## Step 1: Mapping of regular entity types

Regular entities are mapped in this step. Customer and Admin_staff will be mapped in the next step since they are overlapping specialised entities.

**USER**

| user_id | Name | Surname | User_type | Email | Password | Phone_number | APIKey | Street_number | Street_name | Suburb | City | ZIP_code |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

**Rating**

| rating_id | Review | Score |
|---|---|---|

**Products**

| product_id | Name | Price | Description |
|---|---|---|---|

**Retailer**

| retail_id | Name | Email | Street_number | Street_name | Suburb | City | ZIP_code |
|---|---|---|---|---|---|---|---|

## Step 8: Mapping specialisation and generalisation

We need to do this step early because we need to use CUSTOMER in the coming steps. User is totally overlappingly specialised into two types: CUSTOMER and ADMIN_STAFF. We use approach 8A in the slides of L14: That is, create a relation for each subclass (customer and admin_staff) with the key of the superclass (user) and the attributes of this subclass. The key of the superclass will be the key of the subclasses, and links it to the superclass.

**USER**

| user_id | Name | Surname | User_type | Email | Password | Phone_number | APIKey | Street_number | Street_name | Suburb | City | ZIP_code |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

**CUSTOMER**

| user_id | Profile_picture |
|---|---|

**ADMIN_STAFF**

| user_id | Salary | Position |
|---|---|---|

**RATING**

| rating_id | Review | Score |
|---|---|---|

**PRODUCT**

| product_id | Name | Price | Description |
|---|---|---|---|

**RETAILER**

| retail_id | Name | Email | Street_number | Street_name | Suburb | City | ZIP_code |
|---|---|---|---|---|---|---|---|

## Step 2: Mapping of weak entity types

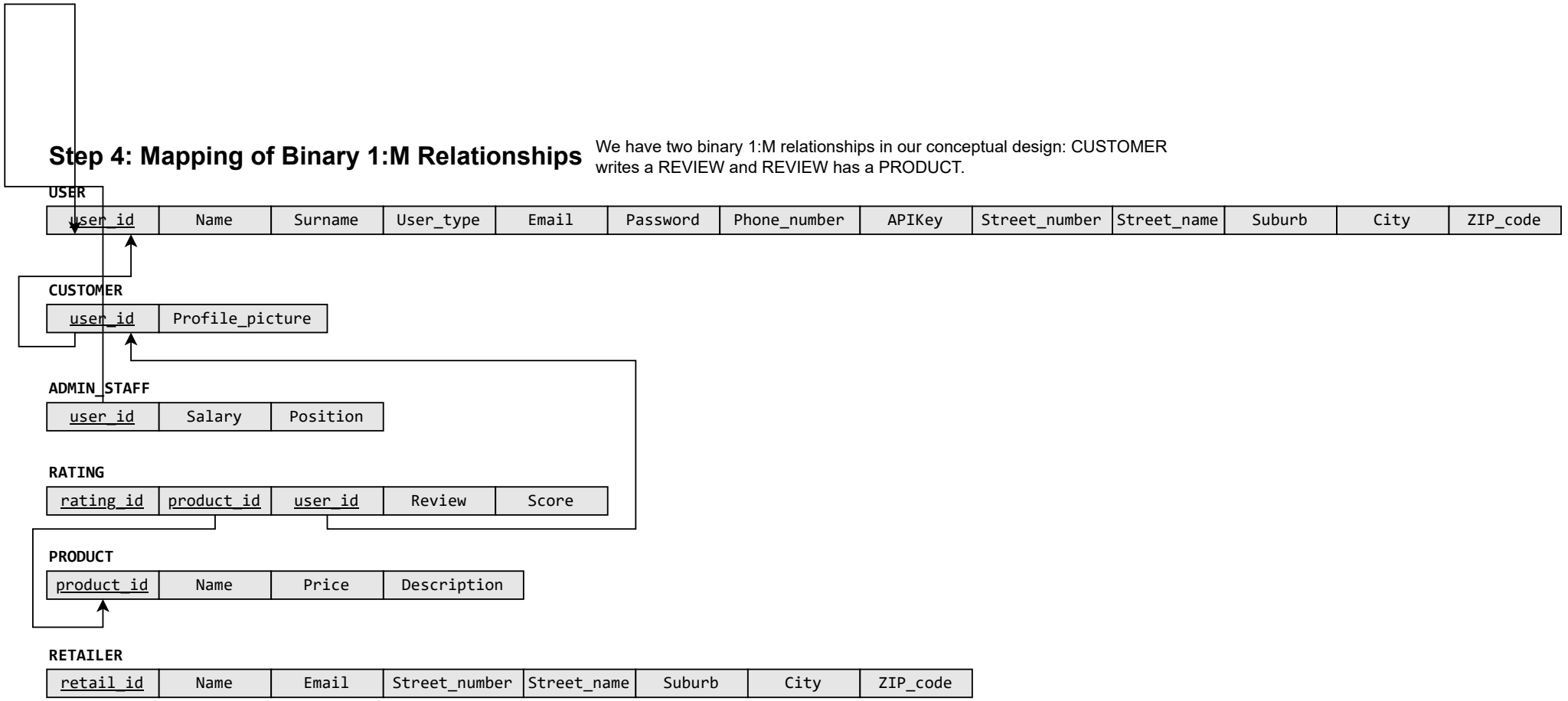
We do not have weak entities in our conceptual design. We do nothing in this step. RATING is not a weak entity since ratings can be updated over time, ratings may need to be deleted, editied or moderated independetly of the user or product that it is linked to.

