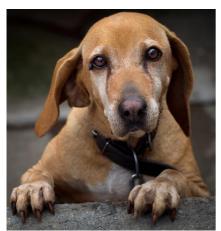SEMESTER 2 2017

DOG PARLOUR DATABASE

MARKS: 100 Marks (15%)

DUE DATE: 11:59PM, 26th of October 2017

WHAT TO SUBMIT: SQL Script accompanied by a PDF Report

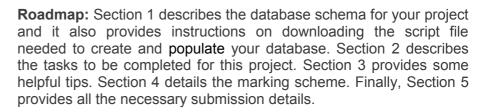**WHERE TO SUBMIT:** Electronic Submission via Blackboard

OVERVIEW

The goal of this project is to gain practical experience in applying several database management concepts using the Oracle DBMS.

Your task is to first populate your database with the supplied schema and data, then design, implement, and test the appropriate queries to perform the tasks explained in the next sections.

You must work on this project **individually**. The standard academic integrity rules apply. Plagiarism will be taken seriously and punished appropriately in accordance to the guidelines published by the School of Information Technology and Electrical Engineering (ITEE).

Roadmap: Section 1 describes the database schema for your project and it also provides instructions on downloading the script file needed to create and populate your database. Section 2 describes the tasks to be completed for this project. Section 3 provides some helpful tips. Section 4 details the marking scheme. Finally, Section 5 provides all the necessary submission details.

GOOD LUCK WITH YOUR PROJECT!

SECTION 1: DOG PARLOUR DATABASE

THE DATABASE

Doggo Paradise is an Australian dog grooming company. It has established a store in each Brisbane suburb. Monumental growth is expected for the company within this financial year and its legacy system will not be able to handle the demand. Hence, a migration from *Doggo Paradise's* old proprietary point of sales systems to a new database software is required, before the company expands. A data dump was created from the old system and it does not meet the business objectives that would be expected for a business expanding so rapidly.

The business dataset is stored in 7 different tables according to the following schema:

1. **DOG_BREEDS** – includes a list of all dog breeds

2. **DOGS** – includes details regarding a dog owned by a specific customer

3. **CUSTOMERS** – highlights all of the basic information required by a customer to register with *Doggo Paradise,* such as their name and date of birth for example.

4. **STORES** – this table will showcase the area in which the store presides as well as it's ID

5. **SERVICE_HISTORY** – will include the dogs ID, store IDs and service IDs. The table will also showcase whether the service has been finished and will have a generated message which is dependent on the finished column.

6. **SERVICES_HISTORY_DETAIL** – this table will highlight the specific service's ID, its start time and end time as well as its name.

7. **SERVICES** – this table highlights all of the services that *Doggo Paradise* conducts and also establishes their price.
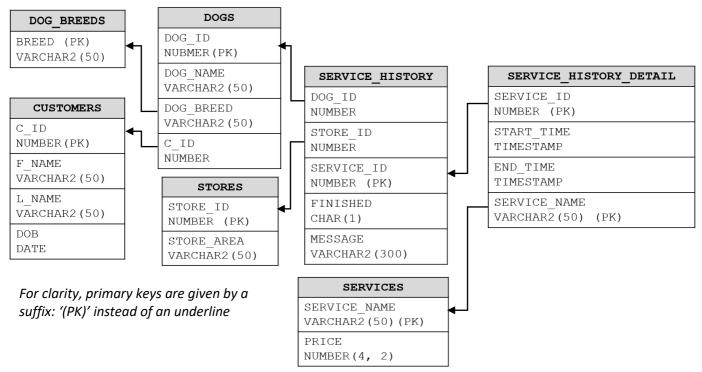
THE DATABASE SCHEMA

DOG_BREEDS

BREED (PK) VARCHAR2(50)

DOGS

DOG_ID NUBMER(PK)
DOG_NAME VARCHAR2(50)
DOG_BREED VARCHAR2(50)
C_ID NUMBER

CUSTOMERS

C_ID NUMBER(PK)
F_NAME VARCHAR2(50)
L_NAME VARCHAR2(50)
DOB DATE

STORES

STORE_ID NUMBER (PK)
STORE_AREA VARCHAR2(50)

SERVICE_HISTORY

DOG_ID NUMBER
STORE_ID NUMBER
SERVICE_ID NUMBER (PK)
FINISHED CHAR(1)
MESSAGE VARCHAR2(300)

SERVICE_HISTORY_DETAIL

SERVICE_ID NUMBER (PK)
START_TIME TIMESTAMP
END_TIME TIMESTAMP
SERVICE_NAME VARCHAR2(50) (PK)

