**REPORT**

Database Design and Testing for Delphii

**1. Introduction:**

Every individual agency now-a-days maintains a database to keep each document record and fetch them properly and efficiently when needed. It helps the agencies to manage their overall work in a proper and perfect way. Delphi Promotions is a promotion company and it handles advertising campaigns in order to promote the products of its clients. Based on suitable portfolio, the client chooses a particular campaign and Delphi produces specific adverts and arranges for the advertising space on relevant mediums. Delphi requires the tracking of its clients and campaigns with respective details properly in order to maintain efficiency, accuracy and scalability. This report aims at the design and development of a database for Delphi Promotions. This database will be useful to keep the record of all staffs, clients, campaigns, advertisements with their respective components and placements, meeting details. The job of overall management and dealing with the clients will become easier using this database. It will help the company to prevent the losses that may happen in dealing all the business using only paper trails. The company will also be able to maintain further communication with their clients after the respective campaign is finished. This will help the marketing sector of the company to flourish in future.

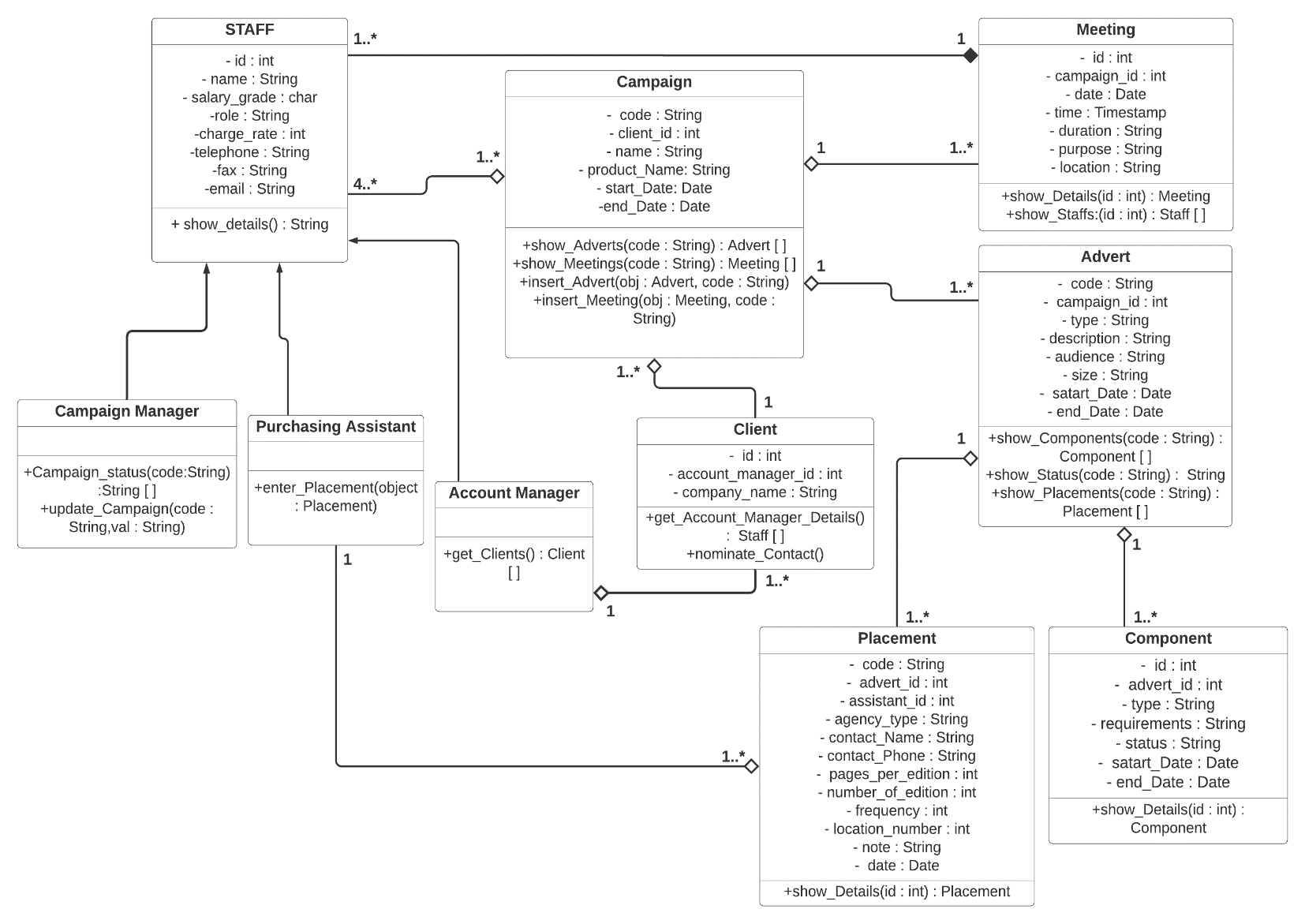
**Aims:**

1. Identify the necessary data and objects handled by the promotion company.
2. Develop the class diagram showing the interactions between objects of different classes.
3. Develop the database maintaining the interactions and using necessary constraints implied by the company.
4. Develop queries to fetch out necessary data as per the requirements of the company.
5. Use the efficiency of data management through the database for further marketing.

**Objectives:**

1. To keep the staff and client details for better management.
2. To keep the details of each of the campaign in order to monitor their status and update them efficiently when needed.
3. To maintain the details of staffs who attended the meetings in order to monitor their performance.
4. To help the clients in nominating suitable staff as their contact in corresponding campaign.
5. To assist the campaign managers in monitoring the status of each adverts and the job of placements.
6. To find out the eligible list of staff those can be assigned as account manager for clients.

