

Introduction

The aim of this report is to design and create a database for an Airline Booking System. To implement this, I examined the online booking system of easyJet, a major European budget airline carrier. Through my examination of easyjet.com I carried out an Entity Discovery to help reverse-engineer how easyJet's database system may be structured.

EasyJet is a major airline carrier working in partnership with other companies that make up the wider Travel, Tourism and Overseas Business markets. While making an online booking, recommendations for Hotels, Car Hire, Travel Insurance etcetera are offered for customer convenience. To maintain the scope of this project aims, I have decided to focus solely on the specifics of flight purchases and other non-flight related extras have been omitted from the database. The database has been populated with a selection sample data that will be used to demonstrate the operation of the database using SQL queries.

Within this report I will discuss the design considerations that went in to the creation of the database, the key relationships involved in an airline booking system. I will then discuss how the database model was implemented and using SQL queries demonstrate its functionality.

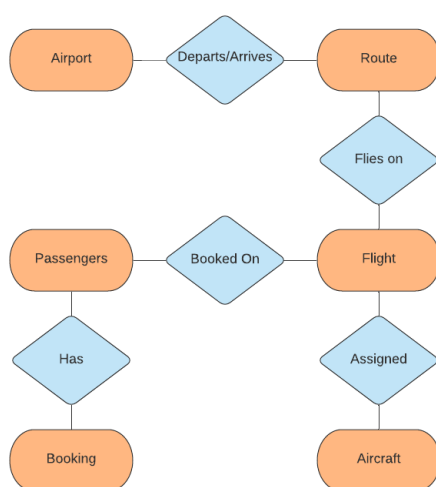
Design Considerations and Core Relationships

An Entity-Relationship (ER) model “adopts the more natural view that the real world consists of entities and relationships.” [1] An ER model for an airline booking database should reflect the real-world relationships and interactions that occur during that process.

They core relationships I have identified from an Airline Booking System is that:

1. A Booking is made for one or more Passengers
2. Passengers are booked on to a flight(s).
3. Flights have an assigned Aircraft and Route that takes them from a Departure Airport to an Arrival Airport.

I have demonstrated this below with an initial simple Entity-Relation Diagram:

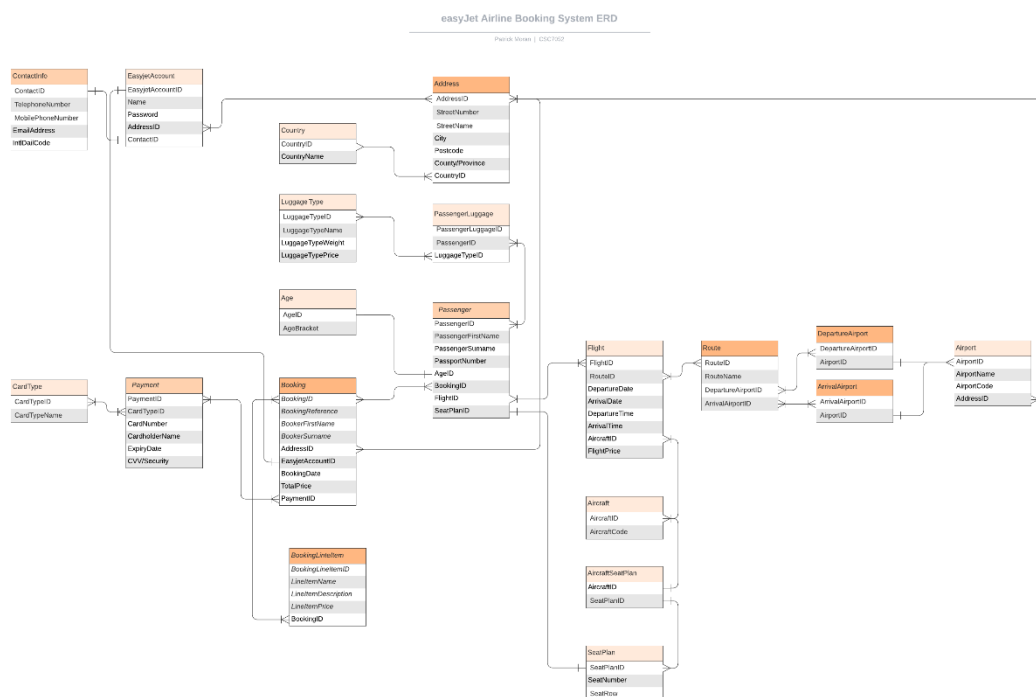


[2]

This simplified diagram demonstrates the primary relationship in an Airline Booking System; that is between Passengers and Flights.

