Eugen Saraci Matricola:1073559

Print PDFs Online - PPDFO

Abstract

Il progetto consiste nella realizzazione di una base di dati per un sito che permette la stampa di file PDF a prezzi agevolati. L'utente avrà la possibilità di caricare i propri file PDF e modifcarne il formato per poi farli stampare all'azienda che provvederà ad inviare il prodotto finale al cliente. Nell' interfaccia web è presente una sezione dedicata allo svolgimento delle operazioni elementari per la gestione del sito, essa è riservata agli utenti amministratori.

Analisi dei requisiti

L'entità principale che si vuole rappresentare è l'**ordine** di cui vogliamo sapere l'indirizzo di spedizione, il costo finale, la data, lo stato (in attesa di conferma, in attesa di rimborso, concluso, rimborso accettato, rimborso rifiutato), l'utente che ha richiesto l'ordine e l'eventuale utilizzo di un coupon per ottenere sconti sul prezzo finale. Il prezzo dell'ordine viene determinato dal file caricato e dal formato con cui si decide di stampare il pdf, del **file** ci interessa l'autore dell' upload, il titolo, il numero di pagine prima della formattazione, la dimensione, e il timestamp di quando è stato caricato. L'utente col file caricato passa successivamente alla fase di modifica del file: la **stampa**, di essa ci interessa principalmente il formato. Le possibili modifiche effettuate dal **formato** sono l'orientamento, il colore (biaco e nero, colorato), la rilegatura (se presente o meno, lato lungo, lato corto) e le pagine per foglio. Il formato utilizzato da qualsiasi stampa è A4.

Il **cliente** è un tipo di utente che potrà effettuare un ordine attraverso i passaggi di login, upload, modifica formato e pagamento di cui ci interessa solamente l'indirizzo paypal. Un cliente ha i seguenti attributi: un id, un username, una password, un' email, il totale delle pagine stampate ed il totale dei soldi spesi. Una volta effettuato l'ordine l'utente dovrà restare in attesa della conferma di tale ordine da parte di un amministratore e potrà annullare l'ordine in qualsiasi momento prima della conferma. Dopo la conferma un cliente avrà la possibilità di richiedere un rimborso, in questo caso un'amministratore potrà decidere se approvare o rifiutare il rimborso richiesto, in entrambi i casi l'ordine verrà concluso. Un utente ha la possibiltà di aggiungere fino a 3 indirizzi di spedizione, essi sono caratterizzati da un id, il nome dell'utente, il cognome, la via, la città, il CAP e la regione. Gli indirizzi possono essere eliminati per aggiungerne altri. Ad un cliente è data la possibilità di moficare username, password ed email. Un cliente non può registrarsi al sito e nemmeno cancellarsi.

Un **amministratore** è un tipo di utente con id, username, password ed email, esso non ha la possiblità di effetuare ordini, ma potrà gestirne lo stato, potrà inoltre aggiungere coupon specificando il codice e lo sconto in percentuale. Potrà creare account per i clienti e modificarne i dati, avrà accesso inoltre ad una sezione dedicata ai file caricati ma non stampati da almeno 24 ore ed ai file il cui formato è stato modifcato ma che non sono stati usati in un ordine da almeno 24 ore, questi file sono detti orfani e possono essere rimossi dall'amministratore.

Progettazione concettuale

L'unico attributo che può assumere valore NULL è il coupon utilizzato in un ordine. Le classi più importanti avranno sempre un attributo in cui è specificato l'id dell'utente che l'ha creata.

Lista delle classi:

Utente: rappreseneta una persona cui è permessa il login all'interno del sito

- Id
- username
- password
- email

  Sottoclassi:
  Cliente: accede all'interfaccia di upload,modifca,acquisto del sito.
  - pagine stampate
  - soldi spesi

  Amministratore: accede al pannello di amministrazione del sito, ha gli stessi attributi di un utente generico.

