Basi di Dati

Progetto A.A. 2022/2023

NOLEGGIO DI FILM

0257344

Luca Moretti

Indice

1. Descrizione del Minimondo

Si vuole realizzare un sistema informativo per la gestione di un negozio per il noleggio delle videocassette e DVD, tenendo conto delle seguenti informazioni.
Il sistema mantiene le informazioni relative a tutte le persone impiegate presso il negozio. Per ciascun impiegato sono noti il codice fiscale, il nome, ed un recapito. Inoltre si vuole tenere traccia della carica che ha rivestito in un determinato periodo (per esempio, cassiere, commesso, addetto di magazzino, ...) considerando che tale qualifica può cambiare nel tempo. Le informazioni sul personale sono gestite dal proprietario del negozio, che è in grado di inserire nuovo personale nel sistema e di visualizzare report mensili ed annuali sulle ore lavorate dai dipendenti. I turni di lavoro sono inseriti, su base mensile, sempre dal proprietario. I film disponibili presso il negozio sono identificati dal titolo e dal nome del regista; inoltre sono noti l'anno in cui il film è stato girato, l'elenco degli attori principali del film, il costo corrente di noleggio della videocassetta ed eventualmente i film disponibili presso la catena di cui il film in questione rappresenta la versione "remake". Per ogni film è nota la collocazione all'interno del negozio. In particolare, sono noti il settore, la posizione all'interno del settore ed il numero di copie in cui il film è disponibile. Ciascun settore è identificato attraverso un codice numerico univoco all'interno del negozio.I clienti della catena, al momento della registrazione al negozio, ricevono una tessera cliente. Per ciascun cliente devono essere mantenute tutte le informazioni anagrafiche e viene associato anche un numero arbitrario di recapiti (telefono, email, cellulare) a cui possono essere contattati. Quando un cliente effettua un noleggio, viene registrata la data entro cui il film dovrà essere restituito. Il personale della catena può gestire l'anagrafica dei clienti e gestire gli ordini. Inoltre, può visualizzare in ogni momento, per ciascun centro di servizio, quali titoli sono associati ad un noleggio scaduto e quali sono i clienti che hanno effettuato tali noleggi.

2. Analisi dei Requisiti

Identificazione dei termini ambigui e correzioni possibili

Linea	Termine	Nuovo termine	Motivo correzione
3	Persone Impiegate	Impiegati	Le persone impiegate presso il negozio vengono definite come impiegati, per unificare i sinonimi.
4	Recapito	telefono, email, cellulare	Evidenziare in modo esplicito quali sono le tipologie di recapito.
5	Carica	Incarico	Utilizzo un termine non ambiguo al posto di carica, per evidenziare il concetto di incarico.
6	Qualifica	Incarico	Utilizzo un termine non ambiguo al posto di qualifica, per evidenziare il concetto di incarico.
7-8	Personale	Impiegato	Le informazioni sul nuovo personale, e il personale stesso, non sono altro che sinonimi indicanti lo stesso concetto di Impiegato.
9	Dipendenti	Impiegati	I dipendenti che lavorano all'interno del negozio, vengono definiti come impiegati per eliminare ridondanze.
12	Videocassetta	Copia di Film	La videocassetta presente nel negozio non è altro che un altro modo di definire una tipologia di copia di un film, ma la tipologia di copia non viene richiesta dalla specifica.
12-13	i film disponibili presso la catena di cui il film in questione rappresenta la versione "remake"	Associazione	Esemplificare la frase e evidenziare l'esistenza dell' associazione tra il film e il suo remake.
14-21	Film	Copia di film	Utile a differenziare il concetto di film(astratto) dalla copia di film(fisica) presente all'interno del negozio.
17	Catena	Negozio	Come concordato con il committente, la specifica si riferisce ad un solo negozio e non ad una catena di negozi.
17-18	Informazioni anagrafiche	Nome,Cognome sesso,luogo di nascita, data di nascita, età, codice fiscale	Evidenziare in modo esplicito quali sono i dati anagrafici di riferimento.
21	Personale della catena	Impiegato	Come precedentemente definito, il personale della catena non sono altro che gli impiegati del negozio, in questo caso si parla solamente di impiegati.

22	Ordini	Noleggio	Disambiguare il concetto di ordine che è fuorviante ed utilizziamo il concetto di noleggio.
23	Centro Di Servizio	Negozio	Centro di servizio, in questo caso non è altro che il negozio.
23	Titoli	Copia di film	I titoli associati ad un noleggio, si intendono le copie noleggiate all' interno del negozio.

Specifica disambiguata

Si vuole realizzare un sistema informativo per la gestione di un negozio per il noleggio delle videocassette e DVD, tenendo conto delle seguenti informazioni.

Il sistema mantiene le informazioni relative a tutti gli **impiegati** presso il negozio. Per ciascun **impiegato** sono noti :

- Codice Fiscale;
- Nome;
- Un recapito (telefono,email,cellulare);
- L' incarico che ha rivestito in un determinato periodo (per esempio, cassiere, commesso, addetto di magazzino, ...) considerando che tale incarico può cambiare nel tempo.

.Le informazioni degli impiegati sono gestite dal **proprietario del negozio**, che può:

- Inserire un nuovo impiegato nel sistema;
- Visualizzare report mensili ed annuali sulle ore lavorate dagli impiegati;
- Inserire mensilmente i turni di lavoro.

I **film** disponibili presso il negozio sono identificati da:

- Titolo;
- Nome del regista.

Inoltre per i **film** sono noti:

- l'anno in cui è stato girato;
- l'elenco degli attori principali;
- Il numero di copie in cui è disponibile;
- I film disponibili associati al "remake".(Associazione)

Inoltre della **copia di film**:

- Il costo corrente di noleggio;
- Settore;
- La posizione all'interno del settore.

Ciascun **settore** :

- Identificato attraverso un codice numerico univoco all'interno del negozio.

I **clienti** al momento della registrazione ricevono:

- Tessera cliente (Identificato da codice);

Per ciascun **cliente** sono noti:

- Nome;
- Cognome;
- Sesso;
- Luogo di nascita;
- Data di nascita;
- Codice Fiscale;
- Numero arbitrario di recapiti (telefono, email, cellulare).

Quando un **cliente effettua un noleggio**:

- Viene registrata la data entro cui la copia di film dovrà essere restituita.

Un **impiegato** può gestire:

- Anagrafica dei clienti;
- Noleggi.

Inoltre un **impiegato** può visualizzare in ogni momento, per ciascun negozio:

- Quali copie di film sono associate ad un noleggio scaduto;
- Quali clienti che hanno effettuato tali noleggi.

Glossario dei Termini

Termine	Descrizione	Sinonimi	Collegamenti
Impiegato	Lavoratore all'interno del negozio, può ricoprire diversi incarichi e gestire informazioni dei clienti.	Persone impiegate, personale, dipendenti, personale della catena.	Clienti, copia di film, noleggio,proprietario del negozio.
Proprietario del negozio	Colui che gestisce il negozio, può gestire tutte le informazioni riguardanti gli impiegati.		Impiegato.
Copia di film	Copia fisica del film, diversa dal suo concetto. Contenuta in varie copie all'interno del negozio.	Film, film disponibili, videocassette, titoli.	Film, noleggio, cliente, impiegato.
Film	Concetto astratto di film, indica concettualmente quale film è stato girato.		Copia di film.
Cliente	Cliente del negozio. Può registrarsi e noleggiare copie di film.		Copia di film, impiegato, noleggio.
Noleggio	Azione compiuta dal cliente per prenotare una copia di film per una certa data.	Ordine.	Cliente, copia di film, impiegato.
Turno	Azione compiuta dall'impiegato in una giornata lavorativa. Può essere gestito dal proprietario.		Impiegato.
Settore	Zona del negozio che ospita copie di film.		Copia di film.

Raggruppamento dei requisiti in insiemi omogenei

Frasi di carattere generale
Si vuole realizzare un sistema informativo per la gestione di un negozio per il noleggio delle videocassette e DVD, tenendo conto delle seguenti informazioni.

Frasi relative a impiegato
Il sistema mantiene le informazioni relative a tutti gli **impiegati** presso il negozio. Per ciascun **impiegato** sono noti : ● Codice Fiscale; ● Nome; ● Un recapito (telefono,email,cellulare); ● L' incarico che ha rivestito in un determinato periodo (per esempio, cassiere, commesso, addetto di magazzino, ...) considerando che tale incarico può cambiare nel tempo. Un **impiegato** può gestire: ● Anagrafica dei clienti; ● Noleggi. Inoltre un **impiegato** può visualizzare in ogni momento, per ciascun negozio: ● Quali copie di film sono associate ad un noleggio scaduto; ● Quali clienti che hanno effettuato tali noleggi. .Le informazioni degli impiegati sono gestite dal **proprietario del negozio**, che può: ● Inserire un nuovo impiegato nel sistema; ● Visualizzare report mensili ed annuali sulle ore lavorate dagli impiegati; ● Inserire mensilmente i turni di lavoro.

Frasi relative a proprietario del negozio
Le informazioni degli impiegati sono gestite dal **proprietario del negozio**, che può:

- Inserire un nuovo impiegato nel sistema;
- Visualizzare report mensili ed annuali sulle ore lavorate dagli impiegati;
- Inserire mensilmente i turni di lavoro.

Frasi relative a copia di film

Inoltre della **copia di film**:

- Il costo corrente di noleggio;
- Settore;
- La posizione all'interno del settore.

Inoltre per i **film** sono noti:

- Il numero di copie in cui è disponibile.

Quando un **cliente effettua un noleggio**:

- Viene registrata la data entro cui la copia di film dovrà essere restituita.

Inoltre un **impiegato** può visualizzare in ogni momento, per ciascun negozio:

- Quali copie di film sono associate ad un noleggio scaduto.

Frasi relative a film

I **film** disponibili presso il negozio sono identificati da:

- Titolo;
- Nome del regista.

Inoltre per i **film** sono noti:

- l'anno in cui è stato girato;
- l'elenco degli attori principali;
- Il numero di copie in cui è disponibile;
- I film disponibili associati al "remake".(Associazione)

Frasi relative a cliente

I **clienti** al momento della registrazione ricevono:

- Tessera cliente (Identificato da codice);

Per ciascun **cliente** sono noti:

- Nome;
- Cognome;
- Sesso;
- Luogo di nascita;
- Data di nascita;
- Codice Fiscale;
- Numero arbitrario di recapiti (telefono, email, cellulare).

Quando un **cliente effettua un noleggio**:

- Viene registrata la data entro cui la copia di film dovrà essere restituita.

Un **impiegato** può gestire:

- Anagrafica dei clienti;

Inoltre un **impiegato** può visualizzare in ogni momento, per ciascun negozio:

- Quali clienti che hanno effettuato tali noleggi.

Frasi relative a noleggio

Quando un **cliente effettua un noleggio**:

- Viene registrata la data entro cui la copia di film dovrà essere restituita

Inoltre della **copia di film**:

- Il costo corrente di noleggio;

Un **impiegato** può gestire:

- Noleggi.

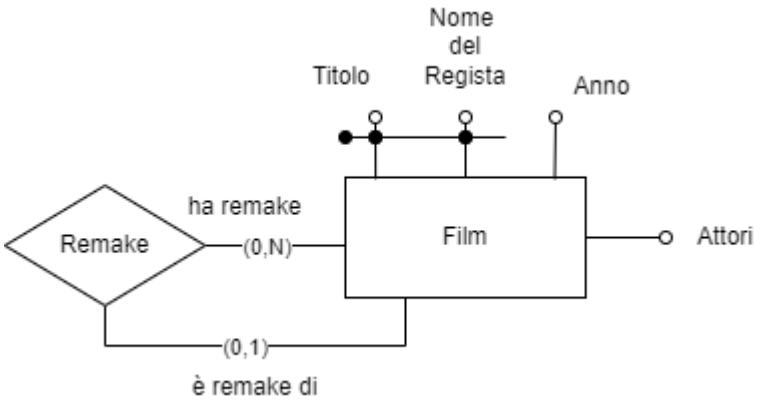
Inoltre un **impiegato** può visualizzare in ogni momento, per ciascun negozio:

- Quali copie di film sono associate ad un noleggio scaduto;
- Quali clienti che hanno effettuato tali noleggi.

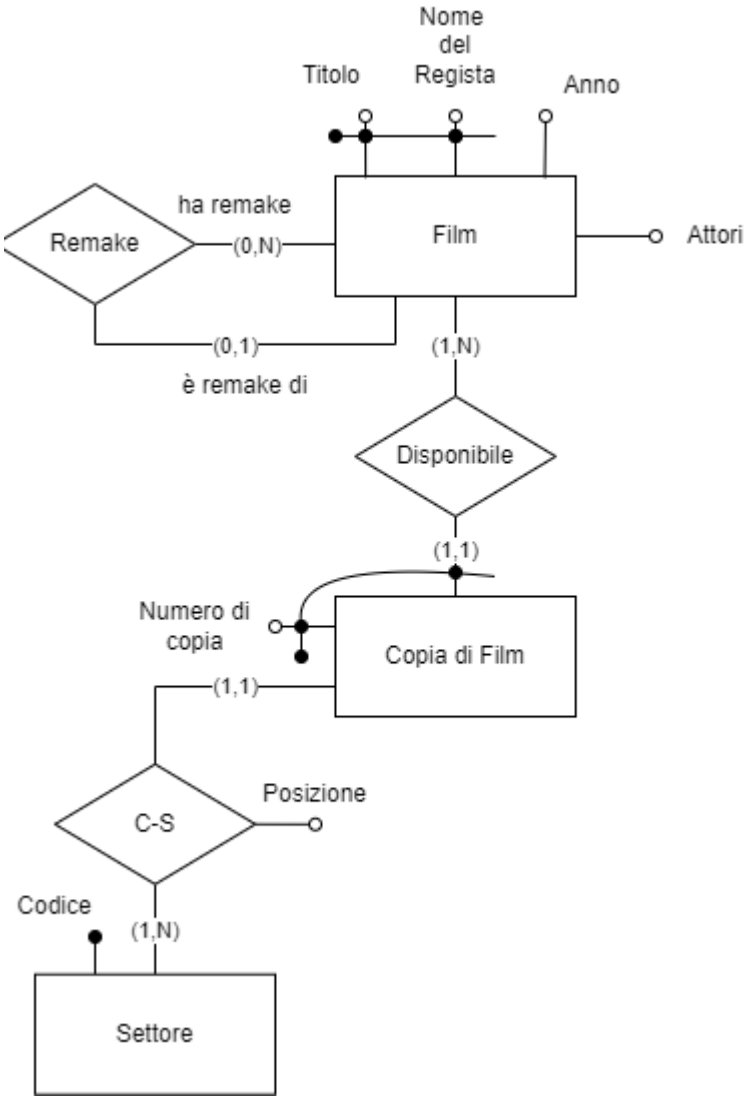
3. Progettazione concettuale

Costruzione dello schema E-R

Per la realizzazione dello schema E-R utilizzo la strategia inside-out o a macchia d'olio, partendo dal concetto fondamentale del minimondo: il concetto di film. Il film come evidenziato nel raggruppamento dei requisiti, viene identificato: dal titolo e dal nome del regista. Sono noti inoltre: l'anno in cui è stato girato, l'elenco degli attori principali e il numero di copie in cui è disponibile dato ridondante che può essere facilmente ricavabile. Infine si vuole tener traccia dei film disponibili associati al remake; questo nello schema E-R viene tradotto come relazione ricorsiva non simmetrica, che chiameremo Remake, questa relazione indica quale film ha remake con cardinalità (0,N) infatti un film può o non avere uno o più remake, mentre il verso della relazione è remake di, indica del remake qual'è il film originale. In questo caso la cardinalità è (0,1) poiché può esistere o meno il remake di un film.