SERVICES

SERVICE_NAME VARCHAR2(50)(PK)
PRICE NUMBER(4, 2)

For clarity, primary keys are given by a suffix: '(PK)' instead of an underline

TABLE DESCRIPTION

DOG_BREEDS

COLUMN_NAME	TYPE	DESCRIPTION
BREED	VARCHAR2(50)	The name of each Dog breed

CUSTOMERS

COLUMN_NAME	TYPE	DESCRIPTION
C_ID	NUMBER (PK)	The Customer ID for a specific customer.
F_NAME	VARCHAR2(50)	The first name of the customer.
L_NAME	VARCHAR2(50)	The last name of the customer.
DOB	DATE	The date of birth for a customer.

DOGS

COLUMN_NAME	TYPE	DESCRIPTION
DOG_ID	NUMBER(PK)	The Dog ID for a specific dog.
DOG_NAME	VARCHAR2(50)	The dog's name.
DOG_BREED	VARCHAR2(50)	The dog's breed.
C_ID	NUMBER	The Customer ID for a specific customer.

STORES

COLUMN_NAME	TYPE	DESCRIPTION
STORE_ID	NUMBER(PK)	The Store ID for a specific *Doggo Paradise* Store.
STORE_AREA	VARCHAR2(50)	The store area (e.g., Carindale)

SERVICE_HISTORY

COLUMN_NAME	TYPE	DESCRIPTION
DOG_ID	NUMBER	The Dog ID for a specific dog.
STORE_ID	NUMBER	The Store ID for a specific *Doggo Paradise* Store.
SERVICE_ID	NUMBER(PK)	The Service ID for a particular service.
FINISHED	CHAR(1)	Boolean flag to determine if the service has been finished.
MESSAGE	VARCHAR2(300)	Message generated on text message enquiry from a customer.

SERVICE_HISTORY_DETAIL

COLUMN_NAME	TYPE	DESCRIPTION
SERVICE_ID	NUMBER(PK)	Highlights the Service ID for a particular service.
START_TIME	TIMESTAMP	Highlights the start time of the specific service.
END_TIME	TIMESTAMP	Highlights the end time of the specific service.
SERVICE_NAME	VARCHAR2(50) (PK)	States the name of the service conducted.

SERVICES

COLUMN_NAME	TYPE	DESCRIPTION
SERVICE_NAME	VARCHAR2(50) (PK)	States the name of the service conducted.
PRICE	NUMBER (4, 2)	States the dollar value of each individual service.

Note that all the attributes and their types can be found by *describing* the different table. For example, `DESC CUSTOMERS;`

DATABASE CONSTRAINTS

The following table lists the database constraints that are applied to our database.

PRIMARY KEYS

CONSTRAINT NAME - PK_DOG_BREEDS		No. 1
Table.Column	DOG_BREEDS.BREED	
Description	BREED is the primary key of DOG_BREEDS	

CONSTRAINT NAME – PK_DOGS		No. 2
Table.Column	DOGS.DOG_ID	
Description	DOG_ID is the primary key of DOGS	

CONSTRAINT NAME - PK_CUSTOMERS		No. 3
Table.Column	CUSTOMERS.C_ID	
Description	C_ID is the primary key of CUSTOMERS	

CONSTRAINT NAME – PK_STORES		No. 4
Table.Column	STORES.STORE_ID	
Description	STORE_ID is the primary key of STORES	

CONSTRAINT NAME - PK_SERVICE_HISTORY		No. 6
Table.Column	SERVICE_HISTORY.SERVICE_ID	
Description	SERVICE_ID is the primary key of SERVICE_HISTORY	

CONSTRAINT NAME - PK_SHD		No. 7
Table.Column	(SERVICE_HISTORY_DETAIL.SERVICE_ID, SERVICE_HISTORY_DETAIL.SERVICE_NAME)	
Description	SERVICE_ID is the primary key of SERVICE_HISTORY_DETAIL	

CONSTRAINT NAME - PK_SERVICES		No. 8
Table.Column	SERVICES.SERVICE_NAME	
Description	SERVICE_NAME is the primary key of SERVICES	

FOREIGN KEYS

CONSTRAINT NAME – FK_DOG_BREED		No. 1
Table.Column	DOGS.DOG_BREED and DOG_BREEDS.BREED	
Description	DOGS.DOG_BREED refers to DOG_BREEDS.BREED	

CONSTRAINT NAME – FK_C_ID		No. 2
Table.Column	DOGS.C_ID and CUSTOMERS.C_ID	
Description	DOGS.C_ID refers to CUSTOMERS.C_ID	

