Esame di stato 2007 libreria online

Traccia

Un negozio on line chiede che sia progettato e realizzato un database per l'organizzazione e la gestione di un portale per la vendita di libri su Internet.
Il negozio richiede che:

- il Catalogo dei libri sia organizzato per:
  - Reparti (in ordine alfabetico: *architettura e urbanistica, arte, classici greci e latini, cucina e casa, diritto, economia e management, fantascienza e fantasy, filosofia, fumetti, ...*)
- ciascun Reparto sia organizzato in libri:
  - "Da non perdere" caratterizzati da uno sconto del 20% sul prezzo di copertina
  - "I più venduti" caratterizzati dal maggior numero di copie vendute negli ultimi 30 giorni
  - "Ultimi arrivi" caratterizzati dai titoli aggiunti negli ultimi 30 giorni
  - "Offerte speciali" caratterizzate da uno sconto del 25% sul prezzo di copertina
  - "Remainders" caratterizzati da uno sconto del 50% sul prezzo di copertina
- ciascun libro in negozio sia archiviato con le seguenti ulteriori informazioni:
  - titolo, autore, prezzo di copertina, sconto (eventuale), dati (numero di pagine, rilegato, ...),anno di pubblicazione, editore, collana, immagine (foto della copertina), data di archiviazione
- gli utenti abilitati all'acquisto on line siano registrati con i seguenti dati:
  - nome e cognome, indirizzo, codice di avviamento postale, città, nazione, telefono, fax, e-mail, password, numero di carta di credito, tipo di carta di credito (Visa, CartaSi, Mastercard, ...), data di scadenza della carta di credito
- gli utenti possano chiedere, in fase di acquisto, che il negozio emetta fattura
- il Carrello Acquisti sia organizzato come segue:
  - numero di copie acquistate, titolo, autore, prezzo, disponibilità (giorni, settimane, ...), togli dal carrello (casella di spunta)
- il riepilogo degli ordini sia organizzato come segue:
  - data dell'ordine, numero d'ordine
  - dettagli dell'ordine (riferimento articolo, quantità, titolo, prezzo di copertina, prezzo scontato, stato dell'ordine, evasione prevista dalla data dell'ordine).

Il candidato, dopo aver fatto le eventuali ipotesi aggiuntive:

a. Fornisca:
   1. lo schema concettuale e lo schema logico del database
   2. la definizione delle relazioni in linguaggio SQL.

b. Implementi le seguenti query:
   1. *Ricerca per Reparto*: scelto un Reparto, il numero di libri "Da non perdere" ed il numero di libri "I più venduti" con i relativi dettagli
   2. *Ordini*: gli ordini in corso, con dettagli, di un dato utente.

c. Scriva in un linguaggio lato server, il codice di almeno una delle seguenti pagine del portale:
   1. con accesso libero, la pagina utile a visualizzare i Reparti e, per ciascun reparto, la pagina che implementa la query n. 1;
   2. con accesso riservato agli utenti registrati, la composizione degli ordini in corso.

Analisi
La traccia propone la realizzazione di un portale WEB e tutte le indicazioni proposte fanno riferimento più agli aspetti di presentazione del portale che alla reale struttura della banca dati che va quindi dedotta dalle specifiche richieste e non ricavata direttamente dalle informazioni fornite.
In particolare:

- Il "catalogo dei libri" non è una entità ma l'obiettivo principale di presentazione del portale.
- I "nomi dei reparti" costituiscono un esempio di una possibile suddivisione che può essere modificata nel tempo.
- L'"organizzazione dei reparti" è un esempio di una possibile categorizzazione nell'ambito del reparto che può essere modificata nel tempo sia in termini di struttura (altre categorie) che di contenuti (variazione degli sconti)
- Il "carrello degli acquisti" non è una struttura da archiviare permanentemente in banca dati ma un insieme di informazioni legate alla sessione di lavoro dell'utente autenticato.
- Il "riepilogo degli ordini" è una particolare "vista" dei dati archiviati di ciascun utente.

Ipotesi sulla struttura delle organizzazioni librarie
I libri pubblicati in tutto il mondo sono identificati in modo univoco dal codice ISBN. Si tratta di un codice parlante numerico a 13 cifre che identifica in modo gerarchico area del linguaggio,editore e titolo. Questo identificativo si presenta come una ottima chiave candidata per l'identificazione dei libri. Da questo punto di vista un libro è associato ad un solo editore anche se è possibile che titoli uguali siano pubblicati da diversi editori ma sono visti come libri distinti (potrebbero differire per traduzione, rilegatura, costo …).
Non è invece richiesta dal problema l'identificazione delle singole copie di un libro, problema che si presenterebbe invece in una biblioteca che deve gestire i prestiti, e neppure la determinazione delle giacenze in magazzino.
Una "collana editoriale" è un insieme di testi appartenenti allo stesso genere o riguardanti lo stesso argomento o comunque aventi una caratteristica comune, pubblicati da una casa editrice, identificati dal nome dato alla collana stessa e da una veste tipografica comune, in genere diversa da quella di altre collane della stessa casa editrice.
Un libro può essere scritto da più autori quindi si presenta una situazione di molteplicità. Sebbene questa soluzione non sia esplicitamente richiesta è da considerarsi indispensabile per una efficace indicizzazione dei libri, prestazione fondamentale in una libreria ondine.
Le immagini di copertina vengono archiviate sotto forma di dati BLOB.

Accesso autenticato
Per la sezione autenticata si prevede una autenticazione di sessione (server side) basata su cookies client side.
Alla prima apertura di una sessione il primo script che viene invocato registra un cookie di sessione nel browser, il quale successivamente lo trasmetterà a tutti gli script della stessa sessione.
Alla prima apertura di sessione viene anche effettuata la richiesta delle credenziali.
Le credenziali per l'accesso sono registrate nella banca dati stessa sotto forma di id_utente e password cifrata con la codifica MD5.
Il superamento della verifica delle credenziali porta al mantenimento della sessione mediante la trasmissione da parte del client del cookie client side e da parte del server delle corrispondenti variabili di sessione server side che garantiscono la validità delle credenziali.
In questo modo dopo una prima autenticazione l'utente ha accesso alle altre pagine autenticate senza dovere rinnovare l'inserimento delle credenziali a patto di non chiudere il browser.
Il logout o la chiusura del browser porta alla rimozione delle variabili di sessione e quindi alla perdita delle credenziali.
Oltre all'autenticazione dei clienti del servizio è necessario prevedere una separata autenticazione per gli amministratori del servizio che hanno accesso, diversamente dai clienti alle pagine di manutenzione della banca dati. Per questa autenticazione si prevede una separata tabella in banca dati indipendente da quella dei clienti. Tale tabella, non collegata alle altre entità, non verrà riportata in banca dati.

