

FIT3171 Databases

Creating, Populating and Manipulating Database - Run Monash (RM)

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, 26 May 2022, 4:30 PM (AEST) / 2:30 (MYT)</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

Run Monash (RM) is a running carnival which is held on separate dates at various Monash campuses during different seasons (Summer, Autumn, Winter and Spring) of the year. The carnival naming convention that Run Monash uses is RM <season name> Series <campus name> <year>. So, for example, a carnival to be held during the Autumn season at the Clayton campus in 2022 will be named **RM Autumn Series Clayton 2022**.

Anyone can attend an RM Carnival, the carnivals are open to the public as well as Monash staff and students. A carnival is run on a particular date, in a particular location and only lasts for one day. RM only runs one carnival on any particular date. During a carnival a range of events are offered from the following list (only some may be offered):

- Marathon 42.2 Km
- Half Marathon 21.1 Km
- 10 Km Run
- 5 Km Run
- 3 Km Community Run/Walk

Run Monash expects to offer around 10 - 20 such events across all carnivals in a given year.

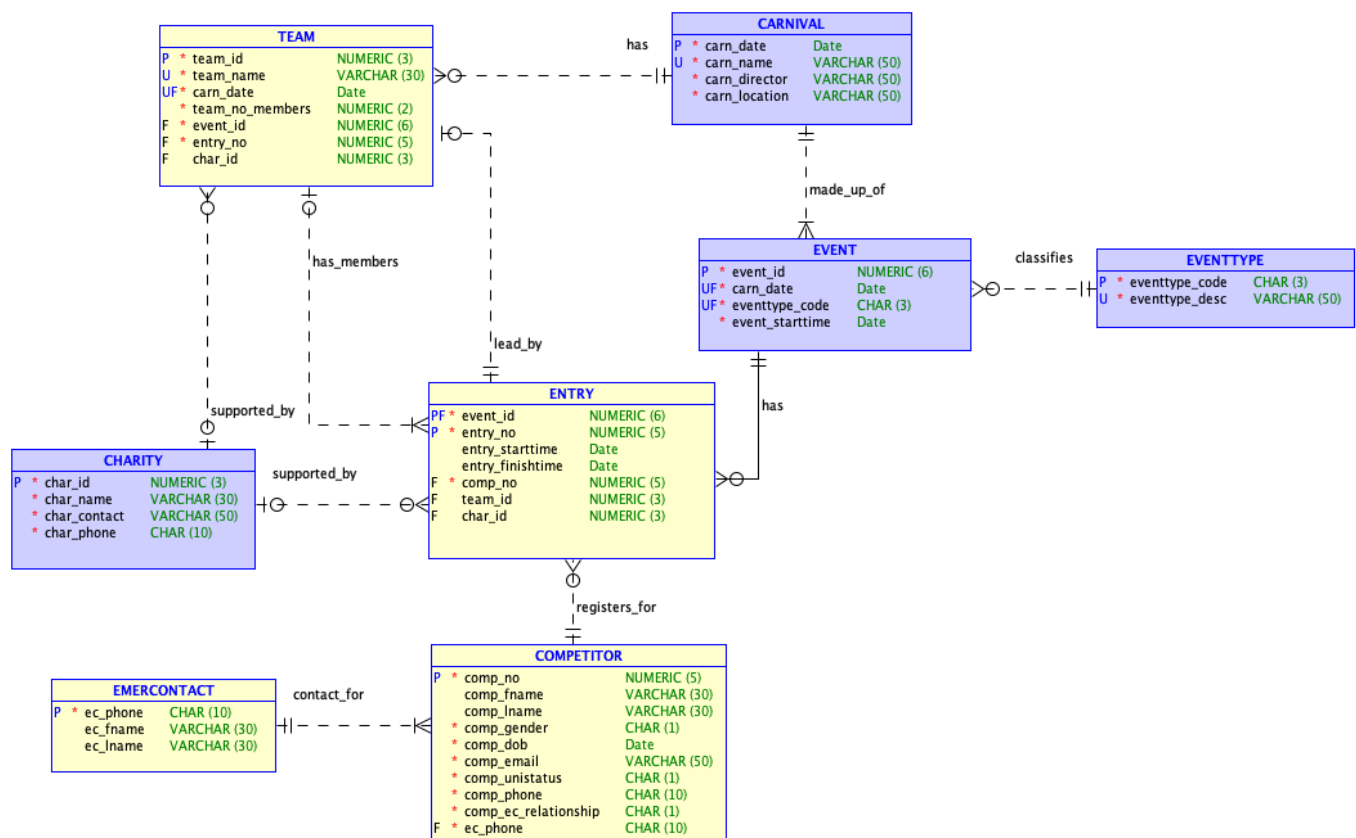
When a competitor initially registers for Run Monash, they are assigned a unique competitor number. A competitor is required to provide details of an emergency contact at the time of registration. The relationship to the competitor can be Parent (P), Guardian (G), Partner (T) or Friend (F).

