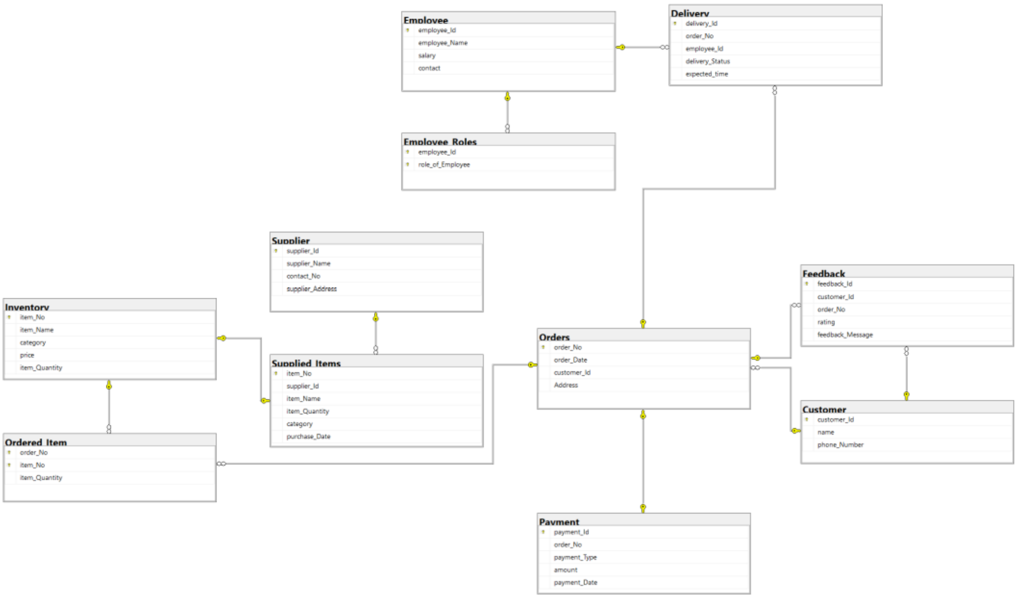
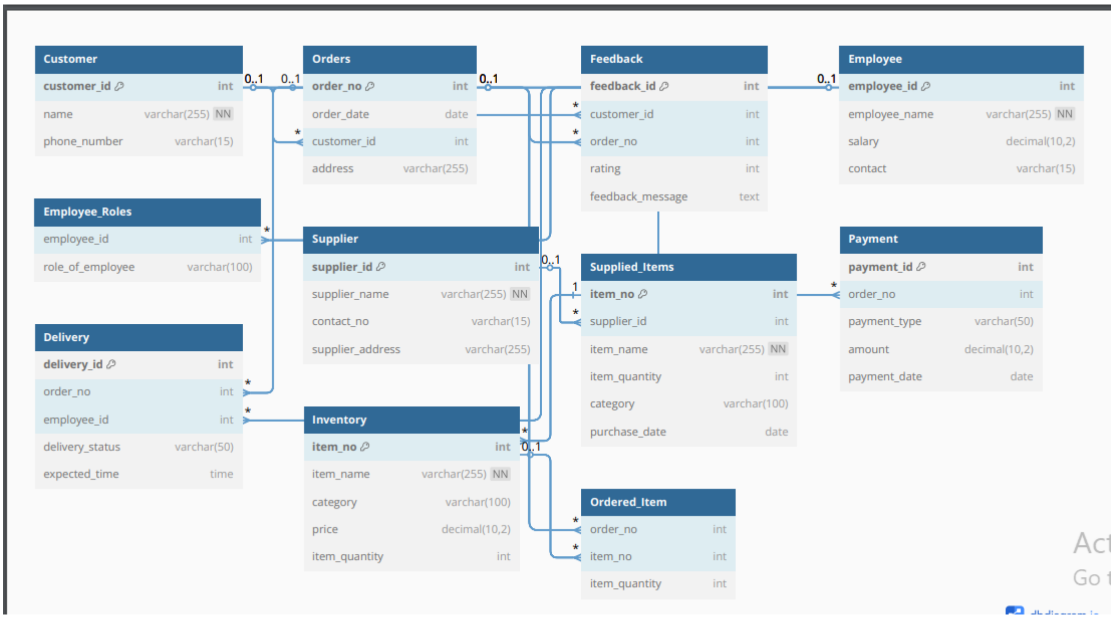
Database Deliverable - IV

Submitted To: *Sir Durraiz Waseem*

Submitted By:
Hamza Naveed
Huzaifa Shabbir
Hamda Saeed

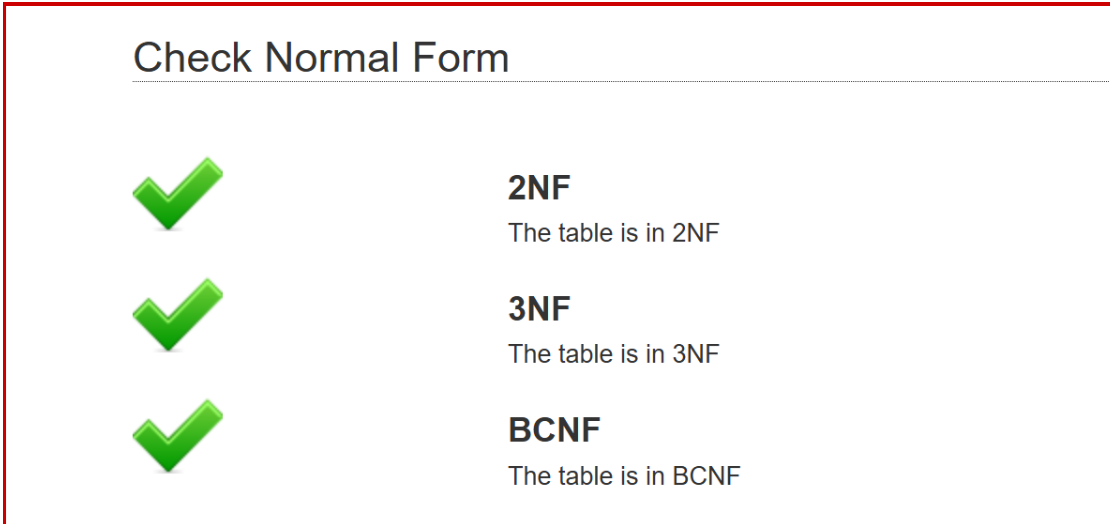
- **Initial Database Diagram from Deliverable-II**





- # Normalization Steps with FD’s

Table 1:
Customer Table
Attributes:
customer_Id,name,phone_Number
FD’s:
customer_Id-> name
customer_Id-> phone_Number
phone_Number -> customer_Id
phone_Number -> name
CK:customer_Id , phone_Number

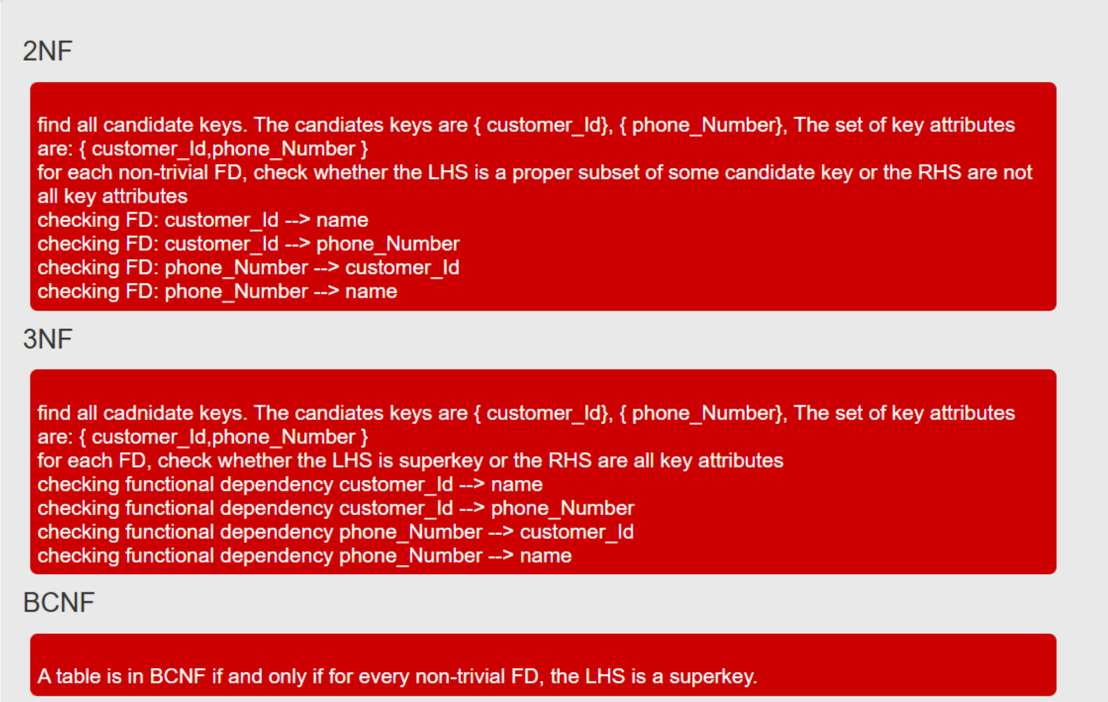


Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF



2NF

find all candidate keys. The candiates keys are { customer_Id}, { phone_Number}, The set of key attributes are: { customer_Id,phone_Number }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: customer_Id --> name
checking FD: customer_Id --> phone_Number
checking FD: phone_Number --> customer_Id
checking FD: phone_Number --> name

