DATABASE SYSTEMS

GROUP 3

COURSEWORK 2
Implementation

Daniel Lorant Gelencser, Nasir Iqbal Sherwani, Mohamed Yusuf Rafiq, Mohammed Sadeq Rahman, Mohammad Imranoor Rahman

Contents

Relational Schema - From CWK 1

Formatting key: **PrimaryKey,** *ForeignKey*, **CompositeKey.**

4NF Relational Schema Diagram

-Customer(**customerID**, firstname, surname, phoneNumber, email)

-Booking(**bookingID**, arrivalDate, departureDate, numberOfAdults, numberOfChilds, *customerID, roomNo, floorNo, transactionID*)

-Transaction(**transactionID**, amount, currency, date, *customerID*)

-Room(**roomNo**, ***floorNo***, *facilityID*)

-Facility(**faciltyID**, *numberOfBeds*, miniFridge)

-NoB(**numberOfBeds**, bedType)

-Floor(**floorNo**, *roomType*)

-RoomPrice(**roomType**, basicPrice)

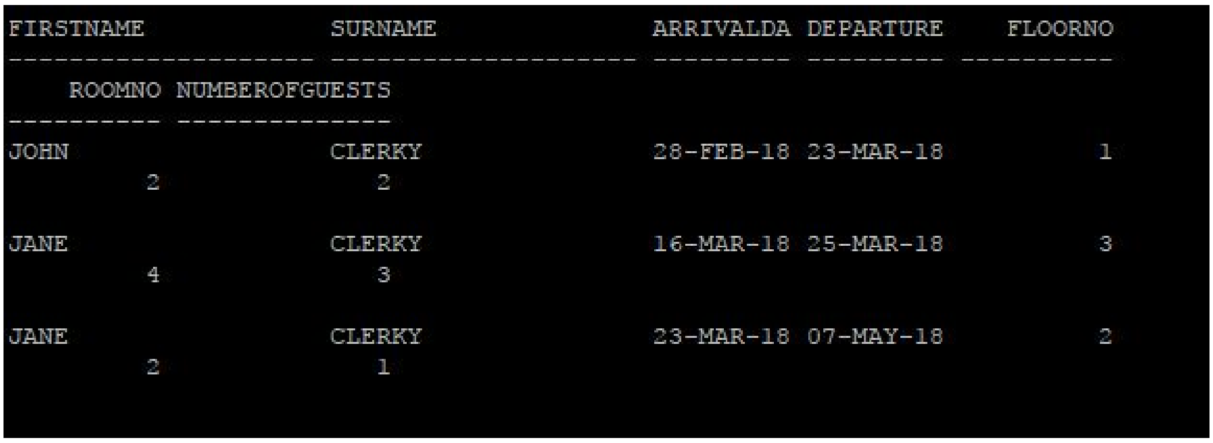
-Guest(**guestID**, firstname, surname, under16)

-GuestBooking(***bookingID, guestID***)

Create Views

1. View created for the Clerky family to view their bookings

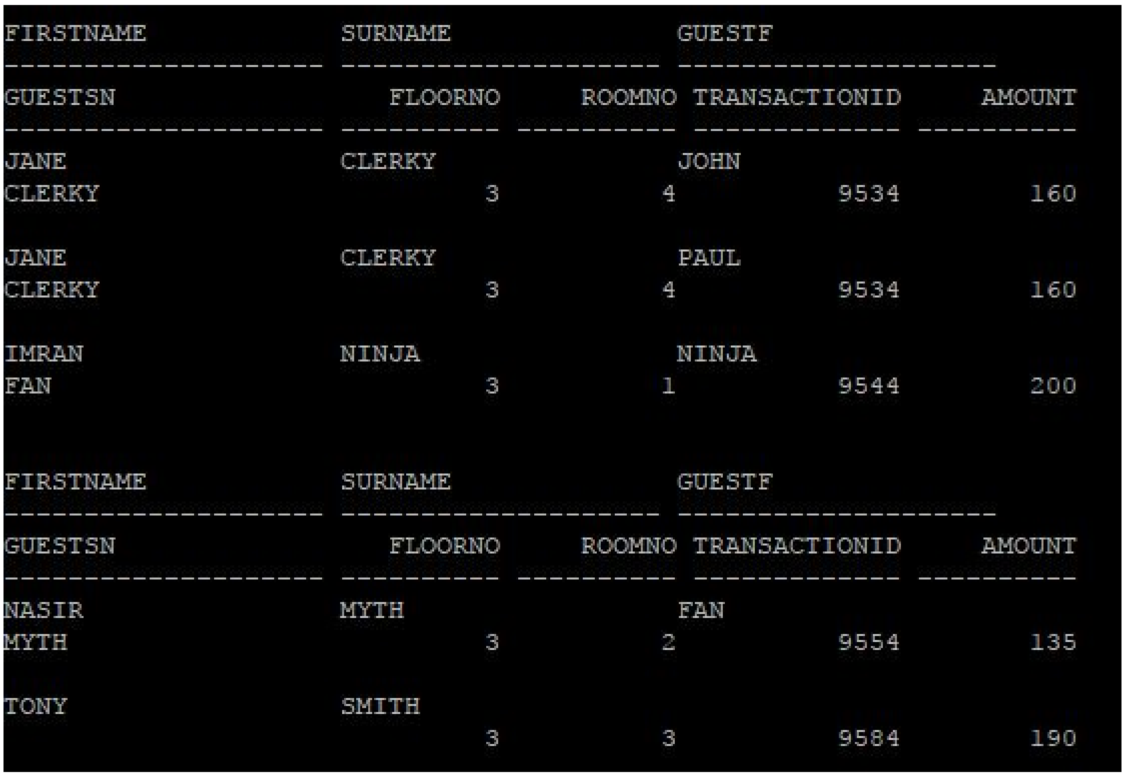
```
CREATE OR REPLACE VIEW CLERKYBOOKING AS
SELECT FIRSTNAME, SURNAME, ARRIVALDATE, DEPARTUREDATE, FLOORNO,
ROOMNO, (NUMBEROFADULTS+NUMBEROFCHILDS) AS NUMBEROFGUESTS
FROM BOOKING INNER JOIN CUSTOMER ON
CUSTOMER.CUSTOMERID=BOOKING.CUSTOMERID
WHERE CUSTOMER.CUSTOMERID IN ('7369','7379');
```



```
FIRSTNAME            SURNAME              ARRIVALDA DEPARTURE    FLOORNO
-------------------- -------------------- --------- --------- ----------
    ROOMNO NUMBEROFGUESTS
---------- --------------
JOHN                 CLERKY               28-FEB-18 23-MAR-18          1
         2              2

JANE                 CLERKY               16-MAR-18 25-MAR-18          3
         4              3

JANE                 CLERKY               23-MAR-18 07-MAY-18          2
         2              1
```

