

**COURSE : DATABASE
SYSTEMS AND SECURITY
(7CI022)**

STUDENT ID:

**OLOYEDE FESTUS OYEBISI /
2345336**

Table of contents

INTRODUCTION	3
Scenario	3
Business Rules	3
Information Needs	4
Data Modelling.....	5
ENTITY-RELATIONSHIP (ER) MODEL	6
Entities & Attributes.....	6
Relationships	7
Logical Model.....	9
Relational Model	10
NORMALIZATION.....	11
Normalization Process	11-15
Table Creation and Population	<i>Error! Bookmark not defined.</i>
Creating tables	16-17
Populating tables	17-20
Querying Database	20-25
Security.....	<i>Error! Bookmark not defined.</i>
Integrity	23-25
Data Ethics	25
CONCLUSION	26

INTRODUCTION

SCENARIO

This report is based on the design of an information system for a hotel (hotel reservation system) named “FLOW HOTEL AND SUITES”. This system facilitates the process of booking and managing hotel accommodations. It allows guests to search for available rooms, make reservations, and sometimes customize their stay with specific preferences.

Additionally, this information system assists the managers, the owners/executives, and shareholders of the hotel to keep track of the income made by the company over a period. In other words, this encourages transparency. It also provides tools to manage room inventory, track reservations, manage guest records, handle check-ins and check-outs, process payments, and generate reviews.

Generally, this system will streamline the booking process for both the hotel guests and staffs, thereby enhancing efficiency and improving the overall guest experience. This will undoubtedly help the organisation to perform financial, administrative, and operational analysis, thereby helping in making key decisions that aid in the advancements of the company.

BUSINESS RULES

Pinpointed business rules may be defined as follows:

- Each guest makes one or more reservations and each reservation is made by a guest, but not all guests make reservations.
- Each guest may or may not leave one or more reviews after their stay
- Each room accommodates one or more guests.
- For each reservation, a deposit or full payment must be made.

INFORMATION NEEDS

There are different levels of information needed by the stakeholders in the organisation/establishment.

OPERATIONAL

- Guests need to know the type of rooms available, so they can make reservations.
- Guests need to know the cost of rooms they are reserving.
- The organisation needs information on all guests making reservations and the type of room that is commonly booked to draw meaningful insights.
- The organisation needs the biodata of all reviews made to know where to improve and maintain.
- The organisation needs information on all reservations made, check-in and check-out dates.

ADMINISTRATIVE

- The organisation needs information of guests and reservations made for reference purposes and to efficiently manage bookings and room allocations.
- The organisation needs information of all guests checking in, along with other occupants during their stay.

ACCOUNTING AND BUDGETING

- The organization needs information on all payments made and respective payment methods for budgeting and transparency.
- The organisation needs information on all reservation deposits made and balance to be made available before check-in.

DATA MODELLING

This information system is designed to record and accommodate the information needs of all stakeholders of the organization by the provision of relevant data, in line with the business rules that has been defined. This information system would be designed with the following entities and attributes:

1. **GUEST:** This entity provides information about each guest coming into the hotel. Attributes include "**GUEST_ID**" (guest unique identifier/primary key), NAME (name of guest), ADDRESS (address of guest), GENDER (gender of guest), POSTCODE (postcode of guest), EMAIL_ADDRESS (email address of guest), MOBILE_NUMBER (mobile number of guest), and ROOMS_ID (rooms unique identifier).
2. **ROOM:** This entity provides info on the rooms the hotel has to offer. Attributes include "**ROOM_ID**" (room unique identifier/primary key), ROOM_TYPE (type of room on offer), ROOM_COST_PER_NIGHT (cost of the room per night), and MAXIMUM_OCCUPANCY (maximum number of occupants available).
3. **RESERVATION:** This entity provides information on the reservations made by customers. Attributes include "**RESERVATION_ID**" (reservation unique identifier/primary key), CHECK_IN_DATE (proposed check in date for the reservation), CHECK_OUT_DATE (proposed check out date for the reservation), RESERVATION_COST (cost of the reservation), and GUEST_ID (guest unique identifier).
4. **PAYMENT:** This entity provides information on the payments made by the customers. Attributes include "**PAYMENT_ID**" (payment unique identifier/primary key), PAYMENT_DATE (date payment was made), PAYMENT_METHOD (method of payment), PAYMENT_AMOUNT (amount of that particular payment), and RESERVATION_ID (reservation unique identifier).
5. **REVIEW:** This entity provides information on the reviews left by customers after leaving. Attributes include "**REVIEW_ID**" (review unique identifier/primary key), RATINGS (ratings on the reviews), COMMENTS (comments left in the reviews), and GUEST_ID (guest unique identifier).

ENTITY-RELATIONSHIP (ER) MODEL

An Entity-Relationship model is a diagram which explains and shows the logical structure of a database. The ER model displays the relationships between entities in a certain database. The main components of an ER model are Entities, Relationships, and Attributes.

ENTITIES & ATTRIBUTES

The entities of this database have been defined as follows (primary keys are boldened):

1. Guest: **GUEST_ID**, NAME, ADDRESS, POSTCODE, GENDER, EMAIL_ADDRESS, MOBILE_NUMBER and ROOM_ID. The foreign key here is ROOMS_ID.
2. Room: **ROOM_ID**, ROOM_TYPE, ROOM_COST_PER_NIGHT, and MAXIMUM_OCCUPANCY.
3. Reservation: **RESERVATION_ID**, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, and GUEST_ID. The foreign key here is GUEST_ID.
4. Payment: **PAYMENT_ID**, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, and RESERVATION_ID. The foreign key here is RESERVATION_ID.
5. Review: **REVIEW_ID**, RATINGS, COMMENTS, and GUEST_ID. The foreign key here is GUEST_ID.

RELATIONSHIPS

These are the associations between two or more entities of a database. The following relationships are defined for this database:

1. Guest and Reservation:
 - Relationship: Guest (1..1) makes Reservations (0..*)
Each guest can make zero to many reservations. Each reservation is made by only one guest.
 - Cardinality: One to Many (1:N)

- Participation: Source optional & Target mandatory

2. Room and Guest:

- Relationship: Rooms (1..1) can harbour Guests (0..*).