File

- id
- titolo
- pagine
- dimensione

Indirizzo

- id
- nome
- cognome
- via
- citta
- cap
- regione

  Le sottoclassi indirizzi in uso ed indirizzi cancellati hanno gli stessi attributi di Indirizzo

Ordine

- id
- data
- prezzo
- coupon
- stato

Lista delle relazioni:

CLIENTE-FILE: upload

Un cliente può caricare 0 o N file, ma un file se presente può esser stato caricato da solo un utente, inoltre essendo un file distinto dall'id nessun file potrà essere caricato da più utenti.

CLIENTE-ORDINE: stampa

Un cliente può non fare ordini o farne N, un'ordine essendo distinto da id ed altri attributi univoci che cambiano in base al cliente non può essere fatto da più utenti.

CLIENTE-FILE: stampa

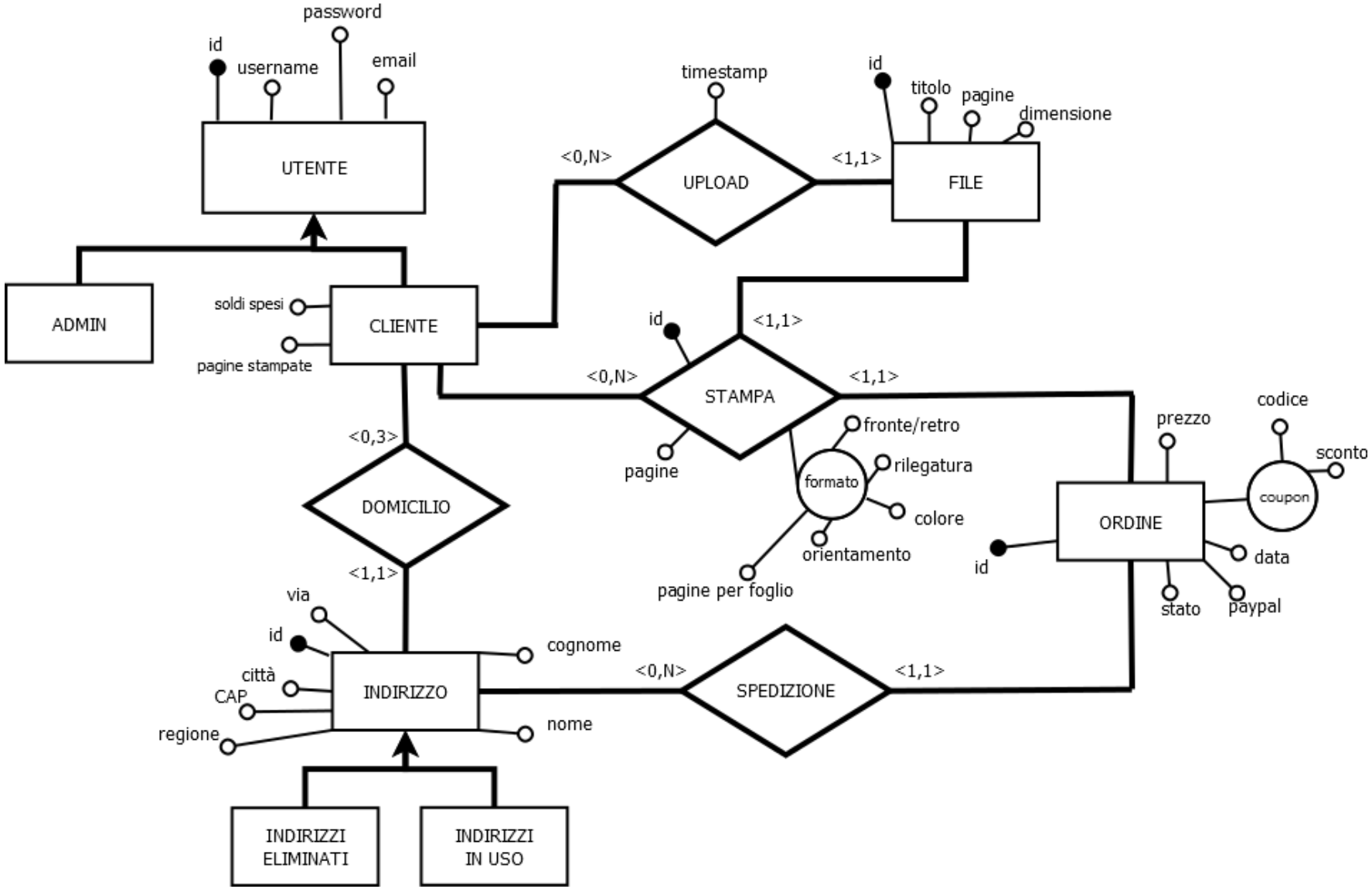
Come CLIENTE-ORDINE: stampa.