2. View created to check VIPS on the floor containing magnificent rooms, VIPS include customers and their guests

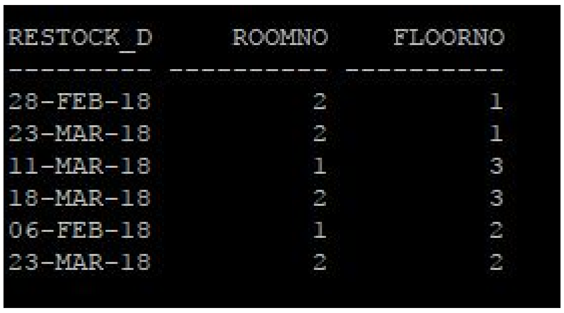
```
CREATE OR REPLACE VIEW VIPS AS SELECT CUSTOMER.FIRSTNAME,
CUSTOMER.SURNAME, GUEST.FIRSTNAME AS GUESTF, GUEST.SURNAME AS
GUESTSN, FLOORNO, ROOMNO, TRANSACTION.TRANSACTIONID,
TRANSACTION.AMOUNT FROM ((BOOKING
INNER JOIN CUSTOMER ON BOOKING.CUSTOMERID=CUSTOMER.CUSTOMERID
LEFT JOIN GUESTBOOKING ON BOOKING.BOOKINGID=GUESTBOOKING.BOOKINGID
LEFT JOIN GUEST ON GUEST.GUESTID=GUESTBOOKING.GUESTID
)INNER JOIN TRANSACTION ON BOOKING.TRANSACTIONID =
TRANSACTION.TRANSACTIONID )
WHERE BOOKING.FLOORNO = '3';
```



```
FIRSTNAME            SURNAME              GUESTF
-------------------- -------------------- --------------------
GUESTSN                 FLOORNO     ROOMNO TRANSACTIONID     AMOUNT
-------------------- ---------- ---------- ------------- ----------
JANE                 CLERKY               JOHN
CLERKY                        3          4          9534        160

JANE                 CLERKY               PAUL
CLERKY                        3          4          9534        160

IMRAN                NINJA                NINJA
FAN                           3          1          9544        200


FIRSTNAME            SURNAME              GUESTF
-------------------- -------------------- --------------------
GUESTSN                 FLOORNO     ROOMNO TRANSACTIONID     AMOUNT
-------------------- ---------- ---------- ------------- ----------
NASIR                MYTH                 FAN
MYTH                          3          2          9554        135

TONY                 SMITH
                              3          3          9584        190
```

3. View created to display the dates the that mini fridges need to be displayed by finding rooms that have the mini fridge facility and then checking to see when they have occupants arriving

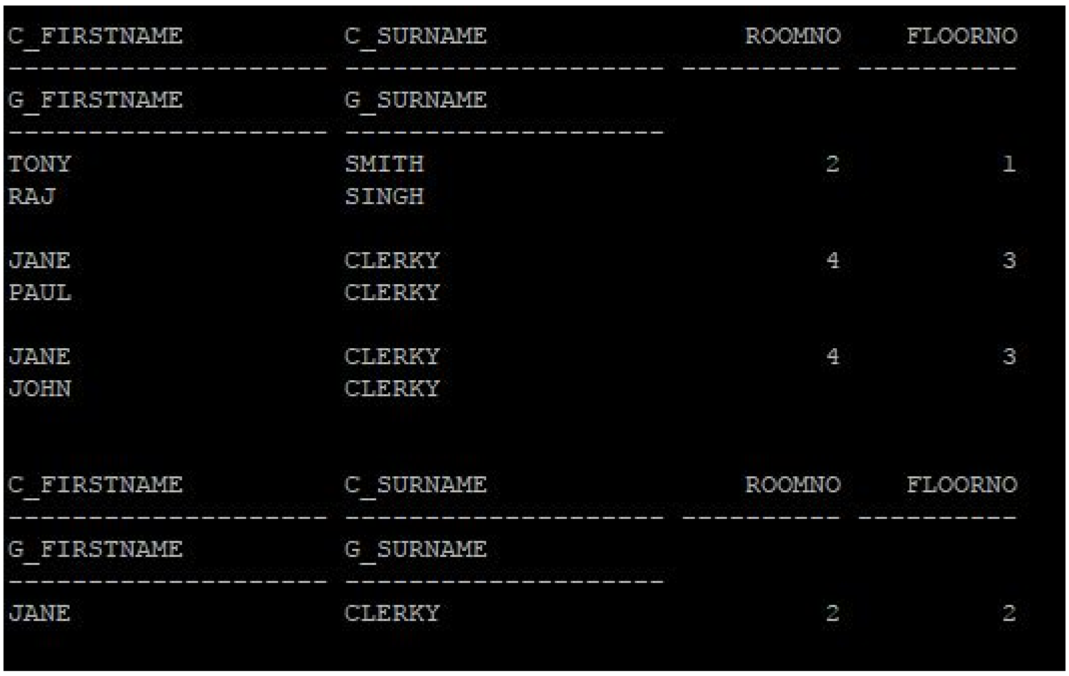
```
CREATE OR REPLACE VIEW STOCKMINIFRIDGE AS SELECT ARRIVALDATE AS
RESTOCK_DATE, BOOKING.ROOMNO, BOOKING.FLOORNO FROM BOOKING INNER
JOIN ROOM ON BOOKING.FLOORNO=ROOM.FLOORNO AND BOOKING.ROOMNO =
ROOM.ROOMNO INNER JOIN FACILITY ON ROOM.FACILITYID=FACILITY.FACILITYID
WHERE MINIFRIDGE='YES';
```



```
RESTOCK_D      ROOMNO    FLOORNO
--------- ---------- ----------
28-FEB-18          2          1
23-MAR-18          2          1
11-MAR-18          1          3
18-MAR-18          2          3
06-FEB-18          1          2
23-MAR-18          2          2
```

4. View created that contains all the rooms that are being used at the current date

```
CREATE OR REPLACE VIEW ROOMSUSED AS
SELECT CUSTOMER.FIRSTNAME AS C_FIRSTNAME, CUSTOMER.SURNAME AS
C_SURNAME, BOOKING.ROOMNO, BOOKING.FLOORNO, GUEST.FIRSTNAME AS
G_FIRSTNAME, GUEST.SURNAME AS G_SURNAME
FROM BOOKING
LEFT JOIN CUSTOMER ON BOOKING.CUSTOMERID=CUSTOMER.CUSTOMERID
LEFT JOIN GUESTBOOKING ON BOOKING.BOOKINGID= GUESTBOOKING.BOOKINGID
LEFT JOIN GUEST ON GUEST.GUESTID=GUESTBOOKING.GUESTID
WHERE ARRIVALDATE <= CURRENT_DATE AND DEPARTUREDATE >=
CURRENT_DATE;
```