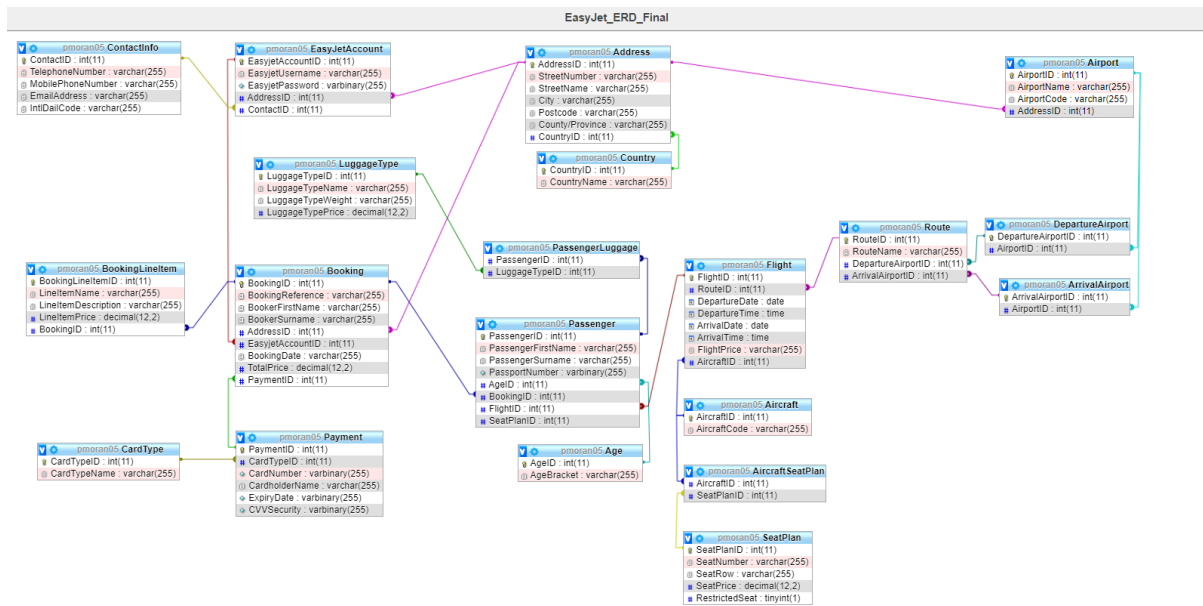
- Multiple bookers can create bookings for multiple Passengers onboard one or many flights, either as a single journey from Airport A-B, a return journey from Airport A-B and B-A or as part of a multi-stag journey from Airport-A-B-C.
- Each Flight has an assigned Route from a Departure Airport to an Arrival Airport, and an Aircraft assigned to it. Routes can be assigned multiple Flights and associated Aircraft to allow for concurrent flights to occur along a route.
- Each Booking is linked to an EasyJet Account, this allows for individual Bookers to make multiple bookings linked to the same account.
- Each Passenger in a booking can have multiple items of luggage of different weights and prices.
- Each Aircraft has a seat plan in which passengers are allocated seats.

Below is a finalised ER diagram that shows the Entities their Attributes and Primary Keys, and relationships between Entities, Attributes. Later I will examine some of these attributes, explaining the design decisions that were made in the creation of the database.



[3]

Below is a final design of the implemented database for the Airline Booking System. This design shows Entity Tables, its attributes and data types, Primary and Foreign Keys and Foreign Key Constraints. The design shows that almost every table has Primary Keys with Data Type Integer. **PassengerLuggage** and **AircraftSeatPlan** are the exceptions to this, in which they have Composite Key, this allows a Many to Many relationship to be formed between **Passenger** and **Luggage** tables, and **Aircraft** and **SeatPlan** tables.



[4]

Booking a Flight

Every Passenger needs a Booking to be made with payment before they can be confirmed on a flight. This booking is done by a single person on behalf of all passengers within their booking. *BookerFirstName* and *BookerSurname* allow Bookers to be referenced by name within a booking, but because of the potential for common or repeated names a randomly generated alpha-numeric *BookingReference* is also created, while this increases the probability of uniqueness within a booking it is not guaranteed method with potential for repeated character sequences possible.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	BookingID	int(11)			No	None		AUTO_INCREMENT
2	BookingReference	varchar(255)	latin1_swedish_ci		No	None		
3	BookerFirstName	varchar(255)	latin1_swedish_ci		No	None		
4	BookerSurname	varchar(255)	latin1_swedish_ci		No	None		
5	AddressID	int(11)			No	None		
6	EasyjetAccountID	int(11)			No	None		
7	BookingDate	varchar(255)	latin1_swedish_ci		No	None		
8	TotalPrice	decimal(12,2)			No	None		
9	PaymentID	int(11)			No	None		

[5]

Within the **Booking** Table, the Primary Key is *BookingID*, this incremented integer ensures that each booking has a unique identification within the system.

There are three Foreign Key constraints in the **Booking** Table: *AddressID*, *EasyjetAccountID* and *PaymentID*. This allows the Addresses within the database are normalised out to an **Address** Table, we will find that Airport addresses have also been normalised out to the same **Address** Table, so that all addresses are grouped and maintained together. This design choice can be seen implemented below, where *BookingID* '2' and '19' have both been made by *EasyjetAccountID* '1'.

Options		BookingID	BookingReference	BookerFirstName	BookerSurname	AddressID	EasyjetAccountID	BookingDate	TotalPrice	PaymentID
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete		2	f45e325	Charles	Calderon	27	1	25/11/2020	200.88	1
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete		19	s455sa88	Charles	Calderon	27	1	25/11/2020	129.98	1

[6#1]

When making a booking an EasyJet Account may be created, this allows the booker to store their details, have more access to manage their booking and create a history of linked past and future

bookings. The **EasyjetAccount** table has a PK EasyjetAccountID and two foreign key constraints with *AddressID* and *ContactID*. This has allowed each EasyjetAccount to be linked with an Address and Contact information for the Booker.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	EasyjetAccountID	int(11)			No	None		AUTO_INCREMENT
2	EasyjetUsername	varchar(255)	latin1_swedish_ci		No	None		
3	EasyjetPassword	varbinary(255)			No	None		
4	AddressID	int(11)			No	None		
5	ContactID	int(11)			No	None		

[7]

Within the **Contact** table we can see the Auto-Incrementing *ContactID* that allows the Foreign Key Constraint between the EasyjetAccount table. This table records either or both of a TelephoneNumber and MobilePhoneNumber. Within the Sample Data we have instances where people have one or the other, or the same number repeated, as people often have access to more than one contact number, in the form of a landline and a mobile phone number this allows both to be recorded if required. While it would be possible to consolidate this in to one PhoneNumber table, this would lead to a data loss where two different entries did exist. International Dialling Codes are unique to countries, we can see some similar data types where both 0044 and +44 are representative of a UK based phone number. As an improvement on this design choice *IntlDialCode* could be normalised out, and linked to the **Country** table.

