CPSC 304 Project Cover Page

Milestone #: 2

Date: March 1 , 2024

Group Number: 36

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Wendy Li	38036695	lwqz	wendyli.org@gmail.com
Antonia Tykei	82340886	atykei	antoniatykei@gmail.com
Wendy Tso	34159368	wendytso	wendywktso@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

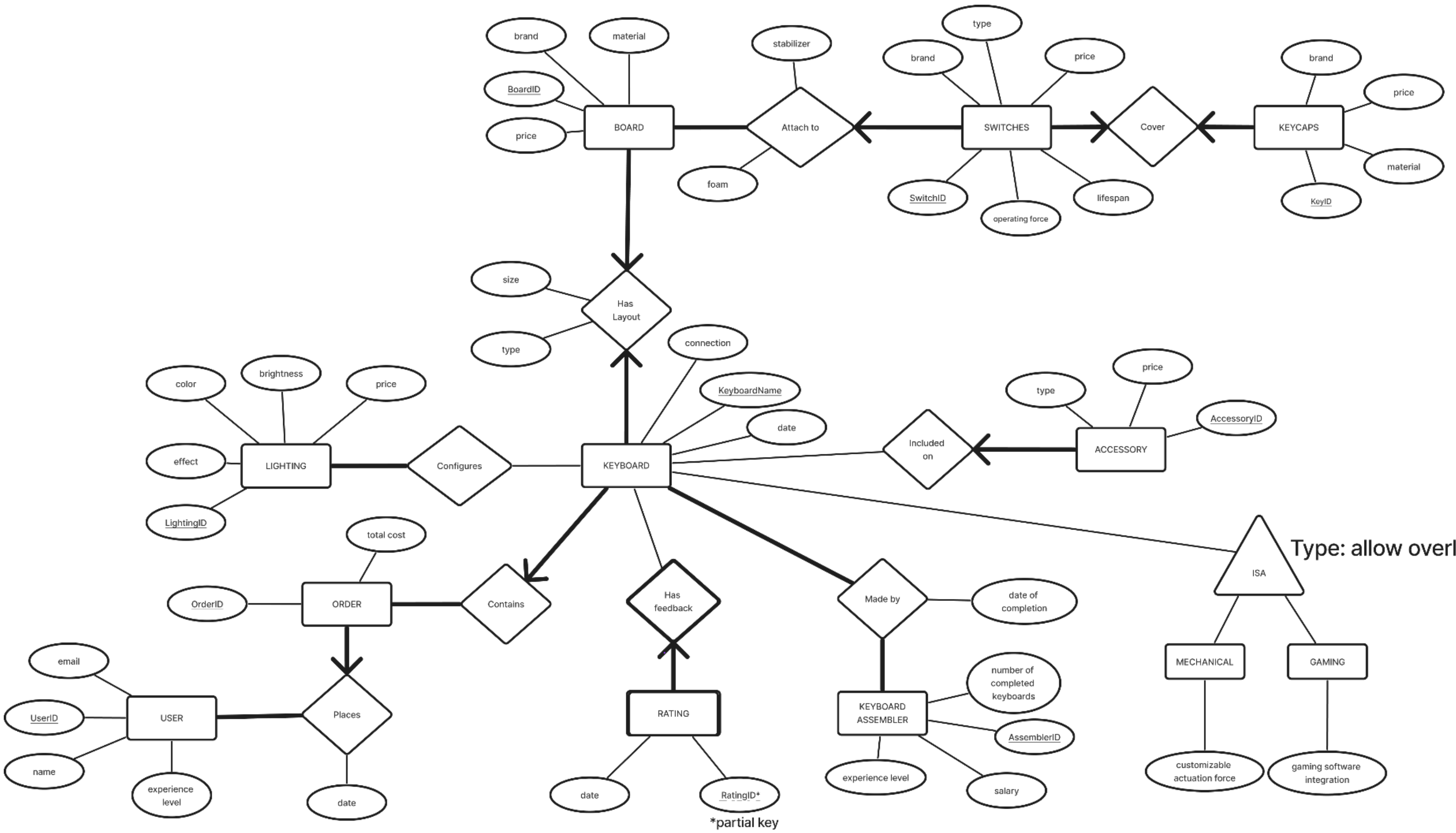
Brief summary (~2-3 sentences) of the project to help TAs remember details.

The domain of this project is mechanical keyboard customization. Putting together and building a mechanical keyboard requires many different components that interact and combine in different ways. Our project aims to create a comprehensive informational database and ordering system that allows users to create, design, organize, and place orders for their customized keyboards.

ER diagram, which may be the same as Milestone 1 or different. Note any changes made and reasons for them.