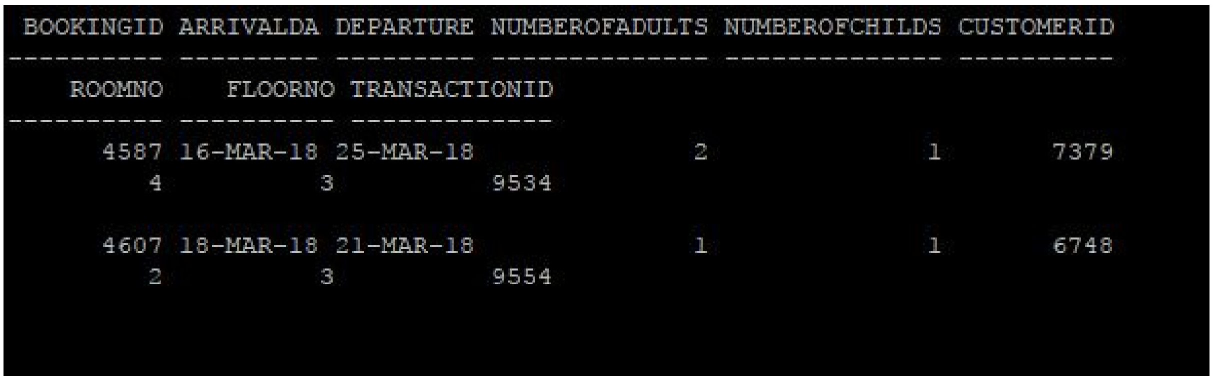


```
C_FIRSTNAME          C_SURNAME                ROOMNO    FLOORNO
-------------------- -------------------- ---------- ----------
G_FIRSTNAME          G_SURNAME
-------------------- --------------------
TONY                 SMITH                         2          1
RAJ                  SINGH

JANE                 CLERKY                        4          3
PAUL                 CLERKY

JANE                 CLERKY                        4          3
JOHN                 CLERKY


C_FIRSTNAME          C_SURNAME                ROOMNO    FLOORNO
-------------------- -------------------- ---------- ----------
G_FIRSTNAME          G_SURNAME
-------------------- --------------------
JANE                 CLERKY                        2          2
```

Canned Queries

1. Display all customers that arrived sometime last week. Can be used to go over records of bookings when checking for customers that may have damaged a room or perhaps left something behind.

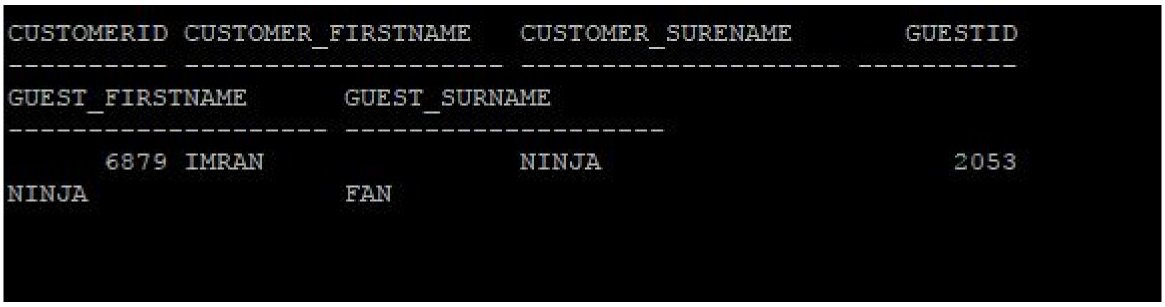
SELECT * FROM BOOKING where ARRIVALDATE >= next_day(trunc(sysdate), 'MONDAY') - 14 and ARRIVALDATE < next_day(trunc(sysdate), 'MONDAY') - 7;



```
 BOOKINGID ARRIVALDA DEPARTURE NUMBEROFADULTS NUMBEROFCHILDS CUSTOMERID
---------- --------- --------- -------------- -------------- ----------
    ROOMNO    FLOORNO TRANSACTIONID
---------- ---------- -------------
      4587 16-MAR-18 25-MAR-18              2              1       7379
         4          3          9534

      4607 18-MAR-18 21-MAR-18              1              1       6748
         2          3          9554
```

2. Display all characters and guests who's surnames start with a character, in this instance the character is N. All records that have N as the starting letter will be displayed no matter how many characters follow the letter because of the % wildcard character

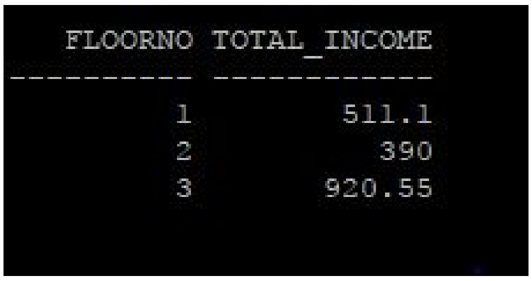
SELECT C.CUSTOMERID, C.FIRSTNAME AS CUSTOMER_FIRSTNAME, C.SURNAME AS CUSTOMER_SURENAME, GB.GUESTID, G.FIRSTNAME AS GUEST_FIRSTNAME, G.SURNAME AS GUEST_SURNAME FROM CUSTOMER C FULL OUTER JOIN BOOKING B ON C.CUSTOMERID = B.CUSTOMERID INNER JOIN GUESTBOOKING GB ON B.BOOKINGID = GB.BOOKINGID INNER JOIN GUEST G ON GB.GUESTID = G.GUESTID WHERE C.SURNAME LIKE 'N%';



```
CUSTOMERID CUSTOMER_FIRSTNAME   CUSTOMER_SURENAME       GUESTID
---------- -------------------- -------------------- ----------
GUEST_FIRSTNAME      GUEST_SURNAME
-------------------- --------------------
      6879 IMRAN                NINJA                      2053
NINJA                FAN
```

3. Displays the total income made by every floor of the hotel, can be used for accounting purposes.

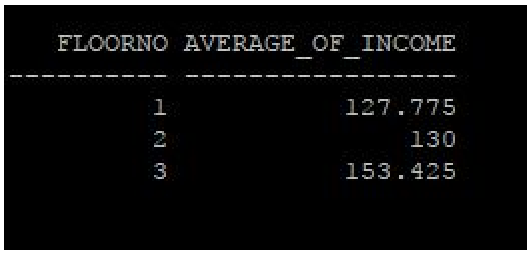
SELECT B.FLOORNO, SUM(T.AMOUNT) AS TOTAL_INCOME FROM TRANSACTION T INNER JOIN BOOKING B ON T.CUSTOMERID = B.CUSTOMERID INNER JOIN CUSTOMER C ON B.CUSTOMERID = C.CUSTOMERID WHERE B.FLOORNO = 1 OR B.FLOORNO = 2 OR B.FLOORNO = 3 GROUP BY B.FLOORNO;



```
   FLOORNO TOTAL_INCOME
---------- ------------
         1        511.1
         2          390
         3       920.55
```

4. Displays the average income made by each floor of the hotel so far, can be used to review business decisions or change services.

