ParkMe

The easy way to park.

Alessandro Romeo

ParkMe

The easy way to park.

Il parcheggio è spesso visto come una fonte di stress e perdita di tempo, ma non se ne può fare a meno. Fortunatamente, l'innovazione tecnologica sta aiutando in questo ambito.

“*ParkMe*” è una soluzione che semplifica la ricerca del parcheggio, ottimizzando tempo e spazio.

“*ParkMe*” è una web app progettata per la gestione dei parcheggi nella città di Messina; è stata creata per offrire una soluzione completa ed efficiente per gestire i parcheggi nella città. Il sistema è basato su un database relazionale per archiviare informazioni sui parcheggi, consentendo agli utenti e agli operatori di accedere alle informazioni in tempo reale.

La web app può aiutare a ridurre la congestione del traffico e aumentare la sicurezza stradale, poiché gli utenti possono trovare rapidamente un parcheggio disponibile, evitando di circolare inutilmente alla ricerca di un posto auto.

Ottimizzare la mobilità ha tre principali risultati:

- Meno inquinamento: quando la mobilità è più veloce, le emissioni di gas si riducono. La naturale conseguenza è una migliore qualità dell’aria.
- Meno traffico: se nel prendere l’automobile si è già risolto il problema del parcheggio, si evita di congestionare le strade.
- Economia: trovare un parcheggio in tempi ridotti comporta meno stress e meno conseguenze in ottica lavorativa, oltre che personale.

Inoltre, in base alla progettazione, "*ParkMe*” potrebbe profilare il comportamento dell’utente utilizzatore: numero di macchine in possesso, giorni di preferenza, orari di preferenza, zone cittadine prediligiate, ecc.

Anche le amministrazioni comunali possono trarre beneficio dall'utilizzo del sistema, grazie alla profilazione degli utenti possono migliorare i suggerimenti e generare report analitici, monitorare l'utilizzo dei parcheggi e pianificare meglio la loro gestione. Inoltre, la stessa amministrazione comunale potrebbe mettere a disposizione di coloro che utilizzano *“ParkMe”,* mezzi green (ad esempio monopattini elettrici), utili soprattutto per stalli in zone lontane dal centro.

Un’ulteriore possibile implementazione sarebbe quella di mettere a disposizione degli utenti finali, hub/armadietti all’interno dei parcheggi chiusi, con la funzione di consegna spesa/pacchi al diretto hub.

In sintesi, "*ParkMe*" rappresenta una soluzione innovativa e conveniente per gestire i parcheggi in città, migliorando l'esperienza dell'utente e ottimizzando l'utilizzo dello spazio e del tempo.

Struttura della Web App

Funzionalità per gli utenti:

In primo luogo, la web app consente agli utenti di verificare la disponibilità di parcheggi (stalli con e senza colonnina di ricarica per auto ibride/elettriche) in una zona direttamente sulla mappa. In secondo luogo, permette anche di prenotare un posto auto in una zona circoscritta prima di recarsi al parcheggio, evitando di dover cercare un posto libero. *"ParkMe"* consente di pagare il costo del parcheggio (al minuto) con un semplice click e con la stessa semplicità eventualmente terminare la sosta anticipatamente. Inoltre, gli utenti possono visualizzare le tariffe dei parcheggi disponibili e scegliere quella più adatta alle loro esigenze. I clienti possono anche acquistare abbonamenti dinamici (settimanali, mensili, trimestrali, annuali, etc.) da vari operatori per accedere a tariffe scontate. Infine, gli utenti possono inviare feedback tramite la web app per segnalare eventuali problemi o per fare proposte.

Funzionalità per gli operatori:

Ad ogni operatore è affidato un tetto massimo di stalli. Gli operatori hanno accesso a una dashboard personalizzata per monitorare in tempo reale lo stato dei parcheggi sotto la loro gestione, la disponibilità, gli introiti e i costi associati. Gli operatori possono gestire i pagamenti, gli abbonamenti da offrire ai propri clienti e i clienti stessi. La dashboard consente agli operatori di aggiornare e gestire i dati dei parcheggi, nonché di visualizzare statistiche e report per monitorare le performance dei parcheggi.

Funzionalità per ente comunale:

"*ParkMe*" offre un'interfaccia intuitiva all'ente, tramite la quale esso ha la possibilità di svolgere diverse attività. Tramite una dashboard dedicata sarà possibile: assegnare gli stalli agli operatori, monitorare l'utilizzo degli stalli, modificare le tariffe degli stalli, monitorare gli introiti generati dagli stalli, e gestire gli operatori assegnati agli stalli, visualizzando i dati degli operatori e la loro attività.

