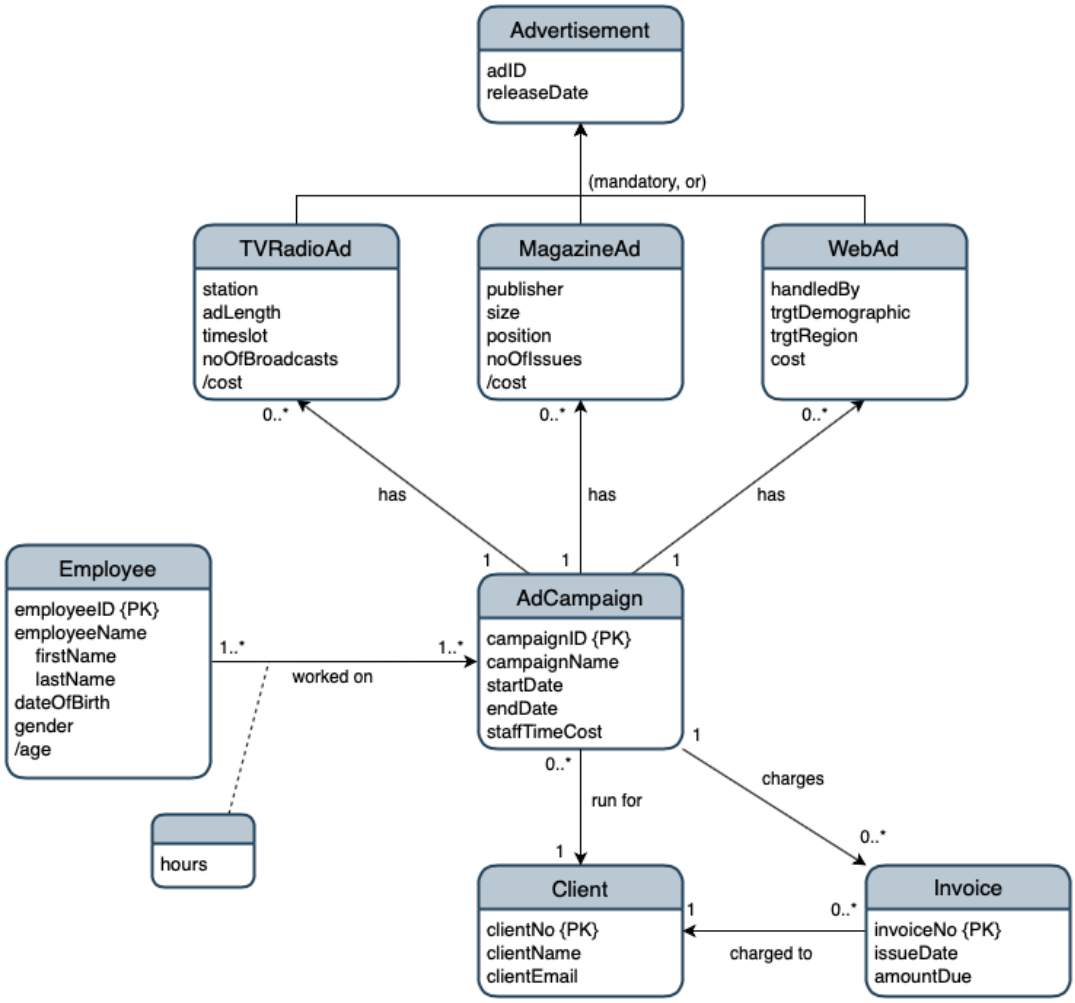
Database Management Systems (F28DM)
Coursework 1

Group 16
Andrew Philip, Navin Suresh Cordano, Varun Senthil Kumar, Gem Dias

Conceptual Model - Entity-Relationship Diagram



Notes:

- The subclasses of `'Advertisement'` are weak entities that cannot exist without a relation to an `'AdCampaign'`.
- All clients of HW Marketing are assumed to be companies and hence a single attribute (`'clientName'`) is used to store the name of the company.