CLIENTE-INDIRIZZO: domicilio

Un cliente può non avere indirizzi o averne al massimo 3, un indirizzo se inserito deve necessariamente avere un solo "autore" e nel caso in cui più utenti abbiano lo stesso indirizzo (via, città ecc...) esso sarà sempre distinto dall'id.

INDIRIZZO-ORDINE: spedizione

Un indirizzo può non essere usato per qualche spedizione oppure può essere usato per N spedizioni, un'ordine essendo sempre distinto da un non avrà mai due indirizzi identici.


password
id
username
email
UTENTE
timestamp
id
titolo
pagine
dimensione
<0,N>
UPLOAD
<1,1>
FILE
ADMIN
soldi spesi
CLIENTE
pagine stampate
id
<1,1>
<0,N>
STAMPA
<1,1>
pagine
fronte/retro
rilegatura
formato
colore
orientamento
pagine per foglio
<0,3>
DOMICILIO
<1,1>
prezzo
codice
sconto
coupon
ORDINE
id
data
stato
paypal
via
id
città
CAP
regione
INDIRIZZO
cognome
nome
<0,N>
SPEDIZIONE
<1,1>
INDIRIZZI
ELIMINATI
INDIRIZZI
IN USO

Progettazione logica

Gerarchie

La gerarchia Utente(Cliente, Amministratore) è stata fatta collassare in alto, la classe generica è ora Utente che è caratterizata dai suoi attributi precedenti più quelli tipici di Cliente, è stato inoltre introdotto un campo per specificare se l'utente è amministratore o meno. La gerarchia Indirizzi(indirizzi in uso, cancellati) è stata fatta collassare in basso con la presenza quindi di due classi aventi gli stessi attributi della classe madre.

Per ogni entità presente precedentemente è stata introdotta una chiave sintetica per poter facilitare l'individuazione dei vari elementi. E' stata inoltre introdotta in più tabelle la ridondanza dell' identificatore utente (id) in modo da permettere l'immediato riconoscimento dell'autore di un ordine, una stampa, un'indirizzo, un file.

Relazioni

La relazione Domicilio è stata rimossa e l'identificatore dell'utente è stato aggiunto alla tabella indirizzi.

La relazione Upload è stata accorpata con File, ora File ha anche l'attributo timestamp e, come specificato prima, anche per file è stato introdotto un campo id relativo all'utente che ha effettuato il caricamento.

La relazione Spedizione è stata rimossa e l'attributo identificativo dell'indirizzo è stato aggiunto a Ordine.

La relazione Stampa è ora un entità ed assume come campi oltre agli attributi precedenti anche gli id rispettivi di utente, file e formato.

Attributi

L'attributo formato è ora una tabella ed ha come campi gli attributi precedenti più un campo identificativo.

Coupon è ora una tabella ed ha come campi gli attributi precedenti con l'aggiunta di un campo identificativo.

Schema Logico

PK = Primary Key, FK = Foreign Key, U=UNIQUE, se non specificato gli attributi sono NOT NULL.

Utente

- id smallint(6) PK
- admin tinyint(1)
- email varchar(64) U
- username varchar(32) U
- password varchar(32)
- soldi_spesi decimal(10,2)
- tot_pagine_stampate int(11)

Indirizzo

- id_indirizzo smallint(6) PK
- link_utente smallint(6) FK ➔ Utente.id
- nome text
- cognome text
- via varchar(32)
- citta varchar(16)
- regione varchar(32)
- CAP mediumint(5)

Indirizzo_log

- id_indirizzo smallint(6) PK
- link_utente smallint(6)
- nome text
- cognome text
- via varchar(32)
- citta varchar(16)
- regione varchar(32)
- CAP mediumint(5)

