# BUS5DWR: Assignment 1 (40%)

Answers to this assignment may be provided in a copy of this Word document under the question or in the space provided.

**Part 1 [7 marks]**

A fictitious train company has the following data extract (imagine there are thousands of rows similar to these).

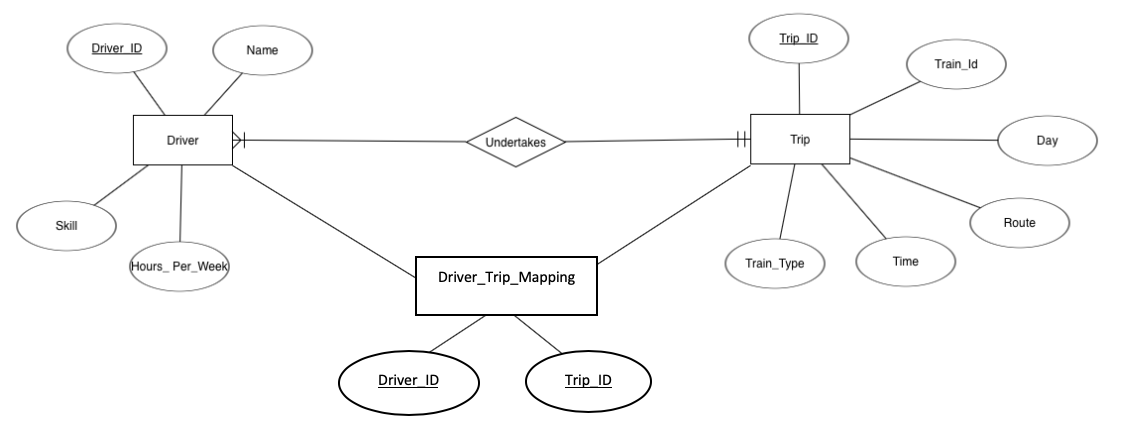


Using the ERDPlus modelling tool (<https://erdplus.com/#/standalone> **—** no need to register), create an ER diagram and relational schema diagram of a structured database design in second normal form consisting of two or more named tables capturing the above data. [1+1=2 marks]

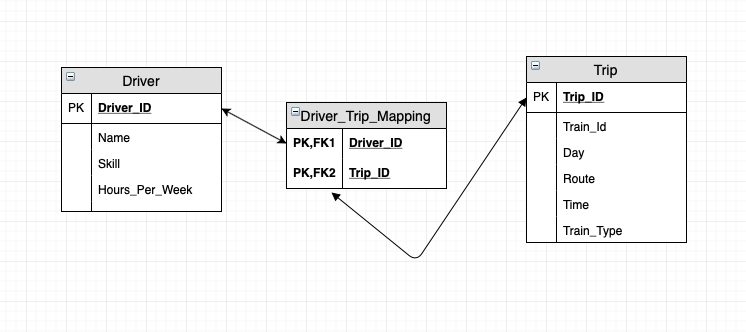
* In the ER diagram indicate the cardinality of each relationship (one-one, one-many etc). [1 mark]
* In the relational schema diagram show the attribute names and primary key(s) of each table. [1 mark]

Include screen captures/image exports of the diagrams in your submission.

**ENTITY RELATIONSHIP DIAGRAM :-**



**RELATIONAL SCHEMA DIAGRAM:-**

****

Note the design needs to enable the following types of queries easily:

1. How many trains are simultaneously running at 2pm on a weekday?

ANSWER) It can be inferred from the design that only **one** train is running at 2pm on a weekday i.e. the train running from Flinders st to Frankston

|  |  |  |  |
| --- | --- | --- | --- |
| Trip\_id | Train\_id | Route | Train\_type |
| 001 | 1001 | Flinders St-Frankston | Comeng |

1. How many trains depart from a particular station?

ANSWER) It can be inferred that in totality there are **four** trains that are departing from a particular station.

|  |  |  |
| --- | --- | --- |
| Trip\_id | Train\_id | Departing Station |
| 001 | 1001 | Flinders St |
| 002 | 1001 | Frankston |
| 003 | 1002 | Flinders St |
| 004 | 1002 | Pakenham |

From the table, it can be concluded that there are :

2 Trains are departing from Flinders St

1 Train is departing from Frankston

1 Train is departing from Pakenham

Justify your design choice and explain why it is in second normal form. [2+1=3 marks]

ANSWER) In the above shown design, the original table is already in 1NF as the column values are atomic but the original table given to us is not in the 2NF due to the presence of partial dependencies as there are some columns( Route, Day, Driver Name, Skills etc) that are functionally dependent on any one part of the composite key( Trip id and Driver id) and also there are insert, update and delete anomalies in the table.

