Technologies for Information Systems

Part I (10 points)

prof. L. Tanca – February 20th, 2018

Available time: 25 minutes

Last Name _____

First Name _____

Student ID _____ **Signature**_____

1) List and describe the main dimensions of Data Quality

2) Describe the Box-Plot method for displaying the distribution of data, using an example to illustrate it clearly

- **During this part of the exam, students are not allowed to consult books or notes.**
- **Students should answer the theoretical questions using their own words, in order for the teachers to be able to assess their real level of understanding.**

Technologies for Information Systems

Part II (23 points)

prof. L. Tanca – February 20th, 2018

Available Time: 2h 00m

Last Name _____

First Name _____

Student ID _____ **Signature**_____

PoliCoach is a transport company offering coach services in Italy. PoliCoach issues tickets for the travels; tickets are strictly personal, and may comprehend multiple travels. The price of a specific travel may vary depending on the other travels with which it is sold, therefore the total price of a ticket cannot be decomposed into the prices of the component travels.

UniCoach too is a transport company offering coach services, but it operates in France, Spain and Portugal. UniCoach issues personal and group tickets, and each ticket is valid for a single travel. The price of a travel for each person is fixed.

PoliCoach and UniCoach have now merged into a unique company named ***UniPoliCoach***. The UniPoliCoach ownership asks you to integrate the relational databases of the two companies into a unique relational database. You must perform the integration ensuring to lose the least possible amount of information.

The original relational schemas of the two sources are reported below.

PoliCoach

VEHICLE (PlateNumber, Manufacturer, Displacement, NrSeats)
TRAVEL (TravelId, Date, DepartureCity, DestinationCity, LengthMiles, VehiclePlateNumber)
CUSTOMER (CustomerId, SSN, BirthDate, Name) *// The Name attribute contains both first name and last name, separated by a blank space; within the Name attribute value the first name is always composed of a single word.*
TICKET (CustomerId, PurchaseTimestamp, Class, Price)
TICKETTRAVEL (CustomerId, PurchaseTimestamp, TravelId)

UniCoach

COACH (PlateNumber, Brand, NrSeats)
CITY (CityName, Country, NrInhabitants, Seaside) *// Seaside is a boolean attribute that is true if the city is on the sea.*
TRIP (TripId, Date, LengthKm, PricePerPerson, DepartureCity, DestinationCity, CoachPlateNumber)
PASSENGER (SSN, BirthDate, GivenName, Surname, Address)
TICKET (TicketId, PurchaseTimestamp, Class, TripId)
TICKETPASSENGER (TicketId, SSN)

Notes:

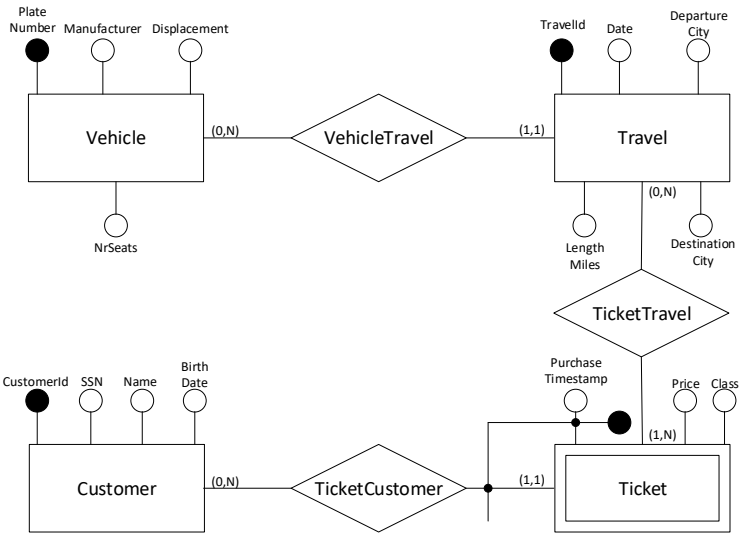
- You can assume that the people in the two data sources are disjoint.
- 1 mile is equal to 1.609 km.

