

FIT9132 Introduction to Databases - Assignment 2A

Creating, Populating and Manipulating Database - Monash New Smile

Purpose	<p>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. This task covers learning outcomes:</p> <ol style="list-style-type: none"> 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.
Your task	<p>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</p>
Value	<p>25% of your total marks for the unit</p>
Due Date	<p>Thursday, 12th October 2023, 4:30 PM</p>
Submission	<ul style="list-style-type: none"> ● Via Moodle Assignment Submission ● FIT GitLab check ins will be used to assess history of development
Assessment Criteria	<ul style="list-style-type: none"> ● 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).
Late Penalties	<ul style="list-style-type: none"> ● 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	<p>See Moodle Assessment page</p>
Feedback	<p>Feedback will be provided on student work via:</p> <ul style="list-style-type: none"> ● general cohort performance ● specific student feedback ten working days post submission ● a sample solution

INSTRUCTIONS

Your task for this assignment is to create, populate and manipulate a database which can be used to support the activities of a dental clinic called Monash New Smile (MNS).

Monash New Smile (MNS) provides dental services such as fillings, scalings, extractions etc to its patients. For each service, MNS records a service code, service description and the MNS standard fee for this service. The actual fee charged to a patient for a particular service may be varied from this standard fee. These services are provided by a range of professionals (providers) that MNS employs. Each provider is assigned a provider code. MNS records the provider's title, name, the specialisation, and the room number where they normally treat patients. Each provider may have a particular specialisation (one only), but some providers may not have any specialisation. Some services require a particular provider while other services are able to be provided by a number of providers.

MNS patients are assigned a patient number. The company records the patient's name, date of birth, the residential address, contact phone number and contact email address (all patients are required to provide a contact number (mobile phone) and an email address for the purpose of confirming appointments). Patients must also provide another person's name and phone number for emergency purposes, which is also recorded.

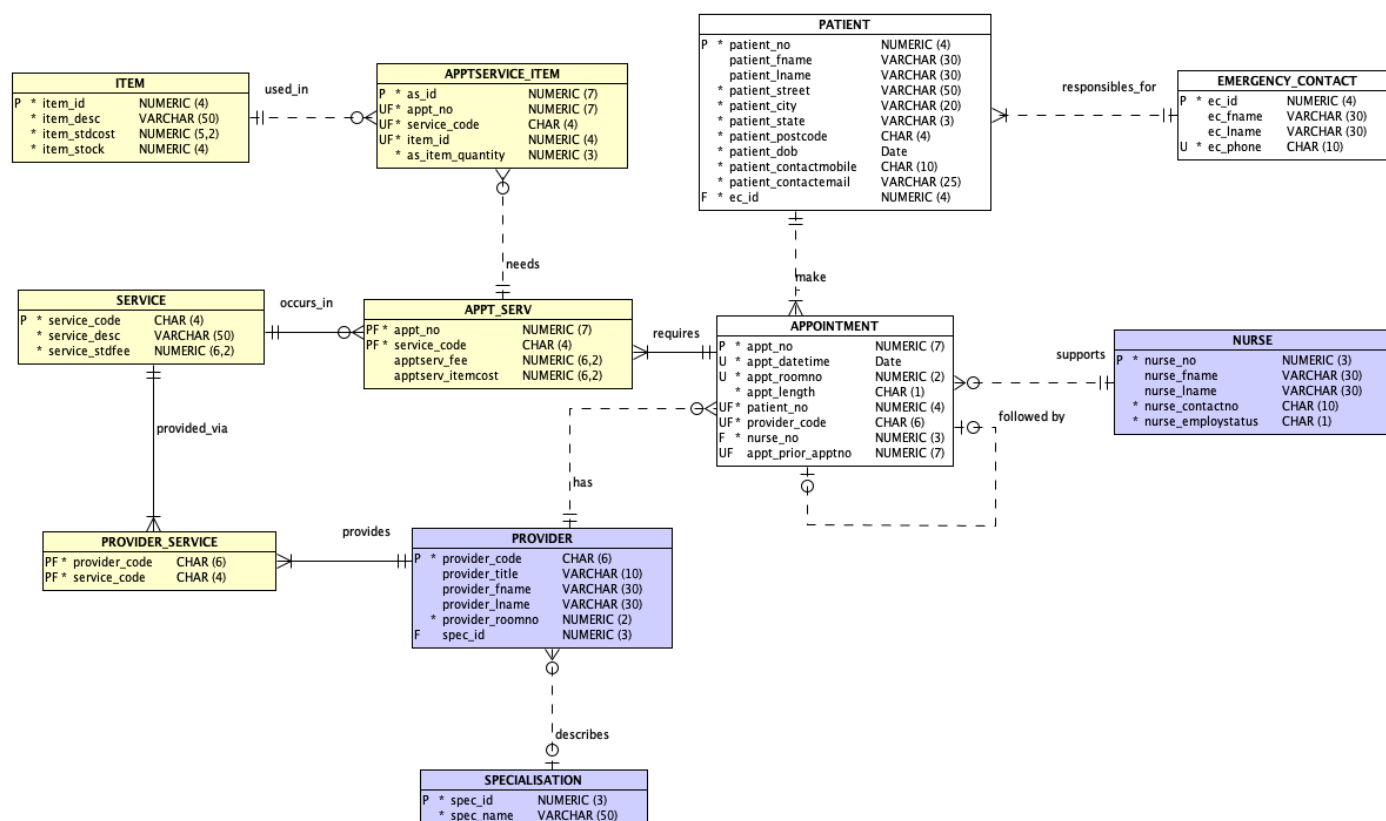
Patients contact Monash New Smile and make appointments to see a provider. Each appointment is booked with only a single provider, the provider is assigned when the appointment is first recorded. The system also ensures that patients, providers, and rooms are not double booked (e.g. a patient cannot have two scheduled appointments at the same date time). There is a possibility that a patient needs more than one appointment per day (e.g. having an X-Ray procedure in the morning and coming back in the afternoon for re-evaluation). An appointment can be a follow up of a prior appointment. MNS records the prior appointment number for each follow up appointment. For example, appointment number 4 is a follow up of appointment number 1, and appointment number 31 is a follow up of appointment 4.

