FIT9132 Introduction to Databases - Assignment 2A

Creating, Populating and Manipulating Database - TimeShare Australia

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. 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.
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
Value	**25%** of your total marks for the unit
Due Date	Wednesday, **24th May 2023**, 4:30 PM (AEST) / 2:30 PM (MYT)
Submission	● SQL Portfolio 1 (submission in week 9) via Moodle Assignment 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).
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

Your task for this assignment is to create, populate and manipulate a database which can be used to support the activities of a holiday timeshare system - TimeShare Australia. If you are unfamiliar with time sharing, this link will provide some background.

Timeshare is a holiday system that permits members to purchase (own) a set amount of time (in the form of points) at a particular resort that they can use for holidays every year as a lifetime commitment. Through TimeShare Australia members of a particular resort can convert their purchase into a holiday at other resorts which are part of the system.

Several different companies jointly operate TimeShare Australia. For each company which is part of the system the companies ABN, name, CEO's name, contact phone number and address of the registered office are recorded. Companies either build or purchase resorts to add to the resorts which they manage.To be a member of TSA a company must be operating at least one resort.

Each resort is assigned a unique resort identifier across the entire TSA system and is only managed by one company. The system needs to record the resort's name, its street address, the contact phone number, the year in which it was built or purchased and the current cost for a member to purchase 1000 points.

A resort member, who is identified by a member number within the resort (ie. each resort has a member 1), has their name, home address, contact email and contact phone number recorded. Members purchase points from the resort at the time they sign up to be a member, for example, the purchase of 1000 points may entitle the member to one week's holiday every year at their home resort, or possibly two weeks in perpetuity. Members may only purchase points from their home resort (the resort they signed up to). The number of points which a member purchases is recorded. Members can only be a member of one resort across the TSA network. Current members of a resort may recommend a new person to join their resort - TSA wish to record which current member made a recommendation which resulted in a new member joining.

Since TimeShare Australia encompasses several companies and the resorts, they each manage, they have set up the system such that members can use their points from their home resort to holiday at any resort within the system, provided they have sufficient points available.

Each resort consists of a number of cabins. Each cabin is numbered within the resort (ie. each resort has a cabin 1). The number of bedrooms in the unit (between 1 to 4 bedrooms), the units sleeping capacity, the type of cabin bathroom (ie. ‘I’ - inside the cabin or ‘C’ - outside shared common bathroom), a unit description and the cost per day, expressed in member points, are recorded.

A member of TSA can book a unit at any resort which is available and is within their membership points. When they book, the starting date of the booking, the end date of the booking, the number of adults and the number of children holidaying are recorded. The system records the points cost to the member for this booking. At this point in the design process, you may disregard any "exchange" fees that TSA charge for holidaying outside a member's home resort.

TSA records details of points of interest that members might wish to visit during their holiday, such as parks, museums, etc which may be in the same town as the resort or in other close by towns. For each town in which a resort is located, or which has a point of interest, TSA records the town id, and the centre of the town's latitude and longitude. The town name, state of Australia, the average summer and winter day temperatures and its population are also recorded. Even though a POI (such as the Great Western Desert) may span several geographical locations, TSA assigns a Point of Interest to only a single town. A town may have two points of interest of the same name.

For points of interest, TSA records the street address and town in which the point of interest is located and the name of the point of interest (e.g. Phillip Island Community & Craft Gallery) - a given town may have two points of interest of the same name. TSA also records the opening hours (open and close time) if appropriate, a brief description of the point of interest and the type of point of interest e.g. art gallery.

To assist members to get the best from their holidays at a resort, TSA invite members to complete a review of points of interest in the local area that they have visited. Not all members will accept the invitation. Those that do will complete a review and rate the POI as 1, 2, 3, 4 or 5 with 5 being the best to visit. TSA publishes these individual member reviews so potential visitors can make a more informed decision on what to visit. TSA also uses these accumulated reviews to calculate a rating level for each POI they have recorded based on the average rating expressed to one decimal point eg. 4.2..