**2. Class Diagram**

****

**Figure:** Class diagram of Delphi system

**Assumptions:**

1. A staff of the company of any particular role can work in one or more than one campaigns.
2. Each campaign has exactly one client but each client can involve in more than one campaigns.
3. Each campaign can have multiple adverts but each advert will belong to exactly one campaign. The campaign finishes when all of its adverts are completed.
4. Each advert can have multiple components but each component belongs to exactly one advert according to their respective requirements. The advert finishes when the design of all the components inside that advert is completed.
5. Each advert can have multiple placements but each placement belongs to exactly one advert. Each placement is supervised by a placement assistant who is a staff of the promotion company attached to that campaign.
6. Each component will have a status column which can hold three values. Not started, ongoing and ready.

**Constraints:**

1. Each table in the database has an id which is a primary key. This id uniquely identifies each row and it can’t be null.
2. Campaign, advert and placement tables maintain a code that uniquely identifies each object and can’t be null. This code will be useful for the application developer to query the database.
3. Each one to many relation in the database will be identified by a foreign key which can’t be null and the type of this key need to match the value type in the parent table. This foreign key will need the on delete cascade option enabled.
4. Tables with start and end date attributes will include check constraint with condition that start date is lower than end date.
5. Attributes that can’t have null values will need the not null option enabled.

**3. Implementation**

This section represents the definitions of the tables in the database.

1. SALARY Table

CREATE TABLE SALARY

(

SALARY\_ID NUMBER,

SALARY\_GRADE VARCHAR2(20) not null,

SALARY NUMBER not null,

constraint salary\_pk PRIMARY KEY(SALARY\_ID)

);

1. STAFF Table

CREATE TABLE STAFF

(

STAFF\_ID NUMBER,

STAFF\_NAME VARCHAR2(50) not null,

SALARY\_ID NUMBER not null,

STAFF\_ROLE VARCHAR2(40) not null,

CHARGE\_OUT\_RATE NUMBER not null,

TELEPHONE\_NO VARCHAR2(30) not null,

FAX\_NO VARCHAR2(30) not null,

EMAIL\_ID VARCHAR2(40) not null,

constraint staff\_pk PRIMARY KEY(STAFF\_ID),

constraint staff\_fk\_1 FOREIGN KEY(SALARY\_ID) references

SALARY(SALARY\_ID) on delete cascade

);