An appointment will require one or more services depending on the work required. For an appointment, MNS records the appointment date and time and the room number in which the consultation (appointment) will take place - in some circumstances this room may not be the provider's normal room, for example, due to the need to use specialist equipment. The reception staff, based on the patient's requirements, set an appointment length as either short (30 minutes), standard (60 minutes) or long (2 hours). The staff who schedule appointments will also ensure that the allocated provider is available and can provide the services which will be required during this appointment, your design is not required to enforce this, although these staff must be able to look up which providers provide which services if required.

Providers require the assistance of one dental nurse during an appointment. Each nurse is assigned a nurse number - MNS also records the nurse's name and their contact number. One nurse is assigned to each appointment. Nurses employed by MNS are hired based on three employment status: Casual (C), Contract (T), and Fulltime (F).

Each service provided at a particular appointment may require several items, for example fillings require Porcelain Etch and Silane. MNS records the quantity of items needed in each service provided at a particular appointment. Each item is assigned a unique identifier. MNS also records the item's description, standard cost and the number of items that are currently on hand. This figure is needed to make decisions about placing item purchase orders.

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



The schema/insert file for creating this model (mns_schema_insert.sql) is available in the archive ass2a_student.zip - this file partially creates the Monash New Smile tables and populates several of the tables (those shown in purple on the supplied model) - you should read this schema carefully and be sure you understand the various data requirements. **You must not alter the schema file in any manner**, it must be used as supplied. **Please note the yellow tables will not be used in any manner in this assignment and should be ignored (they will be used in Assignment 2B).**

Steps for working on Assignment 2A

1. Download the Assignment 2A Required Files (ass2a_student.zip) archive from Moodle
2. Extract the zip archive and place the contained files in your local repository in the folder /Assignments/Ass2A. Do not add the zip archive to your local repo. Then add, commit and push them to the FITGitLab server.
3. Run mns_schema_insert.sql
4. Write your answer for each task in its respective file (e.g. write your answer for Task 1 in T1-mns-schema.sql and so on).
5. Save, add, commit and push the file/s **regularly while you are working on the assignment**

6. 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. Note that the **filenames must not be changed** - you must submit files with exactly the same names as those supplied in the provided archive (ass2a_student.zip).

