RELAZIONE PROGETTO BASI DI DATI

Aronne Brivio – 831154

1) Introduzione

Il progetto prevede la progettazione e realizzazione di una base di dati con rispettivo portale web associato per la gestione di tornei sportivi di quattro tipologie differenti: all’italiana, ad eliminazione diretta, misto e libero.

In particolare i tornei all'italiana sono strutturati in modo tale che tutti i partecipanti si scontrino una e una sola volta contro tutti gli altri iscritti. Per questo tipo di torneo è prevista una situazione di parità ed in base al risultato della gara viene alimentata la classifica dei giocatori assegnando al singolo giocatore 3 punti per ogni vittoria, 1 per ogni pareggio e 0 per ogni sconfitta.
I vincitori del torneo saranno i giocatori che hanno totalizzato un punteggio pari o maggiore di tutti gli altri avversari, ciò significa che per ogni torneo all'italiana può esserci più di un vincitore.

I tornei a eliminazione diretta non prevedono invece il pareggio in una singola gara, né più di un vincitore per torneo. Le singole gare sono strutturate in fasi: la prima fase è costituita da n/2 gare, con n pari al numero di partecipanti al torneo. Accedono ad ogni fase successiva i vincitori delle gare della fase precedente, fino ad arrivare all'ultima fase, composta da una sola gara, da cui emergerà il vincitore del torneo.
Nel caso in cui in una fase i giocatori risultassero essere dispari, uno di questi, scelto casualmente, viene automaticamente “promosso” alla fase successiva.

I tornei misti sono composti da una prima parte organizzata in gironi all'italiana e da una seconda parte a eliminazione diretta. I partecipanti vengono quindi divisi in gruppi per gironi e al termine di questi i vincitori si scontrano a eliminazione diretta.
Il pareggio in questo tipo di tornei è previsto solo nelle gare che appartengono alla fase a gironi all'italiana.

I tornei liberi infine non seguono una struttura specifica, ma è l'organizzatore del torneo stesso a creare le gare nelle quali si scontreranno i partecipanti. Anche in questi tornei è previsto il pareggio e i vincitori vengono eletti in base alla classifica, alimentata con la stessa modalità dei tornei all'italiana.

2) Progettazione concettuale

Per lo sviluppo dello schema ER è stata seguita una metodologia di tipo Top-Down.
Inizialmente si sono identificati i concetti principali quali Torneo, Utente e Gara; e successivamente si sono rifinite le relazioni tra loro.
L'entità Utente è stata successivamente suddivisa in Amministratore, Organizzatore e Giocatore in una generalizzazione totale sovrapposta.
Per quanto riguardo i tornei, è stata identificata una entità principale, Torneo appunto, la quale prevede tre specializzazioni parziali non sovrapposte rispetto alla tipologia: Misto, Eliminazione, Italiana e Libero.

2.1) Entità

1) Utente
È composta dagli attributi:

- *id*: rappresenta univocamente un utente iscritto al portale ed è chiave primaria (vedere la sezione “Scelte progettuali” per maggiori informazioni sull'utilizzo dell'attributo *id*)
- *username*: rappresenta univocamente un utente iscritto al portale
- *password*: attributo utilizzato per proteggere l'accesso dell'utente al portale
- *nome*: nome dell'utente
- *cognome*: cognome dell'utente
- *ban*: attributo utilizzato per riconoscere gli utenti a cui è stato bloccato l'accesso alla quasi totalità delle funzionalità della piattaforma.

L'entità Amministratore è una specializzazione di Utente e riconosce gli utenti che possiedono i permessi di amministrazione della piattaforma.
Per Organizzatore si intende una specializzazione dell'entità Utente che riconosce gli utenti che possiedono i permessi di organizzatore, quindi la possibilità di creare e gestire i tornei.
L'entità Giocatore è sì una specializzazione di Utente, ma ridondante, poiché tutti gli utenti iscritti al portale risultano essere allo stesso tempo anche giocatori.

2) Torneo
Questa entità rappresenta un generico torneo ed è composta da diversi attributi:

- *id*: rappresenta univocamente un torneo ed è chiave primaria (vedere la sezione “Scelte progettuali” per maggiori informazioni sull'utilizzo dell'attributo *id*)
- *nome*: nome del torneo dato dall'organizzatore
- *edizione*: edizione del torneo, insieme al nome rappresenta univocamente un torneo
- *tipo*: tipologia del torneo
- *data inizio*: data di inizio del torneo, corrisponde alla data della prima gara
- *data fine*: data di fine del torneo, corrisponde alla data dell'ultima gara. Eccetto nel torneo misto, viene calcolata automaticamente
- *data di chiusura iscrizione*: data di scadenza delle iscrizioni, dopo la quale nessun giocatore può più iscriversi al torneo
- *quota iscrizione*: quota da versare per iscriversi ad un torneo
- *numero partecipanti*: numero massimo di partecipanti al torneo (minimo 2)

Per l'entità Torneo sono presenti quattro specializzazioni, a seconda della tipologia dello stesso, quindi Libero, Misto, Italiana e Eliminazione.