1. **Source schema reverse engineering**. Provide, for each input data source, the reverse engineering from the logical schema to the conceptual model (ER graph). (5 points)
2. **Schema integration**. Design an integrated global conceptual schema (ER graph) for *UniPoliCoach* capturing all the data coming from both *PoliCoach* and *UniCoach*, and provide the corresponding global logical schema. In more detail, follow these steps:
    a. *Related concept identification and conflict analysis and resolution*. Write a table as shown in the exercise sessions, using the following columns: "PoliCoach concept", "UniCoach concept", "Conflict", "Solution". (3.5 points)
    b. *Integrated conceptual schema* (ER graph). (4 points)
    c. *Conceptual to logical translation of the integrated schema*. (2.5 points)
3. **Query answering and mapping definition**. Consider the query Q: "Find city and country of destination of the first-class tickets for travels with length greater than 300 km".
    a. *Query formulation*. Consider query Q posed on the logical schema of *UniPoliCoach* and write it in SQL. (1.5 points)
    b. *Mapping definition*. Write the GAV mappings between the schema of *UniPoliCoach* and the two sources using SQL. Write the mappings only for the tables used to answer query Q**.** (4 points)
    c. *Query rewriting*. Show the rewriting of Q on the two data sources using SQL. (2.5 points)

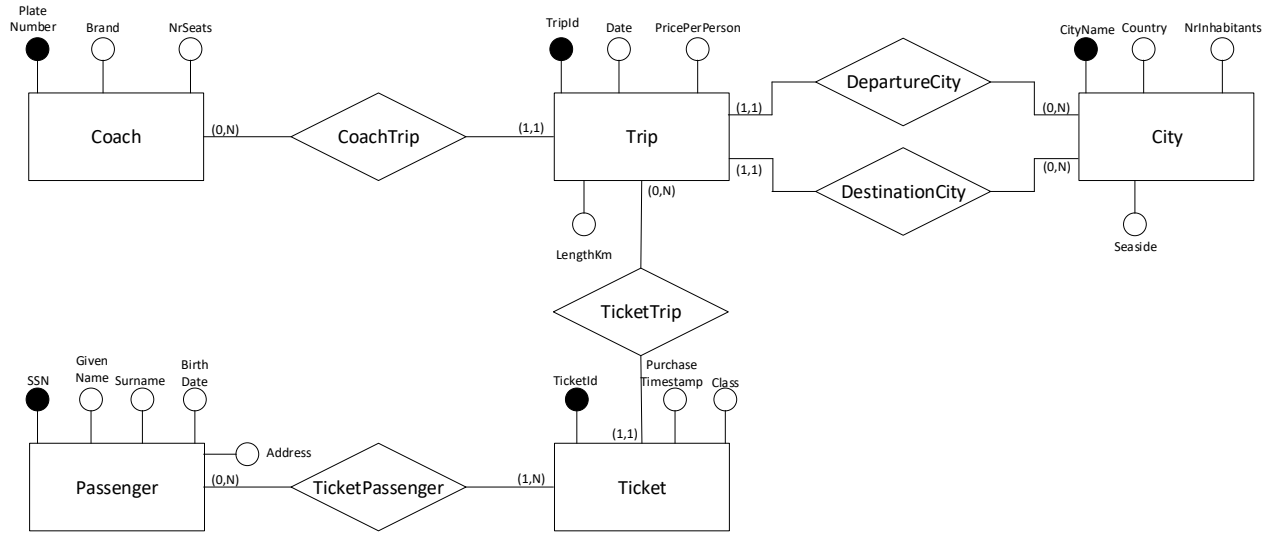
SOLUTION

1. Source schema reverse engineering

PoliCoach



UniCoach

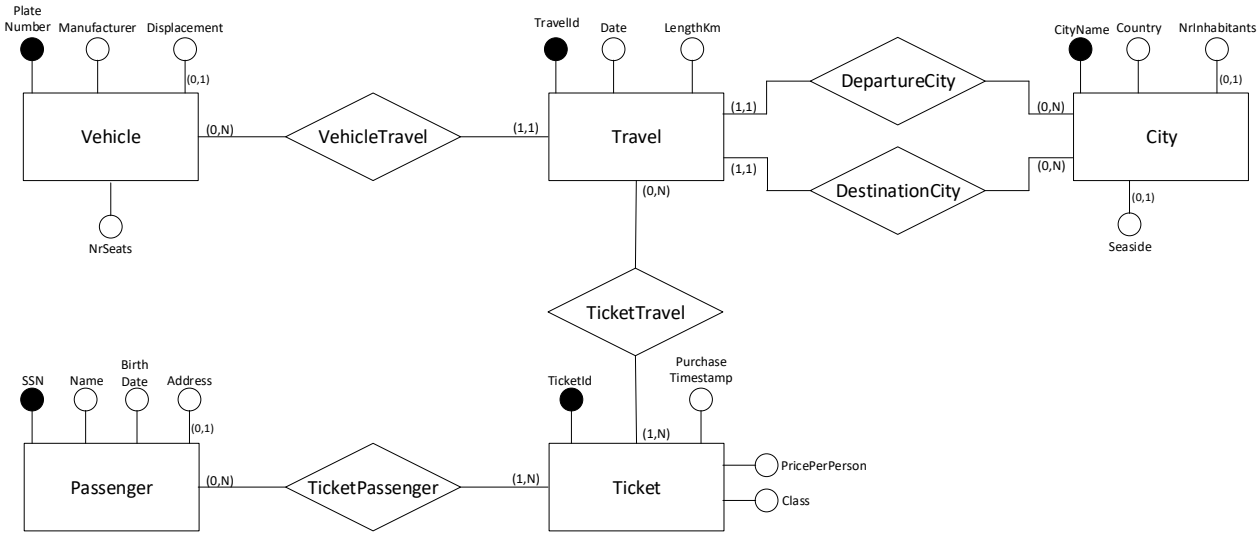


2. Schema integration

2a) Related concept identification + conflict analysis and resolution

PoliCoach	UniCoach	Conflict	Solution
Vehicle	Coach	Name conflicts	
		- Entity name	Vehicle
		- Manufacturer → Brand	Manufacturer