ContactID	TelephoneNumber	MobilePhoneNumber	EmailAddress	IntlDailCode
1		07712522405	CharlesJCalderon@armyspy.com	0044
3		078 1954 4188	DarrenMWeaver@jourrapide.com	+44
4		078 4121 6712	PearlGBrewer@dayrep.com	+44
5	077 0205 0372	077 0205 0372	MarthaCThompson@armyspy.com	+44
6		078 4081 5960	MatildaFraser@dayrep.com	+44
7	04.31.14.42.24		ZdenekPerillard@jourrapide.com	33
8	69 599 45 36		WiktoriaCzerwinska@armyspy.com	48

[8#2]

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
<input type="checkbox"/> 1	PassengerID	int(11)			No	None		AUTO_INCREMENT
<input type="checkbox"/> 2	PassengerFirstName	varchar(255)	latin1_swedish_ci		No	None		
<input type="checkbox"/> 3	PassengerSurname	varchar(255)	latin1_swedish_ci		No	None		
<input type="checkbox"/> 4	PassportNumber	varbinary(255)			No	None		
<input type="checkbox"/> 5	AgeID	int(11)			No	None		
<input type="checkbox"/> 6	BookingID	int(11)			No	None		
<input type="checkbox"/> 7	FlightID	int(11)			No	None		
<input type="checkbox"/> 8	SeatPlanID	int(11)			No	None		

[9]

As every booking can have multiple Passengers there is an FK constraint of *BookingID* in the Passenger table, this allows all Passengers to be linked to a booking. Within the **Passenger** Table the Primary Key *PassengerID* creates uniqueness for each person within the passenger table. *FlightID* is a Foreign Key constraint between the **Passenger** and **Flight** tables as a passenger, this allows the same person to be a passenger on multiple flights. EasyJet has three Age Brackets for passengers on their flights; Infant Under 2 years old, Child 2-15 years old and adult 16 years +, this Age table has been normalised out with an FK constraint between *AgeID* in the **Passenger** and **Age** tables.

All Passengers on an EasyJet flight are allowed 1 piece of Hand Luggage which can be brought in to the Cabin with them. For an additional charge they can bring extra Hold Luggage or Sports Equipment. To allow a passenger to book multiple items of luggage I created a **PassengerLuggage** table which acts as a composite key between the **Passenger** and the **LuggageType**.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	PassengerID		int(11)	No	None			Change Drop More
<input type="checkbox"/>	2	LuggageTypeID		int(11)	No	None			Change Drop More

[10]

The **LuggageType** Table gives a breakdown of the different weight restrictions and prices of individual items of Luggage that it is possible to select as part of a booking. EasyJet offer a greater variety of options for sports equipment within 3 set price bands. For simplification of the data I have compounded these in to three options of Large, Medium and Small Sports Equipment.

LuggageTypeID	LuggageTypeName	LuggageTypeWeight	LuggageTypePrice
1	Cabin Baggage	No Weight Restriction	0.00
2	Hold Baggage	15KG	23.24
3	Hold Baggage	23KG	25.74
4	Hold Baggage	26KG	37.74
5	Hold Baggage	29KG	49.74
6	Hold Baggage	32KG	61.74
7	Large Sports Equipment		45.00
8	Medium Sports Equipment		37.00
9	Small Sports Equipment		32.00

[11#3]

The **BookingLinelItem** table allows each Booking to be connected with an itemised list of everything in the booking that has a cost value. These are linked through the *BookingID* FK constraint. The **BookingLinelItem** table contains the PK *BookingLinelItemID*, *LinelItemName*, *LinelItemDescription* and a *LinelItemPrice*.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	BookingLinelItemID	int(11)			No	None		AUTO_INCREMENT
2	LinelItemName	varchar(255)	latin1_swedish_ci		No	None		
3	LinelItemDescription	varchar(255)	latin1_swedish_ci		No	None		
4	LinelItemPrice	decimal(12,2)			No	None		
5	BookingID	int(11)			No	None		

[12]

The *LinelItemPrice* is not constrained by any other tables where Price is an attribute, such as Flight, **SeatPlan** or **LuggageType**. This allows each booking to have a static payment price from the time they made the booking, that is independent of any variations in pricing made by the Airline Carrier.



	BookingLinelItemID	LinelItemName	LinelItemDescription	LinelItemPrice	BookingID
<input type="checkbox"/> Edit Copy Delete	63	Belfast-Krakow	Flight from Belfast to Krakow	24.99	14
<input type="checkbox"/> Edit Copy Delete	64	Krakow-Belfast	Flight from Krakow to Belfast	34.99	14
<input type="checkbox"/> Edit Copy Delete	65	Cabin Baggage	One item of carry-on hand luggage	0.00	14
<input type="checkbox"/> Edit Copy Delete	66	29KG Hold Baggage	One item of hold luggage max 29kg	49.24	14
<input type="checkbox"/> Edit Copy Delete	67	Standard Seat	Pre-booked Seat in Rows 14-31	4.99	14
<input type="checkbox"/> Edit Copy Delete	68	Cabin Baggage	One item of carry-on hand luggage	0.00	14
<input type="checkbox"/> Edit Copy Delete	69	29KG Hold Baggage	One item of hold luggage max 29kg	49.24	14
<input type="checkbox"/> Edit Copy Delete	70	Standard Seat	Pre-booked Seat in Rows 14-31	4.99	14

☐ Check all With selected: [Edit](#) [Copy](#) [Delete](#) [Export](#)

+ Options
TotalItemsOrdered
168.44

[13#4]

