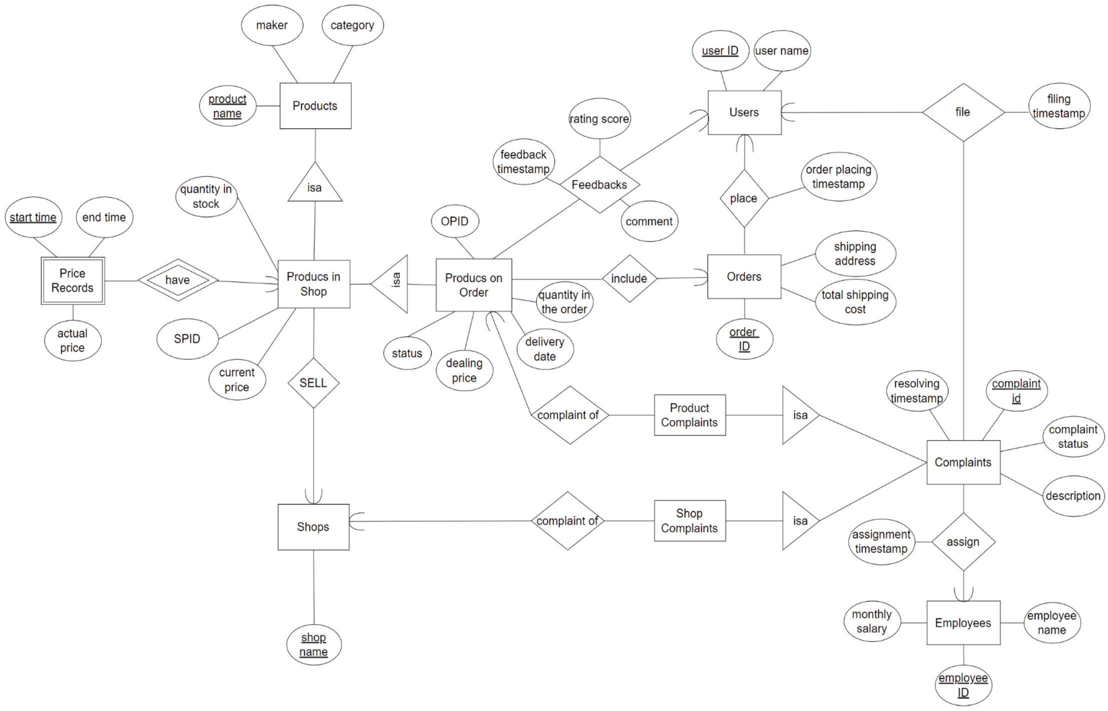
Lab3 DSS2 Team2

A. <u>ER Diagram Design:</u>

*The diagram shows our final version of ER diagram design, some minor adjustments were made for simplification and clarity.



B. <u>Generation of Normalised Database Schemas</u>

Notes:

1. Explanation of acronyms: BCNF stands for Boyce-Codd Normal Form, 3NF stands for Third Normal Form, FD stands for Functional Dependancy.
2. Given BCNF is stricter than 3NF, if a relation is in BCNF, then it must be in 3NF. We will use the property directly in the “Check 3NF” portion.
3. We omitted the SPID and OPID for structural clarity since the subclass entities could be fully defined using existing attributes.

USERS (<u>user_id</u>, name)

Keys	user_id, user_name
Primary Key	user_id
Non-trivial FDs	user_id → user_name

Check 3NF	The relation is in BCNF as **USERS** only has two attributes. Hence, it is 3NF.

SHOPS (<u>shop_name</u>)

Keys	shop_name
Primary Key	shop_name
Non-trivial FDs	None
Check 3NF	The relation is in BCNF as there is no FD associated with it. Hence, it is 3NF.

PRODUCTS (<u>product_name</u>, maker, category)
Assumption for products:
1. product_name is unique for every different product (e.g. apple fruit and apple phone)

Keys	product_name
Primary Key	product_name
Non-trivial FDs	product_name → maker, category
Check 3NF	The relation is in BCNF as every LHS of every non-trivial FD is a superkey.

ORDERS (<u>order_id</u>, shipping_cost, shipping_addr, order_timestamp, user_id)
Assumptions for orders:
1. Users may ask for products to be delivered at multiple different addresses so user_id cannot determine shipping_addr.
2. At one order timestamp, a user can click the button and place only one order so user_id and Date-time can imply order_id. (timestamp is similar to unix epoch time)
3. shipping_addr alone cannot determine shipping_cost as the shipping cost depends on type of product (product_name), the location of the warehouse of shop (shop_name).