Files

- id_file smallint(6) PK
- title varchar(64) U
- proprietario smallint(6) FK ➔Utente.id
- timestamp timestamp
- pagine smallint(6)
- size int(11)

Formato

- id_formato varchar(5) PK
- orientamento enum('verticale','orizzontale')
- fronte_retro tinyint(1)
- colore tinyint(1)
- rilegatura enum('no','lato_corto','lato_lungo')
- righe tinyint(1)
- colonne tinyint(1)

Stampa

- id_stampa smallint(6) PK
- utente_stampa smallint(6) FK ➔ Utente.id
- file_stampa smallint(6) FK ➔ Files.id_file
- formato_stampa varchar(5) FK ➔ Formato.id_formato
- pagine_finali smallint(6)

Coupon

- id_coupon smallint(6) PK
- codice_coupon varchar(16) U
- sconto smallint(2)

Ordine

- id_ordine smallint(6) PK
- utente_ordine smallint(6) FK ➔Utente.id
- stampa_ordine smallint(6) FK ➔ Stampa.id_stampa
- indirizzo_ordine smallint(6) FK ➔ Indirizzo.id_indirizzo
- coupon_ordine varchar(16) FK ➔ Coupon.codice_coupon
- data_ordine timestamp
- prezzo decimal(10,2)
- paypal varchar(64)
- stato enum('In attesa','Concluso','Rimborsato','Richiesta rimborso','Rifiutato')

