FIT2094 Databases - Assignment 2

Creating, Populating and Manipulating Databases - Paris Arrow Transit (PAT)

Purpose	Students will be asked to implement, via SQL, a small database in the Oracle RDBMS from a provided logical model case study, followed by the insert of appropriate data to the created tables. Once populated the database will be used to carry out specified DML commands and make specified changes to the database structure via SQL. Students will then use SQL and NoSQL to write queries to produce specified output. This task covers learning outcomes:
	1. Apply the theories of the relational database model.
	3. Implement a relational database based on a sound database design.
	4. Manage data that meets user requirements, including queries and transactions.
	5. Contrast the differences between non-relational database models and the relational database model.
Your task	This is an open book, **individual task**. The final output for this task will be a set of tables and data implemented in the Oracle RDBMS. In addition students will create a set of relational (Oracle) and non relational (MongoDB) queries which meet the user requirements.
Value	**40%** of your total marks for the unit
Due Date	Wednesday, **30th October 2024, 4:30 PM** **(note: staff support is unavailable after business hours)**
Submission	● Via Moodle Assignment Submission ● FIT GitLab check ins will be used to assess history of development
Assessment Criteria	● Application of relational database principles. ● Handling of transactions and the setting of appropriate transaction boundaries. ● Application of SQL statements and constructs to create and alter tables including the required constraints and column comments, populate tables, modify existing data in tables, and modify the "live" database structure to meet the expressed requirements (including appropriate use of constraints). ● Application of SQL select statements to produce outputs that meet user requirements. ● Mapping of relational database data into non relational database data structure. ● Application of MongoDB operations to produce outputs that meet user requirements.
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 design a model for Paris Arrow Transit (PAT). Paris Arrow Transit is a private company subcontracted by the Olympic Federation to transport officials during the Olympic competition. The company realises that it needs a completely new computerised system to more efficiently manage and record its services during the Games. You have been asked to develop a database system that can meet PAT's needs, which are detailed below.

PAT owns a fleet of vehicles. Each vehicle is identified by its 17-character vehicle identification number (VIN), the company also records the registration plate (7 characters such as AB126FD), the make, such as Peugeot, the current odometer reading and the number of passengers which the vehicle can transport.
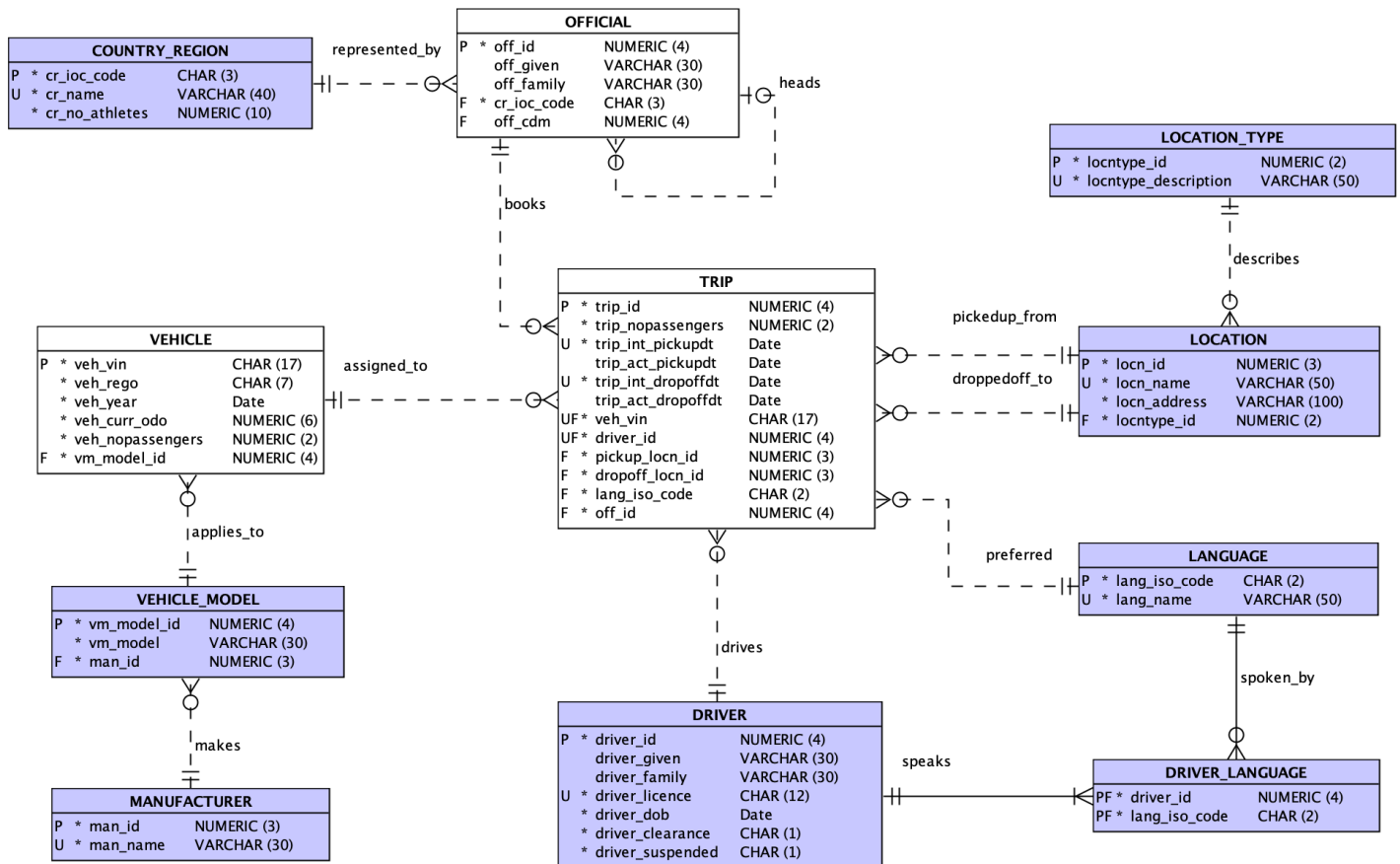
206 National Olympic Committee's (NOC) will compete at the Paris Games - each represents a particular country or region of the world. For each country/region, an identifying IOC code is recorded as well as the name of the country/region and its number of athletes. Each NOC will enter a team into the competition. The PAT system must track all officials who are part of these teams since they are the ones who will need to book official vehicles. One member of each NOC team is designated as the Chef De Mission (the team manager). PAT will record the official's Olympic ID and name for each official. The system records the manager (Chef De Mission) for each official. The Chef De Missions is listed as an official but has no manager since they are the team manager.

