#### CASE STUDY

#### OF

#### AIRLINE TICKET

#### RESERVATION

#### MANAGEMENT SYSTEM

#### IN

#### RELATIONAL DATABASE

#### DESIGN

#### Chitkara University Office Photos | Glassdoor

CASE STUDY IN RELATIONAL DATABASE DESIGN

TITLE: AIRLINE TICKET RESERVATION MANAGEMENT SYSTEM

STUDENT’S NAME: KARAN AHUJA (2010990362)

ADISH JAIN (2010990028)

ADITI MAINI (2010990029)

AMIT GUPTA (2010990060)

GUIDE: Dr. SUSHEELA HOODA

**ABSTRACT**

One case study “Airline Ticket Reservation Management System” is presented. Input for this case study is taken from its informal specification to a relational schema using entity-relationship modeling and its translation to relational model, to database schema, to implementation of the database, to interactive SQL querying of the installed database (SQL/Oracle).Airline reservation System is a computerized system used to store and retrieve information and conduct transactions related to air travel. The project is aimed at exposing the relevance and importance of Airline Reservation Systems. It is projected towards enhancing the relationship between customers and airline agencies through the use of Airline Reservation System, and thereby making it convenient for the customers to book the flights as when they require such that they can utilize the software to make reservations.The software has two parts. First is user part and the administrator part. User part is used as a front end and administrator is the back end. Administrator is used by airline authority. It will allow the customers to access database and allow new customers to sign up for online access.The main purpose of the software is to reduce the manual errorsinvolved in the airline reservation process and make it convenient for the customers to book the flights as when they require such that they can utilize this software to make reservations, modify reservations or cancel a particular reservation.

**ACKNOWLEDGEMENTS**

I would like to express my gratitude to all of those who made it possible to complete this thesis, in particular to my Teacher Dr.Susheela Hooda. I would also like to thank my family for their understanding and continuous support.

CONTENTS

**Abstract**

**Acknowledgments**

**Chapter 1: Introduction**

**Chapter 2: Name of Case Study**

* Case Study INFORMAL DESCRIPTION
* Terminologies and Symbols of E-R diagram
* Case study LOGICAL MODEL(ER Diagram)
* Case study Physical schema
* Case study INTERACTIVE QUERIES

**Chapter 3 : Conclusion**

**Bibliography**

**Chapter 1: INTRODUCTION**

Database Management System:

Database Management Systems (DBMS) are software systems used to store, retrieve, and run queries on data. A DBMS serves as an interface between an user and a database, allowing users to create, read, update, and delete data in the database. DBMS manages the data, and the database schema, allowing for data to be manipulated or extracted by users and other programs. This helps provide data security, data integrity, concurrency, and uniform data administrative procedures. DBMS optimizes the organization of data by following a database schema design technique called normalization, which splits a large table into smaller tables when any of its attributes have redundancy in values. DBMS offers many benefits over traditional file systems, including flexibility and a more complex backup system. Database management systems can be classified based on a variety of criteria such as the data model, the database distribution, or user numbers. The most widely used types of DBMS software are relational, distributed, hierarchical, and network. For Example, MySQL, Oracle etc. are popular commercial DBMS used in different applications. Relational database management systems (RDBMS) are the most popular data model because of its user-friendly interface. It is based on normalizing data in the rows and columns of the tables. This is a viable option when you need a data storage system that is scalable, flexible, and able to manage lots of information.

Relational Database Management System:

Relational database management systems (RDBMS) are the most popular data model because of its user-friendly interface. A relational database management system (RDBMS) is a collection of programs and properties that enables us and others to create, update, manages and interact with a relational database. RDBMS store data in the form of tables, with most commercial relational database management systems using Structured Query Language (SQL) to access the database. However, since SQL was invented after the initial development of the relational model, it is not necessary for RDBMS use. The RDBMS is the most popular database system among organizations across the world. It provides a dependable method of storing and retrieving large amounts of data while offering a combination of system performance , ease of implementation and better than the basic File system. An RDBMS is a type of database management system (DBMS) that stores data in a row-based table structure which connects related data elements. An RDBMS includes functions that maintain the security, accuracy, integrity and consistency of the data. This is different than the file storage used in a DBMS. In a DBMS, data is kept in a hierarchical form, whereas an RDBMS utilizes a table where the headers are used as column names and the rows contain the corresponding values.

ER Diagram:

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design relational databases in the fields of database management, business information systems, education and research. Also known as ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.

| Entity Symbol | Name | Description |
| --- | --- | --- |
| Strong Entity Symbol | Strong entity | These shapes are independent from other entities, and are often called parent entities, since they will often have weak entities that depend on them. They will also have a primary key, distinguishing each occurrence of the entity. |
| Weak Entity Symbol | Weak entity | Weak entities depend on some other entity type. They don't have primary keys, and have no meaning in the diagram without their parent entity. |
| Associative entity symbol | Associative entity | Associative entities relate the instances of several entity types. They also contain attributes specific to the relationship between those entity instances. |

