Step 1: Mapping of regular entity types Regular entities are mapped in this step.

rate_limits

ip_address	endpoint	request_count	window_start	last_request

User

id	name	surname	user_type	email	password	salt	phone_number	apikey	street_number	street_name	suburb	city	zip_code

Rating

id	score	description	updated_at

Product

id	name	description	Image_url	category

Retailer

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 disjoint 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.

rate_limits

ip_address	endpoint	request_count	window_start	last_request

User

id	name	surname	user_type	email	password	salt	phone_number	apikey	street_number	street_name	suburb	city	zip_code

Customer

user_id

Admin_staff

user_id	salary	position

Rating

id	score	description	updated_at

Product

id	name	description	Image_url	category

Retailer

id	name	email	street_number	street_name	suburb	city	zip_code

Step 2: Mapping of weak entity types We map weak entities in this step. WATCHLIST is the only weak entity in our EERD, since it depends on both a CUSTOMER and a PRODUCT to exist: Without them, the watchlist entry cannot exist. RATING is not a weak entity since ratings can be updated over time, ratings may need to be deleted, edited or moderated independetly of the user or product that it is linked to.

rate_limits

ip_address	endpoint	request_count	window_start	last_request

User

id	name	surname	user_type	email	password	salt	phone_number	apikey	street_number	street_name	suburb	city	zip_code

Customer

user_id

Admin_staff

user_id	salary	position

Rating

id	score	description	updated_at

Product

id	name	description	Image_url	category

Retailer

id	name	email	street_number	street_name	suburb	city	zip_code

watchlist

cust_id	product_id	created_at

Step 3: Mapping of Binary 1:1 relationships We do not have binary 1:1 relationships in our EERD. We do nothing in this step

Step 4: Mapping of Binary 1:M relationships We have four 1:M relationships in our EERD: CUSTOMER creates WATCHLIST, PRODUCT is_invloded WATCHLIST (both of these are already mapped in Step 2), CUSTOMER writes_a RATING and PRODUCT has RATING. We use Approach 1 in the slides of L14

rate_limits

ip_address	endpoint	request_count	window_start	last_request

User

id	name	surname	user_type	email	password	salt	phone_number	apikey	street_number	street_name	suburb	city	zip_code

Customer

user_id

Admin_staff

user_id	salary	position

Rating

id	product_id	user_id	score	description	updated_at

Product

id	name	description	Image_url	category

Retailer

id	name	email	street_number	street_name	suburb	city	zip_code

watchlist

cust_id	product_id	created_at

Step 4: Mapping of Binary N:M relationships We have one binary N:M relationship in our EERD: PRODUCT supplied_by RETAILER. We create a new relation called SUPPLIED_BY to support this and we add a price attribute since it is an attribute that belongs to the supplied_by relationship.

rate_limits

ip_address	endpoint	request_count	window_start	last_request

User

id	name	surname	user_type	email	password	salt	phone_number	apikey	street_number	street_name	suburb	city	zip_code

Customer

user_id

Admin_staff

user_id	salary	position

Rating

id	product_id	user_id	score	description	updated_at

Product

id	name	description	Image_url	category

Retailer

id	name	email	street_number	street_name	suburb	city	zip_code

watchlist

cust_id	product_id	created_at

Supplied_By

Retailer_id	Product_id	Price

Step 6: Mapping of multivalued attributes We do not have multivalued attributes in our EERD. We do nothing in this step.

Step 7: Mapping of N-ary relationships We do not have N-ary relationships in our EERD. 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 specialisations in our EERD.

Step 9: Mapping unions We do not have unions in our EERD. We do nothing in this step.

FINAL RELATIONAL MAPPING

rate_limits

ip_address	endpoint	request_count	window_start	last_request

User

id	name	surname	user_type	email	password	salt	phone_number	apikey	street_number	street_name	suburb	city	zip_code

Customer

user_id

Admin_staff

user_id	salary	position

Rating

id	product_id	user_id	score	description	updated_at

Product

id	name	description	Image_url	category

Retailer

id	name	email	street_number	street_name	suburb	city	zip_code

watchlist

cust_id	product_id	created_at

Supplied_By

Retailer_id	Product_id	Price