

**GROUP ASSIGNMENT**

**TECHNOLOGY PARK MALAYSIA**

**CT004-3-3-ADVBS**

**ADVANCE DATABASE SYSTEM**

**Sushant Rauniyar (NP000140)**

**Evan Thapa Magar (NP000130)**

**Sandhya Rai (NP000138)**

**HAND OUT DATE: 04 November 2020**

**HAND IN DATE: 13 January 2021**

**WEIGHTAGE:** **50%**

**INSTRUCTIONS TO CANDIDATES:**

**1 Submit your assignment at the administrative counter.**

**2 Students are advised to underpin their answers with the use of references (cited using the Harvard Name System of Referencing).**

**3 Late submission will be awarded zero (0) unless Extenuating Circumstances (EC) are upheld.**

**4 Cases of plagiarism will be penalized.**

**5 The assignment should be bound in an appropriate style (comb bound or stapled).**

**6 Where the assignment should be submitted in both hardcopy and softcopy, the softcopy of the written assignment and source code (where appropriate) should be on a CD in an envelope / CD cover and attached to the hardcopy.**

**7 You must obtain 50% overall to pass this module.**

**Table of Contents**

[1. Case Study 1](#_Toc61436174)

[2. Introduction 2](#_Toc61436175)

[3. Assumption 2](#_Toc61436176)

[4. Entity Relationship Modelling 3](#_Toc61436177)

[4.1. Entity Relationship Diagram 3](#_Toc61436178)

[4.2. Data Table Structure 4](#_Toc61436179)

[4.3. Normalization 6](#_Toc61436180)

[4.3.1. UNF 7](#_Toc61436181)

[4.3.2. 1NF 7](#_Toc61436182)

[4.3.3. 2NF 8](#_Toc61436183)

[4.3.4. 3NF 9](#_Toc61436184)

[5. Optimization Strategy 10](#_Toc61436185)

[5.1. De-Normalization 10](#_Toc61436186)

[5.2. Duplicating none-key attributes in one to many relationship 11](#_Toc61436187)

[5.3. SQL Optimization Method 13](#_Toc61436188)

[5.4. Suitable Indexing in Table Columns 13](#_Toc61436189)

[6. Constraints and Triggers 14](#_Toc61436190)

[6.1. Constraints 14](#_Toc61436191)

[6.1.1. Primary Key 14](#_Toc61436192)

[6.1.2. Foreign Key 15](#_Toc61436193)

[6.1.3. NOT NULL 16](#_Toc61436194)

[6.2. Triggers 17](#_Toc61436195)

[6.2.1. Reservation Cancellation 17](#_Toc61436196)

[6.2.2. Flight Delay 18](#_Toc61436197)

[6.2.3. Reschedule Reservation 20](#_Toc61436198)

[6.3. Stored Procedure 21](#_Toc61436199)

[6.3.1. Insert Passenger and Meal Material to a New Impermanent Table 21](#_Toc61436200)

[6.3.2. Customer Info Enquiry via Passport Number 21](#_Toc61436201)

[6.3.3. Insert New Reservation 22](#_Toc61436202)

[7. SQL Queries 24](#_Toc61436203)

[7.1. Member 1: Sushant Rauniyar 24](#_Toc61436204)

[7.1.1. Query (i) Create a query which shows direct flights only for given dates, source & destination. 24](#_Toc61436205)

[7.1.2. Queries (ii) Create a query which shows aircraft code, class code, and expected revenue for each class code, along with the total revenue of each aircraft for a given airline in a single journey. 24](#_Toc61436206)

[7.1.3. Queries (iii) Create a query which shows all passenger numbers with their corresponding descriptions of reservation status for a specific airline. 25](#_Toc61436207)

[7.1.4. Queries (iv) Create a query which shows the name of airline that has been most frequently travelled through by the passengers for specified source and destination in given range of dates. 26](#_Toc61436208)

[7.1.5. Queries (v) Create a query which provides, for each age category of passengers, the following information: 27](#_Toc61436209)

[7.1.6. Queries (vi) Create a query which shows the airline name offering maximum number of journey routes along with names of source and destination. 28](#_Toc61436210)

[7.1.7. Queries (vii) Develop one additional query of your own which provides information that would be useful for the business. Marks will be awarded depending on the technical skills shown and the relevance of the query. 29](#_Toc61436211)

[7.2. Member 2: Sandhya Rai 30](#_Toc61436212)

[7.2.1. Query (viii) Create a query which displays flight details, such as, the aircraft code, regular fare, and discounted fare for the first class. A 25% discount is being offered. Label the column as Aircraft, Regular First Class fare, and Discounted First Class fare. 30](#_Toc61436213)

[7.2.2. Query (ix) Create a query which displays the sorted details of flights to given city code with the least duration flight displayed first. 31](#_Toc61436214)

[7.2.3. Query (x) Create a query which displays the types of non-vegetarian meals offered on flights 32](#_Toc61436215)

[7.2.4. Query (xi) Creates a query which shows the names of countries to which TSI provides a flight reservations. Ensure that duplicate country names are eliminated from the list. 33](#_Toc61436216)

[7.2.5. Query (xii) Create a query which provides, for each airline, the following information: 34](#_Toc61436217)

[7.2.6. Query (xiii) Create a query which shows the names of the meal options available on the given airline. 35](#_Toc61436218)

[7.2.7. Query (xiv) Develop one additional query of your own which provides information that would be useful for the business. Marks will be awarded depending on the technical skills shown and the relevance of the query. 36](#_Toc61436219)

[7.3. Member 3: Evan Thapa Magar 37](#_Toc61436220)

[7.3.1. Query (xv) Create a query which shows the minimum, maximum, and average journey hours for flights to given city code. Display column headings as, Minimum duration, Maximum duration, and Average duration respectively. 37](#_Toc61436221)

[7.3.2. Query (xvi) Create a query which shows the journey date, number of booked seats, and class name for given passenger. 38](#_Toc61436222)

[7.3.3. Query (xvii) Create a query which shows the names of meals not requested by any passenger. 39](#_Toc61436223)

[7.3.4. Query (xviii) Create a query which shows the details of passengers booked through a specified airline in a given date for multi-city flights. 40](#_Toc61436224)

[7.3.5. Query (xix) Create a query which provides, for each airline, the following information: 41](#_Toc61436225)

[7.3.6. Query (xx) Create a query which shows the details of passengers who have availed any extra services for a given flight on specified date. 42](#_Toc61436226)

[7.3.7. Query (xxi) Develop one additional query of your own which provides information that would be useful for the business. Marks will be awarded depending on the technical skills shown and the relevance of the query. 43](#_Toc61436227)

[8. Personal Reflection Report 44](#_Toc61436228)

[8.1. Sushant Rauniyar (NP000140) 44](#_Toc61436229)

[8.2. Sandhya Rai (NP000138) 45](#_Toc61436230)

[8.3. Evan Thapa Magar (NP000130) 46](#_Toc61436231)

[9. References 47](#_Toc61436232)

**List of Figures**

[Figure 1 Entity Relation Diagram 3](#_Toc61436233)

[Figure 2 Proper indexing in table columns 13](#_Toc61436234)

[Figure 3 Primary Key 14](#_Toc61436235)

[Figure 4 Foregin Key 15](#_Toc61436236)

[Figure 5 NOT NULL statements 16](#_Toc61436237)

[Figure 6 Reservation Cancellation Trigger 17](#_Toc61436238)

[Figure 7 Result of Reservation Cancellation (1) 17](#_Toc61436239)

[Figure 8 Result of Reservation Cancellation (2) 18](#_Toc61436240)

[Figure 9 Result of Flight Delay (1) 19](#_Toc61436241)

[Figure 10 Result of Flight Delay (2) 19](#_Toc61436242)

[Figure 11 Trigger of Reschedule Reservation (1) 20](#_Toc61436243)

[Figure 12 Passenger\_Meal Procedure 21](#_Toc61436244)

[Figure 13 Results of customer info enquiry via passport number 22](#_Toc61436245)

[Figure 14 Stored Procedure new\_reservation (3) 23](#_Toc61436246)

[Figure 15 Solution and result of query (i) 24](#_Toc61436247)

[Figure 16 Solution and result of query (ii) 24](#_Toc61436248)

[Figure 17 Solution and result of query (iii) 25](#_Toc61436249)

[Figure 18 Solution and result of query (iv) 26](#_Toc61436250)

[Figure 19 Solution and result of query (v) 27](#_Toc61436251)

[Figure 20 Solution and results of query (vi) 28](#_Toc61436252)

[Figure 21 Solution and results of query (vii) 29](#_Toc61436253)

[Figure 22 Solution of query (viii) 30](#_Toc61436254)

[Figure 23 Results of query (viii) 30](#_Toc61436255)

[Figure 24 Solution of query (ix) 31](#_Toc61436256)

[Figure 25 Results of query (ix) 31](#_Toc61436257)

[Figure 26 Solution of query (x) 32](#_Toc61436258)

[Figure 27 Results of query (x) 32](#_Toc61436259)

[Figure 28 Solution of query (xi) 33](#_Toc61436260)

[Figure 29 Results of query (xi) 33](#_Toc61436261)

[Figure 30 Solution of query (xii) 34](#_Toc61436262)

[Figure 31 Results of query (xii) 34](#_Toc61436263)

[Figure 32 Solution of query (xiii) 35](#_Toc61436264)

[Figure 33 Results of query (xiii) 35](#_Toc61436265)

[Figure 34 Solution of query (xiv) 36](#_Toc61436266)

[Figure 35 Results of query (xiv) 36](#_Toc61436267)

[Figure 36 Solution of query (xv) 37](#_Toc61436268)

[Figure 37 Results of query (xv) 37](#_Toc61436269)

[Figure 38 Solution of query (xvi) 38](#_Toc61436270)

[Figure 39 Results of query (xvi) 38](#_Toc61436271)

[Figure 40 Solution of query (xvii) 39](#_Toc61436272)

[Figure 41 Results of query (xvii) 39](#_Toc61436273)

[Figure 42 Solution of query (xviii) 40](#_Toc61436274)

[Figure 43 Results of query (xviii) 40](#_Toc61436275)

[Figure 44 Solution of query (xix) 41](#_Toc61436276)

[Figure 45 Results of query (xix) 41](#_Toc61436277)

[Figure 46 Solution of query (xx) 42](#_Toc61436278)

[Figure 47 Results of query (xx) 42](#_Toc61436279)

[Figure 48 Solution of query (xxi) 43](#_Toc61436280)

[Figure 49 Results of query (xxi) 43](#_Toc61436281)

**List of Tables**

[Table 1 Structure of Data Tables 4](#_Toc61436282)

[Table 2 UNF 7](#_Toc61436283)

[Table 3 1NF 7](#_Toc61436284)

[Table 4 2NF 8](#_Toc61436285)

[Table 5 3NF 9](#_Toc61436286)

[Table 6 Many to Many Relationship Diagram 10](#_Toc61436287)

[Table 7 De-normalization Diagram 11](#_Toc61436288)

[Table 8 : One to Many Relationship 12](#_Toc61436289)

[Table 9 Avoid Duplication of key 12](#_Toc61436290)

# Case Study

TRAVEL SAFE INTERNATIONAL (TSI) is a prominent corporation in supplying several airlines with global delivery networks for selling tickets. Its offices are located in New York. In one of four classes, passengers can be booked- First, Corporate, Luxury Economic, and Economic. For flight reservations, single way, roundtrip, and multiple trips choices are open. As various airlines, several kinds of meals choices are obtainable. The flight period determines whether to serve refreshments, single meals, or several meals. Special programs, such as preschool, are offered by providing special meals to children aged between 2 and 11.

