IBERU

1. ABSTRACT

Iberu è un'applicazione progettata per agevolare il monitoraggio della dieta, dell'attività fisica e del sonno dei suoi clienti al fine di promuovere uno stile di vita sano e bilanciato.
L'applicazione permette agli utenti di tenere sotto controllo l'apporto calorico fornendo accesso ad un vasto database alimentare per la registrazione dei pasti ed il tracciamento dei macronutrienti principali. Iberu inoltre consente agli utenti di aggiungere le proprie ricette preferite, offrendo resoconti alimentari accurati e personalizzati.
La funzionalità di monitoraggio dell'attività fisica include un'ampia gamma di esercizi per tutte le tipologie di allenamento. Inoltre, l'applicazione offre la possibilità di consultare lo storico degli allenamenti al fine di migliorare la qualità ed efficacia nel tempo.
Iberu facilita anche il monitoraggio del sonno consentendo agli utenti di registrare sia la durata che la qualità del riposo notturno. L'applicazione fornisce inoltre informazioni dettagliate sulle diverse fasi del sonno, consentendo agli utenti di identificare eventuali problemi.

2. ANALISI DEI REQUISITI

2.1 Descrizione testuale

Dalle analisi di mercato effettuate dal team di sviluppatori di Iberu l'applicazione potrebbe raggiungere un massimo di 20000 utenti ciascuno caratterizzato da nome, cognome, indirizzo di posta elettronica e dati personali quali altezza, genere ed età. Ogni utente può configurare il volume di attività fisica ed il peso ideale che intende raggiungere, inoltre l'applicazione fornisce la possibilità di impostare la dieta attuale o di individuare un regime alimentare desiderato.
Ogni regola alimentare è caratterizzata da nome, descrizione e fattori moltiplicativi specifici volti a regolare l'assunzione giornaliera di calorie, carboidrati, grassi, proteine e d'acqua.
I clienti possono inserire all'interno della piattaforma tutte le informazioni relative al loro lifestyle, tra cui gli alimenti assunti, le rispettive quantità e l'orario dei pasti così da ottenere una misurazione accurata di calorie e macronutrienti e migliorare il tracciamento energetico. Di ogni alimento il database conserverà: nome, marca, calorie e nutrienti.
I pasti possono essere personalizzati, ad esempio ricette più o meno complesse oppure cibi già presenti all'interno del database. Ogni ricetta è caratterizzata dal nome, dalla durata, dal procedimento di preparazione, dalla durata di cottura,dal peso finale ,dalla difficoltà e dagli alimenti di cui è composta. L'applicazione inoltre offre la funzionalità di tracciamento della quantità d'acqua assunta nell'arco della giornata memorizzando sia l'ora che la quantità.
Iberu fornisce all'utente un quadro completo dei traguardi raggiunti nel proprio percorso fitness mediante la registrazione e il monitoraggio delle misurazioni di peso, massa magra e massa grassa nel corso del tempo.
Nell'ambito del monitoraggio dell'attività fisica Iberu identifica due macro tipologie di allenamento: Palestra e Cardio, per ciascuna potranno essere memorizzate le calorie bruciate, la media del battito cardiaco, la durata e l'intensità dello sforzo fisico. Se l'attività fisica viene svolta in palestra è possibile salvare il gruppo muscolare colpito, mentre per l'attività fisica cardiovascolare è possibile registrare la durata dell'allenamento anaerobico, la durata dell'allenamento aerobico, la distanza percorsa ed il dislivello. L'applicazione fornisce un database di movimenti ed esercizi, suddivisi in cardiovascolari e pesistica, ciascuno dei quali include nome, tempo di recupero e descrizione dell'esercizio. Gli esercizi di pesistica indicano il numero consigliato di ripetizioni, i set e le calorie bruciate per ripetizione, mentre gli esercizi cardiovascolari mostrano la distanza minima consigliata, la tipologia e le calorie consumate per ogni 100 metri.
Nell'ambito del monitoraggio del sonno, Iberu fornisce una panoramica dettagliata tenendo traccia dell'orario in cui l'utente si è addormentato, l'ora in cui si è alzato e la qualità complessiva del riposo notturno. Inoltre, l'applicazione offre un'analisi approfondita degli orari e delle durate dei risvegli, nonché delle diverse fasi del sonno, mostrandone la durata,la tipologia, una descrizione e il battito cardiaco medio corrispondenti a ciascuna fase.

2.2 Glossario dei termini

Termine	Descrizione	Sinonimi	Collegamenti
Utente	Persona che utilizza l'applicazione Iberu	cliente	Dieta, Pasto, Acqua, Peso, Allenamento, Sonno
Dieta	Un'alimentazione corretta, sana ed equilibrata volta a soddisfare le esigenze fisiologiche dell'organismo.	regime alimentare, regola alimentare	Utente
Pasto	Momento specifico della giornata in cui l'utente si nutre attraverso una o più vivande.		Utente, Alimento

Alimento	Sostanza che sopperisce al dispendio di energie e fornisce materiali indispensabili alla reintegrazione.	cibo	Pasto, Alimento
Ricetta	Alimento o bevanda ottenuta da molteplici ingredienti attraverso specifici procedimenti (es. cottura, impasto ecc…).	pasto personalizzato	Alimento
Acqua	Assunzione giornaliera di acqua da parte dell'utente.		Utente
Peso	Misurazione della massa dell'utente.	misurazione	Utente
Allenamento	Pratica dell'esercizio fisico al fine di raggiungere un adattamento fisiologico.	esercizio fisico, attività fisica	Utente, Esercizio
Palestra	Tipologia specifica di allenamento svoltasi con l'utilizzo di pesi e/o macchinari.	esercizio fisico, attività fisica	Esercizio
Cardio	Tipologia specifica di allenamento che coinvolge il sistema cardiovascolare, ovvero il cuore, i vasi sanguigni e i polmoni	esercizio fisico, attività fisica	Esercizio
Esercizio	Movimento specifico che viene eseguito con l'obiettivo di migliorare la forza, la resistenza, la flessibilità o l'equilibrio.	movimento	Allenamento
Pesistica	Tipologia di esercizio che prevede l'utilizzo di pesi per massimizzare lo sforzo muscolare.	movimento	Allenamento
Cardiovascolare	Tipologia di esercizio volto a migliorare l'efficienza del sistema cardiovascolare.	movimento	Allenamento
Sonno	Attività di riposo notturno per il ripristino dell'efficienza fisica o psichica	riposo notturno	Utente, Fase, Risveglio
Fase	Fasi di suddivisione del riposo che si alternano costantemente.		Sonno
Risveglio	Risvegli coscienti o meno che avvengono durante il riposo notturno dell'utente		Sonno

2.3 Operazioni sul database

Operazione	L/S	Frequenza (giorno)
Ricerca calorie e macronutrienti principali consumati durante un pasto	L	180000
Registrazione alimento in un pasto	S	240000
Ricerca calorie bruciate durante un allenamento	L	4000
Registrazione di un nuovo allenamento	S	5500
Registrazione di un nuovo alimento nel database	S	50
Registrazione del riposo notturno	S	5500
Registrazione nuova misurazione del peso	S	8000
Visualizzazione qualità e durata del riposo notturno	L	4000
Consultazione calorie e macronutrienti relativi alla propria dieta	L	72000
Visualizzazione delle ricette	L	50000

3. PROGETTAZIONE CONCETTUALE

3.1 Lista delle entità

Legenda:
Gli identificatori delle entità sono gli attributi sottolineati.
Tutti i campi sono di default NOT NULL se non specificato altrimenti .

