Easy Hotel Control – Dokumentacja Bazy Danych

Bazę danych do systemu EHC zaprojektowano w języku MySQL w środowisku MySQL Workbench 8.0.

1. Lista tabel bazy danych.

Zaprojektowana baza danych zawiera 9 tabel. Przedstawione zostały poniżej.

a) Tabela *room*:

roomID	floorNumber	label	roomStatus	dayPrice	lastClean	lastService	staffID	isCleaningNeeded	isServiceNeeded	roomDescription	capacity
1	1	101	1	40	2019-05-28 20:17:38	2019-05-28 20:00:32	5	0	0	SGL	1

Rys. 1.1 – kolumny tablicy *room*

Poszczególne kolumny pełnią następujące funkcje:

- **roomID** - stanowi unikalny identyfikator każdego z pokojów.
- **floorNumber** – przechowuje informację o piętrze na którym znajduje się pokój.
- **label** – trzycyfrowy numer pokoju (pierwsza cyfra stanowi prefiks, który wskazuje na piętro.
- **roomStatus** – status pokoju, mówiący czy pokój jest aktualnie wolny, czy nie.
- **dayPrice** – dzienna cena za dany pokój.
- **lastClean** – informacja o czasie ostatniego sprzątania pokoju.
- **lastService** – informacja o czasie ostatniego serwisu pokoju.
- **staffID** – id personelu, który ostatnio dokonał wszelkich prac w pokoju (sprzątaczka / serwis).
- **isCleaningNeeded** – flaga, świadcząca o tym, czy dany pokój wymaga usługi sprzątaczki.
- **isServiceNeeded** – flaga, świadcząca o tym, czy dany pokój wymaga usługi serwisowej.
- **roomDescription** – słowny opis pokoju.
- **capacity** – informacja o maksymalnej liczbie osób, jaka może przebywać w pokoju.

b) Tabela *reservation*:

resID	fk_clientID	fk_roomID	reservationStatus	reservationStart	reservationEnd	advanceValue	advanceStatus	peopleAmmount
1	4	2	0	2019-06-18 18:00:00	2019-06-20 10:00:00	60	1	2

Rys. 1.2 – kolumny tablicy *reservation*

Poszczególne kolumny pełnią następujące funkcje:
- **resID** – unikalne ID danej rezerwacji.
- **fk_clientID** – informacja o ID klienta rezerwującego.
- **fk_roomID** – ID rezerwowanego pokoju.
- **reservationStatus** – status rezerwacji.
- **reservationStart** – data zameldowania w hotelu / pokoju.
- **reservationEnd** – data wymeldowania z hotelu / pokoju.
- **advanceValue** – wartość zaliczki.
- **advanceStatus** – status zaliczki (zapłacona, bądź nie).
- **peopleAmmount** – liczba osób, które będą w pokoju w trakcie pobytu.

c) Tabela *clients*:

clientID	clientName	clientSurname	e_mail	phoneNumber	clientRegistrationTime	street	address	postalCode	city	clientPESEL	idCardNumber	advancePaid
5	Jan	Kowalski	jkow@gmail.com	168596124	2019-05-19 13:33:17	Kościuszki	7B/3	65-143	Sosnowiec	14250637891	SHW14523	NULL

Rys. 1.3 – kolumny tablicy *clients*

Poszczególne kolumny pełnią następujące funkcje:

- **clientID** – unikalne ID klienta.
- **clientName / clientSurname** – imię i nazwisko klienta.
- **e_mail** – e-mail klienta.
- **phoneNumber** – numer telefonu.
- **clientRegistrationTime** – data zarejestrowania klienta w bazie.
- **street / address / postalCode / city** – pola adresu klienta.
- **clientPESEL** – pesel klienta.
- **idCardNumber** – numer dowodu osobistego.
- **advancePaid** – flaga wskazująca, czy klient zapłacił zaliczkę.

d) Tabela *users*:

userID	permissionGiven	userLogin
1	1	easyhotelcontrol@gmail.com

Rys. 1.4 – kolumny tablicy *users*

Poszczególne kolumny pełnią następujące funkcje:

- **userID** – unikalne ID użytkownika
- **permissionGiven** – nadane uprawnienia
- **userLogin** – login / e-mail użytkownika

e) Tabela *permissions*:

permissionID	permissionDescription
1	ADMIN

Rys. 1.5 – kolumny tablicy *permissions*

Poszczególne kolumny pełnią następujące funkcje:

- **permissionID** – ID danego rodzaju uprawnienia.
- **permissionDescription** – słowny opis rodzaju uprawnienia.

f) Tabela *roomstate* (tabela pomocnicza):

stateID	stateDesc
0	FREE
1	RESERVED
2	OCCUPIED

Rys. 1.6 – kolumny tablicy *roomstate*

Poszczególne kolumny pełnią następujące funkcje:
- **stateID** – unikalne ID statusu pokoju.
- **stateDesc** – słowny opis statusu pokoju.