The child's price be contingent on whether the child is "in the lap" or "in the seat." The airlines will responsibility amongst 10-15% adult price while commuting with a child in lap, while the air company will be responsibility the child fare while traveling with the infant in a wheelchair. There is a limit of 4 travelers per boarding on certain airlines. Many airlines limit the right of a child up to 17 years of age to fly alone, but some airlines provide youngsters from 5 to 17 years of age with unaccompanied minor service to ensure that the child is boarded on the plane. For all unaccompanied minors, the program is mandatory and open to extra expenses. Unaccompanied little service while traveling with a parental or a grownup age 18 or older is not required for children aged 17 and underage. A single charge will be charged in each direction for the two children traveling together.

Wheelchair service is offered by several airlines as well. Joint restrictions and regulations related to aircraft reservation, including but not restricted to the luggage limit, are observed. According to the ticket's validity, cancellations and seat modifications can be made with individual extra costs. Regarding ticket updates, once the journey resumes, the receipt requisite be rereleased, and all voyages obligation be finished within one year of the initial time of departure. If the trip does not begun, the receipt must be rereleased, and the journey essential commence contained by one year of the start of the new journey.

If a passenger abandons or variations their itinerary less than four periods before the planned departure, the ticket's full value will be forfeited.

# Introduction

TRAVEL SAFE INTERNATIONAL (TSI) is a prominent corporations in supplying several airlines with global delivery networks for selling tickets. Its offices are located in New York. TSI provides the reservation system's core features, such as flight booking, passenger registration, and flights rescheduling.

A database must be generated for storing various details relevant to flight booking, passenger registration, and others, with specific criteria given. Given the need to consider each airline's laws and regulations, queries would be tailored to satisfy market needs and deliver supplementary purposes to sustain TSI for longstanding processes.

The Entity-Relationship idea is produced by citrating all characteristics and relations with other objects in a simple view to display all the entities. In this database construction, limitations, stored measures, and generates the market needs and manage the record logically and structurally. Other than that, to increase the efficiency of the database, optimization techniques would also be applied.

# Assumption

For building a database, specific business criteria are given but not explicitly defined. To style the databank more structural and supportable, quantities of theories were brought up. The hypotheses posed are listed below.

* Some carriers limit only four customers per reservation.
* Most airlines provide numerous facilities, like solitary child services, wheelchair, and for youngster.
* Every plane serves a diverse meal plan based on the period of the aircraft
* Every plane has a dissimilar amount of seats.

All aircraft are be able to be distinguished different types, one way that reaches explicitly the endpoint, one roundtrip that enables travelers to leave and reappearance with one reservation, and one latter city alternative that allows flights to transit and change flights to the specific destination.

# Entity Relationship Modelling

## Entity Relationship Diagram

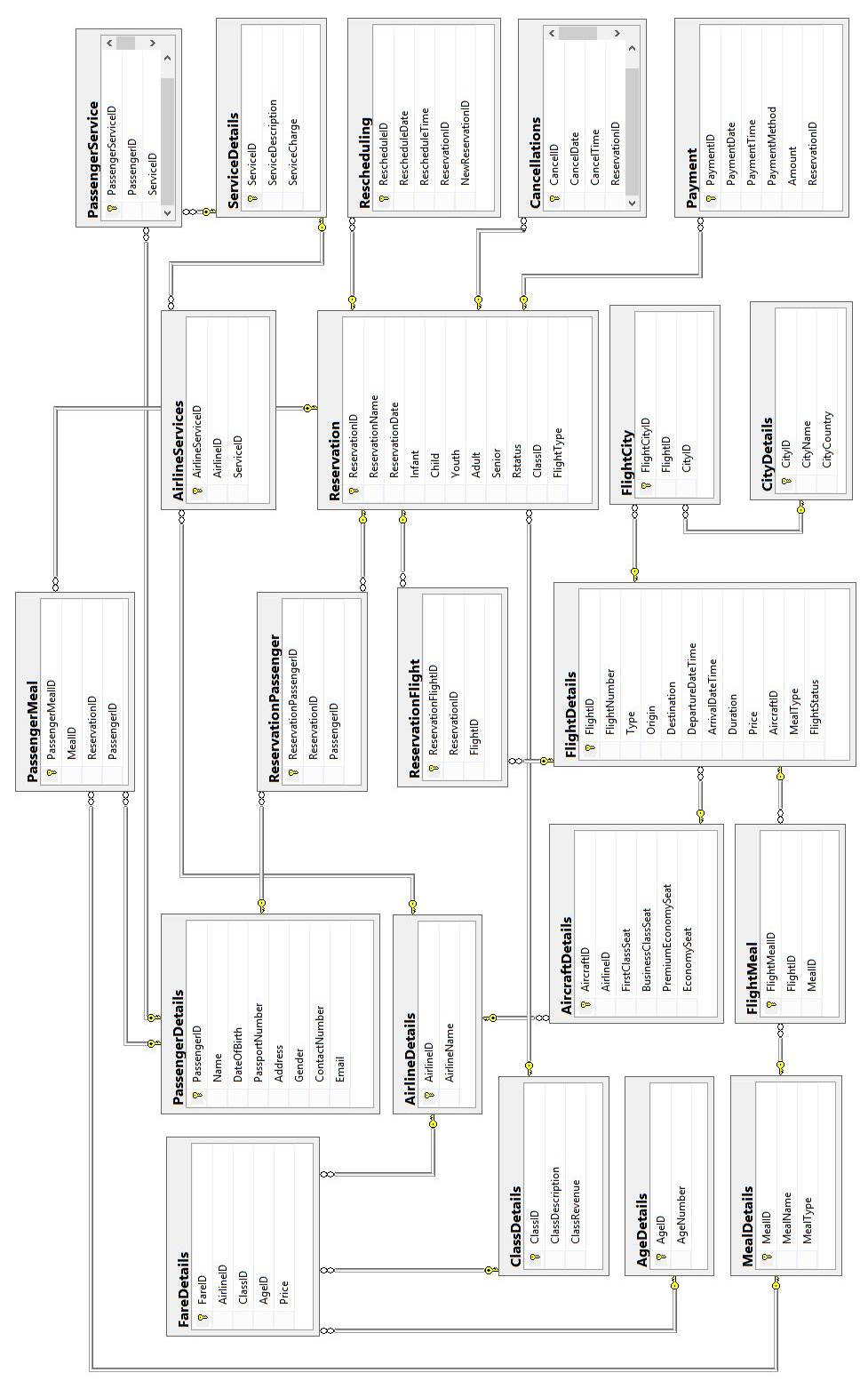


Figure 1 Entity Relation Diagram

## Data Table Structure

Table 1 Structure of Data Tables

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Table Name** | **Attribute Name** | **Data Type** | **Allow NULL** | **Primary Key** | **Foreign Key** |
| **AgeDetails** | AgeID | varchar(50) | No | Yes | No |
| AgeNumber | int | No | No | No |
| **AircraftDetails** | AircraftID | nvarchar(20) | No | Yes | No |
| AirlineID | nvarchar(20) | No | No | Yes |
| FirstClassSeat | int | No | No | No |
| BusinessClassSeat | int | No | No | No |
| PremiumEconomySeat | int | No | No | No |
| EconomySeat | int | No | No | No |
| **AirlineDetails** | AirlineID | nvarchar(20) | No | Yes | No |
| AirlineName | varchar(50) | No | No | No |
| **AirlineServices** | AirlineServiceID | nvarchar(50) | No | Yes | No |
| AirlineID | nvarchar(20) | No | No | Yes |
| ServiceID | nvarchar(50) | No | No | Yes |
| **Cancellations** | CancelID | nvarchar(50) | No | Yes | No |
| CancelDate | date | No | No | No |
| ReservationID | nvarchar(50) | No | No | Yes |
| **CityDetails** | CityID | varchar(50) | No | Yes | No |
| CityName | varchar(50) | No | No | No |
| CityCountry | varchar(50) | No | No | No |
| **ClassDetails** | ClassID | varchar(50) | No | Yes | No |
| ClassDescription | varchar(50) | No | No | No |
| ClassRevenue | int | No | No | No |
| **FareDetails** | FareID | nchar(10) | No | Yes | NO |
| AirlineID | nvarchar(20) | No | No | Yes |
| ClassID | varchar(50) | No | No | Yes |
| AgeID | varchar(50) | No | No | Yes |
| Price | decimal(18,0) | No | No | No |
| **FlightDetails** | FlightID | nvarchar(50) | No | Yes | No |
| FlightNunmber | nvarchar(50) | No | No | No |
| Type | varchar(50) | No | No | No |
| Origin | nvarchar(50) | No | No | No |
| Destination | nvarchar(50) | No | No | No |
| DepartureDateTime | datetime | No | No | No |
| ArrivalDateTime | datetime | No | No | No |
| Duration | datetime | No | No | No |
| Price | decimal(18,0) | No | No | No |
| AircraftID | nvarchar(20) | No | No | Yes |
| MealType | varchar(50) | No | No | No |
| FlightStatus | varchar(50) | No | No | No |
| **FlightCity** | FlightCityID | nvarchar(50) | No | YES | NO |
| FlightID | nvarchar(50) | No | No | Yes |
| CityID | varchar(50) | No | No | Yes |
| **FlightMeal** | FlightMealID | nvarchar(50) | No | Yes | No |
| FlightID | nvarchar(50) | No | No | Yes |
| MealID | nvarchar(50) | No | No | Yes |
| **MealDetails** | MealID | nvarchar(50) | No | Yes | No |
| MealName | varchar(50) | No | No | No |
| MealType | varchar(50) | No | No | No |
| **PassengerDetails** | PassengerID | nvarchar(50) | No | Yes | No |
| Name | varchar(50) | No | No | No |
| DateOfBirth | date | No | No | No |
| PassportNumber | nvarchar(50) | No | No | No |
| Address | nvarchar(max) | No | No | No |
| Gender | varchar(50) | No | No | No |
| ContactNumber | numeric(18,0) | No | No | No |
| Email | nvarchar(50) | No | No | No |

## Normalization

Normalization is a strategy of the database design that eradicates device redundancy and removes redundant structures such as Insertion, Update, and Deletion Incongruities. The laws of normalization break more oversized tables into smaller tables and link them using relationships.

SQL Normalization aims to erase redundant figures and confirm the logical storage of data. With the First Normal Form implementation, Edgar Codd's inventor proposed data normalization and started to extend the theory with the Second and Third Normal Forms. (Guru99, 2018)

Normalization is predominantly used for two reasons.

* Eliminating unusable information.
* Certifying data addictions makes logic, i.e., reasonably processed knowledge.

**First Normal Form (1NF)**

1NF follows these guidelines:

* It should only have minute valued attributes/columns.
* Ethics deposited in a pilaster should be of the identical domain
* All the columns in a table should have unique names.

