CPSC 304 Project Cover Page

Milestone #: 2

Date: July 26, 2023

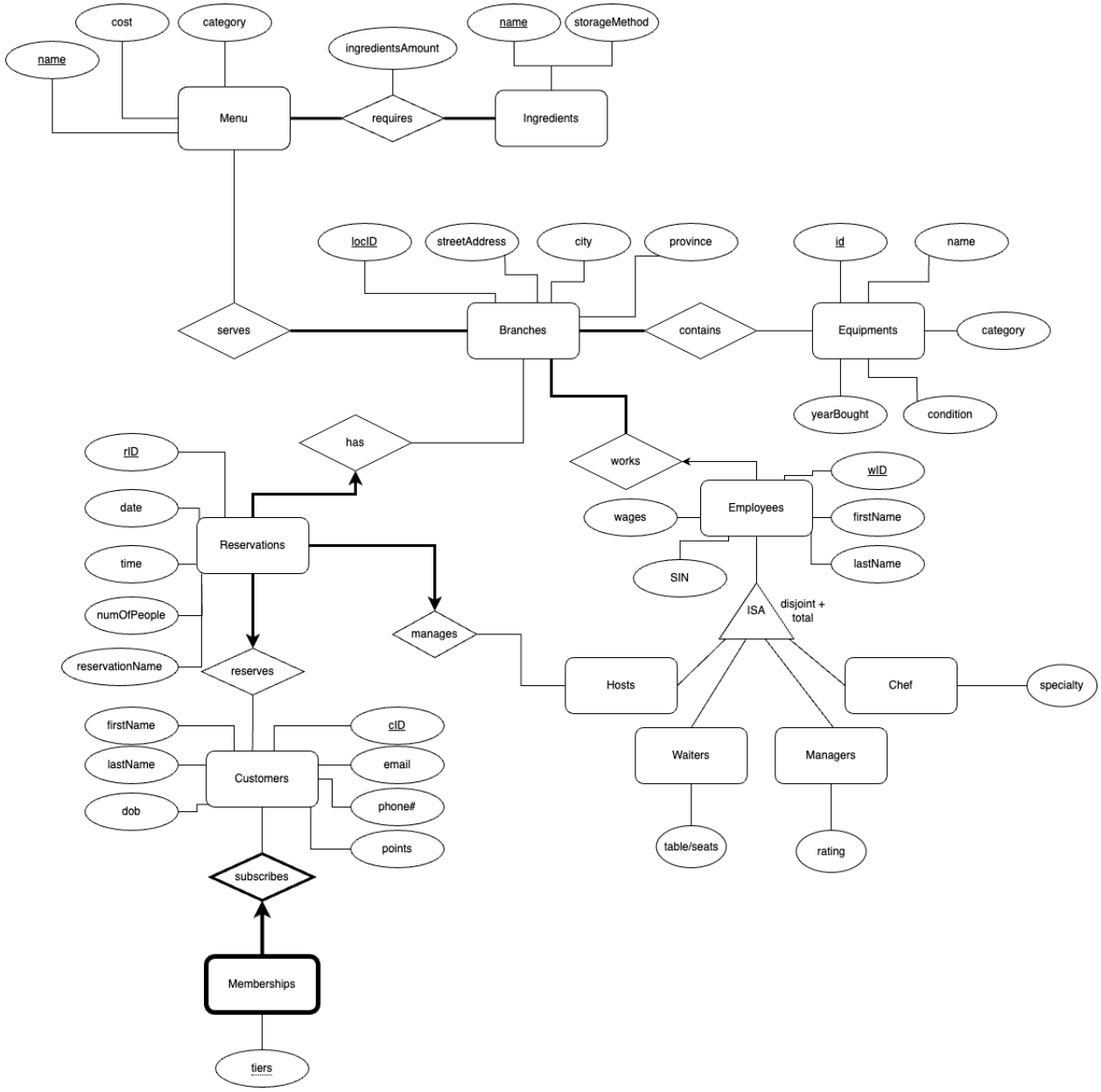
Group Number: 7

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Julian Widjaja	73873796	i0k7n	julian.widjaja02@gmail.com
Hannah Shiang	47799820	p1l1e	hanhanshiang6@gmail.com
Michealea Faustine	44619948	r1l70	michealeafaustin@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