Sistema di supporto e manutenzione:

"*ParkMe*" include un sistema di supporto e manutenzione per garantire che la web app funzioni correttamente e per risolvere eventuali problemi tecnici. La web app implementa un chat bot, il quale risponde alle domande più frequenti. Gli operatori possono segnalare problemi e richiedere assistenza attraverso la dashboard. Un possibile team di supporto e manutenzione sarà responsabile di risolvere i problemi e di garantire che la web app sia sempre disponibile per gli utenti.

Implementazione del Database:

Il database è stato progettato per garantire la sicurezza dei dati degli utenti e degli operatori. Per garantire l'integrità e la sicurezza dei dati, il database include anche meccanismi di autenticazione e autorizzazione per limitare l'accesso a determinate funzionalità solo agli utenti autorizzati. Inoltre, il database include anche meccanismi di backup e ripristino per garantire che i dati siano sempre disponibili in caso di problemi tecnici.

ANALISI DEI REQUISITI

Utente finale (End user):

Ogni utente è identificato da un codice univoco (corrispondente ad un nome utente o e-mail). Di ciascun utente viene memorizzato il codice fiscale, nome e cognome, sesso, luogo e data di nascita, indirizzo e numero di telefono, targhe delle auto in possesso.

Operatore (Agent):

Le operazioni dedicate agli operatori vengono gestite dagli Impiegati degli stessi. Ad ogni operatore corrisponde il proprio Bilancio.

Impiegato dell'operatore (Agent employee):

Ogni Impiegato è identificato tramite il proprio account aziendale. Quest'ultimo è generato al momento dell'assunzione di ciascun impiegato. Di ogni impiegato si conserva, data di assunzione e fine contratto, nome, cognome, numero di telefono, e-mail (privata), sesso, data e luogo di nascita, indirizzo di residenza.

Stalli/Posti auto (Parking space):

Ogni stallo ha un codice alfa numerico (es. A1, A2) tramite il quale si identifica il posto all'interno di una delle zone della città. Alcuni stalli possiedono anche una colonnina di ricarica. Inoltre, è fondamentale verificare che il posto risulti libero o occupato, e in questo ultimo caso inizio e fine (anche se provvisoria) della sosta. Ulteriore importante verifica, un posto può risultare prenotato.

Bilancio (Balance):

Il bilancio si riferisce sia ai guadagni inerenti all'ente che agli operatori, che i costi di entrambi. I guadagni dell'ente derivano interamente dall'offerta degli operatori per vincere la gara d'appalto, ed i costi sono di manutenzione (es. delle strade, stipendi degli ausiliari del traffico etc.). Al contrario i guadagni dei singoli operatori derivano dalle singole soste e dagli abbonamenti effettuati, e quindi i pagamenti dei singoli utenti finali.

Tariffa (Price):

Le tariffe degli stalli sono gestite dall'ente. La tariffa potrebbe variare in base alla posizione o al periodo dell'anno (anche in corrispondenza di giorni festivi).

Abbonamento (Subscription):

Gli utenti possono effettuare un piano di abbonamento ad un operatore per accedere a tariffe scontate. Del piano di abbonamento si salvano data di iscrizione e data di fine, codice di pagamento.

Prenotazione (Booking):

Un posto auto può essere prenotato da soltanto un utente. Viene salvato l'orario della prenotazione, la quale ha una scadenza.

Pagamento (Payment):

Un posto auto viene pagato da soltanto un utente nei confronti di un operatore. Viene salvato un codice identificativo del pagamento, data di pagamento effettuato.

Ente (Body):

Le operazioni dedicate all'ente vengono gestite dagli Impiegati dello stesso. L'ente avrà il proprio bilancio.

Impiegato dell'ente (Body employee):

Ogni Impiegato è identificato tramite il proprio account aziendale. Quest'ultimo è generato al momento dell'assunzione di ciascun impiegato. Di ogni impiegato si conserva, data di assunzione e fine contratto presso l'Ente, nome, cognome, numero di telefono, e-mail (privata), sesso, data e luogo di nascita, indirizzo di residenza.

FAQ:

Vengono memorizzate domande e risposte frequenti, da sfruttare tramite chatbot.

Profilo (Account):

Vengono raccolte usernames e le passwords di end users e/o employees mediante codifica one-way (hash). Per ogni account viene salvata la data di registrazione; nel caso dei Dipendenti (operatore o ente) l'account non sarà più valido alla fine del contratto lavorativo.