g) Tabela *services* (tabela pomocnicza):

serviceID	serviceType	serviceTag	serviceHourPrice
1	Wynajem roweru	STD	10

Rys. 1.7 – kolumny tablicy *services*

Poszczególne kolumny pełnią następujące funkcje:
- **serviceID** – unikalne ID danej usługi.
- **serviceType** – typ usługi.
- **serviceTag** – rodzaj usługi.
- **serviceHourPrice** – cena za godzinę wynajęcia danej usługi.

h) Tabela *itemservices* (tabela pomocnicze):

itserviceID	serviceType	serviceTag	serviceItemPrice
1	Butelka szampana	VIP	250

Rys. 1.8 – kolumny tablicy *itemservices*

Poszczególne kolumny pełnią następujące funkcje:
- **itserviceID** – unikalne ID danej usługi.
- **serviceType** – typ usługi.
- **serviceTag** – rodzaj usługi.
- **serviceItemPrice** – cena zakupu jednej sztuki danej usługi.

i) Tabela *clientsservices*:

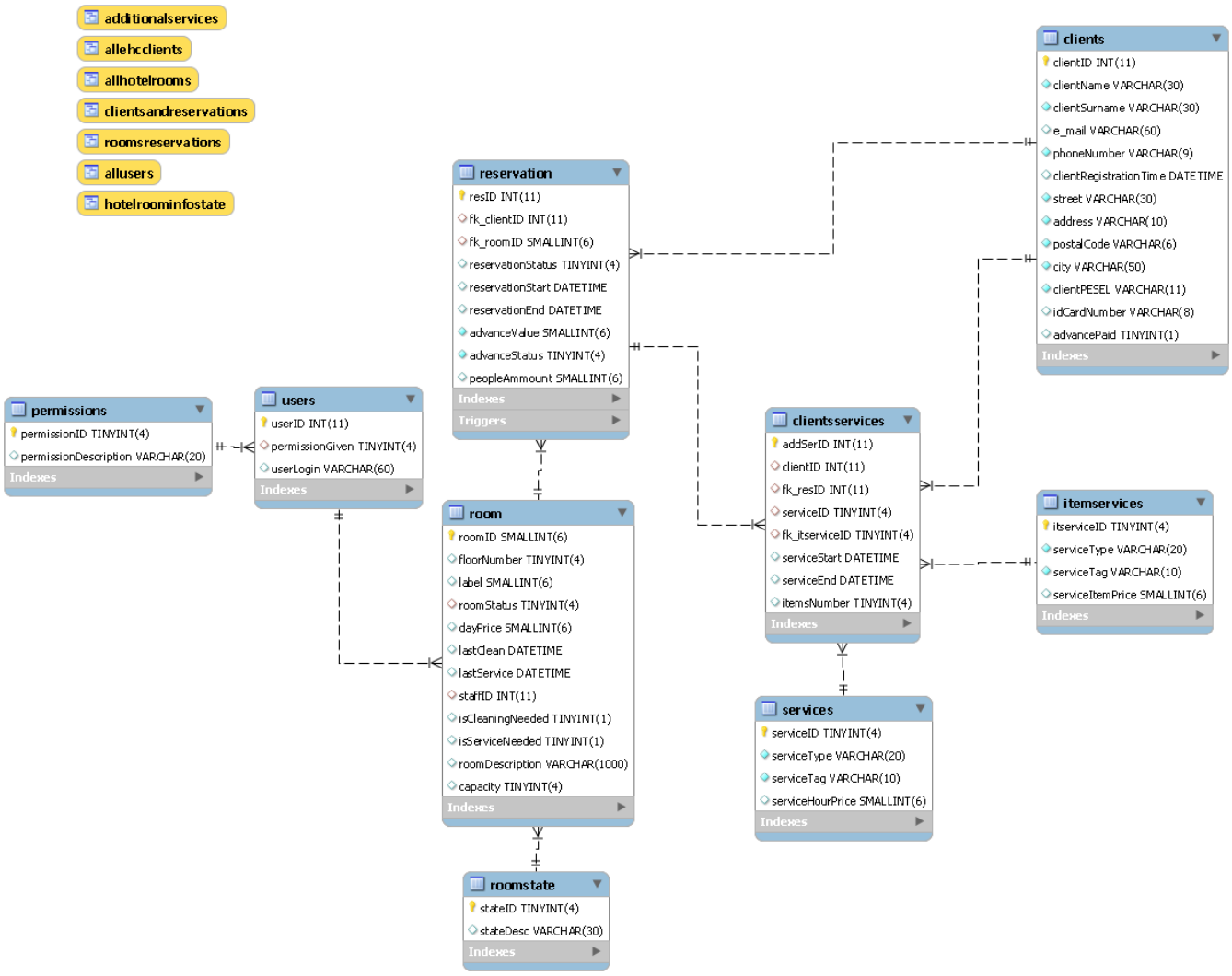
addSerID	clientID	fk_resID	serviceID	fk_itserviceID	serviceStart	serviceEnd	itemsNumber

Rys. 1.9 – kolumny tablicy *clientsservices*

Poszczególne kolumny pełnią następujące funkcje:
- **addSerID** – ID zakupu usługi.
- **clientID** – ID klienta zamawiającego usługę.
- **fk_resID** – ID rezerwacji, której dotyczy zamówiona usługa.

