ICT 285 ASSIGNMENT 1

YIN ZHANPENG

34742217

2023

# ICT285 Databases

## Assignment 1

### **Individual Assignment (20% of the final mark for the unit)**

***Please check the LMS for the due date.***

## Submission Instructions:

**Please submit a SINGLE Word document using the link in Moodle**. The due date is on the LMS. You MUST include your student number as part of your document. Your name and student number MUST also be included within the assignment document.

## From the Unit Information Guide:

* It is your responsibility to **keep a copy of any assignment handed in for assessment**. Make sure you keep a copy of the final (submitted) version as well as the backups you make as you create the assignment. It is recommended that you also keep a hard copy of any written assignments.
* It is especially **important to maintain regular backups of all your practical work**. You should ensure you keep a separate backup on a thumb drive, cloud or similar. It is recommended that you keep both a hard copy and an electronic copy.
* Late submission will be penalised at a rate of five marks (out of a possible 100) per day (or part thereof). **The deadline for submission will be specified in LMS and on the assignment**. Each assignment will have a final date by which the assignment will be accepted for marking. Assignments not received by the final date WILL NOT BE MARKED and will be recorded as a 0.
* Under exceptional circumstances an extension can be provided. Extensions for assignment submissions can only be granted if requested in advance of the due date for submission, and with a good reason. Applications for extensions should be made to the Unit Coordinator, not to the student's lecturer/tutor who is unable to grant extensions. If an extension is granted, the Unit Coordinator will tell you at the time of granting it whether any penalty in marks will apply to the submitted work.
* This unit uses software called Ouriginal when viewing work that you submit. **Ouriginal is a pattern-matching system** designed to compare work submitted by students with other sources from the internet, journals/periodicals, and previous submissions. Its primary purpose is to detect any submitted work that is not original and provide a thorough comparison between the submitted document and the original sources.
* The University **takes academic integrity very seriously**. Instances of academic misconduct include Submitting work that has been produced by someone else or either in part of whole using Artificial Intelligence tools. More information about academic integrity is contained within the Murdoch Academic Passport (MAP) unit <https://www.murdoch.edu.au/mymurdoch/support-advice/learning-study/murdoch-academic-passport>

This assignment requires you to answer several questions on relational database principles and SQL, and to design a database based on a case study.

The assignment addresses the following learning outcomes for the unit:

1. Apply your knowledge of relational database principles and theory to create effective and efficient database designs.
2. Define, create, and manage relational database systems using SQL.

You will need to ensure that your materials reflect a **high professional standard\***. The marks allocated for the Assignment will be allotted in terms of the percentages shown in the following table.

|  |  |
| --- | --- |
| Question 1: Relational algebra | 20 |
| Question 2: SQL Select queries | 20 |
| Question 3: Further SQL | 15 |
| Question 4: Normalisation | 20 |
| Question 5: Conceptual design | 25 |
| **Total** | **100** |

**\*Formatting and presenting your assignment appropriately is important, because the assignment includes marks for the overall organisation and presentation of the document (up to 5 marks).** This includes marks for things such as:

* Standard Font Size and Colour i.e., Arial/12.
* Formatting and layout: double line spacing, page number, separate sections (Headings/Subheadings), aligning paragraphs.
* Includes a title page and table of contents. Correct use of equations, footers, and headers.
* Good technical style, avoiding excessive jargon, grammar, and spelling errors.

Before you submit your assignment, make sure to check the formatting and overall presentation of your document. Make sure the file type is a **single** MS word document. If you're not sure about what’s required contact your teaching staff or UC.

# Question 1: Relational algebra (20 marks)

A construction company works with a database that stores information about contractors, materials, and sites. The Consignment relation records instances of a contractor delivering materials for a construction site.

The schema for the database used in this question is as follows: (note that primary keys are shown underlined, **foreign keys in bold**).

Graphical user interface, text, application

Description automatically generated

Provide **Relational Algebra** (NOT SQL) queries, with description as necessary to find the following information. Each question is worth 2 marks.

