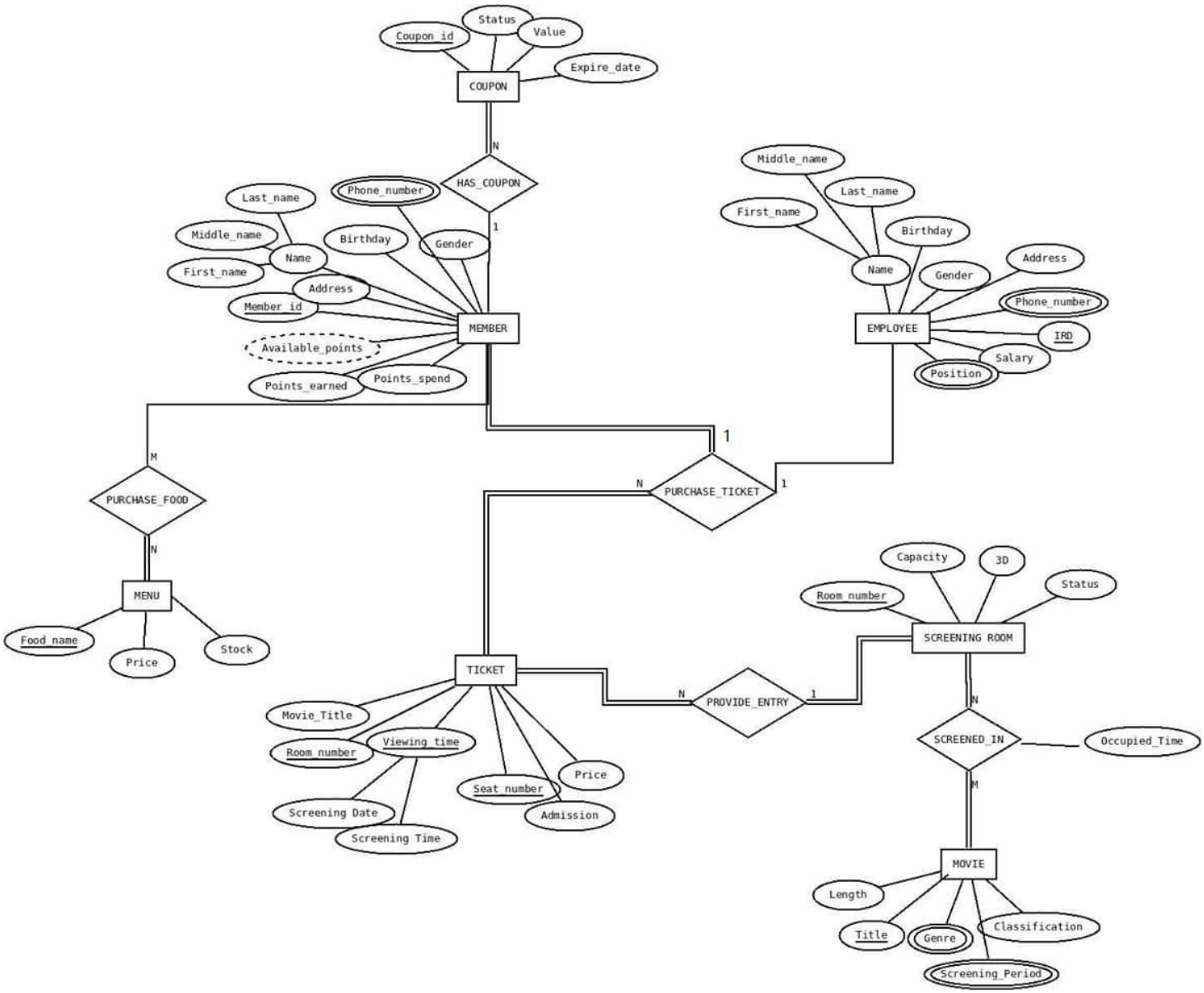
COSC344 Assignment 2

