**CRISTINA AILOAEI (266543) IT-2Y-S18**

**Chapter 4**

**4.8**

The relation *Room* has as foreign key the attribute *hotelNo*.

The relation *Booking* has as foreign keys the attributes *hotelNo*, *roomNo* and *guestNo*.

The entity integrity constraint states that in a base relation no attribute of a primary key can be null. As all our foreign keys are either a primary key in their home relation or are part of the attributes which create a primary key, the referential integrity constraint is applied.

**4.9**

**Hotel**

|  |  |  |
| --- | --- | --- |
| hotelNo | hotelName | city |
| H01 | Grosvenor | London |
| H02 | Eurobar | Oxford |
| H03 | New Union | Menchester |

**Room**

|  |  |  |  |
| --- | --- | --- | --- |
| roomNo | hotelNo | type | price |
| 10 | H01 | single | 40 |
| 12 | H01 | double | 65 |
| 21 | H01 | family | 75 |
| 33 | H02 | family | 80 |
| 41 | H02 | double | 50 |
| 16 | H03 | single | 33 |

**Booking**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| hotelNo | guestNo | dateFrom | dateTo | roomNo |
| H01 | G03 | 16-07-2018 | 20-07-2018 | 10 |
| H01 | G01 | 20-08-2018 | 29-08-2018 | 21 |
| H02 | G02 | 17-06-2018 | 25-06-2018 | 33 |
| H03 | G04 | 20-04-2018 | 25-04-2018 | 16 |

**Guest**

|  |  |  |
| --- | --- | --- |
| guestNo | guestName | guestAddress |
| G01 | William Jensen | Stefansgade 7 st. th. Roskilde |
| G02 | Emma Nielsen | Allegade 63 Skanderborg |
| G03 | Mathias Hansen | Ostergade 13 Fredericia |
| G04 | Tobias Pedersen | Sundvej 44 Horsens |

**Appropriate constraints:**

Check constraint to allow only the values *single*, *double* and *family* in the “type” field in “Room” relation.

Check constraint to allow only values between 10 and 100 in the “price” field in “Room” relation.

Check constraint to allow only values between 1 and 100 in the “roomNo” field in “Room” relation.

Check constraint to allow only values between *01-01-2018* and *31-12-2019* in the “dateFrom” and “dateTo” fields in “Room” relation.

Check constraint to allow only values between *H01* and *H99* in the “hotelNo” field in “Hotel” relation.

Check constraint to allow only values between *G01* and *G99* in the “guestNo” field in “Guest” relation.

**Chapter 5**

**5.12**

a) RA: π hotelNo, hotelName, city (Hotel)

TRC: { h.hotelNo, h.hotelName, h.city | Hotel(h) }

DRC: { h1, h2, h3 | Hotel(h1 , h2 , h3 ) }

b) RA: π roomNo( σ type = 'single' ∧ price < 20 (Room))

TRC: { r.roomNo | Room(r) ∧ r.type = 'single' ∧ r.price < 20 }

DRC: {r1 | ∃r4 (Room(r1 , 'single', r4 ) ∧ r4 < 20 ) }

c) RA: π guestName, guestAddress (Guest)

TRC: { g.guestName, g.guestAddress | Guest(g)}

DRC: { g2, g3 | Guest(g2, g3) }

d) RA: π type, price(Room ⋈ σ hotelName = 'Grosvenor'(Hotel))

TRC: { r.roomType, r.price | Room(r) ∧ ∃h (Hotel(h) ∧ h.hotelName = 'Grosvenor' ∧ h.hotelNo = r.hotelNo) }

DRC: { r3 , r4 | ∃h1 , h3 (Room(h1 , r3 , r4 ) ∧ Hotel(h1 , 'Grosvenor', h3 ))

e) RA: Guest ⋉ σ dateFrom ≤ CURRENT\_DATE ∧ dateTo ≥ CURRENT\_DATE ∧ hotelName = 'Grosvenor' (Hotel ⋈ Booking)

