# CSE 3241 Project Checkpoint 04 – Functional Dependencies and Normal Forms

Names   Cenglin Bao, Wen Fang, Patrick Flanagan, Kai Li   Date 4/5/2020

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.**
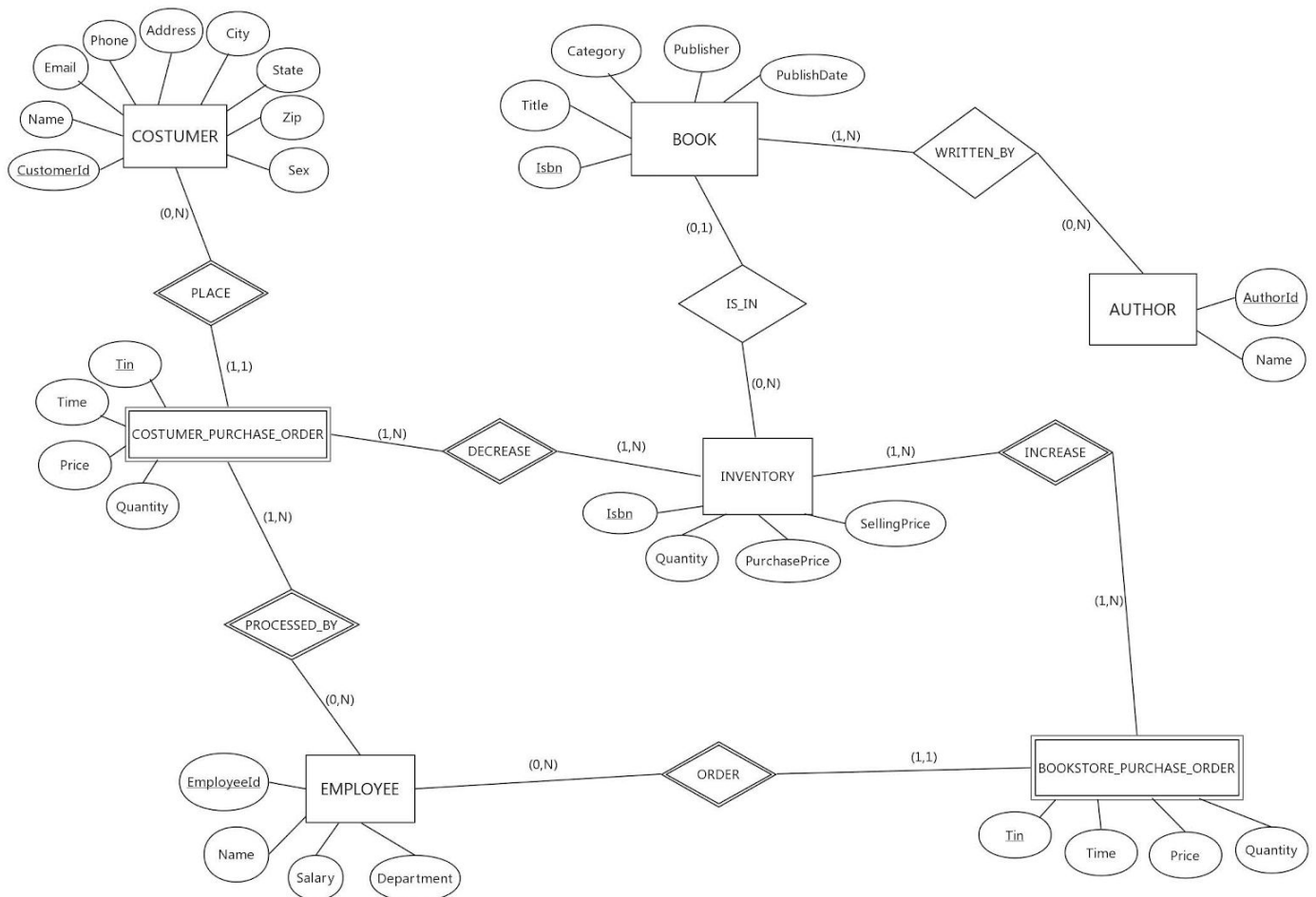
Note: Primary keys are underlined, foreign keys are colored red.
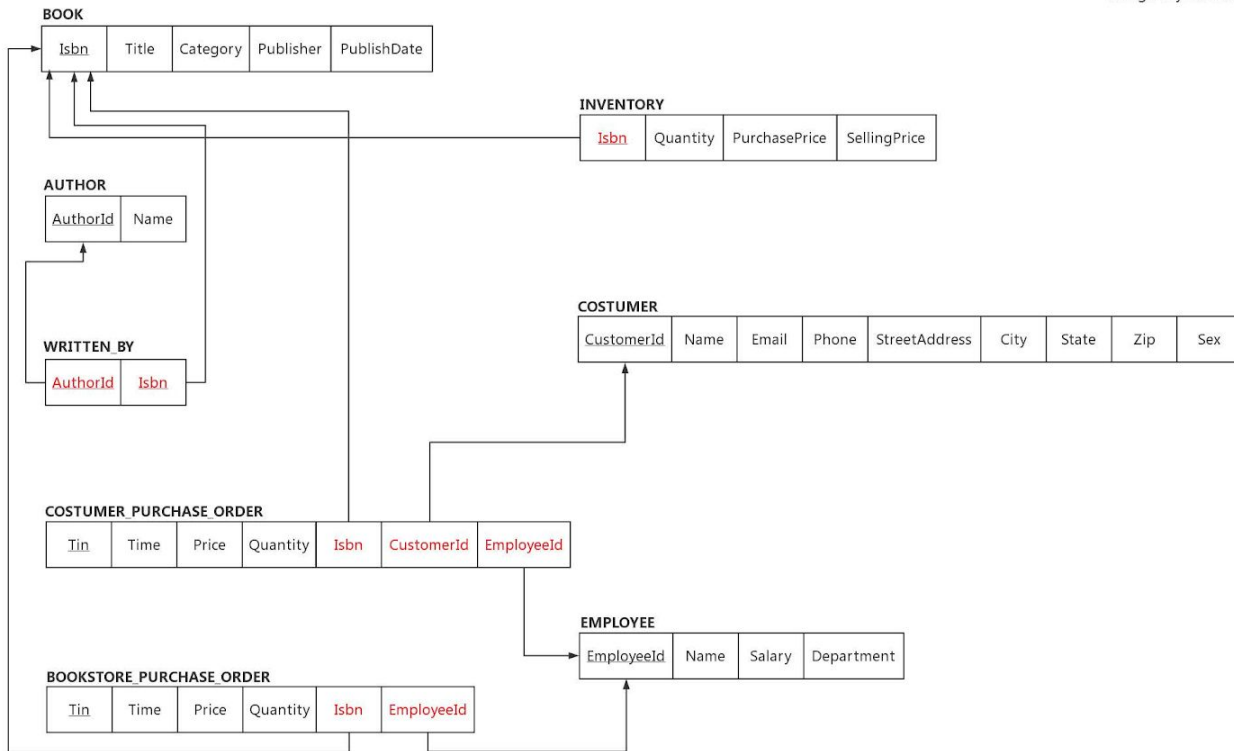
2. For each relation schema in your model, indicate the functional dependencies. Think carefully about what you are modeling here - make sure you consider all the possible dependencies in each relation and not just the ones from your primary keys. For example, a customer's credit card number is unique, 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).