NOTE:

* You can use the symbols σ, Π, etc or the words ‘PROJECT’, ‘RESTRICT’ etc as you prefer.
* You do not need to try to make efficient queries – just correct ones.
* Where you use a join, always show the join condition.
* Provide a description to explain how the query will run.

1. Show the quantity of Materials delivered on site-ID “Site04”.

PROJECT Quantity (RESTRICT Site-ID = 'Site04' (Consignment))

1. Show the weight of the Wood type material named “Beautiful Grey Door”.   
     
   PROJECT Weight (RESTRICT MaterialName = 'Beautiful Grey Door' AND Type = 'Wood' (Material))
2. Show the Site name and Material name of any Materials where more than 75 of the material has been delivered to a particular site.  
     
   PROJECT Site.Name, Material.MaterialName ((RESTRICT Quantity > 75 (Consignment)) ⨝ Site ⨝ Material)
3. Find the name of any Contractor who have delivered materials to the site called “Humpty Doo Place” or to the site called “Noonamah Place”.  
     
   PROJECT Contractor.Name ((RESTRICT Site.Name = 'Humpty Doo Place' OR Site.Name = 'Noonamah Place' (Consignment ⨝ Site)) ⨝ Contractor)
4. Find the name of any Contractor who have delivered materials to the site called “Humpty Doo Place” and to the site called “Noonamah Place”.  
     
   PROJECT Contractor.Name ((RESTRICT Site.Name = 'Humpty Doo Place'   
   (Consignment ⨝ Site)) ⨝ Contractor) ∩ PROJECT Contractor.Name ((RESTRICT Site.Name = 'Noonamah Place' (Consignment ⨝ Site)) ⨝ Contractor)
5. List the names of Contractors who have delivered the material called “Precious Bamboo”.  
     
   PROJECT Contractor.Name ((RESTRICT Material.MaterialName = 'Precious Bamboo' (Consignment ⨝ Material)) ⨝ Contractor)
6. Find the names of sites that have had materials delivered by “Move Forward Transport” or a contractor located in Wagga Wagga.  
     
   PROJECT Site.Name ((RESTRICT Contractor.Name = 'Move Forward Transport' OR Contractor.Address = 'Wagga Wagga' (Consignment ⨝ Contractor)) ⨝ Site)
7. List the details of any Contractor who has delivered materials to a site that stared in 2023 and (the site is) located in Alice Springs.  
     
   PROJECT C-ID, Name, Phone, Address ((RESTRICT Site.StartYear = 2023 AND Site.Address = 'Alice Springs' (Consignment ⨝ Site)) ⨝ Contractor)
8. Find the name of any material that was not used on a site that commenced in 2022.  
     
   PROJECT Material.MaterialName (Material) - PROJECT Material.MaterialName (RESTRICT Site.StartYear = 2022 (Consignment ⨝ Site) ⨝ Material)
9. List the name of any material that has been delivered to all sites that commenced in 2023.   
     
   PROJECT Material.MaterialName (Material) - PROJECT Material.MaterialName ((PROJECT M-ID (Material) × PROJECT Site-ID (RESTRICT StartYear = 2023 (Site))) - Consignment)

# Question 2: SQL – SELECT queries (20 marks)

**Provide SQL queries and the result tables for the following (20 marks):**

*Please ensure that you include the result table as well as your SQL; you can copy and paste this from either your ssh client or SQL Developer. Each query is worth 2 marks. These tables exist in Rhea and are owned by the user dtoohey. You may, if you wish, create your own copies of the tables under your own account. If you do so, you should ensure that you copy the sample data in dtoohey’s tables.*

*These queries are based on the View Ridge Gallery database you have been using in the Lab sessions. Please see Chapters 6 and 7 of Kroenke for background to the case and table structures.*

*Marks are allocated not only for correct answers, but also for* **best practice** *in the creation of the queries. You should also include a description along with each query to explain how it will run.*

1. Find the details of any works of art along with the full name of the Artist who created the work that have at least two copies recorded in the database (i.e., a work that is listed in the database more than two times).

