**Instituto Tecnológico y de Estudios Superiores de Occidente**

**DEPARTMENT OF ELECTRONICS, SYSTEMS AND COMPUTING**

**RELATIONAL DATABASES**

Logo, company name

Description automatically generated

**Final PROJECT: Earth BnB**

PRESENTS

**Aissa Berenice Gonzalez Fosado 752224 Data Science Engineering**

**Sofía Maldonado García 751299 Data Science Engineering**

**Priscila Cervantes Ramírez 750722 Data Science Engineering**

**Jesús Vargas Pacheco 750962 Systems Engineering**

Professor: Mtra. Ivonne del Rocio López Robles

Date: <21 11 2024>

CONTENTS

[PROJECT NAME 4](#_Toc163402484)

[INTRODUCTION 4](#_Toc163402485)

[1. PROBLEM STATEMENT 4](#_Toc163402486)

[1.1 BUSINESS RULES 5](#_Toc163402487)

[1.2 FORMATS USED 7](#_Toc163402488)

[1.3 OPERATIONS CARRIED OUT BY THE BUSINESS 7](#_Toc163402489)

[2. RELATIONAL DATABASE DESIGN 8](#_Toc163402490)

[2.1 Extended Entity Relationship Diagram (EER) 8](#_Toc163402491)

[2.2 Table Diagram 9](#_Toc163402492)

[3. NORMALIZATION 10](#_Toc163402493)

[3.1 Table: LISTINGS 10](#_Toc163402494)

[3.1.1 Initial Table 10](#_Toc163402495)

[3.1.2 FIRST NORMAL FORM 10](#_Toc163402496)

[3.1.2 SECOND NORMAL FORM 10](#_Toc163402497)

[3.1.3 THIRD NORMAL FORM 10](#_Toc163402498)

[3.2 Table: USERS 10](#_Toc163402499)

[3.2.1 Initial Table 10](#_Toc163402500)

[3.2.2 FIRST NORMAL FORM 11](#_Toc163402501)

[3.2.2 SECOND NORMAL FORM 11](#_Toc163402502)

[3.2.3 THIRD NORMAL FORM 11](#_Toc163402503)

[3.3 Table: REVIEWS 11](#_Toc163402504)

[3.3.1 Initial Table 11](#_Toc163402505)

[3.3.2 FIRST NORMAL FORM 11](#_Toc163402506)

[3.3.2 SECOND NORMAL FORM 11](#_Toc163402507)

[3.3.3 THIRD NORMAL FORM 11](#_Toc163402508)

[3.4 Table: FEES 12](#_Toc163402509)

[3.4.1 Initial Table 12](#_Toc163402510)

[3.4.2 FIRST NORMAL FORM 12](#_Toc163402511)

[3.4.2 SECOND NORMAL FORM 12](#_Toc163402512)

[3.4.3 THIRD NORMAL FORM 12](#_Toc163402513)

[3.5 Table: CALENDAR 12](#_Toc163402514)

[3.5.1 Initial Table 12](#_Toc163402515)

[3.5.2 FIRST NORMAL FORM 12](#_Toc163402516)

[3.5.2 SECOND NORMAL FORM 13](#_Toc163402517)

[3.5.3 THIRD NORMAL FORM 13](#_Toc163402518)

[4. DB2 DATABASE IMPLEMENTATION 14](#_Toc163402519)

[5. DEFINE AND IMPLEMENT TYPICAL BUSINESS PROCESSES 15](#_Toc163402520)

[5.1 DEFINE EACH BUSINESS PROCESSES 15](#_Toc163402521)

[5.2 IMPLEMENT TYPICAL BUSINESS PROCESSES: SQL OPERATIONS 20](#_Toc163402522)

[6 BASIC QUERY DESIGN – SQL SELECT 25](#_Toc163402523)

[7. STATISTICAL REPORTS: ADVANCED SQL 30](#_Toc163402524)

[8. CONCLUSIONS 36](#_Toc163402525)

[9. RECOMMENDATIONS AND REFLEXIONS 36](#_Toc163402526)

[10. BIBLIOGRAPHICAL SOURCES 36](#_Toc163402527)

[11 . ANNEX 37](#_Toc163402528)

# PROJECT NAME

Earth Bed and Breakfast Database Management System

# INTRODUCTION

The *earthbnb* company is dedicated to generating real estate reservations. Since opening, he has recorded all the operations of his business, making notes in notebooks, and keeping some controls in an MS Excel file.

The business operations are as follows:

a) Management and inventory of rental property, property owners are responsible for providing information about their properties and registering them on the company's site.

b) Reservation Management: property reservations must be paid in full to be confirmed, otherwise they remain in pending status

c) Calculation of Profits (revenue), cut of reservations and profits during a certain month

d) Reception of complaints about problems presented during the stay at a property

e) Reception of payments or cancellations of reservations

# 1. PROBLEM STATEMENT

Over time business have had problems with:

a) identify when properties are deregistered, or inactive for rental

b) keep track of the calculation of profits

c) keep track of payments, therefore, it is difficult to know if the reservation should be canceled or not

d) keep track of reservations

d) keep track of payments, taxes, fees

e) avoid duplication of data, nd

f) generate statistical reports quickly.

It is very clear that earthbnb needs administrative software to support it to solve the previously mentioned problems.

The decision has been made to address the problem by implementing each of the business operations within an DBMS. Later - at another time - they will develop a graphical interface that facilitates the editing/consultation of data and that communicates with the DBMS.

For the moment, those in charge of operating the business will manipulate the information directly in the DBMS through SQL queries and/or stored procedures.

## 1.1 BUSINESS RULES

|  |  |
| --- | --- |
| **RESERVATION** | |
| **1** | Every Reservation generates a transaction and must be registered |
| **2** | The same guest can be booked many times. |
| **3** | You can only include one rental property at a time in a reservation and it must be available on the selected dates before generating the reservation. |
| **4** | Each reservation has a unique identifier, which is used as a reference in all relationships or uses of the reservation. |
| **5** | Each reservation is paid in a single payment, which happens at the time of creating the reservation. |
| **6** | The reservation can be CONFIRMED or CANCELED. |
| **7** | Reservations have a limit of people included, if there are more people they must be paid as an extra. |
| **RESERVATION PAYMENTS** | |
| **8** | Payment is always electronic and can be made with a credit card, debit card, PayPal, Apple Pay, Google Wallet. |
| **9** | Payment is always in a single payment and a single payment method. |
| **RESERVATION CANCELLATION** | |
| **10** | If a Reservation is canceled, you must release the calendar dates |
| **11** | In case of cancellation, it is necessary to record the date/time of the cancellation request, as well as the date/time of system processing. |
| **12** | Once the reservation is cancelled, the reserved dates are released, and it is available again.  12.5. Then, update payment status as REFUND, and set Reservation as CANCELED. |
| **PROPERTY MANAGEMENT** | |
| **13** | Only properties that exist in the catalog can be rented. |
| **14** | Each property has a unique identifier, which is used as a reference in all relationships or uses of the reservation. |
| **15** | The property owner determines the rental price and cleaning fee. |
| **16** | Property owners may have at least one rental property, and may have more than one property available and active. |
| **17** | The types of properties can be: Entire home/apt, Hotel room, Private room, Shared room |
| **18** | Properties must explicitly have availability dates. |
| **19** | The property owner can determine the minimum and maximum number of nights to rent his/her property(s). |
| **PROPERTY AVAILABILITY CALENDARS** | |
| **20** | As a result of the reservation, the availability of dates for said property is updated |
| **21** | The company asks for visibility to the total number of days of the property available in 1 year. That is, of the 365 days of the year, we need to determine how many days the property will be available, the owner can configure these days as they wish. |
| **RATES/FEEs ON RESERVATIONS** | |
| **22** | The rates at service fees and property taxes will be determined by the type of property(Entire home/apt, Hotel room, Private room, Shared room), are fixed and never changes |
| **23** | As Service Fee: Entire home/apt – 16% , Hotel room – 15%, Private room – 10%,Shared room-10% of total |
| **24** | As Tax Fee: Entire home/apt – 16% , Hotel room – 16%, Private room – 16%,Shared room-16% of total |
| **27** | A property cleaning fee is always charged and is charged and executed only once, regardless of the number of days of stay. |
| **USERS OR HOSTS** | |
| **28** | Each user has a unique identifier, which is used as a reference in all relationships or uses of the reservation. |
| **29** | Users may or may not review their stay at the property |
| **30** | Any user can become a property host |
| **31** | Property hosts may or may not have verifications on social networks such as Facebook, Twitter, Instagram, LinkedIn. |
| **COMPLAINTS ABOUT STAYS** | |
| **32** | Users can open complaints about their experience at a property, or not. |
| **33** | If there are complaints, they can be: open or closed. |
| **34** | Any user can make reviews, as long as they have an associated reservation. |
| **35** | When a user makes a review, they will at least assign a rating of 1 – 5, with 5 being the maximum and highest rating for the stay, they can add comments about the experience. |

## 1.2 FORMATS USED

We used the Following formats given as PFD Files:

**User Profile**

**Host Profile**

**Review**

**Complaint**

**Booking Cancelled**

**New Booking Confirmed Version 1**

**New Booking Confirmed Version 2**

## 1.3 OPERATIONS CARRIED OUT BY THE BUSINESS

After assessing business rules and formats used, it’s been determined that the business processes targeted to solution are:

* *Creation of Reservations*
* *Cancellation of reservations*
* *Property Review Registry*
* *Register of Complaints about properties*
* *Variable rate registration*