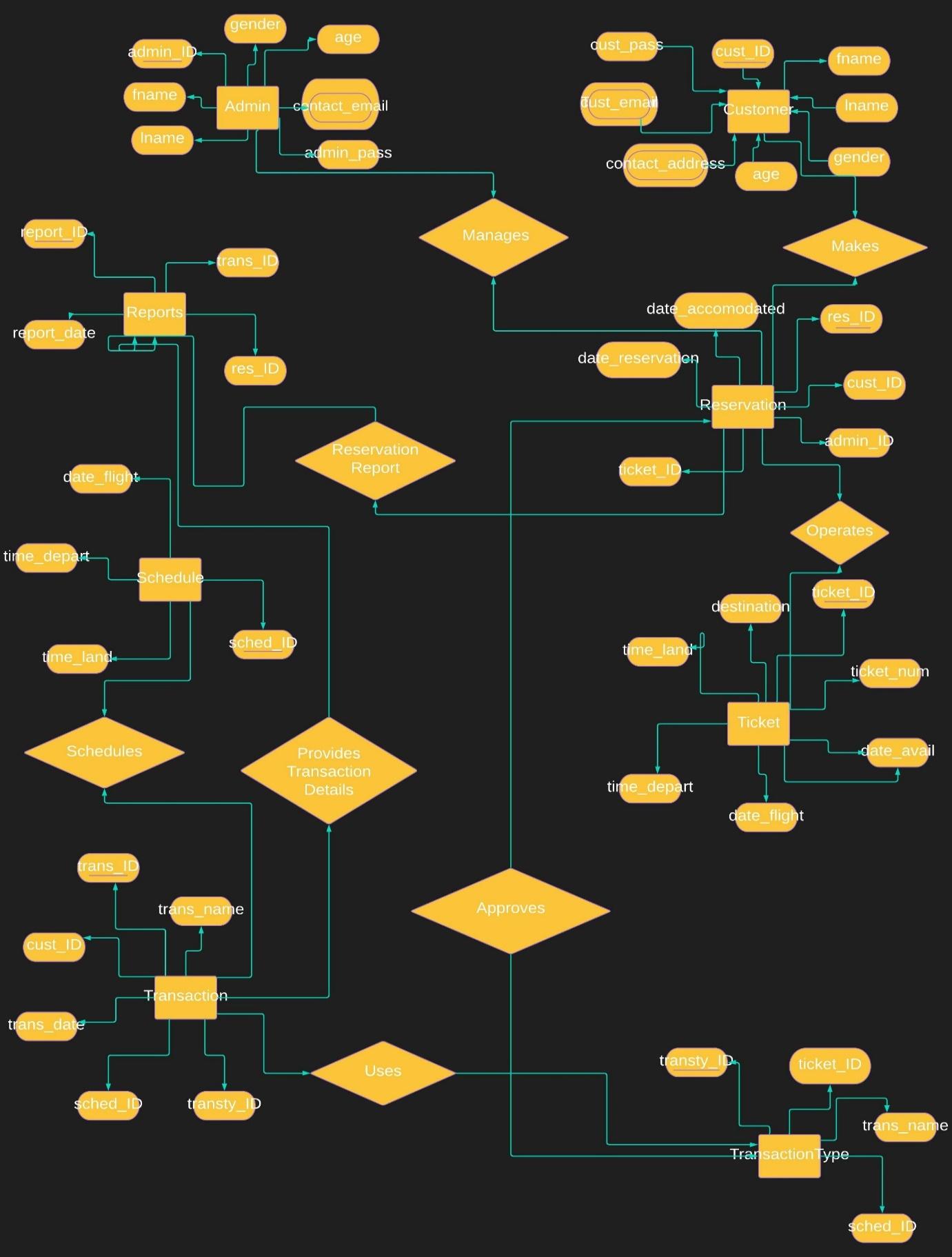
ERD relationship symbols

Within entity-relationship diagrams, relationships are used to document the interaction between two entities. Relationships are usually verbs such as assign, associate, or track and provide useful information that could not be discerned with just the entity types.

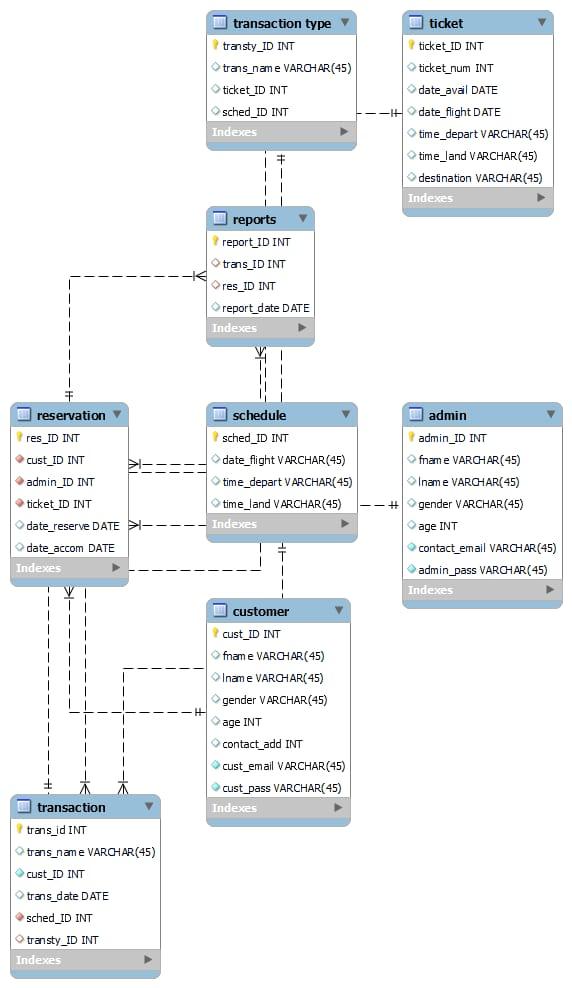
| Relationship Symbol | Name | Description |
| --- | --- | --- |
| Relationship Symbol | Relationship | Relationships are associations between or among entities. |
| Weak Relationship Symbol | Weak relationship | Weak Relationships are connections between a weak entity and its owner. |