Keys	[order_id, {user_id, order_timestamp}]
Primary Key	order_id
Non-trivial FDs	order_id → shipping_cost, shipping_addr, Date-time, user_id {user_id, order_timestamp} → order_id {user_id, order_timestamp} → shipping_cost, shipping_addr

Check 3NF	The relation is in BCNF as every LHS of every non-trivial FD is a superkey, hence it is 3NF.

COMPLAINTS (<u>complaint_id</u>, description, filing_timestamp, complaint_status, employee_id, handling_timestamp, user_id)
Assumptions for complaints:
1. We assume that each user can only file one complaint at one timestamp, and each employee can only handle one complaint at one timestamp. (Timestamp is the unix epoch time)

Keys	[complaint_id, {user_id, filing_timestamp}, {employee_id, handling_timestamp}]
Primary Key	complaint_id
Non-trivial FDs	complaint_id → description, filing_timestamp, complaint_status, employee_id, Handled-date-time, user_id {user_id, filing_timestamp} → complaint_id {employee_id, handling_timestamp} → complaint_id
Check 3NF	All left hand sides of the nontrivial FDs contain a key, hence, the table is in 3NF.

COMPLAINTS-ON-SHOPS (<u>complaint_id</u>, shop_name)

Keys	complaint_id
Primary Key	complaint_id
Non-trivial FDs	complaint_id → shop_name
Check 3NF	The relation is in BCNF as it only contains 2 attributes. Hence, it is 3NF.

COMPLAINTS-ON-PRODUCTS (<u>complaint_id</u>, product_name, order_id, shop_name)

Keys	complaint_id
Primary Key	complaint_id
Non-trivial FDs	complaint_id → order_id, product_name, shop_name
Check 3NF	The relation is in BCNF as every LHS of every non-trivial FD is a superkey. Hence, it is 3NF.

EMPLOYEES (employee_id, employee_name, salary)

Keys	employee_id
Primary Key	employee_id
Non-trivial FDs	employee_id → employee_name, salary
Check 3NF	The relation is in BCNF as every LHS of every non-trivial FD is a superkey. Hence, it is 3NF.

PRODUCTS-IN-SHOPS (<u>product_name</u>, <u>shop_name</u>, price, quantity_in_stock)

Keys	{product_name, shop_name}
Primary Key	{product_name, shop_name}
Non-trivial FDs	{product_name, shop_name} → price, quantity_in_stock
Check 3NF	The relation is in BCNF as every LHS of every non-trivial FD is a superkey. Hence, it is 3NF.

PRODUCTS-IN-ORDERS* (<u>product_name</u>, <u>shop_name</u>, <u>order_id</u>, user_id, price, status, quantity_on_order, delivery_date, feedback_rating_score, feedback_comment, feedback_timestamp)

Assumptions for products-in-orders*:

1. Based on real life scenario, we assume that each user can only provide feedback one time on the products they purchased to prevent "click farming". Thus, feedback is considered as a many-to-one relation between user and product-in-order. And the attributes of feedback and the key of user (user_id) are merged into the product-in-order schema.

Keys	{product_name, shop_name, order_id}
Primary Key	{product_name, shop_name, order_id}
Non-trivial FDs	{order_id, product_name, shop_name} → user_id, price, delivery_date, quantity_on_order, status, feedback_rating_score, feedback_comment, feedback_timestamp order_id →user_id

Check 3NF	order_id →user_id indicates a violation of 3NF. Therefore, we need to perform 3NF normalization.

Step 1: Derive Minimal Basis	1. Transform FDs:
	{order_id, product_name, shop_name} → user_id,
	{order_id, product_name, shop_name} → price,
	{order_id, product_name, shop_name} → delivery_date,
	{order_id, product_name, shop_name} → quantity_on_order,
	{order_id, product_name, shop_name} → status,
	{order_id, product_name, shop_name} → feedback_rating_score,
	{order_id, product_name, shop_name} → feedback_comment,
	{order_id, product_name, shop_name} → feedback_timestamp,
	order_id →user_id
	
	2. Remove redundant FDs
	Except for {order_id, product_name, shop_name} → user_id is implied by order_id →user_id, all FDs are not redundant
	
	3. Remove redundant attributes on left hand side
	No redundancies
	Final minimal basis: {
	{order_id, product_name, shop_name} → price,
	{order_id, product_name, shop_name} → delivery_date,
	{order_id, product_name, shop_name} → quantity_on_order,
	{order_id, product_name, shop_name} → status,
	{order_id, product_name, shop_name} → feedback_rating_score,
	{order_id, product_name, shop_name} → feedback_comment,
	{order_id, product_name, shop_name} → feedback_timestamp,
	order_id →user_id
	}

