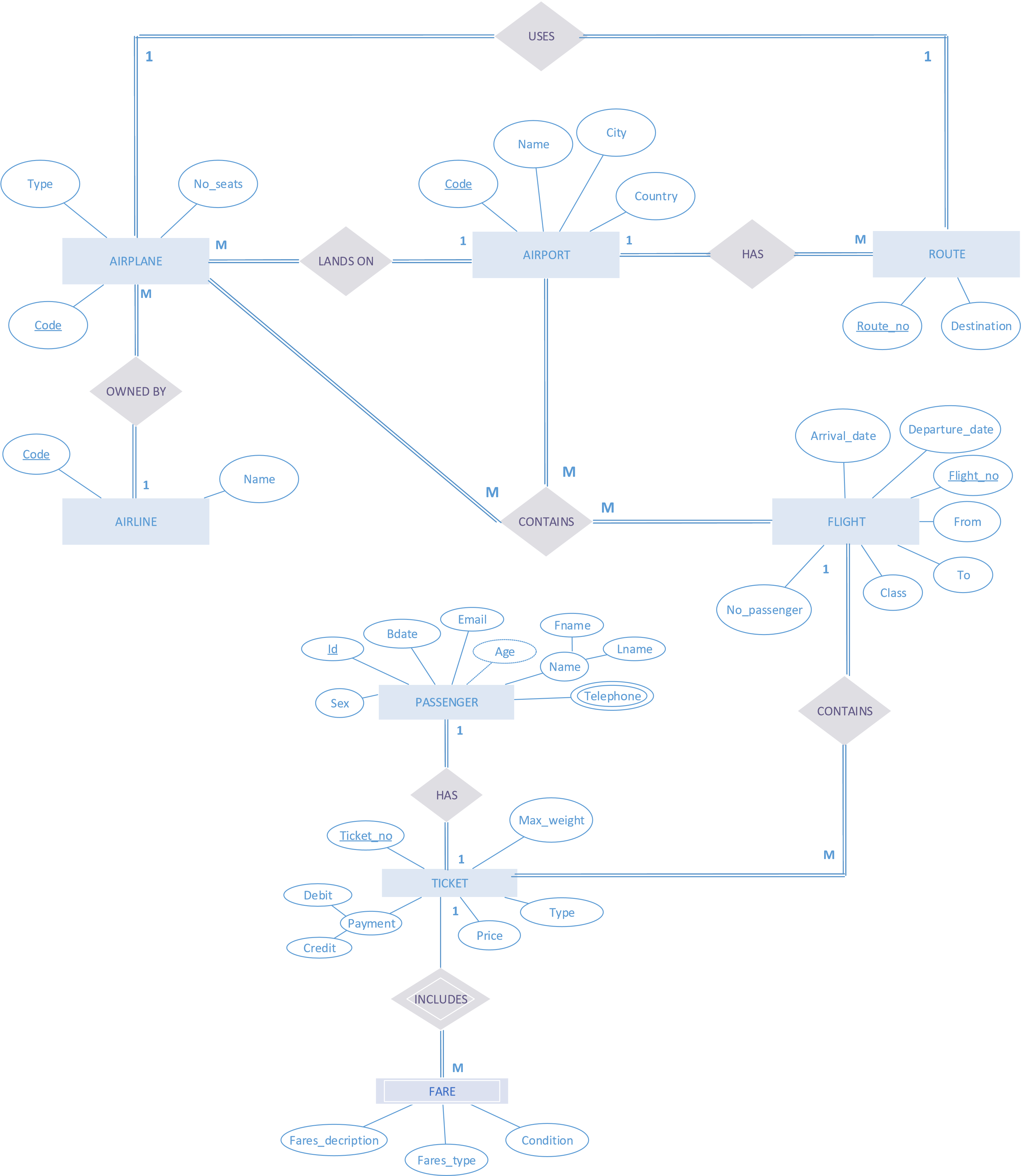
**Airline Online Reservation**

|  |  |  |
| --- | --- | --- |
| Name | ID | E-mail |
| Aalaa | **20201818** |  |
| Abdelrahman | **20201840** |  |
| Ahmed | **20201802** |  |
| Clara | **20201861** |  |
| Roudina | **20201828** |  |

**Scope**

The name of our database is " AIRLINE ONLINE RESERVATION DATABASE " . With the help of this system , customers can view all different flight's availability with different timings for a particular date and it also allow them to reserve a seat , cancel reservation or modify it . The database is very flexible to insert new data or modify an existence data about a passenger , add or delete columns , retrieve data about specific passenger, remove table with all constraints that make it secure ( not to allow anyone to make changes in it ) .



