

Contents

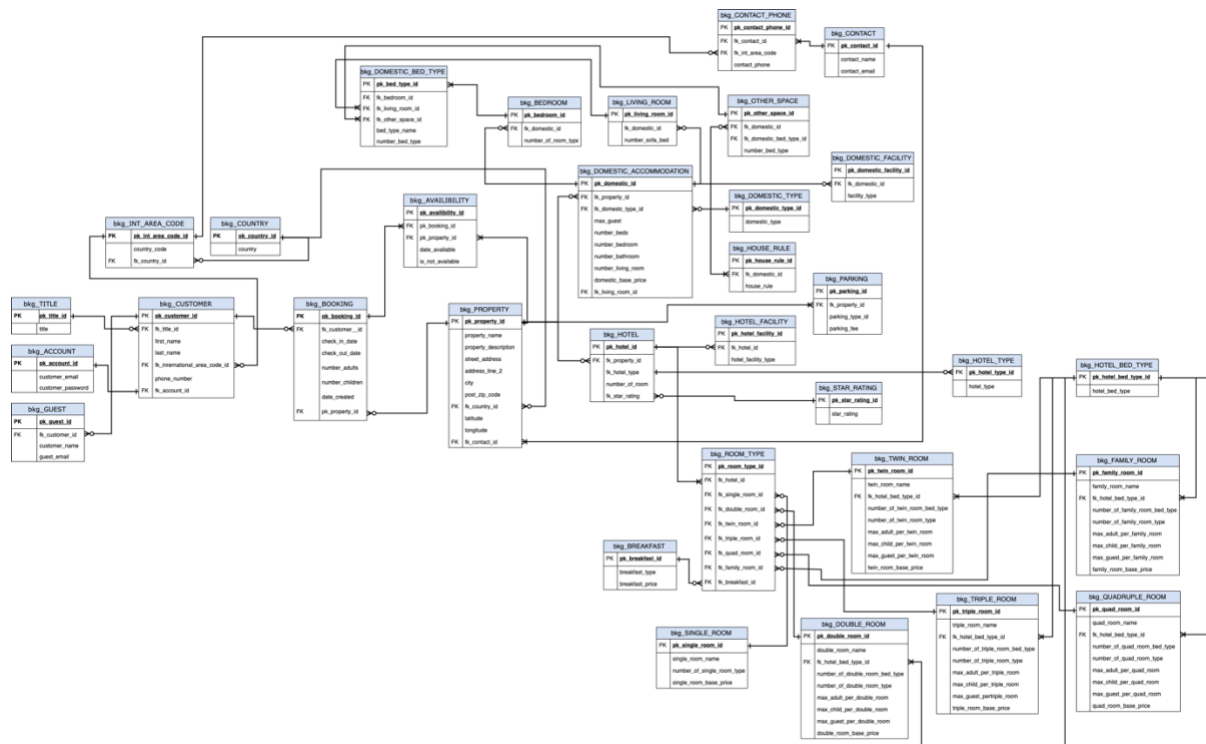
Contents	1
1. Introduction	2
Figure 1. Initial Team ER Diagram Design	2
1.1 <i>Assumptions and Scope</i>	2
2. Database Design: Overview and Functionality	3
Figure 2. Final ER Diagram Design	5
2.1 <i>Design Decisions</i>	5
2.2 <i>Keys</i>	6
2.3 <i>Data Types</i>	6
Figure 3 & 4. Data Types & Key/Non-Key Attributes	7
2.4 <i>Normalisation</i>	8
Figure 5. Example of Second Normal Form	8
Figure 6. Examples of Third Normal Form	8
3. Business Intelligence	9
Figure 7 Earnings vs Potential Earnings	9
4. Improvements	10
5. Conclusion	10
APPENDIX	11
Figure 1.1 Entity Relationship Diagram	11
Figure 2.1 Data Dictionary	11
Figure 3.1/4.1 Data Types & Key/Non-Key Attributes Data tables	19
Figure 7.1 Earnings Vs Potential Earnings Example Query	19
DATABASE VIVA	21
1. <i>Customer search for a property, book and pay for their stay</i>	21
2. <i>Delete a customer record</i>	28
3. <i>Delete and update a property record</i>	29

1. Introduction

The purpose of this project, report and VIVA was to develop a relational database which mirrored the operational aspects of Booking.com (<https://www.booking.com/>). Primarily, attempting to limit redundant data and inconsistent dependencies to allow for flexibility and development of the database with reference to the first principles of Normalisation.

The initial approach taken to gain a better understanding of the functionality, relationships and types of data that would need to be stored in the database, was to perform a thorough investigation of the Booking.com websites user interface(UI) and incorporate any prior knowledge each design member had of the site. From conception, we split these areas into three subsidiaries: Customer, Booking and Properties to perform attribute and entity discovery as a starting reference to aid in our initial Entity Relationship diagram which is detailed in [Figure 1](#).

Figure 1. Initial Team ER Diagram Design



1.1 Assumptions and Scope

Given the magnitude of Booking.com, initial assumptions were made to limit the scope of the project;

Out of scope

- Any section that takes the user away from the site (including flights, flights + hotels, car rentals, tours & activities and airport taxis).

- Guest reviews are to be considered out of scope

And aid in the design and functionality of the database;

Assumptions

- Language selection of the site would be limited to English
- One Currency will be used, in this instance GBP (£)
- A *selection* of data related to filter types would be sufficient in representing their functionality and purpose in the database
- An application will manage the amount of rooms available on a given day and prevent these being shown to the customer if unavailable.

2. Database Design: Overview and Functionality

The database has been developed to aid the functionality of the UI on the Booking.com website. The database facilitates the storage of data for users, including accounts and payment information, property listings with photos, available rooms and related facilities and amenities, booking details, related guest information and payment credentials and related user data. It aims to provide useful, relevant and up to date information in assisting a customer making, updating and cancelling a booking, giving businesses dynamic control of pricing and availability of rooms, and assisting them in making future business decisions on relevant and purposeful data.

A user can create an account to either list a property or book a property by providing their email address and a password (which is AES Encrypted). The user account can either be linked to a customer or a property contact, dependent on which will determine what data would be required to be store about the user.

A booking can be made by a customer which can be for one or more rooms at a property. The number of adults and children per booking are stored to aid in providing relevant rooms that are available (It is assumed an application would determine what rooms would be available to display to the user) to book. A user may be travelling for business purposes which can be recorded, to aid in providing suitable properties when searching. A customer can store their card details to streamline the booking process when completing a booking. A customer can have many card types. Note: CVV numbers are not stored in the database in order to aid customer data protection. CVV numbers would be required through Booking.com UI at each payment attempt. Customer card numbers are protected with AES Encryption.

Details about a booking are stored in the booking details which store the details relating to the room in which they have booked, the number of adults and children booked, and check in and check out date. Guest details are stored against the booking detail of a booking to provide detail to the property where booked, who will be staying with them. A customer who creates a booking can book themselves as a guest or for another person. Transaction details are stored against a booking including last four digits, payment amount and payment date. Booking.com allows customers numerous options for customers to pay. For the purpose of this project, the customer would pay at the time of booking (It is assumed that

transaction details would only be stored against a booking once this action has been completed).

One or many properties can be listed by a user when they have created an account (it is assumed that an application would determine the purpose of an account at creation, whether a user has selected 'list a property' or 'register') A user that lists a property need only provide their full name and contact number. A property contact can have more than one contact phone number.

A variety of properties can be listed on Booking.com which is represented by the accommodation type table. A commission fee is charged by Booking.com every month to each property which is represented by the commission table. Dependant on the property type selected, a star rating may be applied (It is assumed that an application would only allow a star rating to be selected for relevant properties, in this case a hotel). Details about the property must be added including property address and a minimum of one photo.

Each property has the option to offer facilities and services, including parking, internet and languages spoken for international travellers. These details are related to each property through the accommodation details table. Each property can offer a cancellation policy which is also represented in the accommodation detail table.