The ER Diagram



dob = date of birth

Changes:

1. deleted attribute ‘amount’ from Ingredients and created descriptive attribute ‘ingredientsAmount’ to requires
    - We realized that it doesn’t make sense if every menu items use the same amount of a specific ingredients. Hence, we moved it to a descriptive attribute in the

'requires' relationship so every menu items can specify the amount needed for an ingredient

2. renamed attribute 'age' to 'yearBought' in the Equipments entity
    - We realized that we need to either manually update the age every year or automatically change the age every year for the attribute 'age'. However, for now we think that it will be easier for us to store the year when the equipment was bought so we don't need to update it occasionally.
3. renamed attribute 'name' to 'reservationName' in the Reservations entity
    - We realized that 'name' is little bit to general so we want to renamed it to a more descriptive name
4. renamed 'Waiters / Waitress' entity to 'Waiters'
    - We just want to shorten the name of the entity and avoid using any symbols
5. renamed attribute 'table/seats' to 'tableSection' in the Waiters entity
    - 'table/seats' gives quite a vague description so we changed it to a more descriptive name

The schema derived from our ER diagram

Menu(<u>name</u>: char(50), cost: float, category: char(50))

Ingredients(<u>name</u>: char(50), storageMethod: char(50))

IngredientsRequired(**<u>menuName</u>**: char(50) , **<u>ingredientsName</u>**: char(50), ingredientsAmount: int)

Customers(<u>cID</u>: int, email: char(50), phoneNum: char(10), points: int, firstName: char(50), lastName:char(50), dob: date) (email cannot be null and must be unique; phoneNum cannot be null and must be unique; points cannot be null; candidate keys: email, phoneNum)

MembershipsSubscribes(**<u>cID</u>**: int, <u>tiers</u>:char(50))

Branches(<u>locID</u>: int , streetAddress: char(50), city: char(50), province: char(50))

MenuServed(**<u>menuName</u>**: char(50) , **<u>locID</u>**: int)

Equipments(<u>id</u>: int, name: char(50), category: char(50), condition: char(50), yearBought: int)

EquipmentContained (**<u>locID</u>** : int, **<u>equipID</u>** : int)

Hosts (<u>wID</u> : int, firstName: char(50), lastName: char(50), wages: float, SIN: char(9), **locID**: int) (SIN cannot be null and must be unique; candidate keys: SIN)

Waiters (wID : int, firstName: char(50) , lastName: char(50), wages: float, SIN: char(9), tableSection: char(50), **locID**: int) (SIN cannot be null and must be unique; candidate keys: SIN)

Managers (wID : int, firstName: char(50), lastName: char(50), wages: float, SIN: char(9), rating: float, **locID**: int) (SIN cannot be null and must be unique; candidate keys: SIN)

Chef (wID : int, firstName: char(50), lastName: char(50), wages: float, SIN: char(9), speciality: char(50), **locID**: int) (SIN cannot be null and must be unique; candidate keys: SIN)

Reservations(rID: int , **cID**: int**, locID**: int **, wID**: int **,** date: date, time: time, numOfPeople: int, reservationName: char(50)) (cID must not be null, locID must not be null, wID must not be null)

Note:
Underlined attributes are the primary keys
Bolded attributes are the foreign keys

Functional Dependencies (FDs)

Menu:
name -> cost
name -> category

Ingredient:
name -> storageMethod

IngredientsRequired:
menuName, ingredientsName -> ingredientsAmount

Reservations:
rID-> date
rID- > time
rID -> numOfPeople
rID -> reservationName
rID -> cID
rID -> locID
rID -> wID