Each room can harbour zero to multiple guests, and each guest can only be in one room.

- Cardinality: One to Many (1:N)
- Participation: Source optional, Target mandatory.

3. Guest and Review:

- Relationship: Guest (1..1) can make Reviews (0..*)

Each guest can make zero to many reviews, and each review must be made by a guest.

- Cardinality: One to Many (1:N)
- Participation: Source optional, Target Mandatory

4. Reservation and Payment:

- Relationship: Reservation (1..1) must be made with Payments (1..*)

Each reservation must be made with one or more payments, and each payment must have been actioned by a reservation.

- Cardinality: One to Many (1:N)
- Participation: Source mandatory, Target mandatory.

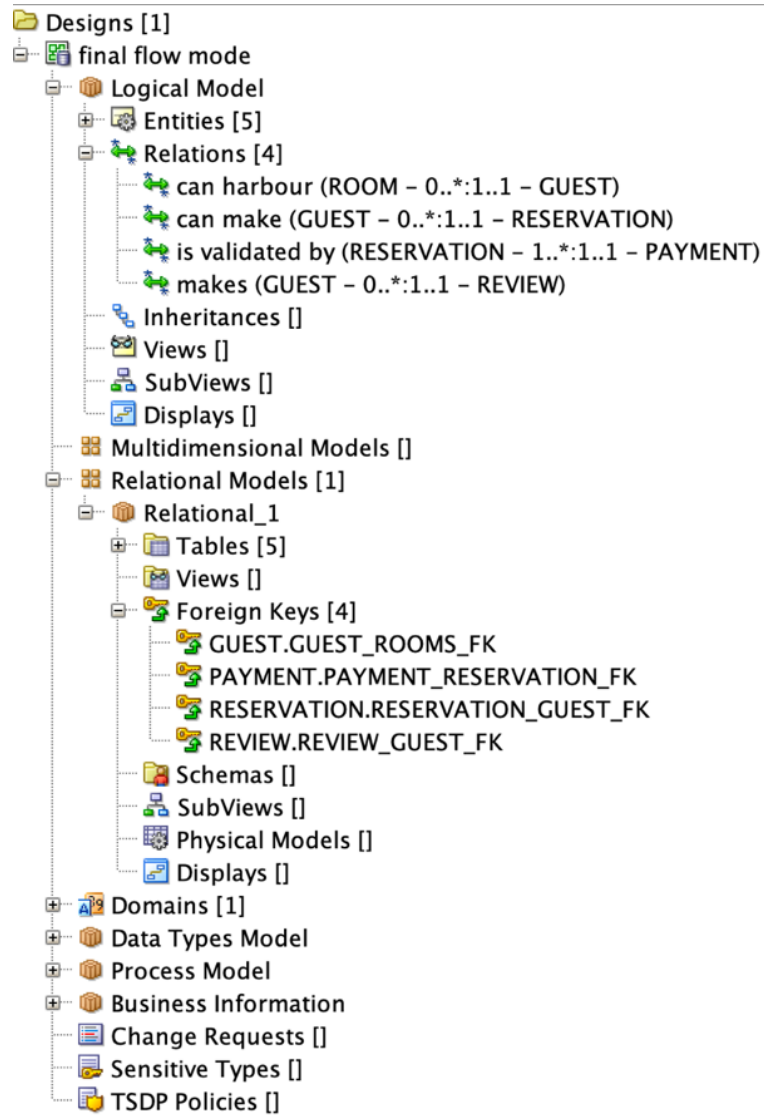


Fig 1: The Entity relationship definitions

LOGICAL MODEL

The logical model is built using Oracle SQL Data Modeller by defining entities, attributes and the relationship between attributes. Figure 1 shows this step and Figure 2 shows logical model.

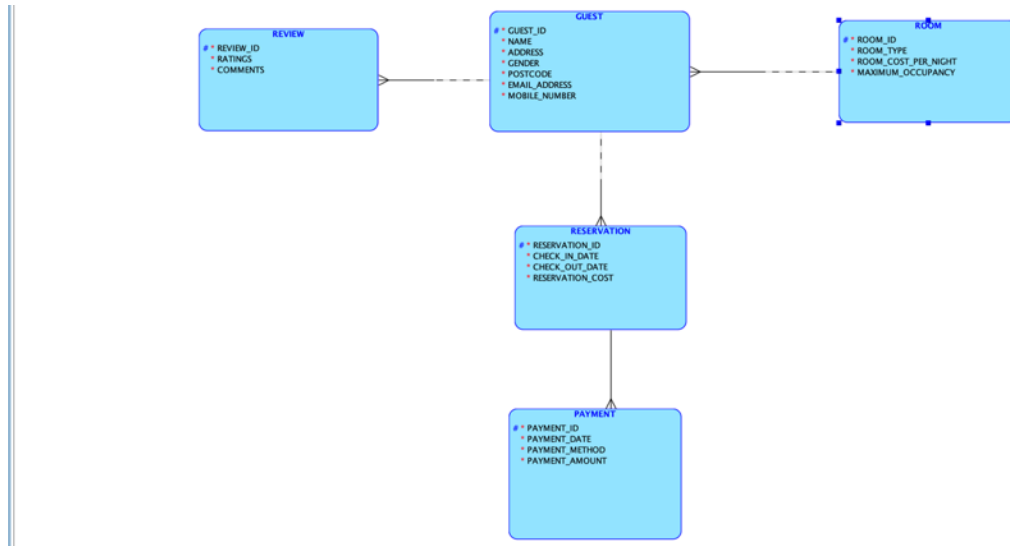


Figure 2: Logical ER Model

Relational Model

Oracle SQL Data Modeller automatically engineers a relational model based once the relationships are established on the logical model. Figure 3 shows the automatically engineered relational model.