The final SQL scripts you submit **MUST NOT** contain SPOOL or ECHO commands (you may include them as you work but must comment them out before submission). Please carefully read the Marking Guide on pages 16 and 17.

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

- Week 6 Workshop and Week 7 Applied - Creating & Populating the Database
- Week 7 Workshop and Week 8 Applied - Insert, Update, Delete (DML) and Transaction Management
- Week 8 Workshop and Week 9 Applied - SQL Part I - Basic
- Week 9 Workshop and Week 10 Applied - SQL Part II- Intermediate
- Week 10 Workshop and Week 11 Applied - SQL Part III - Advanced

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 2A 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.

TASKS

ENSURE your id and name are shown at the top of any file you submit.

GIT STORAGE

Your work for these tasks **MUST** be saved in your individual local working directory (repo) in the Assignment 2A folder and **regularly pushed to the FIT GitLab server to build a clear history of development of your approach**. A minimum of ten pushes to the FIT GitLab server is required (2 pushes per file). Please note ten 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.

TASK 1: DDL (16 marks):

For this task you are required to add to **T1-mns-schema.sql**, the CREATE TABLE and CONSTRAINT definitions for the EMERGENCY_CONTACT, PATIENT and APPOINTMENT tables in the positions indicated by the comments in the script.

The table below provides details of the meaning of the attributes in the missing three tables. You **MUST use identical table 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. **You must use delete RESTRICT/NO ACTION for all FK constraints**. These new DDL commands *must be hand-coded, not generated in any manner (generated code will not be marked)*.

Table name	Attribute name	Meaning
EMERGENCY_CONTACT		
	ec_id	Emergency contact identifier
	ec_fname	Emergency contact first name
	ec_lname	Emergency contact last name
	ec_phone	Emergency contact phone number
PATIENT		
	patient_no	Patient number (unique for each patient)
	patient_fname	Patient first name
	patient_lname	Patient last name
	patient_street	Patient residential street address
	patient_city	Patient residential city
	patient_state	Patient residential state - NT, QLD, NSW, ACT, VIC,

		TAS, SA, or WA
	patient_postcode	Patient residential postcode
	patient_dob	Patient date of birth
	patient_contactmobile	Patient contact mobile number
	patient_contactemail	Patient contact email address
	ec_id	Patient emergency contact identifier
APPOINTMENT		
	appt_no	Appointment number (identifier)
	appt_datetime	Date and time of the appointment
	appt_roomno	Room in which appointment is scheduled to take place
	appt_length	Length of appointment - Short, Standard or Long (S, T or L)
	patient_no	Patient who books the appointment
	provider_code	Provider who is assigned to the appointment
	nurse_no	Nurse who is assigned to the appointment
	appt_prior_apptno	Prior appointment number which leads to this appointment (this is required to be unique)

To test your code you will need to first run the provided script **mns_schema_insert.sql** to create the other required tables. **mns_schema_insert.sql**, the head of this schema file, contains the drop commands for **all** tables in this model. If you have problems with Task 1 and/or Task 2 simply rerun **mns_schema_insert.sql** which will cause **all** tables to be dropped (including any you have created as part of task 2) and correct the issues in your script. **Do not add DROP TABLE statements** to either of your Task 1 or Task 2 scripts.

TASK 2: Populate Sample Data (24 marks):

Before proceeding with Task 2, you must ensure you have successfully run the file **mns_schema_insert.sql** (which **must not be edited in any way**) followed by the extra definitions that you added in Task 1 above (T1-mns-schema.sql). Note that the **mns_schema_insert** SQL commands nor any of your task 1 SQL code may be added to/reproduced in your Task 2 script.