In the above example we can see the *BookingLineItem* table with the inset Sum Query for the *LineItemPrice* for *BookingID* '14'. This Booking contains a Flights from Belfast to Krakow at £24.99, Krakow to Belfast at £34.99. Hold Baggage has been booked on both legs of the journey at £49.24 each and a Standard Seat at £4.99 has been booked for each flight.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	PaymentID 	int(11)			No	None		AUTO_INCREMENT
2	CardTypeID 	int(11)			No	None		
3	CardNumber	varbinary(255)			No	None		
4	CardholderName	varchar(255)	latin1_swedish_ci		No	None		
5	ExpiryDate	varbinary(255)			No	None		
6	CVVSecurity	varbinary(255)			No	None		

[14]

The **Payment** table is linked to the **Booking** table through its Primary Key. CardType has been normalised out to a **CardType** table, covering Mastercard, Visa Debit, American Express etc, as this is non-key data to the Payment table.

While in many instances *CardholderName* will be similar to *BookerFirstName* and *BookerSurname* from the **Booking** table, by design I decided to keep this attributes unlinked, as a card belonging to be a different person or with a name variation could be used on their Bank Information.

The table below shows an SQL query for the **Booking** Table and **Payment** Table. This shows the relationship between individual bookings referenced by *BookingID* and the *PaymentID*. *PaymentID* '5' shows that *Martha C Thompson* has made two separate payments for *BookingID* '10' and '22'.

BookingID	BookingReference	TotalPrice	PaymentID	CardNumber	CardholderName	ExpiryDate	CVVSecurity
2	f455e325	200.88	1	0x21b748dfb32ec71425f45c6aaa98ae5e9a873ed24c3d046b...	Charles Calderon	0xe7d046ddc7f2912192fed28841de04ec	0xf20bf5aafa9bc8b07f1f3f014566424e
4	zqe9i0	76.96	2	0xeb263471c750df98d9263253c121c0953f1a1a16db4fad2...	Darren M Weaver	0x24bbb1d0cd5324d3b34eb3418711fe2	0xc0bbd8d03af732ad8fa335c32353c42be
6	7wognjrr	381.36	4	0x0bbc9b7f1da6f55365a98f54eb8ecf3a3b70759dec610c46...	Pearl Brewer	0xb0b2c3acc6b519e18f20fe80a1f34c18	0xeb8743c631430618c9a4173867802c56
10	kynzq96f	156.44	5	0x7ab1e608138f56936fb95c511d679c08cac55928abb67457...	Martha C Thompson	0x4efb10a8899dbfb8ea78be4131b1700	0xc3178b350537ae6956f4b8fb2db53765
11	tovh12ev	237.92	6	0xb96d7f1846f999dea510b35ac20a446070c08e0b25e6485...	Malida Fraser	0x0ea18582c954db72a7037f6ee36367	0x363b9e770c1c1cfa5c7c4ac31058db5
13	ls9um8jo	249.86	7	0xdbc3470eb0107083020b7327ace2398272a212159ba291...	Zdenek Ferliard	0xf5a948b878c2927d6639026f4cbe2d	0x27071f37f26c3e35ee8e6f9a9804
14	xq554d8	168.44	8	0x8acc027d9f9036ade294ade00107736ecbb1a2627e6674...	Wiktoria Czerwinska	0xe940811c696ee18e13b13bbe861b9c29	0xe0e533e4d800503993c079336b8ea75
15	azaijglt	143.44	9	0x7ce21eb28beaa31ecd56e684724cb15c1c9ea682abb77b...	Katrin Frei	0x51e64afe18909574fd3130add56755a	0xdaaca323b519f13107922407b6de021
16	17ba0ciu	451.82	10	0x912a01be23d9ce0ad9147be960b7c1b70e052e1b7f118b...	Erin Alexander	0xb4e25bdf1157d2a03c5d14a6f48dca9	0xc2c8b5b8cbbf9dce8d51a5e76f9c4b
17	zmvvofrr	103.96	11	0x97f7f7af34181a888421aa488c8b05ea27f44588dc6374...	Theodore Bergeron	0x3b498726bf1e502ca0601b67153c83e3	0xc6cf955059a89a4281ec6f1bc30664
18	yu85vst6	118.96	12	0xc60afe2c9c2f599a0adca3c930a48172399fe914daafdd2...	Coby Docherty	0x8240280e2e6ea29eb5fd4af991a079d	0x6063c32cfd728529291ee92214ce702
19	s455ea88	129.98	1	0x21b748dfb32ec71425f45c6aaa98ae5e9a873ed24c3d046b...	Charles Calderon	0xe7d046ddc7f2912192fed28841de04ec	0xf20bf5aafa9bc8b07f1f3f014566424e
20	wfb89g12	179.40	2	0xeb263471c750df98d9263253c121c0953f1a1a16db4fad2...	Darren M Weaver	0x24bbb1d0cd5324d3b34eb3418711fe2	0xc0bbd8d03af732ad8fa335c32353c42be
22	hk15f735	137.92	5	0x7ab1e608138f56936fb95c511d679c08cac55928abb67457...	Martha C Thompson	0x4efb10a8899dbfb8ea78be4131b1700	0xc3178b350537ae6956f4b8fb2db53765

[15#5]

As security of Personal Data is a legal requirement the *CardNumber*, *ExpiryDate* and CVV/Security number have been encrypted using AES Encryption within the database. This method of encryption has also been used within the **EasyjetAccount** and **Passenger** Table, to encrypt *EasyjetPasswords* and *PassportNumber*.

