FIT9132 Introduction to Databases - Assignment 2B

Database - TimeShare Australia

Purpose	Students will be asked to use SQL and NoSQL to write queries to produce specified output. This task covers learning outcomes: 1. Apply the theories of the relational database model. 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.
Your task	This is an open book, **individual task**. The final output for this task will be a set of relational (RA and Oracle) and non relational (MongoDB) queries which meet the user requirements.
Value	**35%** of your total marks for the unit
Due Date	Friday, **9th June 2023**, 4:30 PM (AEST) / 2:30 PM (MYT)
Submission	● SQL Portfolio 2 (submission in week 10) and SQL Portfolio 3 (submission in week 11) 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 algebra operations to produce outputs that meet user requirements ● 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.
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

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. If you are unfamiliar with time sharing, this link will provide some background.

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.
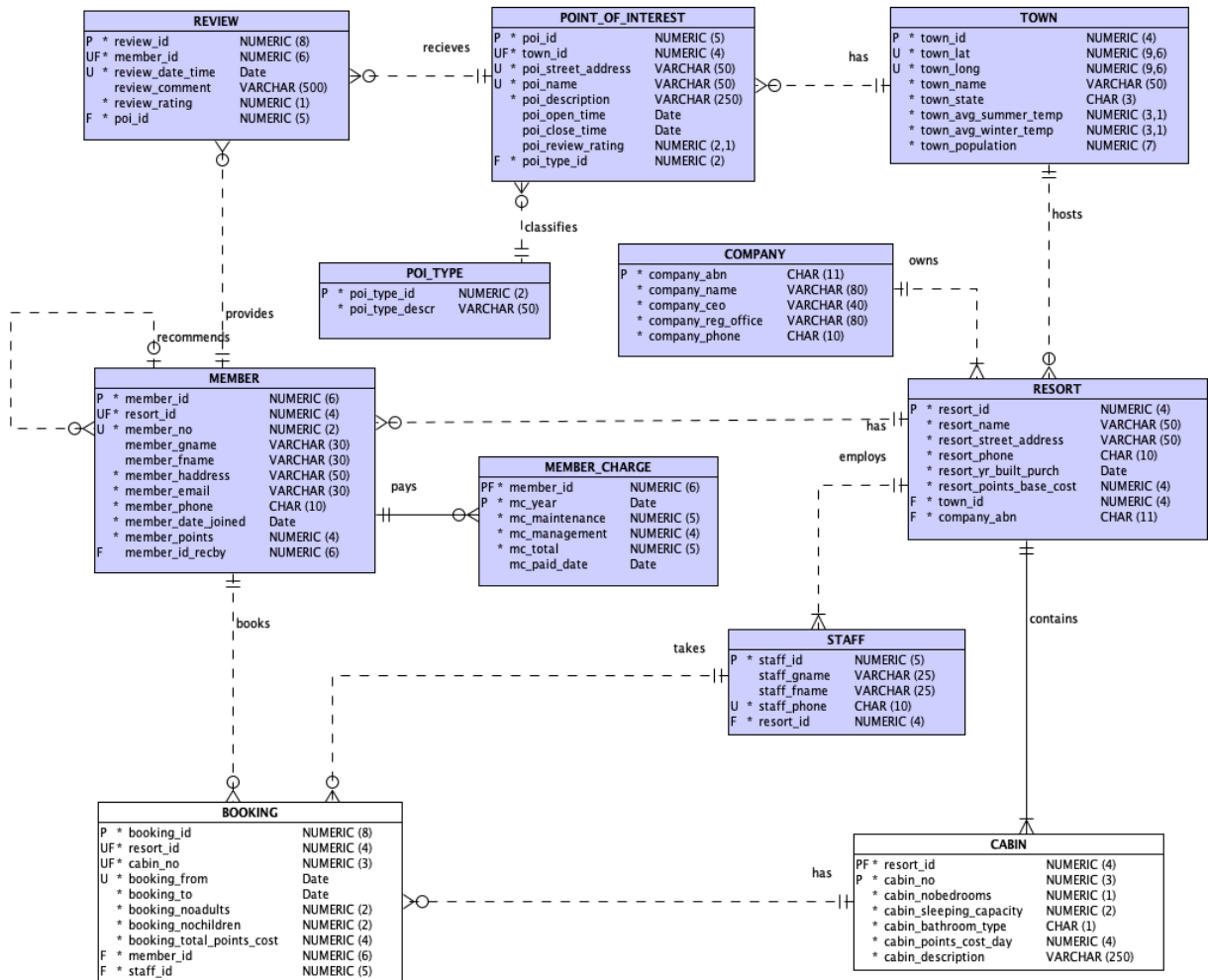
To assist members to get the best from their holidays at a resort, TSA invite members to complete a review of points of interest in the local area that they have visited. Not all members will accept the invitation. Those that do will complete a review and rate the POI as 1, 2, 3, 4 or 5 with 5 being 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)

