controllo disgiunzione UTENTE -> DIRETTORE MAGAZZINO \*/
* delimiter $$
* create function ControlloDisgiunzioneDirettoreMagazzino(x varchar(15)) returns int deterministic begin
* select count(\*) into @var1 from direttoremagazzino
* where direttoremagazzino.Soggetto=x;
* if(@var1>0) then return 0; end if;
* return 1;
* end $$
* delimiter ;
* /\* FINE: funzione per il controllo disgiunzione UTENTE -> DIRETTORE MAGAZZINO \*/
* /\* INIZIO: funzione per il controllo disgiunzione DIRETTORE MAGAZZINO -> UTENTE \*/
* delimiter $$
* create function ControlloDisgiunzioneUtente(x varchar(15)) returns int deterministic begin
* select count(\*) into @var1 from utente
* where utente.Soggetto=x;
* if(@var1>0) then return 0; end if;
* return 1;
* end $$
* delimiter ;
* /\* FINE: funzione per il controllo disgiunzione DIRETTORE MAGAZZINO -> UTENTE \*/
* **TRIGGER**
* /\* INIZIO: trigger per il controllo della disgiunzione UTENTE -> DIRETTORE MAGAZZINO \*/
* delimiter $$
* create trigger V\_SoggettoISASogUtISASogDir\_1
* before insert on utente
* for each row
* begin
* if(ControlloDisgiunzioneDirettoreMagazzino(new.Soggetto)=0) then
* signal sqlstate '45000'
* set message\_text='Questo UTENTE è già un DIRETTORE MAGAZZINO esistente';
* end if;
* end $$
* delimiter ;
* /\* FINE: trigger per il controllo della disgiunzione UTENTE -> DIRETTORE MAGAZZINO \*/
* /\* INIZIO: trigger per il controllo della disgiunzione DIRETTORE MAGAZZINO -> UTENTE \*/
* delimiter $$
* create trigger V\_SoggettoISASogUtISASogDir\_2
* before insert on direttoremagazzino
* for each row
* begin
* if(ControlloDisgiunzioneUtente(new.Soggetto)=0) then
* signal sqlstate '45000'
* set message\_text='Questo DIRETTORE MAGAZZINO è già un UTENTE esistente';
* end if;
* end $$
* delimiter ;
* /\* FINE: trigger per il controllo della disgiunzione DIRETTORE MAGAZZINO -> UTENTE \*/

**-V.Direttore**: Per ogni AZIENDA deve esistere uno ed un solo DIPENDENTE che è anche un DIRETTORE e che è un impiegato corrente:

* Per ogni istanza (Az,Dip) di “impiegato corrente” con Az appartenente a AZIENDA e Dip appartenente a DIPENDENTE, deve esistere l’istanza (Dip,Dir) di “ISA-Dip” con Dip appartenente a DIRETTORE il cui numero di tale istanza deve essere >0 e <2;
* **SQL**
* **FUNZIONI**
* /\* INIZIO: Controllo numero DIRETTORI \*/
* delimiter $$
* create function ControlloDirettori(x varchar(15)) returns int deterministic begin
* /\* Prima cerchiamo l'azienda per cui lavora x \*/
* select Azienda into @var1 from impiegatocorrente
* where impiegatocorrente.Dipendente=x;
* /\* Poi vediamo quanti direttori ci sono in tale azienda e se sono più di 0 significa che già c'è un direttore \*/
* select count(\*) into @var2 from impiegatocorrente
* inner join direttore on
* direttore.Dipendente=impiegatocorrente.Dipendente
* and impiegatocorrente.Azienda=@var1;
* if(@var2=0) then return 1;
* end if;
* return 0;
* end $$
* delimiter ;
* /\* FINE: Controllo numero DIRETTORI \*/
* **TRIGGER**
* /\* V\_Direttore \*/
* /\* Si è pensato onde evitare che non ci sia nessun direttore per un'azienda che quando si inserisce il primo DIPENDENTE automaticamente esso è il direttore \*/
* /\* Prima di tale operazione varrà eseguito un controllo per appunto controllare quanti direttori ci sono (max 1) \*/
* /\* INIZIO \*/
* delimiter $$
* create trigger V\_Direttore
* after insert on impiegatocorrente
* for each row
* begin
* if(ControlloDirettori(new.Dipendente)=1) then
* insert into direttore(Dipendente)
* values(new.Dipendente);
* end if;
* end $$
* delimiter ;
* /\* FINE \*/
* /\* INIZIO: Trigger per gestire l'inserimento di un solo DIRETTORE \*/
* delimiter $$
* create trigger V\_Direttore\_2
* before insert on direttore
* for each row
* begin
* if(ControlloDirettori(new.Dipendente)=1) then
* set new.Dipendente=new.Dipendente;
* else
* signal sqlstate '45000'
* set message\_text='Esiste già un direttore per questa azienda!';
* end if;
* end $$
* delimiter ;
* /\* FINE: Trigger per gestire l'inserimento di un solo DIRETTORE \*/