We have included all the feedback from our TA, specifically ensuring that we have the correct number of entities and relationships and ensuring that naming of attributes is clear and consistent. We evaluated and changed some of the relationships (e.g removed some weak entities, removed the aggregation, introduced new entities, etc.). We have also tried to rename our relations to verbs as recommended.

https://www.figma.com/file/22vXWAx0YyCxYUsz6HvhN3/304-ER-Diagram?type=whiteboard&node-id=0-1&t=neT2ePY7J5fxmZQc-0



Type: allow overlap, yes covering

*partial key

Schema derived from the ER diagram, including the translation into the relational model:

- USER (UserID: varchar, name: varchar, email: varchar, experience level: varchar)
  - *Experience level NOT NULL*

- PLACES_ORDER (OrderID: varchar, **UserID:** varchar, date: date, total cost: real)
  - *FK: UserID references USER*

- KEYBOARD_CONTAINS (KeyboardName: varchar, **OrderID:** varchar, connection: varchar, date: date)
  - *FK: OrderID references PLACES_ORDER*

- MECHANICAL (**KeyboardName**: varchar, customizable actuation force : varchar)
  - *FK: KeyboardName references (‘superclass’) KEYBOARD_CONTAINS*

- GAMING (**KeyboardName**: varchar, gaming software integration: varchar)
  - *FK: KeyboardName references (‘superclass’) KEYBOARD_CONTAINS*

- KEYBOARD_ASSEMBLER (AssemblerID: int, experience level: varchar, salary: real, number of completed keyboards: int)
  - *experience level is NOT NULL*
  - *number of completed keyboard is NOT NULL*

- MADE_BY (**AssemblerID:** int, **KeyboardName:** varchar, date of completion: date)
  - *Number of keyboard completed set DEFAULT 0*
  - *FK: KeyboardName references KEYBOARD_CONTAINS*
  - *FK: AssemblerID references KEYBOARD_ASSEMBLER*

- RATING_HAS_FEEDBACK (RatingID: varchar, **KeyboardName**: varchar, date: date)
  - *FK: KeyboardName references KEYBOARD_CONTAINS*

- LIGHTING (LightingID: int, brightness: int, price: real, color: varchar, effect: varchar)
  - *Color is NOT NULL*

- CONFIGURES (**LightingID**: int, **KeyboardName**: varchar)
  - *FK: KeyboardName references KEYBOARD_CONTAINS*
  - *FK: LightingID references LIGHTING*

- ACCESSORY_INCLUDED_ON (AccessoryID: varchar, price: real, type: varchar, **KeyboardName:** varchar)
  - *FK: KeyboardName references KEYBOARD_CONTAINS*

- BOARD_HAS_LAYOUT (BoardID: int, brand: varchar, material: varchar, price: real, size: real, type: varchar, **KeyboardName:** varchar)
  - *FK: KeyboardName UNIQUE (one to one - so it is CK)*

- SWITCHES_ATTACH_TO (SwitchID: int, operating force: varchar, brand: varchar, type: varchar, price: real, lifespan: int, **BoardID**: int, **KeyID:** varchar)
  - *BoardID is NOT NULL*
  - *FK: BoardID references BOARD_HAS_LAYOUT*
  - *FK: KeyID references KEYCAP*

- KEYCAP (KeyID: varchar, price: real, material: varchar, brand: varchar, **SwitchID:** int)
  - *SwitchID is UNIQUE*

Functional Dependencies (FDs):

Identify the functional dependencies in your relations, including the ones involving all candidate keys (including the primary key). PKs and CKs are considered functional dependencies and should be included in the list of FDs. You do not need to include trivial FDs such as A → A.

USER (UserID, name, email, experience level)
UserID → experience level
UserID → email
UserID → name
email → name

PLACES_ORDER (OrderID, **UserID,** date, total cost)
OrderID → UserID
OrderID → date
OrderID → total cost

KEYBOARD_CONTAINS (KeyboardName, **OrderID,** connection, date)
KeyboardName → connection
KeyboardName → date
KeyboardName → OrderID

MECHANICAL (**KeyboardName**, customizable actuation force)
KeyboardName → customizable actuation force

GAMING (**KeyboardName**, gaming software integration)
KeyboardName → gaming software integration

KEYBOARD_ASSEMBLER (AssemblerID, experience level, salary, number of completed keyboards)
AssemblerID → experience level
AssemblerID → number of completed keyboard
experience level, number of completed keyboard → salary

MADE_BY (**AssemblerID**, **KeyboardName**, date of completion)
AssemblerID, Keyboard Name → date of completion

RATING_HAS_FEEDBACK (RatingID, **KeyboardName**, date)
RatingID, KeyboardName → date

