School of Computing and Information Systems

INFO90002

Database Systems and Information Modelling

Practice Exam 2 SOLUTIONS

Semester 2 2022

Reading Time: 15 minutes

Writing Time: 120 minutes (2 Hours)

This exam has 8 pages including this page

ATTEMPT ALL QUESTIONS IN ANY ORDER

Authorised Materials:

- **None**

While you are undertaking this assessment, you MUST NOT

- make use of any messaging or communication technology
- record, screenshot, stream, upload or in any known format duplicate this document
- record, screenshot, stream, upload or in any known format duplicate your solutions
- make use of any world wide web or internet based resources such as wikipedia, github, stackoverflow, google, Weichat or any known search engine / messaging services
- act in a manner that could be regarded as providing assistance to a student who is undertaking this assessment or in the future will be undertaking this assessment
- seek assistance from any other student who is undertaking this assessment or in the future will be undertaking this assessment

Instructions to Students:

- This exam is in 8 sections. Attempt all questions in all sections.
- We recommend using pencil and paper for modelling questions to save time
- The total for this exam is 100 marks representing 50% of your final assessment
- Attempt **all** questions which are of unequal marks value
- **PLEASE DO NOT USE RED font colour**

IMPORTANT

- **Answer questions in the allocated space for answers.**
- **The official exam language is English. Sections of the submission in languages other than English will NOT be assessed and will be marked as 0.**
- Before submitting your solution document, check that the diagram(s) is/are readable. It is your responsibility to ensure your answers are readable and make sense to the marker.

The work you submit **must be based on your own knowledge and skills** and without the assistance of any other person. You MUST NOT directly copy work that you have not authored (e.g. slide notes, websites, other student's study notes).

Q1. ER Modelling (30 Marks)

TeaTime

TeaTime is a platform for tea lovers. To use the system, users must install the TeaTime app on their phone and set up a user profile that contains: a login name (their email address), password (encrypted and stored as a string of 64 characters), short screen name e.g. "DonkeyKong64" (which other users will see), profile photo, and a “My Cuppa” entry - their favourite tea type, chosen from a list of standard drink names such as “Peppermint Tea” and “Black Tea with Honey”. We plan to have up to 2 million customers (end users) accessing the system.

While a user’s phone is switched on, the TeaTime app sends the phone’s current location to the server once per minute. We store these, so that the system knows where a given user is “now”, as well as the history of where the user has been.

All locations in this system are recorded as a pair of numbers representing latitude and longitude. Latitudes are between -90 and 90 degrees (south pole to north pole) while longitudes are between -180 and 180 degrees (west or east of the prime meridian in Greenwich). We will use a precision of 4 decimal places, which is about 11 metres at the equator. For example, the Doug McDonell building at UniMelb is at latitude -37.7989, longitude 144.9627.