Team 4

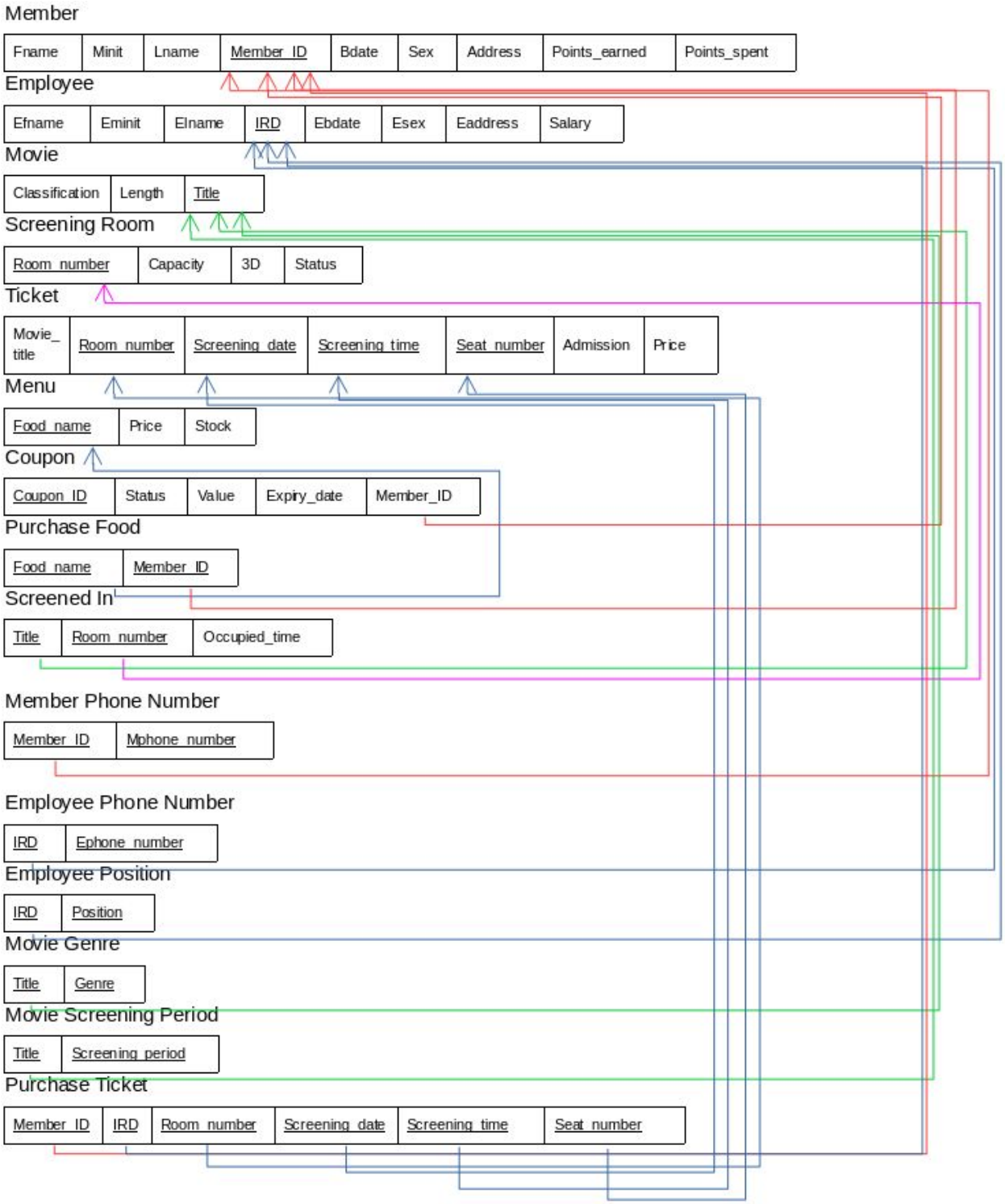
Members:

Mitchell, Caleb (Leader)
Ruiter, Jessy
Liu, Julia
Fielding-Woodmass, Callum

Revised ER Diagram



Relation Schema



Normalisation

MEMBER

Fname	Minit	Lname	Member_ID	Bdate	Sex	Address	Points_earned	Points_spent

Step1: 1NF
MEMBER is in 1NF because all attribute values are single atomic values.
Step2: 2NF
MEMBER is in 2NF because its primary key contains a single attribute, there is no partial dependency.
Step3: 3NF
MEMBER is in 3NF because no nonprime attribute is functionally determined by another nonprime attribute, that is, there is no transitive dependency of a nonprime attribute on the primary key.
Step4: BCNF
MEMBER is in BCNF because there is not a functional dependency where a nonprime attribute determines a key attribute.

EMPLOYEE

Efname	Eminit	Elname	IRD	Ebdate	Esex	Eaddress	Salary

Step1: 1NF
EMPLOYEE is in 1NF because all attribute values are single atomic values.
Step2: 2NF
EMPLOYEE is in 2NF because its primary key contains a single attribute, there is no partial dependency.
Step3: 3NF
EMPLOYEE is in 3NF because no nonprime attribute is functionally determined by another nonprime attribute, that is, there is no transitive dependency of a nonprime attribute on the primary key.
Step4: BCNF
EMPLOYEE is in BCNF because there is not a functional dependency where a nonprime attribute determines a key attribute.

MOVIE

Title	Length	Classification

Step1: 1NF
MOVIE is in 1NF because all attribute values are single atomic values.
Step2: 2NF
MOVIE is in 2NF because its primary key contains a single attribute, there is no partial dependency.
Step3: 3NF
MOVIE is in 3NF because no nonprime attribute is functionally determined by another nonprime attribute, that is, there is no transitive dependency of a nonprime attribute on the primary key.
Step4: BCNF
MOVIE is in BCNF because there is not a functional dependency where a nonprime attribute determines a key attribute.

SCREENING_ROOM

Room_number	Capacity	3D	Status

Step1: 1NF
SCREENING_ROOM is in 1NF because all attribute values are single atomic values.
Step2: 2NF
SCREENING_ROOM is in 2NF because its primary key contains a single attribute, there is no partial dependency.
Step3: 3NF

SCREENING_ROOM is in 3NF because no nonprime attribute is functionally determined by another nonprime attribute, that is, there is no transitive dependency of a nonprime attribute on the primary key.
Step4: BCNF
SCREENING_ROOM is in BCNF because there is not a functional dependency where a nonprime attribute determines a key attribute.

TICKET

Movie_title	Room_number	Screening_date	Screening_time	Seat_number	Admission	Price

Step1: 1NF
TICKET is in 1NF because all attribute values are single atomic values.
Step2: 2NF
TICKET is not in 2NF because of {Room_number, Screening_date, Screening_time}->Movie_Title. To make it in 2NF, we put it into 2 different relations.

TICKET1

Room_number	Screening_date	Screening_time	Seat_number	Admission	Price

TICKET_MOVIETITLE

Room_number	Screening_date	Screening_time	Movie_Title

Step3: 3NF
Both TICKET1 and TICKET_MOVIETITLE are in 3NF because there is no transitive dependency.
Step4: BCNF
Relation TICKET_MOVIETITLE, TICKET1 are in BCNF because there is no nonprime attribute determines any key attributes.

MENU

Food_name	Price	Stock

Step1: 1NF
MENU is in 1NF because all attribute values are single atomic values.
Step2: 2NF
MENU is in 2NF because its primary key contains a single attribute, there is no partial dependency.
Step3: 3NF
MENU is in 3NF because no nonprime attribute is functionally determined by another nonprime attribute, that is, there is no transitive dependency of a nonprime attribute on the primary key.
Step4: BCNF
MENU is in BCNF because there is not a functional dependency where a nonprime attribute determines a key attribute.