**Second Normal Form (2NF)**

Second Normal Form should follows these rules:

* States within First Normal form.
* Should not have Partial Dependency.

**Third Normal Form (3NF)**

Third Normal Form should follows these rules:

* States within 2NF.
* Do not have Transitive Reliance. (Studytonight, 2019)

### UNF

Table 2 UNF

|  |
| --- |
| AirlineID, AirlineName, AircraftID, FirstClassSeat, BusinessClassSeat, PremiumEconomySeat, EconomySeat, FareID, ClassID, AgeID, AgeNumber, FlightID, FlightNumber, Type, Origin, Destination, DepartureDateTime, ArrivalDateTime, Duration, MealType, Price, FlightMealID, MealID, MealName, PassengerMealID, ReservationID, PassengerID, ClassDescription, ClassRevenue, ReservationName, ReservationDate, Infant, Child, Youth, Adult, Senior, RStatus, FlightType, ReservationPassengerID, Name, DateOfBirth, PassportNumber, Address, Gender, ContactNumber, Email, ServiceID, ServiceDescription, ServiceCharge, ReservationFlightID, CityID, CityName, CityCountry, PaymentID, PaymentDate, PaymentTime, PaymentMethod, Amount, AirlineServiceID, RescheduleID, RescheduleDate, RescheduleTime, NewReservationID, CancelID, CancelDate, CancelTime, FlightStatus, FlightCityID |

### 1NF

Table 3 1NF

|  |  |
| --- | --- |
| AirlineID, FareID, AgeID, AircraftID, FlightMealID, FlightID, ClassID, ReservationID, ReservationPassengerID, MealID, PassengerMealID, PassengerID, ReserveFlight, PaymentID, CityID, ServiceID, AirlineServiceID, RescheduleID, CancelID, FlightCityID | AirlineName, FirstClassSeat, BusinessClassSeat, PremiumEconomySeat, EconomySeat, AgeNumber, FlightNumber, Type, Origin, Destination, DepartureDateTime,  ArrivalDateTime, Duration, MealType, Price, MealName, ClassDescription, ClassRevenue, ReservationName, ReservationDate, Infant, Child, Youth, Adult, Senior, RStatus, Type, Name, DateOfBirth, PassportNumber, Address, Gender, ContactNumber, Email, ServiceDescription, ServiceCharge, CityName, CityCountry, PaymentDate, PaymentTime, PaymentMethod, Amount, RescheduleDate, RescheduleTime, NewReservationID, CancelDate, CancelTime, FlightStatus |

### 2NF

Table 4 2NF

|  |  |
| --- | --- |
| AirlineID | AirlineName |
| AgeID | AgeNumber |
| AirlineID, ServiceID | AirlineServiceID |
| AircraftID,AirlineID | FirstClassSeat, BusinessClassSeat, PremiumEconomySeat, EconomySeat |
| AirlineID, ClassID, AgeID | FareID, Price |
| FlightID, MealID | FlightMealID |
| FlightID, CityID | FlightCityID |
| FlightID, AircraftID | FlightNumber, Type, Origin, Destination, DepartureDateTime, ArrivalDateTime, Duration, MealType, Price, FlightStatus |
| MealID | MealName, MealType |
| ClassID | ClassDescription, ClassRevenue |
| RID, ClassCode | RName, RDate, Infant, Child, Youth, Adult, Senior, RStatus. FlightType |
| MealCode, RID, PID | PassengerMealCode |
| RID, PID | ReservationPassengerCode |
| PID, ServiceCode | PName, DOB, PassportNo, Address, Gender, ContactNo, Email |
| ServiceCode | ServiceDesc, ServiceCharge |
| RID, FlightCode | ReserveFlight |
| RID,PaymentID | PaymentDate, PaymentTime, PaymentMethod, Amount |
| RescheduleID, RID | RescheduleDate, RescheduleTime, NewRID |
| CancelID, RID | CancelDate, CancelTime |

### 3NF

Table 5 3NF

|  |  |
| --- | --- |
| FareID | Price |
| FlightNumber | Type, Origin, Destination, DepartureDateTime, ArrivalDateTime, Duration, MealType, Price |

# Optimization Strategy

## De-Normalization

De-normalization is a way of endeavoring to improve the deliver output of a record by inserting dismissed data or combination data as to unite different data from the database and produce a new data category view. It is used to combine several database data and generate a new group view of data. Below will show example of de-normalization such as combining two Many-to-many relationship tables. For example:

* Many Passenger can choose many PassengerMeal
* PassengerMeal can have more than one Meal

Below shows the many-to-many relationship diagram, in order to get access for data from Passenger and Meal. An inner join statement is considered.

Table 6 Many to Many Relationship Diagram

|  |  |  |
| --- | --- | --- |
| **PassengerDetails** | **PassengerMeal** | **MealDetails** |
| PassengerID (PK)  Name  DateOfBirth  PassportNumber  Address  Gender  ContactNumber  Email | PassengerMealID(PK)  MealID(FK)  ReservationID(FK)  PassengerID(FK) | MealID(PK)  MealName  MealType |

Table 7 De-normalization Diagram

|  |
| --- |
| **PassengerDetail\*** |
| PassengerID(PK)  Name  DateOfBirth  PassportNumber  Address  Gender  ContactNumber  Email  MealName  MealType |

From above table, when the de-normalization is applied where both detail of Passenger and Meal will be merged together as shown in table 2. The benefits of using the de-normalization are improved performance, the need for fewer joins, and the ability to maintain history information. It allows staff of aircraft easier to arrange food before flight and distribute the food on plane. It can also be used as a reference when there are some mistakes on food distribution to avoid any misunderstanding. In other view on this system, de-normalization enables the means to maintain function of history look-up on previous transaction and it will helps the system to be easier to manage.

## Duplicating none-key attributes in one to many relationship

Two tables that share similar characteristics that make a partnership are a one-to-many relationship as relationship should be a parent table which includes a child entity element. The Passenger table should include the Service ID as an international key in the Passenger Information table, based on TSI. In order to access data from the parent table, the Service table's data must use a joint statement.

The JOIN statements used in the database are reduced using table 4 above, which will greatly recover the performance of the record and the simplicity of the question statement as it has more disk size, and in a short amount of time, any employee can understand the simplicity of the process.

Table 8 : One to Many Relationship

|  |  |
| --- | --- |
| **PassengerDetails** | **ServiceDetails** |
| PassengerID (PK)  ServiceID  Name  DateOfBirth  PassportNumber  Address  Gender  ContactNumber  Email | ServiceID (PK)  ServiceDescription  ServiceCharge |

Table 9 Avoid Duplication of key

|  |
| --- |
| **PassengerDetails** |
| PassengerID (PK)  Name  DateOfBirth  PassportNumber  Address  Gender  ContactNumber  Email  ServiceDescription  ServiceCharge |

Using above table 4, it reduces the JOIN statements used in database which will enhance the database performance greatly and through the simplicity of the query statement. It saves more database space and the simplicity of the system can be learnt by any staff in a short amount of time.

## SQL Optimization Method

SQL articulation is essentially recycled to extract figures the record. A few approaches that can get comparable results for writing queries. The implementation is not the fastest because of the builder's ordinary messed up, which is mistaken for the best approach to handling the matter. The list below will show how to increase the performance which that can be optimized.

Avoid full table scan - SQL query does not include primary key or index which the full scan occur. Example:

|  |
| --- |
| SELECT PassengerID, DateOfBirth, Name from Passenger where Name = ‘Lim Hong Jun’ |

## Suitable Indexing in Table Columns

On executing the queries, correct indexing on the table section would significantly advance the database's execution. Appointing essential keys to every single table in the database is the best course for ordering. Subsequently, where organizing functions that are inferable from the tables are physically organized, data may be retrieved quickly based on the leading critical sector. Only similar tables can be assigned to the index, where information can be exchanged between tables.

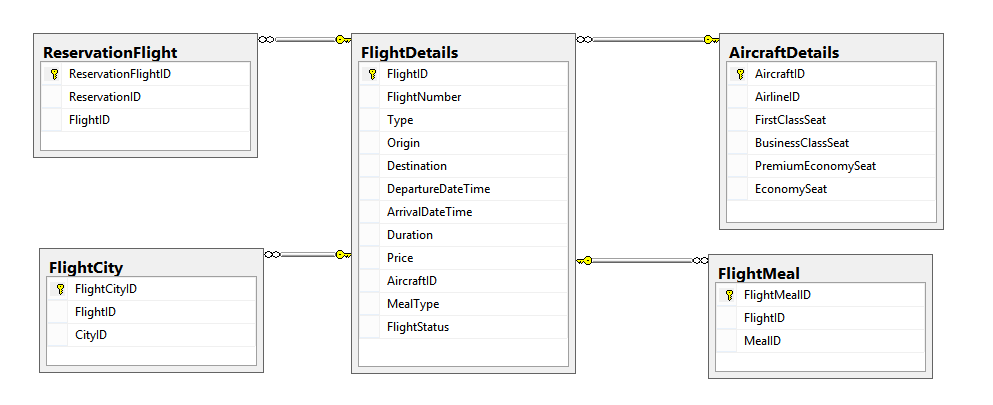


Figure 2 Proper indexing in table columns

Though the development of indexes may increase the record's general performance, the after effect can occur if there are numerous files. Developers would have to enter all applicable tables internally to execute a modest query. Therefore, since the capability of the record is massive, it would be a real threat.

# Constraints and Triggers

## Constraints

Constraints permit operators to track database laws and can influence occupational logic, record veracity and table layout.

### Primary Key

Primary key constraint is Not Null and is considered a Unique Constraint in the sense that it must contain data inside and does not repeat the same value. It helps the user to easily and quickly identify a specific data in a table.