# 

# 2. RELATIONAL DATABASE DESIGN

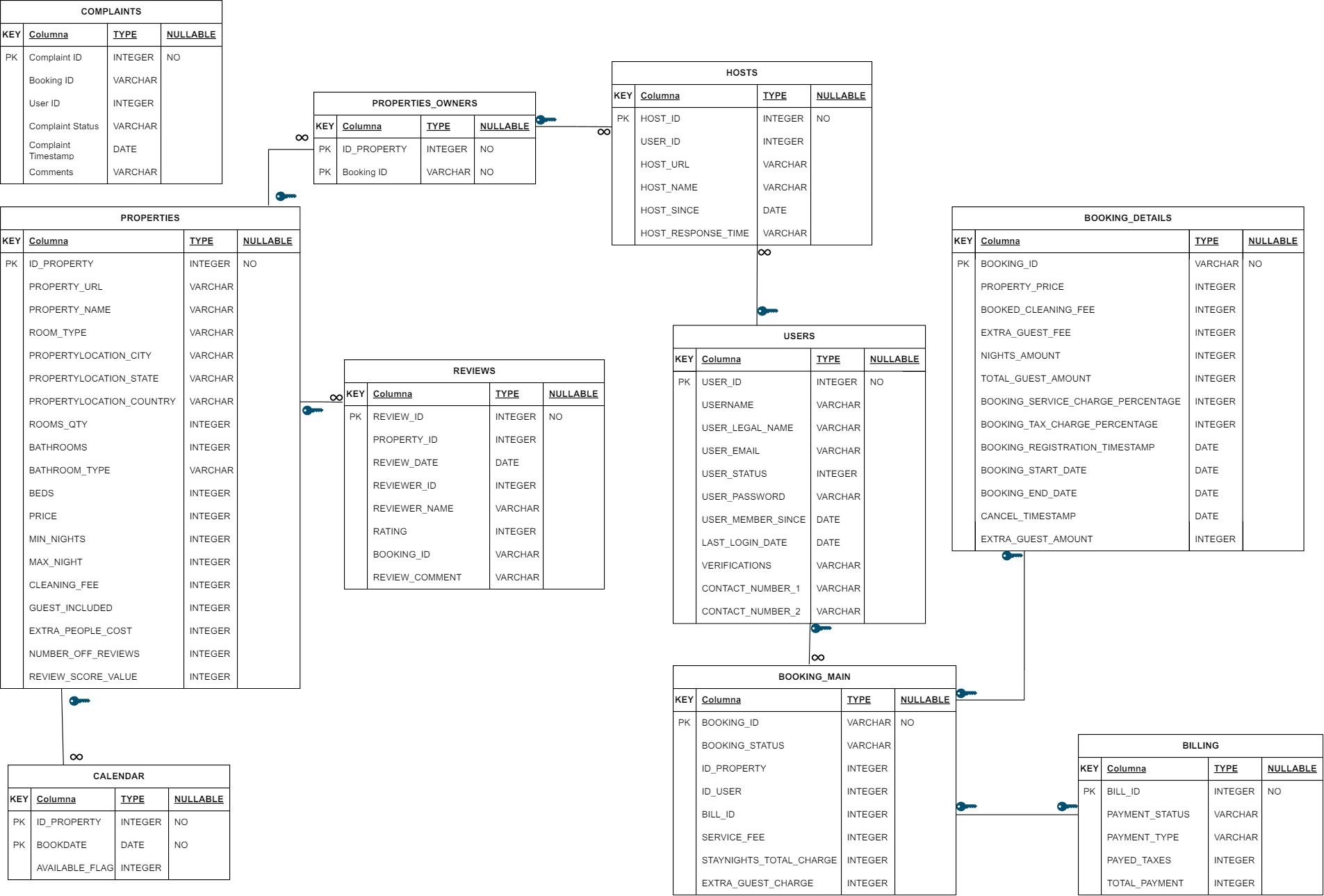
## 2.1 Extended Entity Relationship Diagram (EER)

A black screen with white text

Description automatically generated

## 2.2 Table Diagram

*The Table Diagram designed, which is also representing the EER on 5.1 below:*

**

# 3. NORMALIZATION

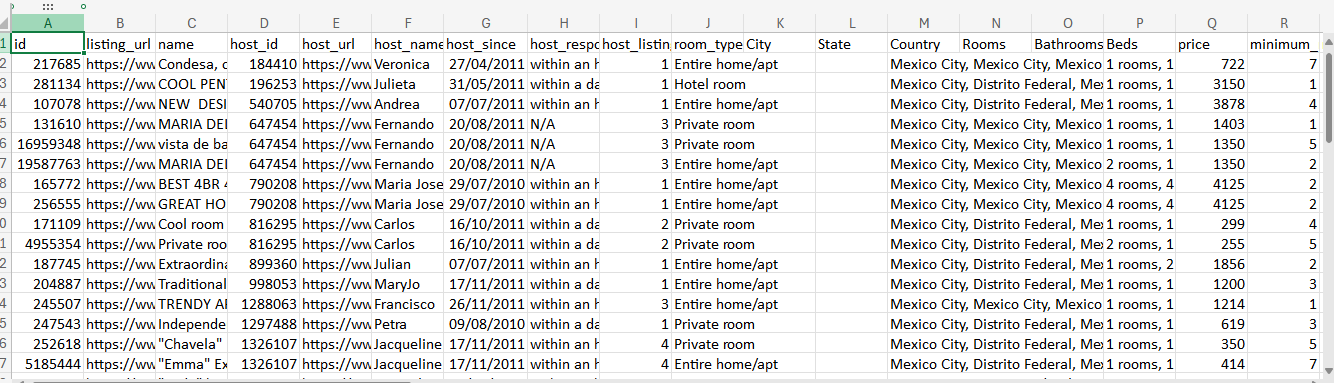
After analyzing the raw data provided by the business, it’s been decided to normalize so we can make it part of this solution, the data provided listed below:

* LISTING/PROPERTY DATA
* LISTING AVAILABILITY CALENDARS
* RATES/FEES SERVICE BASED ON PROPERTY TYPE
* USERS
* REVIEWS

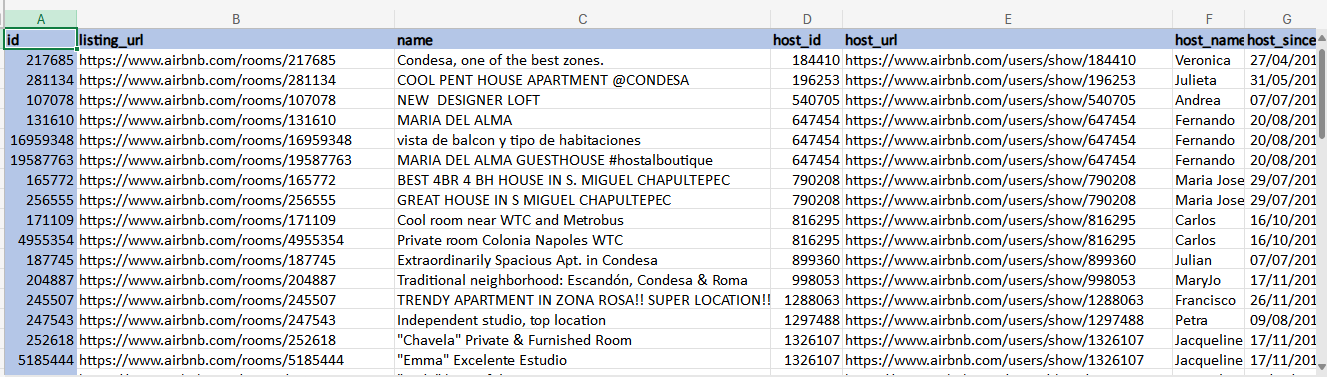
<below, from section 6.1 to 6.5 add the normalization forms details*– remove this line afterwards>*

## 3.1 Table: LISTINGS

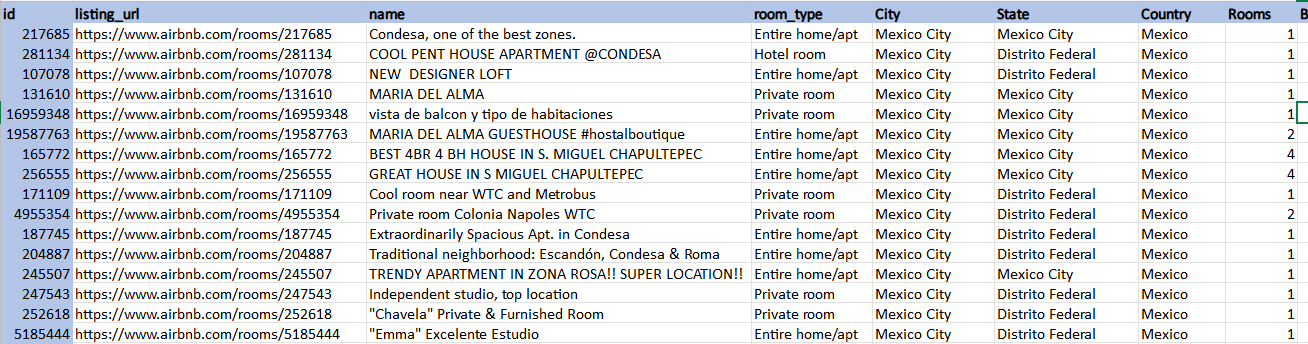
### 3.1.1 Initial Table



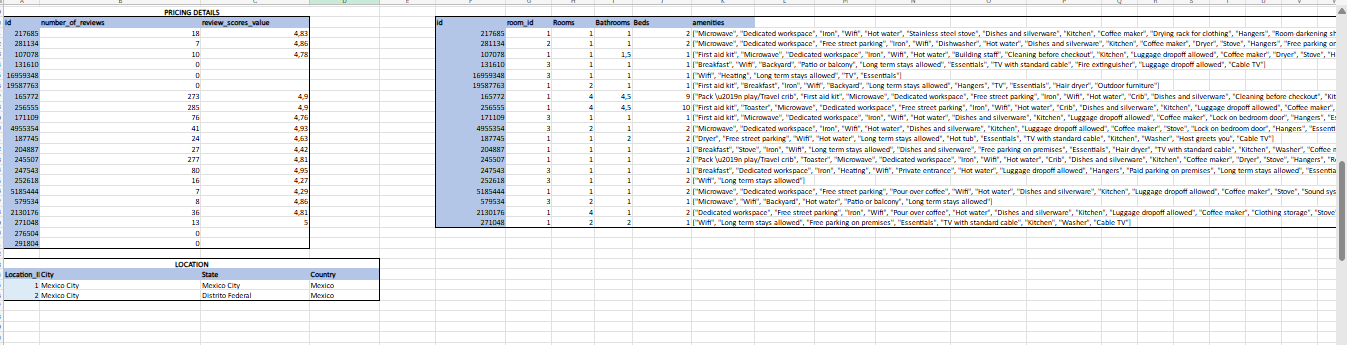
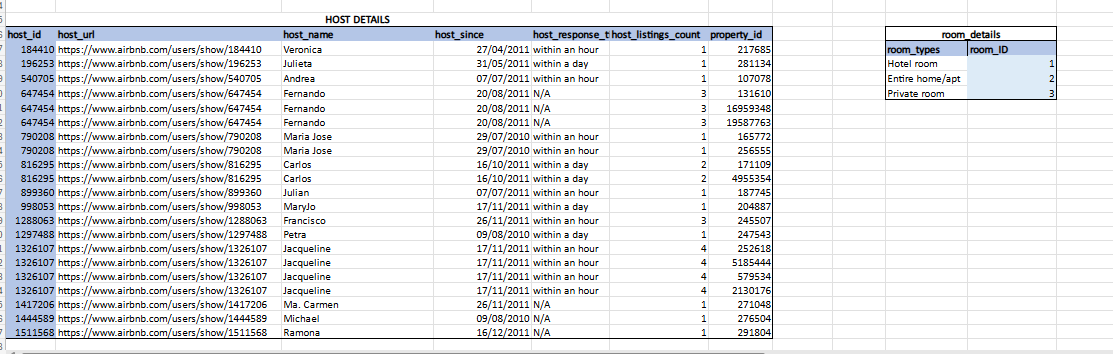
### 3.1.2 FIRST NORMAL FORM



