Technical University of Denmark

Department of Applied Mathematics and Computer Science

TEST exam 2024 **based on** the exam in 2023. (Q5 has been divided into two questions, and all requests about showing the output have been removed.)

02170 Database Systems – Written Examination Part 1

Examination date: 16th May 2023
Course title: Database Systems
Course number: 02170
Aids allowed: All
Exam duration: 3 hours
Weighting: Part 1: 50 %, Part 2: 50 %.
The weighting is only indicative and will ultimately be decided during the evaluation.

This document constitutes Part 1 of the written examination.
Part 2 is a digital multiple choice assignment which should be accessed separately.
The two parts can be solved in any order.

Your answers to the questions in this document should be inserted at the right place in the ***answers.sql*** file also provided in this part of the exam. There must at most be one answer to each question. Remember to write your student ID at the top of the file.

Feel free to write comments to your answers. Comments must be placed in SQL comments, i.e. just after two hyphens (--) without line breaks.

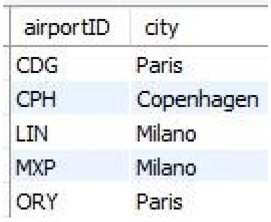
The provided ***airline.sql*** file can be used to create and populate the database instance shown in this document. You are welcome to use it while answering the questions. Note that referential integrity constraints are *deliberately omitted* from *airline.sql*, and they must be added for the database to be consistent.

When you have completed the *answers.sql* file, you should submit it digitally on eksamen.dtu.dk.
[This should only be done at the real exam.]

A database for an airline:

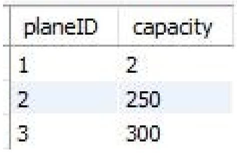
In both parts of this exam a database named *Airline* is considered. The database contains information about flights offered by an airline company named AirDK, passengers on the flights, airports, planes, and customers. Below you find the relation schemas of the database and a relation instance of each of these, and some explanations are given.

Airport(<u>airportID</u>, city)



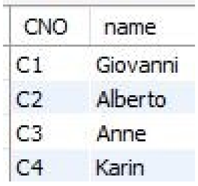
airportID	city
CDG	Paris
CPH	Copenhagen
LIN	Milano
MXP	Milano
ORY	Paris

Plane(<u>planeID</u>, capacity)



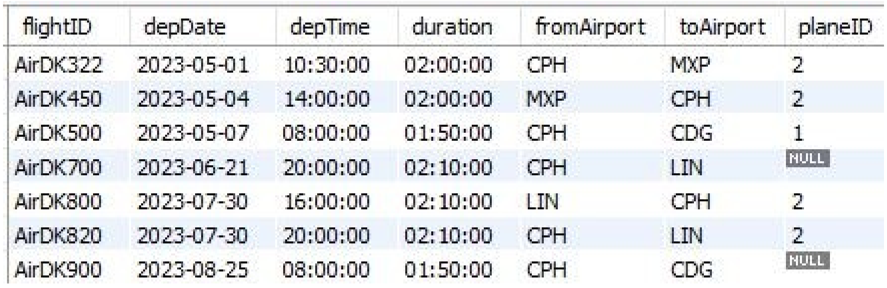
planeID	capacity
1	2
2	250
3	300

Customer(<u>CNO</u>, name)



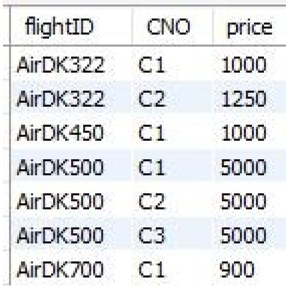
CNO	name
C1	Giovanni
C2	Alberto
C3	Anne
C4	Karin

Flight(<u>flightID</u>, depDate, depTime, duration, fromAirport, toAirport, planeID)



flightID	depDate	depTime	duration	fromAirport	toAirport	planeID
AirDK322	2023-05-01	10:30:00	02:00:00	CPH	MXP	2
AirDK450	2023-05-04	14:00:00	02:00:00	MXP	CPH	2
AirDK500	2023-05-07	08:00:00	01:50:00	CPH	CDG	1
AirDK700	2023-06-21	20:00:00	02:10:00	CPH	LIN	NULL
AirDK800	2023-07-30	16:00:00	02:10:00	LIN	CPH	2
AirDK820	2023-07-30	20:00:00	02:10:00	CPH	LIN	2
AirDK900	2023-08-25	08:00:00	01:50:00	CPH	CDG	NULL

FlightPassengers(<u>flightID</u>, <u>CNO</u>, price)



flightID	CNO	price
AirDK322	C1	1000
AirDK322	C2	1250
AirDK450	C1	1000
AirDK500	C1	5000
AirDK500	C2	5000
AirDK500	C3	5000
AirDK700	C1	900

Airport: Each airport has a unique *airportID* and a *city* in which it is situated.

Plane: Each plane has a unique *planeID* and a *capacity* stating the number of passengers that can be seated in the plane.

Flight: Each flight has a unique *flightID*, a departure date (*depDate*) and a departure time (*depTime*) (in the CET time zone), and a *duration*. It also has a departure airport (*fromAirport*) and a destination airport (*toAirport*). Furthermore, a specific plane (given by its *planeID*) can be allocated to the flight. (Initially no plane is allocated, but a plane must be allocated before departure.)

Customer: Before a customer can be booked on a flight, the customer must be registered with a unique customer number *CNO* and a *name* in the *Customer* table.

FlightPassengers: In this table it is registered which customers are booked as passengers on which flights, and which individual *price* each passenger of a flight has paid for that flight.

Question 1 State an SQL query, which returns a table containing the flightID of those flights that are flying to Milano city.

Question 2 State an SQL query, which returns a table containing the flightID of flights for which the highest price (paid by a passenger) is less than 1200.

Question 3 State an SQL query, which returns a table containing the customer number of each customer who has not been booked on a flight to the CPH airport.

Question 4 State an SQL query, which returns a table containing the flightID and the number of free seats for each flight which goes from the CPH airport and which has got a plane allocated. For the number of free seats, use attribute name *freeseats*.

Hint: It is assumed that each passenger booked on a flight will be assigned exactly one seat on that flight. Furthermore, the airline sometimes overbooks the flight. In such a case the number of free seats will be negative.

Question 5 Define an SQL trigger named *Flight_Before_Insert*, which automatically raises a signal when a new row having the same *fromAirport* and *toAirport* is attempted to be inserted in the *Flight* table.

Question 6 Show an example of an SQL statement which causes this trigger to raise a signal.