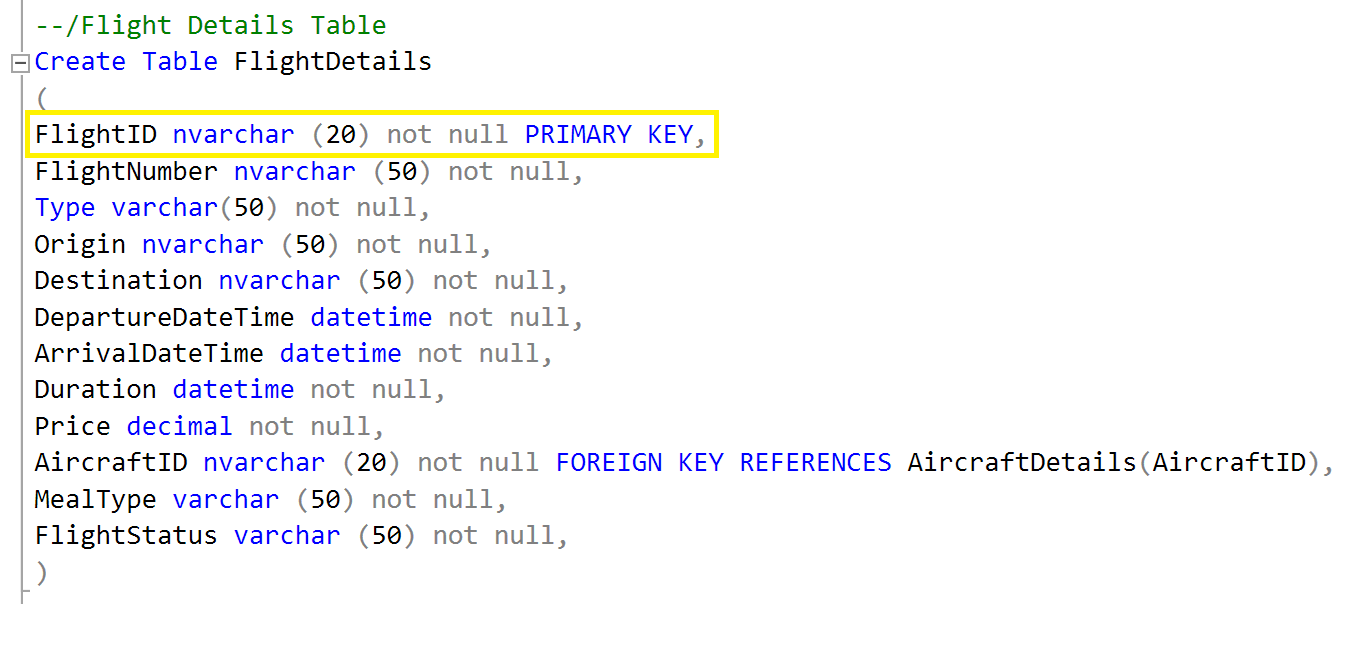


Figure 3 Primary Key

Above figure shows, the primary key is set in FlightDetails table. This FlightID will have foreign key references in other tables which will use it to access the FlightDetails table to obtain data. An example would be that a different table would be able to access a specific row of data in FlightDetails via the unique primary key, ensuring that only a single type of result will be listed since there would be no duplicate primary key fields.

### Foreign Key

The primary international restriction guarantees that the data in one table is referential honesty to match values in another table (W3Schools, 2014). In other tables that carry their info, the foreign key itself is a primary.

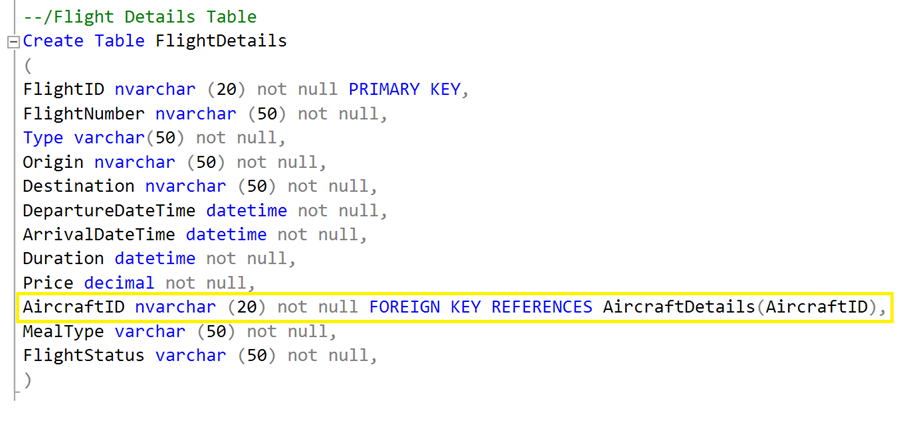


Figure 4 Foregin Key

The figure above shows a foreign key constraint has been added to the AircraftID in the FlightDetails table, which will link the field to its reference which is the AircraftID primary key inside the AircraftDetails table. This enables functions such as referential integrity where the DBMS is able to check whether a foreign key column in a table contains the same data as the ones found in the primary key column in the original table. For this example shown here, AircraftID is used to determine which aircraft will be used for the created flight.

### NOT NULL

NOT NULL constraints are used to enforce a rule that a column must always contain valid and applicable data- in short, that a column may not contain NULL. (Codeidol.com, 2012) It will automatically alert developers if the database found that there’s an empty column haven’t filled. Adopting NOT NULL constraints will not just help a developer to maintain a database with increased ease, it prevents events such as the developer forgetting to input necessary data as it will generate an alert to developer. It also prevents errors such as linking of data that will lead to no result show as NULL foreign key. Thus, most of the time NOT NULL constraints are needed to avoid unnecessary errors.

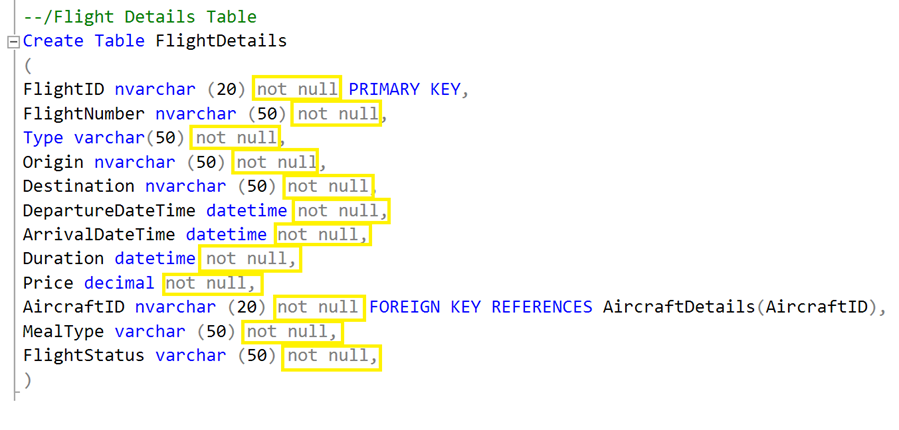


Figure 5 NOT NULL statements

The query above shows the usage of NOT NULL in FlightDetails table. This is to ensure that all the column data are filled so that other tables and queries can refer to the required data.

## Triggers

### Reservation Cancellation

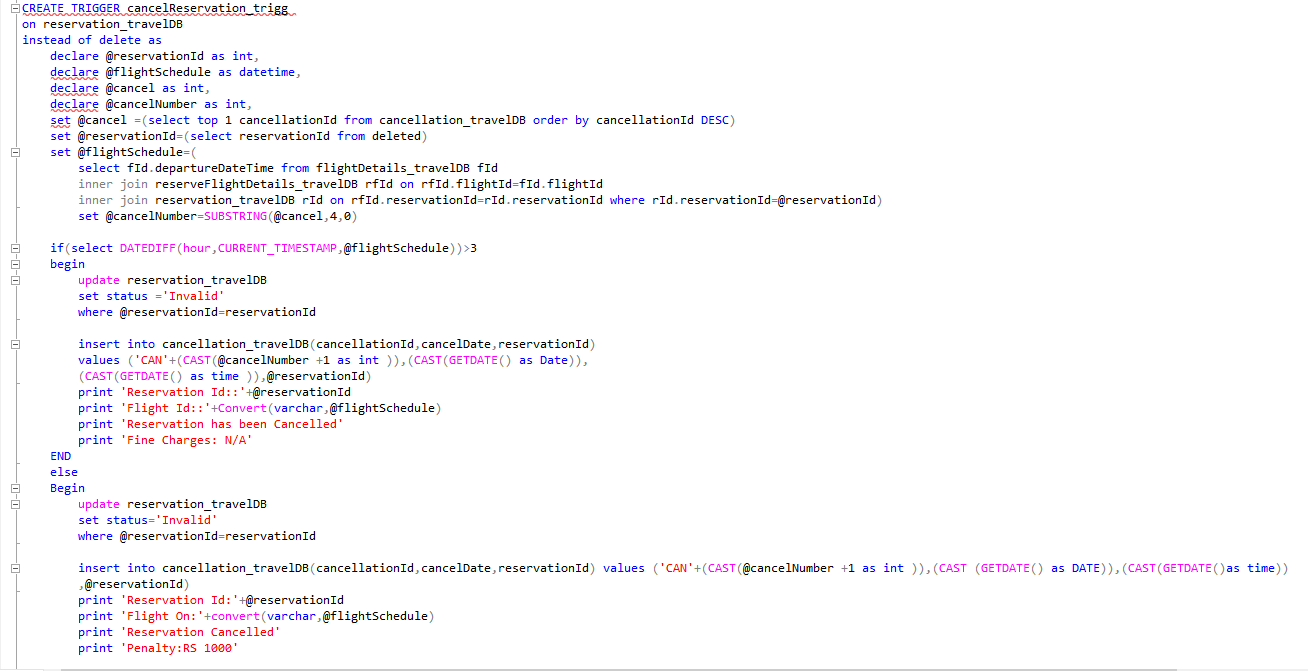


Figure 6 Reservation Cancellation Trigger

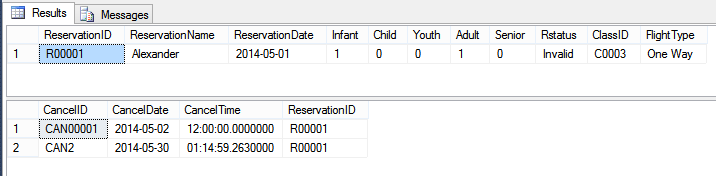
**Output:** 

Figure 7 Result of Reservation Cancellation (1)

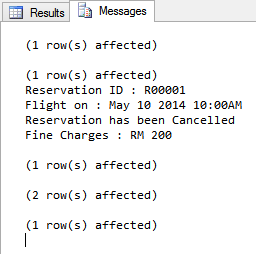
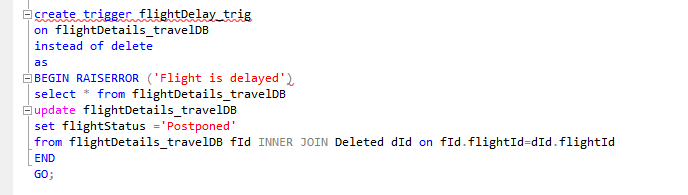


Figure 8 Result of Reservation Cancellation (2)

Simultaneously, the removed record will be created in the Termination table to show that the customer no extended reserves this particular ticket. Extra than all persons who canceled their booking three hours before the aircraft would allow the token to mislay its worth according to business plan, and interest costs will be paid.

The cause will be reviewed on the flight schedule to ensure that the data fits the chosen reservation ID. The machine would then exhibition posts that indicate that the information is completely changed.

### Flight Delay



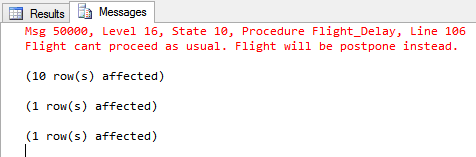


Figure 9 Result of Flight Delay (1)

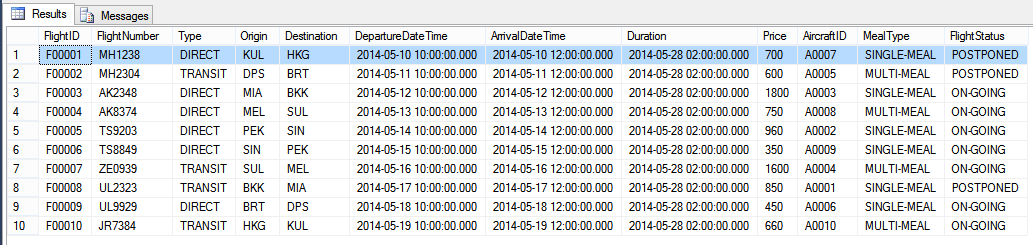


Figure 10 Result of Flight Delay (2)

To stop any flight records from being inadvertently or deliberately deleted, this Trigger was introduced. This uninstall order would instead change the status from on-going to defer if any unexpected conditions occur, such as Heavy Snow or Lightning Dip, where the aeronautical must be behind or adjourned. It helps workers, travelers to recognize condition of such airlifts easily.