- **serviceID** – ID usługi (cechowanej godzinowo).
- **fk_itserviceID** – ID usługi (cechowanej sztukami).
- **serviceStart / serviceEnd** – początek i koniec świadczenia usługi (jeżeli klient zamówił usługę godzinową, np. wynajął rower).
- **itemsNumber** – liczba sztuk, którą zamówił klient (w przypadku skorzystania z usługi na sztuki).

2. Relacje między tabelami / diagram EER



Rys. 2.1. Diagram EER zaprojektowanej bazy danych

3. Projekty szczegółowe tabel

a) Tabela *room*:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
capacity	tinyint(4)		YES			select,insert,update,references	
dayPrice	smallint(6)		YES			select,insert,update,references	
floorNumber	tinyint(4)		YES			select,insert,update,references	
isCleaningNeeded	tinyint(1)		YES			select,insert,update,references	
isServiceNeeded	tinyint(1)		YES			select,insert,update,references	
label	smallint(6)		YES			select,insert,update,references	
lastClean	datetime		YES			select,insert,update,references	
lastService	datetime		YES			select,insert,update,references	
roomDescription	varchar(1000)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
roomID	smallint(6)		NO			select,insert,update,references	auto_increment
roomStatus	tinyint(4)		YES			select,insert,update,references	
staffID	int(11)		YES			select,insert,update,references	

Rys. 3.1. Kolumny tabeli *room* wraz z typami danych

Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	roomID
☑	roomID	BTREE	YES	roomID
☑	staffID	BTREE	NO	staffID
☑	roomStatus	BTREE	NO	roomStatus

Rys. 3.2. Indeksy w tabeli *room*

Name	Schema	Table	Column	Referenced Sch...	Referenced Table	Referenced Col...
reservation_ibfk_2	ehc_2	reservation	fk_roomID	ehc_2	room	roomID
room_ibfk_4	ehc_2	room	roomStatus	ehc_2	roomstate	stateID
room_ibfk_3	ehc_2	room	staffID	ehc_2	users	userID

Rys. 3.3. Klucze obce w encji *room*

b) Tabela *reservation*:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
advanceStatus	tinyint(4)		NO			select,insert,update,references	
advanceValue	smallint(6)		NO			select,insert,update,references	
fk_clientID	int(11)		YES			select,insert,update,references	
fk_roomID	smallint(6)		YES			select,insert,update,references	
peopleAmmount	smallint(6)		YES			select,insert,update,references	
reservationEnd	datetime		YES			select,insert,update,references	
reservationStart	datetime		YES			select,insert,update,references	
reservationStatus	tinyint(4)		YES			select,insert,update,references	
resID	int(11)		NO			select,insert,update,references	auto_increment

Rys. 3.4. Kolumny tabeli *reservation* wraz z typami danych

Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	resID
☑	fk_clientID	BTREE	NO	fk_clientID
☑	fk_roomID	BTREE	NO	fk_roomID

Rys. 3.5. Indeksy w tabeli *reservation*

Name	Event	Timing	Created	SQL Mode	Definer	Client Character...	Connection Coll...	Database Collation
preventReservationOnNonExisti...	INSERT	BEFORE	2019-05-17 21:1...	STRICT_TRANS...	root@localhost	utf8mb4	utf8mb4_0900_...	utf8mb4_0900_...
preventReservationOnNonExisti...	INSERT	BEFORE	2019-05-21 22:0...	STRICT_TRANS...	root@localhost	utf8mb4	utf8mb4_0900_...	utf8mb4_0900_...

Rys. 3.6. Wyzwalacze dla encji *reservation*

Użyte powyżej wyzwalacze, uaktywniają się przed dokonaniem dodania rezerwacji. Zapobiegają rezerwowaniu na nieistniejący pokój, czy też na nieistniejącego klienta.