Load the EMERGENCY_CONTACT, PATIENT and APPOINTMENT tables with **your own test data** using the supplied **T2-mns-insert.sql** script file, and SQL commands which will insert as a minimum (at least), the following sample data:

- 5 EMERGENCY_CONTACT entries
 - Involve at least 2 people being the emergency contact for more than two patients
- 10 PATIENT entries
 - Involve at least 5 adult patients and 5 under age (under 18 years old) patients

- 15 APPOINTMENT entries
 - All appointments which you add must be scheduled on three specific days in 2023. You may pick any three dates between 1 May 2023 and 31 August 2023
 - Involve some parallel appointments (ie. two or more appointments scheduled at the same date and time)
 - Involve at least 5 different providers
 - Involve at least 5 different nurses
 - Involve at least 3 follow up appointments

In adding this data, you must ensure that the data you insert will make full use of the various features you have coded in Task1. For example, if you have coded a check clause with permitted values of 1, 2 and 3 then your inserted data must use all three of these permitted values.

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 (i.e. **NOT** make use of sequences) and must consist of **values below 100**.
3. The data added must be sensible (e.g. the patient data must mimic the real data; the patient, provider, nurse and room must not be double booked; the following appointment for a provider must not be scheduled before the previous appointment for that provider is completed; and other real scenario constraints).

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 this task you must not modify any data or add any further data to the tables which were populated by the `mns_schema_insert.sql` script.

For all subsequent questions (Task 3 onwards) you are NOT permitted to:

- manually lookup an attribute/s in the database to obtain *any* value,
- manually calculate values (including dates/times) external to the database, e.g. on a calculator and then use such values in your answers. ***ALL necessary calculations must be carried out as part of your SQL code***, or
- assume any contents in the database - rows in a table are potentially in a constant state of change

Your answers must recognise the fact that you have been given, with the supplied insert file, only a small sample snapshot of a multiuser database, as such you must operate on the basis that there will be ***more data in all the tables of the database than you have been given. 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.***

You must ONLY use the data as provided in the text of the questions. Failure to adhere to this requirement will result in a mark of 0 for the relevant question.

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).

TASK 3: DML (20 marks):

Your answers for this task (Task 3) must be placed in the supplied SQL script **T3-mns-dm.sql**

For this task 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 *Orthodontics* you may search in the database using the provided string exactly as supplied. When a name is supplied you may break the name into title, first name and last name, for example, *Dr. Gerry ELLIOTT* can be split into *Dr.*, *Gerry* and *ELLIOTT*, again note that the case must be maintained as it was supplied.

- (a) Oracle sequences are going to be implemented in the database for the subsequent insertion of records into the database for the EMERGENCY CONTACT, PATIENT, and APPOINTMENT tables. Provide the CREATE SEQUENCE statements to create three sequences which could be used to provide primary key values for these three tables. These sequences must start at 100 and increment by 5.

Immediately prior to each create sequence command, place an appropriate DROP SEQUENCE command so that it will cause the sequence to be dropped before being created if it exists. Please note that there can only be three sequences introduced and used in Task 3.

[2 marks]

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) On 1st September 2023, *Jonathan Robey* (phone number: 0412523122) made an appointment for his two kids (*Laura* and *Lachlan*) for a general dental check-up. Jonathan registered himself as the emergency contact of his two kids. Previously this family has not been involved with MNS.

The reception then created two short (S) appointments for:

- *Laura* on 4th September 2023 at 3:30 PM, and
- *Lachlan* on 4th September 2023 at 4:00 PM.

Both kids were seen by *Dr. Bruce STRIPLIN*. You may assume that there is only one provider named *Dr. Bruce STRIPLIN* in the system. Nurse Chelsea Ford (nurse number: 6) was assigned to assist the provider in this appointment.

Take the necessary steps in the database to record the required entries for these appointments. You may assume that Jonathan is only registered as the emergency contact of *Laura* and *Lachlan* in the system (i.e Jonathan is not a patient at MNS and does not

currently wish to be registered as a patient). You may make up your own values for any required attributes where a value has not been supplied, for example a date of birth.

[8 marks]

- (c) Based on the diagnosis made by Dr Bruce STRIPLIN during the appointment on 4th September, *Lachlan* needed a follow up appointment for a tooth extraction procedure. The follow up appointment was scheduled 10 days after this first appointment on the 4th of September. The follow up appointment time was 4PM. The assigned nurse for this follow up appointment was nurse Katie (nurse number: 14) and the procedure would be carried by *Dr. Bruce STRIPLIN*. Take the necessary steps in the database to record the required entry for this follow up appointment.