Translation into Relational Schema

- Underlined attributes represent primary keys.
- Italicised attributes represent foreign keys.
- Underlined and italicised attributes represent keys that are both, primary keys and foreign keys.

```
Client (
      clientNo: integer(5),
      clientName: varchar(100),
      clientEmail: varchar(100)
)

AdCampaign (
      campaignID: integer(5),
      campaignName: varchar(100),
      startDate: date,
      endDate: date,
      staffTimeCost: decimal(10, 2),
      clientNo: integer(5)
)

Employee (
      employeeID: integer(5),
      firstName: varchar(50),
      lastName: varchar(50),
      dateOfBirth: date,
      gender: enum('M', 'F', 'N')
)

WorkedOn (
      employeeID: integer(5),
      campaignID: integer(5),
      hours: smallint
)


Invoice (
      invoiceNo: integer(5),
      campaignID: integer(5),
      clientNo: integer(5),
      issueDate: date,
      amountDue: decimal(10, 2)
)
```

```
TvRadioAd (
      campaignID: integer(5),
      adId: integer(5),
      releaseDate: date,
      station: varchar(50),
      adLength: smallint,
      timeslot: enum('prime time', day, 'night'),
      noOfBroadcasts: smallint
)

MagazineAd (
      campaignID: integer(5),
      adId: integer(5),
      releaseDate: date,
      publisher: varchar(50),
      size: decimal(3, 2),
      position: enum('inside front', 'inside back', 'other'),
      noOfIssues: smallint
)

WebAd (
      campaignId: integer(5),
      adId: integer(5),
      releaseDate: date,
      handledBy: varchar(50),
      trgtDemographic: varchar(100),
      trgtRegion: varchar(50),
      cost: decimal(10, 2)
)
```