LIGHTING (LightingID, brightness, price, color, effect)
LightingID → price

LightingID → color
LightingID → effect
LightingID → brightness
color → brightness

CONFIGURES (**LightingID**, **KeyboardName**)
LightingID → KeyboardName
Keyboard → LightingID

ACCESSORY_INCLUDED_ON (AccessoryID, price, type, **KeyboardName**)
AccessoryID → price
AccessoryID → type
AccessoryID → KeyboardName

BOARD_HAS_LAYOUT (BoardID, brand, material, price, size, type, **KeyboardName**)
BoardID → brand
BoardID → material
BoardID → price
BoardID → size
BoardID → type
BoardID → KeyboardName

SWITCHES_ATTACH_TO (SwitchID, operating force, brand, type, price, lifespan, **BoardID**, **KeyID**)
SwitchID → operating force
SwitchID → brand
SwitchID → type
SwitchID → price
SwitchID → lifespan
SwitchID → BoardID
SwitchID → KeyID

KEYCAP (KeyID, price, material, brand, **SwitchID**)
KeyID → material
KeyID → brand
KeyID → switchID
KeyID → price

Normalization:

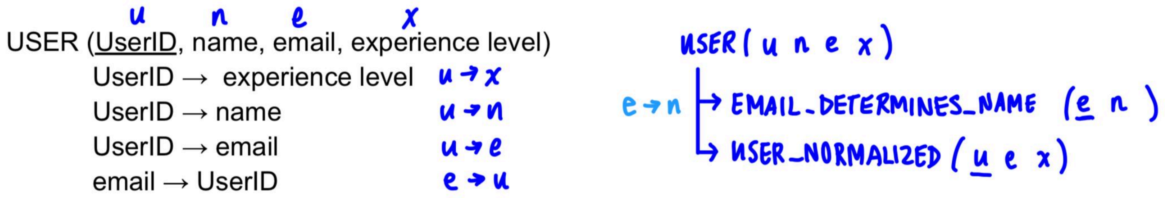
Normalize each of your tables to be in 3NF or BCNF. Give the list of tables, their primary keys, their candidate keys, and their foreign keys after normalization.

USER (UserID, name, email, experience level)

UserID ⟶ experience level
UserID ⟶ email
UserID ⟶ name
email ⟶ name

We can deduce that UserID is the Candidate Key (CK), because it's the only attribute that is only on LHS, and UserID+ = USER.

The USER relation is NOT in 3NF or BCNF because <email ⟶ name> FD violates BCNF (email is not CK), let us decompose to:



u n e x
USER (UserID, name, email, experience level)
UserID → experience level u → x
UserID → name u → n
UserID → email u → e
email → UserID e → u

USER (u n e x)
e → n
→ EMAIL_DETERMINES_NAME (e n)
→ USER_NORMALIZED (u e x)

USER_NORMALIZED(UserID, **email**, experience level)

EMAIL_DETERMINES_NAME(email, name)

PLACES_ORDER (OrderID, **UserID,** date, total cost)

OrderID ⟶ UserID
OrderID ⟶ date
OrderID ⟶ total cost

We can deduce that OrderID is CK because OrderID+ = PLACES_ORDER.

We observe that all FDs have CK on LHS, thus we may conclude that PLACES_ORDER is in BCNF.

KEYBOARD_CONTAINS (KeyboardName, **OrderID,** connection, date)

KeyboardName ⟶ connection
KeyboardName ⟶ date
KeyboardName ⟶ OrderID

We can deduce that KeyboardName is CK because
KeyboardName+ = KEYBOARD_CONTAINS.

We observe that all FDs have CK on LHS, thus we may conclude that KEYBOARD_CONTAINS is in BCNF.

MECHANICAL (**KeyboardName**, customizable actuation force)

KeyboardName ⟶ customizable actuation force

We can deduce that KeyboardName is CK because KeyboardName+ = MECHANICAL .

We observe that all FDs have CK on LHS, thus we may conclude that MECHANICAL is in BCNF.

GAMING (**KeyboardName**, gaming software integration)

KeyboardName → gaming software integration

We can deduce that KeyboardName is CK because KeyboardName+ = GAMING .
We observe that all FDs have CK on LHS, thus we may conclude that GAMING is in BCNF.