3) Gara
Entità che rappresenta una singola competizione tra due giocatori.
È composta da:

- *id*: rappresenta univocamente una gara ed è chiave primaria (vedere la sezione “Scelte progettuali” per maggiori informazioni sull'utilizzo dell'attributo *id*)
- *data*: data in cui viene disputata
- *fase*: attributo opzionale utilizzato per riconoscere la fase della quale fa parte la gara all'interno di un torneo a eliminazione diretta
- *girone*: attributo opzionale utilizzato per riconoscere il girone del quale fa parte la gara all'interno di un torneo misto

2.2) Relazioni
1) Iscritto a
Relazione tra Utente e Torneo utilizzata per identificare la partecipazione di un giocatore ad un torneo.
Questa relazione prevede i seguenti attributi:

- *IDutente*: chiave esterna che identifica l'utente partecipe della relazione
- *IDtorneo*: chiave esterna che identifica il torneo partecipe della relazione
- *approvato*: utilizzato per distinguere le iscrizioni approvate o meno dall'organizzatore del torneo stesso
- *sconto/rincaro*: rappresenta la percentuale di sconto (o rincaro) sulla quota di iscrizione al torneo
- *ban*: utilizzato per riconoscere gli iscritti al torneo che sono stati squalificati durante lo svolgimento dello stesso

2) Partecipa a
Relazione tra Utente e Gara che identifica la partecipazione di un giocatore ad una gara di un torneo, con i seguenti attributi:

- *IDutente*: chiave esterna che identifica l'utente partecipe della relazione
- *IDgara*: chiave esterna che identifica la gara partecipe della relazione
- *punteggio*: punteggio ottenuto dal giocatore in seguito al risultato della gara, utilizzato per alimentare la classifica
- *risultato*: risultato ottenuto dal giocatore nella gara

3) Vince_t
Relazione tra Utente e Torneo che identifica i vincitori di un torneo.
Attributi:

- *IDutente*: chiave esterna che identifica l'utente partecipe della relazione
- *IDtorneo*: chiave esterna che identifica il torneo partecipe della relazione

4) Vince_g
Relazione tra Utente e Gara che identifica il vincitore ad una gara.
Attributi:

- *IDutente*: chiave esterna che identifica l'utente partecipe della relazione
- *IDgara*: chiave esterna che identifica la gara partecipe della relazione

5) Riedizione
Relazione tra due Tornei che identifica la riedizione di uno di questi.
Possiede i seguenti attributi:

- *IDriedizione*: chiave esterna che identifica il torneo partecipe alla relazione
- *IDtorneo*: chiave esterna che identifica il torneo (riedizione) partecipe alla relazione

6) Organizza
Relazione tra un Utente e un Torneo che identifica l'organizzatore di un torneo.
Attributi:

- *IDutente*: chiave esterna che identifica l'utente partecipe della relazione
- *IDtorneo*: chiave esterna che identifica il torneo partecipe della relazione

7) Parte di
Relazione tra una Gara e un Torneo che identifica di quale torneo la gara fa parte.
Attributi:

- *IDgara*: chiave esterna che identifica la gara partecipe della relazione
- *IDtorneo*: chiave esterna che identifica il torneo partecipe della relazione

3) Scelte progettuali

Per quanto riguarda le tabelle che rappresentano le entità è stato deciso di utilizzare come chiave primaria un campo numerico *id*, il che permette di identificare univocamente un record, velocizzare l'indicizzazione nel caso di tabelle con molti record (confronto tra interi più veloce rispetto al confronto tra stringhe e richiedente meno byte di memoria) e semplificare l'invio di informazioni tramite GET e POST, poiché non vengono utilizzate chiavi composte.
Ad esempio per l'entità *Utente* si sarebbe potuto utilizzare *username* come chiave primaria, in *Torneo* invece la coppia (*nome*, *edizione*).
Per i motivi sopra citati sono state invece imposte delle clausole *unique* su questi attributi e impostato l'*id* come chiave primaria.
Inoltre le entità che identificano il tipo di utente e il tipo di torneo, seppur assolutamente ridondanti, soni state mantenute in modo da avere la stessa struttura (inteto a 4 byte) delle chiavi esterne in ogni tabella della base di dati.
Il campo *id* non è invece presente nelle tabelle che rappresentano le relazioni tra le entità, poiché in queste è stata usata come chiave primaria la coppia di chiavi esterne che identificano le due entità in relazione.

Le singole gare sono state progettate come una competizione tra due giocatori (o potenzialmente due squadre), quindi negli attributi del torneo non verrà dichiarato il numero di partecipanti alla singola gara. In sostituzione è stato scelto di dare la possibilità, per ogni torneo, di impostare un limite massimo di giocatori, raggiunto il quale l'iscrizione a tale torneo non sarà più possibile.

I risultati delle singole gare possono essere aggiornati solo dall'organizzatore del torneo di cui fanno parte (o dall'amministratore del portale) e non dai partecipanti della gara stessa.