SELECT w.\*, a.FirstName || ' ' || a.LastName AS ArtistFullName

FROM dtoohey.WORK w

JOIN dtoohey.ARTIST a ON w.ArtistID = a.ArtistID

JOIN (

SELECT Title

FROM dtoohey.WORK

GROUP BY Title

HAVING COUNT(\*) > 1

) subquery ON w.Title = subquery.Title;  
A screenshot of a computer

Description automatically generated with low confidence

1. List the details of any work of art (including the Artist who created the work) that has an Expressionist style.  
     
   SELECT w.\*, a.FirstName || ' ' || a.LastName AS ArtistFullName

FROM dtoohey.WORK w

JOIN dtoohey.ARTIST a ON w.ArtistID = a.ArtistID

WHERE EXISTS (

SELECT 1

FROM dtoohey.WORK

WHERE Description LIKE '%Expressionist%'

AND WorkID = w.WorkID

);  
  
A screenshot of a computer

Description automatically generated with medium confidence

1. List the details of the works of art (including the Artist who created the work, and the acquisition and asking price details) currently held in the gallery (*i.e., works of art that have not been sold).*

*SELECT w.\*, a.FirstName || ' ' || a.LastName AS ArtistFullName, t.DateAcquired, t.AskingPrice*

*FROM dtoohey.WORK w*

*INNER JOIN dtoohey.ARTIST a ON w.ArtistID = a.ArtistID*

*INNER JOIN dtoohey.TRANS t ON w.WorkID = t.WorkID*

*WHERE t.DateSold IS NULL;*

*A screenshot of a computer

Description automatically generated with low confidence*

1. List the sales for each customer made in year 2007 (*i.e., when a customer purchases a work of art from the Gallery, a transaction line is created. For a purchase, there will be values in the DateSold and SalesPrice columns).* The query should include the details of the customer, the transaction and the work of art purchased.

SELECT c.\*, t.\*, w.\*

FROM dtoohey.CUSTOMER c

JOIN dtoohey.TRANS t ON c.CustomerID = t.CustomerID

JOIN dtoohey.WORK w ON t.WorkID = w.WorkID

WHERE EXTRACT(YEAR FROM t.DateSold) = 2007 AND t.DateSold IS NOT NULL AND t.SalesPrice IS NOT NULL;

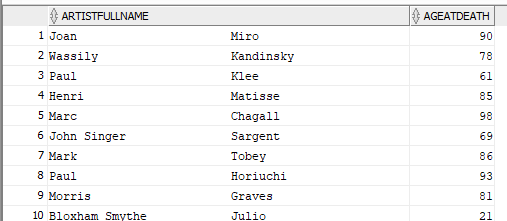


1. List the names of the deceased artists and the number of years of age they were when they died (*for example, an artist born in 1950 and deceased in 2001 has an age of 51).*

SELECT FirstName || ' ' || LastName AS ArtistFullName, DateDeceased - DateOfBirth AS AgeAtDeath

FROM dtoohey.ARTIST

WHERE DateDeceased IS NOT NULL;



1. The sum of the acquisition price of works of art for each year (f*or example, if there were two works of art purchased for $1500 and $1000 in 2019, and one work of art purchased for $500 in 2020, then the sums would be $2500 and $500, for 2019 and 2020 respectively*).

SELECT TRUNC(DateAcquired, 'YEAR') AS AcquisitionYear, SUM(AskingPrice) AS TotalAcquisitionPrice

FROM dtoohey.TRANS

GROUP BY TRUNC(DateAcquired, 'YEAR');  
A picture containing text, screenshot, font, number

Description automatically generated

1. Calculate the profit made on works of art that have been sold in year 2010 (*i.e., the profit/loss on an individual work of art is the difference between the acquisition price and the sales price*).