CONSTRAINT NAME – FK_SH_DOG_ID		No. 3
Table.Column	SERVICE_HISTORY.DOG_ID and DOGS.DOG_ID	
Description	SERVICE_HISTORY.DOG_ID refers to DOGS.DOG_ID	

CONSTRAINT NAME – FK_SH_STORE_ID		No. 4
Table.Column	SERVICE_HISTORY.STORE_ID and STORES.STORE_ID	
Description	SERVICE_HISTORY.STORE_ID refers to STORES.STORE_ID	

CONSTRAINT NAME – FK_SHD_SERVICE_ID		No. 5
Table.Column	SERVICE_HISTORY_DETAIL.SERVICE_ID and SERVICES_HISTORY.SERVICE_ID	
Description	SERVICE_HISTORY_DETAIL.SERVICE_ID refers to SERVICES_HISTORY.SERVICE_ID	

CONSTRAINT NAME – FK_SERVICE_NAME		No. 6
Table.Column	SERVICE_HISTORY_DETAIL.SERVICE_NAME and SERVICES.SERVICE_NAME	
Description	SERVICE_HISTORY_DETAIL.SERVICE_NAME refers to SERVICES.SERVICE_NAME	

ADDITIONAL CONSTRAINTS

CONSTRAINT NAME – NN_DOG_NAME		No. 1
Table.Column	DOGS.DOG_NAME	
Description	DOGS.DOG_NAME is not null	

CONSTRAINT NAME – NN_DOB		No. 2
Table.Column	CUSTOMERS.DOB	
Description	CUSTOMERS.DOB is not null	

CONSTRAINT NAME – NN_PRICE		No. 3
Table.Column	SERVICES.PRICE	
Description	SERVICES.PRICE is not null	

CONSTRAINT NAME – CK_FINISHED		No. 4
Table.Column	SERVICE_HISTORY.FINISHED	
Description	Check: SERVICE_HISTORY. FINISHED must be equal to 'T' or 'F'.	

CONSTRAINT NAME – CK_DOB		No. 5
Table.Column	CUSTOMERS.DOB	
Description	Check: Customers were born in the year 1998 or earlier.	

CONSTRAINT NAME – CK_START_TIME_END_TIME		No. 6
Table.Column	SERVICE_HISTORY_DETAIL.START_TIME and SERVICE_HISTORY_DETAIL.END_TIME	
Description	Check: START_TIME is not greater than or equal to END_TIME	

CONSTRAINT NAME – CK_SERVICE_DATE		No. 7
Table.Column	SERVICE_HISTORY_DETAIL.END_TIME	
Description	Check: SERVICE_HISTORY_DETAIL .END_TIME is not greater than or equal than dates in the year 2018	

SECTION 2: ASSIGNMENT TASKS

Note: For any tasks that require reporting execution times, the results should be presented using screenshots.

TASK 0: DATABASE

a) You need to execute the `seed.sql` file and populate your database before completing the following tasks. However, the command for executing `seed.sql` <u>must not</u> be included in your final answer script.

b) Execute a commit to ensure the data is saved to the database.

TASK 1: CONSTRAINTS

a) You might notice that not all the constraints listed in Section 1 (Database Constraints) were created. Write a single SQL SELECT command to find all the constraints that are currently implemented in the database.

b) Implement the **<u>missing</u>** constraints that are defined in Section 1.

TASK 2: TRIGGERS

a) Create a sequence named **SEQ_CUSTOMER** and a trigger named **TR_CUSTOMER_ID** that will automatically generate a sequence of numbers for the C_ID column. This sequence should start at 10,000 and increment by 1.

b) Create a sequence named **SEQ_SERVICE_HISTORY** and a trigger named **TR_SERVICE_ID** that will automatically generate a sequence of numbers for the SERVICE_ID column. This sequence should start at 125,000 and increment by 1.

c) A customer can send a SMS to *Doggo Paradise* to inquire if their dog is ready to be picked up. Create a trigger **TR_SERVICE_HISTORY_MESSAGE**, which is activated at inserting or updating in **SERVICE_HISTORY**. Your trigger should auto-populate the description column according to the strict template below. *(Note: use "BEFORE" trigger)*:

If the FINISHED flag is 'T' then:

```
Hi <Full Customer Name>, your dog <dog-name> of breed: <breed-name> is ready for pick-up at <store-area>.
```

If the FINISHED flag is 'F' then:

```
Hi <Full Customer Name>, your dog <dog-name> of breed: <breed-name> is not ready to be picked up yet.
```