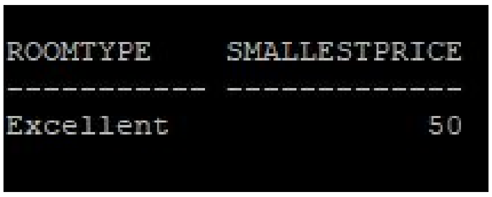
SELECT B.FLOORNO, AVG(T.AMOUNT) AS TOTAL_INCOME FROM TRANSACTION T INNER JOIN BOOKING B ON T.CUSTOMERID = B.CUSTOMERID INNER JOIN CUSTOMER C ON B.CUSTOMERID = C.CUSTOMERID WHERE B.FLOORNO = 1 OR B.FLOORNO = 2 OR B.FLOORNO = 3 GROUP BY B.FLOORNO;



```
   FLOORNO AVERAGE_OF_INCOME
---------- -----------------
         1           127.775
         2               130
         3           153.425
```

5. Display the cheapest room, can be used when searching for rooms based on the price attribute. This query focuses on displaying the least expensive room. The MIN function is implemented here for that specific purpose.

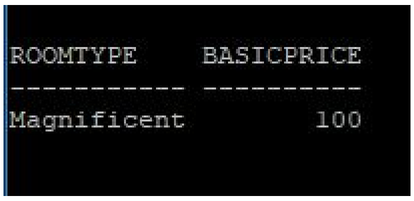
SELECT roomType, basicPrice AS SmallestPrice FROM ROOMPRICE WHERE basicPrice = (SELECT MIN(basicPrice) FROM ROOMPRICE);



```
ROOMTYPE    SMALLESTPRICE
----------- -------------
Excellent              50
```

6.This query focuses on displaying the most expensive room. The MAX function is implemented here for that specific purpose.

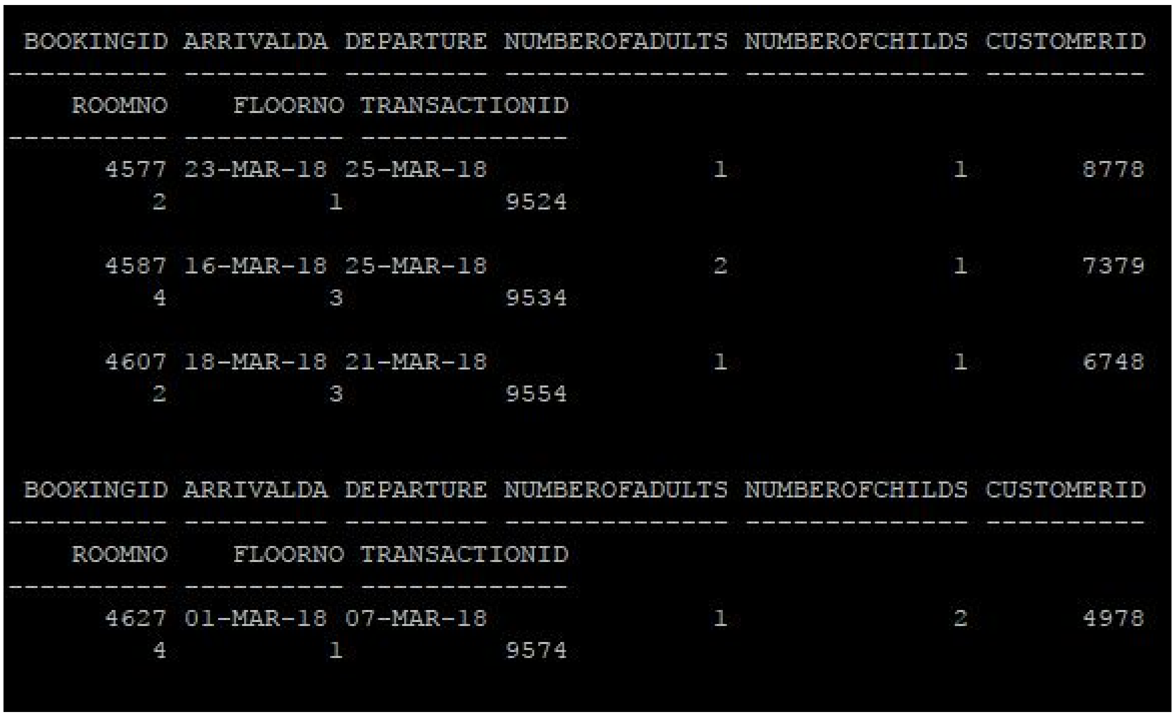
SELECT roomType, basicPrice FROM ROOMPRICE WHERE basicPrice = (SELECT MAX(basicPrice) FROM ROOMPRICE);



```
ROOMTYPE    BASICPRICE
----------- ----------
Magnificent        100
```

7.This query checks the booking table for customers that do not have any guests that are under 16 (children).

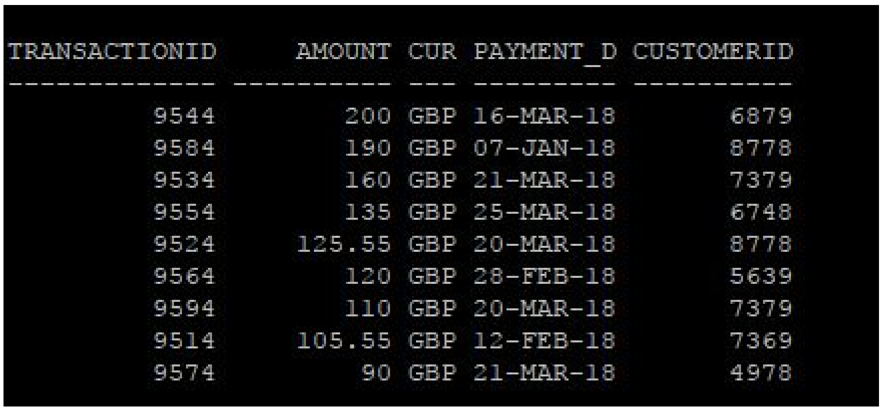
SELECT * FROM BOOKING WHERE NOT NUMBEROFCHILDS < 1;



```
 BOOKINGID ARRIVALDA DEPARTURE NUMBEROFADULTS NUMBEROFCHILDS CUSTOMERID
---------- --------- --------- -------------- -------------- ----------
    ROOMNO    FLOORNO TRANSACTIONID
---------- ---------- -------------
      4577 23-MAR-18 25-MAR-18              1              1       8778
         2          1          9524

      4587 16-MAR-18 25-MAR-18              2              1       7379
         4          3          9534

      4607 18-MAR-18 21-MAR-18              1              1       6748
         2          3          9554


 BOOKINGID ARRIVALDA DEPARTURE NUMBEROFADULTS NUMBEROFCHILDS CUSTOMERID
---------- --------- --------- -------------- -------------- ----------
    ROOMNO    FLOORNO TRANSACTIONID
---------- ---------- -------------
      4627 01-MAR-18 07-MAR-18              1              2       4978
         4          1          9574
```

8.This query will present all transactions in a descending order taking the amount into account.