SELECT w.WorkID, w.Title, (t.SalesPrice - t.AskingPrice) AS Profit

FROM dtoohey.WORK w

JOIN dtoohey.TRANS t ON w.WorkID = t.WorkID

WHERE TO\_CHAR(t.DateSold, 'YYYY') = '2010';  
A screenshot of a computer

Description automatically generated with medium confidence

1. Which artist has had the most works of art sold, and how many of the artist’s works have been sold?

SELECT ArtistFullName, WorksSold

FROM (

SELECT a.FirstName || ' ' || a.LastName AS ArtistFullName, COUNT(\*) AS WorksSold

FROM dtoohey.ARTIST a

JOIN dtoohey.WORK w ON a.ArtistID = w.ArtistID

JOIN dtoohey.TRANS t ON w.WorkID = t.WorkID

WHERE t.DateSold IS NOT NULL

GROUP BY a.FirstName, a.LastName

ORDER BY WorksSold DESC

)

WHERE ROWNUM = 1;



1. Sales of which artist’s works have resulted in the highest average profit (*i.e., the average of the profits made on each sale of works by an artist*), and what is that amount?

SELECT ArtistFullName, AverageProfit

FROM (

SELECT a.FirstName || ' ' || a.LastName AS ArtistFullName, AVG(t.SalesPrice - t.AskingPrice) AS AverageProfit

FROM dtoohey.ARTIST a

JOIN dtoohey.WORK w ON a.ArtistID = w.ArtistID

JOIN dtoohey.TRANS t ON w.WorkID = t.WorkID

WHERE t.DateSold IS NOT NULL

GROUP BY a.FirstName, a.LastName

ORDER BY AverageProfit DESC

)

WHERE ROWNUM = 1;  


1. Customer name of any customers who have an interest in ALL artists.

SELECT c.FirstName || ' ' || c.LastName AS CustomerFullName

FROM dtoohey.CUSTOMER c

WHERE NOT EXISTS (

SELECT a.ArtistID

FROM dtoohey.ARTIST a

WHERE NOT EXISTS (

SELECT 1

FROM dtoohey.CUSTOMER\_ARTIST\_INT ca

WHERE ca.CustomerID = c.CustomerID

AND ca.ArtistID = a.ArtistID

)

);



# Question 3: Further SQL (15 marks)

1. **Provide ALL of the SQL statements** required to insert the details of the following: A customer “Bill Board”, of 404 Faraway Street, Darwin, NT, 0810, Australia (email: [B.Board@somemailservice.com](mailto:B.Board@somemailservice.com)) has sold a work of art called “Lona Missa” by the renowned Russian artist, Marsha Mellow (b. 1900) to the Gallery (*i.e., the Gallery* ***purchased it from him*)**. It is a unique Oil and collage and is 5.5 x 6.1 inch signed by the Artist. The purchase price was $55000, and the transaction took place on 2nd May 2023. (7 marks)

-- Create a table for customers

CREATE TABLE customers\_new (

id INTEGER PRIMARY KEY,

name VARCHAR2(255) NOT NULL,

address VARCHAR2(255) NOT NULL,

city VARCHAR2(255) NOT NULL,

state VARCHAR2(255) NOT NULL,

zip VARCHAR2(255) NOT NULL,

country VARCHAR2(255) NOT NULL,

email VARCHAR2(255) NOT NULL

);

-- Insert the customer data

INSERT INTO customers\_new (id, name, address, city, state, zip, country, email)

VALUES (1, 'Bill Board', '404 Faraway Street', 'Darwin', 'NT', '0810', 'Australia', 'B.Board@somemailservice.com');

SELECT \* FROM customers\_new;

-- Create a table for artists

CREATE TABLE artists\_new (

id INTEGER PRIMARY KEY,

name VARCHAR2(255) NOT NULL,

birth\_year INTEGER

);

-- Insert the artist data

INSERT INTO artists\_new (id, name, birth\_year)

VALUES (1, 'Marsha Mellow', 1900);

SELECT \* FROM artists\_new;

-- Create a table for artworks

CREATE TABLE artworks (

id INTEGER PRIMARY KEY,

title VARCHAR2(255) NOT NULL,

artist\_id INTEGER NOT NULL,

type VARCHAR2(255) NOT NULL,

width REAL NOT NULL,

height REAL NOT NULL,

FOREIGN KEY (artist\_id) REFERENCES artists\_new(id)

);

-- Insert the artwork data

-- Create a sequence to generate unique values for the id column

CREATE SEQUENCE artworks\_id\_seq;

