首先，由ODL方法建立如下的数据模型：

Interface Ticket(key ticketID){ // 机票

Attribute string ticketID; // 机票编号

Attribute string class; // 舱位等级

Attribute string seatNumber; // 座位号

Attribute integer price; // 价格

Relationship Passenger ticketOwner inverse Passenger:: ownsTicket; // 联系乘客并指定反向联系

Relationship Flight ticketFlight inverse Flight:: hasTicket; // 联系航班并指定反向联系

}

Interface Flight(key flightNumber) { // 航班

Attribute string flightNumber; // 航班号

Attribute string duration; // 航班时长

Attribute string takeOffTime; // 起飞时间

Attribute string landingTime; // 降落时间

Attribute date flightDate; // 航班日期

Relationship Airplane flightAirplane inverse Airplane:: hasFlight; // 联系飞机并指定反向联系

Relationship Airport takeOffAirport inverse Airport::inboundFlight; // 联系出发机场并指定反向联系

Relationship Airport landingAirport inverse Airport::departureFlight; // 联系到达机场并指定反向联系

Relationship Airlines operatedBy inverse Airlines:: operates; // 联系航空公司并指定反向联系

Relationship Set<Ticket> hasTicket inverse Ticket::ticketFlight; // 联系机票集合并指定反向联系

}

Interface Runway (key runwayID){ // 跑道

Attribute string runwayID; // 跑道编号

Attribute integer length; // 跑道长度

Attribute integer width; // 跑道宽度

Relationship Airport runwayAirport inverse Airport:: hasRunway; // 联系机场并指定反向联系

}

Interface Airlines (key airlinesName) { // 航空公司

Attribute string airlinesName; // 航空公司名称

Attribute string abbreviation; // 航空公司缩写

Attribute integer numOfPlanes; // 飞机数量

Relationship Set<Flight> operates inverse Flight::operatedBy; // 联系航班集合并指定反向联系

Relationship Set<Airplane> hasAirplane inverse Airplane::belongsTo; // 联系飞机集合并指定反向联系

Relationship Set<Captain> employs inverse Captain::employedBy; // 联系机长集合并指定反向联系

}

Interface Captain(key captainID) { // 机长

Attribute string captainName; // 机长姓名

Attribute string captainsex; // 性别

Attribute integer captainID; // 机长编号

Attribute integer captainPhone; // 机长电话

Attribute date captainbirthdate; // 出生日期

Relationship Airlines employedBy inverse Airlines::employs; // 联系所属航空公司并指定反向联系

Relationship Airplane flies inverse Airplane::flightCaptain; // 联系所飞飞机并指定反向联系

}

Interface Passenger(key passengerID) {//乘客

Attribute string passengerName; // 乘客姓名

Attribute integer passengerID; // 乘客编号

Attribute integer passengerIphone; // 联系电话

Attribute date passengerbirthdate; // 出生日期

Attribute string credit; // 信用

Relationship Set<Ticket> ownsTicket inverse Ticket::ticketOwner; // 联系拥有的机票集合并指定反向联系

}

Interface Airplane (key airplaneID) {//飞机

Attribute integer airplaneID; // 飞机编号

Attribute string airplaneType; // 飞机类型

Attribute integer maxPassenger; // 最大载客量

Attribute date airplaneCreateDate; // 生产日期

Relationship Set<Flight> hasFlight inverse Flight::flightAirplane; // 联系执行的航班

Relationship Airlines belongsTo inverse Airlines:: hasAirplane; // 联系所属的航空公司

Relationship Airport locatedAt inverse Airport:: hasAirplane; // 联系所在的机场

Relationship Captain flightCaptain inverse Captain:: flies; // 联系驾驶飞机的机长

}

Interface Airport (key airportName){//机场

Attribute string airportName; // 机场名称

Attribute string airportlocation; // 机场所在地

Attribute date airportbuildTime; // 建造日期

Attribute string airportLevel; // 机场等级

Relationship Set<Airplane> hasAirplane inverse Airport:: locatedAt; // 联系所有的飞机集合

Relationship Set<Flight> inboundFlight inverse Flight::takeOffAirport; // 联系起飞的航班

Relationship Set<Flight> departureFlight inverse Flight::landingAirport; // 联系降落的航班