Name	Schema	Table	Column	Referenced Sch...	Referenced Table	Referenced Col...
clientsservices_ibfk_3	ehc_2	clientsservices	fk_resID	ehc_2	reservation	resID
reservation_ibfk_1	ehc_2	reservation	fk_clientID	ehc_2	clients	clientID
reservation_ibfk_2	ehc_2	reservation	fk_roomID	ehc_2	room	roomID

Rys. 3.7. Klucze obce w encji *reservation*

c) Tabela *clients*:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
address	varchar(10)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
advancePaid	tinyint(1)		YES			select,insert,update,references	
city	varchar(50)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
clientID	int(11)		NO			select,insert,update,references	auto_increment
clientName	varchar(30)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
clientPESEL	varchar(11)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
clientRegistrationTime	datetime		YES			select,insert,update,references	
clientSurname	varchar(30)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
e_mail	varchar(60)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
idCardNumber	varchar(8)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
phoneNumber	varchar(9)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
postalCode	varchar(6)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
street	varchar(30)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	

Rys. 3.8. Kolumny tabeli *clients* wraz z typami danych

Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	clientID
☑	clientPESEL	BTREE	YES	clientPESEL
☑	clientPESEL_2	BTREE	YES	clientPESEL
☑	idCardNumber	BTREE	YES	idCardNumber
☑	e_mail	BTREE	YES	e_mail

Rys. 3.9. Indeksy w tabeli *clients*

Name	Schema	Table	Column	Referenced Sch...	Referenced Table	Referenced Col...
clientsservices_ibfk_1	ehc_2	clientsservices	clientID	ehc_2	clients	clientID
reservation_ibfk_1	ehc_2	reservation	fk_clientID	ehc_2	clients	clientID

Rys. 3.10. Klucze obce w tabeli *clients*

d) Tabela *clientsservices*:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
addSerID	int(11)		NO			select,insert,update,references	auto_increment
clientID	int(11)		YES			select,insert,update,references	
fk_itserviceID	tinyint(4)		YES			select,insert,update,references	
fk_resID	int(11)		YES			select,insert,update,references	
itemsNumber	tinyint(4)		YES			select,insert,update,references	
serviceEnd	datetime		YES			select,insert,update,references	
serviceID	tinyint(4)		YES			select,insert,update,references	
serviceStart	datetime		YES			select,insert,update,references	

Rys. 3.11. Kolumny tabeli *clientsservices* wraz z typami danych

Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	addSerID
☑	clientID	BTREE	NO	clientID
☑	serviceID	BTREE	NO	serviceID
☑	fk_resID	BTREE	NO	fk_resID
☑	fk_itserviceID	BTREE	NO	fk_itserviceID

Rys. 3.12. Indeksy w tabeli *clientsservices*

Name	Schema	Table	Column	Referenced Sch...	Referenced Table	Referenced Col...
clientsservices_ibfk_1	ehc_2	clientsservices	clientID	ehc_2	clients	clientID
clientsservices_ibfk_4	ehc_2	clientsservices	fk_itserviceID	ehc_2	itemservices	itserviceID
clientsservices_ibfk_3	ehc_2	clientsservices	fk_resID	ehc_2	reservation	resID
clientsservices_ibfk_2	ehc_2	clientsservices	serviceID	ehc_2	services	serviceID

Rys. 3.13. Klucze obce w tabeli *clientsservices*

e) Tabela *services*:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
serviceHourPrice	smallint(6)		YES			select,insert,update,references	
serviceID	tinyint(4)		NO			select,insert,update,references	auto_increment
serviceTag	varchar(10)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
serviceType	varchar(20)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	

Rys. 3.14. Kolumny tabeli *services* wraz z typami danych

Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	serviceID
☑	serviceID	BTREE	YES	serviceID

Rys. 3.15. Indeksy w tabeli *services*

Name	Schema	Table	Column	Referenced Sch...	Referenced Table	Referenced Col...
clientsservices_ibfk_2	ehc_2	clientsservices	serviceID	ehc_2	services	serviceID

Rys. 3.16. Klucze obce w encji *services*

f) Tabela *itemservices*:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
itserviceID	tinyint(4)		NO			select,insert,update,references	auto_increment
serviceItemPrice	smallint(6)		YES			select,insert,update,references	
serviceTag	varchar(10)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
serviceType	varchar(20)		NO	utf8mb4	utf8mb4_0900_...	select,insert,update,references	

Rys. 3.17. Kolumny tabeli *itemservices* wraz z typami danych

Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	itserviceID
☑	itserviceID	BTREE	YES	itserviceID

Rys. 3.18. Indeksy w tabeli *itemservices*

Name	Schema	Table	Column	Referenced Sch...	Referenced Table	Referenced Col...
clientsservices_ibfk_4	ehc_2	clientsservices	fk_itserviceID	ehc_2	itemservices	itserviceID

Rys. 3.19. Klucze obce w encji *itemservices*

g) Tabela *users*:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
permissionGiven	tinyint(4)		YES			select,insert,update,references	
userID	int(11)		NO			select,insert,update,references	auto_increment
userLogin	varchar(60)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references	