-- Create a trigger to automatically populate the id column

CREATE OR REPLACE TRIGGER artworks\_id\_trg

BEFORE INSERT ON artworks

FOR EACH ROW

BEGIN

:new.id := artworks\_id\_seq.NEXTVAL;

END;

/

INSERT INTO artworks (title, artist\_id, type, width, height)

VALUES ('Lona Missa', (SELECT id FROM artists\_new WHERE name = 'Marsha Mellow'), 'Oil and collage', 5.5, 6.1);

SELECT \* FROM artworks;

DESCRIBE artworks;

-- Create a table for transactions

CREATE TABLE transactions (

id INTEGER PRIMARY KEY,

artwork\_id INTEGER NOT NULL,

customers\_id INTEGER NOT NULL,

price REAL NOT NULL,

date TEXT NOT NULL,

FOREIGN KEY (artwork\_id) REFERENCES artworks (id),

FOREIGN KEY (customers\_id) REFERENCES customers\_new (id)

);

DESC artworks;

DESC customers\_new;

-- Insert the transaction data

INSERT INTO transactions (artwork\_id, customer\_id, price, date)

VALUES (

(SELECT id FROM artworks WHERE title = 'Lona Missa'),

(SELECT id FROM customers WHERE name = 'Bill Board'),

55000,

'2023-05-02'

);

1. You have been given the following specifications of a simple database for a local company that keeps record of Employees, Departments, and Offices.



**Give the SQL to create the Department table.** You may assume that the Employee and Office tables have already been created, and that the EmpolyeeID and OfficeID colums are of the data type VARCHAR2(8). The status of the department refers to the current state of the department and can only be Active or Inactive. (5 marks)

CREATE TABLE Department (

DepartmentID VARCHAR2(8) PRIMARY KEY,

DepartmentName VARCHAR2(255) NOT NULL,

EmployeeID VARCHAR2(8) NOT NULL,

OfficeID VARCHAR2(8) NOT NULL,

Status VARCHAR2(8) CHECK (Status IN ('Active', 'Inactive')),

FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID),

FOREIGN KEY (OfficeID) REFERENCES Office(OfficeID)

);

1. Your Depratment table must also include a record of the Budget which is allocated each year to cover its running expenses. **Provide the SQL to amend the original table design to allow for this change in requirement.** It is most unlikely (impossible) that a budget would ever be more than 30000 annually. (3 marks)

ALTER TABLE Department

ADD (Budget NUMBER(8, 2) CHECK (Budget <= 30000));

# Question 4: Normalisation (20 marks)

The following question is based upon a Car-Service relation which records the details of transactions occurring in an Automotive Mechanic and Repair business. You may assume the data are representative.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Customer Number** | **Customer NAME** | **Customer DOB** | **Service DATE** | **Car Rego NUMBER** | **Car made** | **Car Model** | **Service FEE** | **Staff NAME** | **Staff speciality** |
| 008 | Justin Case | 8/08/1999 | 9/5/2023 | 1h568 | BMW | Z4 | 200 | Ali | General |
| 123 | Liv Long | 8/08/1999 | 8/4/2022 | 1h985 | Toyota | Corolla | 400 | Chip | Auto body |
| 852 | Sonny Day | 20/10/2004 | 9/5/2023 | 1h895 | BMW | Z4 | 200 | Ali | General |
| 159 | Olive Hoyl | 29/03/1998 | 2/10/2022 | 1h127 | Kia | Soul | Fifty | Holly | Service tech |
| 008 | Justin Case | 8/08/1999 | 13/8/2023 | 1h458 | Haval | H6 | 200 | Ali | General |
| 101 | Corey Ander | 03/01/1991 | 15/9/2022 | 1h325 | Fiat | 500e | 150 | Chip | Auto body |
| 999 | Walter Melon | 30/01/2001 | 8/4/2022 | 1h854 | Holden | Astra | One fifty | Ali | 99999 |
| 101 | Corey Ander | 03/01/1991 | 15/8/2019 | 1h325 | Red | 500e | 260 | Ali | General |

You have been asked to design a relational database for this system. You know that there are problems with the current design and that it will need to be modified to work effectively.

