MONASH University

FACULTY OF
INFORMATION
TECHNOLOGY

FIT3171 Databases - Assignment 2

Creating, Populating and Manipulating Databases - Monash Art Gallery (MAG)

Purpose	Students will be asked to implement, via SQL, a small database in the Oracle RDBMS from a provided logical model case study, followed by the insert of appropriate data to the created tables. Once populated the database will be used to carry out specified DML commands and make specified changes to the database structure via SQL. Students will then use SQL and NoSQL to write queries to produce specified output and use PL/SQL to enforce business rules. This task covers learning outcomes: 1. Apply the theories of the relational database model. 3. Implement a relational database based on a sound database design. 4. Manage data that meets user requirements, including queries and transactions. 5. Contrast the differences between non-relational database models and the relational database model. 6. Develop programming structures within a database backend.
Your task	This is an open book, **individual task**. The final output for this task will be a set of tables and data implemented in the Oracle RDBMS. In addition students will create a set of relational (Oracle) and non relational (MongoDB) queries which meet the user requirements, and code procedures and triggers to enforce business rules.
Value	**40%** of your total marks for the unit
Due Date	Wednesday, **14th February 2024**, 11:55 PM - New Deadline (note: staff support is unavailable after business hours)
Submission	● Via Moodle Assignment Submission ● FIT GitLab check ins will be used to assess history of development
Assessment Criteria	● Application of relational database principles. ● Handling of transactions and the setting of appropriate transaction boundaries. ● Application of SQL statements and constructs to create and alter tables including the required constraints and column comments, populate tables, modify existing data in tables, and modify the "live" database structure to meet the expressed requirements (including appropriate use of constraints). ● Application of SQL select statements to produce outputs that meet user requirements. ● Mapping of relational database data into non relational database data structure.

	● Application of MongoDB operations to produce outputs that meet user requirements. ● Development of Procedures and Triggers (PL/SQL) to enforce business rules and data integrity
Late Penalties	● 10% deduction per calendar day or part thereof for up to one week ● Submissions more than 7 calendar days after the due date will receive a mark of zero (0) and no assessment feedback will be provided.
Support Resources	See Moodle Assessment page
Feedback	Feedback will be provided on student work via: ● general cohort performance ● specific student feedback ten working days post submission ● a sample solution

INSTRUCTIONS

Monash Art Gallery (MAG) is a company which accepts artworks from artists and offers these artworks to galleries around the country for display and potential sale. A registered MAG customer may, while the artwork is being displayed, decide to purchase the item. MAG charges twenty percent of the price at which the artwork is sold as their commission (note this is not recorded as part of the model). The gallery is paid a standard percentage of the sale price as its commission, this percentage is negotiated per gallery and so may vary from gallery to gallery (this gallery percentage is recorded as part of the model). The remainder of the sale price goes to the artist.

MAG assigns an artist code to each artist which the company represents. MAG records the artist's name, their contact address and telephone number (not all artists supply a telephone number). When an artist has completed an artwork which they wish to sell through MAG, they contact the company and offer the work to be sold by MAG.

All artworks accepted by MAG are assigned an artwork number specific to a particular artist. For example artist 1234 will have artworks 1, 2, 3 etc and artist 4567 will also have artworks 1, 2, 3 etc. The title of the artwork, the date the work was accepted into the MAG system and the minimum payment which the artist is prepared to accept for their work is recorded. The work may be sold for any price above this minimum payment such that the artist receives at least this minimum amount and the gallery and MAG commission requirements are satisfied.

A gallery is identified by a gallery id. MAG records the name of the gallery, the gallery manager's name, the address of the gallery and the contact phone number for the gallery (all galleries are required to provide a unique contact number).

Art collectors who are interested in purchasing an artwork must register with MAG as a customer before they are able to make any purchase. Each customer is identified by their customer id. The customer's name, address and contact phone number are recorded. If the customer is a business customer their business name is also recorded.

A gallery considers the artworks which MAG has on offer and then requests one or more artworks and displays the artwork(s) in its gallery with the intention of generating a sale. The date at which the display starts is recorded. If the item generates little interest then the gallery will return the artwork to MAG. At a later date, after it is back at MAG, the gallery might request and display the same artwork again.