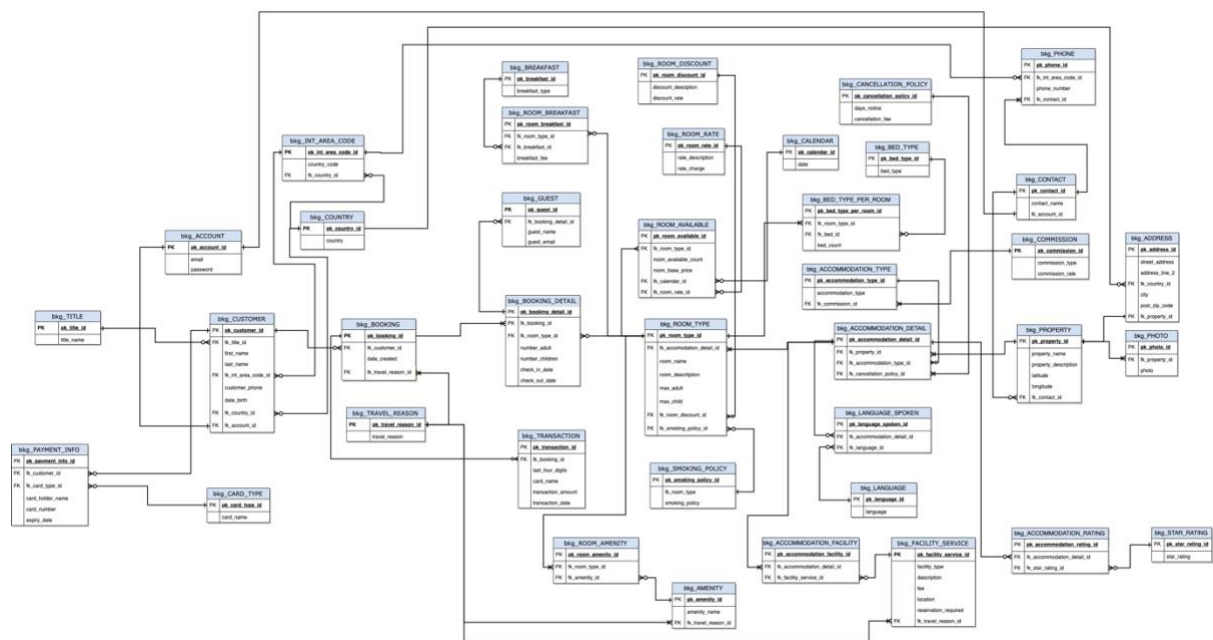
A property can have many different room types. The room type has a max adult and max child limit to avoid overbooking (It is assumed that an application would only allow a room to be booked in a booking if the total number of adults and children per booking detail did not exceed the total amount of adults and children a room can accommodate). A room can apply a discount if the property decides to allow this. Each room can either have a smoking or non-smoking policy applied.

Each room will have one or many beds, or a variety of different bed types which is represented in the beds per room type table. Each room can have a host of amenities that are specific to each room and one or many breakfast options available to each room. Each breakfast can apply a fee.

A property can have multiple rooms of the same type, that can be available on different dates, which is represented by the room available count and calendar id (which relates to the calendar table) in the room available table. A room will have a base price and can have a rate applied to it if a property is aware of an event over a particular time period. A rate can be applied to a room on an individual date basis. This is represented in the room rate table.

The diagram in [Figure 2](#) was used to develop the database in MySQL. An entity relationship diagram can be found in the appendix ([Figure 1.1 Entity Relationship Diagram](#)). Rather than modelling the database on all business views of Booking.com, the database was created to facilitate the core functionality of the website, from creating an account, listing a property, making a booking and completing a transaction. This approach was taken to gain experience in database design and implementation and allow focus and attention to be centred around this core functionality.

Figure 2. Final ER Diagram Design



2.1 Design Decisions

Design decisions were made in every iteration of the design phase, development and implementation of the database. After the initial ER diagram, the decision was made to de-normalise the design and clarify relationships between each entity (for example, a customer can book a room type in a property rather than an entire property). The introduction of the booking detail table allowed a customer to book many rooms under one booking. This decision was key, as it allowed facts about the booking to be related to the booking itself (guest information, room type, check in and out dates).

Progressing through the database design, adhering to the normalisation principles, a key addition of the room available table was discovered. This allowed for granularity in room selection, and the option to specify a price per room per day (The base price attribute was initially located in the room type entity, but this only allowed for a room type to be priced the same no matter what date). As this developed, room rate and room discount were incorporated to allow for further control of room pricing.

When defining the relationships between each table, it was made apparent that there were incorrectly defined relationships between tables such as, the facility service table and the accommodation table. This didn't allow for specificity to a particular accommodation type for example, and if an accommodation type record were to be deleted, it would delete the facility linked to this, and remove this facility linked to any other property, causing unreliable data, and thus, losing data integrity. This was a many to many relationship, and with the introduction of the accommodation facility table, allowed for removal of records without removal of required data that could belong to other accommodation records. The same relationships were defined with the introduction of the room amenity, room breakfast and bed type per room tables, which allowed a room type and all its related information to be

removed without affecting other records with related data, reducing repeated data and limiting redundancy.

Defining the actions upon update and delete in the relationships between each table allowed for a customer for example, if they desired to be removed completely, to remove all information related to them, without removing related booking, transaction or guest information, and similarly with a property. These actions were pivotal in ensuring that the booking details remained intact. With the introduction of tables such as, bed type per room and room amenity, this allowed for the database to be easily maintained, as any *NULL* values in these tables (other than the booking, and booking details table) represented already deleted related records and could be easily removed.

Security was considered throughout, with sensitive data incorporating AES Encryption on these attribute fields (i.e. card number in bkg_PAYMENT_INFO table and password in bkg_ACCOUNT table) to protect customer data, and protect the integrity and reputation of the business.

2.2 Keys

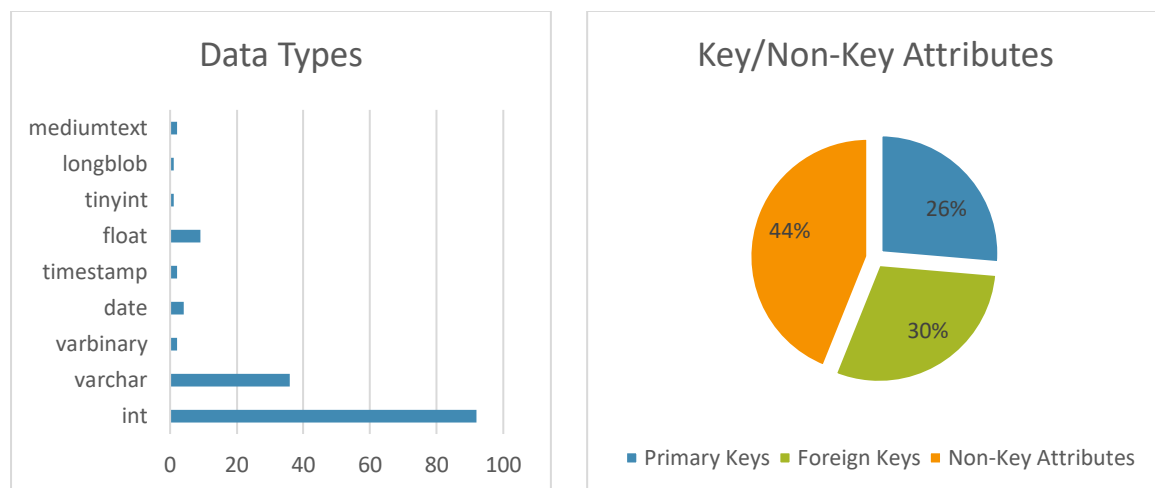
Each entity has a primary key used to uniquely identify the detail related to each record. Each primary key in the database is a data type int(10) auto increment to allow for easy creation of records, avoiding repetition and maintain uniqueness.

pk_booking_id is a primary key used in bkg_BOOKING table to uniquely identify a booking created by a customer on a given date, and to link any details relating to this particular booking. fk_booking_id in bkg_BOOKING_DETAIL table is a foreign key related to the pk_booking_id in bkg_BOOKING that allows these details to be related. By applying these foreign key constraints, data can be easily and correctly maintained, limiting data redundancy and maintaining data integrity. By defining the actions this added further control in maintaining data integrity which was discussed in 2.1 Design Decisions.

2.3 Data Types

A data dictionary can be found in the appendix ([Figure 2.1 Data Dictionary](#)) detailing all data related to each table, attribute, data type, default values, primary and foreign keys (identified as PK or FK in the KEYS column) , relation and actions between each table and comments with any further detail about an attribute. Charts have been included to depict the amount of data types used in the database and the frequency of primary, foreign and non-key attributes.

Figure 3 & 4. Data Types & Key/Non-Key Attributes



Ints are the most predominantly used data types as a large proportion of the attributes were either a primary or foreign key, allowing relationships between entities to be easily created. The int data type was used for all primary keys as it could be easily auto incremented allowing for easy creation and maintenance of data records. For the likes of star rating, we limited the int size to 1 as it wasn't necessary to allow for any more than this (star ratings typically range from 1 – 5 so there was no need to accommodate a 10 digit number). Similarly, tinyint was used for reservation required to represent a Boolean option.

The varchar data type was used second to int, as this allowed for descriptive data for attributes of each entity, for example, room type. The use of mediumtext was used for attributes like room description and property description as varchar may have become restrictive if a description was to be longer than the data type could accommodate, meaning relevant data could be cut short without realising. The varbinary datatype was used to allow for AES Encrypted fields such as password and card number to facility security protocols and protect customer data.

The decimal data type float was used with attributes that stored pricing information, including base price per room, rate charge and transaction amount.

The longblob data type was used to allow properties the functionality to store photos depicting their property and to help aid in a customer's decision to book with them.

Time related data types were key in ensuring bookings could be made for a specific date (check in and check out) and that rooms were available. This is apparent in the calendar table, which was introduced to limit repeating data. Timestamp was used to record when a booking was made and when a transaction had been completed.

2.4 Normalisation

First Normal Form is achieved widely throughout the database. Many design decisions throughout the design process aimed to achieve a minimum of first normal form. Within the bkg_CUSTOMER table the customer title was normalised to a separate entity consisting of title name and title id. Other examples of this are evident throughout the database, for example, bkg_BREAKFAST and bkg_COMMISSION.