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)

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)

recieves

has

classifies

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)

hosts

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)

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

RESORT

P	* resort_id	NUMERIC (4)
	* resort_name	VARCHAR (50)
	* resort_street_address	VARCHAR (50)
	* resort_phone	CHAR (10)
	* resort_yr_built_purch	Date
	* resort_points_base_cost	NUMERIC (4)
F	* town_id	NUMERIC (4)
F	* company_abn	CHAR (11)

has

pays

employs

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)

Please note the white tables will not be used in any manner in this assignment and should be ignored.

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 2B folder and **regularly pushed to the FIT GitLab server to build a clear history of development of your approach**. Any submission with less than eight pushes to the FITGitLab server will incur a grade penalty of 8 marks. Please note eight 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: Relational Database Queries - Relational Algebra (8 marks):

Your answers for this task (Task 1) must be placed in the supplied Ms Word document **T1-tsa-ra.docx**. Once you have completed all questions, download or print the document as a pdf file and name the file as **T1-tsa-ra.pdf**.

For this task you are required to write the **relational algebra operations** for the following queries (your answer must show an understanding of query efficiency).

Example:
Question:
    List the id, name, and address of all resorts which have point base cost greater than $3000.
Answer:
    $R = \pi_{resort\_id, resort\_name, resort\_address}(\sigma_{resort\_point\_base\_cost>3000} \text{RESORT})$
OR

    $R1 = \sigma_{resort\_point\_base\_cost>3000} \text{RESORT}$
    $R = \pi_{resort\_id, resort\_name, resort\_address} (R1)$

List of symbols for copying/pasting as you enter your answers below:

project: π, select: σ, join: ⋈, intersect: ∩, union: ∪, minus: -

(a) List the id, name and state of all towns which do not have any point of interest.

[2 marks]

(b) List the id, name, street address and description of all points of interests which fall under 'Nature and Wildlife' type and have a review rating above 3.

[2 marks]

(c) List member id, member given name, poi id, poi name, review date time, review rating and review comment of all reviews written for POIs which are located in a town named Broome (latitude:-17.9644, longitude:122.2304)

[4 marks]

TASK 2: Relational Database Queries - SQL (56 marks):

****

Before attempting this task, **drop all tables under your account**.
In arriving at your solutions for Task 2 you are ONLY permitted to use the SQL structures, syntax and functions **which have been covered within this unit**:

- 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 or marked**.

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*,
- assume any contents in the database - rows in a table are potentially in a constant state of change, or
- use views and/or PL/SQL

Your answers must recognise the fact that you have been given 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 <u>multiple users will be operating in the tables at the same time</u>. 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**.

****

In the Monash Oracle database, this TimeShare Australia set of tables has been created under the user "TSA". To use these tables you need to add the prefix "TSA" to the table names that you use in an SQL statement. So, if you want to retrieve data from RESORT table you need to write:

```
select * from tsa.resort;
```

Your answers for this task (Task 2) must be placed in the supplied SQL script **T2-tsa-select.sql.**

You are only permitted to code **a single select statement for each question** below.
For each question sample output showing the **form** of what you are required to produce is provided. Note this *is the form of the output ONLY ie. the appearance*, the data you return will be different.