Iscrizione al servizio
L'iscrizione viene effettuata autonomamente dall'utente che compila una form con i dati.
Se i dati sono corretti l'iscrizione viene registrata e vengono presentati all'utente un nome utente (id numerico che costituisce la chiave primaria della tabella utenti) ed una password generata automaticamente con un randomizzatore.

Carrello
Il carrello viene realizzato come variabile temporanea di sessione.
Utilizzando la sessione già esistente a causa dell'autenticazione viene creato un array associativo di sessione denominato 'carrello'. La chiave associativa di ciascun elemento dell'array 'carrello' è costituita dalla chiave primaria del libro che si intende acquistare mentre il valore è costituito dalla quantità di libri che si intendono acquistare con quella chiave primaria. La pagina degli acquisti prevede, tramite la trasmissione a se stessa di opportuni parametri di GET, la possibilità di eseguire le seguenti operazioni sull'array 'carrello':

- inserimento nel carrello di un libro che non era presente (creazione di un nuovo elemento nell'array con chiave associativa pari alla chiave primaria del libro e valore 1 `$_SESSION['carrello'][$isbn]=1;`)
- aggiunta al carrello di un libro che era già presente (incremento di uno del valore memorizzato nell'array in corrispondenza della chiave primaria del libro `$_SESSION['carrello'] [$isbn]++;`)
- rimozione di tutte le copie di un libro dal carrello (cancellazione dell'elemento di corrispondente chiave primaria dall'array `unset($_SESSION['carrello'][$isbn]);`)
- svuotamento del carrello (cancellazione dell'intero array di sessione 'carrello' `unset($_SESSION['carrello']);`)

Nell'array di sessione 'carrello' sono memorizzati solo la chiave primaria (chiave associativa dell'array) e la quantità (valore dell'elemento corrispondente) perché tutti gli altri dati di presentazione richiesti nel carrello sono estratti dalla banca dati.
La conferma dell'ordina determina l'archiviazione dei dati del carrello nell'archivio permanente degli ordini.
La chiusura della sessione porta alla perdita dei dati del carrello.

Gestione degli ordini
Al momento della conferma di un carrello le informazioni contenute nel carrello diventano un ordine confermato. L'ordine è un dato di tipo storico (archiviazione permanente) contenente attributi multipli (i libri che fanno parte dell'ordine con le loro quantità e il loro prezzo storico, cioè del momento dell'ordine). E' quindi necessario realizzare una entità debole che individui univocamente ciascun ordine confermato legata 1:N agli utenti ed N:N ai libri.

Gestione della carta di credito
Per la effettiva gestione della carta di credito sia di tipo tradizionale sia di tipo a "micropagamento" è necessario che il portale si appoggi ad un "gateway bancario" .

Gestione delle immagini
Le immagini di copertina vengono archiviate sotto forma di dati BLOB. In questo modo si ottiene una totale congruenza della banca dati che non dipende da files esterni ma solo dai dati contenuti in banca dati facilitando le operazioni di backup, restore e porting del sistema.
L'upload delle immagini (solo di tipo JPG) viene effettuato dall'utente amministratore attraverso una form di upload. La risorsa disponibile sul file system locale viene trasferita in un file temporaneo del server identificato da un nome univoco.
Lo script di upload legge il file temporaneo in una variabile locale e la inserisce nella query di inserimento (creazione di un nuovo libro) o modifica (modifica di un libro esistente).
Per la presentazione della copertina nelle pagine pubbliche viene utilizzato un script accessorio che genera un'uscita con mime-type 'image/jpeg' parametrizzato in modo da potere estrarre una qualsiasi immagine di copertina dalla banca dati.

Gestione delle procedure temporizzate

Alcune informazioni richieste fanno riferimento a dati che cambiano automaticamente al passare del tempo (libri più venduti negli ultimi 30 giorni, titoli aggiunti negli ultimi 30 giorni). Questa funzione può essere realizzata con un "trigger" se il DBMS lo implementa altrimenti può essere realizzata con un funzione temporizzata del sistema operativo ospite (cron in ambiente UNIX/LINUX oppure "operazioni pianificate" in ambiente Windows).
Nel primo caso la procedura, lanciata ogni giorno ad un orario prefissato, estrae dagli ordini i conteggi relativi alle vendite di ciascun libro con restrizione agli ultimi 30 giorni e modifica aggiungendo, mantenendo o togliendo le associazioni tra libri e categoria "Più venduti".
Nel secondo caso un'altra procedura, anch'essa lanciata una volta al giorno, verifica la scadenza dei 30 giorni dall'inserimento in banca dati togliendo l'associazione alla categoria "Ultimi arrivi" che era stata applicata per default al primo inserimento.

Individuazione delle entità.

Entità '*utente*': è l'anagrafica degli utenti registrati. Mentre la consultazione del portale è libera per potere effettuare è necessario autenticarsi mediante un nome utente e password che sono forniti su richiesta dell'utente mediante la compilazione di una form.
Gli attributi semplici dell'entità sono:

- *id_utente:*chiave primaria numerica ad autoincremento che viene comunicata all'utente con la conferma di registrazione. Questo numero verrà usato dall'utente come username per gli accessi autenticati.
- *Password*: testo che memorizza in forma cifrata con codifica MD5 la password assegnata all'utente; generata automaticamente da un algoritmo di randomizzazione al momento della registrazione. Viene comunicata in chiaro all'utente con la conferma della registrazione.
- *cognome*:testo che contiene il cognome inserito dall'utente (obbligatorio)
- nome: testo che contiene il nome inserito dall'utente (obbligatorio)
- *citta*: testo che contiene la città di residenza inserita dall'utente (obbligatorio)
- *cap*: testo che contiene il cap della città di residenza inserito dall'utente (obbligatorio)
- *email*: testo che contiene un indirizzo email inserito dall'utente (facoltativo)
- telefono: testo che contiene il numero di telefono inserito dall'utente (facoltativo)
- *fax*: testo che contiene il numero di fax inserito dall'utente (facoltativo)
- *numero_carta*: testo che contiene il numero di carta inserito dall'utente (obbligatorio)
- *data_scadenza*: data di scadenza della carta in inserita dall'utente (obbligatorio)

