

Relational Schema

We changed our relational schema from the first coursework based on the feedback we had received. Below is the updated relational schema which makes more sense for our design.

Bookings

bookingID	customerID	startDate	duration	totalPrice
------------------	-------------------	-----------	----------	------------

RoomBookings

bookingID	roomNumber	price
------------------	-------------------	-------

GuestRooms

bookingID	roomNumber	guestID
------------------	-------------------	----------------

Customers

customerID	firstName	lastName	dob	contactNumber	contactEmail
-------------------	-----------	----------	-----	---------------	--------------

CustomerAddresses

customerID	doorNumber	streetName	postCode
-------------------	------------	------------	----------

Guests

guestID	firstName	lastName	dob
----------------	-----------	----------	-----

Rooms

roomNumber	roomType	roomFloor
-------------------	-----------------	-----------

RoomPrices

roomType	basePrice
-----------------	-----------

CREATETABLES, SAMPLE TEST DATA AND CREATEVIEWS FILES ARE ATTACHED WITHIN THE .ZIP. SEE THESE FILES TO SEE COMMANDS FOR CREATETABLE, INSERTED TEST DATA AND CREATEVIEW COMMANDS. DO NOT IGNORE.

View Definitions

Customers Who Are Guests:

This creates a view of all the customers who are guests which means they will be staying in the hotel with their guests. This is useful as you can see which customers are actually staying in the hotel because many customers may pay for the booking but not actually stay.

```
CREATE VIEW CustomersWhoAreGuests AS
SELECT t1.CustomerID, t2.GuestID, t1.FirstName, t1.LastName, t1.DoB
FROM customers t1
INNER JOIN guests t2 ON t2.FirstName=t1.FirstName AND t2.LastName=t1.LastName AND
t2.DoB=t1.DoB;
```

CUSTOMERID	GUESTID	FIRSTNAME	LASTNAME	DOB
4517	99999	Hamilton	Bentley	08-NOV-31
6448	99998	Dai	Gordon	18-NOV-57
4932	99997	Kiona	Waller	08-NOV-57
4619	99996	Stuart	Holcomb	16-JUL-70
1449	99995	Coby	Ewing	16-OCT-37

Children Who Are Guests:

This creates a view of all the guests who are children under the age of 12. This is useful as the children activity coordinator for the hotel could see how many children there and which guests are children so they can organise activities and set up adequate facilities for them if needed.

```
CREATE VIEW Children AS
SELECT t1.GuestID, t1.FirstName,
TRUNC(MONTHS_BETWEEN(sysdate,t1.DoB)/12)
Age
FROM guests t1
INNER JOIN guestrooms t2 ON
t2.GuestID=t1.GuestID
WHERE
TRUNC(MONTHS_BETWEEN(sysdate,t1.DoB)/12)
<= 12;
```

GUESTID	FIRSTNAME	AGE
92790	Jorden	0
54850	Jason	0
33471	Brett	5
65622	Lisandra	3
28113	Aubrey	2
39619	Ronan	11
11865	Farrah	11
85150	Felix	1
31413	Tyrone	0
43189	Jacob	3
10702	Edan	10
GUESTID	FIRSTNAME	AGE
57730	Trevor	6
45712	Iris	4
87796	Callum	7
12663	Jamal	6
79930	Coby	9
93919	Odessa	5
86118	Daryl	6
97016	Harper	11
18222	Brennan	4
94764	Demetrius	0
92429	Lyle	7
GUESTID	FIRSTNAME	AGE
14376	Sierra	7
50578	Cheyenne	1
53386	Kenyon	6
35604	Shannon	3

Average Duration of Stay According to Room Type:

This view calculates the average duration of stay for bookings according to room type. This is helpful to the business advisors of the hotel as it allows them to gather information about how long people usually stay in a room and room type. They can then adjust their offers to encourage more people to book a certain type of room.