1. CLIENT Table

CREATE TABLE CLIENT

(

CLIENT\_ID NUMBER,

COMPANY\_NAME VARCHAR2(40) not null,

ACC\_MANAGER\_ID NUMBER not null,

constraint client\_pk PRIMARY KEY(CLIENT\_ID),

constraint client\_fk\_1 FOREIGN KEY(ACC\_MANAGER\_ID) references

STAFF(STAFF\_ID) on delete cascade

);

1. CAMPAIGN Table

CREATE TABLE CAMPAIGN

(

CAMPAIGN\_ID NUMBER,

CAMPAIGN\_NAME VARCHAR2(50) not null,

CAMPAIGN\_CODE VARCHAR2(15) not null unique,

CLIENT\_ID NUMBER not null,

CAMPAIGN\_PRODUCT VARCHAR2(40) not null,

START\_DATE DATE,

END\_DATE DATE,

constraint campaign\_pk PRIMARY KEY(CAMPAIGN\_ID),

constraint campaign\_fk\_1 FOREIGN KEY(CLIENT\_ID) references CLIENT(CLIENT\_ID) on

delete cascade,

constraint campaign\_date\_check CHECK(END\_DATE > START\_DATE)

);

1. MEETING Table

CREATE TABLE MEETING

(

MEETING\_ID NUMBER,

CAMPAIGN\_ID NUMBER not null,

DATE\_and\_TIME TIMESTAMP not null,

DURATION VARCHAR2(20) not null,

PURPOSE VARCHAR2(300) not null,

LOCATION VARCHAR2(100) not null,

constraint meeting\_pk PRIMARY KEY(MEETING\_ID),

constraint meeting\_fk\_1 FOREIGN KEY(CAMPAIGN\_ID) references

CAMPAIGN(CAMPAIGN\_ID) on delete cascade

);

1. STAFF\_CAMPAIGN Table

This table is an intermediate table and it stores the data that represents the many to many relation in between STAFF and CAMPAIGN tables.

CREATE TABLE STAFF\_CAMPAIGN

(

STAFF\_CAMPAIGN\_ID NUMBER,

STAFF\_ID NUMBER not null,

CAMPAIGN\_ID NUMBER not null,

constraint staff\_campaign\_pk PRIMARY KEY(STAFF\_CAMPAIGN\_ID),

constraint staff\_campaign\_fk\_1 FOREIGN KEY(STAFF\_ID) references STAFF(STAFF\_ID)

on delete cascade,

constraint staff\_campaign\_fk\_2 FOREIGN KEY(CAMPAIGN\_ID) references

CAMPAIGN(CAMPAIGN\_ID) on delete cascade,

constraint staff\_campaign\_unique UNIQUE(STAFF\_ID, CAMPAIGN\_ID)

);

1. STAFF\_MEETING Table

This table represents the many to many relation between STAFF and MEETING tables.

CREATE TABLE STAFF\_MEETING

(

STAFF\_MEETING\_ID NUMBER,

STAFF\_ID NUMBER not null,

MEETING\_ID NUMBER not null,

constraint staff\_meeting\_pk PRIMARY KEY(STAFF\_MEETING\_ID),

constraint staff\_meeting\_fk\_1 FOREIGN KEY(STAFF\_ID) references STAFF(STAFF\_ID)

on delete cascade,

constraint staff\_meeting\_fk\_2 FOREIGN KEY(MEETING\_ID) references

MEETING(MEETING\_ID) on delete cascade,

constraint staff\_meeting\_unique UNIQUE(STAFF\_ID, MEETING\_ID)

);

1. ADVERT Table

CREATE TABLE ADVERT

(

ADVERT\_ID NUMBER,

ADVERT\_CODE VARCHAR2(15) not null unique,

CAMPAIGN\_ID NUMBER not null,

ADVERT\_TYPE VARCHAR2(20) not null,

BRIEF\_DESC VARCHAR2(400) not null,

AUDIENCE VARCHAR2(300) not null,

ADVERT\_SIZE VARCHAR2(300) not null,

START\_DATE DATE,

END\_DATE DATE,

constraint advert\_pk PRIMARY KEY(ADVERT\_ID),

constraint advert\_check\_date CHECK(END\_DATE > START\_DATE),

constraint advert\_fk\_1 FOREIGN KEY(CAMPAIGN\_ID) references

CAMPAIGN(CAMPAIGN\_ID) on delete cascade

);