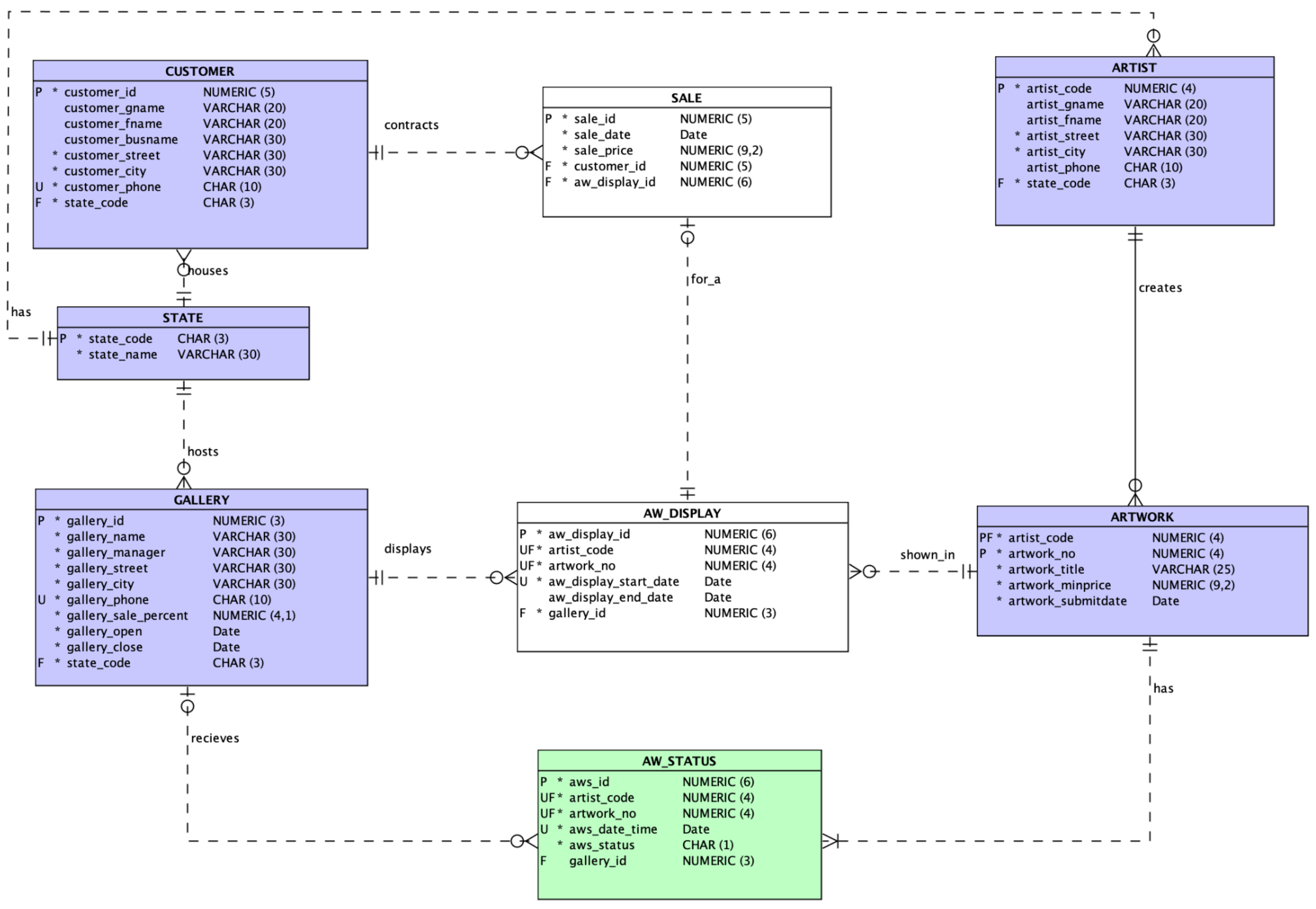
When an artwork from MAG stock is sold, the sale is assigned a unique sale id. The artwork, date of the sale, sale price and customer who purchased the item is recorded. MAG is interested in identifying which display of the artwork generated the sale.

To allow tracking the status of an artwork, MAG would like to have recorded the status of each artwork at the current point in time and its full status history (i.e. all statuses which have been recorded for the artwork must be permanently recorded). MAG would like to record five possible values for this status:

- W - in MAG storage at the MAG central warehouse
- T - in transit (being shipped to/from a gallery), *include to/from which gallery id*
- G - located at a gallery, *include gallery id*
- S - sold, or
- R - returned to the artist

These five possible values are fixed and will not need to be extended. Once an artwork has been returned to an artist, it cannot be accepted back into the MAG system. For such returned items, the artwork entry remains in the MAG system ie. its row in the ARTWORK table is not deleted.