UTENTE

-Indirizzo posta elettronica →varchar(50),PK
-Nome →varchar(50)
-Cognome →varchar(50)
-Età →tinyint unsigned && <=200
-Genere →bool
-Altezza(in cm) →decimal (5,2) unsigned && <=300
-Volume_attività(da 0-5) →tinyint unsigned &&<= 5 anche NULL

DIETA

-Nome_dieta →varchar(30),PK
-Descrizione_dieta →varchar(500) , anche NULL
-Moltiplicatore_calorie →decimal (2,1) unsigned
-Moltiplicatore_proteine →decimal (2,1) unsigned
-Moltiplicatore_carboidrati →decimal (2,1) unsigned
-Moltiplicatore_grassi →decimal (2,1) unsigned
-Moltiplicatore_acqua →decimal (2,1) unsigned

PASTO

Identificatore esterno: attributo "*Indirizzo posta elettronica*" dell'entità **Utente.**

-Data_ora_pasto →timestamp, PK
-Calorie_pasto (in g) →smallint unsigned
-Proteine_pasto (in g) →smallint unsigned
-Carboidrati_pasto (in g) →smallint unsigned
-Grassi_pasto (in g) →smallint unsigned

ALIMENTO

-Id_alimento →int unsigned, PK
-Nome_alimento →varchar(50)
-Marca_alimento → varchar(50) anche NULL
-Calorie_alimento (in g) →varchar(50)
-Proteine_alimento (in g) →smallint unsigned
-Carboidrati (in g) →smallint unsigned
-Fibre (in g) →smallint unsigned
-Zuccheri (in g) →smallint unsigned
-Grassi (in g) →smallint unsigned
-Colesterolo (in g) →smallint unsigned
-Saturi (in g) →smallint unsigned
-Monoinsaturi (in g) →smallint unsigned
-Polinsaturi (in g) →smallint unsigned
-Sale_alimento (in g) →smallint unsigned

RICETTA

-Nome_ricetta →varchar(50),PK
-Peso_finale(in kg) →decimal (6,2) unsigned
-Durata_preparazione(in min) →time,anche NULL && >=00:00:00 && <= 23:59:59
-Durata_cottura →time , anche NULL && >=00:00:00 && <= 23:59:59
-Procedimento →varchar(500),anche NULL
-Difficoltà →tinyint unsigned,>=1 <5

ACQUA

Identificatore esterno: attributo "*Indirizzo posta elettronica*" dell'entità **Utente.**

-Id_acqua →timestamp,PK
-mL →smallint unsigned

PESO

-Data_misurazione →date ,PK
-Peso(in kg) →decimal (6,2) unsigned

-Percentuale_massa_grassa →decimal (4,2) unsigned, anche NULL
-Percentuale_massa_magra →decimal (4,2) unsigned, anche NULL

ALLENAMENTO

Identificatore esterno: attributo "*Indirizzo posta elettronica*" dell'entità **Utente.**

-Data_allenamento →date, PK
-Calorie_consumate →smallint unsigned
-Battito_allenamento →tinyint unsigned, anche NULL, <200
-Durata_allenamento(in min) →decimal (5,2) unsigned, anche NULL
-Intensità →tinyint unsigned , <10 &&>0 , anche NULL

ALLENAMENTO PALESTRA

Eredita l'identificatore "*data_allenamento*" dall'entità **Allenamento.**

-Gruppo_muscolare→varchar(50)
-Tipologia_allenamento →varchar(50)

ALLENAMENTO CARDIO

Eredita l'identificatore "*data_allenamento*" dall'entità **Allenamento.**

-Durata_anaerobico →decimal (3,2) unsigned
-Durata_aerobico →decimal (5,2) unsigned
-Distanza →decimal (6,2) unsigned
-Dislivello →decimal (6,2) unsigned, anche NULL

ESERCIZIO

-Nome_esercizio →varchar(50),PK
-Recupero →tinyint unsigned, anche NULL
-Descrizione_esercizio →varchar(50),anche NULL

ESERCIZIO PESISTICA

Eredita l'identificatore "*nome_esercizio*" dall'entità **Esercizio.**

-Ripetizioni_consigliate →tinyint unsigned
-Set_consigliati →tinyint unsigned
-Calorie_ripetizione →decimal (4,2) unsigned

ESERCIZIO CARDIOVASCOLARE

Eredita l'identificatore "*nome_esercizio*" dall'entità **Esercizio.**

-Distanza_minima →decimal (5,2) unsigned
-Tipologia_esercizio_cardio →varchar(50)
-Calorie_100m →tinyint unsigned

SONNO

Identificatore esterno: attributo "*Indirizzo posta elettronica*" dell'entità **Utente.**

-Data_ora_riposo →timestamp,PK
-Durata_riposo (in min) →decimal (5,2) unsigned
-Qualità→tinyint unsigned, <=5

RISVEGLIO

Identificatore esterno: attributo "*data_ora_riposo*" dell'entità **Sonno.**

-Orario_risveglio →time,PK
-Durata_risveglio →decimal (4,2) unsigned

FASE

-Tipologia_fase →varchar(50),PK
-Descrizione_fase →varchar(250)

GENERALIZZAZIONI :

1. La generalizzazione tra l'entità **Allenamento** e le entità **Palestra** e **Cardio** è totale ed esclusiva.
2. La generalizzazione tra l'entità **Esercizio** e le entità **Pesistica** e **Cardiovascolare** è totale ed esclusiva.

ATTRIBUTI COMPOSTI :

1. Nell'Entità **Ricetta** l'attributo composto "*preparazione*" contiene:
   - "*procedimento*","*durata_cottura*","*durata_preparazione*"
   - "*peso_finale*","*difficoltà*"
2. Nella Relazione **Segue** l'attributo composto "*nutrienti_dieta*" presenta i seguenti attributi:
   - "*calorie_dieta*", "*carboidrati_dieta*", "*proteine_dieta*", "*grassi_dieta*", "*acqua_dieta*'
3. Nell'Entità **Pasto** l'attributo composto "*nutrienti_pasto*" presenta i seguenti attributi :
   - *"moltiplicatore_acqua", "moltiplicatore_grassi", "moltiplicatore_calorie", "moltiplicatore_proteine", "moltiplicatore_carboidrati"*

4. Nell'entità **Alimento** l'attributo composto "*nutrienti_alimento*" è a sua volta suddiviso in due attributi composti *"grassi_alimento"* e "*carboidrati_alimento*":
    - Attributi *"nutrienti_alimento": calorie_alimento", "proteine_alimento", "sale_alimento"*
    - Attributi "*grassi_alimento*": *"colesterolo", "saturi", "polinsaturi", "monoinsaturi", "grassi"*
    - Attributi "*carboidrati_alimento*": *"zuccheri", "fibre", "carboidrati"*

VINCOLI DI INTEGRITA':