Gli attributi semplici città e cap non vengono normalizzati perché nell'ipotesi di una gestione internazionale del portale non è pensabile prevedere una look-up per tutte le città e cap del mondo.
L'attributo semplice 'nazione' viene normalizzato e diventa una tabella di look-up associata 1:N con l'entità utente.
L'attributo semplice 'carta' viene normalizzato e diventa una tabella di look-up associata 1:N con l'entità utente.
SOLUZIONE ALTERNATIVA: si può ipotizzare di non normalizzare l'attributo 'nazione' perdendo però il controllo di congruenza sui nomi nazione.
SOLUZIONE ALTERNATIVA: si può ipotizzare di non normalizzare l'attributo 'carta' perdendo però il controllo di congruenza sui tipi di carta.
SOLUZIONE ALTERNATIVA: si può ipotizzare di consentire il pagamento con più di una carta di credito. In questo caso è necessario prevedere una associazione N:N tra utente e carta. Gli attributi numero_carta e data_scadenza diventano multipli e quindi vanno spostati nell'associazione N:N.

Entità '*nazione*': è una tabella di look-up contenente l'insieme delle nazioni di residenza. Ogni nazione è identificata univocamente da una chiave artificiale a codice parlante definita dalle norme del codice fiscale. Il codice è alfanumerico formato dal un carattere alfabetico 'Z' seguito da una stringa numerica di tre cifre (ad esempio Albania è 'Z100', Sri Lanka è 'Z209').
Ad ogni codice è associata una descrizione.

Entità '*carta*': è una tabella di look-up contenente l'insieme delle carte di credito accettate dal portale. Ogni carta è identificata univocamente da una chiave artificiale numerica ad autoincremento. Ad ogni codice è associata una descrizione.

Entità '*libro*': è l'anagrafica dei libri presenti nel catalogo.
Gli attributi semplici dell'entità sono:

- *isbn*:chiave primaria di tipo testo. Si tratta di un codice parlante numerico a 13 cifre che identifica in modo gerarchico area del linguaggio,editore e titolo. Il codice identifica una particolare edizione di un particolare autore del libro quindi lo stesso libro dal punto di vista di autore, titolo e contenuti può avere registrazioni distinte.
- *titolo*: testo che contiene il titolo del libro (obbligatorio).
- *prezzo*:archiviazione di tipo valuta che contiene il prezzo attuale del libro.
- *sconto*: intero che contiene la percentuale di sconto applicato attualmente al libro. Questo sconto, che è facoltativo (default 0), è applicato individualmente al libro e quindi si somma ad eventuali altri sconti applicati in base all'appartenenza a particolari categorie.
- *n_pagine*: intero che contiene il numero di pagine del libro nella corrente edizione(obbligatorio).
- *rilegato*: informazione logica che distingue i testi rilegati da quelli economici(obbligatorio).
- *anno_pubbl*: data di pubblicazione del libro (obbligatorio).
- *immagine*: BLOB che contiene l'immagine di copertina del libro (obbligatorio).
- *tempo_consegna*: intero che contiene il numero di giorni necessari per la consegna.
- *data_archivio*: timestamp che viene generato automaticamente all'inserimento della registrazione e contiene la data di inserimento.

L'attributo semplice 'collana' viene normalizzato e diventa una tabella di look-up associata 1:N con l'entità libro.
Si ipotizza che una collana sia edita da una sola casa editrice e quindi viene eliminato l'attributo semplice 'editore' che viene associato indirettamente attraverso la collana in cui il libro è inserito.
Ogni libro si trova in un solo reparto e quindi il libro è associato 1:N all'entità reparto.
Tutti i reparti sono divisi in categorie nello stesso modo, viene quindi realizzata una associazione diretta 1:N tra libro e categoria nell'ipotesi che un libro possa appartenere ad una sola categoria.
SOLUZIONE ALTERNATIVA: si può ipotizzare che un libro possa appartenere contemporaneamente a più categorie. In tale caso l'associazione tra libro e categoria diventa N:N ma è necessario prevedere vincoli espliciti in modo da evitare incongruenze.

Entità '*autore*': è l'anagrafica degli autori dei libri. Ogni autore è identificato univocamente da una chiave artificiale numerica ad autoincremento. Gli attributi semplici sono i testi 'cognome' e 'nome'.

Entità '*collana*': è una tabella di look-up contenente l'insieme delle collane editoriali disponibili. Ogni collana è identificata univocamente da una chiave artificiale numerica ad autoincremento. Per ogni collana esiste un attributo semplice descrizione. Ogni collana è edita da un editore quindi l'entità è associata 1:N all'entità editore. In questo modo ogni libro è indirettamente associato ad un editore.

Entità '*editore*': è una tabella di look-up contenente l'insieme degli editori disponibili. Ogni editore è identificato univocamente da una chiave artificiale numerica ad autoincremento. Per ogni editore esiste un attributo semplice descrizione.

Entità '*reparto*': è una tabella di look-up contenente l'insieme dei reparti presenti nel portale. Ogni reparto è identificato univocamente da una chiave artificiale numerica ad autoincremento. Per ogni reparto esiste un attributo semplice descrizione. Poiché tutti i reparti sono organizzati nello stesso modo il libri sono associati direttamente sia al reparto sia alla categoria organizzativa senza l'inserimento di gerarchie.