ROOMTYPE	AVGDUR
-----	-----
Excellent	15
Deluxe	15
Magnificent	16

```
CREATE VIEW RoomTypeAvgDur AS
SELECT DISTINCT t1.RoomType, ROUND(AVG(t2.Duration),0) AS AVGDUR
FROM rooms t1
INNER JOIN roombookings t3 ON t3.RoomNumber=t1.RoomNumber
INNER JOIN bookings t2 ON t2.BookingID=t3.BookingID
GROUP BY t1.RoomType;
```

Total Price of Bookings:

This view gives the total price of all bookings made by a customer and lists their name and contact emails. This is useful because they can be contacted via their email with the booking confirmation and price paid. This means all customers can be notified of their purchases.

```
CREATE VIEW BookingConfirmations AS
SELECT t1.CustomerID, t1.FirstName, t1.LastName, t1.ContactEmail, t2.TotalPrice
FROM customers t1
INNER JOIN bookings t2 ON t2.CustomerID = t1.CustomerID;
```

CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
4517 Hamilton	Bentley		9000
nec@orciinconsequat.com			
5927 Shana	Pena		500
enim.consequat@nunc.org			
8397 Hilel	Floyd		800
mollis.Duis@mitemporlorem.ca			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
7213 Zahir	Preston		2300
eleifend.vitae.erat@magnaPraesent.ca			
3351 Yuri	McFarland		3000
commodo@justo.net			
4481 Gabriel	French		750
a.scelerisque@Pellentesqueultrices.co.uk			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
6235 Sawyer	Trevino		500
Nunc.quis.arcu@molestieSed.com			
1843 Hayden	Owens		700
a@at.edu			
8367 Dalton	Flowers		900
egestas.Aliquam@enimcommodo.edu			

CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
6559 Felix	Donovan		700
placerat.augue.Sed@Morbisit.ca			
9578 Dante	Hill		1300
a@metuseuerat.com			
1280 Zenaida	Knowles		4800
mauris.aliquam.eu@morbistristiqueSenectus.co.uk			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
6908 Noel	Madden		400
nunc.ac.mattis@bibendum.ca			
9378 Velma	Lyons		4600
vel@massa.com			
8637 Chiquita	Blanchard		1800
ornare.Fusce.mollis@iaculisnec.ca			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
7788 Kaden	Boyle		5000
tempor@vitaaeliquet.ca			
8401 Linda	Simon		850
iaculis.odio@velitAliquam.org			
3410 Brynn	Dodson		350
scelerisque@loremluctusut.co.uk			

CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
7943 Elaine	Juarez		2100
eu.accumsan.sed@egestas.co.uk			
7670 Murphy	McDonald		2300
ac@vestibulummassarutrum.net			
3189 Kenyon	Villarreal		4400
faucibus.orci.luctus@quamquis.co.uk			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
9888 Peter	Kramer		1400
convallis@ac.co.uk			
1998 Reece	Manning		350
enim@parturientmontesnascetur.co.uk			
9468 Jared	Wiggins		250
gravida.nunc.sed@Cum.co.uk			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
5027 Kermit	Dodson		200
ac.turpis.egestas@nislQuisque.ca			
3960 Dennis	Holcomb		200
cursus@Sed.co.uk			
4985 Wade	Franklin		3200
gravida.molestie@duiCum.org			

CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
4292 Summer	Griffin		1600
augue.id.ante@Suspendissesagittis.edu			
9010 Thane	Saunders		7200
arcu@milaciniamattis.edu			
3389 Quinlan	Hebert		1500
ante.Maecenas@ultrices.edu			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
5939 Darryl	Gay		2400
scelerisque.neque@ornarelectus.ca			
9194 Jin	Gonzales		1200
consequat.enim@In.co.uk			
9028 Kimberley	Dorsey		1800
pretium.neque.Morbi@Donec.com			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
8168 Phelan	Britt		2800
mi.Duis@quismassaMauris.net			
4084 Akeem	Brewer		400
eros@ornareFuscemollis.edu			
4227 Chloe	Serrano		3800
posuere.cubilia@nectellus.org			

CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
3111 Noah	Harrington		5000
ante@ullamcorper magnaSed.edu			
8937 Yardley	Mendez		2000
Nullam@enimMauris.com			
4723 Nolan	Mendez		4400
molestie.in.tempus@cursus.in.com			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
5203 Heather	Haney		600
sed@sociis natoque penatibus.edu			
5002 Ciaran	Tillman		4400
In.condimentum@urna.ca			
6091 Leah	Noble		5200
sagittis.felis.Donec@turpis.com			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
6382 August	Browning		2500
malesuada@Sedpharetra felis.co.uk			
2128 Chelsea	Diaz		200
orci@ipsum Suspendisse.ca			
7605 Celeste	Weaver		9600
tellus.lorem@vitaesodales.net			

CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
3560 Cody	McMahon		3400
arcu.eu@elitpellesquea.net			
2270 Blaze	Webb		5200
dui@luctus.org			
6259 Shay	Zamora		1200
ultrices@fermentumconvallisligula.com			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
1819 Gage	McKenzie		4000
vulputate.lacus@Aliquam.edu			
1028 Whilemina	Barton		2000
sem.semper@erat.ca			
3667 Declan	Crawford		8000
tortor@Donecconsectetuer.net			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
4643 Gabriel	Ryan		1200
eleifend.Cras.sed@inceptoshymenaeosMauris.co.uk			
3896 Flynn	Battle		3600
dui.quis@lectusa.org			
3113 Shelley	Stevenson		4050
a@odioAliquamvulputate.co.uk			

CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
1000 Brent	Fitzpatrick		450
iaculis.lacus@Donec.edu			
3449 Maggy	Hall		3150
enim.Curabitur.massa@acipsum.co.uk			
8489 Seth	Richardson		2600
aliquam@adipiscing elit.ca			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
2729 Rogan	Diaz		5400
Donec.est@Nullamlobortisquam.org			
4106 Deborah	Mays		1350
arcu@molestietellusAenean.ca			
8624 Molly	Brock		200
lacinia.Sed.congue@semper.net			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
3772 Montana	Petty		8100
Proin.non@leoVivamusnibh.edu			
6497 Maxwell	Odonnell		2400
odio.Etiam.ligula@vestibulumneceuismod.edu			
1449 Coby	Ewing		20800
Cras.lorem.lorem@ligulaAeneaneuismod.org			

CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
3772 Montana	Petty		8100
Proin.non@leoVivamusnibh.edu			
6497 Maxwell	Odonnell		2400
odio.Etiam.ligula@vestibulumneceuismod.edu			
1449 Coby	Ewing		20800
Cras.lorem.lorem@ligulaAeneaneuismod.org			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
6671 Charissa	Mendez		19600
est@tempor.com			
9318 Bevis	McDowell		6000
diam.Pellentesque@Donec.org			
1057 Kasper	Burgess		20000
Nunc.commodo@maurisrhoncus.edu			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
1008 Lester	Randall		21550
massa.rutrum.magna@Craspellentesque.org			
4619 Stuart	Holcomb		8800
eu.odio.tristique@nonleoVivamus.edu			
4932 Kiona	Waller		3500
Quisque.ornare.tortor@nascetur.ca			
CUSTOMERID	FIRSTNAME	LASTNAME	
CONTACTEMAIL			TOTALPRICE
6448 Dai	Gordon		16400
eget.tincidunt@convallisdolor.com			

Queries

Rooms Availability with RoomType:

This query uses a left join on the *rooms* and *roombookings* tables to find out which of the rooms in the hotel are available (i.e have not been booked) and what type of room it is. It then displays the room numbers and corresponding type of these rooms.

```
SELECT t1.RoomType, t1.RoomNumber
FROM rooms t1
LEFT JOIN roombookings t2 ON t2.RoomNumber =
t1.RoomNumber
WHERE t2.RoomNumber IS NULL
ORDER BY t1.RoomType;
```

ROOMTYPE	ROOMNUMBER
-----	-----
Deluxe	702
Deluxe	676
Deluxe	204
Excellent	8
Excellent	373
Magnificent	893
Magnificent	807
Magnificent	598
Magnificent	638
Magnificent	531

Average Price of Bookings:

This query takes the *TotalPrice* column from the *bookings* table and finds the average price of all bookings, this includes bookings where more than one room has been booked by the same customer.

```
SELECT ROUND(AVG(TotalPrice), 2)
AS Average
FROM bookings;
```

AVERAGE

4031.43

Duration of Bookings Greater Than the Average Price of Bookings:

This query displays how long people stayed in the hotel for bookings for which the prices are greater than the average price of bookings.

```
SELECT Duration, TotalPrice
FROM bookings
WHERE TotalPrice > (SELECT
ROUND(AVG(TotalPrice), 2)
FROM bookings);
```

