

Challenge Oct-2016

Rebooking Passengers from Cancelled Flights

A solution with DT5GL by Jack Jansonius – 04 August 2021

A particularly interesting challenge was delivered by Michael Parish in October 2016.

However, I am surprised that a solution based on the database query language SQL combined with business rules in decision tables has not yet been submitted.

SQL can be seen as a specification of the desired result and can thus be called a pure 4GL (focused on the what, not the how).

In my opinion, it is especially interesting to include SQL less and less within 3GL languages (i.e. embedding SQL in procedural languages) or supplementing it with other 3GL programming constructs (e.g. PL/SQL or Transact-SQL) and more and more within 5GL languages (based on business rules in decision tables).

Therefore, now my solution with DT5GL: 3 decision tables (5GL/DT) and 5 (very simple) database queries (4GL/SQL).

Flight	From	To	Dep	Arr	Capacity	Status
UA123	SFO	SNA	1/1/07 6:00 PM	1/1/07 7:00 PM	5	cancelled
UA456	SFO	SNA	1/1/07 7:00 PM	1/1/07 8:00 PM	2	scheduled
UA789	SFO	SNA	1/1/07 9:00 PM	1/1/07 11:00 PM	2	scheduled
UA1001	SFO	SNA	1/1/07 11:00 PM	1/2/07 5:00 AM	0	scheduled
UA1111	SFO	LAX	1/1/07 11:00 PM	1/2/07 5:00 AM	2	scheduled

Name	Status	Miles	Flight
Jenny	gold	500000	UA123
Harry	gold	100000	UA123
Igor	gold	50000	UA123
Dick	silver	100	UA123
Tom	bronze	10	UA123

RULES

1. Alternate flight must depart from the same place as the cancelled flight
2. Alternate flight must arrive at the same place as the cancelled flight
3. Alternate flight must depart after the cancelled flight
4. There must be room on the alternate flight
5. Passenger status determines who gets allocated first

Results

Jenny is confirmed on UA456 departing SFO at 1/1/2007 19:00:00 arriving SNA at 1/1/2007 20:00:00
Harry is confirmed on UA456 departing SFO at 1/1/2007 19:00:00 arriving SNA at 1/1/2007 20:00:00
Igor is confirmed on UA789 departing SFO at 1/1/2007 21:00:00 arriving SNA at 1/1/2007 23:00:00
Dick is confirmed on UA789 departing SFO at 1/1/2007 21:00:00 arriving SNA at 1/1/2007 23:00:00
Tom could not be rebooked

Implementation of the decision tables in DT5GL:

SQLite_database: "Database/Flight.sqlite"

Table 0: Determine cancelled flight, alternate flight and first passenger to rebook

```
If:
| 0| 1| 2| 3|
'Cancelled flight'          | Y| Y| Y| N|
'Passenger to rebook'      | Y| Y| N| -|
'Alternative flight for cancelled flight' | Y| N| -| -|
Then:
PreAction is Continued     | X| | | |
PreAction is NoRebook     | | X| | |
PreAction is NextFlight   | | | X| |
PreAction is Finished     | | | | X|
# .....
```

Table 1: Determine passenger with highest prio and rebook

```
If:
| 0| 1| 2| 3|
PreAction.getvalue is Continued | Y| Y| Y| N|
'Next passenger to rebook'      | Y| Y| N| -|
'Next passenger has higher prio' | Y| N| -| -|
Then:
Action is Swap                 | X| | | |
Action is NoSwap              | | X| | |
Action is Rebook               | | | X| |
Action is Finished             | | | | X|
# .....
```

rTable 2: Determine prio next passenger

```
If:
| 0| 1| 2| 3| 4|
current_passenger.status = "gold" | Y| N| N| N| N|
cancelled_passenger.status = "gold" | Y| Y| N| N| N|
current_passenger.status = "silver" | -| -| Y| N| N|
cancelled_passenger.status = "silver" | -| -| Y| Y| N|
cancelled_passenger.miles > current_passenger.miles | Y| -| Y| -| Y|
Then:
'Next passenger has higher prio' | X| X| X| X| X|
# .....
```

Proposition: 'Cancelled flight'
Obtain_instance_from_database_view: cancelled_flight

Proposition: 'Alternative flight for cancelled flight'
Obtain_instance_from_database_view: alternative_flight

Proposition: 'Passenger to rebook'
Obtain_instance_from_database_view: cancelled_passenger

Proposition: 'Next passenger to rebook'
Obtain_instance_from_database_view: cancelled_passenger

