

**Faculty Computer Science and Information Technology**

**Semester 1, 2018/2019**

**WIA 2001**

**DATABASE**

**Assignment**

**MASTERPLAN HELPDESK SYSTEM**

Lab: Monday MM6 (G7)

|  |  |
| --- | --- |
| **Name** | **Registration Number** |
| Cheong Kah Seng | WIC 170020 |
| Aida Syafiqah Bt Mohd Azli | WIF 170012 |
| Emira Asyikin Bt Abdul Rahim | WIF 170038 |
| Goh Sin Ni | WIF 170051 |
| Liow Jing Wen | WIF 170071 |

Contents

[Contents 2](#_Toc532216269)

[1.0 Introduction 3](#_Toc532216270)

[2.0 Business Rules 5](#_Toc532216271)

[3.0 System Objectives and Scope 6](#_Toc532216272)

[4.0 User Requirements 7](#_Toc532216273)

[5.0 Entities, Attributes, Relationship, Cardinality and Constraints 8](#_Toc532216274)

[5.1.1 COMPANY 8](#_Toc532216275)

[5.1.2 REQUEST 9](#_Toc532216276)

[5.1.3 TEAM\_USER 10](#_Toc532216277)

[5.1.4 SERIAL\_NUMBER 11](#_Toc532216278)

[5.1.5 REQUEST\_FEEDBACK 12](#_Toc532216279)

[6.0 Entity Relationship (ER) Diagram 14](#_Toc532216280)

[7.0 Normalization Process 15](#_Toc532216281)

[8.0 Implementable ER Diagram 19](#_Toc532216282)

[9.0 Conclusion 20](#_Toc532216283)

[10.0 SQL Statement 21](#_Toc532216284)

[11.0 Screenshots of Tables 35](#_Toc532216285)

[12.0 Difficulties and Challenge Faced During the Database Development 44](#_Toc532216286)

[13.0 Lesson learned 45](#_Toc532216287)

[Appendix 46](#_Toc532216288)

1.0 Introduction

1.1 Business Operation

Masterplan Consulting Sdn Bhd began its operations back in 1995 with a portfolio of providing top-notch defence and security solutions to government and government related bodies. Masterplan had a good reputation as the years grew and they have since gone on to cement themselves as a pioneering ICT solutions provider with a who’s who list of respectable clienteles within the nation.

All Masterplan support team has been requested to follow the SOP (Standard Operation Procedure) in their work. For instance, they have to ask every company before proceeding to solve the company’s problem. Every support teams in Masterplan have to follow a defined process flow which will be explain further in 1.2 Description of the Process Flow.

The support teams are working 24/7 in shifting manner on call support. They handle all the calls from different agency from different region. There is total 3 shifts in total (8 hours per shift), 6 staffs per shift. They use only one centralized call centre to support nationwide. Call centre is the first level support for all the customers before delegated to the second level support(engineers). The main reasons of doing this is to make sure the respond time to the customers within 2 hours from the case log and make sure all the cases are resolved. Masterplan stated that customer satisfaction is always the top priority.

1.2 Description of the Process Flow

The basic process flow of the company is shown as Diagram 1.1.

Diagram 1.1 Basic process flow

Diagram above shows the basic process flow of the business operation. Incoming call is the first business operation. It is a centralize centre to receive a complaint or reports from their existing customers.

Step of identifying project have to be done soon after identifying company’s information. Due to current Masterplan support operation consist only one call centre, the entire call from the multiple project will call to the same support team. In order for the support team to assign a right resource to a right project, he/she need to be identified a particular project once call is received. The advantages of doing this step is the miscommunication among all project teams can be reduces. A proper report can also be generated according to the project. By following the proper way of assigning project to the right project team, Key Performance Index (KPI) of each project team can also be measured easily.

The following step is to identify issue type. Masterplan Consulting Sdn Bhd is an ICT solution provider, all the projects they currently handling in a multiple platform which means that all the solution for each project is different from other projects from perspective of hardware, software, application and overall solution design. System is able to identify the cases by entering the serial number (provided that all serial number have been pre-registered in the system). By pre-identifying issue type, more precise information will be provided.

Finally, the final step of the business operation is to refer to engineer. Once the case submitted, a list of support engineer will be displayed in which based on the request type. The support team will choose a right engineer which is available to follow up the case. The respective assigned engineer must make sure the assigned request is closed (solved). This is also a way to measure KPI for each engineer and their efficiency to resolve a problem.

2.0 Business Rules

The business rules of the company’s helpdesk system are:

1. Company owns many SERIAL\_NUMBER.
2. COMPANY asks for many REQUESTs.
3. Request is asked by one COMPANY.
4. ReQUEST belongs to one SERIAL\_NUMBER
5. REQUEST is created by one TEAM\_USER.
6. REQUEST is assigned to one TEAM\_USER.
7. REQUEST has many REQUEST\_FEEDBACK.
8. TEAM\_User creates many REQUESTs.
9. TEAM\_USER assigns many REQUESTs.
10. TEAM\_USER creates one REQUEST\_FEEDBACK.
11. Request\_Feedback is created by one TEAM\_USER.
12. REQUEST\_FEEDBACK is created for one REQUEST.
13. Serial\_Number belongs to one COMPANY.
14. SERIAL\_NUMBER belongs to many REQUEST

3.0 System Objectives and Scope

3.1 System Objectives

* To maintain and update the request from each company.
* To improve and keep track on the efficiency of support teams.
* To identify inefficient models which has the most occurrence of malfunction.
* To improve the efficiency for solving of a request.

3.2 Scope

1. Helpdesk system of Masterplan Sdn Bhd is to solve problems faced by each client’s company and this helps to strengthen the relationship with clients.
2. Admin of helpdesk system identifies the team which a helpdesk system user belongs to in the helpdesk system such as installation team, maintenance team, repair team, front desk team and others.
3. Front desk team in charge of receiving calls from client’s company for requests.
4. Upon receiving calls, front desk team will create a request and key-in relevant information from the calling company.
5. Front desk team will enter necessary company’s details such as serial number, and model. Front desk will select the type of the request based on the team of user. For example, ‘installation team’ will be selected when caller request for installation.
6. Front desk team will then set priority of the request, whether it’s low, medium or high.
7. Request will also be assigned to corresponding team chief to solve the company’s problem.
8. Assigned user receives request with status “opened”.
9. Assigned user checks for serial number and model from request to better understand the case.
10. Assigned user also checks for company details such as email, office number and address for contacting the concerned company.
11. Assigned user provides feedback throughout the solving of a request to update on the progress.
12. Assigned user can upload relevant files such as picture, video or text files for a particular request when it is necessary.
13. After the request has been resolved, the assigned user closes the case request.
14. If necessary, assigned user is allowed to create another request for subsequent maintenance or upgrading of the request.
15. Users of helpdesk system can generate and check on reports to identify unsolved cases.
16. Administrative team of helpdesk system can generate and check on reports regarding support team’s KPI and inefficient models.