Customers:
cID -> email
cID -> phoneNum
cID -> firstName
cID -> lastName

cID -> points
cID -> dob
email -> cID
email -> phoneNum
email -> firstName
email -> lastName
email -> points
email -> dob
phoneNum -> cID
phoneNum -> email
phoneNum -> firstName
phoneNum -> lastName
phoneNum -> points
phoneNum -> dob

(candidate keys: email, phoneNum)

Branches:
locID -> streetAddress
locID -> city
locID -> province

Hosts:
wID -> firstName
wID -> lastName
wID -> wages
wID -> SIN
wID -> locID

Waiters:
wID -> firstName
wID -> lastName
wID -> wages
wID -> SIN
wID -> locID
wID -> tableSection
SIN -> wID
SIN -> firstName
SIN -> lastName
SIN -> wages
SIN -> locID
SIN -> tableSection
tableSection -> wages

(candidate key: SIN)

Managers:
wID -> firstName
wID -> lastName
wID -> wages
wID -> SIN
wID -> locID
wID -> rating
SIN -> wID
SIN -> firstName
SIN -> lastName
SIN -> wages
SIN -> locID
SIN -> rating

(candidate key: SIN)

Chef:
wID -> firstName
wID -> lastName
wID -> wages
wID -> SIN
wID -> locID
wID -> speciality
SIN -> wID
SIN -> firstName
SIN -> lastName
SIN -> wages
SIN -> locID
SIN -> speciality
speciality -> wages

(candidate key: SIN)

Equipments:
id -> name
id -> category
id -> condition
id -> yearBought
name -> category

Normalization

Menu(name: char(50), cost: float, category: char(50))

Ingredients(name: char(50), storageMethod: char(50))

IngredientsRequired(**menuName**: char(50) , **ingredientsName**: char(50), ingredientsAmount: int)

Customers(cID: int, email: char(50), phoneNum: char(10), points: int, firstName: char(50), lastName:char(50), dob: date) (email cannot be null and must be unique; phoneNum cannot be null and must be unique; points cannot be null; candidate keys: email, phoneNum)

MembershipsSubscribes(**cID**: int, tiers:char(50))

Branches(locID: int , streetAddress: char(50), city: char(50), province: char(50))