Entità '*categoria*': è una tabella di look-up contenente l'insieme delle categorie organizzative presenti nel portale. Ogni categoria è identificata univocamente da una chiave artificiale numerica ad autoincremento. Per ogni categoria un attributo semplice descrizione ed un attributo numerico facoltativo sconto.
Si ipotizza che ogni libro possa appartenere solo ad una singola categoria ma un libro può anche non appartenere ad alcuna categoria.
Quando un libro viene inserito per in archivio viene associato automaticamente nella categoria 'ultimi arrivi' e l'associazione viene rimossa automaticamente dallo procedura temporizzata che viene eseguita ogni giorno quando sono passati 30 giorni dal suo timestamp.
Quando un libro supera la soglia di vendita fissata dalla procedura temporizzata viene automaticamente inserito nella categoria 'più venduti' e quindi viene rimosso da ogni altra categoria a cui apparteneva in precedenza.
Queste due prime categorie non prevedono uno sconto.
L'inserimento nelle categorie scontate è manuale.

Individuazione delle associazioni.

Associazione utente/nazione '*risiede*': ogni utente deve essere associato ad una nazione di residenza selezionata dall'elenco di lookup delle nazioni. Esiste quindi una associazione 1:N tra nazione ed utente. La relazione è parziale dal lato nazione (una nazione può non avere alcun utente residente) e totale dal lato utente (un utente deve risiedere in una nazione)

Associazione utente/carta '*tipo_carta*': ogni utente deve possedere una carta di credito selezionata dall'elenco di lookup delle carte di credito previste dal portale.i comuni del territorio di interesse della community. Esiste quindi una associazione 1:N tra carta e utente. La relazione è parziale dal lato carta (un tipo di carta può non essere usato da alcun utente) e totale dal lato utente (un utente deve possedere un tipo di carta)

Associazione libro/autore '*libro_autore*': ogni libro può essere scritto da uno o più autori, ogni autore può avere scritto nessuno, uno o più libri. Esiste quindi una associazione N:N tra autore e libro. La relazione è parziale dal lato autore (un autore può non avere scritto alcun libro) e totale dal lato libro (un libro deve essere stato scritto da un autore)

Associazione libro/collana '*inserito_in*': ogni libro deve essere inserito in una collana che a sua volta è edita da un editore.. Esiste quindi una associazione 1:N tra collana e libro. La relazione è totale dal lato libro (un libro deve essere inserito in una collana) e parziale dal lato collana (una collana può non avere alcun libro)

Associazione collana/editore '*edita_in*': ogni collana deve essere edita da un editore. Esiste quindi una associazione 1:N tra editore e collana. La relazione è totale dal lato collana (una collana deve essere edita da un editore) e parziale dal lato editore (una editore può non avere alcuna collana)

Associazione libro/reparto '*si_trova_in*':ogni libro deve essere inserito in un reparto. Esiste quindi una associazione 1:N tra reparto e libro. La relazione è totale dal lato libro (un libro deve essere inserito in un reparto) e parziale dal lato reparto (un reparto può non avere alcun libro)

Associazione libro/categoria '*appartiene_a*': ogni libro può essere inserito in una categoria. Esiste quindi una associazione 1:N tra categoria e libro. La relazione è parziale da entrambi i lati.

Entità debole '*ordine*': quando un carrello viene confermato diventa ordine. Ogni ordine oltre alle informazioni generali ha come attributi multipli i riferimenti ai libri acquistati quindi viene realizzato come entità debole associata 1:N all'utente ed N:N al libro.
Gli attributi semplici sono:

- *n_ordine*: chiave primaria numerica ad autoincremento che identifica l'ordine attraverso un numero progressivo
- *data_ord*: data di emissione dell'ordine generata come data corrente al momento della conferma dell'ordine
- *data_prev*: data prevista di consegna calcolata come la data più avanzata ottenuta sommando alla data corrente il tempo di consegna di ciascun libro contenuto nell'ordine.
- *fattura*: informazione logica che memorizza la richiesta di emissione della fattura.
- *stato*: stato di avanzamento dell'ordine sotto forma di un set di dati compreso tra i seguenti valori:
    - IN_CORSO
    - CONSEGNATO
    - ANNULLATO

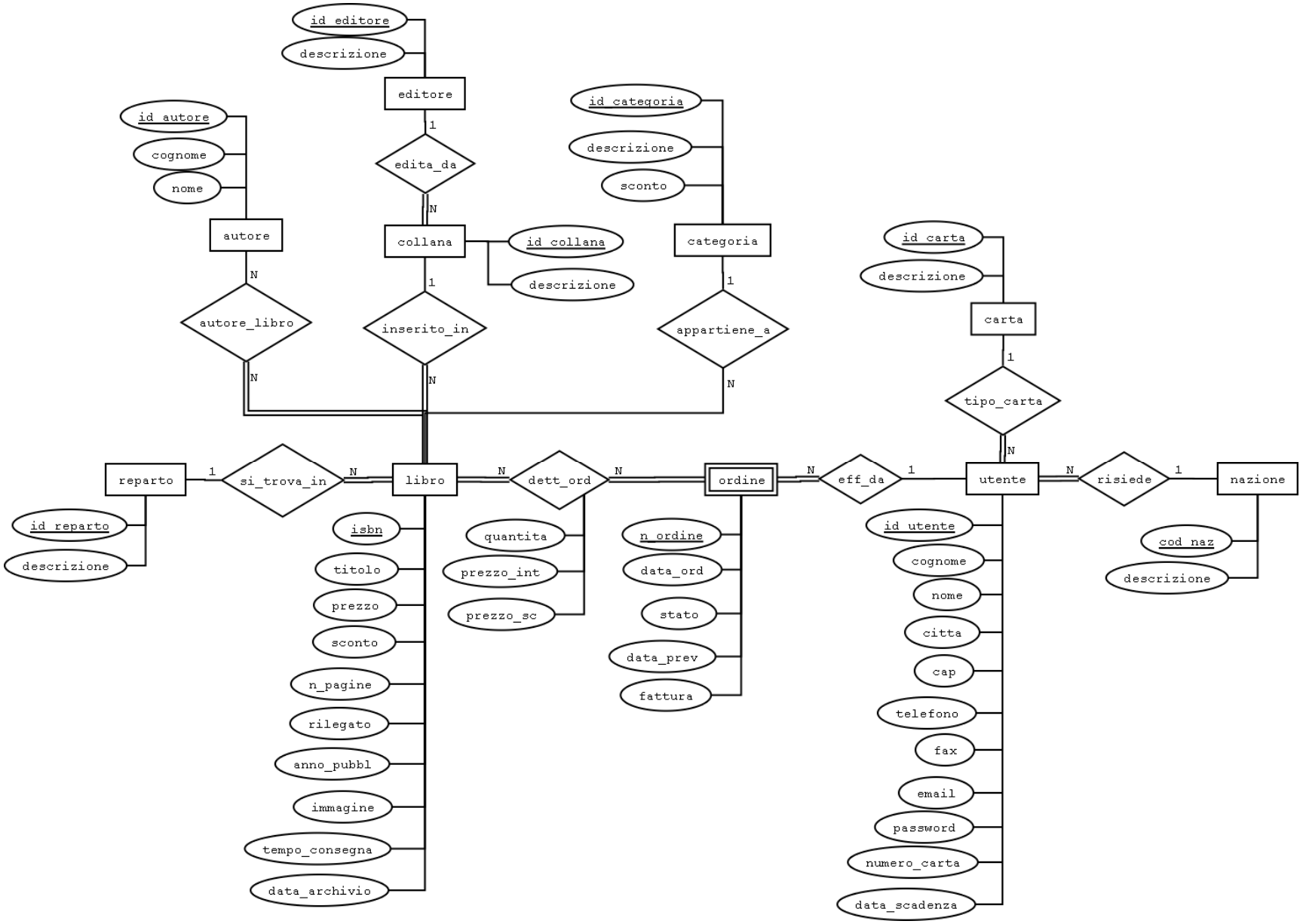
Ogni ordine deve essere associata ad un utente attraverso una associazione 1:N
I libri acquistati con un ordine costituiscono attributo multiplo quindi qgni ordine deve essere associato ad uno o più libri attraverso una associazione N:N; questa associazione possiede gli attributi multipli quantità prezzo intero e prezzo scontato.