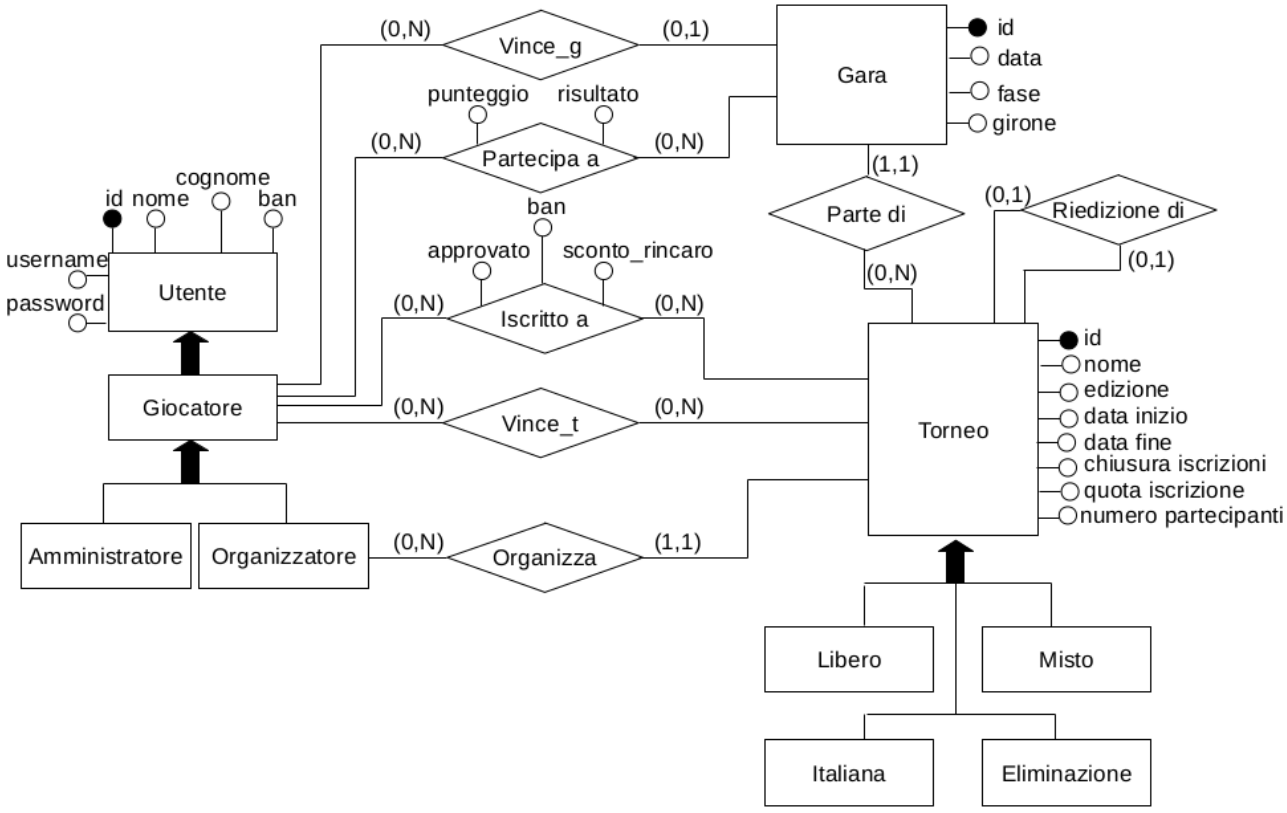
La modifica della quota d'iscrizione per il singolo iscritto ad un torneo è stata espressa attraverso il campo *sconto_rincaro* della relazione “Iscritto a” che rappresenta la percentuale di sconto relativa alla quota base (se è negativa si parla di sconto, se positiva di rincaro).
Tale percentuale può essere modificata solamente prima dell'approvazione dell'iscrizione.

Per ogni gara, nella relazione “Partecipa a”, il punteggio corrisponde ai punti accumulati dal giocatore in base all'esito della gara e vengono utilizzati per alimentare la classifica del torneo. Il risultato invece corrisponde al numero di punti totalizzati dal giocatore durante la gara.
La struttura “a punteggi” della classifica è stata mantenuta anche nel caso di tornei a eliminazione diretta, sebbene per questo tipo di competizione non sia necessaria, così da elaborare la classifica in modo semplice e uguale per ogni tipologia di torneo.

A seconda della tipologia di utente (non registrato, registrato, organizzatore o amministratore) il menù di navigazione del portale presenta voci differenti, che permettono di intraprendere azioni differenti.

Ogni nuovo utente del portale è un semplice giocatore: potrà sì iscriversi ai tornei, ma non potrà crearne. Per poter essere promosso ad organizzatore è necessario l'intervento dell'amministratore che dovrà effettuare tale operazione dalla pagina di gestione degli utenti.

Il portale dispone di una modalità di “notifica” che permette ai giocatori di tenere traccia delle approvazioni delle proprie iscrizioni e dell'eventuale annullamento di un torneo a cui sono iscritti e agli organizzatori della richiesta di iscrizione da parte di un utente ad un proprio torneo.