Implementation of the Schema in MySQL

```sql
-- Creates table for Client
CREATE TABLE IF NOT EXISTS Client (
    clientNo INT(5),
    clientName VARCHAR(100) NOT NULL,
    clientEmail VARCHAR(100) NOT NULL,
    PRIMARY KEY (clientNo)
) ENGINE=INNODB;

-- Inserts values for Client
INSERT INTO Client VALUES(85231, 'Fitbit', 'ads@fitbit.com');
INSERT INTO Client VALUES(74236, 'Subway', 'promtions@subway.com');
INSERT INTO Client VALUES(86798, 'Intel', 'marketing@intel.com');
INSERT INTO Client VALUES(20880, 'Samsung', 'marketing@samsung.com');
INSERT INTO Client VALUES(32652, 'Starbucks', 'promtions@starbucks.com');


-- ----------------------------
```

```sql
-- Creates table for AdCampaign
CREATE TABLE IF NOT EXISTS AdCampaign (
    campaignID INT(5),
    campaignName VARCHAR(100) NOT NULL,
    startDate DATE,
    endDate DATE,
    staffTimeCost DECIMAL(10, 2) DEFAULT 0,
    clientNo INT(5) NOT NULL,
    PRIMARY KEY (campaignID)
) ENGINE=INNODB;

-- Inserts values for AdCampaign
INSERT INTO AdCampaign VALUES(1, 'Fitbit Versa 3 Smartwatch', '2020-03-16', '2021-12-14', 740, 85231);
INSERT INTO AdCampaign VALUES(2, 'Subway Wednesday Deals', '2020-07-27', '2022-05-13', 250, 74236);
INSERT INTO AdCampaign VALUES(3, 'Fitbit Charge 4 Special Edition', '2021-04-29', '2021-06-22', 440, 85231);
INSERT INTO AdCampaign VALUES(4, 'Samsung Galaxy Note 21 5G Launch', '2021-02-19', '2021-08-24', 870, 20880);
INSERT INTO AdCampaign VALUES(5, 'Starbucks Rewards Program', '2022-05-12', '2022-11-26', 480, 32652);


-- ----------------------------

-- Creates table for Employee
CREATE TABLE IF NOT EXISTS Employee (
    employeeID INT(5),
    firstName VARCHAR(50) NOT NULL,
    lastName VARCHAR(50) NOT NULL,
    dateOfBirth DATE,
    gender ENUM('M', 'F', 'N'),
    PRIMARY KEY (employeeID)
) ENGINE=INNODB;

-- Inserts values for Employee
INSERT INTO Employee VALUES(245, 'Nelson', 'Hoffman', '1989-02-18', 'M');
INSERT INTO Employee VALUES(727, 'Bryce', 'West', '1998-03-06', 'M');
INSERT INTO Employee VALUES(282, 'Monica', 'Todd', '1982-06-21', 'F');
INSERT INTO Employee VALUES(943, 'Malcolm', 'Murdock', '1988-11-18', 'M');
INSERT INTO Employee VALUES(634, 'Alexandra', 'Lyon', '1976-04-27', 'F');
INSERT INTO Employee VALUES(395, 'Marcos', 'Roman', '1977-12-29', 'M');


-- ----------------------------
```

```sql
-- Creates table for WorkedOn
CREATE TABLE IF NOT EXISTS WorkedOn (
    employeeID INT(5),
    campaignID INT(5),
    hours SMALLINT DEFAULT 0,
    PRIMARY KEY (employeeID, campaignID)
) ENGINE=INNODB;

-- Inserts values for WorkedOn
INSERT INTO WorkedOn VALUES(727, 5, 20);
INSERT INTO WorkedOn VALUES(245, 2, 25);
INSERT INTO WorkedOn VALUES(634, 1, 24);
INSERT INTO WorkedOn VALUES(395, 4, 35);
INSERT INTO WorkedOn VALUES(634, 4, 17);
INSERT INTO WorkedOn VALUES(282, 5, 12);
INSERT INTO WorkedOn VALUES(245, 3, 28);
INSERT INTO WorkedOn VALUES(943, 1, 2);
INSERT INTO WorkedOn VALUES(943, 4, 31);
INSERT INTO WorkedOn VALUES(727, 3, 16);
INSERT INTO WorkedOn VALUES(282, 4, 4);
INSERT INTO WorkedOn VALUES(245, 1, 48);
INSERT INTO WorkedOn VALUES(727, 2, 15);
INSERT INTO WorkedOn VALUES(634, 5, 16);


-- ----------------------------

-- Creates table for Invoice
CREATE TABLE IF NOT EXISTS Invoice (
    invoiceNo INT(5),
    campaignID INT(5) NOT NULL,
    clientNo INT(5) NOT NULL,
    issueDate DATE NOT NULL,
    amountDue DECIMAL(10, 2) NOT NULL,
    PRIMARY KEY (invoiceNo)
) ENGINE=INNODB;

-- Inserts values for Invoice
INSERT INTO Invoice VALUES(6887, 1, 85231, '2020-05-20', 61204.25);
INSERT INTO Invoice VALUES(6889, 2, 74236, '2020-04-16', 40075.85);
INSERT INTO Invoice VALUES(6131, 1, 85231, '2021-01-14', 3126.00);
INSERT INTO Invoice VALUES(6272, 3, 85231, '2020-11-08', 14882.82);
INSERT INTO Invoice VALUES(5291, 1, 85231, '2020-07-03', 5232.92);
INSERT INTO Invoice VALUES(2483, 2, 74236, '2020-02-23', 8283.64);
INSERT INTO Invoice VALUES(8232, 4, 20880, '2021-01-22', 7327.53);
INSERT INTO Invoice VALUES(4381, 3, 85231, '2020-05-08', 21992.39);
INSERT INTO Invoice VALUES(5823, 2, 74236, '2020-08-23', 3722.83);


-- ----------------------------
```

```sql
-- Creates table for TVRadioAd
CREATE TABLE IF NOT EXISTS TVRadioAd (
    campaignID INT(5),
    adId INT(5),
    releaseDate DATE NOT NULL,
    station VARCHAR(50) NOT NULL,
    adLength SMALLINT NOT NULL,
    timeslot ENUM('prime time', 'day', 'night') NOT NULL DEFAULT 'day',