When a carnival is being offered, Run Monash contacts all registered competitors and provides details of the carnival date and what events are on offer. A competitor can only attend one event at a particular carnival. Every entry is assigned an entry number. The entry number is reused in each event. Run Monash also, on the carnival day, using official timing devices, Run Monash records the entrants starting and finishing times.

Teams are identified by a unique team name which the team manager must select when they first create the team. This team manager can then add/invite other competitors from the carnival to join their team. Team names are unique only within a given carnival. A given team name may be reused by different competitors in a different carnival as teams are recreated for each carnival depending on which competitors have entered an event for the carnival.

Run Monash wishes to record, as part of the stored data, how many members are on each team. Teams may also nominate a charity for which they will raise funds, although not all teams will do so. All charities for which funds can be raised must first be approved by Run Monash. Note that an individual competitor may be supporting a charity as an individual and also the same or a different charity as a team member.

A model to represent this system has been developed:



The schema/insert file for creating this model (rm-schema-insert.sql) is available in the archive ass2-student.zip - this file partially creates the Run Monash 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.

Steps for working on Assignment 2

1. Download the Assignment 2 Required Files (ass2-student.zip) archive from Moodle
2. Extract the zip archive and place the contained files in your local (MoVE or local HDD) repository in the folder /Assignments/Ass2. Do not add the zip archive to your local repo. Then add, commit and push them to the FITGitLab server.
3. Run rm-schema-insert.sql
4. Write your answer for each task in its respective file (e.g. write your answer for Task 1 in T1-rm-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 (if you are using MoVE you will need to download them to your local HDD first - do not attempt to upload from MoVE). 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 final SQL scripts you submit **MUST NOT** contain SPOOL or ECHO commands (you may include them as you work but must remove them before submission). Please carefully read the Marking Guide document.

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 2 folder and regularly pushed to the FIT GitLab server to build a clear history of development of your approach. Any submission with less than five pushes to the FITGitLab server will incur a grade penalty of 10 marks. Please note five 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.

TASK 1: DDL (20 marks)

DATA DEFINITION LANGUAGE :
1) CREATE , DROP AND ALTER

For this task you are required to add to **T1-rm-schema.sql**, the **CREATE TABLE** and **CONSTRAINT** definitions which are missing from the supplied partial schema script in the positions indicated by the comments in the script.

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

Table name	Attribute name	Meaning
EMERCONTACT		
	ec_phone	Emergency contact's phone number (unique identifier)
	ec_fname	Emergency contact's first name
	ec_lname	Emergency contact's last name
COMPETITOR		
	comp_no	Unique identifier for a competitor
	comp_fname	Competitor's first name
	comp_lname	Competitor's last name
	comp_gender	Competitor's gender ('M' for male, 'F' for female, or 'U' for 'Undisclosed')
	comp_dob	Competitor's date of birth
	comp_email	Competitor's email



	comp_unistatus	Competitor's university student/staff status ('Y' for Yes or 'N' for No)
	comp_phone	Competitor's phone number
	comp_ec_relationship	Emergency contact relationship to competitor ('P' for Parent, 'G' for Guardian, 'T' for Partner, or 'F' for Friend)
ENTRY		
	entry_no	Entry number (unique for each event)
	entry_starttime	The entrant start time
	entry_finishtime	The entrant finish time
TEAM		
	team_id	Team identifier (unique)
	team_name	Team name
	team_no_members	Number of team members