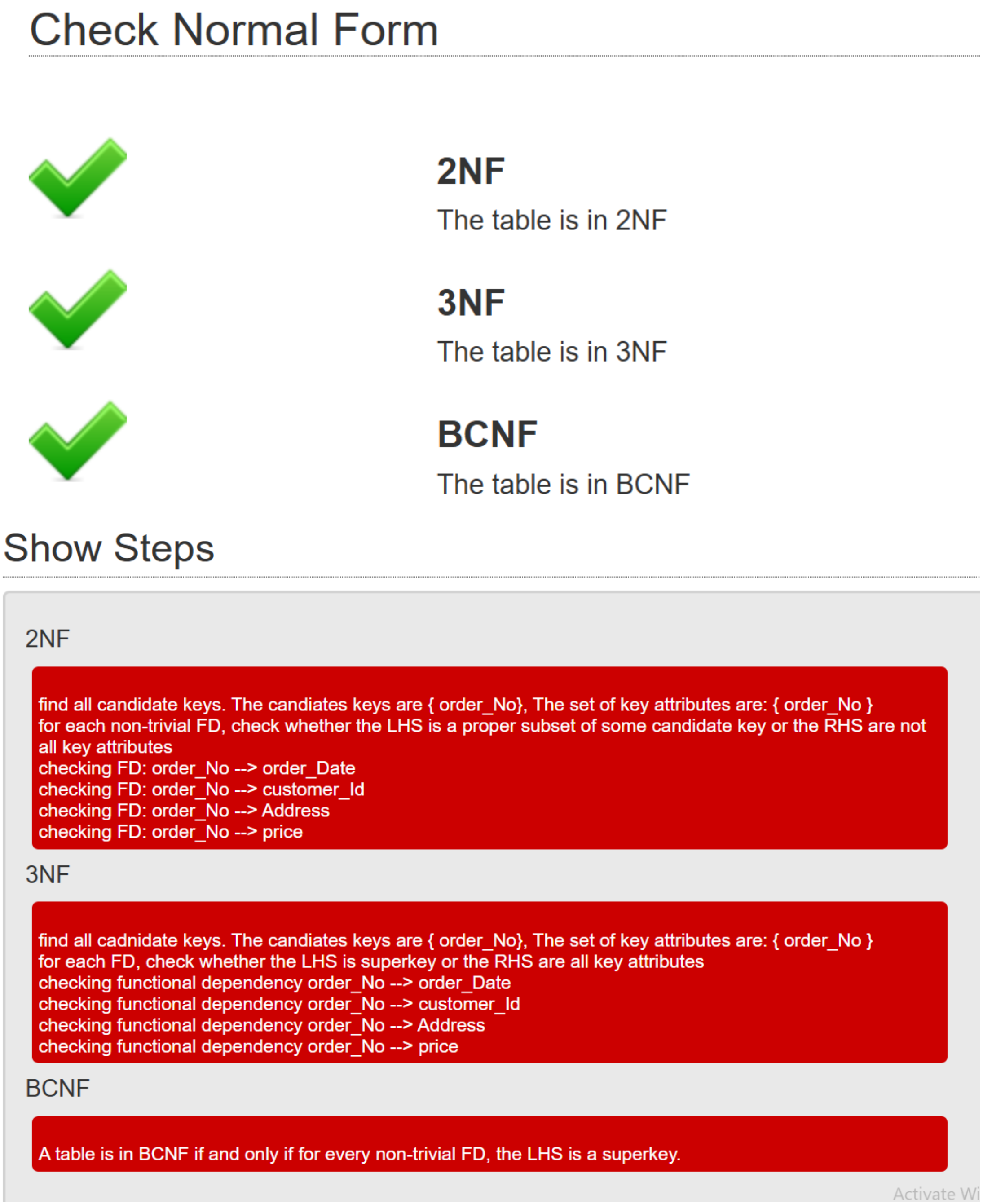
3NF

find all cadnidate keys. The candiates keys are { customer_Id}, { phone_Number}, The set of key attributes are: { customer_Id,phone_Number }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency customer_Id --> name
checking functional dependency customer_Id --> phone_Number
checking functional dependency phone_Number --> customer_Id
checking functional dependency phone_Number --> name

BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

Table 2:
Order Table
Attributes:
order_No,order_Date,customer_Id,Address,price
FD’s:
order_No -> order_Date
order_No -> customer_Id
order_No -> Address
order_No -> price



Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF

Show Steps

2NF

find all candidate keys. The candiates keys are { order_No}, The set of key attributes are: { order_No }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: order_No --> order_Date
checking FD: order_No --> customer_Id
checking FD: order_No --> Address
checking FD: order_No --> price

3NF

find all cadnidate keys. The candiates keys are { order_No}, The set of key attributes are: { order_No }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency order_No --> order_Date
checking functional dependency order_No --> customer_Id
checking functional dependency order_No --> Address
checking functional dependency order_No --> price

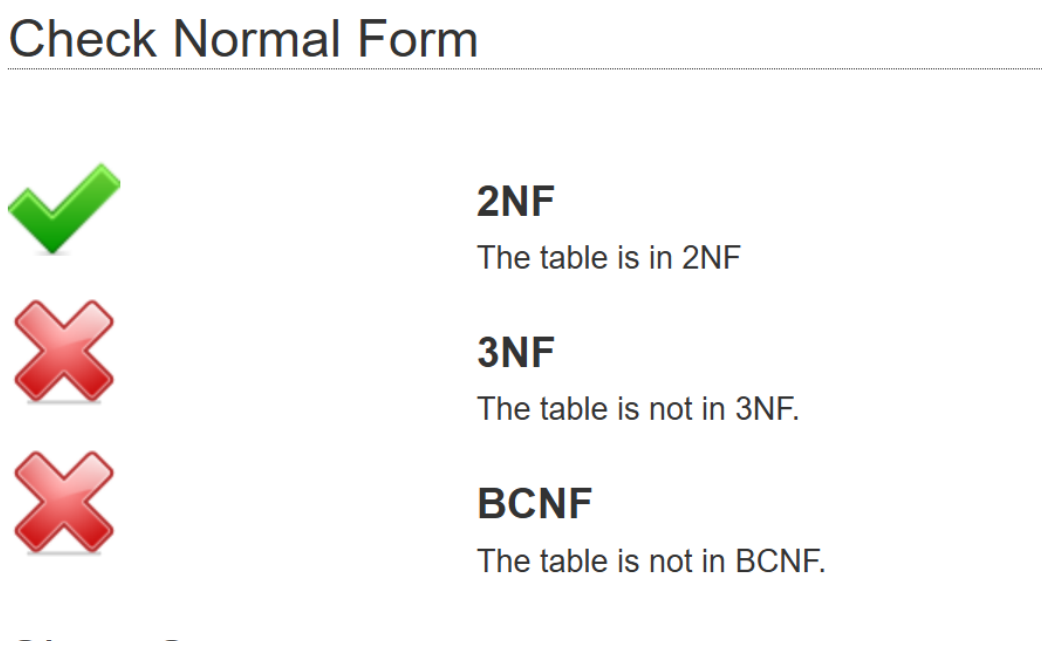
BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

Table 3:
FeedBack Table:
Attributes:
feedback_Id,customer_Id,order_No,rating,feedback_Message
FD’s:
feedback_Id -> customer_Id
feedback_Id -> order_No
feedback_Id -> rating
feedback_Id -> feedback_Message
(customer_Id,order_No)-> rating
(customer_Id,order_No)-> feedback_Message
order_No -> customer_Id

Candidate Keys Found

- feedback_Id

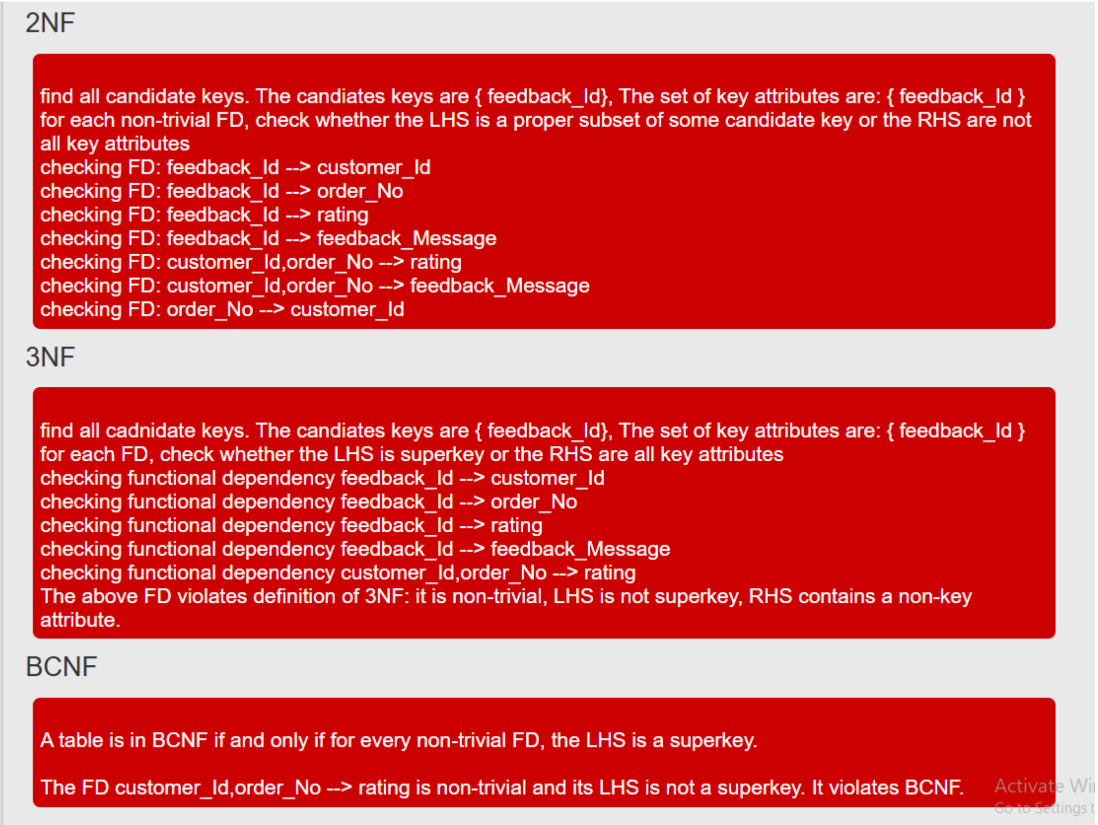


Check Normal Form

✓ **2NF**
The table is in 2NF

✗ **3NF**
The table is not in 3NF.

✗ **BCNF**
The table is not in BCNF.