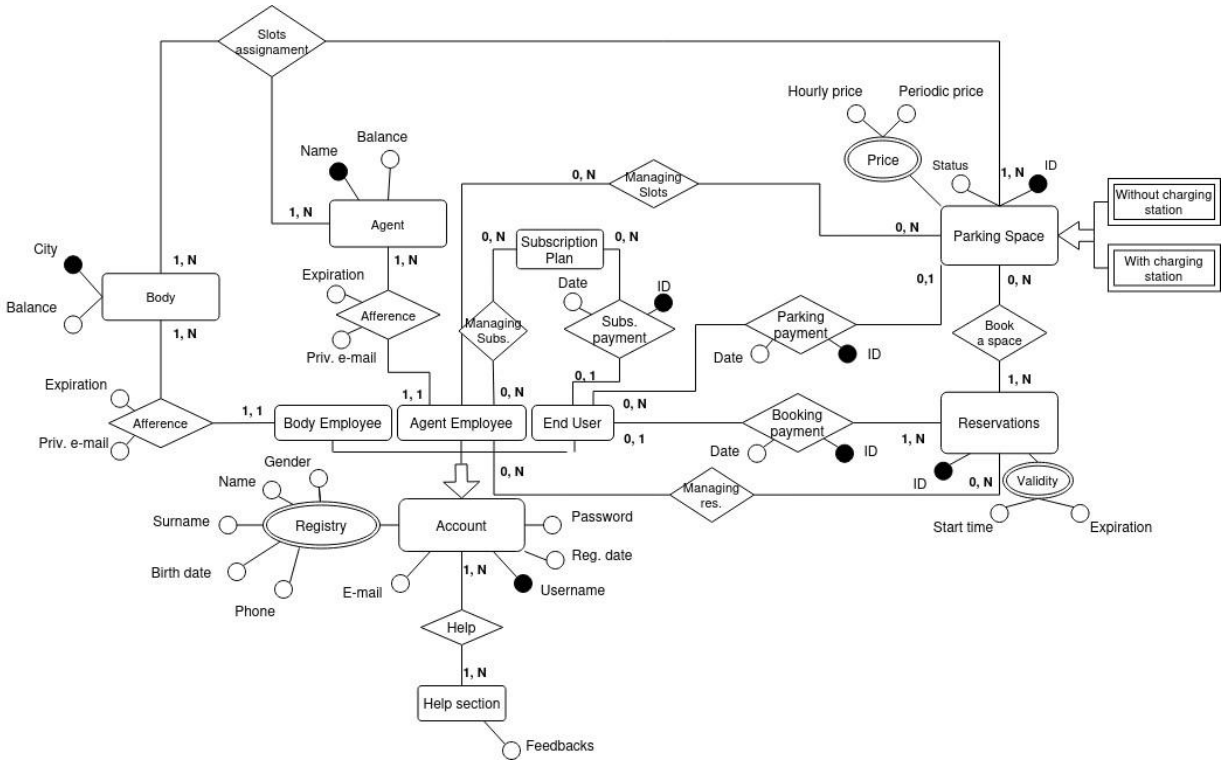
Feedbacks:

Vengono registrati i feedback da parte degli utenti. Viene salvato lo username dell'utente che ha creato il feedback, la data e l'ultima azione effettuata sulla web app.