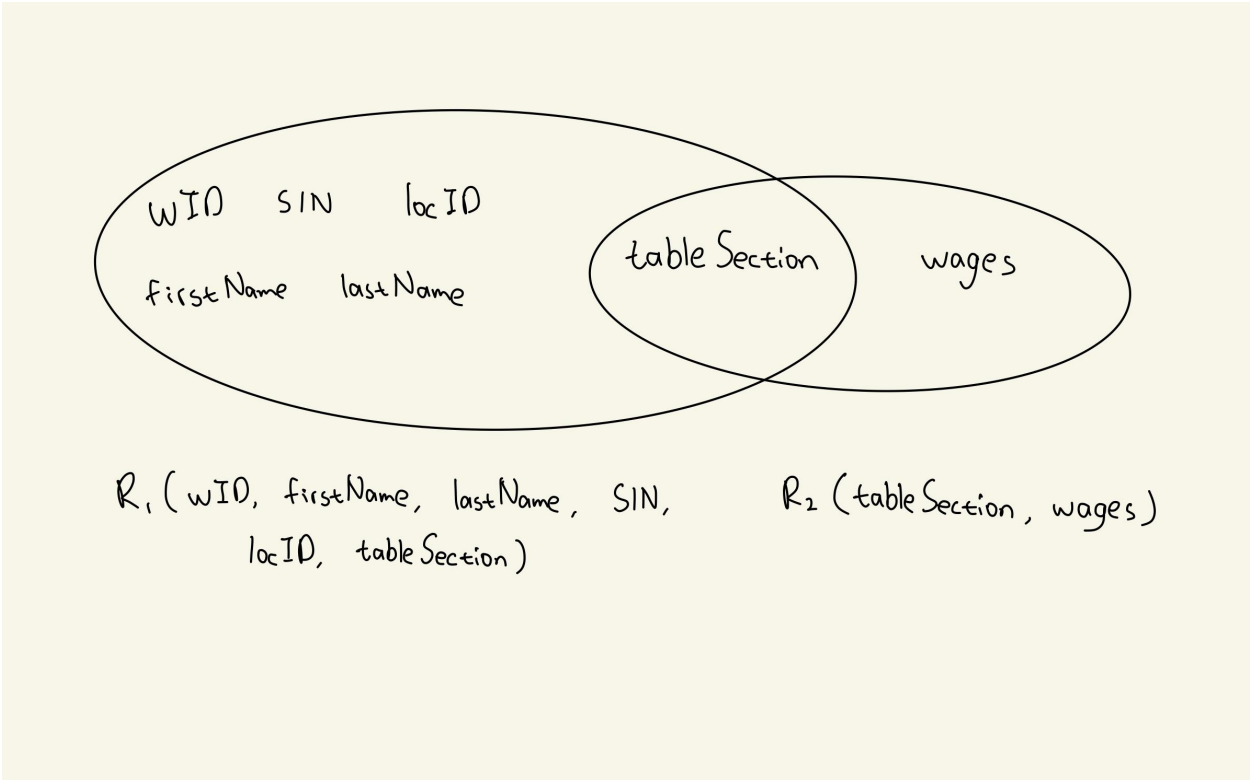
MenuServed(**menuName**: char(50) , **locID**: int)

Waiters:
$\{wID\}^+$ = {wID, firstName, lastName, wages, SIN, tableSection, locID}
$\{SIN\}^+$ = {wID, firstName, lastName, wages, SIN, tableSection, locID}
$\{tableSection\}^+$ = {tableSection, wages}

candidate keys: wID, SIN

Steps for decomposition:

1. FD: tableSection -> wages (non-trivial dependency)
   the functionality dependency above violates the BCNF because tableSection is not a superkey for Waiters. Hence, Waiters is not in BCNF.
2. We will decompose Waiters into two relations as shown below.



3. Let's check the first relation to the FDs.
   $\{wID\}^+$ = {wID, firstName, lastName, SIN, tableSection, locID}
   $\{SIN\}^+$ = {wID, firstName, lastName, SIN, tableSection, locID}

   These FDs are not applicable for this relation:
   - wID -> wages
   - SIN -> wages
   - tableSection -> wages

   The first relation is in BCNF with wID and SIN as the candidate keys.

   Let's check the second relation to the FDs.
   $\{tableSection\}^+$ = {tableSection, wages}
   tableSection -> wages is the only FD applicable for this relation and it doesn't violate BCNF

   The second relation is in BCNF with tableSection as the primary and candidate key.

4. I will name the first relation 'WaitersInfo' and the second relation 'WaitersTableSection'

WaitersInfo (<u>wID</u> : int, firstName: char(50) , lastName: char(50), SIN: char(9), **tableSection**: char(50), **locID**: int) (SIN cannot be null and must be unique; candidate keys: SIN)

WaitersTableSection(<u>tableSection</u>: char(50), wages: float)

Chef:
$\{wID\}^+$ = {wID, firstName, lastName, wages, SIN, specialty, locID}
$\{SIN\}^+$ = {SIN, wID, firstName, lastName, wages, specialty, locID}
$\{specialty\}^+$ = {specialty, wages}
keys: wID, SIN
Steps for decomposition:

1) specialty -> wages are non-trivial dependency, but specialty is not a superkey of Chef, so Chef is not in BCNF
2) Decompose into two tables; ChefInfo(wID : int, firstName: char(50), lastName: char(50), SIN: char(9), specialty: char(50), **locID**: int) and ChefSpecialty(specialty: char(50), wages: float)
3) Check if ChefInfo is in BCNF: $\{wID\}^+$ = {wID, firstName, lastName, SIN, specialty, locID), $\{SIN\}^+$ = {SIN, wID, firstName, lastName, specialty, locID) the keys are only wID and SIN and there are no non-trivial FD, so ChefInfo is in BCNF
4) Check if ChefSpecialty is in BCNF: $\{specialty\}^+$ = {specialty, wages} the key is specialty only, so the FD specialty -> wages hold.