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Fname | Lname | Sex | Email | Bdate | Ticket\_no |

**Passenger**

**Phone**

|  |  |
| --- | --- |
| ID | Telephone |

**Ticket**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Price | Credit | Debit | Flight\_no | Type | Max\_weight |

**Fare**

|  |  |  |  |
| --- | --- | --- | --- |
| Fares\_description | Condition | Fares\_type | Ticket\_no |

**Route Airline**

|  |  |  |  |
| --- | --- | --- | --- |
| No | Destination | Airport\_code | Airplane\_code |

|  |  |
| --- | --- |
| code | Name |

**Airport**

|  |  |  |  |
| --- | --- | --- | --- |
| City | Country | code | Name |

**Airplane**

|  |  |  |  |
| --- | --- | --- | --- |
| code | Type | No\_seats | Airline\_code |

**Flight**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | From | To | Class | Arrival\_date | Depature\_date | No\_passenger |

**Plane\_flight\_port**

**Airport-Airplane**

|  |  |  |
| --- | --- | --- |
| Airplane\_code | Airport\_code | Flight\_no |

|  |  |
| --- | --- |
| Airport\_code | Airplane\_code |

**Description**

Let’s explain our Database in a simple way to be understandable :

First of all , We have tables connected with each other to facilitate the data access path in the database.

* **Tables ( Entities ) with their columns ( Attributes ) :**

1. Here we have the first table which is called PASSENGER that has more than one column (attribute) : ID ( which is considered to be the primary key of this table so it will be underlined ) , Name ( which is considered as a composite attribute of fname , lname so we write them directly as columns ) , sex , BDate , Email , age , telephone ( which is considered as multivalued attribute so we will take it off from this table to be in another table with the primary key ) .
2. The Second table is called TICKET that has more than one column (attribute) : No ( which is considered to be the primary key of this table so it will be underlined ) , price , type , max\_weight , payment which is considered as a composite attribute of credit , debit so we write them directly as columns ) .
3. The Third table is called FARES that has more than one column (attribute) : type , description , condition . It is considered as weak entity.
4. The Fourth table is called FLIGHT that has more than one column (attribute) : No ( which is considered to be the primary key of this table so it will be underlined ) , class , from , to , no\_of\_passengers , arrival\_date, departure\_date .
5. The Fifth table is called AIRPLANE that has more than one column (attribute) : code ( which is considered to be the primary key of this table so it will be underlined ) , type , no\_of\_seats .
6. The Sixth table is called AIRLINE that has more than one column (attribute) : code ( which is considered to be the primary key of this table so it will be underlined ) , name .
7. The Seventh table is called AIRPORT that has more than one column (attribute) : code ( which is considered to be the primary key of this table so it will be underlined ) , name , city , country .
8. The Eighth table is called ROUTE that has more than one column (attribute) : No ( which is considered to be the primary key of this table so it will be underlined ) , destination .

* **Relations between these tables ( Entities ) :**

1. [ has ] :

* Between PASSENGER & TICKET tables .
* Its cardinality : 1 to 1 relationship cause each passenger has only one ticket .
* Its participation : must – must cause each passenger must have a ticket .

Because of its cardinality ( 1 to 1 ) , we take one of the primary keys of each table & put it in the other one as foreign key . Here we take Ticket\_no ( primary key of TICKET table ) & put it in the PASSENGER table as foreign key .

1. [ include ] :

* Between TICKET & FARES tables .
* Its cardinality : 1 to M relationship cause each ticket include more than one fares .
* Its participation : may – must cause each ticket may include fares or not .
* It's considered as a Weak Relationship .

Because of its cardinality ( 1 to M ) , we take the primary key of the 1-side & put it in the M-side as foreign key . Here we take Ticket\_no ( primary key of TICKET table ) & put it in the FARES table as foreign key .