Based on these requirements a data model has been created for MAG:


CUSTOMER
P * customer_id NUMERIC (5)
customer_gname VARCHAR (20)
customer_fname VARCHAR (20)
customer_busname VARCHAR (30)
* customer_street VARCHAR (30)
* customer_city VARCHAR (30)
U * customer_phone CHAR (10)
F * state_code CHAR (3)
contracts
SALE
P * sale_id NUMERIC (5)
* sale_date Date
* sale_price NUMERIC (9,2)
F * customer_id NUMERIC (5)
F * aw_display_id NUMERIC (6)
ARTIST
P * artist_code NUMERIC (4)
artist_gname VARCHAR (20)
artist_fname VARCHAR (20)
* artist_street VARCHAR (30)
* artist_city VARCHAR (30)
artist_phone CHAR (10)
F * state_code CHAR (3)
houses
for_a
creates
has
STATE
P * state_code CHAR (3)
* state_name VARCHAR (30)
hosts
GALLERY
P * gallery_id NUMERIC (3)
* gallery_name VARCHAR (30)
* gallery_manager VARCHAR (30)
* gallery_street VARCHAR (30)
* gallery_city VARCHAR (30)
U * gallery_phone CHAR (10)
* gallery_sale_percent NUMERIC (4,1)
* gallery_open Date
* gallery_close Date
F * state_code CHAR (3)
displays
AW_DISPLAY
P * aw_display_id NUMERIC (6)
UF * artist_code NUMERIC (4)
UF * artwork_no NUMERIC (4)
U * aw_display_start_date Date
aw_display_end_date Date
F * gallery_id NUMERIC (3)
shown_in
ARTWORK
PF * artist_code NUMERIC (4)
P * artwork_no NUMERIC (4)
* artwork_title VARCHAR (25)
* artwork_minprice NUMERIC (9,2)
* artwork_submitdate Date
has
recieves
AW_STATUS
P * aws_id NUMERIC (6)
UF * artist_code NUMERIC (4)
UF * artwork_no NUMERIC (4)
U * aws_date_time Date
* aws_status CHAR (1)
F gallery_id NUMERIC (3)

The schema/insert file for creating this model (mag_initialSchemaInsert.sql) is available in the archive ass2_student.zip - this file partially creates the Monash Art Gallery tables and populates several of the tables (those shown in purple and green on the supplied model) - you should read this schema carefully and be sure you understand the various data requirements.

IMPORTANT points for you to observe, when completing this assignment, are:

1. The ass2-student.zip archive also contains seven script files for you to code your answers in, **you should ensure these files are regularly pushed to GitLab server so a clear development history is available for the marker to verify (a minimum of fourteen pushes are required).** In each file, you ***must*** fill in the header details with your name and student ID before beginning any work. Your SQL script files **must not include any SPOOL or ECHO commands**. Although you might include such commands when testing your work **they must be removed before submission** (a -10 marks grade penalty will be applied if your documents contain spool or echo commands).

2. You are free to make assumptions if needed. However, ***your assumptions must align with the details here and in the Ed Assignment 2 forum*** and must be clearly documented (see the required submission files).

   REMEMBER you must keep up to date with the Ed Assignment 2 forum where further clarifications may be posted (this forum is to be treated as your client). **Please be careful to ensure you do not post anything which includes your reasoning, logic or any part of your work to this assignment forum as doing so violates Monash plagiarism/collusion rules.**

3. Views ***must not*** be used in arriving at any solutions for the tasks you are required to complete as part of this assessment.

4. In handling dates with SQL, the default date format must not be assumed; ***you must make use of the TO_DATE and TO_CHAR functions where appropriate***.

5. **ANSI joins must be used** where the joining of tables is required.

6. In completing the following tasks, you must ***design your test data so that you always get output for the queries specified below*** **-** this may require you to add further data as you move through completing the required tasks. Such extra data MUST be added as part of Task 2 (ie. as part of your load of test data). ***Queries that are correct but do not produce any output ("no rows selected" message) using your test data will lose 50% of the marks allocated***, so you should carefully check your test data and ensure it thoroughly validates your SQL queries.

Steps for working on Assignment 2

1. Download the Assignment 2 Required Files zip archive (ass2-student.zip) from Moodle

2. Extract the zip archive and place the contained files in your local repository in the folder
   /Assignments/Ass2
   Do not add the zip archive to your local repo.

3. Examine the extracted files i.e. read carefully through the files and be sure you understand their content.

4. In each of the answer supplied scripts fill in the header details with your name and student ID. Then add, commit and push them to the FITGitLab server.

