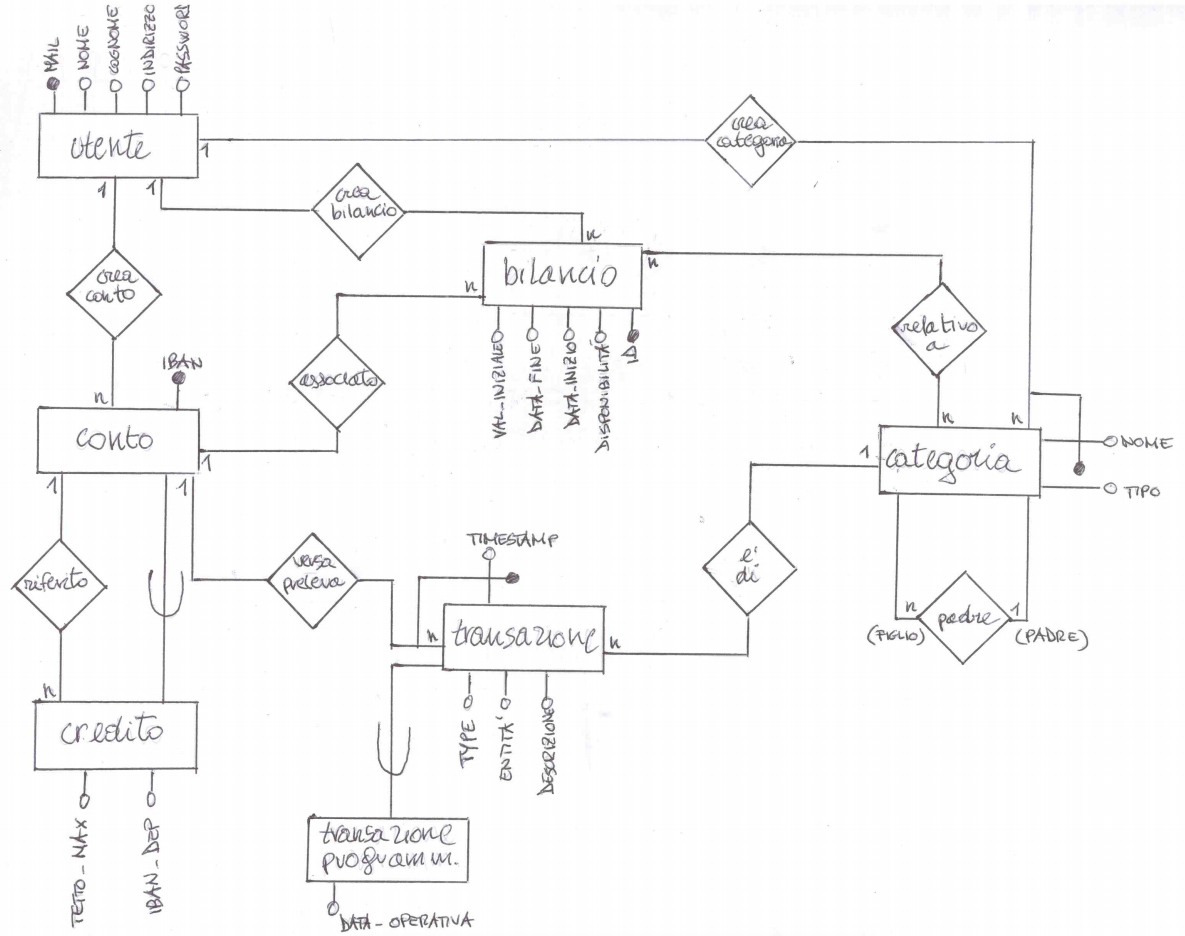
DOCUMENTAZIONE TECNICA

# SCHEMA ENTITY-RELATIONSHIP



**SCHEMA RELAZIONALE**

**Legenda:**

chiave primaria

chiave esterna

chiave primaria e chiave esterna

**UTENTE:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| mail | nome | cognome | indirizzo | password |
| character varying(100) | character varying(30) | character varying(30) | character varying(100) | character varying(20) |

