SWEN304 2021

Database System Engineering

Exam Revision (with Answers)

1. Query Processing and Optimisation

Suppose the following relational database schema is used by *InterCity Coachlines* to manage information on their bus transportation network.

S = {

Stop ({StopName, City, Location}, {StopName}),

Line ({LineId, LineName}, {LineId}),

Service ({LineId, ServiceNo, From, To}, {LineId, ServiceNo}),

Destination ({LineId, ServiceNo, StopName, ArrTime, DepTime},

{LineId, ServiceNo, StopName })

}

I = {

Service [LineId] $\subseteq$ Line [LineId],

Service [From] $\subseteq$ Stop [StopName],

Service [To] $\subseteq$ Stop [StopName],

Destination [(LineId, ServiceNo)] $\subseteq$ Service [(LineId, ServiceNo)],

Destination [StopName] $\subseteq$ Stop [StopName]

}

Not Null Constraints = {

Null (Service, From) = Not,

Null (Service, To) = Not,

}

InterCity Coachlines runs multiple bus lines. Each bus line is served by one or more services. On each bus line, the services are identifiable by their numbers (*ServiceNo*), but services on different bus lines may have the same service number.

InterCity Coachlines uses many bus stops all across New Zealand. Each service has two bus termini (*From, To*). Each service has a scheduled route consisting of a sequence of destinations. Each destination is a bus stop where a service arrives at a given time and then departs. The two bus termini are also considered as destinations: For the initial stop (*From*) of a service the arrival time (*ArrTime*) is set to NULL, while for the terminal stop (*To*) of a service the departure time (*DepTime*) is set to NULL.

a) [6 marks] Suppose you want to know the arrival times of services on the GreatSights line that go from Auckland Central to Wellington Parliament. To answer this question you can issue the following SQL query against the database:

SELECT ArrTime

FROM Line NATURAL JOIN Service NATURAL JOIN Destination

WHERE LineName = 'Great Sights' AND From = 'Auckland Central'
AND To = 'Wellington Parliament' AND StopName = 'Tongariro';

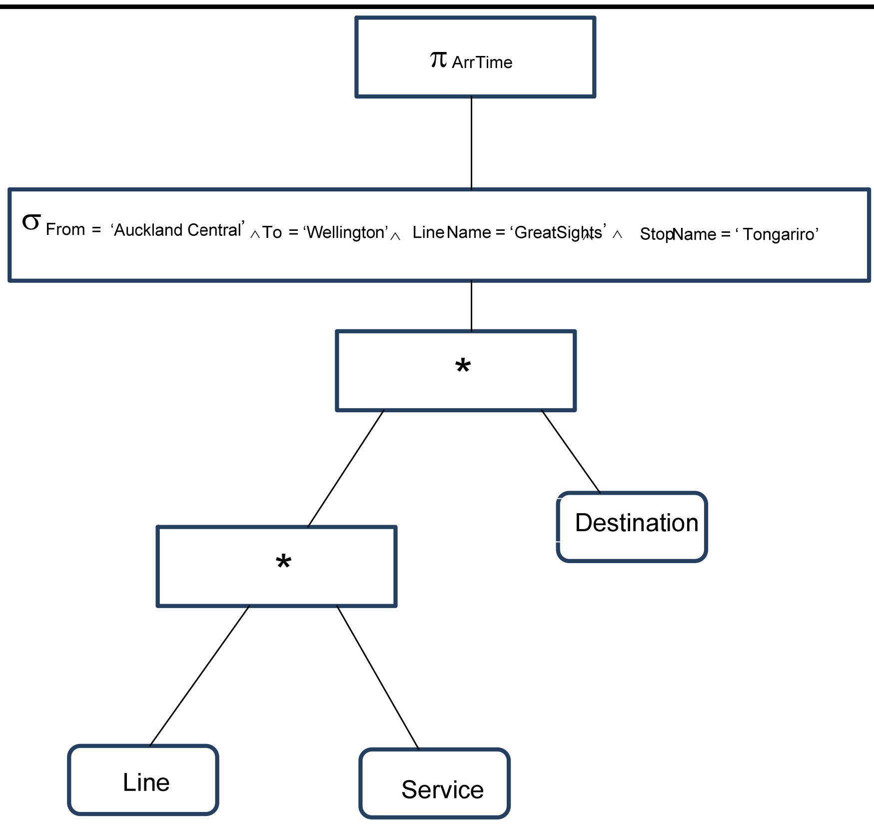
Transfer the above SQL query in relational algebra.

