School of Computing and Information Systems

INFO90002

Database Systems and Information Modelling SOLUTION

Practice Exam 1

Semester 1 2021

Reading Time: 15 minutes

Writing Time: 120 minutes (2 Hours)

This exam has 10 pages including this page
ATTEMPT ALL QUESTIONS IN ANY ORDER

Authorised Materials:

While you are undertaking this assessment, you are permitted to

- make use of any textbook, lecture slides (including soft copies)
- MySQL Workbench is supported for E.R. modelling questions
- Draw IO (diagrams.net) for Chen conceptual models or any suitable modelling equivalent
- Any lecture notes, books, laptop, PC
- You are free to use the course materials and your laptop/PC in this exam.

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

- The total for this exam is 100 marks
- Attempt **all** 9 questions which are of unequal marks value
- Be sure to number your answers to ensure your questions is marked
- Questions can be answered in any order
- **PLEASE DO NOT USE RED font colour or pens.**
- You may revisit/edit your exam answers throughout
- You must not communicate with other students whilst taking this exam, e.g. using messaging, chat rooms or email

IMPORTANT

- **Your file upload must be a single Word document before the elapsed time.**
- **No other document format will be assessed (e.g. Pages, txt, .SQL, etc).**
- **Email submissions will not be assessed.**
- **Every question attempt must be numbered (e.g. Q2C, Q1, Q6) to ensure it is assessed.**
- **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 copy directly the text from any material without attribution.

Q1. ER Modelling (20 marks)

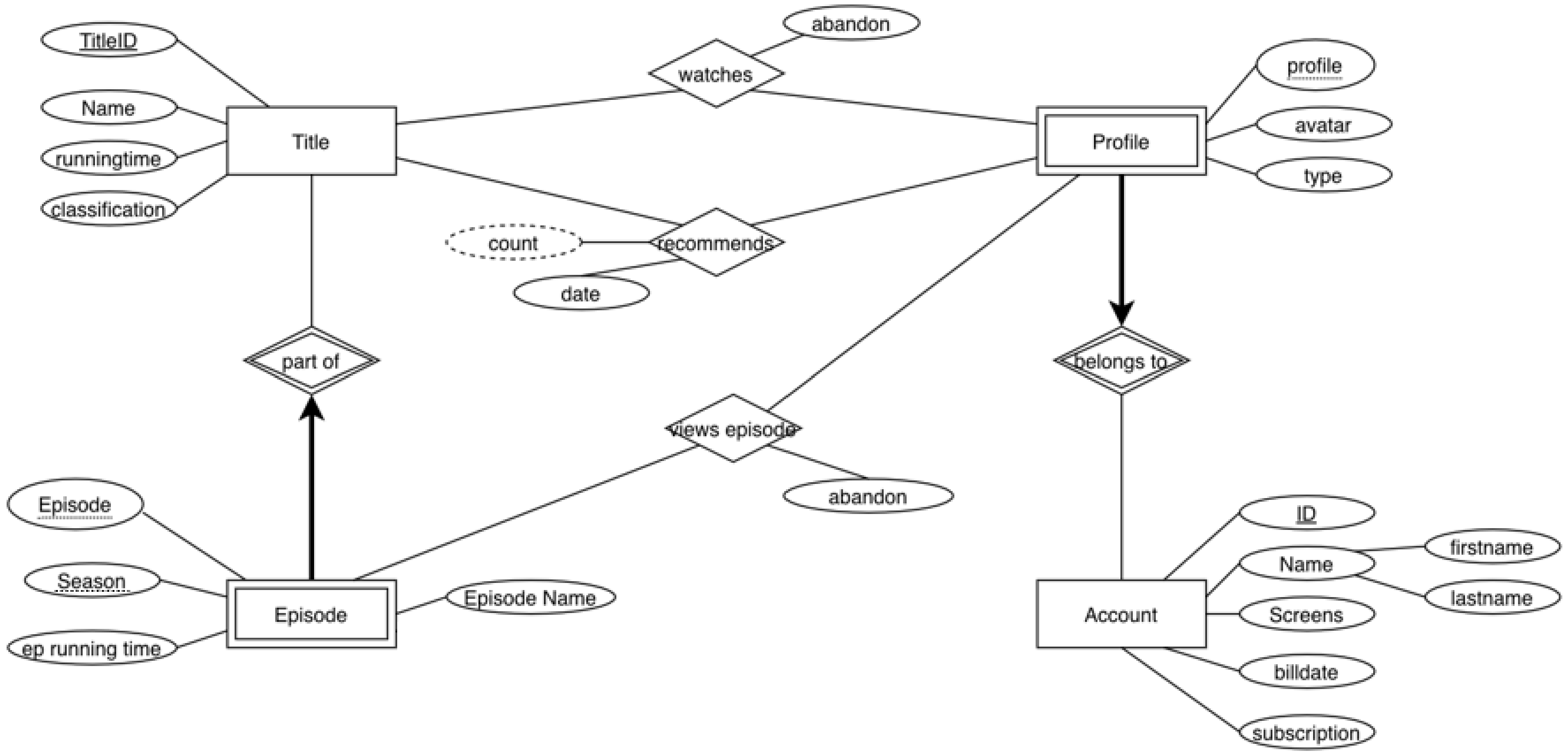
Q1. Criterion Classics is an on-demand streaming service that streams old television shows (e.g. ‘Skippy’) and movies (‘Lawrence of Arabia’), including black-and-white movies (‘Casablanca’) and television shows ("I dream of Lucy"). Each Criterion Classics account can have multiple profiles associated with it – for example, a family account could have one profile for the parents and one profile for each child. Profiles of type ‘Child’ can only view titles classified as ‘U’ (Universal) and ‘G’ (General). Each profile is stored with an avatar or digital image. About each account we store a name ('Mary Lawson') the monthly subscription price, the number of concurrent screens allowed and the day of the month (18th) the account is charged and the name of the subscription plan (Basic, Standard, Premium).

