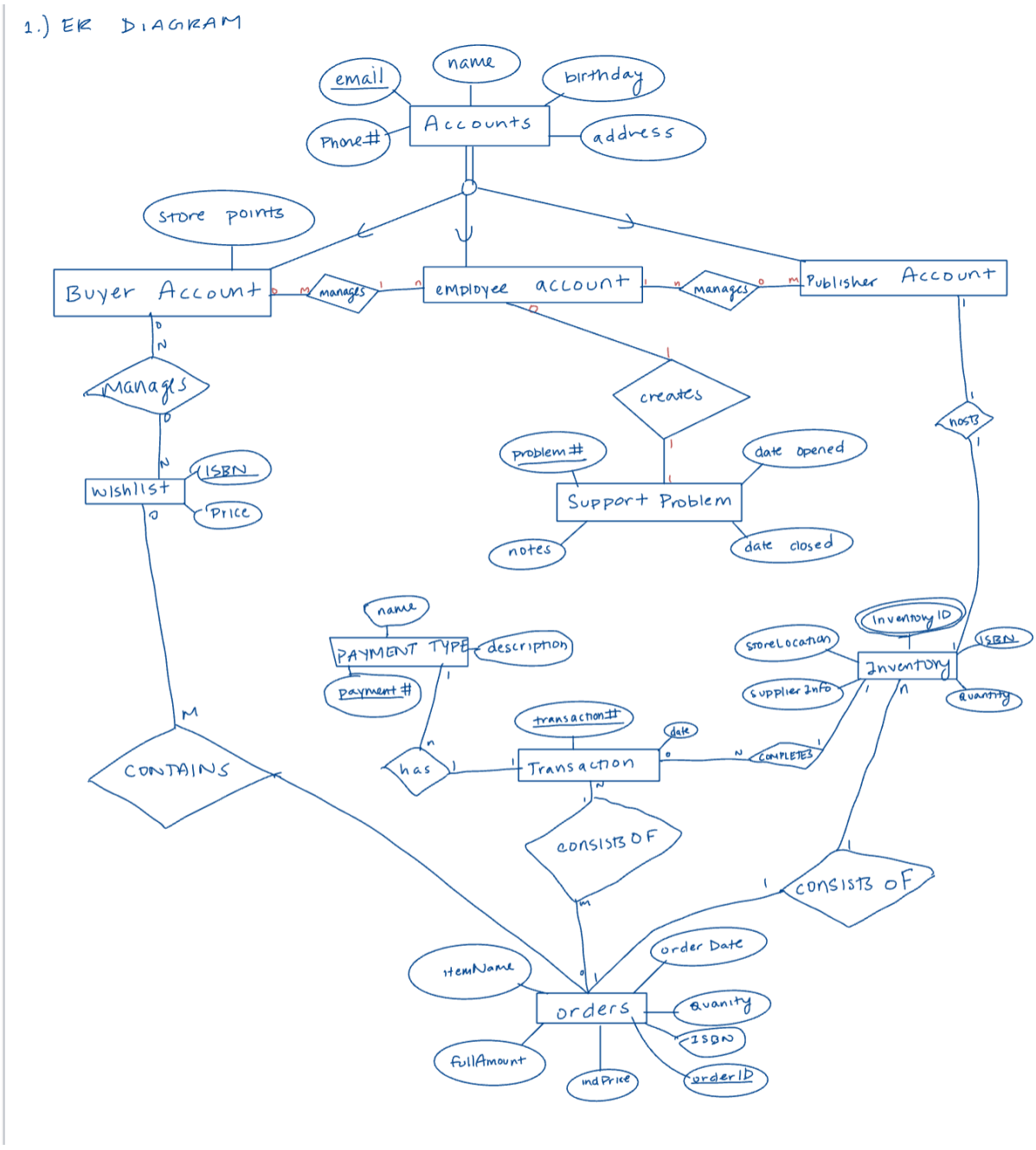
CSE 3241 Project Checkpoint 04 – Functional Dependencies and Normal Forms

Names	Date
Anmol Kumar and Jay Ramesh	

In a **NEATLY TYPED** document, provide the following:

1. Provide a current version of your ER Diagram and Relational Model as per Project Checkpoint 03. **If you were instructed to change the model for Project Checkpoint 03, make sure you use the revised versions of your models.**

1.) ER DIAGRAM



2. For each relation schema in your model, indicate the functional dependencies. Think carefully about what you are modeling here - make sure you consider all the possible dependencies in each relation and not just the ones from your primary keys. For example, a customer's credit card number is unique, and so will uniquely identify a customer even if you have another key in the same table (in fact, if the customer can have multiple credit card numbers, the dependencies can get even more involved).

3. For each relation schema in your model, determine the highest normal form of the relation. If the relation is not in 3NF, rewrite your relation schema so that it is in at least 3NF.

ACCOUNT
(Account) = {Email, Name, Birthday, PhoneNumber, Address}
Primary Key = Email

3NF = {Email → Name, Birthday, PhoneNumber, Address}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key. Under the assumption that the phone number and address can have multiple accounts.

AUTHORS
(Authors) = {ISBN, Authors_Names}
Primary Key = Combination of both ISBN and Authors_Names

3NF = {ISBN, Authors_Names}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Since there are no non-key attributes, there are no transitive dependencies.

BUYER_ACCOUNT
(Buyer_Account) = {Buyer_Email, Name, Birthday, PhoneNumber, Address, StorePoints}
Primary Key = Buyer_Email

3NF = {Buyer_Email→ Name, Birthday, PhoneNumber, Address, StorePoints}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key. Under the assumption that the phone number and address can have multiple accounts.

BUYER_MANAGES_WISHLIST
(Buyer_Manages_WishList) = {Buyer_Email, Book_ISBN, Book_Price}
Primary Key = Buyer_Email, Book_ISBN

3NF = {Buyer_Email, Book_ISBN → Book_Price}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies.

Every non-key trait/attribute depends on a singular superkey and not another non-key. Under the assumption that the combination of the Buyer's email and the book ISBN will provide a unique identifier key.

BUYER_RECEIPT
(Buyer_Receipt) = {Order ID, Transaction_ID, Receipt_Data, Transaction_Total}
Primary Key = Order_ID, Transaction_ID

3NF = {Order ID, Transaction_ID → Receipt_Data, Transaction_Total}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key. Each Order_ID is unique to each order that occurs.

CATALOG
(Catalog) = {ISBN, Title, Publisher, Year, Price, Category}
Primary Key = ISBN

3NF = {ISBN → Title, Publisher, Year, Price, Category}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key.

CUSTOMER_SERVICE
(Customer_Service) = {Case_Number, Account_Email, Date, Problem_Description, Employee_In_Charge}
Primary Key = Case_Number

3NF = {Case_Number → Account_Email, Date, Problem_Description, Employee_In_Charge}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key. Under the assumption that an account can have multiple problems, and an employee can be in charge of multiple accounts

EMPLOYEE_ACCOUNT
(Employee_Account) = {Email, Name, Birthday, PhoneNumber, Address}
Primary Key = Email

3NF = {Emaill → Birthday, PhoneNumber, Address}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key. Under the assumption that the phone number and address can have multiple accounts.

EMPLOYEE_MANAGES_BUYER
(Employee_Manages_Buyer) = {Employee_Email, Buyer_Email}
Primary Key = Combination of both Employee_Email and Buyer_Email

3NF = {Employee_Email, Buyer_Email}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Since there are no non-key attributes, there are no transitive dependencies.

EMPLOYEE_MANAGES_SELLER
(Employee_Manages_Seller) = {Employee_Email, Seller_Email}
Primary Key = Combination of both Employee_Email and Seller_Email

3NF = {Employee_Email,Seller_Email}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Since there are no non-key attributes, there are no transitive dependencies.

INVENTORY
(Inventory) = {ISBN, Inv_Quanity}
Primary Key = ISBN

3NF = {ISBN → Inv_Quanity}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key.