2NF

find all candidate keys. The candiates keys are { feedback_Id}, The set of key attributes are: { feedback_Id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: feedback_Id --> customer_Id
checking FD: feedback_Id --> order_No
checking FD: feedback_Id --> rating
checking FD: feedback_Id --> feedback_Message
checking FD: customer_Id,order_No --> rating
checking FD: customer_Id,order_No --> feedback_Message
checking FD: order_No --> customer_Id

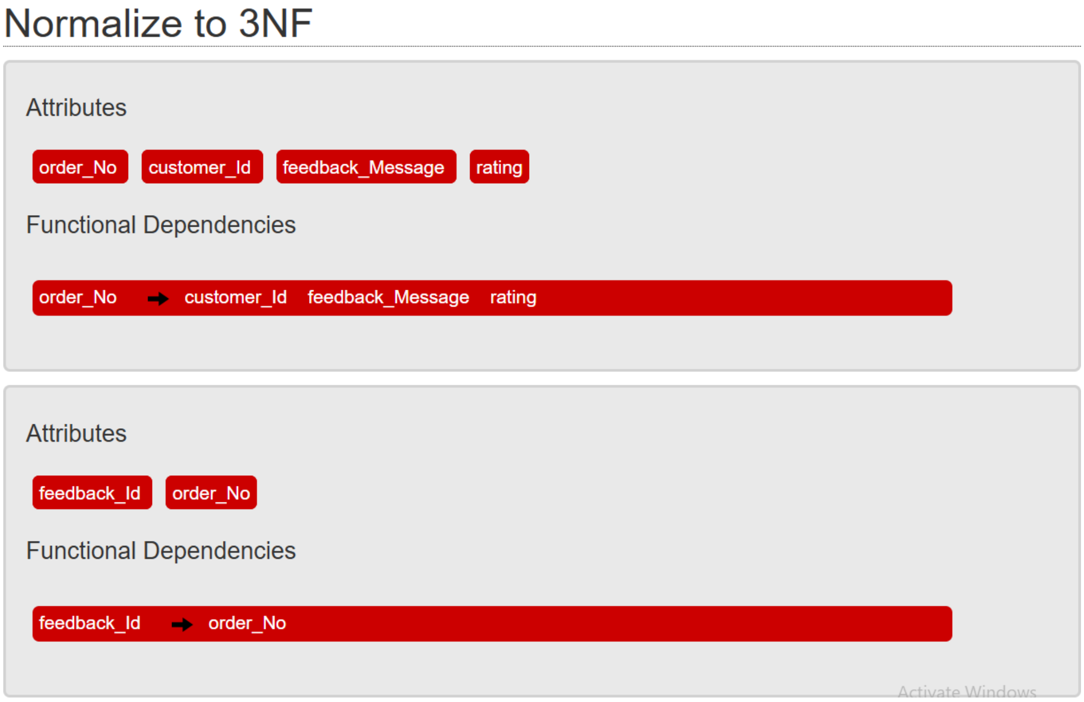
3NF

find all cadnidate keys. The candiates keys are { feedback_Id}, The set of key attributes are: { feedback_Id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency feedback_Id --> customer_Id
checking functional dependency feedback_Id --> order_No
checking functional dependency feedback_Id --> rating
checking functional dependency feedback_Id --> feedback_Message
checking functional dependency customer_Id,order_No --> rating
The above FD violates definition of 3NF: it is non-trivial, LHS is not superkey, RHS contains a non-key attribute.

BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

The FD customer_Id,order_No --> rating is non-trivial and its LHS is not a superkey. It violates BCNF.



Normalize to 3NF

Attributes

order_No customer_Id feedback_Message rating

Functional Dependencies

order_No → customer_Id feedback_Message rating

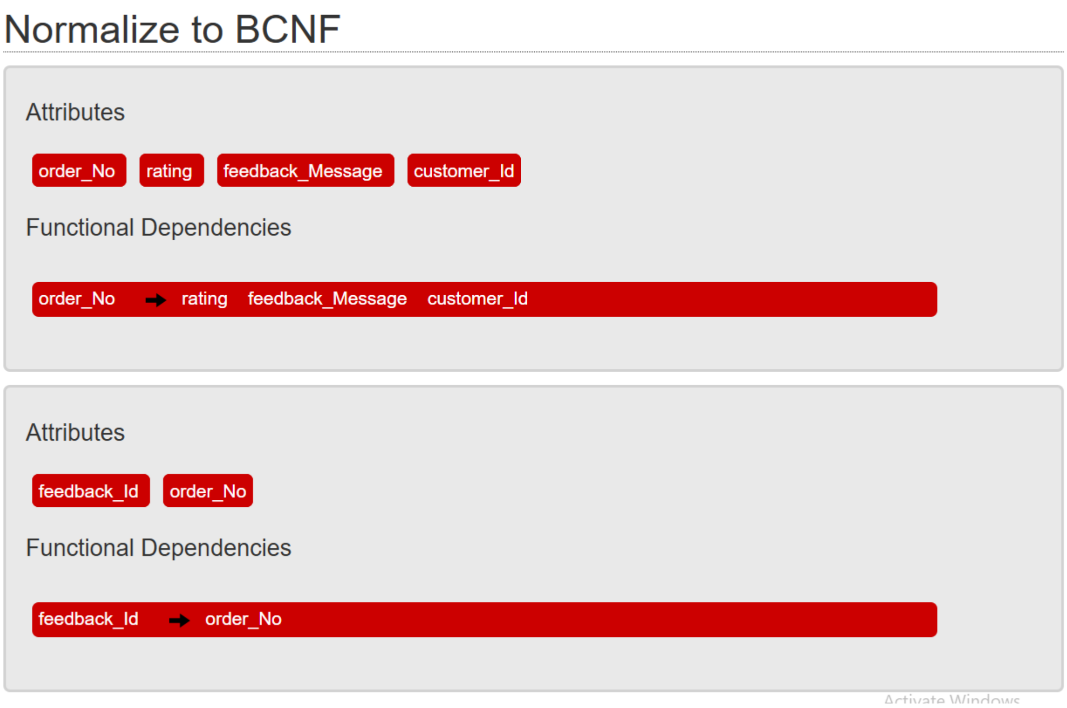
Attributes

feedback_Id order_No

Functional Dependencies

feedback_Id → order_No

Now it is in BCNF Too



Normalize to BCNF

Attributes

order_No rating feedback_Message customer_Id

Functional Dependencies

order_No → rating feedback_Message customer_Id

Attributes

feedback_Id order_No

Functional Dependencies

feedback_Id → order_No

Table 4:
Employee Table:
Attributes:
employee_Id,employee_Name,salary,contact
FD's:

employee_Id → employee_Name

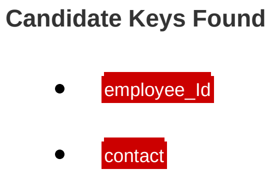
employee_Id→ salary

employee_Id→ contact

contact → employee_Id

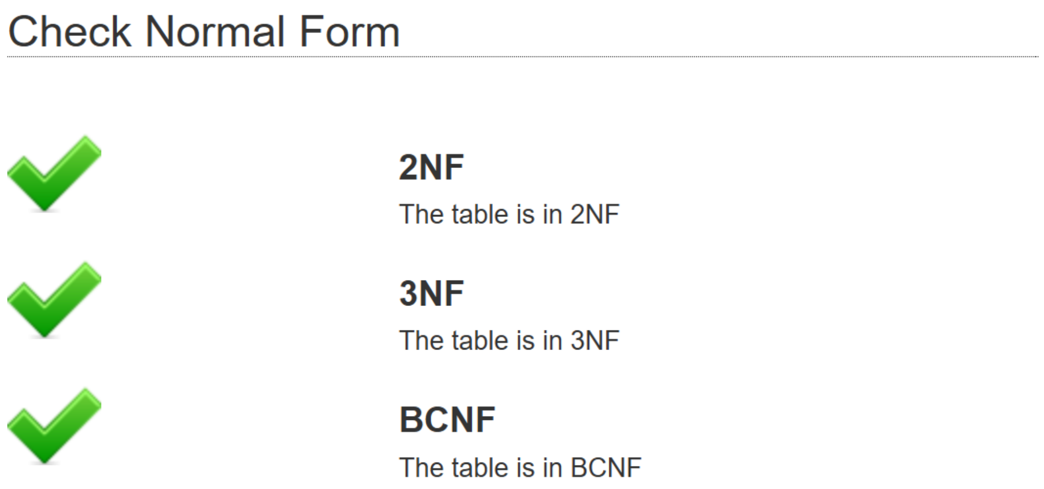
contact → employee_Name

contact → salary



Candidate Keys Found

- employee_Id
- contact

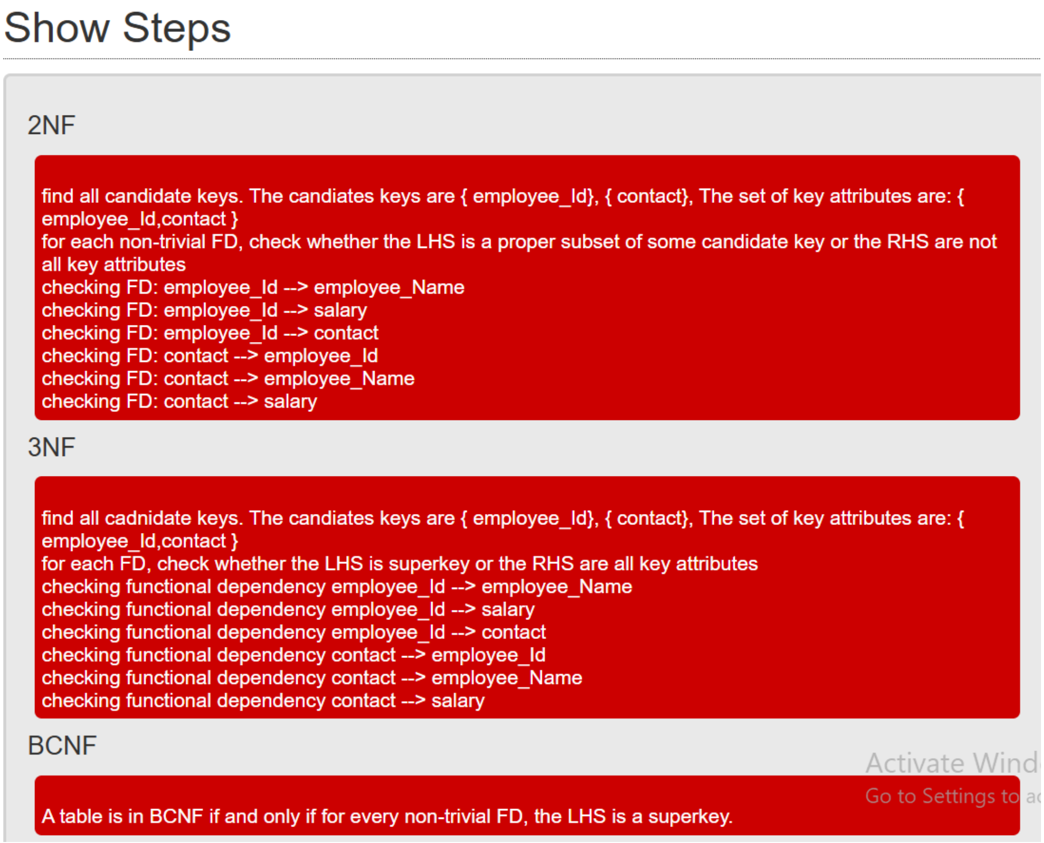


Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF



Show Steps

2NF

find all candidate keys. The candiates keys are { employee_Id}, { contact}, The set of key attributes are: { employee_Id,contact }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: employee_Id --> employee_Name
checking FD: employee_Id --> salary
checking FD: employee_Id --> contact
checking FD: contact --> employee_Id
checking FD: contact --> employee_Name
checking FD: contact --> salary

