University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: June 7th

Group Number: 7

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Emma Park (Heayoung)	33281130	y6f0b	emma95@student.ubc.ca
Shumin Wang	70072111	g5o9x	shumin11@student.ubc.ca
Mingyue (Miranda) Tang	13159264	g0v3o	mtang78@student.ubc.ca

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

---

2. Brief summary of the project

Our project is to implement a comprehensive Lab Inventory Database to address the challenges faced by researchers in managing laboratory supplies and equipment. It focuses on tracking, organizing, and maintaining the stock of various items, supplies, and equipment used in laboratory operations. The application will significantly enhance the efficiency and effectiveness of researchers within the lab environment.

3. ER diagram

Please refer to the last page.

4. Schema derived from the ER diagram.

a. List the table definition (e.g., Table1(attr1: domain1, attr2: domain2, ...)). Make sure to include the domains for each attribute.

b. Specify the primary key (PK), candidate key, (CK) foreign keys (FK), and other constraints (e.g., not null, unique, etc.) that the table must maintain.

- Items(Full Name: char[20], Catalog number: integer, Description: char[100], Units: char[20], Quantity: integer)
- Chemicals(Expiry date: date, **Catalog number**: integer)
- Equipments(Maintenance frequency: char[20], **Catalog number**: integer)
- Keep(**ShelfID**: integer, **Number**: integer, **Building name**: char[20], **Catalog number**: integer, Date: date)
- Cabinet_In(ShelfID: integer, **Number**: integer, **Building name**: char[20])
- Room(Number: integer, Building name: char[20])
- Use(**Catalog number**: integer, **User ID**: char[20], Date: date)
- Lab Members(Name: char[20], Email: char[20], User ID: char[20], Phone: char[20])
- Involve(**User ID**: char[20], **ID:** integer, Enroll date: date)
- Lab(Name: char[20], ID: integer, Address: char[50])
- Lab Manager(Admin ID: char[20], Name: char[20], Email: char[20], Phone: char[20], **ID**: integer) (ID needs to be unique and not null)
- Chemical Waste_Dispose(Name: char[20], ID: integer, Description: char[200], **Admin ID**: char[20], Date: date) (Admin ID cannot be null)
- Vendors(Name: char[20], Email: char[20], Address: char[50], Phone: char[20])
- Purchase(**Catalog numbe**r: integer, **Admin ID**: char[20], **Name**: char[20], **Address: char[50]**, Date: date, Unit price: integer)

5. Functional Dependencies (FDs)

1. Items: Catalog Number -> Full Name, Quantity, Units, Description
2. Items: Full Name -> Units (violates 3NF)
3. Chemicals: Catalog Number -> Expiry date
4. Equipments: Catalog Number -> Maintenance frequency
5. Keep: ShelfID, Number, Building name, Catalog number -> Date
6. Use: Catalog number, User ID -> Date
7. Lab Members: User ID -> Name, Email, Phone
8. Involve: User ID, ID -> Enroll date
9. Lab: ID -> Name, Address
10. Lab Manager: Admin ID -> Name, Email, Phone, ID
11. Chemical Waste_Dispose: ID -> Name, Description, Admin ID, Date
12. Vendors: Name -> Email, Address, Phone
13. Purchase: Catalog number, Admin ID, Name -> Data, Unit price

6. 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.

You should show the steps taken for the decomposition. Should there be errors, and no work is shown, no partial credit can be awarded without steps shown. The format should be the same as Step 3, with tables listed similar to Table1(attr1:domain1, attr2:domain2, …). ALL Tables must be listed, not only the ones post normalization.

Solution:

We only have items that violate both 3NF and BCNF. So decomposition is done below:

Items: Catalog Number -> Full Name, Quantity, Units, Description

Items: Full Name -> Units

Minimal Cover:

Catalog Number -> Full Name;

Catalog Number ->Quantity;

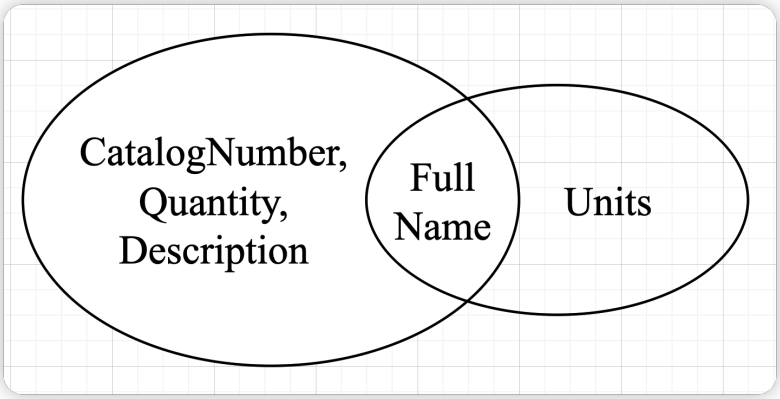
Catalog Number -> Units;

(redundant with Catalog Number -> Full Name + Full Name -> Units, so remove)

Catalog Number -> Description;

Full Name -> Units

Now we get the minimal cover, we will use lossless join method to get all decomposition.