-**V.CompraVendiUtente**: Un utente non può comprare un annuncio che egli stesso ha pubblicato:

* Per ogni (U,AP) di “ordine” con U appartenente a UTENTE isa SOGGETTO e AP appartenente a ANNUNCIO, non deve esistere un’istanza (U,AC) con AC appartenente a ANNUNCIO ACQUISTATO dove AC=AP;
* **SQL:**
* **FUNZIONI**
* delimiter $$
* create function Controlla\_proprietario(x int,y varchar(15)) returns int deterministic begin
* /\* controllo che non sta comprando un suo annuncio \*/
* select count(\*) into @var1 from annuncio
* inner join soggetto
* on annuncio.Soggetto=y
* and soggetto.CF=y
* and annuncio.codice=x;
* /\* se è almeno impegnato con questa relazione allora non può comprare \*/
* if(@var1>0) then
* return 0;
* end if;
* /\* altrimenti può comprare \*/
* return 1;
* end $$
* delimiter ;
* /\* FINE: funzione che viene usata per il trigger del vincolo che un utente non può comprare un suo stesso annuncio \*/
* **TRIGGER**
* /\* INIZIO: trigger per controllare che un utente non sta comprando un suo annuncio \*/
* delimiter $$
* create trigger V\_CompraVendiUtente\_1
* before insert on annuncioacquistato
* for each row
* begin
* if(Controlla\_proprietario(new.Annuncio,new.Soggetto)=1) then /\* se restituisce 1 allora può comprarlo, else si \*/
* set new.Annuncio=new.Annuncio;
* set new.Soggetto=new.Soggetto;
* set new.DataOra=new.DataOra;
* else
* signal sqlstate '45000'
* set message\_text='Non puoi comprare un tuo annuncio!';
* end if;
* end $$
* delimiter ;
* /\* FINE: trigger per controllare che un utente non sta comprando un suo annuncio \*/

-**V.DirettoreMagazzino**: Un direttore di un magazzino non può acquistare annunci;

* Per ogni istanza D di DIRETTORE MAGAZZINO isa SOGGETTO non deve esistere l’istanza (D,AC) di “ordine” con AC appartenente a ANNUNCIO ACQUISTATO isa ANNUNCIO;
* **SQL:**
* **FUNZIONI**
* /\* INIZIO: Controllo che chi sta comprando non è un DIRETTORE MAGAZZINO \*/
* delimiter $$
* create function ControllaCompratore(x int,y varchar(15)) returns int deterministic begin
* /\* controllo se y è un DIRETTORE MAGAZZINO \*/
* select count(\*) into @var1 from soggetto
* inner join direttoremagazzino
* on direttoremagazzino.Soggetto=soggetto.CF
* and soggetto.CF=y;
* /\* se @var1 > 0 allora sicuramente è un direttore \*/
* if(@var1>0) then
* return 0;
* end if;
* /\* altrimenti non lo è e quindi può comprare \*/
* return 1;
* end $$
* delimiter ;
* /\* FINE: Controllo che chi sta comprando non è un DIRETTORE MAGAZZINO \*/
* **TRIGGER**
* /\* INIZIO: V\_DirettoreMagazzino \*/
* delimiter $$
* create trigger V\_DirettoreMagazzino
* before insert on annuncioacquistato
* for each row
* follows V\_CompraVendiUtente\_1
* begin
* if(ControllaCompratore(new.Annuncio,new.Soggetto)=0) then /\* se restituisce 0 allora un DIRETTORE MAGAZZINO sta ACQUISTANDO \*/
* signal sqlstate '45000'
* set message\_text='Un DIRETTORE di MAGAZZINO non può comprare un ANNUNCIO!';
* end if;
* end $$
* delimiter ;
* /\* FINE: V\_DirettoreMagazzino \*/