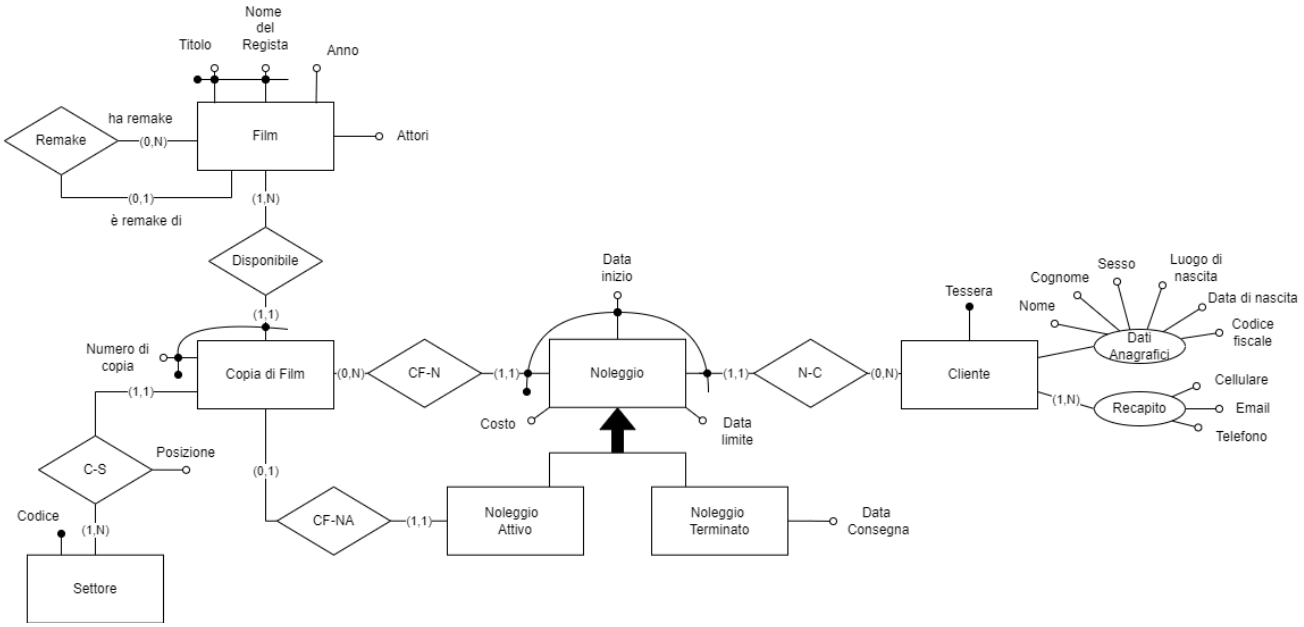


Direttamente collegato al concetto di film c'è la sua copia. Per realizzare ciò viene utilizzato un pattern di tipo instance-of, per dividere infatti il concetto di film che è astratto dalla sua copia fisica e aggiungendo il numero di copia, per discriminare a quale copia mi riferisco nell'associazione disponibile; nella quale l'entità film partecipa con cardinalità (1,N) infatti un film deve essere disponibile in più copie e l'entità copia di film partecipa con cardinalità (1,1) deriva dal pattern instance of. Della copia inoltre bisogna tener traccia del settore in cui essa si trova, settore che è identificato da un codice, ed anche la posizione all' interno del settore.



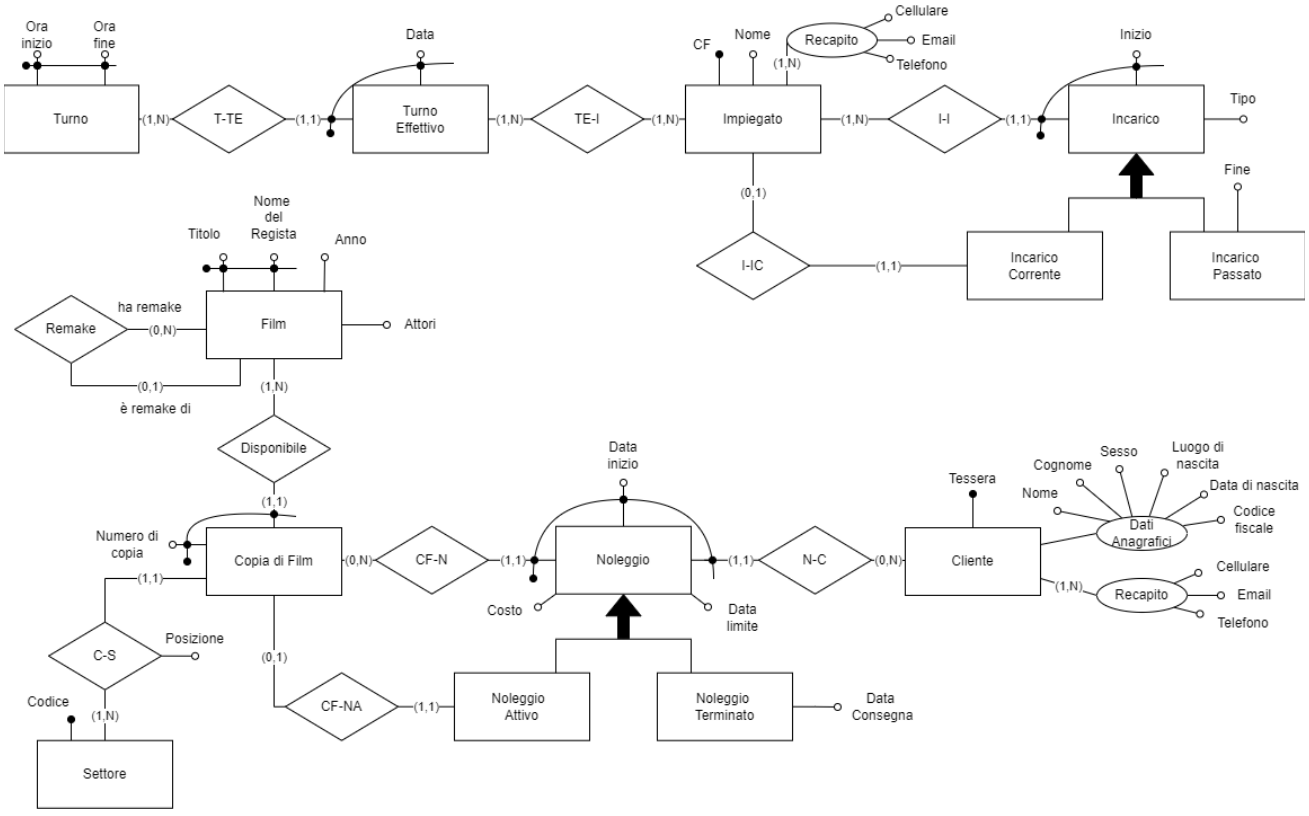
Dopo aver introdotto il concetto di copia di film, viene quasi automatico parlare del cliente e conseguentemente del concetto di noleggio, fondamentale nel minimondo di riferimento. Del cliente sappiamo che al momento della registrazione, viene identificato da una tessera con un codice, un numero arbitrario di recapiti (telefono, email, cellulare) e i dati anagrafici (nome, cognome, sesso, luogo di nascita, data di nascita, codice fiscale). Entrambi (dati anagrafici, recapito) indicati come attributi composti, del recapito troviamo anche la cardinalità (1,N) poiché ne vogliamo un numero arbitrario, ma almeno 1. Il collante tra le due entità (copia di film e cliente) come precedentemente detto è il noleggio. Di quest'ultimo bisogna conoscere: il costo e della data limite entro la quale la copia di film deve essere restituita, è utile infatti dividere il concetto di noleggio attivo e terminato utilizzando un pattern di storicizzazione. Il noleggio infatti è un'entità debole che viene identificata dalla copia di film (che viene noleggiata), dal cliente (che la noleggia) e dalla data di noleggio. L'entità debole noleggio partecipa infatti alle associazioni CF-N e N-C con cardinalità (1,1) come

precedentemente detto, mentre la copia film partecipa all'associazione CF-N con cardinalità (0,N) poiché può o non può essere noleggiata una o più copie di film, d'altra parte anche l'entità cliente partecipa all'associazione N-C con cardinalità (0,N), perchè un cliente può noleggiare, o meno, diverse copie di film. Vogliamo tener traccia inoltre del noleggio attivo di una copia di film, utilizzando l'associazione CF-NA, in questa associazione la copia di film partecipa con cardinalità (0,1) poiché una copia può essere o non essere noleggiata ma può far parte solamente di un noleggio attivo alla volta, infatti il noleggio attivo partecipa con cardinalità (1,1) poiché una e una sola copia di film può far parte di un noleggio attivo alla volta.



Parliamo ora del concetto di impiegato. Per gli impiegati sono noti codice fiscale, nome, un recapito (telefono,email,cellulare), l'incarico che ha rivestito in un determinato periodo (per esempio, cassiere, commesso, addetto di magazzino, ...) considerando che tale incarico può cambiare nel tempo e il turno poiché il proprietario del negozio deve poter gestire i turni. Per il concetto di incarico viene utilizzato un pattern di storicizzazione. L'incarico infatti è un'entità debole identificata da una data di inizio e da almeno un impiegato che lo ricopre. L'incarico dunque è un'entità debole, che partecipa con cardinalità (1,1) all' associazione I-I per quanto detto precedentemente, mentre l'impiegato partecipa con cardinalità (1,N) poiché almeno uno, ma anche più di uno possono fare lo stesso incarico. Come evidenziato dalla specifica inoltre, bisogna tener traccia dell'incarico corrente che un determinato impiegato svolge, questo viene reso disponibile dall'associazione I-IC in cui l'impiegato partecipa con cardinalità (0,1) poichè può o non può rivestire un certo incarico in un determinato periodo, mentre l'incarico corrente partecipa con vincoli (1,1) perchè un incarico corrente è svolto da un impiegato. Infine viene inserito il concetto di turno con un instance-of per

dividerlo dal turno effettivo che viene svolto dall'impiegato.L'entità debole turno effettivo è identificata da un turno e una data, utile per discriminare i diversi turni effettivi, e assumendo che un impiegato svolge un turno al giorno. Inoltre l'entità debole partecipa all'associazione TE-I con cardinalità (1,N) poiché lo stesso turno effettivo può essere svolto da più impiegati mentre quest' ultimo partecipa con cardinalità (1,N) poiché un impiegato svolge uno o più turni effettivi.
Il turno invece è identificato da un'ora di inizio e di fine, che nella relazione T-TE partecipa con cardinalità (1,N) poiché riferito da più turni effettivi, mentre l'entità debole turno effettivo partecipa con cardinalità (1,1) dovuto al pattern instance-of. Il concetto di impiegato è distaccato dagli altri concetti poiché collegato solo dal punto di vista delle operazioni, così come il proprietario del negozio che non viene rappresentato nel modello concettuale.



Integrazione finale

Non c'è bisogno di alcuna integrazione poiché non vi sono conflitti. Lo schema finale è quello sopra riportato.

Regole aziendali

- I **Noleggi Attivi** vengono considerati scaduti se la data al momento del report è maggiore della data limite.
- Un **Impiegato** non può svolgere un turno effettivo se non ha un incarico corrente.
- Un **Cliente** non può noleggiare due volte la stessa copia prima di riconsegnarla.
- Un **Cliente** non può noleggiare una nuova copia se ha un noleggio scaduto.
- Un **Noleggio** dura una settimana.

Dizionario dei dati

Entità	Descrizione	Attributi	Identificatori
Film	Concetto astratto di film, indica concettualmente quale film è stato girato.	Anno, attori, numero di copie.	Titolo, nome del regista.
Copia di film	Copia fisica del film, diversa dal suo concetto. Contenuta in varie copie all'interno del negozio.	Settore, posizione.	Titolo, nome del regista, numero di copia.
Noleggio	Azione compiuta dal cliente per prenotare una copia di film per una certa data.	Costo, data limite.	Titolo, nome del regista, numero di copia. data, tessera.
Noleggio Attivo	Azione effettivamente eseguita da un cliente, che noleggia una copia di film.	Costo, data limite.	Titolo, nome del regista, numero di copia. data, tessera.
Noleggio Terminato	Azione compiuta da un cliente, che non ha restituito una copia di film.	Costo, data limite , data consegna.	Titolo, nome del regista, numero di copia. data, tessera.
Cliente	Cliente del negozio. Può registrarsi e noleggiare copie di film.	Dati anagrafici (nome, cognome, sesso, luogo di nascita, data di nascita, codice fiscale), recapito (telefono,email, cellulare).	Tessera.
Impiegato	Lavoratore all'interno del negozio, può ricoprire diversi incarichi e gestire informazioni dei clienti.	Nome, recapito (telefono,email, cellulare).	Codice fiscale.
Incarico	Lavoro svolto da un impiegato in un determinato periodo.	Tipo.	Codice fiscale, inizio.

Incarico Corrente	Lavoro svolto correntemente da un impiegato.	Tipo.	Codice fiscale, inizio.
Incarico Passato	Lavoro che ha svolto un impiegato in un determinato periodo.	Tipo, fine.	Codice fiscale, inizio.
Turno	Orario lavorativo all' interno del negozio.		Ora inizio, ora fine.
Turno effettivo	Effettiva esecuzione del turno di lavoro in una specifica data.		Ora inizio, ora fine, data.