An example of an inserted SERVICE_DESCRIPTION:

```
Hi John Smith, your dog Sam of breed: Jack Russel is ready for pick-up at Doggo Paradise Carindale.
```

d) Run the following SQL statements to test the new triggers.

```
INSERT INTO CUSTOMERS (F_NAME, L_NAME, DOB)
VALUES ('Luke', 'Cheung', '08-OCT-1996');

INSERT INTO SERVICE_HISTORY (DOG_ID, STORE_ID, FINISHED)
VALUES (1234, 30, 'F');
```

TASK 3: VIEWS

a) CREATE a view V_DOG_BREED_STATISTICS that contains the **total**, **mean** and **standard deviation** of money spent at *Doggo Paradise* per Dog Breed. Hence, there should be four columns in that view, including the Dog Breed.

b) CREATE a materialized view MV_DOG_BREED_STATISTICS that contains the same information required by the virtualized view created in 3a.

c) Run the following two queries and record the execution time of each:

```
SELECT * FROM V_DOG_BREED_STATISTICS;

SELECT * FROM MV_DOG_BREED_STATISTICS;
```

In the report, explain the reason for any differences in the execution times between the two queries.

4: FUNCTION BASED INDEXES

a) Construct a query to find the longest time to perform a **Dental Checkup** on a dog, together with information about the dog that received that checkup (i.e., DOG_ID and DOG_NAME) as well as the store where that checkup took place.

b) Create a function based index **IDX_SERVICE_TIME** to decrease the time for running the query in 4a.

c) Rerun the query from 4a and record the time that it takes for the query to run. If there is any difference, explain how the index has affected the running time.

TASK 5: BITMAP INDEXING

a) Calculate the query execution time to find the total number of each service performed.

b) Create a bitmap index **BIDX_SERVICE** that would potentially decrease the time that it will take for the query in 5a to run.

c) Rerun the query from 5a and record the time that it takes for the query to run. Explain how the index has affected the running time.

d) What are the advantages and disadvantages constructing this bitmap index, justify your answer.

TASK 6: EXECUTION PLAN & ANALYSIS

a) You need to obtain your unique dog ID (DOG_ID) before completing the rest of Task 6. To do this, run the following query in SQL*Plus, after substituting your 8-digit student number for the placeholder. **Record the dog ID that you obtained at the top of your** `answer.sql` **file.**

Template	Example (If student number was 12345678)
`SELECT GET_UNIQUE_SNUMBER(<StudentNumber>) FROM DUAL;`	`SELECT GET_UNIQUE_SNUMBER(12345678) FROM DUAL;`

b) Write a single SQL SELECT command to find all B+ tree indexes that are currently implemented in the database for the **STORES**, **SERVICE_HISTORY** and **SERVICE_HISTORY_DETAIL** tables. For each index, retrieve the index name, column name and table name. Verify that there is a **PK_STORES** index, two **PK_SHD** and a **PK_SERVICE_HISTORY** indexes before proceeding to Task 6c**.**

c) Create a query that finds the total number of <u>all</u> services performed for a specific dog. The dog in question is given by your personal dog ID (DOG_ID). Each individual service preformed is contained in **SERVICE_HISTORY_DETAIL** table. Record the execution time and explain the query processing steps taking place in that plan.

d) Write SQL statements to drop the following keys:
   - The foreign key, **FK_SHD_SERVICE_ID** from **SERVICE_HISTORY_DETAIL**.
   - The primary key, **PK_SHD** from **SERVICE_HISTORY_DETAIL**.
   - The primary key, **PK_SERVICE_HISTORY** from **SERVICE_HISTORY**.

   Run those statements and save it to your `student_number.sql` file. Verify that both **PK_SHD** indexes and **PK_SERVICE_HISTORY** has been deleted and rerun the query from Task 6c. Explain any difference between this plan and the one in 6c. Record the execution time and explain the query processing steps taking place in that plan.

e) Analyze the index **PK_STORES** and use the values from the INDEX_STATS table to answer following questions about the B+ Tree Index:

   - What is the height of the tree?

   - How many leaf blocks are in the tree?

   - How many block accesses are needed for a full direct scan of the STORES table?

   The **USER_TABLES** view may be useful for answering this question.

   You may also find this Oracle Documentation useful for this question:
   https://docs.oracle.com/cd/B28359_01/server.111/b28320/statviews_5119.htm#REFRN29025

SECTION 3: TIPS AND HELP

- To execute a commit, use the following command. This ensures that any changes to the database have been saved.

```
commit;
```

- If you wish to change computers you can save your database to a file with the following command. Replace `C##s1234567` with your username and `mypassword` with your password.:

```
HOST EXP C##s1234567/mypassword file=H:\assignment.dmp
owner=C##s1234567;
```

  And restored to another computer using the following command again replacing `C##s1234567` with your username and password:

```
HOST IMP C##s1234567/mypassword@localhost file=H:\assignment.dmp
full=y
```

  Note that the username and password are for the database user on the computer you are currently using.