An official will book a trip with PAT to transport members of their team between various locations. A trip only involves a single vehicle. An official may use PAT's services multiple times (even during a single day); the only limiting factor is whether a suitable vehicle and driver is available at the date and time they wish to book travel. The vehicle is booked in the name of the official making the booking; the details of the actual passengers do not need to be recorded. The official booking the trip will also indicate the preferred language to be spoken during the trip, so that the travellers can be understood by the driver and vice versa. Only a single preferred language for the trip will be indicated/recorded. The intended pick-up date and time and the projected drop-off date and time are recorded for each booked trip. In addition, PAT records the number of passengers needing transport for each booked trip.

PAT vehicles are driven by the company's drivers. Each driver is assigned a unique driver ID. The driver's name (given and family), licence number (12 characters in length), date of birth and the level of security clearance granted to the driver are recorded. Due to issues/problems that may arise, a driver may need to be suspended while a particular matter is investigated; while suspended a driver cannot drive any PAT vehicle - the system must be able to flag this. Driver security clearances are set at either F to represent Full or R to represent Restricted; the default should be R. Only a single driver drives for a particular booked trip.

PAT records the languages that a driver speaks – some drivers speak several languages. To record languages, PAT will use ISO639-1 two-character language codes, for example, EN as English and ZH as Chinese – PAT records the ISO639-1 code and the name of the language.

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

MONASH University

COUNTRY_REGION

P	*	cr_ioc_code	CHAR (3)
U	*	cr_name	VARCHAR (40)
	*	cr_no_athletes	NUMERIC (10)

represented_by

OFFICIAL

P	*	off_id	NUMERIC (4)
		off_given	VARCHAR (30)
		off_family	VARCHAR (30)
F	*	cr_ioc_code	CHAR (3)
F		off_cdm	NUMERIC (4)

heads

books

LOCATION_TYPE

P	*	locntype_id	NUMERIC (2)
U	*	locntype_description	VARCHAR (50)