### 3.1.2 SECOND NORMAL FORM

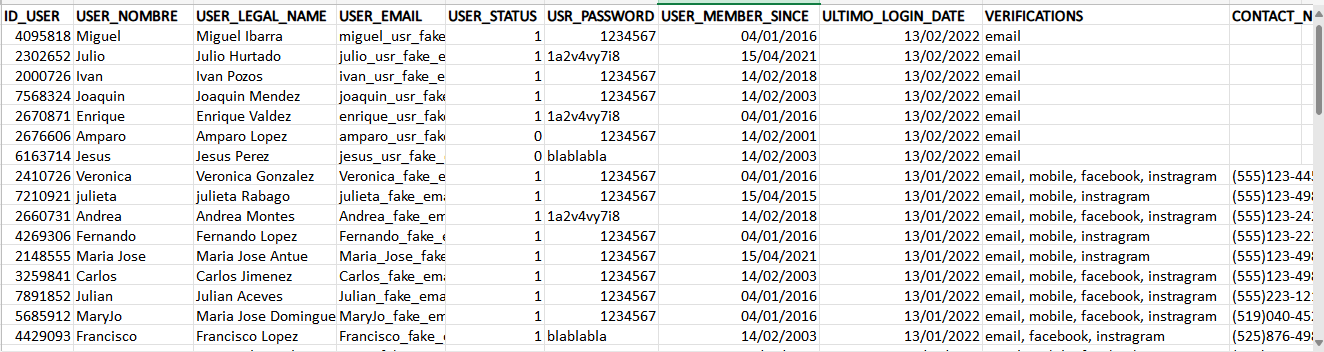


### 3.1.3 THIRD NORMAL FORM

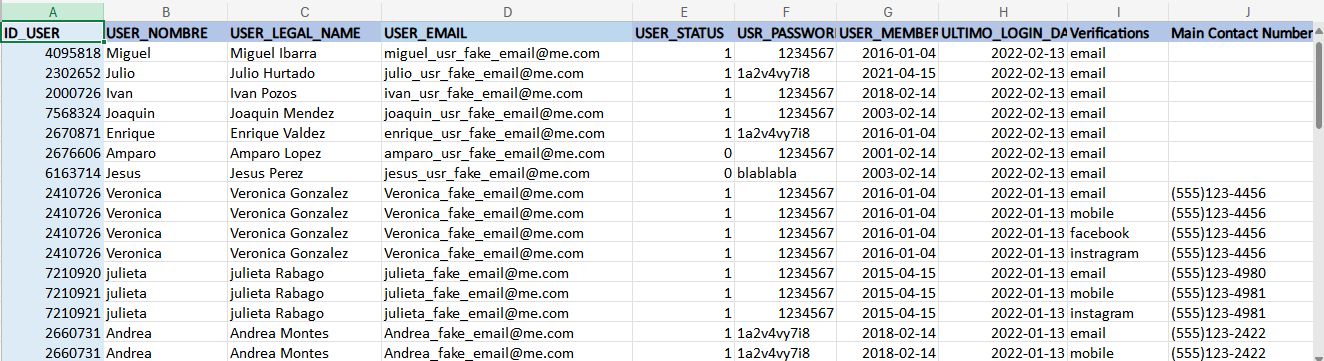


## 3.2 Table: USERS

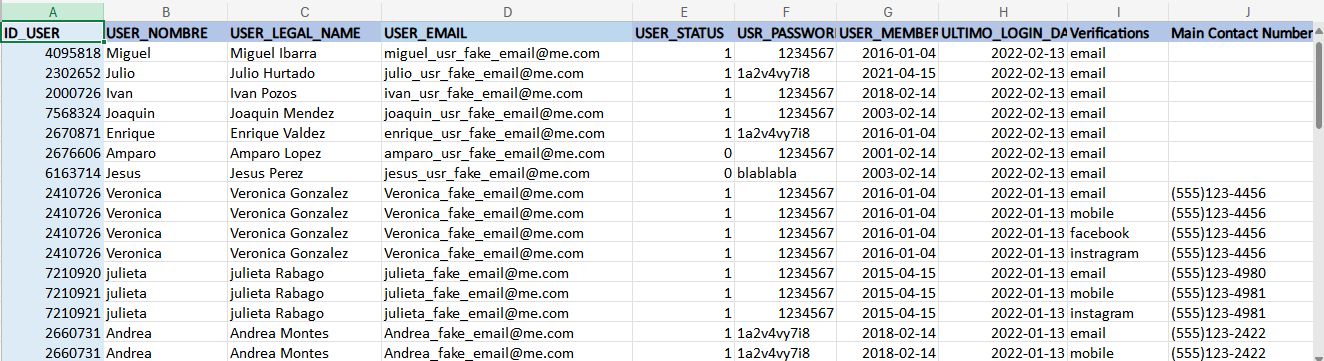
### 3.2.1 Initial Table



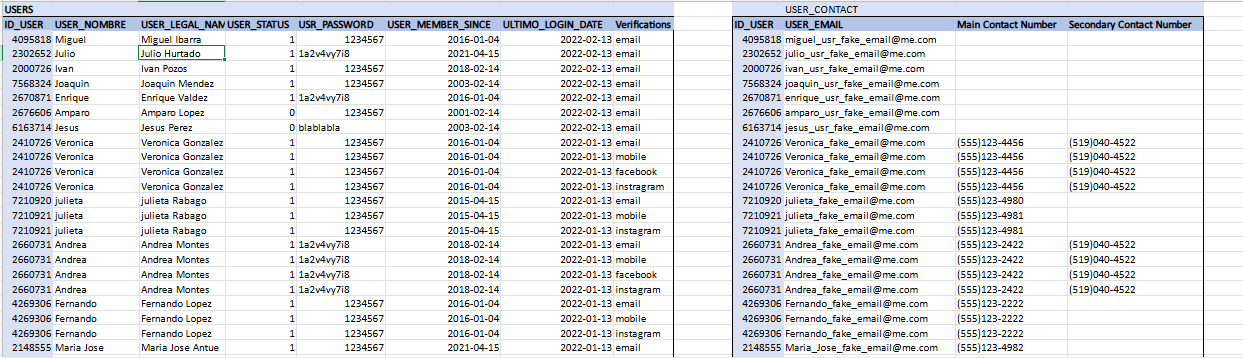
### 3.2.2 FIRST NORMAL FORM



### 3.2.2 SECOND NORMAL FORM

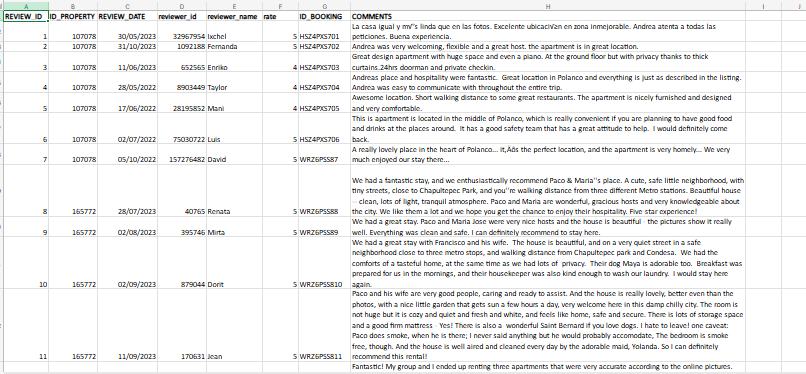


### 3.2.3 THIRD NORMAL FORM

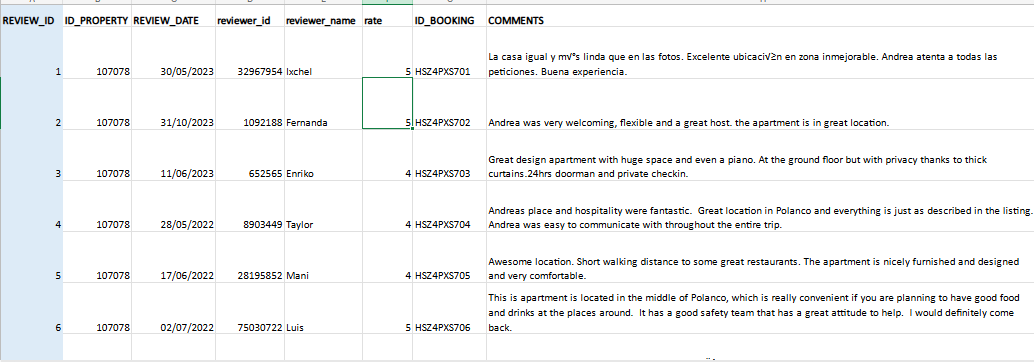


## 3.3 Table: REVIEWS

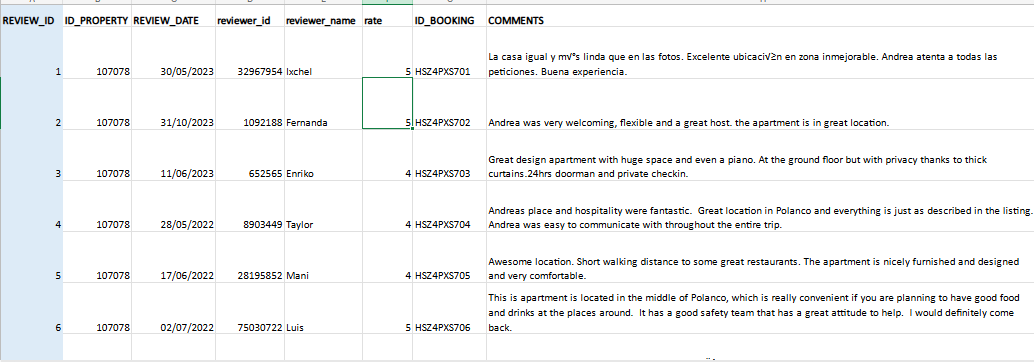
### 3.3.1 Initial Table



### 3.3.2 FIRST NORMAL FORM



### 3.3.2 SECOND NORMAL FORM



### 3.3.3 THIRD NORMAL FORM

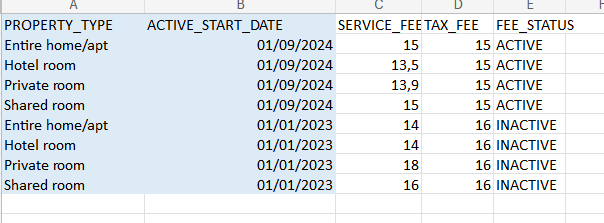


## 3.4 Table: FEES

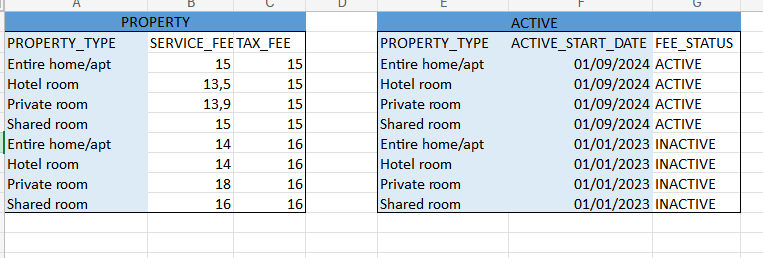
### 3.4.1 Initial Table



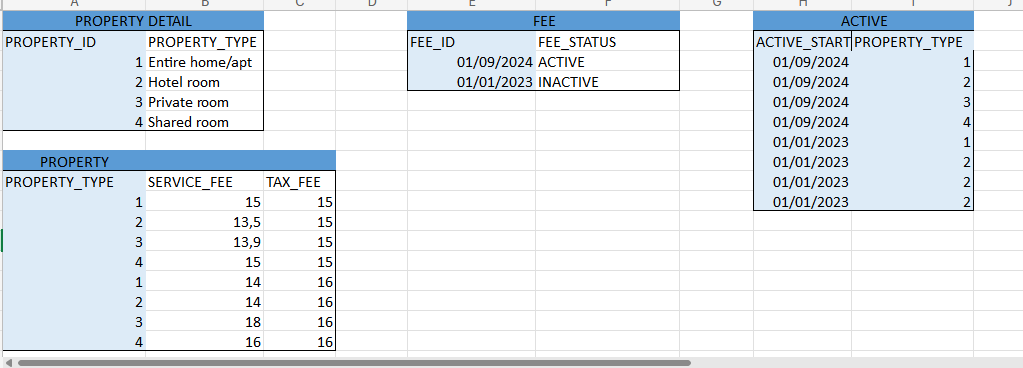
### 3.4.2 FIRST NORMAL FORM



### 3.4.2 SECOND NORMAL FORM

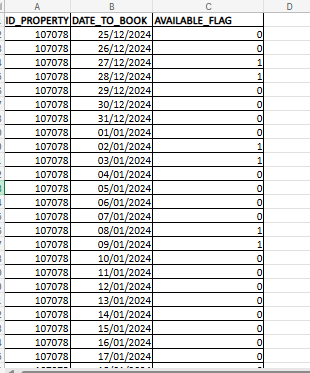