4.0 User Requirements

User requirements are identified and listed as below.

4.1 Maintaining data and records (by User)

* To maintain (enter, update, and delete) data on company.
* To maintain (enter, update, and delete) data on user.
* To maintain (enter, update, and delete) data on serial number.
* To maintain (enter and update) data of request.
* To maintain (enter and update) data of request feedback.

4.2 Searching data and records (by User)

* To perform searches on company details.
* To perform searches on details of user who is assigned to request.
* To perform searches on serial number to get its respective model.

4.3 Tracking data and record (by User)

* To track the status of request.
* To track the detail of company for in-door service.
* To track the detail of serial number.
* To track the files uploaded for request feedback.

4.4 Generating reports (by User)

* To report on the most reported model.
* To report on the most reported brand.
* To report on the number of requests, total time taken and average time taken for requests solved by team(s).
* To report on the number of requests, total time taken and average time taken for requests solved by team users.
* To report on the team users with opened requests.
* To report on the number of requests opened per month.

5.0 Entities, Attributes, Relationship, Cardinality and Constraints

From the business rules, relevant entities and attributes are identified.

5.1 Entity, Attributes, Attribute Types and Constraints

Tables below show the attributes for each entity along with the attribute type and constraint applied.

5.1.1 COMPANY

Table 5.1 shows details of COMPANY entity.

Table 5.1 Details of COMPANY Entity

|  |  |  |
| --- | --- | --- |
| **ENTITIY NAME:** | COMPANY | |
| **DESCRIPTION:** | Company which reports for problem | |
| **ATTRIBUTES** | **TYPES** | **CONSTRAINTS** |
| COMPANYID | INT | Primary Key  Not Null  Auto Incremental |
| COMPANYNAME | VARCHAR | Not Null |
| HPNO | VARCHAR | Not Null |
| ADDRESS | VARCHAR | Not Null |
| POSTCODE | INT | Not Null |
| STATE | VARCHAR | Not Null |
| COUNTRY | VARCHAR | Not Null |
| CONTACTPERSON | VARCHAR | Not Null |
| OFFICENO | VARCHAR | Not Null |
| FAXNO | VARCHAR |  |
| EMAIL | VARCHAR | Not Null |
| STATUS | VARCHAR | Not Null |

5.1.2 REQUEST

Table 5.2 shows details of REQUEST entity.

Table 5.2 Details of REQUEST Entity

|  |  |  |
| --- | --- | --- |
| **ENTITIY NAME:** | REQUEST | |
| **DESCRIPTION:** | All data recorded from company’s request for project case | |
| **ATTRIBUTES** | **TYPES** | **CONSTRAINTS** |
| REQUESTID | INT | Primary Key  Not Null  Auto Incremental |
| COMPANYID | INT | Foreign Key  Not Null |
| SERIALNOID | INT | Foreign Key  Not Null |
| DESCRIPTION | VARCHAR |  |
| TYPE | INT | Foreign Key  Not Null |
| PRIORITY | VARCHAR | Not Null |
| ASSIGNEDTO | INT | Foreign Key  Not Null |
| STATUS | VARCHAR | Not Null |
| OPENEDDATE | TIMESTAMP | Not Null |
| CLOSEDDATE | TIMESTAMP |  |

5.1.3 TEAM\_USER

Table 5.3 shows details of TEAM\_USER entity.

Table 5.3 Details of TEAM\_USER Entity

|  |  |  |
| --- | --- | --- |
| **ENTITIY NAME:** | TEAM\_USER | |
| **DESCRIPTION:** | User of helpdesk system | |
| **ATTRIBUTES** | **TYPES** | **CONSTRAINTS** |
| USERID | INT | Primary Key  Not Null  Auto Incremental |
| USERNAME | VARCHAR | Not Null |
| PASSWORD | VARCHAR | Not Null |
| TEAMID | INT | Primary Key  Not Null |
| TEAMNAME | VARCHAR | Not Null |
| DESCRIPTION | VARCHAR |  |
| FULLNAME | VARCHAR | Not Null |
| HPNO | VARCHAR | Not Null |
| OFFICENO | VARCHAR | Not Null |
| FAXNO | VARCHAR |  |
| EMAIL | VARCHAR | Not Null |
| STATUS | VARCHAR | Not Null |

5.1.4 SERIAL\_NUMBER

Table 5.4 shows details of SERIAL\_NUMBER entity.

Table 5.4 Details of SERIAL\_NUMBER Entity

|  |  |  |
| --- | --- | --- |
| **ENTITIY NAME:** | SERIAL\_NUMBER | |
| **DESCRIPTION:** | Serial number for request | |
| **ATTRIBUTES** | **TYPES** | **CONSTRAINTS** |
| SERIALNOID | INT | Primary Key  Not Null  Auto Incremental |
| SERIALNO | VARCHAR | Not Null |
| DESCRIPTION | VARCHAR |  |
| MODELID | INT | Not Null |
| MODELNAME | VARCHAR | Not Null |
| BRAND | VARCHAR | Not Null |
| COMPANYID | INT | Foreign Key  Not Null |
| STATUS | VARCHAR | Not Null |

5.1.5 REQUEST\_FEEDBACK

Table 5.5 shows details of REQUEST\_FEEDBACK entity.

Table 5.5 Details of REQUEST\_FEEDBACK Entity

|  |  |  |
| --- | --- | --- |
| **ENTITIY NAME:** | REQUEST\_FEEDBACK | |
| **DESCRIPTION:** | Feedback for each project’s request | |
| **ATTRIBUTES** | **TYPES** | **CONSTRAINTS** |
| FEEDBACKID | INT | Primary Key  Not Null  Auto Incremental |
| FEEDBACKDES | VARCHAR | Not Null |
| FILEID | INT | Not Null |
| FILENAME | VARCHAR | Not Null |
| FILEPATH | VARCHAR | Not Null |
| FILESIZE | VARCHAR | Not Null |
| REQUESTID | INT | Foreign Key  Not Null |
| UPDATEDBY | INT | Foreign Key  Not Null |
| UPDATEDON | TIMESTAMP | Not Null |

5.2 Entity, Relationship and Cardinality

Table 5.6 shows the relationship between each entity and the cardinality associated accordingly.