    noOfBroadcasts SMALLINT NOT NULL DEFAULT 1,
    PRIMARY KEY (campaignID, adId)
) ENGINE=INNODB;

-- Inserts values for TVRadioAd
INSERT INTO TVRadioAd VALUES(1, 1001, '2021-01-25', 'Comedy Central', 23, 'prime
time', 15);
INSERT INTO TVRadioAd VALUES(2, 2532, '2021-02-17', '89.4 FM', 16, 'day', 4);
INSERT INTO TVRadioAd VALUES(1, 2300, '2020-05-10', 'HBO', 28, 'day', 12);
INSERT INTO TVRadioAd VALUES(3, 1421, '2021-08-12', 'National Geographic', 33,
'prime time', 40);
INSERT INTO TVRadioAd VALUES(3, 1239, '2021-09-12', 'BBC News', 25, 'night', 8);


-- ----------------------------

-- Creates table for MagazineAd
CREATE TABLE IF NOT EXISTS MagazineAd (
    campaignID INT(5),
    adId INT(5),
    releaseDate DATE NOT NULL,
    publisher VARCHAR(50) NOT NULL,
    size DECIMAL(3, 2) NOT NULL,
    position ENUM('inside front', 'inside back', 'other') NOT NULL DEFAULT 'other',
    noOfIssues SMALLINT NOT NULL DEFAULT 1,
    PRIMARY KEY (campaignID, adId)
) ENGINE=INNODB;

-- Inserts values for MagazineAd
INSERT INTO MagazineAd VALUES(2, 1001, '2020-09-18', 'TIME', 4.5, 'inside front',
45);
INSERT INTO MagazineAd VALUES(2, 2365, '2021-12-13', 'Men''s Health', 20.20,
'inside front', 90);
INSERT INTO MagazineAd VALUES(1, 2150, '2021-02-28', 'Vogue', 25.15, 'other', 120);
INSERT INTO MagazineAd VALUES(4, 6356, '2021-07-23', 'New York Times', 100.00,
'inside front', 65);
INSERT INTO MagazineAd VALUES(4, 3425, '2021-03-26', 'GQ', 32.05, 'inside back',
45);


-- ----------------------------
```

```sql
-- Creates table for WebAd
CREATE TABLE IF NOT EXISTS WebAd (
    campaignID INT(5),
    adId INT(5),
    releaseDate DATE NOT NULL,
    handledBy VARCHAR(50) NOT NULL,
    trgtDemographic VARCHAR(100) NOT NULL,
    trgtRegion VARCHAR(50) NOT NULL,
    cost DECIMAL(10, 2) NOT NULL,
    PRIMARY KEY (campaignID, adId)
) ENGINE=INNODB;

-- Inserts values for WebAd
INSERT INTO WebAd VALUES(1, 2300, '2021-07-17', 'Google', '18-35 year olds', 'UAE',
300.00);
INSERT INTO WebAd VALUES(2, 2365, '2020-12-23', 'Bing', 'professionals', 'India',
210.32);
INSERT INTO WebAd VALUES(2, 2532, '2021-02-03', 'YouTube', 'students', 'Australia',
900.15);


-- ------------------------------------------------------------


-- ------------------------------------------------------------

-- Adds foreign key constraint for client number in AdCampaign
-- Assumes that ad campaign information is deleted when a client stops working with
HW Marketing
ALTER TABLE AdCampaign
ADD CONSTRAINT AdCmpgnClientNo_fk
    FOREIGN KEY (clientNo)
    REFERENCES Client(clientNo)
    ON DELETE CASCADE ON UPDATE CASCADE;



-- WorkedOn details are retained to track costs for the ad campaigns

-- Adds foreign key constraint for employee ID in WorkedOn
ALTER TABLE WorkedOn
ADD CONSTRAINT WorkedOnEmpID_fk
    FOREIGN KEY (employeeID)
    REFERENCES Employee(employeeID)
    ON DELETE NO ACTION ON UPDATE CASCADE;

-- Adds foreign key constraint for campaign ID in WorkedOn
ALTER TABLE WorkedOn
ADD CONSTRAINT WorkedOnCmpnID_fk
    FOREIGN KEY (campaignID)
    REFERENCES AdCampaign(campaignID)
    ON DELETE NO ACTION ON UPDATE CASCADE;
```

```sql
-- Invoice details are retained to track costs of ad campaigns and expected
payments from clients

-- Adds foreign key constraint for campaign ID in Invoice
ALTER TABLE Invoice
ADD CONSTRAINT InvoiceCmpgnID_fk
    FOREIGN KEY (campaignID)
    REFERENCES AdCampaign(campaignID)
    ON DELETE NO ACTION ON UPDATE CASCADE;

-- Adds foreign key constraint for client number in Invoice
ALTER TABLE Invoice
ADD CONSTRAINT InvoiceClientNo_fk
    FOREIGN KEY (clientNo)
    REFERENCES Client(clientNo)
    ON DELETE NO ACTION ON UPDATE CASCADE;



-- Ads are deleted when their ad campaign is deleted as all ads must be related to
an ad campaign

-- Adds foreign key constraint for campaign ID in TVRadioAd
ALTER TABLE TVRadioAd
ADD CONSTRAINT TVRadioCmpgnID_fk
    FOREIGN KEY (campaignID)
    REFERENCES AdCampaign(campaignID)
    ON DELETE CASCADE ON UPDATE CASCADE;



-- Adds foreign key constraint for campaign ID in MagazineAd
ALTER TABLE MagazineAd
ADD CONSTRAINT MagCmpgnID_fk
    FOREIGN KEY (campaignID)
    REFERENCES AdCampaign(campaignID)
    ON DELETE CASCADE ON UPDATE CASCADE;



-- Adds foreign key constraint for campaign ID in WebAd
ALTER TABLE WebAd
ADD CONSTRAINT WebCmpgnID_fk
    FOREIGN KEY (campaignID)
    REFERENCES AdCampaign(campaignID)
    ON DELETE CASCADE ON UPDATE CASCADE;
```

