An overview of the database design ERD

A list of any reasonable design assumptions you have made

An explanation of some of the primary keys, foreign key constraints between tables and the data

types used in your database;

A discussion and justification for some of the important design decisions you made;

A discussion and justification for some of the important normalisation decisions you

made;

A discussion and justification for some improvements you might make to the design;

Evidence of your SQL queries (these can simply be pasted into the appendix). The

report is essentially a wider extension of the work you need to compete for your video demo. You are

welcome to re-use content prepared for your individual video demo.

Justifications:

* Date fields e.g. booking start and end or card valid from and valid to have no validation to ensure that the end date is after start date
  + This would rely on implementation from the front end developer to ensure this was correctly validated
* Region table
  + removed as jet2 has a flexible definition of country, ie it has Balearics as a country and not a region. Ibiza is nested under Balearics, not under spain
* Different addresses may be something to talk about in improvements
* Timezones are in British time and probably localised in front-end (?)
* *gps\_address* “wraps” address for entities that require both an address and a gps location
* booking duration encoded by flight datetimes?
* Booking contact stores more info than passenger, as only 1 passenger has to provide the extra fields
* Talk about how decisions were based on website
* Talk about gps/ address relationship
* Image\_url text type as no upper size limit on url
  + <https://www.sistrix.com/ask-sistrix/technical-seo/site-structure/url-length-how-long-can-a-url-be>
* Star rating
* Flight pricing via route, and pricing can be adjusted for time of day and time via a datetime range. I wanted to place price here, as it allows quick addition of new flights
  + Timeline

    Description automatically generated
* A reviewer must be a passenger
* Hotel facility types
  + Graphical user interface, text, application, email

    Description automatically generated
  + Appear to be same across various hotels
* Hotel facilities and room facilities don’t actually have much in common
  + Hotel facilities have images, names, types and descriptions
  + Room facilities is a simple list of strings
    - Room facilties seems to be room type specific, and can be seen being used in multiple different hotel room types
    - 2 diff hotels:
    - A picture containing text, indoor, screenshot

      Description automatically generated
    - Graphical user interface, application

      Description automatically generated
* Hotels have a couple of extra bits of information
  + Additional info – nullable – not always present
  + Background pattern

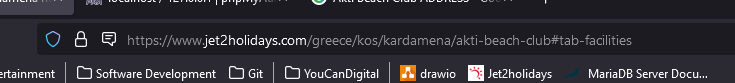
    Description automatically generated with low confidence
  + Figure 1note no additional info (its nullable) Background pattern

    Description automatically generated with low confidence
* Foreign keys match their corresponding primary key, and are kept distinct via their fully qualified name, eg, room\_image.image\_id vs image.image\_id
* Reviews are from trip advisor, and are external to jet2s databvase, likely accessed via an api
* Prices stored as dec(13,4) as per this allows very expensive rooms potentially, and uses a fixed point decimal arithmetic library after MySQL 5.0, making currency calculations very accurate and not prone to rounding errors
* <https://www.globalpaymentsintegrated.com/en-us/blog/2019/11/25/pci-rules-for-storing-credit-card-numbers-in-a-database> “It is important to note that these statements apply to Cardholder Data (16-digit Primary Account Number, expiration date, cardholder name), and do not apply to Sensitive Authentication Data (Track Data, PIN, PIN Block, CVV). Sensitive Authentication Data (SAD) can never be stored after authorization.”
* recent searches are simply stored on local storage by website and not in the database.
  + Graphical user interface, text

    Description automatically generated
  + Graphical user interface, text, application

    Description automatically generated
* [https://www.jet2holidays.com/destinations/croatia#](https://www.jet2holidays.com/destinations/croatia)
  + Countries have description
  + No mention at all of currency
* Removed language and currency tables as this information was not findable on the website
* Left country\_description nullable as only destination countries have this attribute (eg, search for NI)
* Country code not visible anywhere – removed
* country\_group table added to show this relationship for searching purposes
*  Text

  Description automatically generated
* Graphical user interface, text, application

  Description automatically generated
* Above shows that a country can be a child only once, and a parent / child relationship is ALSO unique
* WEBSITE MIRRORS Our country / region/ resort/ hotel structure in its urls
* 
* STAR RATING AND REVIEW RATING ARE NOT THE SAME THING!

