Queen Mary
University of London

First Degrees in Engineering by Course Units

May 2013 14:30 - 17:00

ECS740 Databases Duration: 2 hours 30 minutes

YOU ARE NOT PERMITTED TO READ THE CONTENTS OF THIS QUESTION PAPER UNTIL INSTRUCTED TO DO SO BY AN INVIGILATOR.

Answer ALL Questions

CALCULATORS ARE NOT PERMITTED IN THIS EXAMINATION.

COMPLETE ALL ROUGH WORKINGS IN THE ANSWER BOOK AND CROSS THROUGH ANY WORK WHICH IS NOT TO BE ASSESSED.

IMPORTANT NOTE:
THE ACADEMIC REGULATIONS STATE THAT POSSESSION OF UNAUTHORISED MATERIAL AT ANY TIME WHEN A STUDENT IS UNDER EXAMINATION CONDITIONS IS AN ASSESSMENT OFFENCE AND CAN LEAD TO EXPULSION FROM QMUL.

PLEASE CHECK NOW TO ENSURE YOU DO NOT HAVE ANY NOTES, MOBILE PHONES OR UNATHORISED ELECTRONIC DEVICES ON YOUR PERSON. IF YOU HAVE ANY THEN PLEASE RAISE YOUR HAND AND GIVE THEM TO AN INVIGILATOR IMMEDIATELY. PLEASE BE AWARE THAT IF YOU ARE FOUND TO HAVE HIDDEN UNAUTHORISED MATERIAL ELSEWHERE, INCLUDING TOILETS AND CLOAKROOMS IT WILL BE TREATED AS BEING FOUND IN YOUR POSSESSION. UNAUTHORISED MATERIAL FOUND ON YOUR MOBILE PHONE OR OTHER ELECTRONIC DEVICE WILL BE CONSIDERED THE SAME AS BEING IN POSSESSION OF PAPER NOTES. MOBILE PHONES CAUSING A DISRUPTION IS ALSO AN ASSESSMENT OFFENCE.

EXAM PAPERS CANNOT BE REMOVED FROM THE EXAM ROOM.

Examiners:
Thomas Roelleke
Tony Stockman

Question 1 (Database History & Design)

a) Describe the three main concepts of the ERM. Give an example for each concept, refer to the notion of type and set, and relate the concepts to object-oriented modelling.

[5 marks]

SOLUTION:

Entities, Relationships and Attributes.

OO: objects, classes, class hierarchy (inheritance), attributes, methods.

An entity corresponds to a real world object (e.g. a student, a university).

An entity is a type and set. For example: type student, set of students.

OO: An entity corresponds to a class, a type, and to an object.

A relationship is a link (an association) between two entities (objects).

For example: peter.EnrolledAt(qmul): entity-based (set of instances).

Student.EnrolledAt(University): type-based.

OO: A relationship corresponds to a method (an attribute method).

An attribute is a feature characterising an entity (an object).

For example: Student.Name: attribute (same in ERM and OO).

b) Describe the principal mapping of the ERM into the relational model. Thereby, explain how the cardinality of relationships affects the mapping.

[5 marks]

SOLUTION:

For each entity and for each relationship, a relational table is created.

For 1:1 relationships, the relationship can be migrated in either side/entity.

For 1:N relationships (e.g. One lecturer teaches many lectures), migrate into the N side: Lecture(Id, Date, Room, LecturerId).

For N:M relationships (N students register for M modules), no migration; separate relation required.

c) What were the main milestones in DB history?

[5 marks]

SOLUTION:

60s: hierarchical model (IBM DB system; IMS)

70s: relational model, Codd paper

80s: relational database management systems become widely available (Oracle, IBM DB2, Sybase). Main features: SQL and transaction management.

90s: object-oriented DB technology / systems; became part of RDBMs.