(a) For each town which has points of interest (POI) and the town contains more than one POI of any type, display the number of POI's of all such POI types in the town.

Your output must show the town id, town name, poi type id, the poi type description and the number of POI for each POI type in the town that is greater than one. The output must be ordered by town id and for a given town by poi type description.

Your output must have the *form* shown below:

TOWN_ID	TOWN_NAME	POI_TYPE_ID	POI_TYPE_DESCR	POI_COUNT
5	Alice Springs	3	Nature and Wildlife	5
6	Broome	2	Historic places and Museum	2

Here, for example, town id 5 has five points of interest which are of type 3 Nature and Wildlife

[6 marks]

(b) Find the member/s who have made the highest number of recommendations for new members.

Your output must show the member id, the member name in a single column, the members home resort id (where they became a member), the home resort name and the number of recommendations they have made which have resulted in new members.

The output must be ordered by resort id, and within a given resort by member id.

Your output must have the *form* shown below:

MEMBER_ID	MEMBER_NAME	RESORT_ID	RESORT_NAME	NUMBER_OF_RECOMMENDATIONS
2	Florida Goldhawk	1	Byron Bay Exclusive Resort	3
20	Rochelle	6	Gympie Luxury Resort	3

[8 marks]

(c) For every point of interest (POI) show the highest review rating, the lowest review rating and the average review rating.

Your output must show the poi id, poi name, the max review rating, the minimum review and the average rating. Where a POI has not been rated to display 'NR' for the maximum/minimum and average values.

The output must be ordered by poi id.

Your output must have the *form* shown below:

POI_ID	POI_NAME	MAX_RATING	MIN_RATING	AVG_RATING
1	Alice Springs Telegraph Station Historical Reserve	5	3	4.3
2	Arakwal National Park	5	4	4.5
3	Gympie Gold Mining and Historical Museum	5	5	5.0
4	Broome Historical Museum	NR	NR	NR
5	Sun Picture Gardens	NR	NR	NR

[8 marks]

(d) For all points of interest (POI) show the percentage of all reviews that have been completed which are for that point of interest.

Your output must show the POI name, the POI type description, the name of the town in which the POI is located and the town's location. The town location must be shown as a single column 35 characters in width; right aligned, in the form 'Lat: -23.698000 Long: 133.880700'. The output must also show the number of reviews which have been completed for this POI and the percentage of all reviews this represents. If no reviews have been completed for a POI show 'No reviews completed' in the percent of reviews column.

Order the output by town name and within a town name with the highest number of reviews first. Where in a given town, several poi type descriptions have the same number of reviews, order them by the poi_name.

Your output must have the *form* shown below:

POI_NAME	POI_TYPE_DESCR	TOWN_NAME	TOWN_LOCATION	REVIEWS_COMPLETED	PERCENT_OF_REVIEWS
Anzac Hill	Nature and Wildlife	Alice Springs	Lat: -23.698000 Long: 133.880700		39.09%
Adelaide House	Historic places and Museum	Alice Springs	Lat: -23.698000 Long: 133.880700		26.06%
Alice Springs Desert Park	Nature and Wildlife	Alice Springs	Lat: -23.698000 Long: 133.880700		26.06%
Tanami Desert - NT	Nature and Wildlife	Alice Springs	Lat: -23.698000 Long: 133.880700	0	No reviews completed

Here, for example, Adelaide House has received 2 reviews which represent 6.06% of all reviews given and Tanami Desert - NT has had no reviews recorded.

[10 marks]

(e) For all members whose home resort is not located in a city called '*Byron* ~~Bryon~~ *Bay*' in '*NSW*' and who joined based on a recommendation from another member, and whose total member charge that they have paid to date is less than the average total member charge for all members of their home resort. Show the members home resort id, resort name, their member no, member name in a single column, date joined, the details of the member who recommended them in a single column (member no and name eg. '1 Florida Goldhawk') left aligned and the total member charges they have paid rounded to the nearest dollar. The total charges should be shown with a leading $ sign, and be right aligned in a column 13 characters wide.