Improvements

* Return flights have no timezone specified. To find the time localisation bants we’d need to complete payment, so that is effectively out of scope. British package holiday provider – British times when timezone not specific
* Passport storing local time smells bad – but seems appropriate
* Seat\_count should perhaps be moved into a plane table, that holds information of the type of plane, and its seat count
* GDPR?
* ONLINE SECURITY BILL IN HOUSE OF COMMONS?
* 2 field uniqueness constraints eg a booking contact can make many reviews, but only 1 per hotel?
* Room type occupancies also contain an option (max children attribute) I did not implement this however
* Allow extended check/in out from hotels as a cost item – and its corresponding database implementation
* Short list login system not implemented in our database
* Storing last 4 digits of long card number in payment which would be called via stored procedure
* Addition of paypal table
  + Paypal\_payment\_id
  + Booking\_id
  + Payment\_amount
* Technically there is some overlap between town\_city and region. Specifically however regions are used to group up resorts whereas town\_cities are used in specific addresses so they have slightly different use cases, but there is some reprition there. A better design would have accounted for this
* Some hotels contain information other than what we have in our database, and designing a method of storing this information against certain hotels would be an improvement
* Text

  Description automatically generated
* Star rating perhaps more appropriate to store as a int/ bit(boolean) combo so the front end does not have to extract the number and the plus out of the string
  + Timeline

    Description automatically generated
  + A picture containing table

    Description automatically generated
* Holiday types/ groupings could be implemented
* Not delighted by the destination name nullable/ or use country name setup
  + There are abstractions around country that are difficult to tease apart

Assumptions

* If there is a flight available we are assuming there is a room available? Something like that?
* As there is no login functionality (other than shortlists) when you enter youre details as a booking contact, a new record is inserted, and this new record allows you a single review on that hotel

**Gareth Assumptions/Discussion Points when creating database**

* Some VARCHAR fields being left at 255 even if likely number of characters would be less (e.g. Passport number incase some countries have longer numbers)
  + Similarly title\_name\_abbreviation although I might change this to 10
* However, some have been restricted e.g. airport\_iata\_code as it will only ever be 3 chars

**Country**

* Included a country description which can be null as some countries don’t have descriptions
  + E.g. NI, UK as “fly-from” destinations- countries are in table but don’t have description the way e.g. Spain etc. would have

**GPS**

* *gps\_address* “wraps” address for entities that require both an address and a gps location

**Passenger vs Booking Passenger**

* Separated into two tables as booking passenger needs to have more information stored
  + Telephone numbers were only requested against booking contact on website
* GDPR stipulates that you only hold as much information as is required

**Telephone Type**

* Normalised out as can have office, home, mobile, emergency numbers- many to many I think??

**Airport**

* Airport links to gps\_id via foreign key

**Route**

* Route links twice to airport\_id for departure\_airport\_id & arrival\_airport\_id

**Flight**

* Flight time is a derived attribute. It is derived from departure and arrival datetimes
* **Date times** are stored as utc datetimes, and localised by developer on front end. All dates on website do not stipulate timezone but as jet2 specifically a British package holiday provider it seems reasonable to assume all website times are displayed in British time. Passport is left wityh no timezone specified as theser dates are likely to be local times.
* **Dates** are stored with no timezone
* Datetimes used instead of timestamps as their maximum value is much higher, therefore more future proof

**Flight Price**

* Flight price added with date range for dynamic/flexible pricing
* Datetime selected so pricing can vary not only by date but by flight time
* Validation for start/end handled via front end developer
* FK relationship on flight\_id from flight (1-many as one flight can have many prices)

**Review**

* Review table is linked to booking\_contact\_id via a 1:0 relationship, as a booking contact can be in 0 or 1 reviews, and booking contact
* Design decision made as a person booking a second holiday in the future would get a new ‘booking\_contact\_id’ which would just re-use some of the foreign key data
  + Still ensures that at most 1 review can be left from each booking
* Review date is set to current timestamp which will truncate to the date portion only as this is all the website displays

**Review rating**

* Normalised as review ratings can only be 1 to 5
  + Whereas hotel star ratings could be 3 plus, 4 plus etc

**Image**

* Image table will serve multiple purposes as image id will be foreign key for various “child” image tables in 1-many relationships e.g.
  + Hotel images
  + Room Images
  + Review images
* Image tables are linked to their respective use cases (be it a room type image or a review image) via a 1:0 relationship. That is images can be used 0 or 1 times in a room type

**Hotel**

* Website has additional info for some (but not all hotels)
  + Therefore design decision was made to have a nullable additional info attribute as it’s not mandatory
* However, the website always has details for lift count and floor count so these were included as non-nullable fields
* Check in and out times are specific to that hotel and timezone so therefore decision was made to store in local time
  + As opposed to utc timezones for flights etc. which require to be standardised for clarity & consistency
* Star rating normalised to be id field as foreign key
* Features – derived attributes mostly, but also has lift and floor count