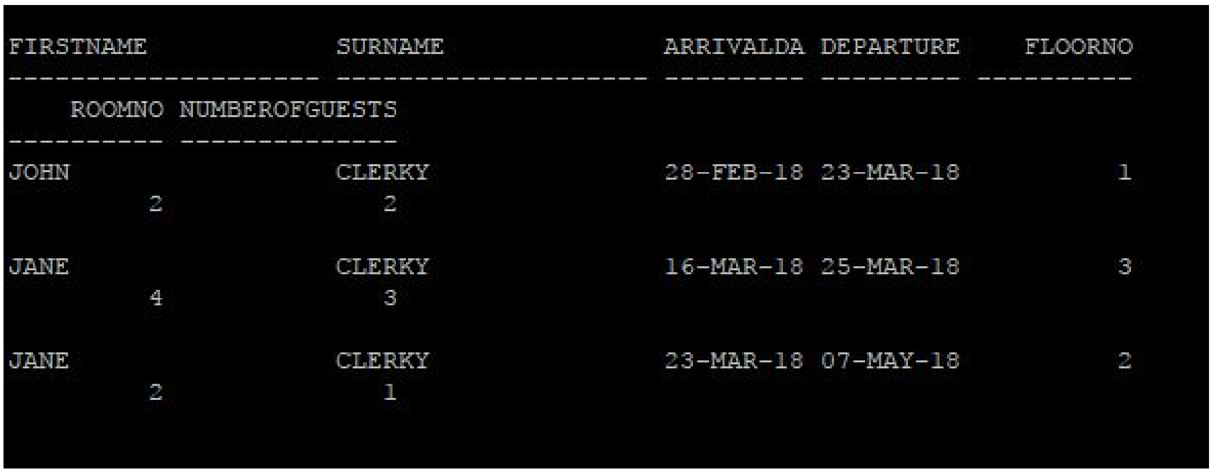
SELECT * FROM TRANSACTION ORDER BY AMOUNT DESC;



```
TRANSACTIONID     AMOUNT CUR PAYMENT_D CUSTOMERID
------------- ---------- --- --------- ----------
         9544        200 GBP 16-MAR-18       6879
         9584        190 GBP 07-JAN-18       8778
         9534        160 GBP 21-MAR-18       7379
         9554        135 GBP 25-MAR-18       6748
         9524     125.55 GBP 20-MAR-18       8778
         9564        120 GBP 28-FEB-18       5639
         9594        110 GBP 20-MAR-18       7379
         9514     105.55 GBP 12-FEB-18       7369
         9574         90 GBP 21-MAR-18       4978
```

9. This query will display all possible records from the clerky booking (View table).

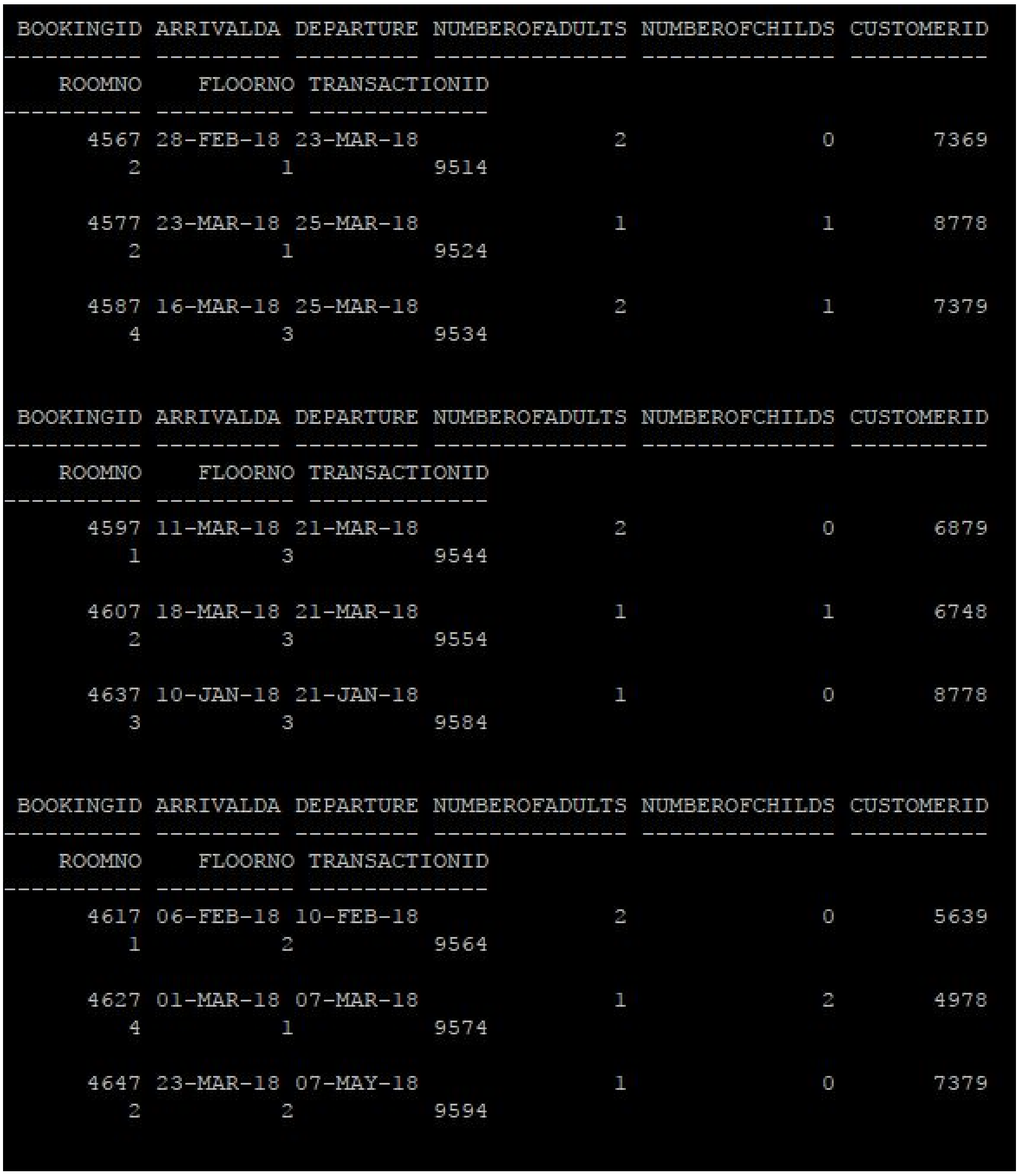
SELECT * FROM CLERKYBOOKING;



```
FIRSTNAME            SURNAME              ARRIVALDA DEPARTURE    FLOORNO
-------------------- -------------------- --------- --------- ----------
    ROOMNO NUMBEROFGUESTS
---------- --------------
JOHN                 CLERKY               28-FEB-18 23-MAR-18          1
         2              2

JANE                 CLERKY               16-MAR-18 25-MAR-18          3
         4              3

JANE                 CLERKY               23-MAR-18 07-MAY-18          2
         2              1
```