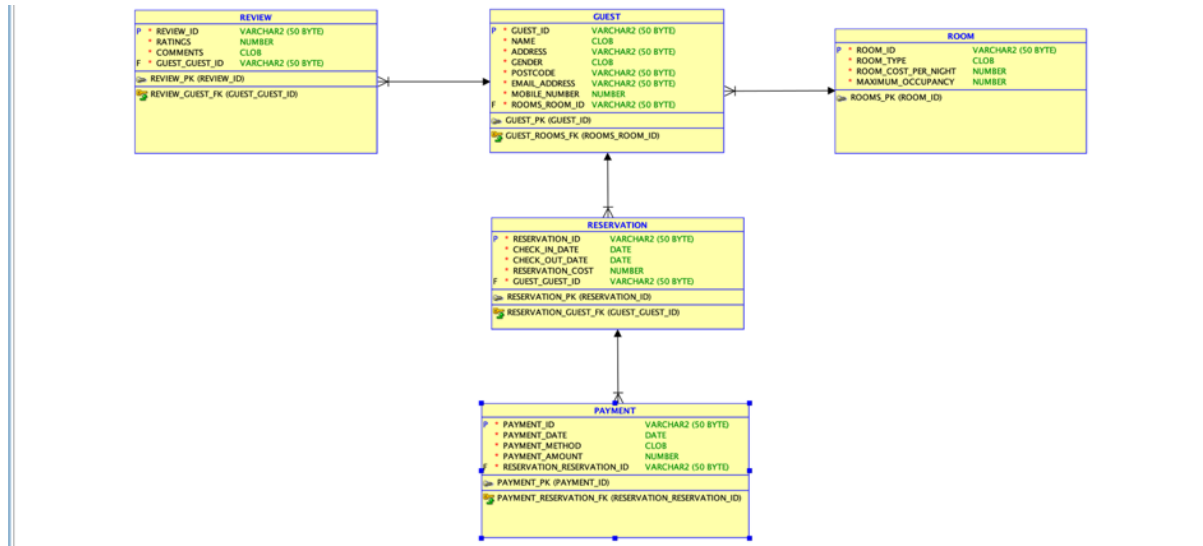


Figure 3: Relational ER Model

NORMALIZATION

Normalization refers to the process of organizing data in a database efficiently. It involves breaking down large tables into smaller ones and defining relationships between them to reduce redundancy.

Figure 4 shows a dummy dataset for Guest's reservations in an unnormalized form (UNF). I would attempt to normalize this relation to 3NF.

NORMALIZATION PROCESS

This describes the steps taken to normalize my dataset from Unnormalized form to Third normal form in relation to FLOW HOTEL'S information system:

1. **Unnormalized Form (UNF):** This refers to a context where tables or relations have not been structured to eliminate redundancy and dependency issues. This means the table contains repeating groups i.e., there is more than one value at the intersection of each row and column. Figure 4 below shows my dummy data is not unnormalized because it contains no repeating groups. Upon further analysis of the relation table, I have identified **PAYMENT_ID** as the key attribute because it fully functionally determines every attribute in the relation. I will label this relation (entity) **PAYMENT**.

PMNT_ID	RESERVATION_ID	GUEST_ID	NAME	ROOM_ID	ROOM TYPE	ROOM COST	MAXIMUM OCCUPANCY	CHECK IN DATE	CHECK OUT DATE	RESERVATION COST	PAYMENT DATE	PAYMENT METHOD	PAYMENT AMT	ADDRESS	POSTCODE	GENDER	EMAIL ADDRESS	MOBILE NUMBER	REVIEW ID	RATINGS	COMMENTS
01	FLOW001	INNO0001	EROUJIMMY	C111	DELUXE	\$150.00	3	05/02/1999	05/05/1999	\$50	05/02/1999	CASH	\$150	1, WOLFPLANA, WOLVERHAMPTON	WV11876	MALE	EROUJIMMY@GMAIL.COM	44734586	R1	5	EXCELLENT
02	FLOW002	INNO0002	FESTUS OLOFIDE	C102	STANDARD	\$100.00	2	05/02/1999	05/06/1999	\$50	05/02/1999	CASH	\$100	3, WOLFPLANA, WOLVERHAMPTON	WV11777	MALE	FESTUSOLOFIDE@GMAIL.COM	44741474	R2	4	VERY GOOD
03	FLOW003	INNO0003	MERCY JOHNSON	C105	ROYAL	\$200.00	4	05/02/1999	05/05/1999	\$50	05/02/1999	CARD	\$200	2, BAVERCE WAY, WOLVERHAMPTON	WV11775	MALE	MERCYJOHNSON@GMAIL.COM	44802039	R3	5	EXCELLENT
04	FLOW004	INNO0004	DELLESHAPRU	C002	DELUXE	\$150.00	3	08/02/1999	05/06/1999	\$50	05/02/1999	CARD	\$150	3, BAVERCE WAY, WOLVERHAMPTON	WV11776	MALE	DELLESHAPRU@GMAIL.COM	44802040	R4	5	EXCELLENT
05	FLOW005	INNO0005	TOBI SAMIA	C100	DELUXE	\$150.00	3	05/05/1999	05/08/1999	\$50	05/05/1999	CARD	\$150	9, WESTLAND DRIVE, WOLVERHAMPTON	WV10440	MALE	TOBISAMIA@GMAIL.COM	44740904	R5	5	EXCELLENT
06	FLOW006	INNO0006	EZEKIANU SANDO	C406	DELUXE	\$150.00	3	05/08/1999	05/10/1999	\$50	05/08/1999	CASH	\$150	8, WOLFPLANA, WOLVERHAMPTON	WV10445	MALE	EZEKIANUSANDO@GMAIL.COM	44232024	R7	4	VERY GOOD
07	FLOW007	INNO0007	GIDEON KIMITHA	C203	STANDARD	\$100.00	2	05/09/1999	05/11/1999	\$50	05/09/1999	CASH	\$100	5, FRODOCK, WOLVERHAMPTON	WV15402	FEMALE	GIDEONKIMITHA@GMAIL.COM	44679421	R6	5	EXCELLENT
08	FLOW008	INNO0008	PASCAL CROSS	C001	EXECUTIVE	\$250.00	5	05/11/1999	05/12/1999	\$50	05/11/1999	CARD	\$250	2, BOBCHALTON, WOLVERHAMPTON	WV10442	MALE	PASCALCROSS@GMAIL.COM	44223789	R10	5	EXCELLENT
12	FLOW0012	INNO00010	INDO GROSS	C105	ROYAL	\$200.00	4	04/05/1999	10/5/1999	\$50	14/5/1999	CASH	\$200	5, ADAPLEANE, TELFORD	SV159BT	MALE	INDOGROSS@GMAIL.COM	44679806	R12	5	EXCELLENT