Step 2: Combine FDs in the minimal basis	The minimal basis becomes: {
	{order_id, product_name, shop_name} → price, delivery_date, quantity_on_order, status, feedback_rating_score, feedback_comment, feedback_timestamp,
	order_id →user_id
	}

Step 3: Create table for remained FDs	The following two tables are formed: **PRODUCTS-IN-ORDERS (<u>product_name</u>, <u>shop_name</u>, <u>order_id</u>, price, status, quantity_on_order, delivery_date, feedback_rating_score, feedback_comment, feedback_timestamp)** **ORDER-USER (<u>order_id</u>, user_id)**
Step 4: Check if original keys are contained	The original key {product_name, shop_name, order_id} is contained in the schema **PRODUCTS-IN-ORDERS**.

Step 5: Remove redundant tables	The schema **ORDER-USER (<u>order_id</u>, user_id)** is deemed <u>redundant</u> since the relation is completely included in the aforementioned schema **ORDERS.** Hence, the table is removed from our final result.
Result	The original table **PRODUCTS-IN-ORDERS*** is decomposed to the following table: **PRODUCTS-IN-ORDERS (<u>product_name</u>, <u>shop_name</u>, <u>order_id</u>, price, status, quantity_on_order, delivery_date, feedback_rating_score, feedback_comment, feedback_timestamp)**

PRODUCTS-IN-ORDERS (<u>product_name</u>, <u>shop_name</u>, <u>order_id</u>, price, status, quantity_on_order, delivery_date, feedback_rating_score, feedback_comment, feedback_timestamp)
Note: The relation is generated due to the normalization of the original relation **PRODUCTS-IN-ORDERS***.

Keys	{product_name, shop_name, order_id}
Primary Key	{product_name, shop_name, order_id}
Non-trivial FDs	{order_id, product_name, shop_name} → price, delivery_date, quantity_on_order, status, feedback_rating_score, feedback_comment, feedback_timestamp
Check 3NF	The relation is in 3NF since the left hand side of the only nontrivial FD contains the key.

PRICE-HISTORY (<u>product_name</u>, <u>shop_name</u>, <u>start_date</u>, end_date, price)
Assumptions for price-history:
1. For simplicity, we assume that the price of a product sold in a shop can only be changed at the start of each day (i.e. the price of a product will remain the same throughout one day)

Keys	[{product_name, shop_name, start_date}, {product_name, shop_name, end_date}]
Primary Key	{product_name, shop_name, start_date}
Non-trivial FDs	{product_name, shop_name, start_date} → price, end_date {product_name, shop_name, end_date} → price, start_date
Check 3NF	The relation is in BCNF as every LHS of every non-trivial FD is a superkey.

APPENDIX C: INDIVIDUAL CONTRIBUTION FORM

Name	Individual Contribution to Submission 1 (Lab 1)	Percentage of Contribution	Signature
cheng zhengxing	join group discussion, suggest possible improvements, modify the details of ER graphics	16.66%	
zhang tianyu	join group discussion, suggest possible improvements, modify the details of ER graphics	16.66%	
zhou runbing	join group discussion, suggest possible improvements, modify the details of ER graphics	16.66%	
chirstopher arif setiadharma	double-checking the relationships and entity attributes	16.66%	
an ruyi	design the ER relation, create skeleton of design, create ER diagram, write report, join group discussion	16.66%	
peng wenxuan	design the ER relation, create skeleton of design, create ER diagram, write report, join group discussion	16.66%	彭文轩

Name	Individual Contribution to Submission 2 (Lab 3)	Percentage of Contribution	Signature
cheng zhenxing	Write up for entities sets, analyze their attributes and FDs, join group discussion	16.66%	
zhang tianyu	Write up for entities sets, analyze their attributes and FDs, join group discussion	16.66%	
zhou runbing	Write up for entities sets, analyze their attributes and FDs, join group discussion	16.66%	
chirstopher arif setiadharma	Write up for entities sets, analyze their attributes and FDs, join group discussion	16.66%	
an ruyi	Write up for entities sets, analyze their attributes and FDs, join group discussion	16.66%	
peng wenxuan	Write up for entities sets, analyze their attributes and FDs, join group discussion	16.66%	彭文轩

Name	Individual Contribution to Submission 3 (Lab 5)	Percentage of Contribution	Signature