10. This query will display all booking records in the database.

SELECT * FROM BOOKING;



```
 BOOKINGID ARRIVALDA DEPARTURE NUMBEROFADULTS NUMBEROFCHILDS CUSTOMERID
---------- --------- --------- -------------- -------------- ----------
    ROOMNO    FLOORNO TRANSACTIONID
---------- ---------- -------------
      4567 28-FEB-18 23-MAR-18              2              0       7369
         2          1          9514

      4577 23-MAR-18 25-MAR-18              1              1       8778
         2          1          9524

      4587 16-MAR-18 25-MAR-18              2              1       7379
         4          3          9534


 BOOKINGID ARRIVALDA DEPARTURE NUMBEROFADULTS NUMBEROFCHILDS CUSTOMERID
---------- --------- --------- -------------- -------------- ----------
    ROOMNO    FLOORNO TRANSACTIONID
---------- ---------- -------------
      4597 11-MAR-18 21-MAR-18              2              0       6879
         1          3          9544

      4607 18-MAR-18 21-MAR-18              1              1       6748
         2          3          9554

      4637 10-JAN-18 21-JAN-18              1              0       8778
         3          3          9584


 BOOKINGID ARRIVALDA DEPARTURE NUMBEROFADULTS NUMBEROFCHILDS CUSTOMERID
---------- --------- --------- -------------- -------------- ----------
    ROOMNO    FLOORNO TRANSACTIONID
---------- ---------- -------------
      4617 06-FEB-18 10-FEB-18              2              0       5639
         1          2          9564

      4627 01-MAR-18 07-MAR-18              1              2       4978
         4          1          9574

      4647 23-MAR-18 07-MAY-18              1              0       7379
         2          2          9594
```