ANSWER

$\pi_{ArrTime}\ \sigma_{LineName = 'Great\ Sights' \wedge From = 'Auckland\ Central' \wedge To = 'Wellington\ Parliament' \wedge StopName = 'Tongariro'}$
(Destination *(Line* Service))

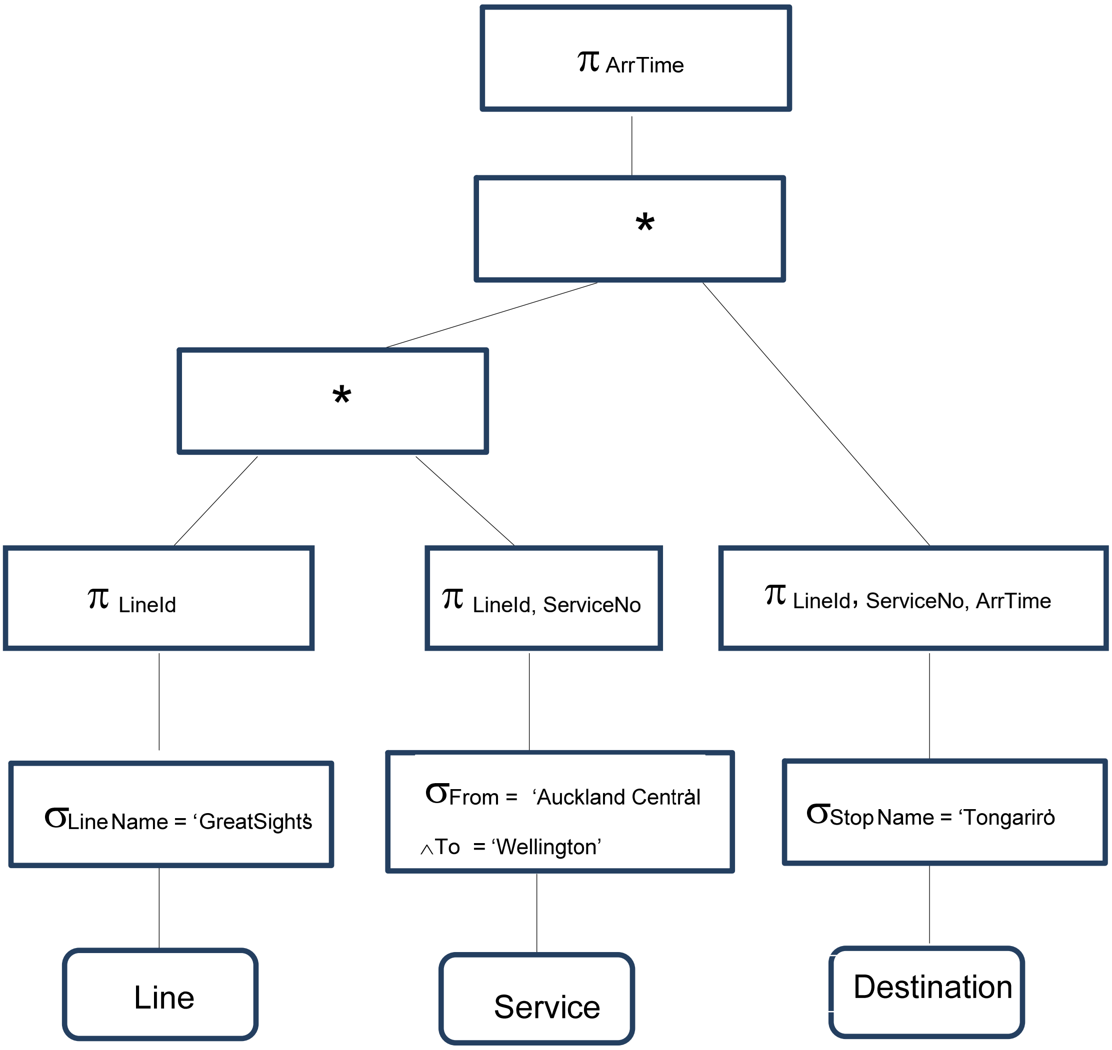
b) [6 marks] Draw a query tree for your relational algebra query in part **a)**