5. Run mag_initialSchemaInsert.sql from the supplied zip archive to set up the initial state of the database

6. Write your answer for each task in its respective file (e.g. write your answer for task 1 in T1-mag-schema.sql and so on).

7. Save, add, commit and push the file/s **regularly** while you are working on the assignment

8. Finally, when you have completed all tasks, upload all required files from your local repository to Moodle. Check that the files you have uploaded are the correct files (download them from Moodle into a temporary folder and check they are correct). After you are sure they are correct, submit your assignment.

****

For all assignment tasks, **your final script must run as a script without errors**. **Any task script which runs with an error will receive a maximum grade of half of the task's marks -1**. For example if your task 1 script runs with an error, regardless of the contained code, your maximum grade will be 9/2 => 4.5 - 1 = 3.5 marks. This will be applied even if the error is simply a forgotten semicolon. Thus **please carefully check that your final scripts for all tasks run without error**.

In arriving at your solutions for assignment 2 you are ONLY permitted to use the SQL/PL-SQL/NoSQL structures and syntax **which have been covered within this unit**:

- Topic 6 Workshop and Applied 7 - Creating & Populating the Database
- Topic 7 Workshop and Applied 8 - Insert, Update, Delete (DML) and Transaction Management
- Topic 8 Workshop and Applied 9 - SQL Part I - Basic and Intermediate
- Topic 9 Workshop and Applied 10 - SQL Part II - Advanced
- Topic 10 Workshop and Applied 11 - PL/SQL
- Topic 11 Workshop and Applied 12 - Non Relational Database

SQL syntax and commands outside of the covered work, as detailed above, will NOT be accepted/marked.

Views must not be used in completing these tasks.

You must also keep up to date with the Ed Assignment 2 forum where further clarifications may be posted. Please be careful to **ensure you do not publicly post anything which includes your reasoning, logic, or any part of your work to this forum**, *doing so violates Monash plagiarism/collusion rules* and has significant academic penalties. Attend a consultation session, use a private Ed post, or email your allocated tutor to raise such questions.

****

GIT STORAGE

Your work for these tasks MUST be saved in your individual local working directory (repo) in the Assignment 2 folder and **regularly pushed to the FIT GitLab server to build a clear history of development of your approach**. A minimum of fourteen pushes to the FIT GitLab server is required (2 pushes per file). Please note fourteen pushes is a *minimum*, in practice we would expect significantly more. All commits must include a **meaningful commit message** which clearly describes what the particular commit is about and **must be correctly assigned to your valid GitLab author**.

You must regularly check that your pushes have been successful by logging in to the web interface of the FIT GitLab server; you must not simply *assume* they are working. Before submission, via Moodle, you **must** log in to the web interface of the GitLab server and ensure your submission files are present on the GitLab server and that their names are unchanged.

****

Assignment Tasks

TASK 1: DDL [12 mks]

ENSURE your **id and name are shown at the top of any file you submit**.
For this task you are required to add to **T1-mag-schema.sql**, the CREATE TABLE and CONSTRAINT definitions which are missing from the supplied partial schema script in the positions indicated by the comments in the script.

The table below provides details of the meaning of the attributes in the missing two tables. You **MUST** use exactly the same relation and attribute names as shown in the data model above to name the tables and attributes which you add. The attributes must be in the same order as shown in the model. These new DDL commands *must be hand-coded, not generated in any manner (generated code will not be marked)*.

Table name	Attribute name	Meaning
SALE		
	sale_id	Identifier for sale
	sale_date	Date sale was closed
	sale_price	Price customer paid for artwork
	customer_id	Identifier for customer
	aw_display_id	Identifier for aw_display
AW_DISPLAY		
	aw_display_id	Identifier for aw_display
	artist_code	Identifier for artist
	artwork_no	Identifier for artwork within this artist
	aw_display_start_date	Date this artwork display in the gallery began
	aw_display_end_date	Date this artwork display in the gallery ends. The display end date must be after the display start date.
	gallery_id	Identifier for Gallery

To test your code you will need to first run the provided script mag_initialSchemaInsert.sql to create the other required tables. mag_initialSchemaInsert.sql contains at the head of the file the drop commands for ***all*** tables in this model. If you have problems with task 1 simply rerun mag_initialSchemaInsert.sql which will cause all tables to be dropped and correct the issues in your script. **DO NOT add drop table statements to T1-mag-schema.sql**

TASK 2: INSERT [20 mks]

Before proceeding with Task 2, you must ensure you have run the file mag_initialSchemaInsert.sql (which **must not be edited in any way**) followed by the extra definitions that you added in Task 1 above (T1-mag-schema.sql).