Schema ER che comprende le Entità, le Relazioni e le scelte progettuali precedentemente descritte

4) Progettazione logica

4.1) Ristrutturazione dello schema ER

Prima di passare allo schema logico della base di dati è necessario ristrutturare lo schema ER eliminando le generalizzazioni e gli attributi composti e multivalore.

Lo schema non presenta attributi composti e multivalore.

Per quanto riguarda l'entità *Utente* sono state rimosse tutte le generalizzazioni introducendo un nuovo attributo intero per definire la natura dell'utente (viene assunto che ogni utente iscritto al portale sia un giocatore).
A questo attributo è stata associata un'entità *Gruppo* con due attributi, *id* e *nome*, così da utilizzare un intero (l'*id* del gruppo) come chiave esterna nell'entità *Utente*.

Le generalizzazioni dell'entità *Torneo* sono state gestite allo stesso modo: è stato introdotto un attributo numerico *tipo* che corrisponde all'attributo *id* dell'entità *Tipo torneo*.

Inoltre sono state aggiunte altre due entità: *Girone* e *Notifica*.
Un *Girone* può essere definito come un torneo all'italiana, che però è parte integrante di un torneo misto.

L'entità *Girone* possiede i seguenti attributi:

- *id*: identifica univocamente un girone (è chiave primaria)
- *numero*: numero del girone in riferimento al torneo misto di cui fa parte
- *data inizio*: data di inizio del girone, corrisponde alla data della sua prima gara
- *data fine*: data di fine del girone, corrisponde alla data della sua ultima gara
- *numero giocatori*: numero di giocatori che partecipano al girone
- *numero gare*: numero di gare che compongono il girone

L'entità *Notifica* rappresenta le notifiche inviate agli utenti, utilizzate per riferire la richiesta di iscrizione ad un torneo, l'approvazione della stessa e l'eventuale annullamento di un torneo.
Possiede i seguenti attributi:

- *id*: identifica univocamente una notifica (è chiave primaria)
- *titolo*: titolo della notifica
- *descrizione*: descrizione della notifica
- *data*: data di invio della notifica
- *letto*: flag che identifica se una notifica è stata letta o meno

Per quanto riguarda l'entità *Girone*, esso è legato a *Torneo* attraverso la relazione *Parte di*.

L'entità *Notifica* è legata invece a *Utente* attraverso la relazione *Riceve* e a *Torneo* attraverso la relazione *Riferito a*.