R1 (FullName, Units)

R2 (CatalogNumber, FullName, Quantity, Description)

=> R3 (CatalogNumber, FullName)

=> R4 (CatalogNumber, Quantity)

=> R5 (CatalogNumber, Description)

Since CatalogNumber is the primary key in R2, and there's no other FDs in R2, we can then do optimization and combine R3, R4, R5 into one => R2 (CatalogNumber, FullName, Quantity, Description)

Now we split the relation "Items" into 2 smaller relations: "Items" and "ItemUnit"

To be specific:

- Items(Full Name: char[20], Catalog number: integer, Description: char[100], Quantity: integer)
- ItemUnit(Full Name: char[20], Units: char[20])

All the other relations should be the same as Question #4.

- Chemicals(Expiry date: date, **Catalog number**: integer)
- Equipments(Maintenance frequency: char[20], **Catalog number**: integer)
- Keep(**ShelfID**: integer, **Number**: integer, **Building name**: char[20], **Catalog number**: integer, Date: date)
- Cabinet_In(ShelfID: integer, **Number**: integer, **Building name**: char[20])
- Room(Number: integer, Building name: char[20])

- Use(**Catalog number**: integer, **User ID**: char[20], Date: date)
- Lab Members(Name: char[20], Email: char[20], User ID: char[20], Phone: char[20])
- Involve(**User ID**: char[20], **ID:** integer, Enroll date: date)
- Lab(Name: char[20], ID: integer, Address: char[50])
- Lab Manager(Admin ID: char[20], Name: char[20], Email: char[20], Phone: char[20], **ID**: integer) (ID needs to be unique and not null)
- Chemical Waste_Dispose(Name: char[20], ID: integer, Description: char[200], **Admin ID**: char[20], Date: date) (Admin ID cannot be null)
- Vendors(Name: char[20], Email: char[20], Address: char[50], Phone: char[20])
- Purchase(**Catalog numbe**r: integer, **Admin ID**: char[20], **Name**: char[20], **Address: char[50]**, Date: date, Unit price: integer)

7. 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.

Items(Full Name: char[20], Catalog number: integer, Description: char[100], Quantity: integer)

```
CREATE TABLE Items (
        Catalog number          INTEGER         PRIMARY KEY,
        Full Name               CHAR(20),
        Description             CHAR(100),
        Quantity                INTEGER
);
```

ItemUnit(Full Name: char[20], Units: char[20])

```
CREATE TABLE ItemUnit (
        Full Name               CHAR(20)        PRIMARY KEY,
        Unit                    CHAR(20)
);
```

Chemicals(Expiry date: date, **Catalog number**: integer)

```
CREATE TABLE Chemicals (
        Catalog number          INTEGER         PRIMARY KEY,
        Expiry date             DATE
);
```

Equipments(Maintenance frequency: char[20], **Catalog number**: integer)

```
CREATE TABLE Equipments (
        Catalog number          INTEGER         PRIMARY KEY,
        Maintenance frequency   CHAR(20)
);
```

Keep(**ShelfID**: integer, **Number**: integer, **Building name**: char[20], **Catalog number**: integer, Date: date)

```
CREATE TABLE Keep (
        ShelfID                 INTEGER
        Number                  INTEGER,
        Building name           CHAR(20),
        Catalog number          INTEGER,
        Date                    DATE,
        PRIMARY KEY             (ShelfID, Number, Building name, Catalog
                                number),
        FOREIGN KEY             (SelfID, Number, Building name)
            REFERENCES          Cabinet_In(ShelfID, Number, Building Name),
        FOREIGN KEY             (Catalog number)
            REFERENCES          Items(Catalog number)
```

);

Cabinet_In(ShelfID: integer, **Number**: integer, **Building name**: char[20])

```
CREATE TABLE Cabinet_In (
        ShelfID                 INTEGER,
        Number                  INTEGER,
        Building name           char(20),
        PRIMARY KEY             (ShelfID, Number, Building name),
        FOREIGN KEY             (Number, Building name)
            REFERENCES          Room(Number, Building Name)
);
```

Room(Number: integer, Building name: char[20])

```
CREATE TABLE Room (
        Number                  INTEGER,
        Building name           char(20),
        PRIMARY KEY             (Number, Building name)
);
```