TRC: {g | Guest(g) ∧ ∃ b, h(Booking(b) ∧ Hotel(h) ∧ b.guestNo = g.guestNo ∧ b.hotelNo = h.hotelNo

∧ b.dateFrom ≤ CURRENT\_DATE ∧ b.dateTo ≥ CURRENT\_DATE ∧ h.hotelName = 'Grosvenor') }

DRC: {g1, g2, g3 | Guest (g1, g2, g3) ∧ ∃ h1, h3, b3, b4, b5 (Hotel (h1, 'Grosvenor', h3)

∧ Booking (h1, g1, b3, b4, b5) ∧ b3 ≤ CURRENT\_DATE ∧ b4 ≥ CURRENT\_DATE)}

f) RA: π roomNo, type, price, guestName (Room ⟕ (σ hotelName = 'Grosvenor' (Hotel)

⋈ σ dateFrom ≤ CURRENT\_DATE ∧ dateTo ≥ CURRENT\_DATE(Booking) ⋈ Guest))

TRC: GrosvenorRoom(r) ≔ Room(r) ∧ ∃h(Hotel(h) ∧ h.hotelName = 'Grosvenor' ∧ h.hotelNo = r.hotelNo)

CurrentBooking(g, r) ≔ Guest(g) ∧ ∃b(Booking(b) ∧ b.hotelNo = r.hotelNo ∧ b.roomNo = r.roomNo

∧ b.dateFrom ≤ CURRENT\_DAY ∧ b.dateTo ≥ CURRENT\_DAY ∧ b.guestNo = g.guestNo)

{r.roomNo, r.type, r.price, g.guestName|BR(r) ∧ (CB(g, r) ∨(≦ ∃g’ CB(g’, r) ∧ g = (null, null, null, null)))}

DRC: GrosvenorRoom(r1, r2, r3, r4) ≔ Room(r1, r2, r3, r4) ∧ ∃ h3 Hotel(r2, 'Grosvenor', h3)

CurrentBooking(g2, r1, r2) ≔ ∃ g1, g3 (Guest(g1, g2, g3) ∧ ∃ b3, b4(Booking(r2, g1, b3, b4, r1)

∧ b3 ≤ CURRENT\_DAY ∧ b4 ≥ CURRENT\_DAY)) {r1 r3, r4, g2 | ∃ r2 BR(r1, r2, r3, r4) ∧ (CB(g2, r1, r2) ∨ (¬ ∃g2’ CB(g2’, r1, r2) ∧ g2 = null))}

g) RA: π Room.\*, guestName σ (hotelName = 'Grosvenor' ∧ dateFrom <= CURRENT\_DATE ∧ CURRENT\_DATE <= dateTo) ((Room ⋈ Hotel) (Booking ⋈ Guest))

**Chapter 6**

**6.7**

SELECT \* FROM Hotel;

**6.8**

SELECT \* FROM Hotel WHERE city = ‘London’;

**6.9**

SELECT guestName, guestAddress FROM Guest WHERE guestAddress LIKE '%London%' ORDER BY guestName ASC;

**6.10**

SELECT \* FROM Room WHERE price < 40 AND type IN ('family','double') ORDER BY price;

**6.11**

SELECT \* FROM Booking WHERE dateTo IS NULL;

**6.12**

SELECT COUNT(hotelNo) FROM Hotel;

**6.13**

SELECT AVG(price) FROM Room;

**6.14**

SELECT SUM(price) FROM Room WHERE type = 'double';

**6.15**

SELECT COUNT(DISTINCT guestNo) FROM Booking WHERE (dateFrom >= '01-08-2018' and dateTo <= '31-08-2018) OR (dateFrom <= '01-08-2018' and dateTo >= '01-08-2018) OR (dateFrom <= '31-08-2018' and dateTo >= '31-08-2018);

