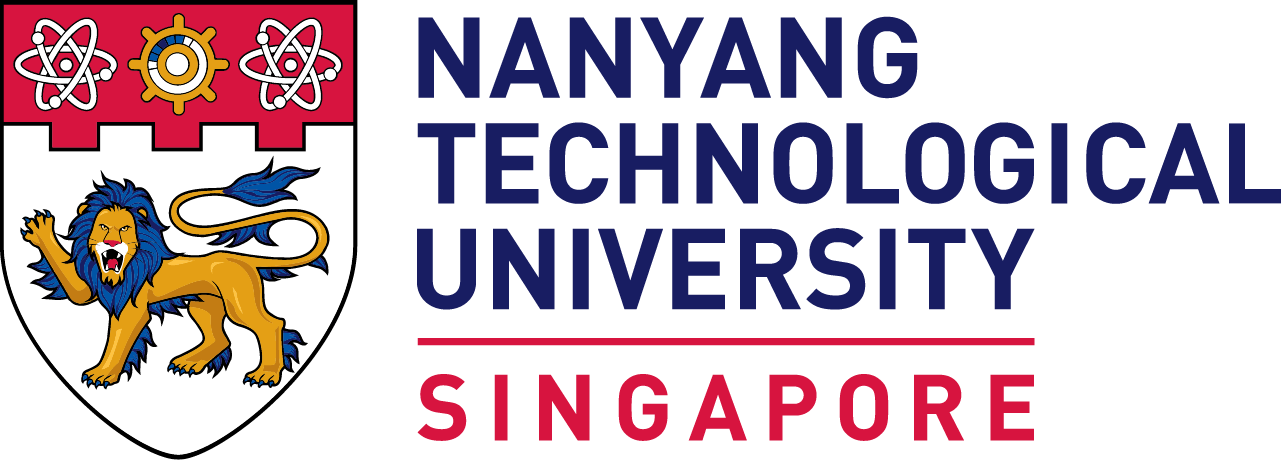
****

**CZ2007 SS3 Group 2 Lab 2 Report**

Members:

Eugene Poh Yang Quan

Wong Yi Pun

Roy Lau Run-Xuan

Chua Zi Jian

Ryan

Koh Jun Kai

# Contents Page No.

[Contents](#_nc40izx8nsh8)  2

[PRICE-HISTORY (PName, SName, Start-date, End-date, Price)](#_e2fotfwcd8os) 3

[PRODUCTS (PName, Maker, Category)](#_tlmxx0u07h6n) 4

[PRODUCTS-IN-ORDERS (PName, SName, OID, OPID, OPrice, OQuantity, Delivery-date, Status)](#_6389ckybjtfm) 5

[PRODUCTS-IN-SHOPS (PName, SName, SPID, SPrice, SQuantity)](#_b1qzs0tzsca2) 6

[COMPLAINTS (ID, UID, Text, Status, Filed-date-time)](#_7ueqaz2h77jo) 8

[COMPLAINTS-ON-SHOPS (ID, Sname)](#_fstc268glia2) 9

[COMPLAINTS-ON-ORDERS (ID, OID)](#_f9ay4xqphvy3) 10

[EMPLOYEE (ID, Name, Salary)](#_ozfeivps8ceg) 11

[ORDERS (OID, UID, Date-time, Shipping-address)](#_cd2oesnze7fu) 12

[FEEDBACK (UID, PName, SName, OID, Date-time, Rating, Comment)](#_cik0i07z9sur) 13

[SHOPS (SName)](#_5w5i6ibxr6kv) 15

[HANDLED (ID, handled-date-time, Employee-ID)](#_3809hbcgeyye) 15

[USERS (UID, UName)](#_606lin1pshdq) 16

[APPENDIX C: INDIVIDUAL CONTRIBUTION FORM](#_d88er4dc7lcg) 17

# 

# PRICE-HISTORY (PName, SName, Start-date, End-date, Price)

Assumptions:

* No price change within the same day.

Functional Dependencies:

1. PName, SName, Start-date, End-date → Price

Closures:

* {PName, SName, Start-date, End-date}+ = {PName, SName, Start-date, End-date, Price}

Key:

* {PName, SName, Start-date, End-date}

Primary key:

* {PName, SName, Start-date, End-date}

3NF:

* FD-1 has a key on the LHS and is the only non-trivial FD.
* PRICE-HISTORY is in 3NF.

# 

# 

# PRODUCTS (PName, Maker, Category)

Assumptions:

* The same product made by different maker has different names
  + E.G phones made by Samsung are Galaxy X, phones made by Apple are IPhones.
* Maker can make multiple different categories of products.

Functional Dependencies:

1. PName → Maker, Category

Closures:

* {PName}+ = {PName, Maker, Category}
* {Maker}+ = {Maker}
* {Category}+ = {Category}

Key:

* PName

Primary key:

* PName

3NF:

* FD-1 has a key on the LHS and is non-trivial.
* PRODUCTS is in 3NF.