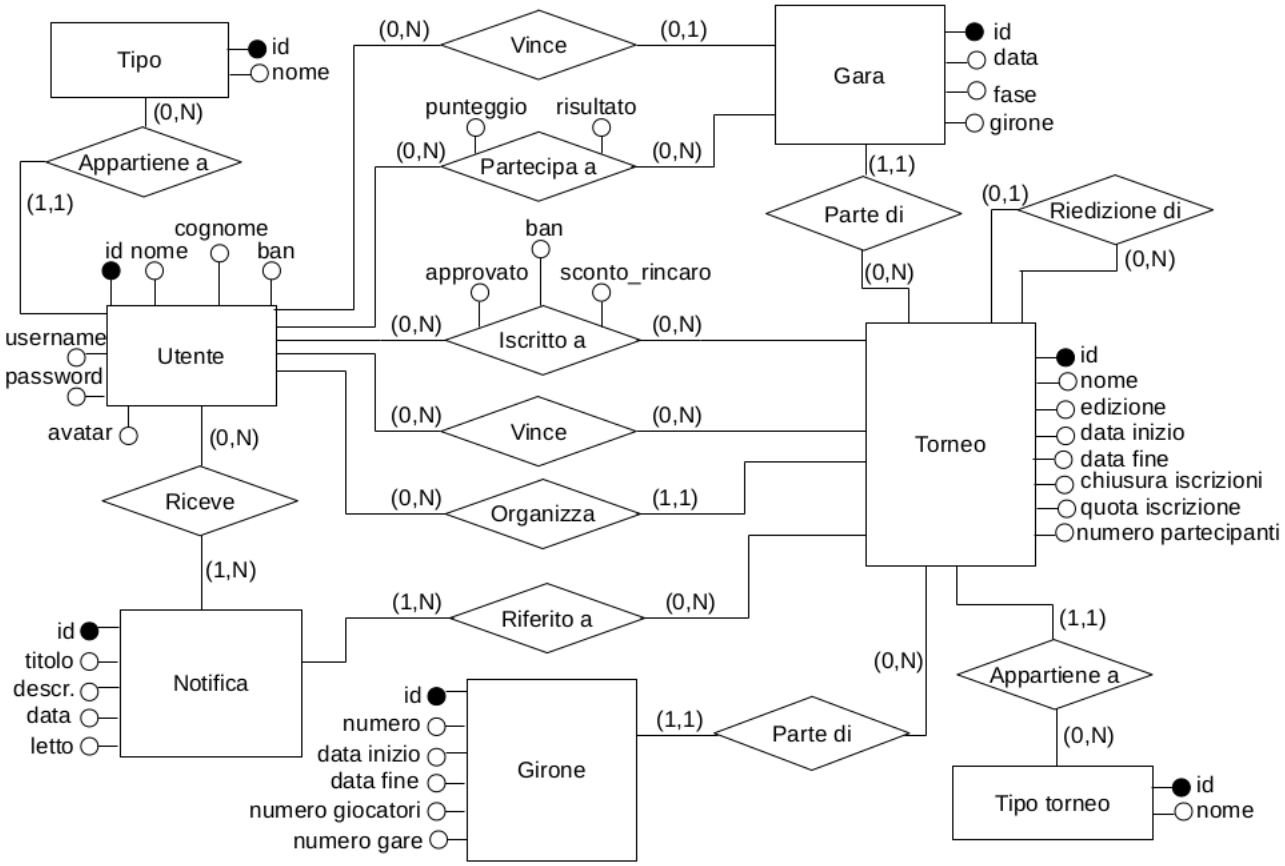


Diagramma ER ristrutturato e completo

4.2) Traduzione in schema logico
Per la traduzione dello schema ER in schema logico è stata utilizzata la seguente legenda:
- chiave primaria
- chiave esterna
- *attributo opzionale
- chiave candidata

Entità
Utente(id, username, password, nome, cognome, IDorg, *avatar, ban)
Gruppi(id, nome)
Torneo(id, nome, edizione, chiusura iscrizioni, data inizio, *data fine, quota iscrizione, numero partecipanti, admin, tipo, *numero gironi, *IDriedizione)
Tipo_torneo(id, nome)
Gara(id, data, *fase, *girone, *vincitore, IDtorneo)
Girone(id, numero, data inizio, *data fine, *numero giocatori, *numero gare, IDtorneo)
Notifica(id, titolo, *descrizione, data, letto, IDutente, IDtorneo)

Relazioni
Vince(IDutente, IDtorneo)
Iscritto_a(IDutente, IDtorneo, approvato, sconto_rincaro, ban)
Partecipa_a(IDutente, IDtorneo, *punteggio, *risultato)

➔ Tutte le relazioni sopra descritte risultano essere in Terza Forma Normale, ottimale per quanto riguarda la ridondanza e l'incoerenza dei dati nel database.

5) Funzioni

Di seguito sono elencate tutte le funzioni presenti nel database, utilizzate per inserire, modificare e ricevere i dati, effettuare controlli e generare automaticamente i tornei.

Funzioni utente

login_corretto(idu INT, p char(32))
Controlla se la password inserita corrisponde a quella relativa all'utente che sta cercando di effettuare l'accesso. Ritorna 1 se corrisponde, 0 altrimenti.

iscrivi (u CHAR(20), p CHAR(32), n CHAR(20), c CHAR(20))
Iscrive un utente al portale con i parametri passati. Di default imposta il nuovo utente come un normale giocatore. Ritorna 1 se la procedura è andata a buon fine.

esiste (u CHAR(20))
Restituisce 1 se nella base di dati esiste un utente con lo username passato come argomento, 0 altrimenti.

get_nome (idu INT)
Restituisce il nome completo (nome e cognome) dell'utente.

get_img (idu INT)
Restituisce il nome con cui è stato salvato sul server l'avatar dell'utente.

get_org(idu INT)
Restituisce l'id del gruppo di cui l'utente fa parte.

mod_pass(idu INT, o CHAR(32), n CHAR(32))
Modifica la password dell'utente controllando prima che la vecchia password corrisponda con quella inserita. Se la procedura va a buon fine restituisce 1, 0 altrimenti.

set_nome(idu INT, n CHAR(20), c CHAR(20))
Modifica il nome dell'utente (nome e cognome) restituendo 1 se la procedura va a buon fine.

set_avatar (idu INT, a CHAR(24))
Modifica l'avatar dell'utente restituendo 1 se la procedura va a buon fine.

make_organizzatore(idu INT)
Promuove un utente a organizzatore restituendo 1 se la procedura va a buon fine.

iscrivi_torneo(idu INT, idt int)
Iscrive un giocatore ad un torneo restituendo 1 se la procedura va a buon fine.

revoca_iscr(idu INT, idt INT)
Revoca l'iscrizione di un utente ad un torneo restituendo 1 se la procedura va a buon fine.

approve_iscr(idu INT, idt INT)
Approva l'iscrizione di un utente ad un torneo restituendo 1 se la procedura va a buon fine.

is_iscr_approved(idu INT, idt INT)
Restituisce 1 se l'iscrizione di un utente ad un torneo è stata approvata, 0 altrimenti.

ban_local(idu INT, idt INT)
Espelle un giocatore da un torneo. In questo modo il giocatore in questione risulterà ultimo nella classifica del torneo e risulterà sconfitto a tavolino in tutte le gare non ancora disputate.

is_ban_local(idu INT, idt INT)
Verifica se un utente è stato espulso da un torneo, restituendo 1 se lo è e 0 altrimenti.

ban(idu INT)
Disabilita un utente alla quasi totalità delle funzioni del portale restituendo 1 se la procedura va a buon fine, 0 altrimenti.

unban(idu INT)
Riabilita l'accesso di un utente al portale restituendo 1 se la procedura va a buon fine, 0 altrimenti.

is_ban(idu INT)
Verifica se l'accesso al portale per un utente è stato disabilitato, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

is_iscritto(idu INT, idt INT)
Verifica l'iscrizione di un utente ad un torneo, restituendo 1 se è iscritto, 0 altrimenti.

agg_quota(idu INT, idt INT, q INT)
Aggiorna la quota di iscrizione ad un torneo per un utente, restituendo 1 se la procedura va a buon fine.