Rys. 3.20. Kolumny tabeli *users* wraz z typami danych

Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	userID
☑	userLogin	BTREE	YES	userLogin
☑	permissionGiven	BTREE	NO	permissionGiven

Rys. 3.21. Indeksy w tabeli *users*

Name	Schema	Table	Column	Referenced Sch...	Referenced Table	Referenced Col...
room_ibfk_3	ehc_2	room	staffID	ehc_2	users	userID
users_ibfk_1	ehc_2	users	permissionGiven	ehc_2	permissions	permissionID

Rys. 3.22. Klucze obce w encji *users*

h) Tabela *permissions*:

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
permissionDescription	varchar(20)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update,references	
permissionID	tinyint(4)		NO			select,insert,update,references	auto_increment

Rys. 3.23. Kolumny tabeli *permissions* wraz z typami danych

Indexes in Table

Visible	Key	Type	Uni...	Columns
☑	PRIMARY	BTREE	YES	permissionID
☑	permissionID	BTREE	YES	permissionID
☑	permissionID_2	BTREE	YES	permissionID

Rys. 3.24. Indeksy w tabeli *permissions*

Name	Schema	Table	Column	Referenced Sch...	Referenced Table	Referenced Col...
users_ibfk_1	ehc_2	users	permissionGiven	ehc_2	permissions	permissionID

Rys. 3.25. Klucze obce w encji *permissions*

4. Widoki – kod źródłowy

```sql
CREATE VIEW clientsandreservations AS
    SELECT
        clients.clientID AS 'clientID',
        clients.clientName AS 'clientName',
        clients.clientSurname AS 'clientSurname',
        clients.e_mail AS 'clientEmail',
        clients.clientPESEL AS 'pesel',
        room.label AS 'roomLabel',
        room.floorNumber AS 'floorNumber',
        room.roomDescription AS 'roomDescription',
        room.dayPrice * ABS(DATEDIFF(reservation.reservationStart,
                        reservation.reservationEnd)) AS 'totalPrice',
        reservation.resID AS 'reservationID',
        reservation.fk_roomID AS 'roomID',
        reservation.reservationStart AS 'reservationStart',
        reservation.reservationEnd AS 'reservationEnd'
    FROM
        reservation
            LEFT JOIN
        clients ON reservation.fk_clientID = clients.clientID
            LEFT JOIN
        room ON reservation.fk_roomID = room.roomID
    ORDER BY reservation.resID;
```

Rys. 4.1. Widok *clientsandreservations*

Powyższy widok ma za zadanie kumulować informacje o klientach i rezerwacjach przez nich dokonanych.

```sql
CREATE VIEW roomsreservations AS
    SELECT
        room.roomID AS 'roomID',
        room.floorNumber AS 'floorNumber',
        room.label AS 'roomLabel',
        room.roomStatus AS 'roomStatus',
        room.capacity AS 'capacity',
        reservation.resID AS 'reservationID',
        reservation.reservationStart AS 'reservationStart',
        reservation.reservationEnd AS 'reservationEnd',
        reservation.peopleAmmount AS 'peopleAmmount',
        reservation.reservationStatus AS 'reservationStatus',
        clients.clientID AS 'clientID'
    FROM
        room
            LEFT JOIN
        reservation ON room.roomID = reservation.fk_roomID
            INNER JOIN
        clients ON reservation.fk_ClientID = clients.ClientID
    ORDER BY room.roomID;
```

Rys. 4.2. Widok *roomsreservations*

Widok roomsreservations, podaje informacje o pokojach i rezerwacjach na nie.

```
50 •  CREATE VIEW allusers AS
51        SELECT
52            users.userID AS 'userID',
53            users.userLogin AS 'userLogin',
54            permissions.permissionID AS 'permissionID',
55            permissions.permissionDescription AS 'permissionDescription'
56        FROM
57            users
58                LEFT JOIN
59            permissions ON users.permissionGiven = permissions.permissionID
60        ORDER BY users.permissionGiven;
```

Rys. 4.3. Widok *allusers*

Widok allusers służy wyświetlaniu informacji o użytkownikach systemu EHC.

```
63 •  CREATE VIEW allehcclients AS
64        SELECT
65            clients.clientID AS 'clientID',
66            clients.clientName AS 'clientName',
67            clients.clientSurname AS 'clientSurname',
68            clients.e_mail AS 'clientEmail',
69            clients.clientPESEL AS 'pesel',
70            clients.phoneNumber AS 'phoneNumber',
71            clients.city AS 'city',
72            clients.street AS 'street',
73            clients.address AS 'address',
74            clients.postalCode AS 'postalCode'
75        FROM
76            clients
77        ORDER BY clients.clientName ASC , clients.clientSurname ASC;
```