### Reschedule Reservation



Figure 11 Trigger of Reschedule Reservation (1)

The trigger that is implanted in reschedule\_reservation was designed to meet the business requirements which is to reschedule the Reservation ID if any passengers want to made changes to his/her reservation. The trigger will be triggered as soon as the data is inserted to the “Rescheduling” table. The trigger will simply check whether the reservation status linked to the “ReservationID” given by the passenger has been reissued. If the Reservation status is indicates “Reissued” or “Invalid”, the data would not be inserted into the database, instead, a rollback will be performed and an error message will be displayed.

In opposite, if the Reservation status indicates “Booked”, the data will be inserted into “Rescheduling” table, and it will update the existing reservation status to “Reissued” based on the “ReservationID” provided by passenger. After that, a new ReservationID will be generate and inserted to “ReservationDetails” table as well as update “ReservationPassenger” table to ensure the new “ReservationID” is attach with the “PassengerID” that passengers provide. In the meantime, the existing “ReservationID” that has been reissued will become invalid.

## Stored Procedure

### Insert Passenger and Meal Material to a New Impermanent Table

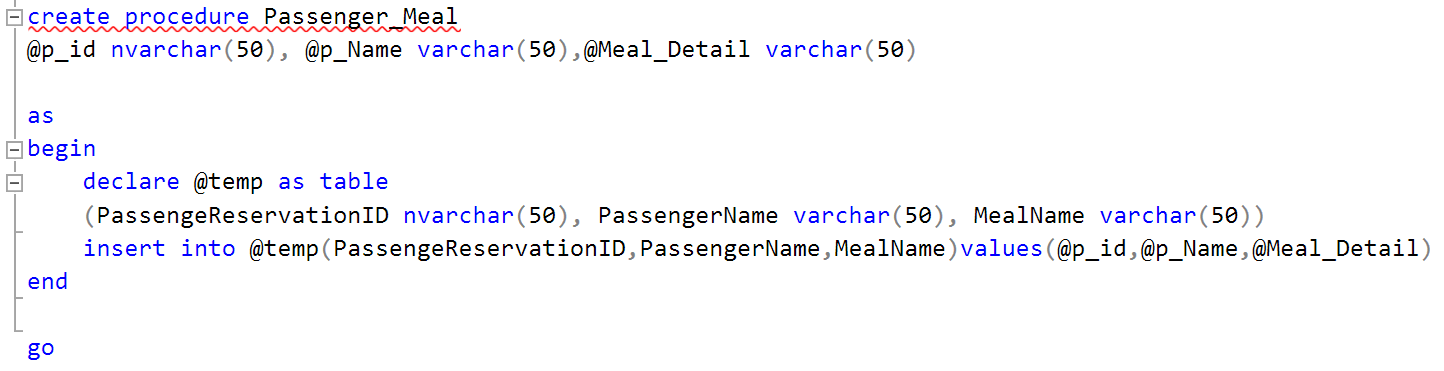
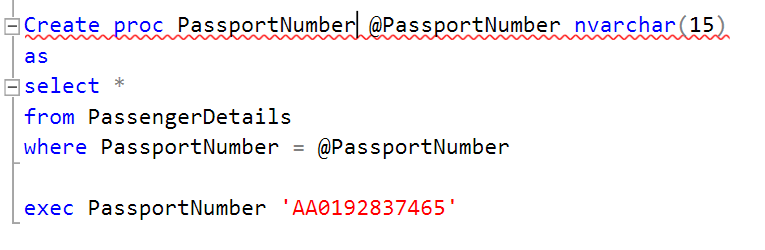


Figure 12 Passenger\_Meal Procedure

The "Passenger-Meal" protocol is here to addition records from the "PassengerDetails" and "MealDetails" table and accumulation data in a brief table to display the appropriate production. This system is rummage-sale to signify which travelers have requested what food on a particular trip.

### Customer Info Enquiry via Passport Number



**Output:**

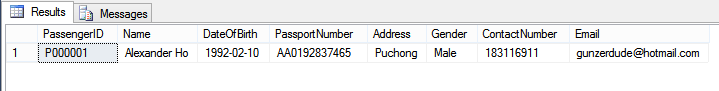


Figure 13 Results of customer info enquiry via passport number

It so-called protocol helps employees to check for customer details via passport numbers quickly. Since an individual can be marked uniquely by passport numbers, it can be a second primary key. Information about passengers is given during the registration process. For this specific user, the machine would check the input passport number and scan the passenger database.

### Insert New Reservation

The stored procedure for new\_reservation will be used to insert a new reservation record into “Reservation” table, “PassengerDetails” table, “ReservationPassenger” table, “ReservationFlight” table, and “Payment” table. Before insertion to related tables, the stored procedure will check the total number of passenger to ensure the number of passenger is not more than 4 passengers. If this limit is exceeded, the new record would not be inserted into tables and an error messaged will be displayed. On the flipside, when number of passengers is less than the limit, the record will be stored into related tables.



Figure 14 Stored Procedure new\_reservation (3)

# SQL Queries

## Member 1: Sushant Rauniyar

### Query (i) Create a query which shows direct flights only for given dates, source & destination.

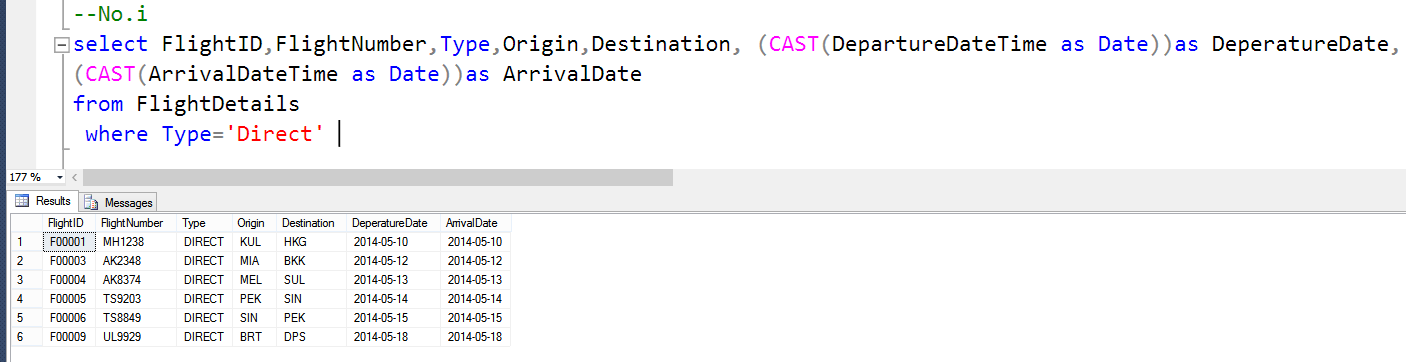


Figure 15 Solution and result of query (i)

This query will show all flight information regarding direct flight.

### Queries (ii) Create a query which shows aircraft code, class code, and expected revenue for each class code, along with the total revenue of each aircraft for a given airline in a single journey.

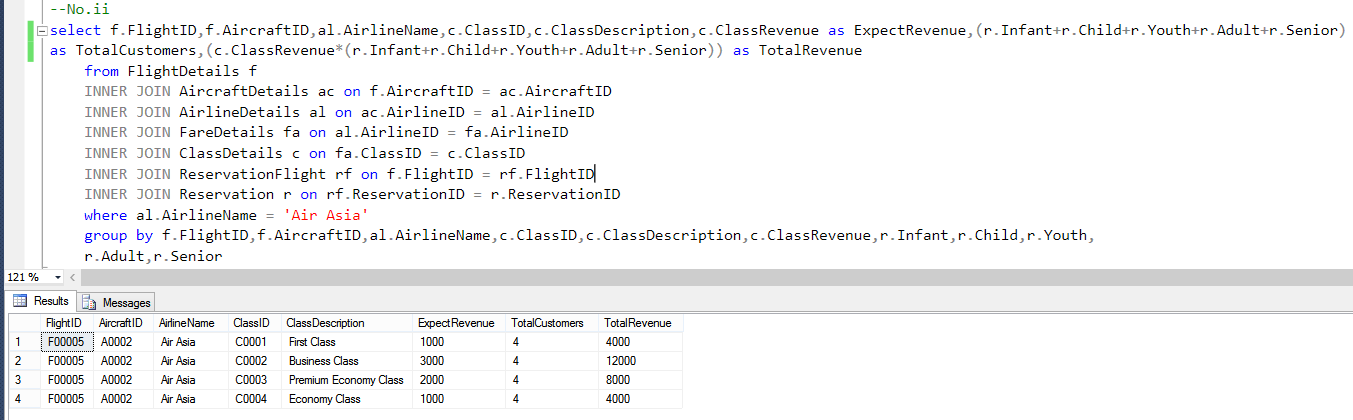


Figure 16 Solution and result of query (ii)

Expect Revenue for First Class is 10000, Business Class is 8000, Premium Economy Class is 5000 and Economy Class is 3000. The results will be varied and the total revenue will be based on how many total customers purchase the ticket.

### Queries (iii) Create a query which shows all passenger numbers with their corresponding descriptions of reservation status for a specific airline.

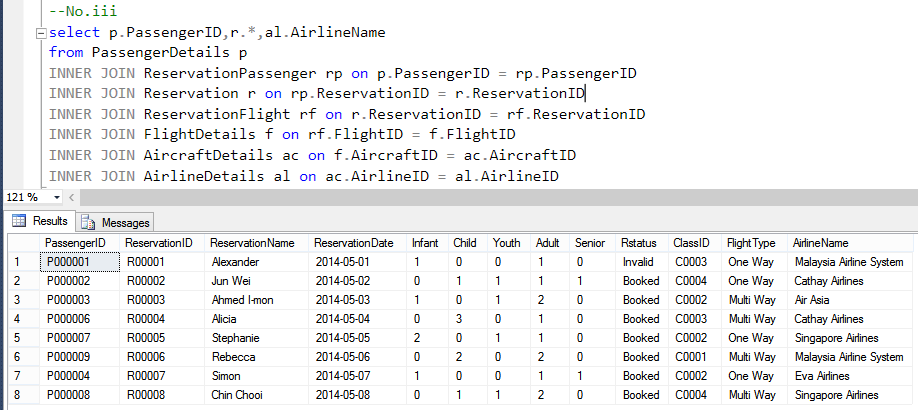


Figure 17 Solution and result of query (iii)

This query will allow user to look for all reservation information regarding what airline company chosen, how many passengers, types of flight and the reservation status.

### Queries (iv) Create a query which shows the name of airline that has been most frequently travelled through by the passengers for specified source and destination in given range of dates.