Table 5.6 Relationship and Cardinality between Entities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NO** | **ENTITY** | **RELATION-SHIP** | **CARDINALITY** | **ENTITY** |
| 1 | **COMPANY** | Owns | 1:M | SERIAL\_NUMBER |
| Asks for | 1:M | REQUEST |
| 2 | **REQUEST** | Asked by | 1:1 | COMPANY |
| Belongs to | 1:1 | SERIAL\_NUMBER |
| Is created by  Is assigned to | 1:1 | TEAM\_USER |
| Has | 1:M | REQUEST\_FEEDBACK |
| 3 | **TEAM\_USER** | Creates  Assigns | 1:M | REQUEST |
| Creates | 1:1 | REQUEST\_FEEDBACK |
| 4 | **REQUEST\_FEEDBACK** | Is created by | 1:1 | TEAM\_USER |
| Is created for | 1:1 | REQUEST |
| 5 | **SERIAL\_NUMBER** | Belongs to | 1:1 | COMPANY |
| Belongs to | 1:M | REQUEST |

6.0 Entity Relationship (ER) Diagram

In this project, Crow’s Foot Model has been used to show the Entity Relationship (ER) diagram of the company helpdesk, as shown in Diagram 6.1. Crow’s Foot Model shows all the attributes with data types for every entity.

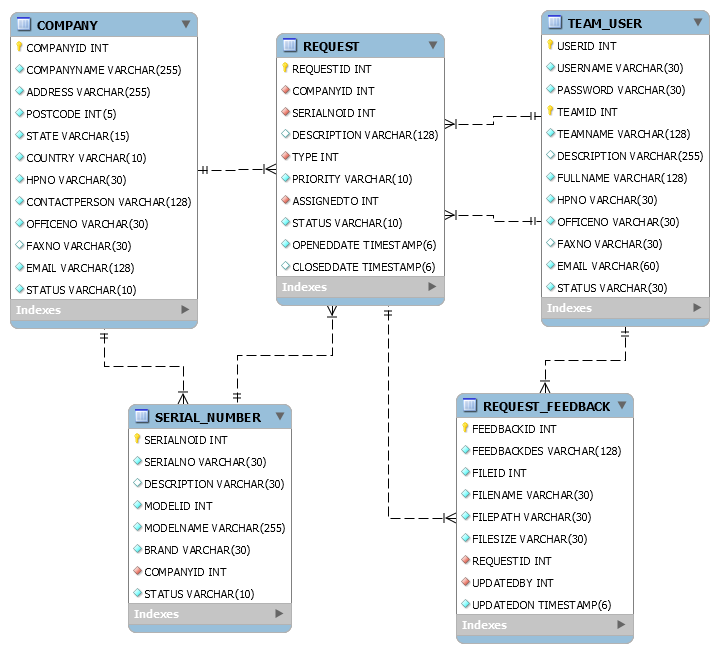


Diagram 6.1 Crow’s Foot Model

7.0 Normalization Process

An ER Diagram is done based on the business rules and user requirements. However, it is found that the ER Diagram in 6.0 ER Diagram can further be normalized into a higher normal form to produce a better database design. Thus, a normalization process is performed on the entities which are found to have transitive dependency such as SERIAL\_NUMBER, TEAM\_USER and REQUEST\_FEEDBACK.

For the SERIAL\_NUMBER entity, transitive dependency can be eliminated by creating another entity of MODEL to store all model-related information. The normalization process for SERIAL\_NUMBER entity can be referred to in Diagram 7.1 Normalisation of SERIAL\_NUMBER Entity.

For the REQUEST\_FEEDBACK entity, transitive dependency can be eliminated by creating another entity of REQUEST\_FILE to store all information of files uploaded per request. The normalization process for REQUEST\_FEEDBACK entity can be referred to in Diagram 7.2 Normalisation of REQUEST\_FEEDBACK Entity.