Rys. 4.4. Widok *allehcclients*

Widok allehcclients podaje informacje o klientach.

```
80 •  CREATE VIEW allhotelrooms AS
81        SELECT
82            room.floorNumber AS 'floorNumber',
83            room.label AS 'roomLabel',
84            room.dayPrice AS 'dayPrice',
85            room.roomID AS 'roomID',
86            room.roomDescription AS 'roomDescription',
87            room.roomStatus AS 'roomStatus',
88            room.capacity AS 'roomCapacity'
89        FROM
90            room
91        ORDER BY room.floorNumber ASC , room.label ASC;
```

Rys. 4.5. Widok *allhotelrooms*

```sql
CREATE VIEW hotelroominfostate AS
    SELECT
        room.floorNumber AS 'floorNumber',
        room.label AS 'roomLabel',
        room.roomDescription AS 'roomDescription',
        room.roomID AS 'roomID',
        room.lastClean AS 'lastCleaning',
        room.lastService AS 'lastService',
        room.isCleaningNeeded AS 'cleaningNeeded',
        room.isServiceNeeded AS 'serviceNeeded',
        room.staffID AS 'staffID',
        room.capacity AS 'roomCapacity',
        roomstate.stateDesc AS 'roomState'
    FROM
        room
            INNER JOIN
        roomstate ON room.roomStatus = roomstate.stateID
    ORDER BY room.floorNumber ASC , room.label ASC;
```

Rys. 4.6. Widok *hotelroominfostate*

Powyższy widok podaje informacje o pokojach i ich statusach.

```sql
CREATE VIEW additionalservices AS
    SELECT
        clientsservices.addSerID AS 'serviceOrderID',
        clients.clientName AS 'clientName',
        clients.clientSurname AS 'clientSurname',
        services.serviceID AS 'timeServiceID',
        services.serviceType AS 'serviceType',
        services.serviceHourPrice AS 'hourPrice',
        itemservices.itserviceID AS 'itemServiceID',
        itemservices.serviceType AS 'itemServiceType',
        itemservices.serviceItemPrice AS 'oneItemPrice',
        clientsservices.fk_resID AS 'reservationID',
        clientsservices.serviceStart AS 'serviceStart',
        clientsservices.serviceEnd AS 'serviceEnd',
        clientsservices.itemsNumber AS 'itemsNumber'
    FROM clients
            LEFT OUTER JOIN
        clientsservices ON clients.clientID = clientsservices.clientID
            LEFT OUTER JOIN
        services ON services.serviceID = clientsservices.serviceID
            INNER JOIN
        itemservices ON itemservices.itserviceID = clientsservices.fk_itserviceID
    ORDER BY clients.clientID;
```

Rys. 4.7. Widok *additionalservices*

Widok additionalservices wyświetla informacje o klientach i zamówionych przez nich usługach dodatkowych podczas pobytu w hotelu.

5. Procedury – krótki opis funkcjonalności

5.1.Procedury dotyczące klientów

a)
```
CREATE PROCEDURE procedureClientSelectEmail(paramEmail VARCHAR(60))
```

Procedura ma za zadanie wybrać dane klienta wskazanego odpowiednim adresem e-mail.

b)
```
CREATE PROCEDURE procedureSelectAllPESEL()
```

Procedura wybiera wszystkie numery PESEL zapisane w bazie.

c)
```
CREATE PROCEDURE procedureClientSelectByPESEL(paramPESEL VARCHAR(11))
```

Procedura wybiera dane klienta na podstawie podanego numeru PESEL.

d)
```
CREATE PROCEDURE procedureClientSelect(paramName VARCHAR(30), paramSurname
VARCHAR(30), paramPESEL VARCHAR(11))
```

Procedura wybierająca klienta wskazanego przez imię, nazwisko oraz PESEL.

e)
```
CREATE PROCEDURE procedureShowClients()
```

Wyświetlanie wszystkich danych (rekordów) o wszystkich klientach w bazie.

f)
```
CREATE PROCEDURE procedureClientAdd(paramClName VARCHAR(30),
paramClSurname VARCHAR(30), paramEMail VARCHAR(60),
paramPhNumber VARCHAR(9),paramStreet VARCHAR(30),
paramAddress VARCHAR(120),paramPostalCode VARCHAR(6),paramCity VARCHAR(50),
paramPESEL VARCHAR(11),paramIDNum VARCHAR(8))
```

Procedura dodająca nowego klienta do bazy danych.

g)
```
CREATE PROCEDURE reserveAdditionalTimeTypedService
(paramPesel VARCHAR(11),paramResID INT,paramStart DATETIME,
paramEnd DATETIME, paramServiceNum TINYINT)
```

Procedura umożliwiająca odnotowanie zamówienia przez klienta usługi dodatkowej (cechowanej czasowo – np. wynajem roweru na jedną godzinę).

h)
```
CREATE PROCEDURE reserveAdditionalItemTypedService
(paramPesel VARCHAR (11),paramResID INT,
paramStart DATETIME,paramServiceNum TINYINT, paramItemsNumber TINYINT)
```

Procedura umożliwiająca odnotowanie zamówienia przez klienta usługi dodatkowej (cechowanej ilością sztuk).