# 

# PRODUCTS-IN-ORDERS (PName, SName, OID, OPID, OPrice, OQuantity, Delivery-date, Status)

Assumptions:

* Cannot have two of the same product, from the same shop, in the same order but with different OPID.

Functional Dependencies:

1. OPID, OID → OPrice, OQuantity, delivery-date, status, PName, SName
2. PName, SName, OID → OPID

Closures:

* {PName, SName, OID}+  = {PName, SName, OID, OPID, OPrice , OQuantity, delivery-date, status}
* {OPID,OID}+  = {OPID, OID, OPrice, OQuantity, delivery-date, status, PName, SName}

Key:

* PName, SName, OID
* OPID,OID

Primary Key:

* PName, SName, OID

3NF:

* FD-1 has a key on the LHS and is non-trivial.
* FD-2 has a key on the LHS and is non-trivial.
* PRODUCTS-IN-ORDERS is in 3NF.

# 

# PRODUCTS-IN-SHOPS (PName, SName, SPID, SPrice, SQuantity)

Functional Dependencies:

1. SPID → SPrice, SQuantity
2. SName, PName → SPID, SPrice, SQuantity

Closures:

* {SName, PName}+ = {PName,SName,SPID,SPrice,SQuantity}

Key:

* SName, PName

Primary key:

* SName, PName

3NF violation:

* SPID → SPrice, SQuantity ; LHS is not a key and RHS does not belong to a key

Minimal Basis:

* + Step 1: Transform so that RHS only one attribute
* SPID → SPrice
* SPID → SQuantity
* SName, PName → SPID
* SName, PName → SPrice
* SName, PName → SQuantity
  + Step 2: Remove redundant FDs by assuming FD doesn’t exist. Check if LHS closure contains RHS
* Remove SPID → SPrice
  + {SPID}+ = {SPID, SQuantity}
* Remove SPID → SQuantity
  + {SPID}+ = {SPID,SPrice }
* Remove SName, PName → SPID
  + {SName, PName}+ = {SName, PName, SPrice, SQuantity}
* Remove SName, PName → SPrice
  + {SName, PName}+ = {SName, PName SPID, SPrice, SQuantity}
  + SName, PName → SPrice is redundant
* Remove SName, PName → SQuantity
  + {SName, PName}+ = {SName, PName SPID, SPrice, SQuantity}
  + SName, PName → SQuantity is redundant
* FDs:
  + SPID → SPrice
  + SPID → SQuantity
  + SName, PName → SPID
  + Step 3: Composite LHS, remove each one and check if closure still contains RHS
* Remove SName
  + {PName}+ = {PName}
* Remove PName
  + {SName}+ = {SName}
* FDs:
  + SPID → SPrice,
  + SPID → SQuantity
  + SName, PName → SPID

Combine FDs

* FDs:
  + SPID → SPrice, SQuantity
  + SName, PName → SPID

Create table for each FD remained

* R1(SPID, SPrice, SQuantity), R2(SName, PName, SPID)

If none of the tables contains key, create a table that contains a key of R

* NA

Remove redundant tables

* NA

Decomposed relations:

* R1(SPID, SPrice, SQuantity)
* R2(SName, PName, SPID)

# 

# COMPLAINTS (ID, UID, Text, Status, Filed-date-time)

Assumptions:

* May have the same text from different complaints at different shops.
* Multiple complaints can be filed at the same time by the same user or by different users.
* Multiple complaints can be handled at the same time by the same employee or by different employees.

Functional Dependencies:

1. ID → Text, Status, Filed-date-time, Handled-date-time, Employee ID, UID

Closures:

* {ID}+ = {ID, Text, Status, Filed-date-time, Handled-date-time, Employee ID, UF}

Key:

* ID

Primary key:

* ID

3NF:

* The LHS of the non-trivial FDs is a key, So the relation is in 3NF.

# 

# COMPLAINTS-ON-SHOPS (ID, Sname)

Assumptions:

* NA

Functional Dependencies:

* ID → Sname

Closures:

* {ID}+ ={ID, Sname}

Keys:

* ID

Primary Keys:

* ID

3NF:

* The LHS of the non-trivial FDs is a key, So the relation is in 3NF.
* Two-attribute relation is always in BCNF, which is also in 3NF.

# 

# COMPLAINTS-ON-ORDERS (ID, OID)

Assumptions:

* NA

Functional Dependencies:

1. ID → OID

Closures:

* {ID}+ ={ID, OID}

Keys:

* ID

Primary Keys:

* ID

3NF:

* The LHS of the non-trivial FDs is a key, So the relation is in 3NF.
* Two-attribute relation is always in BCNF, which is also in 3NF.

# 

# EMPLOYEE (ID, Name, Salary)

Assumptions:

* NA

Functional Dependencies:

1. ID → Name, Salary

Closures:

* {ID}+ = {ID, Name, Salary}

Key:

* ID

Primary Key:

* ID

3NF:

* FD-1 has a key on the LHS and is non-trivial.
* EMPLOYEE is in 3NF.

# 

# 

# ORDERS (OID, UID, Date-time, Shipping-address)

Assumption:

* User can order from multiple devices ; user can make more than one order at any one time.