Figure 4: UNF/1NF/2NF PAYMENT DATA TABLE

2. **1st Normal Form (1NF):** For a table or set of data to be in 1st Normal Form, it means it does not contain any repeating groups. This can be seen in figure 4 above, where my data table does not contain any repeating groups, with the primary key **PAYMENT_ID**, uniquely identifying each row in the table.

3. **2nd Normal Form (2NF):** For a table to be in 2NF, the table must be in 1NF and contain no partial dependencies. Partial dependencies between entities must have been reduced. This means that the relation must also have a primary key of a single attribute. My primary key for this table is **PAYMENT_ID**, as it fully functionally determines every other attribute, and as seen in figure 4, this is functional dependency definition for the **PAYMENT** relation:

PAYMENT_ID \Rightarrow RESERVATION_ID, GUEST_ID, NAME, ROOM_ID, ROOM_TYPE, ROOM_COST, MAXIMUM_OCCUPANCY, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, ADDRESS, POSTCODE, GENDER, EMAIL_ADDRESS, MOBILE_NUMBER, REVIEW_ID, RATINGS, COMMENTS.

4. **3rd Normal Form (3NF):** A table is in 3NF if it is in 2NF, and has no transitive dependencies for non-prime attributes. The main goal of 3NF is to eliminate redundancy and prevent data anomalies by ensuring every non-prime attribute is not only fully functionally dependent on the primary key, but also directly dependent on it. Transitive dependencies which exist in the 2NF PAYMENT relation include:

Transitive Dependency 1: **RESERVATION_ID \Rightarrow GUEST_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST.**

Transitive Dependency 2: **ROOM_ID \Rightarrow ROOM_TYPE, ROOM_COST_PER_NIGHT, MAXIMUM_OCCUPANCY.**

Transitive Dependency 3: **REVIEW_ID \Rightarrow GUEST_ID, RATINGS, COMMENTS.**

Transitive Dependency 4: **GUEST_ID \Rightarrow NAME, ROOM_ID, ADDRESS, POSTCODE, GENDER, EMAIL_ADDRESS, MOBILE_NUMBER.**

To remove these transitive dependencies from the **PAYMENT** relation, transitive attributes must be eliminated from the relation. Foreign keys then emerge from the parent **PAYMENT** relation, and TD1, TD2, TD3, TD4 can be used to create the **RESERVATION**, **ROOM**, **REVIEW**, and **GUEST** entities respectively. **ROOM_ID** emerges as a foreign key in the **GUEST** entity, and **GUEST_ID** emerges as a foreign key in the **RESERVATION** and **REVIEW** entities. **RESERVATION_ID** is also foreign key in the final **PAYMENT** entity. This leaves our final 3NF **PAYMENT** TABLE as;

PAYMENT_ID \Rightarrow **PAYMENT_DATE**, **PAYMENT_METHOD**, **PAYMENT_AMOUNT**, **RESERVATION_ID**.

After this, the data base and the individual relations are now in 3NF as shown in the figures and tables below:

1	PAYMENT_ID	RESERVATION_ID	PAYMENT_DATE	PAYMENT_METHOD	PAYMENT_AMOUNT
2	P001	FLOW001	05/02/1999	CASH	\$150
3	P002	FLOW002	05/03/1999	CASH	\$100
4	P003	FLOW003	05/02/1999	CARD	\$200
5	P004	FLOW 004	05/03/1999	CARD	\$150
6	P006	FLOW006	05/05/1999	CARD	\$150
7	P007	FLOW007	05/08/1999	CASH	\$150
8	P008	FLOW008	05/09/1999	CASH	\$100
9	P010	FLOW0010	05/11/1999	CARD	\$250
10	P012	FLOW0012	14/5/1999	CASH	\$200

Figure 5: 3NF PAYMENT TABLE

1	RESERVATION_ID	GUEST_ID	CHECK_IN_DATE	CHECK_OUT_DATE	RESERVATION_COST
2	FLOW001	INN00001	05/02/1999	05/05/1999	\$50
3	FLOW002	INN00002	05/03/1999	05/06/1999	\$50
4	FLOW003	INN00006	05/02/1999	05/05/1999	\$50
5	FLOW 004	INN00003	08/03/1999	05/06/1999	\$50
6	FLOW006	INN00004	05/05/1999	05/08/1999	\$50
7	FLOW007	INN00005	05/08/1999	05/10/1999	\$50
8	FLOW008	INN00007	05/09/1999	05/11/1999	\$50
9	FLOW0010	INN00008	05/11/1999	05/12/1999	\$50
10	FLOW0012	INN000010	04/05/1999	16/5/1999	\$50

Figure 6: 3NF RESERVATION TABLE

ROOM_ID	ROOM_TYPE	ROOM_COST	MAXIMUM_OCCUPANCY
C111	DELUXE	\$150.00	3
C102	STANDARD	\$100.00	2
C105	ROYAL	\$200.00	4
C002	DELUXE	\$150.00	3
C100	DELUXE	\$150.00	3
C406	DELUXE	\$150.00	3
C203	STANDARD	\$100.00	2
C301	EXECUTIVE	\$250.00	5
C105	ROYAL	\$200.00	4