Funzioni torneo

new_torneo(n CHAR(20), e INT, t INT, inizio DATE, chiusura DATE, part INT, idu INT, q DECIMAL(5,2))

Inserisce un nuovo torneo non misto nella base di dati, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

edit_torneo(idt INT, t INT, inizio DATE, chiusura DATE, part INT, idu INT, q DECIMAL(5,2))

Modifica gli attributi di un torneo non misto, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

new_torneo_misto(n CHAR(20), e INT, t INT, inizio DATE, chiusura DATE, part INT, idu INT, q DECIMAL(5,2), g INT)

Inserisce un nuovo torneo misto nella base di dati, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

edit_torneo_misto(idt INT, t INT, inizio DATE, chiusura DATE, part INT, idu INT, q DECIMAL(5,2), g INT)

Modifica gli attributi di un torneo misto, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

new_torneo_ried(idt INT, inizio DATE, chiusura DATE, q DECIMAL(5,2))

Inserisce un nuovo torneo basato su una precedente edizione dello stesso, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

elimina_torneo(idt INT)

Elimina un torneo e con esso anche i suoi gironi, le gare e i rispettivi risultati, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

get_nomet(idt INT)

Restituisce il nome del torneo.

get_edizione(idt INT)

Restituisce l'edizione del torneo.

get_tipo(idt INT)

Restituisce il tipo del torneo.

get_fine(idt INT)

Restituisce la data di fine del torneo.

get_inizio(idt INT)

Restituisce la data d'inizio del torneo.

set_fine(idt INT)

Modifica la data di fine di un torneo (libero), impostandola come la data dell'ultima gara dello stesso, restituendo 1 se la procedura va a buon fine e 0 altrimenti.

registra_risultato(idg INT, g1 INT, pg1 INT, g2 INT, pg2 INT)
Registra il risultato di una gara, restituendo 1 se la procedura va a buon fine e 0 altrimenti.

is_ended(idt INT)
Restituisce 1 se il torneo in questione è finito, 0 altrimenti.

ins_vincitore(idt INT)
Inserisce nella tabella Vince il/i vincitore/i di un torneo, restituendo 1 se la procedura va a buon fine e 0 altrimenti.

is_fase_ended(idt INT, f INT)
Restituisce 1 se la fase del torneo è finita, 0 altrimenti.

is_fase_ita_ended(idt INT)
Restituisce 1 se la parte all'italiana di un torneo misto è finita (se tutti i suoi gironi sono conclusi), 0 altrimenti.

has_gare(idt INT)
Restituisce 1 se il torneo possiede almeno una gara, 0 altrimenti.

has_fine(idt INT)
Restituisce 1 se per un torneo è stata impostata una data di fine, 0 altrimenti.

ins_gara(idt INT, p1 INT, p2 INT, d DATE)
Inserisce una gara relativa ad un torneo, restituendo 1 se la procedura va a buon fine e 0 altrimenti.

reached_iscr_date(idt INT)
Restituisce 1 se è stata raggiunta o superata la data di chiusura delle iscrizioni per un torneo, 0 altrimenti.

reached_start_date(idt INT)
Restituisce 1 se è stata raggiunta o superata la data di inizio per un torneo, 0 altrimenti.

reached_max_iscr (idt INT)
Restituisce 1 se è stato raggiunto il numero massimo di iscritti per un torneo, 0 altrimenti.

has_fasi(idt INT)
Restituisce 1 se il torneo (misto) in questione possiede già delle fasi, 0 altrimenti.

Funzioni di generazione dei tornei

genera_torneo(idt INT)

Funzione che calcola il numero dei partecipanti di un torneo e richiama una delle seguenti specializzazioni a seconda del tipo dello stesso, restituendo il valore di ritorno delle funzioni chiamate.

genera_torneo_italiana (idt INT, num_part INT)

Genera le gare di un torneo all'italiana in base al numero di iscritti, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

genera_torneo_eliminazione (idt INT, num_part INT)

Funzione che richiama *genera_fase()*, restituendo il valore di ritorno della stessa.

genera_fase (idt INT, fase INT, num_part INT)

Genera una fase del torneo a eliminazione diretta, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

genera_fase_succ(idt INT)

Controlla se la fase attuale del torneo a eliminazione diretta si è conclusa, in questo caso chiama *genera_fase()* o *ins_vincitore()*, restituendo il valore di ritorno della stessa.