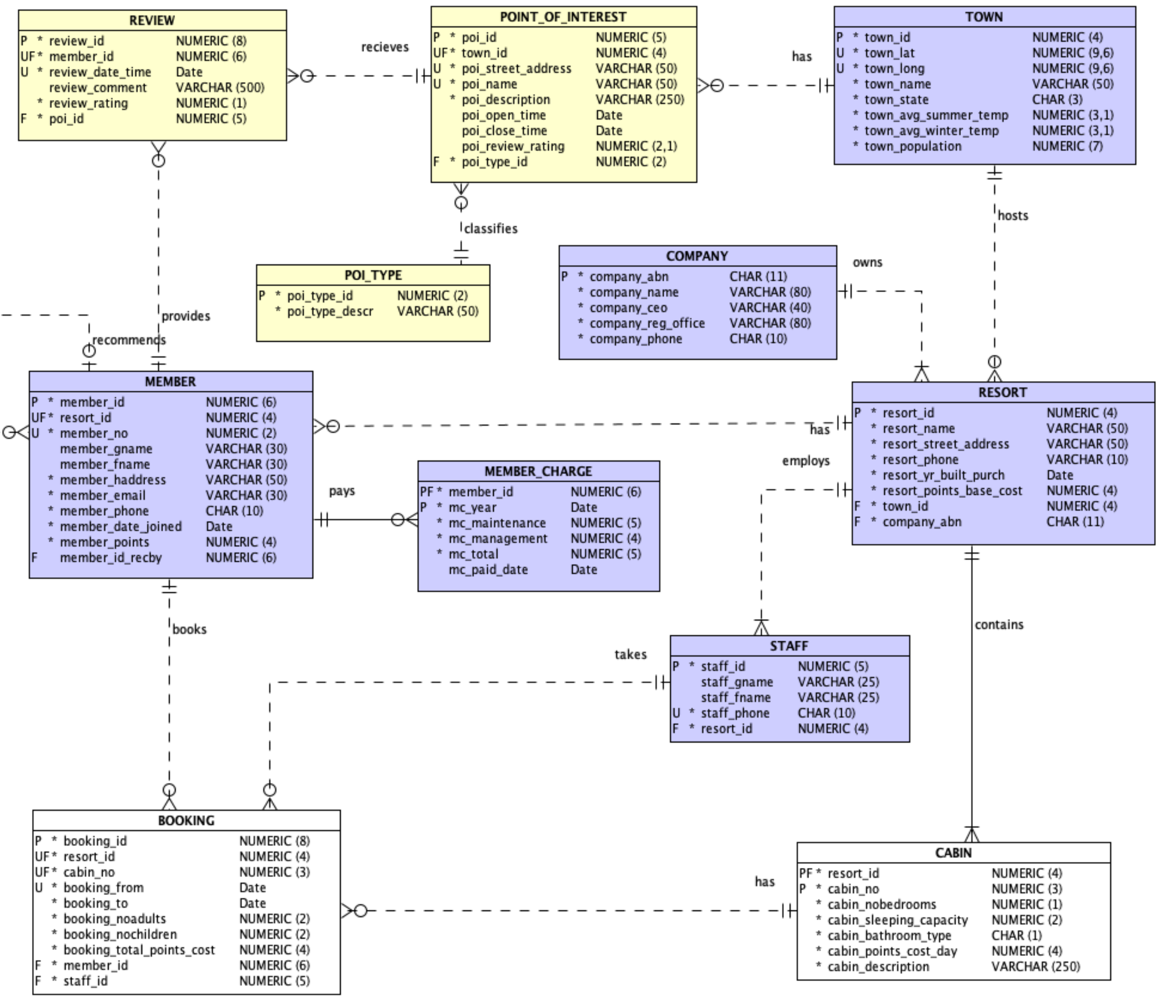
When a member purchases membership at a resort they will make the initial payment of several thousand dollars to buy into the resort as a member. This initial payment is made only once. Members also sign a contract which binds them to annual charges which cover the running and upkeep of their resort. These annual charges are of two types:

i. Annual Management Fee - this is the costs for the expenses involved in running the resort such as salaries, office expenses etc

ii. Annual Maintenance Fee - this is the costs incurred by the resort in upkeep and update/renewal of the resort facilities such as repainting a unit etc

The two fees above are worked out by the resort for the full calendar year and then apportioned to members based on the value of their membership points. A member, for example, with 2000 points will pay double the charges of a member with 1000 points since they "own" twice as much of the resort. The detailed records of the management and maintenance costs are maintained by TSA outside the system you are developing. Management and maintenance fees are rounded to the nearest dollar when charging to members.