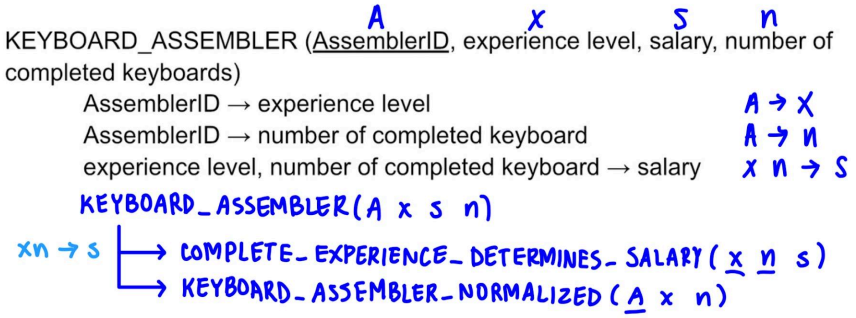
KEYBOARD_ASSEMBLER (AssemblerID, experience level, salary, number of completed keyboards)

AssemblerID → experience level

AssemblerID → number of completed keyboard

experience level, number of completed keyboard → salary

We can deduce that AssemblerID is the Candidate Key (CK), because it’s the only attribute that is only on LHS, and AssemblerID+ = KEYBOARD_ASSEMBLER .
The KEYBOARD_ASSEMBLER relation is NOT in 3NF or BCNF because the <experience level, number of completed keyboard → salary> FD violates BCNF (experience level, number of completed keyboard is not CK), let us decompose to:



KEYBOARD_ASSEMBLER (AssemblerID, experience level, salary, number of completed keyboards)

A x S n

AssemblerID → experience level   A → X

AssemblerID → number of completed keyboard   A → n

experience level, number of completed keyboard → salary   X n → S

KEYBOARD_ASSEMBLER(A x s n)

xn → s

→ COMPLETE_EXPERIENCE_DETERMINES_SALARY(x n s)

→ KEYBOARD_ASSEMBLER_NORMALIZED(A x n)

COMPLETE_EXPERIENCE_DETERMINES_SALARY (experience level, number of completed keyboards, salary)

KEYBOARD_ASSEMBLER_NORMALIZED (AssemblerID, **experience level**, **number of completed keyboards**)

MADE_BY (**AssemblerID**, **KeyboardName**, date of completion)

AssemblerID, KeyboardName → date of completion

We can deduce that (AssemblerID, KeyboardName) is CK because
(AssemblerID, KeyboardName)+ = MADE_BY.
We observe that all FDs have CK on LHS, thus we may conclude that MADE_BY is in BCNF.

RATING_HAS_FEEDBACK (RatingID, **KeyboardName**, date)

RatingID, KeyboardName → date

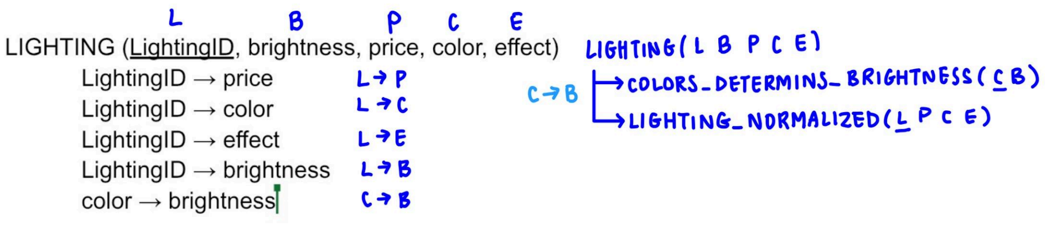
We can deduce that (RatingID, KeyboardName) is CK because
(RatingID, KeyboardName)+ = RATING_HAS_FEEDBACK .

We observe that all FDs have CK on LHS, thus we may conclude that RATING_HAS_FEEDBACK is in BCNF.

LIGHTING (LightingID, brightness, price, color, effect)

LightingID ⟶ price
LightingID ⟶ color
LightingID ⟶ effect
LightingID ⟶ brightness
color ⟶ brightness

We can deduce that LightingID is the Candidate Key (CK), because it’s the only attribute that is only on LHS, and LightingID+ = LIGHTING .
The LIGHTING relation is NOT in 3NF or BCNF because the
<color ⟶ brightness> FD violates BCNF (color is not CK), let us decompose to:



LIGHTING (LightingID, brightness, price, color, effect)

LightingID → price
LightingID → color
LightingID → effect
LightingID → brightness
color → brightness

L
B
P
C
E
LIGHTING(L B P C E)
L→P
L→C
L→E
L→B
C→B
C→B
→COLORS_DETERMINS_BRIGHTNESS(C B)
→LIGHTING_NORMALIZED(L P C E)

COLOR_DETERMINES_BRIGHTNESS (color, brightness)

LIGHTING_NORMALIZED (LightingID, price, **color**, effect)

CONFIGURES (**LightingID**, **KeyboardName**)
We can deduce that (LightingID, KeyboardName) is CK because
(LightingID, KeyboardName)+ = CONFIGURES .
We observe that all FDs have CK on LHS, thus we may conclude that CONFIGURES is in BCNF.