For the TEAM\_USER entity, transitive dependency can be eliminated by creating another entity of TEAM to store all information in regard to teams. The normalization process for TEAM\_USER entity can be referred to in Diagram 7.3 Normalisation of TEAM\_USER Entity.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UNF:** SERIAL\_NUMBER (SERIALNOID, SERIALNO, DESCRIPTION, MODELID, MODELNAME, BRAND, COMPANYID, STATUS)  **1NF:** SERIAL\_NUMBER (SERIALNOID, SERIALNO, DESCRIPTION, MODELID, MODELNAME, BRAND, COMPANYID, STATUS)  **2NF:** SERIAL\_NUMBER (SERIALNOID, SERIALNO, DESCRIPTION, MODELID, MODELNAME, BRAND, COMPANYID, STATUS)   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | SERIALNOID | SERIALNO | DESCRIPTION | MODELID | MODELNAME | BRAND | COMPANYID | STATUS | | 1 | SGH0010001 | AD/DNS | 1 | HP DL360 GEN9 8SFF CTO | HP | 1 | ACTIVE | | 2 | SGH0010002 | AD/DNS | 1 | HP DL360 GEN9 8SFF CTO | HP | 2 | ACTIVE | | | |
| **3NF**  **Entity: SERIAL\_NUMBER**  (SERIALNOID, SERIALNO, DESCRIPTION, MODELID, COMPANYID, STATUS)   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | SERIALNOID | SERIALNO | DESCRIPTION | MODELID | COMPANYID | STATUS | | 1 | SGH0010001 | AD/DNS | 1 | 1 | ACTIVE | | 2 | SGH0010002 | AD/DNS | 1 | 2 | ACTIVE | | **Entity: MODEL** (MODELID, MODELNAME, BRAND)   |  |  |  | | --- | --- | --- | | MODELID | MODELNAME | BRAND | | 1 | HP DL360 GEN9 8SFF CTO | HP | | |
| Diagram 7.1 Normalisation of SERIAL\_NUMBER Entity | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UNF:** REQUEST\_FEEDBACK (FEEDBACKID, FEEDBACKDES, FILEID, FILENAME, FILEPATH, FILESIZE, REQUESTID, UPDATEDBY, UPDATEDON)  **1NF:** REQUEST\_FEEDBACK (FEEDBACKID, FEEDBACKDES, FILEID, FILENAME, FILEPATH, FILESIZE, REQUESTID, UPDATEDBY, UPDATEDON)  **2NF:** REQUEST\_FEEDBACK (FEEDBACKID, FEEDBACKDES, FILEID, FILENAME, FILEPATH, FILESIZE, REQUESTID, UPDATEDBY, UPDATEDON)   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | FEEDBACKID | FEEDBACKDES | FILEID | FILENAME | FILEPATH | FILESIZE | REQUESTID | UPDATEDBY | UPDATEDON | | 8 | Problem fixed. | 4 | image4 | Users\Repair | 9 KB | 8 | 9 | 13/11/2018 | | 8 | Problem fixed. | 5 | image5 | Users\Repair | 20 KB | 8 | 9 | 13/11/2018 | | |
| **3NF**  **Entity: REQUEST\_FEEDBACK**  (FEEDBACKID, FEEDBACKDES, REQUESTID, UPDATEDBY, UPDATEDON)   |  |  |  |  |  | | --- | --- | --- | --- | --- | | FEEDBACK ID | FEEDBACKDES | REQUESTID | UPDATED BY | UPDATED ON | | 8 | Problem fixed. | 8 | 9 | 13/11/2018 | | **Entity: REQUEST\_FILE**  (FILEID, FILENAME, FILEPATH, FILESIZE, FEEDBACKID)   |  |  |  |  |  | | --- | --- | --- | --- | --- | | FILEID | FILENAME | FILEPATH | FILESIZE | FEEDBACKID | | 4 | image4 | Users\Repair | 9 KB | 8 | | 5 | image5 | Users\Repair | 20 KB | 8 | |
| Diagram 7.2 Normalization of REQUEST\_FEEDBACK Entity | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UNF:** USER (USER ID, USERNAME, PASSWORD, TEAMID, TEAMNAME, DESCRIPTION, FULLNAME, HPNO, OFFICENO, FAXNO, EMAIL, STATUS)  **1NF:** USER (USER ID, USERNAME, PASSWORD, TEAMID, TEAMNAME, DESCRIPTION, FULLNAME, HPNO, OFFICENO, FAXNO, EMAIL, STATUS)  **2NF:** USER (USER ID, USERNAME, PASSWORD, TEAMID, TEAMNAME, DESCRIPTION, FULLNAME, HPNO, OFFICENO, FAXNO, EMAIL, STATUS)   |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | USER ID | USER NAME | PASS WORD | TEAMID | TEAMNAME | DESCRIPTION | FULLNAME | HPNO | OFFICE NO | FAX NO | EMAIL | STATUS | | 2 | USER\_ DEV | 123! @#ABC | 2 | DEVELOPER | DEVELOPER OF THE SYSTEM | DEVELOPER | 012-3456789 | 03-12340002 | - | DEVELOPER @MP.COM.MY | ACTIVE | | 3 | USER\_ DEV2 | 123! @#ABC | 2 | DEVELOPER | DEVELOPER OF THE SYSTEM | DEVELOPER2 | 012-3456790 | 03-12340003 | - | DEV2 @MP.COM.MY | ACTIVE | |
| **3NF**  **Entity: TEAM\_USER** (USERID, USERNAME, PASSWORD, TEAMID, FULLNAME, HPNO, OFFICENO, FAXNO, EMAIL, STATUS)   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | USERID | USERNAME | PASSWORD | TEAMID | FULLNAME | HPNO | OFFICENO | FAXNO | EMAIL | STATUS | | 2 | USER\_DEV | 123!@#ABC | 2 | DEVELOPER | 012-3456789 | 03-12340002 | - | DEVELOPER@MP.COM.MY | ACTIVE | | 3 | USER\_DEV2 | 123!@#ABC | 2 | DEVELOPER2 | 012-3456790 | 03-12340003 | - | DEV2@MP.COM.MY | ACTIVE |   **Entity: TEAM** (TEAMID, TEAMNAME, DESCRIPTION)   |  |  |  | | --- | --- | --- | | TEAMID | TEAMNAME | DESCRIPTION | | 2 | DEVELOPER | DEVELOPER OF THE SYSTEM | |
| Diagram 7.3 Normalisation of TEAM\_USER Entity |

8.0 Implementable ER Diagram

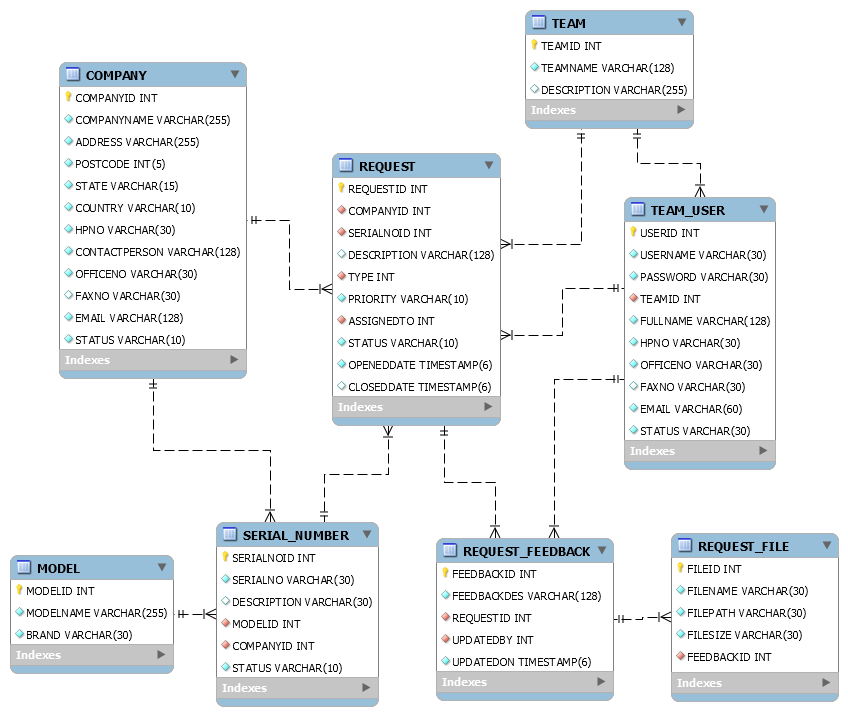
Diagram 8.1 indicates the implementable ER Diagram after normalization. 

Diagram 8.1 Implementable ER Diagram

9.0 Conclusion

In conclusion, with the help of Entity-Relationship diagram we can create the required database and perform queries. Helpdesk system of MasterPlan Sdn Bhd helps to strengthen the company's relationship with its clients. Clients of MasterPlan can always rely on the helpdesk system to solve problems of their respective project efficiently. With such database designed, the company can manage all client's details and project requests effectively and hence, MasterPlan has established itself as a trustworthy brand that have been serving various government bodies for decades. To make this successful, we must understand the business processes and business rules thoroughly. This project assignment requires essential attributes of teamwork such as communication, commitment, providing support, and sharing ideas and responsibility. Each team member gives excellent cooperation throughout the process.