Order the output by resort id and then within resort id by member number.

Your output must have the *form* shown below:

RESORT_ID	RESORT_NAME	MEMBER_NO	MEMBER_NAME	DATE_JOINED	RECOMMENDED_BY_DETAILS	TOTAL_CHARGES
7	Taree Exclusive Resort	2	Haleigh Bonifacio	15-May-2022	1 Florida Goldhawk	$3900
7	Taree Exclusive Resort	3	Rose	15-May-2022	1 Florida Goldhawk	$3900

[10 marks]

(f) You have been provided with a special function (geodistance) in your Monash Oracle account which returns the great circle distance between two points expressed in latitude and longitude. The function is called via geodistance (lat_city1, long_city1, lat_city2, long_city2).

As an example, given two towns:

Town	Latitude	Longitude
Surfers Paradise	-28.000767	153.429642
Currumbin	-28.135731	153.486923

The great circle distance between these two towns can be calculated via:

```
SELECT
    geodistance(- 28.000767, 153.429642, - 28.135731, 153.486923)
FROM
    dual;
```

which yields a great circle (or air) distance of 16.03 Km

Based on the Town and POI data you have been supplied with, TSA would like to create a list of points of interest (poi's) close to their various resorts to help guests who would like to tour the local area.

The output must show the resort id, resort name and for each point of interest within 100 Km great circle distance of the resort; the poi name, ~~poi street address~~, poi town, poi state, poi opening time and the great circle distance to the POI in a column called 'DISTANCE'. Within 100 Km is inclusive (ie 100 Km from the resort or less). The output must be ordered by the resort name and for a given resort by the separation in kms. Your SQL query **MUST NOT use a cross join** as part of your solution.

Your output must have the *form* shown below:

RESORT_ID	RESORT_NAME	POI_NAME	POI_TOWN	POI_STATE	POI_OPENING_TIME	DISTANCE
5	Byron Bay Super Resort	Broken Head Nature Reserve	Byron Bay	NSW	Not Applicable	0.0 Kms
6	Gympie Luxury Resort	Gympie Gold Mining and Historical Museum	Gympie	QLD	Not Applicable	0.0 Kms
6	Gympie Luxury Resort	Woondum National Park	Mothar Mountain	QLD	Not Applicable	12.4 Kms
6	Gympie Luxury Resort	Kingaroy Heritage Museum	Kingaroy	QLD	09:00 AM	90.9 Kms
6	Gympie Luxury Resort	Kingaroy Observatory	Taabinga	QLD	Not Applicable	95.0 Kms
8	King Resort at Kingaroy	Kingaroy Heritage Museum	Kingaroy	QLD	09:00 AM	0.0 Kms

[14 marks]

TASK 3: Non Relational Database Queries - MongoDB (16 marks):

Your answers for this task (Task 3) must be placed in the supplied sql file **T3-tsa-json.sql** and the supplied MongoDB script file **T3-tsa-mongo.mongodb.js**

(a) Write an SQL statement in **T3-tsa-json.sql** to generate a collection of JSON documents using the following structure/format from the **TSA** tables. Each document in the collection represents a town and resorts within the town. Note that the town name in this structure is the combination of town name and town state.

```
[
    {
        "_id": 1,
        "name": "Byron Bay, NSW",
        "location": {
            "latitude": -28.6474,
            "longitude": 153.602
        },
        "avg_temperature": {
            "summer": 23.5,
            "winter": 16.9
        },
        "no_of_resorts": 2,
        "resorts": [
            {
                "id": 1,
                "name": "Byron Bay Exclusive Resort",
                "address": "1 Karma Road",
                "phone": "0212429423",
                "year_built": 2012,
                "company_name": "Byron Holiday"
            },
            {
                "id": 5,
                "name": "Byron Bay Super Resort",
                "address": "675 Lennon Street",
                "phone": "0224811234",
                "year_built": 2016,
                "company_name": "Byron Holiday"
            }
        ]
    },
    … //partial collection only
]
```

[5 marks]

Write the MongoDB commands for the following questions, 3(b) - 3(e), in the supplied MongoDB script file named **T3-tsa-mongo.mongodb.js**.