EasyjetAccountID	EasyjetPassword	PassengerID	PassportNumber
1	0x266388b8c4043d15406d3da0a81a2335	1	0xba8b556c333cbf262a932cfdd44ce089
3	0x425c1244076e6a0d9b4f62da471bf6d1	2	0xf7a6a3b1a38275cca429c7941a1fd8cd
6	0x7f86b4ef427af0e2d6e0471a5a81fb44	3	0xba8b556c333cbf262a932cfdd44ce089
9	0xc4024cef78c1c5e09b93d6699fb7ab5e	4	0xf7a6a3b1a38275cca429c7941a1fd8cd
10	0xc412148265fb754ebd1c2238a56b512b	5	0x37297d7df832a0dcf21f7d853e52ae57

[17#7]

For each of these Encrypted Data types a unique Encryption Key has been used to increase security and lessen secure data loss in the event of a leak of an individual key.

Flights, Routes and Airports

All Passengers are linked to a Flight through the *FlightID* FK constraint. The Flight Table contains a *DepartureDate*, *DepartureTime*, *ArrivalDate* and *ArrivalTime* and a *FlightPrice*.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
<input type="checkbox"/> 1	FlightID	int(11)			No	None		AUTO_INCREMENT
<input type="checkbox"/> 2	RouteID	int(11)			No	None		
<input type="checkbox"/> 3	DepartureDate	date			No	None		
<input type="checkbox"/> 4	DepartureTime	time			No	None		
<input type="checkbox"/> 5	ArrivalDate	date			No	None		
<input type="checkbox"/> 6	ArrivalTime	time			No	None		
<input type="checkbox"/> 7	FlightPrice	varchar(255)	latin1_swedish_ci		No	None		
<input type="checkbox"/> 8	AircraftID	int(11)			No	None		

[18]

Each Flight has an individual price. These prices are set by the Airline Company based on a number of variables, such as Demand, Fuel Costs, Time of Flight and Date of Travel. As such these prices can be updated by the Airline company to reflect changes in what they consider the cost Value of the flight. This could be in the form of reduction in price during a Sale Event or an increase in cost as the flight date approaches. The table below shows variance in prices of flights on the same route.

FlightID	RouteID	DepartureDate	DepartureTime	ArrivalDate	ArrivalTime	FlightPrice	AircraftID
1	1	2020-12-06	07:30:00	2020-12-06	08:05:00	24.99	1
2	1	2020-12-06	12:30:00	2020-12-06	13:05:00	36.99	1
3	1	2020-12-06	17:30:00	2020-12-06	18:05:00	17.99	1

[19#8]

Each **Flight** is also linked to the **Aircraft** table through the *AircraftID* FK constraint. This allows a change in the data to occur if an Aircraft scheduled on a flight was replaced with another.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
<input type="checkbox"/> 1	AircraftID	int(11)			No	None		AUTO_INCREMENT
<input type="checkbox"/> 2	AircraftCode	varchar(255)	latin1_swedish_ci		No	None		

[20]

Each Flight travels along a set Route, this **Route** has been normalised out through the *RouteID* FK constraint so that multiple flights can be linked to the same route. This can be seen below where FlightID '1', '2' and '3' are all linked to *RouteID* '1' BFSGLA (Belfast-Glasgow) Route. These flights are all on the same *DepartureDate* and *ArrivalDate*, in this case the 2020-Dec-06.

FlightID	RouteID	DepartureDate	DepartureTime	ArrivalDate	ArrivalTime	FlightPrice	AircraftID	RouteID	RouteName	DepartureAirportID	ArrivalAirportID
1	1	2020-12-06	07:30:00	2020-12-06	08:05:00	24.99	1	1	BFSGLA	1	2
2	1	2020-12-06	12:30:00	2020-12-06	13:05:00	36.99	1	1	BFSGLA	1	2
3	1	2020-12-06	17:30:00	2020-12-06	18:05:00	17.99	1	1	BFSGLA	1	2
4	2	2020-12-06	08:30:00	2020-12-06	09:05:00	24.99	1	2	GLABFS	2	1

[21#9]

An Aircraft can be assigned to different Flights and Routes. This can be demonstrated above, where the 4 flights listed are all completed by *AircraftID* '1'. In this case the Aircraft is completing multiple Return Journeys between Belfast-Glasgow and Glasgow-Belfast, with a time difference between *ArrivalTime* of one flight and *DepartureTime* of the return leg to allow for passenger disembarkation and embarkation before the next departure.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	AirportID	int(11)			No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/> 2	AirportName	varchar(255)	latin1_swedish_ci		No	None			Change Drop More
<input type="checkbox"/> 3	AirportCode	varchar(255)	latin1_swedish_ci		No	None			Change Drop More
<input type="checkbox"/> 4	AddressID	int(11)			No	None			Change Drop More

[22]

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
<input type="checkbox"/> 1	RouteID	int(11)			No	None		AUTO_INCREMENT
<input type="checkbox"/> 2	RouteName	varchar(255)	latin1_swedish_ci		No	None		
<input type="checkbox"/> 3	DepartureAirportID	int(11)			No	None		
<input type="checkbox"/> 4	ArrivalAirportID	int(11)			No	None		

[23]

The **Route** table and **Airport** Table are linked through two intermediary tables **DepartureAirport** and **ArrivalAirport**. This allows all Airports to be linked to either end of a route. In the table below we can see four different Routes, all with *ArrivalAirportID* '1' which is associated with Belfast International Airport. The **DepartureAirport** and **ArrivalAirport** tables are linked to the **Airport** Table through the *AirportID FK constraint*.