Attribute: current_passenger.miles Type: Integer
Attribute: cancelled_passenger.miles Type: Integer

```
Database_view: cancelled_flight
With_attributes:
flight, from, to, dep, arr, capacity, status
Query:
  SELECT *
  FROM Flights
  WHERE status = 'cancelled'
  LIMIT 1
End_Query
```

```
Database_view: cancelled_passenger
With_attributes:
name, status, miles, flight, flightstatus
Query:
  SELECT *
  FROM Passenger
  WHERE Flight = '%s'
  LIMIT 1 OFFSET %s
With_arguments: cancelled_flight.flight, cancelled_passenger.auto_index
```

```
Database_view: alternative_flight
With_attributes:
flight, from, to, dep, arr, capacity, status
Query:
  SELECT *
  FROM Flights
  WHERE [From] = '%s' AND
        [To] = '%s' AND
        Dep > '%s' AND
        Capacity > 0 AND
        Status = 'scheduled'
  ORDER BY Dep
  LIMIT 1
With_arguments: cancelled_flight.from, cancelled_flight.to, cancelled_flight.dep
```

```
Initial_database_table: init_current_passenger
Query:
CREATE TEMP TABLE Current_passenger(
  Name          VARCHAR (25),
  Status        VARCHAR (10),
  Miles         INT,
  Flight        VARCHAR (10),
  FlightStatus  VARCHAR (10)
)
End_Query
```

```
Database_view: current_passenger
With_attributes:
name, status, miles, flight, flightstatus
Query:
  SELECT *
  FROM Current_passenger
  LIMIT 1
End_Query
```

GoalAttribute: PreAction
Repeat_until: Finished

Case: Finished

```
Print: "=====  
Print: "No passengers to process anymore."  
Print: "The rebooking service has been finished"  
Print: "=====
```

Case: Continued

```
Print: "#REM# -- print nothing"  
>SQL: "INSERT INTO Current_passenger (Name, Status, Miles, Flight, Flightstatus) "  
<SQL: "VALUES ('%s', '%s', %s, '%s', '%s') " cancelled_passenger.name  
cancelled_passenger.status cancelled_passenger.miles cancelled_passenger.flight  
cancelled_passenger.flightstatus
```

Case: NoRebook

```
Print: "%s => %s could not be rebooked." cancelled_flight.flight  
cancelled_passenger.name  
>SQL: "UPDATE Passenger "  
-SQL: " SET Flightstatus = 'not_rebooked' "  
<SQL: " WHERE Name = '%s' " cancelled_passenger.name
```

Case: NextFlight

```
Print: "No more passengers to rebook for flight %s." cancelled_flight.flight  
>SQL: "UPDATE Flights "  
-SQL: " SET Status = 'completed' "  
<SQL: " WHERE Flight = '%s' " cancelled_flight.flight
```

GoalAttribute: Action
Repeat_until: Rebook, Finished

Case: Finished

```
Print: "#REM# -- print nothing"
```

Case: Swap

```
Print: "#REM# -- print nothing"  
>SQL: "DELETE FROM Current_passenger "  
<SQL: "WHERE Name = '%s' " current_passenger.name  
>SQL: "INSERT INTO Current_passenger (Name, Status, Miles, Flight, Flightstatus) "  
<SQL: "VALUES ('%s', '%s', %s, '%s', '%s') " cancelled_passenger.name  
cancelled_passenger.status cancelled_passenger.miles cancelled_passenger.flight  
cancelled_passenger.flightstatus
```

Case: NoSwap

```
Print: "#REM# -- print nothing"
```

Case: Rebook

```
Print: "%s => %s is confirmed on %s departing %s at %s arriving %s at %s."  
cancelled_flight.flight current_passenger.name alternative_flight.flight  
alternative_flight.from alternative_flight.dep alternative_flight.to  
alternative_flight.arr  
>SQL: "UPDATE Passenger "  
-SQL: " SET Flight = '%s', " alternative_flight.flight  
-SQL: " Flightstatus = 'scheduled' "  
<SQL: " WHERE Name = '%s' " current_passenger.name  
  
>SQL: "UPDATE Flights "  
-SQL: " SET Capacity = Capacity - 1 "  
<SQL: " WHERE Flight = '%s' " alternative_flight.flight  
  
>SQL: "DELETE FROM Current_passenger "  
<SQL: "WHERE Name = '%s' " current_passenger.name
```

```

Initial_database_setup: delete_passengers
Query:
    DELETE FROM Passenger
End_Query

Initial_database_setup: insert_new_passengers
Query:
    INSERT INTO Passenger
    VALUES
    ('Tom', 'bronze', 10, 'UA123', 'cancelled'),
    ('Harry', 'gold', 100000, 'UA123', 'cancelled'),
    ('Igor', 'gold', 50000, 'UA123', 'cancelled'),
    ('Dick', 'silver', 100, 'UA123', 'cancelled'),
    ('Jenny', 'gold', 500000, 'UA123', 'cancelled')
End_Query

Initial_database_setup: delete_flights
Query:
    DELETE FROM Flights
End_Query

Initial_database_setup: insert_new_flights
Query:
    INSERT INTO Flights
    VALUES
    ('UA123', 'SFO', 'SNA', '2007-01-01 18:00', '2007-01-01 19:00', 5, 'cancelled'),
    ('UA456', 'SFO', 'SNA', '2007-01-01 19:00', '2007-01-01 20:00', 2, 'scheduled'),
    ('UA789', 'SFO', 'SNA', '2007-01-01 21:00', '2007-01-01 23:00', 2, 'scheduled'),
    ('UA1001', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
    ('UA1111', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled')
End_Query