### 3.4.3 THIRD NORMAL FORM

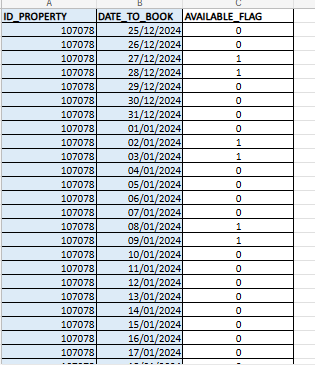


## 3.5 Table: CALENDAR

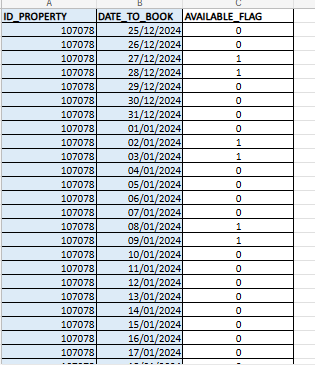
### 3.5.1 Initial Table



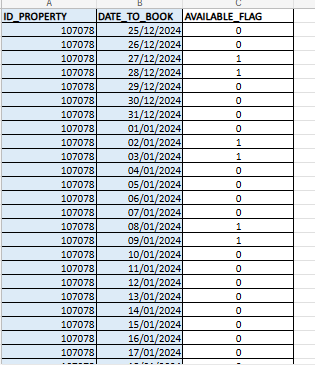
### 3.5.2 FIRST NORMAL FORM



### 3.5.2 SECOND NORMAL FORM

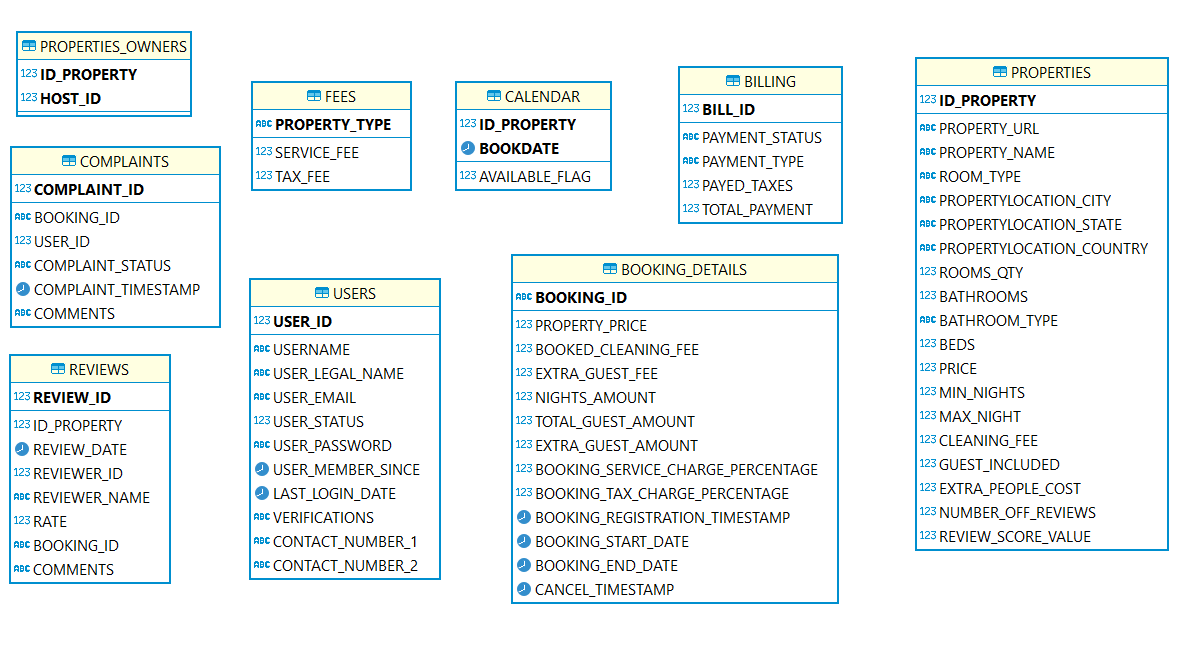


### 3.5.3 THIRD NORMAL FORM



LINK: [12 Project Progress Normalization 1FN-2FN.xlsx](https://iteso01-my.sharepoint.com/:x:/g/personal/aissa_gonzalez_iteso_mx/EXxVTnTcDdhNmwkhuFeJaEkBqKRGlp1_3BYIMslw-RztpA?e=9bQ3rW)

# 4. DB2 DATABASE IMPLEMENTATION



# 5. DEFINE AND IMPLEMENT TYPICAL BUSINESS PROCESSES

Describe what are the steps, and its corresponding explicit SQL transactions and impacted tables to carry out each of these business operations using IBM DB2/DBEAVER.

1. Creation of Reservations
2. Cancellation of reservations
3. Property Review Registry
4. Register of Complaints about properties