COUPON

Coupon_ID	Status	Value	Expiry_date	Member_ID

Step1: 1NF
COUPON is in 1NF because all attribute values are single atomic values.
Step2: 2NF
COUPON is in 2NF because its primary key contains a single attribute, there is no partial dependency.
Step3: 3NF
COUPON is in 3NF because no nonprime attribute is functionally determined by another nonprime attribute, that is, there is no transitive dependency of a nonprime attribute on the primary key.
Step4: BCNF
COUPON is in BCNF because there is not a functional dependency where a nonprime attribute determines a key attribute.

PURCHASE_FOOD

Food_name	Member_ID

PURCHASE_FOOD doesn't require normalisation because all attributes are key attributes.

SCREENED IN

Title	Room_number	Occupied_Time

Step1: 1NF
SCREENED_IN is in 1NF because all attribute values are single atomic values.
Step2: 2NF
SCREENED_IN is not in 2NF because Title can determine how long the room is occupied (movie length), which is partial dependency.
To get it into 2NF, SCREENED_IN can be put into 2 relations.

SCREENED_IN1

Title	Room_number

SCREENED_IN_TIME

Title	Occupied_Time

Step3: 3NF
SCREENED_IN1 and SCREENED_IN_TIME are in 3NF because there is no transitive dependency.
Step4: BCNF
SCREENED_IN1 and SCREENED_IN_TIME are in BCNF.

MEMBER_PHONE_NUMBER

Member_ID	Mphone_number

MEMBER_PHONE_NUMBER doesn't require normalisation because all attributes are key attributes.

EMPLOYEE_PHONE_NUMBER

IRD	Ephone_number

EMPLOYEE_PHONE_NUMBER doesn't require normalisation because all attributes are key attributes.

EMPLOYEE_POSITION

IRD	Position

EMPLOYEE_POSITION doesn't require normalisation because all attributes are key attributes.

MOVIE_GENRE

Title	Genre

MOVIE_GENRE doesn't require normalisation because all attributes are key attributes.

MOVIE_SCREENING_PERIOD

Title	Screening_period

MOVIE_SCREENING_PERIOD doesn't require normalisation because all attributes are key attributes.

PURCHASE_TICKET

Member_ID	IRD	Room_number	Screening_date	Screening_time	Seat_number

PURCHASE_TICKET doesn't require normalisation because all attributes are key attributes.

