University of British Columbia, Vancouver

Department of Computer Science

CPSC 304 Project Cover Page

Milestone #:___2

Date:_25, October 2021

Group Number:_____77

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Wenkai Cai	90787698	q6d3d	wenkai101cai@gmail.com
Heng Wu	31696198	v4b4c	black14369@gmail.com
Michelle Chen	65406415	u0f7w	hsiangyi1025@gmail.com

Changes from M1:

1. Corrected Meal to Contains so that Meal can be associated with multiple orders instead of one.

2. The arrow from customer to Place, this change allows customers to place multiple orders instead of only one.

3. We decided to switch the weak entity from staff to workstation, since workstation can only belong to a store, now the workstation can be identified by both (StoreId, WorkstationName).

Staff now have a unique id and can only work in exactly one store.

4. Changed the workstation weak entity key from name to ID, and now it has an additional attribute of name.

5. We found out the many to one relation between Meal and workstation is ambiguous, the original case a meal can only be made by one store and workstation, which means the meal exists in only one store. So we changed it to many to many with total participation with Meals.

6. Added additional attributes to membership to normalize when violating 3NF or BCNF.

- Type: Types of Memberships offered to customers.
- Number: A number must match a type to identify a member, which means each type can have the same number.
- Level: Each type of membership has a set of levels.
- Expense: Record the total money spent.
- Points: Record the total points earned.
- Discount: Discounts offered for the type and level of membership.

7. Changed Store entity's attributes. Removed StoreID and changed primary key to StreetAddress, City, Province. Added PostalCode, LocalDiscount.

Store
Store(StoreName: Char, StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char, StorePostalCode: Char, LocalDiscount: Char)
FDs:
StoreStreetAddress, StoreCity, StoreProvince → StoreName, StorePostalCode, LocalDiscount
StoreCity, StorePostalCode → StoreProvince
StoreCity, StoreProvince → LocalDiscount

Primary Key: (StoreName: Char, StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char)
Candidate Key: (StoreName: Char, StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char, StorePostalCode: Char)

Normalize Store
Step 1: Put FDs in standard form (have only one attribute on RHS)
StoreStreetAddress, StoreCity, StoreProvince → StorerName,
StoreStreetAddress, StoreCity, StoreProvince → StorePostalCode,
StoreStreetAddress, StoreCity, StoreProvince → LocalDiscount,
StoreCity, StorePostalCode → StoreProvince
StoreCity, StoreProvince → LocalDiscount

Step2: Minimize LHS of each FD
Nothing to do with this step

Step3: Delete Redundant FDs
Nothing to do with this step

Final answer:
StoreStreetAddress, StoreCity, StoreProvince → StorerName,
StoreStreetAddress, StoreCity, StoreProvince → StorePostalCode,
StoreStreetAddress, StoreCity, StoreProvince → LocalDiscount,
StoreCity, StorePostalCode → StoreProvince
StoreCity, StoreProvince → LocalDiscount

StoreCity, StorePostalCode → StoreProvince, violates BCNF in Store, so decompose Store1(StoreCity, StorePostalCode, StoreProvince), Store2(StoreName, StoreStreetAddress, StoreCity, StorePostalCode, LocalDiscount)

StoreCity, StorePostalCode → LocalDiscount, violates BCNF in Store2, so decompose Store3(StoreCity, StorePostalCode, LocalDiscount), Store4(StoreName, StoreStreetAddress, StoreCity, StorePostalCode)

Final answer:
Store1(StoreCity: Char, StorePostalCode: Int, StoreProvince: Char), Store3(StoreCity: Char, StorePostalCode: Char, LocalDiscount: Char), Store4(StoreStreetAddress: Char, StoreCity: Char, StorePostalCode: Char, StoreName: Char)

```
CREATE TABLE Store1 {
        storeCity               CHAR(40),
        storePostalCode         CHAR(40),
        storeProvince           CHAR(40),
        PRIMARY KEY(storeCity, storePostalCode)
};
```

storeCity	storePostalCode	storeProvince
Vancouver	V3H 1V6	BC
Vancouver	H7Y 7E4	BC
Calgary	O9U 4R2	AB
Calgary	X4I 8N5	AB
Burnaby	G4R 7U8	BC

```
CREATE TABLE Store3 {
        storeCity               CHAR(40),
        storePostalCode         CHAR(40),
        localDiscount           CHAR(40),
        PRIMARY KEY(storeCity, storePostalCode)
};
```