ChefInfo(wID : int, firstName: char(50), lastName: char(50), SIN: char(9), **specialty**: char(50), **locID**: int) (SIN cannot be null and must be unique; candidate keys: SIN)

ChefSpecialty(specialty: char(50), wages: float)

Hosts (wID : int, firstName: char(50), lastName: char(50), wages: float, SIN: char(9), **locID**: int) (SIN cannot be null and must be unique; candidate keys: SIN)

Managers (wID : int, firstName: char(50), lastName: char(50), wages: float, SIN: char(9), rating: float, **locID**: int) (SIN cannot be null and must be unique; candidate keys: SIN)

Equipments:
[Before normalization: Equipments(id: int, name: char(50), category: char(50), condition: char(50), yearBought: int)]
$\{id\}^+$ = {id, name, category, condition}
$\{name\}^+$ = {name, category}

Step 1: Functional dependency name -> category is a non-trivial dependency but name is not a superkey thus it violates BCNF.

Step 2: Decompose Equipment by splitting it into two tables. EquipmentsMain(id: int, name: char(50), condition: char(50), yearBought: int) and EquipmentsName(name: char(50), category: char(50)).

Step 3a: Check if they violate BCNF. EquipmentsMain(id: int, name: char(50), condition: char(50), yearBought: int): id -> name, condition, yearBought holds for {id}+ = {id, name, category, condition} and does not violate BCNF. EquipmentsName(name: char(50), category: char(50)) name -> category holds for {name}+ = {name, category} and does not

violate BCNF. As both relations does not violate BCNF the decomposition is done and we end up with two tables, EquipmentsMain and EquipmentName.

EquipmentsMain(id: int, **name**: char(50), condition: char(50), yearBought: int)

EquipmentsName(name: char(50), category: char(50))

EquipmentContained (**locID** : int, **equipID** : int)

Reservations(rID: int , **cID**: int**, locID**: int **, wID**: int **,** date: date, time: time, numOfPeople: int, reservationName: char(50)) (cID must not be null, locID must not be null, wID must not be null)