1. [ contain ] :

* Between FLIGHT & TICKET tables .
* Its cardinality : 1 to M relationship cause each flight has more than one ticket .
* Its participation : must – must cause each flight must have a ticket .

Because of its cardinality ( 1 to M ) , we take the primary key of the 1-side & put it in the M-side as foreign key . Here we take Flight\_no ( primary key of FLIGHT table ) & put it in the TICKET table as foreign key .

1. [ contain ] :

* Between AIRPLANE , AIRPORT & FLIGHT tables ( Ternary Relationship ).
* Its cardinality : M to M to M relationship cause airport has more than one airplane & can manage more than one flight .
* Its participation : must – must-must cause airport must have more than one airplane & must manage more than one flight .

Because of its cardinality ( M to M to M ) , we take the primary keys of each table ( airport\_code , airplane\_code , flight\_no) & put them in a new table which is called P-F-P . All of them together will be the primary key of the new table ( P-F-P table ).

1. [ owned by ] :

* Between AIRLINE & AIRPLANE tables .
* Its cardinality : 1 to M relationship cause an airline own many airplanes .
* Its participation : must – must cause an airline must own airplanes & airplanes must be owned by airline .

Because of its cardinality ( 1 to M ) , we take the primary key of the 1-side & put it in the M-side as foreign key . Here we take airline\_code ( primary key of AIRLINE table ) & put it in the AIRPLANE table as foreign key .

1. [ land on] :

* Between AIRPORT & AIRPLANE tables .
* Its cardinality : 1 to M relationship cause many airplanes lands on one airports
* Its participation : must – must cause airplane must land on airport .

Because of its cardinality ( 1 to M ) , we take the primary key of the 1-side & put it in the M-side as foreign key , But we take both foreign keys & put them in different table . We put airplane\_code & airport\_code in that table , airplane\_code refers to ( code ) in AIRPLANE table & airport\_code refers to ( code ) in AIRPORT table .

1. [ has ] :

* Between AIRPORT & ROUTE tables .
* Its cardinality : 1 to M relationship cause an airport has many routes .
* Its participation : must – must cause an airport must has routes .

Because of its cardinality ( 1 to M ) , we take the primary key of the 1-side & put it in the M-side as foreign key . Here we take airport\_code ( primary key of AIRPORT table ) & put it in the ROUTE table as foreign key .

1. [ use ] :

* Between AIRPLANE & ROUTE tables .
* Its cardinality : 1 to 1 relationship cause an airplane uses only one route .
* Its participation : must – must cause an airplane must use a route .

Because of its cardinality ( 1 to 1 ) , we take one of the primary keys of each table & put it in the other one as foreign key so we take airplane\_code & put it in ROUTE table as foreign key.

* **Tables after simplifying Relationships :**

1. PASSENGER

* Columns : ID , fname , lname , sex , BDate , Email ,age , Ticket\_no .
* Primary key : ID .
* Foreign key : Ticket\_no refers to ( No ) in TICKET table .

1. PASSENGER\_TELEPHONE

* Columns : ID , telephone .
* Primary key : ID , telephone together
* Foreign key : ID refers to ( ID ) in PASSENGER table .

1. TICKET

* Columns : No , type , price , max\_weight , credit , debit , Flight\_no .
* Primary key : No .
* Foreign key : Flight\_no refers to ( No ) in FLIGHT table .

1. FARES

* Columns : type , description , condition , Ticket\_no .
* Primary key : no primary key
* Foreign key : Ticket\_no refers to ( No ) in TICKET table .

1. FLIGHT

* Columns : No , class , from , to , arrival\_date , departure\_date , no\_of\_passenger
* Primary key : No .
* Foreign key : no foreign key

1. AIRPLANE

* Columns : code , type , no\_of\_seats , airline\_code .
* Primary key : code .
* Foreign key : airline\_code refers to ( code ) in AIRLINE table .

1. AIRLINE

* Columns : code , name .
* Primary key : code .
* Foreign key : no foreign key .

1. AIRPORT

* Columns : code , name , city , country .
* Primary key : code .
* Foreign key : no foreign key .

1. ROUTE

* Columns : No , destination , airport\_code , airplane\_code .
* Primary key : No .
* Foreign key : airport\_code refers to ( code ) in AIRPORT table , airplane\_code refers to ( code ) in AIRPLANE table.