10.0 SQL Statement

10.1 Maintaining data and records (by User)

|  |
| --- |
| 1. To maintain (enter, update, and delete) data on company. |
| **1.1 To enter data on company**  INSERT INTO COMPANY (COMPANYID, COMPANYNAME, ADDRESS, POSTCODE, STATE, COUNTRY, HPNO, CONTACTPERSON, OFFICENO, FAXNO, EMAIL, STATUS)  VALUES ('2', 'MINISTRY OF EDUCATION MALAYSIA/KEMENTERIAN PENDIDIKAN MALAYSIA(MOE/KPKKPM)', 'BLOK E8, KOMPLEKS E, PUSAT PENTADBIRAN KERAJAAN PERSEKUTUAN,62604, PUTRAJAYA', '62604', 'PUTRAJAYA', 'MALAYSIA', '03-80008000', 'HAJI ZAIDI BIN YAZID', '03-80008000, '03-88846411', 'ZAIDI.YAZID@MOE.GOV.MY', 'ACTIVE'); |
| **1.2 To update data on company**  UPDATE COMPANY  SET STATUS = 'INACTIVE'  WHERE COMPANYID = 1; |
| **1.3 To delete data on company**  DELETE COMPANY  WHERE COMPANYID = 1; |

|  |
| --- |
| 1. To maintain (enter, update, and delete) data on user. |
| **2.1 To enter data on user**  INSERT INTO TEAM\_USER (USERID, USERNAME, PASSWORD, TEAMID, FULLNAME, HPNO, OFFICENO, EMAIL, STATUS)  VALUES ('2', 'USER\_DEV', '123!@#ABC', '2', 'DEVELOPER', '012-3456789','03-12340002', 'DEVELOPER@MP.COM.MY', 'ACTIVE'); |
| **2.2 To update data on user**  UPDATE TEAM\_USER  SET HPNO = '010-0000000'  WHERE USERID = 1; |
| **2.3 To delete data on user**  DELETE TEAM\_USER  WHERE USERID = 2; |

|  |
| --- |
| 1. To maintain (enter, update, and delete) data on serial number. |
| **3.1 To enter data on serial number**  INSERT INTO SERIAL\_NUMBER (SERIALNOID, SERIALNO, DESCRIPTION, MODELID, COMPANYID, STATUS)  VALUES ('153', '9Q28HD2', 'WORKSTATION', '15', '7', 'ACTIVE'); |
| **3.2 To update data on serial number**  UPDATE SERIAL\_NUMBER  SET DESCRIPTION = 'WORKSTATION'  WHERE SERIALNOID = 152; |
| **3.3 To delete data on serial number**  DELETE SERIAL\_NUMBER  WHERE SERIALNOID = 152; |

|  |
| --- |
| 1. To maintain (enter and update) data on request. |
| **4.1 To enter data of request**  INSERT INTO REQUEST (REQUESTID, COMPANYID, SERIALNOID, DESCRIPTION, TYPE, PRIORITY, ASSIGNEDTO, STATUS, OPENEDDATE, CLOSEDDATE)  VALUES ('1', '1', '13', 'REPAIR', '5', ‘MEDIUM’, '8', 'CLOSED', '09-APR-18', '14-APR-18'); |
| **4.2 To update data of request**  UPDATE REQUEST  SET PRIORITY = 'HIGH'  WHERE REQUESTID = 1; |

|  |
| --- |
| 1. To maintain (enter and update) data on request feedback. |
| **5.1 To enter data of request feedback**  INSERT INTO REQUEST\_FEEDBACK (FEEDBACKID, FEEDBACKDES, REQUESTID, UPDATEDBY, UPDATEDON)  VALUES ('2', 'File named "image11" has been updated.', '1', '8', '10-APR-18'); |
| **5.2 To update data of request feedback**  UPDATE REQUEST\_FEEDBACK  SET FEEDBACKDES = 'File named "image1" has been updated.'  WHERE FEEDBACKID = 2; |

10.2 Searching data and records (by User)

|  |  |
| --- | --- |
| 1. Find the username of user where TEAMID = 3. | |
| SELECT USERID, USERNAME FROM TEAM\_USER  WHERE TEAMID = 3  AND TEAMID IN  (SELECT TEAMID FROM TEAM  WHERE TEAM\_USER.STATUS = 'ACTIVE'); |  |

|  |  |
| --- | --- |
| 1. Find the serial number where COMPANYID = 5. | |
| SELECT SERIALNOID, SERIALNO FROM SERIAL\_NUMBER  WHERE COMPANYID = 5  AND SERIALNOID IN  (SELECT SERIALNOID  FROM COMPANY  WHERE COMPANY.STATUS = 'ACTIVE'); |  |

|  |
| --- |
| 1. **Find the detail of company in IOI that the serial number for the company is/are still active.** |
| SELECT \* FROM COMPANY  WHERE ADDRESS LIKE '%IOI%'  AND COMPANYID IN  (SELECT COMPANYID FROM SERIAL\_NUMBER  WHERE SERIAL\_NUMBER.STATUS = 'ACTIVE'); |

|  |
| --- |
| 1. **Search the details of user who is assigned to request.** |
| SELECT REQUEST.REQUESTID, REQUEST.DESCRIPTION, REQUEST.ASSIGNEDTO, TEAM\_USER.USERNAME, TEAM\_USER.HPNO, TEAM\_USER.EMAIL, REQUEST.STATUS FROM REQUEST, TEAM\_USER  WHERE REQUEST.ASSIGNEDTO IN  (SELECT USERID FROM TEAM\_USER  WHERE STATUS = 'ACTIVE')  AND REQUEST.ASSIGNEDTO = TEAM\_USER.USERID  ORDER BY ASSIGNEDTO; |

|  |
| --- |
| 1. **Find the inactive serial number of product where COMPANYID = 1.** |
| SELECT COMPANY.COMPANYNAME, MODEL.MODELNAME AS MODEL, SERIALNO FROM SERIAL\_NUMBER  INNER JOIN COMPANY ON  COMPANY.COMPANYID = SERIAL\_NUMBER.COMPANYID  INNER JOIN MODEL ON MODEL.MODELID = SERIAL\_NUMBER.MODELID  WHERE COMPANY.COMPANYID = 1  AND SERIALNO IN  (SELECT SERIALNO FROM SERIAL\_NUMBER  WHERE STATUS = 'INACTIVE'); |