1. La somma degli attributi "*percentuale_massa_grassa"* e *"percentuale_massa_magra"* non può superare 100.
2. Le date di pasti, allenamenti, misurazione del peso non possono superare la data odierna.
3. Un utente può salvare al massimo una misurazione del peso al giorno
4. Le fasi del sonno per 1 data sommate assieme non possono avere una durata maggiore della durata del sonno
5. Risulta possibile registrare al massimo una sessione di allenamento al giorno
6. Nelle Entità **Alimento**, **Pasto**, **Dieta** e nella Relazione **Segue** i valori relativi alle calorie totali non possono essere inferiori a (carboidrati + proteine) * 4 + grassi * 9

3.2 Lista delle relazioni e cardinalità

Utente - Dieta: **Segue**
- Un utente può seguire al massimo un solo regime alimentare -> (0, 1)
- Una dieta può essere seguita da vari utenti (o da nessuno) -> (0, n)

La relazione presenta l'attributo composto *"nutrienti_dieta*" e gli attributi "*peso_obiettivo*" e *"tipologia_dieta"*.
(Un utente può seguire solo una dieta in caso di cambi si eliminerà la tupla con la relazione precedente. Lo storico delle diete seguite non è verificabile.)

Dieta-Alimento**: Appartenente**
- Una dieta può contenere molteplici alimenti -> (1, n)
- Un cibo può appartenere a varie diete (o a nessuna) -> (0, n)

Ricetta-Alimento: **Contiene**
- Una ricetta contiene almeno un alimento -> (1, n)
- Un alimento può essere contenuto in più ricette (come in nessuna) -> (0, n)

La relazione possiede l'attributo *"quantità_ingrediente"*.

Alimento-Pasto**: Costituito**
- Un alimento può essere presente in molteplici pasti (o nessuno) -> (0, n)
- Un pasto deve avere almeno un alimento -> (1, n)

La relazione presenta l'attributo *"quantità_alimento"*.

Utente-Pasto**: Mangia**
- Un utente può mangiare molteplici pasti (o nessuno) -> (0, n)
- Uno specifico pasto può essere mangiato da un singolo utente -> (1, 1)

Utente-Acqua: **Assume**
- Un utente può memorizzare quanti mL di acqua ha bevuto varie volte (o non farlo mai) -> (0, n)
- Le quantità di acqua bevuta appartengono ad un ed un solo utente -> (1, 1)

La relazione presenta l'attributo "data_ora_acqua"

Pasto-Acqua: **Bevuto**
- Un pasto può salvare la quantità di mL di acqua bevuta (o non farlo) -> (0, 1)
- Le quantità di acqua bevuta appartengono ad un ed un solo utente -> (1, 1)

Utente-Allenamento: **Svolge**
- Un utente può svolgere molteplici allenamenti al giorno (come non allenarsi mai) -> (0, n)
- Un allenamento può essere svolto da una sola persona -> (1, 1)

Utente-Sonno: **Dorme**
- Un utente può registrare diverse sessioni di sonno (o nessuna) -> (0, n)
- Una sessione di sonno si riferisce a un unico utente -> (1, 1)

Sonno-Fase: **Suddiviso**
- Una sessione di sonno può essere suddivisa in tante fasi (almeno uno) -> (1, n)
- Una fase può far parte di varie sessioni di riposo (come di nessuna) -> (0, n)

La relazione possiede gli attributi *"durata_fase*" e *"battito_fase"*.

Sonno-Risveglio: **Interrotto**
- Il sonno può essere interrotto da risvegli -> (0, n)
- Un risveglio appartiene ad una specifica sessione di riposo-> (1, 1)

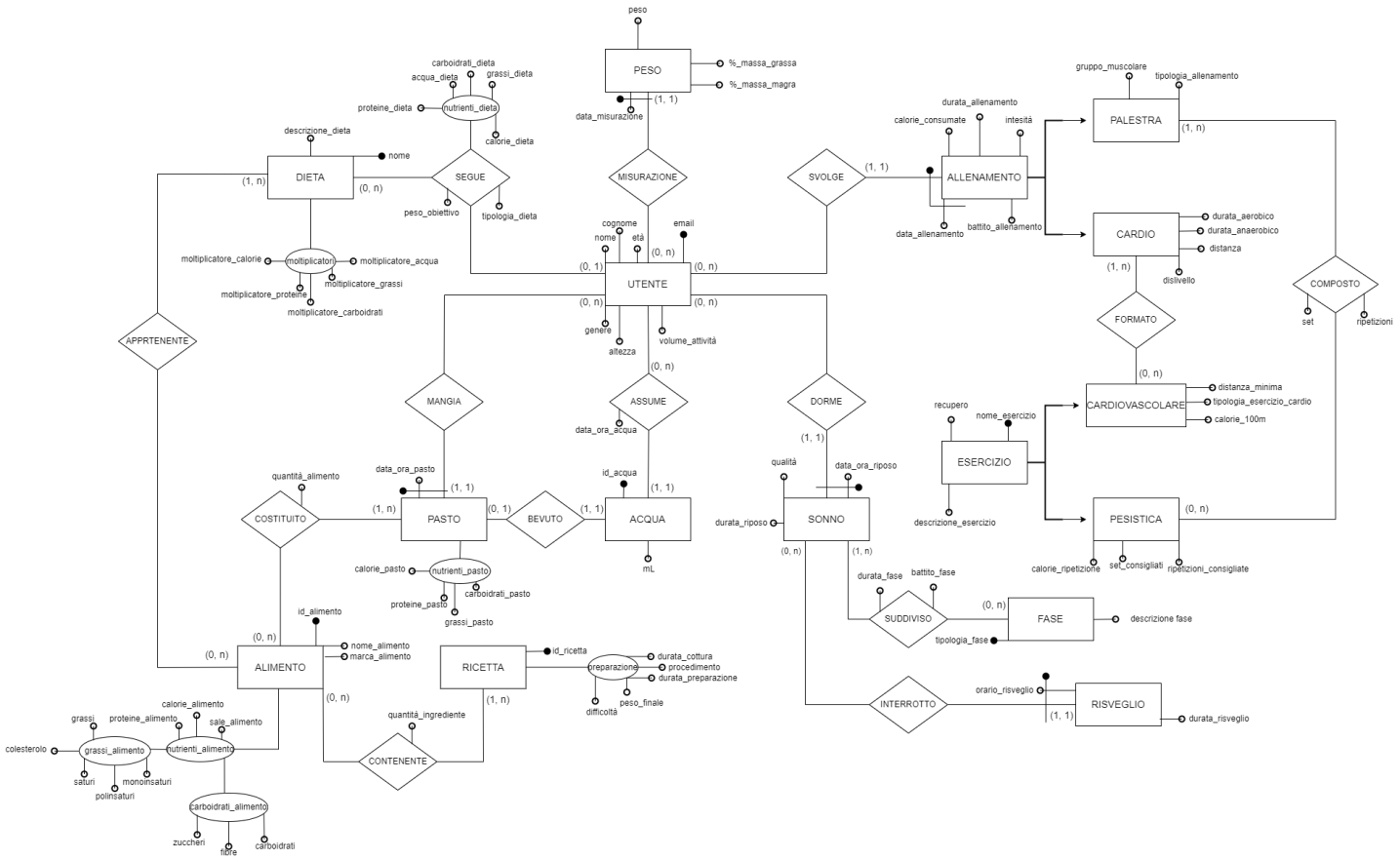
Palestra-Pesistica: **Composto**
- L'allenamento in palestra può essere costituito da diversi esercizi di pesistica (almeno uno) ->(1,n)
- Un esercizio di pesistica può essere stato eseguito svariate volte (come mai) -> (0, n)

La relazione possiede gli attributi *“set”* e *“ripetizioni”*.