ANSWER



c) **[9 marks]** Optimise the query tree in part **b)** using heuristic rules for algebraic query optimisation as discussed in our lectures. Explain which heuristic rules you can apply. Draw the optimised query tree.

ANSWER



Rules applied:

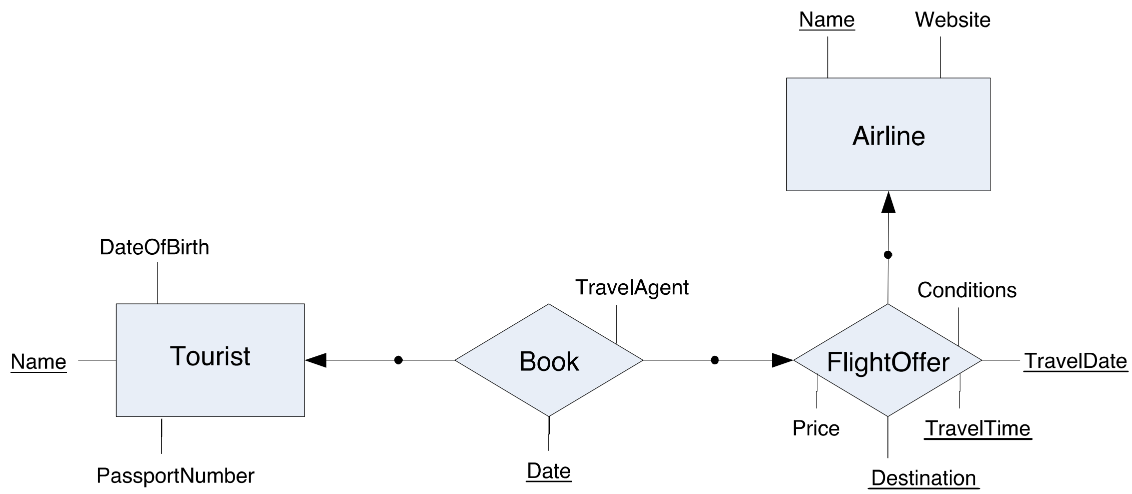
Apply Move select operations down the tree (Rule 6)

Keeping in intermediate relations only the attributes needed by subsequent operations by applying project (π) operations as early as possible (Rule 7)

Note that we can also change the order of JOINS so that the very restrictive select operation could be applied as early as possible (Rule 9). Because we assume there is only one line named GreatSights, JOIN starts with the smallest relation. Hence, we can keep the order of JOINs as it is.

2. EER to RDM

Consider the extended ER diagram below. Transform the corresponding extended ER schema into a relational database schema.



List all the relation schemas in your relational database schema. For each relation schema, list all attributes, primary key, Not Null constraints, and the foreign keys.