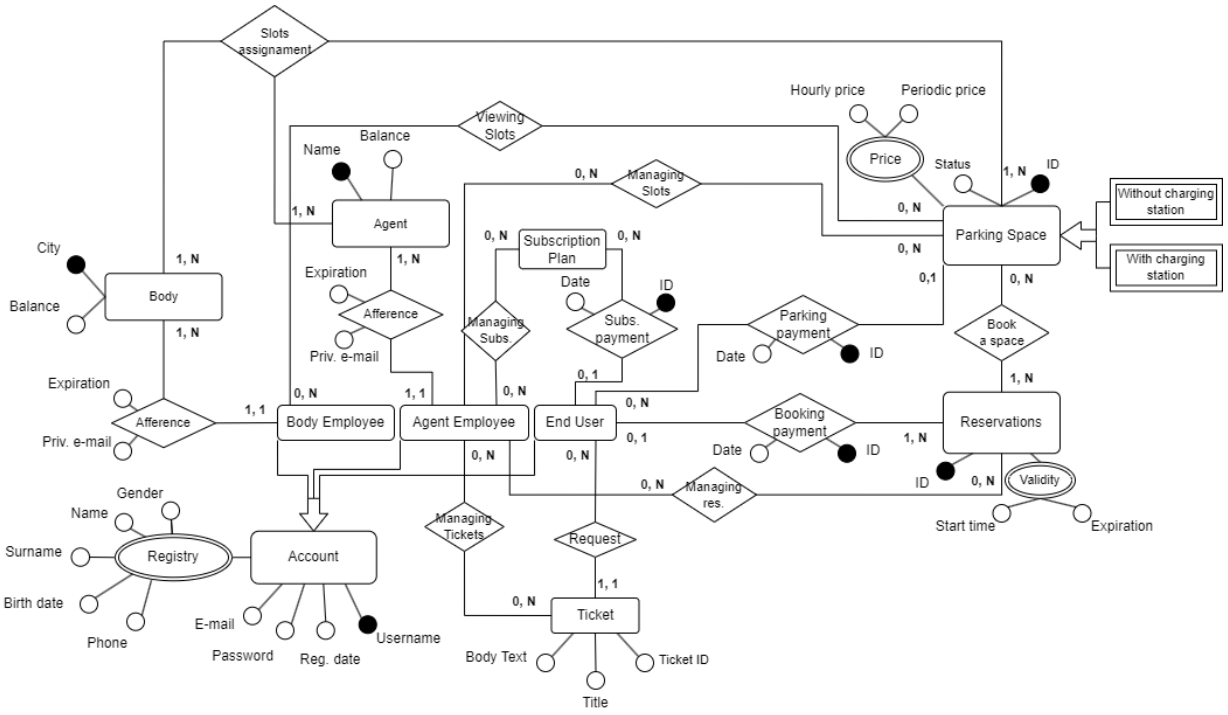
GLOSSARIO DEI TERMINI

Termine	Descrizione	Sinonimi	Collegamenti
End user	Colui che effettivamente parcheggia.	Utente finale	Abbonamento, Pagamento, Prenotazione, Posto auto, Feedback, Account
Ente	Ente comunale.	Body	Operatore, Tariffa, Bilancio
Operatore	Azienda vincitrice della gara d'appalto per gli stalli.	Agents	Bilancio, Ente, Abbonamento, Pagamento
Agent employee	Colui che lavora per un operatore.	Impiegato/Dipendente operatore	Abbonamento, Posto auto, Account
Body employee	Colui che lavora per l'ente.	Impiegato/Dipendente ente	Tariffa, Account
Abbonamento	Piano di sottoscrizione ad un operatore.	Subscription	End User, Operatore
Account	Account utente.	Profilo	Agent employee, Body employee, End User
Prenotazione	Prenotazione posto auto.	Booking	End User, Posto auto
Pagamento	Pagamento parcheggio.	Payment	End User, Operatore

Modello Concettuale (Prima Versione)



Modello Concettuale (Seconda Versione)



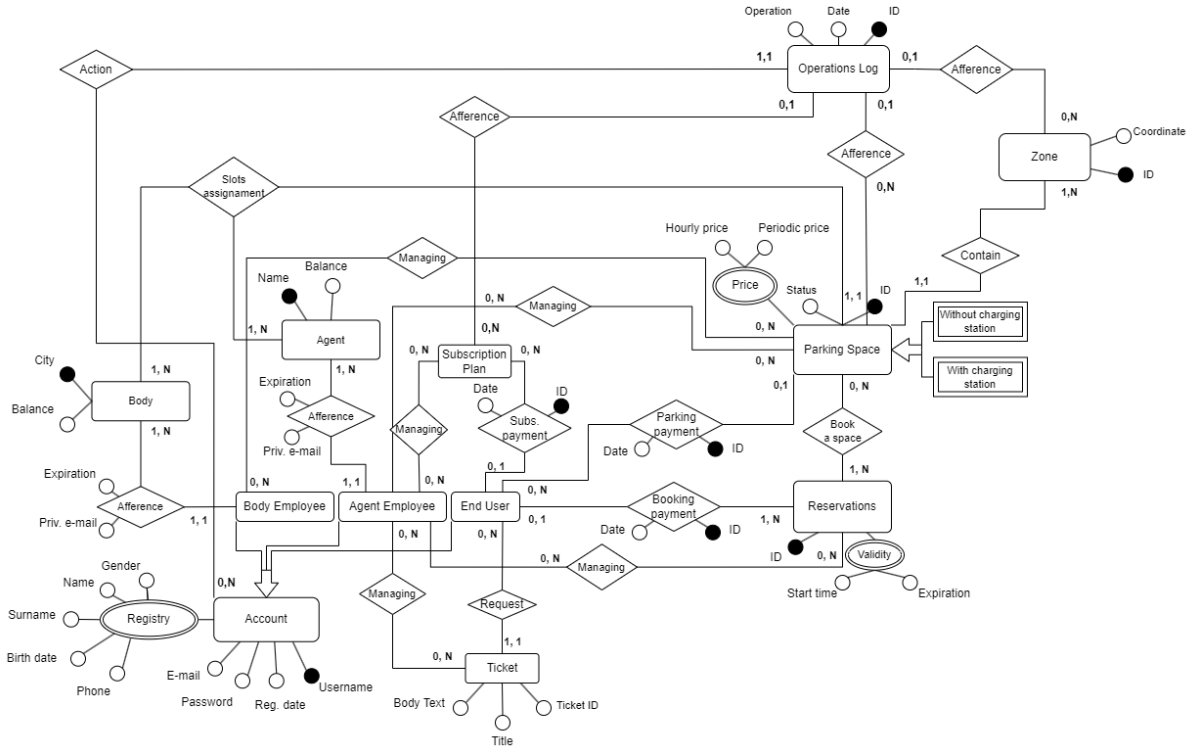
Note:

La relazione “Managing Slots” permette agli impiegati degli operatori di visualizzare lo stato dei propri parcheggi ma anche gestirne le tariffe.

La relazione “Managing Res.” permette agli impiegati degli operatori di visualizzare le prenotazioni effettuate dagli utenti finali.

La relazione “Viewing Slots” permette agli impiegati degli enti di visualizzare soltanto lo stato di tutti i parcheggi.

Modello Concettuale (Versione Finale)



Note:

È stata aggiunta l’entità “Operations Log”, la quale tiene traccia di tutte le operazioni compiute dagli impiegati, ed è implementata eventualmente in modo da essere in grado di monitorare anche le azioni dell’utente.

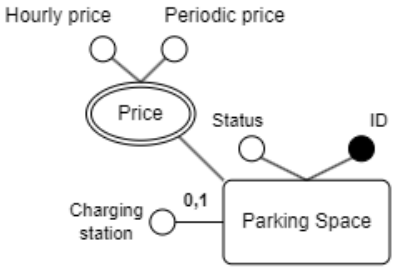
È stata aggiunta l’entità “Zone”, la quale contiene i singoli parcheggi, la quale possiede un ID e le coordinate corrispondenti.

Ristrutturazione

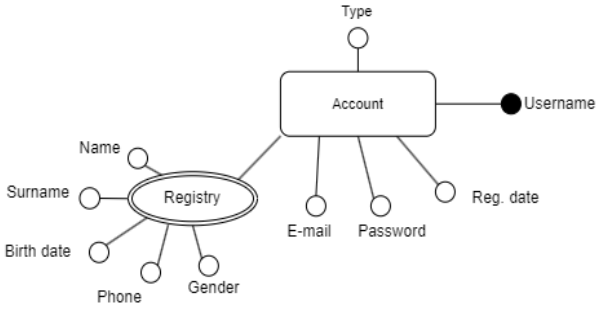
Per quanto riguarda eventuali modifiche di ristrutturazione del modello concettuale, si è deciso di eliminare l'entità "Reservation" in quanto risultata ridondante e complessa. Ovviamente, sono state risolte le generalizzazioni, implementando le diversificazioni tramite l'ausilio di appositi attributi alle entità.

Di seguito le modifiche apportate:

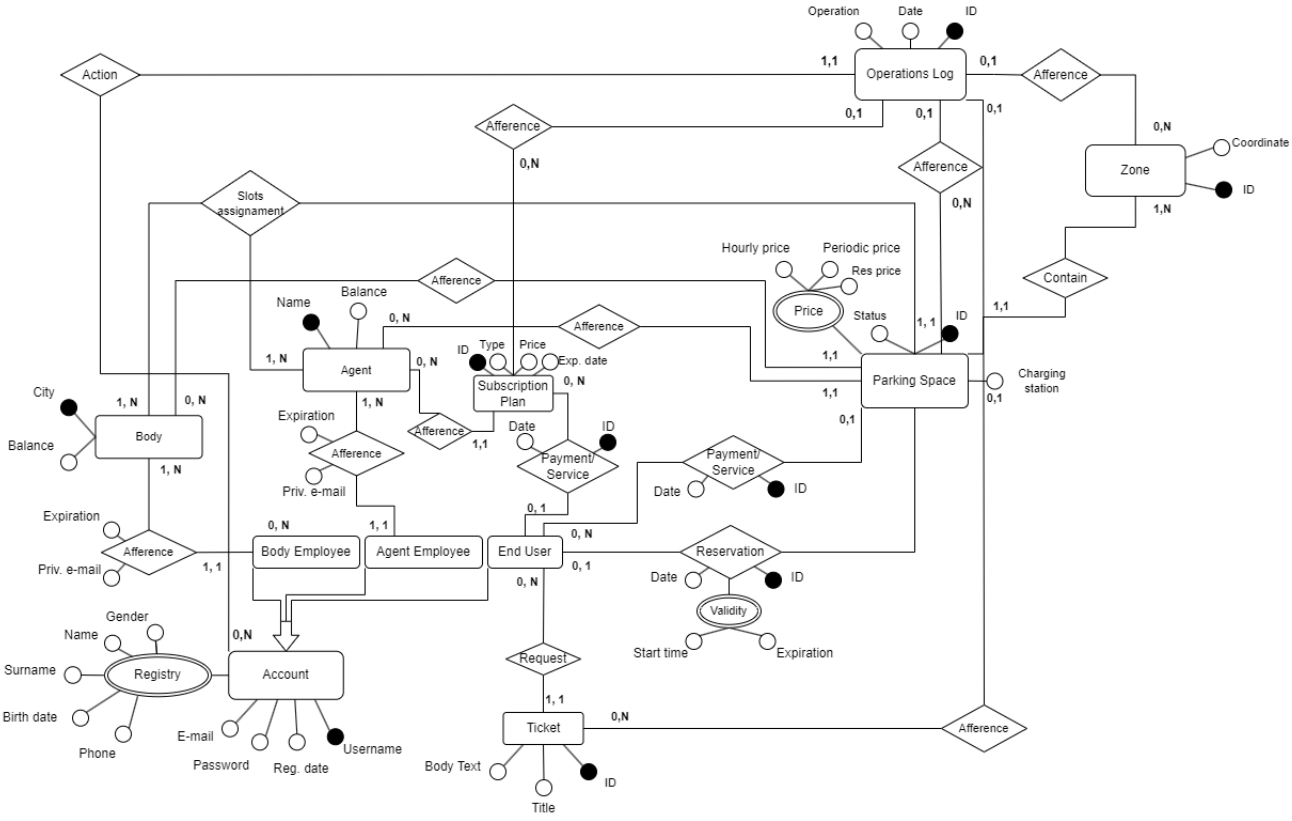
Entità Parking Space



Entità Account



Infine, viene riportato il modello completo ristrutturato:



Modello logico

Dopo aver effettuato la fase di ristrutturazione, è possibile procedere con la stesura del modello logico.

Account (Username, e-mail, password, reg.data, type, name, surname, birth date, phone, gender, priv_email_employee, expiration_contract_employee)

Agent (Name, Balance)

Body (City, Balance)

Payment (ID, Date, ID_user, Product_name)

Operations_Log (ID, Action, Date)

Parking Space (ID, Status, ID_body, ID_agent, Hourly price, Periodic price, Charging Station)

Reservations (ID, Price, Starting_time, Ending_time)

Ticket (ID, Title, Body text, ID_user)

Subscription plan (ID, Type, Price, Expiration date, ID_agent, ID_user)

Zone (ID, Coords)