```

A first testrun:

```

File to read...: RebookFinal - 1 flight - v3.52.txt
dt5gl-v3.52 rel.19/07/21 - 2021-07-30 13:00:38.
All decision tables (except the rTables) are checked and ok.
SQLite_database: "Database/Flight.sqlite"
Proposition: 'Cancelled flight'
Proposition: 'Alternative flight for cancelled flight'
Proposition: 'Passenger to rebook'
Proposition: 'Next passenger to rebook'
Attribute: current_passenger.miles      Type: Integer
Attribute: cancelled_passenger.miles    Type: Integer
Database_view: cancelled_flight
Database_view: cancelled_passenger
Database_view: alternative_flight
Initial_database_table: init_current_passenger
Database_view: current_passenger
GoalAttribute: PreAction
GoalAttribute: Action
Initial_database_setup: delete_passengers
Initial_database_setup: insert_new_passengers
Initial_database_setup: delete_flights
Initial_database_setup: insert_new_flights

UA123 => Jenny is confirmed on UA456 departing SFO at 2007-01-01 19:00 arriving SNA at 2007-01-01 20:00.
UA123 => Harry is confirmed on UA456 departing SFO at 2007-01-01 19:00 arriving SNA at 2007-01-01 20:00.
UA123 => Igor is confirmed on UA789 departing SFO at 2007-01-01 21:00 arriving SNA at 2007-01-01 23:00.
UA123 => Dick is confirmed on UA789 departing SFO at 2007-01-01 21:00 arriving SNA at 2007-01-01 23:00.
UA123 => Tom could not be rebooked.
No more passengers to rebook for flight UA123.
=====
No passengers to process anymore.
The rebooking service has been finished
=====

```

Testrun variation 1: 1 cancelled flight; 15 passengers

Initial_database_table: insert_new_passengers

Query:

```
INSERT INTO Passenger
VALUES
('Tom', 'bronze', 10, 'UA123', 'cancelled'),
('Harry', 'gold', 100000, 'UA123', 'cancelled'),
('Igor', 'gold', 50000, 'UA123', 'cancelled'),
('Dick', 'silver', 100, 'UA123', 'cancelled'),
('Jenny', 'gold', 500000, 'UA123', 'cancelled'),

('Tomb', 'bronze', 11, 'UA123', 'cancelled'),
('Harryb', 'gold', 100001, 'UA123', 'cancelled'),
('Igorb', 'gold', 50001, 'UA123', 'cancelled'),
('Dickb', 'silver', 101, 'UA123', 'cancelled'),
('Jennyb', 'gold', 500001, 'UA123', 'cancelled'),

('Tomc', 'bronze', 12, 'UA123', 'cancelled'),
('Harryc', 'gold', 100002, 'UA123', 'cancelled'),
('Igorc', 'gold', 50002, 'UA123', 'cancelled'),
('Dickc', 'silver', 102, 'UA123', 'cancelled'),
('Jennyc', 'gold', 500002, 'UA123', 'cancelled')
```

End_Query

Initial_database_table: insert_new_flights

Query:

```
INSERT INTO Flights
VALUES
('UA123', 'SFO', 'SNA', '2007-01-01 18:00', '2007-01-01 19:00', 15, 'cancelled'),
('UA456a', 'SFO', 'SNA', '2007-01-01 19:02', '2007-01-01 20:00', 2, 'scheduled'),
('UA789a', 'SFO', 'SNA', '2007-01-01 21:02', '2007-01-01 23:00', 2, 'scheduled'),
('UA1001a', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
('UA1111a', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled'),

('UA456b', 'SFO', 'SNA', '2007-01-01 19:01', '2007-01-01 20:00', 2, 'scheduled'),
('UA789b', 'SFO', 'SNA', '2007-01-01 21:01', '2007-01-01 23:00', 2, 'scheduled'),
('UA1001b', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
('UA1111b', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled'),

('UA456c', 'SFO', 'SNA', '2007-01-01 19:00', '2007-01-01 20:00', 2, 'scheduled'),
('UA789c', 'SFO', 'SNA', '2007-01-01 21:00', '2007-01-01 23:00', 2, 'scheduled'),
('UA1001c', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
('UA1111c', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled')
```