Indexes

1) Client names in the Client table

```
-- Indexes client names
CREATE INDEX clientName ON Client(clientName);
```

This column is indexed as client names may be used to:

- search for a client's contact information
- select a particular client in join queries

2) Campaign names in the AdCampaign table

```
-- Indexes ad campaign names
CREATE INDEX campaignName ON AdCampaign(campaignName);
```

This column is indexed as campaign names may be used to select a particular campaign, especially in join queries. This is important as the AdCampaign table has relations to almost every other table in the database.

3) Employee names in the Employee table

```
-- Indexes employee names
CREATE INDEX employeeName ON Employee(firstName, lastName);
```

This column is indexed as employee names may be used to search for employee details. It may also be used in join queries to find hours worked by an employee.

4) Release dates in Advertisement tables

```
-- Indexes release dates for TVRadio, Magazine and Web ads
CREATE INDEX TVRadioReleaseDate ON TVRadioAd(releaseDate);
CREATE INDEX MagazineReleaseDate ON MagazineAd(releaseDate);
CREATE INDEX WebReleaseDate ON WebAd(releaseDate);
```

These columns are indexed as advertisements will likely need to be sorted in order of release date. This is useful for:

- getting information about ads that need to be released soon
- finding details of ads released in a particular time period

5) Invoice issue dates in Invoice table

```
-- Indexes issue dates for invoices
CREATE INDEX invoiceIssueDate ON Invoice(issueDate);
```

This column is indexed as invoices will likely be sorted in order of issue date. This is useful for:

- finding recently issued invoices for collection of payments
- finding invoices issued in a particular time period for accounting purposes

Queries over the Database

a) Find all the staff that have NOT previously worked on a RADIO advert campaign.

```
-- Shows names of employees that haven't worked on a TV or Radio ad campaign
SELECT DISTINCT CONCAT(firstName, ' ', lastName) AS employeeName
FROM Employee
WHERE employeeID NOT IN
      (SELECT DISTINCT W.employeeID
       FROM WorkedOn AS W
            INNER JOIN TVRadioAd AS T
            ON W.campaignID = T.campaignID);
```

Output:

```
+--------------+
| employeeName |
+--------------+
| Marcos Roman |
| Monica Todd  |
+--------------+
2 rows in set (0.001 sec)
```

b) For each client calculate how many adverts of each type (e.g. radio, web) have been marketed by the company.

```
-- Shows number of ads of each type for each client using left joins
SELECT C.clientName,
       IFNULL(T.count, 0) AS TVRadioAds,
       IFNULL(M.count, 0) AS MagazineAds,
       IFNULL(W.count, 0) AS WebAds
FROM Client AS C

     LEFT JOIN

     (SELECT A.clientNo, COUNT(*) AS count
      FROM AdCampaign AS A
           INNER JOIN TVRadioAd AS T
           ON A.campaignID = T.campaignID
      GROUP BY A.clientNo) AS T
     ON C.clientNo = T.clientNo

     LEFT JOIN
```

```
    (SELECT A.clientNo, COUNT(*) AS count
     FROM AdCampaign AS A
          INNER JOIN MagazineAd AS M
          ON A.campaignID = M.campaignID
     GROUP BY A.clientNo) AS M
    ON C.clientNo = M.clientNo

    LEFT JOIN

    (SELECT A.clientNo, COUNT(*) AS count
     FROM AdCampaign AS A
          INNER JOIN WebAd AS W
          ON A.campaignID = W.campaignID
     GROUP BY A.clientNo) AS W
    ON C.clientNo = W.clientNo;
```