Load.sql

```
DROP TABLE purchase_food;
DROP TABLE screened_in1;
DROP TABLE screened_in_time;
DROP TABLE member_phone_number;
DROP TABLE employee_phone_number;
DROP TABLE employee_position;
DROP TABLE movie_genre;
DROP TABLE movie_screening_period;
DROP TABLE purchase_ticket;
DROP TABLE ticket_movietitle;
DROP TABLE member cascade constraints;
DROP TABLE movieemployee cascade constraints;
DROP TABLE movie cascade constraints;
DROP TABLE screening_room cascade constraints;
DROP TABLE ticket1 cascade constraints;
DROP TABLE menu cascade constraints;
DROP TABLE coupon;


CREATE TABLE member
	(fname		VARCHAR2(30)  NOT NULL,
	minit		CHAR,
	lname		VARCHAR2(30)  NOT NULL,
	member_id	CHAR(6)		PRIMARY KEY,
	bdate		DATE,
	sex		CHAR,
	address		VARCHAR2(30),
	points_earned	INT,
	points_spent	INT
);

INSERT INTO member VALUES
	('John', 'M', 'Smith', '000001', TO_DATE('01-01-01', 'DD-MM-YYYY'),
	'M', '1 Generic Grove', 100, 100
);
INSERT INTO member VALUES
	('Steve', 'N', 'Jensen', '000002', TO_DATE('02-02-02', 'DD-MM-YYYY'),
	'M', '2 Every Street', 57, 20
);
INSERT INTO member VALUES
	('Kate', 'O', 'Buscemi', '000003', TO_DATE('03-03-03', 'DD-MM-YYYY'),
	'F', '3 Some Road', 350, 1
);
```

```
COMMIT;


/*
Name had to be changed
as table 'employee' already
exists, due to work from
the labs
*/

CREATE TABLE movieemployee
        (efname   VARCHAR2(30)              NOT NULL,
        eminit   CHAR,
        elname   VARCHAR2(30)               NOT NULL,
        ird          CHAR(9)             PRIMARY KEY,
        ebdate   DATE,
        esex       CHAR,
        eaddress VARCHAR2(30),
        salary  INT
);

INSERT INTO movieemployee VALUES
        ('Jenny', 'K', 'Burg', '123456789', TO_DATE('17-06-1996', 'DD-MM-YYYY'),
        'F', '192 Castle Street', 25000
);
INSERT INTO movieemployee VALUES
        ('Tracey', 'S', 'Sandler', '987654321',
        TO_DATE('23-03-1989', 'DD-MM-YYYY'),
        'M', '17 Leith Street', 25001
);
INSERT INTO movieemployee VALUES
        ('Skittles', 'M', 'Woodhousen', '123454321',
        TO_DATE('12-08-1986', 'DD-MM-YYYY'),
        'M', '1 Slayer Ave', 35000
);
COMMIT;

CREATE TABLE movie
        (title       VARCHAR2(50)   PRIMARY KEY,
        length                NUMBER(8, 2),    --in minutes
        classification VARCHAR2(3)
);

INSERT INTO movie VALUES
        ('Electric Boogaloo', 189.20, 'PG');
INSERT INTO movie VALUES
        ('Electric Boogaloo 2', 189.20, 'PG');
INSERT INTO movie VALUES
        ('Electric Boogaloo 3', 189.20, 'PG');
COMMIT;
```

```
CREATE TABLE screening_room
	(room_number	VARCHAR2(4)	PRIMARY KEY,
	capacity		INT,
	threed			CHAR NOT NULL,  /*Y or N - also doesn't like
					numbers at the start of the name
					*/
	status			VARCHAR2(15)   /*screening/not in use
					/cleaning/etc*/
);

INSERT INTO screening_room VALUES
	('TH01', 50, 'Y', 'screening');
INSERT INTO screening_room VALUES
	('TH02', 50, 'N', 'not in use');
INSERT INTO screening_room VALUES
	('TH03', 50, 'N', 'cleaning');
COMMIT;

CREATE TABLE ticket1
	(room_number	VARCHAR2(4),
	screening_date	DATE,
	screening_time	TIMESTAMP,
	seat_number	VARCHAR2(4),
	admission	VARCHAR2(10),
	price		INT		NOT NULL,
	PRIMARY KEY(room_number, screening_date, screening_time, seat_number)
	);

INSERT INTO ticket1 VALUES
	('TH01', TO_DATE('22-04-2011', 'DD-MM-YYYY'),
	TO_TIMESTAMP('08:30', 'HH24:MI'),'F17', 'child', 10
	);
INSERT INTO ticket1 VALUES
	('TH02',TO_DATE('23-04-2011', 'DD-MM-YYYY'),