Relationship Set<Runway> hasRunway inverse Runway::runwayAirport; // 联系拥有的跑道集合

}

将以上数据模型转化为关系模型：（其中加粗下划线表示键码）

Ticket (**ticketID**, class, seatNumber, price, passengerID, flightNumber)

Flight(**flightNumber,** duration, takeOffTime, landingTime, flightDate, airplaneID ,airportName1, airportName2, airlinesName)

Runway(**runwayID**, length, width, airportName)

Airlines(**airlinesName**, abbreviation, numOfPlanes)

Captain(captainName, captainsex, **captainID**, captainPhone, captainbirthdate, airlinesName, airplaneID)

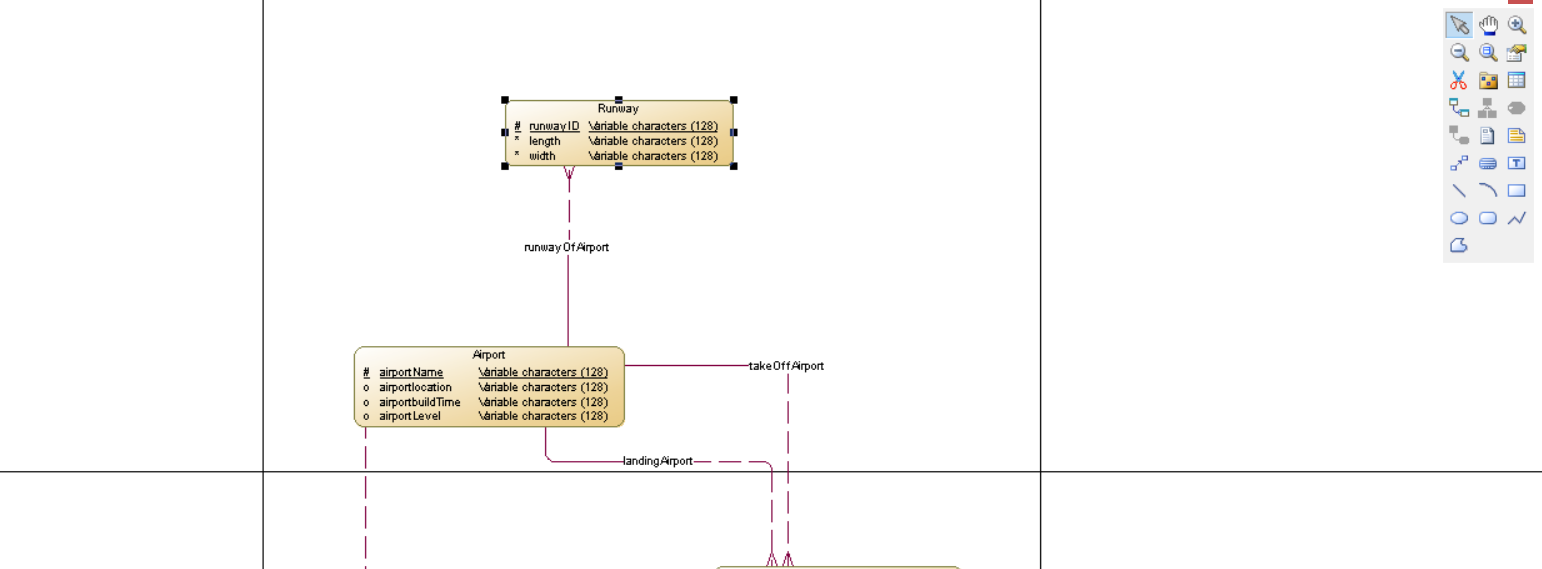
Passenger(passengerName, **passengerID**, passengerIphone, passengerbirthdate, credit)