ERD attributes are characteristics of the entity that help users to better understand the database. Attributes are included to include details of the various entities that are highlighted in a conceptual ER diagram.

| Attribute Symbol | Name | Description |
| --- | --- | --- |
| Attribute Symbol | Attribute | Attributes are characteristics of an entity, a many-to-many relationship, or a one-to-one relationship. |
| Multivalued Attribute Symbol | Multivalued attribute | Multivalued attributes are those that are can take on more than one value. |
| Derived Attribute Symbol | Derived attribute | Derived attributes are attributes whose value can be calculated from related attribute values. |
| Relationship Symbol | Relationship | Relationships are associations between or among entities. |

LOGICAL ER DIAGRAM:

PHYSICAL ER DIAGRAM:

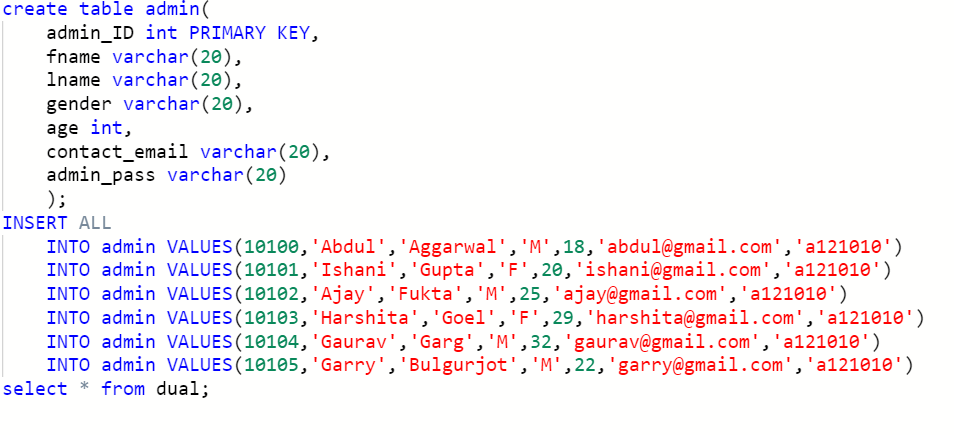


**Brief Introduction:**

Remember the days when to buy tickets for a vacation, you had to call up ticket agents, find the airlines, search for deals, and so on? Well, since the Internet took off, things haven't been the same. Now you have many Web sites from both third-party vendors and airlines offering anything from cars to cruises to vacation packages, with deal comparisons and what have you. Clearly, the consumer is the winner! As a part of your learning experience using Web Logic Server Internet- and Web-enabled applications have revolutionized the way businesses are carried out., We build a simple airline ticket booking system, which will model for a real-world. We'll begin by briefly outlining the features that will be provided by your airline ticket booking system. This Airline reservation helps users and the admin to access the details as they needed, as we the user wants the each and every detail of the our flight and all other things so this database will provide us with the same For example, the customer table will provide each and every necessary detail of customer and Ticket table will provide the details of the tickets and all other things related to it. For the transaction and billing problems to be sorted we have different tables such as Transaction and Transaction type table this will provide the details of each transaction and its type and all the details associated with it and the all type of data is managed by the admin and the report is created of each type of things happening on the web application. It contains information on schedules and fares and contains a database of reservations (or passenger name records) and of tickets issued. Airlines use databases for storing passenger information when tickets are booked and when passengers check-in at the airport.

**STRUCTURE OF CASE STUDY:**

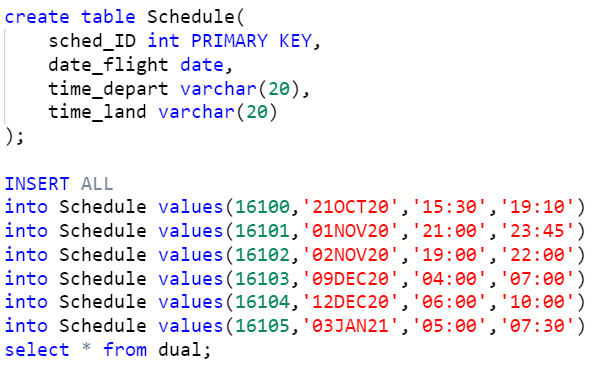
**Admin table**: All the details of the admin and the admin\_ID is the primary key through which it can access all the details.



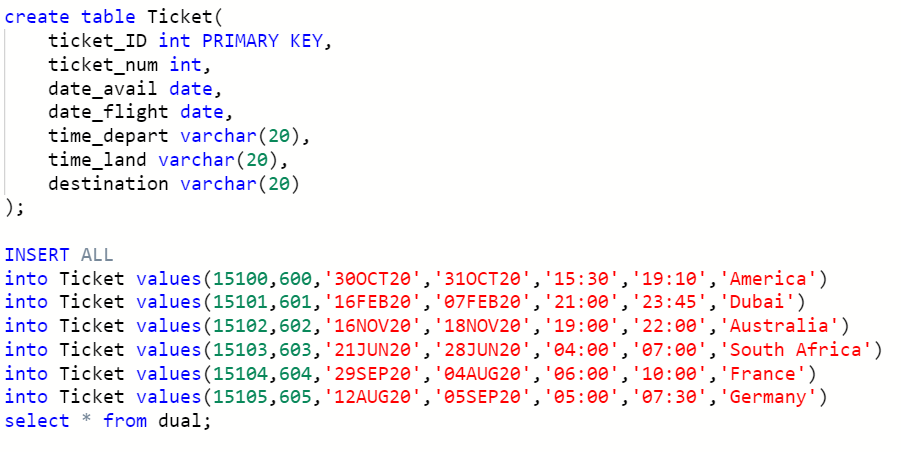
**Customer table:** All the details of the customer in which cust\_ID as the primary key is used to access all the customer details.