Each movie is an individual title. TV shows can have one or more seasons and up to 24 episodes in a season. Each title of a TV show (e.g. “Breaking Bad”) will also have episodes title (e.g. “Ozymandias”) as well as season number (5) and episode number (14) as well as its running time 47:35 (minutes and seconds) and classification ('MA').

Criterion Classics stores the length of each title/episode in minutes and seconds, as well as the elapsed time a particular profile stopped watching the title/episode. This is so users can 'continue watching' where they left off if they have to stop watching for any reason. If a user watches the title to the end, the abandon time is equal to the title/episode length time.

By collecting and storing this information, Criterion Classics can make recommendations to other users based on profile data analysis. Criterion Classics makes recommendations about rating appropriate titles to all profiles. Criterion needs to store the name of the title, the date it was suggested to a profile and the number of times (total count) a particular title has been recommended to a user. Using this information and the viewing information of a profile it can determine if its suggested viewing algorithm is working.

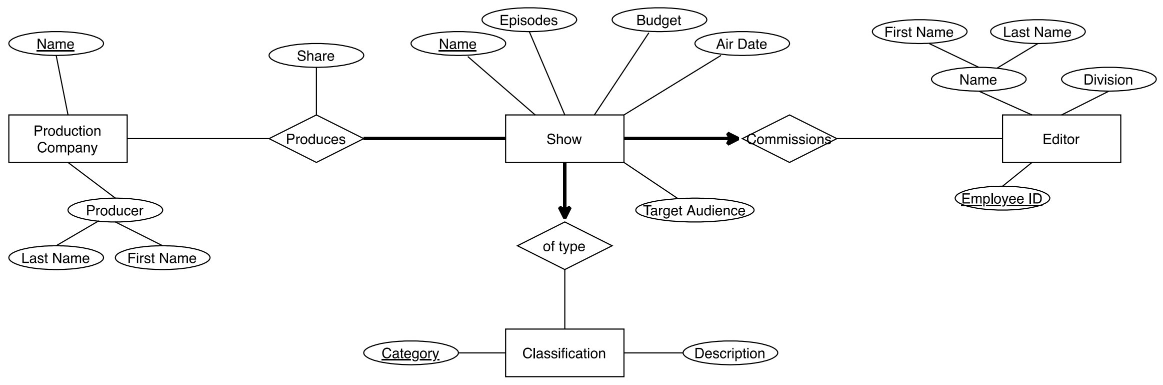
Q1. You are asked to model a **conceptual** Model of the Criterion Classics case study in **Chen** notation. State any assumptions you have made.