ACCESSORY_INCLUDED_ON (AccessoryID, price, type, **KeyboardName**)

AccessoryID ⟶ price
AccessoryID ⟶ type
AccessoryID ⟶ KeyboardName

We can deduce that (AccessoryID, KeyboardName) is CK because
(AccessoryID, KeyboardName)+ = ACCESSORY_INCLUDED_ON .
We observe that all FDs have CK on LHS, thus we may conclude that ACCESSORY_INCLUDED_ON is in BCNF.

BOARD_HAS_LAYOUT (BoardID, brand, material], price, size, type, **KeyboardName**)

BoardID ⟶ brand
BoardID ⟶ material
BoardID ⟶ price
BoardID ⟶ size

BoardID → type

BoardID → KeyboardName

We can deduce that BoardID is CK because BoardID+ = BOARD_HAS_LAYOUT .

We observe that all FDs have CK on LHS, thus we may conclude that BOARD_HAS_LAYOUT is in BCNF.

SWITCHES_ATTACH_TO (<u>SwitchID</u>, operating force, brand, type, price, lifespan, **BoardID**, **KeyID**)

SwitchID → operating force

SwitchID → brand

SwitchID → type

SwitchID → price

SwitchID → lifespan

SwitchID → BoardID

SwitchID → KeyID

We can deduce that SwitchID is CK because SwitchID+ = SWITCHES_ATTACH_TO .

We observe that all FDs have CK on LHS, thus we may conclude that SWITCHES_ATTACH_TO is in BCNF.

KEYCAP (<u>KeyID</u>, price, material, brand, **SwitchID**)

KeyID → material

KeyID → brand

KeyID → switchID

KeyID → price

We can deduce that KeyID is CK because KeyID+ = KEYCAP .

We observe that all FDs have CK on LHS, thus we may conclude that KEYCAP is in BCNF.

<u>LIST OF NORMALIZED TABLES:</u>

- USER_NORMALIZED (<u>UserID</u>: varchar, **email:** varchar, experience level: varchar)
    - *FK: email references* EMAIL_DETERMINES_NAME

- EMAIL_DETERMINES_NAME (<u>email</u>**:** varchar, name: varchar)

- PLACES_ORDER (<u>OrderID</u>: varchar, **UserID:** varchar, date: date, total cost: real)
    - *FK: UserID references USER*

- KEYBOARD_CONTAINS (KeyboardName: varchar, **OrderID:** varchar, connection: varchar, date: date)
  - *FK: OrderID references PLACES_ORDER*

- MECHANICAL (**KeyboardName**: varchar, customizable actuation force : varchar)
  - *FK: KeyboardName references (‘superclass’) KEYBOARD_CONTAINS*

- GAMING (**KeyboardName**: varchar, gaming software integration: varchar)
  - *FK: KeyboardName references (‘superclass’) KEYBOARD_CONTAINS*

- KEYBOARD_ASSEMBLER_NORMALIZED (AssemblerID: int, **experience level**: varchar, **number of completed keyboards**: int)
  - *FK: experience level references* COMPLETE_EXPERIENCE_DETERMINES_SALARY
  - *FK: number of completed keyboards references* COMPLETE_EXPERIENCE_DETERMINES_SALARY

- COMPLETE_EXPERIENCE_DETERMINES_SALARY (experience level, number of completed keyboards, salary)

- MADE_BY (**AssemblerID:** int, **KeyboardName:** varchar, date of completion: date)
  - *Number of keyboard completed set DEFAULT 0*

- RATING_HAS_FEEDBACK (RatingID: varchar, **KeyboardName**: varchar, date: date)
  - *FK: KeyboardName references KEYBOARD_CONTAINS*

- LIGHTING_NORMALIZED (LightingID: int, price: real, **color**: varchar, effect: varchar)
  - *FK: color references* COLOR_DETERMINES_BRIGHTNESS, color is NOT NULL

- COLOR_DETERMINES_BRIGHTNESS (color: varchar, brightness: int)

- CONFIGURES (**LightingID**: int, **KeyboardName**: varchar)
    - *FK: KeyboardName references KEYBOARD_CONTAINS*
    - *FK: LightingID references LIGHTING*

- ACCESSORY_INCLUDED_ON (AccessoryID: varchar, price: real, type: varchar, **KeyboardName:** varchar)
    - *FK: KeyboardName references KEYBOARD_CONTAINS*

- BOARD_HAS_LAYOUT (BoardID: int, brand: varchar, material: varchar, price: real, size: real, type: varchar, **KeyboardName:** varchar)
    - *FK: KeyboardName UNIQUE (one to one - so it is CK)*