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


REVIEW
P * review_id NUMERIC (8)
UF * member_id NUMERIC (6)
U * review_date_time Date
review_comment VARCHAR (500)
* review_rating NUMERIC (1)
F * poi_id NUMERIC (5)
recieves
POINT_OF_INTEREST
P * poi_id NUMERIC (5)
UF * town_id NUMERIC (4)
U * poi_street_address VARCHAR (50)
U * poi_name VARCHAR (50)
* poi_description VARCHAR (250)
poi_open_time Date
poi_close_time Date
poi_review_rating NUMERIC (2,1)
F * poi_type_id NUMERIC (2)
has
TOWN
P * town_id NUMERIC (4)
U * town_lat NUMERIC (9,6)
U * town_long NUMERIC (9,6)
* town_name VARCHAR (50)
* town_state CHAR (3)
* town_avg_summer_temp NUMERIC (3,1)
* town_avg_winter_temp NUMERIC (3,1)
* town_population NUMERIC (7)
classifies
hosts
POI_TYPE
P * poi_type_id NUMERIC (2)
* poi_type_descr VARCHAR (50)
COMPANY
P * company_abn CHAR (11)
* company_name VARCHAR (80)
* company_ceo VARCHAR (40)
* company_reg_office VARCHAR (80)
* company_phone CHAR (10)
owns
provides
recommends
MEMBER
P * member_id NUMERIC (6)
UF * resort_id NUMERIC (4)
U * member_no NUMERIC (2)
member_gname VARCHAR (30)
member_fname VARCHAR (30)
* member_haddress VARCHAR (50)
* member_email VARCHAR (30)
* member_phone CHAR (10)
* member_date_joined Date
* member_points NUMERIC (4)
F member_id_recby NUMERIC (6)
has
RESORT
P * resort_id NUMERIC (4)
* resort_name VARCHAR (50)
* resort_street_address VARCHAR (50)
* resort_phone VARCHAR (10)
* resort_yr_built_purch Date
* resort_points_base_cost NUMERIC (4)
F * town_id NUMERIC (4)
F * company_abn CHAR (11)
employs
pays
MEMBER_CHARGE
PF * member_id NUMERIC (6)
P * mc_year Date
* mc_maintenance NUMERIC (5)
* mc_management NUMERIC (4)
* mc_total NUMERIC (5)
mc_paid_date Date
books
takes
STAFF
P * staff_id NUMERIC (5)
staff_gname VARCHAR (25)
staff_fname VARCHAR (25)
U * staff_phone CHAR (10)
F * resort_id NUMERIC (4)
contains
BOOKING
P * booking_id NUMERIC (8)
UF * resort_id NUMERIC (4)
UF * cabin_no NUMERIC (3)
U * booking_from Date
* booking_to Date
* booking_noadults NUMERIC (2)
* booking_nochildren NUMERIC (2)
* booking_total_points_cost NUMERIC (4)
F * member_id NUMERIC (6)
F * staff_id NUMERIC (5)
has
CABIN
PF * resort_id NUMERIC (4)
P * cabin_no NUMERIC (3)
* cabin_nobedrooms NUMERIC (1)
* cabin_sleeping_capacity NUMERIC (2)
* cabin_bathroom_type CHAR (1)
* cabin_points_cost_day NUMERIC (4)
* cabin_description VARCHAR (250)

The schema/insert file for creating this model (tsa_schema_insert.sql) is available in the archive ass2a_student.zip - this file partially creates the TimeShare Australia 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 tsa_schema_insert.sql
4. Write your answer for each task in its respective file (eg. write your answer for Task 1 in T1-tsa-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 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.

****

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**. Any submission with less than ten pushes to the FITGitLab server will incur a grade penalty of 9 marks. Please note ten pushes is a *minimum*, in practice we would expect significantly more.

Before submission via Moodle, you **must** log into the web interface of the GitLab server and ensure your files are present in your individual repo and that their names are unchanged.

TASK 1: DDL (16 marks):

For this task you are required to add to **T1-tsa-schema.sql**, the CREATE TABLE and CONSTRAINT definitions for the BOOKING and CABIN 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 two 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
BOOKING		
	booking_id	surrogate key added to replace BOOKING composite PK
	resort_id	Resort identifier, for this booking
	cabin_no	Cabin number within the resort, for this booking
	booking_from	Date booking from
	booking_to	Date booking to
	booking_noadults	Booking number of adults
	booking_nochildren	Booking number of children
	booking_total_points_cost	Total cost to the member in points for this booking
	member_id	Unique member id across TSA for member who made this booking
	staff_id	Staff identifier of staff member who took this booking

CABIN		
	resort_id	Resort identifier
	cabin_no	Cabin number within the resort
	cabin_nobedrooms	Number of bedrooms in cabin (between 1 and 4 bedrooms)
	cabin_sleeping_capacity	Cabin sleeping capacity
	cabin_bathroom_type	Type of cabin bathroom: I - Inside cabin bathroom C - outside common bathroom
	cabin_points_cost_day	Number of members points the cabin costs per day
	cabin_description	Cabin description

To test your code you will need to first run the provided script **tsa_schema_insert.sql** to create the other required tables. **tsa_schema_insert.sql**, at the head of the file, contains the drop commands for ***all*** tables in this model. If you have problems with Task 1 and/or Task 2 simply rerun **tsa_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 tsa_schema_insert.sql (which **must not be edited in any way**) followed by the extra definitions that you added in Task 1 above and the provided code for the trigger (T1-tsa-schema.sql). You are not required to understand the trigger code, it is simply to help you when adding new bookings by reporting the members current points so you can cross check your inserted data.