RouteID	RouteName	DepartureAirportID	ArrivalAirportID
2	GLABFS	2	1
4	LTNBFS	3	1
11	BCNBFS	6	1
16	KRKBFS	10	1

[24#10]

Both Airports and Passengers have Addresses as an Attribute. I have captured Address information for both in the **Address** Table, this decision was made because at a base level all addresses are made up of similar structures. The Primary Key *AddressID* is used as a Foreign Key Constraint between both the **EasyjetAccount** and the **Airport** tables. Within the Address table I have captured *StreetNumber*, *StreetName*, *City*, *Postcode*, *County/Province* and *CountryID*. This highlighted a number of discrepancies in how addresses work in different Countries, and for entities such as Personal Home Addresses and Business Addresses for large entities like Airports. As we can see from the table below, *AddressID* '8' is the entry for Edinburgh Airport, it does not have a listed *StreetNumber* or *StreetName*, however, the PK does allow meaningful connection between it and the **Airport** table.

AddressID	StreetNumber	StreetName	City	Postcode	County/Province	CountryID
7	1-3	FLUGHAFENSTR.	HAMBURG	22335	HAMBURG	5
8			EDINBURGH	EH12 9DN	MIDLOTHIAN	2
9		AVENUE CHARLES DE GAULLE	PARIS	95700	ROISSY-EN-FRANCE	4
26	1	KAPITANA MIECZYSLAWA MEDWECKIEGO	BALICE	32-083	WOJEWÓDZTWO MAŁOPOLSKIE	7
27	91	Layburn Court	Leeds	LS24 5JN	yorkshire	3

[25#11]

CountryID has been normalised out within the Address table. This decision was taken as almost every country has an airport, excluding the Principalities of Monaco, Andorra and Liechtenstein and the City States of Vatican City, San Marino. Passengers from any country could book a flight also. *City* could have been normalised out in the same way, but I took the decision against this for clarity, particularly around airport addresses, where on occasion Airports will be named after the nearest major city, but isn't located in the city itself, such as *AddressID* '26' above, which is linked to Krakow Airport, but is located in the city of Balice.

The table below shows a Join of the **Country** table and the **Address** table for CountryID '3' which represents England. Here we can see some repetition within the County/Province column, where two instances of Surrey are occurring. As Counties/Provinces have different relative meanings across different countries have different relative meanings across different countries, I decided that this would be a difficult process to normalise in a meaningful way.

CountryID	CountryName	AddressID	StreetNumber	StreetName	City	Postcode	County/Province	CountryID
3	England	3			MANCHESTER	M90 1QX	LANCASHIRE	3
3	England	4			LUTON	LU2 9LY	BEDFORDSHIRE	3
3	England	27	91	Layburn Court	Leeds	LS24 5JN	yorkshire	3
3	England	28	19	Font Street	Hendon	SR2 8EY	Surrey	3
3	England	32	57	Town Lane	South Godstone	RH9 1FP	Surrey	3

[26#12]

Potential Improvements to the Design:

I will now highlight a number of potential improvements to the design that became apparent through testing, but was unable to implement due to time constraints.

EasyJet allow passengers to select a seat for a Set Price or to be randomly allocated a seat at no extra charge. Within my database, there is no distinction on whether a seat was selected by the passenger or randomly allocated. Therefore, in the **BookingLineItem** table selected seats have been displayed with a cost price. As an improvement on this database, I would create a **SeatSelection** table, this would potentially have *SelectedSeat* attribute which would have an entry either True or False, if True, then the seat selected by the passenger would have an associated price which would appear in the **BookingLineItem** table, otherwise there would be no additional price for the seat they are randomly allocated based on available seats remaining from the **AircraftSeatPlan**.

I believe the **SeatPlan** design could also be improved upon by creating a **SeatType** table, this would allow smaller blocks of SeatNumbers and rows to be grouped together with linked prices, such as "Extra Legroom", "Up Front" and "Standard", this would potentially make the **SeatPlan** table more manageable.

The **LuggageType** table could be further normalised out in to **HoldBaggageWeight** and **SportsEquipmentType** so that there is no repetition of data in the form of Hold Baggage and Sports Equipment.

The **Aircraft** table currently only holds the **AircraftID** and **AircraftCode**, this table could be expanded to include other attributes of the Aircraft that might be applicable such as **AircraftModel** Currently EasyJet operates a fleet exclusively of Airbus A320 aircraft, so I felt that this would be redundant information within the scope of this project, but it is feasible that as Aircraft reach the end of their service life, older models will be replaced with newer models, meaning there could be a need for Aircraft models to be stored. **AircraftDateOfRegistration** could also be important information to store within this database, so that EasyJet is aware of the Age of the aircraft within their fleet, and able to plan and upgrade as necessary.

Conclusion

In conclusion, the database design I have created meets the requirements of an Online Booking System such as used by EasyJet. Bookings are able to be made on behalf of multiple passengers, additions to the booking can be made such as selected seats and additional luggage. Foreign Key Constraints help to maintain the data in a more manageable format which can be analysed in isolation

from the database as whole. Payment details and private Personal data and can be taken and stored securely using AES encryption, providing customers with a level of protection of their information, distinct Encryption Keys have been used for each encrypted data type to increase security within the database. For the purposes of demonstration these Encryption Keys have been kept relevant to the datatype which isn't as secure a method as possible. Account Details can be stored within a UserAccount to create a Booking History to be created. Aircraft can be added to Flights and Routes which Depart and Arrive at Airports. The design allows EasyJet to update their route map, by adding new Airports and Routes to the database. While there are some limitations in the database which could be improved upon, the database system does function as it is intended.