Di seguito la creazione delle tabelle nel database:

```
CREATE TABLE `Account` (
  `username` varchar(50) NOT NULL,
  `email` varchar(255) NOT NULL,
  `password` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT
NULL,
  `reg_date` datetime DEFAULT CURRENT_TIMESTAMP,
  `type` enum('body_emp','sup_body_emp','end_user','agent_emp','sup_agent_emp')
CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `name` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `surname` varchar(50) NOT NULL,
  `birth_date` date DEFAULT NULL,
  `phone` varchar(20) DEFAULT NULL,
  `gender` enum('M','F','O') DEFAULT NULL,
  `Body_id` varchar(50) DEFAULT NULL,
  `Agent_id` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT
NULL,
  `priv_email_employee` varchar(255) DEFAULT NULL,
  `exp_emp_date` date DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `Agent` (
  `NAME` varchar(50) NOT NULL,
  `balance` float NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```
CREATE TABLE `Body` (
  `city` varchar(50) NOT NULL,
  `balance` float NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `Operations_Log` (
  `id` int NOT NULL,
  `action` enum('Modifica Abbonamento','Modifica Tariffa','Visualizzazione
Zona','Risoluzione Ticket','Assegnazione posto','Rimozione posto','Aggiunta
Impiegato','Rimozione Impiegato','Rimozione Abbonamento') CHARACTER SET utf8mb4
COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `zone_id` enum('A','B','C','D','E') DEFAULT NULL,
  `slot_id` varchar(4) DEFAULT NULL,
  `sub_id` int DEFAULT NULL,
  `res_id` int DEFAULT NULL,
  `emp_id` varchar(50) DEFAULT NULL,
  `user_id` varchar(50) NOT NULL,
  `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `Parking_Space` (
  `id` varchar(4) NOT NULL,
  `STATUS` enum('Available','Occupied','Reserved','Out of order') NOT NULL,
  `zone_id` enum('A','B','C','D','E') CHARACTER SET utf8mb4 COLLATE
utf8mb4_0900_ai_ci DEFAULT NULL,
  `id_body` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT
NULL,
  `id_agent` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT
NULL,
  `hourly_price` decimal(10,2) DEFAULT NULL,
  `periodic_price` decimal(10,2) DEFAULT NULL,
  `parking_ending_time` datetime DEFAULT NULL,
  `charging_station` tinyint(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `Payment` (
  `id` int NOT NULL,
  `amount` decimal(10,2) NOT NULL,
  `id_user` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `id_parking` varchar(4) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT
NULL,
  `id_reservation` int DEFAULT NULL,
  `id_subscription` int DEFAULT NULL,
  `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `Reservation` (
  `id` int NOT NULL,
  `id_user` varchar(50) NOT NULL,
  `id_parking` varchar(4) NOT NULL,
```

```
  `price` decimal(10,2) NOT NULL,
  `starting_time` datetime NOT NULL,
  `ending_time` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `Subscription` (
  `id` int NOT NULL,
  `title` varchar(20) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `type` enum('Settimanale','Mensile','Trimestrale','Semestrale','Annuale')
CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `price` decimal(10,2) NOT NULL,
  `expiration_date` date DEFAULT NULL,
  `id_agent` varchar(50) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `Ticket` (
  `id` int NOT NULL,
  `title` varchar(255) NOT NULL,
  `body_text` text NOT NULL,
  `answer` text,
  `employee` varchar(50) DEFAULT NULL,
  `id_user` varchar(50) NOT NULL,
  `status` enum('Aperto','Chiuso') NOT NULL DEFAULT 'Aperto',
  `date` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

CREATE TABLE `Zone` (
  `id` enum('A','B','C','D','E') NOT NULL,
  `Coords` text
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

ALTER TABLE `Account`
  ADD PRIMARY KEY (`username`),
  ADD UNIQUE KEY `email` (`email`),
  ADD KEY `Agent_rel` (`Agent_id`),
  ADD KEY `Body_rel` (`Body_id`);

--
-- Indici per le tabelle `Agent`
--
ALTER TABLE `Agent`
  ADD PRIMARY KEY (`NAME`);

--
-- Indici per le tabelle `Body`
--
ALTER TABLE `Body`
  ADD PRIMARY KEY (`city`);

--
```

```sql
-- Indici per le tabelle `Operations_Log`
--
ALTER TABLE `Operations_Log`
  ADD PRIMARY KEY (`id`),
  ADD KEY `Operations_Log_rel1` (`user_id`),
  ADD KEY `Operations_Log_rel2` (`slot_id`),
  ADD KEY `Operations_Log_rel3` (`sub_id`),
  ADD KEY `Operations_Log_rel4` (`res_id`),
  ADD KEY `Operations_Log_rel5` (`zone_id`),
  ADD KEY `Operations_Log_rel6` (`emp_id`);

--
-- Indici per le tabelle `Parking_Space`
--
ALTER TABLE `Parking_Space`
  ADD PRIMARY KEY (`id`),
  ADD KEY `id_body` (`id_body`),