The new database system was initiated on the 1st June 2023, at this point in time all artworks held by MAG were recorded as being present in the Warehouse (i.e. assigned a status W as of the 1st June 2023).

Load the AW_STATUS, AW_DISPLAY and SALE tables with **your own test data** using the supplied **T2-mag-insert.sql** script file, and SQL commands which will insert *as a minimum*, the following sample data:

(i) 10 AW_DISPLAY entries
- Involved at least 2 different display galleries
- Included at least 3 different artwork display start dates
- Included at least 5 different artworks
- Involved at least 2 different artists

(ii) 4 SALE entries

(iii) Required AW_STATUS entries to support the AW_DISPLAY and SALE data you add

In adding this data, you must ensure that the test data thoroughly tests the model as supplied, to ensure your schema is correct (you are not required to submit or code fail tests, all insert statements must execute correctly).

Your inserted data must conform to the following rules:
1. You may treat all the data that you add as a single transaction since you are setting up the initial test state for the database.
2. The primary key values for this data should be hardcoded values (ie. **NOT** make use of sequences) and must consist of values below 100.
3. Dates used must be chosen between the 1st June 2023 and 30th December 2023.
4. The data added must be sensible (eg. the display start date should be after the artwork submitted date; the transit time should be a minimum of 2 hrs; ; etc).

For this task **ONLY,** Task 2, you may look up and include values for the loaded tables/data directly where required. However, if you wish, you can still use SQL to get any non-key values.

In carrying out Task 2 you must not modify any data or add any further data to the tables which were populated by the mag_initialSchemaInsert.sql script except the AW_STATUS table (ie. you must not modify any data in the purple tables but may, if needed, add data to the green table).

Design your test data so that you get output for the SQL scripts/queries specified below - this may require you to add further data as you move through completing the required tasks.

TASK 3: DML [23 mks]

Your answers for this task must be placed in the SQL file T3-mag-dml.sql

For this, and *all subsequent Tasks* **you are NOT permitted to**:

- manually lookup a value in the database, obtain its primary key or manually obtain the highest/lowest value in a column,
- manually calculate values external to the database, e.g. on a calculator and then use such values in your answers. ***Any necessary calculations must be carried out as part of your SQL code***, or
- assume any particular contents in the database - rows in a table are potentially in a constant state of change

Your answers must recognise the fact that you are dealing with only a very small sample snapshot of a multiuser database, as such you must operate on the basis that there will be ***more data in all of the tables of the database than you currently have access to and thus data will be in a constant state of change. Your answers must work regardless of the extra quantity of this extra "real" data and the fact that multiple users will be operating in the tables at the same time. You must take this aspect into consideration when writing SQL statements.***

For any following SQL tasks, **your SQL must correctly manage transactions and use sequences to generate new primary keys for numeric primary key values** (under no circumstances may a new primary key value be hardcoded as a number or value).

You must ONLY use the data as provided in the text of the questions.

For Task 3 you are required to complete the following sub-tasks in the same order they are listed. Where you have been supplied with a string contained in *italics*, such as *Karma Art* you may search in the database using the string *as listed*. Where a particular case (upper case, lower case, etc.) for a word is provided *you must only use that case*. When a name is supplied you may break the name into first name and last name, for example, Septi Hartati can be split into Septi and Hartati, again note that the case must be maintained as it was supplied. **Failure to adhere to these requirements, such as by changing the case of a provided string will result in grade penalty**.

(a) Oracle sequences are going to be implemented in the database for the subsequent insertion of records into the database for the AW_DISPLAY, AW_STATUS and SALE tables.

Provide the CREATE SEQUENCE statement to create three sequences which could be used to provide primary key values for the AW_DISPLAY, AW_STATUS and SALE tables. All sequences should start at 100 and increment by 10. Immediately prior to the create sequence commands, place appropriate DROP SEQUENCE commands so they will cause the sequences to be dropped before being created if they exist. Please note that there can only be these three sequences introduced and used in Task 3.

[1 mark]

Question 3b, 3c, 3d and 3e are related questions. You can use the information provided below as needed in any part of Task 3.

(b) Suppose it is now 11 AM on 30th December 2023, a new artwork called *Shattered glass* has just been received by the MAG central warehouse from the artist with artist code 1. The minimum payment this artist is prepared to accept for this artwork is $25000. You may assume that artist code 1 only has one artwork titled *Shattered glass*. Take the necessary steps in the database to record the required entries for this new arrival (for this task you may make changes to the provided ARTWORK table).