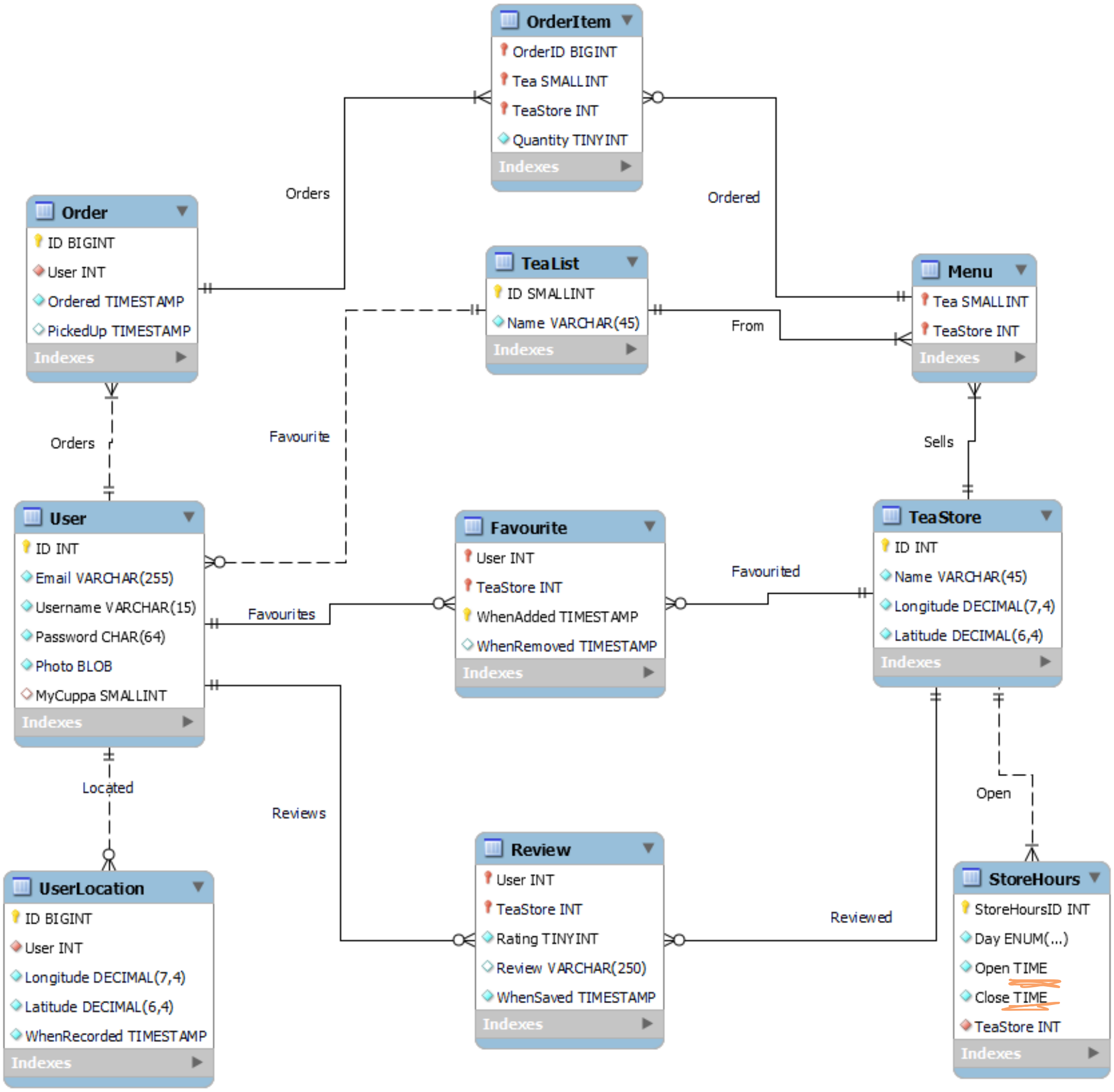
The names, locations and opening hours of about 600 tea stores are available via the app. Users can browse stores either in an alphabetical list or via a map. When viewing a store's profile, users can see reviews and photos uploaded by other users, and the store's average rating. The store's opening hours are recorded as an opening and closing time for each day of the week (for example, a café might open Mondays 8am to 6pm, Tuesdays 9:30am to 7pm, and so on). Each café stores a menu of the teas it sells: these must be drawn from our standard list mentioned above.

Users can rate stores. A rating consists of a whole number between 0 and 10, along with an optional piece of text (up to about 30 words). A given user can only rate a given tea store once. Users can mark particular stores as favourites. These can be viewed in a list. Users can later “unfavourite” the tea store if they wish, and yet later “favourite” it again. We keep a history of these favourites and unfavourites. Customers can use the app to order drinks from a tea store. To do this a user first selects a tea store, then chooses how many of each tea type(s) they want from the store's menu. We keep track of when orders are placed and when customers later pick up the order.

Q.1. Draw a physical model in Crow’s Foot notation for this case study. Be sure to write down any assumptions you make.

(30 marks)

SOLUTION:


OrderItem
OrderID BIGINT
Tea SMALLINT
TeaStore INT
Quantity TINYINT
Indexes
Orders
Ordered
Order
ID BIGINT
User INT
Ordered TIMESTAMP
PickedUp TIMESTAMP
Indexes
TeaList
ID SMALLINT
Name VARCHAR(45)
Indexes
From
Menu
Tea SMALLINT
TeaStore INT
Indexes
Orders
Favourite
Sells
User
ID INT
Email VARCHAR(255)
Username VARCHAR(15)
Password CHAR(64)
Photo BLOB
MyCuppa SMALLINT
Indexes
Favourites
Favourite
User INT
TeaStore INT
WhenAdded TIMESTAMP
WhenRemoved TIMESTAMP
Indexes
Favourited
TeaStore
ID INT
Name VARCHAR(45)
Longitude DECIMAL(7,4)
Latitude DECIMAL(6,4)
Indexes
Located
Reviews
Open
UserLocation
ID BIGINT
User INT
Longitude DECIMAL(7,4)
Latitude DECIMAL(6,4)
WhenRecorded TIMESTAMP
Indexes
Review
User INT
TeaStore INT
Rating TINYINT
Review VARCHAR(250)
WhenSaved TIMESTAMP
Indexes
Reviewed
StoreHours
StoreHoursID INT
Day ENUM(...)
Open TIME
Close TIME
TeaStore INT
Indexes

Q2. SQL (20 marks)

Given the schema in Figure 2, write a single SQL statement to correctly answer each of the following questions (2A – 2D). DO NOT USE VIEWS to answer questions.