End_Query

```
UA123 => Jennyc is confirmed on UA456c departing SFO at 2007-01-01 19:00 arriving SNA at 2007-01-01 20:00.
UA123 => Jennyb is confirmed on UA456c departing SFO at 2007-01-01 19:00 arriving SNA at 2007-01-01 20:00.
UA123 => Jenny is confirmed on UA456b departing SFO at 2007-01-01 19:01 arriving SNA at 2007-01-01 20:00.
UA123 => Harryc is confirmed on UA456b departing SFO at 2007-01-01 19:01 arriving SNA at 2007-01-01 20:00.
UA123 => Harryb is confirmed on UA456a departing SFO at 2007-01-01 19:02 arriving SNA at 2007-01-01 20:00.
UA123 => Harry is confirmed on UA456a departing SFO at 2007-01-01 19:02 arriving SNA at 2007-01-01 20:00.
UA123 => Igorc is confirmed on UA789c departing SFO at 2007-01-01 21:00 arriving SNA at 2007-01-01 23:00.
UA123 => Igorb is confirmed on UA789c departing SFO at 2007-01-01 21:00 arriving SNA at 2007-01-01 23:00.
UA123 => Igor is confirmed on UA789b departing SFO at 2007-01-01 21:01 arriving SNA at 2007-01-01 23:00.
UA123 => Dickc is confirmed on UA789b departing SFO at 2007-01-01 21:01 arriving SNA at 2007-01-01 23:00.
UA123 => Dickb is confirmed on UA789a departing SFO at 2007-01-01 21:02 arriving SNA at 2007-01-01 23:00.
UA123 => Dick is confirmed on UA789a departing SFO at 2007-01-01 21:02 arriving SNA at 2007-01-01 23:00.
UA123 => Tom could not be rebooked.
UA123 => Tomb could not be rebooked.
UA123 => Tomc could not be rebooked.
No more passengers to rebook for flight UA123.
=====
No passengers to process anymore.
The rebooking service has been finished
=====
```

Testrun variation 2: 3 cancelled flights; 15 passengers

Initial_database_setup: insert_new_passengers

Query:

```
INSERT INTO Passenger
VALUES
('Tom', 'bronze', 10, 'UA123', 'cancelled'),
('Harry', 'gold', 100000, 'UA123', 'cancelled'),
('Igor', 'gold', 50000, 'UA123', 'cancelled'),
('Dick', 'silver', 100, 'UA123', 'cancelled'),
('Jenny', 'gold', 500000, 'UA123', 'cancelled'),

('Tomb', 'bronze', 11, 'XA123', 'cancelled'),
('Harryb', 'gold', 100001, 'XA123', 'cancelled'),
('Igorb', 'gold', 50001, 'XA123', 'cancelled'),
('Dickb', 'silver', 101, 'XA123', 'cancelled'),
('Jennyb', 'gold', 500001, 'XA123', 'cancelled'),

('Tomc', 'bronze', 12, 'YA123', 'cancelled'),
('Harryc', 'gold', 100002, 'YA123', 'cancelled'),
('Igorc', 'gold', 50002, 'YA123', 'cancelled'),
('Dickc', 'silver', 102, 'YA123', 'cancelled'),
('Jennyc', 'gold', 500002, 'YA123', 'cancelled')
```

End_Query

Initial_database_setup: insert_new_flights

Query:

```
INSERT INTO Flights
VALUES
('UA123', 'SFO', 'SNA', '2007-01-01 18:00', '2007-01-01 19:00', 5, 'cancelled'),
('UA456', 'SFO', 'SNA', '2007-01-01 19:10', '2007-01-01 20:00', 2, 'scheduled'),
('UA789', 'SFO', 'SNA', '2007-01-01 21:10', '2007-01-01 23:00', 2, 'scheduled'),
('UA1001', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
('UA1111', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled'),

('XA123', 'SFO', 'SNA', '2007-01-01 18:00', '2007-01-01 19:00', 5, 'cancelled'),
('UA456B', 'SFO', 'SNA', '2007-01-01 19:09', '2007-01-01 20:00', 2, 'scheduled'),
('UA789B', 'SFO', 'SNA', '2007-01-01 21:09', '2007-01-01 23:00', 2, 'scheduled'),
('UA1001B', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
('UA1111B', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled'),

('YA123', 'SFO', 'SNA', '2007-01-01 18:00', '2007-01-01 19:00', 5, 'cancelled'),
('UA456C', 'SFO', 'SNA', '2007-01-01 19:08', '2007-01-01 20:00', 2, 'scheduled'),
('UA789C', 'SFO', 'SNA', '2007-01-01 21:08', '2007-01-01 23:00', 2, 'scheduled'),
('UA1001C', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
('UA1111C', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled')
```

End_Query

```
UA123 => Jenny is confirmed on UA456C departing SFO at 2007-01-01 19:08 arriving SNA at 2007-01-01 20:00.
UA123 => Harry is confirmed on UA456C departing SFO at 2007-01-01 19:08 arriving SNA at 2007-01-01 20:00.
UA123 => Igor is confirmed on UA456B departing SFO at 2007-01-01 19:09 arriving SNA at 2007-01-01 20:00.
UA123 => Dick is confirmed on UA456B departing SFO at 2007-01-01 19:09 arriving SNA at 2007-01-01 20:00.
UA123 => Tom is confirmed on UA456 departing SFO at 2007-01-01 19:10 arriving SNA at 2007-01-01 20:00.
No more passengers to rebook for flight UA123.
XA123 => Jennyb is confirmed on UA456 departing SFO at 2007-01-01 19:10 arriving SNA at 2007-01-01 20:00.
XA123 => Harryb is confirmed on UA789C departing SFO at 2007-01-01 21:08 arriving SNA at 2007-01-01 23:00.
XA123 => Igorb is confirmed on UA789C departing SFO at 2007-01-01 21:08 arriving SNA at 2007-01-01 23:00.
XA123 => Dickb is confirmed on UA789B departing SFO at 2007-01-01 21:09 arriving SNA at 2007-01-01 23:00.
XA123 => Tomb is confirmed on UA789B departing SFO at 2007-01-01 21:09 arriving SNA at 2007-01-01 23:00.
No more passengers to rebook for flight XA123.
YA123 => Jennyc is confirmed on UA789 departing SFO at 2007-01-01 21:10 arriving SNA at 2007-01-01 23:00.
YA123 => Harryc is confirmed on UA789 departing SFO at 2007-01-01 21:10 arriving SNA at 2007-01-01 23:00.
YA123 => Tomc could not be rebooked.
YA123 => Igorc could not be rebooked.
YA123 => Dickc could not be rebooked.
No more passengers to rebook for flight YA123.
=====
No passengers to process anymore.
The rebooking service has been finished
=====
```