[3 marks]

- (d) On 10th September, Jonathan called the receptionist to reschedule *Lachlan's* follow up appointment due to important school commitments. The follow up appointment was shifted to four days later (i.e. 14 days after the first appointment) at 4PM. The assigned provider and nurse remained the same. Make these required changes to the data in the database.

[4 marks]

- (e) On 15th September *Dr. Bruce STRIPLIN* had to leave the country due to an emergency family matter and was not able to work for one week. All *Dr. Bruce STRIPLIN* appointments between 15th September and 22nd September (inclusive) had to be cancelled. Take the necessary steps in the database to remove these appointments from the system.

[3 marks]

TASK 4: DATABASE MODIFICATIONS (20 marks):

Your answers for these tasks (Task 4) must be placed in the supplied SQL script **T4-mns-alter.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. Also remember as stated on page 7 ... "there will be **more data in all the tables of the database than you have been given. 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.**"...

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

- (a) MNS would like to be able to easily determine the total number of appointments for each patient in the system.

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

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.

[4 marks]

- (b) From this point forward, MNS would like to allow a patient to register more than one emergency contact person if they wish. Change the database to meet this requirement. Note that you must not lose any of the existing emergency contact data.

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 eg. desc customer to show any table structural changes.

[8 marks]

- (c) To ensure that nurse skills are kept up to date, nurses may be required to undertake initial/refresher training. This training is provided by other nurses. A given nurse may be trained by many other nurses. A given nurse may act as a trainer for many other nurses. The data below shows some samples of nurse training logs:

**Monash New Smile
NURSE TRAINING LOG BOOK**

Trainee:
Nurse No 8
Nurse First Name Kaitlyn
Nurse Last Name Rivers

Nurse Trainer No	Nurse Trainer First Name	Nurse Trainer Last Name	Start Date Time	End Date Time	Description
12	Mia	Blackall	27/07/2023 10:00	27/07/2023 16:00	Decontamination of scalling instruments
6	Chelsea	Ford	04/08/2023 11:00	04/08/2023 13:00	Initial examination data entry
6	Chelsea	Ford	04/08/2023 14:00	04/08/2023 15:00	Patient record keeping
12	Mia	Blackall	08/08/2023 9:00	08/08/2023 11:00	Maintaining dental x-ray machine

**Monash New Smile
NURSE TRAINING LOG BOOK**

Trainee:
Nurse No 7
Nurse First Name James
Nurse Last Name -

Nurse Trainer No	Nurse Trainer First Name	Nurse Trainer Last Name	Start Date Time	End Date Time	Description
12	Mia	Blackall	27/07/2023 13:00	27/07/2023 16:00	Scalling instrument maintenance
12	Mia	Blackall	08/08/2023 09:00	08/08/2023 11:00	Cleaning dental x-ray machine
6	Chelsea	Ford	28/08/2023 14:00	28/08/2023 15:00	Scalling instrument maintenance - refreshment

Change the database to meet this requirement. Note that you do not need to populate the training data, you only need to provide the structure in which data will be stored.