storeCity	storePostalCode	localDiscount
Burnaby	G6Y 8Z6	5%
Burnaby	O9Z 5M7	5%
Vancouver	U7F 4T0	5%
Toronto	A9B 1F1	7%
Calgary	M0B 6H6	10%

```
CREATE TABLE Store4 {
        storeCity               CHAR(40),
        storePostalCode         CHAR(40),
        storeStreetAddress      CHAR(80),
        storeName               CHAR(40),
        PRIMARY KEY(storeCity, storePostalCode)
};
```

storeCity	storePostalCode	storeStreetAddress	storeName
Vancouver	F7N 9A6	1234 W2 Ave	Store2
Vancouver	F1Z 9U9	3456 W7 Ave	Store4
Burnaby	J1J 7Y7	2234 KingsWay St	Store7
Toronto	Y2F 5C9	4122 Harvord St	Store99
Calgary	H9Z 6G1	5642 Kiki Ave	Store2

Workstation
Workstation(**StoreStreetAddress**: Char, **StoreCity**: Char, **StoreProvince**: Char, WorkstationID: Int, WorkstationName: Char) (StoreStreetAddress, StoreCity, StoreProvince cannot be null)
FDs:
StoreStreetAddress, StoreCity, StoreProvince, WorkstationID → WorkstationName
Trivial:
StoreStreetAddress, StoreCity, StoreProvince → StoreName, StorePostalCode, LocalDiscount

Primary Key: (StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char, WorkstationID: Int)
Candidate Key: (StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char, WorkstationID: Int)

Normalize Workstation
StoreStreetAddress, StoreCity, StoreProvince, WorkstationID → WorkstationName
In BCNF, there is nothing to do.

```
CREATE TABLE WorkStation {
        storeStreetAddress    CHAR(80)          NOT NULL,
        storeCity             CHAR(40)          NOT NULL,
        storeProvince         CHAR(40)          NOT NULL,
        workStationID         INT,
        workStationName       CHAR(40),
        PRIMARY KEY (storeStreetAddress, storeCity, storeProvince, WorkstationID),
        FOREIGN KEY (storeStreetAddress, storeCity, storeProvince) REFERENCES Store
ON DELETE CASCADE
};
```

storeStreetAddress	storeCity	storeProvince	workStationID	workStationName
1831 W37 Ave	Vancouver	BC	01	Station1
1831 W37 Ave	Vancouver	BC	02	Station2
1223 Abord Ave	Calgary	AB	01	Station1
1234 W16 Ave	Vancouver	BC	01	Station5
1221 Asot St	Toronto	ON	02	Station6

Meal
Meal(MealName: Char, MealPrice: Int)
FDs:
MealName → MealPrice

Primary Key: Meal(MealName: Char)
Candidate Key: Meal(MealName: Char)

Normalize Meal
MealName → MealPrice
In BCNF, there is nothing to do.

```
CREATE TABLE Meal {
        mealName        CHAR(40)    PRIMARY KEY,
        mealPrice       INT
};
```

mealName	mealPrice
spicy chicken	12
bbq chicken	10
bbq chicken	10
Large Fries	10
small diet coke	4

FriedChicken
FriedChicken(MealName: Char, MealPrice: Int, FriedChickenFlavour: Char)
FDs:
MealName → MealPrice
MealName → FriedChickenFlavour

Primary Key: Meal(MealName: Char)
Candidate Key: Meal(MealName: Char)

Normalize FriedChicken
MealName → MealPrice
MealName → FriedChickenFlavour
In BCNF, there is nothing to do.

```
CREATE TABLE FriedChicken {
        mealName            CHAR(40)      PIMARY KEY,
        mealPrice           INT,
        friedChickenFlavour CHAR(40)
};
```

mealName	mealPrice	friedChickenFlavour
spicy chicken	12	spicy
bbq chicken	10	bbq
bbq chicken	10	bbq
butter milk chicken	15	butter milk
buffalo chicken	8	buffalo

Drink
Drink(MealName: Char, MealPrice: Int, DrinkFlavour: Char, DrinkSize: Char)
FDs:
MealName → MealPrice
MealName → DrinkFlavour, DrinkSize

Primary Key: Meal(MealName: Char)
Candidate Key: Meal(MealName: Char)

Normalize Drink
MealName → MealPrice
MealName → DrinkFlavour, DrinkSize
In BCNF, there is nothing to do.

```
CREATE TABLE Drink {
        mealName        CHAR(40)      PIMARY KEY,
        mealPrice       INT,
        drinkFlavour    CHAR(40),
        drinkSize       CHAR(40)
};
```

mealName	mealPrice	drinkFlavour	drinkSize
large coke	5	coke	large
medium coke	4	coke	medium
medium coffee	4	coffee	medium
large sprite	5	sprite	large
large iced tea	5	iced tea	large