10.3 Tracking data and record (by User)

|  |
| --- |
| 1. **Keep track the request(s) that was/were opened.** |
| SELECT REQUESTID, REQUEST.DESCRIPTION, COMPANYNAME AS COMPANY, SERIALNO AS SERIAL\_NUMBER, TEAMNAME AS TYPE, PRIORITY, USERNAME AS ASSIGNED\_TO, REQUEST.OPENEDDATE, EXTRACT (DAY FROM (SYSDATE - OPENEDDATE)) AS DATE\_DIFF  FROM REQUEST  INNER JOIN SERIAL\_NUMBER ON  SERIAL\_NUMBER.SERIALNOID = REQUEST.SERIALNOID  INNER JOIN COMPANY ON  COMPANY.COMPANYID = REQUEST.COMPANYID  INNER JOIN TEAM ON TEAM.TEAMID = REQUEST.TYPE  INNER JOIN TEAM\_USER ON  TEAM\_USER.USERID = REQUEST.ASSIGNEDTO  WHERE REQUEST.STATUS = 'OPENED'  ORDER BY OPENEDDATE DESC; |

|  |
| --- |
| 1. **Find the details of Company Ministry of Education, MOE where the company request is still opened.** |
| SELECT \* FROM COMPANY  WHERE COMPANYNAME LIKE '%MOE%'  AND COMPANYID IN  (SELECT COMPANYID FROM REQUEST  WHERE REQUEST.STATUS = 'OPENED'); |

|  |
| --- |
| 1. **Find the serial number (SGH0010013) with the priority of the created request.** |
| SELECT SERIAL\_NUMBER.SERIALNO AS SERIALNO, MODEL.MODELNAME AS MODEL, SERIAL\_NUMBER.DESCRIPTION, COMPANY.COMPANYNAME AS COMPANY, REQUEST.PRIORITY, COUNT (REQUEST.REQUESTID) AS REQUEST\_ PRIORITY \_COUNT  FROM SERIAL\_NUMBER  INNER JOIN MODEL ON MODEL.MODELID = SERIAL\_NUMBER.MODELID  INNER JOIN COMPANY ON  COMPANY.COMPANYID = SERIAL\_NUMBER.COMPANYID  INNER JOIN REQUEST ON  REQUEST.SERIALNOID = SERIAL\_NUMBER.SERIALNOID  WHERE SERIAL\_NUMBER.COMPANYID IN  (SELECT COMPANYID FROM COMPANY  WHERE COMPANY.STATUS = 'ACTIVE')  AND SERIALNO = 'SGH0010013'  GROUP BY SERIALNO, MODELNAME, SERIAL\_NUMBER.DESCRIPTION, COMPANYNAME, REQUEST.PRIORITY; |

|  |
| --- |
| 1. **Find the file uploaded by USERID = 9.** |
| SELECT REQUEST.DESCRIPTION AS REQUEST\_TITLE, REQUEST\_FEEDBACK.FEEDBACKDES AS FEEDBACK, REQUEST\_FILE.FILENAME, REQUEST\_FILE.FILEPATH, REQUEST\_FEEDBACK.UPDATEDON  FROM REQUEST\_FILE, REQUEST\_FEEDBACK, REQUEST  WHERE REQUEST\_FEEDBACK.FEEDBACKID IN  (SELECT FEEDBACKID FROM REQUEST\_FEEDBACK  WHERE  REQUEST\_FEEDBACK.FEEDBACKID = REQUEST\_FILE.FEEDBACKID  AND REQUEST.REQUESTID IN  (SELECT REQUESTID FROM REQUEST  WHERE REQUEST\_FEEDBACK.REQUESTID = REQUEST.REQUESTID AND REQUEST.ASSIGNEDTO = 9)  ) ORDER BY FILEID; |

10.4 Generating reports (by User)

|  |  |
| --- | --- |
| 1. **To find the most reported brand** | |
| SELECT MODEL.BRAND, COUNT (MODEL.BRAND) AS REPORTEDTIMES  FROM MODEL  INNER JOIN SERIAL\_NUMBER  INNER JOIN REQUEST ON  SERIAL\_NUMBER.SERIALNOID = REQUEST.SERIALNOID  ON MODEL.MODELID = SERIAL\_NUMBER.MODELID  GROUP BY MODEL.BRAND  ORDER BY REPORTEDTIMES DESC; |  |

|  |
| --- |
| 1. **To find the most reported model** |
| SELECT MODELNAME, BRAND, COUNT (MODELNAME) AS REPORTEDTIMES  FROM MODEL  INNER JOIN SERIAL\_NUMBER  INNER JOIN REQUEST ON SERIAL\_NUMBER.SERIALNOID = REQUEST.SERIALNOID  ON MODEL.MODELID = SERIAL\_NUMBER.MODELID  GROUP BY MODEL.MODELNAME, MODEL.BRAND  ORDER BY REPORTEDTIMES DESC; |

|  |
| --- |
| 1. **To find the number of requests, total time taken, and average time taken for requests solved by teams where the total solving times is more than 10 days.** |
| SELECT TEAM.TEAMID, TEAM.TEAMNAME, COUNT (TEAM.TEAMID) AS SOLVED\_CASES, SUM (EXTRACT (DAY FROM (REQUEST.CLOSEDDATE - REQUEST.OPENEDDATE))) AS TOTAL\_SOLVING\_TIMES  , AVG (EXTRACT (DAY FROM (REQUEST.CLOSEDDATE - REQUEST.OPENEDDATE))) AS AVERAGE\_SOLVING\_TIMES  FROM TEAM  INNER JOIN TEAM\_USER  INNER JOIN REQUEST ON TEAM\_USER.USERID = REQUEST.ASSIGNEDTO  ON TEAM.TEAMID = TEAM\_USER.TEAMID  GROUP BY TEAM.TEAMID, TEAM.TEAMNAME  HAVING SUM (EXTRACT (DAY FROM (REQUEST.CLOSEDDATE - REQUEST.OPENEDDATE))) > 10  ORDER BY SOLVED\_CASES DESC; |

|  |
| --- |
| 1. **To find the number of requests, total time taken, and average time taken for requests solved by team users** |
| SELECT TEAM\_USER.USERID, TEAM\_USER.FULLNAME, COUNT (TEAM\_USER.USERID) AS SOLVED\_CASES, SUM (EXTRACT (DAY FROM (REQUEST.CLOSEDDATE - REQUEST.OPENEDDATE))) AS TOTAL\_SOLVING\_TIMES  , ROUND (AVG (EXTRACT (DAY FROM (REQUEST.CLOSEDDATE - REQUEST.OPENEDDATE)))) AS AVERAGE\_SOLVING\_TIMES  FROM TEAM\_USER  INNER JOIN REQUEST ON TEAM\_USER.USERID = REQUEST.ASSIGNEDTO  GROUP BY TEAM\_USER.USERID, TEAM\_USER.FULLNAME  ORDER BY SOLVED\_CASES DESC; |

|  |
| --- |
| 1. **To find the number of requests, total time taken, and average time taken for requests solved by team** |
| SELECT TEAM.TEAMNAME AS REQUEST\_TYPE, COUNT (TEAM.TEAMID) AS SOLVED\_CASES, SUM (EXTRACT (DAY FROM (REQUEST.CLOSEDDATE - REQUEST.OPENEDDATE))) AS TOTAL\_SOLVING\_TIMES, ROUND (AVG(EXTRACT (DAY FROM (REQUEST.CLOSEDDATE - REQUEST.OPENEDDATE)))) AS AVERAGE\_SOLVING\_TIMES  FROM TEAM  INNER JOIN REQUEST ON TEAM.TEAMID = REQUEST.TYPE  GROUP BY TEAM.TEAMID, TEAM.TEAMNAME  ORDER BY SOLVED\_CASES DESC; |