[4 marks]

(c) The above artwork titled *Shattered glass* is then transited from the MAG central warehouse to Art Smart gallery (ph: 0490556646) on 1st January 2024 at 1:00 PM.

2 hours and 30 minutes later, the gallery informed MAG by a phone call that the artwork arrived safely at their location. On the next day, the gallery places the artwork on display for a duration of 14 days.

Make these required changes to the data in the database.

[6 marks]

(d) On the 4th January 2024 at 11:30 AM, the artwork titled *Shattered glass* was sold to a customer named Lois Hawkshaw (ph: 0458708402) at the price of $29,499.99. The gallery was then instructed to stop displaying this artwork on this sale date.

Make these required changes to the data in the database.

[6 marks]

(e) Before delivering the sold artwork to the customer's address, MAG was informed that the customer was involved in illegal financial activities (the customer was accused of involvement in a money laundering case). As a result, MAG management decided to cancel the sale (remove the sale record and related status data from the system) and have the payment refunded to the customer. The artwork was placed back in the gallery for continuing display for sale based on its original gallery display start date.

Make these required changes to the data in the database.

[6 marks]

TASK 4: DATABASE MODIFICATIONS (15 marks):

Your answers for these tasks (Task 4) must be placed in the supplied SQL script **T4-mag-mods.sql**

The required changes must be made to the "live" database (the database *after* you have completed tasks 1, 2 and 3) not by editing and executing your schema file again. Before carrying out the work below, please ensure that you have completed tasks 1, 2 and 3 above.

If in answering these questions you need to create a table, please place a drop table statement immediately prior to your create table statement.

(a) MAG would like to be able to easily determine the total number of artworks that each artist has submitted which have been sold or are currently available for sale (i.e., not including the ones that have been returned).

Based on the data which is currently stored in the system, this attribute must be initialised to the correct current number of artworks. Add a new attribute which will record this requirement.

As part of your solution provide appropriate select and desc statements to show the changes you have made. Select to show any data changes which have occurred and desc tablename e.g. `desc customer` to show any table structural changes.

[6 marks]

(b) MAG has observed that many customers seem to purchase artworks from the same artist. They want to record the number of artworks that each customer has purchased from each artist so as to assist with future marketing opportunities for particular artists.

Change the database to meet this requirement. Your solution must record the correct number of artworks that each customer has purchased from each artist based on the data that is currently stored in the database.

As part of your solution provide appropriate select and desc statements to show the changes you have made. Select to show any data changes which have occurred and desc tablename e.g. `desc customer` to show any table structural changes.

[9 marks]

TASK 5: PL/SQL [15 mks]

Your answers for this task (Task 5) must be placed in the supplied SQL script **T5-mag-plsql.sql.**

(a) From this point onwards, MAG wants to automatically check whether the sale price is valid when an artwork is being sold.

The minimum selling price includes the commission payment to the gallery where it was sold, the commission payment to MAG and the minimum payment to the artist. For example, an artwork has been provided by an artist with an indicated minimum payment to the artist of $1,400.00. This artwork is to be sold by a gallery with a 10% commission. An estimate of the minimum selling price must include the minimum payment to the artist, the gallery commission and the MAG commission (20%). Here since 30% is commissions, the $1,400.00 must represent 70% of the sold price, thus the actual selling price would need to be at least $2,000.00 (artist payment $1,400.00, gallery commission $200.00 and MAG commission $400.00) ie. $2,000.00 is the minimum sales price.

Create **one** trigger to check if the inserted sale price is above the required minimum selling price when a SALE record is inserted. If the sale price is below the required minimum selling price, the sale must be cancelled (i.e. must not be inserted into the system).

[6 marks]

(b) Write a stored procedure called **prc_insert_sale** which handles the insert of a new artwork sale. The procedure only handles one SALE insertion at a time.

The procedure requires:

- Four input arguments
  - p_customer_id (ie. the customer id of the purchaser)
  - p_artist_code
  - p_artwork_no
  - p_sale_price
- One output arguments
  - p_output

The procedure must check if the customer is a valid customer and whether the inputted artwork is currently on display. Once these values are checked, the procedure must insert/update the necessary records in the relevant tables. The sale date is the date when the record is inserted into the system. You may use the sequences created in Task 3a to generate the PK values.