Figure 7: 3NF ROOM TABLE

1	GUEST_ID	NAME	ROOM_ID	ADDRESS	POSTCODE	GENDER	EMAIL_ADDRESS	MOBILE_NUMBER
2	INN00001	EBUKAJIMMY	C111	1,WULFRUNA,WOLVERHAMPTON	WV1 876	MALE	BG@YAHOO.COM	44734586
3	INN00002	FESTUS OLOYEDE	C102	3,WULFRUNA,WOLVERHAMPTON	WV1 777	MALE	FF@YAHOO.COM	44747474
4	INN00006	MERCY JOHNSON	C105	2,BAYICE WAY,WOLVERHAMPTON	WV1 775	MALE	VB@YAHOO.COM	44892839
5	INN00003	DELE SHAPIRU	C002	3,BAYICE WAY,WOLVERHAMPTON	WV1 776	MALE	BS@YAHOO.COM	44892840
6	INN00004	TOBISANYA	C100	9,WESTLAND DRIVE,WOLVERHAMPTON	WV2 AA3	MALE	VB@YAHOO.COM	44740904
7	INN00005	EZEIHUAKU SANDRA	C406	8,WULFRUNA,WOLVERHAMPTON	WV3 AAS	MALE	FG@YAHOO.COM	44212324
8	INN00007	GIDEON SMITH	C203	5,FROLICK,WOLVERHAMPTON	WW5 432	FEMALE	HN@GMAIL.COM	44673421
9	INN00008	PASCAL GROSS	C301	2,BOB CHARLTON,WOLVERHAMPTON	WV1 AA2	MALE	PO@GMAIL.COM	44223789
10	INN000010	INGO GROSS	C105	5,AGAPE LANE,TELFORD	SY5 98T	MALE	DE@YAHOO.COM	44673908

Figure 8: 3NF GUEST TABLE

1	REVIEW ID	GUEST_ID	RATINGS	COMMENTS
2	R1	INN00001	5	EXCELLENT
3	R2	INN00002	4	VERY GOOD
4	R3	INN00006	5	EXCELLENT
5	R4	INN00003	5	EXCELLENT
6	R6	INN00004	5	EXCELLENT
7	R7	INN00005	4	VERY GOOD
8	R8	INN00007	5	EXCELLENT
9	R10	INN00008	5	EXCELLENT
10	R12	INN000010	5	EXCELLENT

Figure 9: 3NF REVIEW TABLE

This [DUMMY DATA SHEET](#) shows the relations and transformations after each normalization process from UNF to 1NF to 2NF, and also 3NF for better understanding.

TABLE CREATION AND POPULATION

The database is created and populated using Oracle Live SQL. Five tables were created for GUEST, ROOM, RESERVATION, PAYMENT, and REVIEW. The SQL statements used in creating, altering, and populating these tables are shown in the following sections.

CREATING TABLES

The GUEST, ROOM, RESERVATION, PAYMENT, and REVIEW tables are created using CREATE TABLE statements, as shown in the figures below:

```
1 v CREATE TABLE guest (  
2     guest_id      VARCHAR2(50 BYTE) NOT NULL,  
3     name          CLOB NOT NULL,  
4     address       VARCHAR2(50 BYTE) NOT NULL,  
5     gender        CLOB NOT NULL,  
6     postcode      VARCHAR2(50 BYTE) NOT NULL,  
7     email_address VARCHAR2(50 BYTE) NOT NULL,  
8     mobile_number NUMBER NOT NULL,  
9     rooms_room_id VARCHAR2(50 BYTE) NOT NULL  
10 );
```

Figure 10: CREATE TABLE GUEST STATEMENT

```
43 v CREATE TABLE room (  
44     room_id          VARCHAR2(50 BYTE) NOT NULL,  
45     room_type        CLOB NOT NULL,  
46     room_cost_per_night NUMBER NOT NULL,  
47     maximum_occupancy NUMBER NOT NULL  
48 );
```

Figure 11: CREATE TABLE ROOM STATEMENT


```

24 v CREATE TABLE reservation (
25     reservation_id  VARCHAR2(50 BYTE) NOT NULL,
26     check_in_date   DATE NOT NULL,
27     check_out_date  DATE NOT NULL,
28     reservation_cost NUMBER NOT NULL,
29     guest_guest_id  VARCHAR2(50 BYTE) NOT NULL
30 );

```

Figure 12: CREATE TABLE RESERVATION STATEMENT

```

14 v CREATE TABLE payment (
15     payment_id      VARCHAR2(50 BYTE) NOT NULL,
16     payment_date    DATE NOT NULL,
17     payment_method  CLOB NOT NULL,
18     payment_amount  NUMBER NOT NULL,
19     reservation_reservation_id VARCHAR2(50 BYTE) NOT NULL
20 );

```

Figure 13: CREATE TABLE PAYMENT STATEMENT

```

34 v CREATE TABLE review (
35     review_id      VARCHAR2(50 BYTE) NOT NULL,
36     ratings        NUMBER NOT NULL,
37     comments       CLOB NOT NULL,
38     guest_guest_id VARCHAR2(50 BYTE) NOT NULL
39 );

