RMIT UNIVERSITY

School of Science

ISYS1055 Database Concepts

Assessment 1: Database Design

Assessment type: PDF

Word limit: N/A

Due Date: 10 Apr at 23:59 (Melbourne time)

Weighting: 20%, 20 Marks

Overview

The objective of this assignment is to measure your understanding of the basic concepts in the relational database model and using entity-relationship model for database design. The assessment is in two parts, split into four tasks which cover Basic ER Modelling and Basic Relational Modelling. The tasks are as follows.

Part A: Entity-Relationship Modelling (12 Marks)

1. Design and plan for the implementation of a database system, diagramming the design to a high standard using UML notation through the diagramming tool Lucidchart.

2. Model the activities of an organisation and present the model as an Entity-Relationship (ER) diagram. Analyse this ER diagram, and possibly modify it, based on additional client requirements.

3. Map an ER diagram into a relational database schema, showing every step of the mapping.

Part B: Relational Database Model (8 Marks)

4. Answer a series of short questions about a Relational Database model.

To complete this assessment, you must be familiar with Lucidchart, which is covered during the Week 1-4 activities.

Assessment criteria

This assessment will measure your ability to:

- Describe various data modelling and database system technologies.
- Explain the main concepts for data modelling and characteristics of database systems.

Course learning outcomes

This assessment is relevant to the following course learning outcomes:

CLO1	Describe the underlying theoretical basis of the relational database model and apply the theories into practice.
CLO2	Explain the main concepts for data modelling and characteristics of database systems.
CLO3	Develop a sound database design using conceptual modelling mechanisms such as entity-relationship diagrams.
CLO4	Develop a database based on a sound database design.

Part A: Entity-Relationship Modelling

Task 1: Designing an Entity-Relationship Model

Food Saver Case Study

Food Saver (FS) is a supermarket that sells a variety of food products in Australia. The following are the requirements for managing data about staff, products and customers for FS.

Stores

FS has stores throughout Australia. Each store is allocated several staff members. One staff member manages the operations of the store. Each staff member works at a specific store only. The data describing a store includes a unique store number, an address (i.e., street, suburb, and postcode) and several contact types (e.g., email, fax, phone). FS keeps track of which staff members work at each store, including the manager (and the date each staff member starts his/her position at the store he/she is working at).

Staff

It is important to distinguish between two types of FA staff members (i.e., supervisors and workers) as supervisors have additional responsibilities. Supervisors are responsible for the day-to-day activities of a team of workers. Each supervisor is responsible for all staff at a particular store. The data for each staff member includes a unique staff number, name, address, position, and salary.

Products

FS has a range of products that it sells to customers in Australia. At any given time at a store, a particular product has a specific quantity in stock which must be tracked. Other data for each product includes a unique product number, a name, description and sale price.

Customers

When customers register on the FS website, the system records their unique email address, their name, delivery address (i.e., street, suburb, and postcode) and payment details (Credit Card number, expiry date and security code).

Orders

Once registered, customers can order products from the website, selecting the product and quantity to be added to their order. Each order has a unique order number and order date.

Based on the given description, model the given business rules, and present your model as an Entity-Relationship (ER) diagram. Carefully state any assumptions that you make. In your ER diagram, you must properly denote all applicable concepts, including weak or strong entities, keys, composite or multi-valued attributes, relationships and their cardinality and participation constraints.

If you cannot represent any of this information in the ER model, clearly explain what limitations in the ER model restrict you from representing your model.

You **must** use UML notation and the diagramming tool Lucidchart to draw your diagram. Your diagram must be drawn to a high standard with minimal clutter. You are **not** required to map the ER model to relational model.

A special note: This is an open-ended question with many different models that can be derived. Your model is assessed based on how accurately it represents business rules described above.