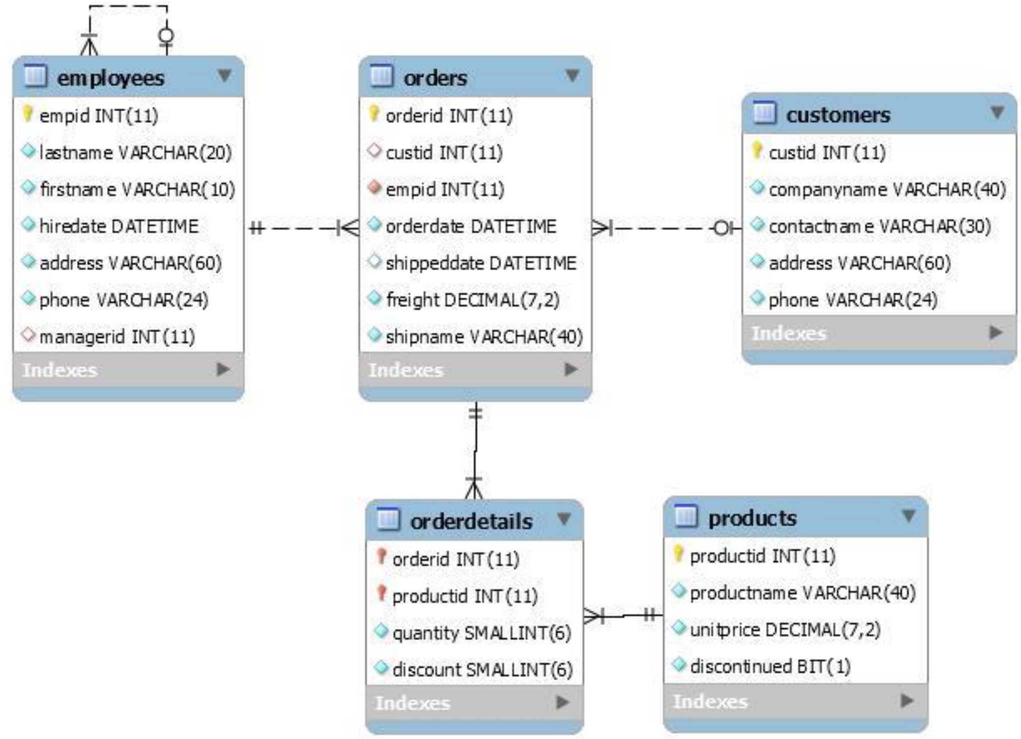


Figure 1 Company Schema

Q.2A. Write a query that returns customers (company names) and the details of their orders (orderid and orderdate), including customers who placed no orders.

(3 marks)

```
SELECT C.companyname, O.orderid, O.orderdate
FROM Customers C
LEFT OUTER JOIN Orders O
ON O.custid = C.custid;
```

Q.2B. Write a query that returns the first name and last name of employees whose manager was hired prior to 01/01/2002.

(4 marks)

```
SELECT E.firstname, E.lastname
FROM Employees E
INNER JOIN Employees MNGR
ON E.managerid = MNGR.empid
WHERE MNGR.hiredate < '2002-01-01';
```

Q.2C. Write a query that returns customers (customer ID) whose company name is ‘Google’, and for each customer return the total number of orders and total quantities for all products that were not discontinued (‘1’ means discontinued, ‘0’ not discontinued).

(5 marks)

```
SELECT C.custid, COUNT(O.orderid) AS numorders, SUM(OD.quantity) AS
totalqty
FROM Customers C
INNER JOIN Orders O
ON O.custid = C.custid
INNER JOIN OrderDetails OD
ON OD.orderid = O.orderid
INNER JOIN Products P
ON OD.productid = P.productid
WHERE C.company = 'Google' AND P.discontinued = '0'
GROUP BY C.custid;
```

Q.2D. Write a query that returns the ID and company name of customers who placed orders in 2007 but not in 2008.