4. Progettazione logica

Volume dei dati

Il sistema mantiene i dati relativi ai noleggi e ai clienti per un periodo non superiore a 10 anni. Lo stesso vale per gli impiegati. Partendo dai concetti principali, quali: film, copia di film e cliente.
Assumendo che nel negozio sono presenti 1000 film diversi (di cui il 20% hanno remake quindi 200) e che ogni film abbia 10 copie e mensilmente arrivano 5 nuovi titoli(per un totale di 60 film e 1200 copie annue), dei clienti invece viene assunto che in 1 anno il negozio ne raccoglie 3000 clienti (considerando 20 clienti giornalieri di cui il 50% sono ricorrenti). Per il noleggio viene considerato che quotidianamente un cliente noleggia due copie di film. I noleggi terminati e attivi considerati con un rapporto differente, con il tempo i noleggi terminati saranno molto maggiori, inoltre dei noleggi il 10% sono scaduti. Il negozio ha 3 incarichi differenti disponibili: cassieri/e, magazzinieri/e, commessi/e; di questi vi sono 4 posti ciascuno per un totale di 12. Assumendo che in un anno 4 impiegati cambiano lavoro, in totale, annualmente ci saranno 16 nuovi impiegati annuali e considerando che il 10% cambia mensilmente incarico(19 incarichi diversi annuali). Dei turni invece viene assunto che ci sono 4 differenti orari lavorativi(8-12, 14-18. 12-18, 8-16) mentre per i turni effettivi vengono considerati mediamente 6 giorni lavorativi esclusi i festivi (per un totale di 300 giorni annui che per una media di 12 impiegati).

Concetto nello schema	Tipo[1]	Volume atteso
Film	E	1600
Copia di film	E	16000
Cliente	E	30000
Noleggio	E	60000
Noleggio Attivo	E	6000
Noleggio Terminato	E	54000
Impiegato	E	160
Incarico	E	190
Incarico Corrente	E	12
Incarico Passato	E	178
Turno	E	4
Turno Effettivo	E	36000
Remake	R	320
Disponibile	R	16000
CF-N(Copia film - Noleggio)	R	60000
CF-NA(Copia film - Noleggio Attivo)	R	6000
N-C(Noleggio - Cliente)	R	60000

[1] Indicare con E le entità, con R le relazioni

I-I(Impiegato - Incarico)	R	190
I-IC(Impiegato - Incarico Corrente)	R	12
T-TE	R	36000
TE-I	R	20000

Tavola delle operazioni

Nel sistema vi sono due tipologie di utenti: gli impiegati e il proprietario del negozio.

Gli **impiegati** possono gestire:

- Anagrafica dei clienti;
- Noleggi.

Un impiegato inoltre deve poter visualizzare e gestire i film e le copie di film presenti nel negozio, e deve poter registrare un nuovo cliente.

L' impiegato inoltre può visualizzare (report):

- Quali copie di film sono associate ad un noleggio scaduto;
- Quali clienti che hanno effettuato tali noleggi.

Il **proprietario del negozio** che può:

- Inserire un nuovo impiegato nel sistema;
- Inserire mensilmente i turni di lavoro;
- Visualizzare report mensili ed annuali sulle ore lavorate dagli impiegati.

Infine avremo bisogno di un'operazione che prevede il login.

Cod.	Descrizione	Frequenza attesa
I01	Stampa la lista dei film disponibili.	20/giorno
I02	Stampa la lista delle copie disponibili di un film.	20/giorno
I03	Aggiungi copia di film.	1/mese
I04	Stampa la lista dei noleggi attivi di un cliente.	5/giorno
I05	Trova copia di film	20/giorno
I06	Aggiungi noleggio.	20/giorno
I07	Termina noleggio.	20/giorno
I08	Modifica anagrafica clienti.	1/settimana
I09	Registrazione cliente.	10/giorno
I10	Report copie di film associate a noleggio scaduto.	1/mese
I11	Report dei clienti che hanno noleggi scaduti.	1/settimana
P01	Inserire nuovo impiegato nel sistema	4/anno

P02	Cambiare incarico ad un impiegato	1/mese
P03	Inserire turno di lavoro di un impiegato	1/mese
P04	Modificare turno effettivo di lavoro di un impiegato	1/settimana
P05	Rimuovere turno effettivo di lavoro di un impiegato	1/settimana
P06	Report mensile ore lavorate dagli impiegati	1/mese
P07	Report annuale ore lavorate dagli impiegati	1/anno
L01	Login	50/giorno

Costo delle operazioni

I00 - Stampa la lista dei film			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Film	E	1000	L
Costo Totale: 1000		**Frequenza:** 20/giorno	
Accessi/giorno: 20000			

I01 - Stampa la lista dei film disponibili			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Noleggio Attivo	E	6000	L
CF-NA	R	6000	L
Copia Film	E	10000	L
Disponibile	R	10000	L
Film	E	1000	L
Costo Totale: 33000		**Frequenza:** 20/giorno	
Accessi/giorno: 660000			

I02 - Stampa la lista delle copie disponibili di un film			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Copie di film	E	10	L
Disponibile	R	10	L
Film	E	1	L
Costo Totale: 21		**Frequenza:** 20/giorno	
Accessi/giorno: 420			

I03 - Aggiungi copia di film

Concetto	Costrutto	Accessi	Tipo
Film	E	1	S
Copie di film	E	1	S
Disponibile	R	1	S
Costo Totale: 6		**Frequenza:** 1/mese	
Accessi/mese: 6			

I04 - Stampa la lista dei noleggi attivi di un cliente			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Cliente	E	1	L
Noleggio - Cliente	R	360	L
Noleggio Attivo	E	360	L
Copia film - Noleggio	R	360	L
Copie di film	E	300	L
Disponibile	R	300	L
Film	E	300	L
Costo Totale: 1981		**Frequenza:** 5/giorno	
Accessi/giorno: 9905			

I05 - Aggiungi noleggio			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Noleggio Attivo	E	1	S
Noleggio - Cliente	R	1	S
Copia film - Noleggio Attivo	R	1	S
Costo Totale: 6		**Frequenza:** 20/giorno	
Accessi/giorno: 120			

I06 - Termina noleggio			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Noleggio Attivo	E	1	S
Copia film - Noleggio Attivo	R	1	S
Noleggio Terminato	E	1	S
Copia film - Noleggio	R	1	S
Costo Totale: 8		**Frequenza:** 20/giorno	
Accessi/giorno: 160			

I07 - Modifica anagrafica dei clienti			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Cliente	E	1	S
Costo Totale: 2		**Frequenza:** 1/settimana	

Accessi/settimana: 2

I08 - Registrazione cliente			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Cliente	E	1	S
Costo Totale: 2		**Frequenza:** 10/giorno	
Accessi/giorno: 20			

I09 - Report copie di film associate a noleggio scaduto			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Noleggio Terminato	E	54000	L
Copia film - Noleggio	R	5400	L
Noleggio Attivo	E	6000	
Copia film - Noleggio attivo	R	600	L
Copia di film	E	6000	L
Disponibile	R	6000	L
Film	E	600	L
Costo Totale: 78600		**Frequenza:** 1/mese	
Accessi/mese: 78600			

I10 - Report dei clienti che hanno noleggi scaduti			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Noleggio	E	6000	L
Noleggio - cliente	R	6000	L
Cliente	E	3000	L
Costo Totale: 15000		**Frequenza:** 1/settimana	
Accessi/settimana: 15000			

P01 - Inserire nuovo impiegato nel sistema			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Impiegato	E	1	S
Impiegato - incarico corrente	R	1	S
Incarico corrente	E	1	S
Costo Totale: 6		**Frequenza:** 4/anno	
Accessi/anno: 24			

P02 - Cambiare incarico ad un impiegato			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Incarico Corrente	E	1	S
Incarico Passato	E	1	S

Impiegato - Incarico	R	1	S
Impiegato - Incarico Corrente	R	1	S
Costo Totale: 8		**Frequenza:** 1/mese	
Accessi/mese: 8			

P03 - Inserire turno di lavoro a un impiegato			
Concetto	**Costrutto**	**Accessi**	**Tipo**
TE-I	R	1	S
Turno Effettivo	E	1	S
T-TE	R	1	S
Costo Totale: 6		**Frequenza:** 1/mese	
Accessi/mese: 6			

P04 - Modificare turno effettivo di lavoro			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Turno Effettivo	E	1	S
TE-I	R	1	S
T-TE	R	1	S
Costo Totale: 6		**Frequenza:** 1/settimana	
Accessi/settimana: 6			

P05 - Rimuovere turno effettivo di lavoro			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Impiegato	E	1	L
TE-I	R	1	S
Costo Totale: 3		**Frequenza:** 1/settimana	
Accessi/settimana: 3			

P06 - Report mensile sulle ore lavorate dagli impiegati			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Impiegato	E	12	L
TE-I	R	312	L
Turno Effettivo	E	150	L
TE-T	R	150	L
Turno	E	4	L
Costo Totale: 952		**Frequenza:** 1/mese	
Accessi/mese: 952			

P07 - Report annuale sulle ore lavorate dagli impiegati

Concetto	Costrutto	Accessi	Tipo
Impiegato	E	19	L
TE-I	R	5947	L
Turno Effettivo	E	2973	L
TE-T	R	2973	L
Turno	E	4	L
Costo Totale: 11916		**Frequenza:** 1/anno	
Accessi/anno: 11916			

L01 - Login			
Concetto	**Costrutto**	**Accessi**	**Tipo**
Login	E	1	L
Costo Totale: 50		**Frequenza:** 50/giorno	
Accessi/giorno: 50			

Ristrutturazione dello schema E-R

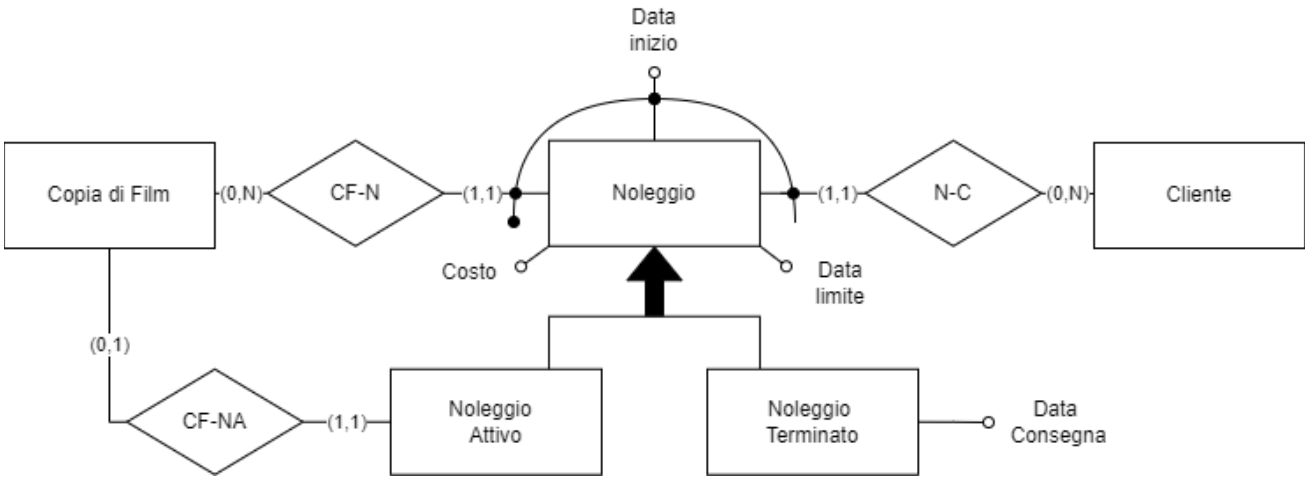
1 - Analisi delle ridondanze

Non sono presenti ridondanze.

2 - Eliminazione delle generalizzazioni

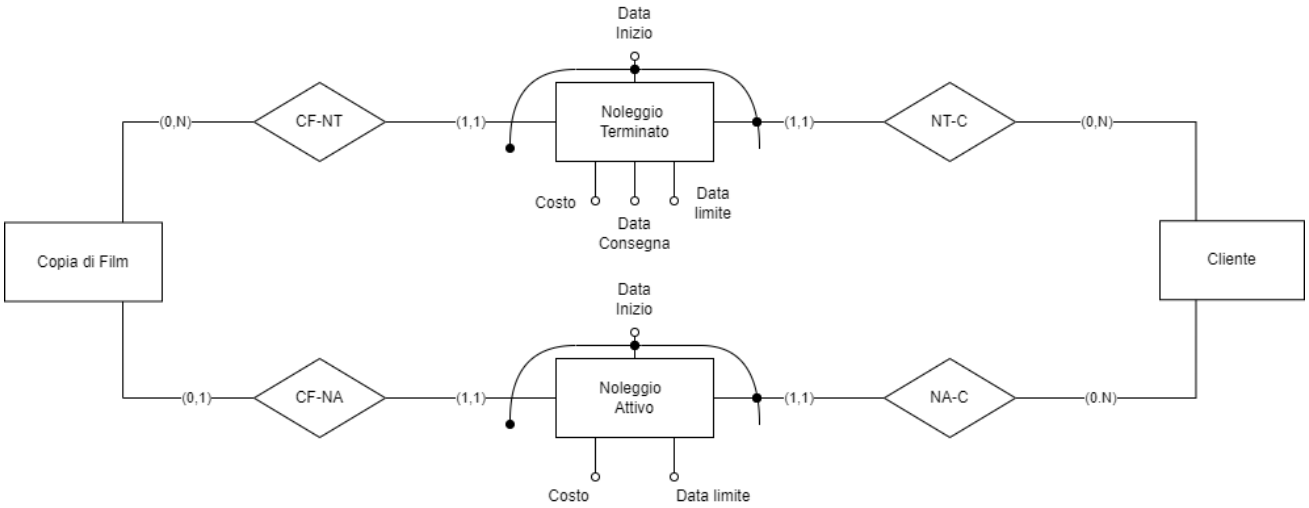
Vanno eliminate le generalizzazioni di noleggio e di incarico presenti nello schema e dovuti ai pattern di storicizzazione.