3NF

find all cadnidate keys. The candiates keys are { employee_Id}, { contact}, The set of key attributes are: { employee_Id,contact }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency employee_Id --> employee_Name
checking functional dependency employee_Id --> salary
checking functional dependency employee_Id --> contact
checking functional dependency contact --> employee_Id
checking functional dependency contact --> employee_Name
checking functional dependency contact --> salary

BCNF

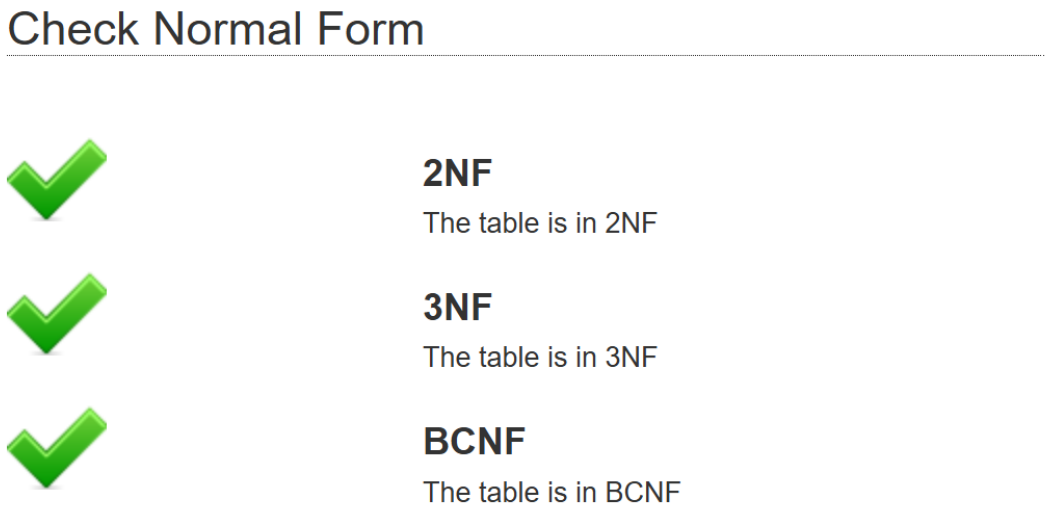
A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

Table 5:

```
Employee_Roles:
Attributes:
employee_Id, role_of_Employee
FD’s:
(employee_Id, role_of_Employee) → (no other attributes)
```

Candidate Keys Found

- employee_Id role_of_Employee



Check Normal Form

✓ **2NF**

The table is in 2NF

✓ **3NF**

The table is in 3NF

✓ **BCNF**

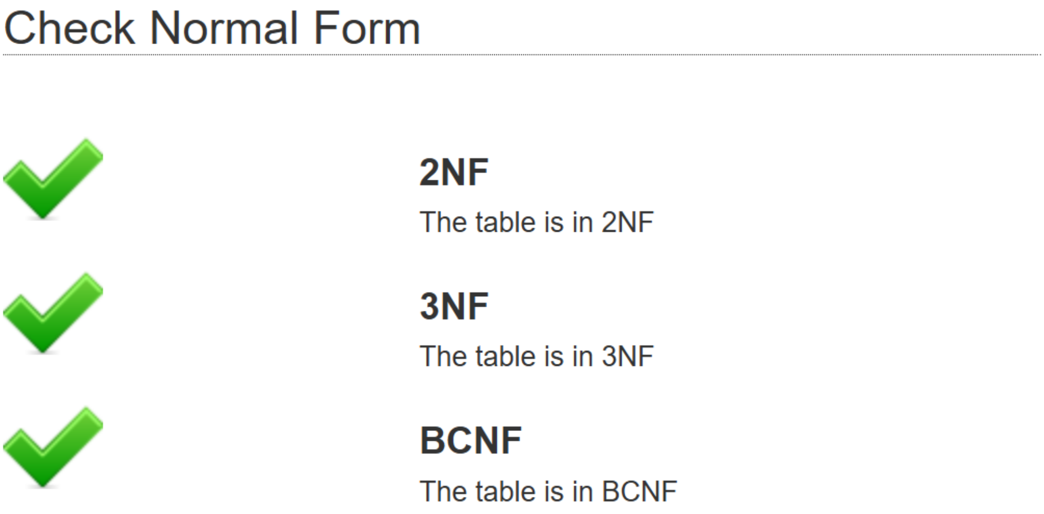
The table is in BCNF

Table 6:

```
Supplier:
Attributes:
supplier_Id, supplier_Name, contact_No, supplier_Address
FD’s:
supplier_Id -> supplier_Name
supplier_Id -> contact_No
supplier_Id -> supplier_Address
contact_No -> supplier_Id
contact_No -> supplier_Name
contact_No -> supplier_Address
```

Candidate Keys Found

- supplier_Id
- contact_No

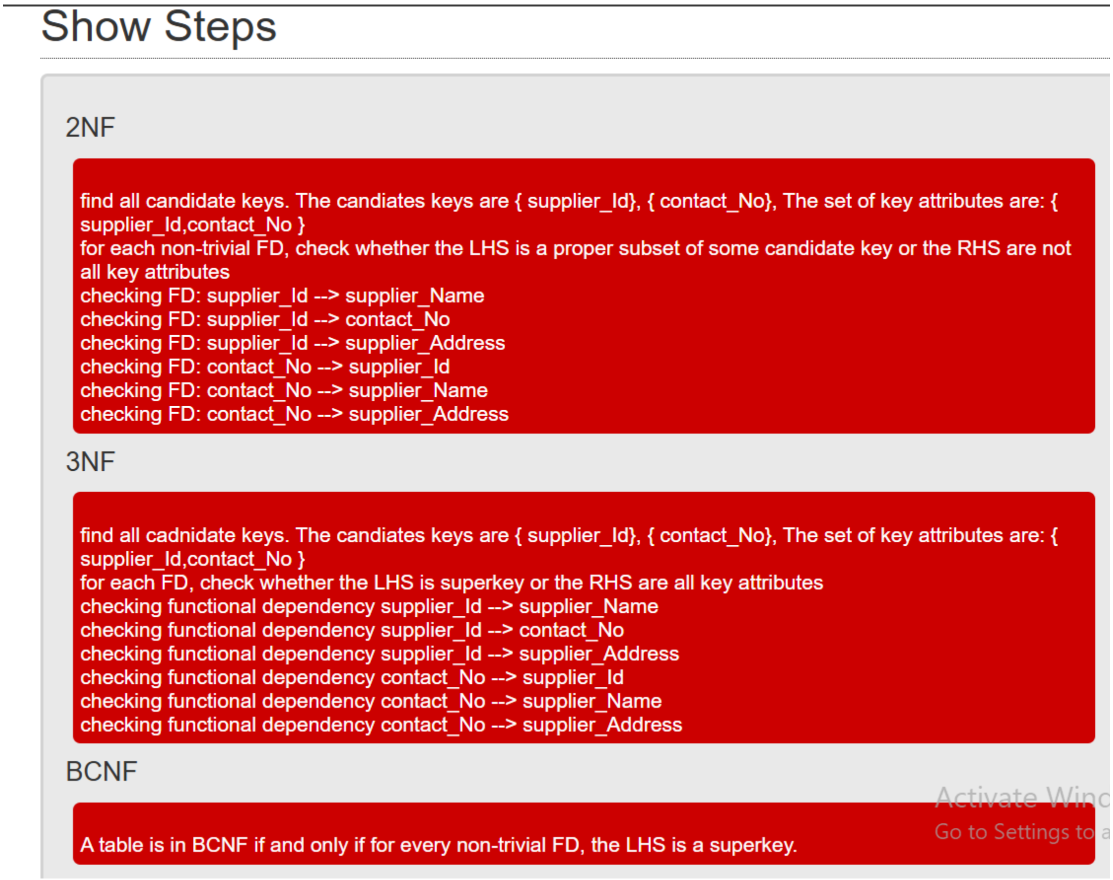


Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF



Show Steps

2NF

find all candidate keys. The candiates keys are { supplier_Id}, { contact_No}, The set of key attributes are: { supplier_Id,contact_No }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: supplier_Id --> supplier_Name
checking FD: supplier_Id --> contact_No
checking FD: supplier_Id --> supplier_Address
checking FD: contact_No --> supplier_Id
checking FD: contact_No --> supplier_Name
checking FD: contact_No --> supplier_Address

3NF

find all cadnidate keys. The candiates keys are { supplier_Id}, { contact_No}, The set of key attributes are: { supplier_Id,contact_No }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency supplier_Id --> supplier_Name
checking functional dependency supplier_Id --> contact_No
checking functional dependency supplier_Id --> supplier_Address
checking functional dependency contact_No --> supplier_Id
checking functional dependency contact_No --> supplier_Name
checking functional dependency contact_No --> supplier_Address

BCNF

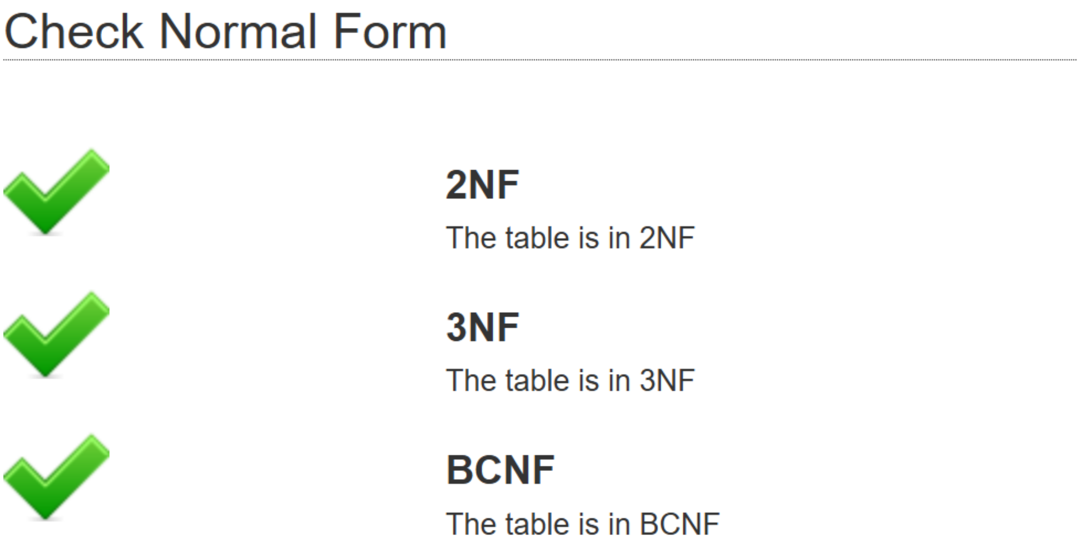
A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