## 5.1 DEFINE EACH BUSINESS PROCESSES

|  |  |
| --- | --- |
| PROCESS 1 | BOOKING CREATION |
| STEP 1 | Validate or calculate all columns values for **ABNB.BOOKING\_DETAILS**   * BOOKING ID * PROPERTY PRICE * BOOKED CLEANING FEE * EXTRA GUEST FEE * NIGHTS AMOUNT * TOTAL GUEST AMOUNT * EXTRA GUEST AMOUNT * BOOKING SERVICE CHARGE PERCENTAGE * BOOKING TAX CHARGE PERCENTAGE * BOOKING REGISTRATION DATE/TIME * BOOKING START DATE * BOOKING END DATE   REFER to ANNEX 1 to get formulas. |
| STEP 2 | INSERT ALL VALUES FOR **ABNB.BOOKING\_DETAILS** Corresponding to the new BOOKING. |
| STEP 3 | Validate or calculate all columns values for **ABNB.BOOKING\_MAIN** like   * BOOKING\_ID * ID\_PROPERTY * ID\_USER * BILL\_ID * BOOKING\_STATUS * SERVICE\_FEE * STAYNIGHTS\_TOTAL\_CHARGE * EXTRA\_GUEST\_CHARGE   REFER to ANNEX 1 to get formulas,  for instance,  SERVICE\_**FEE**= ((NIGHTS\_AMOUNT \*PROPERTY\_PRICE) + (NIGHTS\_AMOUNT \*EXTRA\_GUEST\_FEE \*EXTRA\_GUEST\_AMOUNT))\* (BOOKING\_SERVICE\_CHARGE\_PERCENTAGE/100)  **STAYNIGHTS\_TOTAL\_CHARGE**= (NIGHTS\_AMOUNT \*PROPERTY\_PRICE + NIGHTS\_AMOUNT )  **EXTRA\_GUEST\_CHARGE** = (NIGHTS\_AMOUNT \*EXTRA\_GUEST\_FEE \*EXTRA\_GUEST\_AMOUNT) |
| STEP 4 | INSERT/UPDATE AT **ABNB.CALENDAR** DATA, LIKE  AVAILABLE\_FLAG=1 for booked property. |
| STEP 5 | Validate or calculate all columns values for PAYMENT DETAILS AT **ABNB.BILLING**   * BILL\_ID * PAYMENT\_STATUS * PAYMENT\_TYPE * PAYED\_TAXES * TOTAL\_PAYMENT   Then, INSERT DATA RELATED TO ABNB.BILLING |

|  |  |
| --- | --- |
| PROCESS 2 | RESERVATION/BOOKING CANCEL |
| STEP 1 | Prepare all data to update the previously data for the Booking to cancel.  Like   * ID\_PROPERTY * BOOKING\_ID * BILL\_ID * Book dates to release |
| STEP 2 | UPDATE **ABNB.CALENDAR** for all book dates to release for the corresponding properties.  SET AVAILABLE\_FLAG AS 0 |
| STEP 3 | UPDATE BOOKING\_STATUS to CANCELLED at **ABNB.BOOKING\_MAIN** for the corresponding Booking number. |
| STEP 4 | UPDATE PAYMENT\_STATUS to REFUND at **ABNB. BILLING** for the corresponding Booking number/BILL\_ID. |

|  |  |
| --- | --- |
| PROCESS 3 | REGISTER A COMPLAINT ABOUT A RESERVATION |
| STEP 1 | Prepare all data to register a complaint related to existing Booking.  Like   * COMPLAINT ID * BOOKING\_ID * USER ID who is setting the complaint * COMPLAINT STATUS * COMPLAINT DATE * COMPLAINT COMMENTS |
| STEP 2 | INSERT as needed at **ABNB.COMPLAINTS** |

|  |  |
| --- | --- |
| PROCESS 4 | REGISTER A REVIEW ABOUT A RESERVATION |
| STEP 1 | Prepare all data to register a review related to existing Booking.  REVIEW ID  ID PROPERTY  REVIEW DATE  REVIEWER ID  REVIEWER NAME  RATE  BOOKING ID  COMMENTS |
| STEP 2 | INSERT as needed at **ABNB.REVIEWS** |
| STEP 3 | UPDATE Review related columns at **ABNB.PROPERTIES** tables, like   * NUMBER\_OF\_REVIEWS, like avg(rate) based on existing reviews, example REVIEW\_SCORES\_VALUE=(SELECT avg(rate) FROM REVIEWS WHERE ID\_PROPERTY = 276504) * REVIEW\_SCORES\_VALUE, add +1 on from latest existing data. |

## 5.2 IMPLEMENT TYPICAL BUSINESS PROCESSES: SQL OPERATIONS

Based on the processes defined, you shall resolve each problem below by explicitly and clearly stating each SQL transaction to execute to achieve it.

*Be aware that for some problems, you can take the data straight-forward from the annex formats, for some others, you shall obtain and calculate all the attributes of each table corresponding to the reservation, obtaining the information from the tables available in the database*.

|  |  |
| --- | --- |
| 1 | Implement Reservation as Annex 2 |
|  | **INSERT** **INTO** BOOKING\_DETAILS (BOOKING\_ID,PROPERTY\_PRICE,BOOKED\_CLEANING\_FEE,EXTRA\_GUEST\_FEE,NIGHTS\_AMOUNT,TOTAL\_GUEST\_AMOUNT,EXTRA\_GUEST\_AMOUNT,BOOKING\_SERVICE\_CHARGE\_PERCENTAGE,BOOKING\_TAX\_CHARGE\_PERCENTAGE,BOOKING\_REGISTRATION\_TIMESTAMP,BOOKING\_START\_DATE,BOOKING\_END\_DATE,CANCEL\_TIMESTAMP)  **VALUES** (  **'ALB557788'**,  (**SELECT** PRICE **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY = 2130176),  (**SELECT** CLEANING\_FEE **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY = 2130176),  (**SELECT** EXTRA\_PEOPLE\_COST **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY = 2130176),  4,  6,  (6 - (**SELECT** GUEST\_INCLUDED **FROM** AIRBNB.PROPERTIES **WHERE**  ID\_PROPERTY=2130176)),  (**SELECT** SERVICE\_FEE  **FROM** AIRBNB.FEES  **INNER** **JOIN** AIRBNB.PROPERTIES  **ON** FEES.PROPERTY\_TYPE = PROPERTIES.ROOM\_TYPE  **WHERE** ID\_PROPERTY = 2130176),  (**SELECT** TAX\_FEE  **FROM** AIRBNB.FEES  **WHERE** PROPERTY\_TYPE =  (**SELECT** ROOM\_TYPE  **FROM** AIRBNB.PROPERTIES  **WHERE** ID\_PROPERTY = 2130176)),  **'2024-09-10'**,  **'2024-12-30 15:00:00'**,  **'2025-01-03 12:00:00'**,  **NULL**  ); |
|  | **INSERT** **INTO** AIRBNB.BOOKING\_MAIN (BOOKING\_ID, BOOKING\_STATUS, ID\_PROPERTY, ID\_USER, BILL\_ID, SERVICE\_FEE, STAYNIGHTS\_TOTAL\_CHARGE, EXTRA\_GUEST\_CHARGE)  **VALUES**(  **'ALB557788'**,  **'CONFIRMED'**,  2130176,  5685912,  998877,  (**SELECT**  ((NIGHTS\_AMOUNT \* PROPERTY\_PRICE) +  (EXTRA\_GUEST\_FEE \* NIGHTS\_AMOUNT \* EXTRA\_GUEST\_AMOUNT))  \* ((BOOKING\_SERVICE\_CHARGE\_PERCENTAGE / 100))  **FROM**  BOOKING\_DETAILS  **WHERE**  BOOKING\_ID = **'ALB557788'**),  (**SELECT**  (NIGHTS\_AMOUNT \* PROPERTY\_PRICE)  **FROM**  BOOKING\_DETAILS  **WHERE**  BOOKING\_ID = **'ALB557788'**),  (**SELECT**  (NIGHTS\_AMOUNT \* EXTRA\_GUEST\_FEE \* EXTRA\_GUEST\_AMOUNT)  **FROM**  BOOKING\_DETAILS  **WHERE**  BOOKING\_ID = **'ALB557788'**)  ); |
|  | **INSERT** **INTO** AIRBNB.BILLING (BILL\_ID,PAYMENT\_STATUS,PAYMENT\_TYPE,PAYED\_TAXES,TOTAL\_PAYMENT)  **SELECT**  998877,  **'CONFIRMED'**,  **'APPLE PAY'**,  (bd.BOOKING\_TAX\_CHARGE\_PERCENTAGE / 100) \* (bm.SERVICE\_FEE +  bm.STAYNIGHTS\_TOTAL\_CHARGE + bm.EXTRA\_GUEST\_CHARGE),  (bm.SERVICE\_FEE + bm.STAYNIGHTS\_TOTAL\_CHARGE + bm.EXTRA\_GUEST\_CHARGE +  bd.BOOKED\_CLEANING\_FEE)  **FROM**  AIRBNB.BOOKING\_MAIN bm  **JOIN**  AIRBNB.BOOKING\_DETAILS bd **ON** bm.BOOKING\_ID = bd.BOOKING\_ID  **WHERE**  bm.BOOKING\_ID = **'ALB557788'**; |
|  | **UPDATE** AIRBNB.CALENDAR  **SET** AVAILABLE\_FLAG = 0  **WHERE** BOOKDATE **BETWEEN**  (**SELECT** BOOKING\_START\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'ALB557788'**)  **AND** (**SELECT** BOOKING\_END\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'ALB557788'**); |

|  |  |
| --- | --- |
| 2 | Implement Reservation Cancel as Annex 4 |
|  | **SELECT** \* **FROM** ***BOOKING\_DETAILS*** *bd* **WHERE** ***BOOKING\_ID*** = **'XPR5SSS9C9'**    **UPDATE** AIRBNB.CALENDAR  **SET** AVAILABLE\_FLAG = 1  **WHERE** BOOKDATE **BETWEEN**  (**SELECT** BOOKING\_START\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'XPR5SSS9C9'**)  **AND** (**SELECT** BOOKING\_END\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'XPR5SSS9C9'**); |
|  | **UPDATE** AIRBNB.BILLING  **SET** PAYMENT\_STATUS = **'REFUND'**  **WHERE** BILL\_ID = ( **SELECT** BILL\_ID **FROM** AIRBNB.BOOKING\_MAIN **WHERE** BOOKING\_ID = **'XPR5SSS9C9'**); |
|  | **UPDATE** AIRBNB.BOOKING\_MAIN **SET** BOOKING\_STATUS = **'CANCELED'** **WHERE** BOOKING\_ID = **'XPR5SSS9C9'**; |

|  |  |
| --- | --- |
| 3 | Implement a Stay Review as Annex 5 |
|  | **INSERT** **INTO** AIRBNB.REVIEWS (REVIEW\_ID,ID\_PROPERTY,REVIEW\_DATE,REVIEWER\_ID,REVIEWER\_NAME,RATE,BOOKING\_ID,COMMENTS)  **VALUES**(37,**'252618'**,**'2025-01-20'**,3259841,**'Charles'**,4,**'EMIL2170615'**,**'Host was a great host! Responded very quickly,and was very accommodating to our needs/requests. Ckeck-in instructions were super detailed and helpful. The apartment was very clean and well stucked. We could not have asked for a better location. Overall a wonderfull place; we would stay here again easily!'**); |
|  | **UPDATE** AIRBNB.PROPERTIES **SET** NUMBER\_OFF\_REVIEWS = (NUMBER\_OFF\_REVIEWS + 1) **WHERE** ID\_PROPERTY = 252618; |
|  | **UPDATE** AIRBNB.PROPERTIES **SET** REVIEW\_SCORES\_VALUE = (**SELECT** **AVG**(rate) **FROM** AIRBNB.REVIEWS **WHERE** ID\_PROPERTY = 252618) **WHERE** ID\_PROPERTY = 252618; |
|  | **SELECT** \* **FROM** PROPERTIES *p* **WHERE** ID\_PROPERTY = 247543 |