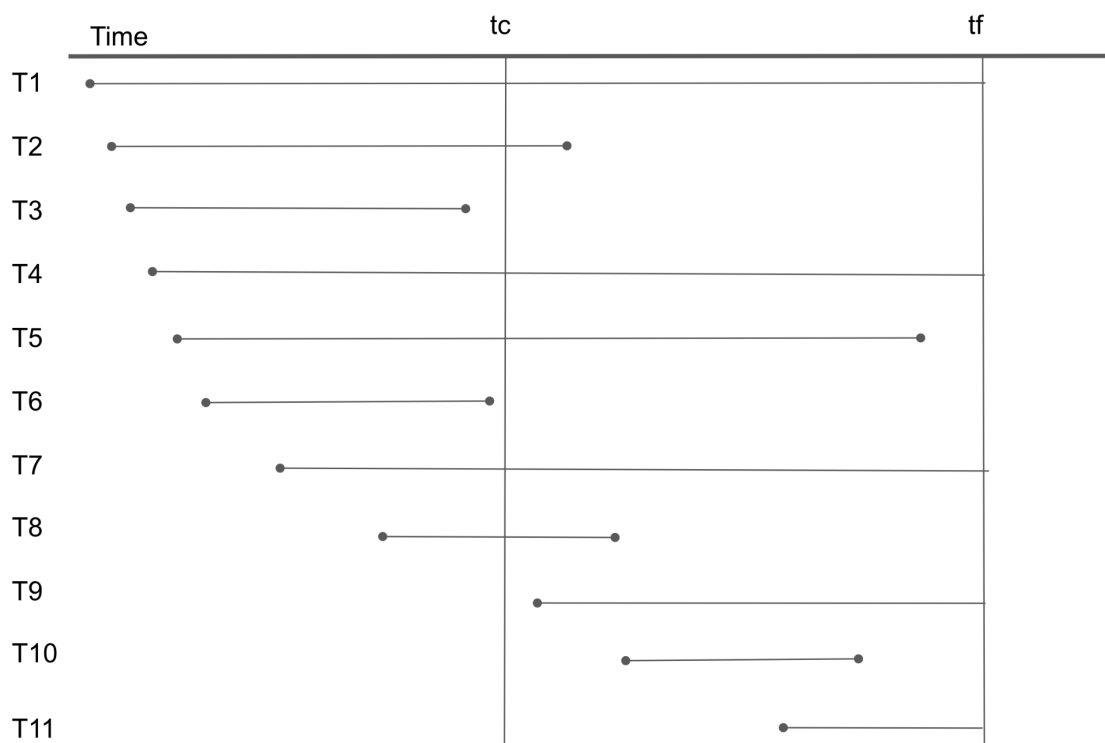
As part of your solution, provide appropriate desc statements to show the changes you have made e.g. `desc customer` to show any table structural changes.

[8 marks]

TASK 5: Transaction Theory (10 marks):

Your answers for this task (Task 5) must be written in an Ms Word document or Google document. Once you have completed Task 5a and 5b, download or print the document as a pdf file and name the file as **T5-mns-transaction.pdf**.

- (a) A database has eleven transactions running as listed below (the time is shown horizontally from left to right):



At time *tc* a checkpoint is taken, at time *tf* the database fails due to a power outage. If the database is a **write through** database, three stages are involved in the recovery process when the database is restarted.

Use the diagram above to discuss what happens at each of the three stages and what transactions are involved.

[5 marks]

- (b) Given the following transaction sequence, copy and paste this sequence into your answer document and clearly indicate what locks are present at each of the indicated times (Time 0 to Time 34).

Cell entries must have the form:

- **S(T_n)** - for a shared lock by T_n,
- **X(T_n)** - for an exclusive lock by T_n or
- **T_n wait T_m** - for a wait of T_n due to T_m (where n and m are transaction numbers).

TIME	TRANS	ACTION	A	B	C	D	E	F	G	H
0	T1	Read A								
1	T2	Read B								
2	T1	Read C								
3	T4	Read D								
4	T5	Read A								
5	T2	Read E								
6	T2	Update E								
7	T3	Read F								
8	T2	Read F								
9	T5	Update A								
10	T1	Commit								
11	T6	Read A								
12	T5	Rollback								
13	T6	Read C								
14	T6	Update C								
15	T7	Read G								
16	T8	Read H								
17	T9	Read G								
18	T9	Update G								
19	T8	Read E								
20	T7	Commit								
21	T9	Read H								

22	T3	Read G								
23	T10	Read A								
24	T9	Update H								
25	T6	Commit								
26	T11	Read C								
27	T12	Read D								
28	T12	Read C								
29	T2	Update F								
30	T11	Update C								
31	T12	Read A								
32	T10	Update A								
33	T12	Update D								
34	T4	Read G								

Complete the following:

- (i) For **each** of the listed items A .. H, what wait states are present at time 34 (the last time listed). Shown the waits in the form:

Item A: T1 waiting on T5 (*note this is an example ONLY*)

Item B: ... etc

- (ii) Prepare a **wait for graph** indicating the state of waiting locks at time 34. Your *wait for graph* can be prepared using any drawing package such as LucidChart or free hand drawn. Paste/insert an image of your wait for graph into your answer document below your answer to (i) above
- (iii) Report if deadlock exists or not, and if it does exist, list the transactions involved.