Fries
Fries(MealName: Char, MealPrice: Int, FriesSize: Char)
FDs:
MealName → FriesSize

Primary Key: Meal(MealName: Char)
Candidate Key: Meal(MealName: Char)

Normalize Fries
MealName → MealPrice
MealName → FriesSize
In BCNF, there is nothing to do.

```
CREATE TABLE Fries {
        mealName        CHAR(40)    PIMARY KEY,
        mealPrice       INT,
        friesSize       CHAR(40)
};
```

mealName	mealPrice	friesSize
large fries	10	large
medium fries	8	medium
medium fries	8	medium
small fries	6	samll
samll fries	6	samll

Order
Order(**Phone**: Int, OrderNumber: Int) (Phone cannot be null)
FDs:
OrderNumber → Phone
Trivial:
Phone → CustomerName, MemberName, MemberNumber, CustomerName

Primary Key: Order(OrderNumber: Int)
Candidate Key: Order(OrderNumber: Int)

Normalize Order
OrderNumber → Phone
In BCNF, there is nothing to do.

```
CREATE TABLE Order {
        orderNumber         INT         PRIMARY KEY,
        phone               INT         UNIQUE,
        FOREIGN KEY (phone) REFERENCES Customer ON DELETE CASCADE
};
```

orderNumber	phone
18	6041234567
19	6042345678
33	6043456789
44	2481234567
45	2482345678

Customer
Customer(**MemberName**: Char, **MemberNumber**: Int, Phone: Int, CustomerName: Char)
FDs:
Phone → CustomerName, MemberName, MemberNumber, CustomerName
Trivial:
MemberName, MemberNumber → MemberType, MemberLevel, MemberExpense, MemberPoints, MemberDiscount

Primary Key: Customer(Phone: Int)
Candidate Key: Customer(Phone: Int)

Normalize Customer
Phone → CustomerName, MemberName, MemberNumber, CustomerName
In BCNF, there is nothing to do.

```
CREATE TABLE Customer {
        phone               INT             PRIMARY KEY,
        memberName          CHAR(40),
        memberNumber        INT,
        customerName        CHAR(40),
        FOREIGN KEY (memberName, memberNumber) REFERENCES Membership
};
```

phone	memberName	memberNumber	customerName
6041234567	Mars Wang	12345	Mars Wang
6042345678	Wenkai Cai	23456	Wenkai Cai
6043456789	Dwayne Johnson	76852	Dwayne Johnson
2481234567	Maddie Ziegler	56789	Maddie Ziegler
2482345678	Heng Wu	24689	Heng Wu

Membership
Membership(MemberName: Char, MemberNumber: Int, MemberType: Char, MemberLevel: Int, MemberExpense: Int, MemberPoints: Int, MemberDiscount: Char)
FDs:
MemberName, MemberNumber → MemberType, MemberLevel, MemberExpense, MemberPoints, MemberDiscount
MemberType, MemberLevel → MemberDiscount
MemberType, MemberExpense → MemberPoints
MemberPoints → MemberLevel

Primary Key: Membership(MemberName: Char, MemberNumber: Int)
Candidate Key: Membership(MemberName: Char, MemberNumber: Int, MemberType: Char, MemberLevel: Int, MemberExpense: Int, MemberPoints: Int)

Normalize MemberShip
Step 1: Put FDs in standard form (have only one attribute on RHS)
MemberName, MemberNumber → MemberType,
MemberName, MemberNumber → MemberLevel,
MemberName, MemberNumber → MemberExpense,
MemberName, MemberNumber → MemberPoints,
MemberName, MemberNumber → MemberDiscount,
MemberType, MemberLevel → MemberDiscount,
MemberType, MemberExpense → MemberPoints
MemberPoints → MemberLevel

Step2: Minimize LHS of each FD
Nothing to do with this step

Step3: Delete Redundant FDs
Nothing to do with this step

Final answer:
MemberName, MemberNumber → MemberType,
MemberName, MemberNumber → MemberLevel,
MemberName, MemberNumber → MemberExpense,
MemberName, MemberNumber → MemberPoints,
MemberName, MemberNumber → MemberDiscount,
MemberType, MemberLevel → MemberDiscount,
MemberType, MemberExpense → MemberPoints
MemberPoints → MemberLevel

MemberType, MemberLevel → MemberDiscount, violates 3NF in Membership, so decompose Member1(MemberType, MemberLevel, MemberDiscount),