genera_torneo_misto(idt INT, p INT)

Genera i gironi della fase all'italiana di un torneo misto attraverso *crea_gironi()*, e appoggiandosi ad una tabella temporanea *partecipa_girone* suddivide i partecipanti al torneo nei gironi creati. Infine chiama la funzione *crea_gare_girone()*, restituendo il valore di ritorno di questa (o quello delle funzioni precedenti nel caso di errori).

crea_gironi(idt INT)

Crea i gironi per un torneo misto, restituendo 1 se la procedura va a buon fine e 0 altrimenti.

crea_gare_girone(idg INT, num_part INT, ngir INT)

Genera le gare di un girone di un torneo misto appoggiandosi alla tabella temporanea *partecipa_girone()* creata dalla procedura chiamante, restituendo 1 se la procedura va a buon fine e 0 altrimenti.

genera_ita_misto(idt INT)

Genera la parte a eliminazione diretta di un torneo misto richiamando *genera_fase_misto()*, restituendo il valore di ritorno della stessa.

genera_fase_misto(idt INT, f INT, num_part INT)

Genera una fase della parte eliminazione diretta di un torneo misto, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

genera_fase_misto_succ(idt INT)

Controlla se la fase attuale della parte a eliminazione diretta di un torneo misto si è conclusa, in questo caso chiama *genera_fase_misto()* o *ins_vincitore()*, restituendo il valore di ritorno della stessa.

Funzioni di notifica

create_notification(idu INT, tit CHAR(20), d CHAR(255), idt INT)

Crea una notifica per un determinato utente e riferita ad un torneo, restituendo 1 se la procedura va a buon fine e 0 altrimenti.

read_all_notifications(idu INT)

Segna come lette tutte le notifiche di un utente, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

leggi_notifica (i INT)

Segna come letta una notifica, restituendo 1 se la procedura va a buon fine, 0 altrimenti.

6) Progettazione fisica e UML

Per l'implementazione fisica si è deciso di utilizzare PostgreSQL come DBMS su macchina Linux e PHP per la gestione del back-end della piattaforma.

Per quanto riguarda il front-end si è deciso di utilizzare un tema basato su Bootstrap per la creazione del template (MVP Theme).
Le pagine infatti sono dinamiche e sfruttano le funzioni PHP *ob_start()*, *ob_get_contents()* e *ob_end_clean()* per “iniettare” l'HTML di ogni pagina nel template.

Le pagine del portale sono contenute nella `root` del progetto.

Sotto `deps/` troviamo invece le dipendenze quali il tema usato, jQuery, Font Awesome, Bootstrap Calendar e altre librerie Javascript con i rispettivi CSS.

All'interno di `imgs/` sono contenuti gli avatar degli utenti, compreso quello di default.

All'interno di `inc/` si trovano invece tutti gli altri file personalizzati (PHP, Javascript e CSS), in particolare:

- `template_all.php` e `template_login.php` sono i template del portale
- *functions.php* contiene delle funzioni PHP per semplificare l'inserimento di elementi HTML quali form, campi di input, modal, bottoni e titoli (la loro funzione è semplicemente quella di *wrapper HTML*), funzioni wrapper per le query (*sql(), ssql()* e *multisqlarr()*) e funzioni per controllare i permessi e le notifiche dell'utente collegato, utilizzate per popolare il menù di navigazione
- *config.php* contiene i parametri di configurazione per la connessione al database e della sessione
- *menus.php* contiene i menù di navigazione personalizzati per l'utente collegato

All'interno di `sql/` sono contenuti invece tutti i file SQL relativi alla creazione di sequenze, tabelle, funzioni e alcuni dati per popolare il database.