You need to write a 1-2 page report that addresses the following:

1. What are the specific problems associated with the current design and why do they arise?
2. How would you change the current design and how does your new design address the problems you have identified with the current design.

To receive high marks for this question, you will need to demonstrate an understanding of the theories discussed in Topics 1, 2 and 3, how they apply to this problem, and justify the changes you are making to the system. **Simply providing the amended design (even if it is correct) will only attract a small percentage of the marks for this question.**

Report: Improving Current Relational **Databases** for **Car Service** Transactions

**Problems** with the **current design:** The current design lacks **validity** and **reliability and has** the following problems:

Redundancy and **data inconsistency:** One **problem** is **that there is** redundancy and data inconsistency. For example, the customer "Justin Case" is duplicated with the same customer number but different service dates, increasing the risk of errors during data **manipulation.**

**Lack** of **data normalization:** Another problem is the **non-observance of** data normalization principles. The design **requires** proper **organization of** data into logical **units,** resulting in data **redundancy** and efficient storage. **For example,** the customer's name and date of birth **are repeatedly stored** for each transaction, **resulting in** unnecessary **storage space.**

**The absent**of **a primary key:** The design **requires** a primary key, making it **difficult** to ensure data integrity and enforce referential integrity constraints. **Insufficient data integrity constraints: More** appropriate data integrity constraints, such as foreign key constraints, are **needed to maintain** data integrity and consistency. This absence **results in** invalid references and data **exceptions.**

**Proposed** Changes and New **Design: To** address the **issues** identified **in** the current design, the following proposed changes are **proposed:**

**Data Normalization:** The new design will strictly adhere to data **normalization principles.** Data **is** organized into separate tables, eliminating **duplication,** and **improving** data integrity. The proposed **form** will include **customers, cars, services, employees,** and **models.**

**Primary** Key **Implementation:** Each table in the new design will have a primary key that uniquely identifies its records. For **example,** the Customers table **could** have CustomerID as the primary **key, while** the Services table **could have** TransactionID as the primary **key.**

**Foreign** Key **Constraints:** The new design will **use** appropriate foreign key constraints to establish relationships between tables and **ensure** referential integrity. For example, the Services table **might contain** foreign key constraints **on** the Customers, Cars, and Staff **tables.**

**Normalized schema:** The new design will **use** a normalized **schema to organize** related data into separate tables. This approach will eliminate **duplication, reduce** data duplication, and **improve** data consistency and storage efficiency. For example, **car model** information will be stored in a **special car model** table, and **employee knowledge** will be **retrieved from** the **employee** table.

**Additional Constraints: New designs** will **include** additional constraints to ensure data integrity and consistency. These **can** include constraints on attribute domains **(such as validation** dates) **With** these proposed changes, the new design aims to **improve** the **efficiency** and reliability of the database.

Adopting a normalized schema will significantly reduce redundancy, improve data integrity, and optimize storage space. **Additionally,** including primary and foreign keys will **increase** data integrity and **make it easier to create** relationships between entities.

**Additional** constraints will further ensure data consistency and reliability.

**Conclusions:** In **summary,** the proposed changes to the current design address redundancy, data inconsistency, lack of normalization, and **insufficient** data integrity constraints.

With its normalized schema, primary and foreign keys, and additional constraints, the new design will **create** a **stronger and** more efficient relational database for recording **car service** transactions.

In the updated design:

A picture containing text, screenshot, diagram, parallel

Description automatically generated

The "Service Fee" has been moved to the "Services" table to associate it directly with each service.

A new "ServiceTypes" table has been added to store different types of services that can be provided.

The "Staff Speciality" has been moved to the "Staff" table, as each staff member can have a specific specialty.

The "Staff Speciality" has been changed to "StaffSpeciality" (without space) for consistency and to avoid potential issues with queries.

A new "Transactions" table has been added to establish a link between customers, services, and staff for each transaction.