Appendix

[1] Chen, Peter (March 1976). "The Entity-Relationship Model - Toward a Unified View of Data". ACM Transactions on Database Systems

[2] Initial ER Diagram *created on lucid.app/lucidchart by P.Moran*

[3] Finalised ER Diagram based on initial group Entity Discovery *created on lucid.app/lucidchart by P.Moran*

[4] Final Database Design implemented through phpMyAdmin using MySQL

[5] Booking Table

[6#1] SQL Query showing all Bookings related to the same EasyjetAccount:

```
SELECT * FROM `Booking` WHERE `EasyjetAccountID` = 1;
```

[7] EasyjetAccount Table

[8#2] A selection of data from the ContactInfo Table:

```
SELECT * FROM `ContactInfo`;
```

[9] Passenger Table

[10] PassengerLuggage Table

[11#3] Data from the LuggageType table:

```
SELECT * FROM `LuggageType`;
```

[12] BookingLinelItem Table

[13#4] Entry for BookingLinelItem table for BookingID '14', with inset in Yellow box of SQL query of Sum of LinelItemPrice.

```
SELECT `BookingLinelItemID`, `LinelItemName`, `LinelItemDescription`, `LinelItemPrice`,  
`BookingID` FROM `BookingLinelItem` WHERE `BookingID`=14;
```

```
SELECT SUM(LinelItemPrice) AS TotalItemsOrdered FROM BookingLinelItem WHERE BookingID  
= 14;
```

[14] Payment Table

[15#5] Join of Booking Table and Payment Table showing Encryption of Data

```
SELECT * FROM Booking INNER JOIN Payment ON Booking.`PaymentID` =  
Payment.`PaymentID`;
```

[16#6] EasyjetAccount Table, showing encryption of EasyJetPassword

```
SELECT EasyjetAccountID, EasyjetPassword FROM EasyJetAccount;
```

[17#7] Passenger Table, showing encryption of PassportNumber

```
SELECT PassengerID, PassportNumber FROM Passenger;
```

[18] Flight Table

[19#8] Flight Table where RouteID is '1'

```
SELECT * FROM `Flight` WHERE ROUTEID=1
```

[20] Aircraft Table

[21#9] Selection of data from Inner Join of Flight and Route tables

```
SELECT * FROM `Flight` INNER JOIN Route ON Flight.RouteID = Route.RouteID;
```

[22] Airport Table

[23] Route Table

[24#10] Route Table for routes with ArrivalAirportID '1'

```
SELECT * FROM `Route` WHERE ArrivalAirportID = 1;
```

[25#11] Sample of Data from the Address Table

```
SELECT * FROM `Address`;
```

[26#12] Tables showing join between Country Table and Address Table, where CountryID is '3'

```
SELECT * FROM Country RIGHT JOIN Address ON Country.CountryID = Address.CountryID  
WHERE Country.CountryID = 3
```

Demonstration SQL Queries

#Booking a flight

```
SELECT * FROM `Flight` WHERE `RouteID` = 1;
```

```
SELECT * FROM `Flight` WHERE `RouteID` = 2;
```

#Adding Bookers Address details

```
INSERT INTO `Address` (`AddressID`, `StreetNumber`, `StreetName`, `City`, `Postcode`,  
`County/Province`, `CountryID`) VALUES (NULL, '63', 'Friar Street', 'Motherwell', 'ML1 1UG', 'South  
Lanarkshire', '2');
```

#Adding Payment Details

```
INSERT INTO `Payment` (`PaymentID`, `CardTypeID`, `CardNumber`, `CardholderName`, `ExpiryDate`,  
`CVVSecurity`) VALUES (NULL, '1', '5167 9112 7181 1958', 'Andrew Wood', '09/2025', '147');
```

```
UPDATE Payment SET CardNumber = AES_ENCRYPT('5167 9112 7181 195', 'paymentSecretKey')  
WHERE PaymentID = 13;
```

```
UPDATE Payment SET CVVSecurity = AES_ENCRYPT('147', 'CVVSecretKey') WHERE PaymentID = 13;
```

```
UPDATE Payment SET ExpiryDate = AES_ENCRYPT('09/2025', 'ExpirySecretKey') WHERE PaymentID =  
13;
```

#Adding Contact Info

INSERT INTO `ContactInfo` (`ContactID`, `TelephoneNumber`, `MobilePhoneNumber`, `EmailAddress`, `IntlDailCode`) VALUES (NULL, '336 58458', '077 0702 8432', 'AndrewWood@armyspy.com', '+44');

#Adding EasyJetAccount

INSERT INTO `EasyJetAccount` (`EasyjetAccountID`, `EasyjetUsername`, `EasyjetPassword`, `AddressID`, `ContactID`) VALUES (NULL, 'Woodsy80', 'niezoo4Exae', '39', '13');

UPDATE EasyJetAccount SET EasyjetPassword = AES_ENCRYPT('niezoo4Exae','PasswordSecretKey') WHERE EasyjetAccountID = 18;

#Adding Booker Details

INSERT INTO `Booking` (`BookingID`, `BookingReference`, `BookerFirstName`, `BookerSurname`, `AddressID`, `EasyjetAccountID`, `BookingDate`, `TotalPrice`, `PaymentID`) VALUES (NULL, '2Y32H085', 'Andrew', 'Wood', '39', '18', '2020/12/01', '0.00', '13');

#Adding Passengers

INSERT INTO `Passenger` (`PassengerID`, `PassengerFirstName`, `PassengerSurname`, `PassportNumber`, `AgeID`, `BookingID`, `FlightID`, `SeatPlanID`) VALUES (NULL, 'Andrew', 'Wood', '324251537', '3', '23', '1', '44');

INSERT INTO `Passenger` (`PassengerID`, `PassengerFirstName`, `PassengerSurname`, `PassportNumber`, `AgeID`, `BookingID`, `FlightID`, `SeatPlanID`) VALUES (NULL, 'Kirsty', 'Wood', '564326593', '3', '23', '1', '45');

UPDATE Passenger SET PassportNumber = AES_ENCRYPT('324251537','PassportSecretKey') WHERE PassengerID = 52;