Task 2: Designing an Entity-Relationship Model

Part A: Initial Design

Sunny Holidays Case Study

Sunny Holidays is an Australian travel company that allows customers to book holiday packages.
You are asked to design a database for managing customer bookings. Requirements for the database are as follows:

- Customers book holiday packages recording the date the booking was made. Each customer has a firstname, lastname, date of birth, and passport number.
- Holiday packages have a package no, and have a set fee (depending on the departure date). They consist of several stop overs, each in a city within a country. Each country could have several stop over cities.
- Each stop over has a sequence number (eg stop number 1, 2 etc).
- The system stores a complete list of cities and countries, some which may not necessarily have an associated holiday stop over (yet).
- All holiday packages have at least one stopover.

Based on the given description, model the business rules of Sunny Holidays, and present your model as an Entity-Relationship (ER) diagram. Carefully state any assumptions that you make. In your ER diagram, you must properly denote all applicable concepts, including weak or strong entities, keys, composite or multi-valued attributes, relationships and their cardinality and participation constraints.

If you cannot represent any of this information in the ER model, clearly explain what limitations in the ER model restrict you from representing your model.

You **must** use UML notation and the diagramming tool Lucidchart to draw your diagram. Your diagram must be drawn to a high standard with minimal clutter. You are **not** required to map the ER model to relational model.

A special note: This is an open-ended question with many different models that can be derived. Your model is assessed based on how accurately it represents business rules described above.

Part B: Client Adjustments

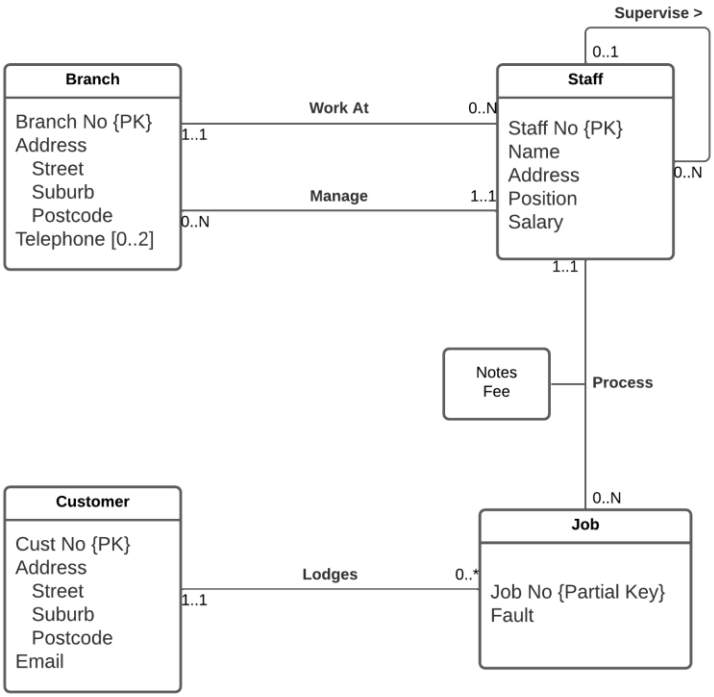
After presenting your ER model to Sunny Holidays management, you are asked if it can be used to perform the following additional tasks.

- Customers arrive at and depart from each stop over on a flight at a specific datetime.
- Flights have a unique code and also a duration.
- Depending on the duration of the stop over, there may be a single nominated accommodation.
- Accommodation has an identifying address and has a phone number.
- The check in and check out datetime is recorded for each accommodation stay.

For each one of the tasks specified above, explain how your ER diagram is supporting it. If it is not possible to achieve any of the tasks above given your current design, state why, modify the model, provide the modified ER diagram (in addition to your original ER diagram), and explain how the new model achieves the missing requirements.

Task 3: Mapping an ER Model to a Relational Database Schema

Consider the following ER diagram, which shows aspects of the business of a computer service company.



You are requested to map the above ER diagram into a relational database schema. Show every step of the mapping. No marks are awarded to the final schema if you do not show the partially built schema at the end of each step. Clearly indicate the primary key (underlined) and foreign keys (with an asterisk) in each relation.