- SWITCHES_ATTACH_TO (SwitchID: int, operating force: char[10], brand: varchar, type: varchar, price: real, lifespan: int, **BoardID**: int, **KeyID:** varchar)
    - *FK: BoardID references BOARD_HAS_LAYOUT, is NOT NULL*
    - *FK: KeyID references KEYCAP*

- KEYCAP (KeyID: varchar, price: real, material: varchar, brand: varchar, **SwitchID:** int)
    - *SwitchID is UNIQUE, is NOT NULL*

SQL DDL Statements:

The SQL DDL statements required to create all the tables from item # 6. The statements should use the appropriate foreign keys, primary keys, UNIQUE constraints, etc.

```
CREATE TABLE USER_NORMALIZED (
        UserID varchar(100),
        email varchar(100),
        experience_level varchar(100) NOT NULL,
        PRIMARY KEY (UserID),
        FOREIGN KEY (email) REFERENCES EMAIL_DETERMINES_NAME (email)
);

CREATE TABLE EMAIL_DETERMINES_NAME (
        email varchar(100) PRIMARY KEY,
        name varchar(100)
);

CREATE TABLE PLACES_ORDER (
        OrderID varchar(100),
        UserID varchar(100),
        date DATE,
        total_cost REAL,
        PRIMARY KEY (OrderID),
        FOREIGN KEY (UserID) REFERENCES USER(UserID)
);

CREATE TABLE KEYBOARD_CONTAINS (
        KeyboardName varchar(100),
        OrderID varchar(100),
        connection varchar(100),
        date DATE,
        PRIMARY KEY (KeyboardName),
        FOREIGN KEY (OrderID)
                REFERENCES PLACES_ORDER(OrderID)
);

CREATE TABLE MECHANICAL (
        KeyboardName varchar(100),
        customizable_actuation_force varchar(100),
        PRIMARY KEY (KeyboardName),
        FOREIGN KEY (KeyboardName)
        REFERENCES KEYBOARD_CONTAINS(KeyboardName)
);

CREATE TABLE GAMING (
    KeyboardName varchar(100),
    gaming_software_integration varchar(100),
    PRIMARY KEY (KeyboardName),
    FOREIGN KEY (KeyboardName)
        REFERENCES KEYBOARD_CONTAINS(KeyboardName)
);

CREATE TABLE KEYBOARD_ASSEMBLER_NORMALIZED (
    AssemblerID INTEGER,
    experience_level varchar(100),
    salary REAL NOT NULL,
    number_of_completed_keyboards INTEGER,
    PRIMARY KEY (AssemblerID)
    FOREIGN KEY (experience_level, number_of_completed_keyboards)
        REFERENCES COMPLETE_EXPERIENCE_DETERMINES_SALARY(
            experience_level, number_of_completed_keyboards)
);

CREATE TABLE COMPLETE_EXPERIENCE_DETERMINES_SALARY (
    experience_level varchar(100),
    salary REAL NOT NULL,
    number_of_completed_keyboards INTEGER,
    PRIMARY KEY (experience_level, number_of_completed_keyboards)
);

CREATE TABLE MADE_BY (
    AssemblerID INTEGER,
    KeyboardName varchar(100),
    date_of_completion DATE,
    PRIMARY KEY (AssemblerID, KeyboardName),
    FOREIGN KEY (AssemblerID) REFERENCES KEYBOARD_ASSEMBLER(AssemblerID),
    FOREIGN KEY (KeyboardName) REFERENCES KEYBOARD_CONTAINS(KeyboardName)
);


CREATE TABLE RATING_HAS_FEEDBACK (
    RatingID varchar(100),
    KeyboardName varchar(100),
    date DATE,
    PRIMARY KEY (RatingID, KeyboardName),
    FOREIGN KEY (KeyboardName)
        REFERENCES KEYBOARD_CONTAINS(KeyboardName)
        ON DELETE CASCADE
);

CREATE TABLE LIGHTING_NORMALIZED (
    LightingID INTEGER,
    price REAL,
    color varchar(100) NOT NULL,
    effect varchar(100) NOT NULL,
    PRIMARY KEY (LightingID),
    FOREIGN KEY (color)
        REFERENCES COLOR_DETERMINES_BRIGHTNESS(color)
        ON UPDATE CASCADE
);

CREATE TABLE COLOR_DETERMINES_BRIGHTNESS(
    color varchar(100),
    brightness INT,
    PRIMARY KEY (color)
);

CREATE TABLE CONFIGURES (
    LightingID INTEGER,
    KeyboardName varchar(100),
    PRIMARY KEY (LightingID, KeyboardName),
    FOREIGN KEY (LightingID) REFERENCES LIGHTING(LightingID),
    FOREIGN KEY (KeyboardName)
        REFERENCES KEYBOARD_CONTAINS(KeyboardName)
);

CREATE TABLE ACCESSORY_INCLUDED_ON (
    AccessoryID varchar(100),
    price REAL,
    type varchar(100),
    KeyboardName varchar(100),
    PRIMARY KEY (AccessoryID, KeyboardName),
    FOREIGN KEY (AccessoryID) REFERENCES ACCESSORIES(AccessoryID),
    FOREIGN KEY (KeyboardName)
        REFERENCES KEYBOARD_CONTAINS(KeyboardName)
```

```
);

CREATE TABLE BOARD_HAS_LAYOUT (
        BoardID INTEGER,
        brand varchar(100),
        Material varchar(100),
        price REAL,
        size REAL,
        type varchar(100),
        KeyboardName varchar(100) UNIQUE,
        PRIMARY KEY (BoardID),
        FOREIGN KEY (KeyboardName)
                REFERENCES KEYBOARD_CONTAINS(KeyboardName)
);

CREATE TABLE SWITCHES_ATTACH_TO (
        SwitchID INTEGER,
        operating_force varchar(100),
        brand varchar(100),
        Type varchar(100),
        price REAL,
        lifespan INTEGER,
        BoardID INTEGER NOT NULL,
        KeyID varchar(100),
        PRIMARY KEY (SwitchID),
        FOREIGN KEY (BoardID) REFERENCES BOARD_HAS_LAYOUT(BoardID)
                ON UPDATE CASCADE
        FOREIGN KEY (KeyID) REFERENCES KEYCAP(KeyID)
);

CREATE TABLE KEYCAP (
        KeyID varchar(100) PRIMARY KEY,
        price REAL,
        material varchar(100),
        brand varchar(100),
        SwitchID INTEGER UNIQUE,
        FOREIGN KEY (SwitchID)
                REFERENCES SWITCHES_ATTACH_TO(SwitchID)
);
```