Testrun variation 3: 3 cancelled flights; 15 passengers/v2

Initial_database_setup: insert_new_passengers

Query:

```
INSERT INTO Passenger
VALUES
('Tom', 'bronze', 10, 'UA123', 'cancelled'),
('Harry', 'gold', 100000, 'UA123', 'cancelled'),
('Igor', 'gold', 50000, 'UA123', 'cancelled'),
('Dick', 'silver', 100, 'UA123', 'cancelled'),
('Jenny', 'gold', 500000, 'UA123', 'cancelled'),

('Tomb', 'bronze', 11, 'XA123', 'cancelled'),
('Harryb', 'gold', 100001, 'XA123', 'cancelled'),
('Igorb', 'gold', 50001, 'XA123', 'cancelled'),
('Dickb', 'silver', 101, 'XA123', 'cancelled'),
('Jennyb', 'gold', 500001, 'XA123', 'cancelled'),

('Tomc', 'bronze', 12, 'YA123', 'cancelled'),
('Harryc', 'gold', 100002, 'YA123', 'cancelled'),
('Igorc', 'gold', 50002, 'YA123', 'cancelled'),
('Dickc', 'silver', 102, 'YA123', 'cancelled'),
('Jennyc', 'gold', 500002, 'YA123', 'cancelled')
```

End_Query

Initial_database_setup: insert_new_flights

Query:

```
INSERT INTO Flights
VALUES
('UA123', 'SFO', 'SNA', '2007-01-01 18:00', '2007-01-01 19:00', 5, 'cancelled'),
('UA456', 'SFO', 'SNA', '2007-01-01 19:10', '2007-01-01 20:00', 2, 'scheduled'),
('UA789', 'SFO', 'SNA', '2007-01-01 21:10', '2007-01-01 23:00', 2, 'scheduled'),
('UA1001', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
('UA1111', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled'),

('XA123', 'SFO', 'SNA', '2007-01-01 21:09', '2007-01-01 23:00', 5, 'cancelled'),
('UA456B', 'SFO', 'SNA', '2007-01-01 19:09', '2007-01-01 20:00', 2, 'scheduled'),
('UA789B', 'SFO', 'SNA', '2007-01-01 21:09', '2007-01-01 23:00', 2, 'scheduled'),
('UA1001B', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
('UA1111B', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled'),

('YA123', 'SFO', 'SNA', '2007-01-01 18:00', '2007-01-01 19:00', 5, 'cancelled'),
('UA456C', 'SFO', 'SNA', '2007-01-01 19:08', '2007-01-01 20:00', 2, 'scheduled'),
('UA789C', 'SFO', 'SNA', '2007-01-01 21:08', '2007-01-01 23:00', 2, 'scheduled'),
('UA1001C', 'SFO', 'SNA', '2007-01-01 23:00', '2007-01-02 05:00', 0, 'scheduled'),
('UA1111C', 'SFO', 'LAX', '2007-01-01 23:00', '2007-01-02 05:00', 2, 'scheduled')
```

End_Query

```
UA123 => Jenny is confirmed on UA456C departing SFO at 2007-01-01 19:08 arriving SNA at 2007-01-01 20:00.
UA123 => Harry is confirmed on UA456C departing SFO at 2007-01-01 19:08 arriving SNA at 2007-01-01 20:00.
UA123 => Igor is confirmed on UA456B departing SFO at 2007-01-01 19:09 arriving SNA at 2007-01-01 20:00.
UA123 => Dick is confirmed on UA456B departing SFO at 2007-01-01 19:09 arriving SNA at 2007-01-01 20:00.
UA123 => Tom is confirmed on UA456 departing SFO at 2007-01-01 19:10 arriving SNA at 2007-01-01 20:00.
No more passengers to rebook for flight UA123.
XA123 => Jennyb is confirmed on UA789 departing SFO at 2007-01-01 21:10 arriving SNA at 2007-01-01 23:00.
XA123 => Harryb is confirmed on UA789 departing SFO at 2007-01-01 21:10 arriving SNA at 2007-01-01 23:00.
XA123 => Tomb could not be rebooked.
XA123 => Igorb could not be rebooked.
XA123 => Dickb could not be rebooked.
No more passengers to rebook for flight XA123.
YA123 => Jennyc is confirmed on UA456 departing SFO at 2007-01-01 19:10 arriving SNA at 2007-01-01 20:00.
YA123 => Harryc is confirmed on UA789C departing SFO at 2007-01-01 21:08 arriving SNA at 2007-01-01 23:00.
YA123 => Igorc is confirmed on UA789C departing SFO at 2007-01-01 21:08 arriving SNA at 2007-01-01 23:00.
YA123 => Dickc is confirmed on UA789B departing SFO at 2007-01-01 21:09 arriving SNA at 2007-01-01 23:00.
YA123 => Tomc is confirmed on UA789B departing SFO at 2007-01-01 21:09 arriving SNA at 2007-01-01 23:00.
No more passengers to rebook for flight YA123.
=====
No passengers to process anymore.
The rebooking service has been finished
=====
```