Cardio-Cardiovascolare: **Formato**

- L’allenamento di Cardio si compone di diversi esercizi cardiovascolari (almeno uno) -> (1, n)
- Un esercizio cardiovascolare può essere stato eseguito svariate volte (come mai) -> (0, n)

3.3 Schema E-R



4. PROGETTAZIONE LOGICA
Ristrutturazione dello schema Entità-Relazione

4.1 Analisi di ridondanze

La prima ridondanza identificata nello schema E-R riguarda l'attributo composto "*nutrienti_pasto*", che rappresenta l'accorpamento di macronutrienti e calorie assunti.

Considerando che gli utenti di Iberu registrano in media tre pasti al giorno, e ogni pasto è composto da circa quattro alimenti, possiamo ottenere i macronutrienti e le calorie acquisiti sommando i valori degli attributi corrispondenti agli alimenti registrati.

Quotidianamente gli accessi in memoria che coinvolgono questi specifici attributi risultano essere effettuati da:

- Operazione 1: Ricerca calorie e macronutrienti principali consumati durante un pasto. In media un utente controlla calorie e macronutrienti che ha assunto all’incirca tre volte per pasto, ovvero ogni volta che deve inserire un nuovo alimento eccetto per il primo.
  [Formula: utenti_totali * pasti_al_giorno * (alimenti_per_pasto - 1)]
- Operazione 2: Registrazione alimento in un pasto. Un utente consuma circa quattro alimenti ogni pasto e mangia tre volte al giorno.
  [Formula: utenti_totali * pasti_al_giorno * alimenti_per_pasto]

Senza ridondanza			*"nutrienti_pasto"*	
=>	Entità/Relazione	Volume	Operazione 1	Operazione 2
Accessi	**Costituito**	24’000’000	(1L x 4)**	1S
	Alimento	120’000	(1L x 4)**	0

OverHead Operazione singola	8	2
Frequenza giornaliera operazione	180’000	240’000
OverHead giornaliero (S vale doppio)	1’440’000	480’000

Con ridondanza			"*nutrienti_pasto*"	
=>	Entità/Relazione	Volume	Operazione 1	Operazione 2
Accessi	**Pasto**	8’000’000	1L	1S
	Costituito	24’000’000	0	1S
	Alimento	120’000	0	1L
OverHead Operazione singola			1	5
Frequenza giornaliera operazione			180’000	240’000
OverHead giornaliero (S vale doppio)			180’000	1’200’000

Conclusioni	"*nutrienti_pasto*"	
	Calcolo	OverHead
Senza ridondanza	1’440’000 + 480’000	1’920’000
Con ridondanza	180’000 + 1’200’000	1’380’000
In seguito all’analisi svolta si è deciso di mantenere la ridondanza relativa all’attributo “*nutrienti_pasto”* poiché comporta un overhead minore sebbene porti ad un’ occupazione di memoria superiore di 480 KB al giorno		

La seconda ridondanza identificata nello schema riguarda l'attributo composto "*nutrienti_dieta*", che rappresenta l'accorpamento delle quantità di macronutrienti e calorie che uno specifico utente può consumare giornalmente. Queste quantità vengono calcolate utilizzando il BMI (Indice di Massa Corporea), ottenuto dividendo il peso per il quadrato dell'altezza. Il calcolo viene quindi moltiplicato per il moltiplicatore corrispondente presente nell'entità **Dieta**.
Per rendere il calcolo ancora più preciso, viene sottratto il valore dell'attributo "*tipologia_dieta*" presente nella relazione **Segue**. Questo attributo rappresenta una percentuale che indica la velocità con cui si desidera perdere o guadagnare peso.
Dalle analisi di mercato condotte dal team di sviluppo di Iberu, emerge che circa il 60% degli utenti iscritti seguirà una dieta, mentre il restante preferirà utilizzare la funzionalità di monitoraggio di calorie e macronutrienti liberamente (39.8%) o non utilizzarla proprio (<0.2%).
Quotidianamente gli accessi in memoria che coinvolgono questi specifici attributi risultano essere effettuati da:

- Operazione 7: Registrazione nuova misurazione del peso. Ogni qual volta che un utente aggiunge una nuova misurazione sarà necessario ricalcolare gli apporti giornalieri di calorie e macronutrienti relativi al BMI aggiornato. [Formula: “*nuovo_peso*” / (“altezza”)^2 * “*moltiplicatore_n*” * (100 - “*tipologia_dieta*”)]
- Operazione 9: Consultazione calorie e macronutrienti relativi alla propria dieta. In media un utente controlla la quantità di calorie e macronutrienti rimanenti a disposizione circa due volte a pasto.

Senza ridondanza			"*nutrienti_dieta*"	
=>	Entità/Relazione	Volume	Operazione 7	Operazione 9
Accessi	**Utente**	20’000	0	1L
	Misurazione	800’000	1S	1L
	Segue	12’000	0	1L

	Dieta	200	0	1L

OverHead Operazione singola	2	4
Frequenza giornaliera operazione	8’000	72’000
OverHead giornaliero (S vale doppio)	16’000	288’000

Con ridondanza			"*nutrienti_dieta*"	
=>	Entità/Relazione	Volume	Operazione 7	Operazione 9
Accessi	**Utente**	20’000	1L	0
	Misurazione	800’000	1S	0
	Segue	12’000	1S	0
	Dieta	200	1L	1L
OverHead Operazione singola			6	1
Frequenza giornaliera operazione			8’000	72’000
OverHead giornaliero (S vale doppio)			48’000	72’000

Conclusioni	"*nutrienti_dieta*"	
	Calcolo	OverHead
Senza ridondanza	16’000 + 288’000	302’000
Con ridondanza	48’000 + 72’000	120’000
In seguito all’analisi svolta si è deciso di mantenere la ridondanza relativa all’attributo “*nutrienti_dieta”* poiché comporta un overhead minore sebbene porti ad un’ occupazione di memoria superiore di 120 KB al giorno		

La terza ridondanza identificata nello schema E-R riguarda l'attributo "*durata_riposo*" nell'entità **Sonno**. Questo attributo può anche essere calcolato sommando gli attributi “*durata_fase*” presenti nella relazione **Suddiviso** e gli attributi “*durata_risveglio*” presenti nell’entità Risveglio.
È importante notare che il monitoraggio del sonno è una funzionalità di nicchia utilizzata principalmente da atleti o appassionati di fitness, che rappresentano meno del 27.5% della community di Iberu.