|  |  |
| --- | --- |
| 4 | Register a Reservation as below:   1. ID booking: ROBL009090 2. property id: 247543 Independent studio, top location 3. total nights: 7 4. Booking Dates: from 2025-03-05 up to 2025-03-11 5. Amount of people: 4 6. Booking for user: 4429093 /ANTHONY 7. Payment type "PAYPAL" 8. Payment transaction id: 557532 9. Booking date: Feb,24 2025 16:00 HRS |
|  | **INSERT** **INTO** BOOKING\_DETAILS (BOOKING\_ID,PROPERTY\_PRICE,BOOKED\_CLEANING\_FEE,EXTRA\_GUEST\_FEE,NIGHTS\_AMOUNT,TOTAL\_GUEST\_AMOUNT,EXTRA\_GUEST\_AMOUNT,BOOKING\_SERVICE\_CHARGE\_PERCENTAGE,BOOKING\_TAX\_CHARGE\_PERCENTAGE,BOOKING\_REGISTRATION\_TIMESTAMP,BOOKING\_START\_DATE,BOOKING\_END\_DATE,CANCEL\_TIMESTAMP)  **VALUES** (  **'ROBL009090'**,  (**SELECT** PRICE **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY = 247543),  (**SELECT** CLEANING\_FEE **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY = 247543),  (**SELECT** EXTRA\_PEOPLE\_COST **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY = 247543),  7,  4,  (4 - (**SELECT** GUEST\_INCLUDED **FROM** AIRBNB.PROPERTIES **WHERE**  ID\_PROPERTY=247543)),  (**SELECT** SERVICE\_FEE  **FROM** AIRBNB.FEES  **INNER** **JOIN** AIRBNB.PROPERTIES  **ON** FEES.PROPERTY\_TYPE = PROPERTIES.ROOM\_TYPE  **WHERE** ID\_PROPERTY = 247543),  (**SELECT** TAX\_FEE  **FROM** AIRBNB.FEES  **WHERE** PROPERTY\_TYPE =  (**SELECT** ROOM\_TYPE  **FROM** AIRBNB.PROPERTIES  **WHERE** ID\_PROPERTY = 247543)),  **'2025-02-24 16:00:00'**,  **'2025-03-05'**,  **'2025-03-11'**,  **NULL**  ); |
|  | **INSERT** **INTO** AIRBNB.BOOKING\_MAIN (BOOKING\_ID, BOOKING\_STATUS, ID\_PROPERTY, ID\_USER, BILL\_ID, SERVICE\_FEE, STAYNIGHTS\_TOTAL\_CHARGE, EXTRA\_GUEST\_CHARGE)  **VALUES**(  **'ROBL009090'**,  **'CONFIRMED'**,  247543,  4429093,  557532,  (**SELECT**  ((NIGHTS\_AMOUNT \* PROPERTY\_PRICE) +  (EXTRA\_GUEST\_FEE \* NIGHTS\_AMOUNT \* EXTRA\_GUEST\_AMOUNT))  \* ((BOOKING\_SERVICE\_CHARGE\_PERCENTAGE / 100))  **FROM**  BOOKING\_DETAILS  **WHERE**  BOOKING\_ID = **'ROBL009090'**),  (**SELECT**  (NIGHTS\_AMOUNT \* PROPERTY\_PRICE)  **FROM**  BOOKING\_DETAILS  **WHERE**  BOOKING\_ID = **'ROBL009090'**),  (**SELECT**  (NIGHTS\_AMOUNT \* EXTRA\_GUEST\_FEE \* EXTRA\_GUEST\_AMOUNT)  **FROM**  BOOKING\_DETAILS  **WHERE**  BOOKING\_ID = **'ROBL009090'**)  ); |
|  | **INSERT** **INTO** AIRBNB.BILLING (BILL\_ID,PAYMENT\_STATUS,PAYMENT\_TYPE,PAYED\_TAXES,TOTAL\_PAYMENT)  **SELECT**  557532,  **'CONFIRMED'**,  **'PAYPAL'**,  (bd.BOOKING\_TAX\_CHARGE\_PERCENTAGE / 100) \* (bm.SERVICE\_FEE +  bm.STAYNIGHTS\_TOTAL\_CHARGE + bm.EXTRA\_GUEST\_CHARGE),  (bm.SERVICE\_FEE + bm.STAYNIGHTS\_TOTAL\_CHARGE + bm.EXTRA\_GUEST\_CHARGE +  bd.BOOKED\_CLEANING\_FEE)  **FROM**  AIRBNB.BOOKING\_MAIN bm  **JOIN**  AIRBNB.BOOKING\_DETAILS bd **ON** bm.BOOKING\_ID = bd.BOOKING\_ID  **WHERE**  bm.BOOKING\_ID = **'ROBL009090'**; |
|  | **UPDATE** AIRBNB.CALENDAR  **SET** AVAILABLE\_FLAG = 0  **WHERE** BOOKDATE **BETWEEN**  (**SELECT** BOOKING\_START\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'ROBL009090'**)  **AND** (**SELECT** BOOKING\_END\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'ROBL009090'**); |

|  |  |
| --- | --- |
| 5 | Implement a Stay Complaint as ebnb\_Complaint\_1 (ANNEX 6) |
|  | **INSERT** **INTO** AIRBNB.COMPLAINTS(COMPLAINT\_ID, BOOKING\_ID, USER\_ID, COMPLAINT\_STATUS, COMPLAINT\_TIMESTAMP, COMMENTS)  **VALUES**(1,**'ROBL009090'**,4429093,**'Open'**, **'2025-04-12 10:00:02'**, **'They opened me 3 hours late, and at checkout time they were pressuring me to leave 2 hours earlier than agreed. It was a very bad experience, I demand a partial refund'**) |

|  |  |
| --- | --- |
| 6 | Register a Review as below:   * ID booking: ROBL009090 * property id: 247543 * Booking Registered Dates: March 15, 2025 18:08HRS * Booked by user: 4429093/Anthony * rate: 2 * comments: It was not nice experience, too much noise and mold. |
|  | **INSERT** **INTO** AIRBNB.REVIEWS (REVIEW\_ID, ID\_PROPERTY, REVIEW\_DATE, REVIEWER\_ID, REVIEWER\_NAME, RATE, BOOKING\_ID, COMMENTS)  **VALUES** (38,247543, **'2025-03-15 18:08:00'**, 4429093, **'Anthony'**, 2, **'ROBL009090'**, **'It was not nice experience, too much noise and mold'**) |
|  | **UPDATE** AIRBNB.PROPERTIES  **SET** NUMBER\_OFF\_REVIEWS = (  **SELECT** **AVG**(rate)  **FROM** REVIEWS  **WHERE** ID\_PROPERTY = 247543)  **WHERE** ID\_PROPERTY = 247543; |
|  | **UPDATE** AIRBNB.PROPERTIES  **SET** REVIEW\_SCORE\_VALUE = REVIEW\_SCORE\_VALUE + 1  **WHERE** ID\_PROPERTY = 247543; |

|  |  |
| --- | --- |
| 7 | Register a Reservation as below:   1. ID booking: WEXE0PYQQQ8  * property id: 291804/ A nice room with great location  1. total nights: 2 2. Booking Dates: from 2025-04-01 al 2025-04-02 3. Amount of people:2  * Booking for user: 2676606Ethan Davis  1. Payment type "Google Wallet" 2. Payment transaction id: 449021 3. Booking date: March,20 2025 19:00 HRS |
|  | **INSERT** **INTO** BOOKING\_DETAILS (BOOKING\_ID,PROPERTY\_PRICE,BOOKED\_CLEANING\_FEE,EXTRA\_GUEST\_FEE,NIGHTS\_AMOUNT,TOTAL\_GUEST\_AMOUNT,EXTRA\_GUEST\_AMOUNT,BOOKING\_SERVICE\_CHARGE\_PERCENTAGE,BOOKING\_TAX\_CHARGE\_PERCENTAGE,BOOKING\_REGISTRATION\_TIMESTAMP,BOOKING\_START\_DATE,BOOKING\_END\_DATE,CANCEL\_TIMESTAMP)  **VALUES** (  **'WEXE0PYQQQ8'**,  (**SELECT** PRICE **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY = 291804),  (**SELECT** CLEANING\_FEE **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY = 291804),  (**SELECT** EXTRA\_PEOPLE\_COST **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY = 291804),  2,  2,  0,  (**SELECT** SERVICE\_FEE  **FROM** AIRBNB.FEES  **INNER** **JOIN** AIRBNB.PROPERTIES  **ON** FEES.PROPERTY\_TYPE = PROPERTIES.ROOM\_TYPE  **WHERE** ID\_PROPERTY = 291804),  (**SELECT** TAX\_FEE  **FROM** AIRBNB.FEES  **WHERE** PROPERTY\_TYPE =  (**SELECT** ROOM\_TYPE  **FROM** AIRBNB.PROPERTIES  **WHERE** ID\_PROPERTY = 291804)),  **'2025-03-20 19:00:00'**,  **'2025-04-01'**,  **'2025-04-02'**,  **NULL**  ); |
|  | **INSERT** **INTO** AIRBNB.BOOKING\_MAIN (BOOKING\_ID, BOOKING\_STATUS, ID\_PROPERTY, ID\_USER, BILL\_ID, SERVICE\_FEE, STAYNIGHTS\_TOTAL\_CHARGE, EXTRA\_GUEST\_CHARGE)  **VALUES**(  **'WEXE0PYQQQ8'**,  **'CONFIRMED'**,  291804,  2676606,  449021,  (**SELECT**  ((NIGHTS\_AMOUNT \* PROPERTY\_PRICE) +  (EXTRA\_GUEST\_FEE \* NIGHTS\_AMOUNT \* EXTRA\_GUEST\_AMOUNT))  \* ((BOOKING\_SERVICE\_CHARGE\_PERCENTAGE / 100))  **FROM**  BOOKING\_DETAILS  **WHERE**  BOOKING\_ID = **'WEXE0PYQQQ8'**),  (**SELECT**  (NIGHTS\_AMOUNT \* PROPERTY\_PRICE)  **FROM**  BOOKING\_DETAILS  **WHERE**  BOOKING\_ID = **'WEXE0PYQQQ8'**),  (**SELECT**  (NIGHTS\_AMOUNT \* EXTRA\_GUEST\_FEE \* EXTRA\_GUEST\_AMOUNT)  **FROM**  BOOKING\_DETAILS  **WHERE**  BOOKING\_ID = **'WEXE0PYQQQ8'**)  ); |
|  | **INSERT** **INTO** AIRBNB.BILLING (BILL\_ID,PAYMENT\_STATUS,PAYMENT\_TYPE,PAYED\_TAXES,TOTAL\_PAYMENT)  **SELECT**  449021,  **'CONFIRMED'**,  **'GOOGLE WALLET'**,  (bd.BOOKING\_TAX\_CHARGE\_PERCENTAGE / 100) \* (bm.SERVICE\_FEE +  bm.STAYNIGHTS\_TOTAL\_CHARGE + bm.EXTRA\_GUEST\_CHARGE),  (bm.SERVICE\_FEE + bm.STAYNIGHTS\_TOTAL\_CHARGE + bm.EXTRA\_GUEST\_CHARGE +  bd.BOOKED\_CLEANING\_FEE)  **FROM**  AIRBNB.BOOKING\_MAIN bm  **JOIN**  AIRBNB.BOOKING\_DETAILS bd **ON** bm.BOOKING\_ID = bd.BOOKING\_ID  **WHERE**  bm.BOOKING\_ID = **'WEXE0PYQQQ8'**; |
|  | **UPDATE** AIRBNB.CALENDAR  **SET** AVAILABLE\_FLAG = 0  **WHERE** BOOKDATE **BETWEEN**  (**SELECT** BOOKING\_START\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'WEXE0PYQQQ8'**)  **AND** (**SELECT** BOOKING\_END\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'WEXE0PYQQQ8'**); |