BOOK
{Isbn} → {Title, Category, Publisher, PublishDate}

AUTHOR
{AuthorId} → {Name}

WRITTEN_BY
{AuthorID, Name}

EMPLOYEE
{EmployeeId} → {Name, Salary, Department}

INVENTORY
{Isbn} → {Quantity, PurchasePrice, SellingPrice}

CUSTOMER

{CustomerId} → {Name, Email, Phone, StreetAddress, City, State, Zip, Sex}

COSTUMER_PURCHASE_ORDER
{Tin} → {Time, Price, Quantity, Isbn, CustomerId, EmployeeId}

BOOKSTORE_PURCHASE_ORDER
{Tin} → {Time, Price, Quantity, Isbn, EmployeeId}

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.

BOOK: BCNF
AUTHOR: BCNF
WRITTEN_BY: BCNF
EMPLOYEE: BCNF
INVENTORY: BCNF
CUSTOMER: BCNF
COSTUMER_PURCHASE_ORDER: BCNF
BOOKSTORE_PURCHASE_ORDER: BCNF

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.

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.

(1) Show the Isbn, title, and total number of sold copies of books that were sold over 1000 copies

CREATE VIEW BOOKS_SOLD_OVER_1000

AS   SELECT B.Isbn, B.Title, sum(P.Quantity)

        FROM BOOK B, CUSTOMER_PURCHASE_ORDER P

        WHERE B.Isbn = P.Isbn

        GROUP BY B.Isbn, B.Title

        HAVING sum(P.Quantity) > 1000;

(2) Provide the list of Customer who purchased more than 20 books

CREATE VIEW CUSTOMERS_FROM

AS   SELECT C.CustomerId, C.Name, sum(P.quantity)

FROM CUSTOMER C, CUSTOMER_PURCHASE_ORDER P

WHERE C.CustomerId = P.CustomerId

GROUP BY C.CustomerId, C.Name

HAVING sum(P.Quantity) > 20;

(3) Provide the avg salary of Departments sold more than 1000 books

CREATE VIEW AVG_SALARY_DEPARTMENT
AS   SELECT E.DEPARTMENT, avg(E.Salary)

FROM EMPLOYEE E, CUSTOMER_PURCHASE_ORDER P

WHERE E.EmployeeId = P.EmployeeId

GROUP BY E.Department

HAVING sum(P.Quantity) > 1000;