5.2. Procedury związane z pokojami

a)
```
CREATE PROCEDURE procedureRoomInfo(paramLabel SMALLINT,
permissionParam TINYINT)
```

Wyświetlanie różnych informacji o pokoju w zależności od przyznanych uprawnień użytkownika.

b)
```
CREATE PROCEDURE procedureTakeRoomOnFloor(paramFloorNumber TINYINT,
permissionParam TINYINT)
```

Wyświetlanie różnych informacji o pokojach na danym piętrze w zależności od przyznanych uprawnień użytkownika.

c)
```
CREATE PROCEDURE procedureAddRoom(paramFloorNumber TINYINT,
paramLabel SMALLINT ,paramDayPrice SMALLINT,
paramRoomDesc VARCHAR(1000),paramCapacity TINYINT)
```

Procedura dodająca nowy pokój do bazy danych.

d)
```
CREATE PROCEDURE deleteRoom(paramLabel SMALLINT)
```

Procedura usuwająca pokój. Note: Uwzględniając klucz obcy **fk_roomID** w encji **reservation** - posiada on zdefiniowaną akcję ON DELETE CASCADE, sprawia ona, że z usunięciem pokoju wiąże się także usunięcie wszystkich powiązanych z nim rezerwacji.

e)
```
CREATE PROCEDURE takeAllRooms()
```

Procedura podająca informacje o wszystkich pokojach.

f)
```
CREATE PROCEDURE takeOneRoom(paramLabel SMALLINT)
```

Procedura wyświetlająca informacje o podanym pokoju.

g)
```
CREATE PROCEDURE updateRoomLastCleaning (paramLabel SMALLINT,
paramUserEmail VARCHAR(60))
```

Procedura aktualizująca stan czystości pokoju – usługa sprzątaczki. Zaktualizowanie czasu ostatniego sprzątania, wpis id pracownika oraz ustawienie flagi świadczącej o potrzebie sprzątania w danym pokoju na 0.

h) `CREATE PROCEDURE updateRoomLastService (paramLabel SMALLINT, paramUserEmail VARCHAR(60))`

Procedura aktualizująca stan serwisu pokoju – usługa serwisanta. Zaktualizowanie czasu ostatniego serwisu, wpis id pracownika oraz ustawienie flagi świadczącej o potrzebie serwisu w danym pokoju na 0.

i) `CREATE PROCEDURE changeCleaningState(paramLabel SMALLINT)`

Procedura zmieniająca stan pokoju – potrzebne sprzątanie.

j) `CREATE PROCEDURE changeServiceState(paramLabel SMALLINT)`

Procedura zmieniająca stan pokoju – potrzebny serwis.

k) `CREATE PROCEDURE showRoomWhereCleaningNeeded()`

Procedura wyświetlająca wszystkie pokoje, gdzie aktualnie potrzebne jest sprzątanie.

l) `CREATE PROCEDURE showRoomWhereServiceNeeded()`

Procedura wyświetlająca wszystkie pokoje, gdzie aktualnie potrzebna jest usługa serwisu.

m) `CREATE PROCEDURE procedureRoomInfoState(paramLabel SMALLINT)`

Procedura wyświetlająca informację statusowe (status pokoju, stan czystości, serwis) o danym pokoju.

n) `CREATE PROCEDURE procedureReturnFloorNumber()`

Procedura zwracająca aktualną liczbę pięter w hotelu.

5.3. Procedury związane z rezerwacjami

a)
```
CREATE PROCEDURE makeRoomReservation(paramLabel SMALLINT,
paramPesel VARCHAR(11), paramStart DATETIME, paramEnd DATETIME,
paramAdvance SMALLINT, paramPeopleAmmount SMALLINT)
```

Procedura dodająca nową rezerwację do bazy danych.

b)
```
CREATE PROCEDURE procedureDeleteReservation(paramResID INT)
```

Usunięcie rezerwacji (np. z losowych przyczyn, rezygnacji, pomyłki itp.)

c)
```
CREATE PROCEDURE markAdvanceAsPaid(paramResID INT)
```

Procedura odznaczająca wpłacenie zaliczki przez klienta.

d)
```
CREATE PROCEDURE markAdvanceNotPaid(paramResID INT)
```

Procedura ustawiająca status zaliczki na 0 – niezapłacona (w razie pomyłki – klient przy rezerwacji domyślnie nie zapłacił zaliczki).

e)
```
CREATE PROCEDURE markRoomAsReserved(paramLabel SMALLINT)
```

Procedura odznaczająca pokój jako zajęty (status pokoju ustawiany na 2 – OCCUPIED – ktoś aktualnie przebywa w pokoju, jest on zajęty przez któregoś z klientów).