|  |  |
| --- | --- |
| 8 | Register a Reservation Cancelation as below:   * ID booking: WEXE0PYQQQ8 |
|  | **UPDATE** AIRBNB.BOOKING\_DETAILS **SET** CANCEL\_TIMESTAMP = **'2025-03-25'** **WHERE** BOOKING\_ID =**'WEXE0PYQQQ8'**; |
|  | **UPDATE** AIRBNB.BOOKING\_MAIN **SET** BOOKING\_STATUS = **'CANCELLED'** **WHERE** BOOKING\_ID =**'WEXE0PYQQQ8'**; |
|  | **UPDATE** AIRBNB.BILLING **SET** PAYMENT\_STATUS = **'REFUND'** **WHERE** BILL\_ID =**'449021'**; |
|  | **UPDATE** AIRBNB.CALENDAR  **SET** AVAILABLE\_FLAG = 1  **WHERE** BOOKDATE **BETWEEN**  (**SELECT** BOOKING\_START\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'WEXE0PYQQQ8'**)  **AND** (**SELECT** BOOKING\_END\_DATE **FROM** AIRBNB.BOOKING\_DETAILS **WHERE** BOOKING\_ID = **'WEXE0PYQQQ8'**); |

|  |  |
| --- | --- |
| 9 | Register a Reservation Review as below:   * property id: 291804 * User, 2676606 Ethan * ID booking WEXE0PYQQQ8 * rate: 5 * REVIEW DATE: 2025-04-20 * comments: This is a very large house and great for groups. We found the location difficult for our purposes of sightseeing in Mexico City. It is a nice quiet neighborhood and we had great sleep. However, car service costs added up for us as we had older people and a baby in our group. The nicer amenities of the home included comfortable beds, quiet nights, cleaning services, the garden and a nice book collection. |
|  | **SELECT** \* **FROM** AIRBNB.REVIEWS **ORDER** **BY** REVIEW\_ID;  **INSERT** **INTO** AIRBNB.REVIEWS  **SELECT**  **MAX**(REVIEW\_ID) + 1, 291804, **'2025-04-20'**, 2676606, **'Ethan'**, 5, **'WEXE0PYQQQ8'**, **'This is a very large house and great for groups. We found the location difficult for our purposes of sightseeing in Mexico City. It is a nice quiet neighborhood and we had great sleep. However, car service costs added up for us as we had older people and a baby in our group. The nicer amenities of the home included comfortable beds, quiet nights, cleaning services, the garden and a nice book collection.'**  **FROM**  AIRBNB.REVIEWS; |
|  | **SELECT** \* **FROM** AIRBNB.PROPERTIES **WHERE** ID\_PROPERTY =291804;  **UPDATE** AIRBNB.PROPERTIES  **SET** REVIEW\_SCORE\_VALUE = (  **SELECT** **AVG**(rate)  **FROM** AIRBNB.REVIEWS  **WHERE** ID\_PROPERTY = 291804)  **WHERE** ID\_PROPERTY = 291804; |
|  | **UPDATE** AIRBNB.PROPERTIES  **SET** NUMBER\_OFF\_REVIEWS = NUMBER\_OFF\_REVIEWS + 1  **WHERE** ID\_PROPERTY = 291804; |

# 6 BASIC QUERY DESIGN – SQL SELECT

Create a SQL Statement that resolve each problem below.

|  |  |
| --- | --- |
| QUERY 1 | Extract all properties available for rent.  List columns as property id, property name, location State, price, rate score value, show Zero(“0”) in each row where the property has NULL value in the "REVIEW\_SCORES\_VALUE" column. |
| SQL solution: | **SELECT**  ID\_PROPERTY,  PROPERTY\_NAME,  PROPERTYLOCATION\_STATE,  PRICE,  **COALESCE**(REVIEW\_SCORE\_VALUE, 0) **AS** *REVIEW\_SCORE\_VALUE*  **FROM**  PROPERTIES *p*; |

|  |  |
| --- | --- |
| QUERY 2 | Extract all properties details and their host name.  List columns as Properties Name, Property Price and host name, so we can see the name's owner for the property. |
| SQL solution: | **SELECT**  *p*.PROPERTY\_NAME,  *p*.PRICE,  *h*.HOST\_NAME  **FROM**  PROPERTIES *p*  **JOIN**  PROPERTIES\_OWNERS *po* **ON** *p*.ID\_PROPERTY = *po*.ID\_PROPERTY  **JOIN**  HOSTS *h* **ON** *po*.HOST\_ID = *h*.HOST\_ID; |

|  |  |
| --- | --- |
| QUERY 3 | Extract all Reservations of any status and the total payment for each one.  List data as booking number, booking status, booking start date, TOTAL\_PAYMENT,  and shows the most recent date first |
| SQL solution: | **SELECT** *bd*.***BOOKING\_ID***, *bm*.***BOOKING\_STATUS***, *bd*.***BOOKING\_START\_DATE***,  *b*.***TOTAL\_PAYMENT***  **FROM** ***BOOKING\_DETAILS*** *bd*  **JOIN** ***BOOKING\_MAIN*** *bm*  **ON** *bd*.***BOOKING\_ID*** = *bm*.***BOOKING\_ID***  **JOIN** ***BILLING*** *b*  **ON** *bm*.***BILL\_ID*** = *b*.***BILL\_ID***  **ORDER** **BY** BOOKING\_START\_DATE **DESC**; |

|  |  |
| --- | --- |
| QUERY 4 | Extract all Confirmed Reservations and the total payment for each one?  List data as the booking number, booking start date, TOTAL\_PAYMENT and shows the most recent date first. |
| SQL solution: | **SELECT**  bd.BOOKING\_ID,  bd.BOOKING\_START\_DATE,  b.TOTAL\_PAYMENT  **FROM**  BOOKING\_DETAILS bd  **JOIN**  BOOKING\_MAIN bm **ON** bd.BOOKING\_ID = bm.BOOKING\_ID  **JOIN**  BILLING b **ON** bm.BILL\_ID = b.BILL\_ID  **WHERE**  bm.BOOKING\_STATUS = **'CONFIRMED'**  **ORDER** **BY**  bd.BOOKING\_START\_DATE **DESC**; |

|  |  |
| --- | --- |
| QUERY 5 | Extract all users are also hosts.  *List* as user id, legal name, email, verifications, and the date since is a host.  sort by most recent date first. |
| SQL solution: | **SELECT**  *u*.USER\_ID,  *u*.USER\_LEGAL\_NAME,  *u*.USER\_EMAIL,  *u*.VERIFICATIONS,  *h*.HOST\_SINCE  **FROM**  USERS *u*  **JOIN**  HOSTS *h* **ON** *u*.USER\_ID = *h*.USER\_ID  **ORDER** **BY**  *h*.HOST\_SINCE **DESC**; |

|  |  |
| --- | --- |
| QUERY 6 | How many booked reservations has properties with price greater than 900 USD?  List data as booking number, property number, property name and its price. |
| SQL solution: | **SELECT** BOOKING\_ID, PROPERTY\_PRICE  **FROM** BOOKING\_DETAILS  **WHERE** PROPERTY\_PRICE > 900; |

|  |  |
| --- | --- |
| QUERY 7 | Extract all reservations have paid above average reservation service fees?  List data as reservation number, and its service fee. |
| SQL solution: | **SELECT** BOOKING\_ID, SERVICE\_FEE  **FROM** BOOKING\_MAIN  **WHERE** SERVICE\_FEE > (**SELECT** **AVG**(SERVICE\_FEE) **FROM** BOOKING\_MAIN); |

|  |  |
| --- | --- |
| QUERY 8 | Extract all properties have open complaints registered.  List the complaint number, property name, month number it was opened, and its comments. |
| SQL solution: | **SELECT** c.COMPLAINT\_ID, c.BOOKING\_ID, p.PROPERTY\_NAME, c.COMMENTS, **EXTRACT** (**MONTH** **FROM** c.COMPLAINT\_TIMESTAMP) **AS** COMPLAINT\_MONTH  **FROM** COMPLAINTS c  **JOIN** BOOKING\_MAIN bm **ON** c.BOOKING\_ID = bm.BOOKING\_ID  **JOIN** PROPERTIES p **ON** bm.ID\_PROPERTY = p.ID\_PROPERTY  **WHERE** c.COMPLAINT\_STATUS = **'Open'**; |