ORDER_DETAILS
(Order_Details) = {Order_ID,, Order_Date, Book_Title, Book_Price}
Primary Key = Order_ID, Book_ISBN

3NF = {Order_ID, Book_ISBN→ Order_Date, Book_Title, Book_Price}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular combination superkey and not another non-key. Each order ID coupled with each individual book ISBN purchased creates a unique identifying primary key.

PAYMENT_STATUS
(Payment_Status) = {Payment_ID, Order_ID, Approval_Status, Approval_Date}
Primary Key = Payment_ID

3NF = {Payment_ID → Order_ID, Approval_Status, Approval_Date}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key

PAYMENT_TYPE
(Payment_Type) = {Payment_ID, Name, Method}
Primary Key = Payment_ID

3NF = {Payment_ID → Name, Method}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key

SELLER_ACCOUNT
(Seller_Account) = {Email, Name, Birthday, PhoneNumber, Address}
Primary Key = Email

3NF = {Emaill → Birthday, PhoneNumber, Address}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key. Under the assumption that the phone number and address can have multiple accounts.

SELLER_UPDATES_CATALOG
(Seller_Updates_Catalog) = {Seller_Email, ISBN, Title, Publisher, Year, Price, Category}
Primary Key = Combination of both Seller_Email and ISBN

3NF = {Seller_Email, ISBN, → Title, Publisher, Year, Price, Category}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular combination superkey and not another non-key. Seller_Email and ISBN, together, create the primary key that keeps this table in 3NF.

TRANSACTION_DETAILS
(Transaction_Details) = {Transaction_ID, Transaction_Date, Order_ID, Transaction_Total, Buyer_Details, Payment_Approved}
Primary Key = Tranaction_ID

3NF = {Transaction_ID → Transaction_Date, Order_ID, Transaction_Total, Buyer_Details, Payment_Approved}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular superkey and not another non-key.

WISHLIST
(Wishlist) = {Email, ISBN, Title, Price, Date_Added}
Primary Key = Combination of both Email and ISBN

3NF = {Email, ISBN → Title, Price, Date_Added}

This table is already in 1NF since all keys/attributes are atomic. Additionally, the table is already in 2NF since there is a composite primary/candidate key. This table is in 3NF because it has no transitive dependencies. Every non-key trait/attribute depends on a singular combination superkey and not another non-key. Email and ISBN, together, create the primary key that keeps this table in 3NF.

4. For each relation schema in your model that is in 3NF but not in BCNF, either rewrite the relation schema to BCNF or provide a short justification for why this relation should be an exception to the rule of putting relations into BCNF.

    a. Don't need to provide BCNF

5. For your database, propose at least two interesting views that can be built from your relations. These views must involve joining at least two tables together each and must include some kind of aggregation in the view. Each view must also be able to be described by a one or two sentence description in plain English. Provide the code for constructing your views along with the English language description of what the view is supposed to be providing.
    a. FULL CATALOG – Displays all the details of each book
        i. ISBN
        ii. Title
        iii. Author
        iv. Publisher
        v. Year
        vi. Price
        vii. Category/Genre
    b. Wishlist Cart – Displays if books in Wishlist are in stock or not

    On the next page is the code for each of the views

6.
```
/* Display a Full Catalog
   the ISBN, Title, Authors, Publisher, and Price of all books */
CREATE VIEW FULL_CATALOG AS
SELECT CATALOG.ISBN, CATALOG.Title, AUTHORS.Authors_Names, CATALOG.Publisher,
CATALOG.Price
FROM CATALOG
INNER JOIN AUTHORS ON CATALOG.ISBN = AUTHORS.ISBN;

/* Display ONE_WISH
   Customer Email, Name, Items in their Wishlist
 */
CREATE VIEW ONE_WISH AS
SELECT BUYER_ACCOUNT.Name, BUYER_ACCOUNT.Email, WISHLIST.Title,
INVENTORY.Inv_Quantity
FROM WISHLIST
INNER JOIN INVENTORY ON WISHLIST.ISBN = INVENTORY.ISBN
INNER JOIN BUYER_ACCOUNT ON WISHLIST.Email = BUYER_ACCOUNT.Email;
```