1. COMPONENT Table

CREATE TABLE COMPONENT

(

COMPONENT\_ID NUMBER,

ADVERT\_ID NUMBER not null,

TYPE VARCHAR2(100) not null,

REQUIREMENTS VARCHAR2(250) not null,

STATUS VARCHAR2(100) not null,

START\_DATE DATE,

END\_DATE DATE,

constraint component\_pk PRIMARY KEY(COMPONENT\_ID),

constraint component\_date\_check CHECK(END\_DATE > START\_DATE),

constraint component\_fk\_1 FOREIGN KEY(ADVERT\_ID) references

ADVERT(ADVERT\_ID) on delete cascade

);

1. PLACEMENT Table

CREATE TABLE PLACEMENT

(

PLACEMENT\_ID NUMBER,

ADVERT\_ID NUMBER not null,

ASSISTANT\_ID NUMBER not null,

PLACEMENT\_CODE VARCHAR2(30) not null,

PL\_AGENCY\_TYPE VARCHAR2(100) not null,

CONTACT\_NAME VARCHAR2(100) not null,

CONTACT\_PHONE VARCHAR2(50) not null,

PAGES\_PER\_EDITION NUMBER,

NUMBER\_OF\_EDITION NUMBER,

FREQUENCY VARCHAR2(100),

NUMBER\_OF\_LOCATIONS NUMBER,

NOTE VARCHAR2(300),

PLACEMENT\_DATE DATE,

constraint placement\_pk PRIMARY KEY(PLACEMENT\_ID),

constraint placement\_fk\_1 FOREIGN KEY(ADVERT\_ID) references

ADVERT(ADVERT\_ID) on delete cascade,

constraint placement\_fk\_2 FOREIGN KEY(ASSISTANT\_ID) references STAFF(STAFF\_ID)

on delete cascade

);

**4. Queries**

This section represents the queries that demonstrates the requirements of the system.

1. This query represents the staff information inside a particular campaign.

SELECT S.STAFF\_ID, S.STAFF\_NAME, S.STAFF\_ROLE, S.CHARGE\_OUT\_RATE CHARGE\_RATE\_PER\_HR\_in\_GBP, S.TELEPHONE\_NO, S.FAX\_NO, S.EMAIL\_ID

FROM STAFF S JOIN STAFF\_CAMPAIGN SC

ON S.STAFF\_ID = SC.STAFF\_ID

WHERE SC.CAMPAIGN\_ID =

(

SELECT CAMPAIGN\_ID

FROM CAMPAIGN

WHERE CAMPAIGN\_CODE = 'C01'

);

1. This query provides the staff information who are working as contact person. It includes the number of campaigns each staff worked as a contact person. This will help the clients to nominate contact persons from the staffs.

SELECT S.STAFF\_ID ID, S.STAFF\_NAME NAME, SA.SALARY SALARY, S.CHARGE\_OUT\_RATE CHARGE\_RATE, S.TELEPHONE\_NO, S.EMAIL\_ID, S.FAX\_NO,

(

SELECT COUNT(\*)

FROM STAFF\_CAMPAIGN

WHERE STAFF\_ID = S.STAFF\_ID

GROUP BY CAMPAIGN\_ID

) NUMBER\_OF\_CAMPAIGNS

FROM STAFF S JOIN SALARY SA

ON S.SALARY\_ID = SA.SALARY\_ID

WHERE UPPER(S.STAFF\_ROLE) = 'CONTACT PERSON';

1. This query gives the advert details of a particular campaign. It will help to generate the summary sheet of a campaign.

SELECT ADVERT\_CODE, ADVERT\_TYPE, BRIEF\_DESC, AUDIENCE, ADVERT\_SIZE, START\_DATE, END\_DATE

FROM ADVERT

WHERE CAMPAIGN\_ID =

(

SELECT CAMPAIGN\_ID

FROM CAMPAIGN

WHERE CAMPAIGN\_CODE = 'C01'

);

1. This query gives the placement details inside a particular advert. It will help the campaign manager to watch the placements of the adverts inside his/her own campaign.