```

Figure 14: CREATE TABLE REVIEW STATEMENT

POPULATING TABLES

Dummy values are inserted into the GUEST, ROOM, RESERVATION, PAYMENT, and REVIEW tables respectively using INSERT statements, as shown in the figures below:

```

14 v INSERT INTO guest(GUEST_ID,NAME,ROOMS_ROOM_ID,ADDRESS,POSTCODE,GENDER,EMAIL_ADDRESS,MOBILE_NUMBER)
15 VALUES
16 ('INN00001','EBUKA JIMMY','C111','1,WULFRUNA , WOLVERHAMPTON', 'WV1 876','MALE','BG@YAHOO.COM',44734586);
17 v INSERT INTO guest(GUEST_ID,NAME,ROOMS_ROOM_ID,ADDRESS,POSTCODE,GENDER,EMAIL_ADDRESS,MOBILE_NUMBER)
18 VALUES
19 ('INN00002','FESTUS OLOYEDE','C102','3,WULFRUNA , WOLVERHAMPTON', 'WV1 777', 'MALE','FF@YAHOO.COM',44747474);
20 v INSERT INTO guest(GUEST_ID,NAME,ROOMS_ROOM_ID,ADDRESS,POSTCODE,GENDER,EMAIL_ADDRESS,MOBILE_NUMBER)
21 VALUES
22 ('INN00006','MERCY JOHNSON','C105','2,BAYICE WAY, WOLVERHAMPTON', 'WV1 775','MALE','VB@YAHOO.COM',44892839);
23 v INSERT INTO guest(GUEST_ID,NAME,ROOMS_ROOM_ID,ADDRESS,POSTCODE,GENDER,EMAIL_ADDRESS,MOBILE_NUMBER)
24 VALUES
25 ('INN00003','DELE SHAPIRU','C002','3,BAYICE WAY, WOLVERHAMPTON', 'WV1 776','MALE','BS@YAHOO.COM',44892840);
26 v INSERT INTO guest(GUEST_ID,NAME,ROOMS_ROOM_ID,ADDRESS,POSTCODE,GENDER,EMAIL_ADDRESS,MOBILE_NUMBER)
27 VALUES
28 ('INN00004','TOBI SANYA','C100','9, WESTLAND DRIVE, WOLVERHAMPTON', 'WV2 AA3','MALE','VB@YAHOO.COM',44740904);
29 v INSERT INTO guest(GUEST_ID,NAME,ROOMS_ROOM_ID,ADDRESS,POSTCODE,GENDER,EMAIL_ADDRESS,MOBILE_NUMBER)
30 VALUES
31 ('INN00005','EZEIHUAKU SANDRA','C406','8,WULFRUNA,WOLVERHAMPTON', 'WV3 AAS','MALE','FG@YAHOO.COM',44212324);
32 v INSERT INTO guest(GUEST_ID,NAME,ROOMS_ROOM_ID,ADDRESS,POSTCODE,GENDER,EMAIL_ADDRESS,MOBILE_NUMBER)
33 VALUES
34 ('INN00007','GIDEON SMITH','C203','5,FROLICK,WOLVERHAMPTON', 'WV5 432','FEMALE','HN@GMAIL.COM',44673421);
35 v INSERT INTO guest(GUEST_ID,NAME,ROOMS_ROOM_ID,ADDRESS,POSTCODE,GENDER,EMAIL_ADDRESS,MOBILE_NUMBER)
36 VALUES
37 ('INN00008','PASCAL GROSS','C301','2,BOB CHARLTON ,WOLVERHAMPTON', 'WV1 AA2','MALE','PO@GMAIL.COM',44223789);
38 v INSERT INTO guest(GUEST_ID,NAME,ROOMS_ROOM_ID,ADDRESS,POSTCODE,GENDER,EMAIL_ADDRESS,MOBILE_NUMBER)
39 VALUES
40 ('INN00010','INGO GROSS','C105','5, AGAPE LANE, TELFORD', 'SY5 98T','MALE','DE@YAHOO.COM',44673908);

```

Figure 15: INSERT INTO GUEST

```

169 v INSERT INTO room (ROOM_ID,ROOM_TYPE,ROOM_COST_PER_NIGHT,MAXIMUM_OCCUPANCY)
170 VALUES
171 ('C111','DELUXE','150.00',3);
172 v INSERT INTO room (ROOM_ID,ROOM_TYPE,ROOM_COST_PER_NIGHT,MAXIMUM_OCCUPANCY)
173 VALUES
174 ('C102','STANDARD','100.00',2);
175 v INSERT INTO room (ROOM_ID,ROOM_TYPE,ROOM_COST_PER_NIGHT,MAXIMUM_OCCUPANCY)
176 VALUES
177 ('C105','ROYAL','200.00',4);
178 v INSERT INTO room (ROOM_ID,ROOM_TYPE,ROOM_COST_PER_NIGHT,MAXIMUM_OCCUPANCY)
179 VALUES
180 ('C002','DELUXE','150.00',3);
181 v INSERT INTO room (ROOM_ID,ROOM_TYPE,ROOM_COST_PER_NIGHT,MAXIMUM_OCCUPANCY)
182 VALUES
183 ('C100','DELUXE','150.00',3);
184 v INSERT INTO room (ROOM_ID,ROOM_TYPE,ROOM_COST_PER_NIGHT,MAXIMUM_OCCUPANCY)
185 VALUES
186 ('C406','DELUXE','150.00',3);
187 v INSERT INTO room (ROOM_ID,ROOM_TYPE,ROOM_COST_PER_NIGHT,MAXIMUM_OCCUPANCY)
188 VALUES
189 ('C203','STANDARD','100.00',2);
190 v INSERT INTO room (ROOM_ID,ROOM_TYPE,ROOM_COST_PER_NIGHT,MAXIMUM_OCCUPANCY)
191 VALUES
192 ('C301','EXECUTIVE','250.00',5);