SQL DDL

```
CREATE TABLE Menu
        (name           CHAR(50)        PRIMARY KEY,
         cost           FLOAT,
         category       CHAR(50)        NOT NULL
)

CREATE TABLE Ingredients
        (name                   CHAR(50)        PRIMARY KEY,
         storageMethod          CHAR(50)
)

CREATE TABLE IngredientsRequired
        (menuName               CHAR(50),
         ingredientsName        CHAR(50),
         ingredientsAmount      INTEGER,
         PRIMARY KEY (menuName, ingredientsName),
         FOREIGN KEY (menuName)
                REFERENCES Menu (name)
                        ON DELETE CASCADE
                        ON UPDATE CASCADE,
         FOREIGN KEY (ingredientsName)
                REFERENCES Ingredients (name)
                        ON DELETE CASCADE
                        ON UPDATE CASCADE
)

CREATE TABLE Customers
        (cID            INTEGER         PRIMARY KEY,
```

```
        email           CHAR(50)        NOT NULL,
        phoneNum        CHAR(10)        NOT NULL,
        points          INTEGER         NOT NULL,
        firstName       CHAR(50),
        lastName        CHAR(50),
        dob             DATE,
        UNIQUE (email, phoneNum)
)

CREATE TABLE MembershipsSubscribes
        (cID    INTEGER,
        tiers   CHAR(50),
        PRIMARY KEY (cID, tiers),
        FOREIGN KEY (cID)
                REFERENCES Customers (cID)
                        ON DELETE CASCADE
                        ON UPDATE CASCADE
)

CREATE TABLE Branches
        (locID                  INTEGER         PRIMARY KEY,
        streetAddress           CHAR(50),
        city                    CHAR(50),
        province                CHAR(50)
)

CREATE TABLE MenuServed
        (menuName   CHAR(50),
        locID           INTEGER,
        PRIMARY KEY (menuName, locID),
        FOREIGN KEY (menuName)
                REFERENCES Menu (name)
                        ON DELETE CASCADE
                        ON UPDATE CASCADE,
        FOREIGN KEY (locID)
                REFERENCES Branches (locID)
                        ON DELETE CASCADE
                        ON UPDATE CASCADE
)

CREATE TABLE WaitersTableSection
        (tableSection   CHAR(50)        PRIMARY KEY,
        wages           FLOAT
)
```

```
CREATE TABLE WaitersInfo
        (wID            INTEGER         PRIMARY KEY,
         firstName      CHAR(50),
         lastName       CHAR(50),
         SIN            CHAR(9)         NOT NULL,
         tableSection   CHAR(50),
         locID          INTEGER,
         UNIQUE (SIN),
         FOREIGN KEY (tableSection)
                REFERENCES WaitersTableSection (tableSection)
                        ON DELETE SET NULL
                        ON UPDATE CASCADE,
         FOREIGN KEY (locID)
                REFERENCES Branches (locID)
                        ON DELETE SET NULL
                        ON UPDATE CASCADE
)

CREATE TABLE ChefSpecialty(
        specialty       CHAR(50),
        wages           FLOAT,
        PRIMARY KEY (specialty)
)

CREATE TABLE ChefInfo(
         wID            INTEGER         PRIMARY KEY,
         firstName      CHAR(50),
         lastName       CHAR(50),
         SIN            CHAR(9)         NOT NULL,
         specialty      CHAR(50),
         locID          INTEGER,
         UNIQUE (SIN),
         FOREIGN KEY (specialty)
                REFERENCES ChefSpecialty(specialty)
                        ON DELETE SET DEFAULT
                        ON UPDATE CASCADE,
         FOREIGN KEY (locID)
                REFERENCES Branches (locID)
                        ON DELETE SET NULL
                        ON UPDATE CASCADE
)

CREATE TABLE Hosts(
```

```
        wID             INTEGER         PRIMARY KEY,
        firstName       CHAR(50),
        lastName        CHAR(50),
        wages           FLOAT,
        SIN             CHAR(9)         NOT NULL,
        locID           INTEGER,
        UNIQUE (SIN),
        FOREIGN KEY (locID)
                REFERENCES Branches (locID)
                        ON DELETE SET NULL
                        ON UPDATE CASCADE
)

CREATE TABLE Manager(
        wID             INTEGER         PRIMARY KEY,
        firstName       CHAR(50),
        lastName        CHAR(50),
        SIN             CHAR(9)         NOT NULL,
        rating          FLOAT,
        locID           INTEGER,
        UNIQUE (SIN),
        FOREIGN KEY (locID)
                REFERENCES Branches (locID)
                        ON DELETE SET NULL
                        ON UPDATE CASCADE
)

CREATE TABLE EquipmentsName(
        name            CHAR(50)        PRIMARY KEY,
        category        CHAR(50),
)

CREATE TABLE EquipmentsMain(
        id              INTEGER         PRIMARY KEY,
        name            CHAR(50),
        condition       CHAR(50),
        yearBought      INTEGER,
        FOREIGN KEY(name)
                REFERENCES EquipmentsName (name)
                        ON DELETE CASCADE
                        ON UPDATE CASCADE
)

CREATE TABLE EquipmentContained(
```

```
    locID           INTEGER,
    equipID         INTEGER,
    PRIMARY KEY (locID, equipID),
    FOREIGN KEY (locID),
        REFERENCES Branches (locID)
            ON DELETE CASCADE
            ON UPDATE CASCADE,
    FOREIGN KEY (equipID)
        REFERENCES EquipmentMain(id)
            ON DELETE CASCADE
            ON UPDATE CASCADE
)

CREATE TABLE RESERVATIONS(
    rID                 INTEGER,
    cID                 INTEGER     NOT NULL,
    locID               INTEGER     NOT NULL,
    wID                 INTEGER     NOT NULL,
    date                DATE,
    time                TIME,
    numOfPeople         INTEGER,
    reservationName     CHAR(50),
    PRIMARY KEY (rID),
    FOREIGN KEY (cID)
        REFERENCES Customers (cID)
            ON DELETE CASCADE
            ON UPDATE CASCADE,
    FOREIGN KEY (locID)
        REFERENCES Branches (locID)
            ON DELETE CASCADE
            ON UPDATE CASCADE,
    FOREIGN KEY (wID)
        REFERENCES Employee (wID)
            ON DELETE CASCADE
            ON UPDATE CASCADE
)
```