TitleID
Name
runningtime
classification
Title
watches
abandon
Profile
profile
avatar
type
recommends
count
date
part of
views episode
abandon
belongs to
Episode
Season
ep running time
Episode
Episode Name
Account
ID
Name
firstname
lastname
Screens
billdate
subscription

Q2. 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).

Q2. Write the SQL DDL for the following Chen conceptual model. There is no need to specify not null. You can use whichever data types you think are appropriate for a MySQL relational database



```
CREATE TABLE Editor
(Employee_ID   INT,
 Firstname VARCHAR(20),
 Lastname VARCHAR(25),
 Division  VARCHAR (40),
 PRIMARY KEY (Employee_ID))

CREATE TABLE Show
(Name CHAR(30),
 Episodes TINYINT,
 Budget DECIMAL(8,2),
 Airdate  DATE,
 Audience   VARCHAR(30),
 Category CHAR(3),
 Editor INT,
 PRIMARY KEY (Name),
 FOREIGN KEY Editor REFERENCES Editor(Employee_ID),
 FOREIGN KEY Category REFERENCES Classification(Category))

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

CREATE TABLE ShowProducer
(Show CHAR(30),
 Producer CHAR(30),
 SHARE TINYINT,
PRIMARY KEY (Show, Producer),
FOREIGN KEY Show REFERENCES Show(Name),
FOREIGN KEY Producer REFERENCES Producer(Name))

CREATE TABLE Producer
(Name  CHAR(30),
 FirstName VARCHAR(20),
 LastName VARCHAR(25),
 PRIMARY KEY (Name))
```

Q3. SQL (15 marks)

This case study is for the SQL model to help contextualise the Physical ER model

Lunch Rider

LunchRider is a new startup in the food delivery business. LunchRider allows people to order lunches from local food vendors and have them delivered by a delivery rider on a bike. When a customer opens the LunchRider app, it first presents a list of local vendors such as cafes, restaurants and snack bars. The customer clicks on a vendor. Then the app shows the meals available from that vendor.

The customer chooses which meals they want from that vendor: for example “2 chili burgers meals" and "1 mango smoothie”. The phone sends this order to the LunchRider server, along with the customer's phone GPS coordinates - the meals will be delivered to this location. Customers can click to “like” a meal and the total number of likes (across all customers) is displayed beside each meal. When a customer’s order is received, LunchRider broadcasts a work offer to riders who are near the customer. Riders see the offer pop up on their app and can press “accept” or “no thanks”.

The chosen rider goes to the vendor, picks up the meals, and delivers them to the customer’s location. We record at what time the rider delivers the order. Payment is automatically deducted from the customer’s credit card and bank transactions are handled by the bank. LunchRiders provide their name, mobile phone number, and date of birth. Whenever a rider is on duty, the rider’s app sends the GPS coordinates about once per minute, allowing us to keep track of each rider’s location.

If someone wants to order a meal, they need to first register as a customer, giving their name, email address, mobile phone number and current credit card. Over time a customer may register more than one credit card. Vendors must register the name and address of their business, including GPS coordinates and a contact email address, and provide the name, price, description (max 1,000 characters) and photo of each meal that they want to sell.

After delivery, the app allows the customer to rate the rider’s service, choosing from 1 star (worst), 2, 3, 4 or 5 stars (best). The app also allows the customer to add the rider to their “favourite” list. LunchRider uses this information to help choose riders for future work offers.

All locations 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). LunchRider uses a precision of 4 decimal places, which is about 11 metres at the equator (smaller in Melbourne). For example, the Doug McDonell building at The University of Melbourne is at latitude -37.7989, longitude 144.9627.