	TO_TIMESTAMP('14:30', 'HH24:MI'),'E18', 'senior', 10
	);
INSERT INTO ticket1 VALUES
	('TH03',TO_DATE('24-04-2011', 'DD-MM-YYYY'),
	TO_TIMESTAMP('23:30', 'HH24:MI'),'A05', 'adult', 10
	);

COMMIT;


CREATE TABLE ticket_movietitle
	(room_number	VARCHAR2(4),
	screening_date	DATE,
	screening_time	TIMESTAMP,
	movie_title	VARCHAR2(30)	NOT NULL,
```

```
        PRIMARY KEY(room_number, screening_date, screening_time)
        );

INSERT INTO ticket_movietitle VALUES
        ('TH01', TO_DATE('22-04-2011', 'DD-MM-YYYY'),
        TO_TIMESTAMP('08:30', 'HH24:MI'), 'Electric Boogaloo'
        );
INSERT INTO ticket_movietitle VALUES
        ('TH02',TO_DATE('23-04-2011', 'DD-MM-YYYY'),
        TO_TIMESTAMP('14:30', 'HH24:MI'), 'Electric Boogaloo 2'
        );
INSERT INTO ticket_movietitle VALUES
        ('TH03',TO_DATE('24-04-2011', 'DD-MM-YYYY'),
        TO_TIMESTAMP('23:30', 'HH24:MI'), 'Electric Boogaloo 3'
        );
COMMIT;


COMMIT;

CREATE TABLE menu
        (food_name      VARCHAR2(50)  PRIMARY KEY,
        price           NUMBER(38,2)  NOT NULL,
        stock           INT
);

INSERT INTO menu VALUES
        ('shitty cheese rolls', 15.45, 200);
INSERT INTO menu VALUES
        ('lesser shitty cheese rolls', 10.45, 100);
INSERT INTO menu VALUES
        ('greater shitty cheese rolls', 25.45, 300);
COMMIT;



CREATE TABLE coupon
        (coupon_id      VARCHAR2(10)  PRIMARY KEY,
        status          VARCHAR2(10),
        value           NUMBER(6,2)   CHECK(value >=10) --minimum coupon value
        expiry_date     DATE,
        member_id       CHAR(6)       REFERENCES member(member_id)
);
INSERT INTO coupon VALUES
        ('A00001', 'active', 10, TO_DATE('19-08-2017', 'DD-MM-YYYY'), '000001');
INSERT INTO coupon VALUES
        ('A00002', 'active', 20, TO_DATE('20-09-2017', 'DD-MM-YYYY'), '000002');
INSERT INTO coupon VALUES
        ('A00003', 'active', 30, TO_DATE('21-10-2017', 'DD-MM-YYYY'), '000003');
COMMIT;
```

```
CREATE TABLE purchase_food
	(food_name VARCHAR2(30)  REFERENCES menu(food_name),
	member_id CHAR(6)   REFERENCES member(member_id),
	PRIMARY KEY(food_name, member_id)
	);

INSERT INTO purchase_food VALUES ('shitty cheese rolls', '000001');
INSERT INTO purchase_food VALUES ('lesser shitty cheese rolls', '000002');
INSERT INTO purchase_food VALUES ('greater shitty cheese rolls', '000003');
COMMIT;

CREATE TABLE screened_in1
	(title	VARCHAR2(30)   REFERENCES movie(title),
	room_number   VARCHAR2(4)	REFERENCES screening_room(room_number),
	PRIMARY KEY(title, room_number)
	);

INSERT INTO screened_in1 VALUES('Electric Boogaloo', 'TH01');
INSERT INTO screened_in1 VALUES('Electric Boogaloo 2', 'TH02');
INSERT INTO screened_in1 VALUES('Electric Boogaloo 3', 'TH03');
COMMIT;

CREATE TABLE screened_in_time
	(title	VARCHAR2(30)   REFERENCES movie(title),
	occupied_time INT, --in minutes
	PRIMARY KEY(title)
	);

INSERT INTO screened_in_time VALUES
	('Electric Boogaloo',100);

INSERT INTO screened_in_time VALUES
	('Electric Boogaloo 2',120);
INSERT INTO screened_in_time VALUES
	('Electric Boogaloo 3',140);
COMMIT;



CREATE TABLE member_phone_number
	(member_id	CHAR(6)		REFERENCES member(member_id),
	mphone_number   VARCHAR2(10),
	PRIMARY KEY(member_id, mphone_number)
	);

INSERT INTO member_phone_number VALUES('000001','0800838383');
INSERT INTO member_phone_number VALUES('000002','0800838383');
INSERT INTO member_phone_number VALUES('000003','0800838383');