1. PLANE-FLIGHT-PORT

* Columns : airport\_code , flight\_no , airplane\_code .
* Primary key : airport\_code , flight\_no & airplane\_code together .
* Foreign key : airport\_code refers to ( code ) in AIRPORT table , flight\_no refers to ( No ) in FLIGHT table , airplane\_code refers to ( code ) in AIRPLANE table .

1. AIRPORT\_AIRPLANE

* Columns : airport\_code , airplane\_code .
* Primary key : Both .
* Foreign key : airport\_code refers to ( code ) in AIRPORT table , airplane\_code refers to ( code ) in AIRPLANE table.
* **Queries :**

1. **First query :** select all from passenger **,** ticket where price > 2000.

(discuss the operational operator bigger than 2000) .

1. **Second query:** select flight number , no of flights that travels in 2020 .

(discuss the aggregate function count that counts the number of flights that travel during 2020) .

1. **Third query :** select fare type , fname as first name, lname as last name , birthdate where birthyaear >= 2000 .

(discuss the inner join that is used to join data in more than one table . Here we join FARES with TICKET with PASSENGER by fare.ticket\_no = ticket.t\_no & ticket.t\_no = passenger.ticket\_no) .

1. **Forth query :** select route number , destination , class ,flight number where class of flight = ' A ' &group by flight number .

(discuss the inner join that is used to join data in more than one table . Here we join ROUTE with AIRPLANE with PLANE\_FLIGHT\_PORT by route.plane\_code = airplane.plane\_code & airplane.plane\_code = plane\_flight\_port.plane\_code & flight.flight\_no = plane.flight.port.plane\_code) .

1. **Fifth query :** select number of flights , flight no that travels to ' BOM ' group by flight no .

(discuss count aggregate function) .

1. **Sixth query :** select fname , lnameas passenger name that flight travel to ' FRA ' , order by passenger name .

(discuss the inner join that is used to join data in more than one table . Here we join FLIGHT with TICKET with PASSENGER by passenger.ticket\_no = ticket.t\_no & ticket.flight\_no = flight.flight\_no ).

1. **Seventh query :** select flight numbers that arrived in April & count there numbers group by flight number .

(discuss count aggregate function to count no of flights in April) .

1. **Eighth query :** select airline code , airline name , flight no that travelled at 02 .

(discuss the inner join that is used to join data in more than one table . Here we join AIRLINE with AIRPLANE with FLIGHT with PLANE\_FLIGHT\_PORT by airline.airline\_code = airplane.airplane\_code & plane\_flight\_port.plane\_code = airplane.airplane\_code & flight.flight\_code = plane\_flight\_port.flight\_no , where date = 02) .

1. **Ninth query :** select \* from passenger whose name contains letter ' a ' .

(discuss alphabetical letters to select the name of passenger whose fname has letter ' a ' in his name ).

**10-Tenth query :** select ticket type, fname , lname as full name whose gender is male .

(discuss concatenation between two attribute to be one attribute & discuss inner join between TICKET with PASSENGER by ticket.t\_no = passenager.ticket\_no where passenager's gender is male ' M ').

**From 11 🡪** **14 : Nested Queries**

**11-Eleventh query :** select fname , lname as passenger name whose flight is booked by credit card.

(discuss inner join between PASSENGER with TICKET by passenger.ticket\_no = ticket.t\_no ).

**12-twelvth query :** select fname , lname as full name , ticket no whose ticket included fares without duplication .

**13- Thirteenth query :** select all from ticket that has maximimum ticket price .

(discuss maximum price from the TICKET tabe).

**14-Fourteenth query :** select all from passenger whose sum of their tickets is bigger than 3500 then group by them by ticket no .

(discuss aggregate function that return total summation of their tickets ).

**15- Fifteenth query :** select fname , lname as full name whose fname contains ' a ' , lname contains ' m ' & gender is female .

(discuss inner join between PASSENGER with TICKET by ticket.t\_no = passenger.ticket\_no where gender is female ' F ' , fname contains letter ' a ' , lname contains letter ' m ').

**16- Sixteenth query :** select all from ticket , fares whose ticket no has fares & gender is female .

(discus nested query & full outer join between TICKET with FARES by ticket.t\_no = fare.ticket\_no) .