f)
```
CREATE PROCEDURE procedureChooseAvailableRoomsAllFloors
(paramStart DATETIME, paramEnd DATETIME)
```

Procedura wyświetlająca dostępne pokoje w zadanym przedziale czasowym – używana w usłudze rezerwacji.

g)
```
CREATE PROCEDURE selectReservation(paramResID INT)
```

Procedura wyświetlająca informacje o kliencie i rezerwacji przez niego dokonanej.

h)
```
CREATE PROCEDURE selectReservationEndingOn(paramEnd DATETIME)
```

Procedura wyświetlająca wszystkie rezerwacje, które kończą się we wskazanym dniu.

i) `CREATE PROCEDURE selectReservationByRoom(paramLabel SMALLINT)`

Procedura wyświetlająca informacje o wszystkich rezerwacjach na dany pokój.

j) `CREATE PROCEDURE procedureActivateReservation(paramResID INT)`

Procedura aktywacji rezerwacji – zameldowanie. Ustawienie statusu rezerwacji na 1 – aktywna.

k) `CREATE PROCEDURE procedureActivateReservedRoom(paramLabel SMALLINT)`

Procedura ustawiająca status pokoju na 2 – OCCUPIED – aktualnie aktywna rezerwacja na danym pokoju. Procedura używana przy usłudze zameldowania, razem z procedurą procedureActivateReservation.

l) `CREATE PROCEDURE procedureCheckOutOfEHC(paramResID INT)`

Procedura dezaktywacji rezerwacji – wymeldowanie. Ustawienie statusu rezerwacji na 2 – po terminie.

k) `CREATE PROCEDURE procedureSetValidRoomState(paramLabel SMALLINT)`

Procedura ustawiająca odpowiedni status pokoju po wymeldowaniu. Gdy po wymeldowaniu, dalej istnieją rezerwacje na wskazany pokój status pokoju ustawiany na 1, w przeciwnym razie na 0.Wywoływana po procedurze procedureCheckOutOfEHC.

l) `CREATE PROCEDURE chooseClientEmailWhereRoomDeleted(paramLabel SMALLINT)`

Zwracanie maila klientów, którzy zarezerwowali wcześniej pokój, który administracja hotelu ma zamiar usunąć (z różnych przyczyn, np. restrukturyzacja hotelu, prace remontowe itd.) w celu umożliwienia późniejszego wysłania maila do nich z informacją o zaistniałym fakcie.

m) `CREATE PROCEDURE selectAllReservations()`

Procedura wyświetlająca informacje o wszystkich pokojach i rezerwacjach.

n) `CREATE PROCEDURE selectReservationsPastNow()`

Procedura zwracająca wszystkie rezerwacje rozpoczynające się później niż aktualna data.

5.4. Procedury związane z użytkownikami

a) `CREATE PROCEDURE addEhcUser(paramEmail VARCHAR(60),paramPermission TINYINT)`

Dodawanie nowego użytkownika systemu.

b) `CREATE PROCEDURE deleteEhcUser(paramEmail VARCHAR(60))`

Usuwanie użytkownika.

c) `CREATE PROCEDURE selectUser (paramEmail VARCHAR(60))`

Zwracanie informacji o kliencie wskazanego adresem e-mail.

d) `CREATE PROCEDURE selectAllUsers ()`

Zwracanie informacji o wszystkich użytkownikach.

e) `CREATE PROCEDURE selectUserEmails (paramPermission TINYINT)`

Zwracanie maili wszystkich użytkowników o zadanych uprawnieniach.

5.5. Procedury związane z raportowaniem

a) `CREATE PROCEDURE procedureReservationBillUp(paramResID INT)`

Procedura podliczająca rezerwację. Zwracanie ceny za wynajem pokoju, usługi dodatkowe klientów w trakcie rezerwacji oraz ich sumowanie.

b) `CREATE PROCEDURE makeDailyReport()`

Procedura odpowiedzialna za dzienny raport. Zwracanie liczby ludzi aktualnie przebywających w hotelu, liczby wolnych / zajętych pokoi, dzienny przychód oraz aktualną datę.

6. Wyzwalacze / triggers

a) `CREATE TRIGGER preventReservationOnNonExistingRoom BEFORE INSERT ON reservation`

Zapobieganie dokonania rezerwacji na nieistniejący pokój.

b) `CREATE TRIGGER preventReservationOnNonExistingClient BEFORE INSERT ON reservation`

Zapobieganie dokonania rezerwacji na nieistniejącego klienta.

c) `CREATE TRIGGER preventInvalidServiceBooking BEFORE INSERT ON clientsservices`

Zapobieganie dokonania zakupu usługi dodatkowej na nieistniejącego klienta.