**BILANCIO:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| id | iban | mail | disponibilità | valore\_iniziale | data\_inizio | data\_scadenza |
| character varying(8) | character varying(32) | character varying(100) | numeric(12,2) | numeric(12,2) | date | date |

**CATEGORIA:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| nome | mail | nome\_padre | mail\_padre | tipo |
| character varying(20) | character varying(100) | character varying(20) | character varying(100) | character(1) |

**CATEGORIA\_BILANCIO:**

|  |  |  |
| --- | --- | --- |
| id | mail | nome |
| character varying(8) | character varying(100) | character varying(20) |

**CONTO:**

|  |  |  |  |
| --- | --- | --- | --- |
| iban | mail | tipologia | ammontare |
| character varying(32) | character varying(100) | character varying(8) | Numeric(12,2) |

**CONTO\_CREDITO:**

|  |  |  |
| --- | --- | --- |
| iban | deposito\_riferimento | tetto\_max |
| character varying(32) | character varying(32) | numeric(12,2) |

**TRANSAZIONE:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| iban | data\_transazione | mail | nome | type | descrizione |
| character varying(32) | Timestamp without time zone | character varying(100) | character varying(20) | character(1) | character varying(100) |

**TRANSAZIONE\_PROGRAMMATA:**

|  |  |  |
| --- | --- | --- |
| iban | data\_transazione | data\_operativa |
| character varying(32) | Timestamp without time zone | date |

# DOTAZIONE SOFTWARE

1. **Pacchetto preconfigurato Postgres dbms/php etc:** Necessario per ospitare la base di dati, contiene lo schema relazionale e lo schema pgagent relativo ai jobs.
2. **pgAdminIII:** Strumento di controllo dotato di interfaccia grafica, facilitante per quanto riguarda l’uso di pgAgent
3. **pgAgent:**  Utility importante per l’implementazione dei jobs e per le operazioni che necessitano di trigger periodici.
4. **phpGraphLib:** Libreria indispensabile per l’implementazione di semplici grafici da inserire all’interno di pagine web.

**Linguaggi:**

1. plpgsql per le funzioni realizzate internamente alla base di dati.
2. Php per le funzioni per l'implementazione dnell'interfaccia web e la comuicazione con la base di dati.
3. Html per l'impaginazione delle contenuti web.

**CONSEGNA**

**Conto**

Un conto rappresenta un deposito economico caratterizzato da un proprio bilancio in costante aggiornamento rispetto alle entrate e uscite che interessano il conto. Un utente può creare liberamente il numero di conti che preferisce. In fase di creazione, occorre specificare un ammontare iniziale e la tipologia del conto. I conti si dividono in due tipologie: conti di deposito e conti di credito. I conti di deposito rappresentano conti in cui le entrate e le spese hanno effetto immediato sul bilancio del conto. Ne sono esempi i conti creati per rappresentare la disponibilità in contanti e i conti bancari . I conti di credito sono conti in cui le uscite sono contabilizzate su un conto di deposito a scadenze fissate nel tempo. Un esempio di conto di credito è un conto creato per rappresentare una carta di credito, il cui ammontare in termini di spesa viene scalato da un conto corrente all’inizio di ogni mese. Pertanto , ogni conto di credito deve necessariamente essere associato a un conto di deposito , dal quale vengono dedotte le spese al momento della scadenza dei crediti. Le entrate sui conti di credito rappresentano il tetto massimo di credito e vengono rinnovate automaticamente al saldo dei crediti.

**Spese e entrate**

Spese e entrate costituiscono una transazione economica associata necessariamente a un conto. Sono caratterizzate da una data , una categoria di spesa/entrata , una descrizione e l’entità economica della transazione. Ogni utente può inoltre modificare le categorie di spesa/entrata previste aggiungendone di nuove o aggiungendo a una categoria esistente una nuova sotto-categoria.