|  |
| --- |
| 1. **To find team users who have opened case** |
| SELECT TEAM\_USER.USERID, TEAM\_USER.FULLNAME AS USERNAME, REQUEST.REQUESTID AS OPENEDREQUESTID, REQUEST\_FEEDBACK.FEEDBACKDES AS FEEDBACK, REQUEST\_FEEDBACK.UPDATEDON  FROM TEAM\_USER, REQUEST, REQUEST\_FEEDBACK  WHERE REQUEST.STATUS = 'OPENED'  AND REQUEST.ASSIGNEDTO = TEAM\_USER.USERID  AND REQUEST.REQUESTID = REQUEST\_FEEDBACK.REQUESTID  ORDER BY OPENEDREQUESTID; |

|  |  |
| --- | --- |
| 1. **To find the number of cases opened per month** | |
| SELECT EXTRACT (month FROM openeddate) "MONTH",  COUNT (openeddate) "REQEUSTSPERMONTH"  FROM REQUEST  GROUP BY EXTRACT (month FROM openeddate)  ORDER BY "MONTH" ASC; |  |

|  |
| --- |
| 1. **To find the number of cases opened per month by user.** |
| SELECT EXTRACT (MONTH FROM REQUEST.OPENEDDATE) "MONTH", TEAM\_USER.FULLNAME AS USERNAME, COUNT (REQUEST.OPENEDDATE) "REQEUSTSPERMONTH"  FROM REQUEST, TEAM\_USER  WHERE REQUEST.ASSIGNEDTO = TEAM\_USER.USERID  GROUP BY EXTRACT (MONTH FROM REQUEST.OPENEDDATE), TEAM\_USER.FULLNAME  ORDER BY "MONTH" ASC; |

|  |
| --- |
| 1. **To find the number of cases opened per month by type.** |
| SELECT EXTRACT (MONTH FROM OPENEDDATE) "MONTH", TEAM.TEAMNAME,  COUNT (OPENEDDATE) "REQEUSTSPERMONTH"  FROM REQUEST, TEAM  WHERE TEAMNAME IN  (SELECT TEAMNAME FROM TEAM  WHERE TEAM.TEAMID = REQUEST.TYPE)  GROUP BY EXTRACT (MONTH FROM OPENEDDATE), TEAMNAME  ORDER BY "MONTH" ASC; |

11.0 Screenshots of Tables

Diagram 11.1 to Diagram 11.8 show the SQL scripts, tables and data for each entity. The attribute, attribute type and constraints of each entity have been identified in implemented ER diagram. To create entities with attribute domain for every entity in Oracle Application Express database, these information are important.

|  |
| --- |
| **SQL**  CREATE TABLE "COMPANY"  ( "COMPANYID" NUMBER,  "COMPANYNAME" VARCHAR2(255) NOT NULL ENABLE,  "ADDRESS" VARCHAR2(255) NOT NULL ENABLE,  "POSTCODE" NUMBER NOT NULL ENABLE,  "STATE" VARCHAR2(15) NOT NULL ENABLE,  "COUNTRY" VARCHAR2(10) NOT NULL ENABLE,  "HPNO" VARCHAR2(30) NOT NULL ENABLE,  "CONTACTPERSON" VARCHAR2(128) NOT NULL ENABLE,  "OFFICENO" VARCHAR2(30) NOT NULL ENABLE,  "FAXNO" VARCHAR2(30),  "EMAIL" VARCHAR2(128) NOT NULL ENABLE,  "STATUS" VARCHAR2(10) NOT NULL ENABLE,  CONSTRAINT "COMPANY\_PK" PRIMARY KEY ("COMPANYID") ENABLE  ); |
| **Table** |
| **Data** |

Diagram 11.1 COMPANY Data Description

|  |
| --- |
| **SQL**  CREATE TABLE "MODEL"  ( "MODELID" NUMBER,  "MODELNAME" VARCHAR2(255) NOT NULL ENABLE,  "BRAND" VARCHAR2(30) NOT NULL ENABLE,  CONSTRAINT "MODEL\_PK" PRIMARY KEY ("MODELID") ENABLE  ); |
| **Table** |
| **Data** |

Diagram 11.2 MODEL Data Description

|  |
| --- |
| **SQL**  CREATE TABLE "REQUEST"  ( "REQUESTID" NUMBER, "COMPANYID" NUMBER NOT NULL ENABLE,  "SERIALNOID" NUMBER NOT NULL ENABLE, "DESCRIPTION" VARCHAR2(128),  "TYPE" NUMBER NOT NULL ENABLE,  "PRIORITY" VARCHAR2(10) NOT NULL ENABLE,  "ASSIGNEDTO" NUMBER NOT NULL ENABLE,  "STATUS" VARCHAR2(10) NOT NULL ENABLE,  "OPENEDDATE" TIMESTAMP (6) NOT NULL ENABLE,  "CLOSEDDATE" TIMESTAMP (6),  CONSTRAINT "REQUEST\_PK" PRIMARY KEY ("REQUESTID") ENABLE  ; ALTER TABLE "REQUEST" ADD CONSTRAINT "REQUEST\_COMPANY" FOREIGN KEY ("COMPANYID")  REFERENCES "COMPANY" ("COMPANYID") ENABLE; ALTER TABLE "REQUEST" ADD CONSTRAINT "REQUEST\_SERIAL\_NUMBER" FOREIGN KEY ("SERIALNOID")  REFERENCES "SERIAL\_NUMBER" ("SERIALNOID") ENABLE; ALTER TABLE "REQUEST" ADD CONSTRAINT "REQUEST\_TEAM" FOREIGN KEY ("TYPE")  REFERENCES "TEAM" ("TEAMID") ENABLE; ALTER TABLE "REQUEST" ADD CONSTRAINT "REQUEST\_USER" FOREIGN KEY ("ASSIGNEDTO")  REFERENCES "TEAM\_USER" ("USERID") ENABLE; |
| **Table** |
| **Data** |