Member2(MemberName, MemberNumber, MemberType, MemberLevel, MemberExpense, MemberPoints)

MemberType, MemberExpense → MemberPoints, violates 3NF in Membership, so decompose Member2(MemberName, MemberNumber, MemberType, MemberLevel, MemberExpense, MemberPoints)
We get Member3(MemberType, MemberExpense, MemberPoints), Member4(MemberName, MemberNumber, MemberType, MemberLevel, MemberExpense)

We lost FD: MemberPoints → MemberLevel, so add Member5(MemberPoints, MemberLevel)

Final answer:
Member1(MemberType: Char, MemberLevel: Int, MemberDiscount: Char),
Member3(MemberType: Char, MemberExpense: Int, MemberPoints: Int),
Member4(MemberName: Char, MemberNumber: Int, MemberType: Char, MemberLevel: Int, MemberExpense: Int),
Member5(MemberPoints: Int, MemberLevel: Int)

```
CREATE TABLE Member1 {
        memberType          CHAR(40),
        memberLevel         INT,
        memberDiscount      CHAR(40),
        PRIMARY KEY(memberType, memberLevel)
};
```

memberType	memberLevel	memberDiscount
A	1	6%off
A	2	7%off
C	3	10%off
B	1	3%off
B	2	4%off

```
CREATE TABLE Member3 {
        memberType          CHAR(40),
        memberExpense       INT,
        memberPoints        INT,
        PRIMARY KEY(memberType, memberExpense)
};
```

memberType	memberExpense	memberPoints
A	500	1000
A	400	800
A	300	600
B	500	500
B	400	400

```
CREATE TABLE Member4 {
        memberName          CHAR(40),
        memberNumber        INT,
        memberType          CHAR(40),
        memberLevel         INT,
        memberExpense       INT,
        PRIMARY KEY(memberName, memberNumber)
};
```

memberName	memberNumber	memberType	memberLevel	memberExpense
Mars Wang	12345	A	2	500
Mars Wang	23456	B	1	500
Dwayne Johnson	76852	A	1	300
Maddie Ziegler	56789	A	3	750
Heng Wu	24689	B	2	750

```
CREATE TABLE Member5 {
        memberPoints        INT,        PRIMARY KEY,
        memberLevel         INT,
};
```

memberPoints	memberLevel
500	2
500	2
300	1
750	3
1000	4

Staff

Staff(StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char, StaffID: Int, StaffName: Char) (StoreStreetAddress, StoreCity, StoreProvince cannot be null)

FDs:

StaffID → StoreStreetAddress, StoreCity, StoreProvince, StaffName

Primary Key: Staff(StaffID: Int)

Candidate Key: Staff(StaffID: Int)

Normalize Staff

StaffID → StoreStreetAddress, StoreCity, StoreProvince, StaffName

In BCNF, there is nothing to do.

```
CREATE TABLE Staff {
        staffID             INT             PRIMARY KEY,
        storeStreetAddress  CHAR(80)        NOT NULL,
        storeCity           CHAR(40)        NOT NULL,
        storeProvince       CHAR(40)        NOT NULL,
        staffName           CHAR(40),
        FOREIGN KEY (storeStreetAddress, storeCity, storeProvince) REFERENCES Store
ON DELETE CASCADE
};
```

staffID	storeStreetAddress	storeCity	storeProvince	staffName
807812	4014 16th Ave	Vancouver	BC	Wenkai Hu
807813	4014 16th Ave	Vancouver	BC	Andy Yu
807814	4014 16th Ave	Vancouver	BC	Lei Wang
807820	4014 16th Ave	Vancouver	BC	Zhu Yu
453762	220 George Street	Calgary	AB	Sandy Wu

SendsMealTo
SendsMealTo(**StaffID**: Int, **StoreStreetAddress**: Char, **StoreCity**: Char, **StoreProvince**: Char, **WorkstationID**: Int) (StaffId, StoreStreetAddress, StoreCity, StoreProvince, WorkstationID cannot be null)
FDs:
All trivial
StaffID → StoreStreetAddress, StoreCity, StoreProvince, StaffName
StoreStreetAddress, StoreCity, StoreProvince, WorkstationID → WorkstationName

Primary Key: SendsMealTo(StaffID: Int, StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char, WorkstationID: Int)
Candidate Key: SendsMealTo(StaffID: Int, StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char, WorkstationID: Int)