ALL CODE

```
SET TERMOUT ON
PROMPT Building demonstration tables.  Please wait.
SET TERMOUT OFF

DROP TABLE CUSTOMER cascade constraints;
DROP TABLE BOOKING cascade constraints;
DROP TABLE TRANSACTION cascade constraints;
DROP TABLE ROOM cascade constraints;
DROP TABLE FACILITY cascade constraints;
DROP TABLE NOB cascade constraints;
DROP TABLE FLOOR_ cascade constraints;
DROP TABLE ROOMPRICE cascade constraints;
DROP TABLE GUEST cascade constraints;
DROP TABLE GUESTBOOKING cascade constraints;

CREATE TABLE CUSTOMER
        (CUSTOMERID NUMBER(4) NOT NULL,
   FIRSTNAME VARCHAR(20),
   SURNAME VARCHAR(20),
   PHONENUMBER VARCHAR(11),
   EMAIL VARCHAR(30),
   CONSTRAINT PK_CUSTOMERID PRIMARY KEY (CUSTOMERID) );

CREATE TABLE BOOKING
        (BOOKINGID NUMBER(4),
        ARRIVALDATE DATE,
        DEPARTUREDATE DATE,
   NUMBEROFADULTS NUMBER(1),
   NUMBEROFCHILDS NUMBER(1),
   CUSTOMERID NUMBER(4) NOT NULL,
   ROOMNO NUMBER(4),
   FLOORNO NUMBER(2),
   TRANSACTIONID NUMBER(5),
   CONSTRAINT PK_BOOKINGID PRIMARY KEY (BOOKINGID)
    );

CREATE TABLE TRANSACTION(
   TRANSACTIONID NUMBER(5) NOT NULL,
   AMOUNT NUMBER(10,4),
   CURRENCY VARCHAR(3),
   PAYMENT_DATE DATE,
   CUSTOMERID NUMBER(4) NOT NULL,
   CONSTRAINT PK_TRANSACTIONID PRIMARY KEY (TRANSACTIONID),
   CONSTRAINT FK_CUSTOMERID FOREIGN KEY (CUSTOMERID) REFERENCES CUSTOMER(CUSTOMERID)
   );

CREATE TABLE ROOM(
   ROOMNO NUMBER(4) NOT NULL,
   FLOORNO NUMBER(2) NOT NULL,
   FACILITYID NUMBER(4) NOT NULL,
   CONSTRAINT PK_ROOMID PRIMARY KEY (ROOMNO, FLOORNO)
   );

CREATE TABLE FACILITY(
   FACILITYID NUMBER(4) NOT NULL PRIMARY KEY,
   NUMBEROFBEDS NUMBER(2),
   MINIFRIDGE VARCHAR(3) );

CREATE TABLE NOB
            (NUMBEROFBEDS NUMBER(2) NOT NULL PRIMARY KEY,
            BEDTYPE VARCHAR2(6));

CREATE TABLE FLOOR_(
   FLOORNO NUMBER(2) NOT NULL PRIMARY KEY,
   ROOMTYPE VARCHAR(11)
   );

CREATE TABLE ROOMPRICE
            (ROOMTYPE VARCHAR2(11) NOT NULL PRIMARY KEY,
            BASICPRICE NUMBER(5));

CREATE TABLE GUEST
            (GUESTID NUMBER(4) NOT NULL PRIMARY KEY,
            FIRSTNAME VARCHAR2(20),
             SURNAME VARCHAR2(20),
            UNDER16 VARCHAR2(1));

CREATE TABLE GUESTBOOKING
            (
   BOOKINGID NUMBER (4) NOT NULL,
   GUESTID NUMBER(4) NOT NULL,
   CONSTRAINT PK_GUESTBOOKINGID PRIMARY KEY (BOOKINGID, GUESTID),
   CONSTRAINT FK_BOOKINGID FOREIGN KEY (BOOKINGID) REFERENCES BOOKING(BOOKINGID),
   CONSTRAINT FK_GUESTID FOREIGN KEY (GUESTID) REFERENCES GUEST(GUESTID));

ALTER TABLE BOOKING ADD FOREIGN KEY (CUSTOMERID) REFERENCES CUSTOMER(CUSTOMERID);
ALTER TABLE BOOKING ADD FOREIGN KEY (ROOMNO, FLOORNO) REFERENCES ROOM(ROOMNO, FLOORNO);
ALTER TABLE BOOKING ADD FOREIGN KEY (FLOORNO) REFERENCES FLOOR_(FLOORNO);
ALTER TABLE BOOKING ADD FOREIGN KEY (TRANSACTIONID) REFERENCES TRANSACTION(TRANSACTIONID);

ALTER TABLE ROOM ADD FOREIGN KEY (FLOORNO) REFERENCES FLOOR_(FLOORNO);
ALTER TABLE ROOM ADD FOREIGN KEY (FACILITYID) REFERENCES FACILITY(FACILITYID);

ALTER TABLE FLOOR_ ADD FOREIGN KEY (ROOMTYPE) REFERENCES ROOMPRICE(ROOMTYPE);
ALTER TABLE FACILITY ADD FOREIGN KEY (NUMBEROFBEDS) REFERENCES NOB(NUMBEROFBEDS);


INSERT INTO CUSTOMER VALUES
            (7369, 'JOHN',  'CLERKY',    '07444444444', 'example@domain.com');
```

```
INSERT INTO CUSTOMER VALUES
        (7379, 'JANE',  'CLERKY',     '07333333333', 'example2@domain.com');
INSERT INTO CUSTOMER VALUES
        (8778, 'TONY',  'SMITH',      '07545454545', 'thirdtothirtyword@domain.co.uk');
INSERT INTO CUSTOMER VALUES
        (6879, 'IMRAN',  'NINJA',     '07456861235', 'ninjaisimran@gmail.com');
INSERT INTO CUSTOMER VALUES
        (6748, 'NASIR',  'MYTH',      '07456845626', 'nasirismyth@gmail.com');
INSERT INTO CUSTOMER VALUES
        (5639, 'SADEQ',  'DAEQUAN',           '07457984212', 'sadeqisdaequan@gmail.com');
INSERT INTO CUSTOMER VALUES
        (4978, 'YUSUF',  'DRAKE',     '07111437841', 'yusufisDrake@outlook.com');

INSERT INTO ROOMPRICE VALUES
   ('Excellent', 50);
INSERT INTO ROOMPRICE VALUES
   ('Deluxe', 75);
INSERT INTO ROOMPRICE VALUES
   ('Magnificent', 100);

INSERT INTO FLOOR_ VALUES
   (1, 'Excellent');
INSERT INTO FLOOR_ VALUES
   (2, 'Deluxe');
INSERT INTO FLOOR_ VALUES
   (3, 'Magnificent');

INSERT INTO NOB VALUES (1, 'DOUBLE');
INSERT INTO NOB VALUES (2, 'SINGLE');
INSERT INTO NOB VALUES (3, 'SINGLE');

INSERT INTO FACILITY VALUES
   (4785, 1, 'YES');
INSERT INTO FACILITY VALUES
   (4786, 2, 'YES');
INSERT INTO FACILITY VALUES
   (4787, 2, 'NO');
INSERT INTO FACILITY VALUES
   (4788, 3, 'NO');

INSERT INTO ROOM VALUES
   (01, 1, 4785);
INSERT INTO ROOM VALUES
   (02, 1, 4786);
INSERT INTO ROOM VALUES
   (03, 1, 4787);
INSERT INTO ROOM VALUES
   (04, 1, 4788);

INSERT INTO ROOM VALUES
   (01, 2, 4785);
```

```
INSERT INTO ROOM VALUES
   (02, 2, 4786);
INSERT INTO ROOM VALUES
   (03, 2, 4787);
INSERT INTO ROOM VALUES
   (04, 2, 4788);

INSERT INTO ROOM VALUES
   (01, 3, 4785);
INSERT INTO ROOM VALUES
   (02, 3, 4786);
INSERT INTO ROOM VALUES
   (03, 3, 4787);
INSERT INTO ROOM VALUES
   (04, 3, 4788);

INSERT INTO TRANSACTION VALUES (9514, 105.55, 'GBP', TO_DATE('12/2/2018', 'DD-MM-YY'), 7369);
INSERT INTO TRANSACTION VALUES (9524, 125.55, 'GBP', TO_DATE('20/3/2018', 'DD-MM-YY'), 8778);
INSERT INTO TRANSACTION VALUES (9534, 160.00, 'GBP', TO_DATE('21/3/2018', 'DD-MM-YY'), 7379);
INSERT INTO TRANSACTION VALUES (9544, 200.00, 'GBP', TO_DATE('16/3/2018', 'DD-MM-YY'), 6879);
INSERT INTO TRANSACTION VALUES (9554, 135.00, 'GBP', TO_DATE('25/3/2018', 'DD-MM-YY'), 6748);
INSERT INTO TRANSACTION VALUES (9564, 120.00, 'GBP', TO_DATE('28/2/2018', 'DD-MM-YY'), 5639);
INSERT INTO TRANSACTION VALUES (9574, 90.00, 'GBP', TO_DATE('21/3/2018', 'DD-MM-YY'), 4978);
INSERT INTO TRANSACTION VALUES (9584, 190.00, 'GBP', TO_DATE('07/1/2018', 'DD-MM-YY'), 8778);
INSERT INTO TRANSACTION VALUES (9594, 110.00, 'GBP', TO_DATE('20/3/2018', 'DD-MM-YY'), 7379);

INSERT INTO BOOKING VALUES (4567, TO_DATE('28/2/2018', 'DD-MM-YY'), TO_DATE('23/3/2018',
'DD-MM-YY'), 2, 0, 7369, 02, 1, 9514);
INSERT INTO BOOKING VALUES (4577, TO_DATE('23/3/2018', 'DD-MM-YY'), TO_DATE('25/3/2018',
'DD-MM-YY'), 1, 1, 8778, 02, 1, 9524);
INSERT INTO BOOKING VALUES (4587, TO_DATE('16/3/2018', 'DD-MM-YY'), TO_DATE('25/3/2018',
'DD-MM-YY'), 2, 1, 7379, 04, 3, 9534);
INSERT INTO BOOKING VALUES (4597, TO_DATE('11/3/2018', 'DD-MM-YY'), TO_DATE('21/3/2018',