Table 7:

```
Payment:
Attributes:
 payment_Id, order_No, payment_Type, amount, payment_Date
FD’s:
payment_Id -> order_No
payment_Id -> payment_Type
payment_Id -> amount
payment_Id -> payment_Date
order_No -> payment_Id
order_No -> payment_Type
order_No -> payment_Type
order_No -> payment_Date
```

Candidate Keys Found

- payment_Id
- order_No

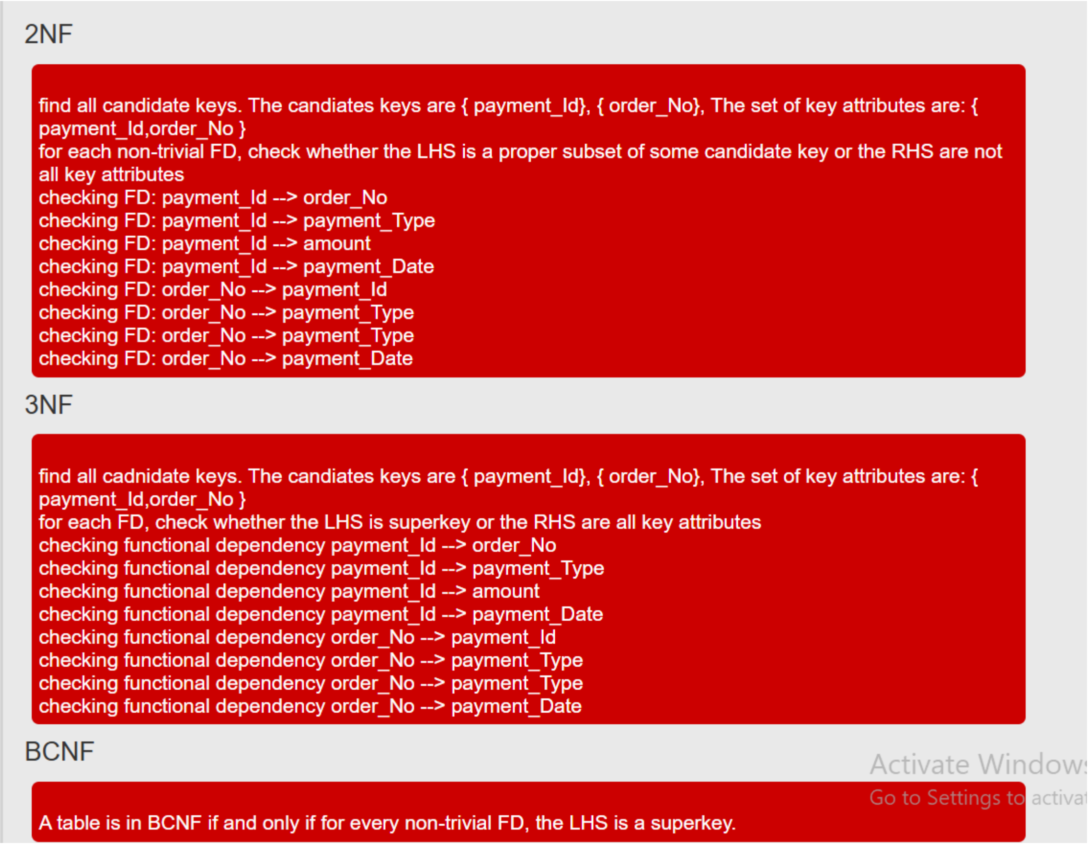


Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF



2NF

```
find all candidate keys. The candiates keys are { payment_Id}, { order_No}, The set of key attributes are: { payment_Id,order_No }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: payment_Id --> order_No
checking FD: payment_Id --> payment_Type
checking FD: payment_Id --> amount
checking FD: payment_Id --> payment_Date
checking FD: order_No --> payment_Id
checking FD: order_No --> payment_Type
checking FD: order_No --> payment_Type
checking FD: order_No --> payment_Date
```

3NF

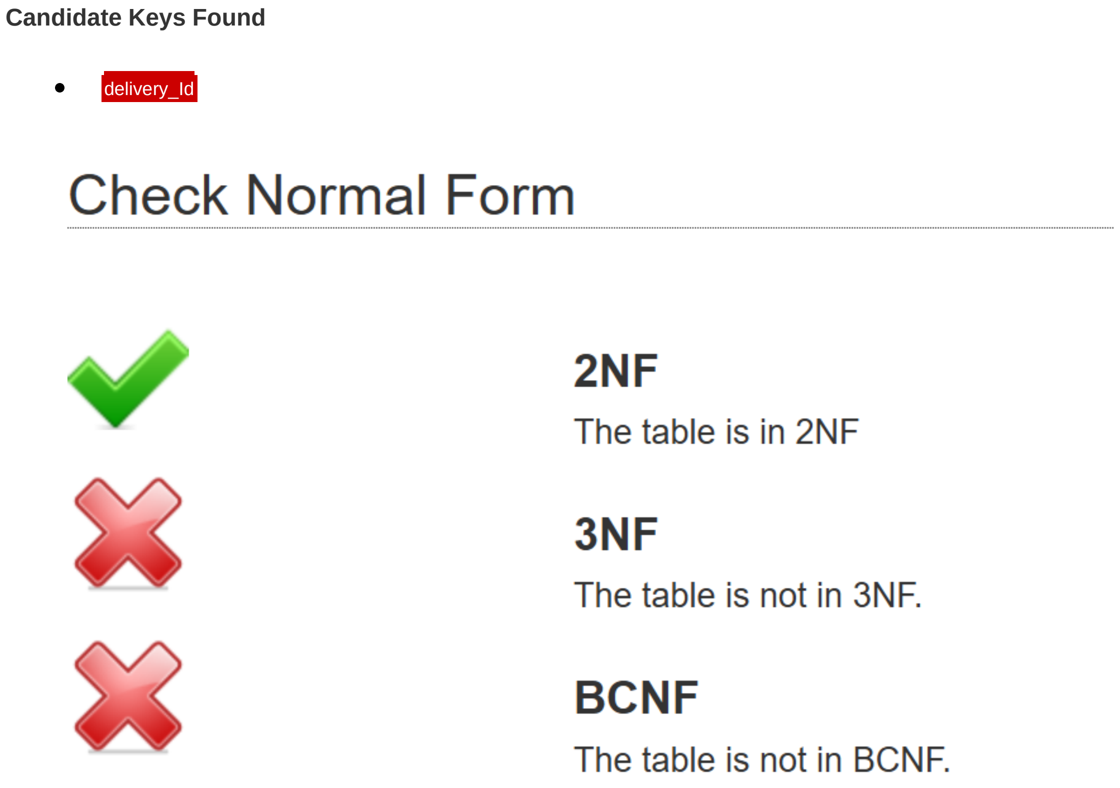
```
find all cadnidate keys. The candiates keys are { payment_Id}, { order_No}, The set of key attributes are: { payment_Id,order_No }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency payment_Id --> order_No
checking functional dependency payment_Id --> payment_Type
checking functional dependency payment_Id --> amount
checking functional dependency payment_Id --> payment_Date
checking functional dependency order_No --> payment_Id
checking functional dependency order_No --> payment_Type
checking functional dependency order_No --> payment_Type
checking functional dependency order_No --> payment_Date
```

BCNF

```
A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.
```

Table 8:

```
Delivery:
Attributes:
delivery_Id, order_No, employee_Id, delivery_Status, expected_time
FD’s:
delivery_Id -> order_No
delivery_Id -> employee_Id
delivery_Id -> delivery_Status
delivery_Id -> expected_time
order_No,employee_Id -> delivery_Status
order_No,employee_Id -> expected_time
```



Candidate Keys Found

- delivery_Id

Check Normal Form

✓ **2NF**

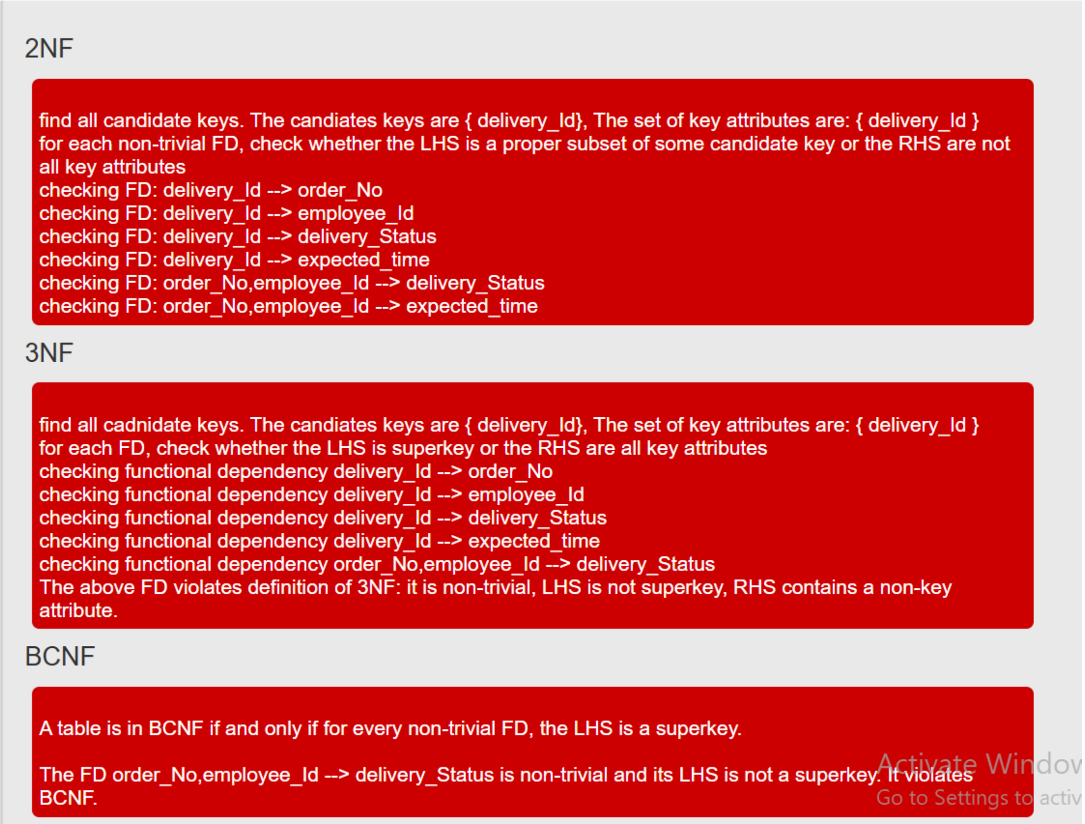
The table is in 2NF

✗ **3NF**

The table is not in 3NF.

✗ **BCNF**

The table is not in BCNF.