To test your code you will need to first run the provided script **rm-schema-insert.sql** to create the other required tables. **rm-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 **rm-schema-insert.sql** which will cause all tables to be dropped and correct the issues in your script.

TASK 2: Populate Sample Data (20 marks)

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

Load the EMERCONTACT, COMPETITOR, ENTRY, and TEAM tables with **your own test data** using the supplied **T2-rm-insert.sql** script file, and SQL commands which will insert **as a minimum**, the following sample data:

- (i) 5 EMERCONTACT entries
 - of the 5 contacts added 3 must be contacts for more than one competitor
- (ii) 15 COMPETITOR entries
 - Have at least 10 competitors who are Monash student/staff
 - Have at least 2 competitors who are not Monash student/staff
 - Included at least 2 competitors who are under 18 years of age
- (iii) 30 ENTRY entries
 - Included at least 10 competitors
 - Included at least 6 events from 3 different carnivals
 - Have at least 5 competitors who join more than 2 events
 - Have at least 2 uncompleted entries (registration only)
- (iv) 5 TEAM entries
 - Have at least 2 teams with more than 2 members
 - At least one team with the same name in different carnivals

In adding this data you must ensure that the test data thoroughly tests the model as supplied, so as to ensure your schema is correct.

Your inserted data must conform to the following rules:

- (i) You may treat all of the data that you add as a single transaction since you are setting up the initial test state for the database.
- (ii) The numeric primary key values for this data should be hardcoded values (i.e. **NOT** make use of sequences) and must consist of values below 100.
- (iii) The data added must be sensible eg. entry finish times should be after entry start times with a sensible running duration.

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 *rm-schema-insert.sql* script.

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

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

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

PL/SQL may ONLY be used for Task 5.

TASK 3: DML (25 marks)**INSERT UPDATE AND DELETE**

Your answers for this task (Task 3) must be placed in the supplied SQL Script **T3-rm-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 quotes, such as 'RM Autumn Series Caulfield 2022' you may search in the database using that exact string, and when a name is supplied you may break the name into first name and last name, for example 'Jack Kai' can be split into 'Jack' and 'Kai'.

- (a) Oracle sequence is going to be implemented in the database for the subsequent insertion of records into the database for the COMPETITOR and TEAM tables.

Provide the CREATE SEQUENCE statements to create two sequences which could be used to provide primary key values for the COMPETITOR and TEAM tables. Both sequences should start at 100 and increment by 1. Immediately prior to the create sequence commands, place appropriate DROP SEQUENCE commands so they will cause the sequences to be dropped before being created if they exist. *Please note that there can only be these two sequences introduced and used in Task 3.*

[1 mark]

- (b) Record two competitors named 'Daniel Kai' and 'Annabelle Kai'. Both of them are Monash students. They nominated 'Jack Kai' (phone number: '0476541234'), their father, as their emergency contact.

Both of them have indicated that they would like to participate in 'RM Autumn Series Caulfield 2022' carnival and enter the '21.1 Km Half Marathon' event. For this carnival, Annabelle will raise funds to support the 'Amnesty International' charity, and Daniel will support 'Beyond Blue' charity.

Make these changes to the data in the database. You may make up sensible data for the rest of attributes. This entire registration should be treated as a single transaction.

[7 marks]

- (c) Two days later, 'Annabelle Kai' indicates that she would like to form a team called 'Kai Speedstars' for the 'RM Autumn Series Caulfield 2022' carnival. Run Monash staff check and confirm that the team name is not currently in use for this carnival, so inform her that such a team can be created. The new team will support the 'Beyond Blue' charity.