'DD-MM-YY'), 2, 0, 6879, 01, 3, 9544);
INSERT INTO BOOKING VALUES (4607, TO_DATE('18/3/2018', 'DD-MM-YY'), TO_DATE('21/3/2018',
'DD-MM-YY'), 1, 1, 6748, 02, 3, 9554);
INSERT INTO BOOKING VALUES (4637, TO_DATE('10/1/2018', 'DD-MM-YY'), TO_DATE('21/1/2018',
'DD-MM-YY'), 1, 0, 8778, 03, 3, 9584);
INSERT INTO BOOKING VALUES (4617, TO_DATE('06/2/2018', 'DD-MM-YY'), TO_DATE('10/2/2018',
'DD-MM-YY'), 2, 0, 5639, 01, 2, 9564);
INSERT INTO BOOKING VALUES (4627, TO_DATE('01/3/2018', 'DD-MM-YY'), TO_DATE('07/3/2018',
'DD-MM-YY'), 1, 2, 4978, 04, 1, 9574);
INSERT INTO BOOKING VALUES (4647, TO_DATE('23/3/2018', 'DD-MM-YY'), TO_DATE('07/5/2018',
'DD-MM-YY'), 1, 0, 7379, 02, 2, 9594);

INSERT INTO GUEST VALUES
   (2013, 'JANE', 'CLERKY', 'N');
INSERT INTO GUEST VALUES
   (2023, 'RAJ', 'SINGH', 'Y');
INSERT INTO GUEST VALUES
   (2033, 'PAUL', 'CLERKY', 'Y');
```

```
INSERT INTO GUEST VALUES
   (2043, 'JOHN', 'CLERKY', 'N');
INSERT INTO GUEST VALUES
   (2053, 'NINJA', 'FAN', 'N');
INSERT INTO GUEST VALUES
   (2063, 'FAN', 'MYTH', 'Y');
INSERT INTO GUEST VALUES
   (2073, 'DAEQUAN', 'FAN', 'N');
INSERT INTO GUEST VALUES
   (2083, 'EASY', 'NOOB', 'Y');
INSERT INTO GUEST VALUES
   (2093, 'SCRUB', 'NOOB', 'Y');

INSERT INTO GUESTBOOKING VALUES
   (4567, 2013);
INSERT INTO GUESTBOOKING VALUES
   (4577, 2023);
INSERT INTO GUESTBOOKING VALUES
   (4587, 2033);
INSERT INTO GUESTBOOKING VALUES
   (4587, 2043);
INSERT INTO GUESTBOOKING VALUES
   (4597, 2053);
INSERT INTO GUESTBOOKING VALUES
   (4607, 2063);
INSERT INTO GUESTBOOKING VALUES
   (4617, 2073);
INSERT INTO GUESTBOOKING VALUES
   (4627, 2083);
INSERT INTO GUESTBOOKING VALUES
   (4627, 2093);

COMMIT;
--**VIEWS**--NEED 4
--1.[view definitions]Create a view for the clerky family that lists all their bookings
CREATE OR REPLACE VIEW CLERKYBOOKING AS
SELECT FIRSTNAME, SURNAME, ARRIVALDATE, DEPARTUREDATE, FLOORNO, ROOMNO,
(NUMBEROFADULTS+NUMBEROFCHILDS) AS NUMBEROFGUESTS
FROM BOOKING INNER JOIN CUSTOMER ON CUSTOMER.CUSTOMERID=BOOKING.CUSTOMERID
WHERE CUSTOMER.CUSTOMERID IN ('7369','7379');


--2.[view definitions]Display a list of all the customers and guests staying in the magnificent rooms ADN HOW
MUCH THEY PAID aka VIPS
CREATE OR REPLACE VIEW VIPS AS SELECT CUSTOMER.FIRSTNAME, CUSTOMER.SURNAME, GUEST.FIRSTNAME
AS GUESTF, GUEST.SURNAME AS GUESTSN, FLOORNO, ROOMNO, TRANSACTION.TRANSACTIONID,
TRANSACTION.AMOUNT FROM ((BOOKING
INNER JOIN CUSTOMER ON BOOKING.CUSTOMERID=CUSTOMER.CUSTOMERID
LEFT JOIN GUESTBOOKING ON BOOKING.BOOKINGID=GUESTBOOKING.BOOKINGID
LEFT JOIN GUEST ON GUEST.GUESTID=GUESTBOOKING.GUESTID
)INNER JOIN TRANSACTION ON BOOKING.TRANSACTIONID = TRANSACTION.TRANSACTIONID )
```

```
WHERE BOOKING.FLOORNO = '3';


--3.[view definitions]Display a list of the dates that rooms with minifridges must be restocked
CREATE OR REPLACE VIEW STOCKMINIFRIDGE AS SELECT ARRIVALDATE AS RESTOCK_DATE,
BOOKING.ROOMNO, BOOKING.FLOORNO FROM BOOKING INNER JOIN ROOM ON
BOOKING.FLOORNO=ROOM.FLOORNO AND BOOKING.ROOMNO = ROOM.ROOMNO INNER JOIN FACILITY ON
ROOM.FACILITYID=FACILITY.FACILITYID WHERE MINIFRIDGE='YES';

--4.[view definitions]Display a list of the dates that customers have departed, so that cleaners would know
when, which rooms and how many beds to clean.
CREATE OR REPLACE VIEW ROOMSUSED AS
SELECT CUSTOMER.FIRSTNAME AS C_FIRSTNAME, CUSTOMER.SURNAME AS C_SURNAME,
BOOKING.ROOMNO, BOOKING.FLOORNO, GUEST.FIRSTNAME AS G_FIRSTNAME, GUEST.SURNAME AS
G_SURNAME
FROM BOOKING
LEFT JOIN CUSTOMER ON BOOKING.CUSTOMERID=CUSTOMER.CUSTOMERID
LEFT JOIN GUESTBOOKING ON BOOKING.BOOKINGID= GUESTBOOKING.BOOKINGID
LEFT JOIN GUEST ON GUEST.GUESTID=GUESTBOOKING.GUESTID
WHERE ARRIVALDATE <= CURRENT_DATE AND DEPARTUREDATE >= CURRENT_DATE;


--**CANNED** --NEED 12
--1.DISPLAY CUSTOMERS WHO'S ARRIVAL DATE GOES FAR BACK AS LAST WEEK
SELECT * FROM BOOKING where ARRIVALDATE >= next_day(trunc(sysdate), 'MONDAY') - 14 and ARRIVALDATE
< next_day(trunc(sysdate), 'MONDAY') - 7;

--2.DISPLAY ALL CUSTOMERS AND GUESTS WHOSE SURNAME STARTS WITH THE LETTER N. --LETTER
SELECT C.CUSTOMERID, C.FIRSTNAME AS CUSTOMER_FIRSTNAME, C.SURNAME AS CUSTOMER_SURENAME,
GB.GUESTID, G.FIRSTNAME AS GUEST_FIRSTNAME, G.SURNAME AS GUEST_SURNAME FROM CUSTOMER C
FULL OUTER JOIN BOOKING B ON C.CUSTOMERID = B.CUSTOMERID INNER JOIN GUESTBOOKING GB ON
B.BOOKINGID = GB.BOOKINGID INNER JOIN GUEST G ON GB.GUESTID = G.GUESTID WHERE C.SURNAME LIKE
'N%';

--3.DISPLAY TOTAL INCOME SO FAR OF ALL BOOKING TRANSACTIONS FROM A SPECIFIC FLOOR ALONG WITH
THE FLOOR NUMBER.
SELECT B.FLOORNO, SUM(T.AMOUNT) AS TOTAL_INCOME FROM TRANSACTION T INNER JOIN BOOKING B ON
T.CUSTOMERID = B.CUSTOMERID INNER JOIN CUSTOMER C ON B.CUSTOMERID = C.CUSTOMERID WHERE
B.FLOORNO = 1 OR B.FLOORNO = 2 OR B.FLOORNO = 3 GROUP BY B.FLOORNO;

--4.Displays the average income made by each floor of the hotel so far, can be used to review business
decisions or change services.
SELECT B.FLOORNO, AVG(T.AMOUNT) AS AVERAGE_OF_INCOME FROM TRANSACTION T INNER JOIN BOOKING
B ON T.CUSTOMERID = B.CUSTOMERID INNER JOIN CUSTOMER C ON B.CUSTOMERID = C.CUSTOMERID WHERE
B.FLOORNO = 1 OR B.FLOORNO = 2 OR B.FLOORNO = 3 GROUP BY B.FLOORNO;

--5.Display the cheapest room, can be used when searching for rooms based on the price attribute. This query
focuses on displaying the least expensive room. The MIN function is implemented here for that specific
purpose.
SELECT roomType, basicPrice AS SmallestPrice FROM ROOMPRICE WHERE basicPrice = (SELECT MIN(basicPrice)
FROM ROOMPRICE);
```

```
--6.This query focuses on displaying the most expensive room. The MAX function is implemented here for that specific purpose.
SELECT roomType, basicPrice FROM ROOMPRICE WHERE basicPrice = (SELECT MAX(basicPrice) FROM ROOMPRICE);

----7. This query checks the booking table for customers that do not have any guests that are under 16 (children).
SELECT * FROM BOOKING WHERE NOT NUMBEROFCHILDS < 1;


--8.This query will present all transactions in a descending order taking the amount into account.
SELECT * FROM TRANSACTION ORDER BY AMOUNT DESC;

--9.This query will display all possible records from the clerky booking (View table)
SELECT * FROM CLERKYBOOKING;

--10. This query will display all records in the database
SELECT * FROM BOOKING;



--11
SET TERMOUT ON
PROMPT Demonstration table build is complete.
```