Senza ridondanza			"*durata_riposo*"	
=>	Entità/Relazione	Volume	Operazione 6	Operazione 8
Accessi	**Sonno**	70’000	0	1L
	Suddiviso	330’000	(1S * 5)**	(1L * 5)**
	Fase	5	(1L * 5)**	0
	Risveglio	45’000	1S	1L
OverHead Operazione singola			17	12
Frequenza giornaliera operazione			5’000	4’000

OverHead giornaliero (S vale doppio)	93’500	48’000

Con ridondanza			"*durata_riposo*"	
=>	Entità/Relazione	Volume	Operazione 6	Operazione 8
Accessi	**Sonno**	70’000	1S	1L
	Suddiviso	330’000	(1S * 5)**	0
	Fase	5	(1L * 5)**	0
	Risveglio	45’000	1S	0
OverHead Operazione singola			19	1
Frequenza giornaliera operazione			5’500	4’000
OverHead giornaliero (S vale doppio)			104’500	4’000

Conclusioni	"*durata_riposo*"	
	Calcolo	OverHead
Senza ridondanza	93’500 + 48’000	141’500
Con ridondanza	104’500 + 4’000	108’500
In seguito all’analisi svolta si è deciso di mantenere la ridondanza relativa all’attributo “*durata_riposo”* poiché comporta un overhead minore sebbene porti ad un’ occupazione di memoria superiore di 40 KB al giorno		

Analizzando lo schema E-R, potrebbe sembrare a prima vista che l'attributo “*calorie_consumate*” sia una ridondanza a causa della presenza degli attributi “*calorie_100m*” nell'entità **Cardiovascolare** e "distanza" nell'entità **Cardio**, o degli attributi “*calorie_ripetizione*” nell'entità **Pesistica** e "*set*" e "*ripetizioni*" nell'entità **Palestra** . Tuttavia, è importante notare che l'attributo “*calorie_consumate*” tiene conto non solo delle calorie bruciate durante l'esecuzione degli esercizi specifici, ma anche delle calorie utilizzate per spostarsi da una postazione all'altra e dei movimenti effettuati durante le pause tra un esercizio e l'altro.

4.2 Eliminazione delle generalizzazioni

I sistemi tradizionali per la gestione delle basi di dati non consentono la rappresentazione attraverso generalizzazioni, risulta quindi necessario trasformare i tre costrutti presenti nello schema E-R in entità e relazioni.

Allenamento -> generalizzazione completa ed esclusiva.
Entità padre: **Allenamento** presenta una chiave primaria non riferita e possiede gli attributi “*data_allenamento*”, “*battito_allenamento*”, “*calorie_consumate*”, “*durata_allenamento*” ed “*intensità*”.
Entità figlie: **Palestra** presenta l’attributo “*gruppo_muscolare*” e la relazione **Composto** con l’entità **Esercizio(Pesistica)**. L’entità **Cardio** possiede gli attributi “*distanza*”, “*dislivello*”, “*durata_aerobico*” e “*durata_anaerobico*” ed è riferita dall’entità **Esercizio(Cardiovascolare)** attraverso la relazione **Formato**.
Per risolvere questa generalizzazione è stato deciso di accorpare solo l’entità **Palestra** all’entità **Allenamento** per le seguenti motivazioni:

1. E’ essenziale mantenere unitaria la relazione **Svolge** altrimenti si avrebbe un aumento ingiustificato dell’OverHead all’entità **Utente.**
2. Replicare i quattro campi dell’entità **Cardio** porterebbe nel caso di allenamenti in palestra a troppi campi NULL.

Esercizio -> generalizzazione completa ed esclusiva.
Entità padre: **Esercizio** presenta gli attributi “*nome_esercizio*”, “*recupero*” e “*descrizione_esercizio*”.
Entità figlie: **Pesistica** presenta gli attributi “*calorie_ripetizione*”, *“ripetizioni_consigliate*” e “*set_consigliati*” e la relazione **Composto** con l’entità **Allenamento.** L’entità **Cardiovascolare** possiede gli attributi “*distanza_minima*”, “*tipologia_esercizio_cardio*” e “*calorie_100m*” ed è riferita dall’entità **Cardio** attraverso la relazione **Formato.**

Per risolvere questa generalizzazione si è ritenuto opportuno rimuovere l'entità padre per le seguenti motivazioni:

1. Le differenze tra le entità figlie sono troppo marcate per permettere un accorpamento nell'entità padre.
2. L'entità **Esercizio** non presenta motivazioni valide, ad esempio relazioni unitarie essenziali, che ne giustifichino il mantenimento.
3. L'accorpamento dell'entità padre nelle entità figlie non porta ad alcun problema di aumento dell'overHead alle entità riferite.

4.3 Eliminazione degli attributi composti

Questa ristrutturazione si rende necessaria poiché il modello relazionale non permette di rappresentare direttamente attributi di questo tipo. Per superare questa limitazione, gli attributi verranno suddivisi in attributi individuali che li compongono.

Alimento -> "*nutrienti_alimento*", "*carboidrati_alimento*" e "*grassi_alimento*" verranno divisi in "*calorie_alimento*", "*proteine_alimento*", "*carboidrati_alimento*", "*zuccheri_alimento*", "*fibre_alimento*", "*grassi_alimento*", "*monoinsaturi_alimento*", "*polinsaturi_alimento*", "*colesterolo_alimento*", "*saturi_alimento*" e "*sale_alimento*"

Ricetta -> "*preparazione*" verrà suddivisa in "*difficoltà*", "*peso_finale*", "*procedimento*", "*durata_procedimento*" e "*durata_cottura*".

Pasto -> "*nutrienti_pasto*" sarà diviso in "*calorie_pasto*", "*carboidrati_pasto*" "*proteine_pasto*" e "*grassi_pasto*".

Dieta -> "*moltiplicatori*" sarà diviso in "*moltiplicatore_acqua*", "*moltiplicatore_grassi*", "*moltiplicatore_calorie*", "*moltiplicatore_proteine*" e "*moltiplicatore_carboidrati*".

Segue -> "*nutrienti_dieta*" verrà diviso in "*calorie_dieta*", "*carboidrati_dieta*", "*grassi_dieta*", "*proteine_dieta*" e "*acqua_dieta*".

4.4 Scelta di identificatori primari

Iberu è un'applicazione sviluppata con l'utente finale come punto di riferimento. Questo è evidente anche osservando lo schema E-R, in cui l'entità **Utente** assume un ruolo centrale e la maggior parte delle relazioni parte da essa. Di conseguenza, la chiave primaria di **Utente** viene utilizzata frequentemente.