```

Figure 16: INSERT INTO ROOM

```

95 v INSERT INTO reservation (RESERVATION_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, GUEST_GUEST_ID)
96 VALUES
97 ('FLOW001', TO_DATE('02-MAY-1999', 'DD-MON-YYYY'), TO_DATE('05-MAY-1999', 'DD-MON-YYYY'), 50, 'INN00001');
98 v INSERT INTO reservation (RESERVATION_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, GUEST_GUEST_ID)
99 VALUES
100 ('FLOW002', TO_DATE('03/MAY/1999', 'DD-MON-YYYY'), TO_DATE('06/MAY/1999', 'DD-MON-YYYY'), 50, 'INN00002');
101 v INSERT INTO reservation (RESERVATION_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, GUEST_GUEST_ID)
102 VALUES
103 ('FLOW003', TO_DATE('02/MAY/1999', 'DD/MON/YYYY'), TO_DATE('05/MAY/1999', 'DD/MON/YYYY'), 50, 'INN00006');
104 v INSERT INTO reservation (RESERVATION_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, GUEST_GUEST_ID)
105 VALUES
106 ('FLOW004', TO_DATE('03/MAY/1999', 'DD/MON/YYYY'), TO_DATE('06/MAY/1999', 'DD/MON/YYYY'), 50, 'INN00003');
107
108 v INSERT INTO reservation (RESERVATION_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, GUEST_GUEST_ID)
109 VALUES
110 ('FLOW006', TO_DATE('05/MAY/1999', 'DD/MON/YYYY'), TO_DATE('08/MAY/1999', 'DD/MON/YYYY'), 50, 'INN00004');
111 v INSERT INTO reservation (RESERVATION_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, GUEST_GUEST_ID)
112 VALUES
113 ('FLOW007', TO_DATE('08/MAY/1999', 'DD-MM-YYYY'), TO_DATE('10/MAY/1999', 'DD/MON/YYYY'), 50, 'INN00005');
114 INSERT INTO reservation (RESERVATION_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, GUEST_GUEST_ID)
115 VALUES
116 ('FLOW008', TO_DATE('09/MAY/1999', 'DD-MM-YYYY'), TO_DATE('11/MAY/1999', 'DD/MON/YYYY'), 50, 'INN00007');
117 INSERT INTO reservation (RESERVATION_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, GUEST_GUEST_ID)
118 VALUES
119 ('FLOW0010', TO_DATE('11/MAY/1999', 'DD-MM-YYYY'), TO_DATE('12/MAY/1999', 'DD/MON/YYYY'), 50, 'INN00008');
120 INSERT INTO reservation (RESERVATION_ID, CHECK_IN_DATE, CHECK_OUT_DATE, RESERVATION_COST, GUEST_GUEST_ID)
121 VALUES
122 ('FLOW0012', TO_DATE('05/APR/1999', 'DD-MM-YYYY'), TO_DATE('16/MAY/1999', 'DD/MON/YYYY'), 50, 'INN000010');

```

Figure 17: INSERT INTO RESERVATION

```

57 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
58 VALUES
59 ('P001', '02/MAY/1999', 'CASH', 150, 'FLOW001');
60 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
61 VALUES
62 ('P002', '03/MAY/1999', 'CASH', 100, 'FLOW002');
63 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
64 VALUES
65 ('P003', '02/MAY/1999', 'CARD', 200, 'FLOW003');
66 INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
67 VALUES
68 ('P004', '03/MAY/1999', 'CARD', 150, 'FLOW004');
69 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
70 VALUES
71 ('P006', '05/MAY/1999', 'CARD', 150, 'FLOW006');
72 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
73 VALUES
74 ('P007', '08/MAY/1999', 'CASH', 150, 'FLOW007');
75 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
76 VALUES
77 ('P008', '09/MAY/1999', 'CASH', 100, 'FLOW008');
78 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
79 VALUES
80 ('P010', '11/MAY/1999', 'CARD', 250, 'FLOW0010');
81 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
82 VALUES
83 ('P012', '14/MAY/1999', 'CASH', 200, 'FLOW0012');

```

Figure 18: INSERT INTO PAYMENT


```

57 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
58 VALUES
59 ('P001', '02/MAY/1999', 'CASH', 150, 'FLOW001');
60 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
61 VALUES
62 ('P002', '03/MAY/1999', 'CASH', 100, 'FLOW002');
63 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
64 VALUES
65 ('P003', '02/MAY/1999', 'CARD', 200, 'FLOW003');
66 INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
67 VALUES
68 ('P004', '03/MAY/1999', 'CARD', 150, 'FLOW004');
69 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
70 VALUES
71 ('P006', '05/MAY/1999', 'CARD', 150, 'FLOW006');
72 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
73 VALUES
74 ('P007', '08/MAY/1999', 'CASH', 150, 'FLOW007');
75 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
76 VALUES
77 ('P008', '09/MAY/1999', 'CASH', 100, 'FLOW008');
78 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
79 VALUES
80 ('P010', '11/MAY/1999', 'CARD', 250, 'FLOW010');
81 v INSERT INTO payment (PAYMENT_ID, PAYMENT_DATE, PAYMENT_METHOD, PAYMENT_AMOUNT, RESERVATION_RESERVATION_ID)
82 VALUES
83 ('P012', '14/MAY/1999', 'CASH', 200, 'FLOW012');

```

Figure 19: INSERT INTO REVIEW

QUERYING DATABASE

QUERY 1: This query below selects payment amount of each guest in ascending order.

```

SELECT guest.name, payment.payment_amount
FROM guest
JOIN reservation
ON guest.guest_id = reservation.guest_id
JOIN payment
ON reservation.reservation_id = payment.reservation_id
ORDER BY payment.payment_amount ASC

```

NAME	PAYMENT_AMOUNT
FESTUS OLOYEDE	100
GIDEON SMITH	100
EBUKA JIMMY	150
TOBI SANYA	150
EZEIHUAKU SANDRA	150
DELE SHAPIRU	150
MERCY JOHNSON	200
INGO GROSS	200
PASCAL GROSS	250

Figure 20: Querying database 1

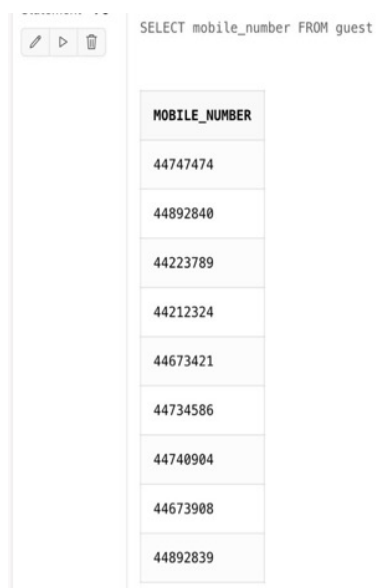
QUERY 2: This query shows a list of reviews with the corresponding guest name and rating using a left join between the review and guest tables.

```
SELECT rv.review_id, g.name, rv.ratings
FROM review rv
LEFT JOIN guest g ON rv.guest_guest_id = g.guest_id
```

