**USERS (UID, UName)**

Keys: UID

Primary Key: UID

Functional Dependencies:

1. UID → UName

USERS is a two-attribute relation and must be in the third normal form.

**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

Keys:

Computing Closures:  
 OID+ = {date-time,shipping-address,UID}

OID is a key

Primary Key: OID

~~Minimal Basis:~~

~~Step 1 (Transform so that RHS only one attribute):~~

~~OID→date-time~~

~~OID→shipping-address~~

~~OID→UID~~

~~Step 2 (Remove redundant FDs, assume FD doesn’t exist, does LHS closure contain RHS?):~~

~~NA~~

~~Step 3 (Composite LHS, remove each one and check if closure still contains RHS):~~

~~NA~~

LHS of every non-trivial FD contains the key of ORDERS. ORDERS is in BCNF. Hence ORDERS does not violate 3NF

The left-hand side of functional dependency 1 contains a key. ORDERS is in the third normal form.

**PRODUCTS-IN-ORDERS (OPID, PName, SName, OID, OPrice, OQuantity, delivery-date, status)**

| **SName** | **PName** | **OID** | **OPID**  **(PName.Sname.OID)** | **OPrice** | **delivery-date** | **OQuantity** | **Status (being processed,shipped,delivered,returned)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| chi | phone | 1 | chi.phone.1 | $500 | 123 |  |  |
| chi | home | 1 | chi.home.1 | $200 | 123 |  |  |
| eng | phone | 1 | eng.phone.1 | $550 | 123 |  |  |
|  |  |  |  |  |  |  |  |
| china | phone | 1 | chi.phone.1 | $200 | 123 | 40 |  |
| chile | phone | 1 | chi.phone.1 | $200 | 123 | 40 |  |
|  |  |  |  |  |  |  |  |
|  |  | 2 | 2 |  |  |  |  |
|  |  |  |  |  |  |  |  |
| china | phone | A | 1 | $100 | 123 | 1 |  |
| china | phone | A | 2 | $100 | 123 | 2 |  |
|  |  | 2 | 1 | $100 | 123 | 1 |  |
| china | phone | B | 1 | $120 | 123 | 2 |  |

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
3. ~~OPID, OID, PName, SName, -> OPrice, OQuantity, delivery-date, status~~

~~PName, SName, OID, OID -> OPID, OID (Augmentation)~~

~~PName, SName, OID -> OPrice, OQuantity, delivery-date, status, PName, SName (transitive)~~

~~PName, SName, OID -> OPrice, OQuantity, delivery-date, status (remove redundant reflexive FD)~~

1. ~~PName, SName, OID -> PName, SName, OID -> OPrice, OQuantity, delivery-date, status, PName, SName (transitive)~~

Key:

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

Key : {PName, SName, OID}

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

Key: {OPID,OID}

Primary Key:

{PName, SName, OID}

The left-hand side of FDs 1 and 2 are keys. PRODUCTS-IN-ORDERS is in 3NF.

~~Minimal Basis:~~

~~Step 1 (Transform so that RHS only one attribute):~~

~~Step 2 (Remove redundant FDs, assume FD doesn’t exist, does LHS closure contain RHS?):~~

~~Step 3 (Composite LHS, remove each one and check if closure still contains RHS):~~

~~Computing Closures:~~

**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, ~~UID~~ -> date-time, rating, comment

{PName, SName, OID}+ = {PName, SName, OID, UID, date-time, rating, comment}

PName, SName, OID -> date-time, rating, comment

{OID}+ = {OID, UID}

The left-hand side of functional dependency 1 contains a key. FEEDBACK is in the third normal form.

Keys:

PName, SName, OID

~~Minimal Basis:~~

~~Step 1 (Transform so that RHS only one attribute):~~

~~Step 2 (Remove redundant FDs, assume FD doesn’t exist, does LHS closure contain RHS?):~~

~~Step 3 (Composite LHS, remove each one and check if closure still contains RHS):~~