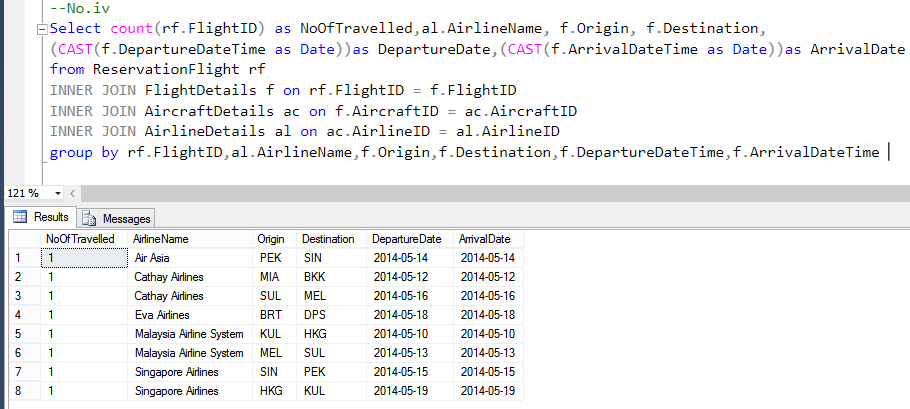


Figure 18 Solution and result of query (iv)

This query will show user regarding the number of travelled for specific airline company for specific journey routes.

### Queries (v) Create a query which provides, for each age category of passengers, the following information:

The total number of infants, children, youths, adults & seniors travelling through specified flight in a single journey operated by a specified airline in given date. Result should contain both detailed breakup & summary for above mentioned categories along with overall summary.

Hint: you may wish to use rollup or cube statements with a query. Some marks will be awarded for the query structure, even if you cannot generate the totals.

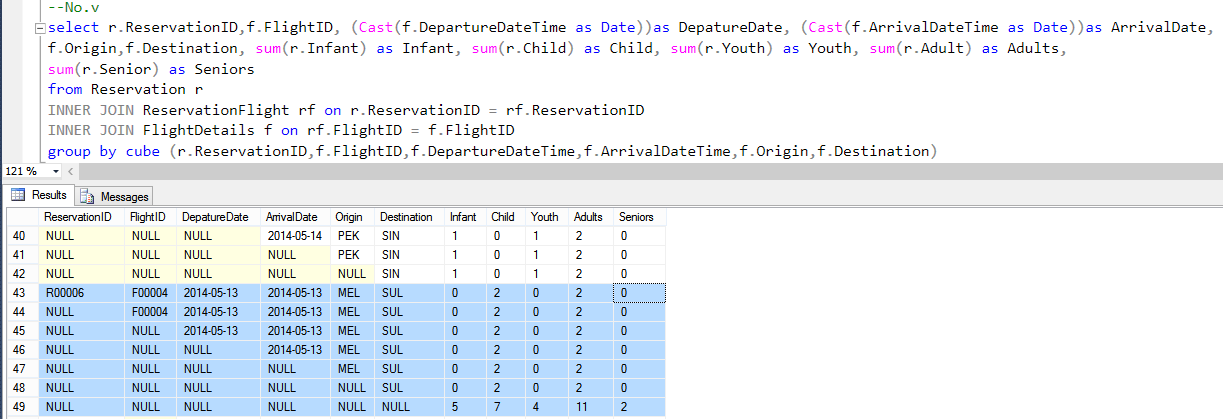


Figure 19 Solution and result of query (v)

As it can be seen that the third highlighted row is the sum of first and second highlighted row.

### Queries (vi) Create a query which shows the airline name offering maximum number of journey routes along with names of source and destination.

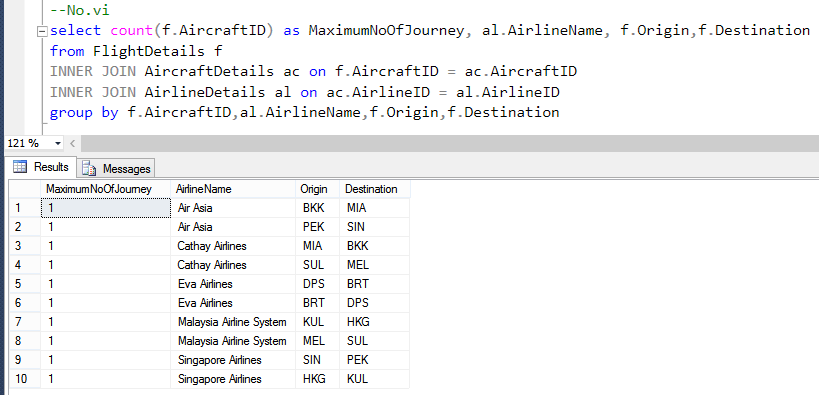


Figure 20 Solution and results of query (vi)

This query will show the user regarding the maximum number of journey which are offered by several airline company with specific journey routes.

### Queries (vii) Develop one additional query of your own which provides information that would be useful for the business. Marks will be awarded depending on the technical skills shown and the relevance of the query.

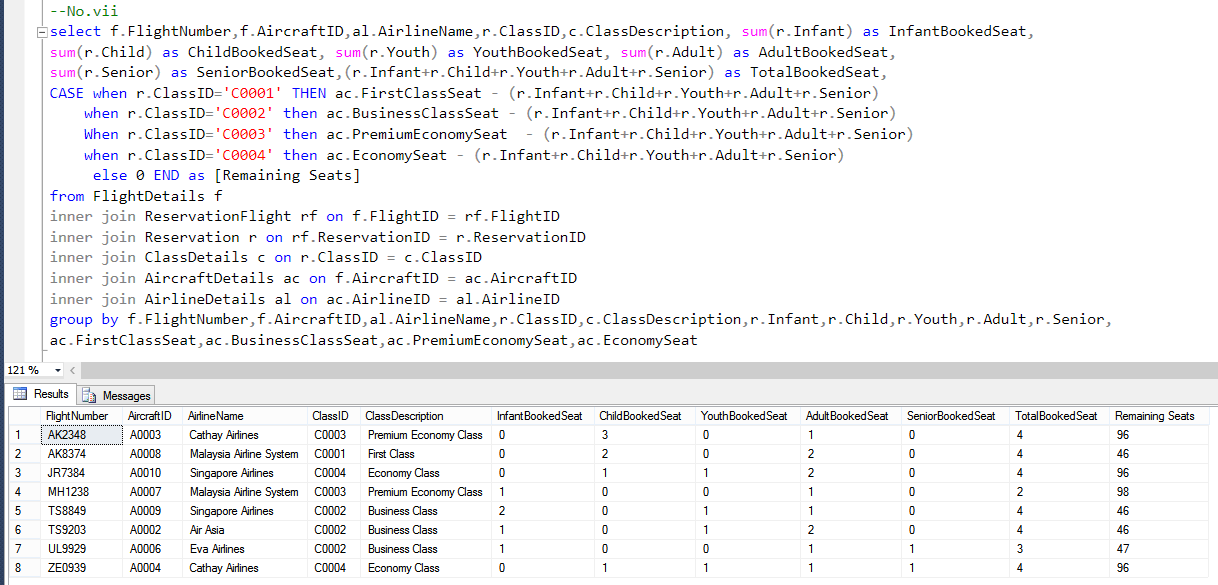


Figure 21 Solution and results of query (vii)

For the own design queries part, I have decided to show the remaining seats for specific flight where information including InfantBookedSeat, ChildBookedSeat and etc will be shown. The actual seats sizes for all aircrafts are set at as follows; first class 50, business class 50, premium economy seat 100 and economy 100. The results for remaining seats column will be based on the subtraction of total booked seat for all age category.

## Member 2: Sandhya Rai

### Query (viii) Create a query which displays flight details, such as, the aircraft code, regular fare, and discounted fare for the first class. A 25% discount is being offered. Label the column as Aircraft, Regular First Class fare, and Discounted First Class fare.

**Solution:**

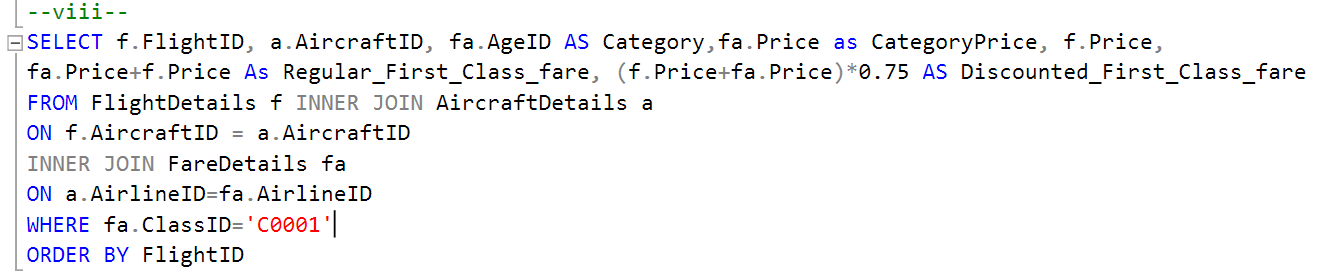


Figure 22 Solution of query (viii)

**Output:**

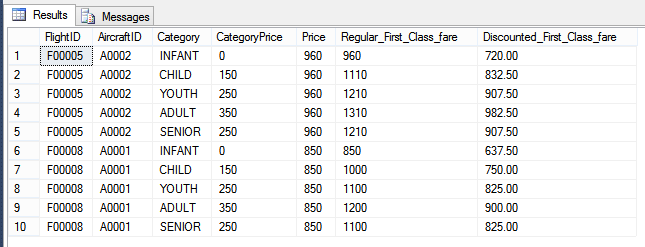


Figure 23 Results of query (viii)

The figure above shows the output of the query (viii) with the class code is “C0001”.

### Query (ix) Create a query which displays the sorted details of flights to given city code with the least duration flight displayed first.

**Solution:**

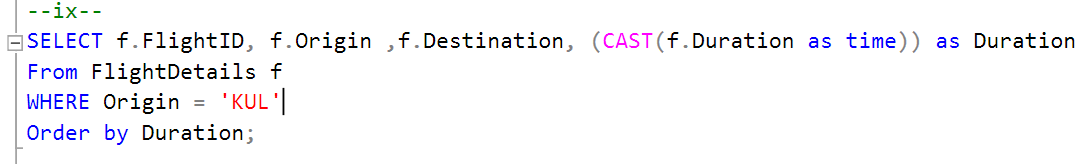


Figure 24 Solution of query (ix)

**Output:**

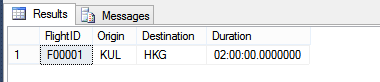


Figure 25 Results of query (ix)

The figure above shows the output of the query (ix) with the given city is “KUL”.

### Query (x) Create a query which displays the types of non-vegetarian meals offered on flights

**Solution:**

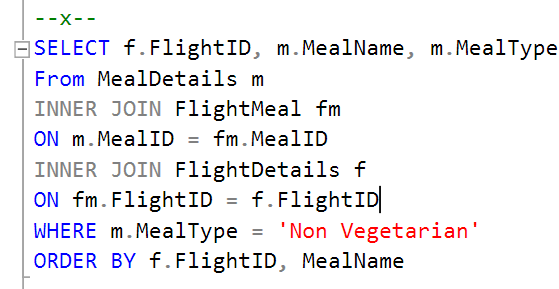


Figure 26 Solution of query (x)

**Output:**

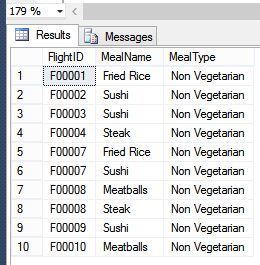


Figure 27 Results of query (x)

The figure above shows the output of query (x) with the meal type is “Non-Vegetarian” as well as sort by FlightCode and MealName.