UPDATE Passenger SET PassportNumber = AES_ENCRYPT('564326593','PassportSecretKey') WHERE PassengerID = 53;

INSERT INTO `Passenger` (`PassengerID`, `PassengerFirstName`, `PassengerSurname`, `PassportNumber`, `AgeID`, `BookingID`, `FlightID`, `SeatPlanID`) VALUES (NULL, 'Andrew', 'Wood', '324251537', '3', '23', '4', '80');

INSERT INTO `Passenger` (`PassengerID`, `PassengerFirstName`, `PassengerSurname`, `PassportNumber`, `AgeID`, `BookingID`, `FlightID`, `SeatPlanID`) VALUES (NULL, 'Kirsty', 'Wood', '564326593', '3', '23', '4', '81');

UPDATE Passenger SET PassportNumber = AES_ENCRYPT('324251537','PassportSecretKey') WHERE PassengerID = 54;

UPDATE Passenger SET PassportNumber = AES_ENCRYPT('564326593','PassportSecretKey') WHERE PassengerID = 55;

#Adding Luggage

INSERT INTO `PassengerLuggage` (`PassengerID`, `LuggageTypeID`) VALUES ('52', '1');

INSERT INTO `PassengerLuggage` (`PassengerID`, `LuggageTypeID`) VALUES ('52', '2');

INSERT INTO `PassengerLuggage` (`PassengerID`, `LuggageTypeID`) VALUES ('53', '1');

INSERT INTO `PassengerLuggage` (`PassengerID`, `LuggageTypeID`) VALUES ('53', '3');

INSERT INTO `PassengerLuggage` (`PassengerID`, `LuggageTypeID`) VALUES ('54', '1');

INSERT INTO `PassengerLuggage` (`PassengerID`, `LuggageTypeID`) VALUES ('54', '2');

INSERT INTO `PassengerLuggage` (`PassengerID`, `LuggageTypeID`) VALUES ('55', '1');

INSERT INTO `PassengerLuggage` (`PassengerID`, `LuggageTypeID`) VALUES ('55', '3');

#BookingLineltem

INSERT INTO `BookingLineltem` (`BookingLineltemID`, `LineltemName`, `LineltemDescription`, `LineltemPrice`, `BookingID`) VALUES (NULL, '1 Adult, BFS-GLA', 'Flight from Belfast to Glasgow', '24.99', '23');

INSERT INTO `BookingLineltem` (`BookingLineltemID`, `LineltemName`, `LineltemDescription`, `LineltemPrice`, `BookingID`) VALUES (NULL, '1 Adult, BFS-GLA', 'Flight from Belfast to Glasgow', '24.99', '23');

INSERT INTO `BookingLineltem` (`BookingLineltemID`, `LineltemName`, `LineltemDescription`, `LineltemPrice`, `BookingID`) VALUES (NULL, 'Standard Seat', 'Pre-booked Seat in Rows 7-9', '6.99', '23');

INSERT INTO `BookingLineltem` (`BookingLineltemID`, `LineltemName`, `LineltemDescription`, `LineltemPrice`, `BookingID`) VALUES (NULL, 'Standard Seat', 'Pre-booked Seat in Rows 7-9', '6.99', '23');

INSERT INTO `BookingLineltem` (`BookingLineltemID`, `LineltemName`, `LineltemDescription`, `LineltemPrice`, `BookingID`) VALUES (NULL, 'Cabin Baggage', 'One item of carry-on hand luggage', '0.00', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, '15KG Hold Baggage', 'One item of hold luggage max 15kg', '23.24', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, 'Cabin Baggage', 'One item of carry-on hand luggage', '0.00', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, '23KG Hold Baggage', 'One item of hold luggage max 23kg', '25.74', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, 'Cabin Baggage', 'One item of carry-on hand luggage', '0.00', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, '1 Adult, GLA-BFS', 'Flight from Belfast to Glasgow', '24.99', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, '1 Adult, GLA-BFS', 'Flight from Belfast to Glasgow', '24.99', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, 'Standard Seat', 'Pre-booked Seat in Rows 14-31', '4.99', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, 'Cabin Baggage', 'One item of carry-on hand luggage', '0.00', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, '15KG Hold Baggage', 'One item of hold luggage max 15kg', '23.24', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, 'Cabin Baggage', 'One item of carry-on hand luggage', '0.00', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, '23KG Hold Baggage', 'One item of hold luggage max 23kg', '25.74', '23');

INSERT INTO `BookingLinItem` (`BookingLinItemID`, `LinItemName`, `LinItemDescription`, `LinItemPrice`, `BookingID`) VALUES (NULL, 'Cabin Baggage', 'One item of carry-on hand luggage', '0.00', '23');

SELECT * FROM `BookingLineItem` WHERE `BookingID` =23;

SELECT SUM(LineItemPrice) AS TotalItemsOrdered FROM BookingLineItem WHERE BookingID = 23;

UPDATE `Booking` SET `TotalPrice` = '216.89' WHERE `Booking`.`BookingID` = 23;

#FlightPriceUpdate

SELECT FlightID, FlightPrice, `DepartureDate`, NOW(), DATEDIFF(`DepartureDate`,NOW()) FROM Flight
WHERE DATEDIFF(`DepartureDate`,NOW()) <7;

UPDATE Flight SET FlightPrice = FlightPrice +(FlightPrice*20/100) WHERE
DATEDIFF(`DepartureDate`,NOW()) <7;

SELECT FlightID, FlightPrice, `DepartureDate`, NOW(), DATEDIFF(`DepartureDate`,NOW()) FROM Flight
WHERE DATEDIFF(`DepartureDate`,NOW()) <7;