Second Normal Form has been achieved in the bkg_ROOM_TYPE table as all non-key attributes are reliant on the primary key. Initially room available attributes were grouped in the bkg_ROOM_TYPE table. If anything were to change with room availability, it could not be done without affecting details about the room itself.

Figure 5. Example of Second Normal Form

pk_room_type_id	room_name	max_adult	max_children	room_description	fk_accommodation_detail_id	fk_room_discount_id	fk_smoking_policy_id
1	Single Corporate Room	1	0	perfect for those on short business trips with all...	1	2	2
2	Art Deco Room	2	0	Since the 1920's and for over 75 years the Art Deco...	1	2	2
3	Bridal Suite	2	0	bridal suite	1	2	2
4	Standard Double Room	2	0	standard double room with all basic amenities	2	1	2
5	Standard Triple Room	3	2	standard triple room perfect for couple with small...	2	2	2

Examples where Third Normal Form has been achieved can be seen in the bkg_ACCOMMODATION_DETAIL table. All attributes in the table are reliant on the primary key and all redundant data has been removed. Bkg_LANGUAGE is another example of where Third Normal Form has been achieved in the database.

Figure 6. Examples of Third Normal Form

bkg_ACCOMMODATION_DETAIL table example

pk_accommodation_detail_id	fk_property_id	fk_accommodation_type_id	fk_cancellation_policy_id
1	1	1	1
2	2	1	1

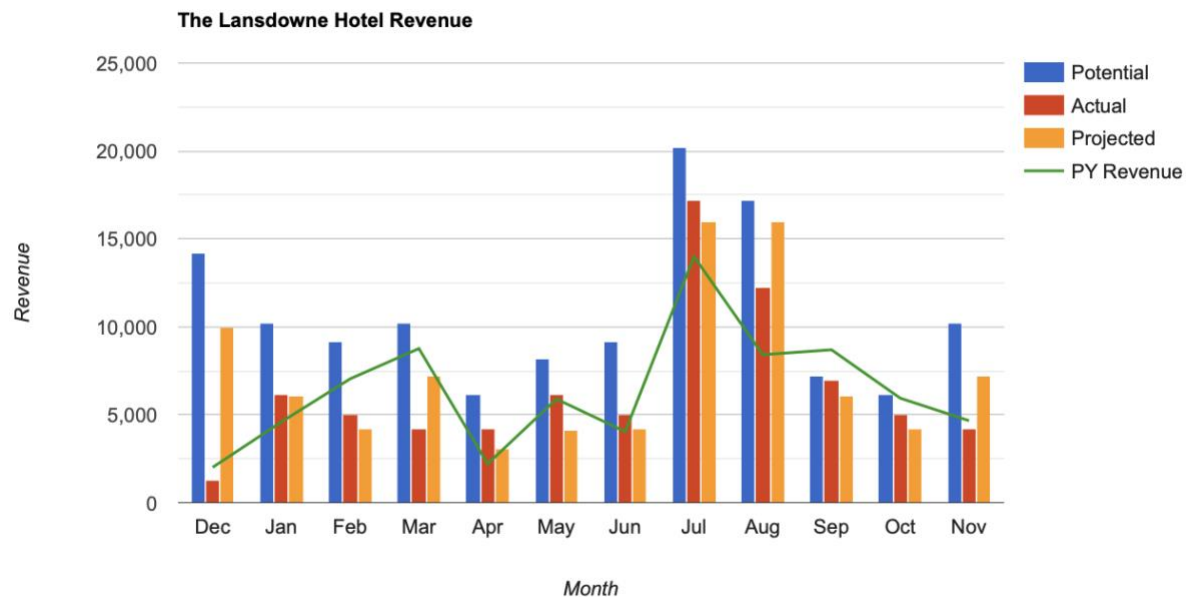
bkg_LANGUAGE table example

pk_language_id	language
1	Albanian
2	Arabic
3	Bulgarian
4	Chinese
5	Dutch
6	English
7	Filipino
8	French
9	Irish
10	Italian
11	Japanese
12	Spanish

3. Business Intelligence

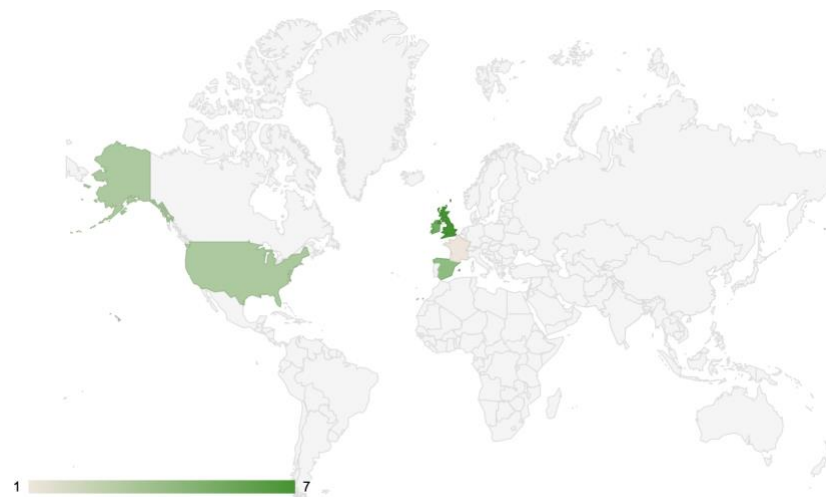
Data stored in the database can be used to facilitate future business decisions for both, properties listed and Booking.com. For example, earnings against potential earnings per property per month can provide insight into performance and compare year on year trends).

Figure 7 Earnings vs Potential Earnings



Dummy data has been used in every month bar Decemeber. [See Figure 7.1 Earnings Vs Potential Earnings Example Query](#) in the appendix for sample query retrieving this info from the database. 'Projected' and 'PY Revenue' are for demonstration purposes only.

Data can also be used to determine user trends. For example, the number of users per country or region on the site (this could prove useful to Booking.com seeing where user traffic is highest to aid in areas like marketing and infrastructure management).



4. Improvements

The database has been designed by a developer with relatively little experience in database design. Given time beyond the limit of this project and further experience, there are areas of the database that could be improved upon;

- Currently, the monetary and payment functions of the database are under developed;
 1. The payments option is relatively simple and only allows for a payment to be made. With further development, allowing for functionality of pay at a later date and pay on arrival would better mirror the views of Booking.com
 2. Rates per room are very rigid. Only one single rate can be applied per room on any given date. The introduction of a rate per room table would allow for multiple rates to be applied to a particular room on a given date.
 3. Considering a percentage rather than a fixed value for all related rates would limit redundant data and allow for dynamic control over rate pricing.
- The use of rollback functions would be particularly useful in regards to bookings, helping improve data integrity so that if something were to occur before the 'complete booking' selection was pressed, nothing would be committed to the database.
- Developing the accommodation detail section of the database to allow for the inclusion of domestic property types would allow for a wider selection of properties for users to book from.
- In hindsight the room type table could have been further normalised to allow for better control over room type and avoidance of repeated data. The introduction of a link table to house the room type may limit repeated data.

5. Conclusion

The overall functionality of the database closely resembles the views of Booking.com and is functional. With the measures taken, the integrity of the data stored in the database is maintained which is evidenced in the Appendix (VIVA Demonstration).

There are areas of the database that would require further development, some of which have been detailed above ([4. Improvements](#)). In reflection, the database appears to satisfy the requirements of the project and contains sufficient demo records to support this.

APPENDIX

Figure 1.1 Entity Relationship Diagram



Figure 2.1 Data Dictionary

TABLE_NAME	COLUMN_NAME	DATA_TYPE	IS_NULLABLE	DEFAULT_VALUE	KEYS	TABLE_RELATIONS	Comments
bkg_ACCOMMODATION_DETAIL	pk_accommodation_detail_id	int(10)	No		PK		AI
bkg_ACCOMMODATION_DETAIL	fk_property_id	int(10)	Yes	NULL	FK	-> bkg_PROPERTY.pk_property_id ON UPDATE RESTRICT ON DELETE CASCADE	
bkg_ACCOMMODATION_DETAIL	fk_accommodation_type_id	int(10)	Yes	NULL	FK	-> bkg_ACCOMMODATION_TYPE.pk_accommodation_type_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_ACCOMMODATION_DETAIL	fk_cancellation_policy_id	int(10)	Yes	NULL	FK	-> bkg_CANCELLATION_POLICY.pk_cancellation_policy_id ON UPDATE RESTRICT ON DELETE SET_NULL	
bkg_ACCOMMODATION_FACILITY	pk_accommodation_facility_id	int(10)	No		PK		AI
bkg_ACCOMMODATION_FACILITY	fk_accommodation_detail_id	int(10)	Yes	NULL	FK	-> bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id ON UPDATE RESTRICT ON DELETE SET_NULL	