describes

TRIP

P	*	trip_id	NUMERIC (4)
	*	trip_nopassengers	NUMERIC (2)
U	*	trip_int_pickupdt	Date
		trip_act_pickupdt	Date
U	*	trip_int_dropoffdt	Date
		trip_act_dropoffdt	Date
UF	*	veh_vin	CHAR (17)
UF	*	driver_id	NUMERIC (4)
F	*	pickup_locn_id	NUMERIC (3)
F	*	dropoff_locn_id	NUMERIC (3)
F	*	lang_iso_code	CHAR (2)
F	*	off_id	NUMERIC (4)

pickedup_from

droppedoff_to

LOCATION

P	*	locn_id	NUMERIC (3)
U	*	locn_name	VARCHAR (50)
	*	locn_address	VARCHAR (100)
F	*	locntype_id	NUMERIC (2)

preferred

LANGUAGE

P	*	lang_iso_code	CHAR (2)
U	*	lang_name	VARCHAR (50)

spoken_by

VEHICLE

P	*	veh_vin	CHAR (17)
	*	veh_rego	CHAR (7)
	*	veh_year	Date
	*	veh_curr_odo	NUMERIC (6)
	*	veh_nopassengers	NUMERIC (2)
F	*	vm_model_id	NUMERIC (4)

assigned_to

applies_to

drives

VEHICLE_MODEL

P	*	vm_model_id	NUMERIC (4)
	*	vm_model	VARCHAR (30)
F	*	man_id	NUMERIC (3)

makes

MANUFACTURER

P	*	man_id	NUMERIC (3)
U	*	man_name	VARCHAR (30)

DRIVER

P	*	driver_id	NUMERIC (4)
		driver_given	VARCHAR (30)
		driver_family	VARCHAR (30)
U	*	driver_licence	CHAR (12)
	*	driver_dob	Date
	*	driver_clearance	CHAR (1)
	*	driver_suspended	CHAR (1)

speaks

DRIVER_LANGUAGE

PF	*	driver_id	NUMERIC (4)
PF	*	lang_iso_code	CHAR (2)

The schema/insert file for creating this model (pat_initialSchemaInsert.sql) is available in the archive ass2_student.zip. This file partially creates the Pets First tables and populates several of them (those shown in purple on the supplied model). Please read this schema carefully and be sure you understand the various data requirements.

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

1. The ass2-student.zip archive also contains seven script files to code your answers in. **You MUST ensure these files are regularly pushed to the GitLab server <u>so that a clear development history is available</u> for the marker to verify your work (a minimum of fourteen pushes are required - 2 pushes per file).** In each file, you *must* fill in the header details with your name and student ID before beginning work. Your SQL script files **must not include any SPOOL or ECHO commands**. Although you might include such commands when testing your work, **they must be removed before submission** (a -10 mark grade penalty will be applied if your documents contain spool or echo commands).

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

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

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

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

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

Steps for working on Assignment 2

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

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

3. Examine the extracted files, i.e. read carefully through them and ensure you understand their content.

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

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

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

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

8. Finally, when you have completed all tasks, separately run each SQL or MongoDB **as a script (not as individual statements) and ensure there are no errors**. Upload all required files from your local repository to Moodle. Check that the files you have uploaded are the correct files (download them from Moodle into a temporary folder and check they are correct). After you are sure they are correct, submit your assignment.

****

For all assignment tasks, **your final script must run as a script without errors except for SQL errors generated by the DROP TABLE/DROP SEQUENCE statements**. **Any task's script that runs with an error will receive a maximum grade of half of the task's available marks -1**. For example, if your task 1 script runs with an error, regardless of the code contained, your maximum grade will be 15/2 => 7.5 - 1 = 6.5 marks. This will be applied even if the error is simply a forgotten semicolon. Thus, **please carefully check that your final scripts for all tasks run without error**.

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

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

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

Views must not be used in completing these tasks.