Viene considerata inizialmente la generalizzazione di noleggio che è debole e include le seguenti entità/relazioni:

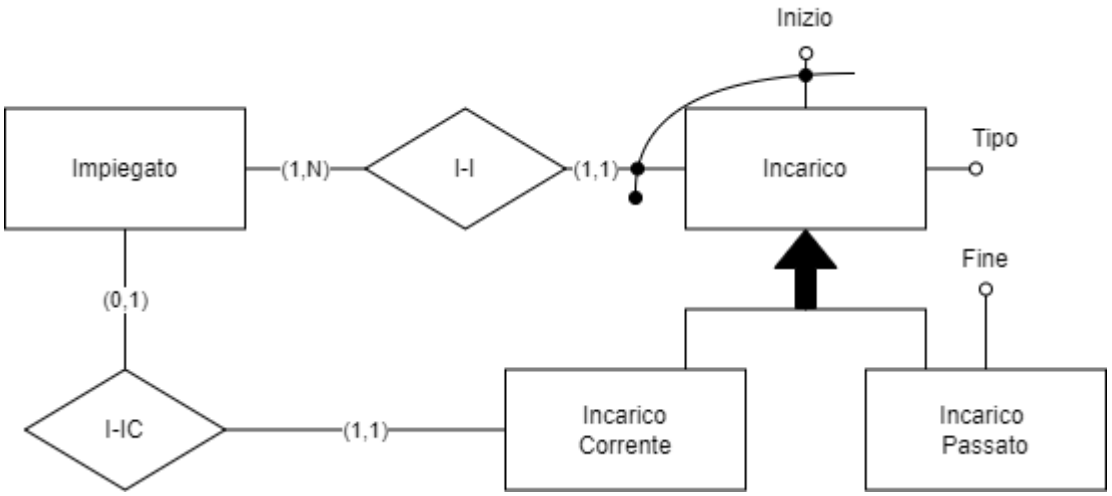


Da questa la generalizzazione può essere eliminata in due modi: accorpando i figli nel genitore o accorpando il genitore nei figli. Le principali differenze che si evincono sono che nella prima scelta

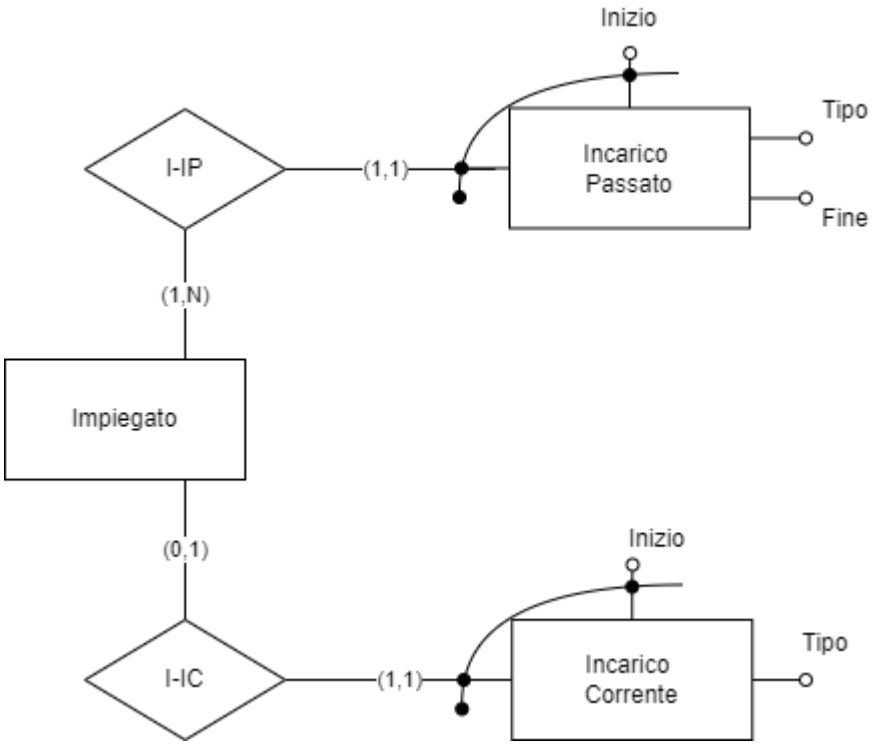
ci sarà solamente una tabella nel database e quindi vi sarà conseguentemente un solo accesso, ma d'altra parte dovrà essere inserito un attributo per differenziare il tipo di noleggio e inoltre ci sarà uno spreco maggiore di memoria dovuto alla presenza di valori null(attributo mancante nel noleggio attivo). La seconda scelta invece permette di avere un risparmio considerevole della memoria dovuto all' assenza di valori null ma un numero maggiori di accessi poiché vi saranno due tabelle separate per le due tipologie di noleggio ed è più semplice fare la distinzione di operazioni che afferiscono solamente ad una particolare tipologia di noleggio. A seguito di questa analisi la scelta che viene presa è quella di eliminare la generalizzazione e di accorpare il genitore nei figli. Lo schema è quello riportato di seguito.



Mentre per quanto riguarda la generalizzazione incarico vi è una simile situazione della precedente in quanto si ricade nella stessa casistica.

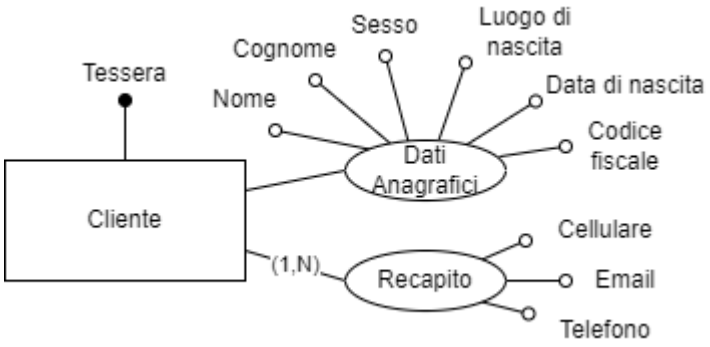


E' possibile quindi dividere completamente i due concetti utilizzando la stessa analisi precedentemente esplicitata. Lo schema viene tradotto nel seguente modo.

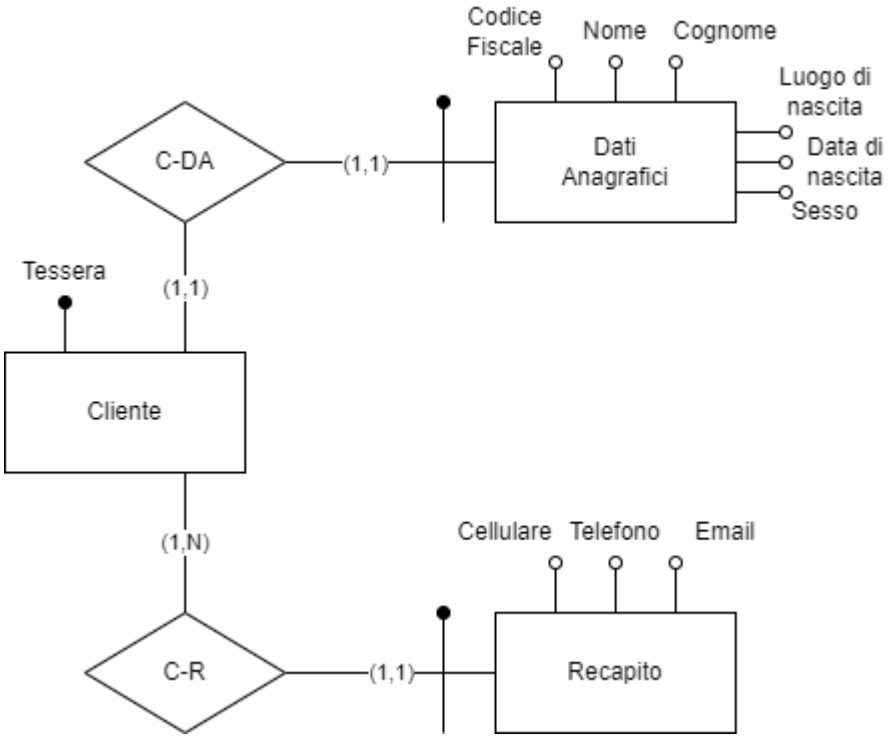


2.1 - Partizionamento di entità

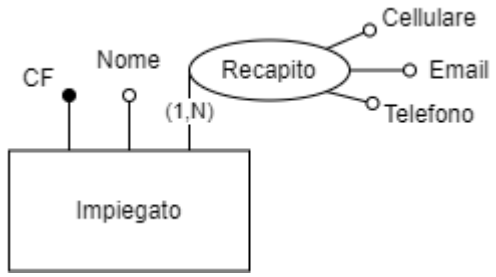
In questa parte verrà trattata la decomposizione verticale di due delle entità cardine del progetto: l'impiegato e il cliente. Entrambe infatti presentano attributi multivalore. Prendendo in considerazione l'entità cliente e i suoi attributi multivalore:



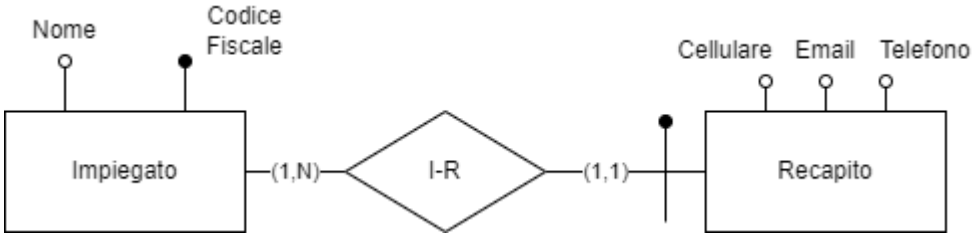
Questi due attributi multivalore che verranno divisi nel seguente modo:



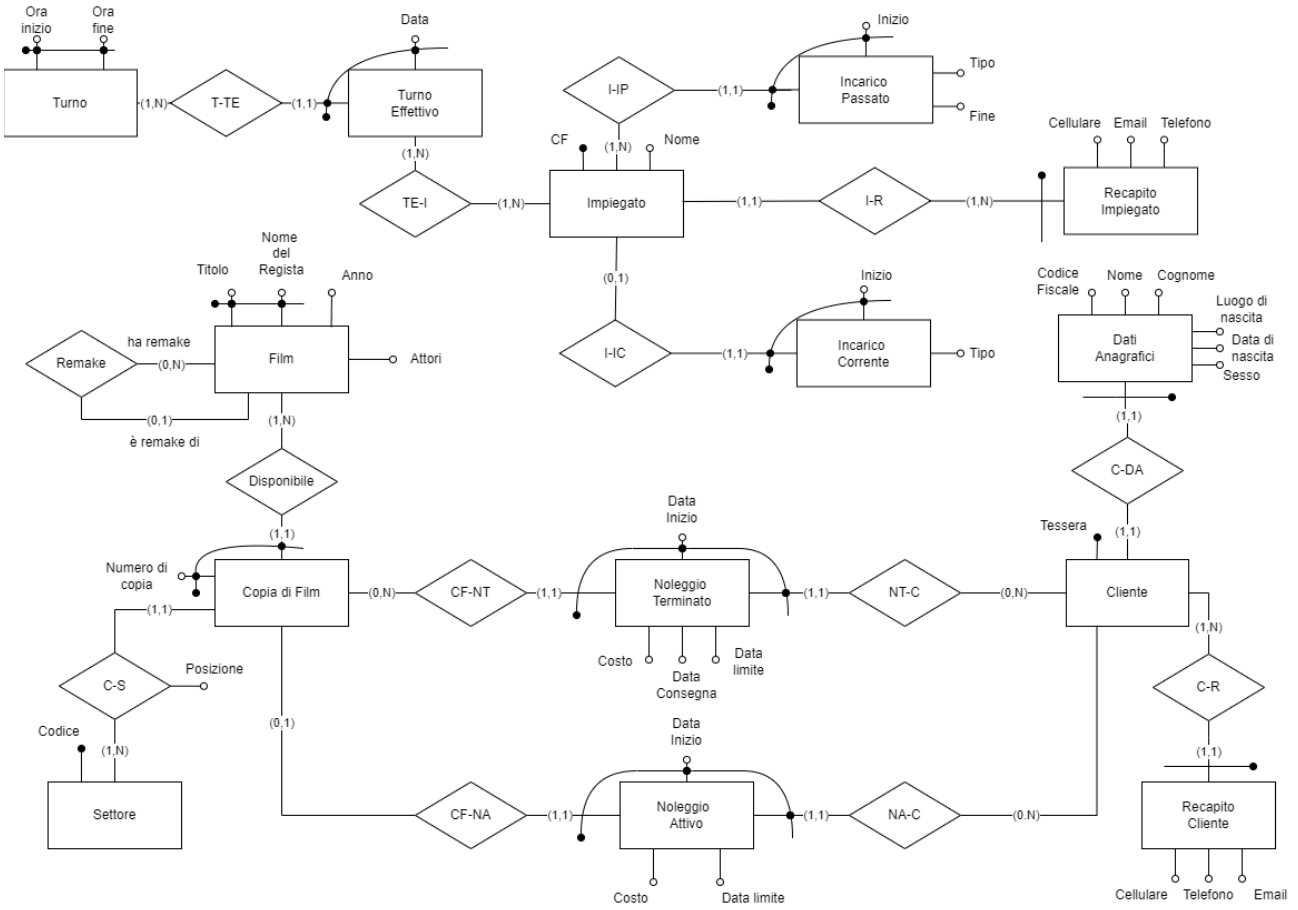
Analogamente viene analizzata l'entità impiegato che si presenta nel seguente modo:



che allo stesso modo viene decomposta verticalmente.



a seguito di questi cambiamenti viene riportato lo schema ristrutturato.



3 - Scelta degli identificatori primari

Entità	**Identificatore**
Film	Titolo, nome del regista.
Copia di Film	Titolo, nome del regista, numero di copia.
Settore	Codice.
Noleggio Terminato	Titolo, nome del regista, numero di copia, data inizio, tessera.
Noleggio Attivo	Titolo, nome del regista, numero di copia, data inizio, tessera.
Cliente	Tessera.
Dati Anagrafici	Tessera.
Recapito Cliente	Tessera.
Impiegato	Codice fiscale.

Incarico Corrente	Codice fiscale, inizio.
Incarico Passato	Codice fiscale, inizio.
Recapito Impiegato	Codice fiscale.
Turno Effettivo	Ora inizio, ora fine, data.
Turno	Ora inizio, ora fine.

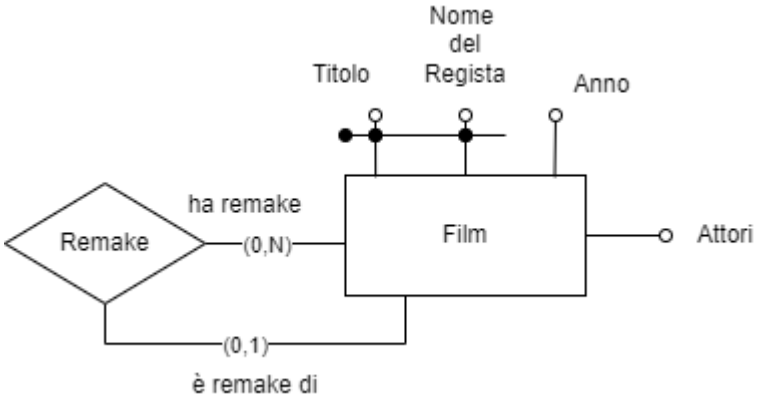
Trasformazione di attributi e identificatori

(TODO)

Traduzione di entità e associazioni

Viene considerato lo schema E-R precedente e step by step tradotto nel modello relazionale.