Output:

```
+------------+------------+-------------+--------+
| clientName | TVRadioAds | MagazineAds | WebAds |
+------------+------------+-------------+--------+
| Fitbit     |          4 |           1 |      1 |
| Intel      |          0 |           0 |      0 |
| Samsung    |          0 |           2 |      0 |
| Starbucks  |          0 |           0 |      0 |
| Subway     |          1 |           2 |      2 |
+------------+------------+-------------+--------+
5 rows in set (0.002 sec)
```

c) Which employee (by name) has charged out the most time across all campaigns?

```
-- Shows the name of the employee who has spent the most time working on a single
campaign
SELECT CONCAT(E.firstName, ' ', E.lastName) AS employeeName,
       A.campaignID,
       A.campaignName,
       W.hours
FROM Employee AS E,
     AdCampaign AS A,
     (SELECT *
      FROM WorkedOn
      WHERE hours = (SELECT MAX(hours) FROM WorkedOn)) AS W
WHERE E.employeeID = W.employeeID
      AND A.campaignID = W.campaignID;
```

Output:

```
+---------------+------------+--------------------------+-------+
| employeeName  | campaignID | campaignName             | hours |
+---------------+------------+--------------------------+-------+
| Nelson Hoffman |          1 | Fitbit Versa 3 Smartwatch |    48 |
+---------------+------------+--------------------------+-------+
1 row in set (0.002 sec)
```

d) Write an UPDATE query to automatically calculate the total staff time cost for a specified advert campaign.

```sql
-- Updates the total staff time cost of the campaign with ID 2
-- Assumes an hourly rate of $10/hr for all employees
UPDATE AdCampaign AS A
       INNER JOIN
       (SELECT campaignID, SUM(hours) AS totalHours
        FROM WorkedOn
        GROUP BY campaignID) AS W
       ON A.campaignID = W.campaignID
SET A.staffTimeCost = W.totalHours * 10
WHERE A.campaignID = 2;
```

Output:

Before UPDATE:

```
+------------+-----------------------+------------+------------+---------------+----------+
| campaignID | campaignName          | startDate  | endDate    | staffTimeCost | clientNo |
+------------+-----------------------+------------+------------+---------------+----------+
|          2 | Subway Wednesday Deals | 2020-07-27 | 2022-05-13 |        250.00 |    74236 |
+------------+-----------------------+------------+------------+---------------+----------+
1 row in set (0.001 sec)
```

After UPDATE:

```
+------------+-----------------------+------------+------------+---------------+----------+
| campaignID | campaignName          | startDate  | endDate    | staffTimeCost | clientNo |
+------------+-----------------------+------------+------------+---------------+----------+
|          2 | Subway Wednesday Deals | 2020-07-27 | 2022-05-13 |        400.00 |    74236 |
+------------+-----------------------+------------+------------+---------------+----------+
1 row in set (0.001 sec)
```

e) Design your own ***complex query*** that contains at least one of the following features.

```sql
-- Shows hours spent by each employee on each campaign where the staff time cost is
less than 500
--
-- Features demonstrated:
-- 1) Joins involving a composite key
--    -> (employeeID, campaignID) in WorkedOn
-- 2) Nested negation, involving NOT EXISTS or NOT IN
--    -> getting ad campaigns where staff time cost is not greater than 500
-- 3) Using MySQL built in functions
--    -> CONCAT, LEFT

SELECT A.campaignID,
       A.campaignName,
       E.employeeID,
       CONCAT(E.lastName, ', ', LEFT(E.firstName, 1), '.') AS employeeName,
       W.hours
FROM WorkedOn AS W
     INNER JOIN Employee AS E
     ON W.employeeID = E.employeeID
     INNER JOIN AdCampaign AS A
     ON W.campaignID = A.campaignID
WHERE A.campaignID NOT IN
      (SELECT campaignID
       FROM AdCampaign
       WHERE staffTimeCost > 500)
ORDER BY A.campaignID ASC, W.hours DESC;
```