-**V.DipendenteAzienda**: Se un dipendente non lavora più nell’azienda dove lavorava, la data e l’ora di fine deve essere strettamente maggiore della data e l’ora dell’inizio:

* Per ogni istanza (D,A) di “impiegato corrente” con D appartenente a DIPENDENTE e A appartenente a AZIENDA, sia i l’attributo inizio di “impiegato corrente”, se (D,A) è anche un’istanza di “impiegato passato”, sia j l’attributo fine di “impiegato passato”, deve valere che j>i;
* **SQL:**
* **FUNZIONI**
* /\* INIZIO: controllo della data di fine DIPENDENTE \*/
* delimiter $$
* create function ControllaDipendente(x timestamp,y int,z varchar(15)) returns int deterministic begin
* /\* controllo se x (dataora) di z (dipendente) è maggiore della sua data di inizio \*/
* select inizio into @var1 from impiegatocorrente
* where impiegatocorrente.Azienda=y
* and impiegatocorrente.Dipendente=z;
* /\* se @var1<x \*/
* if(@var1<x) then
* return 1;
* end if;
* /\* altrimenti non è corretta la dataora \*/
* return 0;
* end $$
* delimiter ;
* /\* FINE: controllo della data di fine DIPENDENTE \*/
* **TRIGGER**
* /\* INIZIO: V\_DipendenteAzienda \*/
* delimiter $$
* create trigger V\_DipendenteAzienda
* before insert on impiegatopassato
* for each row
* begin
* if(ControllaDipendente(new.fine,new.Azienda,new.Dipendente)=0) then /\* se restituisce 0 allora la data di fine non è corretta\*/
* signal sqlstate '45000'
* set message\_text='La data di fine deve essere maggiore della sua data di inizio';
* end if;
* end $$
* delimiter ;
* /\* FINE: V\_DipendenteAzienda \*/

-**V.SpedizioneServizio**: Se un direttore di magazzino ha pubblicato un annuncio allora la sua eventuale spedizione deve essere per forza di servizio:

* Per ogni istanza (D,AV) di “ordine” con D appartenente a DIRETTORE MAGAZZINO isa SOGGETTO e AV appartenente a ANNUNCIO, se AV appartiene a ANNUNCIO VENDUTO isa ANNUNCIO, sia (AV,SP) con SP appartenente a SPEDIZIONE, SP deve per forza assumere il valore SER appartenente a SERVIZIO e non a AUTONOMA;
* **SQL:**
* **FUNZIONI**
* /\* INIZIO: funzione che controlla che l'annuncio è di un DIRETTORE MAGAZZINO o meno \*/
* delimiter $$
* create function ControlloV\_SpedizioneServizio(x int) returns int deterministic begin
* select count(\*) into @var1 from soggetto,direttoremagazzino,annuncio,annunciovenduto,spedizione
* where soggetto.CF=direttoremagazzino.Soggetto
* and annuncio.Soggetto=soggetto.CF
* and annunciovenduto.Annuncio=annuncio.codice
* and spedizione.AnnuncioVenduto=annunciovenduto.Annuncio
* and spedizione.codice=x;
* /\* se @var1 è uguale a zero allora non è di un DIRETTORE MAGAZZINO e quindi non può essere di SERVIZIO la SPEDIZIONE \*/
* if(@var1=0) then return 0;
* end if;
* return 1;
* end $$
* delimiter ;
* /\* FINE: funzione che controlla che l'annuncio è di un DIRETTORE MAGAZZINO o meno \*/
* **TRIGGER**
* /\* INIZIO: V\_SpedizioneServizio \*/
* delimiter $$
* create trigger V\_SpedizioneServizio
* before insert on servizio
* for each row
* begin
* if(ControlloV\_SpedizioneServizio(new.Spedizione)=0) then /\* se restituisce 0 allora non è di un direttore magazzino e quindi non può essere di servizio la spedizione \*/
* signal sqlstate '45000'
* set message\_text='La spedizione per un UTENTE deve essere per forza AUTONOMA';
* end if;
* end $$
* delimiter ;
* /\* FINE: V\_SpedizioneServizio \*/