(8 marks)

```
SELECT custid, companyname
FROM Customers
WHERE custid IN
  (SELECT custid
   FROM Orders
   WHERE orderdate >= '2007-01-01' AND orderdate < '2008-01-01')
   AND custid NOT IN
  (SELECT custid
   FROM Orders
   WHERE orderdate >= '2008-01-01' AND orderdate < '2009-01-01');

--     BETTER option

SELECT DISTINCT Customers.custid, companyname
FROM Customers
INNER JOIN ORDERS O1
ON Customers.custid = O1.custID
WHERE YEAR(orderdate) = 2007
AND Customers.custid NOT IN
      (SELECT DISTINCT custid
       FROM ORDERS O2
       WHERE YEAR(orderdate) = 2008);
```

Q3. Normalisation (16 marks)

A car rental company keeps its records in a spreadsheet which looks like a table below. Note a car cannot be rented for part of the day. CostPerDay is determined on the day of rent by some business rules.

RegNo	Make	Model	DateRented	DateReturned	CostPerDay	CustID	CustName	Mobile
2LB 8NG	Toyota	Camry	1/2/2024	5/2/2024	80	2453	Jay Lee	0411199422
2LB 8NG	Toyota	Camry	6/2/2024	16/2/2024	80	7812	Lou Stone	0466121260
2LA 4TX	Hyundai	i30	4/2/2024	11/2/2024	70	3357	Jason Goh	0431441187

Represent the table as a relation, identify candidate key.

Produce 1NF, 2NF and 3NF.

Legend: PK – underline, FK – italics, PFK – underline+italics

Car (**RegNo**, Make, Model (**DateRented**, DateReturned, CostPerDay, CustID, CustName, Mobile))

1NF

Car (**Regno**, Make, Model)

Rental (**DateRented**, DateReturned, CostPerDay, CustID, CustName, Mobile, ***RegNo***)

2NF

Already 2NF because there are no partial dependencies here

3NF

Transitive Dependencies: CustID -> CustName, Mobile

Customer (**CustID**, CustName, Mobile)

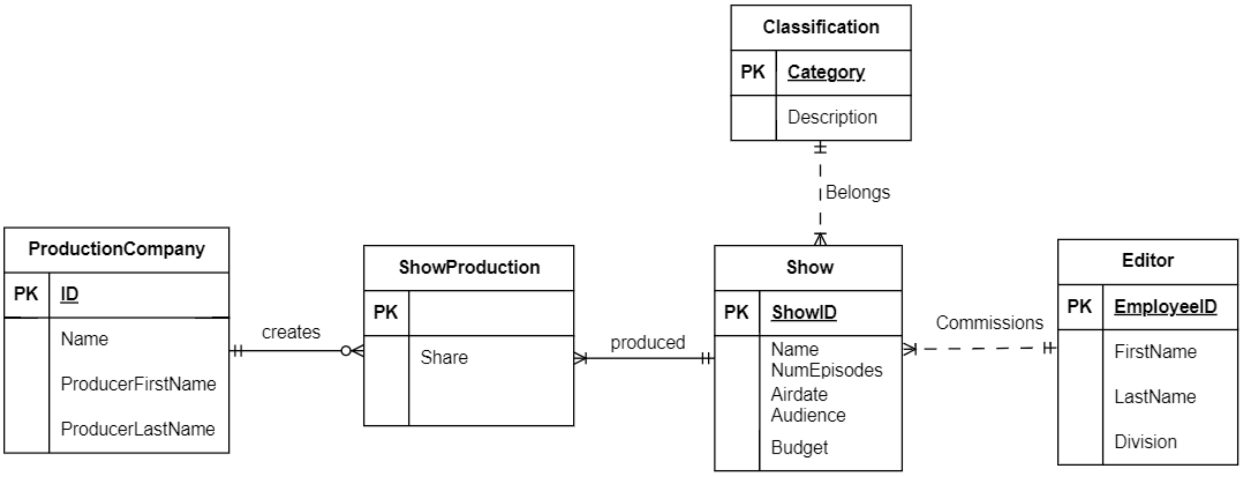
Car (**Regno**, Make, Model)

Rental (**DateRented**, DateReturned, CostPerDay, *CustID*, ***RegNo***)

Q4. SQL DDL (10 marks)

The British Broadcasting Corporation (BBC) has a number of commissioning editors (e.g. Martin Davidson) who have cast a number of landmark shows ("Life on Earth", "A History of Britain") from independent production companies such as names like Hartswood films (Sherlock, Coupling) with producer Sue Vertue. Sometimes more than one production company is involved in the production of a series. Every Show is given a name (e.g. "Sherlock"), a number of episodes (between 6 and 24), a budget (up to several hundred thousand pounds) and an estimated air date for the first episode. The Producers must also nominate who is the target audience (Audiences are graded by age and income demographic) and classify the show (e.g. Documentary, Drama, Comedy, Light Entertainment, Game Show, Reality, News and Current Affairs, Special Event).