# Question 5: Conceptual Design (25 marks)

“One Way Exit” (OWE transport) is a start-up bus transportation business aimed at the local market of travellers (local travellers wanting a one-way ticket out of town!). OWE transport needs a database to track Buses, customers, fares, bus performance, and staff assignment. Since OWE transport is promoted as “…the fastest way out of town!” individual seats are not assigned. More specific notes about OWE transport are listed below:

* Information about a route includes its unique number, its origin, its destination, and estimated departure and arrival times. To reduce costs, OWE transport only has non-stop bus rides (Journeys) with a single origin and destination.
* Journeys are timetabled for a route on one or more dates with a Bus and a Staff assigned to each journey, and the available seats (remaining capacity i.e., seats not taken) noted. In a staff assignment, the staff ID and the role (position) are noted. It is a business requirement that the number of hours that journey staff (i.e., driver, assistant-driver, bus attendant) are on a journey must be recorded. There is no such requirement for non-journey staff (e.g., route supervisor, admin officers, mechanics etc..).
* Busses have a unique registration number, a made, model, year, a capacity, and a scheduled-service date.
* The service record of a buss includes a unique service number, a date, a description, the registration number of the bus, and the staff (mechanic) responsible for the maintenance/repairs.
* Staff have a unique staff ID, a full name (First and Last name), a contact phone, and a job role.
* Customers have a unique customer ID, a contact phone, and a name (it does not have to be their real name).
* A record is maintained of journey bookings including a unique booking ID, a journey number, a customer number, a date, a fare, and the payment method (can be cash, cheque, crypto currency, or credit card). If the payment is by credit card, a credit card number and an expiration date are part of the booking record.

#### What you have to do:

1. Create an entity-relationship diagram showing the data requirements of the system. Your ERD should be able to be implemented in a relational DBMS. You should use the ERD notation we have been using in the lectures, and should include a legend to explain the notation. You should include attributes in the ERD. The use of a drawing tool such as Visio will make this task easier. However, whichever tool you use, you must copy and paste the ERD into a word-processed document. This is because your tutor might not have access to the tools you have used. **Please note that hand-drawn ERDs are not acceptable.**
2. List and explain any assumptions you have made in creating the data model.
3. Follow the best practices taught in this unit! Refer to the lecture slides and learning materials.

#### Some important things to note:

1. **Part of understanding a system at sufficient enough detail to model well, involves asking appropriate questions. If you are not sure about some detail of the case study, you should ask on the Discussion Forum in LMS.**
2. The University email server strips out any Visio (.vsd) files that are sent; even if they are included in a zip archive. So, if you want to send a draft of your design to your lecturer by email, you will need to change the extension to something other than .vsd (.blah works well) or paste the diagram into a word document.
3. Marks will be allocated to each of the following functional areas:
   1. Legend
   2. Entities
   3. Have you included all required entities in the design?
   4. Have you included specialisation hierarchies where appropriate?
   5. Relationships
      1. Have you designed relationships between the entities that will support the functional requirements?
      2. Are your relationships correctly annotated?
   6. Will the overall design support the enterprise requirements?
   7. Following the best practices taught in the unit e.g., correct naming formats (entities, attributes, others), etc..

As Assignment 2 will require you to implement the OWE database, you will need to take into account the feedback you receive on your conceptual design when commencing your logical and physical designs.

Assumptions:

Each route has a unique number and connects a single origin to a single destination.

Each journey is assigned to a single route and has a unique journey ID.

Buses have a unique registration number and can be scheduled for multiple journeys.

Service records are associated with a specific bus and staff member responsible for maintenance/repairs.

Staff members have a unique ID and can have different job roles.

Customers have a unique ID, and their name can be any string value.

Bookings are linked to a specific journey and customer, and payment methods can be cash, cheque, cryptocurrency, or credit card. Credit card details are only required if the payment method is by credit card and are nullable.

Remaining Capacity in the Journey entity represents the number of available seats for a specific journey.

Relationships:

Route has a one-to-many relationship with Journey.

Journey has a one-to-one relationship with Bus.

Journey has a one-to-one relationship with Staff.

Bus has a one-to-many relationship with ServiceRecord.

ServiceRecord has a one-to-one relationship with Staff.

Journey has a one-to-many relationship with Booking.

Customer has a one-to-many relationship with Booking.

Title: OWE transport entity-relationship diagram

A screenshot of a computer flowchart

Description automatically generated with low confidence