Travel	Trip	Name conflicts	
		- Entity name	Travel
		- TravelId → TripId	TravelId
		Data semantics conflicts	
		- LengthMiles → LengthKm	LengthKm
		Structure conflicts	
		- Departure and destination cities are represented as attributes → City is an entity	City is an entity
Customer	Passenger	Name conflicts	
		- Entity name	Passenger
		Key conflict	
		- CustomerId → SSN	SSN
		Structure conflicts	
		- Given name and surname in a unique attribute Name → Two distinct attributes	Just one attribute Name
Ticket	Ticket	Key conflict	
		- CustomerId+ PurchaseTimestamp → TicketId	TicketId
		Cardinality conflicts	
		- A ticket is associated with just one customer → A ticket may be associated with multiple passengers	A ticket may be associated with multiple passengers
		- A ticket may be associated with multiple travels → A ticket is associated with just one trip	A ticket may be associated with multiple travels
		Structure conflicts	
		- The price (per person) of a travel varies for each ticket → The price per person of a trip is fixed	The price (per person) of a travel varies for each ticket

2b) Global conceptual schema



2c) Conceptual to logical translation

VEHICLE (<u>PlateNumber</u>, Manufacturer, NrSeats, Displacement*)
CITY (<u>CityName</u>, Country, NrInhabitants*, Seaside*)
TRAVEL (<u>TravelId</u>, Date, LengthKm, DepartureCity, DestinationCity, VehiclePlateNumber)
PASSENGER (<u>SSN</u>, Name, Birthdate, Address*)
TICKET (<u>TicketId</u>, PurchaseTimestamp, Class, PricePerPerson)
TICKETPASSENGER (<u>TicketId</u>, <u>SSN</u>)
TICKETTRAVEL (<u>TicketId</u>, <u>TravelId</u>)

3. Query answering and mapping definition

3a) Query formulation

Find city and country of destination of the first-class tickets for travels with length greater than 300 km.