**6.16**

SELECT type, price FROM Room WHERE hotelNo = (SELECT hotelNo FROM Hotel WHERE hotelName = ‘Grosvenor’);

**6.17**

SELECT \* FROM Guest WHERE guestNo = (SELECT guestNo FROM Booking WHERE dateFrom <= CURRENT\_DATE AND dateTo >= CURRENT\_DATE AND hotelNo = (SELECT hotelNo FROM Hotel WHERE hotelName = 'Grosvenor'));

**6.18**

SELECT r.\* FROM Room r LEFT JOIN (SELECT g.guestName, h.hotelNo, b.roomNo FROM Guest g, Booking b, Hotel h WHERE g.guestNo = b.guestNo AND b.hotelNo = h.hotelNo AND hotelName= ‘Grosvenor Hotel’ AND dateFrom <= CURRENT\_DATE AND dateTo >= CURRENT\_DATE) AS newrel ON r.hotelNo = newrel.hotelNo AND r.roomNo = newrel.roomNo;

**6.19**

SELECT SUM(price) FROM Booking b, Room r, Hotel h WHERE (b.dateFrom <= CURRENT\_DATE AND b.dateTo >= CURRENT\_DATE) AND r.hotelNo = h.hotelNo AND r.roomNo = b.roomNo AND h.hotelName = ‘Grosvenor’;

**6.20**

SELECT r.\* FROM Room r, Hotel h WHERE r.hotelNo = h.hotelNo AND h.hotelName = 'Grosvenor' and r.roomNo NOT IN (SELECT roomNo FROM Booking b, Hotel h WHERE b.hotelNo = h.hotelNo AND hotelName = 'Grosvenor' AND (dateFrom <= CURRENT\_DATE AND dateTo >= CURRENT\_DATE));

**6.21**

SELECT SUM(price) FROM Room r WHERE roomNo NOT IN (SELECT roomNo FROM Booking b, Hotel h WHERE (dateFrom <= CURRENT\_DATE AND dateTo >= CURRENT\_DATE) AND b.hotelno = h.hotelno AND hotelname = 'Grosvenor');

**6.22**

SELECT hotelNo, COUNT(roomNo) AS count FROM Room GROUP BY hotelNo;

**6.23**

SELECT h.hotelNo, count(roomNo) AS count FROM Room r, Hotel h WHERE r.hotelNo = h.hotelNo AND city = 'London' GROUP BY r.hotelNo;

**6.24**

SELECT AVG(cnt) AS AvgNoBookings FROM (SELECT hotelNo, COUNT(hotelNo) AS cnt FROM Booking b WHERE (b.dateFrom >= '01-08-2018' AND b.dateFrom <= '31-08-2018') GROUB BY hotelNo) AS Relat;

**6.25**

SELECT MAX(cnt) AS MostBooked FROM (SELECT type, COUNT(type) AS cnt FROM Booking b, Hotel h, Room r WHERE r.roomNo = b.roomNo AND b.hotelNo = h.hotelNo AND h.city = 'London' GROUP BY type) AS Relat;

**6.26**

SELECT hotelNo, SUM(price) FROM Room r WHERE roomNo NOT IN (SELECT roomNo FROM Booking b, Hotel h WHERE (dateFrom <= CURRENT\_DATE AND dateTo >= CURRENT\_DATE) AND b.hotelNo = h.hotelNo) GROUP BY hotelNo;

**6.27**

INSERT INTO Hotel(hotelNo, hotelName, city)

VALUES ('H01', 'Grosvenor', 'London'),

('H02', 'Eurobar', 'Oxford'),

('H03', 'New Union', 'Menchester');

INSERT INTO Room(roomNo, hotelNo, type, price)

VALUES

('10', 'H01', 'single', '40'),

('12', 'H01', 'double', '65'),

('21', 'H01', 'family', '75'),

('33', 'H02', 'family', '75'),

('41', 'H02', 'double', '80'),

('16', 'H03', 'double', '50');