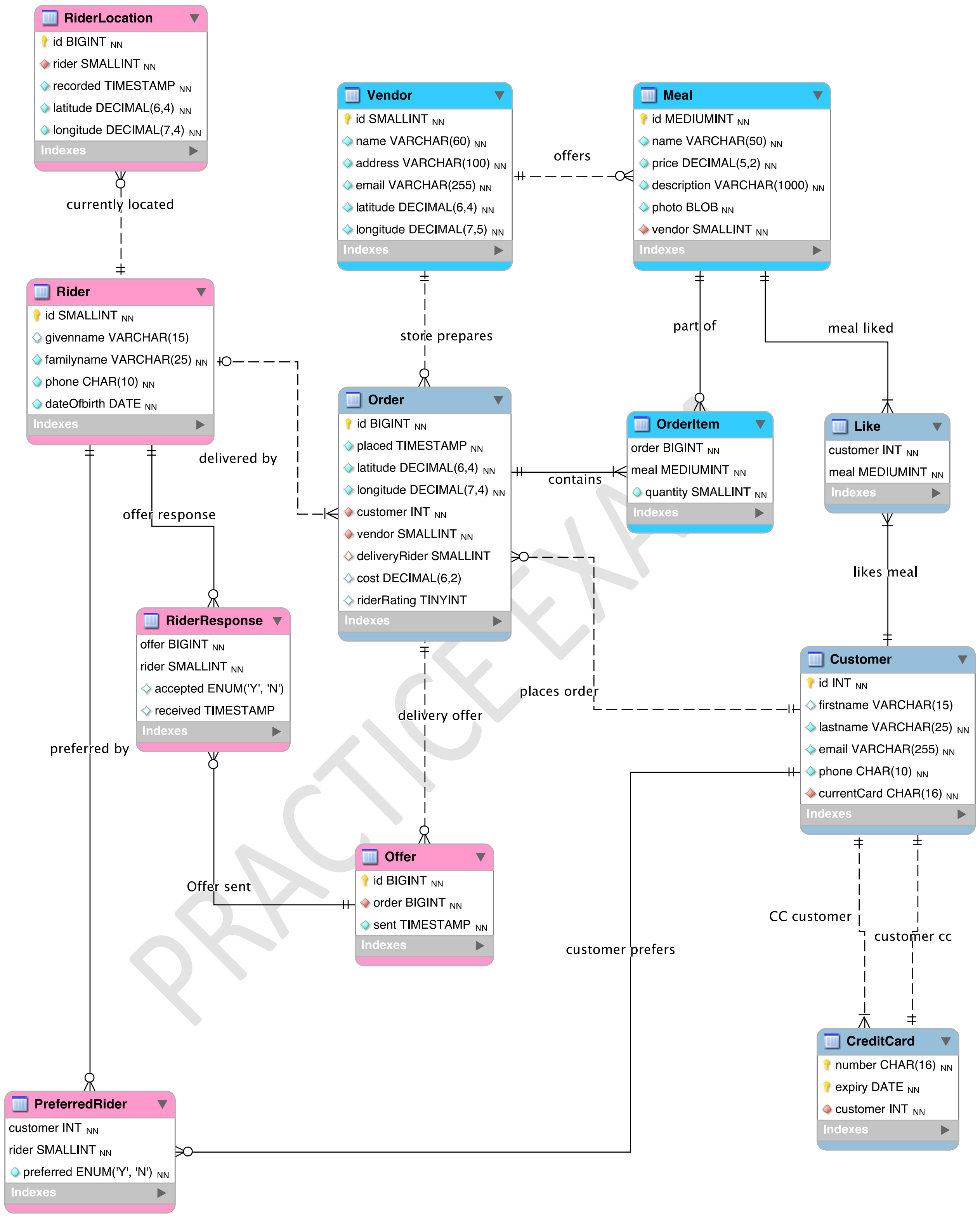


Figure 2: The Lunch Rider Entity Relationship Model

Questions 3A-3E require you to write one single SQL statement per question. Do **not** use views, temporary tables or inline views. Format code for ease of reading. Ensure user-friendly output by renaming columns where appropriate. Values of column fields are in italics e.g. *Tom Hardy*

For Example:

Q. List the name and salary of *Jane Grey*

```
SELECT firstname, lastname, salary
FROM  employee
WHERE firstname = 'Jane'  AND  lastname = 'Grey';
```

Q3A. Write the SQL for the following query:

List all Vendor names who offer a meal where the price is greater than $60.00 and less than $80.00

```
SELECT Vendor.Name
FROM Vendor INNER JOIN MEAL
ON Vendor.ID = Meal.vendor
WHERE Meal.price BETWEEN 60 and 80;
```

Q3B. List the names of riders born in *1997* have accepted an offer in *August 2020*?

```
SELECT givenname, familyname
FROM rider INNER JOIN RiderResponse as RR
ON rider.id = RR.rider
WHERE RR.accepted = 'Y'
AND YEAR(rider.dateOfbirth) = 1997
AND MONTHNAME(RR.received) = 'August'
AND YEAR(RR.received) = 2020;
```

