CS990 Database Fundamentals.

CLASSWORK 2: LOGICAL DESIGN AND SQL

THIS IS AN INDIVIDUAL TASK AND MUST BE ACCOMPLISHED WITHOUT COLLABORATION, COLLUSION OR THE SHARING OF SOLUTIONS. FAILURE TO FOLLOW THIS INSTRUCTION WILL RESULT IN DISCIPLINARY ACTION BEING TAKEN.

Aim Of The Assignment:
Your task during the classwork is to design and construct a database and use it to store and retrieve data. You are advised to read the whole document carefully, paying particular attention to the ***Marking Criteria*** section.

Learning Outcomes:
After completing this assignment you will have gained experience of the process of designing a database system, starting from an informal specification and formulating database queries using SQL.

Task:
Read the scenario below carefully. From the description, produce the following:

1. a logical design for the database
2. the SQL code to construct and populate the logical design tables with a minimal amount of data that is needed to carry out the queries specified.
3. The SQL code to answer the queries listed below along with an explanation of the queries.
4. A listing of the output of each of the queries.
5. A critique of your design and implementation.

The database must be built using Oracle.

Scenario: The Art Gallery.

A database is required to store information concerning an art gallery as follows:

The art gallery holds paintings on behalf of their owners so that the paintings can either be permanently on display in the gallery or on loan to exhibitions around the country. Each owner has a name and may have introduced one or more other owners to the scheme. The catalogue of paintings held by the gallery records the name, date of birth and nationality of each artist, the title of the painting and current location within the gallery or the venue of the exhibition at which the painting is currently on show. In addition, the gallery must record a painting's acquisition date. Some paintings are the work of unknown artists. Also stored for paintings in the gallery, is the room number where the painting is located. All paintings are insured with each painting being insured on a single insurance policy. Each insurance policy covers a single painting. The policy number and the insurance value are recorded for each policy.

An enhanced entity relationship model was developed to represent the database and included the following entities, attributes and relationships:

Entity meaning:

ARTIST	Artists who produced the paintings
OWNER	The paintings' owners
INSURANCE POLICY	The policy that covers each painting
PAINTING	The details of the paintings
IN GALLERY	Paintings in the gallery
ON LOAN	Paintings on loan to exhibitions

List of entities and their attributes

ARTIST(A_id, Name, DOB, Nationality)
OWNER(O_id, Name)
INSURANCE POLICY(Pol_id, Insurance_value)
PAINTING(P_id, Title, Aquisition_date)
IN GALLERY(G_id, Room)
ON LOAN(L_id, Exhibition_venue)

Relationships:

Introduced	OWNER	OWNER	1:N	Opt on both
Owns	OWNER	PAINTING	1:N	Oblig on both
Paints	ARTIST	PAINTING	1:N	Oblig on ARTIST Optional on PAINTING
Covers	INSURANCE POLICY	PAINTING	1:1	Oblig on both

Assumptions:

The Gallery only holds details of artists who have painted pictures in the catalogue.
Artists can paint multiple pictures but a painting is the work of one artist.
Owners can own multiple paintings but each painting is owned by one owner,