### Query (xi) Creates a query which shows the names of countries to which TSI provides a flight reservations. Ensure that duplicate country names are eliminated from the list.

**Solution:**

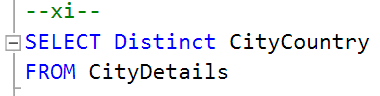


Figure 28 Solution of query (xi)

**Output:**



Figure 29 Results of query (xi)

The figure above shows the output of query (xi) with eliminated duplicate country name.

### Query (xii) Create a query which provides, for each airline, the following information:

The total number of flights scheduled in a given date. Result should contain both detailed breakup & summary for flights for each airline along with overall summary.

Hint: you may wish to use rollup or cube statements with a query. Some marks will be awarded for the query structure, even if you cannot generate the totals.

**Solution:**

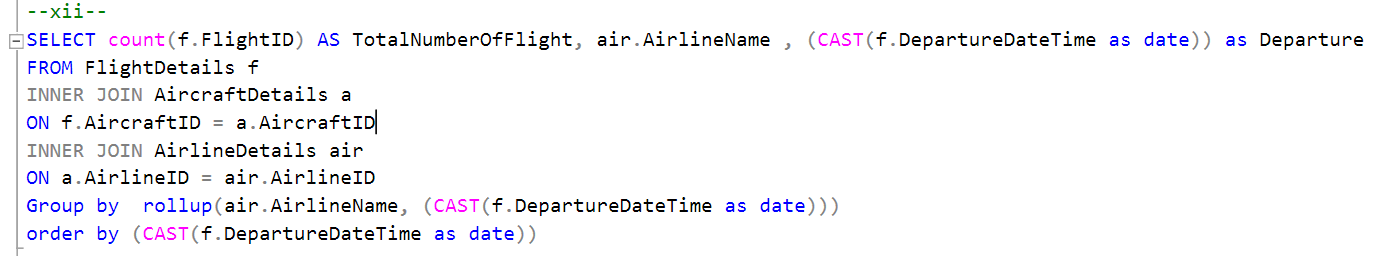


Figure 30 Solution of query (xii)

**Output:**

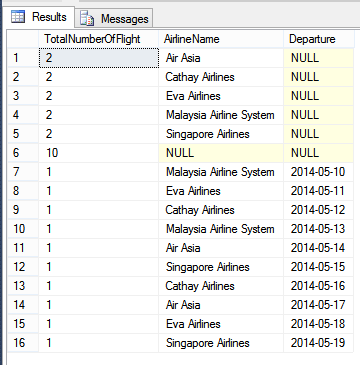


Figure 31 Results of query (xii)

The figure above shows the output of query (xii) with the rollup statements. First it shows the total number of flight based on Airline AirlineName and shows the total of flights. Next, it will categorize the number of flights based on the departure date and airline company name.

### Query (xiii) Create a query which shows the names of the meal options available on the given airline.

**Solution:**

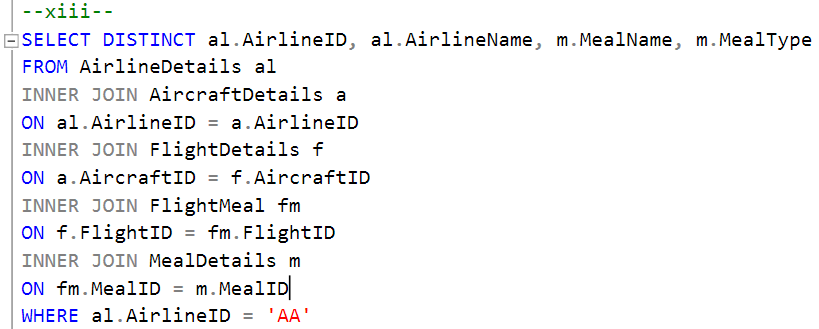


Figure 32 Solution of query (xiii)

**Output:**

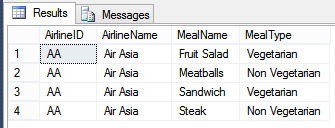


Figure 33 Results of query (xiii)

The figure above shows the output of query (xiii) with the given airline is “AA”.

### Query (xiv) Develop one additional query of your own which provides information that would be useful for the business. Marks will be awarded depending on the technical skills shown and the relevance of the query.

**Solution:**

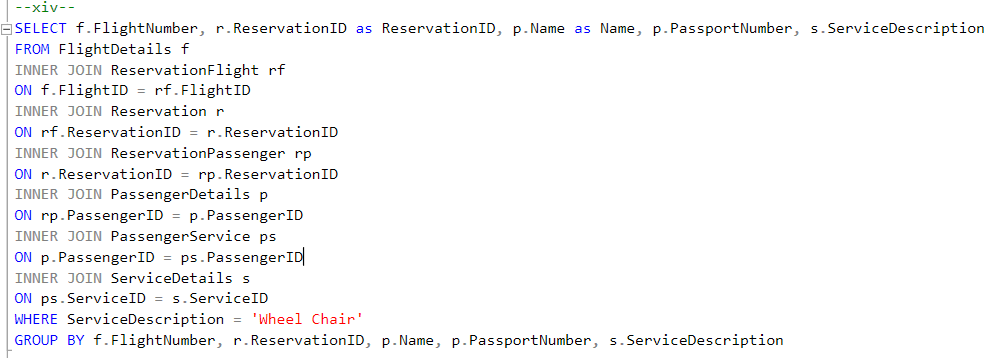


Figure 34 Solution of query (xiv)

**Output:**

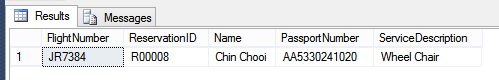


Figure 35 Results of query (xiv)

The figure above shows the output of query (xiv). It displays which passenger requires Service of “Wheel Chair” according to each flight.

## Member 3: Evan Thapa Magar

### Query (xv) Create a query which shows the minimum, maximum, and average journey hours for flights to given city code. Display column headings as, Minimum duration, Maximum duration, and Average duration respectively.

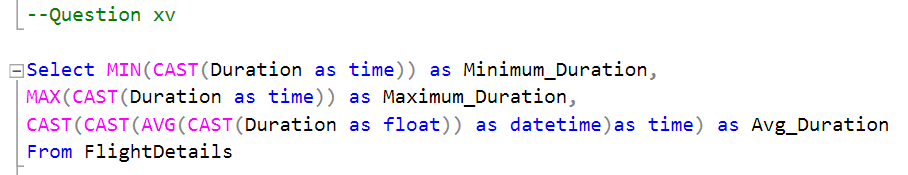


Figure 36 Solution of query (xv)

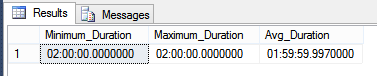


Figure 37 Results of query (xv)

This are the result of minimum duration, maximum duration and average duration.

### Query (xvi) Create a query which shows the journey date, number of booked seats, and class name for given passenger.

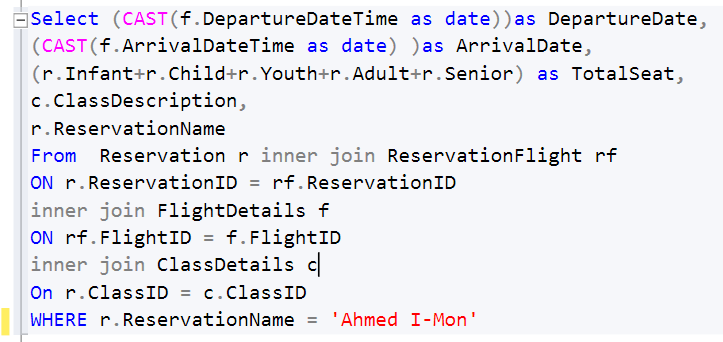


Figure 38 Solution of query (xvi)

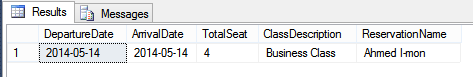


Figure 39 Results of query (xvi)

The passenger has reserved a flight on specific date including the total Seat and the class seat.

### Query (xvii) Create a query which shows the names of meals not requested by any passenger.

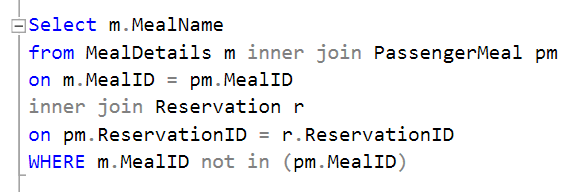


Figure 40 Solution of query (xvii)



Figure 41 Results of query (xvii)

The result should show the MealName of beef, sushi, and salad but due to SQL problems the result cannot be show properly but the way of coding is correct.

### Query (xviii) Create a query which shows the details of passengers booked through a specified airline in a given date for multi-city flights.

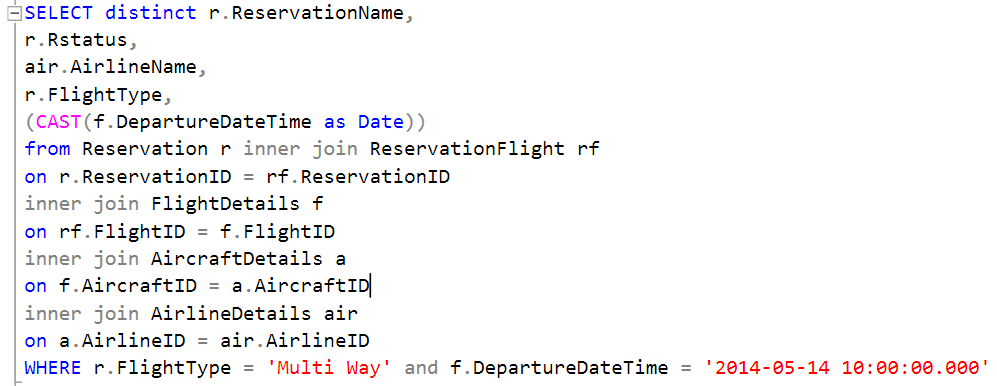


Figure 42 Solution of query (xviii)

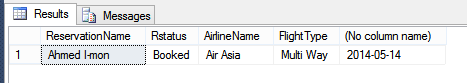


Figure 43 Results of query (xviii)

The result show the by searching specific flight type data ‘multi-city’ and exact date for the passenger booked.

### Query (xix) Create a query which provides, for each airline, the following information:

The total number of unaccompanied children travelling in a given date. Result should contain both detailed breakup & summary for unaccompanied children for each airline along with overall summary.