REVIEW_ID	NAME	RATINGS
R12	INGO GROSS	5
R1	EBUKA JIMMY	5
R2	FESTUS OLOYEDE	4
R4	DELE SHAPIRU	5
R6	TOBI SANYA	5
R10	PASCAL GROSS	5
R7	EZEIHUAKU SANDRA	4
R3	MERCY JOHNSON	5
R8	GIDEON SMITH	5

Figure 21: Querying database 2

QUERY 3: This query shows the mobile numbers of guests in the hotel.



```
SELECT mobile_number FROM guest
```

MOBILE_NUMBER
44747474
44892840
44223789
44212324
44673421
44734586
44740904
44673908
44892839

Figure 22: Querying database 3

SECURITY, INTEGRITY, AND ETHICAL ASPECTS OF DATA GOVERNANCE

SECURITY

Data security is very critical and essential to any organization or business, as a robust data security management and strategy process enables an organization to protect its information and personal data against cyberattacks. It also helps them reduce the risk of human error and insider threats, which continue to be the cause of many data breaches.

Every organization should have policies put in place regarding data access and protection, because organizations are legally obliged to protect customer data from getting stolen or ending up in the wrong hands. High profile hacks can lead to customers losing trust in the business and fines/legal payments to repair damages.

Some general benefits of data security include:

- Keeps the reputation of the company/organization clean, as a great reputation is very important in the business world.
- Gives the organization a competitive edge, as this can easily set them apart from other competitors who may be struggling to do the same.
- Keeps the personal information and records collected by the company safe, secure, and well-protected.

DATA SECURITY IN RELATION TO FLOW HOTEL AND SUITES

Flow Hotel and Suites is a hotel organization that collects different types of personal information from guests, therefore data security practice is paramount to avoid any type of hacks. Clear policies have been put in place regarding data access and our staff members get trained in secure data handling practices.

These are some measures taken by Flow Hotel and Suites to ensure data security in the organization's database:

- **Every staff gets trained:** Every hotel staff has been provided with the fundamental skills to ensure a secure environment for guests and their personal

data. We also conduct regular tests to make sure the employees are keeping abreast with all the learnings and teachings in this aspect.

- **Use of cybersecurity tools:** Security tools such as firewalls, network monitor, and anti-malware have been put in place to protect the hotel against cybersecurity threats.
- **Payment card information encryption:** Strong data encryption methods are employed to make sure card information of our guests are well protected from any type of malicious attacks.

INTEGRITY

Data integrity is the accuracy, consistency, and completeness of data as it is maintained over time and across formats. This involves ensuring that your data collected is error-free, and conforms with the integrity controls and methods of regulatory compliance. This is very important in organizations as poor data quality can lead to:

- Poor decision making
- Potential legal issues
- Inadequate customer service

Types of data integrity and constraints include:

- **Domain Integrity:** This refers to the processes that ensure accuracy in each piece of data included in a domain, or a set of acceptable values a column may contain.
- **Entity Integrity:** This involves the creation of primary keys in order to identify data collected as unique entities, to ensure no data is listed more than once or is null. These are basically the rules governing each relation and their formats. The primary keys in Flow Hotel and Suites database are GUEST_ID, PAYMENT_ID, RESERVATION_ID, ROOM_ID, and REVIEW_ID.

```
11 ALTER TABLE guest ADD CONSTRAINT guest_pk PRIMARY KEY ( guest_id );
```

Figure 23: GUEST ENTITY CONSTRAINT

```
50 ALTER TABLE room ADD CONSTRAINT rooms_pk PRIMARY KEY ( room_id );
```

Figure 24: ROOM TABLE CONSTRAINT

```
32 ALTER TABLE reservation ADD CONSTRAINT reservation_pk PRIMARY KEY ( reservation_id );
```

Figure 25: RESERVATION TABLE CONSTRAINT

```
22 ALTER TABLE payment ADD CONSTRAINT payment_pk PRIMARY KEY ( payment_id );
```

Figure 26: PAYMENT TABLE CONSTRAINT

```
41 ALTER TABLE review ADD CONSTRAINT review_pk PRIMARY KEY ( review_id );
```

Figure 27: REVIEW TABLE CONSTRAINT

- **Referential Integrity:** This is the series of processes used to store and access data uniformly, allowing rules to be embedded into the database's structure regarding the use of foreign keys. This ensures and oversees the logical dependency of a foreign key on a primary key.

In relation to Flow Hotel and Suites, we take Integrity seriously, as this allows our guests to correctly register their information. We ensure that data entered into a row or column reflects the standard allowable value for that domain, row, or column. For example, the MOBILE_NUMBER attribute in the GUEST entity table can only collect digits/numerical entries, and this has been strictly put in place to maintain structure and eliminate any potential violation of the company's data integrity rules.

DATA ETHICS

These are the principles that regulate how an organization collects, protects, and uses data. Organizations must ensure they use data in a consistent manner that's in line with the company values.

In Flow Hotel and Suites, we have taken the following ethical data use routes, to ensure we offer the best services to our hotel guests:

- **Transparency:** This involves clear communication with our guests on what data would be collected, if it will be stored, and who it might be shared with. This ensures that our guests know exactly how the organization would be using their information.
- **Data Privacy:** Personal information collected from our guests such as email address, home address, and mobile number, which can be linked directly to them are well protected from public exposure. Private policies and security protocol have been enacted to secure these data and to also make our guests aware of how their data is stored and used.
- **Individual Agency & Consent:** This allows individuals to consent to, and make a choice on what personal data should be collected, stored, and made accessible to certain staff.

CONCLUSION

We successfully designed an information system for FLOW HOTEL AND SUITES. We defined the scenario, business rules, information needs, and created an appropriate Entity-Relationship (ER) model with proper entity relationships. We also normalized the database to Third normal Form.

The information system was implemented by creating the required entity and constraints for each entity. We also populated each entity with dummy data and queried the tables to draw insights.

In conclusion, we discussed data security, ethics, and integrity using the context of our scenario.