Implementazione del DB

Tabelle

```
-- Struttura della tabella `coupon`
--

DROP TABLE IF EXISTS `coupon`;
CREATE TABLE IF NOT EXISTS `coupon` (
  `id_coupon` smallint(6) NOT NULL AUTO_INCREMENT,
  `codice_coupon` varchar(16) NOT NULL,
  `sconto` smallint(2) NOT NULL,
  PRIMARY KEY (`id_coupon`),
  UNIQUE KEY `codice_coupon_2` (`codice_coupon`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=0 ;


-- Struttura della tabella `files`

DROP TABLE IF EXISTS `files`;
CREATE TABLE IF NOT EXISTS `files` (
  `id_file` smallint(6) NOT NULL AUTO_INCREMENT,
  `title` varchar(64) NOT NULL,
  `proprietario` smallint(6) NOT NULL,
  `pagine` smallint(6) NOT NULL,
  `size` int(11) NOT NULL,
  `timestamp` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
  PRIMARY KEY (`id_file`),
  UNIQUE KEY `title` (`title`),
  KEY `proprietario` (`proprietario`),
CONSTRAINT `files_ibfk_1` FOREIGN KEY (`proprietario`) REFERENCES `utente`
(`id`) ON DELETE NO ACTION ON UPDATE CASCADE

) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=0 ;
```

```sql
-- Struttura della tabella `formato`

DROP TABLE IF EXISTS `formato`;
CREATE TABLE IF NOT EXISTS `formato` (
  `id_formato` varchar(5) NOT NULL,
  `orientamento` enum('verticale','orizzontale') NOT NULL DEFAULT 'verticale',
  `fronte_retro` tinyint(1) NOT NULL DEFAULT '0' COMMENT '1=fronte e retro',
  `colore` tinyint(1) NOT NULL DEFAULT '0',
  `rilegatura` enum('no','lato_corto','lato_lungo') NOT NULL DEFAULT 'no',
  `righe` tinyint(1) NOT NULL DEFAULT '1',
  `colonne` tinyint(1) NOT NULL DEFAULT '1',
  PRIMARY KEY (`id_formato`),
  UNIQUE KEY `id_formato` (`id_formato`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;



-- Struttura della tabella `indirizzo`

DROP TABLE IF EXISTS `indirizzo`;
CREATE TABLE IF NOT EXISTS `indirizzo` (
  `id_indirizzo` smallint(6) NOT NULL AUTO_INCREMENT,
  `link_utente` smallint(6) NOT NULL,
  `nome` text NOT NULL,
  `cognome` text NOT NULL,
  `via` varchar(32) NOT NULL,
  `citta` varchar(16) NOT NULL,
  `regione` varchar(32) NOT NULL,
  `CAP` mediumint(5) NOT NULL,
  PRIMARY KEY (`id_indirizzo`),
  KEY `link_utente` (`link_utente`),
CONSTRAINT `indirizzo_ibfk_1` FOREIGN KEY (`link_utente`) REFERENCES `utente`
(`id`) ON DELETE NO ACTION ON UPDATE CASCADE

) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=0 ;



-- Struttura della tabella `indirizzo_log`

CREATE TABLE IF NOT EXISTS `indirizzo_log` (
  `id_indirizzo` smallint(6) NOT NULL,
  `link_utente` smallint(6) NOT NULL,
  `nome` text NOT NULL,
  `cognome` text NOT NULL,
  `via` varchar(64) NOT NULL,
  `citta` varchar(16) NOT NULL,
  `regione` varchar(32) NOT NULL,
  `CAP` mediumint(5) NOT NULL,
  PRIMARY KEY (`id_indirizzo`),
  KEY `link_utente` (`link_utente`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
-- Struttura della tabella `ordine`

DROP TABLE IF EXISTS `ordine`;
CREATE TABLE IF NOT EXISTS `ordine` (
  `id_ordine` smallint(6) NOT NULL AUTO_INCREMENT,
  `utente_ordine` smallint(6) NOT NULL,
  `stampa_ordine` smallint(6) NOT NULL,
  `indirizzo_ordine` smallint(6) NOT NULL,
  `data_ordine` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `coupon_ordine` varchar(16) DEFAULT NULL,
  `prezzo` decimal(10,2) NOT NULL,
  `paypal` varchar(64) NOT NULL,
  `stato` enum('In attesa','Concluso','Rimborsato','Richiesta
rimborso','Rifiutato') NOT NULL DEFAULT 'In attesa',
  PRIMARY KEY (`id_ordine`),
  KEY `utente_ordine` (`utente_ordine`),
CONSTRAINT `ordine_ibfk_1` FOREIGN KEY (`utente_ordine`) REFERENCES `utente`
(`id`) ON DELETE NO ACTION ON UPDATE CASCADE,
  KEY `stampa_ordine` (`stampa_ordine`),
CONSTRAINT `ordine_ibfk_2` FOREIGN KEY (`stampa_ordine`) REFERENCES `stampa`
(`id_stampa`) ON DELETE NO ACTION ON UPDATE CASCADE
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=0 ;


-- Struttura della tabella `stampa`

DROP TABLE IF EXISTS `stampa`;
CREATE TABLE IF NOT EXISTS `stampa` (
  `id_stampa` smallint(6) NOT NULL AUTO_INCREMENT,
  `utente_stampa` smallint(6) NOT NULL,
  `file_stampa` smallint(6) NOT NULL,
  `formato_stampa` varchar(5) NOT NULL,
  `pagine_finali` smallint(6) NOT NULL,
  PRIMARY KEY (`id_stampa`),
  KEY `utente_stampa` (`utente_stampa`),
CONSTRAINT `stampa_ibfk_1` FOREIGN KEY (`utente_stampa`) REFERENCES `utente`
(`id`) ON DELETE NO ACTION ON UPDATE CASCADE,
  KEY `file_stampa` (`file_stampa`),
CONSTRAINT `stampa_ibfk_2` FOREIGN KEY (`file_stampa`) REFERENCES `files`
(`id_file`) ON DELETE CASCADE ON UPDATE CASCADE,
  KEY `formato_stampa` (`formato_stampa`)
CONSTRAINT `stampa_ibfk_3` FOREIGN KEY (`formato_stampa`) REFERENCES `formato`
(`id_formato`) ON DELETE NO ACTION ON UPDATE CASCADE
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=0 ;


-- Struttura della tabella `utente`

DROP TABLE IF EXISTS `utente`;
CREATE TABLE IF NOT EXISTS `utente` (
  `id` smallint(6) NOT NULL AUTO_INCREMENT,
  `admin` tinyint(1) NOT NULL DEFAULT '0',
  `email` varchar(64) NOT NULL,
  `username` varchar(32) NOT NULL,
  `password` varchar(32) NOT NULL,
  `soldi_spesi` decimal(10,2) NOT NULL DEFAULT '0.00',
  `tot_pagine_stampate` int(11) NOT NULL DEFAULT '0',
  PRIMARY KEY (`id`),
  UNIQUE KEY `username` (`username`),
  UNIQUE KEY `email` (`email`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=0 ;
```