SELECT P.PLACEMENT\_CODE, A.ADVERT\_CODE, P.ASSISTANT\_ID, S.STAFF\_NAME ASSISTANT\_NAME, P.PL\_AGENCY\_TYPE AGENCY\_TYPE, P.CONTACT\_NAME, P.CONTACT\_PHONE, P.FREQUENCY, P.NOTE, P.PLACEMENT\_DATE

FROM PLACEMENT P JOIN ADVERT A

ON P.ADVERT\_ID = A.ADVERT\_ID

JOIN STAFF S

ON P.ASSISTANT\_ID = S.STAFF\_ID

WHERE P.ADVERT\_ID =

(

SELECT ADVERT\_ID

FROM ADVERT

WHERE ADVERT\_CODE = 'A03'

);

1. This query provides the ongoing components inside a particular campaign. It will help the campaign manager to know which adverts in the campaign are ongoing and which are ready to be delivered.

SELECT COMPONENT\_ID, TYPE, REQUIREMENTS, STATUS, START\_DATE, END\_DATE

FROM COMPONENT

WHERE ADVERT\_ID IN

(

SELECT ADVERT\_ID

FROM ADVERT

WHERE CAMPAIGN\_ID = (SELECT CAMPAIGN\_ID FROM CAMPAIGN WHERE

CAMPAIGN\_CODE = ‘C01’)

) AND UPPER(STATUS) = 'ONGOING';

1. This query gives the details of each meeting that happened in a particular campaign. Each meeting also shows the number of staffs that attended that meeting.

SELECT M.DATE\_AND\_TIME, M.DURATION, M.PURPOSE, M.LOCATION,

(

SELECT COUNT(\*)

FROM STAFF\_MEETING

WHERE MEETING\_ID = M.MEETING\_ID

GROUP BY MEETING\_ID

) STAFFS

FROM MEETING M

WHERE M.CAMPAIGN\_ID =

(

SELECT CAMPAIGN\_ID

FROM CAMPAIGN

WHERE CAMPAIGN\_CODE = 'C01'

);

1. **Conclusion**

The database designed for the promotion company meets all of the stated requirements. The staff and the client table maintain the records of all the staffs and clients. Each client detail includes the staff id working as an account manager for that client. The campaign table, advert and meeting table holds the records of all campaigns with their corresponding adverts and meeting details. This meets the requirements of managing all the campaigns in a systematic way. The component and placement table holds the component and placement details of a particular advert. It will help the campaign manager to monitor the status of each advert within the campaign and to monitor the placements of the adverts in their respective mediums.

According to the assumption, the design of the database can change in several ways.

The company can have adverts which are restricted inside a fixed number of types. The system can maintain a separate table for the adverts and there will be a many to many relation in between campaigns and adverts and a separate table will represent this relation where each row will have a different column stating the description of that particular advert.

The one to many relation between campaigns and adverts won’t be needed.

Each advert can have components from a list with fixed number of things. This situation will require a many to many relation between advert and components and it will be represented by a separate intermediate table. Each row in this table will have starting and ending date column.

The company can have a fixed number of placement agencies. The database will maintain a separate table maintaining the details of the placement agencies. There will be a many to many relation between the adverts and placement agencies maintaining the details of each advertising placement. The one to many relation between adverts and placement agencies will be omitted.

Each placement may include more than one assistant belonging to the company staff. The system will require the one to many relation between staffs and placements replaced by a many to many relation represented by an intermediate table.

The company may want to balance the workload between the account managers. The clients will be assigned more than one account managers and an intermediate table between the staffs and the clients will represent this many to many relation.

The database of the overall system will evolve according to the scenario of the business, assumptions about the relations, and constraints about the attributes inside the tables of the database. The entire report represents the aim to develop the database for the company, class diagram showing the higher level details of each class including the relation among them, assumptions on which the database is developed, constraint those are maintained in the table, structure of individual tables, queries to demonstrate the effectiveness of the system and the alternatives which can be followed to extend the scope of the system.