DURATION	TOTALPRICE
-----	-----
20	8000
27	4050
27	5400
27	8100
28	19600
15	6000
20	20000
23	21550
26	20800
22	8800
22	16400

DURATION	TOTALPRICE
-----	-----
24	4800
23	4600
25	5000
22	4400
18	7200
25	5000
11	4400
22	4400
13	5200
24	9600
13	5200

DURATION	TOTALPRICE
-----	-----
18	9000

Total Revenue For April:

This query displays the total amount of revenue the hotel made in the month of April.

```
SELECT SUM(TotalPrice) AS Revenue
FROM bookings
WHERE StartDate BETWEEN '01/APR/18' AND '30/APR/18';
```

REVENUE
27550

Names of Guests Staying in Each Booked Room:

This query inner joins the *guests* table and the *guestrooms* table to find out the names of each guest staying in each booked room. The first name, last name and room number of each guest staying in the said room is given. This data is displayed in ascending order of room number.

```
SELECT t2.RoomNumber, t1.FirstName,
t1.LastName
FROM guests t1
INNER JOIN guestrooms t2 ON t2.GuestID =
t1.GuestID
ORDER BY t2.RoomNumber ASC;
```

ROOMNUMBER	FIRSTNAME	LASTNAME
1	Jeremy	Hodges
2	Libby	Hancock
2	Bradley	French
3	Wilma	Schmidt
4	Lisandra	Jacobson
5	Unity	Savage
5	Odessa	Baldwin
6	Idona	Perry
7	Richard	Mullen
11	Naida	Brooks
11	Kevyn	Stewart

ROOMNUMBER	FIRSTNAME	LASTNAME
35	Lucian	Green
35	Daryl	Espinoza
44	Cathleen	Cameron
63	Chase	Valencia
63	Cara	Harper
81	Nehru	Myers
91	Bianca	Duran
91	Graham	Delgado
92	Ivy	Farmer
92	Coby	Rollins
105	Urielle	Ross

ROOMNUMBER	FIRSTNAME	LASTNAME
128	Kylynn	Finch
147	Yuli	Rose
152	Sierra	Pratt
152	Martin	Zimmerman
154	Wang	Delaney
154	Farrah	Case
162	Hiram	Glass
162	Aphrodite	Wilkins
163	Britanney	Wyatt
166	Cailin	Schmidt
178	Clinton	Howell

ROOMNUMBER	FIRSTNAME	LASTNAME
182	Cooper	McClure
203	Hamilton	Bentley
203	Gil	Porter
225	Eden	Powell
234	Felix	Johnston
234	Gregory	Porter
243	Callum	Dorsey
245	Ila	Rivera
247	Leila	Thomas
247	Ivory	Curtis
273	Kelly	Sparks

ROOMNUMBER	FIRSTNAME	LASTNAME
273	Hall	Carlson
274	Edan	Mills
274	Timon	White
279	Orlando	Estes
279	Vielka	Ferguson
280	Gretchen	Espinoza
288	Tyrone	Matthews
288	May	Dunn
317	Brett	Slater
327	Unity	Joyner
327	Rigel	Kemp

ROOMNUMBER	FIRSTNAME	LASTNAME
336	Callum	Mays
336	Chanda	Blackwell
372	Zeus	Pratt
372	Stuart	Holcomb
375	Kiona	Glenn
375	Lacey	Maxwell
380	Rachel	Frank
380	Leroy	Stokes
389	Kyle	Rasmussen
389	Montana	York
393	Zena	Hart