ANSWER

EER Schema: (optional)
Level 0:
Tourist=({Name, DateOfBirth, PassportNumber}, {Name})
Airline=(Name, Website}, {Name})

Level 1:
FlightOffer=({Airline}, {Conditions, TravelDate, TravelTime, Destination, Price}, {Airline, TravelDate, TravelTime, Destination})

Level 2:
Book=({FlightOffer, Tourist}, {TravelAgent, Date}, {FlightOffer, Tourist, Date})

Relational Database Schema:
S = {

Tourist ({Name, DateOfBirth, PassportNumber}, {Name}),

Airline ({Name, Website}, {Name}),

FlightOffer ({AirlineName, Destination, TravelDate, TravelTime, Price, Conditions}, {AirlineName, Destination, TravelDate, TravelTime}),

Book ({TouristName, AirlineName, Destination, TravelDate, TravelTime, Date, TravelAgent}, {TouristName, AirlineName, Destination, TravelDate, TravelTime, Date})

}

I = {

FlightOffer [AirlineName] $\subseteq$ Airline [Name],

Book [TouristName] $\subseteq$ Tourist [Name],

Book [(AirlineName, Destination, TravelDate, TravelTime)] $\subseteq$

FlightOffer [(AirlineName, Destination, TravelDate, TravelTime)]

}

Not Null Constraints = {

Null (Tourist, PassportNumber) = Not

Null (Airline, Website) = Not,

}

3. Database Normalisation

Consider the following set of attributes used for capturing the data about entries into Photography Competitions:

Entry = {

EntrantName, EntrantEmail, Competition, NumberOfImages, Fees

}

with the following set of functional dependencies:

F = {

EntrantEmail → EntrantName

Competition+NumberOfImages → Fees

EntrantEmail+Competition → NumberOfImages

}

a) Determine all minimal keys of *Entry* with respect to *F*. Justify your answer.

ANSWER

{EntrantEmail, Competition} is a superkey as it implies all other attributes in Entry.

It is a minimal key as none of its proper subsets is a superkey.

It is the only minimal key as any other superkey must contain the attributes EntrantEmail and Competition.

b) Decide whether *Entry* is in Third Normal Form (3NF) with respect to *F*. Justify your answer. If not, determine a lossless Third Normal Form decomposition.

ANSWER

It is not in 3NF. The first two FDs violate the condition for 3NF, i.e. neither LHS a super key nor RHS a prime attribute.

We use the synthesis algorithm to determine a Third Normal Form decomposition.

We obtain one subset from each of the three FDs:

R1={EntrantEmail, EntrantName} with the projected set of FDs

F1={EntrantEmail → EntrantName},

R2={Competition, NumberOfImages, Fees} with the projected set of FDs

F2={Competition+NumberOfImages → Fees}, and

R3={EntrantEmail, Competition, NumberOfImages} with the projected set of FDs

F3={EntrantEmail+Competition → NumberOfImages}

Because the minimal key is in R3, there is no need to add a relation schema which contains the minimal key: R4={EntrantEmail, Competition} with F4={}.

So we decompose Entry into R1, R2 and R3. Each of them is in 3NF with respect to the respective set of FDs.

c) Decide whether *Entry* in Boyce-Codd Normal Form (BCNF) with respect to *F*. Justify your answer. If not, determine a lossless Boyce-Codd Normal Form decomposition. State whether the decomposition is dependency-preserving.

ANSWER

Entry is not in BCNF as the first two FDs violate the condition, i.e., LHS not a super key. Also it was not in 3NF, so it cannot be in BCNF.

The decomposition into 3NF computed before is in BCNF. One possible answer is to point this out and argue that each of the subsets R1, R2, R3 is in BCNF with respect to the respective set of FDs.