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

****

GIT STORAGE

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

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

****

Assignment Tasks

TASK 1: DDL [15 mks]

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

For this task, you must add to T1-pat-schema.sql the CREATE TABLE and CONSTRAINT definitions, which are missing from the supplied partial schema script, in the positions indicated by the script's comments. The table below details the attributes' meaning in the missing three tables. You **MUST** use exactly the same relation and attribute names shown in the data model above to name the tables and attributes 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
OFFICIAL		
	off_id	Identifier for an official
	off_given	Given name for the official
	off_family	Family name for the official
	cr_ioc_code	IOC country code for the official
	off_cdm	Identifier for Chef De Mission for the official
VEHICLE		
	veh_vin	Identifier for vehicle
	veh_rego	Registration plate of vehicle
	veh_year	Year of manufacture of vehicle
	veh_curr_odo	Current odometer reading of vehicle
	veh_nopassenngers	Number of passengers vehicle can seat
	vm_model_id	Identifier for vehicle_model
TRIP		
	trip_id	Identifier for a trip
	trip_nopassengers	Number of passengers for the trip
	trip_int_pickupdt	Intended pickup date and time for the trip
	trip_act_pickupdt	Actual pickup date and time for the trip
	trip_int_dropoffdt	Intended drop-off date and time for the trip
	trip_act_dropoffdt	Actual drop-off date and time for the trip
	veh_vin	Identifier for a vehicle
	driver_id	Identifier for a driver
	pickup_locn_id	Identifier for the pick-up location
	dropoff_locn_id	Identifier for the drop-off location
	lang_iso_code	ISO639-1 two-character language codes
	off_id	Identifier for an official

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

TASK 2: INSERT [20 mks]

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

Load the VEHICLE, OFFICIAL and TRIP tables with **your own test data** using the supplied **T2-pat-insert.sql** script file. Write SQL commands that will insert _as a minimum_ (i.e. you may and should insert more) the following sample data:

  (i)   10 VEHICLE entries
       ● Include at least three vehicle models
  (ii)   10 OFFICIAL entries
       ● Include at least four IOC Countries/Regions
  (iii)   20 TRIP entries
       ● Include at least five vehicles used in more than one trip
       ● Include at least five officials booked more than one trip
       ● Include at least five drivers
       ● Include at least five languages
       ● Include at least two parallel trips (i.e. trips that have the same intended pick-up/drop-off date times and the same pickup/drop-off locations)

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

Your inserted data must conform to the following rules:
  1. Treat all the data you add _as a single transaction_ since you are setting up the initial test state for the database.
  2. The primary key values for this data should be hardcoded values (i.e. **NOT** make use of sequences). If the primary key attribute's datatype is number, it must consist of values below 100.
  3. Trip dates used must be chosen between the 20th July 2024 and 15th August 2024.
  4. The data added must be sensible (e.g., the drop-off date should be after the pick-up date, the vehicle must accommodate the requested number of passengers, etc.).

For this task **ONLY,** Task 2, you may manually look up and include values for the loaded tables/data directly where required. However, you can still use SQL to get any non-key values if you wish. **In carrying out Task 2, you must not modify any data or add any further data to the tables populated by the pat_initialSchemaInsert.sql script. Design your test data to get output for the SQL scripts/queries specified below - this may require you to add further data as you complete the required tasks.**

TASK 3: DML [20 mks]

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

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

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

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

For any following SQL tasks, **your SQL must correctly manage transactions and use sequences to generate new primary keys for numeric primary key values** (a new primary key value cannot be hardcoded as a number or value).

You must ONLY use the data provided in the questions' text.

For Task 3, you must complete the following sub-tasks in the same order they are listed. Where you have been supplied with a string in italics, such as *La Beaujoire Stadium*, you may search in the database using the string *as listed*. Where a particular case for a word is provided, *you must only use that case (same spacing, case, etc) in your SQL code*. When a name is supplied, you may break the name into first name and last name. For example, *James SMITH* can be split into James and SMITH; again, note that the case must be maintained as it was supplied. **Failure to adhere to these requirements, such as changing the case of a provided string, will result in a grade penalty**.

(a) Oracle sequences will be implemented in the database to insert records for the OFFICIAL and TRIP tables.

Provide the CREATE SEQUENCE statements to create sequences that could be used to provide primary key values for the OFFICIAL and TRIP tables. These sequences must start at 100 and increment by 10. Immediately before the create sequence commands, place appropriate DROP SEQUENCE commands so that the sequences will be dropped before being created if they exist. Please note that these are the ONLY sequences that can be introduced and used in Task 3.

[1 mark]

Questions 3b, 3c, and 3d are related questions. You can use the information below in any part of Task 3.

(b) An official and a vehicle needed to be stored in the database.

The official's name is *Franklin Gateau* from *St Vincent and the Grenadines*. Franklin is the only official from this country and the Chef De Mission for this country.