Q4. Write the SQL DDL for the following conceptual model. You can use whichever data types you think are appropriate for a MySQL relational database. You need to use NOT NULL to specify mandatory participation.



```
CREATE TABLE Editor
(Employee_ID   INT,
 Firstname VARCHAR(20) NOT NULL,
 Lastname VARCHAR(25) NOT NULL,
 Division  VARCHAR (40),
 PRIMARY KEY (Employee_ID));


CREATE TABLE ProductionCompany
(ID SMALLINT,
 Name VARCHAR(30),
 ProducerFirstName VARCHAR(20) NOT NULL,
 ProducerLastName VARCHAR(25) NOT NULL,
 PRIMARY KEY (ID));


CREATE TABLE Classification
(Category    CHAR(3),
 Description VARCHAR(60),
 PRIMARY KEY (Category));


CREATE TABLE Show
(ShowID SMALLINT,
 ShowName VARCHAR(40),
 Episodes TINYINT,
 Budget DECIMAL(8,2),
 Airdate  DATE,
 Audience   VARCHAR(30),
 Category CHAR(3) NOT NULL,
 Editor INT NOT NULL,
 PRIMARY KEY (ShowID),
 FOREIGN KEY (Editor) REFERENCES Editor(Employee_ID),
 FOREIGN KEY (Category) REFERENCES Classification(Category));
```

```
CREATE TABLE ShowProduction
(ShowID SMALLINT,
 ProdCompanyID SMALLINT,
 SHARE TINYINT,
 PRIMARY KEY (ShowID, ProdCompanyID),
 FOREIGN KEY (ShowID) REFERENCES Show(ShowID),
 FOREIGN KEY (ProdCompanyID)
           REFERENCES ProductionCompany(ID))
```

Q5. Data Warehousing (10 marks)

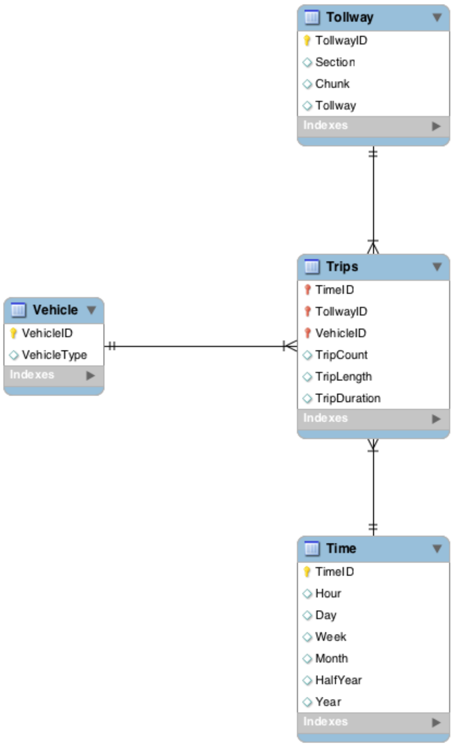
Q5A. Transurban operates toll roads in Australia and the United States of America. Traffic data analysis helps determine road maintenance, RFID reader maintenance, gantry maintenance and capacity management to reduce inefficiencies in its tollway network.

Each part of the tollway is identified by a section (E3), and multiple sections (E1, E2, E3) make up a chunk, (C3) and multiple Chunks make up an entire roadway ("Eastlink"). Many different vehicles on toll roads every day. Each vehicle falls into one category: motorcycles, passenger vehicles, 4WD's, vans, trucks, prime movers, buses, trailers, recreational and miscellaneous machinery (e.g. tractors, cranes, street sweepers, back hoes).

Transurban's management wants to understand the vehicle trips on its road network. They need to understand the number of trips, trip length, trip duration, at different times of the day, on different days and during different period (e.g. Easter holidays, school holidays).

Draw a *star schema* to support the design of this data warehouse, showing the attributes in each table. You do not need to select data types. Clearly display the legend for Primary Key, Foreign Key and Primary Foreign Key.

(8 marks)



Q.5B. Why are star schemas preferred over relational database designs to support decision making?