Q3C. How many customers have ordered and liked the '*Betty Deluxe Burger Meal*' meal ordered from the Melbourne store with coordinates latitude *-37.8156* and longitude *144.9636* ?

```
SELECT m.name, count(l.customer)
FROM Vendor v INNER JOIN Meal m INNER JOIN Like l INNER JOIN OrderItem oi
ON v.ID = m.vendor AND Meal.ID = l.Meal AND oi.Meal = m.ID
WHERE v.latitude = '-37.8156' AND v.longitude = '144.9636'
AND m.name = 'Betty Deluxe Burger Meal'
GROUP BY m.name;
```

Q3D. List the names of riders who are also customers of the Lunch Rider app (assume the rider uses the same phone number as they would as a customer)

```
SELECT firstname, lastname
FROM Customer
WHERE EXISTS
    (SELECT *
     FROM Rider
     Where Rider.phone = Customer.phone);
```

Q3E. List the vendor names who have never had any ordered meal liked by any customer

```
SELECT vendor.name
FROM VENDOR
INNER JOIN MEAL
INNER JOIN Order
ON Vendor.ID = Meal.Vendor AND Meal.ID = OrderItem.meal
WHERE Meal.ID NOT IN
     (SELECT meal
     FROM Like)

-- (NOT EXISTS also possible)
```

Q3F. Name the riders who have received a rating of 4 or above from more than 500 different customers

```
SELECT givenname, familyname
FROM Rider INNER JOIN Order
ON Rider.ID = Order.deliveryrider
WHERE Order.riderrating >= 4
GROUP BY Rider.ID
HAVING COUNT(DISTINCT(Order.customer)) > 500;
```

Q4. Normalisation (15 marks)

The table below is part of the medical records for a Veterinarian Clinic
PRACTICE (Animal_ID, Animal, Animal type, OWNER_ID, Owner, Phone, Consult, PROC_ID, PROC_DESC)

Animal ID	**Animal**	**Animal Type**	**Owner_ID**	**Owner**	**Phone**	**Consult**	**ProcID**	**Description**
317	Ralph	Dog	10	Julie Sumner	0409 673-888	13-Oct-13	101	Annual Checkup
317	Ralph	Dog	10	Julie Sumner	0409 673-888	27-Apr-13	115	Teeth Clean
317	Ralph	Dog	10	Julie Sumner	0409 673-888	14-Oct-14	119	3 month Checkup
398	Zeno	Canary	23	Tony Rijks	0408 322-444	21-Jul-14	105	Parasite treatment
398	Zeno	Canary	23	Tony Rijks	0408 322-444	14-Oct-13	119	3 month Checkup
441	Panda	Short haired cat	47	Helene Hanff	0419 121-212	24-Apr-14	715	Initial Consultation
441	Panda	Short haired cat	47	Helene Hanff	0419 121-212	27-Apr-13	115	Teeth Clean
518	Zeno	Canary	23	Tony Rijks	0408 322-444	1-Mar-15	001	6 month Checkup

The combination of ANIMAL_ID and PROC_ID is the candidate key for the relation. The following functional dependencies hold:

- Animal_ID → ANIMAL, ANIMAL Type, OWNER_ID
- OWNER_ID → OWNER, PHONE
- PROCID → DESCRIPTION
- ANIMAL_ID, PROC_ID → Consult

Q4. Please normalise the data to third normal form (3NF). Be sure to identify all anomalies, and show each stage (1NF, 2NF, 3NF).
Key: **BOLD** primary key *ITALIC* foreign key ***BOLD + ITALIC*** primary foreign key

END OF PRACTICE EXAM: PART 1

1NF All atomic cells
However Insert Update Delete Anomalies exist
- DELETE Zeno the Canary we lose the 6 month checkup procedure (001)
- UPDATE Ralph's owner – multiple updates
- INSERT Have to know the consultation before we can add owner and pet

1NF
PRACTICE(**animalID,** animal, animal_type, owner_ID, owner, phone, **procID**, procedure, consult)
2NF
To be in 2NF - 1NF and No partial functional dependencies
Partial functional dependency exists in PRACTICE
animalID ➔ animal, animal type, owner_ID
ProcID ➔ procedure