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

- 20 CABIN entries
  - Involved at least 5 different resorts
    - at least 2 of these resorts must have more than 2 cabins
- 20 BOOKING entries
  - Involved at least 10 cabins in 5 different resorts
    - at least 3 of these cabins must be booked at least three times
  - Involve at least 10 different members
    - at least 3 of these members have more than one booking

In adding this data, you must ensure that the data you insert will validate the schema you have created by making full use of the various features you have coded (for example check clauses).

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 March 2023 and 30th April 2023.
4. The data added must be sensible (eg. the booking to date should be after the booking from date; the total number of guests must be less than or equal to the cabin capacity; the total cost of a booking must be inline with the cabin cost per day; a booking from the 5th March 2023 until the 12th March 2023 is a 7 day booking).

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 *tsa_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-tsa-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 *Amazing Resort* 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 given name and family name, for example, *Indiana Perrier* can be split into Indiana and Perrier, 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 BOOKING table.
Provide the CREATE SEQUENCE statement to create a sequence which could be used to provide primary key values for the BOOKING table. This sequence must start at 100 and increment by 10. Immediately prior to the 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 one sequence 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) On 30th April 2023, *Awesome Resort* in Broome (latitude:-17.9644, longitude:122.2304) launched a new cabin which has 4 bedrooms, a capacity of 10 people and costs 220 points per day. This new cabin is the fourth cabin in this resort (ie. cabin number 4). Take the necessary steps in the database to record this new cabin. You may assume that there is only one *Awesome Resort* in Broome and you may make up your own values for other required attributes.

[4 marks]

(c) On 1st May 2023, *Noah Garrard* (member number 2 in resort with id 9) made a booking for the new cabin added in point (b) above. He intended to stay with 3 other adults and 4 kids (ie. total of 4 adults and 4 children) in this cabin from Friday, 26th May 2023 to Sunday, 28th May 2023. The booking was added to the system by the resort staff member named Reeba Wildman (phone number: *0493427245*). Take the necessary steps in the database to record the required entry for this new booking.

[4 marks]

(d) The next day, *Noah Garrard* called TSA and extended the booking for one extra day (ie. the proposed check out date will be Monday, 29th May 2023). Make these required changes to the data in the database.

[4 marks]

(e) On the 4th May 2023, before any members holidayed in this new cabin, a fire destroyed the cabin and management decided, due to the costs involved, that they would not replace it. The TSA was then instructed to remove the cabin from the system. Members who have booked in this cabin will be informed by TSA that their booking has been cancelled. Take the necessary steps in the database to remove the cabin from the system.

[7 marks]

TASK 4: DATABASE MODIFICATIONS (20 marks):

Your answers for these tasks (Task 4) must be placed in the supplied SQL script **T4-tsa-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 8 …"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) TSA would like to be able to easily determine the total number of bookings for each cabin 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 bookings*. If there is no booking for a particular cabin, the attribute value must be 0.

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.

[4 marks]

(b) TSA would like to record the role of each staff member for staff management purposes. TSA currently has three different roles for their staff:

Role ID	Role Name	Job Description
A	Admin	Take bookings, and reply to customer inquiries
C	Cleaning	Clean cabins and maintain resort's public area
M	Marketing	Prepare and present marketing ideas and deliverables

Each staff member has one assigned role. TSA would like to record this data in their database and wish to initialise the role for all staff that they have (ie. currently in the system) as Admin staff. TSA would also like to expand the list of roles in the future. Change the database to meet 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 eg. desc customer to show any table structural changes.

[7 marks]

(c) Cabins need to be cleaned after the guest checks out or when the guest requests it, but a cabin can only be scheduled for a clean once for a particular date. To speed up the cleaning process, TSA may assign more than one staff member to do the cleaning. For each staff member involved in a scheduled cleaning for a particular cabin, the cleaning start time and end time are recorded for payroll purposes.

Change the database to meet this requirement. Note that you do not need to populate the cleaning 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 eg. desc customer to show any table structural changes.