The data has been normalized in order to reduce data redundancy, retain data integrity and to avoid the insert, update and delete anomalies. In the new implemented model, two tables are created( Driver and Trip) with an additional associative table (Driver\_Trip\_Mapping). Driver and Trip tables have their primary keys as driver\_id and trip\_id respectively and all the non-key attributes are dependent upon the primary keys in their respective table as shown below:

Driver\_ID 🡪Driver Name, Skill, Hours Per Week

Trip\_ID 🡪 Train\_id,Route,Day,Time,Train\_type

And, now there are no partial dependencies in the table.

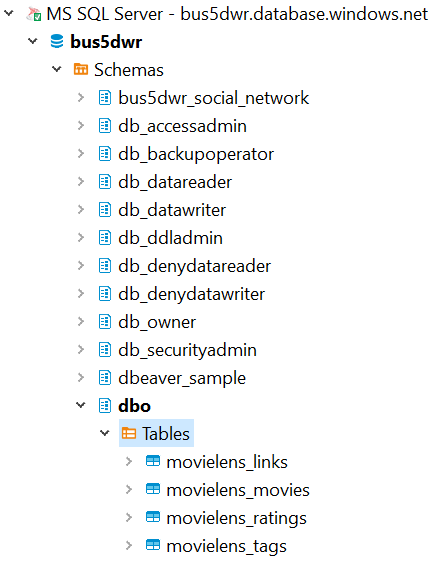
Moreover, as the original table has partial dependencies without full dependencies( when non-key attributes are fully dependent on the composite key),an associative table has also been made to link the two tables by referencing the primary keys of each table. The junction or associative table contains two foreign keys( Driver\_ID and Trip\_ID) that are in a many to one relationship.

Furthermore, the anomalies have also been removed as now we can add a Driver without assigning a trip to him, similarly can delete a record like trip id without deleting the driver’s record and also can update our tables without redundancy.

The table is in 2NF as now in the Driver and Trip tables, all the non-key columns are dependent upon the primary key of the table (therefore no partial dependencies) and there are only keys in the Driver\_Trip\_Mapping table. Also, there are transitive dependencies i.e. some non-key attributes are dependent upon other non-key attributes in the data for example time depends upon route and route depends upon trip\_id; therefore, it needs to be converted into third normal form.

**Part 2 [33 marks in 12 parts]**

We will make use of the MovieLens dataset contained in bus5dwr.dbo within the bus5dwr.database.windows.net server. Please refer to Lecture 1 slides for instructions on how to access the server via DBeaver if you have not accessed it already. The data involves movie ratings and tags and is described in more detail here: <http://files.grouplens.org/datasets/movielens/ml-latest-small-README.html>.



You will see four tables:

    i) bus5dwr.dbo.movielens\_links

    ii) bus5dwr.dbo.movielens\_movies

    iii) bus5dwr.dbo.movielens\_ratings

    iv) bus5dwr.dbo.movielens\_tags

Start by exploring the contents of the tables understanding the meaning of their attributes.