Associazione ordine/utente '*eff_da*': ogni ordine deve essere associato all'utente che l'ha effettuato. Un utente può effettuare più ordini ma un ordine può essere effettuato solo da un singolo utente. Esiste quindi una associazione 1:N tra utente ed ordine. La relazione è parziale dal lato utente (un utente può non avere effettuato alcun ordine) e totale dal lato ordine (un ordine deve essere stato effettuato da un utente)

Associazione ordine/libro 'dett_ord': questa associazione N:N definisce il dettaglio degli ordini, cioè effettua i riferimenti tra un ordine e le sue parti con i dati specifici di ogni parte.
Un ordine può essere composto di uno o più libri e un libro può essere inserito in nessuno, uno o più ordini.
La relazione è parziale dal lato libro (un libro può non essere presente in alcun ordine) e totale dal lato ordine (un ordine deve avere almeno un libro)
Gli attributi dell'associazione sono:
quantità: numero di copie di un certo libro acquistate con un certo ordine
prezzo_int: valuta che indica il prezzo intero che il libro aveva al momento dell'ordine (dato storico)
prezzo_sc: valuta che indica il prezzo scontato che il libro aveva al momento dell'ordine dopo l'applicazione degli eventuali sconti attivi al momento dell'ordine (dato storico)

SOLUZIONE ALTERNATIVA: invece che realizzare il carrello mediante un array di sessione si può sfruttare il fatto che gli acquisti sono riservati esclusivamente agli utenti registrati per creare una entità debole 'carrello' avente una struttura coerente a quella dell'ordine. Al momento della conferma l'istanza di carrello viene copiata nell'entità ordine diventando un ordine confermato.
L'istanza carrello va rimossa in uscita dalla sessione perché i suoi dati hanno validità solo temporanea.
SOLUZIONE ALTERNATIVA: invece che realizzare il carrello mediante un array di sessione si può sfruttare il fatto che gli acquisti sono riservati esclusivamente agli utenti registrati per registrare il carrello come ordine aggiungendo il valore NON_CONFERMATO nello stato.
Al momento della conferma lo stato viene cambiato in IN_CORSO.
L'istanza di ordine NON_CONFERMATO va rimossa in uscita dalla sessione perché i suoi dati hanno validità solo temporanea.

Schema concettuale (modello ER)



Schema logico
Si usa il modello relazionale. Ogni entità ed entità debole viene sostituita con una relazione. Ogni associazione 1:N tra due entità viene sostituita con l'esportazione della chiave primaria del lato 1 come chiave esterna nel lato N. Ogni associazione N:N tra due entità viene sostituita con la generazione di una nuova relazione ottenuta esportando le due chiavi primarie come chiavi esterne e definendo la loro composizione come chiave primaria della nuova relazione.

Relazioni

utente(id_utente:intero,
password: testo,
cognome: testo,
nome: testo,
citta:testo,
cap: testo,
email: testo,
telefono:testo,
fax: testo,
numero_carta: testo,
data_scadenza: data,
id_carta: intero,
cod_naz: testo)

nazione(cod_naz:testo,
descrizione:testo)
carta(id_carta:intero,
descrizione: testo)

libro(isbn:testo,
titolo: testo,
prezzo: valuta,
sconto: intero,
n_pagine: intero,
rilegato: booleano,
anno_pubbl: data,
immagine: blob,
tempo_consegna: intero,
data_archivio: data,
id_collana: intero,
id_categoria: intero,
id_reparto: intero)

autore(id_autore:intero,
cognome:testo,
nome:testo)

libro_autote(isbn: testo,
id_autore:intero)

collana(id_collana:intero,
descrizione: testo,
id_editore: intero)

editore(id_editore:intero,
descrizione: testo)

reparto(id_reparto:intero,
descrizione: testo)

categoria(id_categoria:intero,
descrizione: testo,
sconto:intero)

ordine(n_ordine:intero,
data_ordine: data,
stato:set,
data_prev: data,
fattura: booleano
id_utente: intero)

dett_ord(n_ordine:intero,
isbn:testo,
quantità: intero,
prezzo_int: valuta,
prezzo_sc: valuta)