bkg_ACCOMMODATION_FACILITY	fk_facility_service_id	int(10)	Yes	NULL	FK	-> bkg_FACILITY_SERVICE.pk_facility_service_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_ACCOMMODATION_RATING	pk_accommodation_rating_id	int(10)	No		PK		AI
bkg_ACCOMMODATION_RATING	fk_accommodation_detail_id	int(10)	Yes	NULL	FK	-> bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id ON UPDATE RESTRICT ON DELETE SET_NULL	
bkg_ACCOMMODATION_RATING	fk_star_rating_id	int(1)	Yes	NULL	FK	-> bkg_STAR_RATING.pk_star_rating_id ON UPDATE RESTRICT ON DELETE SET_NULL	
bkg_ACCOMMODATION_TYPE	pk_accommodation_type_id	int(10)	No		PK		AI
bkg_ACCOMMODATION_TYPE	accommodation_type	varchar(255)	No				
bkg_ACCOMMODATION_TYPE	fk_commission_id	int(10)	No		FK	-> bkg_COMMISSION.pk_commission_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_ACCOUNT	pk_account_id	int(10)	No		PK		AI
bkg_ACCOUNT	email	varchar(255)	No				
bkg_ACCOUNT	password	varbinary(255)	No				
bkg_ADDRESS	pk_address_id	int(10)	No		PK		AI
bkg_ADDRESS	street_addresses	varchar(255)	No				
bkg_ADDRESS	address_line_2	varchar(255)	No				
bkg_ADDRESS	fk_country_id	int(10)	Yes	NULL	FK	-> bkg_COUNTRY.pk_country_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_ADDRESS	city	varchar(255)	No				
bkg_ADDRESS	post_zip_code	varchar(255)	No				
bkg_ADDRESS	fk_property_id	int(10)	No		FK	-> bkg_PROPERTY.pk_property_id ON UPDATE RESTRICT ON DELETE CASCADE	
bkg_AMENITY	pk_amenity_id	int(10)	No		PK		AI
bkg_AMENITY	amenity_name	varchar(255)	No				

bkg_AMENITY	fk_travel_reason_id	int(10)	No	2	FK	-> bkg_TRAVEL_REASON.pk_travel_reason_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_BED_TYPE	pk_bed_type_id	int(10)	No		PK		AI
bkg_BED_TYPE	bed_type	varchar(255)	No				
BKG_BED_TYPE_PER_ROOM	pk_bed_type_per_room_id	int(10)	No		PK		AI
BKG_BED_TYPE_PER_ROOM	fk_room_type_id	int(10)	Yes	NULL	FK	-> bkg_ROOM_TYPE.pk_room_type_id ON UPDATE RESTRICT ON DELETE SET_NULL	
BKG_BED_TYPE_PER_ROOM	fk_bed_type_id	int(10)	Yes	NULL	FK	-> bkg_BED_TYPE.fk_bed_type_id ON UPDATE RESTRICT ON DELETE RESTRICT	
BKG_BED_TYPE_PER_ROOM	bed_count	int(10)	No	1			
bkg_BOOKING	pk_booking_id	int(10)	No		PK		AI
bkg_BOOKING	fk_customer_id	int(10)	Yes	NULL	FK	-> bkg_CUSTOMER.pk_customer_id ON UPDATE RESTRICT ON DELETE SET_NULL	
bkg_BOOKING	date_created	timestamp	No	CURRENT_TIMESTAMP			on update CURRENT_TIMESTAMP
bkg_BOOKING	fk_travel_reason_id	int(10)	No	2	FK	-> bkg_TRAVEL_REASON.pk_travel_reason_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_BOOKING_DETAIL	pk_booking_detail_id	int(10)	No		PK		AI
bkg_BOOKING_DETAIL	fk_booking_id	int(10)	No		FK	-> bkg_BOOKING.pk_booking_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_BOOKING_DETAIL	fk_room_type_id	int(10)	Yes	NULL	FK	-> bkg_ROOM_TYPE.pk_room_type_id ON UPDATE NO_ACTION ON DELETE SET_NULL	
bkg_BOOKING_DETAIL	number_adult	int(10)	No	1			
bkg_BOOKING_DETAIL	number_child	int(10)	No	0			
bkg_BOOKING_DETAIL	check_in_date	date	No				
bkg_BOOKING_DETAIL	check_out_date	date	No				

bkg_BREAKFAST	pk_breakfast_id	int(10)	No		PK		AI
bkg_BREAKFAST	breakfast_type	varchar(255)	No				
bkg_CALENDAR	pk_calendar_id	int(10)	No		PK		AI
bkg_CALENDAR	date	date	No				
bkg_CANCELLATION	pk_cancellation_policy_id	int(10)	No		PK		AI
bkg_CANCELLATION	days_notice	int(2)	No				
bkg_CANCELLATION	cancellation_fee	varchar(255)	No				
bkg_CARD_TYPE	pk_card_type_id	int(10)	No		PK		AI
bkg_CARD_TYPE	card_name	varchar(255)	No				
bkg_COMMISSION	pk_commission_id	int(10)	No		PK		AI
bkg_COMMISSION	commission_type	varchar(255)	No				
bkg_COMMISSION	commission_rate	float	No	0			
bkg_CONTACT	pk_contact_id	int(10)	No		PK		AI
bkg_CONTACT	contact_name	varchar(255)	No				
bkg_CONTACT	fk_account_id	int(10)	No		FK	-> bkg_ACCOUNT.pk_account_id ON UPDATE RESTRICT ON DELETE CASCADE	
bkg_COUNTRY	pk_country_id	int(10)	No		PK		AI
bkg_COUNTRY	country	varchar(255)	No				
bkg_CUSTOMER	pk_customer_id	int(10)	No		PK		AI
bkg_CUSTOMER	first_name	varchar(255)	No				
bkg_CUSTOMER	last_name	varchar(255)	No				
bkg_CUSTOMER	fk_country_id	int(10)	Yes	NULL	FK	-> bkg_COUNTRY.pk_country_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_CUSTOMER	fk_account_id	int(10)	Yes	NULL	FK	-> bkg_ACCOUNT.pk_account_id ON UPDATE RESTRICT ON DELETE CASCADE	
bkg_CUSTOMER	date_birth	date	Yes	NULL			
bkg_CUSTOMER	fk_title_id	int(10)	No		FK	-> bkg_TITLE.pk_title_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_CUSTOMER	fk_int_area_code_id	int(10)	No		FK	-> bkg_INT_AREA_CODE.pk_int_area_code_id ON UPDATE RESTRICT ON DELETE RESTRICT	

bkg_CUSTOMER	customer_phone	varchar(255)	No				
bkg_FACILITY_SERVICE	pk_facility_service_id	int(10)	No		PK		AI
bkg_FACILITY_SERVICE	facility_type	varchar(255)	No				
bkg_FACILITY_SERVICE	description	varchar(255)	No	N/A			
bkg_FACILITY_SERVICE	fee	float	No	0			
bkg_FACILITY_SERVICE	fk_travel_reason_id	int(10)	No		FK	-> bkg_TRAVEL_REASON.pk_travel_reason_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_FACILITY_SERVICE	location	varchar(255)	No	N/A			
bkg_FACILITY_SERVICE	reservation_required	tinyint(1)	No	0			
bkg_GUEST	pk_guest_id	int(10)	No		PK		AI
bkg_GUEST	guest_name	varchar(255)	No				
bkg_GUEST	fk_booking_detail_id	int(10)	No		FK	-> bkg_BOOKING_DETAIL.pk_booking_detail_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_GUEST	guest_email	varchar(255)	No				
bkg_INT_AREA_CODE	pk_int_area_code_id	int(10)	No		PK		AI
bkg_INT_AREA_CODE	country_code	varchar(255)	No				
bkg_INT_AREA_CODE	fk_country_id	int(10)	No		FK	-> bkg_COUNTRY.pk_country_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_LANGUAGE	pk_language_id	int(10)	No		PK		AI
bkg_LANGUAGE	language	varchar(255)	No				
bkg_LANGUAGE_SPOKEN	pk_language_spoken_id	int(10)	No		PK		AI
bkg_LANGUAGE_SPOKEN	fk_accommodation_detail_id	int(10)	Yes	NULL	FK	-> bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id ON UPDATE RESTRICT ON DELETE SET_NULL	
bkg_LANGUAGE_SPOKEN	fk_language_id	int(10)	Yes	NULL	FK	-> bkg_LANGUAGE.pk_language_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_PAYMENT_INFO	pk_payment_info_id	int(10)	No		PK		AI