Database situation after Testrun variation 3:

	Flight	From	To	Dep	Arr	Capacity	Status
1	UA123	SFO	SNA	2007-01-01 18:00	2007-01-01 19:00	5	completed
2	UA456	SFO	SNA	2007-01-01 19:10	2007-01-01 20:00	0	scheduled
3	UA789	SFO	SNA	2007-01-01 21:10	2007-01-01 23:00	0	scheduled
4	UA1001	SFO	SNA	2007-01-01 23:00	2007-01-02 05:00	0	scheduled
5	UA1111	SFO	LAX	2007-01-01 23:00	2007-01-02 05:00	2	scheduled
6	XA123	SFO	SNA	2007-01-01 21:09	2007-01-01 23:00	5	completed
7	UA456B	SFO	SNA	2007-01-01 19:09	2007-01-01 20:00	0	scheduled
8	UA789B	SFO	SNA	2007-01-01 21:09	2007-01-01 23:00	0	scheduled
9	UA1001B	SFO	SNA	2007-01-01 23:00	2007-01-02 05:00	0	scheduled
10	UA1111B	SFO	LAX	2007-01-01 23:00	2007-01-02 05:00	2	scheduled
11	YA123	SFO	SNA	2007-01-01 18:00	2007-01-01 19:00	5	completed
12	UA456C	SFO	SNA	2007-01-01 19:08	2007-01-01 20:00	0	scheduled
13	UA789C	SFO	SNA	2007-01-01 21:08	2007-01-01 23:00	0	scheduled
14	UA1001C	SFO	SNA	2007-01-01 23:00	2007-01-02 05:00	0	scheduled
15	UA1111C	SFO	LAX	2007-01-01 23:00	2007-01-02 05:00	2	scheduled

	Name	Status	Miles	Flight	FlightStatus
1	Tom	bronze	10	UA456	scheduled
2	Harry	gold	100000	UA456C	scheduled
3	Igor	gold	50000	UA456B	scheduled
4	Dick	silver	100	UA456B	scheduled
5	Jenny	gold	500000	UA456C	scheduled
6	Tomb	bronze	11	XA123	not_rebooked
7	Harryb	gold	100001	UA789	scheduled
8	Igorb	gold	50001	XA123	not_rebooked
9	Dickb	silver	101	XA123	not_rebooked
10	Jennyb	gold	500001	UA789	scheduled
11	Tomc	bronze	12	UA789B	scheduled
12	Harryc	gold	100002	UA789C	scheduled
13	Igorc	gold	50002	UA789C	scheduled
14	Dickc	silver	102	UA789B	scheduled
15	Jennyc	gold	500002	UA456	scheduled

A commentary on this solution.

To better understand this implementation in dt5gl an explanation of the different constructs of the tool.

First, an explanation of the third decision table:

```
rTable 2: Determine prio next passenger
If:
current_passenger.status = "gold"           | 0| 1| 2| 3| 4|
cancelled_passenger.status = "gold"        | Y| N| N| N| N|
current_passenger.status = "silver"        | -| -| Y| N| N|
cancelled_passenger.status = "silver"      | -| -| Y| Y| N|
cancelled_passenger.miles > current_passenger.miles | Y| -| Y| -| Y|
Then:
'Next passenger has higher prio'          | X| X| X| X| X|
# .....
```

It is always my preference to create a full table first:

```
Table 2: Determine prio next passenger
If:
current_passenger.status = "gold"           | 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|
cancelled_passenger.status = "gold"        | Y| Y| Y| N| N| N| N| N| N| N|
current_passenger.status = "silver"        | -| -| -| -| Y| Y| Y| N| N| N|
cancelled_passenger.status = "silver"      | -| -| -| -| Y| Y| N| Y| N| N|
cancelled_passenger.miles > current_passenger.miles | Y| N| -| -| Y| N| -| -| Y| N|
Then:
'Next passenger has higher prio'          | X| | | X| X| | | X| X| |
# .....
```

and then remove the unneeded columns again.

The order in which the rules are evaluated in a decision table is determined by the order in which the goals (and/or subgoals) are listed in the conclusion section of the table in combination with the columns (or decision rules) that are marked with the letter X.