ANIMAL(**animalid**, animal, animal type, owner_ID, owner, phone
PROCEDURE(**procid**, procedure)
PRACTICE2(***procID***, ***animalID***, consult)
Anomolies
If the owner changes we have multiple updates
3NF
To be in 3NF - 2NF and no transitive functional dependencies
Transitive functional dependencies exist
Owner_ID --> owner, phone not dependent on animal_ID of ANIMAL

PROCEDURE(**procid**, procedure)
PRACTICE2(***procID***, ***animalID***, consult)
ANIMAL2(**animal_id**, animal, animal_type, *owner_ID*)
OWNER(**owner_ID**, owner, phone)

Q5. NoSQL (4 marks)

Q5. There are trade-offs between the principles of ACID and BASE. Discuss the trade-off between availability and consistency for relational and NoSQL databases. Illustrate your answer using either the example of Facebook or the example of Twitter.

(4 marks)

NoSQL databases need to be available all the time. Facebook (or Twitter) needs to ensure that users can access and post to its site and app all the time, even if the data is not consistent. For example, users expect to always see the number of likes on a post (or retweet count on a tweet); they do not mind if that number is slightly outdated.

Relational databases sacrifice availability for consistency. This means the database needs to be consistent all the time for all the users. If there is potential for inconsistencies the system may shutdown, this would cause severe disruption to Twitter/Facebook which rely on being always available.

Q6. Applications (4 marks)

Q6. There are problems with giving end users an SQL interface to access a database. Describe **two** (2) distinct problems, and for each, how providing application software to users solves the problem.

(2+2=4 marks)

Providing users with an interface to a database presents two distinct problems. The first is users would be able to make indiscriminate changes including errors, and would all need to be trained in the SQL language

The second is logically units of work. By providing an application interface you can structure SQL into transactions such as purchasing goods online, transferring funds, or adding student results for subjects.

Q7. Distributed Databases (4 marks)

Q7. Describe how synchronous updates work and the advantages and disadvantages of this approach.

(4 marks)

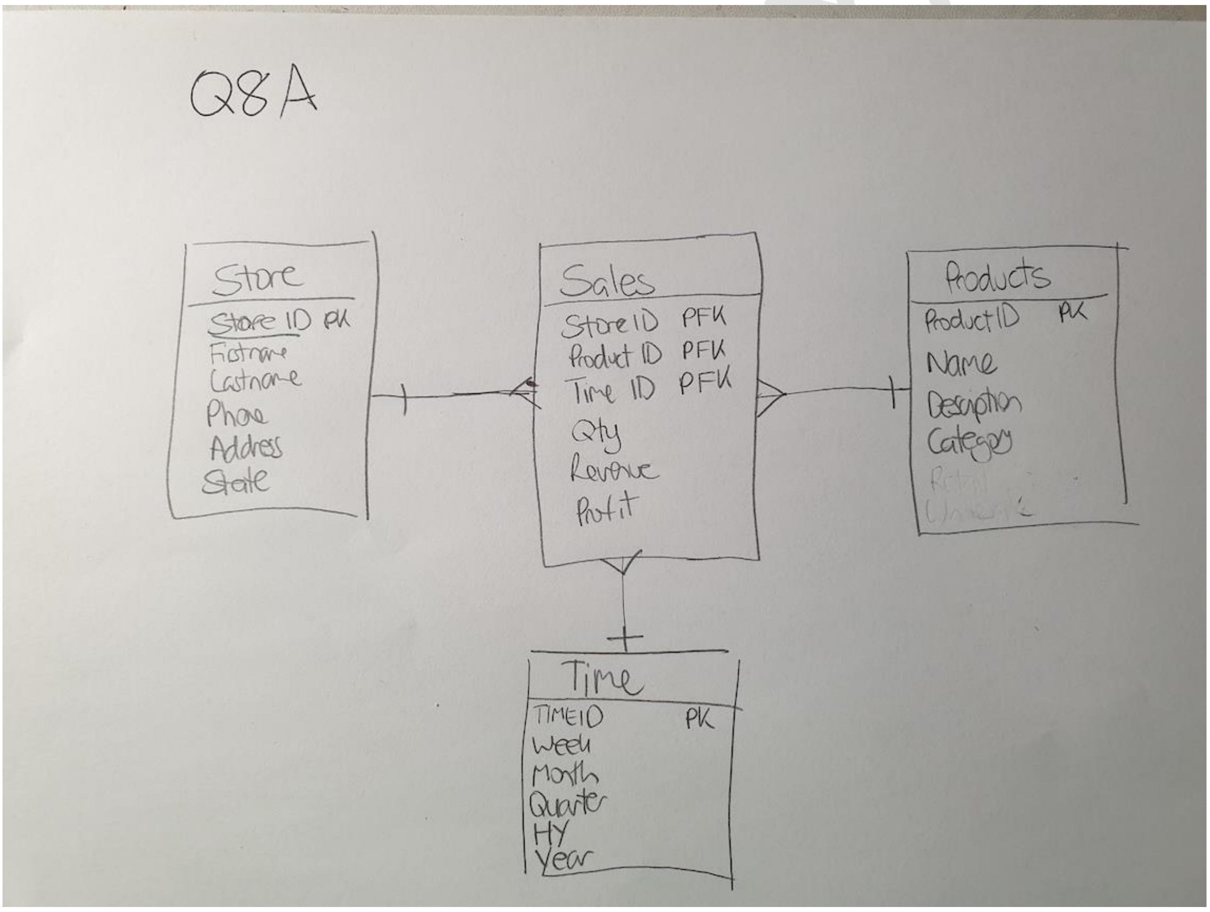
Synchronous updates apply to Distributed Replicated databases. When a change is made on one node/server in the distributed database the change is immediately sent to all other nodes in the distributed database. All users on all nodes see the same current value for all data immediately – therefore providing excellent data integrity
The problem with synchronous replication is that any break in the network would cause an outage, as data will not be consistent. Synchronous replication increases network traffic and may result in slower performance.