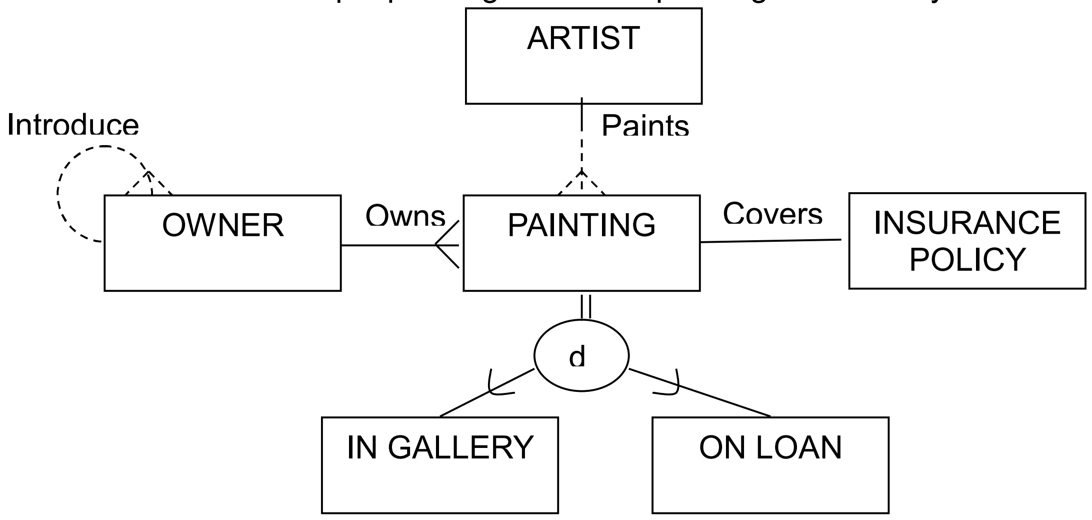

Figure 1. **Enhanced Entity Relationship model diagram.**

Logical design

Table structures must be written down in the following format:
TABLE_NAME(Primary-key-attribute, Non-key-attribute1, Non-key-attribute2.....).

Using the enhanced entity relationship model write down a table structures for each entity taking care that:

- each attribute becomes a column.
- the unique identifier becomes the primary key and is indicated by underlining
- subtype/supertype entities are represented in one of three methods described in the lectures

Use Table CW-1 as a guide to the way of representing the relationships between entities. Write down table structures or modify existing structures to represent relationships in the system.

	1 : 1	1 : N	N : M
Obligatory on neither	New table to represent relationship Post identifiers as candidate keys	New table to represent relationship Post identifiers as candidate keys	New table to represent relationship Post identifiers as candidate keys
Obligatory on one	Post identifier of non-obligatory to obligatory table	New table to represent relationship Post identifiers as candidate keys	-
Obligatory on many	-	Post identifier of "one" table to "many" table	New table to represent relationship Post identifiers as candidate keys
Obligatory on both	Post all attribute into one table	Post identifier of "one" table to "many" table	New table to represent relationship Post identifiers as candidate keys

Table CW-1: Representing relationships in tables.

Creating and loading the database
Use appropriate integrity constraints. Populate each table with a limited set of data i.e. only enough to show that the queries work.

Querying the database
You now need to write some queries on your database. The queries must be useful queries and not artificially constructed simply to fulfil the criteria listed. All queries require a WHERE clause of the form '…WHERE ATTRIBUTE = Value…' to limit the rows returned (Value can be a text, numeric or date value). Write SQL statements that will
(i) carry out a join between two different tables and use the group by clause.
(ii) execute a sub-query.
(iii) execute a correlated-query.
(iv) List the names of owners and the names of the other owners they have introduced. If an owner has not introduced any other owners, their name should be listed in the result.

The output of Oracle SQL queries can be captured in a file by typing:

spool outfile

at the SQL prompt. All screen output is then copied to a file with the name "outfile.lst". The spooling can be stopped by typing:

spool off

at the SQL prompt.

Marking Criteria
Your submission must have a **maximum** of **four** pages (A4, portrait orientation, 11pt font, **NO** coversheets etc, black text on a white background) including **only** the following elements in the order given:

i. A list of the table structures produced by logical design showing the attributes and primary keys. (15 %)
ii. The SQL create statements (including the specification of integrity constraints) for creating the tables. (25%)
iii. The SQL insert statements for populating the tables with a small sample of data. (15%)
iv. The SQL queries together with a narrative explanation of queries (i) – (iii) (do not paraphrase the SQL commands) and its output. (25%)
v. A 200 word critique of your database, highlighting the strengths and weaknesses of your solution and giving reasons for decisions that you have taken in the design and implementation. (20%)

The solution **must** be typed (not hand written). Items (i) to (v) must be combined into a single document in the order shown and submitted through the link on the class Myplace page. The document must not be zipped and must not contain screen shots or other elements that cannot be processed by Turnitin. If a Turnitin error (eg " Turnitin has returned an error with your submission: Your submission

must contain 20 words or more ") is returned when you upload the document, it means that the PDF contains text as images. You must OCR (optical character recognition) the text so that it can be processed by Turnitin. Failure to follow these instructions may result in marks being deducted.
All work submitted will be subjected to Turnitin similarity checking.

Due Date:
The classwork should be submitted only as a **single pdf** file on Myplace by 12.00noon on Monday 9th March 2020.

This exercise is worth 25% of the overall marks for this module