L'analisi si svolge partendo con l'entità film e la sua relazione ricorsiva:



Questo schema si traduce nelle due relazioni:

FILM(Titolo, NomeRegista, Anno, Attori)

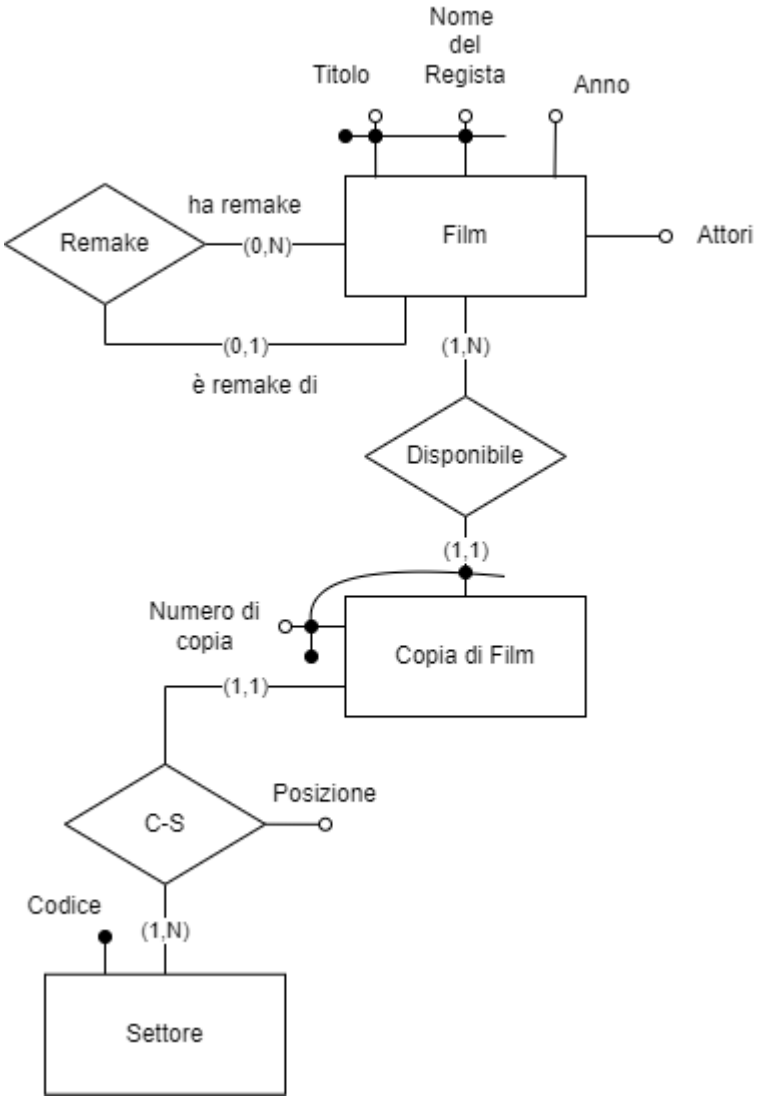
REMAKE(TitoloOriginale, NomeRegistaOriginale, TitoloRemake, NomeRegistaRemake)

Con vincoli:

REMAKE(TitoloOriginale,RegistaOriginale) ⊆ **FILM**(Titolo, NomeRegista)

REMAKE(TitoloRemake,RegistaRemake) ⊆ **FILM**(Titolo, NomeRegista)

Continuando viene presa in analisi la seguente porzione di schema.



Che viene tradotta nel seguente modo:

FILM(Titolo, Regista, Anno, Attori)

REMAKE(TitoloOriginale, RegistaOriginale, TitoloRemake, RegistaRemake)

COPIA DI FILM(TitoloFilm, RegistaFilm, Numero)

COPIA FILM - SETTORE(TitoloCopia, RegistaCopia, NumeroCopia, Settore, Posizione)

SETTORE(Codice)

Con i vincoli:

REMAKE(TitoloOriginale,RegistaOriginale) ⊆ **FILM**(Titolo, NomeRegista)

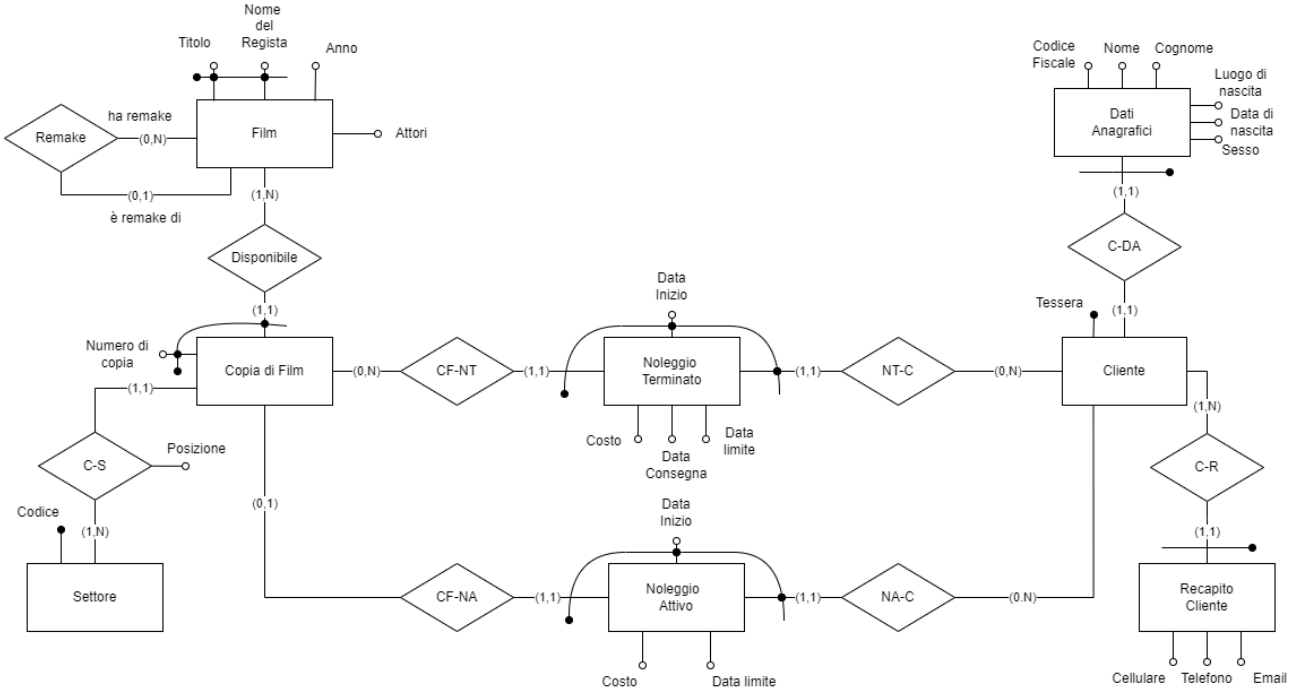
REMAKE(TitoloRemake,RegistaRemake) ⊆ **FILM**(Titolo, NomeRegista)

COPIA DI FILM(TitoloFilm,RegistaFilm) ⊆ **FILM**(Titolo, NomeRegista)

COPIA FILM - SETTORE(TitoloCopia, RegistaCopia, NumeroCopia) ⊆ **COPIA DI FILM**(TitoloFilm, RegistaFilm, Numero)

COPIA FILM - SETTORE(Settore) ⊆ **SETTORE**(Codice)

L'analisi ora passa alla porzione di schema che includono il noleggio e il cliente:



Il noleggio terminato è composto da due associazioni uno a molti, d'altra parte il noleggio attivo è composto da un'associazione uno ad uno e da un'associazione uno a molti. Il cliente invece ha due .

le quali vengono tradotte nel seguente modo:

FILM(Titolo, Regista, Anno, Attori)

REMAKE(TitoloOriginale, RegistaOriginale, TitoloRemake, RegistaRemake)

COPIA DI FILM(TitoloFilm, RegistaFilm, Numero)

COPIA FILM - SETTORE(TitoloCopia, RegistaCopia, NumeroCopia, Settore, Posizione)

SETTORE(Codice)

NOLEGGIO TERMINATO (TitoloCopia, RegistaCopia, NumeroCopia, DataInizio,TesseraCliente, Costo, DataConsegna, DataLimite)

NOLEGGIO ATTIVO(TitoloCopia, RegistaCopia, NumeroCopia, DataInizio, TesseraCliente, Costo, DataLimite)

CLIENTE(Tessera)

RECAPITO CLIENTE(TesseraCliente, Cellulare, Telefono, Email)

DATI ANAGRAFICI(TesseraCliente, CF, Nome, Cognome, Sesso, LuogoNascita, DataNascita)

Con vincoli:

REMAKE(TitoloOriginale,RegistaOriginale) ⊆ **FILM**(Titolo, NomeRegista)

REMAKE(TitoloRemake,RegistaRemake) ⊆ **FILM**(Titolo, NomeRegista)

COPIA DI FILM(TitoloFilm,RegistaFilm) ⊆ **FILM**(Titolo, NomeRegista)

COPIA FILM - SETTORE(TitoloCopia, RegistaCopia, NumeroCopia) ⊆ **COPIA DI FILM**(TitoloFilm, RegistaFilm, Numero)

COPIA FILM - SETTORE(Settore) ⊆ **SETTORE**(Codice)

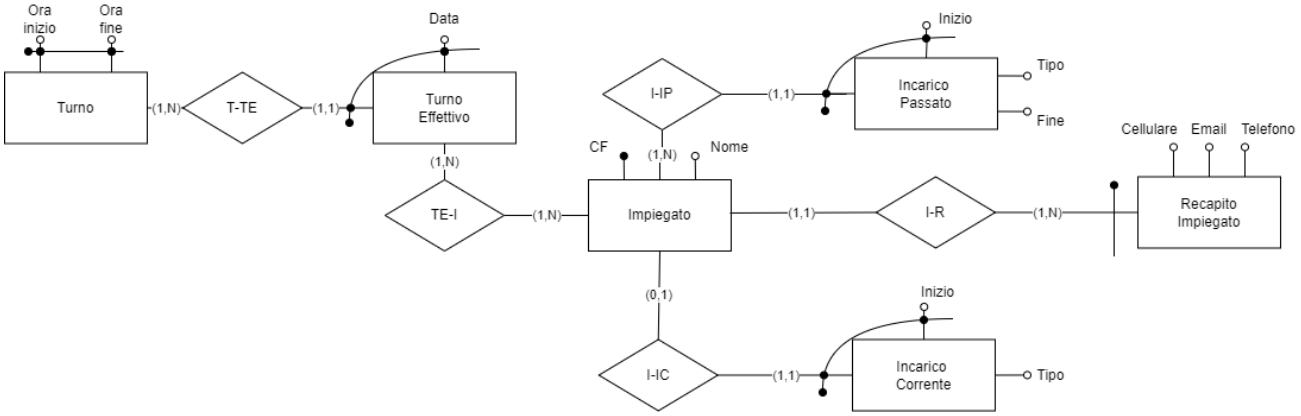
NOLEGGIO TERMINATO(TitoloCopia, RegistaCopia, NumeroCopia) ⊆ **COPIA DI FILM**(TitoloFilm, RegistaFilm, Numero)

NOLEGGIO ATTIVO(TitoloCopia, RegistaCopia, NumeroCopia) ⊆ **COPIA DI FILM**(TitoloFilm, RegistaFilm, Numero)

RECAPITO CLIENTE(TesseraCliente) ⊆ **CLIENTE**(Tessera)

DATI ANAGRAFICI(TesseraCliente) ⊆ **CLIENTE**(Tessera)

Infine viene analizzato il concetto di impiegato con gli incarichi e i turni:



che viene tradotto nel seguente modo:

IMPIEGATO(<u>CF</u>, Nome)

RECAPITO IMPIEGATO(<u>CFI</u>, Cellulare, Email, Telefono)

INCARICO CORRENTE(<u>CFI, Inizio,</u> Tipo)

INCARICO PASSATO(<u>CFI, Inizio</u>, Tipo, Fine)

TURNO(<u>OraInizio, OraFine</u>)

TURNO EFFETTIVO(<u>InizioTurno, FineTurno, Data)</u>

TE-I(<u>InizioEffettivo,FineEffettiva,DataEffettiva</u>, <u>CFI</u>)

Con vincoli:

RECAPITO IMPIEGATO(CFI) ⊆ **IMPIEGATO**(CF)

INCARICO CORRENTE(CFI) ⊆ **IMPIEGATO**(CF)

INCARICO PASSATO(CFI) ⊆ **IMPIEGATO**(CF)

TURNO EFFETTIVO(InizioTurno, FineTurno) ⊆ **TURNO**(OraInizio, OraFine)

TE-I (InizioEffettivo, FineEffettiva, DataEffettiva) ⊆ **TURNO EFFETTIVO** (InizioTurno, FineTurno, Data)

TE-I (CFI) ⊆ **IMPIEGATO** (CF)

Normalizzazione del modello relazionale

Poiché non ci sono dipendenze funzionali non banali, allora le tabelle sono in terza forma normale.

5. Progettazione fisica

Utenti e privilegi

Si prevedono 3 ruoli, per implementare il principle of least privilege:

Ruolo	**Grant in esecuzione**
Impiegato	I00, I01, I02, I03, I04, I05, I06, I07, I08, I09, I10, I11, I12.
Proprietario del negozio	P01, P02, P03, P04, P05.
Login	L1.

Per identificare gli utenti si introduce una tabella Utenti per mantenere le credenziali.