Q8. Data Warehouses (14 Marks)

Sennheiser Australia wants to track information about the selling of audio equipment (microphones, headphones, turntables) via physical retail stores located throughout Australia. You need to design a data warehouse to report information about sales of audio equipment over time. You need to store the retail store (store ID, Manager's first name, Manager's last name, phone number, address, state), and product (product-id, name, description, category, retail price, wholesale price). The sales managers want to find the number of items sold (e.g. turntables), the revenue and profit. The information needs to be accessible by the retail store, its state, product name (e.g. "Sen-300-II") and for different times (week, month, quarter, half year and year).

Q8A. Draw a *star schema* to support the design of this data warehouse, showing the attributes in each table. Be sure to denote PK, FK and PFK.

(8 marks)



Students need to draw a star schema in correct Crow's foot notation

Sales (fact table)	**Store (dimension table)**	**Product (dim. table)**	**Time (dim. table)**
StoreID PFK	StoreID PK	ProductID PK	TimeID PK
ProductID PFK	ManagerFirstName	ProductName	Week
TimeID PFK	ManagerLastName	Description	Month
Quantity	PhoneNumber	Category	Quarter
Revenue	Address		HalfYear
Profit	State		Year

Q8B. Briefly explain what OLTP and OLAP databases are and demonstrate the difference between these two types of databases.

(6 marks)

OLTP are online transactional processing databases designed to collect the data associated with day to day operations of the business. The database design is normalised and the data is volatile in nature and subject to changes. Data is both read and written.

OLAP are online analytical processing databases. They are designed to analyse historic data that is largely static and in a finalised form. The majority of transactions are read only - and long running, and designed for management and strategic - not operational decision making. OLAP databases are usually focused on a single theme (e.g Sales).

The difference between the two systems are listed below

Characteristic	**OLTP**	**OLAP**
Data	Volatile	Finalised
Transactions	Read and Write; Many short transactions	Read only Long running
Database Use	Operational	Management and Strategic
Users	Many users for day to day operations	Managers to make strategic decisions

Q9. Ethics & Security (4 marks)

Q9. What are the advantages and disadvantages of a logical backup when compared with a physical backup of a database?

(4 marks)

The advantage of a logical backup is that it can occur while the database is still running – and therefore not interrupt database availability. The backup is in a series of SQL statements than can be ported to any operating system.

The disadvantages of a logical backup is that it may impact database performance as it is resource intensive. Logical backups are larger in size than physical backups and usually take longer to complete.

END OF EXAM

GOOD LUCK!