Instances

Menu

name	cost	category
Miso and mango soup	7.89	Main Dish
Chervil and prosciutto salad	6.49	Appetizer
Tangerine and ugli fruit salad	6.99	Appetizer
Haddock and pepper pie	9.99	Main Dish
Courgette and ginger cake	6.49	Dessert

Ingredients

name	storageMethod
Onion	Room Temperature
Miso	Refrigerator
Mango	Room Temperature
Lettuce	Refrigerator
Chevril	Refrigerator
Prosciutto	Refrigerator
Tangerine	Room Temperature
Ugli Fruit	Room Temperature
Flour	Room Temperature
Butter	Room Temperature
Haddock	Freezer
Pepper	Room Temperature
Egg	Refrigerator
Sugar	Room Temperature
Courgette	Freezer
Ginger	Room Temperature

IngredientsRequired

menuName	ingredientsName	ingredientsAmount
Miso and mango soup	Onion	2
Miso and mango soup	Miso	1
Miso and mango soup	Mango	1
Chervil and prosciutto salad	Lettuce	3
Chervil and prosciutto salad	Chevril	3
Chervil and prosciutto salad	Prosciutto	2
Tangerine and ugli fruit salad	Lettuce	2
Tangerine and ugli fruit salad	Tangerine	1
Tangerine and ugli fruit salad	Ugli Fruit	2
Haddock and pepper pie	Flour	1
Haddock and pepper pie	Butter	1
Haddock and pepper pie	Sugar	1
Haddock and pepper pie	Haddock	2
Haddock and pepper pie	Pepper	1
Courgette and ginger cake	Flour	1
Courgette and ginger cake	Butter	1
Courgette and ginger cake	Egg	2
Courgette and ginger cake	Sugar	1
Courgette and ginger cake	Courgette	2
Courgette and ginger cake	Ginger	1

Customers

cID	email	phoneNum	firstName	lastName	dob
1	j_smith@hotmail.com	6043324567	John	Smith	1998-08-23
2	bob@gmail.com	7781234912	Bob	Smith	1998-08-01
3	tracyk@outlook.com	1234567892	Tracy	Kim	2001-06-13
4	sarawang@gmail.com	2345678912	Sara	Wang	1972-01-03
5	zoem@hotmail.com	3456789012	Zoe	Miller	1980-04-12

MembershipSubscribes

cID	tiers
1	Gold
2	Silver
3	Gold
4	Bronze
5	Silver

MenuServed

menuName	locID
Miso and mango soup	111
Courgette and ginger cake	111
Miso and mango soup	112
Tangerine and ugli fruit salad	112
Courgette and ginger cake	112
Miso and mango soup	113
Chervil and prosciutto salad	113
Haddock and pepper pie	113
Miso and mango soup	114
Chervil and prosciutto salad	114
Tangerine and ugli fruit salad	114
Haddock and pepper pie	114
Courgette and ginger cake	114
Haddock and pepper pie	115
Courgette and ginger cake	115