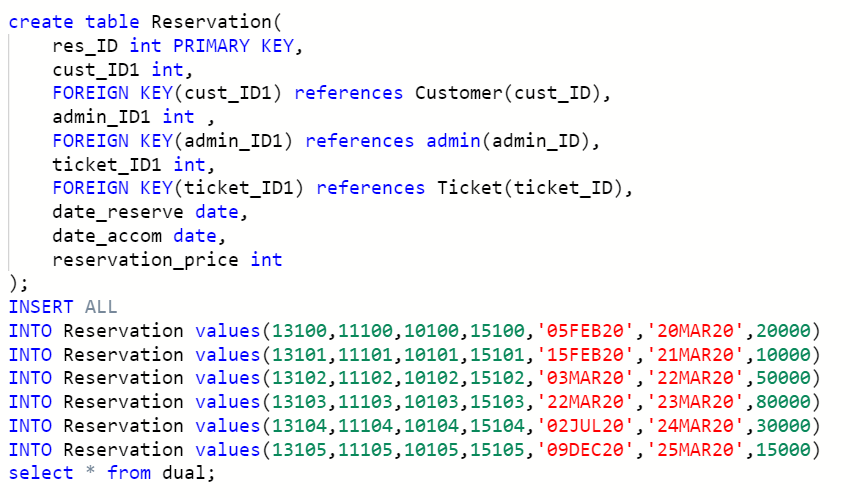
**Schedule table:** It contains all the schedules of the flight in which we used sched\_ID as primary key to get all the details.



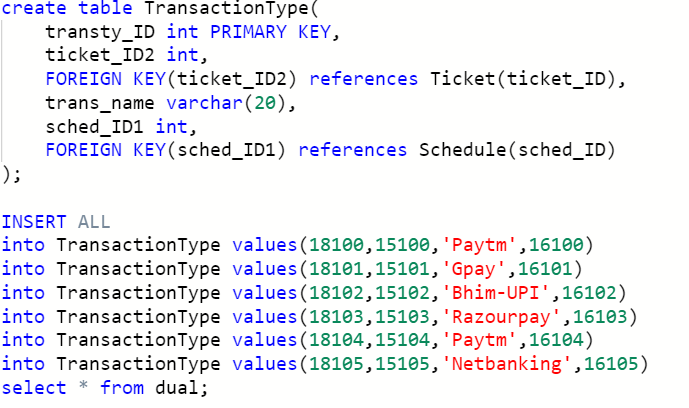
**Ticket table:** It tells all the ticket details and we used ticket\_ID as primary key to fetch all the details on the particular ticket



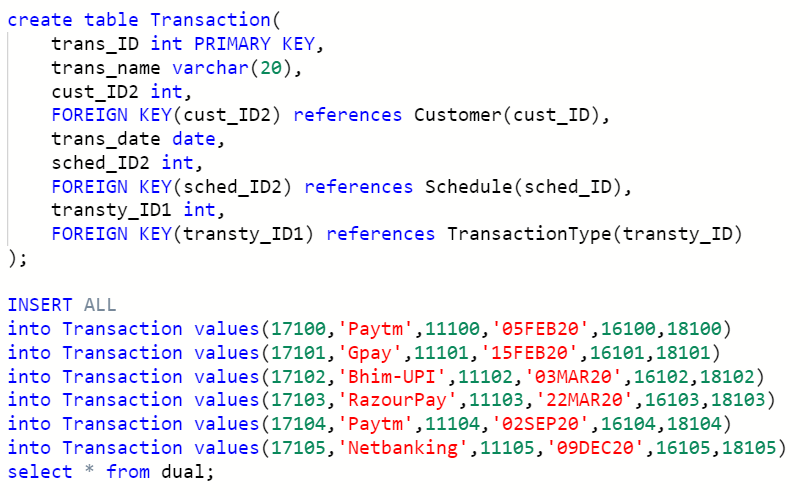
**Reservation table:** It tells about the reservations made by the customer by using the res\_ID we can fetch all the reservation details.



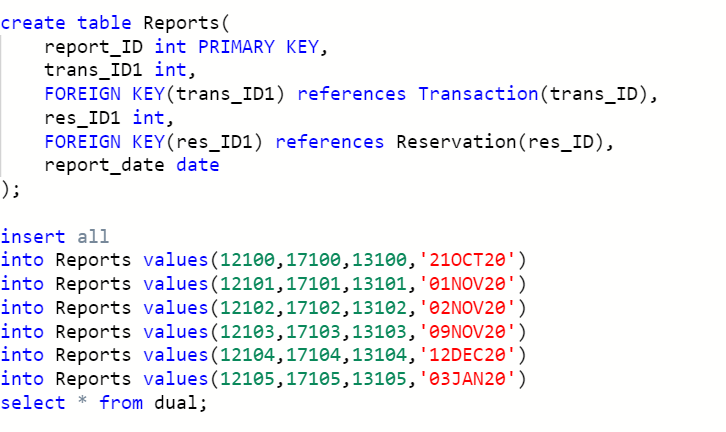
**Transaction Type table:** It tells about the type of the transaction made for the reserving the ticket and we can use transty\_ID as primary key to get transaction type details.



**Transaction table:** It tells about the transaction made by using trans\_ID as primary key to fetch transaction details.