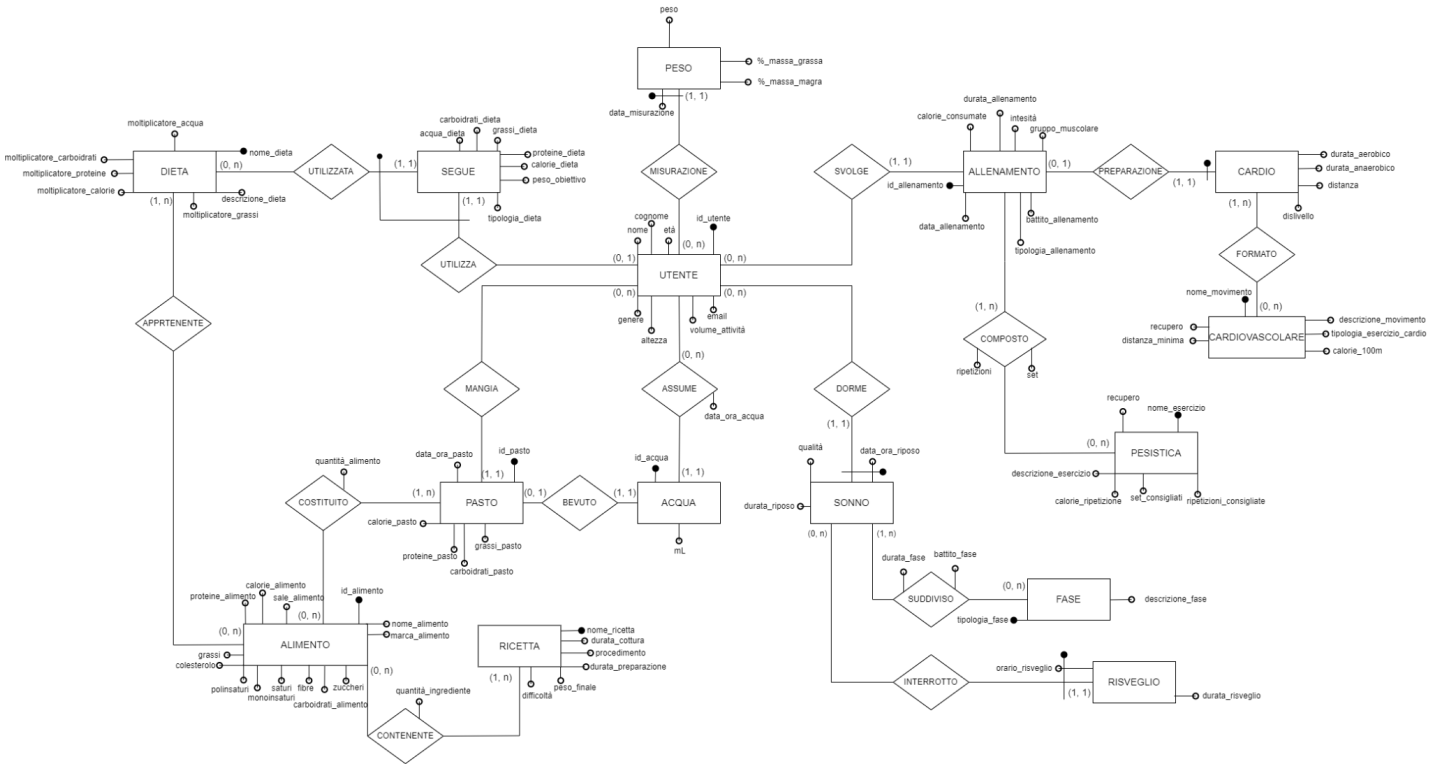
Per le ragioni elencate, è stata preferita l'opzione di utilizzare un identificatore numerico aggiuntivo come chiave primaria invece di utilizzare una stringa di lunghezza fissa *("id_utente"* varchar(50)).

Inoltre data la relazione 1 a 1 tra pasto e acqua come chiave primaria dell'entità è stato inserito un "*id_pasto"* e un *"id_acqua" come* chiave primaria entrambi varchar(50) per evitare la ripetizione di data e ora che sarebbero coincidenti.Questo risulta più efficiente poiché gli utenti di Iberu quando registrano un pasto registrano anche la quantità di liquidi assunti .

E' stato aggiunto id_allenemento perché uno stesso utente può fare più allenamenti nello stesso giorno e la data e ora sarebbe risultata essere troppo pesante come chiave primaria.

E' stata aggiunta la relazione 'segue' sostituendola alla relazione segue perchè un utente può anche non aderire a nessuna dieta quindi ci sarebbero nella sua tabella 7 campi nulli.

4.5 Diagramma ER ristrutturato



4.6 Schema relazionale e vincoli di integrità

Legenda : Gli attributi sottolineati sono le chiavi , gli asterischi indicano che il valore può essere anche nullo e le frecce → indicano le chiavi esterne. in particolare (A→B indica che B è chiave esterna di A.
Utente(id_utente, eta, cognome, nome, genere, altezza, volume_attivita, email)
Segue(id_utente,nome_dieta, grassi_dieta, carboidrati_dieta, acqua_dieta, peso_obiettivo, tipologia_dieta, calorie_dieta, proteine_dieta)
-segue.id_utente→Utente.id_utente
-segue.nome_dieta→Dieta.nome_dieta
Peso (data_misurazione ,id_utente,peso,%_massa_grassa,%_massa_magra)
- peso.id_utente-->utente.id_utente
Dieta(nome_dieta,descrizione_dieta,moltiplicatori_carboidrati,moltiplicatori_proteine,moltiplicatore_calorie, moltiplicatore_grassi, moltiplicatore_acqua)
Alimento(id_alimento, nome_alimento, marca_alimento, zuccheri, carboidrati_alimento, fibre, saturi, monosaturi, polinsaturi, colesterolo, grassi_alimento, poteine_alimento, calorie_alimento, sale_alimento, quantità_alimento)
Appartenente(nome_dieta,id_alimento)
- Appartenente.nome_dieta→Dieta.nome_dieta
- Appartenente.id_alimento→Alimento.id_alimento
Pasto(id_pasto, grassi_pasto, carboidrati_pasto, proteine_pasto, calorie_pasto, data_ora_pasto,id_acqua,id_utente)
-Pasto.id_utente→Utente.id_utente
-Pasto.id_acqua→Acqua-id_acqua
Acqua(id_acqua, ml)
Assume(id_utente, id_acqua, data_ora_acqua)
-Assume.id_acqua→Acqua-id_acqua
-Assume.id_utente→Utente.id_utente
Costituito (id_pasto, id_alimento, quantità_alimento)
- Costituito.id_pasto-->Pasto.id_pasto
- Costituito.id_alimento-->Alimento.id_alimento
Ricetta (id_ricetta,nome_ricetta, durata_cottura, procedimento, durata_preparazione, peso_finale, difficoltà)
Contenente (id_ricetta, id_alimento, quantità_ingrediente)
- Contenente.id_ricetta-->Ricetta.id_ricetta
- Contenente.id_alimento →Alimento.id_alimento
Allenamento(id_allenamento, calorie_consumate, durata_allenamento, data_allenamento, gruppo_muscolare, battito_allenamento, tipologia_allenamento, intensita, id_utente)
-Allenamento.id_utente→Utente.id_utente
Pesistica(nome_esercizio, ripetizioni_consigliate, descrizione_esercizio, calorie_ripetizione, set_consigliati, recupero)
Composto (id_allenamneto, nome_esercizio, set, ripetizioni)
- Composto.id_allenamento--> Allenamento.id_allenamento
- Composto.nome_esercizio--> Esercizio.nome_esercizio
Cardio (id_allenamento, duranta_aerobico, durata_anaerobicodistanza, dislivello)
- Cardio.id_allenamento--> Allenamento.id_allenamento
Cardiovascolare(nome_movimento,descrizione_movimento,tipologia_esercizio_cardio,calorie_100m, recupero,distanza_minima)
Formato(nome_movimento, id_allenamento)
- Formato.nome_movimento--> Cardiovascolare.nome_moviemento
- Formato.id_allenamento--> Cardio.id_allenamento
Sonno (data_ora_riposo, id_utente, qualità,durata_riposo)
- Sonno.id_utente-->Utente.id_utente
Fase (tipologia_fase, descrizione_fase)
Suddiviso(data_ora_riposo, id_utente, data_ora_riposo, tipologia_fase, battito_fase, durata_fase)
- Suddivio.id_utente-->Sonno.id_utente
- Suddivio.data_ora_riposo-->Sonno.data_ora_riposo
- Suddiviso.tipologia_fase-->Fase.tipologia_fase
Risveglio(orario_risveglio, data_ora_riposo, d_utente, durata_risveglio)
- Risveglio.id_utente-->Sonno.id_utente
- Risveglio.data_ora_riposo-->Sonno.data_ora_riposo