Make these changes to the data in the database. You may assume 'Annabelle Kai' is the only competitor of that name in the 'RM Autumn Series Caulfield 2022' carnival. These changes must be treated as a single transaction.

[7 marks]



update

- (d) One day later, 'Daniel Kai' indicates that he has suffered an injury in training and would like to change his entry (called downgrading) for the 'RM Autumn Series Caulfield 2022' carnival from the '21.1 Km Half Marathon' to the '10 Km Run'. He has also indicated that he will join the 'Kai Speedstars' team for this carnival.

Make these changes to the data in the database. You may assume 'Daniel Kai' is the only competitor of that name in the 'RM Autumn Series Caulfield 2022' carnival. These changes must be treated as a single transaction.

[5 marks]

- (e) One week later, 'Daniel Kai' indicates that his injury has gotten a lot worse and that he will need to withdraw from the 'RM Autumn Series Caulfield 2022' carnival. 'Daniel Kai' also indicates that he is looking forward to competing in the next 2022 carnival.

delete

On the other hand, 'Annabelle Kai' has asked a few friends to join her team for this carnival but she is not sure if any have actually taken up the offer, so she directs Run Monash that her team, 'Kai Speedstars', should be disbanded. She will still participate in the carnival as an individual runner and she will support the 'Beyond Blue' charity

Make these changes to the data in the database. You may assume 'Daniel Kai' and 'Annabelle Kai' are the only competitors of that name in the 'RM Autumn Series Caulfield 2022' carnival. These changes must be treated as a single transaction.

[5 marks]

TASK 4: DATABASE MODIFICATIONS (15 marks)

Your answers for these tasks (Task 4) must be placed in the supplied SQL script

T4-rm-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.

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

- (a) Run Monash has decided that they would like to have the elapsed time (finish time - start time) for a runner in a particular event stored as part of the system, rather than having to calculate it every time it is required.

Add a new attribute which will record the runner's elapsed time in an event. The time should be stored as the number of minutes elapsed to two decimal places eg. 26.82

This attribute must be initialised to the correct elapsed time based on the data which is currently stored in the system. You should note that the system may contain registrations for future events which currently do not have either a start or finish time.

[2 marks]

- (b) Some teams indicated that they would like to support more than one charity in future carnivals. They also want to store the percentage (0 to 100) of total raised funds that goes to each charity.

For example, Team 'Super Stars' supports 'Beyond Blue' and 'Amnesty International' charities in the 'RM Autumn Series Caulfield 2022' carnival. 70% of raised funds will be donated to 'Beyond Blue' and the other 30% will be donated to 'Amnesty International'.

Modify the database structure to meet this new requirement.

[6 marks]

- (c) Run Monash requires a lot of officials to run a carnival so they decided to allow registered competitors to act as officials in future carnivals. A competitor may register as an official in multiple carnivals. Each official has one role in a carnival but they may change to another role in a different carnival. The roles include time keeper, marshal, starter, and first aid. The list of roles may be expanded as required.

Modify the database structure to meet this new requirement

[7 marks]

TASK 5: PL/SQL (20 marks)

Your answers for these tasks (Task 5) must be placed in the supplied SQL script
T5-rm-alter.sql

The required PL/SQL must be made to the "live" database (the database *after* you have completed tasks 1, 2, 3, and 4) **not** by editing and executing your schema file again. Before carrying out the work below, please ensure that you have completed tasks 1, 2, 3 and 4 above.

- (a) Run Monash have noted that several competitors have enrolled in multiple events in the same carnival which they do not wish to occur. Write a trigger to prevent this issue in future carnivals.

[3 marks]

- (b) Run Monash has decided to record the fastest elapsed time for each event type for all future carnivals. Add two new attributes in the EVENTTYPE table named eventtype_record and eventtype_recordholder to store the fastest elapsed time for each event type and the competitor (competitor number) who holds the record. You may assume that only a single competitor will hold each record and that the record holder will only be replaced by a new single competitor if their elapsed time is less than the current eventtype_record.