Strutture di memorizzazione

Tabella <utenti>		
Colonna	**Tipo di dato**	**Attributi[2]**
username	VARCHAR(45)	PK, NN
password	CHAR(30)	NN
ruolo	ENUM('impiegato', 'proprietario')	NN

Tabella <film>		
Colonna	**Tipo di dato**	**Attributi[2]**
titolo	VARCHAR(45)	PK, NN
nome_regista	VARCHAR(45)	PK, NN
anno	YEAR(4)	NN
attori	TINYTEXT	NN

Tabella <remake>		
Colonna	**Tipo di dato**	**Attributi[2]**
titolo_originale	VARCHAR(45)	PK,NN
regista_originale	VARCHAR(45)	PK, NN
titolo_remake	VARCHAR(45)	PK, NN
regista_remake	VARCHAR(45)	PK, NN

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

Tabella <copie_film>		
Colonna	**Tipo di dato**	**Attributi[2]**
titolo_film	VARCHAR(45)	PK, NN
regista_film	VARCHAR(45)	PK,NN
numero	INT	PK,NN

Tabella <settore>		
Colonna	**Tipo di dato**	**Attributi[2]**
codice	VARCHAR(45)	PK, NN

Tabella <copie_film_settore>		
Colonna	**Tipo di dato**	**Attributi[2]**
titolo_copia	VARCHAR(45)	PK,NN
regista_copia	VARCHAR(45)	PK,NN
numero_copia	INT	PK,NN
settore	VARCHAR(45)	NN
posizione	TINYTEXT	NN

Tabella <cliente>		
Colonna	**Tipo di dato**	**Attributi[2]**
tessera	VARCHAR(30)	PK,NN,AI

Tabella <noleggio_attivo>		
Colonna	**Tipo di dato**	**Attributi[2]**
titolo_copia	VARCHAR(45)	PK, NN
regista_copia	VARCHAR(45)	PK,NN
numero_copia	INT	PK,NN
data_inizio	DATE	PK,NN
tessera_cliente	TINYINT	PK,NN
costo	DECIMAL(5,2)	NN
data_limite	DATE	NN

Tabella <noleggio_terminato>		
Colonna	**Tipo di dato**	**Attributi[2]**
titolo_copia	VARCHAR(45)	PK, NN
regista_copia	VARCHAR(45)	PK,NN

numero_copia	INT	PK,NN
data_inizio	DATE	PK,NN
tessera_cliente	TINYINT	PK,NN
costo	FLOAT	NN
data_limite	DATE	NN
data_consegna	DATE	NN

Tabella <recapito_cliente>		
Colonna	**Tipo di dato**	**Attributi[2]**
tessera_cliente	VARCHAR(30)	PK,NN
cellulare	VARCHAR(30)	UQ,NN
telefono	VARHCAR(30)	UQ,NN
email	VARCHAR(45)	UQ,NN

Tabella <dati_anagrafici>		
Colonna	**Tipo di dato**	**Attributi[2]**
tessera_cliente	INT	PK,NN
cf	VARCHAR(45)	UQ,NN
nome	VARCHAR(45)	NN
cognome	VARCHAR(45)	NN
sesso	ENUM('M','F')	NN
luogo_nascita	VARCHAR(45)	NN
data_nascita	DATE	NN

Tabella <impiegato>		
Colonna	**Tipo di dato**	**Attributi[2]**
cf	VARCHAR(45)	PK,NN
nome	VARCHAR(45)	NN

Tabella <recapito_impiegato>		
Colonna	**Tipo di dato**	**Attributi[2]**
cf_impiegato	VARCHAR(45)	PK,NN
cellulare	VARCHAR(30)	UQ,NN
telefono	VARHCAR(30)	UQ,NN
email	VARCHAR(45)	UQ,NN

Tabella <incarico_corrente>

Colonna	Tipo di dato	Attributi[2]
cf_impiegato	VARCHAR(45)	PK,NN
tipo	ENUM('cassiere','magazziniere','commesso')	NN
inizio	DATE	NN

Tabella <incarico_passato>		
Colonna	**Tipo di dato**	**Attributi[2]**
cf_impiegato	VARCHAR(45)	PK,NN
tipo	ENUM('cassiere','magazziniere','commesso')	NN
inizio	DATE	NN
fine	DATE	NN

Tabella <turno>		
Colonna	**Tipo di dato**	**Attributi[2]**
ora_inizio	INT	PK,NN
ora_fine	INT	PK,NN

Tabella <turno_effettivo>		
Colonna	**Tipo di dato**	**Attributi[2]**
inizio_turno	INT	PK,NN
fine_turno	INT	PK,NN
data_turno	DATE	PK,NN

Tabella <turno_impiegato>		
Colonna	**Tipo di dato**	**Attributi[2]**
inizio_effettivo	INT	NN
fine_effettiva	INT	NN
data_effettiva	DATE	PK,NN
cf_impiegato	VARCHAR(45)	PK,NN

Indici

Poiché non vengono aggiunti indici questa sezione non viene compilata.

Trigger

```
--
-- TRIGGER before_add_rental
--
CREATE TRIGGER before_add_rental
BEFORE INSERT ON noleggio_attivo
FOR EACH ROW
BEGIN
        DECLARE existing_rentals INT;
   -- Conta quante copie dello stesso film l'utente ha già noleggiato
   SELECT COUNT(*) INTO existing_rentals
   FROM noleggio_attivo
   WHERE tessera_cliente = NEW.tessera_cliente
   AND titolo_copia = NEW.titolo_copia
   AND regista_copia = NEW.regista_copia;
   -- Se l'utente ha già noleggiato una copia uguale, genera un errore
   IF existing_rentals > 0 THEN
      SIGNAL SQLSTATE 'TR000'
      SET MESSAGE_TEXT = 'Questo utente ha già noleggiato una copia di questo film.';
   END IF;
END!


--
-- TRIGGER before_mod_worker
--
CREATE TRIGGER before_mod_worker
BEFORE INSERT ON incarico_passato
FOR EACH ROW
BEGIN
   DECLARE cf_count INT;

   -- Conta quante volte il CF dell'impiegato appare nella tabella
```

```
    SELECT COUNT(*) INTO cf_count
    FROM incarico_passato
    WHERE cf_impiegato = NEW.cf_impiegato
    AND INizio <= NEW.fine
    AND fine >= NEW.INizio;

    -- Se il CF appare più di una volta nello stesso giorno, genera un errore
    IF cf_count > 0 THEN
        SIGNAL SQLSTATE 'TRP02'
            SET MESSAGE_TEXT = 'Impossibile cambiare ruolo due volte allo stesso impiegato nello
stesso giorno.';
    END IF;
END!



--
-- TRIGGER prevent_duplicate_turn
--
CREATE TRIGGER prevent_duplicate_turn
BEFORE INSERT ON turno_impiegato
FOR EACH ROW
BEGIN
   DECLARE turno_count INT;

    -- Conta quante volte lo stesso CF_impiegato ha già un turno nello stesso giorno
    SELECT COUNT(*) INTO turno_count
    FROM turno_impiegato
    WHERE cf_impiegato = NEW.cf_impiegato
    AND data_effettiva = NEW.data_effettiva;

    -- Se il CF_impiegato ha già un turno nello stesso giorno, genera un errore
    IF turno_count > 0 THEN
        SIGNAL SQLSTATE 'TRP04'
```

```
        SET MESSAGE_TEXT = 'Impossibile inserire più di un turno per lo stesso impiegato nello stesso giorno.';
    END IF;
END!

--
-- TRIGGER elimina_turno_delete
--
CREATE TRIGGER elimina_turno_delete
AFTER DELETE ON turno_impiegato
FOR EACH ROW
BEGIN
   DECLARE var_inizio INT;
    DECLARE var_fine INT;
    DECLARE var_data DATE;
    DECLARE count_impiegati INT;

    -- Estrai i dati di inizio, fine e data dall'aggiornamento
    SET var_inizio = OLD.inizio_effettivo;
    SET var_fine = OLD.fine_effettiva;
    SET var_data = OLD.data_effettiva;

    -- Conta quanti impiegati hanno ancora il turno per quella data e ore
    SELECT COUNT(*) INTO count_impiegati
    FROM turno_impiegato
    WHERE inizio_effettivo = var_inizio
    AND fine_effettiva = var_fine
    AND data_effettiva = var_data;

    -- Se nessun impiegato ha più il turno, rimuovi l'entry da turno_effettivo
    IF count_impiegati = 0 THEN
        DELETE FROM turno_effettivo
        WHERE inizio_turno = var_inizio
```

```
        AND fine_turno = var_fine
        AND data_turno = var_data;
    END IF;
END!

--
-- TRIGGER elimina_turno_update
--
CREATE TRIGGER elimina_turno_update
AFTER UPDATE ON turno_impiegato FOR EACH ROW
BEGIN
    DECLARE var_inizio INT;
    DECLARE var_fine INT;
    DECLARE var_data DATE;
    DECLARE count_impiegati INT;

    -- Estrai i dati di inizio, fine e data dall'aggiornamento
    SET var_inizio = OLD.inizio_effettivo;
    SET var_fine = OLD.fine_effettiva;
    SET var_data = OLD.data_effettiva;

    -- Conta quanti impiegati hanno ancora il turno per quella data e ore
    SELECT COUNT(*) INTO count_impiegati
    FROM turno_impiegato
    WHERE inizio_effettivo = var_inizio
    AND fine_effettiva = var_fine
    AND data_effettiva = var_data;

    -- Se nessun impiegato ha più il turno, rimuovi l'entry da turno_effettivo
    IF count_impiegati = 0 THEN
        DELETE FROM turno_effettivo
        WHERE inizio_turno = var_inizio
        AND fine_turno = var_fine
```

```
      AND data_turno = var_data;
   END IF;
END!
```

Eventi

Non sono stati implementati eventi.

Viste

Non sono stati implementate Viste.