Codice SQL per la generazione di due utenti elementari.

```
INSERT INTO `utente` (`id`, `admin`, `email`, `username`, `password`,
`soldi_spesi`, `tot_pagine_stampate`) VALUES
(1, 1, 'admin@ppdfo.it', 'admin', 'admin', '0.00', 0),
(2, 0, 'user@ppdfo.it', 'user', 'user', '0.00', 0);
```

Funzioni e Procedure

Le due seguenti procedure vengono invocate da un trigger e servono per aggiornare i contatori di soldi_spesi e tot_pagine_stampate di un utente.

```
DROP PROCEDURE IF EXISTS `add_soldi_pagine`
CREATE PROCEDURE `add_soldi_pagine`(IN `valore` DECIMAL(10,2), IN `uid`
SMALLINT(6), IN `pag_piu` INT(6))
    NO SQL
BEGIN
DECLARE valore_old DECIMAL(10,2);
DECLARE valore_new DECIMAL(10,2);
DECLARE pag_old INT(11);
DECLARE pag_new INT(11);
SELECT soldi_spesi, tot_pagine_stampate INTO valore_old, pag_old FROM utente
WHERE id = uid;
SET valore_new = valore + valore_old;
SET pag_new = pag_piu + pag_old;
UPDATE utente SET soldi_spesi=valore_new, tot_pagine_stampate=pag_new WHERE
id=uid;
END
```

```
DROP PROCEDURE IF EXISTS `sub_soldi_pagine`
CREATE PROCEDURE `sub_soldi_pagine`(IN `valore` DECIMAL(10,2), IN `uid`
SMALLINT(6), IN `pag_meno` INT(6))
    NO SQL
BEGIN
DECLARE valore_old DECIMAL(10,2);
DECLARE valore_new DECIMAL(10,2);
DECLARE pag_old INT(11);
DECLARE pag_new INT(11);
SELECT soldi_spesi, tot_pagine_stampate INTO valore_old, pag_old FROM utente
WHERE id = uid;
SET valore_new = valore_old - valore;
SET pag_new = pag_old - pag_meno;
UPDATE utente SET soldi_spesi=valore_new, tot_pagine_stampate=pag_new WHERE
id=uid;
END
```

Triggers

Il seguente trigger si occupa di copiare in una tabella di backup un indirizzo cancellato da un utente in modo che in caso di necessità sia sempre possibile risalire all'indirizzo di spedizione di un vecchio ordine.

```
DROP TRIGGER IF EXISTS `del_indirizzo`;
CREATE TRIGGER `del_indirizzo` AFTER DELETE ON `indirizzo`
 FOR EACH ROW BEGIN
INSERT INTO indirizzo_log VALUES (old.id_indirizzo, old.link_utente, old.nome,
old.cognome, old.via, old.citta, old.regione, old.CAP);
END
```

Il trigger seguente è anch'esso sulla tabella indirizzo, il trigger viene invocato quando un utente inserisce un indirizzo, esso controlla che l'utente non abbia già raggiunto il limite di indirizzi (cioè 3), in tal caso solleva un errore.

```
DROP TRIGGER IF EXISTS `max_3_indirizzi`;
CREATE TRIGGER `max_3_indirizzi` BEFORE INSERT ON `indirizzo`
 FOR EACH ROW BEGIN
DECLARE nr_indirizzi INT;
SET nr_indirizzi = 0;
SELECT COUNT(*) INTO nr_indirizzi FROM indirizzo WHERE indirizzo.link_utente =
NEW.link_utente;
IF (nr_indirizzi = 3) THEN
CALL raise_error;
END IF;
END
```