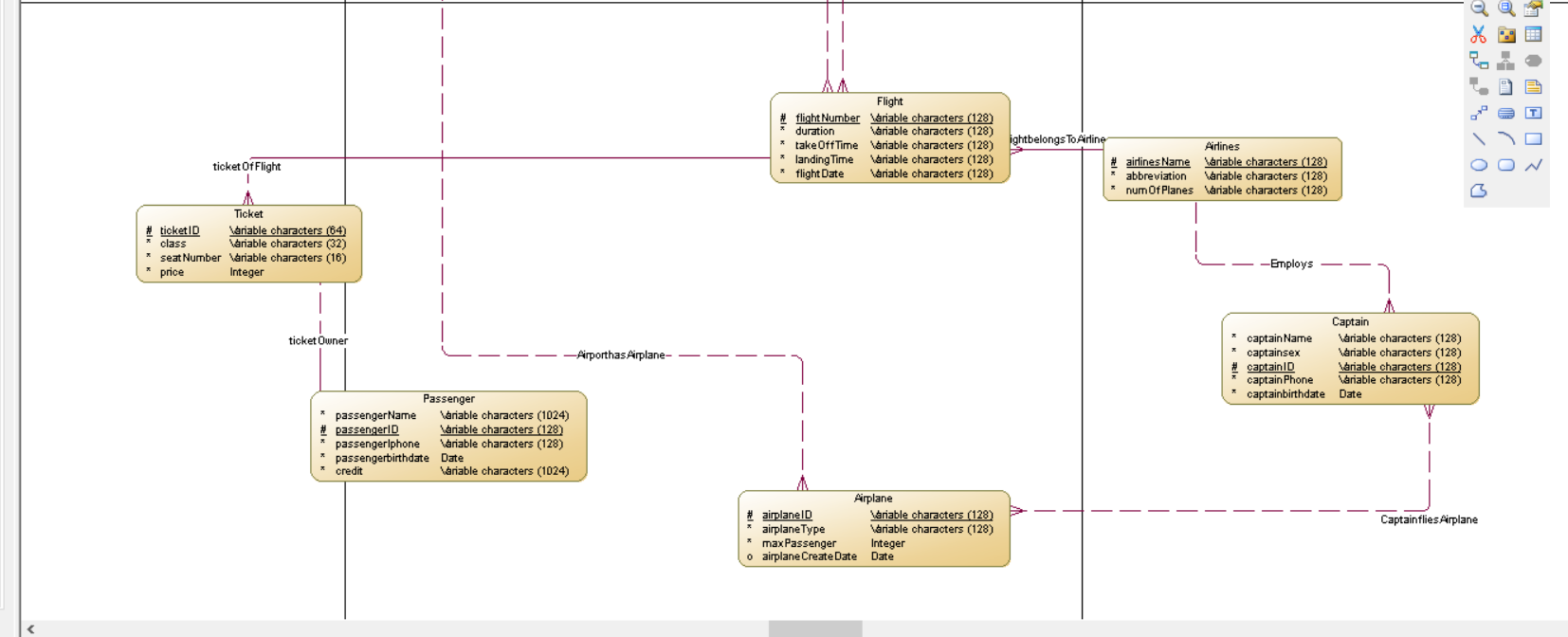
Airplane(**airplaneID**, airplaneType, maxPassenger, airplaneCreateDate, airlinesName, airportName)

Airport(**airportName**, airportlocation, airportbuildTime, airportLevel)

将以上关系模型规范化：由于每一个多值依赖（函数依赖）的左侧均为超键码，所以该关系模式是第四范式。同时也是第一范式、第二范式、第三范式、BC范式。

接着，我们用过ER方法建立数据模型。以下图片是“多情自古伤别离\_参考答案.cdm”文件的截图。





将以上ER图建立的数据模型转化为关系模型，得到：（其中加粗下划线表示键码）

Ticket (**ticketID**, class, seatNumber, price,)

Flight(**flightNumber,** duration, takeOffTime, landingTime, flightDate,)

Runway(**runwayID**, length, width)

Airlines(**airlinesName**, abbreviation, numOfPlanes)

Captain(captainName, captainsex, **captainID**, captainPhone, captainbirthdate)

Passenger(passengerName, **passengerID**, passengerIphone, passengerbirthdate, credit)

Airplane(**airplaneID**, airplaneType, maxPassenger, airplaneCreateDate)

Airport(**airportName**, airportlocation, airportbuildTime, airportLevel)

ticketOwner(**ticketID, passengerID**)

ticketOfFlight(**ticketID,** flightNumber)

AirporthasAirplane(**airplaneID**,airportName)

runwayOfAirport(**runwayID**,airportName)

landingAirport(**flightNumber**,airportName)

takeOffAirport(**flightNumber**,airportName)

FlightbelongsToAirlines (**flightNumber,** airlinesName)

Employs(**captainID,** airlinesName)

CaptainfliesAirplane(**captainID, airplaneID**)

由于每一个多值依赖（函数依赖）的左侧均为超键码，所以该关系模式是第四范式。同时也是第一范式、第二范式、第三范式、BC范式。