After making these changes, write a single trigger which will automatically:

- calculate the elapsed time attribute added in 4(a) when the finish time is updated (ie. the competitor finished the run), and
- update the 'eventtype_record' and 'eventtype_recordholder' when the new elapsed time is faster than the current eventtype_record time. Note that, initially, the value of these two attributes is null.

[8 marks]

- (c) Write a stored procedure called `event_registration` which handles the competitor's entry for an event. When a competitor registers for an event, they may also choose to create or join a team. Your procedure must take four input parameters: competitor number, carnival date, event type description and team name. The procedure must:
- check if the input carnival date and event name are valid
 - check if the competitor wants to create/join a team (ie. they provide a team name) and, if provided, whether the team name exists for the input carnival:
 - If the team does not exist then the procedure should add a new team and assign this competitor as the team leader, or
 - if the team exists then the procedure should add the entrant into the existing team

The output from this procedure, returned by an OUT parameter, must be a status string indicating successful registration in the event or the issue that prevented registration on this attempt. For a failed registration you are not required to report all possible errors which would occur, only the first found by your procedure.

[9 marks]

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

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

Submission Requirements

Due Date: Thursday 26th May 2022 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:

- T1-rm-schema.sql
- T2-rm-insert.sql
- T3-rm-dm.sql
- T4-rm-alter.sql
- T5-rm-plsql.sql

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 five independent SQL scripts. The individual files must also have been pushed to the FIT GitLab server with an appropriate history as you developed your solutions (a minimum of five pushes - 1 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 actually submitting, that you download the submission and double-check its contents.

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

Submission status

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

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

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

Resubmission

If you wish to resubmit your assignment you must email your tutor, provide your full details as listed in the Unit Information (see 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 2 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 requirements, If it does the answer will be accepted and graded appropriately.

Marking Criteria	Items Assessed
TASK 1 DDL 20 marks	
DDL Creation of tables	Maximum 9.5 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 generated schema used
DDL implementation of non PK database constraints	Maximum 10.5 marks - Non PK Constraints: <ul style="list-style-type: none"> • Marks awarded for correct implementation of non PK constraints: <ul style="list-style-type: none"> ◦ CHECK ◦ UNIQUE ◦ FK
TASK 2 Populate Sample Data 20 marks	
Insert of required items test data	Maximum 10 marks - Insert of data: <ul style="list-style-type: none"> • Marks awarded for correct insert of required data <ul style="list-style-type: none"> ◦ 5 EMERCONTACT entries ◦ 15 COMPETITOR entries ◦ 30 ENTRY entries ◦ 5 TEAM entries • Marks awarded for correct management of transactions
Insert of valid test data	Maximum 10 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 25 marks	
	Maximum 25 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 15 marks	
	Maximum 15 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 PL/SQL 20 marks	
	Maximum 20 marks - Satisfy brief requirements: <ul style="list-style-type: none"> Marks awarded (a) - (c) for PL/SQL code which meets the expressed requirement. Marks awarded for writing a test harness for each question (a) - (c) which includes both successful and failed tests (all errors your code raises must be tested) .
Penalty Criteria	Penalty Applied
Limited/No push of model to FITGitLab server resulting in lack of development history.	If less than five pushes showing a clear development history a grade deduction of 10 marks applied . Note that the expectation is that you would push significantly more times than this.
Use of <ul style="list-style-type: none"> PL/SQL in Task 1 - Task 4, VIEWS, and/or SET ECHO or SPOOL commands 	Use of PL/SQL in Task 1 - Task 4, VIEWS, inclusion of SET ECHO/SPOOL commands in submitted scripts will result in a grade deduction of 10 marks being applied . *** PL/SQL may ONLY be used for Task 5 ***
Incorrect application of relational database principles	Marks will be deducted, based on the particular question, where basic relational model principles have been violated. For example, creation of a table which is not in 3NF