Output:

```
+------------+---------------------------------+------------+--------------+-------+
| campaignID | campaignName                    | employeeID | employeeName | hours |
+------------+---------------------------------+------------+--------------+-------+
|          2 | Subway Wednesday Deals          |        245 | Hoffman, N.  |    25 |
|          2 | Subway Wednesday Deals          |        727 | West, B.     |    15 |
|          3 | Fitbit Charge 4 Special Edition |        245 | Hoffman, N.  |    28 |
|          3 | Fitbit Charge 4 Special Edition |        727 | West, B.     |    16 |
|          5 | Starbucks Rewards Program       |        727 | West, B.     |    20 |
|          5 | Starbucks Rewards Program       |        634 | Lyon, A.     |    16 |
|          5 | Starbucks Rewards Program       |        282 | Todd, M.     |    12 |
+------------+---------------------------------+------------+--------------+-------+
7 rows in set (0.002 sec)
```

f) **A transaction** containing multiple CRUD (Create, Read, Update, Delete) operations, e.g. you may look up a person, change some value, and then check the update has happened, or you may insert some new data that requires multiple tables to be updated.

```sql
-- Starts a new transaction
START TRANSACTION;

-- Updates the WorkedOn table to add 8 hours to Bryce West's record on the
'Starbucks Rewards Program' campaign (Update)
UPDATE WorkedOn
SET hours = hours + 8
WHERE employeeID = (SELECT employeeID
                    FROM Employee
                    WHERE firstName = 'Bryce' AND lastName = 'West')
      AND
      campaignID = (SELECT campaignID
                    FROM AdCampaign
                    WHERE campaignName = 'Starbucks Rewards Program');

-- Updates the staff time cost for the 'Starbucks Rewards Program' campaign (same
as query (d)) (Update)
UPDATE AdCampaign AS A
       INNER JOIN
       (SELECT campaignID, SUM(hours) AS totalHours
        FROM WorkedOn
        GROUP BY campaignID) AS W
       ON A.campaignID = W.campaignID
SET A.staffTimeCost = W.totalHours * 10
WHERE A.campaignID = (SELECT campaignID
                      FROM AdCampaign
                      WHERE campaignName = 'Starbucks Rewards Program');

-- Displays the update staff time cost in the AdCampaign table (Read)
SELECT campaignID, campaignName, staffTimeCost
FROM AdCampaign
WHERE campaignName = 'Starbucks Rewards Program';

-- Reverts to state before transaction
ROLLBACK;
```

Output:

Before transaction:

```
+------------+---------------------------+---------------+
| campaignID | campaignName              | staffTimeCost |
+------------+---------------------------+---------------+
|          5 | Starbucks Rewards Program |        480.00 |
+------------+---------------------------+---------------+
1 row in set (0.001 sec)
```

Before ROLLBACK:

```
+------------+---------------------------+---------------+
| campaignID | campaignName              | staffTimeCost |
+------------+---------------------------+---------------+
|          5 | Starbucks Rewards Program |        560.00 |
+------------+---------------------------+---------------+
1 row in set (0.001 sec)
```

After ROLLBACK:

```
+------------+---------------------------+---------------+
| campaignID | campaignName              | staffTimeCost |
+------------+---------------------------+---------------+
|          5 | Starbucks Rewards Program |        480.00 |
+------------+---------------------------+---------------+
1 row in set (0.002 sec)
```

Group Member Contributions

Member	Contributions
Andrew	Relational Schema for Employee, Client and Invoice
Navin	Relational Schema for TVRadioAd, MagazineAd and WebAd
Varun	TVRadioAd and MagazineAd Entities for ER Diagram; Example data for Invoice, TVRadioAd, MagazineAd and WebAd
Gem	Client, AdCampaign, Employee, Invoice and WebAd Entities for ER Diagram; CREATE TABLE queries; Example data for Client, AdCampaign, Employee and WorkedOn tables; Indexes; Queries a) to f)