# 7. STATISTICAL REPORTS: ADVANCED SQL

Create an SQL statement to resolve each of the reports requested by the client.

|  |  |
| --- | --- |
| STATISTICAL REPORT 1 | How many reservations are there for the room types?  *List* *columns as expected output* |
| SQL solution: | **SELECT** ROOM\_TYPE,  **COUNT**(BOOKING\_ID)*"Reservations by room type"*  **FROM** PROPERTIES *p*  **JOIN** BOOKING\_MAIN *bm*  **ON** *P*.ID\_PROPERTY = *BM*.ID\_PROPERTY  **GROUP** **BY** ROOM\_TYPE; |

|  |  |
| --- | --- |
| STATISTICAL REPORT 2 | Extract the top two most common payment methods, it also shows the total payments you have.  *List columns as expected output* |
| SQL solution: | **SELECT** PAYMENT\_TYPE,  **COUNT**(PAYMENT\_TYPE) **AS** *"Total Payments"*  **FROM** BILLING *b*  **GROUP** **BY** PAYMENT\_TYPE  **FETCH** **FIRST** 2 **ROWS** **ONLY**; |

|  |  |
| --- | --- |
| STATISTICAL REPORT 3 | What is the total payment sum for each recorded payment status, showing first the largers "SUM OF TOTAL PAYMENTS BY STATUS"  *List columns as expected output* |
| SQL solution: | **SELECT** PAYMENT\_STATUS,  **SUM**(TOTAL\_PAYMENT) *"Sum of total payments by Status"*  **FROM** BILLING *b*  **GROUP** **BY** PAYMENT\_STATUS  **ORDER** **BY** **SUM**(TOTAL\_PAYMENT) **DESC**; |

|  |  |
| --- | --- |
| STATISTICAL REPORT 4 | List the number of Cancelled and/or Confirmed reservations as of today  *List columns as expected output* |
| SQL solution: | **SELECT** BOOKING\_STATUS,  **COUNT**(BOOKING\_STATUS) *"Quantity"*  **FROM** BOOKING\_MAIN *bm*  **GROUP** **BY** BOOKING\_STATUS; |

|  |  |
| --- | --- |
| STATISTICAL REPORT 5 | List how many properties are available in each city.  *List columns as expected output* |
| SQL solution: | **SELECT** PROPERTYLOCATION\_CITY,  **COUNT**(ID\_PROPERTY) *"Quantity"*  **FROM** PROPERTIES *p*  **GROUP** **BY** PROPERTYLOCATION\_CITY  **ORDER** **BY** **COUNT**(ID\_PROPERTY) **DESC**; |

|  |  |
| --- | --- |
| STATISTICAL REPORT 6 | List how many properties are available in each state.  *List columns as expected output* |
| SQL solution: | **SELECT** PROPERTYLOCATION\_STATE,  **COUNT**(PROPERTYLOCATION\_STATE) *"Quantity"*  **FROM** PROPERTIES *p*  **GROUP** **BY** PROPERTYLOCATION\_STATE; |

|  |  |
| --- | --- |
| STATISTICAL REPORT 7 | What is the average CLEANING FEE for each type of property (PROPERTY\_TYPE)  *List columns as expected output* |
| SQL solution: | **SELECT** ROOM\_TYPE,  **AVG**(CLEANING\_FEE) *"Average Cleaning Fee"*  **FROM** PROPERTIES *p*  **GROUP** **BY** ROOM\_TYPE  **ORDER** **BY** **AVG**(CLEANING\_FEE) **DESC**; |

|  |  |
| --- | --- |
| STATISTICAL REPORT 8 | What is the minimum and maximum price of each type of property  *List columns as expected output* |
| SQL solution: | **SELECT**  ROOM\_TYPE,  **MIN**((MIN\_NIGHTS\*PRICE) + CLEANING\_FEE) **AS** *MINIMUM\_PRICE*,  **MAX**((MAX\_NIGHT\*PRICE)+ CLEANING\_FEE + (GUEST\_INCLUDED\*EXTRA\_PEOPLE\_COST)) **AS** *MAXIMUM\_PRICE*  **FROM** PROPERTIES  **GROUP** **BY** ROOM\_TYPE |

|  |  |
| --- | --- |
| STATISTICAL REPORT 9 | How many users registered per month on the platform?  *List columns as expected output* |
| SQL solution: | **SELECT**  **EXTRACT** (**YEAR** **FROM** USER\_MEMBER\_SINCE) **AS** YEAR\_JOINED,  **COUNT**(\*) **AS** USER\_COUNT  **FROM** USERS  **GROUP** **BY** **EXTRACT** (**YEAR** **FROM** USER\_MEMBER\_SINCE)  **ORDER** **BY** YEAR\_JOINED; |

|  |  |
| --- | --- |
| STATISTICAL REPORT 10 | Lists the number of available and BOOKED dates for each property. Shows the legend "AVAILABLE" and "BOOKED" as the case may be instead of the 1/0 in the AVAILABLE\_FLAG column.  *List columns as expected output* |
| SQL solution: | **SELECT**  ID\_PROPERTY,  **CASE**  **WHEN** AVAILABLE\_FLAG = 0 **THEN** **'AVAILABLE'**  **ELSE** **'BOOKED'**  **END** **AS** *"Availability"*,  **COUNT**(**CASE** **WHEN** AVAILABLE\_FLAG = 0 **THEN** 1 **ELSE** 0 **END**) *"Available Days"*  **FROM** CALENDAR *c*  **GROUP** **BY** ID\_PROPERTY, AVAILABLE\_FLAG; |

# 8. CONCLUSIONS

In this project, we gained much more than just a few new skills. It was an intensive experience that required hard work and effective organization to get everything done.

We began by learning the basics, starting with the Extended Entity Relationship Diagram (EERD). Using these diagrams, we mapped out the initial steps of the project and established relationships within our raw data.

One of the most critical and challenging topics we tackled was the Three Normal Forms (3NF). This was a crucial step in "cleaning" our raw data, and learning how to properly apply all three normal forms took considerable effort. With this foundation in place, we were ready to dive into SQL, where we applied a variety of techniques learned in class to write efficient code and complete the required tasks.

Although these weeks were incredibly challenging, the experience was rewarding. Not only did we master the concepts mentioned, but we also gained many other valuable insights along the way, making this project a truly enriching learning experience.

# 9. RECOMMENDATIONS AND REFLEXIONS

1. We found the exercises in class very useful; we recommend continuing with them, we were able to practice with the teacher and ask her questions as they came up.
2. We recommend scheduling some class time for working on the project, rather than doing it all at home. In our opinion, it was a challenging project, and at times, several questions arose that weren't always addressed immediately, and it was hard for us to reach out at times.
3. We would recommend making the instructions for homework and projects more concise, as it took us some time to fully understand them on the first read-through.

# 10. BIBLIOGRAPHICAL SOURCES

* Simplilearn. (2024, 23 julio). *The Ultimate Guide on SQL Basics*. Simplilearn.com. <https://www.simplilearn.com/tutorials/sql-tutorial/what-is-sql>

# 11 . ANNEX

#### annex 1

|  |  |  |
| --- | --- | --- |
|  | TABLE.COLUMN | COMPUTE |
| 1 | **BOOKING\_DETAILS.PROPERTY\_PRICE** | Shall be pulled from PROPERTIES.PRICE |
| 2 | **BOOKING\_DETAILS.CLEANING\_FEE** | Shall be pulled from PROPERTIES.CLEANING\_FEE |
| 3 | **BOOKING\_DETAILS. EXTRA\_GUEST\_AMOUNT** | shall calculated based on booking TOTAL\_GUEST\_AMOUNT minus PROPERTY.GUEST\_INCLUDED. |
| 4 | **BOOKING\_DETAILS. TOTAL\_GUEST\_AMOUNT** | Shall be pulled from PROPERTIES.EXTRA\_PEOPLE\_COST |
| 5 | **BOOKING\_DETAILS.**  **BOOKING\_TAX\_CHARGE\_PERCENTAGE** | Shall be taken from FEES table, pull TAX\_FEE filtering on the PROPERTY\_TYPE |
| 6 | **CALENDAR.BOOKED\_DATE** | If the booked dates are not already in the CALEDAR TABLE for the corresponding ID\_PROPERTY it shall be INSERTED into CALENDAR. |
| 7 | **BOOKING\_MAIN.SERVICE\_FEE** | shall be calculated based on:(NIGHTS\_AMOUNT x PROPERTY\_PRICE) + EXTRA\_GUEST\_CHARGE |
| 8 | **BOOKING\_MAIN. STAYNIGHTS\_TOTAL\_CHARGE\_** | the STAYNIGHTS\_TOTAL\_CHARGE, shall be computed based on NIGHTS\_AMOUNT X PROPERTY\_PRICE |
| 9 | **BOOKING\_MAIN.EXTRA\_GUEST\_CHARGE** | shall consider, BOOKING\_DETAILS.EXTRA\_GUEST\_AMOUNT, x BOOKING\_DETAILS.TOTAL\_GUEST\_AMOUNT x and BOOKING\_DETAILS.NIGHTS\_AMOUNT |
| 10 | **BILLING.PAYED\_TAXES** | shall be computed based on BOOKING\_TAX\_CHARGE\_PERCENTAGE X (SERVICE\_FEE +STAYNIGHTS\_TOTAL\_CHARGE\_ + EXTRA\_GUEST\_CHARGE) |
| 11 | **BILLING.TOTAL\_PAYMENT** | shall be calculated based on  + BOOKING\_MAIN.SERVICE\_FEE  + BOOKING\_MAIN.STAYNIGHTS\_TOTAL\_CHARGE  + BOOKING\_MAIN.EXTRA\_GUEST\_CHARGE  + BOOKING\_DETAILS.BOOKED\_CLEANING\_FEE  + BILLING.PAYED\_TAXES |

#### annex 2

#### A screenshot of a phone Description automatically generated

#### annex 3

A screenshot of a hotel ticket

Description automatically generated

#### annex 4

A screenshot of a email

Description automatically generated

A screenshot of a ticket

Description automatically generated

#### annex 5

A screenshot of a hotel review

Description automatically generated

#### annex 6

A screenshot of a phone

Description automatically generated

#### annex 7

HOST PROFILE

A screenshot of a personal profile

Description automatically generated

#### annex 8

USER PROFILE

A screenshot of a computer

Description automatically generated

#### annex N