Vincoli di integrità

Vincoli di integrità referenziale

utente.cod_naz ⊆ nazione.cod_naz
utente.id_carta ⊆ carta.id_carta
utente.id_gruppo ⊆ gruppo.id_gruppo
libro.id_collana ⊆ collana.id_collana
collana.id_editore ⊆ editore.id_editore
libro.id_reparto ⊆ reparto.id_reparto
libro_autore.id_libro ⊆ libro.id_libro
libro_autore.id_autore ⊆ autore.id_autore
libro.id_categoria ⊆ categoria.id_categoria | ∅
ordine.id_utente ⊆ utente,id_utente
dett_ord.n_ordine ⊆ ordine.n_ordine
dett_ord.isbn ⊆ libro.isbn

Modello fisico

```
CREATE TABLE utente (
     id_utente int(11) NOT NULL auto_increment,
     password varchar(32) NOT NULL default '',
     cognome varchar(80) NOT NULL default '',
     nome varchar(80) NOT NULL default '',
     citta varchar(80) NOT NULL default '',
     cap varchar(5) NOT NULL default '',
     telefono varchar(80) NOT NULL default '',
     fax varchar(80) NOT NULL default '',
     email varchar(255) NOT NULL default '',
     numero_carta varchar(20) NOT NULL default '',
     data_scadenza DATE NOT NULL,
     cod_naz varchar(4) NOT NULL default '',
     id_carta int(11) NOT NULL default '0',
     PRIMARY KEY (id_utente),
     KEY cod_naz(cod_naz),
     KEY id_carta(id_carta),
     FOREIGN KEY (cod_naz) REFERENCES nazione(cod_naz),
     FOREIGN KEY (id_carta) REFERENCES carta(id_cata)
);

CREATE TABLE nazione (
     cod_naz varchar(4) NOT NULL default '',
     descrizione varchar(80) NOT NULL default '',
     PRIMARY KEY (cod_naz)
);

CREATE TABLE carta (
     id_carta int(11) NOT NULL auto_increment,
     descrizione varchar(20) NOT NULL default '',
     PRIMARY KEY (id_carta)
);

CREATE TABLE libro (
     isbn varchar(13) NOT NULL auto_increment,
     titolo varchar(255) NOT NULL default '',
     prezzo decimal(5,2) NOT NULL default 0,
     sconto int(11) NOT NULL default 0,
     n_pagine int(11) NOT NULL default 0,
     rilegato tinyint(1) NOT NULL default 0,
     anno_pubbl YEAR NOT NULL default '0000',
     immagine BLOB,
     tempo_consegna int(11) NOT NULL default 0,
     data_archivio TIMESTAMP
     id_reparto int(11) NOT NULL default '0',
     id_collana int(11) NOT NULL default '0',
     id_categoria int(11) NOT NULL default '0',
     PRIMARY KEY (isbn),
     KEY id_reparto(id_reparto),
     KEY id_collana(id_collana),
     KEY id_categoria(id_categoria),
     FOREIGN KEY (id_reparto) REFERENCES reparto(id_reparto),
     FOREIGN KEY (id_collana) REFERENCES carta(id_collana)
);

CREATE TABLE reparto (
     id_reparto int(11) NOT NULL auto_increment,
```

```
     descrizione varchar(20) NOT NULL default '',
     PRIMARY KEY (id_reparto)
);

CREATE TABLE collana (
     id_collana int(11) NOT NULL auto_increment,
     descrizione varchar(20) NOT NULL default '',
     id_editore int(11) NOT NULL default '0',
     PRIMARY KEY (id_collana),
     KEY id_editore(id_editore),
     FOREIGN KEY (id_editore) REFERENCES editore(id_editore)
);

CREATE TABLE editore (
     id_editore int(11) NOT NULL auto_increment,
     descrizione varchar(20) NOT NULL default '',
     PRIMARY KEY (id_editore)
);

CREATE TABLE categoria (
     id_categoria int(11) NOT NULL auto_increment,
     descrizione varchar(20) NOT NULL default '',
     sconto int(11),
     PRIMARY KEY (id_categoria)
);

CREATE TABLE autore (
     id_autore int(11) NOT NULL auto_increment,
     cognome varchar(20) NOT NULL default '',
     nome varchar(20) NOT NULL default '',
     PRIMARY KEY (id_autore)
);

CREATE TABLE libro_autore (
     id_autore int(11) NOT NULL,
     isbn varchar(13) NOT NULL,
     PRIMARY KEY (id_autore,isbn),
     FOREIGN KEY (id_autore) REFERENCES autore(id_autore),
     FOREIGN KEY (isbn) REFERENCES libro(isbn)
);

CREATE TABLE ordine (
     n_ordine int(11) NOT NULL auto_increment,
     data_ord DATE NOT NULL,
     data_prev DATE NOT NULL,
     stato set('IN_CORSO','CONSEGNATO','ANNULLATO') NOT NULL ,
     fattura TINYINT(1) NOT NULL default 0,
     id_utente int(11) NOT NULL default '0',
     PRIMARY KEY (n_ordine),
     KEY id_utente(id_utente),
     FOREIGN KEY (id_utente) REFERENCES utente(id_utente)
);

CREATE TABLE dett_ord (
     n_ordine int(11) NOT NULL,
     isbn varchar(13) NOT NULL,
     quantita int(11) NOT NULL,
     prezzo_int decimal(5,2) NOT NULL,
     prezzo_sc decimal(5,2) NOT NULL,
```

```
    PRIMARY KEY (n_ordine,isbn),
    FOREIGN KEY (n_ordine) REFERENCES ordine(n_ordine),
    FOREIGN KEY (isbn) REFERENCES libro(isbn)
);
```

Interrogazioni

B1: Ricerca per Reparto: scelto un Reparto, il numero di libri "Da non perdere" ed il numero di libri "I più venduti" con i relativi dettagli

Si effettua una giunzione naturale tra libro, reparto e categoria. Questa giunzione esclude tutti i libri che non hanno alcuna categoria. Una ulteriore restrizione riduce la query ai libri di un particolare reparto (architettura nell'esempio)

Sul risultato della giunzione viene applicata una partizione in base alla categoria e sul risultato della partizione viene effettuata una ulteriore restrizione per mantenere solo le partizioni relative alle due categorie coinvolte.