Today:

- object-oriented (object-relational) data model
- datawarehousing (business intelligence)
- unstructured and semi-structured data (XML, information retrieval)
- NoSQL, triplet stores (RDF)

d) Provide a basic ERM diagram for the university scenario with students, courses, and modules. Students register for courses, and students sit exams of modules. There are first sits and resits. The exam marks have to be recorded, and the database system shall keep track of who entered and updated the exam marks. Specify the relational model, and underline the attributes that form the primary keys, and list the foreign keys.

[5 marks]

SOLUTION:

Diagram: Entities: student, course, module, exam.

Relationships: studentRegistersForCourse, studentSitsExam

Relational model: a table per entity, and a table per relationship

Tables for entities:

student(Id, Name, ...)

course(Id, Title, ...)

module(Id, Title, ...)

e.g.: module("ECS740", "Databases", ...)

exam(Id, ModuleId, Location, Date)

# ModuleId models the n:1 relationship between Exam and Module.

Tables for relationships:

registers(StudentId, CourseId, ...)

sits(StudentId, ExamId, Mark, Sit{FirstSit,Resit,Retake}, EnteredBy, UpdatedBy)

Foreign keys in entity tables: exam.ModuleId

Foreign keys relationship tables: registers.StudentId, registers.CourseId, sits.StudentId, sits.ExamId

e) List five of the main database applications. Describe the nature of each application briefly.

[5 marks]

SOLUTION:

See slide in first lecture.

Six DB applications presented in the lecture:

1. Traditional: Banks, Libraries
2. Multimedia databases
3. Geographic information systems
4. Datawarehouse / Business Intelligence
5. Real-time and active DB technology
6. Web (e-commerce, internet banking)

Question 2 (SQL & Relational Model)

Given the following relational DB schema.

```
Plane(PLID, Name, DateOfReg)

Airport(AID, Name, Country, City)

Passenger(PID, FName, LName, Nationality, DOB, PassportNumber)

Flight(FlightNumber, PlaneID, Date, DepAirport, DestAirport, DepTime, Delay, Status)

# Status: is one of "scheduled", "delayed", "cancelled", "arrived", "baggage in hall".

WeatherForecast(Date, City, Source, Text)

# Source: is one of "BBC", "Met Office", "MeteoGroup".
```

a) Identify and specify the referential integrity constraints inherent in the above schema. Specify the create table statement for “Flight”.

[5 marks]

SOLUTION:

The attributes DepAirport and DestAirport in relation “Flight” are foreign keys.

The airports have to exist in relation “Airport”.

```
CREATE TABLE Flight (
FlightNumber varchar(5),
DepAirport varchar(5),
DestAirport varchar(5),
...,
foreign key ( DepAirport ) references Airport ( AID ),
foreign key ( DestAirport ) references Airport ( AID )
foreign key ( PlaneId ) references Plane ( PID )
);
```

b) SQL Query: Find all flights going from Stansted to Dortmund that are delayed.

[5 marks]

SOLUTION:

```
SELECT *
FROM Flight, Airport Airport1, Airport Airport2
WHERE DepAirport = Airport1.AID
AND Airport1.Name = 'Stansted'
AND DestAirport = Airport2.AID
AND Airport2.Name = 'Dortmund'
AND Status = 'delayed';
```

c) SQL Query: Find all airports with at least one flight to a destination (city) in France. Use a nested SQL query.

[5 marks]

SOLUTION:

```
SELECT Airport.Name
FROM Airport A1
WHERE EXISTS
(SELECT Flight.FlightNumber
FROM Flight, Airport A2
WHERE DepAirport = A1.AID
AND DestAirport = A2.AID
AND A2.Country = 'France'
);
```

d) Schema refinement and normalisation.

i) One requirement is to list airports where a weather warning is "active", which means flights are likely to be delayed or diverted. Which extension of the schema do you propose to satisfy this requirement?

ii) Note that the schema of Airport contains the attributes City and Country. What do you propose regarding the normalisation of the schema?

[5 marks]

SOLUTION:

i) Add two tables:

Entity table: WeatherForecast(ID, Date, Source, Description, ...)