<u>NOTE</u>:
During the exam also the following alternate equivalent text of the query was provided to the students:
Find city and country of destination of the travels with length greater than 300 Km associated with first-class tickets.

SELECT C.CityName, C.Country
FROM City **AS** C, Travel **AS** Tr, TicketTravel **AS** TT, Ticket **AS** Ti
WHERE C.CityName=Tr.DestinationCity **AND** Tr.TravelId=TT.TravelId **AND** TT.TicketId=Ti.TicketId **AND**
Ti.Class=1 **AND** Tr.LengthKm>300

3b) GAV mapping definition

```
CREATE VIEW UniPoliCoach.City (CityName, Country, NrInhabitants, Seaside) AS (
        SELECT DepartureCity, 'Italy', null, null
        FROM PoliCoach.Travel

        UNION

        SELECT DestinationCity, 'Italy', null, null
        FROM PoliCoach.Travel

        UNION

        SELECT CityName, Country, NrInhabitants, Seaside
        FROM UniCoach.City
)


CREATE VIEW UniPoliCoach.Travel (TravelId, Date, LengthKm, DepartureCity, DestinationCity,
VehiclePlateNumber) AS (
        SELECT KeyGenTravel(TravelId, 'PoliCoach'), Date, DepartureCity, DestinationCity,
                LengthMiles*1.609, VehiclePlateNumber
        FROM PoliCoach.Travel

        UNION

        SELECT KeyGenTravel(TripId, 'UniCoach'), Date, DepartureCity, DestinationCity, LengthKm,
                CoachPlateNumber
        FROM UniCoach.Trip
)


CREATE VIEW UniPoliCoach.Ticket (TicketId, PurchaseTimestamp, Class, PricePerPerson) AS (
        SELECT KeyGenTicket(CustomerId||PurchaseTimestamp, 'PoliCoach'), PurchaseTimestamp, Class,
                Price
        FROM PoliCoach.Ticket

        UNION

        SELECT KeyGenTicket(Ti.TicketId, 'UniCoach'), Ti.PurchaseTimestamp, Ti.Class, Tr.PricePerPerson
        FROM UniCoach.Ticket AS Ti, UniCoach.Trip AS Tr
        WHERE Ti.TripId=Tr.TripId
)


CREATE VIEW UniPoliCoach.TicketTravel (TicketId, TravelId) AS (
        SELECT KeyGenTicket(CustomerId||PurchaseTimestamp, 'PoliCoach'), KeyGenTravel(TravelId,
                'PoliCoach')
        FROM PoliCoach.TicketTravel
```

```
    UNION

    SELECT KeyGenTicket(TicketId, 'UniCoach'), KeyGenTravel(TripId, 'UniCoach')
    FROM UniCoach.Ticket
)
```

3c) Query rewriting

Find city and country of destination of the first-class tickets for travels with length greater than 300 km.

```
SELECT Tr.DestinationCity, 'Italy'
FROM Travel AS Tr, TicketTravel AS TT, Ticket AS Ti
WHERE Tr.TravelId=TT.TravelId AND TT.CustomerId=Ti.CustomerId AND
    TT.PurchaseTimestamp=Ti.PurchaseTimestamp AND Ti.Class=1 AND Tr.LengthMiles*1.609>300

UNION

SELECT C.CityName, C.Country
FROM City AS C, Trip AS Tr, Ticket AS Ti
WHERE C.CityName=Tr.DestinationCity AND Tr.TripId=Ti.TripId AND Ti.Class=1 AND Tr.LengthKm>300
```