**Board Type**

* Only finite number of options e.g. self-catering, half, all inclusive etc so these will be one to many for hotel board type as many hotels can have the same board types
* Boards – shared across hotels, BUT, can have a little unique blurb
  + Text

    Description automatically generated with medium confidence
  + Left this description as nullable, so that it may be omitted if required (eg *half board* in above image)

**Hotel Board Type**

* Investigating website shows that website has cost based on “standard” option e.g. self-catering but then you can select additional options e.g. All inclusive plus from a list with details of additional surcharge per person
  + SQL query to work out additional cost, how many passengers then dynamically update pricing associated with a potential booking
  + This additional cost would be contained in the booking\_line\_item table
* Reason 13,4 was selected for decimal attributes as it’s to do with GAAP compliance
* Board type description is in here rather than ‘board type’ table as while hotels tend to offer similar board options, their descriptions for each tend to vary
  + Narrative again is “being as representative of the real-world scenario as possible

**Booking**

* Booking contains details of both flights used as foreign keys from the same ‘flight\_id’ attribute in ‘flight’ table
  + 1 to many for each
* Booking reference is generated uniquely by external code when making a booking
* A booking contact can use the same record if they make another booking?
* The booking does not care which passenger goes in which room, which is why room id is stored at room\_booking level and not booking\_passengers level. Perhaps after finalising a booking the website would assign passengers to rooms but this is speculation
* Need to add more design decisions etc. here I reckon
* Included a total\_booking\_cost which will be derived from foreign key relationships
  + Design choice as future proofing- having this data already available will reduce transaction once database has many records

**Room Type**

* Room type is specific to hotel as different hotels have alternative descriptions for “same” classification of room
* Website displays min and max occupants for each room so design decision made to include these in this table
* Quantity of each room type is included in this table
  + Availability model will use this info, coupled with number of that room\_type booked over a given time period
    - SQL queries to demonstrate this
    - Could be used to dynamically update pricing depending non availability
* ROOM type containing min and max passenger ages, because the room type defines the customers able to stay there
* Room types are unique to hotels, look to be manually entered **(see below images)**
* A screenshot of a bedroom

  Description automatically generated with medium confidence
* Rooms are linked to hotels by the room type, which was a design choice to ensure each hotel has its own room types data (hotel\_id in room vs in room\_type)

**Room Type Price**

* Valid from and valid to date included to set dynamic pricing
* Date selected rather than datetime as pricing will only be applicable to a day (not a time)

**Hotel Facility Type**

* Hotel\_facility\_type\_name no longer foreign key as we binned the “parent” facility table for room and hotel as little crossover
* Hotel\_facility\_type\_description removed as identified that different hotels could have different descriptions
  + “Real-world scenario” justification

**Resort**

* Investigating website shows country then broken down into “resorts”, grouping together hotels
  + We previously had this in as region before developing a better understanding of the jet2holidays.com website

**Hotel resort**

* Hotel\_resort\_hotel\_id is foreign key from hotel
  + 1-1 relationship between hotel\_id as hotel can only belong to one resort
* Hotel\_resort\_resort\_id one to many foreign key from hotel\_resort

**Hotel Facility Image**

* Hotel\_facility\_type\_id 1-many from hotel\_facility\_type table
* Image\_id 1-many from image table

**Room Facility**

* Come back to this one

**Card Vendor**

* Normalised out as per 1NF (I think)

**Payment Card**

* Speak about justification for salted hash storage of sensitive stuff.
  + Hash card number, password at a minimum
* CCV number not stored for PCI SSS (??) regulations and etc etc
* Payment card holder details are accessible via the booking contact > address relationship

**Room Image**

* Room\_type\_id 1-many relationship via FK from Room-type
* Image\_id is 1-0/1 from image table

**Payment\_card**

* Encryption will be handled via SQL transaction- secure password as two way
* Expiry date validation will be applied by front end developer
* Foreign key 1-many relationship from booking table on ‘booking\_contact\_id’
* Foreign key 1-many relationship from card\_vendor on ‘card\_vendor\_id’
* CVV code not stored due to regulations (more blurb around this somewhere in notes)
* Card details associated with a booking not a person- more regs

**Payment**

* Payment amount always in GBP as British holiday website
* Possibility to hold last 4 digits

Graphical user interface, application

Description automatically generatedf