Alternatively, the decomposition algorithm can be applied. Using the FD EntrantEmail → EntrantName one obtains the subsets S1={EntrantEmail, EntrantName} and S2={EntrantEmail, Competition, NumberOfImages, Fees}, with F1={EntrantEmail → EntrantName}, and F2={Competition+NumberOfImages → Fees, EntrantEmail+Competition → NumberOfImages }

S1 is in BCNF but S2 is not due to FD Competition+NumberOfImages → Fees. Hence S2 is further decomposed alone FD Competition+NumberOfImages → Fees into S21={Competition, EntraintEmail, NumberOfImages } and S22={Competition, NumberOfImages,Fees}, with F21 = { EntrantEmail+Competition → NumberOfImages } and F22 = { Competition+NumberOfImages → Fees }

S1, S21 and S22 all are in BCNF with respect to their projected sets of FDs. Note that S1, S21 and S22 are just R1, R2 and R3 from before. So we get the same decomposition again.

All the three FDs are preserved after the decomposition.

4. Extended Entity Relationship Model

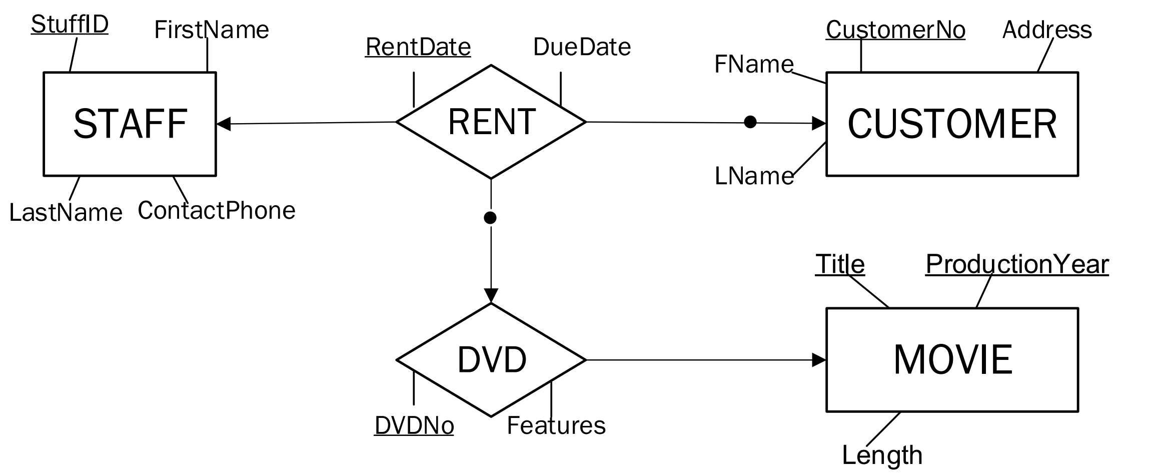
Suppose Wellington's most popular DVD rental shop (Rent'n'Watch) hires you as a data architect to design a new DVD rental database for them. They provide you with the following list of requirements:

- For customers, it is required to store their customer number, first name, last name, and address.
- For staff members, it is required to store their staff id, first name, last name, and contact phone number.
- Rent'n'Watch rents out DVDs containing movies. Each DVD contains one movie. For a movie there may be many DVD copies.
- For each movie, it is required to store its title, production year, and length.
- Each DVD has its unique DVD number and some features.
- When a customer hires a DVD from a certain staff member, the hiring date and due date will be recorded.

Draw an EER diagram of the new database for Rent'n'Watch. Show the entity types and relationship types, and their attributes on the diagram, with entity type keys underlined, and key components marked explicitly (a dot on the edge of a key component).

***Note**: Use the notation and conventions for drawing Extended ER diagrams that we introduced in the lectures. If you prefer to use another notation that differs from the one we used in lectures, clearly define all differences.*

ANSWER



CUSTOMER({CustomerNo, Fname, Lname, Address}, {CustomerNo})

STAFF({StaffID, FirstName, LastName, ContactPhone}, {StaffID})

MOVIE({Title, ProductionYear, Length}, {Title, ProductionYear})

DVD({DVDNo, Features}, {MOVIE}, { DVDNo})

RENT({CUSTOMER, STAFF, DVD}, {RentDate, DueDate}, {DVD, RentDate})