  ADD KEY `id_agent` (`id_agent`),
  ADD KEY `Parking_Space_ibfk_3` (`zone_id`);

--
-- Indici per le tabelle `Payment`
--
ALTER TABLE `Payment`
  ADD PRIMARY KEY (`id`),
  ADD KEY `id_user` (`id_user`),
  ADD KEY `id_subscription` (`id_subscription`),
  ADD KEY `id_reservation` (`id_reservation`),
  ADD KEY `id_parking` (`id_parking`);

--
-- Indici per le tabelle `Reservation`
--
ALTER TABLE `Reservation`
  ADD PRIMARY KEY (`id`),
  ADD KEY `id_user` (`id_user`),
  ADD KEY `id_parking` (`id_parking`);

--
-- Indici per le tabelle `Subscription`
--
ALTER TABLE `Subscription`
  ADD PRIMARY KEY (`id`),
  ADD KEY `id_agent` (`id_agent`);

--
-- Indici per le tabelle `Ticket`
--
ALTER TABLE `Ticket`
  ADD PRIMARY KEY (`id`),
```

```
  ADD KEY `id_user` (`id_user`),
  ADD KEY `Ticket_ibfk_2` (`employee`);

--
-- Indici per le tabelle `Zone`
--
ALTER TABLE `Zone`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT per le tabelle scaricate
--


--
-- AUTO_INCREMENT per la tabella `Operations_Log`
--
ALTER TABLE `Operations_Log`
  MODIFY `id` int NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT per la tabella `Payment`
--
ALTER TABLE `Payment`
  MODIFY `id` int NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;

--
-- AUTO_INCREMENT per la tabella `Reservation`
--
ALTER TABLE `Reservation`
  MODIFY `id` int NOT NULL AUTO_INCREMENT;

--
-- AUTO_INCREMENT per la tabella `Subscription`
--
ALTER TABLE `Subscription`
  MODIFY `id` int NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=6;

--
-- AUTO_INCREMENT per la tabella `Ticket`
--
ALTER TABLE `Ticket`
  MODIFY `id` int NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;

--
-- Limiti per le tabelle scaricate
--


--
-- Limiti per la tabella `Account`
--
```

```
ALTER TABLE `Account`
  ADD CONSTRAINT `Agent_rel` FOREIGN KEY (`Agent_id`) REFERENCES `Agent` (`NAME`)
ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `Body_rel` FOREIGN KEY (`Body_id`) REFERENCES `Body` (`city`) ON
DELETE RESTRICT ON UPDATE RESTRICT;

--
-- Limiti per la tabella `Operations_Log`
--
ALTER TABLE `Operations_Log`
  ADD CONSTRAINT `Operations_Log_rel1` FOREIGN KEY (`user_id`) REFERENCES `Account`
(`username`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `Operations_Log_rel2` FOREIGN KEY (`slot_id`) REFERENCES
`Parking_Space` (`id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `Operations_Log_rel3` FOREIGN KEY (`sub_id`) REFERENCES
`Subscription` (`id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `Operations_Log_rel4` FOREIGN KEY (`res_id`) REFERENCES
`Reservation` (`id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `Operations_Log_rel5` FOREIGN KEY (`zone_id`) REFERENCES `Zone`
(`id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  ADD CONSTRAINT `Operations_Log_rel6` FOREIGN KEY (`emp_id`) REFERENCES `Account`
(`username`) ON DELETE RESTRICT ON UPDATE RESTRICT;

--
-- Limiti per la tabella `Parking_Space`
--
ALTER TABLE `Parking_Space`
  ADD CONSTRAINT `Parking_Space_ibfk_1` FOREIGN KEY (`id_body`) REFERENCES `Body`
(`city`),
  ADD CONSTRAINT `Parking_Space_ibfk_2` FOREIGN KEY (`id_agent`) REFERENCES `Agent`
(`NAME`),
  ADD CONSTRAINT `Parking_Space_ibfk_3` FOREIGN KEY (`zone_id`) REFERENCES `Zone`
(`id`) ON DELETE RESTRICT ON UPDATE RESTRICT;

--
-- Limiti per la tabella `Payment`
--
ALTER TABLE `Payment`
  ADD CONSTRAINT `Payment_ibfk_1` FOREIGN KEY (`id_user`) REFERENCES `Account`
(`username`),
  ADD CONSTRAINT `Payment_ibfk_2` FOREIGN KEY (`id_subscription`) REFERENCES
`Subscription` (`id`),
  ADD CONSTRAINT `Payment_ibfk_3` FOREIGN KEY (`id_reservation`) REFERENCES
`Reservation` (`id`),
  ADD CONSTRAINT `Payment_ibfk_4` FOREIGN KEY (`id_parking`) REFERENCES
`Parking_Space` (`id`) ON DELETE RESTRICT ON UPDATE RESTRICT;

--
-- Limiti per la tabella `Reservation`
--
```

```sql
ALTER TABLE `Reservation`
  ADD CONSTRAINT `Reservation_ibfk_1` FOREIGN KEY (`id_user`) REFERENCES `Account`
(`username`),
  ADD CONSTRAINT `Reservation_ibfk_2` FOREIGN KEY (`id_parking`) REFERENCES
`Parking_Space` (`id`) ON DELETE RESTRICT ON UPDATE RESTRICT;

--
-- Limiti per la tabella `Subscription`
--
ALTER TABLE `Subscription`
  ADD CONSTRAINT `Subscription_ibfk_1` FOREIGN KEY (`id_agent`) REFERENCES `Agent`
(`NAME`);

--
-- Limiti per la tabella `Ticket`
--
ALTER TABLE `Ticket`
  ADD CONSTRAINT `Ticket_ibfk_1` FOREIGN KEY (`id_user`) REFERENCES `Account`
(`username`),
  ADD CONSTRAINT `Ticket_ibfk_2` FOREIGN KEY (`employee`) REFERENCES `Account`
(`username`) ON DELETE RESTRICT ON UPDATE RESTRICT;
```