Diagram 11.3 REQUEST Data Description

|  |
| --- |
| **SQL**  CREATE TABLE "REQUEST\_FEEDBACK"  ( "FEEDBACKID" NUMBER,  "FEEDBACKDES" VARCHAR2(128) NOT NULL ENABLE,  "REQUESTID" NUMBER NOT NULL ENABLE,  "UPDATEDBY" NUMBER NOT NULL ENABLE,  "UPDATEDON" TIMESTAMP (6) NOT NULL ENABLE,  CONSTRAINT "REQUEST\_FEEDBACK\_PK" PRIMARY KEY ("FEEDBACKID") ENABLE  ); ALTER TABLE "REQUEST\_FEEDBACK" ADD CONSTRAINT "REQUEST\_FEEDBACK\_REQUEST" FOREIGN KEY ("REQUESTID")  REFERENCES "REQUEST" ("REQUESTID") ENABLE; ALTER TABLE "REQUEST\_FEEDBACK" ADD CONSTRAINT "REQUEST\_FEEDBACK\_USER" FOREIGN KEY ("UPDATEDBY")  REFERENCES "TEAM\_USER" ("USERID") ENABLE; |
| **Table** |
| **Data** |

Diagram 11.4 REQUEST\_FEEDBACK Data Description

|  |
| --- |
| **SQL**  CREATE TABLE "REQUEST\_FILE"  ( "FILEID" NUMBER,  "FILENAME" VARCHAR2(30) NOT NULL ENABLE,  "FILEPATH" VARCHAR2(30) NOT NULL ENABLE,  "FILESIZE" VARCHAR2(30) NOT NULL ENABLE,  "FEEDBACKID" NUMBER NOT NULL ENABLE,  CONSTRAINT "REQUEST\_FILE\_PK" PRIMARY KEY ("FILEID") ENABLE  ); ALTER TABLE "REQUEST\_FILE" ADD CONSTRAINT "REQUEST\_FILE\_FEEDBACK" FOREIGN KEY ("FEEDBACKID")  REFERENCES "REQUEST\_FEEDBACK" ("FEEDBACKID") ENABLE; |
| **Table** |
| **Data** |

Diagram 11.5 REQUEST\_FILE Data Description

|  |
| --- |
| **SQL**  CREATE TABLE "SERIAL\_NUMBER"  ( "SERIALNOID" NUMBER,  "SERIALNO" VARCHAR2(30) NOT NULL ENABLE,  "DESCRIPTION" VARCHAR2(30),  "MODELID" NUMBER NOT NULL ENABLE,  "COMPANYID" NUMBER NOT NULL ENABLE,  "STATUS" VARCHAR2(10) NOT NULL ENABLE,  CONSTRAINT "SERIAL\_NUMBER\_PK" PRIMARY KEY ("SERIALNOID") ENABLE  ); ALTER TABLE "SERIAL\_NUMBER" ADD CONSTRAINT "SERIAL\_NUMBER\_COMPANY" FOREIGN KEY ("COMPANYID")  REFERENCES "COMPANY" ("COMPANYID") ENABLE; ALTER TABLE "SERIAL\_NUMBER" ADD CONSTRAINT "SERIAL\_NUMBER\_MODEL" FOREIGN KEY ("MODELID")  REFERENCES "MODEL" ("MODELID") ENABLE; |
| **Table** |
| **Data** |

Diagram 11.6 SERIAL\_NUMBER Data Description

|  |
| --- |
| **SQL**  CREATE TABLE "TEAM"  ( "TEAMID" NUMBER,  "TEAMNAME" VARCHAR2(128) NOT NULL ENABLE,  "DESCRIPTION" VARCHAR2(255),  CONSTRAINT "TEAM\_PK" PRIMARY KEY ("TEAMID") ENABLE  ); |
| **Table** |
| **Data** |

Diagram 11.7 TEAM Data Description

|  |
| --- |
| **SQL**  CREATE TABLE "TEAM\_USER"  ( "USERID" NUMBER,  "USERNAME" VARCHAR2(30) NOT NULL ENABLE,  "PASSWORD" VARCHAR2(30) NOT NULL ENABLE,  "TEAMID" NUMBER NOT NULL ENABLE,  "FULLNAME" VARCHAR2(128) NOT NULL ENABLE,  "HPNO" VARCHAR2(30) NOT NULL ENABLE,  "OFFICENO" VARCHAR2(30) NOT NULL ENABLE,  "FAXNO" VARCHAR2(30),  "EMAIL" VARCHAR2(60) NOT NULL ENABLE,  "STATUS" VARCHAR2(30) NOT NULL ENABLE,  CONSTRAINT "TEAM\_USER\_PK" PRIMARY KEY ("USERID") ENABLE  ); ALTER TABLE "TEAM\_USER" ADD CONSTRAINT "TEAM\_USER\_TEAM" FOREIGN KEY ("TEAMID")  REFERENCES "TEAM" ("TEAMID") ENABLE; |
| **Table** |
| **Data** |

Diagram 11.8 TEAM\_USER Data Description

12.0 Difficulties and Challenge Faced During the Database Development

During the development of the database, there were some difficulties and challenges faced. The first challenge is the collection of data from MasterPlan Company. At the beginning of the project, we must make an appointment with the representatives from the company in order to fully understand their business processes and data flow. It became a challenge because we have to make at least two meetings with the company representatives since the information we got regarding data workflow from the first meeting was ambiguous to our team members. However, the representatives from the company have made his information clearer during the second meeting and our group has successfully obtained useful information from the company as requested.

The second challenge is the lack of face-to-face communication among group members. Since all of our members are living at different residential colleges and some even stay outside of the campus, it is difficult for us to have a face-to-face meeting regularly to discuss our work flow and ideas. Thus, we have opted to use WhatsApp group chat as a platform for ideas and problems discussion among ourselves. Even though it is an excellent virtual platform for group discussion, miscommunication and misunderstanding can happen between text messages and this leads to confusion.

13.0 Lesson learned

There are a number of lessons learned throughout the execution of this group project. The purpose of this team project assignment is to provide students the opportunity to put theory into practice and gain proficiency using current database management software. It is used as a mechanism for helping us learn and actually implement different components of database languages (in the form of Oracle) and different features of a database management system.

Next, through this project, we are able to develop teamwork spirit. Good interpersonal communication is a pivotal key in a group project. Through group discussion, we manage to improve our communication skills. In a group project we must learn to tolerate with each other and be open-minded.

Lastly, this group project also helps us to receive social support and encouragement to take risks, develop new approaches to resolving differences, and establish a shared identity with other group members. We also learn that complex tasks can be broken down into smaller parts and steps. Thus, we can properly plan the project and manage time wisely and the work can be done on schedule.

Appendix

****

A picture took with interviewee from Masterplan at his office.

Start from left: Goh Sin Ni, Mr Lufti (Interviewee), Cheong Kah Seng, Liow Jing Wen, Emira Asyikin Bt Abdul Rahim, Aida Syafiqah Bt Mohd Azli