[5 marks]

Submission Requirements

Due Date: Thursday, 12th October 2023 at 4:30 PM

*Please note, if you need to resubmit, you **cannot** depend on your tutors' 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 five files you are **required** to submit to Moodle and must exist in your individual FIT Gitlab repository under Ass2A folder:

- T1-mns-schema.sql
- T2-mns-insert.sql
- T3-mns-dm.sql
- T4-mns-alter.sql
- T5-mns-transaction.pdf

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

Do not zip these files into one zip archive, submit four independent SQL scripts and one pdf file. The individual files must also have been pushed to the FIT GitLab server with an appropriate history as you developed your solutions (a minimum of ten pushes - 2 per file, however we would *strongly recommend more than this*). **Please ensure your commit comments are meaningful.**


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

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 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.

Resubmission

If you wish to resubmit your assignment you must email your tutor, provide your full details as listed below and request that they reopen your submission for a second submission. Note if this resubmission is after the due date/time the submission will be regarded as late.

When you contact your tutor (or workshop leader) via email, please ensure you clearly include your full name, unit code and applied class number as part of every email you send so they can identify who the message has come from. This will ensure we can respond as quickly and accurately as possible.

You must NOT assume that your tutor will be available if you require a resubmission close to the due date/time - they may have classes or not be available for other reasons, so do not leave submission to the very last minute.

Marking Guide

Submitted code will be assessed against an optimal solution for this task - this optimal solution will be available as a sample solution after Assignment 2A has been graded. Given that parts of this task involve SQL, 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 16 marks	
DDL Creation of tables	Maximum 10 marks - Create table: <ul style="list-style-type: none"> 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: <ul style="list-style-type: none"> Marks awarded for correct implementation of non-PK constraints
TASK 2 Populate Sample Data 24 marks	
Insert of required items test data	Maximum 12 marks- Insert of data: <ul style="list-style-type: none"> Marks awarded for correct insert of required data Marks awarded for correct management of transactions
Insert of valid test data	Maximum 12 marks - Valid data inserted: <ul style="list-style-type: none"> Marks awarded for validity of data inserted <ul style="list-style-type: none"> meets the requirements expressed in the assignment brief Marks awarded for correct management of dates when inserting
Task 3 DML 20 marks	
	Maximum 20 marks - Satisfy brief requirements: <ul style="list-style-type: none"> 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 20 marks	
	Maximum 20 marks - Satisfy brief requirements: <ul style="list-style-type: none"> Marks awarded (a) - (c) 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 Transaction Theory 10 marks	
	Maximum 10 marks - Satisfy brief requirements: <ul style="list-style-type: none"> Marks awarded (a) - (b) for correct answers and explanations Mark penalty for incorrect lock table notation
Correct use of Git 10 marks	
	Maximum 10 marks - Git used appropriately: <ul style="list-style-type: none"> Marks awarded for ten pushes showing a clear development history for the files (minimum two pushes per file) Marks awarded for correct Git author details used in pushes (see week 2 Applied notes Appendix) Marks awarded for the use of meaningful commit messages (ie. not blank or of the form "Push1")
Penalties	
Use of <ul style="list-style-type: none"> VIEWS SET ECHO or SPOOL commands, and/or PL/SQL 	Use of VIEWS, inclusion of SET ECHO/SPOOL, and/or PL/SQL commands in Task 1 - Task 4, will result in a grade deduction of 10 marks being applied.
Incorrect application of relational database principles	Marks will be deducted, based on any question, where basic relational model principles have been violated. For example, creation of a table which is not in 3NF
Late submission penalty	- 10 marks for each 24 hours late or part thereof

Final Assignment Mark Calculation

Total: 100 marks, recorded as a grade out of 25