**Bilancio**

Un bilancio rappresenta una statistica relativa a uno o più conti o bilanci. Attraverso i rapporti deve essere possibile: avere un saldo per periodo di ogni conto/bilancio. Confrontare, in un periodo di tempo scelto dall’utente , diversi bilanci. Avere una rappresentazione della percentuale di spesa in ogni categoria in un periodo arbitrariamente scelto dall’utente. Avere statistiche generali , che considerino cioè tutti i conti, circa ad esempio la quantità media di spesa, le categorie in cui si registrano le maggiori spese , le principali categorie di entrata. In generale, ogni rapporto va inteso come una particolare vista della base di dati che consenta di estrarre i dati necessari alla composizione del rapporto desiderato e effettui le necessarie aggregazioni.

**Utente**

Ogni utente ha un proprio profilo personale e accesso esclusivo ai propri conti , bilanci e rapporti. Un utente è caratterizzato dai propri dati personali e da un profilo che determina alcune opzioni di default dell’utente, come ad esempio la valuta da utilizzare.

**FUZIONI REALIZZATE**

**check\_bilancio\_user\_iban:**

Funzione richiamata da un trigger su inserimento nella tabella *bilancio*, controlla che il conto a cui si vuole riferire il nuovo bilancio esista e appartenga effettivamente al richiedente.

**check\_categoria:**

Funzione richiamata da un trigger su inserimento nella tabella *categoria*, controlla l’integrità della base di dati nel momento in cui un utente voglia creare una nuova categoria. Controlla che la categoria padre sia appartenente al richiedente e che la nuova categoria sia dello stesso tipo(entrata/spesa) della categoria padre.

**check\_bilancio:**

Funzione richiamata da un trigger su eliminazione di una tupla dalla tabella *bilancio*. Se il credito residuo su bilancio è maggiore di 0 la funzione riaccredita sul conto di riferimento l’intero importo.

**delete\_conti\_credito:**

Funzione richiamata da un trigger su eliminazione di una tupla dalla tabella *conto\_credito*. Se esistono bilanci associati al conto di credito questi vengono eliminati, infine viene riaccreditato l’intero ammontare del conto di credito sul conto di deposito di riferimento.

**effettua\_transazione:**

Funzione richiamata da un trigger su inserimento e aggiornamento di una tupla nella tabella *transazione*. A seconda del tipo della transazione(entrata/spesa) la funzione accredita/scala i l’importo della transazione. Se esiste uno o più bilanci associati alla categoria di spesa/entrata l’ammontare viene scalato dal/i bilanci.

**effettua\_transazione\_programmata:**

Funzione richiamata da uno scheduler impostato in pgAgent, la funzione viene richiamata ogni giorno, se la data corrente coincide con la *data\_operativa* della transazione allora viene aggiornata la tupla corrispondente nella tabella transazione(richiamando così la funzione *effettua\_transazione*). Nel caso in cui la data corrente non coincida con quella operativa della transazione, nessuna operazione verrà effettuata.

**fill\_bilancio:**

Funzione richiamata da un trigger su inserimento nella tabella *bilancio*. La funzione accredita l’ammontare massimo ammissibile sul bilancio, scalando l’importo dal conto di riferimento.

**fill\_cred\_from\_dep:**

Funzione richiamata da un trigger su inserimento nella tabella *conto\_credito*. Se la liquidità del conto di deposito di riferimento lo permette, la funzione accredita un ammontare pari al tetto massimo sul conto di credito. Nel caso in cui la liquidità non sia sufficiente, verrà sollevata un’eccezione dalla base di dati.

**insert\_root\_categories:**

Funzione richiamata da un trigger su inserimento nella tabella *utente*. La funzione crea le categorie di base: spesa e entrata.

**refill\_bilancio:**

Funzione richiamata da uno scheduler impostato in pgAgent, la funzione viene richiamata ogni giorno, se la data corrente coincide con la data di scadenza del bilancio la liquidità di quest’ultimo viene riportata all’equivalente del tetto massimo. La funzione inoltre aggiorna la data di scadenza del bilancio.

**update\_conti\_credito:**

Funzione richiamata da uno scheduler impostato in pgAgent, la funzione viene richiamata l’ultimo giorno di ogni mese. L’ammontare di ogni conto di credito presente sulla base di dati viene riportato, se possibile, all’ammontare iniziale massimo. Le somme di denaro corrispondenti vengono ovviamente scalati dai relativi conti di deposito associati.