5.QUERY E INDICI
5.1 Query

Le query più significative individuate sono 7:

1. Mostrare tutte le ricette che utilizzano gli ingredienti inseriti dall'utente

```
SELECT r.nome_ricetta, r.peso_finale, r.durata_preparazione, r.durata_cottura, r.procedimento, r.difficolta
FROM ricetta r
JOIN contenente c
ON r.nome_ricetta = c.nome_ricetta
WHERE c.id_alimento =11
GROUP BY r.nome_ricetta
HAVING COUNT (*) = 1
ORDER BY r.difficolta DESC;
```

	nome_ricetta [PK] character varying (50)	peso_finale numeric (6,2)	durata_preparazione time without time zone	durata_cottura time without time zone	procedimento character varying (500)	difficolta smallint
1	Lasagne al Forno	500.00	01:30:00	00:45:00	Preparare il ragù, cuocere le lasagne, comporre gli strati e infornare.	3
2	Pollo Arrosto	1000.00	00:15:00	01:30:00	Preparare il pollo, insaporire con le erbe, infornare e cuocere fino a doratura.	3
3	Risotto ai Funghi	350.00	00:30:00	00:20:00	Preparare il soffritto, tostare il riso, aggiungere i funghi e cuocere a fuoco len...	3
4	Pasta alla Carbonara	400.00	00:15:00	00:10:00	Cuocere la pasta, saltare la pancetta e un uovo, unire il tutto e servire.	2

2. Per la dieta inserita dall'utente visualizzare quali sono i cibi più usati in ordine di utilizzo

```
SELECT al.nome_alimento, COUNT(*) AS migliori
FROM Segue s
JOIN Dieta d
ON s.nome_dieta = d.nome_dieta
JOIN Appartenente a
ON d.nome_dieta = a.nome_dieta
JOIN Alimento al
ON al.id_alimento = a.id_alimento
WHERE s.nome_dieta = 'Mediterranean'
GROUP BY al.nome_alimento
ORDER BY migliori DESC;
```

	nome_alimento character varying (50)	migliori bigint
1	Ecolab - Lime - A - Way 4/4 L	2
2	Cheese - Parmigiano Reggiano	2
3	Passion Fruit	2
4	Beer - Mcauslan Apricot	2
5	Sauce - Apple, Unsweetened	2
6	Beef - Roasted, Cooked	2
7	Vinegar - Red Wine	2
8	Wine - Beringer Founders Est...	2
9	Bread - Calabrese Baguette	2
10	Wine - Bourgogne 2002, La	2
11	Pastry - Apple Muffins - Mini	2
12	Squid - U - 10 Thailand	2
13	Potatoes - Yukon Gold 5 Oz	2
14	Grapes - Green	2
15	Beans - Yellow	2
16	Wine - Ruffino Chianti Classico	2

3. Visualizzare la durata media del sonno e delle relative 5 fasi di un utente inserito

```
SELECT u.nome, u.cognome, ROUND(us.media_riposo, 2) AS media_riposo_utente, ROUND(usa.media_add, 2) AS
media_add, ROUND(usl.media_sl, 2) AS media_sl, ROUND(usp.media_sp, 2) AS media_sp, ROUND(uspe.media_spe,
2) AS media_spe, ROUND(usr.media_r, 2) AS media_r
        FROM utente u
        JOIN sonno s
        ON s.id_utente = u.id_utente
        JOIN (SELECT u.id_utente, AVG(s.durata_riposo) AS media_riposo
        FROM sonno s JOIN utente u ON u.id_utente = s.id_utente
        WHERE u.nome = 'Matteo' AND u.cognome = 'Scavazza' GROUP BY u.id_utente) AS us
        ON us.id_utente = u.id_utente
        JOIN (SELECT u.id_utente, AVG(s.durata_fase) AS media_add
        FROM suddiviso s JOIN utente u ON u.id_utente = s.id_utente
        WHERE tipologia_fase = 'Stadio1:addormentamento' AND u.nome = 'Matteo' AND u.cognome = 'Scavazza'
GROUP BY u.id_utente) AS usa
        ON usa.id_utente = u.id_utente
        JOIN (SELECT u.id_utente, AVG(s.durata_fase) AS media_sl
        FROM suddiviso s JOIN utente u ON u.id_utente = s.id_utente
```

```
        WHERE tipologia_fase = 'Stadio2:sonno_leggero' AND u.nome = 'Matteo' AND u.cognome = 'Scavazza' GROUP
BY u.id_utente) AS usl
        ON usl.id_utente = u.id_utente
        JOIN (SELECT u.id_utente, AVG(s.durata_fase) AS media_sp
        FROM suddiviso s JOIN utente u ON u.id_utente = s.id_utente
        WHERE tipologia_fase = 'Stadio3:sonno_profondo' AND u.nome = 'Matteo' AND u.cognome = 'Scavazza'
GROUP BY u.id_utente) AS usp
        ON usp.id_utente = u.id_utente
        JOIN (SELECT u.id_utente, AVG(s.durata_fase) AS media_spe
        FROM suddiviso s JOIN utente u ON u.id_utente = s.id_utente
        WHERE tipologia_fase = 'Stadio4:sonno_profondo_effettivo' AND u.nome = 'Matteo' AND u.cognome =
'Scavazza' GROUP BY u.id_utente) AS uspe
        ON uspe.id_utente = u.id_utente
        JOIN (SELECT u.id_utente, AVG(s.durata_fase) AS media_r
        FROM suddiviso s JOIN utente u ON u.id_utente = s.id_utente
        WHERE tipologia_fase = 'Stadio5:fase_rem' AND u.nome = 'Matteo' AND u.cognome = 'Scavazza' GROUP BY
u.id_utente) AS usr
        ON usr.id_utente = u.id_utente
        GROUP BY u.nome, u.cognome, us.media_riposo, usa.media_add, usl.media_sl, usp.media_sp,
uspe.media_spe, usr.media_r  ;
```

	nome character varying (50)	cognome character varying (50)	media_riposo_utente numeric	media_add numeric	media_sl numeric	media_sp numeric	media_spe numeric	media_r numeric
1	Matteo	Scavazza	426.00	2100.00	1980.00	1710.00	2640.00	2295.00

4. Mostrare tutte le ricette che hanno alimenti che non appartengono a nessuna dieta

```
SELECT r.nome_ricetta
FROM ricetta r
WHERE r.nome_ricetta NOT IN(SELECT r.nome_ricetta
  FROM ricetta r
  JOIN contenente c ON r.nome_ricetta = c.nome_ricetta
  JOIN alimento a ON a.id_alimento = c.id_alimento
  JOIN appartenente ap ON ap.id_alimento = a.id_alimento
  GROUP BY r.nome_ricetta)
GROUP BY r.nome_ricetta
```

	nome_ricetta [PK] character varying (50)
1	Insalata di Pollo
2	Pancakes
3	Risotto ai Frutti di Mare
4	Zuppa di Pomodoro