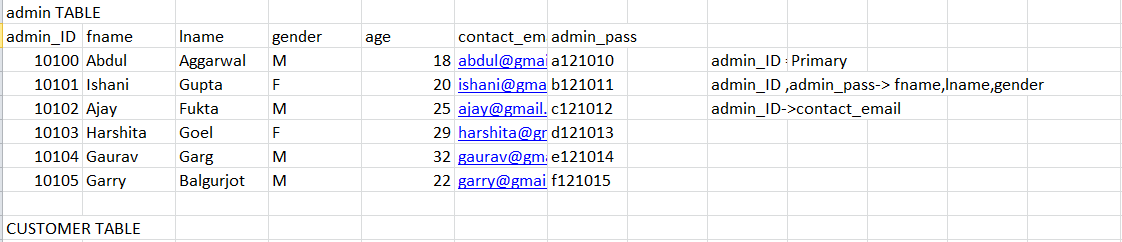


**Reports table:** It makes the reports for the reservation and transaction made by the admin .he uses report\_ID to get all the report.

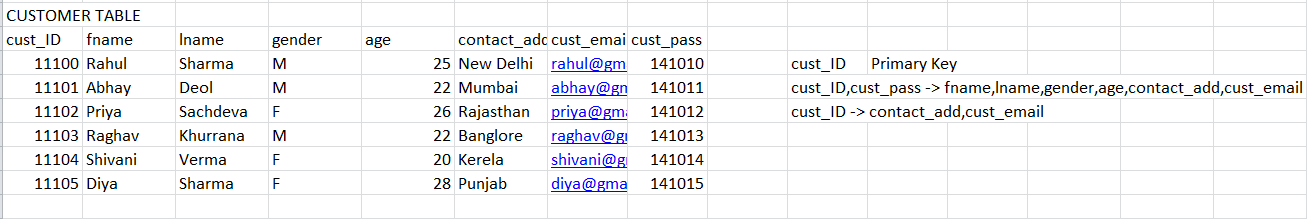


**KEYS AND FUNCTIONAL DEPENDENCIES:**

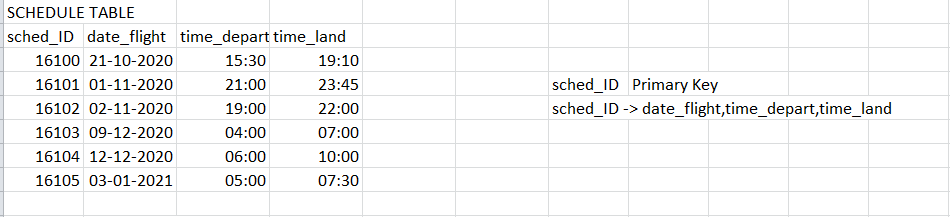
**Admin table**:



**Customer table:**



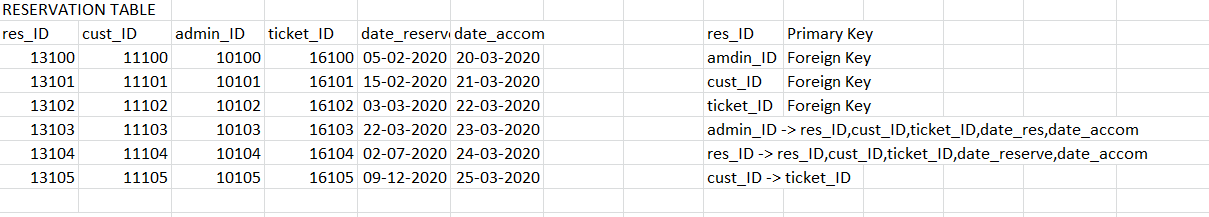
**Schedule table:**



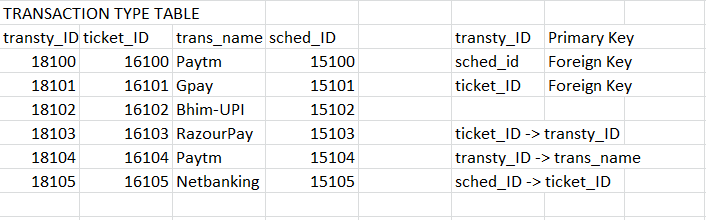
**Ticket table:**



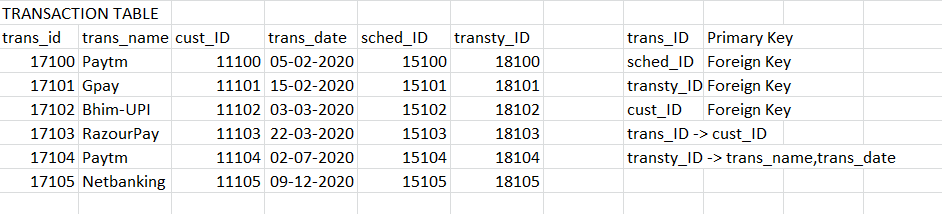
**Reservation table:**



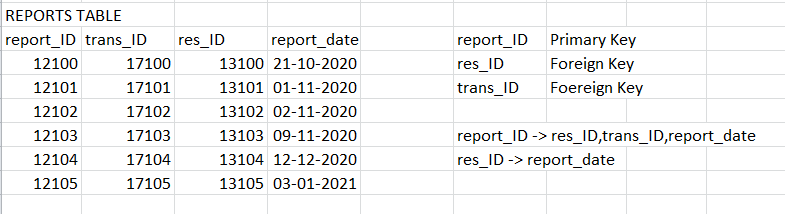
**Transaction Type table:**



**Transaction table:**



**Reports table:**

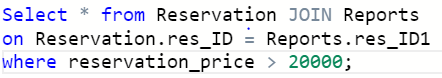


|  |  |  |
| --- | --- | --- |
|  |  |  |

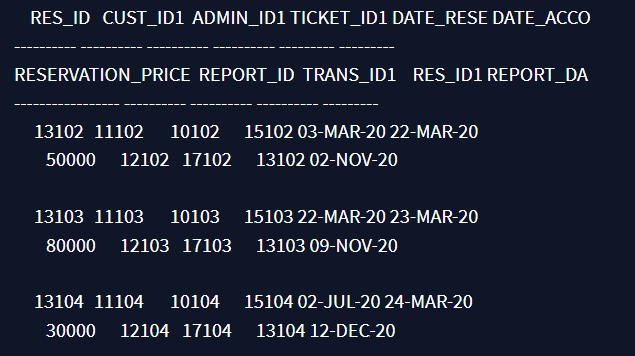
**INTERACTIVE QUERIES:**

**QUERY1:**

**To get the report where reservation price is greater than 20000**

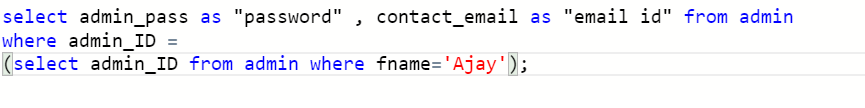


**OUTPUT:**

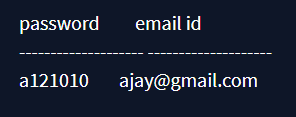


**QUERY 2:**

**To get the email id and password of admin Ajay**

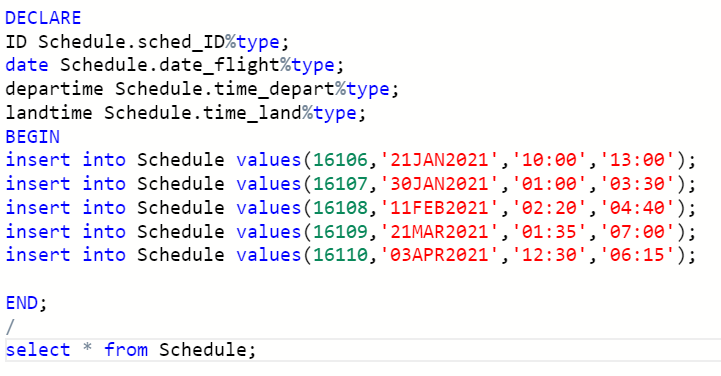


**Output:**

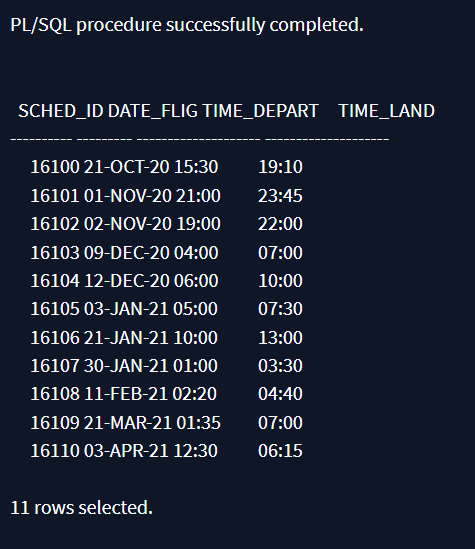


**QUERY 3:**

**To insert more flight schedule using PL/SQL queries in Schedule table.**

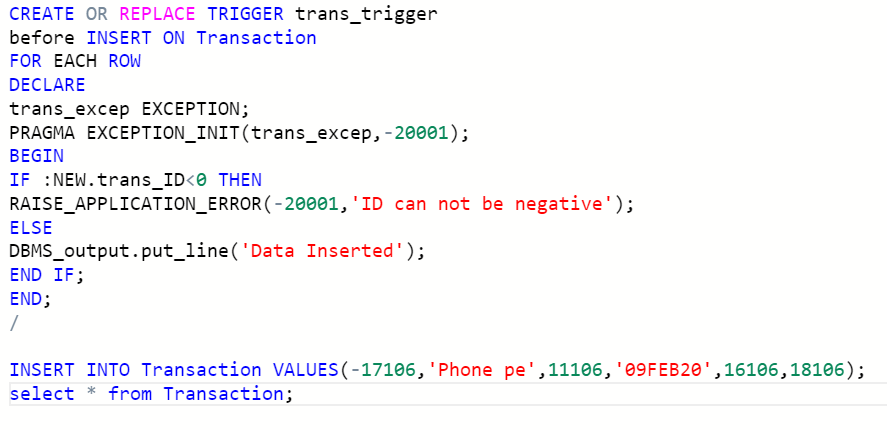


**Output**:

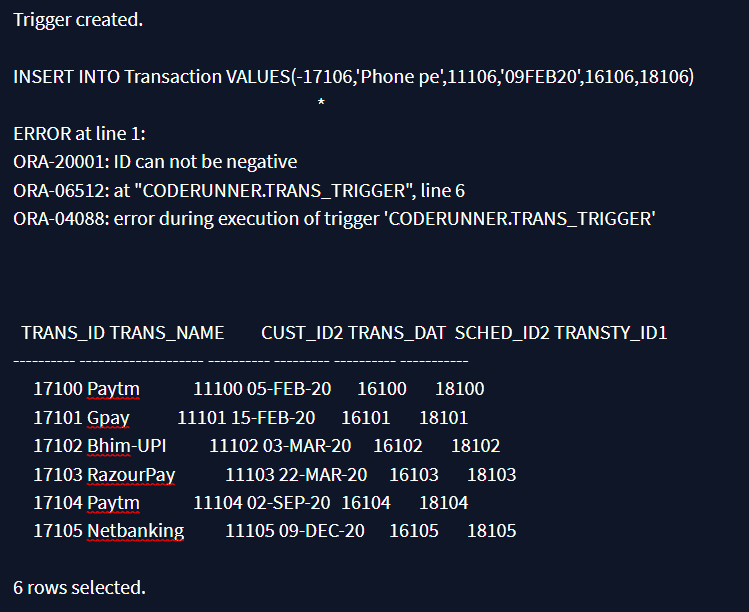


**QUERY 4:**

**To create a trigger for transaction table where trans\_ID can’t br negative**

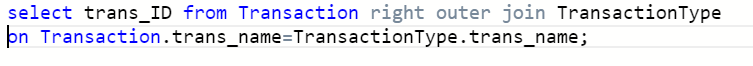


**Output:**

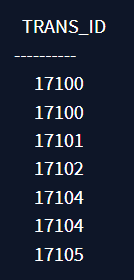


**QUERY 5:**

**To get trans\_ID from where transaction name are equal in both transaction table and transaction type table**



**Output:**

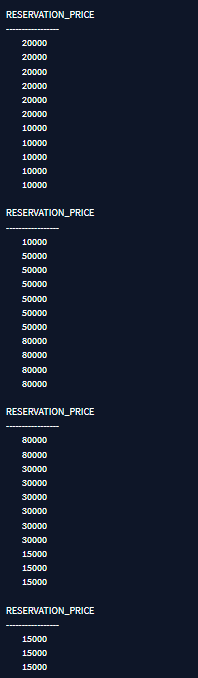


**Query 6:**

**To get reservation price from transactions made**



**Output:**

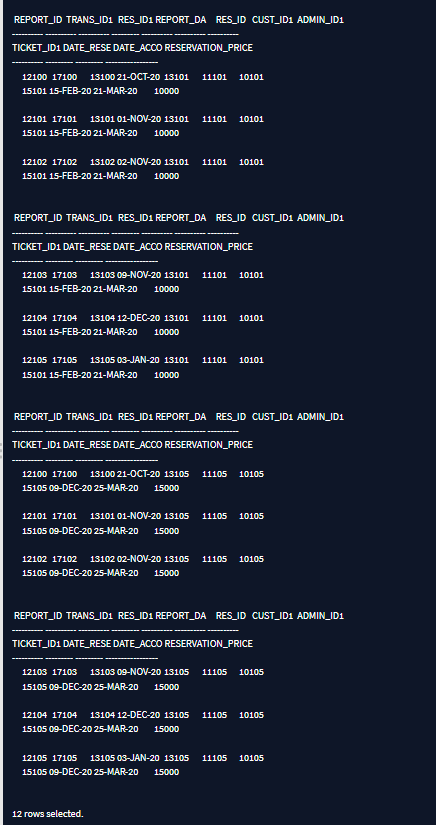


**QUERY 7:**

**To fetch reports where reservation price is less than 20000**



**Output:**

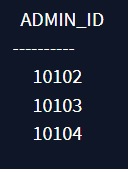