2NF

find all candidate keys. The candiates keys are { delivery_Id}, The set of key attributes are: { delivery_Id }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: delivery_Id --> order_No
checking FD: delivery_Id --> employee_Id
checking FD: delivery_Id --> delivery_Status
checking FD: delivery_Id --> expected_time
checking FD: order_No,employee_Id --> delivery_Status
checking FD: order_No,employee_Id --> expected_time

3NF

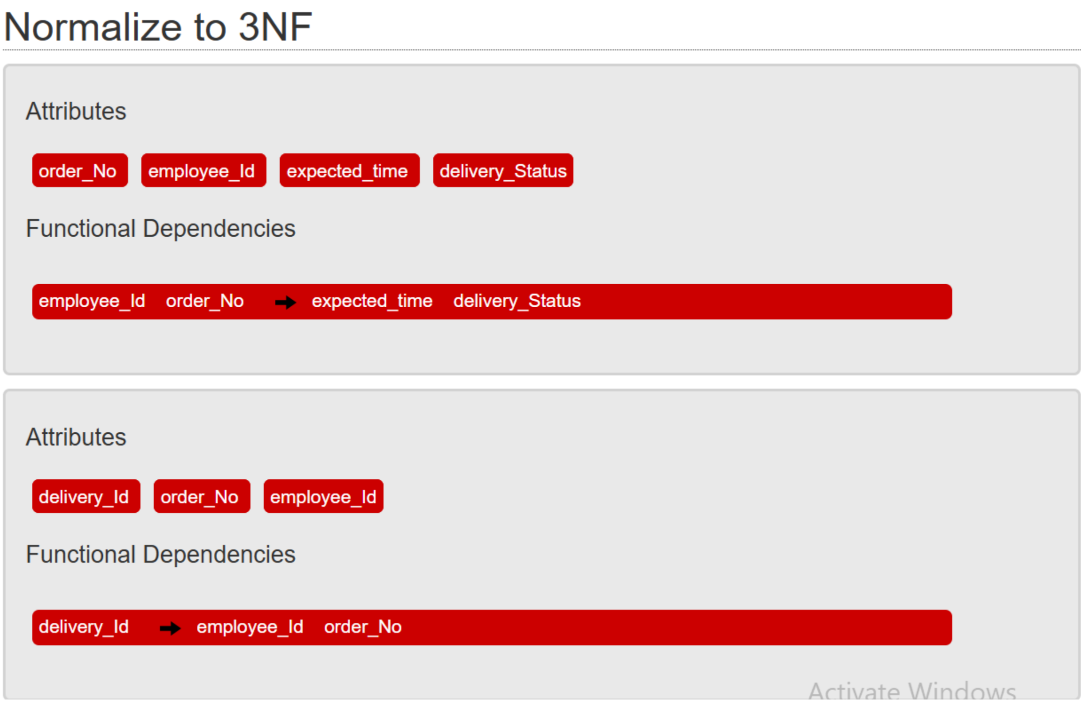
find all cadnidate keys. The candiates keys are { delivery_Id}, The set of key attributes are: { delivery_Id }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency delivery_Id --> order_No
checking functional dependency delivery_Id --> employee_Id
checking functional dependency delivery_Id --> delivery_Status
checking functional dependency delivery_Id --> expected_time
checking functional dependency order_No,employee_Id --> delivery_Status
The above FD violates definition of 3NF: it is non-trivial, LHS is not superkey, RHS contains a non-key attribute.

BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

The FD order_No,employee_Id --> delivery_Status is non-trivial and its LHS is not a superkey. It violates BCNF.

Normalization:



Normalize to 3NF

Attributes

order_No employee_Id expected_time delivery_Status

Functional Dependencies

employee_Id order_No → expected_time delivery_Status

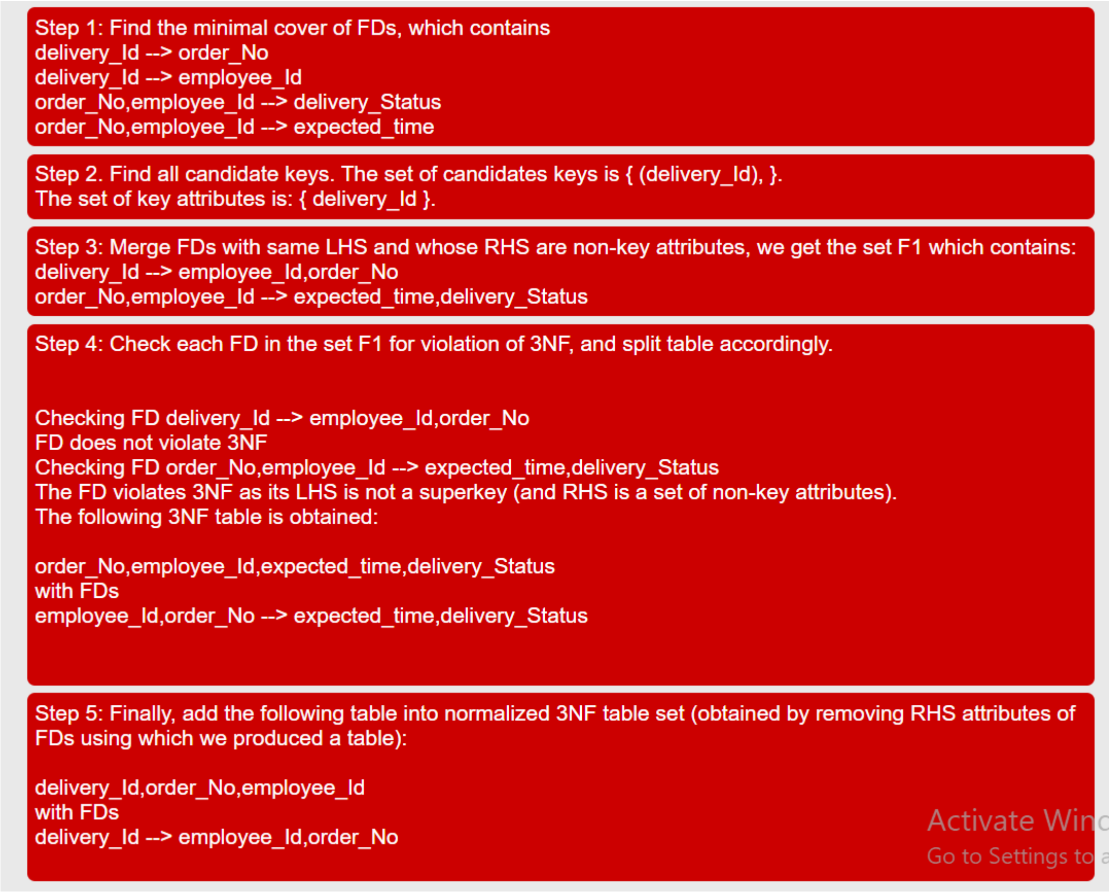
Attributes

delivery_Id order_No employee_Id

Functional Dependencies

delivery_Id → employee_Id order_No

Steps:



Step 1: Find the minimal cover of FDs, which contains
delivery_Id --> order_No
delivery_Id --> employee_Id
order_No,employee_Id --> delivery_Status
order_No,employee_Id --> expected_time

Step 2. Find all candidate keys. The set of candidates keys is { (delivery_Id), }.
The set of key attributes is: { delivery_Id }.

Step 3: Merge FDs with same LHS and whose RHS are non-key attributes, we get the set F1 which contains:
delivery_Id --> employee_Id,order_No
order_No,employee_Id --> expected_time,delivery_Status

Step 4: Check each FD in the set F1 for violation of 3NF, and split table accordingly.

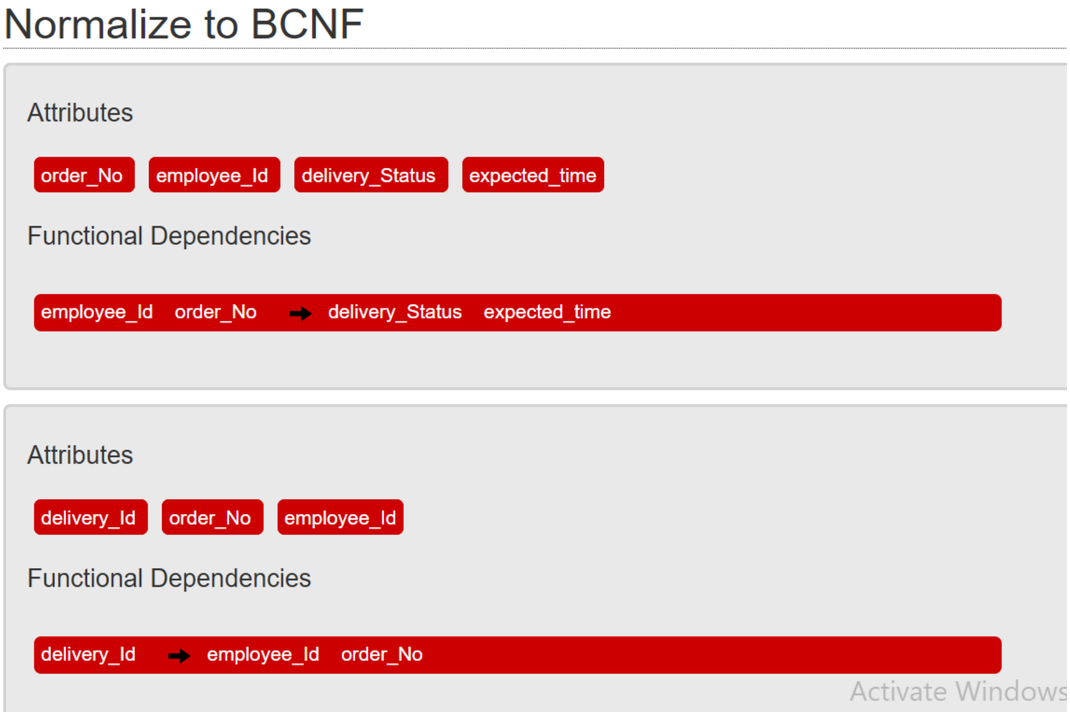
Checking FD delivery_Id --> employee_Id,order_No
FD does not violate 3NF
Checking FD order_No,employee_Id --> expected_time,delivery_Status
The FD violates 3NF as its LHS is not a superkey (and RHS is a set of non-key attributes).
The following 3NF table is obtained:

order_No,employee_Id,expected_time,delivery_Status
with FDs
employee_Id,order_No --> expected_time,delivery_Status

Step 5: Finally, add the following table into normalized 3NF table set (obtained by removing RHS attributes of FDs using which we produced a table):

delivery_Id,order_No,employee_Id
with FDs
delivery_Id --> employee_Id,order_No

Now it’s in BCNF too:



Normalize to BCNF

Attributes

order_No employee_Id delivery_Status expected_time

Functional Dependencies

employee_Id order_No → delivery_Status expected_time

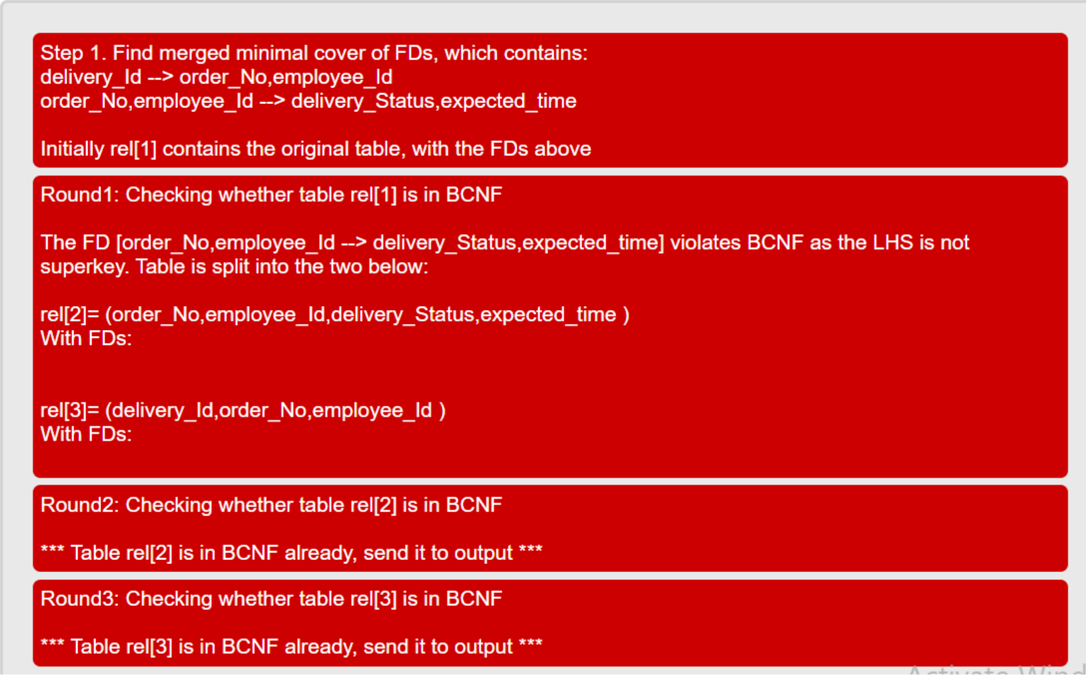
Attributes

delivery_Id order_No employee_Id

Functional Dependencies

delivery_Id → employee_Id order_No

Steps:



Step 1. Find merged minimal cover of FDs, which contains:
delivery_Id --> order_No,employee_Id
order_No,employee_Id --> delivery_Status,expected_time

Initially rel[1] contains the original table, with the FDs above

Round1: Checking whether table rel[1] is in BCNF

The FD [order_No,employee_Id --> delivery_Status,expected_time] violates BCNF as the LHS is not superkey. Table is split into the two below:

rel[2]= (order_No,employee_Id,delivery_Status,expected_time)
With FDs:

rel[3]= (delivery_Id,order_No,employee_Id)
With FDs:

Round2: Checking whether table rel[2] is in BCNF

*** Table rel[2] is in BCNF already, send it to output ***

Round3: Checking whether table rel[3] is in BCNF

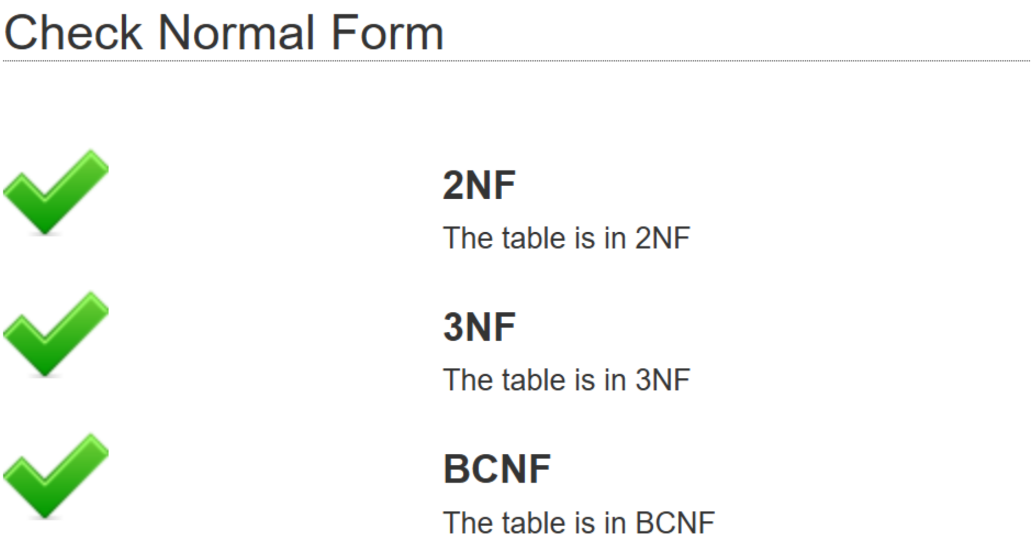
*** Table rel[3] is in BCNF already, send it to output ***

Table 9:

```
Inventory:
Attributes:
item_No, item_Name, category, price, item_Quantity
FD’s:
item_No -> item_Name
item_No -> category
item_No -> price
item_No -> item_Quantity
```

Candidate Keys Found

- item_No

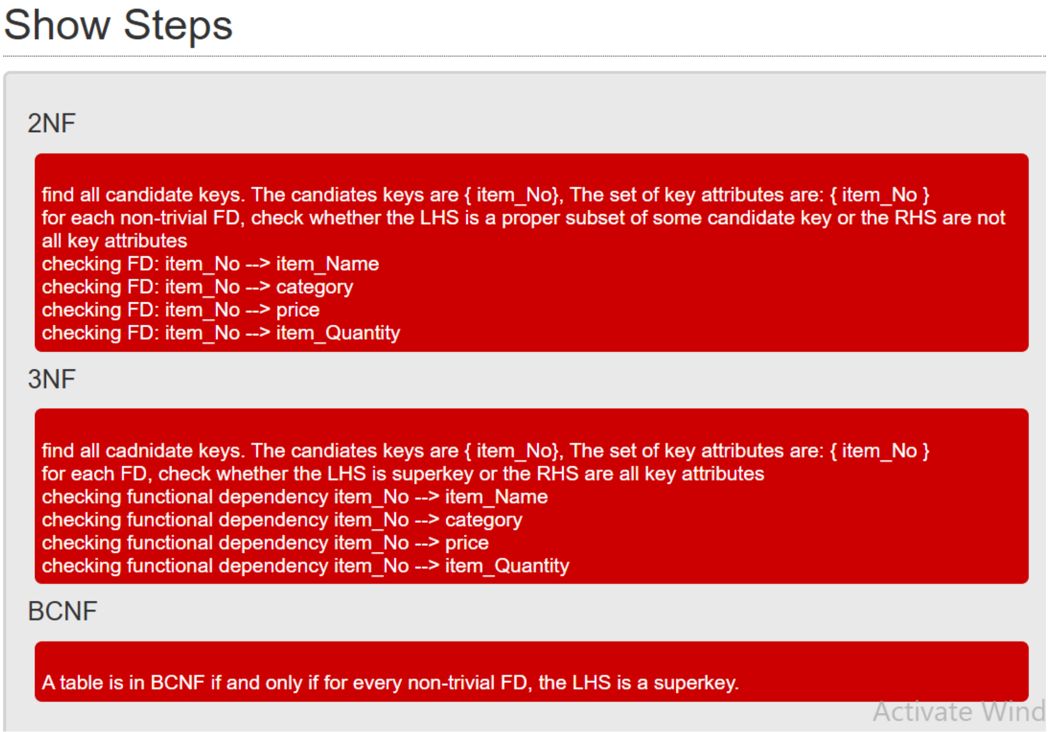


Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF



Show Steps

2NF

find all candidate keys. The candiates keys are { item_No}, The set of key attributes are: { item_No }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: item_No --> item_Name
checking FD: item_No --> category
checking FD: item_No --> price
checking FD: item_No --> item_Quantity

3NF

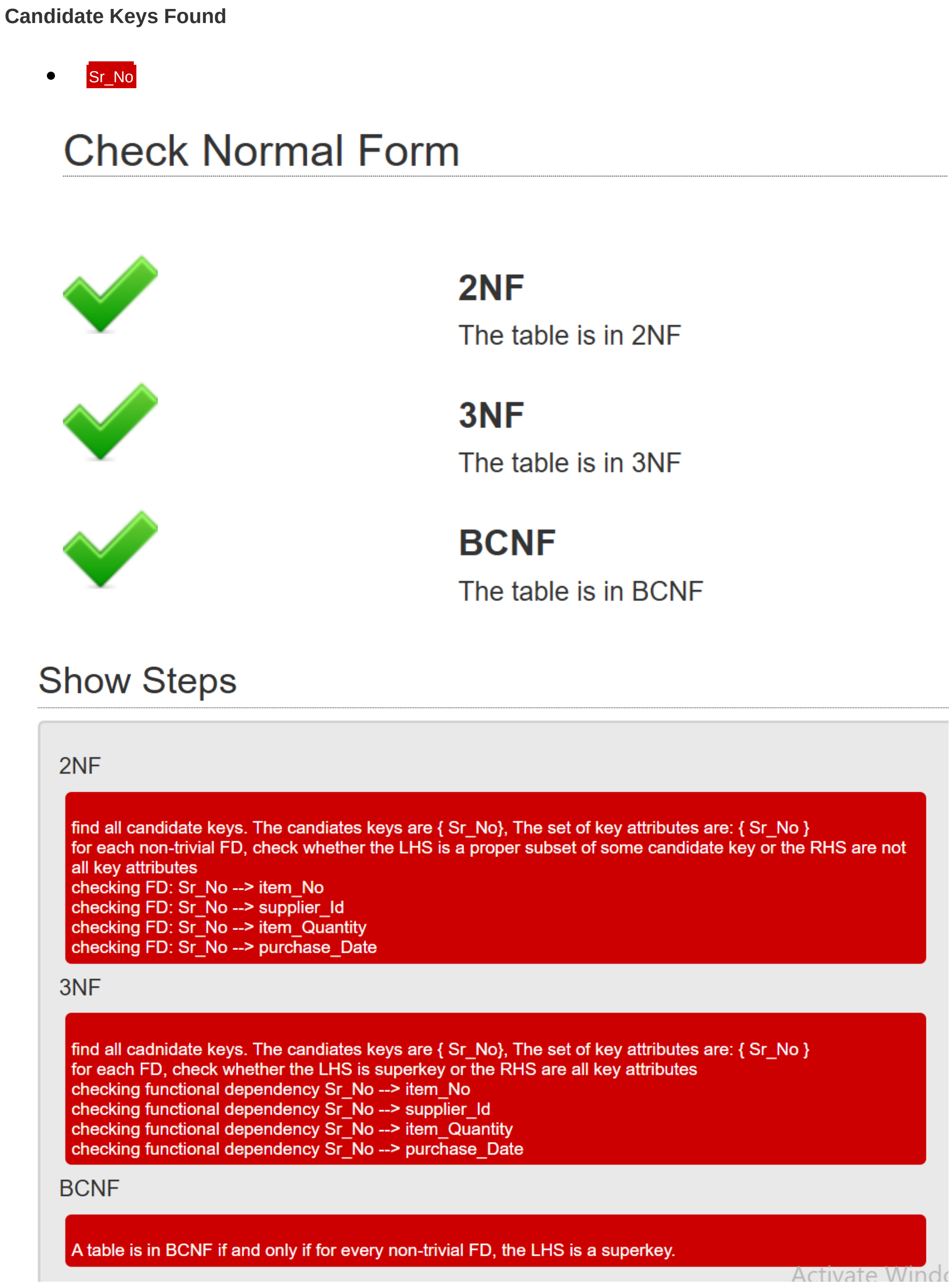
find all cadnidate keys. The candiates keys are { item_No}, The set of key attributes are: { item_No }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency item_No --> item_Name
checking functional dependency item_No --> category
checking functional dependency item_No --> price
checking functional dependency item_No --> item_Quantity

BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

Table 10:

```
Supplied_Items:
Attributes:
Sr_No, item_No, supplier_Id, item_Quantity, purchase_Date
FD’s:
Sr_No -> item_No
Sr_No -> supplier_Id
Sr_No -> item_Quantity
Sr_No -> purchase_Date
```



Candidate Keys Found

- Sr_No

Check Normal Form

✓ **2NF**

The table is in 2NF

✓ **3NF**

The table is in 3NF

✓ **BCNF**

The table is in BCNF

Show Steps

2NF

find all candidate keys. The candiates keys are { Sr_No}, The set of key attributes are: { Sr_No }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: Sr_No --> item_No
checking FD: Sr_No --> supplier_Id
checking FD: Sr_No --> item_Quantity
checking FD: Sr_No --> purchase_Date

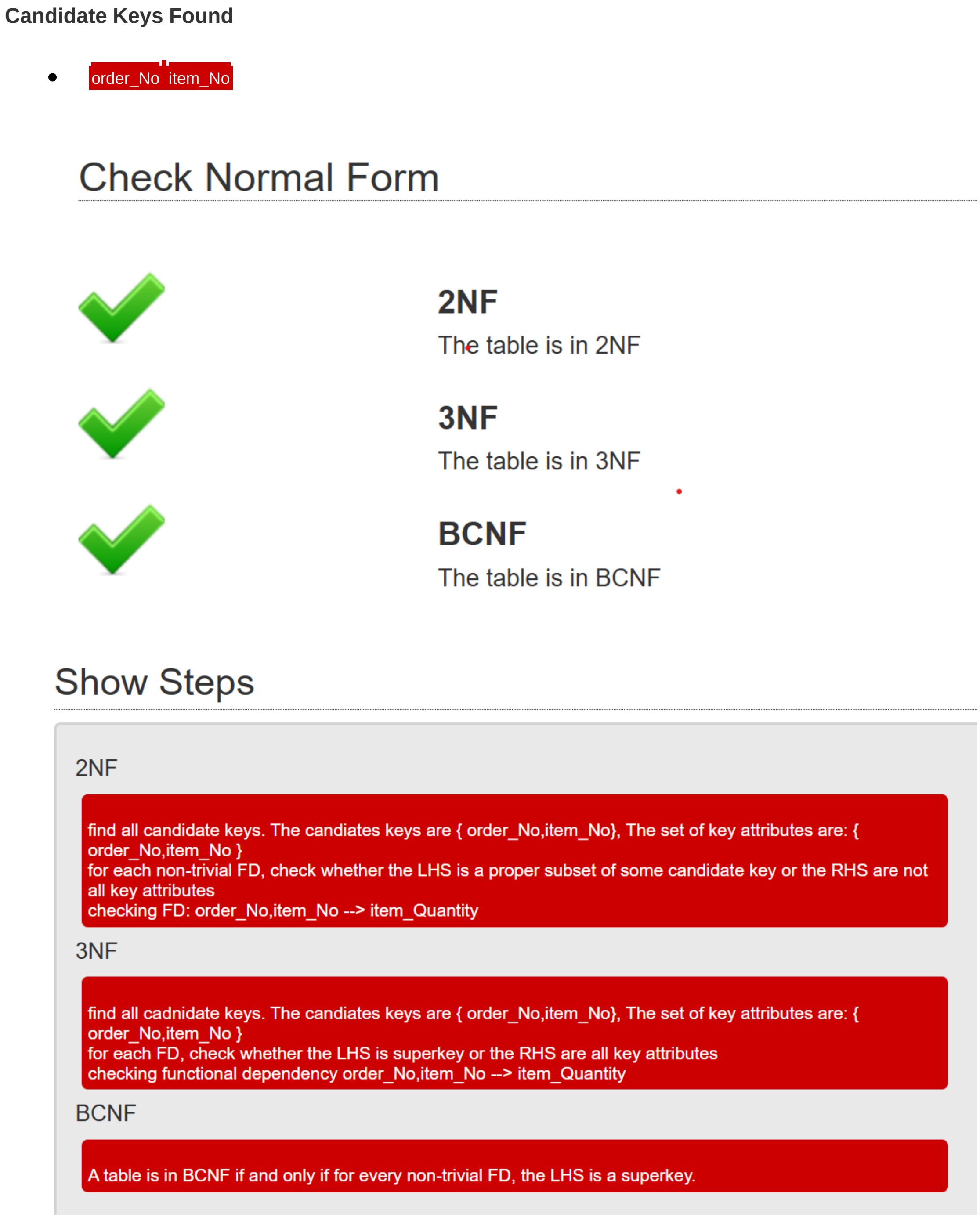
3NF

find all cadnidate keys. The candiates keys are { Sr_No}, The set of key attributes are: { Sr_No }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency Sr_No --> item_No
checking functional dependency Sr_No --> supplier_Id
checking functional dependency Sr_No --> item_Quantity
checking functional dependency Sr_No --> purchase_Date

BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

```
Table 11:
Ordered_Item:
Attributes:
order_No, item_No, item_Quantity
FD’s:
order_No,item_No -> item_Quantity
```



Candidate Keys Found

- order_No item_No

Check Normal Form

✓ **2NF**
The table is in 2NF

✓ **3NF**
The table is in 3NF

✓ **BCNF**
The table is in BCNF

Show Steps

2NF

find all candidate keys. The candiates keys are { order_No,item_No}, The set of key attributes are: { order_No,item_No }
for each non-trivial FD, check whether the LHS is a proper subset of some candidate key or the RHS are not all key attributes
checking FD: order_No,item_No --> item_Quantity

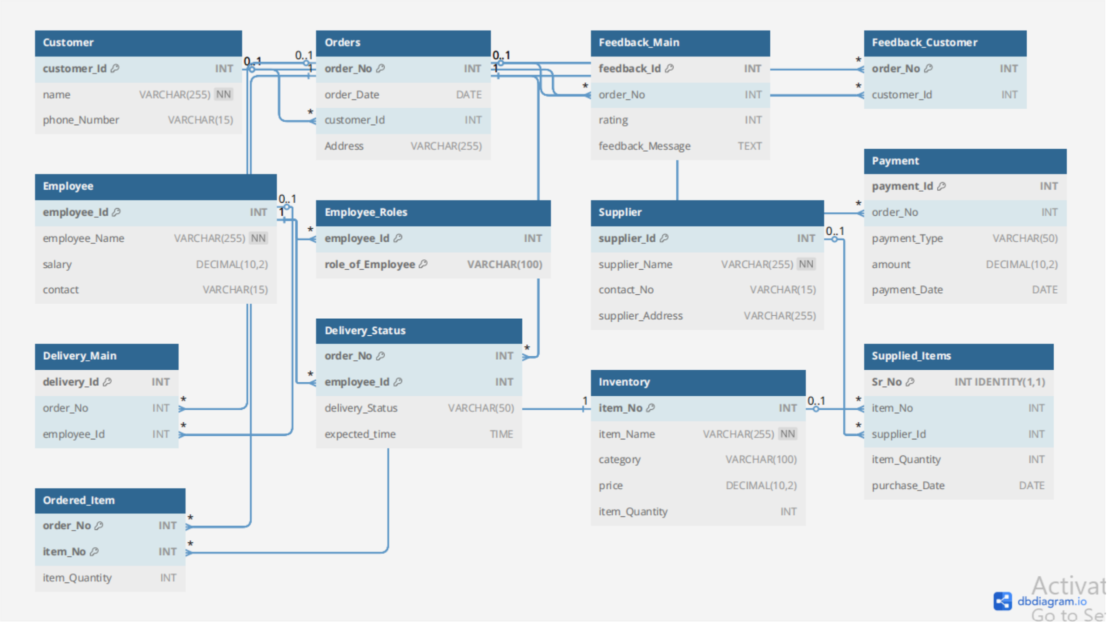
3NF

find all cadnidate keys. The candiates keys are { order_No,item_No}, The set of key attributes are: { order_No,item_No }
for each FD, check whether the LHS is superkey or the RHS are all key attributes
checking functional dependency order_No,item_No --> item_Quantity

BCNF

A table is in BCNF if and only if for every non-trivial FD, the LHS is a superkey.

● ERD After Normalization:



● Observations and Remarks

While normalizing the Quick_Mart database, I gained valuable insights into the structure and integrity of relational data:

1. Functional Dependencies (FDs) were critical to identify accurately, especially in tables with composite keys like Feedback and Ordered_Item.
2. Moving from 1NF to BCNF helped eliminate partial and transitive dependencies, reducing redundancy and improving data consistency.
3. I chose to use surrogate keys (e.g., feedback_Id, payment_Id) to simplify table relationships and avoid overly complex composite keys.
4. Real-world constraints, such as unique contact numbers, played a key role in defining alternate and candidate keys.
5. The normalization process clarified table purposes and improved scalability and maintainability, especially with many-to-many relationships.
6. A key challenge was balancing normalization depth with design simplicity, particularly in maintaining referential integrity across related tables.