Branches

locID	streetAddress	city	province
111	792 Montreal Road	Ottawa	Ontario
112	1257 Beaver Creek	Thornhill	Ontario
113	4752 Tolmie Street	Vancouver	British Columbia
114	1221 Robson Street	Vancouver	British Columbia
115	4795 Robson Street	Vancouver	British Columbia

WaitersInfo

wID	firstName	lastName	SIN	tableSection	locID
1	Joe	Smith	123456789	VIP Second Floor	111
3	Tom	Miller	234567891	First Floor General	112
5	Tina	Johnson	345678901	First Floor General	111
7	Alex	Wang	456789012	Second Floor General	113
9	Jenny	Kim	567890123	VIP First Floor	111

WaitersTableSection

tableSection	wages
VIP Second Floor	23.31
First Floor General	18.23
First Floor Banquet	20.31
Second Floor General	18.55
VIP First Floor	23.11

ChefInfo

wID	firstName	lastName	specialty	locID
200	Oliwer	Dennir	Head	111
204	Janice	Patel	Sous	111
206	Sion	Jordan	Saucier	111
208	Kelly	Weis	Pastry	111
210	Whitney	Dunn	Prep	111
212	Jerry	Dunn	Sous	112
214	Diana	Cherry	Pastry	112
216	Eryn	Knapp	Prep	112
218	Eryn	Le	Sous	113
220	Jakob	Singh	Prep	113
222	Shauna	Stark	Pastry	114
224	Lauren	Gordon	Prep	114
226	Edmun	Gordon	Sous	115
228	Essa	Cantrell	Prep	115

ChefSpecialty

specialty	wages
Head	39.95
Sous	29.95
Saucier	20.55
Pastry	18.35
Prep	17.25
Line	16.75

Managers

wID	firstName	lastName	wages	SIN	rating	locID
16	Jamilah	Gayle	25.5	112345678	8.9	113
17	Kris	Lena	26.12	111234567	9	115
18	Joost	Taavetti	24.33	111123456	7.5	112
19	Helga	Aitor	25.41	111112345	8.1	111
20	Davide	Artie	24.61	111111234	7.5	114

Hosts

wID	firstName	lastName	wages	SIN	locID
11	Tom	John	21.53	203948576	111
12	Jim	Jones	19.98	564738291	112
13	Anna	Hills	21.21	543526169	113
14	Tim	Jones	20.12	123456787	114
15	Tina	John	20.34	123456765	115

EquipmentMain

id	name	condition	yearBought
1	Table	Good	2020
2	Stand Mixer	Fair	2005
3	Chair	Good	2021
4	Christmas Lights	Poor	1999
5	Christmas Tree	Very Good	2022

EquipmentsName

name	category
Table	Furniture
Stand Mixer	Appliance
Chair	Furniture
Christmas Lights	Decoration
Christmas Tree	Decoration

EquipmentContained

locID	equipID
111	1
111	3
112	1
112	2
112	3
113	1
113	3
114	1
114	3
115	1
115	2
115	3
115	4
115	5

Reservations

rID	cID	locID	wID	date	time	numOfPeople	reservationName
1000	1	111	11	2023-08-01	18:20:00	4	John
1001	2	113	13	2023-08-01	19:00:00	3	Bob's birthday
1002	1	111	11	2023-08-03	19:30:00	8	Marketing team social
1003	4	112	12	2023-08-04	18:00:00	2	Wang's anniversary
1004	3	113	13	2023-08-04	18:20:00	3	Tracy
1005	2	114	14	2023-08-05	19:00:00	6	Bob's birthday