Part B

Task 4: Relational Database Model

This section contains the schema and a database instance for the Employee database that stores employee data for an organisation. The data includes items such as personal info (e.g., name, phone, salary), departments of the organisation (e.g., name and location of each department, who the manager is), jobs (e.g., titles, salary range), and a history for past contracts with each employee. A database instance is shown in Figure 2 followed by the database schema.

Employees

employee_id	first_name	last_name	phone_number	hire_date	empjob_id	salary	department_id
50	Adam	Smith	1234	26/10/2009	22	$66,000	2
66	Tom	Moosa	1235	10/12/2016	10	$140,000	2
10	Jonny	Deans	1236	9/09/2015	33	$70,000	1
12	Adam	Jones	1247	8/08/2019	10	$138,000	1
18	Joseph	Ryan	1277	5/05/2020	10	$150,000	3

Departments

department_id	department_name	manager_id	location_id
1	IT Services	12	10
2	Accounting	66	20
3	Human Resource	18	30

Jobs

job_id	job_title	min_salary	max_salary
10	Dep Manager	$120,000	$150,000
22	Accountant	$60,000	$80,000
33	Programmer	$60,000	$80,000
45	Senior Programmer	$70,000	$120,000

Locations

location_id	street_address	postal_code	city	state_province	country_id
10	123 Collins St	3000	Melbourne	VIC	1
20	222 Bourke St	3000	Melbourne	VIC	1
30	555 Swanston St	3000	Melbourne	VIC	1

Countries

country_id	country_name
1	Australia
2	Vietnam
3	Spain

Job History

employee_id	start_date	end_date	job_id	department_id
10	1/01/2001	10/04/2002	33	1
10	11/04/2002	20/08/2002	33	1
12	1/01/1998	5/10/2003	33	1
12	6/10/2003	6/10/2004	33	1
12	7/10/2004	7/08/2009	33	1

Figure 2: Employee Database Instance

The database schema is shown below, and the meaning of most attributes is self-explanatory. “Job History” is simplified as “JobHistory”. Primary keys are underlined, and foreign keys are annotated with a *.

Employees(employee_id, first_name, last_name, phone_number, hire_date, empjob_id, salary, department_id)

Departments(department_id, department_name, manager_id*, location_id*)

Jobs(job_id, job_title, min_salary, max_salary)

Locations(location_id, street_address, postal_code, city, state_province, country_id*)

Countries(country_id, country_name)

JobHistory(employee_id*, start_date, end_date, job_id*, department_id*)

The following table further clarifies the connection between the keys across multiple tables.

Foreign Key		Primary Key
Job History.employee_id	-------->	Employees.employee_id
Departments.manager_id	-------->	Employees.employee_id
Job History.department_id	-------->	Departments.department_id
Job History.job_id	-------->	Jobs.job_id
Locations.country_id	-------->	Countries.country_id
Departments.location_id	-------->	Locations.location_id

Figure 3: Keys

The following questions must be answered based on the given database schema and instance. Where explanation is required, each answer should be a SHORT passage of at most several lines.

QUESTION 4.1: Does the database schema ensure that there is a department associated with each employee? Explain your answer.

Questions 4.2: What change (if any) would you need to make so that the database can store multiple job histories for the same employee with the same start and end dates but with different titles?

QUESTION 4.3: The Human Resource department has recently changed to have three sub-departments (i.e., Ongoing Staff Department, Casuals Department, and External Contractors Department). Now, each sub-department is supposed to have a separate manager. Additionally, the managers of all Human Resource sub-departments now must report to a single Director (i.e., "Director of Human Resources"). Temporarily and until the new managers are hired, Joseph has been appointed to the management of all three sub-departments as well to the single role of Director of Human Resources.