(2 marks)

Star schemas support Data Warehouses which are organized around facts (business measures) and dimensions that help with managerial decision making. Star schemas are denormalised, making it faster to aggregate and query data. Faster aggregates and query data is required due to the large volume of data stored in data warehouses.

Q6. Security and Backups (10 marks)

Q6A. One of the technical safeguards of database systems is access control. Which type of access is based on the "need to know" principle? Explain how this principle works with an example.

(4 marks)

The "need to know" principle is the core of mandatory access control (MAC); it works on user clearance levels which restricts (or allows) user access to objects. Users can only access objects cleared at their levels or lower access levels. The access to objects is centrally controlled by the policy. Users cannot transfer access to owned objects to other users.

MAC model is often enforced in government systems or at NASA with labels assigned to objects, e.g. "confidential", "secret", "top secret". When a user tries to access a "top secret" object, the clearance level is checked before allowing or denying access.

Q6B. Learning Management Systems (LMS) use Role-based Access Control (RBAC). Explain this statement.

(2marks)

In LMS, e.g. in Canvas, when a user account is created, that user is allocated a role, e.g. Student, Instructor, Tutor, Marker. Students have read only access to Modules whereas Instructors have read and write access. A user in the Student role can only see his/her marks whereas users who are Instructors, Tutors, Markers can read and write marks for all students.

Q6C Illustrate the concept of an SQL injection. Describe a scenario and a string a hacker could enter that would jeopardise database security.

(4 marks)

SQL injections are used by hackers to insert their own commands into the developers' SQL statements. If user input becomes part of the pre-existing statement it is possible to edit that statement so that it does what the hacker wants (e.g. shows the database structure, such as table names, or deletes or modifies data). For example, let's say a developer has a statement with the direct input of user name and password

```
SELECT *
FROM User
WHERE username = ' @name '
and password = ' @pw ';
```

A hacker can enter username as **' or 1=1; --**
Or 1=1 – will evaluate the expression to true telling the system that username is valid
-- will discard any SQL following them including password validation so that access is granted and only hacker's commands are executed.

Q7. Transactions (10 marks)

Q7A. What is the inconsistent retrieval problem? Describe the problem and use your own example to demonstrate the answer.

(4 marks)

The inconsistent retrieval problem is where data in one long running read session (SQL SELECT) is reading data that has been modified in other sessions. At the end of the read session it may have read unmodified and modified data from other sessions.

In the table below Session 1 is calculating the interest payment for credit card payments for all credit card accounts at a bank. Session 2, Alice is paying off her entire credit card balance (a series of update statements). In Session 3 Bob is adding more credit to his credit card balance - another update.

Timestamp	**Session 1** Query Calculating credit card interest	**Session 2** Alice	**Session 3** Bob
1	SELECT begins	Balance 673.50	Balance 99.50
23	Reads Alice's balance		
34		Alice pays off her entire balance	
47			Bob adds $49.50 to his credit
55	Reads Bob's balance		
60	Finishes running query		

Table 6. Transaction log demonstrating inconsistent retrieval problem.

In this scenario, the query is not including Alice's payment, but is including Bob's purchase. This is an inconsistent retrieval of data.

Q7B. A sales company decided to increase salaries by 2% across all jobs. Write a transaction to increase annual salary as specified.

The Jobs table was created using DDL:

CREATE TABLE Jobs (JobCode int, JobTitle char(15), AnnualSalary decimal(10,2), primary key(JobCode));

Hint: it is a better practice to use variables instead of hardcoded values.
Syntax example: SET @name='John';

(6 marks)

```
START TRANSACTION;
SELECT * FROM Jobs; -- optional
SET @increase = 0.02;
UPDATE Jobs SET AnnualSalary = AnnualSalary * (1 + @increase);
COMMIT;
```

Q8. NoSQL (4 marks)

Q8. Domain integrity can be violated in NoSQL databases whereas Relational databases will report a domain integrity violation. Discuss the benefits and risks of violating domain integrity.

(4 marks)

Benefits of domain integrity – data stored in the database must comply with the rules of that domain (e.g. Date datatype must be stored as a date).
Disadvantage of domain integrity – is that we are limited by the databases capability to what types of domains we can store (e.g. spatial data)

Benefit of violating domain integrity – we can store many different types of data that is thematically related on a topic e.g. novel, movie, reviews, critical essays, pictures they do not confirm to any specific domain integrity and can be stored. We can store data as we collect it without having to modify it (schema on read).

Disadvantage of violating domain integrity
Data accuracy, data integrity and data reliability are all compromised, resulting in poor data quality.

END OF EXAM