Stored Procedures e transazioni

```
--
-- PROCEDURE login
--
CREATE PROCEDURE login (IN var_username VARCHAR(45), IN var_password CHAR(40))
BEGIN
         SELECT ruolo
   FROM  utenti
   WHERE username=var_username AND pwd=SHA1(var_password);
END!
GRANT EXECUTE ON PROCEDURE login TO utente_login!

--
-- PROCEDURE I00  film_list
--
CREATE PROCEDURE film_list()
BEGIN
   SELECT *
   FROM Film ;
END!
GRANT EXECUTE ON PROCEDURE film_list TO impiegato!
```

```
--
-- PROCEDURE I01  available_film_list
--
CREATE PROCEDURE available_film_list()
BEGIN
   SELECT DISTINCT F.titolo, F.regista, F.anno, F.attori
   FROM Film AS F
   JOIN Copia_Film AS CF ON F.titolo = CF.titolo_film AND F.regista = CF.regista_film
    LEFT JOIN Noleggio_Attivo AS NA ON CF.titolo_film = NA.titolo_copia AND CF.regista_film
= NA.regista_copia AND CF.numero = NA.numero_copia
   WHERE NA.titolo_copia IS NULL;
END!
GRANT EXECUTE ON PROCEDURE available_film_list TO impiegato!

--
-- PROCEDURE I02  available_copy_list
--
CREATE PROCEDURE available_copy_list(IN var_titolo VARCHAR(55), IN var_regista
VARCHAR(45))
BEGIN
        DECLARE custom_error CONDITION FOR SQLSTATE 'I0200';
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
                ROLLBACK;  -- Esegui il rollback in caso di errore
      RESIGNAL;
   END;
        IF EXISTS (SELECT titolo,regista FROM film WHERE titolo = var_titolo and regista =
var_regista) THEN
                SELECT CF.titolo_film, CF.regista_film, CF.numero
                FROM copia_film CF
                LEFT JOIN noleggio_attivo NA ON CF.titolo_film = NA.titolo_copia AND
CF.regista_film = NA.regista_copia AND CF.numero = NA.numero_copia
```

```
        WHERE NA.titolo_copia IS NULL AND CF.titolo_film = var_titolo AND CF.regista_film = var_regista;
    ELSE
        SIGNAL custom_error SET MESSAGE_TEXT = 'Il film che hai scelto potrebbe essere errato o non esistente, perfavore riprova';
    END IF;
END!
GRANT EXECUTE ON PROCEDURE available_copy_list TO impiegato!

--
-- PROCEDURE I03  add_copy
--
CREATE PROCEDURE add_copy(IN var_titolo VARCHAR(55), IN var_regista VARCHAR(45), IN var_num VARCHAR(10))
BEGIN
    DECLARE count_film INT;
  DECLARE i INT DEFAULT 0;
  DECLARE last_num int;
    DECLARE var_num_as_int INT;
  DECLARE custom_error CONDITION FOR SQLSTATE 'I0300';
    DECLARE EXIT HANDLER FOR SQLEXCEPTION
  BEGIN
        ROLLBACK;  -- Esegui il rollback in caso di errore
    RESIGNAL;
  END;
  SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
    START TRANSACTION;
        IF var_num REGEXP '^[0-9]{1,2}$' THEN
        SET var_num_as_int = CAST(var_num AS SIGNED);
            SELECT COUNT(*) INTO count_film
            FROM film
            WHERE titolo = var_titolo AND regista = var_regista;
            IF count_film = 0 THEN
```

```
                SIGNAL custom_error SET MESSAGE_TEXT = 'Il film a cui hai provato ad aggiungere copie potrebbe essere errato o non esistente, perfavore riprova';
            ELSE
                SELECT MAX(numero) INTO last_num
                FROM copia_film
                WHERE titolo_film = var_titolo AND regista_film = var_regista;

                WHILE i < var_num DO
                    SET last_num = last_num + 1;

                    -- Inserisci una nuova copia nella tabella copia_film
                    INSERT INTO copia_film (titolo_film, regista_film, numero)
                    VALUES (var_titolo, var_regista, last_num);

                    SET i = i + 1;
                END WHILE;
            END IF;
        ELSE SIGNAL custom_error SET MESSAGE_TEXT = 'Invalid input: il numero di copie deve essere un intero non maggiore di 99';
        END IF;
    COMMIT;
END!
GRANT EXECUTE ON PROCEDURE add_copy TO impiegato!


--
-- PROCEDURE I04  available_rents
--
CREATE PROCEDURE available_rents(IN var_tessera varchar(30))
BEGIN
    DECLARE custom_error CONDITION FOR SQLSTATE 'I0400';
    DECLARE EXIT HANDLER FOR SQLEXCEPTION
  BEGIN
```

```
            ROLLBACK;  -- Esegui il rollback in caso di errore
      RESIGNAL;
   END;
        IF EXISTS (SELECT tessera FROM cliente WHERE tessera = var_tessera) THEN
            SELECT *
            FROM noleggio_attivo
            WHERE tessera_cliente = var_tessera;
        ELSE SIGNAL custom_error SET MESSAGE_TEXT = 'Tessera cliente non disponibile o
errata';
   END IF;
END!
GRANT EXECUTE ON PROCEDURE available_rents TO impiegato!

--
-- PROCEDURE I05  add_rental
--
CREATE PROCEDURE add_rental(IN var_titolo VARCHAR(55), IN var_regista
VARCHAR(45),IN var_tessera varchar(30),IN var_costo VARCHAR(10))
BEGIN
        DECLARE num_expired INT;
        DECLARE first_num INT;
   DECLARE var_data_inizio DATE;
   DECLARE var_data_limite DATE;
   DECLARE custom_error CONDITION FOR SQLSTATE 'I0500';
   DECLARE decimal_costo DECIMAL(5,2);
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
            ROLLBACK;  -- Esegui il rollback in caso di errore
      RESIGNAL;
   END;
        SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
        START TRANSACTION;
            -- controllo su prezzo  --
```

```
IF var_costo REGEXP '^[0-9]{1,3}(\\.[0-9]{1,2})?$' THEN
SET decimal_costo = CAST(var_costo AS DECIMAL(5, 2));
    -- controllo su tessera cliente --
    IF EXISTS (SELECT tessera FROM cliente WHERE tessera = var_tessera) THEN
        -- controllo su esistenza di copia di film --
        IF EXISTS (SELECT * FROM copia_film WHERE titolo_film = var_titolo AND regista_film = var_regista) THEN
    -- controllo se cliente ha noleggio scaduto --
            -- controllo se utente ha noleggio scaduto --
            IF NOT EXISTS(SELECT * FROM noleggio_attivo WHERE tessera_cliente = var_tessera AND data_limite < CURDATE()) THEN
                -- prendi la prima copia disponibile che non fa parte di un noleggio attivo --
                SELECT MIN(numero) INTO first_num
                FROM copia_film
                WHERE titolo_film = var_titolo AND regista_film = var_regista
                AND (titolo_film, regista_film, numero) NOT IN (
                    SELECT titolo_copia, regista_copia, numero_copia
                    FROM noleggio_attivo);
                IF first_num IS NOT NULL THEN
                    -- inizo noleggio --
                    SELECT CURDATE() INTO var_data_inizio;
                    -- fine noleggio --
                    SELECT DATE_ADD(CURDATE(), INTERVAL 1 WEEK) INTO var_data_limite;
                    INSERT INTO noleggio_attivo(titolo_copia,regista_copia,numero_copia,data_inizio,tessera_cliente,costo,data_limite)
                    VALUES (var_titolo,var_regista,first_num,var_data_inizio,var_tessera,decimal_costo,var_data_limite);
```

```
                                        ELSE SIGNAL custom_error SET MESSAGE_TEXT
= 'Questo film non ha copie disponibili per essere noleggiate';
                                        END IF;
                                ELSE
                                        SIGNAL SQLSTATE '45000'
                                                SET MESSAGE_TEXT = 'L\'utente ha già
noleggi scaduti';
                                END IF;
                        ELSE SIGNAL custom_error
                                SET MESSAGE_TEXT = 'Nessuna copia disponibile per il
film';
                        END IF;
                ELSE SIGNAL custom_error
                        SET MESSAGE_TEXT = 'Tessera cliente errata o non esistente';
                END IF;
        ELSE SIGNAL
                custom_error SET MESSAGE_TEXT = 'Prezzo non valido';
        END IF;
   COMMIT;
END!
GRANT EXECUTE ON PROCEDURE add_rental TO impiegato!


--
-- PROCEDURE I06  end_rental
--
CREATE PROCEDURE end_rental(IN var_titolo VARCHAR(55), IN var_regista
VARCHAR(45),IN var_tessera varchar(30), IN var_data_inizio VARCHAR(20))
BEGIN
        DECLARE var_end_rental DATE;
        DECLARE var_data_date DATE;
   DECLARE custom_error CONDITION FOR SQLSTATE 'I0600';
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
```

```
BEGIN
        ROLLBACK;  -- Esegui il rollback in caso di errore
   RESIGNAL;
END;
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
    START TRANSACTION;
    -- controllo su tessera cliente --
    IF EXISTS (SELECT tessera FROM cliente WHERE tessera = var_tessera) THEN
        -- controllo su data --
        IF(CAST(var_data_inizio AS DATE)) IS NOT NULL THEN
            SET var_data_date = CAST(var_data_inizio AS DATE);
            -- controllo copia film che sta in un noleggio attivo --
            IF EXISTS (SELECT * FROM noleggio_attivo WHERE titolo_copia = var_titolo AND regista_copia = var_regista AND tessera_cliente = var_tessera AND data_inizio = var_data_date) THEN
                SET var_end_rental = CURDATE();
                INSERT INTO noleggio_terminato (titolo_copia, regista_copia, numero_copia, data_inizio, tessera_cliente, costo, data_limite, data_consegna)
                SELECT titolo_copia,regista_copia, numero_copia, data_inizio, tessera_cliente, costo, data_limite, var_end_rental
                FROM noleggio_attivo
        WHERE titolo_copia = var_titolo AND regista_copia = var_regista AND tessera_cliente = var_tessera AND data_inizio = var_data_date;

                DELETE FROM noleggio_attivo
                WHERE titolo_copia = var_titolo
                AND regista_copia = var_regista
                AND data_inizio = var_data_inizio
                AND tessera_cliente = var_tessera;

            ELSE SIGNAL custom_error SET MESSAGE_TEXT = 'Copia film inesistente o non correntemente attiva in un noleggio';
            END IF;
```

```
            ELSE SIGNAL custom_error SET MESSAGE_TEXT = 'Data errata per favore riprova';
            END IF;
        ELSE SIGNAL custom_error SET MESSAGE_TEXT = 'Tessera cliente errata o non esistente';
        END IF;
   COMMIT;
END!
GRANT EXECUTE ON PROCEDURE end_rental TO impiegato!

--
-- PROCEDURE I07  mod_customer_registry
--
CREATE PROCEDURE mod_customer_registry(IN var_tessera VARCHAR(45) ,IN var_cf VARCHAR(45), IN var_nome VARCHAR(45), IN var_cognome VARCHAR(45), IN var_sesso VARCHAR(10), IN var_luogo_nascita VARCHAR(45), IN var_data_nascita VARCHAR(20))
BEGIN
        DECLARE var_old_cf VARCHAR (45);
   DECLARE custom_error CONDITION FOR SQLSTATE 'I0700';
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
            ROLLBACK;  -- Esegui il rollback in caso di errore
      RESIGNAL;
   END;
   SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
        START TRANSACTION;
   -- controllo cf --
   IF NOT EXISTS(SELECT tessera FROM cliente WHERE tessera = var_tessera) THEN
   SIGNAL custom_error SET MESSAGE_TEXT = 'Tessera cliente non presente o errata, per favore riprova';
   END IF;
        -- controllo nome e cognome--
```

```
IF NOT var_nome REGEXP '^[A-Za-zÀ-ÿ\\s\'-]+$' OR NOT var_cognome REGEXP '^[A-Za-zÀ-ÿ\\s\'-]+$' THEN
 SIGNAL custom_error SET MESSAGE_TEXT = 'Nome o cognome errati non possono contenere numeri o essere vuoti';
 END IF;
 -- controllo sesso --
 IF var_sesso NOT IN ('M','F') THEN
 SIGNAL custom_error SET MESSAGE_TEXT = 'Sesso errato prova a inserire M oppure F';
 END IF;
 -- controllo data --
 IF(CAST(var_data_nascita AS DATE)) IS NULL THEN
 SIGNAL custom_error SET MESSAGE_TEXT = 'Errore formato data';
 END IF;
 -- controllo nuovo codice fiscale --
 SELECT cf INTO var_old_cf FROM dati_anagrafici WHERE tessera_cliente = tessera;
 IF var_old_cf <> var_cf THEN
        -- controllo esistenza cf --
        IF EXISTS(SELECT cf FROM dati_anagrafici WHERE cf = var_cf) OR var_cf IS NULL THEN
        SIGNAL custom_error SET MESSAGE_TEXT = 'Codice fiscale gia presente';
        END IF;
        -- aggiorna i dati se codice fiscale è diverso--
   UPDATE dati_anagrafici
      SET cf = var_cf, nome = var_nome, cognome = var_cognome , sesso = var_sesso, luogo_nascita = var_luogo_nascita, data_nascita = var_data_nascita
   WHERE tessera_cliente = var_tessera;
 ELSE
        -- aggiorna i dati se codice fiscale è uguiale --
    UPDATE dati_anagrafici
        SET nome = var_nome, cognome = var_cognome , sesso = var_sesso, luogo_nascita = var_luogo_nascita, data_nascita = var_data_nascita
        WHERE tessera_cliente = var_tessera;
    END IF;
```

```
END!
GRANT EXECUTE ON PROCEDURE mod_customer_registry TO impiegato!


--
-- PROCEDURE I08  add_customer
--
CREATE PROCEDURE add_customer(IN var_tessera VARCHAR(45) ,IN var_cf VARCHAR(45), IN var_nome VARCHAR(45), IN var_cognome VARCHAR(45), IN var_sesso VARCHAR(10), IN var_luogo_nascita VARCHAR(45), IN var_data_nascita VARCHAR(20) ,IN var_cellulare VARCHAR(30) , IN var_telefono VARCHAR(30) , IN var_email VARCHAR(45))
BEGIN
   DECLARE custom_error CONDITION FOR SQLSTATE 'I0800';
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
                ROLLBACK;  -- Esegui il rollback in caso di errore
      RESIGNAL;
   END;
   SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
        START TRANSACTION;
   -- controllo cf --
    IF EXISTS(SELECT tessera FROM cliente WHERE tessera = var_tessera) OR var_tessera IS NULL THEN
   SIGNAL custom_error SET MESSAGE_TEXT = 'Tessera cliente gia presente';
   END IF;
        -- controllo cf --
   IF EXISTS(SELECT cf FROM dati_anagrafici WHERE cf = var_cf) OR var_cf IS NULL THEN
   SIGNAL custom_error SET MESSAGE_TEXT = 'Codice fiscale gia presente';
   END IF;
   -- controllo nome e cognome--
        IF NOT var_nome REGEXP '^[A-Za-zÀ-ÿ\\s\'-]+$' OR NOT var_cognome REGEXP '^[A-Za-zÀ-ÿ\\s\'-]+$' THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Nome o cognome errati non possono contenere numeri o essere vuoti';
```

```
END IF;
-- controllo sesso --
IF var_sesso NOT IN ('M','F') THEN
SIGNAL custom_error SET MESSAGE_TEXT = 'Sesso errato prova a inserire M oppure F';
END IF;
-- controllo data --
IF(CAST(var_data_nascita AS DATE)) IS NULL THEN
SIGNAL custom_error SET MESSAGE_TEXT = 'Errore formato data';
END IF;
    -- controllo su telefono e cellulare clieni --
IF NOT var_telefono REGEXP '^[0-9]+$'  OR NOT var_cellulare REGEXP '^[0-9]+$' THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Numero di telefono o cellulare';
END IF;
IF NOT var_email REGEXP '^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$' THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Indirizzo email non valido';
END IF;
-- controllo unicita recapiti --
    IF EXISTS(SELECT * FROM recapito_cliente WHERE telefono = var_telefono) THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Telefono gia utilizzato per favore riprova';
END IF;
    IF EXISTS(SELECT * FROM recapito_cliente WHERE cellulare = var_cellulare) THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Cellulare gia utilizzato per favore riprova';
END IF;
    IF EXISTS(SELECT * FROM recapito_cliente WHERE email = var_email) THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Email gia utilizzato per favore riprova';
END IF;
-- se supera controlli allora inserisci i dati --
INSERT INTO cliente (tessera)
VALUE (var_tessera);

    INSERT INTO dati_anagrafici(tessera_cliente, cf, nome, cognome, sesso, luogo_nascita,
data_nascita)
```

```
        VALUES (var_tessera, var_cf, var_nome,var_cognome, var_sesso, var_luogo_nascita,
var_data_nascita);

   INSERT INTO recapito_cliente (tessera_cliente, cellulare, telefono, email)
   VALUES (var_tessera, var_cellulare, var_telefono, var_email);
   COMMIT;
END!
GRANT EXECUTE ON PROCEDURE add_customer TO impiegato!



--
-- PROCEDURE I09  report_copy
--
CREATE PROCEDURE report_copy()
BEGIN
   DECLARE custom_error CONDITION FOR SQLSTATE 'I0900';
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
                ROLLBACK;  -- Esegui il rollback in caso di errore
      RESIGNAL;
   END;
                SELECT *
      FROM copia_film
      WHERE (titolo_film, regista_film, numero) IN (
      SELECT titolo_copia, regista_copia, numero_copia
      FROM noleggio_attivo
      WHERE CURDATE() > data_limite);
END!
GRANT EXECUTE ON PROCEDURE report_copy TO impiegato!

--
-- PROCEDURE I10  report_customer
--
```

```
CREATE PROCEDURE report_customer()
BEGIN
   DECLARE custom_error CONDITION FOR SQLSTATE 'I1000';
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
                ROLLBACK;  -- Esegui il rollback in caso di errore
      RESIGNAL;
   END;