In the first table:

```
Table 0: Determine cancelled flight, alternate flight and first passenger to rebook
If:
'Cancelled flight'                       | 0| 1| 2| 3|
'Passenger to rebook'                   | Y| Y| N| -|
'Alternative flight for cancelled flight' | Y| N| -| -|
Then:
PreAction is Continued                   | X| | | |
PreAction is NoRebook                    | | X| | |
PreAction is NextFlight                  | | | X| |
PreAction is Finished                    | | | | X|
# .....
```

a value for the goalattribute PreAction will be derived in the order:
Continued, NoRebook, NextFlight, Finished.

Thus, to prove PreAction is Continued, the first column of the table will be evaluated, since this column is marked for this value. If this evaluation fails, the reasoning mechanism will make an attempt for PreAction is NoRebook, and for that the second decision rule in the table will be evaluated, etc.

A successful evaluation of the first decision rule means:

1. There is a flight with status 'cancelled',
2. There is a passenger for this flight, with flight status 'cancelled' and
3. For the flight with status 'cancelled' alternative flights are available, as determined by the 4 rules of the challenge (same departure airport, same arrival airport, departure after the cancelled flight and available seats).

Now the passenger found for the cancelled flight must first be stored in the variable 'Current_passenger', then in the next table for the next goalattribute Action this passenger is compared with any other passengers for the same cancelled flight, after which the passenger with the highest priority is then booked onto the alternative flight.

Thus, if the evaluation of the first decision rule succeeds, the corresponding case instruction of the proven goalattribute for the value Continued will be executed first:

Case: Continued

```
>SQL: "INSERT INTO Current_passenger (Name, Status, Miles, Flight, Flightstatus) "  
<SQL: "VALUES ('%s', %s, %s, '%s', '%s') " cancelled_passenger.name  
cancelled_passenger.status cancelled_passenger.miles cancelled_passenger.flight  
cancelled_passenger.flightstatus
```

This approach could be called a reduced (limited, restricted) bubble sort: only available seats on alternative flights are prioritized for passengers. Why sort 100 passengers by priority if there are only 2 alternative flights available with 2 available seats each?

Thus, the variable 'Current_passenger' always contains the passenger with the highest priority and comes to the surface per available seat on an alternative flight.

For this variable, I created a temporary table in the database tool used (sqlite¹):

```
CREATE TEMP TABLE Current_passenger (  
    Name          VARCHAR (25),  
    Status        VARCHAR (10),  
    Miles         INT,  
    Flight        VARCHAR (10),  
    FlightStatus VARCHAR (10)  
)
```

but possibly I will develop other variables for this in the future.

¹ Of course, any other database tool can be used instead of sqlite; although differences in syntax are always possible.

The program consists of 2 repeatable goalattributes: PreAction and Action:

Table 0: Determine cancelled flight, alternate flight and first passenger to rebook

```

If:
'Cancelled flight'           | 0| 1| 2| 3|
'Passenger to rebook'      | Y| Y| Y| N|
'Alternative flight for cancelled flight' | Y| N| -| -|
Then:
PreAction is Continued     | X| | | |
PreAction is NoRebook     | | X| | |
PreAction is NextFlight   | | | X| |
PreAction is Finished     | | | | X|
# .....
# Repeat_until: Finished

```

Table 1: Determine passenger with highest prio and rebook

```

If:
PreAction.getvalue is Continued | 0| 1| 2| 3|
'Next passenger to rebook'    | Y| Y| Y| N|
'Next passenger has higher prio' | Y| N| -| -|
Then:
Action is Swap                | X| | | |
Action is NoSwap              | | X| | |
Action is Rebook              | | | X| |
Action is Finished            | | | | X|
# .....
# Repeat_until: Rebook, Finished

```

The goalattribute PreAction is repeated until the value Finished is proven, as specified:

```

GoalAttribute: PreAction
Repeat_until: Finished

```

and the next goalattribute Action is repeated until the values Rebook or Finished are proven, because of:

```

GoalAttribute: Action
Repeat_until: Rebook, Finished

```

The second table (for deriving the goalattribute Action) should only be executed if the previous goalattribute has returned the value Continued; hence the first condition in this table. Note, however, that here is formulated:

```

PreAction.getvalue is Continued instead of PreAction is Continued

```

As soon as a goalattribute also appears as a condition in another table, it stops being a goalattribute and becomes an ordinary attribute. To prevent that, ".getvalue" is appended (which, by the way, also bypasses the backward reasoning mechanism if the goalattribute turns out to have no value at all).

An alternative decision table for determining priority.