Functional Dependencies:

1. OID → Date-time, Shipping-address, UID

Closures:

* {OID}+ = {Date-time,Shipping-address,UID}

Key:

* OID

Primary Key:

* OID

3NF:

* FD-1 has a key on the LHS and is non-trivial.
* ORDERS is in 3NF.

# 

# FEEDBACK (UID, PName, SName, OID, Date-time, Rating, Comment)

Assumption:

* User can submit feedback from multiple devices ; user can submit more than one feedback at any one time
* If user submits a feedback on the same product, it will overwrite the previous feedback
* May have same text in comments from different feedbacks
* May have same ratings from different feedbacks

Functional Dependencies:

1. OID → UID
2. PName, SName, OID → Date-time, Rating, Comment

Closures:

* {PName,SName,OID}+ = {PName, SName, OID, Date-time, Rating, Comment, UID}

Keys:

* PName, SName, OID

Primary Keys:

* PName, SName, OID

3NF violation:

* OID → UID; LHS is not a key and RHS does not belong to a key.

Minimal Basis:

* Step 1: Transform so that RHS only one attribute
  + PName, SName, OID → Date-time
  + PName, SName, OID → Rating
  + PName, SName, OID → Comment
  + OID → UID
* Step 2: Remove redundant FDs by assuming FD doesn’t exist. Check if LHS closure contains RHS
  + Remove PName, SName, OID → Date-time
    - {PName, SName, OID}+ = {PName, SName, OID, Rating, Comment, UID}
  + Remove PName, SName, OID → Rating
    - {PName, SName, OID}+ = {PName, SName, OID, Date-time, Comment, UID}
  + Remove PName, SName, OID → Comment
    - {PName, SName, OID}+ = {PName, SName, OID, Date-time, Rating, UID}
  + Remove OID → UID
    - {OID}+ = {OID}
* Step 3: For composite LHS, remove each one and check if closure of LHS still contains RHS:
  + Remove SName
    - {PName, OID}+ = {PName, OID, UID}
  + Remove PName
    - {SName, OID}+ = {SName, OID, UID}
  + Remove OID
    - {SName, PName}+ = {SName, PName}
  + FDs:
    - PName, SName, OID → Date-time
    - PName, SName, OID → Rating
    - PName, SName, OID → Comment
    - OID → UID

Combine FDs

* FDs:
  + PName, SName, OID → Date-time, Rating, Comment
  + OID → UID

Create table for each FD remained

* R1(PName, SName, OID, Date-time, Rating, Comment), R2(OID, UID)

If none of the tables contains key, create a table that contains a key of R

* NA

Remove redundant tables

* NA

Decomposed relations:

* R1(PName, SName, OID, Date-time, Rating, Comment)
* R2(OID, UID)

# 

# SHOPS (SName)

Assumptions:

* NA

Functional Dependencies:

1. NA

Closures:

* NA

Key:

* SName

Primary key:

* SName

3NF:

* No FD.
* SHOPSis in 3NF.

# HANDLED (ID, handled-date-time, Employee-ID)

Assumptions:

* Multiple complaints can be handled at the same time by the same employees or by different employees

Functional dependencies

* ID -> handled-date-time, employee-id

Closures

Key

Primary key : ID

3NF

# 

# USERS (UID, UName)

Assumptions:

* NA

Functional Dependencies:

1. UID → UName

Closures:

* (UID}+ = {UID, UName}

Key:

* UID

Primary Key:

* UID

3NF:

* USERS is a two-attribute relation and must be in the third normal form.
* Two-attribute relation is always in BCNF, which is also in 3NF.

# APPENDIX C: INDIVIDUAL CONTRIBUTION FORM

| Name | Individual Contribution to Submission 1 (Lab 3) | Percentage of Contribution | Signature |
| --- | --- | --- | --- |
| Eugene Poh Yang Quan | * Conversion of ER diagram into relational schema * Derived keys, primary keys and functional dependencies * Checking of 3NF violation * Normalization of 3NF violation relation schema | 16.6 |  |
| Wong Yi Pun | * Conversion of ER diagram into relational schema * Derived keys, primary keys and functional dependencies * Checking of 3NF violation * Normalization of 3NF violation relation schema | 16.6 |  |
| Roy Lau Run-Xuan | * Conversion of ER diagram into relational schema * Derived keys, primary keys and functional dependencies * Checking of 3NF violation * Normalization of 3NF violation relation schema | 16.6 |  |
| Chua Zi Jian | * Conversion of ER diagram into relational schema * Derived keys, primary keys and functional dependencies * Checking of 3NF violation * Normalization of 3NF violation relation schema | 16.6 |  |
| Ryan | * Conversion of ER diagram into relational schema * Derived keys, primary keys and functional dependencies * Checking of 3NF violation * Normalization of 3NF violation relation schema | 16.6 |  |
| Koh Jun Kai | * Conversion of ER diagram into relational schema * Derived keys, primary keys and functional dependencies * Checking of 3NF violation * Normalization of 3NF violation relation schema | 16.6 |  |