The vehicle is an *ALPHARD* manufactured by *TOYOTA.* You may assume that *TOYOTA* only produced one model called *ALPHARD*. The vehicle identification number (VIN) is *1C4SDHCT9FC614231*, which seats up to 6 passengers.

Take the necessary steps in the database to record the required entries for this vehicle and official. When inserting this data, you may make up (invent) any other required information.

[5 marks]

(c) Franklin booked two trips.

The first trip was booked for 30 July 2024 at 12:30 PM from *Olympic and Paralympic Village* to *Porte de la Chapelle Arena,* and it was scheduled to arrive 1 hour and 30 minutes later.

The second trip was booked for 30 July 2024 at 8:00 PM from *Porte de la Chapelle Arena* to *Olympic and Paralympic Village,* and it was scheduled to arrive 1 hour and 15 minutes later.

For both trips, the vehicle assigned was the Toyota Alphard with VIN *1C4SDHCT9FC614231*; the allocated driver was Claire Robert (licence number: *55052a543210*), and the preferred language was *English*.

Take the necessary steps in the database to record the required trip entries. Both trip bookings <u>must be treated as a single transaction</u>. When inserting this data, you may make up (invent) any other required information.

[8 marks]

(d) As scheduled, on 30 July 2024 at 12:30 PM, Claire Robert picked up the *St Vincent and the Grenadines* team from the *Olympic and Paralympic Village* and dropped them at the *Porte de la Chapelle Arena.* The actual trip took 1 hour and 45 minutes.

Then, at 5 PM on the same day, Claire got into an accident. Due to the short notice and the unavailability of other drivers, all booked (incomplete) trips allocated to Claire for the rest of the day must be cancelled (i.e. removed from the system). PAT informed the officials who booked the trips and reimbursed them with taxi vouchers.

Make these required changes to the data in the database.

[6 marks]

TASK 4: DATABASE MODIFICATIONS (13 marks):

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

The required changes must be made to the "live" database (the database *after* you have completed tasks 1, 2 and 3, in which other users must be assumed to be active). You MUST not edit and execute your schema file again. Before completing the work below, please ensure you have completed tasks 1, 2 and 3 above. If, in answering these questions, you need to create a table, please place a drop table statement immediately before your create table statement.

(a) PAT would like to store each official's role. The list of roles, which includes General, Administrator, Head Coach, Coach, and Physician, will remain unchanged. The default role is General. The Chef de Mission of each country must be assigned an Administrator role.

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

[5 marks]

(b) PAT would like to allow complaints about the driver's behaviour on a particular trip. The official who requested the trip will submit these complaints. Multiple complaints may be lodged for a particular trip. An official cannot make two complaints for a particular trip at the same time.

PAT provides some categories for the complaints. The current categories are *late arrival, rude behaviour, poor driving, and failing to assist*. Each category has a specific demerit point: a late arrival or failure to assist incurs one demerit point, and rude behaviour or poor driving incurs two demerit points. PAT wants to add more categories in the future.

The system stores each complaint's date, time, category, and detailed comment. It also stores the trip for which the complaint was made.

The PAT staff member follows up on the complaint and investigates whether it is valid. If it is, it will be flagged in the system as a valid complaint.

Change the database structure to support these new business rules. You are not required to add the complaint data (you must add the category data as listed in this task). You only need to provide the structure so PAT can store complaints.

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

[8 marks]

TASK 5: SQL Queries [15 mks]

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

If you need to add further data for this task, you must return and add it as part of task 2 and then rerun pat_initialSchemaInsert.sql and all tasks (tasks 1, 2, 3 and 4) before running task 5.

You can only code **a single select statement for each question** below. Note that an SQL select statement begins with the SELECT keyword and ends with a semicolon (;) - within this statement, the SELECT keyword can be used multiple times.

Where you have been supplied with a string in *italics*, you must search the database using the provided string precisely as supplied. Where you need to show a full name, you must not have any extra spaces (e.g., leading space or extra space in the middle of the name).

Please remember VIEWS or PL/SQL (including anonymous blocks BEGIN…END) **must not be used**. **Your SQL code used must be restricted to the syntax covered in your unit.**

Sample partial output showing the form of what you must produce is provided. Note that this is the form of the output ONLY, i.e., the appearance only; the data you return will differ.