Nella proiezione viene effettuato il conteggio dei libri appartenenti a ciascuna categoria e vengono anche riportati i dati di dettaglio delle categorie selezionate (descrizione ed eventuale sconto)

```
SELECT
     t1.id_reparto,
     t1.descrizione,
     COUNT(t2.isbn) AS numero_libri,
     t3.descrizione,
     t3.sconto
FROM
     reparto AS t1,
     libro AS t2,
     categoria AS t3
WHERE
     t1.id_reparto=t2.id_reparto
AND  t2.id_categoria=t3.id_categoria
AND  t1.descrizione LIKE 'architettura%'
GROUP BY
     t3.descrizione
HAVING
     t3.descrizione LIKE 'Da non perdere'
OR   t3.descrizione LIKE 'I più venduti'
```

SOLUZIONE ALTERNATIVA
Si può spezzare la query in due query simili ciascuna riferita ad una singola categoria.
In questo caso non è necessaria la clausola HAVING

Q1A

```
SELECT
     t1.id_reparto,
     t1.descrizione,
     COUNT(t2.isbn) AS numero_libri,
     t3.descrizione,
     t3.sconto
FROM
     reparto AS t1,
     libro AS t2,
     categoria AS t3
WHERE
     t1.id_reparto=t2.id_reparto
AND  t2.id_categoria=t3.id_categoria
AND  t1.descrizione LIKE 'architettura%'
AND  t3.descrizione LIKE 'Da non perdere'
GROUP BY
     t3.descrizione
```

Q1A

```
SELECT
     t1.id_reparto,
     t1.descrizione,
     COUNT(t2.isbn) AS numero_libri,
     t3.descrizione,
     t3.sconto
FROM
     reparto AS t1,
     libro AS t2,
     categoria AS t3
WHERE
     t1.id_reparto=t2.id_reparto
AND  t2.id_categoria=t3.id_categoria
AND  t1.descrizione LIKE 'architettura%'
AND  t3.descrizione LIKE 'I più venduti'
GROUP BY
     t3.descrizione
```

B2: Ordini: gli ordini in corso, con dettagli, di un dato utente..
Si effettua una giunzione naturale tra utente che fornisce i dati dell'utente , ordine che fornisce i dati generali dell'ordine, dett_ord che fornisce i dettagli dell'ordine libro che fornisce i dati del libro.
Una restrizione limita agli ordini di uno specifico utente e agli ordini ancora in corso.
Poichè la query può essere usata solo in sessione l'id utente è preso dall'array di sessione.

```
SELECT
     t2.n_ordine,
     t2.data_ord,
     t2.data_prev,
     t2.fattura,
     t4.isbn,
     t4.titolo,
     t3.quantita,
     t3.prezzo_int,
     t3.prezzo_sc
FROM
     utente AS t1,
     ordine AS t2,
     dett_ord AS t3,
     libro AS t4
WHERE
     t1.id_utente=t2.id_utente
AND
     t2.id_ordine=t3.id_ordine
AND
     t4.id_libro=t3.id_libro
AND
     t1.id_utente=$_SESSION['id_utente']
AND
     t2.stato='IN_CORSO'
ORDER BY
     t2.n_ordine,t4.isbn
```

Premesse comuni per tutti gli script lato server

Ambiente operativo

Si ipotizza di usare una piattaforma xAMP composta da:

- x sistema operativo Windows o Linux
- A web server Apache
- M sql server MySQL
- P script interprete PHP

Connessione al database

Il server MySQL si presenta come un server TCP che risponde sulla porta 3306.
L'interpreter PHP dispone di una libreria di funzioni di interfaccia con il server MySQL.
Si può quindi realizzare uno script di connessione da includere in ogni script che necessiti dell'accesso alla banca dati (connect.php).

```
<?php
//connette al SQL server sullo stesso host,
//utente nobody, password 'qwerty'
$sock=mysql_connect('localhost','nobody','qwerty');
//se la connessione fallisce termina segnalando l'errore
if ($sock==0) die(mysql_error());
//seleziona il database sulla connessione
$ris=mysql_select_db('ebook',$sock);
//se la selezione fallisce termina segnalando l'errore
if ($ris==0) die(mysql_error());
?>
```

La connessione avviene da parte dell'agente (utente fittizio) 'nobody'. Affinché la connessione abbia successo è necessario che l'amministratore del database garantisca l'accesso all'utente nobody con il seguente DCL:

```
GRANT SELECT,INSERT,UPDATE,DELETE
ON ebook.*
TO nobody@localhost
IDENTIFIED BY 'qwerty'
```

Per motivi di sicurezza l'accesso alla banca dati è limitato agli script che si trovano sullo stesso host dell'sql server (localhost)
Gli script che necessitano di una connessione devono includere lo script di connessione:

```
<?php
//termina se l'inclusione fallisce
require 'connect.php'
?>
```

Autenticazione

Si ipotizza di realizzare una autenticazione di sessione PHP.
L'autenticazione di sessione si basa sulla creazione di una sessione all'avvio di ogni script:

```
<?php session_start(); ?>
```

L'avvio di una sessione rende disponibili allo script le variabili di sessione lato server contenute nell'array $_SESSION[]. La prima volta che viene lanciato uno script di sessione il server forza la registrazione di un cookie di sessione nel client (i cookies devono essere abilitati per l'host che fa la richiesta). Il client mantiene il cookie di sessione fino alla rimozione da parte del server o fino alla chiusura del browser e ogni volta che richiede una pagina via GET o POST invia anche il cookie che consente al server di estrarre le corrispondenti variabili lato server ($_SESSION[]). In questo modo due pagine di una stessa sessione condividono le stesse variabili globali. L'autenticazione di sessione si basa sulla verifica dell'esistenza di una variabile di sessione che identifica l'utente. Supponiamo di chiamare 'id_utente' la chiave associativa dell'elemento dell'array $_SESSION ($_SESSION['id_utente']) che identifica l'utente e supponiamo che ad autenticazione superata contenga il nome dell'utente. Ogni pagina che deve essere sottoposta ad autenticazione deve contenere il seguente script (session.php):

```
<?php
//crea una sessione o riprende una sessione già aperta
session_start();
//verifica se già autenticato
if (!isset($_SESSION['id_utente'])) { //non esiste 'username'
//non autenticato: redireziona alla form di login
header("location: login.php");
}
//se arrivo qui vuole dire che sono autenticato: mostro la pagina che mi include
?>
```