- If the terminal width for SQL*Plus is too small, and it is causing text to wrap, it can be increased by using the following command:

```
SET LINESIZE 200;
```

  Where 200 is the number of characters wide the terminal is to be set to.
- If columns are too wide, you can use (where COLUMN_NAME is the name of the column)

```
COLUMN COLUMN_NAME FORMAT A30;
```

- This assignment will be tested on Oracle Database 12c Enterprise Edition with the SQL*Plus 12.1 client.

SECTION 4: MARKING SCHEME

TASK NUMBER	TASK PART	MARKS
1	A	1
	B	7
2	A	4
	B	3
	C	10
	D	2
3	A	7
	B	4
	C	7
4	A	8
	B	1
	C	5
5	A	2
	B	1
	C	5
	D	5
6	B	5
	C	3
	D	8
	E	2
Presentation	-	10
Total Marks	-	100

Presentation marks will be allocated based on the formatting of your `student_number.sql` file and the final report you submit. Elements such as a title page, commenting and overall document structure will all be considered when marking this criterion. Factors that significantly increase a tutors marking time, will cause you to lose marks, for example, not following the template `student_number.sql` script provided to formulate your answer.

6A is worth 0 marks because the set up establishes your answers for 6B, 6C and 6D. Hashes will compute the same number across all platforms and installations of Oracle 12c EE.

SECTION 5: SUBMISSION DELIVERABLES

The project is due at **11:59pm on the 26th of October 2017.**

You are required to turn in two files (use your 8-digit student ID to name your files):

1. `student_number.pdf`: A report answering all the questions in Section 2 including all the necessary SQL statements and their outputs (please be sensible when including query output, any output close to the size of a page can be shown by just including the first 10 lines and the last 10 lines – reports including excessive pages of query output lose presentation marks). **student_number needs to be replaced with your student number just before submitting to Turnitin. E.g.** `12345678.pdf`

2. `student_number.sql`: A script file that includes all your SQL statements. **student_number needs to be replaced with your student number just before submitting to Blackboard. E.g.** `12345678.sql`

Your **report** file should include the following content:

- Answer to all the questions in Section 2
- If you are asked to write SQL statements, you need to include those statements in your report
- When you execute a SQL statement, if Oracle products any output (e.g. query results, query execution time, query plan, etc), you need to include the output as well. Execution times are required to be presented using screenshots.

Your **script** file is in plain text format. You must make sure that your script file can be executed on the ITEE lab computers using the "@" command. The same SQL statements in your script file should be also be copied and pasted into your report file (as explained above). Even though the script file does not introduce any new information in comparison to the report file, it is intended to help the lecturer/tutors quickly check the correctness of your SQL statements before checking the details in your report file.

Do **not** add the create table and insert statements from the provided script file into your submission. Doing so will result in a score of 0 for presentation marks, and may affect the results of your queries as well.

GOOD LUCK WITH YOUR PROJECT!

SECTION 6: VERSION HISTORY

VERSION 1.3

- Added note to Task 2c on trigger activation time.
- Question 3 on views is updated to include only 3 tasks instead of 4
- Refined the wording of Tasks 5b and 5c.
- Modified 6c to count the individual services performed for a given dog instead of visits.
- Altered 6d to drop constraints in **SERVICE_HISTORY_DETAIL** and **SERVICE_HISTORY**
- Fixed minor typos in the Database Constraints
- Updated mark distribution.
- Updated `student_number.sql`

VERSION 1.2

- Fixed minor typos in Database Constraints
    - Foreign Key No.1 Description: **DOG.DOG_BREEDS → DOG.DOG_BREED**
    - Additional Constraints No.5: **CUSTOMER.DOB → CUSTOMERS.DOB**
    - Foreign Key No.6 Typo: **SERVICES_HISTORY_DETAIL → SERVICE_HISTORY_DETAIL**
- Clarified Additional Constraint No.7
- New table of description for the table **SERVICES**.

VERSION 1.1

- Fixed typo in Question 2c. The trigger is to be built on **SERVICE_HISTORY** and not **SERVICE_HISTORY_DETAIL**.
- Fixed error in Question 6d. No longer asking to verify that **PK_STORES** index has been deleted.

VERSION 1.0

Initial release of specification