COMMIT;
```

```
CREATE TABLE employee_phone_number
	(ird		CHAR(9)		REFERENCES movieemployee(ird),
	ephone_number  VARCHAR2(10),
	PRIMARY KEY(ird, ephone_number)
	);

INSERT INTO employee_phone_number VALUES('123456789', '0800323232');
INSERT INTO employee_phone_number VALUES('123454321', '0800323232');
INSERT INTO employee_phone_number VALUES('987654321', '0800323232');
COMMIT;


CREATE TABLE employee_position
	(ird		CHAR(9)		REFERENCES movieemployee(ird),
	position   VARCHAR2(20),
	PRIMARY KEY(ird, position)
	);

INSERT INTO employee_position VALUES('123456789', 'big boss man');
INSERT INTO employee_position VALUES('123454321', 'popcorn handler');
INSERT INTO employee_position VALUES('987654321', 'mean snack man');
COMMIT;


CREATE TABLE movie_genre
	(title  VARCHAR2(30) REFERENCES movie(title),
	genre VARCHAR2(10),
	PRIMARY KEY(title, genre)
	);

INSERT INTO movie_genre VALUES('Electric Boogaloo', 'Sci-Fi');
INSERT INTO movie_genre VALUES('Electric Boogaloo 2', 'Comedy');
INSERT INTO movie_genre VALUES('Electric Boogaloo 3', 'Horror');
COMMIT;

CREATE TABLE movie_screening_period
	(title			VARCHAR2(50)   REFERENCES movie(title),
	screening_period_start	DATE,/*Can't have a range, must have a start
				and end*/
	screening_period_end	DATE,
	PRIMARY KEY(title, screening_period_start, screening_period_end)
	);

INSERT INTO movie_screening_period VALUES
	('Electric Boogaloo',
	TO_DATE('07-01-17', 'DD-MM-YYYY'),
	TO_DATE('07-02-17', 'DD-MM-YYYY')
	);
```

```
INSERT INTO movie_screening_period VALUES
	('Electric Boogaloo 2',
	TO_DATE('08-03-17', 'DD-MM-YYYY'),
	TO_DATE('08-04-17', 'DD-MM-YYYY')
	);
INSERT INTO movie_screening_period VALUES
	('Electric Boogaloo 3',
	TO_DATE('09-05-17', 'DD-MM-YYYY'),
	TO_DATE('09-06-17', 'DD-MM-YYYY')
	);
COMMIT;

CREATE TABLE purchase_ticket
	(member_id	CHAR(6)		REFERENCES member(member_id),
	ird		CHAR(9)		REFERENCES movieemployee(ird),
	room_number	VARCHAR2(4),
	screening_date	DATE,
	screening_time	TIMESTAMP,
	seat_number	VARCHAR2(4),
	FOREIGN KEY (room_number, screening_date, screening_time, seat_number)
	REFERENCES
	ticket1(room_number, screening_date, screening_time, seat_number),
	PRIMARY KEY
	(member_id, ird, room_number, screening_date, screening_time, seat_number)
	);

INSERT INTO purchase_ticket VALUES
	('000001','123456789','TH01', TO_DATE('22-04-2011', 'DD-MM-YYYY'),
	TO_TIMESTAMP('08:30', 'HH24:MI'),'F17'
	);
INSERT INTO purchase_ticket VALUES
	('000002','123456789','TH02', TO_DATE('23-04-2011', 'DD-MM-YYYY'),
	TO_TIMESTAMP('14:30', 'HH24:MI'),'E18'
	);
INSERT INTO purchase_ticket VALUES
	('000003','987654321','TH03', TO_DATE('24-04-2011', 'DD-MM-YYYY'),
	TO_TIMESTAMP('23:30', 'HH24:MI'),'A05'
	);
COMMIT;
```

Teamwork Summary

The initial task of fixing up errors from the first assignment was done collaboratively, and Julia revised the ER diagram accordingly. Each part of this assignment was completed mostly by a separate group member, with input from other members on difficult or opinion based parts. Julia focused on the normalisation, Callum on the sql database, Caleb on the relation schema. Jessy wasn't sufficiently notified of our first meeting so wasn't present, but helped out during the second meeting. The date and time of the third and final meeting was agreed by all members for 11am Wednesday the 23rd which Jessy was absent from. She followed this up with an explanation on Friday saying she misread the email.