The structure of the procedure has been provided in the T5-mag-plsql.sql. <u>You must not change this structure</u> (i.e. you must not change the parameter names and order).

[9 marks]

For each of these PL/SQL questions, as part of your answer, you must create a set of SQL commands which will demonstrate the successful operation of your trigger/stored procedure (test harness) - these tests are part of the awarded marks for each question. Place these commands below your trigger/stored procedure definition for each of the tasks. **You may do manual look up** when writing the test harness.

Ensure your trigger/stored procedure definition finishes with a slash(/) followed by a blank line as detailed in the topic 10 workshop and applied 11. In addition, when coding your triggers/procedure, you must provide output messages where appropriate.

TASK 6: MongoDB [10 mks]

Your answers for this task (Task 6) must be placed in the supplied sql file **T6-mag-json.sql** and the supplied MongoDB script file **T6-mag-mongo.mongodb.js**

(a) Write an SQL statement in **T6-mag-json.sql** to generate a collection of JSON documents using the following structure/format. Each document in the collection represents an artist and their submitted artwork details. The list must only include artists who submitted at least one artwork. Note that the _id in this structure is the artist code, the no_of_artworks is the number of artworks submitted by the artist.

```
{
    "_id": 7,
    "name": "Weston Stearndale",
    "address": {
      "street": "39512 Kipling Road",
      "city": "Leongatha",
      "state": "Victoria"
    },
    "phone": "0417905216",
    "no_of_artworks": 2,
    "artworks": [
      {
        "no": 1,
        "title": "Orange Veils",
        "minimum_price": 12900
      },
      {
        "no": 2,
        "title": "Saint Francis of Assisi",
        "minimum_price": 34536.9
      }
    ]
  }
```

[5 marks]

Write the MongoDB commands for the following questions, 6(b) - 6(d), in the supplied MongoDB script file named **T6-mag-mongo.mongodb.js**.

(b) Create a new collection and insert all documents generated in 6(a) above into MongoDB. Provide a drop collection statement right above the create collection statement. You may pick any collection name. After the documents have been inserted, use an appropriate db.find command to list all the documents you added.

[1 mark]

(c) List the name and phone number of all artists who submitted at least two different artworks OR have at least one artwork with a minimum price above $40000.

[2 marks]

(d) Weston Stearndale ("_id": 7) submitted another artwork. This is Weston's third artwork (ie. number 3) and the title of the artwork is "Purple Sky". MAG set the minimum price as $25000.

Use an appropriate db.find command before making the change so that you illustrate which document/s will be changed.

Write the necessary MongoDB commands to add this new artwork into the collection.

Use an appropriate db.find command after making the change so that you illustrate/confirm the change which was made.

[2 marks]

Submission Requirements

Due Date: **Wednesday, 14th February 2024 at 11:55 PM**

*Please note, if you need to resubmit, you **cannot** depend on your staffs' availability, for this reason, please be VERY CAREFUL with your submission.* **It is strongly recommended that you submit several hours before this time to avoid such issues.**

For this assignment there are seven files you are **required** to submit:

- T1-mag-schema.sql
- T2-mag-insert.sql
- T3-mag-dml.sql
- T4-mag-mods.sql
- T5-mag-plsql.sql
- T6-mag-json.sql
- T6-mag-mongo.mongodb.js

If you need to make any comments to your marker/tutor please place them at the head of each of your solution scripts/answers in the "Comments for your marker:" section.

Do not zip these files into one zip archive, submit seven independent SQL scripts. The individual files must also have been pushed to the FIT GitLab server with an appropriate history as you developed your solutions.

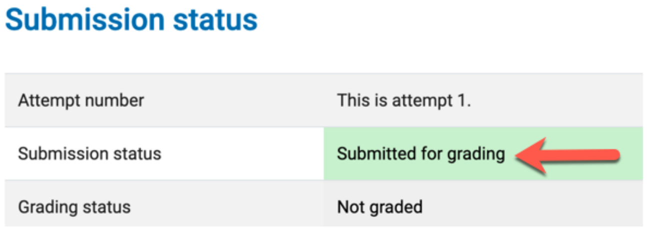
Late submission will incur penalties at the rate of -10 marks for every 24 hours the submission is late.

The seven files must also exist in your FITGitLab server repo and *show a clear history of development* (a **minimum** of two pushes per file).

Please note we **cannot mark any work on the GitLab Server**, you need to ensure that you submit correctly via Moodle since it is only in this process that you complete the required student declaration without which work **cannot be assessed**.