**QUERY 8:**

**To select admin\_ID of the admin that manages the customer with age above 20**

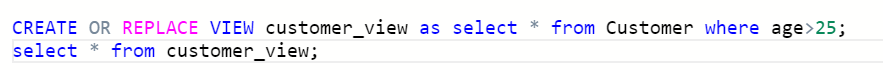


**Output:**



**Query 9:**

**To get a view from Customer table where their age would be more than 25**

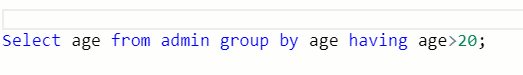


**Output:**



**Query 10:**

**To get admin age where age of admin is greater than 20**



**Output:**



**NORMALISATION**

**Admin Table :-**

Here,

Admin\_ID is Primary Key

admin\_ID is Candidate Key as It is traversing all other attributes.

1NF -> This table is already normalised to 1st Normal Form as it has no multivalued attributes/Atomicity.

2NF -> This table is already normalised to 2nd Normal Form as it is in 1NF and it has no partial Dependency.

3NF -> This table is not Normalised to 3rd Normal Form as condition/Functional Dependency= Candidate Key->Non-Prime is there.

So, we will Decompose the table into two tables.

DECOMPOSED TABLE:

|  |
| --- |
| Admin\_ID |
| Fname |
| Lname |
| Gender |
| Age |
| Contact\_email |
|  |

Table1:- Table2:-

|  |
| --- |
| Admin\_ID |
| Fname |
| Lname |
| Gender |
| Age |
| Contact\_email |
| Admin\_pass |

**Customer Table:-**

Here,

cust\_ID is Primary Key

cust\_ID is Candidate Key as It is traversing all other attributes.

1NF -> This table is already Normalised to 1st Normal Form as it has no multivalued attributes/Atomicity.

2NF -> This table is already normalised to 2nd Normal Form as it is in 1NF and it has no partial Dependency.

3NF -> This table is not Normalised to 3rd Normal Form as condition/Functional Dependency= Candidate Key->NonPrime is there .

So we will Decompose the table into two tables.