    (b) Create a new collection and insert all documents generated in 3(a) above into MongoDB.

[1 mark]

    (c) List all towns which have summer average temperatures greater than 25 degrees and have resort/s which are owned by a company named "Tropical Dream".

[2 marks]

    (d) The management decided to change the name of a company from "Tropical Dream" into "Tropical Heaven". Change the data within the collection to reflect this decision. You may assume that there is only one company named "Tropical Dream" in the collection.

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

[2 marks]

    (e) "Australia Experience" company opened a new resort called "Amazing Resort" (resort id 10) located in a town named "Mount Isa, QLD" (latitude: -20.7256, longitude: 139.4927). Write the necessary MongoDB commands to add this new resort into the collection. You may make up other required data for the insert.

After inserting the details use an appropriate db.find command so that you illustrate/confirm the insert which was made.

[6 marks]

Submission Requirements

Due Date: **Friday, 9th June 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 four files you are **required** to submit to Moodle and must exist in your individual FIT Gitlab repository **under Ass2B folder**:

- T1-tsa-ra.pdf
    - *Note: The source file (T1-tsa-ra.docx) must also exist on your individual FIT GitLab repository and show a clear development history.*
- T2-tsa-select.sql
- T3-tsa-json.sql
- T3-tsa-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 in the "Comments for your marker:" section.

Do not zip these files into one zip archive, submit four independent files. 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 8 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 solutions will be assessed against an optimal solution for this task - this optimal solution will be available as a sample solution after Assignment 2B has been graded.

TASK	Items Assessed
TASK 1 Relational Algebra Queries 8 marks	
	Maximum 8 marks - Satisfy brief requirements: ● Marks awarded, as listed, (a) - (c) for relational algebra which meets the expressed requirement ● Marks awarded for correct use of RA operations and the use of appropriate symbols ● Mark penalty applied if your answer does not show an understanding of query efficiency ie. you must not make use of unnecessary joins, nor carry attributes and tuples up through the query which are not necessary ● 0 marks will be awarded if SQL select statements are provided as solutions for task 1
TASK 2 Relational Database Queries 56 marks	
	Maximum 56 marks - Satisfy brief requirements: ● Marks awarded, as listed, (a) - (f) for SQL code which meets the expressed requirement ● Mark penalty applied if output does not match the form supplied for each question ● Mark penalty applied if date handling and string database lookups are not managed correctly ● Mark penalty applied if column aliases are not used when arithmetic calculation, concatenation, functions, or other output manipulation is used in a query ● Mark penalty applied if manual lookup/calculation is used for values from the database ● 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 3 Non Relational Database Queries - MongoDB 16 marks	
	Maximum 16 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) - (e) 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
PENALTIES APPLIED	
Limited/No push of work to FITGitLab server resulting in lack of development history.	If less than eight pushes showing a clear development history a *grade deduction of 8 marks applied*. Note that the expectation is that you would push significantly more times than this.
Incorrect Git author details used (see week 2 Applied notes Appendix) and/or Blank or non-informative Git Commit message/s (such as "Push 4" - *message must explain what the commit represents*)	If incorrectly identified pushes are made and/or non-informative Git Commit messages are used a *grade deduction of 8 marks will be applied*.
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 in Task 2 will result in a *grade deduction of 8 marks being applied.*
Use of SQL syntax in Task 2 which has not been covered in this unit	Such work will **NOT BE GRADED** (see page 6 of this brief)

Final Assignment Mark Calculation

- 80 marks from the Assignment 2B submission => 28 / 35
  PLUS
- 10 marks from the SQL Portfolio 2 =>   3.5 / 35
- 10 marks from the SQL Portfolio 3 =>   3.5 / 35

Total:100 marks, recorded as a grade out of 35