bkg_PAYMENT_INFO	fk_card_type_id	int(10)	No		FK	-> bkg_CARD_TYPE.pk_card_type_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_PAYMENT_INFO	card_holder_name	varchar(255)	No				
bkg_PAYMENT_INFO	card_number	varbinary(255)	No				
bkg_PAYMENT_INFO	expiry_date	varchar(255)	No				
bkg_PAYMENT_INFO	fk_customer_id	int(10)	No		FK	-> bkg_CUSTOMER.pk_customer_id ON UPDATE CASCADE ON DELETE CASCADE	
bkg_PHONE	pk_phone_id	int(10)	No		PK		AI
bkg_PHONE	fk_contact_id	int(10)	Yes	NULL	FK	-> bkg_CONTACT.pk_contact_id ON UPDATE RESTRICT ON DELETE CASCADE	
bkg_PHONE	fk_int_area_code_id	int(10)	No		FK	-> bkg_INT_AREA_CODE.pk_int_area_code_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_PHONE	phone_number	varchar(255)	No				
bkg_PHOTO	pk_photo_id	int(10)	No		PK		AI
bkg_PHOTO	fk_property_id	int(10)	No		FK	-> bkg_PROPERTY.pk_property_id ON UPDATE RESTRICT ON DELETE CASCADE	
bkg_PHOTO	photo	longblob	No				
bkg_PROPERTY	pk_property_id	int(10)	No		PK		AI
bkg_PROPERTY	property_name	varchar(255)	No				
bkg_PROPERTY	property_description	mediumtext	No				
bkg_PROPERTY	fk_contact_id	int(10)	Yes	NULL	FK	-> bkg_CONTACT.pk_contact_id ON UPDATE RESTRICT ON DELETE CASCADE	
bkg_PROPERTY	latitude	float	No				
bkg_PROPERTY	longitude	float	No				
bkg_ROOM_AMENITY	pk_room_amenity_id	int(10)	No		PK		AI
bkg_ROOM_AMENITY	fk_room_type_id	int(10)	Yes	NULL	FK	-> bkg_ROOM_TYPE.pk_room_type_id ON UPDATE RESTRICT ON DELETE SET_NULL	

bkg_ROOM_AMENITY	fk_amenity_id	int(10)	Yes	NULL	FK	-> bkg_AMENITY.pk_amenity_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_ROOM_AVAILABLE	pk_room_available_id	int(10)	No		PK		AI
bkg_ROOM_AVAILABLE	fk_room_type_id	int(10)	Yes	NULL	FK	-> bkg_ROOM_TYPE.pk_room_type_id ON UPDATE RESTRICT ON DELETE SET_NULL	
bkg_ROOM_AVAILABLE	room_available_count	int(10)	No				
bkg_ROOM_AVAILABLE	fk_calendar_id	int(10)	No		FK	-> bkg_CALENDAR.pk_calendar_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_ROOM_AVAILABLE	room_base_price	float	No	0.01			
bkg_ROOM_AVAILABLE	fk_room_rate_id	int(10)	No		FK	-> bkg_ROOM_RATE.pk_room_rate_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_ROOM_BREAKFAST	pk_room_breakfast_id	int(10)	No		PK		AI
bkg_ROOM_BREAKFAST	fk_room_type_id	int(10)	Yes	NULL	FK	-> bkg_ROOM_TYPE.pk_room_type_id ON UPDATE RESTRICT ON DELETE SET_NULL	
bkg_ROOM_BREAKFAST	fk_breakfast_id	int(10)	Yes	NULL	FK	-> bkg_BREAKFAST.pk_breakfast_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_ROOM_BREAKFAST	breakfast_fee	float	No	0			
bkg_ROOM_DISCOUNT	pk_room_discount_id	int(10)	No		PK		AI
bkg_ROOM_DISCOUNT	discount_description	varchar(255)	No				
bkg_ROOM_DISCOUNT	discount_rate	float	No	0			
bkg_ROOM_RATE	pk_room_rate_id	int(10)	No		PK		AI
bkg_ROOM_RATE	rate_description	varchar(255)	No				
bkg_ROOM_RATE	rate_charge	float	No	0			
bkg_ROOM_TYPE	pk_room_type_id	int(10)	No		PK		AI
bkg_ROOM_TYPE	room_name	varchar(255)	No				
bkg_ROOM_TYPE	max_adult	int(10)	No				
bkg_ROOM_TYPE	max_children	int(10)	No				

bkg_ROOM_TYPE	room_description	mediumtext	No				
bkg_ROOM_TYPE	fk_accommodation_detail_id	int(10)	Yes	NULL	FK	-> bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id ON UPDATE RESTRICT ON DELETE CASCADE	
bkg_ROOM_TYPE	fk_room_discount_id	int(10)	No	2	FK	-> bkg_ROOM_DISCOUNT.pk_room_discount_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_ROOM_TYPE	fk_smoking_policy_id	int(10)	No		FK	-> bkg_SMOKING_POLICY.pk_smoking_policy_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_SMOKING_POLICY	pk_smoking_policy_id	int(10)	No		PK		AI
bkg_SMOKING_POLICY	smoking_policy	varchar(255)	No				
bkg_STAR_RATING	pk_star_rating_id	int(10)	No		PK		AI
bkg_STAR_RATING	star_rating	int(1)	No	1			
bkg_TITLE	pk_title_id	int(10)	No		PK		AI
bkg_TITLE	title_name	varchar(255)	No				
bkg_TRANSACTION	pk_transaction_id	int(10)	No		PK		AI
bkg_TRANSACTION	fk_booking_id	int(10)	No		FK	-> bkg_BOOKING.pk_booking_id ON UPDATE RESTRICT ON DELETE RESTRICT	
bkg_TRANSACTION	last_four_digits	varchar(255)	No				
bkg_TRANSACTION	card_name	varchar(255)	No				
bkg_TRANSACTION	transaction_amount	float	No				
bkg_TRANSACTION	transaction_date	timestamp	No	CURRENT_TIMESTAMP			on update CURRENT_TIMESTAMP
bkg_TRAVEL_REASON	pk_travel_reason_id	int(10)	No		PK		AI
bkg_TRAVEL_REASON	travel_reason	varchar(255)	No				

Figure 3.1/4.1 Data Types & Key/Non-Key Attributes Data tables

DATA TYPE	COUNT
int	91
varchar	36
varbinary	2
date	4
timestamp	2
float	9
tinyint	1
longblob	1
mediumtext	2

KEY/NON-KEY	COUNT
Primary Keys	39
Foreign Keys	44
Non-Key Attributes	65

Figure 7.1 Earnings Vs Potential Earnings Example Query

Potential Earning of a Property Query

```
SELECT bkg_PROPERTY.property_name, ROUND((SUM((bkg_ROOM_AVAILABLE.room_base_price + bkg_ROOM_RATE.rate_charge) * bkg_ROOM_AVAILABLE.room_available_count))) AS 'Total Potential Revenue £'
FROM bkg_ROOM_TYPE
INNER JOIN bkg_ROOM_AVAILABLE ON bkg_ROOM_AVAILABLE.fk_room_type_id = bkg_ROOM_TYPE.pk_room_type_id
INNER JOIN bkg_ACCOMMODATION_DETAIL ON bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id = bkg_ROOM_TYPE.fk_accommodation_detail_id
INNER JOIN bkg_PROPERTY ON bkg_PROPERTY.pk_property_id = bkg_ACCOMMODATION_DETAIL.fk_property_id
INNER JOIN bkg_ROOM_RATE ON bkg_ROOM_RATE.pk_room_rate_id = bkg_ROOM_AVAILABLE.fk_room_rate_id
WHERE bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id = 2 AND bkg_ROOM_AVAILABLE.fk_calendar_id BETWEEN 1 AND 31
GROUP BY bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id;
```

Actual Earning of a Property Query

```
SELECT ROUND((SUM((bkg_ROOM_AVAILABLE.room base price + bkg_ROOM_RATE.rate charge) * DATE
DIFF(check_out_date, check_in_date)) / COUNT(DISTINCT bkg_ROOM_AVAILABLE.fk_calendar_id)),
2) as'Total Actual Revenue £' FROM bkg_BOOKING
INNER JOIN bkg_BOOKING_DETAIL ON bkg_BOOKING_DETAIL.fk_booking_id = bkg_BOOKING.pk_booking
_id
INNER JOIN bkg_ROOM_TYPE ON
bkg_ROOM_TYPE.pk_room_type_id = bkg_BOOKING_DETAIL.fk_room_type_id
INNER JOIN bkg_ROOM_AVAILABLE ON bkg_ROOM_AVAILABLE.fk_room_type_id = bkg_ROOM_TYPE.pk_roo
m_type_id
INNER JOIN
bkg_ACCOMMODATION_DETAIL ON bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id = bkg_ROOM
TYPE.fk_accommodation_detail_id
INNER JOIN bkg_ROOM_RATE ON
bkg_ROOM_RATE.pk_room_rate_id = bkg_ROOM_AVAILABLE.fk_room_rate_id
WHERE bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id = 2
AND bkg_BOOKING_DETAIL.check_in_date > '2019-11-
31' AND bkg_BOOKING_DETAIL.check_in_date < '2020-01-01'
GROUP BY bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id;
```

Location of User Traffic Query

```
SELECT bkg_COUNTRY.country, COUNT(bkg_CUSTOMER.fk_country_id) AS 'Total
Users' FROM bkg_CUSTOMER LEFT JOIN bkg_COUNTRY ON bkg_CUSTOMER.fk_country_id =bkg_COUNTRY.
pk_country_id GROUP BY bkg_CUSTOMER.fk_country_id;
```

Other Business Intelligence Example Queries

Bookings Made in Country/Region Per Month

```
SELECT bkg_COUNTRY.country, COUNT(DISTINCT bkg_BOOKING.pk_booking_id) AS 'Bookings Made In
November' FROM bkg_BOOKING
INNER JOIN bkg_BOOKING_DETAIL ON bkg_BOOKING_DETAIL.fk_booking_id =
bkg_BOOKING.pk_booking_id
INNER JOIN bkg_ROOM_TYPE ON bkg_ROOM_TYPE.pk_room_type_id = bkg_BOOKING_DETAIL.fk_room_typ
e_id
INNER JOIN bkg_ACCOMMODATION_DETAIL ON
bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id = bkg_ROOM_TYPE.fk_accommodation_detai
l_id
INNER JOIN bkg_PROPERTY ON bkg_PROPERTY.pk_property_id =bkg_ACCOMMODATION_DETAIL.fk_proper
ty_id
INNER JOIN bkg_ADDRESS ON bkg_ADDRESS.fk_property_id = bkg_PROPERTY.pk_property_id
INNER JOIN bkg_COUNTRY ON bkg_COUNTRY.pk_country_id= bkg_ADDRESS.fk_country_id WHERE bkg_B
OOKING.date_created BETWEEN '2019-11-01' AND '2019-11-31'
GROUP BY bkg_COUNTRY.country;
```

DATABASE VIVA

1. Customer search for a property, book and pay for their stay

A customer would like to create an account and store their information to create a booking. When the customer presses the register button they will be prompted to create an account and after, choose and verify their password.