Use(**Catalog number**: integer, **User ID**: char[20], Date: date)

```
CREATE TABLE Use (
        Catalog number          INTEGER,
        User ID                 char(20),
        Date                    DATE,
        PRIMARY KEY             (Catalog number, User ID),
        FOREIGN KEY             (Catalog number)
```

```
        REFERENCES          Items(Catalog number),
    FOREIGN KEY             (User ID)
        REFERENCES          Lab members(User ID)
);
```

Lab Members(Name: char[20], Email: char[20], User ID: char[20], Phone: char[20])

```
CREATE TABLE Items (
    User ID                 char(20)            PRIMARY KEY,
    Name                    char(20),
    Email                   char(20),
    Phone                   char(20)
);
```

Involve(**User ID**: char(20), **ID:** integer, Enroll date: date)

```
CREATE TABLE Involve (
    Use ID                  char(20),
    ID                      INTEGER,
    Enroll date             DATE,
    PRIMARY KEY             (User ID, ID),
    FOREIGN KEY             (User ID)
        REFERENCES          Lab member(User ID),
    FOREIGN KEY             (ID)
        REFERENCES          Lab(ID)
);
```

Lab(Name: char[20], ID: integer, Address: char[50])

```
CREATE TABLE Lab (
```

```
        ID                  INTEGER         PRIMARY KEY,
        Name                char(20),
        Address             char(50)
);
```

Lab Manager(Admin ID: integer, Name: char[20], Email: char[20], Phone: char[20], **ID**: integer) (ID needs to be unique and not null)

```
CREATE TABLE Lab Manager (
        Admin ID            INTEGER         PRIMARY KEY,
        Name                char(20),
        Email               char(20),
        Phone               char(20),
        ID                  INTEGER,
        FOREIGN KEY         (ID)
            REFERENCES      Lab(ID)
);
```

Chemical Waste_Dispose(Name: char[20], ID: integer, Description: char[200], **Admin ID**: integer, Date: date) (Admin ID cannot be null)

```
CREATE TABLE Chemical Waste_Dispose(
        ID                  INTEGER             PRIMARY KEY,
        Name                char(20),
        Description         char(200),
        Admin ID            INTEGER,
        Date                DATE,
        FOREIGN KEY         (Admin ID)
            REFERENCES      Lab Manager(Admin ID)
```

);

Vendors(Name: char[20], Email: char[20], Address: char[50], Phone: char[20])

```
CREATE TABLE Vendors (
        Name                    char(20),
        Email                   char(20),
        Address                 char(50),
        Phone                   char(20),
        PRIMARY KEY             (Name, Address)
);
```

Purchase(**Catalog numbe**r: integer, **Admin ID**: integer, **Name**: char[20], **Address**: char[50], Date: date, Unit price: integer)

```
CREATE TABLE Items (
        Catalog number          INTEGER,
        Admin ID                INTEGER,
        Name                    char(20),
        Address                 char(50),
        Date                    DATE
        Unit price              INTEGER
        PRIMARY KEY             (Catalog number, Admin ID, Name, Address),
        FOREIGN KEY             (Catalog number)
            REFERENCES          Items(Catalog number),
        FOREIGN KEY             (Admin ID)
            REFERENCES          Lab Manager(Admin ID),
        FOREIGN KEY             (Name, Address)
```

```
        REFERENCES          Vendor(Name, Address)
);
```