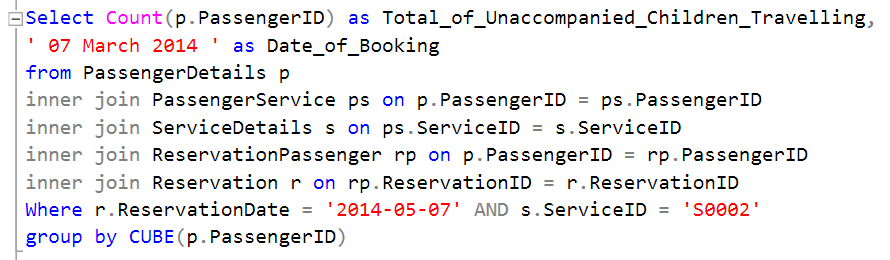


Figure 44 Solution of query (xix)

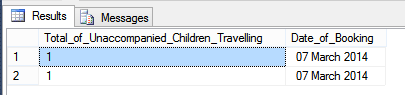


Figure 45 Results of query (xix)

The result are showing the total number of unaccompanied child travelling in specific date given. The result are three people.

### Query (xx) Create a query which shows the details of passengers who have availed any extra services for a given flight on specified date.

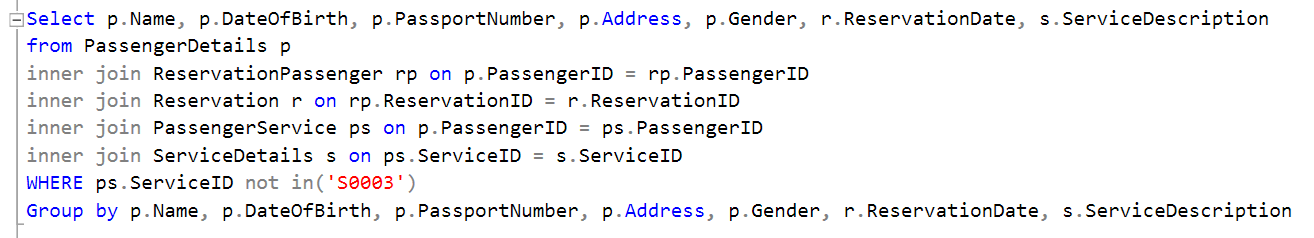


Figure 46 Solution of query (xx)

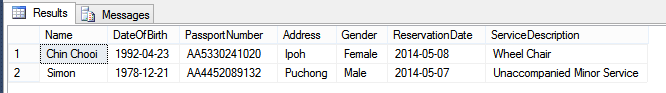


Figure 47 Results of query (xx)

This query is to show the result of people who are having extra service during their flight.

### Query (xxi) Develop one additional query of your own which provides information that would be useful for the business. Marks will be awarded depending on the technical skills shown and the relevance of the query.

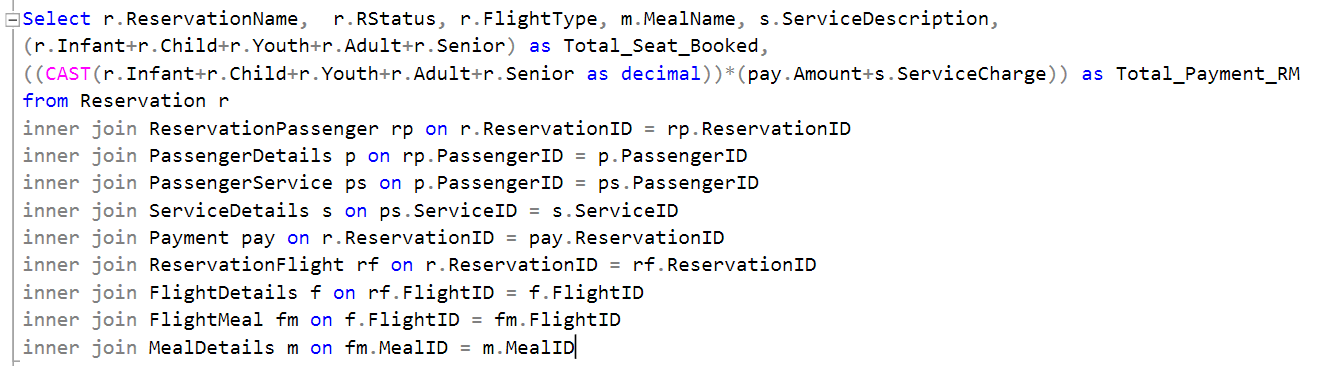


Figure 48 Solution of query (xxi)

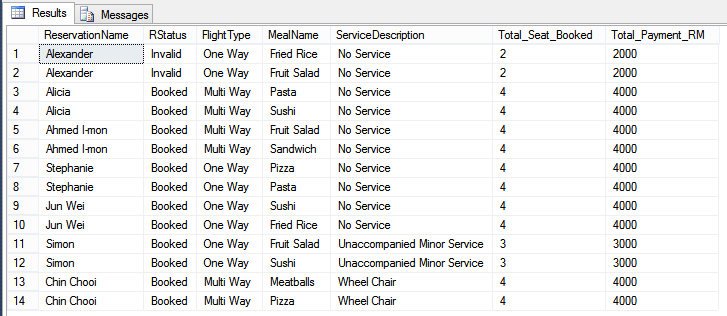


Figure 49 Results of query (xxi)

This result is according to the business strategy, which staff need to check what are the details that should show at their monitor screen. The details are Reservation Name, the status, flight type, meal and service reserved, the number of seat booked and the payment of passenger going to

# Personal Reflection Report

## Sushant Rauniyar (NP000140)

Travel Secure International database structure we expected to form a gathering of three to set up a for this mission. Throughout the project, we worked together to generate outlines, such as the Organization Relationship Diagram (ERD), to meet the coursework's commercial needs. We have already addressed different forms of streamlining methods to ensure honesty and zero waste in the layout of the database tables.

I faced various obstacles in the real improvement of the database when building the database. Still, I could resolve them with dialogs with my gathering people and also Ms. Sandhya Rai, whom I also much appreciate for her comprehension of rehashing her addresses or instructional practice to ensure that I can thoroughly comprehend the advanced database procedures he is seeking to grant us, and whatever is left of my course mates. I could effectively implement prompts and bring operations into the database based on the expertise I have learned from these interactions and make them operate with no problems.

I was entrusted with the inquiries given for part one section for the individual SQL inquiries. Much of the investigations proved very difficult, and some expected me to do supplementary analysis using the web and even discussing information from the APU archive.

I will like to state that it was an exciting and unforgettable experience observing the excellent and low times throughout this database's completion to close my appearance on this assignment. I have taken on many new propelled database capabilities that will prove useful later in programming enhancement projects that use databases, such as my up and coming venture last year.

## Sandhya Rai (NP000138)

During the creation of the database and paper, many lessons were learned. This project also helps me exercise my expertise and skills in specialized databases such as causes, shop process and others, tutorials, and lecture classes.

To create an ERD that all team members would comprehend, few meetings were arranged by the team leader to review mission specifics. The team leader, Mr. Sushant Rauniyar, provided me with a lot of support during the database construction anytime I encounter a conceptual issue with the relationship between tables.

The next component to proceed after database structures is three essential functions: limits, triggers, and store method. This role relates to the mechanism with various consequences. Each participant for this assignment had to include the custom feature above.

The component where more errors occur during development was triggered among these three roles. With the team leader's support and web browsing, I did it to address the issue where the code could not be retrieved and placed in the relevant table.

I want to thank those who supported me during the production of the whole assignment here. The project has been a success, and I would like to express my gratitude on behalf of my teammates to Mr. Jyotir Moy Chatterjee for his kind assistance during the module.

## Evan Thapa Magar (NP000130)

This role has facilitated me acquire a great deal of information and evidence about the catalogue system. Since this is a community job, it has helped me a great deal to work in groups. I learned a lot about the Entity Interaction Diagram (ERD) and expectations in conjunction with my teammates. The case study offered to the group as an assignment was based on these diagrams.

A collection of 7 statements were issued to each member upon which questions had to be introduced. The seven views I picked gave me a lot of information and gained many details about database systems and questions. For their time and commitment, and most of all, their support to accomplish this mission, I thank my community mates. From them and new stuff like rollup and cube comments, I have learned a lot.

Besides what the instructor showed us in the lectures, this group task illuminated me with the meaning of routine procedures and ignited a database like claims. Stored processes have a new table to allow any significant attributes to be held by a user.

The trigger rule allows the user to view, upload, edit, or erase some content. Furthermore, the CAST command that assists in modifying date, number, or figure is an imperative command I discovered.

# References

Guru99, 2018. *Guru99 - What is Normalization? 1NF, 2NF, 3NF, BCNF Database Example.* [Online]   
Available at: https://www.guru99.com/database-normalization.html  
[Accessed 1 January 2021].

Studytonight, 2019. *Studytonight - Normalization of Database.* [Online]   
Available at: https://www.studytonight.com/dbms/database-normalization.php  
[Accessed 1 January 2021].

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Member 1** | **Member 2** | **Member 3** |
| **Student Name** | **Sushant Rauniyar** | **Sandhya Rai** | **Evan Thapa Magar** |
| **Design** |  |  |  |
| **ER Model + Relational Schema** | Check mark, Wingdings font, character code 252 decimal. |  |  |
| **Optimisation Strategy** |  | Check mark, Wingdings font, character code 252 decimal. |  |
| **Triggers and Constraints** |  |  | Check mark, Wingdings font, character code 252 decimal. |
| **Implementation** |  |  |  |
| **Optimisation Strategy** | Check mark, Wingdings font, character code 252 decimal. |  |  |
| **Triggers and Constraints** |  | Check mark, Wingdings font, character code 252 decimal. |  |
| **T-SQL feature** |  |  | Check mark, Wingdings font, character code 252 decimal. |
| **Queries** |  |  |  |
| **Query 1 to 7** | Check mark, Wingdings font, character code 252 decimal. | **N/A** | **N/A** |
| **Query 8 to 14** | **N/A** | Check mark, Wingdings font, character code 252 decimal. | **N/A** |
| **Query 15 to 21** | **N/A** | **N/A** | Check mark, Wingdings font, character code 252 decimal. |
| **Student Signature** |  |  |  |

**Workload Matrix**

**Marking Scheme**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Part A - Group Marks (80%)** | **Max** | **Marks** | | |
|  |  | **Member 1** | **Member 2** | **Member 3** |
| **Design** |  | **Sushant Rauniyar** | **Sandhya Rai** | **Evan Thapa Magar** |
| **ER Model + Relational Schema** | **24** |  |  |  |
| **Optimisation Strategy** | **8** |  |  |  |
| **Triggers and Constraints** | **8** |  |  |  |
| **Implementation** |  |  |  |  |
| **Optimisation Strategy** | **8** |  |  |  |
| **Triggers and Constraints** | **8** |  |  |  |
| **T-SQL feature** | **8** |  |  |  |
| **Queries** |  |  |  |  |
| **Query 1 to 7** | **16** |  | **N/A** | **N/A** |
| **Query 8 to 14** | **16** | **N/A** |  | **N/A** |
| **Query 15 to 21** | **16** | **N/A** | **N/A** |  |
| **Total Group Marks** | **80** |  |  |  |
| **Part B - Individual Marks (20%)** |  | **Member 1** | **Member 2** | **Member 3** |
| **Presentation** | **20** |  |  |  |
| **Overall Marks** | **100** |  |  |  |
| **GRADE** |  |  |  |  |