5. Restituire nome, cognome, esercizio preferito e movimento preferito per gli utenti di genere femminile

```
SELECT u.nome, u.cognome,
    (SELECT p.nome_esercizio
     FROM utente u1
     JOIN allenamento a ON u1.id_utente = a.id_utente
     JOIN composto c ON a.id_allenamento = c.id_allenamento
     JOIN pesistica p ON c.nome_esercizio = p.nome_esercizio
     WHERE u1.genere = false AND u1.id_utente = u.id_utente
     GROUP BY p.nome_esercizio
     ORDER BY COUNT(*) DESC
     LIMIT 1) AS esercizio_preferito,
    (SELECT c.nome_movimento
     FROM utente u3
     JOIN allenamento a ON u3.id_utente = a.id_utente
     JOIN formato f ON a.id_allenamento = f.id_allenamento
     JOIN cardiovascolare c ON f.nome_movimento = c.nome_movimento
     WHERE u3.genere = false AND u3.id_utente = u.id_utente
     GROUP BY c.nome_movimento
     ORDER BY COUNT(*) DESC
```

```
    LIMIT 1) AS movimento_preferito
FROM utente u
WHERE u.genere = false;
```

	nome character varying (50)	cognome character varying (50)	esercizio_preferito character varying (50)	movimento_preferito character varying (50)
1	Giulia	Bianchi	Addominali crunch	[null]
2	Marta	Rossi	Panca piana	[null]
3	Francesca	Verdi	Shoulder press	Nuoto
4	Laura	Gialli	Curl bicipiti	[null]
5	Elisa	Neri	Leg extension	Ellittica
6	Valentina	Rosselli	Addominali crunch	Burpees

6. Visualizza nome, cognome, peso iniziale, peso obiettivo, peso medio e perdita di peso media misurata per ciascun utente di Iberu

```
SELECT u.nome, u.cognome, p.peso as peso_iniziale, s.peso_obiettivo, ROUND(AVG(pp.peso), 2) as peso_medio,
ROUND(AVG(pp.perdita_peso), 3) as perdita_peso_misurazioni
FROM utente u
JOIN segue s ON u.id_utente = s.id_utente
JOIN (SELECT *, peso - LAG(peso, 1) OVER(PARTITION BY id_utente
ORDER BY data_misurazione) perdita_peso
  FROM peso
  WHERE data_misurazione > '2023-03-17'
  ORDER BY data_misurazione) pp ON pp.id_utente = u.id_utente
JOIN (SELECT row_number() OVER(PARTITION BY id_utente order by data_misurazione) as row, id_utente, peso
FROM peso
WHERE data_misurazione >  '2023-03-17') p ON p.id_utente = u.id_utente
WHERE p.row = 1
GROUP BY u.nome, u.cognome, p.peso, s.peso_obiettivo
```

	nome character varying (50)	cognome character varying (50)	peso_iniziale numeric (6,2)	peso_obiettivo smallint	peso_medio numeric	perdita_peso_misurazioni numeric
1	Elisa	Neri	57.40	72	57.28	0.027
2	Francesca	Verdi	59.30	66	58.79	0.009
3	Giulia	Bianchi	59.40	63	60.04	0.082
4	Laura	Gialli	61.20	70	61.83	0.073
5	Lorenzo	Franco	74.50	70	74.20	0.467
6	Marco	Costa	70.20	75	72.28	0.133
7	Marta	Rossi	55.20	68	55.85	0.055
8	Matteo	Munari	75.20	76	74.07	-1.000
9	Matteo	Scavazza	94.50	70	88.27	-0.588
10	Riccardo	Cannaviello	72.10	65	73.70	1.167
11	Riccardo	Fabbian	72.60	75	71.23	0.041
12	Valentina	Rosselli	55.00	68	55.05	-0.018

7. Restituisce gli utenti del genere scelto dall'utente con più allenamenti a settimana e media dei valori nutrizionali

```
SELECT COUNT(*) AS totale_utenti,
      ROUND(AVG(un.calorie), 2) AS media_calorie,
      ROUND(AVG(un.carboidrati), 0) AS media_carboidrati,
      ROUND(AVG(un.proteine), 0) AS media_proteine,
      ROUND(AVG(un.grassi), 0) AS media_grassi
FROM (SELECT u.id_utente,
            AVG(p.calorie_pasto) AS calorie,
            AVG(p.carboidrati_pasto) AS carboidrati,
            AVG(p.proteine_pasto) AS proteine,
            AVG(p.grassi_pasto) AS grassi
     FROM utente u
```

```
JOIN pasto AS p
ON p.id_utente = u.id_utente
WHERE u.volume_attivita >= 2 AND u.genere=true
GROUP BY u.id_utente) AS un
```

	totale_utenti bigint	media_calorie numeric	media_carboidrati numeric	media_proteine numeric	media_grassi numeric
1	3	643.89	59	27	14

5.2 Indici

Al fine di semplificare la ricerca e quindi velocizzare il tempo per ottenere i risultati delle query sono stati utilizzati due indici :ind_name_alimento e difficoltà_ricetta.
Dato che il nome degli alimenti non è una chiave primaria ma viene utilizzato molto in lettura (es: nella query 2) è stato creato un indice della tabella 'alimento' sull' attributo 'nome_alimento'. E' importante evidenziare che al contrario della lettura l'inserimento di nuovi alimenti è meno frequente poiché il database di Iberu contiene già tutti gli alimenti conosciuti e, a regime, non vengono quasi mai aggiornati. (riferimento tabella 2.3).

```
CREATE INDEX ind_name_alimento ON Alimento(nome_alimento);
```

Nel secondo caso per le ricette esse vengono sempre visualizzate in ordine di difficoltà (es: nella query 1) e quindi si può agevolare la lettura creando un indice tra le ricette e le difficoltà Questa decisione è stata presa valutando che l'utente medio visualizza le ricette decisamente più frequentemente di quanto non le inserisca.(riferimento tabella 2.3)

```
CREATE INDEX difficolta_ricetta ON Ricetta(ricetta, difficolta);
```

6.Codice C++

Il codice c++ è costituito da un main, 6 funzioni e un vettore per le 7 query.
Il file.cpp deve essere compilato con il comando:

- g++ codice.cpp -o codice -I /usr/include/postgresql -lpq (utilizzando le librerie di sistema)
- g++ codice.cpp -o codice -L dependencies/lib -lpq (utilizzando le librerie nella cartella corrente)

Per il login con il Database sono definiti i valori dell'utente, delle porta, dell'host, e del database di default in modo tale che utilizzando il backup del database fornito la connessione avvenga sulla macchina locale direttamente.

La prima cosa che il programma chiede è la password per accedere al database, successivamente, mostrerà la lista delle possibili query tra le quali si può scegliere identificate da un numero da 1 a 7. Per eseguire una query bisogna inserire da tastiera il numero della query scelta, mentre per terminare l'esecuzione del programma va inserito '0'.
Alcune query (la numero 1,2,3,5 e 7) richiedono l'inserimento di parametri scelti dall'utente.