ROOMNUMBER	FIRSTNAME	LASTNAME	ROOMNUMBER	FIRSTNAME	LASTNAME
398	Paul	Smith	556	Brennan	Donovan
418	Elizabeth	Lara	572	Gemma	Robbins
425	Ima	Guy	572	Shellie	Kline
425	Bree	Santos	576	Lev	Soto
426	Jesse	Mcknight	576	Sawyer	Stephens
432	Renee	May	589	Ronan	Woodard
469	Baker	Pugh	589	Ashton	Rodriquez
479	Emery	Shields	590	Delilah	Jones
487	Brynne	Curry	597	Brendan	Huffman
487	Brennan	Blanchard	597	Lance	Beck
489	Keely	Burgess	616	Driscoll	Sims
ROOMNUMBER	FIRSTNAME	LASTNAME	ROOMNUMBER	FIRSTNAME	LASTNAME
489	Quin	Pugh	616	Reese	Ochoa
493	Beatrice	Mendez	617	Drake	Jimenez
519	Ivana	Walters	617	Lareina	Hunt
519	Kylie	Morrow	619	Dawn	Adkins
522	Melyssa	Wallace	619	Jacob	Frank
522	Trevor	Johns	624	Todd	Singleton
525	Katelyn	Joyce	637	Lars	Ratliff
525	Brent	Lindsay	637	Kay	Hale
530	Alan	Jefferson	656	Cheryl	Watkins
530	Aline	Barber	656	Odessa	Snider
556	Ross	Mcintosh	675	Coby	Ewing
ROOMNUMBER	FIRSTNAME	LASTNAME	ROOMNUMBER	FIRSTNAME	LASTNAME
675	Wendy	Walters	823	Ian	Mann
683	Courtney	Macias	831	Jason	Barton
693	Iris	Barrera	851	Colin	Webb
701	Cheyenne	Church	851	Rafael	Garner
701	Shannon	Gardner	856	Neve	Aguilar
728	Harper	Pope	860	Brynne	Vasquez
728	Demetrius	Vance	867	Lyle	Zamora
753	Mariko	Vega	867	Maite	Compton
753	Destiny	Foley	868	Hall	Jenkins
756	Jamal	Cross	878	Audra	Brennan
785	Jorden	Coleman	878	Reagan	Solis
ROOMNUMBER	FIRSTNAME	LASTNAME	ROOMNUMBER	FIRSTNAME	LASTNAME
786	Erasmus	Manning	892	Perry	Valdez
786	Mary	Howe	892	Dai	Gordon
805	Kenyon	Cantu	916	Aubrey	Whitehead
805	Kiona	Waller	917	Sylvester	Fowler
806	Josephine	Bartlett	917	Isaiah	Cote
808	Reagan	Ryan	931	Craig	Hays
818	Zelda	Joseph	959	Kenneth	Cooke
818	Dai	Duran	963	Jacqueline	Cook
819	Heather	Anderson	984	Basil	Pittman
822	Kibo	Ballard	984	Sade	Montoya
822	Emma	Tillman	995	Kirby	Goodwin
ROOMNUMBER	FIRSTNAME	LASTNAME	ROOMNUMBER	FIRSTNAME	LASTNAME
995	Slade	Burgess			

Number of Different Types of Rooms:

This query counts the number of each of the three different types of rooms that we have in our hotel, this includes rooms that are booked as well as rooms that are available.

```
SELECT DISTINCT t1.RoomType,
(SELECT COUNT(*) FROM rooms t2 WHERE
t2.RoomType = t1.RoomType)
AS "Number"
FROM rooms t1;
```

ROOMTYPE	Number
Deluxe	44
Magnificent	34
Excellent	32

Rooms Available For the Month of April:

This query displays all the rooms available in the month of April. This means that all these rooms are not booked during this period. As the output includes 100 data items, we will show a snippet of the output to demonstrate it is working.

```
SELECT rooms.RoomNumber, rooms.RoomType
FROM rooms
WHERE RoomNumber NOT IN (
SELECT roomBookings.roomNumber
FROM roomBookings
INNER JOIN bookings ON
roomBookings.BookingID=bookings.BookingID
WHERE bookings.StartDate BETWEEN '01/APR/18'
AND '30/APR/18'
);
```

ROOMNUMBER	ROOMTYPE	ROOMNUMBER	ROOMTYPE
2	Deluxe	203	Deluxe
3	Excellent	204	Deluxe
4	Deluxe	234	Deluxe
5	Excellent	243	Magnificent
6	Magnificent	245	Excellent
7	Deluxe	247	Deluxe
8	Excellent	274	Excellent
11	Magnificent	279	Deluxe
35	Deluxe	280	Magnificent
44	Excellent	288	Deluxe
63	Deluxe	317	Excellent