## Step 3: Mapping of Binary 1:1 Relationships

We do not have binary 1:1 relationships in our conceptual design. We do nothing in this step.

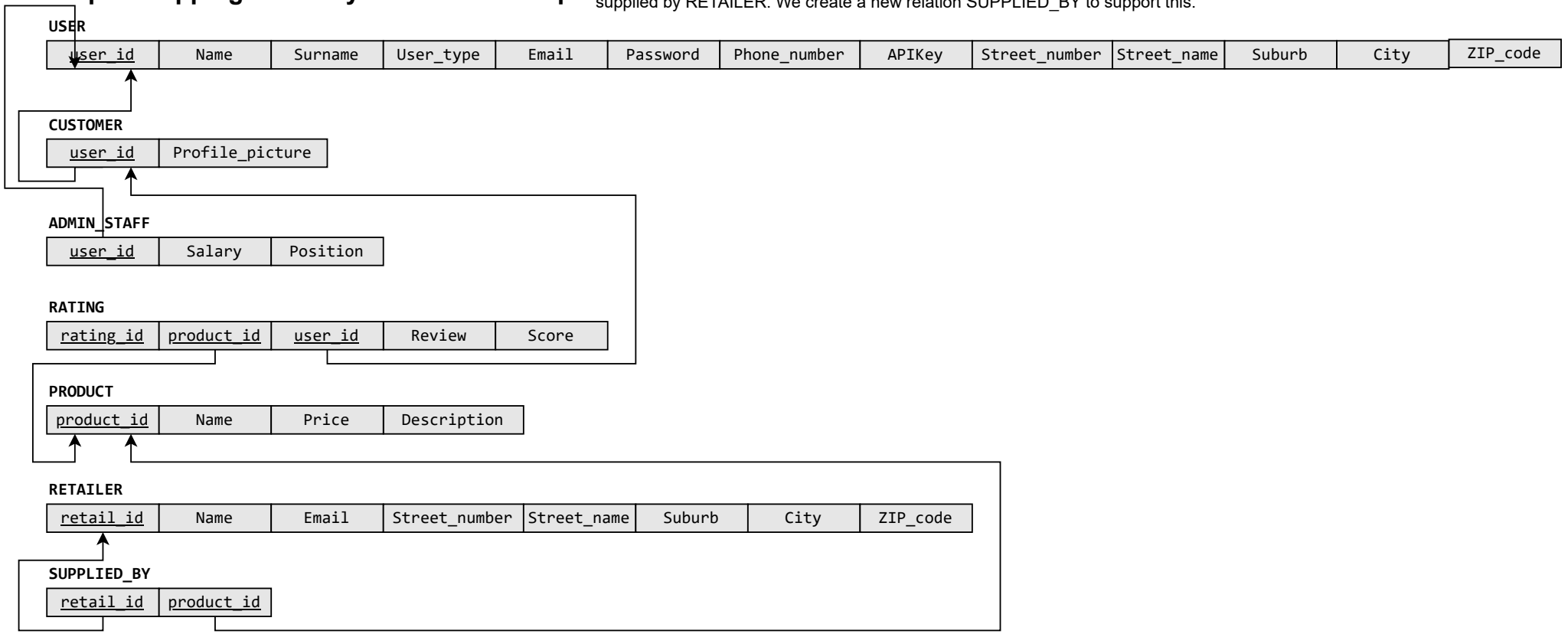
## Step 4: Mapping of Binary 1:M Relationships

We have two binary 1:M relationships in our conceptual design: CUSTOMER writes a REVIEW and REVIEW has a PRODUCT.



## Step 5: Mapping of Binary M:N Relationships

We have one binary M:N relationships in our conceptual design: PRODUCT supplied by RETAILER. We create a new relation SUPPLIED_BY to support this.




## Step 6: Mapping of multivalued attributes

We do not have multivalued attributes in our conceptual design. We do nothing in this step.

## Step 7: Mapping of N-ary relationships

We do not have N-ary relationships in our conceptual design. We do nothing in this step.

## Step 8: Mapping specialisation and generalisation

Step 8 was already done directly after step 1 to ensure that CUSTOMER exists before we map relationships. We do not have other specialisation in our conceptual design.

## Step 9: Mapping unions

We do not have unions in our conceptual design. We do nothing in this step.