It is your responsibility to ENSURE that the files you submit are the correct files - we strongly recommend after uploading a submission, and prior to actually submitting, that you download the submission and double-check its contents.

Your assignment **MUST** show a status of "Submitted for grading" before it will be marked.



Submission status

Attempt number	This is attempt 1.
Submission status	Submitted for grading
Grading status	Not graded

If your submission shows a status of "Draft (not submitted)" it will not be assessed and **will incur late penalties after the due date/time**.

Please ***carefully*** read the documentation under the "Assignment Submission" on the Moodle Assessments page which covers things such as extensions and resubmission.

Marking Guide

Submitted code will be assessed against an optimal solution for these tasks. In some tasks where SQL is involved there are often several alternative approaches possible, such alternatives will be graded based on the code successfully meeting the briefs requirements. If it does, the answer will be accepted and graded appropriately.

Marking Criteria	Items Assessed
TASK 1 DDL 12 marks	
DDL Creation of tables	**Maximum 6 marks - Create table:** ● Marks awarded for correct table DDL ● Marks awarded for correct attributes/data types ● Marks awarded for correct PK definition ● Marks awarded for correct use of column comments ● Mark penalty applied if different table/attribute names used than expressed in the supplied data model ● Mark penalty applied if different order of attributes used than expressed in the supplied data model ● No marks awarded if generated schema used
DDL implementation of non-PK database constraints	**Maximum 6 marks - Non-PK Constraints:** ● Marks awarded for correct implementation of non-PK constraints
TASK 2 Data Insert 20 marks	
Insert of required items test data	**Maximum 10 marks- Insert of data:** ● Marks awarded for correct insert of required data ● Marks awarded for correct management of transactions
Insert of valid test data	**Maximum 10 marks - Valid data inserted:** ● Marks awarded for validity of data inserted ○ meets the requirements expressed in the assignment brief ● Marks awarded for correct management of dates when inserting
Task 3 DML 23 marks	
	Maximum 23 marks - Satisfy brief requirements: ● Marks awarded (a) - (e) for SQL code which meets the expressed requirement ● Mark penalty applied if commit not used appropriately ● Mark penalty applied if date handling and string database lookups not managed correctly

Task 4 Database Modifications 15 marks	
	Maximum 15 marks - Satisfy brief requirements: ● Marks awarded (a) - (b) for SQL code which meets the expressed requirement (including appropriate use of constraints). In making these modifications there must be no loss of existing data or data integrity within the database. ● Mark penalty applied if commit not used appropriately ● Mark penalty applied if column comments not used where required
Task 5 PL/SQL 15 marks	
	Maximum 15 marks - Satisfy brief requirements: ● Marks awarded (a) - (b) for PL/SQL code which meets the expressed requirement. ● Marks awarded for writing a test harness for each question (a) - (b) which includes both successful and failed tests (all errors your code raises must be tested). ● Mark penalty applied if no output messages are provided where appropriate ● Statements which do not execute correctly in Oracle will be awarded a maximum of 50% of the available marks less 1 mark. For example, if a question is worth 6 marks and runs with an error in SQL the *maximum* mark awarded will be 2 marks
Task 6 Non Relational Database Queries - MongoDB 10 marks	
	Maximum 10 marks - Satisfy brief requirements: ● Maximum of 5 marks awarded for creation of a JSON document which matches the supplied document format ● Marks awarded, as listed, (b) - (d) for MongoDB code which meets the expressed requirement ● Mark penalty applied if field names and predicates (such as "&eq") are not enclosed in double quotes ● Statements which do not execute correctly in MongoDB will be awarded a maximum of 50% of the available marks less 1 mark. For example, if a question is worth 6 marks and runs with an error in MongoDB the *maximum* mark awarded will be 2 marks
Correct use of Git 5 marks	
	● Marks awarded for a minimum of fourteen pushes (two per file) showing a clear development history of the work for Assignment 2 ● Marks awarded for correct Git author details used in pushes ● Marks awarded for the use of meaningful commit messages (i.e. not blank or of the form "Push1")

Penalties	
Use of ● VIEWs ● SET ECHO or SPOOL commands, and/or ● PL/SQL	Use of VIEWS, inclusion of SET ECHO/SPOOL, and/or PL/SQL commands (other than in Task 5) will result in a ***grade deduction of 10 marks being applied.*** **PL/SQL can only be used in Task 5**
Late submission	**-10 marks for each 24 hours late or part thereof**

Final Assignment Mark Calculation

Total: 100 marks, recorded as a grade out of 40