1: The First Problem

For table CUSTOMER:

customer-code → name, customer-code → address, customer-code → order-id
order-id → customer-code

For table EMPLOYEE:

employee-number → name, employee-number → department-name, employee-number → department-budget

For table ORDER:

order-id → order-date, order-id → total-value, order-id → customer-code, order-id → employeenumber, order-id → employee-name
(customer-code, employee-number) → order-id

2: The second problem

For table CUSTOMER:

customer-code → (name,address, order-id) is valid in table and it covers entire relational schema then its left hand side is a minimal key (customer-code)

order-id → customer-code and customer-code → (name, address, order-id) then hrough transitivity rule order-id → (name, address, order-id)

order-id → (name,address, order-id) then through augmentation rule (order-id, customer-code) → (customer-code, name,address, order-id)

(order-id, customer-code) → (customer-code, name,address, order-id) is valid in table and it covers entire relational schema then its left hand side is a minimal key (order-id, customer-code)

For table EMPLOYEE:

employee-number → (name, department-name, department-budget) is valid in table and it covers entire relational schema then its left hand side is a minimal key (employee-number)

For table ORDER:

order-id → (order-date, total-value, customer-code, employee-number, employee-name) is valid in table and it covers entire relational schema then its left hand side is a minimal key (order-id)

(customer-code, employee-number) → order-id and order-id → (order-date, total-value, customer-code, employee-number, employee-name) then through transitivity rule (customer-code, employee-number) → (order-date, total-value, customer-code, employee-number, employee-name)

(customer-code, employee-number) → (order-date, total-value, customer-code, employee-number, employee-name) then through augmentation rule (customer-code, employee-number, order-id) → (order-id, order-date, total-value, customer-code, employee-number, employee-name)

(customer-code, employee-number, order-id) → (order-id, order-date, total-value, customer-code, employee-number, employee-name) is valid in table and it covers entire relational schema then its left hand side is a minimal key (customer-code, employee-number, order-id)

3: The third problem

For table CUSTOMER:

the left side of (order-id, customer-code) → (customer-code, name,address, order-id) is not a superkey, but customer-code is a prime attribute, so the schema is in 3NF.

For table EMPLOYEE:

A relational schema employee-number → (name, department-name, department-budget) is in BCNF because does not exist a functional dependency whose left hand side is not a superkey

For table ORDER:

the left side of (customer-code, employee-number, order-id) → (order-id, order-date, total-value, customer-code, employee-number, employee-name) is not asuperkey, but order-id is a prime attribute, so the schema is in 3NF

4: The fourth problem

For table CUSTOMER: Use order-id and customer-code as joint primary keys:

(order-id, customer-code) → (name, address, order-id)

For table ORDER: Use order-id and customer-code as joint primary keys:

(customer-code, order-id) → (order-date, total-value, customer-code, employee-number, employee-name)