                SELECT DISTINCT c.*, da.cf, da.nome, da.cognome, da.sesso, da.luogo_nascita, da.data_nascita, rc.cellulare, rc.telefono, rc.email
                FROM cliente c
                JOIN noleggio_attivo na ON c.tessera = na.tessera_cliente
                JOIN dati_anagrafici da ON c.tessera = da.tessera_cliente
                JOIN recapito_cliente rc ON c.tessera = rc.tessera_cliente
                WHERE CURDATE() > na.data_limite;
END!
GRANT EXECUTE ON PROCEDURE report_customer TO impiegato!



--
-- PROCEDURE P01 add_worker
--
CREATE PROCEDURE add_worker (IN var_cf VARCHAR(45), IN var_nome VARCHAR(45), IN var_cellulare VARCHAR(30) , IN var_telefono VARCHAR(30) , IN var_email VARCHAR(45), IN var_ruolo VARCHAR(20))
BEGIN
        DECLARE custom_error CONDITION FOR SQLSTATE 'P0100';
   DECLARE curr_date DATE;
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
                ROLLBACK;  -- Esegui il rollback in caso di errore
      RESIGNAL;
   END;
```

```
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
    START TRANSACTION;
    -- controllo cf --
IF EXISTS(SELECT cf FROM impiegato WHERE cf = var_cf) OR var_cf IS NULL THEN
SIGNAL custom_error SET MESSAGE_TEXT = 'Codice fiscale gia presente';
END IF;
-- controllo nome --
IF NOT var_nome REGEXP '^[A-Za-zÀ-ÿ\\s\'-]+$' THEN
 SIGNAL custom_error SET MESSAGE_TEXT = 'Nome o cognome errati non possono contenere numeri o essere vuoti';
END IF;
    -- controllo su telefono e cellulare  --
IF (NOT var_telefono REGEXP '^[0-9]+$') OR (NOT var_cellulare REGEXP '^[0-9]+$') THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Numero di telefono o cellulare contengono lettere';
END IF;
IF NOT var_email REGEXP '^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$' THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Indirizzo email non valido';
END IF;
-- controllo unicita recapiti --
    IF EXISTS(SELECT * FROM recapito_impiegato WHERE telefono = var_telefono) THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Telefono gia utilizzato per favore riprova';
END IF;
    IF EXISTS(SELECT * FROM recapito_impiegato WHERE cellulare = var_cellulare) THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Cellulare gia utilizzato per favore riprova';
END IF;
    IF EXISTS(SELECT * FROM recapito_impiegato WHERE email = var_email) THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Email gia utilizzato per favore riprova';
END IF;
-- controllo ruolo --
IF var_ruolo NOT IN ('cassiere','magazziniere','commesso') THEN
      SIGNAL custom_error SET MESSAGE_TEXT = 'Errore incarico seleziona tra (cassiere,magazziniere,commesso)';
```

```
    END IF;

    SET curr_date = CURDATE();
    -- se supera controlli allora inserisci i dati --
    INSERT INTO impiegato (cf, nome)
    VALUE (var_cf,var_nome);

    INSERT INTO recapito_impiegato(cf_impiegaTO, cellulare, telefono, email)
    VALUES (var_cf, var_cellulare, var_telefono, var_email);

    INSERT INTO incarico_corrente (cf_impiegaTO, tipo, inizio)
    VALUES (var_cf, var_ruolo, curr_date);
    COMMIT;
END!
GRANT EXECUTE ON PROCEDURE add_worker TO proprietario!




--
-- PROCEDURE P02 mod_worker
--
CREATE PROCEDURE mod_worker (IN var_cf VARCHAR(45), IN var_ruolo VARCHAR(20))
BEGIN
        DECLARE custom_error CONDITION FOR SQLSTATE 'P0200';
    DECLARE curr_date DATE;
    DECLARE var_old_ruolo VARCHAR(20);
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
                ROLLBACK;  -- Esegui il rollback in caso di errore
       RESIGNAL;
    END;
    SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
        START TRANSACTION;
```

```
        -- controllo cf --
   IF NOT EXISTS(SELECT cf FROM impiegato WHERE cf = var_cf) OR var_cf IS NULL THEN
   SIGNAL custom_error SET MESSAGE_TEXT = 'Codice fiscale errato o inesistente';
   END IF;
   -- controllo ruolo --
   IF var_ruolo NOT IN('cassiere','magazziniere','commesso') THEN
         SIGNAL custom_error SET MESSAGE_TEXT = 'Errore incarico seleziona tra (cassiere,magazziniere,commesso)';
   END IF;
   -- verifica se ruolo in input diverso ruolo effettivo --
   SELECT tipo INTO var_old_ruolo FROM incarico_corrente WHERE cf_impiegaTO = var_cf;
   IF var_old_ruolo <> var_ruolo THEN
                -- inserisci dati nell incarico passato --
                SET curr_date = CURDATE();
      INSERT INTO incarico_passato (cf_impiegaTO, tipo, INizio, fine)
      SELECT cf_impiegaTO, tipo, inizio, curr_date
      FROM incarico_corrente
      WHERE cf_impiegaTO = var_cf;
      -- aggiorna incarico corrente --
      UPDATE incarico_corrente
      SET tipo = var_ruolo, inizio = curr_date
      WHERE cf_impiegaTO = var_cf;
   ELSE
                SIGNAL custom_error SET MESSAGE_TEXT = 'Questo impiegato occupa gia questo ruolo';
        END IF;
   COMMIT;
END!
GRANT EXECUTE ON PROCEDURE mod_worker TO proprietario!

--
-- PROCEDURE P03 all_workers
--
```

```
CREATE PROCEDURE all_workers ()
BEGIN
        SELECT
   i.cf AS CodiceFiscale,
   i.nome AS Nome,
   r.cellulare AS Cellulare,
   r.telefono AS Telefono,
   r.email AS Email,
   ic.tipo AS IncaricoCorrente,
   ic.inizio AS DataInizioIncarico
FROM
   impiegato AS i
LEFT JOIN
   recapito_impiegato AS r ON i.cf = r.cf_impiegaTO
LEFT JOIN
   incarico_corrente AS ic ON i.cf = ic.cf_impiegaTO;
END!
GRANT EXECUTE ON PROCEDURE all_workers TO proprietario!


--
-- PROCEDURE P04 add_workshift
--
CREATE PROCEDURE add_workshift (IN var_cf VARCHAR(45), IN var_inizio VARCHAR(20),
IN var_fine VARCHAR(20), IN var_data VARCHAR(20))
BEGIN
        DECLARE custom_error CONDITION FOR SQLSTATE 'P0400';
   DECLARE var_data_date DATE;
   DECLARE var_inizio_int INT;
   DECLARE var_fine_int INT;
   DECLARE var_old_ruolo VARCHAR(20);
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
   BEGIN
                ROLLBACK;  -- Esegui il rollback in caso di errore
```

```
    RESIGNAL;
  END;
  SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
      START TRANSACTION;
      -- controllo cf --
  IF NOT EXISTS(SELECT cf FROM impiegato WHERE cf = var_cf) OR var_cf IS NULL THEN
  SIGNAL custom_error SET MESSAGE_TEXT = 'Codice fiscale errato o inesistente';
  END IF;
      -- controllo validita ore --
  IF CAST(var_inizio AS SIGNED) IS NULL OR CAST(var_inizio AS SIGNED) IS NULL THEN
  SIGNAL custom_error SET MESSAGE_TEXT = 'Ora inizio turno o ora fine turno non validi';
  END IF;
  SET var_inizio_int = CAST(var_inizio AS SIGNED);
  SET var_fine_int = CAST(var_fine AS SIGNED);
  -- controllo turno --
      IF NOT EXISTS(SELECT * FROM turno WHERE ora_inizio = var_inizio_int AND
ora_fine = var_fine_int) THEN
  SIGNAL custom_error SET MESSAGE_TEXT = 'Non esiste questo turno o è nullo';
  END IF;
  -- controllo date --
      IF(CAST(var_data AS DATE)) IS NULL THEN
  SIGNAL custom_error SET MESSAGE_TEXT = 'Errore formato data';
  END IF;
  SET var_data_date = (CAST(var_data AS DATE));
  -- se esiste un turno effettivo in quel giorno aggiorna il turno dell impiegato --
    IF EXISTS (SELECT * FROM turno_effettivo WHERE inizio_turno = var_inizio_int AND
fine_turno = var_fine_int AND data_turno = var_data_date) THEN
            INSERT INTO turno_impiegato(inizio_effettivo, fine_effettiva, data_effettiva,
cf_impiegato)
      SELECT inizio_turno, fine_turno, data_turno , var_cf
      FROM turno_effettivo
        WHERE inizio_turno = var_inizio_int AND fine_turno = var_fine_int AND data_turno =
var_data_date;
```

```
   -- altrimenti crealo --
   ELSE
                -- aggiungi turno effettivo --
                INSERT INTO turno_effettivo (inizio_turno, fine_turno, data_turno)
      VALUES (var_inizio_int, var_fine_int, var_data_date);
      -- aggiungi turno all impiegato --
      INSERT INTO turno_impiegato(inizio_effettivo, fine_effettiva, data_effettiva, cf_impiegato)
      SELECT inizio_turno, fine_turno, data_turno , var_cf
      FROM turno_effettivo
          WHERE inizio_turno = var_inizio_int AND fine_turno = var_fine_int AND data_turno =
var_data_date;

   END IF;
   COMMIT;
END!
GRANT EXECUTE ON PROCEDURE add_workshift TO proprietario!



--
-- PROCEDURE P05 mod_workshift
--
CREATE PROCEDURE mod_workshift (IN var_cf VARCHAR(45), IN var_inizio VARCHAR(20),
IN var_fine VARCHAR(20), IN var_data VARCHAR(20))
BEGIN
        DECLARE custom_error CONDITION FOR SQLSTATE 'P0500';
   DECLARE var_data_date DATE;
   DECLARE var_inizio_int INT;
   DECLARE var_fine_int INT;
   DECLARE var_old_inizio INT;
   DECLARE var_old_fine INT;
   DECLARE count_turno INT;
   DECLARE var_old_ruolo VARCHAR(20);
```

```
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
                ROLLBACK;  -- Esegui il rollback in caso di errore
       RESIGNAL;
    END;
    SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
        START TRANSACTION;
        -- controllo cf --
    IF NOT EXISTS(SELECT cf FROM impiegato WHERE cf = var_cf) OR var_cf IS NULL THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Codice fiscale errato o inesistente';
    END IF;
        -- controllo validita ore --
    IF CAST(var_inizio AS SIGNED) IS NULL OR CAST(var_inizio AS SIGNED) IS NULL THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Ora inizio turno o ora fine turno non validi';
    END IF;
    SET var_inizio_int = CAST(var_inizio AS SIGNED);
    SET var_fine_int = CAST(var_fine AS SIGNED);
    -- controllo turno --
        IF NOT EXISTS(SELECT * FROM turno WHERE ora_inizio = var_inizio_int AND
ora_fine = var_fine_int) THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Non esiste questo turno o è nullo';
    END IF;
    -- controllo date --
        IF(CAST(var_data AS DATE)) IS NULL THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Errore formato data';
    END IF;
    SET var_data_date = (CAST(var_data AS DATE));
    -- controllo turno effettivo --
    IF NOT EXISTS(
                SELECT *
                FROM turno_impiegato AS ti
                JOIN turno_effettivo AS te
                ON ti.inizio_effettivo = te.inizio_turno
```

```
                AND ti.fine_effettiva = te.fine_turno
                AND ti.data_effettiva = te.data_turno
                WHERE ti.cf_impiegaTO = var_cf
                AND ti.data_effettiva = var_data_date
        ) THEN
        SIGNAL custom_error SET MESSAGE_TEXT = 'Non esiste questo turno o è nullo';
        END IF;
   -- controllo cambio turno --
   SELECT inizio_effettivo,fine_effettiva INTO var_old_inizio, var_old_fine
   FROM turno_impiegato
        WHERE cf_impiegaTO = var_cf AND data_effettiva = var_data_date;
   IF var_old_inizio <> var_inizio_int OR var_old_fine <> var_fine_int THEN
                UPDATE turno_impiegato
      SET inizio_effettivo = var_inizio_int, fine_effettiva = var_fine_int
      WHERE data_effettiva = var_data_date AND cf_impiegaTO = var_cf;
   ELSE
    SIGNAL custom_error SET MESSAGE_TEXT = 'Questo impiegato ricopre gia questo turno in
questa data';
   END IF;
   COMMIT;
END!
GRANT EXECUTE ON PROCEDURE mod_workshift TO proprietario!


--
-- PROCEDURE P06 delete_workshift
--
CREATE PROCEDURE delete_workshift (IN var_cf VARCHAR(45), IN var_inizio
VARCHAR(20), IN var_fine VARCHAR(20), IN var_data VARCHAR(20))
BEGIN
        DECLARE custom_error CONDITION FOR SQLSTATE 'P0600';
   DECLARE var_data_date DATE;
   DECLARE var_inizio_int INT;
   DECLARE var_fine_int INT;
```

```
        DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
                ROLLBACK;  -- Esegui il rollback in caso di errore
       RESIGNAL;
    END;
    SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
        START TRANSACTION;
        -- controllo cf --
    IF NOT EXISTS(SELECT cf FROM impiegato WHERE cf = var_cf) OR var_cf IS NULL THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Codice fiscale errato o inesistente';
    END IF;
        -- controllo validita ore --
    IF CAST(var_inizio AS SIGNED) IS NULL OR CAST(var_inizio AS SIGNED) IS NULL THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Ora inizio turno o ora fine turno non validi';
    END IF;
    SET var_inizio_int = CAST(var_inizio AS SIGNED);
    SET var_fine_int = CAST(var_fine AS SIGNED);
    -- controllo turno --
        IF NOT EXISTS(SELECT * FROM turno WHERE ora_inizio = var_inizio_int AND
ora_fine = var_fine_int) THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Non esiste questo turno o è nullo';
    END IF;
    -- controllo date --
        IF(CAST(var_data AS DATE)) IS NULL THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Errore formato data';
    END IF;
    SET var_data_date = (CAST(var_data AS DATE));
    -- controllo esistenza turno effettivo dell impiegato --
        IF EXISTS(SELECT * FROM turno_impiegato WHERE inizio_effettivo = var_inizio AND
fine_effettiva = var_fine_int AND data_effettiva = var_data_date AND cf_impiegato = var_cf)
THEN
                DELETE FROM turno_impiegato
```

```
        WHERE inizio_effettivo = var_inizio AND fine_effettiva = var_fine_int AND data_effettiva = var_data_date AND cf_impiegato = var_cf;
    ELSE
                    SIGNAL custom_error SET MESSAGE_TEXT = 'Non esiste questo turno per questo impiegato';
    END IF;
    COMMIT;
END!
GRANT EXECUTE ON PROCEDURE delete_workshift TO proprietario!



--
-- PROCEDURE P08 day_workshifts
--
CREATE PROCEDURE day_workshifts(IN var_data VARCHAR(20))
BEGIN
        DECLARE var_data_date DATE;
    DECLARE custom_error CONDITION FOR SQLSTATE 'P0800';
        -- controllo date --
        IF(CAST(var_data AS DATE)) IS NULL THEN
    SIGNAL custom_error SET MESSAGE_TEXT = 'Errore formato data';
    END IF;
    SET var_data_date = CAST(var_data AS DATE);
        SELECT cf_impiegaTO,inizio_effettivo,fine_effettiva, data_effettiva
        FROM turno_impiegato
    WHERE data_effettiva = var_data_date;
END!
GRANT EXECUTE ON PROCEDURE day_workshifts TO proprietario!

--
-- PROCEDURE P08 report_monthly_hours
--
CREATE PROCEDURE report_monthly_hours(IN var_month INT, IN var_year INT)
```

```
BEGIN
  -- Crea una tabella temporanea per immagazzinare i risultati
  CREATE TEMPORARY TABLE temp_monthly_hours (
    cf_impiegaTO VARCHAR(45) NOT NULL,
    total_hours INT
  );

  -- Esegui il calcolo e inserisci i risultati nella tabella temporanea
  INSERT INTO temp_monthly_hours (cf_impiegaTO, total_hours)
  SELECT
    ti.cf_impiegaTO,
    SUM(
      CASE
        WHEN te.inizio_turno <= te.fine_turno THEN te.fine_turno - te.inizio_turno
        ELSE (24 - te.inizio_turno) + te.fine_turno
      END
    ) AS total_hours
  FROM
    turno_impiegato AS ti
  INNER JOIN
    turno_effettivo AS te ON ti.inizio_effettivo = te.inizio_turno
      AND ti.fine_effettiva = te.fine_turno
      AND ti.data_effettiva = te.data_turno
  WHERE
    MONTH(ti.data_effettiva) = var_month
    AND YEAR(ti.data_effettiva) = var_year
  GROUP BY
    ti.cf_impiegaTO;

  -- Seleziona i risultati dalla tabella temporanea
  SELECT * FROM temp_monthly_hours;

  -- Elimina la tabella temporanea
```

```
   DROP TEMPORARY TABLE temp_monthly_hours;
END!
GRANT EXECUTE ON PROCEDURE report_monthly_hours TO proprietario!



--
-- PROCEDURE P09 report_yearly_hours
--
CREATE PROCEDURE report_yearly_hours(IN var_year INT)
BEGIN
   -- Crea una tabella temporanea per immagazzinare i risultati
   CREATE TEMPORARY TABLE temp_yearly_hours (
      cf_impiegaTO VARCHAR(45) NOT NULL,
      total_hours INT
   );

   -- Esegui il calcolo e inserisci i risultati nella tabella temporanea
   INSERT INTO temp_yearly_hours (cf_impiegaTO, total_hours)
   SELECT
      ti.cf_impiegaTO,
      SUM(
         CASE
            WHEN te.inizio_turno <= te.fine_turno THEN te.fine_turno - te.inizio_turno
            ELSE (24 - te.inizio_turno) + te.fine_turno
         END
      ) AS total_hours
   FROM
      turno_impiegato AS ti
   INNER JOIN
      turno_effettivo AS te ON ti.inizio_effettivo = te.inizio_turno
         AND ti.fine_effettiva = te.fine_turno
         AND ti.data_effettiva = te.data_turno
   WHERE
```

```
        YEAR(ti.data_effettiva) = var_year
    GROUP BY
        ti.cf_impiegaTO;

    -- Seleziona i risultati dalla tabella temporanea
    SELECT * FROM temp_yearly_hours;

    -- Elimina la tabella temporanea
    DROP TEMPORARY TABLE temp_yearly_hours;
END;
GRANT EXECUTE ON PROCEDURE report_yearly_hours TO proprietario!
```