Normalize SendsMealTo
This is a relationship so all FDs are trivial so it doesn't violate BCNF.

```
CREATE TABLE SendsMealTo {
        staffID                 INT,
        storeStreetAddress      CHAR(80),
        storeCity               CHAR(40),
        storeProvince           CHAR(40),
        workStationID           INT,
        PRIMARY KEY (staffID, storeStreetAddress, storeCity, storeProvince, workstationID),
        FOREIGN KEY  (staffID) REFERENCES Staff,
        FOREIGN KEY  (storeStreetAddress, storeCity, storeProvince, WorkstationID)
REFERENCES Workstation
};
```

staffID	storeStreetAddress	storeCity	storeProvince	workStationID
807882	1014 Willingdon Ave	Burnaby	BC	02
807812	4014 16th Ave	Vancouver	BC	02
807814	4014 16th Ave	Vancouver	BC	02
807820	4014 16th Ave	Vancouver	BC	03
453762	220 George Street	Calgary	AB	03

Contains
Contains(**OrderNumber**: Int, **MealName**: Char) (OrderNumber cannot be null)
FDs:
All trivial
OrderNumber → Phone
MealName → MealPrice

Primary Key: Contains(OrderNumber: Int, MealName: Char)
Candidate Key: Contains(OrderNumber: Int, MealName: Char)

Normalize Contains
This is a relationship so all FDs are trivial so it doesn't violate BCNF.

```
CREATE TABLE Contains {
        orderNumber         INT,
        mealName            CHAR(40),
        PRIMARY KEY (orderNumber, mealName),
        FOREIGN KEY  (orderNumber) REFERENCES Order,
        FOREIGN KEY  (mealName) REFERENCES Meal
};
```

orderNumber	mealName
23	large Coke
24	medium fries
26	medium fries
41	large fries
41	samll fries

Serves
Serves(**StaffID**: Int, **Phone**: Int)
FDs:
All trivial
StaffID → StoreStreetAddress, StoreCity, StoreProvince, StaffName
Phone → CustomerName, MemberName, MemberNumber, CustomerName

Primary Key: Serves(StaffID: Int, Phone: Int)
Candidate Key: Serves(StaffID: Int, Phone: Int)

Normalize Serves
This is a relationship so all FDs are trivial so it doesn't violate BCNF.

```
CREATE TABLE Serves {
        staffID                 INT,
        phone                   INT,
        PRIMARY KEY (staffID, Phone),
        FOREIGN KEY  (staffID) REFERENCES Staff,
        FOREIGN KEY  (phone) REFERENCES Customer
};
```

staffID	phone
807812	6041236548
807812	7784228456
807814	6048825673
807820	6048825673
453762	7783453911

CookBy
CookBy(**StoreStreetAddress**: Char, **StoreCity**: Char, **StoreProvince**: Char, **WorkstationID**: Int, **MealName**: Char) (MealName cannot be null)
FDs:
All trivial
StoreStreetAddress, StoreCity, StoreProvince → StoreName, StorePostalCode, LocalDiscount
MealName → MealPrice

Primary Key: CookBy(StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char, WorkstationID: Int, MealName: Char)
Candidate Key: CookBy(StoreStreetAddress: Char, StoreCity: Char, StoreProvince: Char, WorkstationID: Int, MealName: Char)

Normalize CookBy
This is a relationship so all FDs are trivial so it doesn't violate BCNF.

```
CREATE TABLE CookBy {
        storeStreetAddress    CHAR(80),
        storeCity             CHAR(40),
        storeProvince         CHAR(40),
        workStationID         INT,
        mealName              INT,
        PRIMARY KEY (storeStreetAddress, storeCity, storeProvince, sorkstationID, mealID),
        FOREIGN KEY  (storeStreetAddress, storeCity, storeProvince, workstationID)
REFERENCES Workstation,
        FOREIGN KEY  (mealID) REFERENCES Meal
};
```

storeStreetAddress	storeCity	storeProvince	workStationID	mealName
2222 W41 Ave	Vancouver	BC	1	Large Coke
2222 W41 Ave	Vancouver	BC	1	Medium Fries
2234 W11 Ave	Vancouver	BC	1	Spicy Chicken
2234 W11 Ave	Vancouver	BC	2	Medium Coffee
3241 Dawn St	Toronto	ON	2	Small Juice

Insertion: Insert a Meal to the Meal table.
Insertion: Insert a Membership to Membership table.
Insertion: Insert a Store to the Store table and Staff table.

Deletion: Delete a store from Store table, workstation is a weak entity on store so on delete cascade also deletes workstation. Since Staff has the foreign key from store, this also affects and deletes the information related to store on the Staff table.
Deletion: Delete a customer from customer table, since order has the on delete cascade relationship with customer, so this will also deletes the information related to customer on the Order table