INSERT Statements:

Provide INSERT statements to populate each table with at least 5 tuples, for meaningful queries later.

```
INSERT INTO USER_NORMALIZED (UserID, Email, Experience_level) VALUES
(‘user1’, 'ambikamod@hotmail.com', 'Intermediate'),
(‘user2’, 'leo@gmail.com', 'Advanced'),
(‘user3’, 'ellie@gmail.com', 'Beginner'),
(‘user4’, 'jon@ubc.com', 'Intermediate'),
(‘user5’, 'essiedavis@gmail.com', 'Advanced');

INSERT INTO EMAIL_DETERMINES_NAME (email, Name) VALUES
('ambikamod@hotmail.com', 'Ambika Mod'),
('leo@gmail.com', 'Leo Woodall'),
('ellie@gmail.com', Eleanor Tomlinson'),
('jon@ubc.com', 'Jonny Weldon'),
('essiedavis@gmail.com', 'Essie Davis');

INSERT INTO PLACES_ORDER (OrderID, UserID, Date, Total_cost) VALUES
('order1', 'user1', '2024-01-01', 50.00),
('order2', 'user2', '2024-01-02', 75.00),
('order3', 'user3', '2024-01-03', 100.00),
('order4', 'user4', '2024-01-04', 125.00),
('order5', 'user5', '2024-01-05', 150.00);

INSERT INTO KEYBOARD_CONTAINS (KeyboardName, OrderID, Connection, Date) VALUES
('keyboard1', ‘order1’, 'USB', '2023-01-15'),
('keyboard2', ‘order2’, 'Wireless', '2023-02-20'),
('keyboard3', ‘order3’, 'Bluetooth', '2023-03-10'),
('keyboard4', ‘order4’, 'USB', '2023-04-05'),
('keyboard5', ‘order5’, 'Wireless', '2023-05-18');

INSERT INTO MECHANICAL (KeyboardName, Customizable_actuation_force) VALUES
('keyboard1', '60g'),
('keyboard2', '65g'),
('keyboard3', '55g'),
('keyboard4', '70g'),
('keyboard5', '62g');

INSERT INTO GAMING (KeyboardName, gaming_software_integration) VALUES
('keyboard1', 'Razer Synapse'),
('keyboard2', 'Logitech G HUB'),
('keyboard3', 'Corsair iCUE'),
('keyboard4', 'SteelSeries Engine'),
('keyboard5', 'ASUS Armoury Crate');

INSERT INTO KEYBOARD_ASSEMBLER_NORMALIZED (AssemblerID, experience_level,
number_of_completed_keyboards) VALUES
(1, 'Intermediate', 3000.00, 10),
(2, 'Advanced', 4000.00, 15),
(3, 'Beginner', 2500.00, 5),
(4, 'Intermediate', 3200.00, 8),
(5, 'Advanced', 4200.00, 12);

INSERT INTO COMPLETE_EXPERIENCE_DETERMINES_SALARY(experience_level,
number_of_completed_keyboards, salary) VALUES
('Intermediate', 10, 3000.00),
('Advanced', 15, 4000.00),
('Beginner', 5,  2500.00),
('Intermediate', 8, 3200.00),
('Advanced', 12, 4200.00);

INSERT INTO MADE_BY (AssemblerID, KeyboardName, date_of_completion) VALUES
(1, 'keyboard1', '2023-01-20'),
(2, 'keyboard2', '2023-02-25'),
(3, 'keyboard3', '2024-01-15'),