Se un'altra pagina della stessa sessione ha fatto l'autenticazione lo script non fa nulla altrimenti redireziona alla form di login. La form di login raccoglie l'username e la password (in chiaro) e le trasferisce ad uno script di verifica delle credenziali. Supponendo che le credenziali ricevute dalla form sia contenute nelle variabili $username e $password il seguente script controlla la validità delle credenziali:

```
<?php
//estrae dalla banca dati un eventuale utente con queste credenziali
$msg="SELECT t1.nome,t1.cognome AS utente
FROM utente AS t1
WHERE id_utente='$id_utente'
AND password=MD5('$password')";
$query=mysql_query($msg,$sock);
if ($row_user=mysql_fetch_assoc($query)) { //trovato accetta credenziali
//crea le variabili di autenticazione di sessione
$_SESSION['id_utente']=$id_utente;
$_SESSION['cognome']=$row_user['cognome'];
$_SESSION['nome']=$row_user['nome'];
}
else { //credenziali rifiutate: rilancia la form di login
header("Location: login.php");
}
?>
```

La selezione delle credenziali dalla banca dati è una ricerca per chiave primaria (id_utente) quindi produce 0 oppure 1 record in uscita; se si ottiene un record ($row_user non nullo) i valori di id_utente, cognome e nome vengono usati come variabili di sessione. Il primo serve per verificare se l'autenticazione ha avuto successo e chi è l'utente autenticato, il secondo serve per segnalare all'utente la permanenza in area autenticata ed il terzo serve per personalizzare le pagine in base al ruolo senza dovere reinterrogare la BD ogni volta. Se $row_user è nullo l'autenticazione è fallita (non si trova la coppia id_utente e cifratura della password in banca dati) quindi si torna alla form di autenticazione mediante una redirezione. Gli script che necessitano di autenticazione devono includere lo script di sessione:

```
<?php require 'session.php' //verif.sessione ed event. lancia login ?>
```

C1 con accesso libero, la pagina utile a visualizzare i Reparti e, per ciascun reparto, la pagina che implementa la query n. 1;
Si realizza una struttura di tipo master/detail.
Una pagina generale mostra un elenco di tutti i reparti e consente il collegamento ad una pagina di dettaglio che mostra i dati riferiti ad un singolo reparto.

Pagina master

```
<?php require 'connect.php' //connessione alla banca dati ?>
$msg="SELECT * FROM reparto";
$query=mysql_query($msg,$sock);
if($query==0) die(mysql_error());
?>
<html>
<head><title>query n.1</title></head>
<body>
<table>
<tr> <!-- riga statica di intestazione -->
<td>id_reparto</td>
<td>descrizione/td>
<td>dettagli</td>
</tr>
<?php while($row_rep=mysql_fetch_assoc($query)) { //riga dinamica ?>
<tr>
<td><?php echo $row_rep['id_reparto']) ?></td>
<td><?php echo stripslashes($row_rep['descrizione']) ?></td>
<td><a href="dett.php?rep=<?php echo $row_rep['id_reparto'])?>"</a></td>
</tr>
<?php } ?>
</table>
</body>
</html>
```

Pagina detail (dett.php)

```
<?php require 'connect.php' //connessione alla banca dati ?>
<?php if (isset($_GET[‘rep’] $rep=$ :GET[‘rep’] ;
else die(’ manca reparto’require ') ;
?>
$msg=" SELECT ... FROM ... WHERE ...
AND t1.descrizione LIKE '$rep%'
GROUP BY ... HAVING ... ";
$query=mysql_query($msg,$sock);
if($query==0) die(mysql_error());
?>
<html>
<head><title>query n.1</title></head>
<body>
<table>
<tr> <!-- riga statica di intestazione -->
<td>id_reparto</td>
<td>descrizione/td>
<td>numero libri</td>
<td>sconto</td>
</tr>
<?php while($row_rep=mysql_fetch_assoc($query)) { //riga dinamica ?>
<tr>
<td><?php echo $row_rep['id_reparto']) ?></td>
<td><?php echo stripslashes($row_rep['descrizione']) ?></td>
<td><?php echo $row_rep['id_reparto']) ?></td>
<td><?php echo $row_rep['sconto']) ?></td>
<td><a href="dett.php?rep=<?php echo $row_rep['id_reparto'])?>"</a></td>
</tr>
<?php } ?>
</table>
</body>
</html>
```

C2 con accesso riservato agli utenti registrati, la composizione degli ordini in corso.
La pagina può essere chiamata solo all'interno di una sessione di un utente registrato. La query utilizza quindi l'id di utente fornito dall'array di sessione.
La pagina mostra un elenco generale dei dettagli degli ordini dell'utente ordinati per ordine e nell'ambito di un ordine per isbn di libro.

```
<?php require 'connect.php' //connessione alla banca dati ?>
<?php require 'session.php' //verifica stato di autenticazione ?>
<?php
$msg=" SELECT ... FROM ... WHERE ... AND t1.id_utente=$_SESSION[‘id_utente’]";
$query=mysql_query($msg,$sock);
?>
<html>
<head><title>query n.2</title></head>
<body>
<?php //mostra dati utente autenticato
echo $_SESSION['id_utente']."-".$_SESSION['cognome']."-".$_SESSION['nome']
?>
<table>
<tr> <!-- riga statica di intestazione -->
<td>n. ordine</td>
<td>data ordine</td>
<td>data cons.</td>
<td>isbn</td>
<td>titolo</td>
<td>quantità</td>
<td>prezzo intero</td>
<td>prezzo scontato</td>
</tr>
<?php while($row_ord=mysql_fetch_assoc($query)) { //riga dinamica ?>
<tr>
<td><?php echo $row_ord['n_ordine'] ?></td>
<td><?php echo $row_ord['data_ord'] ?></td>
<td><?php echo $row_ord['data_prev'] ?></td>
<td><?php echo $row_ord['fattura'] ?></td>
<td><?php echo $row_ord['isbn'] ?></td>
<td><?php echo stripslashes($row_ord['titolo']) ?></td>
<td><?php echo $row_ord['quantita'] ?></td>
<td><?php echo $row_ord['prezzo_int'] ?></td>
<td><?php echo $row_ord['prezzo_sc’] ?></td>
</tr>
<?php } ?>
</table>
</body>
</html>
```