ROOMNUMBER	ROOMTYPE	ROOMNUMBER	ROOMTYPE
81	Excellent	327	Excellent
91	Excellent	336	Excellent
92	Excellent	372	Excellent
105	Deluxe	373	Excellent
128	Deluxe	375	Magnificent
152	Excellent	380	Excellent
154	Magnificent	389	Excellent
163	Magnificent	393	Magnificent
166	Magnificent	398	Magnificent
178	Deluxe	425	Excellent
182	Deluxe	432	Deluxe

ROOMNUMBER	ROOMTYPE
469	Deluxe
479	Excellent
487	Magnificent
489	Deluxe
519	Magnificent
522	Deluxe
525	Excellent
530	Magnificent
531	Magnificent
556	Deluxe
572	Excellent

ROOMNUMBER	ROOMTYPE
576	Excellent
589	Magnificent
590	Magnificent
598	Magnificent
616	Deluxe
617	Magnificent
619	Magnificent
624	Magnificent
637	Magnificent
638	Magnificent
656	Magnificent

ROOMNUMBER	ROOMTYPE
675	Deluxe
676	Deluxe
683	Magnificent
693	Excellent
701	Deluxe
702	Deluxe
728	Deluxe
753	Deluxe
756	Deluxe
785	Magnificent
786	Deluxe

ROOMNUMBER	ROOMTYPE
805	Magnificent
807	Magnificent
808	Deluxe
818	Deluxe
819	Deluxe
822	Magnificent
823	Deluxe
831	Excellent
851	Magnificent
856	Excellent
860	Deluxe

ROOMNUMBER	ROOMTYPE
867	Deluxe
868	Deluxe
878	Deluxe
892	Magnificent
893	Magnificent
916	Excellent
917	Deluxe
931	Deluxe
959	Deluxe
963	Excellent
984	Excellent

ROOMNUMBER	ROOMTYPE
995	Magnificent

Names of Customers Who Paid More, in April, Than the Average Price of Bookings:

This query gets the list of the first and last names of the customers who paid more - in the month of April - than the average price of bookings overall.

```
SELECT t1.FirstName, t1.LastName
FROM customers t1
INNER JOIN bookings t2 ON t2.CustomerID = t1.CustomerID
WHERE t2.StartDate BETWEEN '01/APR/18' AND '30/APR/18'
AND t2.TotalPrice > (SELECT ROUND(AVG(TotalPrice), 2)
FROM bookings);
```

FIRSTNAME	LASTNAME
-----	-----
Rogan	Diaz
Kasper	Burgess

Average Age of Guests:

This query gets the ages of all guests staying in the hotel and takes the average of these ages. It then rounds this number to the nearest integer to give a whole number for the age.

```
SELECT
ROUND(AVG(TRUNC(MONTHS_BETWEEN(sysdate,DoB)/12)),
0) AvgAge
FROM guests;
```

AVGAGE

40

Names and Room Numbers of Guests Who Can Be Regarded As Elderly:

This query gets the first name, last name and room number of all guests who are above the age of 70. This is useful information as the hotel might be able to assist with mobility of these elderly guests.

```
SELECT t1.FirstName, t1.LastName, t2.RoomNumber
FROM guests t1
INNER JOIN guestrooms t2 ON t2.GuestID = t1.GuestID
WHERE
TRUNC(MONTHS_BETWEEN(sysdate,t1.DoB)/12) > 70;
```

FIRSTNAME	LASTNAME	ROOMNUMBER
-----	-----	-----
Courtney	Macias	683
Gemma	Robbins	572
Timon	White	274
Chanda	Blackwell	336
Rachel	Frank	380
Katelyn	Joyce	525
Shellie	Kline	572
Orlando	Estes	279
Mariko	Vega	753
Leroy	Stokes	380
Gregory	Porter	234
-----	-----	-----
Dawn	Adkins	619
Urielle	Ross	105
Zelda	Joseph	818
Hiram	Glass	162
Audra	Brennan	878
Isaiah	Cote	917
Dai	Duran	818
Sylvester	Fowler	917
Graham	Delgado	91
Colin	Webb	851
Unity	Joyner	327
-----	-----	-----
Brendan	Huffman	597
Aline	Barber	530
Perry	Valdez	892
Coby	Ewing	675
Hamilton	Bentley	203