8. INSERT statements to populate each table with at least 5 tuples. You will likely want to have more than 5 tuples so that you can have meaningful queries later on.

```
-- Insert into Items
INSERT INTO Items (Catalog_number, Full_Name, Description, Quantity)
VALUES (1001, 'Chemical A', 'Organic compound used for experiments', 20),
       (1002, 'Chemical B', 'Inorganic salt for laboratory use', 50),
       (1003, 'Equipment A', 'Microscope with high-resolution optics', 5),
       (1004, 'Equipment B', 'Centrifuge for sample separation', 2),
       (1005, 'Glassware A', 'Glass beakers for various volumes', 30);

-- Insert into ItemUnit
INSERT INTO ItemUnit (Full_Name, Units)
VALUES ('Chemical A', 'grams'),
       ('Chemical B', 'grams'),
       ('Equipment A', 'units'),
       ('Equipment B', 'units'),
       ('Glassware A', 'pieces');

-- Insert into Chemicals
INSERT INTO Chemicals (Catalog_number, Expiry_date)
VALUES (1001, '2024-12-31'),
       (1002, '2023-09-15'),
       (1006, '2025-06-30'),
       (1007, '2023-11-30'),
       (1010, '2024-08-01');

-- Insert into Equipments
INSERT INTO Equipments (Catalog_number, Maintenance_frequency)
VALUES (1003, 'Monthly'),
       (1004, 'Quarterly'),
```

```
        (1008, 'Annual'),
        (1009, 'Biennial'),
        (1011, 'Monthly');

-- Insert into Keep
INSERT INTO Keep (ShelfID, Number, Building_name, Catalog_number, Date)
VALUES (1, 1, 'Building A', 1001, '2023-06-01'),
        (2, 3, 'Building B', 1002, '2023-06-02'),
        (1, 2, 'Building A', 1005, '2023-06-05'),
        (3, 4, 'Building C', 1003, '2023-06-06'),
        (2, 1, 'Building B', 1004, '2023-06-07');

-- Insert into Cabinet_In
INSERT INTO Cabinet_In (ShelfID, Number, Building_name)
VALUES (1, 1, 'Building A'),
        (2, 2, 'Building B'),
        (3, 1, 'Building C'),
        (2, 3, 'Building B'),
        (1, 2, 'Building A');

-- Insert into Room
INSERT INTO Room (Number, Building_name)
VALUES (1, 'Building A'),
        (2, 'Building B'),
        (3, 'Building C'),
        (4, 'Building D'),
        (5, 'Building E');

-- Insert into Use
INSERT INTO Use (Catalog_number, User_ID, Date)
VALUES (1001, 'user1', '2023-06-03'),
        (1002, 'user2', '2023-06-04'),
        (1003, 'user3', '2023-06-05'),
        (1004, 'user4', '2023-06-06'),
        (1005, 'user5', '2023-06-07');

-- Insert into Lab_Members
```

```
INSERT INTO Lab_Members (User_ID, Name, Email, Phone)
VALUES ('user1', 'John Smith', 'john.smith@example.com', '123-456-7890'),
       ('user2', 'Jane Doe', 'jane.doe@example.com', '234-567-8901'),
       ('user3', 'Robert Johnson', 'robert.johnson@example.com', '345-678-9012'),
       ('user4', 'Emily Wilson', 'emily.wilson@example.com', '456-789-0123'),
       ('user5', 'Michael Brown', 'michael.brown@example.com', '567-890-1234');

INSERT INTO Involve (User_ID, ID, Enroll_date)
VALUES ('user1', 1, '2022-01-01'),
       ('user2', 1, '2022-02-15'),
       ('user3', 2, '2022-03-10'),
       ('user4', 2, '2022-04-20'),
       ('user5', 3, '2022-05-05');

-- Insert into Lab
INSERT INTO Lab (ID, Name, Address)
VALUES (1, 'Lab 1', 'Building A, Floor 1'),
       (2, 'Lab 2', 'Building B, Floor 2'),
       (3, 'Lab 3', 'Building C, Floor 3'),
       (4, 'Lab 4', 'Building D, Floor 4'),
       (5, 'Lab 5', 'Building E, Floor 5');

-- Insert into Lab_Manager
INSERT INTO Lab_Manager (Admin_ID, Name, Email, Phone, ID)
VALUES ('admin1', 'Jane Doe', 'jane.doe@example.com', '987-654-3210', 1),
       ('admin2', 'Mark Johnson', 'mark.johnson@example.com', '456-789-1230', 2),
       ('admin3', 'Emily Smith', 'emily.smith@example.com', '789-123-4560', 3),
       ('admin4', 'Michael Brown', 'michael.brown@example.com', '321-654-9870', 4),
       ('admin5', 'Sophia Davis', 'sophia.davis@example.com', '654-321-9870', 5);

-- Insert into Chemical_Waste_Dispose
INSERT INTO Chemical_Waste_Dispose (Name, ID, Description, Admin_ID, Date)
VALUES ('Waste A', 1, 'Hazardous waste from experiments', 'admin1', '2023-06-04'),
       ('Waste B', 2, 'Chemical waste for proper disposal', 'admin2', '2023-06-05'),
       ('Waste C', 3, 'Expired chemicals for safe disposal', 'admin3', '2023-06-06'),
       ('Waste D', 4, 'Biohazard waste from biological experiments', 'admin4', '2023-06-07'),
       ('Waste E', 5, 'Toxic waste for specialized treatment', 'admin5', '2023-06-08');
```

-- Insert into Vendors

```
INSERT INTO Vendors (Name, Email, Address, Phone)
VALUES ('Vendor A', 'vendorA@example.com', '123 Main Street', '111-111-1111'),
       ('Vendor B', 'vendorB@example.com', '456 Elm Street', '222-222-2222'),
       ('Vendor C', 'vendorC@example.com', '789 Oak Street', '333-333-3333'),
       ('Vendor D', 'vendorD@example.com', '321 Pine Street', '444-444-4444'),
       ('Vendor E', 'vendorE@example.com', '654 Maple Street', '555-555-5555');
```