The first screenshot shows the 'Sign in' page of the Booking.com account creation process. It includes a header with the Booking.com logo and 'Account' text. Below the header, there's a 'Sign in' section with a sub-header 'You can sign in using your Booking.com account to access our services.' An email address field contains 'mysticrecords@hotmail.com'. A blue 'Next' button is below the field. Below the 'Next' button, there's a horizontal line with 'or' in the center. Below the line, there are two buttons: 'Sign in with Facebook' and 'Sign in with Google'. At the bottom, there's a link 'Don't have an account yet? Sign up'.

The second screenshot shows the 'Create password' page of the Booking.com account creation process. It includes a header with the Booking.com logo and 'Account' text. Below the header, there's a 'Create password' section with a sub-header 'Please choose a password for your Booking.com account registered to mystic@hotmail.com.' There are two password fields: 'Create password' and 'Confirm password'. A blue 'Create account' button is at the bottom.

```
INSERT INTO `bkg_ACCOUNT` (`pk_account_id`, `email`, `password`) VALUES (2, 'mysticrecords@hotmail.com', AES_ENCRYPT('Password123', 'pass'));
```

The customer creates their own password, AES Encrypt is used to maintain security for sensitive information.

Example of How to decrypt password

```
SELECT email, AES_DECRYPT(password, 'pass') FROM bkg_ACCOUNT  
WHERE email = 'mysticrecords@hotmail.com';
```

This information is stored in the account table which is shown in the image below


	pk_account_id	email	password
Edit Copy Delete	1	billwolsey@themerchant.com	0x0f9fb9087039c3060f169e49dc4120c9fddae3e765fa3e31...
Edit Copy Delete	2	mysticrecords@hotmail.com	0x0ae4391c87da51176029a8c651169e1f

The customer can update their profile with their information so that when it comes to making a booking, they don't have to enter their personal information each time. (As detailed in the assumptions, not all fields were used, (display name and address) a selection of attributes were used to demonstrate the functionality of the site).

Your Booking.com account

These details are displayed next to your publicly shared reviews, ratings, photos, etc. Any updates you make here will also appear in past contributions.

Profile picture


[Change photo](#)

The first thing people see is your photo, so show them your best side.

Display name

Your unique display name can be updated as often as you like.

Birthday

15 March 1988

We'll display this information as an age range, e.g. "25-30"

Country/region

United Kingdom

People are curious about where you're from.

For when you book

This information is only used to autofill your details and make it quicker for you to book. Your details will be stored securely and won't be shared publicly.

Title

Mr.

First name

Matt

Last name

Walsh

Phone number

07595052161

Email

mysticrecords@hotmail.c

Addresses

Address

14 mill valley place, Bt14 8fx
Belfast, United Kingdom

[Change home address](#)

```
INSERT INTO `bkg_CUSTOMER` (`pk_customer_id`, `first_name`, `last_name`, `fk_country_id`,
`fk_account_id`, `date_birth`, `fk_title_id`, `fk_int_area_code_id`, `customer_phone`) VALUES
(NULL, 'Matthew', 'Walsh', '1', '2', '1988-03-15', '1', '1', '7595052161');
```

This information is stored in the customer table which is depicted below.

	pk_customer_id	fk_title_id	first_name	last_name	fk_int_area_code_id	customer_phone	date_birth	fk_country_id	fk_account_id
	1	1	Matthew	Walsh	1	7595052161	1988-03-15	1	2

```
SELECT * FROM bkg_CUSTOMER WHERE first_name = 'Matthew' AND last_name = 'walsh'
```

Credit cards

We'll autofill your details and make it easier for you to book properties and additional services via Booking.com. Your payment details are stored securely.

Credit card type

visa Visa

Credit card number

1234123412341234

Cardholder's name

Matthew Walsh

Expiry date

03 2022

☐ Use this card for business bookings

You can store a credit card for business related travel arrangements. This will make it easier for you to invoice those bookings.

☐ Use this card for my reward payments

This will be the card that your rewards are paid to.

[Save change](#)
[Cancel](#)

A customer can opt to store their card details if they wish for future bookings. These details are stored in the payment info table. The customer_id is included in this table so zero or many cards can be related to a customer. Below depicts the details in the database.

pk_payment_info_id	fk_customer_id	fk_card_type_id	card_holder_name	card_number	expiry_date
1	1	2	Matthew Walsh	0x1234123412341234	03/2022

*An example of where it may be beneficial to use a transaction. This information would only be stored in the database once the user clicks the save change button.

```
START TRANSACTION;
INSERT INTO `bkg PAYMENT INFO` (`pk payment info id`, `fk card type id`, `card_holder_name`, `card_number`, `expiry_date`, `fk customer id`) VALUES (NULL, '2', 'Matthew Walsh', '0x1234123412341234', '03/2022', '1');
COMMIT;
```

A Customer has created their account, they would now like to find a hotel in Belfast for 2 adults, checking in on 02/12/2019 and checking out 06/12/2019

Where to next, Matt?

From cosy country homes to funky city flats

Belfast

Mon 2 Dec — Fri 6 Dec

2 adults · 0 children · 1 room

Search

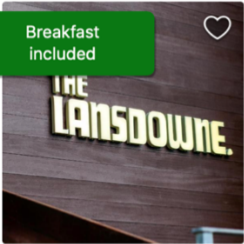
The query below will show all properties with rooms that meet the search criteria entered by the user.

```
SELECT bkg_ROOM_TYPE.pk_room_type_id AS 'Room ID', bkg_ROOM_TYPE.room_name AS 'Room Name', bkg_ROOM_AVAILABLE.room_base_price AS 'Price Per Night (£)', bkg_ROOM_AVAILABLE.room_available count AS 'Available Rooms', bkg_ROOM_TYPE.max adult AS 'Total Adults', bkg_ROOM_TYPE.max children AS 'Total Children', bkg_PROPERTY.property_name AS 'Property', bkg_BED_TYPE.bed_type AS 'Bed Available', bkg_PROPERTY.property_description AS 'Property Description', bkg_ADDRESS.street_address AS 'Address', bkg_ADDRESS.city, bkg_ADDRESS.post_zip_code AS 'PostCode' FROM bkg_PROPERTY

INNER JOIN bkg_ACCOMMODATION_DETAIL ON bkg_ACCOMMODATION_DETAIL.fk_property_id = bkg_PROPERTY.pk_property_id
INNER JOIN bkg_ROOM_TYPE ON bkg_ROOM_TYPE.fk_accommodation_detail_id = bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id
INNER JOIN bkg_ROOM_AVAILABLE ON bkg_ROOM_AVAILABLE.fk_room_type_id = bkg_ROOM_TYPE.pk_room_type_id
INNER JOIN bkg_BED_TYPE_PER_ROOM ON bkg_BED_TYPE_PER_ROOM.fk_room_type_id = bkg_ROOM_TYPE.pk_room_type_id
INNER JOIN bkg_BED_TYPE ON bkg_BED_TYPE.pk_bed_type_id = bkg_BED_TYPE_PER_ROOM.fk_bed_type_id
INNER JOIN bkg_ADDRESS ON bkg_ADDRESS.fk_property_id = bkg_PROPERTY.pk_property_id

WHERE bkg_ADDRESS.city= 'Belfast'
AND bkg_ROOM_TYPE.max adult >= 1
AND bkg_ROOM_AVAILABLE.fk_calendar_id >= 2
AND bkg_ROOM_AVAILABLE.fk_calendar_id <= 6;
```