Key of WeatherForecast: (Date, Source); is unique if each source issues at most one forecast per day.

For simplicity, WeatherForecast.ID.

Relationship table: WeatherWarning(AirportID, WeatherForecastID)

Regarding Flight.Status related to a weather forecast, one could consider a query to check the actual Flight.Status against the existence of a weather warning.

ii) Assuming unique city names, city implies country, that is: the country is a function of city: country(city).

Therefore, (City, Country) should be in a separate table, and for the Airport, it is sufficient to store the City.

e) Explain the translation of the SQL query "SELECT ... FROM ... WHERE" to an expression of the relational algebra.

[5 marks]

SOLUTION:

```
SELECT <attributes> FROM <source1>, <source2>, ... WHERE <condition>;
```

Translation:

Step 1: Join the sources; the join can consider the join conditions in “condition”.

Step 2: Select the tuples based on ¡condition¿

Step 3: Project on the target attributes

Algebra:

```
Project ( Select ( Product (Product (source1, source2),...), <condition>),
<attributes> )
```

Question 3 (Enhanced ERM)

a) Describe the main mechanisms that the Extended Entity Relationship (EER) model provides that are not available in the Entity Relationship (ER) model.

[5 marks]

SOLUTION:

Superclasses/subclasses: entities having superclass/subclass relationships with other entities.

E.g. : Entity “student” is subclass of entity “person”.

Inheritance: entities of subclasses can inherit attributes of their superclasses. E.g. : person(Name,...). Then if “student” is a sub-type, student.Name is from “person”.

Categories: subclass entities which have multiple possible superclasses can be categorised according to which of the available superclasses they belong to.

E.g. an “owner” can be a “person” or a “company”.

Different from shared subclasses where subclass entity inherits from several superclasses.

E.g. a “teaching assistant” inherits from “student” and from “employee”.

b) Design a movie database in the basic ERM (do not use concepts of the enhanced ERM). The movie database comprises movies, directors, casts, actors, plots (movie descriptions), and reception (how the movie was received). For each director and actor, the DB maintains a pointer to a web site describing the person. Particular to this movie DB is that we store also the main reviews (e.g. reviews from rotten tomatoes or empire magazine). A review has a rating: 1-5 stars. The database shall support analytical queries such as “show the average rating of Woody Allen movies”.

(i) Give a textual specification of the ERM.

(ii) Show an ERM diagram that illustrates some of the main components of the ERM.

(iii) Show a more refined diagram that includes cardinalities and the key attributes for strong entities.

[10 marks]

SOLUTION:

(i)

Entities:

movie(MovieId, ...)

actor(ActorId, ..., HomePage)

director(DirectorId, ..., HomePage)

# HomePage: one page per person (actor and director)

plot(PlotId, Text)

reception(ReceptionId, Text)

review(ReviewId, Source, Text, Rating[1-5 stars])

# Entity review has a relationship to "reviewer" (n:1).

# The mapping of the conceptual model will migrate the ReviewerId. # review(ReviewId, ReviewerId, Source, Text, Rating[1-5 stars])

# Source: is rotten tomatoes, or empire magazine

reviewer(ReviewerId, ...)

Relationships:

cast(MovieId, ActorId)

directorOfMovie(DirectorId, MovieId)

reviewOfMovie(ReviewId, MovieId)

authorOfReview(ReviewerId, ReviewId)

# Assumptions:

# The rating is part of the review; one could also model the rating

# as an attribute of the this relationship.

etc

(ii) Level-0 ERM (first-cut); with entities and relationships as in i).

(iii) Strong entities: entities with a key (set of attributes that identifies the entity).

The conceptual model shown in i) contains artificial ID attributes to achieve strong entities. In theory, the conceptual model should only contain natural attributes (e.g. a movie has a title; the MovieId is artificial).

Cardinalities:

cast: N:M: a movie has several actors; actor plays in several movies.

directorOfMovie: 1:N: a director can direct several movies; a movie has one director.

reviewOfMovie: N:1: several reviews for one movie (a review is for exactly one movie)

authorOfReview: 1:N: a review has one author.

c) Improve the basic ERM by applying concepts of the enhanced ERM.