Write SQL queries (using MS SQL Server) that use one or more of the tables to answer the following questions. Please include your code as text rather than an image to enable it to be copy-pasted for assessment.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Question | Answer | Code | Marks |
| 1 | How many users contributed tags? |  | **select** **count**(**DISTINCT** userid) **as** num\_of\_users\_contributing\_tags  **from** dbo.movielens\_tags **where**  tag **IS** **NOT** **NULL**; | 2 |
| 2 | How many movies had exactly 3 genres associated with them? (Comedy|Drama|Romance would be one example.) |  | **with** c **as** (  **select** genres,(LEN(genres)-LEN(**REPLACE**(genres,'|',''))+1) **As** no\_of\_genres **from** dbo.movielens\_movies  )  **select** **count**(no\_of\_genres) **as** no\_movies\_with\_3\_genres  **from** c **where** no\_of\_genres=3;  OR  **select** **count**(\*)  **from** movielens\_movies  **where** genres **like** '%|%|%'  **and** genres **not** **like** '%|%|%|%' | 2 |
| 3 | How many users gave ratings that are not whole numbers (e.g. 3.5, 4.5)? |  | **select** **count**( **DISTINCT** userid) **from**  dbo.movielens\_ratings **where**  rating **not** **like** '\_' **and** rating **not** **like** '\_.0'; | 2 |
| 4 | List the movies whose title (and year) appear more than once in the movielens\_movies table. |  | **select** title **as** repeated\_titles  **from** dbo.movielens\_movies  **group** **by** title  **having** **count**(title)>1; | 2 |
| 5 | How many movies were released between 1985 and 1995 inclusive? |  | **with** y **as** (  **select** **SUBSTRING**(title,len(title)-4,4) **as** year2  **from** dbo.movielens\_movies  **where** title **like** '%(\_\_\_\_)'  )  **select** **count**(year2) **as** num\_movies\_within\_1985\_1995  **from** y **where**  year2 **between** 1985 **and** 1995;  OR  **select** **count**(\*)  **from** movielens\_movies  **where** title **like** '%(198[5-9])%'  **or** title **like** '%(199[0-5])%' |  |
| 6 | Which movies in the movielens\_movies table did not receive a rating? Write a query to list five of these movies. |  | **select** **top** 5 mov.title as title  **from** movielens\_movies mov  **left** **join** movielens\_ratings rat  **on** mov.movieId = rat.movieId  **where** rating **is** **NULL** | 2 |
| 7 | How many users tagged three or more different movies? |  | **with** d **as** (  **select** **count**(**DISTINCT** movieid) **As** no\_of\_movies ,userId **from** dbo.movielens\_tags  **group** **by** userId **having** ((**count**( **distinct** movieid)>=3))  )  **select** **count**(userId) **as** users\_tagging\_three\_or\_more\_movies  **from** d **where** no\_of\_movies>=3; | 3 |
| 8 | What are the names of the five movies that had the most ratings? Show the title and number of ratings in two columns. | Title Num\_ratings  … … | **select** **top** 5 b.title **as** Title,**count**(a.rating) **as** Num\_Ratings  **from** dbo.movielens\_ratings a  **join** dbo.movielens\_movies b  **on** a.movieId=b.movieId  **group** **by** Title  **order** **by** **count**(\*) **DESC**; | 3 |
| 9 | What are the five tags that had the most distinct users contributing them? Show the tag and number of users in two columns. | Tag Num\_users  … … | **SELECT** **top** 5 tag **as** TAG,**count**(**DISTINCT** userId) **as** No\_OF\_Users  **FROM** dbo.movielens\_tags  **group** **by** TAG **order** **by** No\_OF\_Users **DESC**; | 3 |
| 10 | What are the three highest rated movies by average rating where one of the genres is ‘Mystery’ and  the number of ratings for the movie is at least five? Show the movie name and average rating in two columns. |  | **select** **top** 3 a.title **as** Title,  **CAST**(**AVG**(b.rating) **as** **DECIMAL**(10,2)) **as** Average\_Rating  **from** dbo.movielens\_movies a  **join** dbo.movielens\_ratings b  **on** a.movieId=b.movieId  **where** a.genres **like** '%Mystery%'  **group** **by** Title **having** **count**(b.rating)>= 5  **order** **by** Average\_Rating **DESC**; | 4 |
| 11 | Find the five genres with the most distinct tags. Show the genre and tag count in two columns. For example, a movie having two genres Action|Thriller will have its tags contribute to each of the Action and Thriller genres.  Hint: see example B in <https://docs.microsoft.com/en-us/sql/t-sql/functions/string-split-transact-sql?view=sql-server-2017> to see how to split the genres field. | Genre Tag\_count  … … | **select** **top** 5 value **as** Genres,**count**(**distinct** tag) **as** Tag\_count  **from** dbo.movielens\_movies a  **JOIN** dbo.movielens\_tags b  **ON** a.movieId=b.movieId  **cross** apply string\_split(genres,'|')  **group** **by** value  **order** **by** Tag\_count **DESC**; | 4 |
| 12 | Let us attempt to find the highest rated movies, but not simply by computing the average rating, since that would favour movies with a small number of ratings.  We will make use of the following formula.  Weighted rating of a movie = (V\*R+10\*C)/(V+10), where   * V = number of votes for the movie * R = average (mean) rating for the movie * C = the average rating across all movies   This formula represents the equivalent of adding 10 additional ratings to each movie, each equal to the average rating across all movies, then finding the resulting average. Hence a movie with a single rating will not have too high a weighted rating.  Find the names and ratings of the five highest rated movies according to the formula above having the tag ‘funny’. | Title Weighted\_rating  … … | **with** x **as** (  **select** b.title **as** Title, **count**(rating) **as** V,  **CAST**(**avg**(rating) **as** **DECIMAL**(10,2)) **as** R,  (**select** **Cast**(**avg**(rating) **as** **DECIMAL**(10,2))  **from** dbo.movielens\_ratings) **as** C  **from** dbo.movielens\_ratings a  **join** dbo.movielens\_movies b  **on** a.movieId=b.movieId **join**  dbo.movielens\_tags z **on**  a.movieId=z.movieId  **where** tag='funny' **group** **by** b.title  ) **select** **top** 5 title **as** Title,  **CAST**((V\*R+10\*C)/(V+10) **as** **DECIMAL**(10,2)) **as** Weighted\_Average  **from** x  **order** **by** Weighted\_Average **DESC**; | 4 |

Total: 40 marks

## Marking rubrics

The following marking guide will be used by the marker in assessing your work. Please have a look to understand what you need to cover for each question in this assignment.

**Part 1:**

For each part where marks are indicated, full marks will be given for a correct answer, 0.5 marks deducted for each minor error up to the value of that part.

**Part 2:**

Full marks for each faultless SQL statement (i.e. correct output). For each error picked up by the marker, 0.5 marks will be deducted for minor errors and 1 mark for critical errors up to the value of that part.