If the passenger data table is set up more according to the rules of the relational model, the status field would not contain the description of the status, but a status id as a reference to a status table. In this way, new statuses can be added easily:

	Statusid	Status
1	0	Platinum
2	1	Gold
3	2	Silver
4	3	Bronze

and also makes the decision table for determining priority much simpler and more generally applicable:

```
rTable 2: Determine prio next passenger
If:
cancelled_passenger.statusid < current_passenger.statusid      | 0 | 1 |
cancelled_passenger.statusid = current_passenger.statusid      | Y | N |
cancelled_passenger.miles   > current_passenger.miles          | - | Y |
Then:
'Next passenger has higher prio'                                | X | X |
# .....
```

Now the sorting of the passengers is done based on the status id; in addition, if we want to use the status description from the Status table, this must be added to the database view for cancelled_passenger:

```
Database_view: cancelled_passenger
With_attributes:
name, statusid, miles, flight, flightstatus, status
Query:
SELECT Passenger.*, Status.Status
FROM Passenger
INNER JOIN Status on Passenger.Statusid=Status.Statusid
WHERE Flight = '%s'
LIMIT 1 OFFSET %s
With_arguments: cancelled_flight.flight, cancelled_passenger.auto_index
```

The current passenger table (for remembering the highest priority passenger) should also be modified:

```
Initial_database_table: init_current_passenger
Query:
CREATE TEMP TABLE Current_passenger(
  Name          VARCHAR (25),
  Statusid      INT (1),
  Miles         INT,
  Flight        VARCHAR (10),
  FlightStatus  VARCHAR (10),
  Status        VARCHAR (10)
)
End_Query
```

Now the status can be shown in parentheses after the name:

Case: Rebook

```
Print: "%s => %s (%s) is confirmed on %s departing %s at %s arriving %s at %s."
cancelled_flight.flight current_passenger.name current_passenger.status
alternative_flight.flight alternative_flight.from alternative_flight.dep
alternative_flight.to alternative_flight.arr
>SQL: "UPDATE Passenger "
-SQL: "    SET Flight = '%s', "                alternative_flight.flight
-SQL: "        Flightstatus = 'scheduled' "
<SQL: " WHERE Name = '%s' "                current_passenger.name
```

With the following modification to Test run variation 2:

Initial_database_setup: insert_new_passengers

Query:

```
INSERT INTO Passenger
VALUES
('Tom', 3, 10, 'UA123', 'cancelled'),
('Harry', 1, 100000, 'UA123', 'cancelled'),
('Igor', 1, 50000, 'UA123', 'cancelled'),
('Dick', 2, 100, 'UA123', 'cancelled'),
('Jenny', 1, 500000, 'UA123', 'cancelled'),

('Tomb', 3, 11, 'XA123', 'cancelled'),
('Igorb', 1, 50001, 'XA123', 'cancelled'),
('Dickb', 2, 101, 'XA123', 'cancelled'),
('Jennyb', 1, 500001, 'XA123', 'cancelled'),
('Bill', 0, 500001, 'XA123', 'cancelled'),

('Tomc', 3, 12, 'YA123', 'cancelled'),
('Harryc', 1, 100002, 'YA123', 'cancelled'),
('Igorc', 1, 50002, 'YA123', 'cancelled'),
('Dickc', 2, 102, 'YA123', 'cancelled'),
('Jennyc', 1, 500002, 'YA123', 'cancelled')
```

End_Query

is the output then:

```
UA123 => Jenny (Gold) is confirmed on UA456C departing SFO at 2007-01-01 19:08 arriving SNA at 2007-01-01 20:00.
UA123 => Harry (Gold) is confirmed on UA456C departing SFO at 2007-01-01 19:08 arriving SNA at 2007-01-01 20:00.
UA123 => Igor (Gold) is confirmed on UA456B departing SFO at 2007-01-01 19:09 arriving SNA at 2007-01-01 20:00.
UA123 => Dick (Silver) is confirmed on UA456B departing SFO at 2007-01-01 19:09 arriving SNA at 2007-01-01 20:00.
UA123 => Tom (Bronze) is confirmed on UA456 departing SFO at 2007-01-01 19:10 arriving SNA at 2007-01-01 20:00.
No more passengers to rebook for flight UA123.
XA123 => Bill (Platinum) is confirmed on UA456 departing SFO at 2007-01-01 19:10 arriving SNA at 2007-01-01 20:00.
XA123 => Jennyb (Gold) is confirmed on UA789C departing SFO at 2007-01-01 21:08 arriving SNA at 2007-01-01 23:00.
XA123 => Igorb (Gold) is confirmed on UA789C departing SFO at 2007-01-01 21:08 arriving SNA at 2007-01-01 23:00.
XA123 => Dickb (Silver) is confirmed on UA789B departing SFO at 2007-01-01 21:09 arriving SNA at 2007-01-01 23:00.
XA123 => Tomb (Bronze) is confirmed on UA789B departing SFO at 2007-01-01 21:09 arriving SNA at 2007-01-01 23:00.
No more passengers to rebook for flight XA123.
YA123 => Jennyc (Gold) is confirmed on UA789 departing SFO at 2007-01-01 21:10 arriving SNA at 2007-01-01 23:00.
YA123 => Harryc (Gold) is confirmed on UA789 departing SFO at 2007-01-01 21:10 arriving SNA at 2007-01-01 23:00.
YA123 => Tomc could not be rebooked.
YA123 => Igorc could not be rebooked.
YA123 => Dickc could not be rebooked.
No more passengers to rebook for flight YA123.
=====
No passengers to process anymore.
The rebooking service has been finished
=====
```