Il seguente trigger è quello che ha al suo interno la chiamata alle due procedure descritte precedentemente, si occupa di verificare lo stato degli ordini ed in base ad esso di chiamare la corretta procedura.

```
DROP TRIGGER IF EXISTS `gestisci_soldi_pagine`;
CREATE TRIGGER `gestisci_soldi_pagine` AFTER UPDATE ON `ordine`
 FOR EACH ROW BEGIN
DECLARE get_pagine_finali int(6);
SELECT pagine_finali INTO get_pagine_finali FROM stampa JOIN ordine ON
id_stampa=stampa_ordine WHERE id_stampa = new.stampa_ordine;
IF new.stato = 'Rimborsato' THEN
CALL sub_soldi_pagine(new.prezzo, new.utente_ordine, get_pagine_finali);
ELSEIF new.stato='Concluso' THEN
CALL add_soldi_pagine(new.prezzo, new.utente_ordine, get_pagine_finali);
END IF;
END
```

Query significative presenti nel progetto

Lista dei file caricati ma mai passati attraverso il procedimento di modifica del formato e creazione della stampa da almeno 24h:

```
SELECT id_file, title
FROM (files LEFT JOIN stampa ON id_file=file_stampa)
WHERE file_stampa IS NULL  AND timestamp + INTERVAL 1 DAY <= NOW();

id_file     title
5           prova.pdf
```

Lista dei file presenti in una stampa ma mai "evoluti" in un ordine da almeno 24h:

```
SELECT id_file, title
FROM ((files RIGHT JOIN stampa ON id_file=file_stampa)
            LEFT JOIN ordine ON id_stampa=stampa_ordine)
WHERE id_ordine IS NULL AND timestamp + INTERVAL 1 DAY <= NOW();

id_file     title
5           prova.pdf
```

Query significative non presenti nel progetto

Lista degli utenti che hanno caricato un file contenente la parola *pirata* e con più di 10 pagine che è stato utilizzato in una stampa la quale è stata utilizzata in un ordine.

```
SELECT username FROM (((utente INNER JOIN ordine ON id=utente_ordine) INNER JOIN
stampa ON stampa_ordine=id_stampa) INNER JOIN files ON file_stampa=id_file)
WHERE title LIKE '%pirata%'
GROUP BY title
HAVING AVG(pagine)>10

username
user
```

Utenti di cui I file caricati hanno dimensione uguale all'Id di un ordine uniti agli utenti il cui id è identico al prezzo di un ordine in cui è stato usato un coupon con uno sconto del 10%

```
SELECT username FROM ((utente INNER JOIN files as F ON id=F.proprietario) INNER
JOIN ordine ON F.size=id_ordine)
UNION
SELECT username FROM ((utente INNER JOIN ordine as O ON id=prezzo) INNER JOIN
coupon on O.coupon_ordine=codice_coupon) WHERE sconto = 10

username
user
```

Parte WEB, credenziali, istruzioni, avvertenze.

Il sito si apre con la pagina di login, non ci si può registrare, **le credenziali di login sono: admin:admin e user:user**, in base al tipo di utente si verrà rimandati al proprio pannello di controllo admincp.php oppure usercp.php, le due interfacce non sono accessibli da utenti non appartenenti alla categoria per cui l'interfaccia è riservata.

La tabella formato dovrebbe essere una tabella in cui sono elencati tutti i possibili formati che un utente può scegliere, tuttavia ne sono elencati solo 40 e non 120, per cui nella pagina in cui viene chiesto il formato consiglio di non scegliere nel campo "rilegatura" i valori "Lato corto" e "Lato lungo", ma di lasciarla come è impostata di base cioè a "No". Il resto delle combinazioni è presente nel database ed è quindi funzionante.

Nell'interfaccia web non si considerano i tempi di spedizioni e la spedizione stessa, un ordine confermato da un admin equivale ad una virtuale ricezione da parte dell'utente immediata del prodotto senza passare per numeri di tracking e corrieri e rende possibile quindi richiedere immediatamente un rimborso. Non vengono fatti controlli sull'esistenza o meno degli indirizzi degli utenti presumendo che sia a favore dell'utente inserire indirizzi esistenti. I pagamenti vengono effettuati attraverso l'inserimento di una pseudoemail di paypal.

Nel sito sono presenti due utenti, alcuni ordini in sospeso, alcuni file in sospeso, un ordine concluso, un indirizzo ed un coupon col codice "START" che offre lo sconto del 10%.

Il codice php non verrà riportato nella relazione in quanto eccessivo.