-**V.SpedizioneAutonoma**: Se un cliente ha pubblicato un annuncio allora la sua eventuale spedizione deve essere per forza autonoma:

* Per ogni istanza (U,AV) di “ordine” con U appartenente a UTENTE isa SOGGETTO e AV appartenente a ANNUNCIO, se AV appartiene a ANNUNCIO VENDUTO isa ANNUNCIO, sia (AV,SP) con SP appartenente a SPEDIZIONE, SP deve per forza assumere il valore AUT appartenente a AUTONOMA e non a SERVIZIO;
* **SQL:**
* **FUNZIONI**
* /\* INIZIO: funzione che controlla che l'annuncio è di un DIRETTORE MAGAZZINO o meno \*/
* delimiter $$
* create function ControlloV\_SpedizioneServizio\_2(x int) returns int deterministic begin
* select count(\*) into @var1 from soggetto,utente,annuncio,annunciovenduto,spedizione
* where soggetto.CF=utente.Soggetto
* and annuncio.Soggetto=soggetto.CF
* and annunciovenduto.Annuncio=annuncio.codice
* and spedizione.AnnuncioVenduto=annunciovenduto.Annuncio
* and spedizione.codice=x;
* /\* se @var1 è uguale a zero allora non è di un UTENTE e quindi non può essere di SERVIZIO AUTONOMA \*/
* if(@var1=0) then return 0;
* end if;
* return 1;
* end $$
* delimiter ;
* /\* FINE: funzione che controlla che l'annuncio è di un DIRETTORE MAGAZZINO o meno \*/
* **TRIGGER**
* /\* INIZIO: V\_SpedizioneServizio\_2 \*/
* delimiter $$
* create trigger V\_SpedizioneServizio\_2
* before insert on autonoma
* for each row
* begin
* if(ControlloV\_SpedizioneServizio\_2(new.Spedizione)=0) then /\* se restituisce 0 allora non è di un utente e quindi non può essere autonoma la spedizione \*/
* signal sqlstate '45000'
* set message\_text='La spedizione per un DIRETTORE MAGAZZINO deve essere per forza di SERVIZIO';
* end if;
* end $$
* delimiter ;
* /\* FINE: V\_SpedizioneServizio\_2 \*/

-**V.Feedback**: Se esiste un Feedback da parte di un utente deve per forza appartenere ad un annuncio acquistato:

* Se esiste F appartenente a FEEDBACK, allora deve esistere l’istanza (F,AC) di “possiede” dove AC appartiene a ANNUNCIO ACQUISTATO isa ANNUNCIO;
* **SQL**
* **FUNZIONI**
* /\* V\_VerificaCompratore: Solo colui che ha comprato l'annuncio 'x'
* può inserire un feedback per l'annuncio 'x' e lo può
* fare solo se la spedizione di tale annuncio è in 'effettuata'
* ovvero è stata effettuata\*/
* /\* x -> il codice dell'annuncio acquistato
* y -> il CF del compratore
* bisogna controllare se colui che ha comprato x (un ipotetico z) corrisponde a y
* \*/
* delimiter $$
* create function V\_VerificaCompratore(x int, y varchar(15)) returns int deterministic begin
* select distinct(Soggetto) into @var1 from annuncioacquistato as ac
* where ac.Annuncio = x;
* if(@var1 = y) then return 1; /\* allora stai immettendo il CF del corretto UTENTE \*/
* end if;
* return 0;
* end $$
* delimiter ;
* **TRIGGER**
* /\* trigger per il vincolo V\_Feedack \*/
* delimiter $$
* create trigger V\_Feedback
* before insert on feedback
* for each row
* begin
* if(V\_VerificaCompratore(new.AnnuncioAcquistato, new.Utente)=0) then
* signal sqlstate '45000'
* set message\_text='Puoi inserire un feedback solo per l\' annuncio che hai comprato';
* end if;
* end $$
* delimiter $$

**Trigger utili a livello applicazione**

**-V\_SpedizioneNonEff :** E’ stato aggiunto tale trigger in modo tale da aggiungere automaticamente ,quando viene aggiunta una “spedizione”, tale “spedizione” in “noneffettuata” perché essenzialmente all’inizio la spedizione di un annuncio non è stata ancora effettuata.

* /\* INIZIO \*/
* /\* Trigger per far si di inserire automaticamente
* una spedizione in noneffettuata \*/
* delimiter $$
* create trigger V\_SpedizioneNonEff
* after insert on spedizione
* for each row
* begin
* insert into noneffettuata(Data,Spedizione)
* values(NOW(),new.codice);
* end $$
* delimiter ;
* /\* FINE \*/

**-V\_AnnuncioVendutoSpedizione :** E’ stato aggiuntotale trigger per far si che dopo aver venduto annuncio , tale annuncio deve essere spedito e quindi aggiunto in “spedizione”.

* /\* INIZIO \*/
* /\* Trigger per far si di inserire automaticamente
* un annunciovenduto in spedizione \*/
* delimiter $$
* create trigger V\_AnnuncioVendutoSpedizione
* after insert on annuncioacquistato
* for each row
* begin
* insert into spedizione(Data,Utente,AnnuncioVenduto)
* values(NOW(),new.Soggetto,new.Annuncio);
* end $$
* delimiter ;
* /\* FINE \*/

**Operazioni sulla basi di dati: QUERY**

**Op.1: Visualizza il numero di spedizioni per ogni città;**

create view SpedizioniPerCittà(codice,Città)

as select sp.codice, c.nome from spedizione as sp, citta as c, utente as ut, soggetto as sog

where ut.Soggetto = sog.CF

and sp.Utente = ut.Soggetto

and sog.Città = c.nome;

create view SpedizioniPerOgniCittà

as select Città, count(Città) as num\_Spedizioni

from SpedizioniPerCittà

group by Città;

select \* from SpedizioniPerOgniCittà;

**Op.2: Visualizza annunci che non sono stati venduti;**

select \* from annuncio

where codice not in

(select Annuncio from annunciovenduto);

**Op.3: Visualizzare in quale città avvengono più spedizioni;**

select Città from SpedizioniPerOgniCittà as sp

where sp.num\_Spedizioni = any (select max(num\_Spedizioni) from SpedizioniPerOgniCittà);

**Op.4: Visualizzare quali Dipendenti ancora non hanno consegnato le loro spedizioni;**

select distinct(CF) as Dipendente from dipendente as d, servizio as ser, spedizione as sp, effettuata as eff

where ser.Dipendente = d.CF

and sp.codice = ser.Spedizione

and sp.codice not in (select Spedizione from effettuata);

**Op.5: Calcola la media dei feedback ottenuti da un utente;**

select s.CF,avg(f.voto)-1 as media\_feedback from soggetto as s, annuncio as ann, feedback as f, annunciovenduto as av

where av.Annuncio = ann.codice

and f.AnnuncioAcquistato = av.Annuncio

and ann.Soggetto = s.CF

group by s.CF;