(a) Paris Arrow Transport wishes to know how often all locations have been used as pick-up or drop-off locations (trips that have not been completed must be excluded from this count). The report should show the location ID, name, address, type of location, and the number of times the location has been used for pick-up or drop-off. If a particular location has been used once for pick-up and twice for drop-off, the pickup/dropoff count will be three.

Sample partial output is shown below (your output must have the same format and column headings):

```
LOCN_ID LOCN_NAME                                 LOCN_ADDRESS                                         LOCNTYPE_DESCRIPTION PICKUP/DROPOFF_COUNT
------- ----------------------------------------  -------------------------------------------------   -------------------- --------------------
    113 Olympic and Paralympic village            D1, 93200 Saint-Denis, France                       Village                                 9
    114 Champions Park                            75116 Paris, France                                 Tourist attraction                      5
    110 Pierre Mauroy Stadium                     261 Bd de Tournai, 59650 Villeneuve-d'Ascq, France  Stadium                                 4
    117 Tuileries Garden                          75001 Paris, France                                 Tourist attraction                      4
    120 The Basilica of the Sacred Heart of Paris 35 Rue du Chevalier de la Barre, 75018 Paris, France Tourist attraction                     4

    101 Bordeaux Stadium                          Cr Jules Ladoumegue, 33300 Bordeaux, France         Stadium                                 0
    106 La Beaujoire Stadium                      333 Rte de Saint-Joseph, 44300 Nantes, France       Stadium                                 0
    108 Parc des Princes                          24 Rue du Commandant Guilbaud, 75016 Paris, France  Stadium                                 0
```

The output must be ordered to show the location with the most pick-ups/drop-offs first. If several locations have the same number of pick-ups/drop-offs, then order the output by the location ID.

[7 marks]

(b) Paris Arrow Transport would like a report for its weekly pay run covering the period from August 1st to August 7th, 2024, inclusive. Drivers are paid $45.42 per hour for the time they drive a trip. All trips during this period have been completed, and any cancellations have been removed from the system.

If a driver has not completed any trips during this period, show their total payment as 'No Trips'. The total payment must be displayed in the following format: $1234.56. Your output must show the driver ID, the driver's full name, and the total payment. Order the output in ascending order of driver_id.

Sample partial output is shown below (your output must have the same format and column headings):

```
DRIVER_ID FULLNAME           TOTAL_PERIOD_PAYMENT
--------- ------------------ --------------------
     2001 Pierre Martin                    $41.63
     2002 Marie Dupont       No Trips
     2003 Louis Dubois       No Trips
     2004 Antoine Lefevre    No Trips
     2005 Sophie Bernard                    $4.54
     2006 Arman                           $285.39
     2007 Jean Moreau        No Trips
```

[8 marks]

TASK 6: MongoDB [12 mks]

Your answers for this task (Task 6) must be placed in the supplied sql file **T6-pat-json.sql** and the supplied MongoDB script file **T6-pat-mongo.mongodb.js.**
You must not add any further comments to the supplied MongoDB script file nor remove/rename any comments indicated by //

(a) Write an SQL statement in **T6-pat-json.sql** to generate a collection of JSON documents using the following structure/format. Each document in the collection represents a driver, their details, and a summary of their completed trip details. Note that _id in this structure is the driver's ID, and no_of_trips is the number of trips that the driver has completed. You only need to include drivers who already have at least one completed trip.

```
{
    "_id": 2012,
    "name": "Mansour",
    "licence_num": "33022B678901",
    "no_of_trips": 2,
    "suspended": "N",
    "trips_info": [
        {
            "trip_id": 21,
            "veh_vin": "1D4PU4GK7BW972165",
            "pick-up": {
                "location_id": 102,
                "location_name": "Bercy Arena",
                "intended_datetime": "09/08/2024 17:35",
                "actual_datetime": "09/08/2024 17:38"
            },
            "drop off": {
                "location_id": 114,
                "location_name": "Champions Park",
                "intended_datetime": "09/08/2024 18:00",
                "actual_datetime": "09/08/2024 18:05"
            }
        },
        {
            "trip_id": 9,
            "veh_vin": "1C4SDHCT9FC613386",
            "pick-up": {
                "location_id": 112,
                "location_name": "Roland Garros Stadium",
                "intended_datetime": "01/08/2024 14:00",
                "actual_datetime": "01/08/2024 14:01"
            },
            "drop off": {
                "location_id": 116,
                "location_name": "Louvre Museum",
                "intended_datetime": "01/08/2024 14:25",
                "actual_datetime": "01/08/2024 14:28"
            }
        }
    ]
}
```

[6 marks]

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

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

[1 mark]

(c) List the name and licence number of all drivers who have completed trips to *Champions Park* or *Porte de La Chapelle Arena*.