Room ID	Room Name	Price Per Night (£)	Available Rooms	Total Adults	Total Children	Property	Bed Available	Property Description	Address	city	PostCode
2	Art Deco Room	200	3	2	0	The Merchant Hotel	Large bed (King size) / 151 - 80 cm wide	The AA 5 Red Star Merchant Hotel in Belfast is sit...	16 skipper street	Belfast	bt1 4qg
2	Art Deco Room	200	3	2	0	The Merchant Hotel	Large bed (King size) / 151 - 80 cm wide	The AA 5 Red Star Merchant Hotel in Belfast is sit...	16 skipper street	Belfast	bt1 4qg
2	Art Deco Room	200	2	2	0	The Merchant Hotel	Large bed (King size) / 151 - 80 cm wide	The AA 5 Red Star Merchant Hotel in Belfast is sit...	16 skipper street	Belfast	bt1 4qg
2	Art Deco Room	200	2	2	0	The Merchant Hotel	Large bed (King size) / 151 - 80 cm wide	The AA 5 Red Star Merchant Hotel in Belfast is sit...	16 skipper street	Belfast	bt1 4qg
2	Art Deco Room	200	2	2	0	The Merchant Hotel	Large bed (King size) / 151 - 80 cm wide	The AA 5 Red Star Merchant Hotel in Belfast is sit...	16 skipper street	Belfast	bt1 4qg
4	Standard Double Room	100	10	2	0	The Landsdowne Hotel	Double bed / 131 - 150 cm wide	Situated in Belfast, 3.1 miles from SSE Arena, The...	657 Antrim Road	Belfast	BT15 4EF
4	Standard Double Room	100	10	2	0	The Landsdowne Hotel	Double bed / 131 - 150 cm wide	Situated in Belfast, 3.1 miles from SSE Arena, The...	657 Antrim Road	Belfast	BT15 4EF



Breakfast included

The Lansdowne Hotel

Belfast • [Show on map](#) • 2.7 miles from centre

Standard Double Room – 2 adults

1 double bed

Risk free: You can cancel later, so lock in this great price today.

Superb 9.3

157 reviews

4 nights, 2 adults

£305

includes taxes and charges

Breakfast included

FREE cancellation

No prepayment needed

[Choose your room >](#)

The customer would like to find out more detail about the Lansdowne hotel, and find what facilities are available. They would preferably like to know if there is free wifi and parking available.

```

SELECT pk_property_id FROM bkg_PROPERTY WHERE property_name = 'The Landsdowne Hotel';

SELECT facility_type, description, fee, location, reservation_required FROM bkg_PROPERTY
INNER JOIN bkg_ACCOMMODATION_DETAIL ON bkg_ACCOMMODATION_DETAIL.fk_property_id =bkg_PROPERTY.pk_property_id
INNER JOIN bkg_ACCOMMODATION_FACILITY ON bkg_ACCOMMODATION_FACILITY.fk_accommodation_detail_id =bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id
INNER JOIN bkg_FACILITY_SERVICE ON bkg_FACILITY_SERVICE.pk_facility_service_id =bkg_ACCOMMODATION_FACILITY.fk_facility_service_id
WHERE bkg_PROPERTY.pk_property_id = 2;

```

facility_type	description	fee	location	reservation_required
Parking	Private	0	On Site	1
Internet	Wifi	0	Entire Property	0
Function Room 2	used for large corporate events	1000	entire first floor	1
Play Area	playa area and park for children	0	in the gardens	0
Parking	Private	0	On Site	1
Internet	Cable	0	Entire Property	0

The UI provides all relevant information about the hotel to the customer. It is assumed the presentation layer would format this information in a readable layout to aid in the customer making a booking.

Hotel

The Lansdowne Hotel

Great for two travellers

657-659 Antrim Road, Belfast, BT15 4EF, United Kingdom

Great location - show map

Reserve

We Price Match

Situated in Belfast, 3.1 miles from SSE Arena, The Lansdowne Hotel features accommodation with a restaurant, free private parking, a bar and a garden. Among the facilities at this property are a 24-hour front desk and room service, along with free WiFi throughout the property. The accommodation provides evening entertainment and luggage storage space.

All rooms in the hotel are equipped with a TV. The private bathroom is fitted with a shower, a hairdryer and free toiletries.

A Full English/Irish breakfast is available each morning at The Lansdowne Hotel.

The accommodation offers a sun terrace.

The Waterfront Hall is 3.1 miles from The Lansdowne Hotel, while The Belfast Empire Music Hall is 3.7 miles from the property. The nearest airport is George Best Belfast City Airport, 5 miles from the hotel.

Couples particularly like the location — they rated it **8.8** for a two-person trip.

Guests love...

"friendly staff"

20 related reviews

"excellent food"

10 related reviews

"comfortable bed"

8 related reviews

Perfect for a 4-night stay!

Top location: Highly rated by recent guests (8.7)

Breakfast info

Full English/Irish

Very good coffee!

Free WiFi

Free private parking available at the hotel

This query is an example of the relevant information that the website provides the customer.

```

SELECT pk_property_id FROM bkg_PROPERTY WHERE property_name = 'The Lansdowne Hotel';

SELECT property_name, property_description, bkg_ADDRESS.street_address, bkg_ADDRESS.address_line_2, bkg_ADDRESS.city, bkg_ADDRESS.post_zip_code, bkg_ACCOMMODATION_TYPE.accommodation_type, bkg_STAR_RATING.star_rating, bkg_CANCELLATION_POLICY.days_notice, bkg_CANCELLATION_POLICY.cancellation_fee, latitude, longitude FROM bkg_ADDRESS
INNER JOIN bkg_PROPERTY ON bkg_PROPERTY.pk_property_id = bkg_ADDRESS.fk_property_id
INNER JOIN bkg_ACCOMMODATION_DETAIL ON bkg_ACCOMMODATION_DETAIL.fk_property_id = bkg_PROPERTY.pk_property_id
INNER JOIN bkg_ACCOMMODATION_RATING ON bkg_ACCOMMODATION_RATING.fk_accommodation_detail_id = bkg_ACCOMMODATION_DETAIL.pk_accommodation_detail_id
INNER JOIN bkg_STAR_RATING ON bkg_STAR_RATING.pk_star_rating_id = bkg_ACCOMMODATION_RATING.fk_star_rating_id
INNER JOIN bkg_ACCOMMODATION_TYPE ON bkg_ACCOMMODATION_TYPE.pk_accommodation_type_id = bkg_ACCOMMODATION_DETAIL.fk_accommodation_type_id
INNER JOIN bkg_CANCELLATION_POLICY ON bkg_CANCELLATION_POLICY.pk_cancellation_policy_id = bkg_ACCOMMODATION_DETAIL.fk_cancellation_policy_id
WHERE bkg_PROPERTY.pk_property_id = 2;

```

property_name	property_description	street_address	address_line_2	city	post_code	accommodation_type	star_rating	days_notice	cancellation_fee	property_latitude	property_longitude
The Lansdowne Hotel	Situated in Belfast, 3.1 miles from SSE Arena, The...	657 Antrim Rd		Belfast	BT15 4EF	Hotel	3	1	first night	54.6346	5.9371

Room type	Sleeps	Price for 4 nights	Your choices	Select rooms	
Standard Double Room 1 double bed Ensuite bathroom Free WiFi • Shower • Safety Deposit Box • TV • Hairdryer • Iron • Free toiletries • Wake-up service • Upper floors accessible by elevator Prices are per room for 4 nights Included: 20 % VAT, Breakfast	 includes taxes and charges	£305 ⓘ includes taxes and charges	Superb breakfast included ⓘ ✓ FREE cancellation before 23:59 on 30 November 2019 ✓ NO PREPAYMENT NEEDED - pay at the property	<input type="text" value="0"/>	<div>I'll reserve</div> <ul style="list-style-type: none"> It only takes 2 minutes Confirmation is immediate No booking or credit card fees!
Jackpot! This is the cheapest price you've seen in Belfast for your dates! 4 nights (Mon 2 Dec - Fri 6 Dec)					
	 includes taxes and charges	£265 ⓘ includes taxes and charges	Superb breakfast included ⓘ ✓ FREE cancellation before 23:59 on 30 November 2019 ✓ NO PREPAYMENT NEEDED - pay at the property	<input type="text" value="0"/>	

The Customer is happy with the price, facilities and amenities that are available and would like to book two rooms for himself and his partner, and the same room for a friend. With multiple rooms booked, these would both be linked to an overall booking.

The customer would like to find the total cost of his stay for both rooms and show any discounts that apply.

This query calculates the total cost of all rooms related to this booking.