INSERT INTO Booking(hotelNo, guestNo, dateFrom, dateTo, roomNo)

VALUES

('H1', 'G03', '16-07-2018', '20-07-2018', '10'),

('H1', 'G01', '20-08-2018', '29-08-2018', '21'),

('H2', 'G02', '17-06-2018', '25-06-2018', '33'),

('H3', 'G04', '20-04-2018', '25-04-2018', '16'),

INSERT INTO Guest(guestNo, guestName, guestAddress)

VALUES

('G01', ' William Jensen ', 'Stefansgade 7 st.th. Roskilde'),

('G02', ' Emma Nielsen ', ' Allegade 63 Skanderborg '),

('G03', ' Mathias Hansen ', ' Ostergade 13 Fredericia '),

('G04', ' Tobias Pedersen ', ' Sundvej 44 Horsens ');

**6.28**

UPDATE Room SET price = price \* 21/20;

**Chapter 7**

**7.10**

CREATE DOMAIN hnumber char(2) NOT NULL CHECK(value between 'H01' and 'H99');

CREATE TABLE Hotel(

hotelNo hnumber,

hotelName varchar(15),

city varchar(15),

primary key(hotelNo));

**7.11**

CREATE DOMAIN rnumber int CHECK(value between 1 and 100);

CREATE DOMAIN threetype char(6) CHECK(value IN('single', 'double', 'family'));

CREATE DOMAIN pric smallint CHECK(value between '10' and '100');

CREATE TABLE Room(

roomNo rnumber,

hotelNo hnumber,

type threetype,

price pric,

CONSTRAINT PK primary key(roomNo, hotelNo),

foreign key(hotelNo) REFERENCES Hotel);

CREATE DOMAIN gnumber varchar(2) NOT NULL CHECK(value between 'G1' and 'G9');

CREATE DOMAIN dat DATE NOT NULL CHECK(value >= CURRENT\_DATE);

CREATE TABLE Booking(

hotelNo hnumber,

guestNo gnumber,

dateFrom dat,

dateTo dat,

roomNo rnumber,

primary key(hotelNo, guestNo, dateFrom),

foreign key(hotelNo) REFERENCES Hotel,

foreign key(roomNo,hotelNo) REFERENCES Room(roomNo,hotelNo),

foreign key(guestNo) REFERENCES Guest);

ALTER TABLE Booking ADD CONSTRAINT roombk CHECK(NOT EXISTS ( SELECT \* FROM Booking b WHERE b.roomNo == Booking.roomNo AND b.hotelNo == Booking.hotelNo AND b.dateFrom <= Booking.dateTo AND b.dateTo >= Booking.dateFrom));

CREATE TABLE Guest(

guestNo gnumber,

guestName varchar(40) NOT NULL,

guestAddress varchar(40) NOT NULL,

primary key(guestNo));

ALTER TABLE Booking ADD CONSTRAINT roombk CHECK(NOT EXISTS ( SELECT \* FROM Booking b WHERE b.roomNo == Booking.roomNo AND b.hotelNo == Booking.hotelNo AND b.dateFrom <= Booking.dateTo AND b.dateTo >= Booking.dateFrom));

**7.12**

CREATE TABLE BookingTwin(

hotelNo hnumber,

guestNo gnumber,

dateFrom dat,

dateTo dat,

roomNo rnumber,

primary key(hotelNo, guestNo, dateFrom),

foreign key(hotelNo) REFERENCES Hotel(hotelNo),

foreign key(roomNo,hotelNo) REFERENCES Room(roomNo, hotelNo),

foreign key(guestNo) REFERENCES Guest(guestNo));

INSERT INTO BookingTwin (hotelNo, guestNo, dateFrom, dateTo, roomNo)

(SELECT \* FROM Booking WHERE dateTo < '01-01-2007');

DELETE FROM Booking WHERE dateTo < '01-01-2017';