[2 marks]

(d) It has been discovered that driver *Antoine Lefevre* ("_id":2004) made a trip from *Tuileries Garden* (location id: 117) to *Sainte-Chapelle* (location id: 118) on 10/08/2024 and forgot to record it. Because of this mistake, they have now been suspended for failing to follow policy.

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

Write the necessary MongoDB commands to add this trip to Antoine's record, and change their status to suspended. You may make up other required information when adding the trip.

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

[3 marks]

Submission Requirements

Due Date: **Wednesday, 30th October 2024, 4:30 PM**

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

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

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

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

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

Late submission will incur penalties of -5 marks for every 24 hours the submission is late.

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

Please note we **cannot mark any work on the GitLab Server**; you must 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 before submitting, that you download the submission and double-check its contents.

Your assignment **MUST** show a "Submitted for grading" status 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**.

Marking Guide

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

Marking Criteria	Items Assessed
TASK 1 DDL 15 marks	
DDL Creation of tables	**Maximum 7 marks - Create table:** ● Marks awarded for correct table DDL ● Marks awarded for correct attributes/data types ● Marks awarded for correct PK definition ● Mark penalty applied if different table/attribute names used than expressed in the supplied data model ● Mark penalty applied if different order of attributes used than expressed in the supplied data model ● No marks awarded if generated schema used
DDL implementation of non-PK database constraints	**Maximum 8 marks - Non-PK Constraints:** ● Marks awarded for correct implementation of non-PK constraints ● Marks awarded for correct use of column comments
TASK 2 Data Insert 20 marks	
Insert of required items test data	**Maximum 10 marks- Insert of data:** ● Marks awarded for correct insert of required data ● Marks awarded for correct management of transactions
Insert of valid test data	**Maximum 10 marks - Valid data inserted:** ● Marks awarded for validity of data inserted 　○ meets the requirements expressed in the assignment brief ● Marks awarded for correct management of dates when inserting
Task 3 DML 20 marks	
	Maximum 20 marks - Satisfy brief requirements: ● Marks awarded (a) - (d) 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 13 marks	
	Maximum 13 marks - Satisfy brief requirements: ● Marks awarded (a) - (b) for SQL code which meets the expressed requirement (including appropriate use of constraints). In making these modifications there must be no loss of existing data or data integrity within the database. ● Mark penalty applied if commit not used appropriately ● Mark penalty applied if column comments not used where required

Task 5 SQL Queries 15 marks	
	Maximum 15 marks - Satisfy brief requirements: ● Marks awarded, as listed, (a) - (b) 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 6 Non Relational Database Queries - MongoDB 12 marks	
	Maximum 12 marks - Satisfy brief requirements: ● Maximum of 7 marks awarded for creation of a JSON document which matches the supplied document format ● Marks awarded, as listed, (b) - (d) for MongoDB code which meets the expressed requirement ● Mark penalty applied if field names and predicates (such as "&eq") are not enclosed in double quotes ● Statements which do not execute correctly in MongoDB will be awarded a maximum of 50% of the available marks less 1 mark. For example, if a question is worth 6 marks and runs with an error in MongoDB the *maximum* mark awarded will be 2 marks

Correct use of Git 5 marks	
	- Marks awarded for a minimum of fourteen pushes (**two per file**) showing a clear development history of the work for Assignment 2 - Marks awarded for correct Git author details used in pushes - Marks awarded for the use of meaningful commit messages (i.e. not blank or of the form "Push1")
Penalties	
Use of - VIEWs - SET ECHO or SPOOL commands, and/or - PL/SQL	Use of VIEWS, inclusion of SET ECHO/SPOOL, and/or PL/SQL commands will result in a *grade deduction of 10 marks being applied.*
Incorrect file names	If file names do not follow the requirement (i.e., they are changed from the supplied filenames), or if a zip file is submitted instead of seven individual files, a *grade deduction of 5 marks will be applied*.
Late submission	**-5 marks for each 24 hours late or part thereof**

Final Assignment Mark Calculation

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