(i) There are several entities in the movie database that are specialisations of "person". Show the E-ERM. Also, generalise entities such as "plot", "review" and "reception" by introducing the entity “document”.

(ii) Map the E-ERM to the relational schema. List the three options to map the E-ERM concepts to relations in the relational model; explain each option briefly.

[10 marks]

SOLUTION:

i) director is a person

actor is a person

reviewer is a person

review is a document

plot is a document

reception is a document

etc

Relational Model:

person(PersonId, Name, ...)

actor(PersonId, ...): inherits Name from person

director(PersonId, ...)

document(DocId, Text, Author, ...)

review(DocId, Rating, ReviewerId): inherits Text from document; see below for ReviewerId.

plot(DocId): inherits Text from document

reception(DocId): inherits Text from document

# Regarding “document.Author”: for a review, the reviewerId could be # modelled as the author. For a plot, the author is often not explicitly named.

ii) Three options:

- Specialisation, class/subclass relationship

o Multiple relation: a relation per subclass: relation contains superclass and subclass attributes

o Single relation: one relation: concatenation of all subclass attributes

- Shared subclass: multiple inheritance: subclass relation contains only the subclass-specific attributes

- Categories: surrogate key in superclass; is foreign key pointing to subclass entry.

Question 4 (Transaction Management)

a) List the three problems that may occur for concurrent transactions. Explain each problem briefly.

[5 marks]

SOLUTION:

1. The lost update problem: It occurs if two transactions update a variable depending on its value. If the first transaction updates the value, but the second transaction has read the value before it was updated. While T1 (transaction 1) reads the value of an item, the value of that item is changed by T2 (transaction 2).

2. The uncommitted dependency problem: It occurs when one transaction updates a variable and the other transaction reads in the updated value. Unfortunately, it reads in the value before the first transaction was committed, which means that it could be rolled back, reversing the update. The uncommitted dependency problem occurs when T1 is allowed to use the result of another transaction (T2) before T2 has committed the changes

3. The inconsistent analysis problem: If one transaction (T2) reads in several items from the database, while some of the values are updated by another transaction (T1)

b) Draw the precedence graph for the following schedule. Let T1, T2 and T3 be three transactions.

...

[5 marks]

c) Transaction Properties: Explain the notion of ACID.

(i) What does ACID stand for? Explain each concept.

(ii) Is ACID important for banking systems? Why or why not?

(iii) Is ACID important for web-based services (e.g. bbc news pages) where users browse data? Why or why not?

[5 marks]

SOLUTION:

(i) ACID: Atomicity: transaction is indivisble unit

Consistency: transaction starts and ends in consistent DB status

Isolation: independent transactions

Durabiliy: permanent changes

(ii) Very important for banking systems: transfer between bank account systems

(iii) For web-browsing, not important, unless we talk about the back-end management where several journalists update the underlying databases.

d) Serialisable Schedules:

(i) When is a non-serial schedule called serilisable?

Turn over

(ii) When are two schedules in conflict?

[5 marks]

SOLUTION:

(i) A non-serial schedule is called serialisable if there is a way in which the transactions can be executed concurrently without interfering with one another and thereby produce a database state that could be produced by a serial execution.

(ii) Two transactions T1 and T2 are in conflict if an operation in T1 and an operation in T2 force a temporal order between the transactions.

e) Two-phase locking.

(i) Explain the overall idea of 2-phase locking.
(ii) What are the two main phases referred to?
(iii) Explain what does 2-phase locking ensures.
(iv) What are the disadvantages of 2-phase locking?

[5 marks]

SOLUTION:

(i) 2-phase locking: each transaction locks a resource before accessing it.

(ii) growing phase (locking) and shrinking phase (unlocking)

(iii) 2-phase locking ensures serialisability.

(iv) Disadvantages: deadlock might occur; not very efficient.

End of Paper