[9 marks]

TASK 5: Transaction Theory (10 marks):

Your answers for these tasks (Task 5) must be placed in the supplied Ms Word document **T5-tsa-transaction.docx**. Once you have completed Task 5a and 5b, download or print the document as a pdf file named **T5-tsa-transaction.pdf**.

(a) Given the following transaction sequence, complete the supplied document by clearly indicating what locks are present at each of the indicated times (Time 0 to Time 9).

Cell entries must have the form:

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

TIME	TRANS	ACTION	A	B	C	D
0	T1	UPDATE A				
1	T1	UPDATE B				
2	T2	READ C				
3	T2	READ D				
4	T3	UPDATE A				
5	T2	UPDATE C				
6	T1	ROLLBACK				
7	T3	UPDATE C				
8	T2	UPDATE B				
9	T2	UPDATE A				

In the document indicate:

- Does a deadlock exist in this transaction sequence?
- Explain why you came to this conclusion.

[5 marks]

(b) Suppose that you are a manufacturer of product ABC, which is composed of parts A, B, and C. Each time a new product ABC is created, it must be added to the product inventory using the PROD_QOH in a table named PRODUCT. Also, each time the product is created, the parts inventory, using PART_QOH in a table named PART, must be reduced by one for each of the parts A, B, and C used to make the new product ABC.

The current database contents are shown in the table below:

PRODUCT TABLE

PROD_CODE	PROD_QOH
ABC	1205

PART TABLE

PART_CODE	PART_QOH
A	567
B	98
C	549

Starting with this database state, prepare a transaction log based on the structure shown in the Week 7 Workshop slides (slide 33) or figure 10.1 of Coronel and Morris for the production of a new product ABC. The starting TRL_ID will be 101 and TRX_NUM assigned by the database is 601 (based on C&M Chpt 10 Problems Q1).

[5 marks]

Submission Requirements

Due Date: **Wednesday, 24th May 2023 at 4:30 PM AEST / 2:30 PM MYT**

*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-tsa-schema.sql
- T2-tsa-insert.sql
- T3-tsa-dm.sql
- T4-tsa-alter.sql
- T5-tsa-transaction.pdf

  Note: The source file (T5-tsa-transaction.docx) must also exist on your individual FIT GitLab repository and show a clear development history..

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.

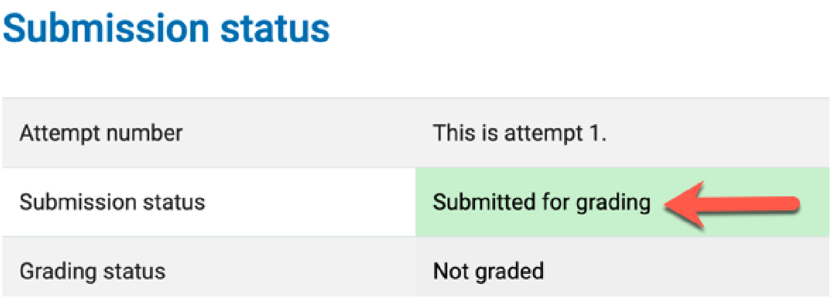
Do not zip these files into one zip archive, submit four independent SQL scripts and one pdf file. The individual files must also have been pushed to the FIT GitLab server with an appropriate history as you developed your solutions (a minimum of ten pushes - 2 per file, however we would *strongly recommend 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 this is 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:** ● 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 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 Populate Sample Data 24 marks	
Insert of required items test data	**Maximum 12 marks- Insert of data:** ● Marks awarded for correct insert of required data ○ 20 CABIN entries ○ 20 BOOKING entries ● Marks awarded for correct management of transactions
Insert of valid test data	**Maximum 12 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 20 marks	
	Maximum 20 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 20 marks	
	Maximum 20 marks - Satisfy brief requirements: ● 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: ● Marks awarded (a) - (b) for correct answers and explanations ● Mark penalty for incorrect lock table notation
Limited/No push of model to FITGitLab server resulting in lack of development history.	If less than ten pushes showing a clear development history a ***grade deduction of 9 marks applied***. Note that the expectation is that you would push significantly more times than this.
Use of ● VIEWs ● SET ECHO or SPOOL commands, and/or ● PL/SQL	Use of VIEWS, inclusion of SET ECHO/SPOOL, and/or PL/SQL (except for the provided trigger) commands in Task 1 - Task 4, will result in a ***grade deduction of 9 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

Final Assignment Mark Calculation

- 90 marks from the Assignment 2A submission => 22.5 / 25
  PLUS
- 10 marks from the SQL Portfolio 1 => 2.5 / 25

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