**Op.6: Visualizzare qual è la categoria dominante;**

create view AnnunciPerCategoria(Categori a, num\_Annunci)

as select nome, count(codice) from categoria as c, annuncio as ann

where ann.Categoria = c.nome

group by c.nome;

select Categoria ,num\_Annunci from AnnunciPerCategoria where num\_Annunci = any

(select max(num\_Annunci) from AnnunciPerCategoria);

**Op.7: Visualizzare i dipendenti che lavorano in un’ azienda situata nella città in cui sono stati venduti più annunci;**

select \* from dipendente as d, azienda as az, impiegatocorrente imp

where imp.Dipendente = d.CF

and imp.Azienda = az.id

and az.Città = (select Città from SpedizioniPerOgniCittà

where num\_Spedizioni = any

(select max(num\_Spedizioni) from SpedizioniPerOgniCittà));

**Op.8: Visualizzare le città in cui avvengono le spedizioni dove non risiedono dei magazzini (è stata utilizzata la VIEW dell’Op.1);**

select distinct(spc.Città) from magazzino as mag, SpedizioniPerOgniCittà as spc, citta as cit

where cit.nome = spc.Città

and spc.Città not in

(select Città from magazzino);

**Op.9: Utenti che hanno acquistato annunci da soggetti concittadini; (stessa città);**

select ac.Soggetto, av.Utente, ann.codice from annuncioacquistato as ac, annunciovenduto as av, Annuncio as ann

where ac.Annuncio = av.Annuncio

and ac.Soggetto != av.Utente

and ac.Annuncio = ann.codice

and av.Annuncio = ann.codice

and ac.Soggetto not in (select CF from soggetto as sog where sog.Città != any (select Città from soggetto as sog2 where sog2.CF = av.Utente));

**Op.10: Inserimento nuovo utente;**

insert into soggetto(CF,DataNascita,email,nome,cognome,Via,civico,CAP,Città)

values('A','1990-01-01','a@gmail.com','Edoardo','Mariani','Via prova',1,00171,'Roma'),

('B','1991-01-01','b@gmail.com','Marco','Rossi','Via prova',2,00191,'Bologna'),

('C','1992-01-01','c@gmail.com','Luigi','Scafati','Via prova',3,00191,'Milano'),

('D','1993-01-01','d@gmail.com','Alex','Da Silva','Via prova',4,00191,'Napoli'),

('E','1994-01-01','e@gmail.com','Federico','Domini','Via prova',5,00121,'Firenze'),

('F','1995-01-01','f@gmail.com','Chiara','Palma','Via prova',6,00121,'Roma'),

('G','1996-01-01','g@gmail.com','Giulia','Bertini','Via prova',7,00121,'Bologna'),

('H','1997-01-01','h@gmail.com','Flavia','Peronaci','Via prova',8,00171,'Napoli'),

('I','1998-01-01','i@gmail.com','Giuseppe','Ponterio','Via prova',1,00171,'Firenze'),

('L','1999-01-01','l@gmail.com','Carla','Ghezzi','Via prova',2,00191,'Bologna');

insert into utente(nickname,media\_feedback,Soggetto)

values('m',DEFAULT,'A'),

('n',DEFAULT,'B'),

('o',DEFAULT,'C'),

('p',DEFAULT,'D'),

('q',DEFAULT,'E'),

('r',DEFAULT,'F'), (

's',DEFAULT,'G'),

('t',DEFAULT,'H');

insert into conto(saldo,Soggetto)

values ('$saldo','$CF');

insert into movimento(importo,DataOra,Conto)

values(0,NOW(),LAST\_INSERT\_ID());

insert into telefono(num,Soggetto)

values('$telefono','$CF');

**Op.11: Inserimento direttore magazzino;**

insert into direttoremagazzino(licenza,Soggetto)

values(true,'I'),

(true,'L');

**Op.12: Acquisto Annuncio;**

insert into annuncioacquistato(DataOra,Annuncio,Soggetto)

values(NOW(),'$codice\_annuncio','$compratore');

insert into annunciovenduto(DataOra,Annuncio,Utente)

values(NOW(),'$codice\_annuncio','$venditore');