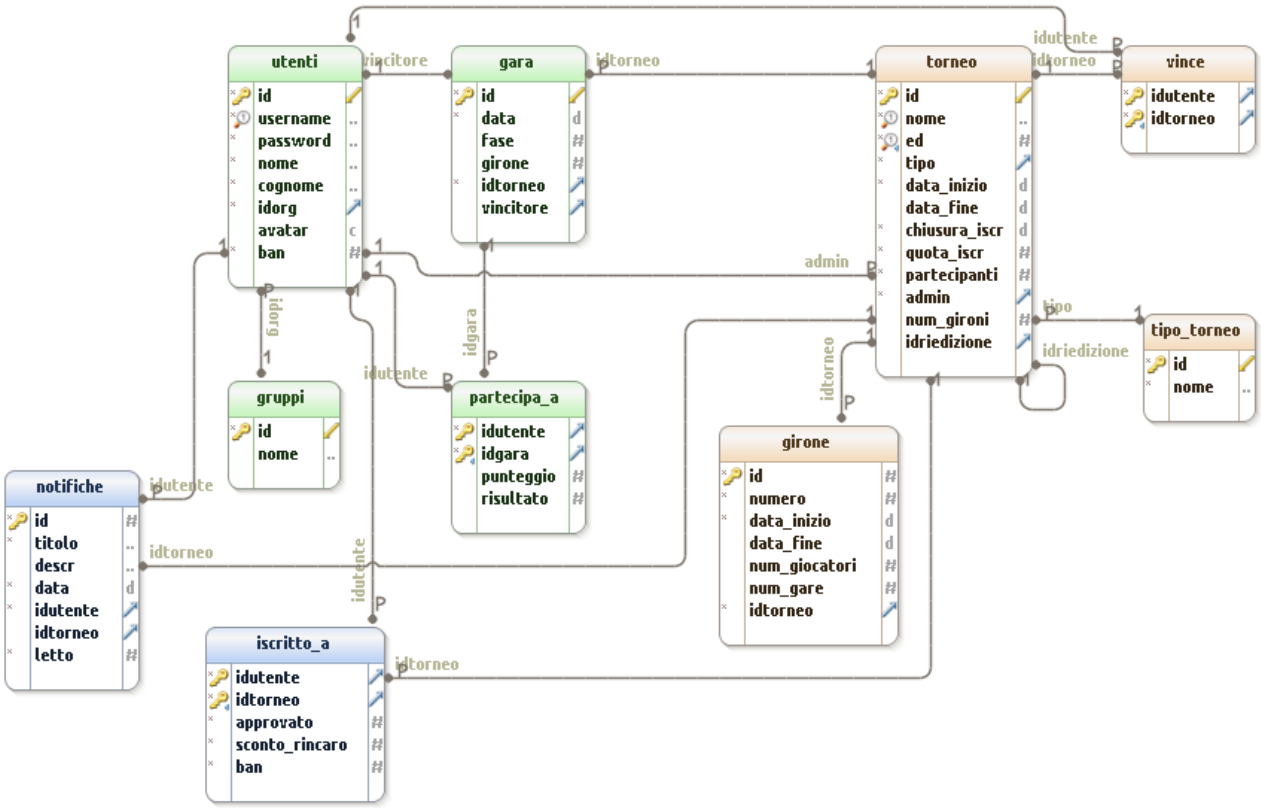

utenti
id
username
password
nome
cognome
idorg
avatar
ban
gara
id
data
fase
girone
idtorneo
vincitore
torneo
id
nome
ed
tipo
data_inizio
data_fine
chiusura_iscr
quota_iscr
partecipanti
admin
num_gironi
idriedizione
vince
idutente
idtorneo
tipo_torneo
id
nome
gruppi
id
nome
partecipa_a
idutente
idgara
punteggio
risultato
girone
id
numero
data_inizio
data_fine
num_giocatori
num_gare
idtorneo
notifiche
id
titolo
descr
data
idutente
idtorneo
letto
iscritto_a
idutente
idtorneo
approvato
sconto_rincaro
ban
vincitore
idtorneo
admin
idorg
idgara
idutente
idtorneo
tipo
idriedizione
idutente
idtorneo
idutente
idtorneo

Grafico UML

Infine, per automatizzare l'inizializzazione del database, è sufficiente recarsi da terminale nella root del progetto ed eseguire da shell il comando `./init_db.sh`: lo script si preoccuperà di chiamare PostgreSQL passandogli come argomenti i file contenuti sotto `sql/`.

```php
<?php

error_reporting(E_ALL ^ (E_NOTICE  | E_DEPRECATED));
ini_set("display_errors", 1);

session_start();

$_SESSION[debug]=0;
$logAllSave=1;

$dbhost="localhost";

$db="aronne";

$user="aronne";
$pass="password";
$port=5432;

$debug=0;

$conn = pg_connect("host=$dbhost port=$port dbname=$db user=$user password=$pass") or

$_SESSION[conn] = $conn;

require "functions.php";
?>
```

config.php, in rosso sono segnate le variabili da modificare

```sh
#!/bin/sh
cd sql
echo "BUILDING DB..."
for i in drop_all.sql sequences.sql tables.sql functions.s
    echo "### Working on $i..."
    psql aronne aronne < $i
done
echo "POPULATING..."
for i in populate.sql populate_Italiano1.sql populate_Libe
    echo "### Working on $i..."
    psql aronne aronne < $i
done
for i in populate_Italiano2.sql populate_Libero2.sql popul
    echo "### Working on $i..."
    psql aronne aronne < $i
done
cd ..
echo "Done!"
exit 0
```

init_db.sh, in rosso sono segnate le parti da modificare

Per poterlo utilizzare sarà necessario modificare lo script in modo da specificare l'utente e il database da modificare (verrà utilizzato lo schema *public*) di tale database.
Inoltre, una volta inizializzato, sarà necessario modificare i parametri di connessione allo stesso contenuti in `inc/config.php`, come nelle immagini sopra.

Nella root del progetto è presente infine un file di testo, `test_users.txt`, contentente una lista di utenti di prova con le relative password utili per il debug.

Il progetto contiene dei dati incompleti a causa dell'impossibilità di aggiornarli ogni giorno, pertanto risultano inconsistenti per quanto riguarda la data di tornei e gare. Per questo è disponibile anche online, con tutti i dati aggiornati giornalmente, al seguente indirizzo: http://dbproj-aroblu94.rhcloud.com.