Your booking details	
Check-in:	Monday, 2 December 2019 from 15:00
Check-out:	Friday, 6 December 2019 until 11:00
Total length of stay:	4 nights
	Travelling on different dates?
You selected:	2 x Standard Double Room
	Change your selection
Your price summary	
2 rooms	£475
20 % VAT	£95
Price (for 3 guests and 4 nights)	£570

```

SELECT bkg BOOKING.pk_booking_id,
(SUM((bkg ROOM_AVAILABLE.room_base_price + bkg ROOM
RATE.rate_charge) * DATEDIFF(check_out_date, check_
in_date)) / COUNT(DISTINCT
bkg_ROOM_AVAILABLE.fk_calendar_id)) as 'TotalCost£(w
ithoutDiscount)'
FROM bkg_BOOKING
INNER JOIN bkg_BOOKING_DETAIL ON bkg_BOOKING_DETAIL.
fk_booking_id =bkg_BOOKING.pk_booking_id INNER JOIN
bkg_ROOM_TYPE ON bkg_ROOM_TYPE.pk_room_type_id = bkg
_BOOKING_DETAIL.fk_room_type_id
INNER JOIN bkg_ROOM_AVAILABLE ON
bkg_ROOM_AVAILABLE.fk_room_type_id = bkg_ROOM_TYPE.p
k_room_type_id
INNER JOIN bkg_ROOM_RATE ON bkg_ROOM_RATE.pk_room_ra
te_id = bkg_ROOM_AVAILABLE.fk_room_rate_id
WHERE bkg BOOKING.pk_booking_id = 1
GROUP BY bkg_BOOKING.pk_booking_id;

```

+ Options

pk_booking_id	TotalCost£(withoutDiscount)
2	880

This query returns if a discount is available for a particular room. It is assumed a program would handle whether or not it should be applied by looking at the guests booked in a room against the total the room accommodates.

```
SELECT bkg_ROOM_DISCOUNT.discount_rate AS 'RatePerRoomPerNight', bkg_ROOM_TYPE.max_adult AS 'TotalAdults', bkg_BOOKING_DETAIL.number_adult AS 'BookedAdults'
FROM bkg_ROOM_DISCOUNT
INNER JOIN bkg_ROOM_TYPE ON bkg_ROOM_TYPE.fk_room_discount_id = bkg_ROOM_DISCOUNT.pk_room_discount_id
INNER JOIN bkg_BOOKING_DETAIL ON
bkg_BOOKING_DETAIL.fk_room_type_id = bkg_ROOM_TYPE.pk_room_type_id
WHERE bkg_BOOKING_DETAIL.fk_booking_id = 1;
```

+ Options

RatePerRoomPerNight	TotalAdults	BookedAdults
10	2	2
10	2	1

In this instance, a discount is available for this room and one room would have it applied per night. In total, £40 discount would be applied to the total booking cost. The results of these queries would be handled by a program to determine the total cost to the customer (£840 in this case).

A booking would only be committed to the database once the customer has confirmed their booking.

The customer id, date the booking was placed and travel reason are specified for a record to be created in the bkg_BOOKING table. A customer can have many bookings.

	pk_booking_id	fk_customer_id	date_created	fk_travel_reason_id
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	2019-11-14 22:58:45	2

Details about the customer booking are stored in bkg_BOOKING_DETAIL table, room_type number of adults and children, check in and out date are all stored so this information can be used to calculate the total cost of the booking. The booking id is included to relate the booking details to the booking. A booking can have many booking details.

	pk_booking_detail_id	fk_booking_id	fk_room_type_id	number_adult	number_child	check_in_date	check_out_date
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	4	2	0	2019-12-02	2019-12-06
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	1	4	1	0	2019-12-02	2019-12-06

A program uses the booking_id, transaction_amount (which is derived from room_base_price, room_rate & room_discount), last_four_digits and card_name to create a record in the bkg_TRANSACTION table.

	pk_transaction_id	fk_booking_id	last_four_digits	card_name	transaction_amount	transaction_date
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	1234	Matthew Walsh	840	2019-11-14 22:49:18

A timestamp records when the payment was fulfilled.

	pk_payment_info_id	fk_customer_id	fk_card_type_id	card_holder_name	card_number	expiry_date
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	2	Matthew Walsh	0x2646543362c306790717a727fc84962d3be2b179b448dc5d...	03/2022

```

SELECT * FROM bkg_BOOKING WHERE fk_customer_id = 1;

SELECT * FROM bkg_BOOKING_DETAIL WHERE fk_booking_id = 1;

SELECT * FROM bkg_TRANSACTION WHERE fk_booking_id = 1;

```

**Below is an example query for a customer making a booking and payment. For demonstration purposes, this data has already been populated into the database.

```

START TRANSACTION;

INSERT INTO `bkg_BOOKING` (`pk_booking_id`, `fk_customer_id`, `date_created`,
`fk_travel_reason_id`) VALUES ('1', '1', 'CURRENT_TIMESTAMP', '2');

SAVEPOINT Create_Booking;

INSERT INTO `bkg_BOOKING_DETAIL` (`pk_booking_detail_id`, `fk_booking_id`,
`fk_room_type_id`, `number_adult`, `number_child`, `check_in_date`, `check_out_date`)
VALUES ('1', '1', '4', '2', '0', '2019-12-02', '2019-12-06');

INSERT INTO `bkg_GUEST` (`pk_guest_id`, `guest_name`, `fk_booking_detail_id`,
`guest_email`) VALUES ('1', 'Matthew Walsh', '1', 'mysticrecords@hotmail.com');

SAVEPOINT Booking_Detail1;

INSERT INTO `bkg_BOOKING_DETAIL` (`pk_booking_detail_id`, `fk_booking_id`,
`fk_room_type_id`, `number_adult`, `number_child`, `check_in_date`, `check_out_date`)
VALUES ('2', '1', '4', '1', '0', '2019-12-02', '2019-11-06');

INSERT INTO `bkg_GUEST` (`pk_guest_id`, `guest_name`, `fk_booking_detail_id`,
`guest_email`) VALUES ('2', 'Peter Griffin', '1', 'SamPucket@Brewery.com');

SAVEPOINT Booking_Detail2;

INSERT INTO `bkg_TRANSACTION` (`pk_transaction_id`, `fk_booking_id`, `last_four_digits`,
`card_name`, `transaction_amount`, `transaction_date`) VALUES ('1', '1', '1234', 'Matthew
Walsh', '840', 'CURRENT_TIMESTAMP');

ROLLBACK TO Create_Booking;

COMMIT;

```

2. Delete a customer record

*As an example to show that data integrity is maintained, if a customer wished to delete their account, all data related to the user would be removed from the database, but booking details would be maintained.

```

SELECT * FROM bkg_CUSTOMER WHERE fk_account_id = 6;
SELECT * FROM bkg_PAYMENT_INFO WHERE fk_customer_id = 2;
SELECT * FROM bkg_BOOKING WHERE fk_customer_id = 2;
SELECT * FROM bkg_BOOKING_DETAIL WHERE fk_booking_id = 2;
SELECT * FROM bkg_TRANSACTION WHERE fk_booking_id = 2;

DELETE FROM bkg_ACCOUNT WHERE pk_account_id = 6;

SELECT * FROM bkg_CUSTOMER WHERE fk_account_id = 6;
SELECT * FROM bkg_PAYMENT_INFO WHERE fk_customer_id = 2;
SELECT * FROM bkg_BOOKING WHERE fk_customer_id = 2;
SELECT * FROM bkg_BOOKING_DETAIL WHERE fk_booking_id = 2;
SELECT * FROM bkg_TRANSACTION WHERE fk_booking_id = 2;

```

When deleting a customer account, we can see that all data related to that customer (including payment info and customer details) have been removed without affecting other records and has maintained data such as title and card type which other records are reliant on.

We can see that booking details remain (minus the customer id) so that booking records can be maintained.

3. Delete and update a property record

Deleting a user account that is used for a property listing has the same functionality as deleting a customer. All records relevant to that user account are removed whilst maintaining data that is relevant to other properties (facility service, language, breakfast and amenity for example).

The actions decided upon update and delete allow for a property lister dynamic control over listing their property.

***For example, if a room closed due to maintenance, the lister would be able to update how many of that room type were available on any given date.

```
SELECT pk_calendar_id FROM bkg_CALENDAR WHERE bkg_CALENDAR.date BETWEEN '2019-12-01' AND '2019-12-03';

SELECT * FROM bkg_ROOM_AVAILABLE WHERE bkg_ROOM_AVAILABLE.fk_room_type_id = 2 AND bkg_ROOM_AVAILABLE.fk_calendar_id BETWEEN 1 and 3;

UPDATE `bkg_ROOM_AVAILABLE` SET `room_available_count` = '1' WHERE bkg_ROOM_AVAILABLE.fk_room_type_id = 2 AND bkg_ROOM_AVAILABLE.fk_calendar_id BETWEEN 1 and 3;

SELECT * FROM bkg_ROOM_AVAILABLE WHERE bkg_ROOM_AVAILABLE.fk_room_type_id = 2 AND bkg_ROOM_AVAILABLE.fk_calendar_id BETWEEN 1 and 3 ORDER BY `fk_room_type_id` ASC;
```

A room of a particular kind may no longer be available in a property and the lister may want to remove all details regarding this room from the site (it is assumed that, if there were bookings already created for this room type on the site, an application would prevent the lister from removing this until the booking has been fulfilled). When the room type is removed, we can see all details specific to that room type have been removed (including availability, bed type, breakfast) without affecting the data in the likes of the breakfast table that other records may be reliant on.

```
SELECT * FROM bkg_ROOM_TYPE WHERE pk_room_type_id = 6;

DELETE FROM bkg_ROOM_TYPE WHERE pk_room_type_id = 6;

SELECT * FROM bkg_ROOM_AVAILABLE WHERE fk_room_type_id = 6;

SELECT * FROM bkg_ROOM_AMENITY WHERE fk_room_type_id = 6;

SELECT * FROM bkg_ROOM_TYPE WHERE pk_room_type_id = 6;

SELECT * FROM bkg_ROOM_AVAILABLE WHERE fk_room_type_id = 6;

SELECT * FROM bkg_ROOM_AMENITY WHERE fk_room_type_id = 6;
```