**Op.13: Inserisci Feedback;**

insert into feedback(voto,AnnuncioAcquistato,Utente)

values('$voto','$annuncio','$utente');

**Op.14: Visualizza Annunci disponibili;**

select \* from annuncio

where codice not in

(select Annuncio from annunciovenduto);

**Op.15: Inserisci Azienda;**

insert into azienda(ragione\_sociale,nome,Città,Via,civico,CAP)

values('$ragione','$nome','$città','$Via','$civico','$CAP');

**Op.16: Inserisci Dipendente;**

insert into dipendente(nome,cognome,DataNascita,CF)

values('$nome','$cognome','$data','$CF');

**Op.17: Inserisci Direttore;**

insert into direttore(Dipendente)

values('$direttore');

**Op.18: Inserisci Impiegato;**

insert into impiegatocorrente(inizio,Azienda,Dipendente)

values(NOW(),'$azienda','$dipendente');

**Op.19: Inserisci Magazzino;**

insert into magazzino(ragione\_sociale,Via,civico,CAP,DirettoreMagazzino,Città)

values('$ragione','$Via','$civico','$CAP','$direttore','$città');

**Op.20: Inserisci SpedizioneAutonoma;**

insert into autonoma(Spedizione) values('$spedizione');

**Op.21: Pubblica Annuncio;**

insert into annuncio(DataOra,periodo\_usato,prezzo,descrizione,Soggetto,Categoria)

values(NOW(),'$periodo\_usato','$prezzo','$descrizione','$soggetto','$categoria');

**Op.22: Inserisci SpedizioneDiServizio:**

insert into servizio(Spedizione,Dipendente,Magazzino) values('$spedizione','$dipendente','$magazzino');

Soddisfazione dei requisiti 2

|  |  |
| --- | --- |
| Deve prendere in considerazione un dominio di interesse che consente di identificare tra le 8 e le 12 entità concettuali principali. Questosenza contare le sotto-entità che compaiono nelle relazioniISA e/o nelle generalizzazioni; | Ci sono già dal diagramma E/R concettuale 13 entità principali (sena contare le sotto-entità e/o le generalizzazioni); |
| Deve avere una complessità tale per cui lo schema ER contiene sia relazioni ISA che generalizzazioni; | Già dal diagramma concettuale si hanno Servizio, Autonoma, Effettuata, NonEffettuata, AnnuncioAcquistato, AnnuncioVenduto; |
| Deve esserci una struttura sufficiente nelle relazioni. Questo comporta che se si vede il diagramma ER come un grafo(in cui i nodi sono le entità e gli archi sono dati dalla partecipazione delle entità nelle relazioni) questo deve contenere dei cicli; | Uno dei tanti cicli presenti è il fatto che un Soggetto pubblica un Annuncio che può essere acquistato da un altro Soggetto (quindi ciclo); |
| Lo schema deve contenere vincoli di cardinalità sulla partecipazione delle entità a relazioni diverse dal valore predefinito (0, n); | Ci sono anche cardinalità (0,1) e (1,1); |
| Lo schema deve contenere alcuni attributi facoltativi e alcuni multi valore; | Si hanno gli attributi facoltativi “nick\_name” e quelli multivalore “telefono” e altri; |
| Devono inoltre esserci dei vincoli esterni(almeno 4) che non possono essere rappresentati nel modello ER; | Ci sono più di 4 vincoli esterni; |
| Le specifiche devono includere un'indicazione dei volumi per le varie entità e relazioni (prestare attenzione alla coerenza tra i volumi e i vincoli di cardinalità dello schema ER); | Guarda indice per le tabelle dei volumi; |
| Le specifiche devono includere un carico di lavoro delle query e delle operazioni più comuni (tra 10 e 15) che sono di interesse nel dominio modellato, con un'indicazione della loro frequenza; | Guarda indice per le QUERY; |