Decomposed Table:

Table1:- Table2:-

|  |
| --- |
| Cust\_ID |
| Fname |
| Lname |
| Gender |
| Age |
| Contact\_add |
| Cust\_email |
| Cust\_pass |
| Contact\_add |
| Cust\_email |
| Cust\_ID |

**Reports:-**

Here,

report\_ID is Primary Key

report\_ID is candidate key

1NF -> This table is already Normalised to 1st Normal Form as it has no multivalued attributes/Atomicity.

2NF -> This table is already normalised to 2nd Normal Form as it is in 1NF and it has no partial Dependency.

3NF -> This table is not Normalised to 3rd Normal Form as condition/Functional Dependency= Non-Prime->Non-Prime is there .

So we will Decompose the table into two tables.

Decomposed Table:

**Table1:- Table2:-**

|  |
| --- |
| **Report\_ID** |
| **Trans\_ID** |
| **Res\_ID** |
| **Report\_date** |
| res\_ID |
| report\_date |

**Reservation Table:-**

Here,

res\_ID is Primary Key

admin\_ID is candidate key

admin\_ID,cust\_ID,ticket\_ID is foreign key

1NF -> This table is already Normalised to 1st Normal Form as it has no multivalued attributes/Atomicity.

2NF -> This table is already Normalised to 2nd Normal Form as it is in 1NF and it has no partial Dependency.

3NF -> This table is not Normalised to 3rd Normal Form as condition/Functional Dependency= Non Prime->NonPrime is there .

So we will Decompose the table into three tables.

Decomposed Table:

Table1:- Table2:-

|  |
| --- |
| Res\_ID |
| Cust\_ID |
| Admin\_ID |
| Ticket\_ID |
| Res\_ID |
| Cust\_ID |
| Ticket\_ID |
| Date\_reserve |
| Date\_accom |

Table 3:-

|  |
| --- |
| Cust\_ID |
| Ticket\_D |

**Schedule Table:-**

Here,

sched\_ID is Primary Key

sched\_ID is candidate key

admin\_ID,cust\_ID,ticket\_ID is foreign key

1NF -> This table is already Normalised to 1st Normal Form as it has no multivalued attributes/Atomicity.

2NF -> This table is already Normalised to 2nd Normal Form as it is in 1NF and it has no partial Dependency.

3NF -> This table is already Normalised to 3rd Normal Form as condition/Functional Dependency= Prime->NonPrime is there also No Transitive Dependency.

**Ticket Table:-**

Here,

ticket\_ID is Primary Key

ticket\_ID is candidate key

1NF -> This table is already Normalised to 1st Normal Form as it has no multivalued attributes/Atomicity.

2NF -> This table is already Normalised to 2nd Normal Form as it is in 1NF and it has no partial Dependency.

3NF -> This table is not Normalised to 3rd Normal Form as condition/Functional Dependency= Candidate Kay->NonPrime is there .

So we will Decompose the table into two tables.

Decomposed Table:

Table 1:- Table2:-

|  |
| --- |
| Ticket\_ID |
| Ticket\_num |
| Ticket\_num |
| Date\_avail |
| Date\_flight |
| Time\_depart |
| Time\_land |
| Destination |

**Transaction Table:-**

Here,

cust\_ID is Primary Key

trans\_ID is Primary key

sched\_ID is foreign key

transty\_ID is foreign key

transty\_ID is Candidate Key

1NF -> This table is already Normalised to 1st Normal Form as it has no multivalued attributes/Atomicity.

2NF -> This table is already Normalised to 2nd Normal Form as it is in 1NF and it has no partial Dependency.

3NF -> This table is not Normalised to 3rd Normal Form as condition/Functional Dependency= Non Prime->Prime is there.

So we will Decompose the table into three tables.

Decomposed Table:

|  |
| --- |
| Transty\_Id |
| Trans\_ID |
| Trans\_name |
| Trans\_date |
| Trans\_ID |
| Cust\_Id |
| Sched\_ID |
| Trans\_ID |

Table1:- Table2:- Table3:-

**Transaction Type:-**

Here,

transty\_ID is Primary Key

sched\_ID is foreign key

ticket\_ID is foreign key

1NF -> This table is already Normalised to 1st Normal Form as it has no multivalued attributes/Atomicity.

2NF -> This table is already Normalised to 2nd Normal Form as it is in 1NF and it has no partial Dependency.

3NF -> This table is not Normalised to 3rd Normal Form as Transitive Dependency is there.

So we will Decompose the table into three tables.

Decomposed Table:

|  |
| --- |
| Sched\_ID |
| Ticket\_Id |

Table1:- Table 2:- Table3:-

|  |
| --- |
| Transty\_Id |
| Ticket\_ID |
| Transty\_ID |
| Trans\_name |

**Conclusion and Future work:**

We are trying to give a live reporting which is updated by Airline Companies so that customer gets a live Flights checking, Available seats, Pricing and also planning to provide seats as per theirs choice so that they can travel very comfortably their journey. We will be trying to provide food facility and choice to customers so that they can feel like their home and more effective amenities. We are also trying to make more attention on Business class people and their requirements. Our future planning is to take this project towards an Android App and QR Code Scanning. So that a Customer can easily contact to the Airlines and they are getting quick Services from Airlines. We also want in future to place in market so that customer can take more advantages and saves their important time. We are also finding and approaching to companies which are using this type of software.

**BIBLIOGRAPHY:**

* [**https://www.lucidchart.com/**](https://www.lucidchart.com/)
* [**https://codequotient.com/code**](https://codequotient.com/code)
* [**https://www.flightslogic.com/airline-ticket-reservation-system.php**](https://www.flightslogic.com/airline-ticket-reservation-system.php)