The following SQL statements are intended to record all the changes required in the database instance. Will they work? If they are sufficient to achieve the requirements specified above, explicitly mention they are sufficient. If there are any shortcomings, identify them and briefly justify your answer.

```
UPDATE Departments SET department_name='Human Resource – Ongoing Staff' WHERE
department_id=3;
INSERT INTO Departments VALUES(4, 'Human Resource – Casual', 18, 30);
INSERT INTO Departments VALUES(5, 'Human Resource – Exteral Contractors', 18, 30);
```

QUESTION 4.4: The employee named Adam Smith has recently change his job to become a Programmer. The following SQL statement intends to make the required changes in the database instance to reflect Adam's promotion.

```
UPDATE Employees SET empjob_id=33 WHERE first_name,last_name='Adam Smith';
```

After running the above query, consider the request "find all the past contracts that Adam Smith used to have". Can this request be completed using the given database schema and after the above statement is run? If yes, explain how the request can be answered. If no, explain what is missing and how it should be fixed.

QUESTION 4.5: Explain what the result of executing the following SQL statement on the database instance will be.

```
 UPDATE LOCATIONS SET location_id=40 WHERE location_id=30;
```

QUESTION 4.6: Write an SQL statement to create the Locations table including all the constraints, assuming all the tables that Locations depends on already exist in the database. Make reasonable assumptions for the data type associated with each field. Your SQL statement must be valid for SQLite Studio environment and free of any errors.

QUESTION 4.7: Write an SQL statement to create the Job History table including all the constraints, assuming all the tables that Job History depends on already exist in the database. Make reasonable assumptions for the data type associated with each field. Your SQL statement must be valid for SQLite Studio environment and free of any errors.

QUESTION 4.8: A new branch is opening up in Spain. You are asked to update the given database instance so that it includes the new Location. The location ID is "40", the street address is "666 Diablo st", the postcode is "46001", the city is "El Carmen", the state_province is "Valencia". Your SQL statement must be valid for SQLite Studio environment, free of any errors, and compatible/consistent with existing data in the instance in Figure 2.

Submission format

You should submit one PDF document with all answers together. Do not submit Word files.

You must use Lucidchart to work on Part 1 of your assignment. You may use Word or any other word processor to compile your submission. Use section titles to indicate which question you are answering. At the end, convert your answer sheet into PDF format. Microsoft Word has the option of saving your document in PDF format. If the conversion option is not available on your system or word processor, there are free PDF converters online you can utilise (e.g., http://convertonlinefree.com/).

Academic integrity and plagiarism

Academic integrity is about honest presentation of your academic work. It means acknowledging the work of others while developing your own insights, knowledge, and ideas.

You should take extreme care that you have:

- Acknowledged words, data, diagrams, models, frameworks and/or ideas of others you have quoted (i.e., directly copied), summarised, paraphrased, discussed, or mentioned in your assessment through the appropriate referencing methods.
- Provided a reference list of the publication details so your reader can locate the source if necessary. This includes material taken from Internet sites.

If you do not acknowledge the sources of your material, you may be accused of plagiarism because you have passed off the work and ideas of another person without appropriate referencing, as if they were your own.

RMIT University treats plagiarism as a very serious offence constituting misconduct.

Plagiarism covers a variety of inappropriate behaviours, including:

- Failure to properly document a source
- Copyright material from the internet or databases
- Collusion between students

For further information on our policies and procedures, please refer to the University website.

Referencing guidelines

Where referencing is required, use RMIT Harvard referencing style for this assessment.

Refer to the RMIT Easy Cite referencing tool to see examples and tips on how to reference in the appropriated style. You can also refer to the library referencing page for more tools such as EndNote, referencing tutorials and referencing guides for printing.

Penalties for late submissions

Late submissions of assignments will be penalised as follows. For 1 to 5 days late (including weekends), a penalty of 10% (i.e., 10% out of total marks, not 10% out of your marks) per day. For assignments more than 5 days late, 100% penalty applies.

Assessment declaration

When you submit work electronically, you agree to the assessment declaration.