(4, 'keyboard4', '2024-02-10'),
(5, 'keyboard5', '2024-02-23');

INSERT INTO RATING_HAS_FEEDBACK (RatingID, KeyboardName, date) VALUES
(‘rating1’, 'keyboard1', '2023-01-22'),
(‘rating2’, 'keyboard2', '2023-02-28'),
(‘rating3’, 'keyboard3', '2023-03-18'),
(‘rating4’, 'keyboard4', '2023-04-12'),
(‘rating5’, 'keyboard5', '2023-05-25');

INSERT INTO LIGHTING_NORMALIZED (LightingID, price, color, effect) VALUES
(1, 20.00, 'RGB', 'Static'),
(2, 30.00, 'Backlight', 'Pulse'),
(3, 40.00, 'White', 'Wave'),
(4, 25.00, 'RGB', 'Wriggle'),
(5, 35.00, 'Purple', 'Ripple');

INSERT INTO COLOR_DETERMINES_BRIGHTNESS (color, brightness) VALUES
```

```
(‘RGB’, 200000),
(‘Backlight’, 300),
(‘White’, 294849),
(‘Purple’, 55),
(‘Blue’, 71);

INSERT INTO CONFIGURES (LightingID, KeyboardName) VALUES
(1, 'keyboard1'),
(2, 'keyboard2'),
(3, 'keyboard3'),
(4, 'keyboard4'),
(5, 'keyboard5');

INSERT INTO ACCESSORY_INCLUDED_ON (AccessoryID, price, type, KeyboardName) VALUES
(‘accessory1’, 15.00, 'Wrist Rest', 'keyboard1'),
(‘accessory2’, 10.00, 'Keycap Puller', 'keyboard2'),
(‘accessory3’, 20.00, 'Cable Organizer', 'keyboard3'),
(‘accessory4’, 12.00, 'Switch Opener', 'keyboard4'),
(‘accessory5’, 18.00, 'Desk Mat', 'keyboard5');

INSERT INTO BOARD_HAS_LAYOUT (BoardID, brand, material, price, size, type, KeyboardName)
VALUES
(1, 'Logitech', 'Plastic', 100.00, 16.5, 'Full-size', 'keyboard1'),
(2, 'Corsair', 'Aluminum', 150.00, 14.0, 'TKL', 'keyboard2'),
(3, 'Razer', 'Plastic', 80.00, 18.0, 'Sixty', 'keyboard3'),
(4, 'SteelSeries', 'Metal', 120.00, 16.0, 'Fourty', 'keyboard4'),
(5, 'Ducky', 'PBT Plastic', 130.00, 15.0, 'Full-size', 'keyboard5');

INSERT INTO SWITCHES_ATTACH_TO (SwitchID, operating_force, brand, type, price, Lifespan,
BoardID, KeyID) VALUES
(1, '45g', 'Cherry', 'Linear', 15.00, 50000000, 1, ‘keycap1’),
(2, '50g', 'Gateron', 'Tactile', 14.00, 50000000, 2, ‘keycap2’),
(3, '55g', 'Kailh', 'Clicky', 14.50, 50000000, 3, ‘keycap3’),
(4, '60g', 'Outemu', 'Linear', 13.50, 50000000, 4, ‘keycap4’),
(5, '65g', 'ZealPC', 'Tactile', 16.00, 50000000, 5, ‘keycap5’);

INSERT INTO KEYCAP (KeyID, price, material, brand, SwitchID) VALUES
(‘keycap1’, 20.00, 'ABS Plastic', 'GMK', 1),
(‘keycap2’, 30.00, 'PBT Plastic', 'EnjoyPBT', 2),
(‘keycap3’, 20.50, 'POM Plastic', 'Akko', 3),
(‘keycap4’, 30.50, 'PBT Plastic', 'Ducky', 4),
(‘keycap5’, 15.00, 'ABS Plastic', 'Mistel', 5);
```