THE UNIVERSITY OF QUEENSLAND
AUSTRALIA
CREATE CHANGE

INFS1200 Assignment 1: Modules 1 and 2

Due: Friday 5th September @ 3 PM — Weighting: 25%

Full Name	Student ID
Caleb Yates	*S49886351*

Overview

This assignment has two sections. **Section 1** will test your knowledge on conceptual modelling via the Entity Relationship (ER) model and ability to apply it for capturing important aspects of a system that need to be stored in a database. **Section 2** of this assignment tests your knowledge on the relational model and your ability to map an ER model to a relational schema.

Assignment 1 is worth 25 course marks. Marks for individual questions are as shown. Please consult the course profile for late penalties that apply to this assessment item. This assignment must be completed **individually**.

Submission

All parts of Section 1 and Section 2 will be submitted through an electronic marking tool called Gradescope, which will also be used for marking and providing grades. Work on and submit your assignment as follows:

1) Enter your answers in the Word version of this document.
2) Save the document as a PDF and upload that PDF to Gradescope. Ensure that you are submitting to the correct portal. Ensure all of your answers are visible in the PDF.

Do not alter the layout of this document; altering it in anyway may attract penalties. Changes that do not alter the document's layout (e.g. highlighting) are acceptable. Do not submit your answers in a separate document. Do not hand-write your answers.

Academic integrity and plagiarism

The University has strict policies regarding academic integrity and plagiarism. Penalties for engaging in unacceptable behaviour can range from loss of grades in a course to expulsion from UQ. You are required to read and understand the policies on academic integrity and plagiarism in the course profile (Section 6.1). If you have any questions regarding an acceptable level of collaboration with your peers, please see either the lecturer or your demonstrator for guidance. Remember that ignorance is not a defence!

In particular, you are permitted to use generative AI tools to help you complete this assessment task. However, if you do, **submit complete copies of your interactions with any AI tools** in the Generative AI submission portal provided on Gradescope. An AI documentation template will be provided on Blackboard; see that document for instructions regarding how to reference Generative AI for this assignment. If you have not used generative AI, you do not need to submit anything in the Generative AI submission portal.

Please note that if you use generative AI but fail to acknowledge this by submitting your interactions, it will be considered misconduct as you are claiming credit for work that is not your own, and may also constitute student misconduct under the Student Code of Conduct.

Section 1 – Conceptual Modelling (12 Marks)

Questions in Section 1 contain a brief UoD that provides contextual information regarding a system or organisation. You will need to create an Entity Relational (ER) diagram or Extended Entity Relational (EER) diagram based on the UoD, using the notation taught in this course. If the UoD is unclear regarding specific aspects of the brief, you may note assumptions alongside your diagram. Your assumptions must not conflict with or violate any aspects of the UoD. Only make assumptions if you think they are necessary to complete your diagram. Your diagram must adhere to the notation guide provided on Blackboard.

Question 1.1 [4 marks]

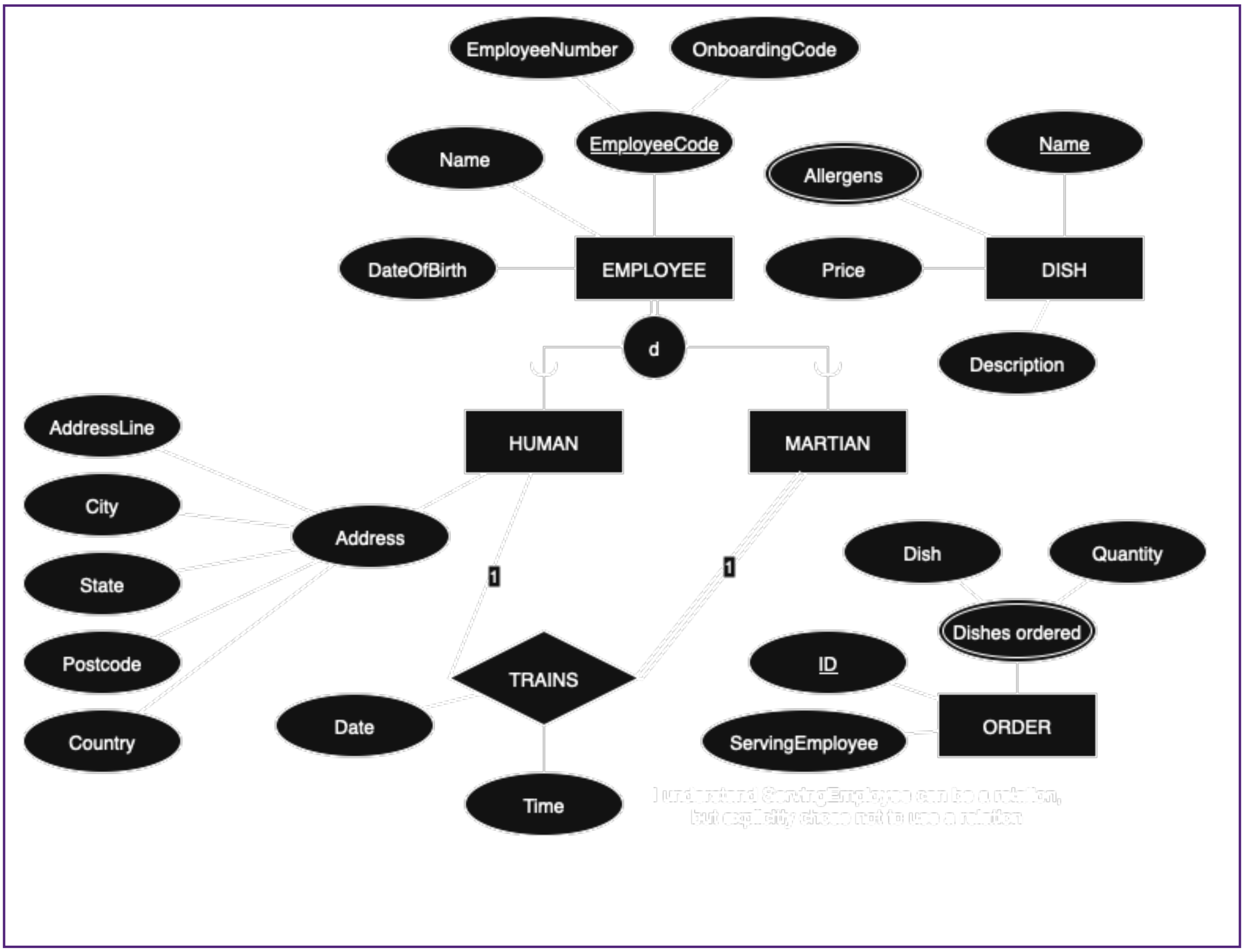
SpaceMex is soon to be the first taco stand franchise on Mars. The details of their database are as follows:

Employees are identified by their employee code, which is a combination of their employee number and their onboarding code. Their name and date of birth is also stored. Employees are either humans or Martians. For human employees, their Earth address is stored, consisting of an address line, city, state, postcode and country.

Martians are uneducated in the art of taco making, so every Martian employee must be trained by a human employee. The date and time of the training is also recorded.

Dishes at *SpaceMex* each have a unique name, as well as a price and description. For each dish, a list of potential allergens is stored.

Orders have unique order IDs. For each order, the dishes included in the order are recorded, as well as the employee who served the order. An order can contain multiple servings of the same dish, so we record the quantity of each dish included in the order.



I understand ServingEmployee can be a relation, but explicitly chose not to use a relation

Question 1.2 [5 marks]

Hilbert's Hotel requires a relational database to keep track of their very large number of guests. The details of this database are as follows:

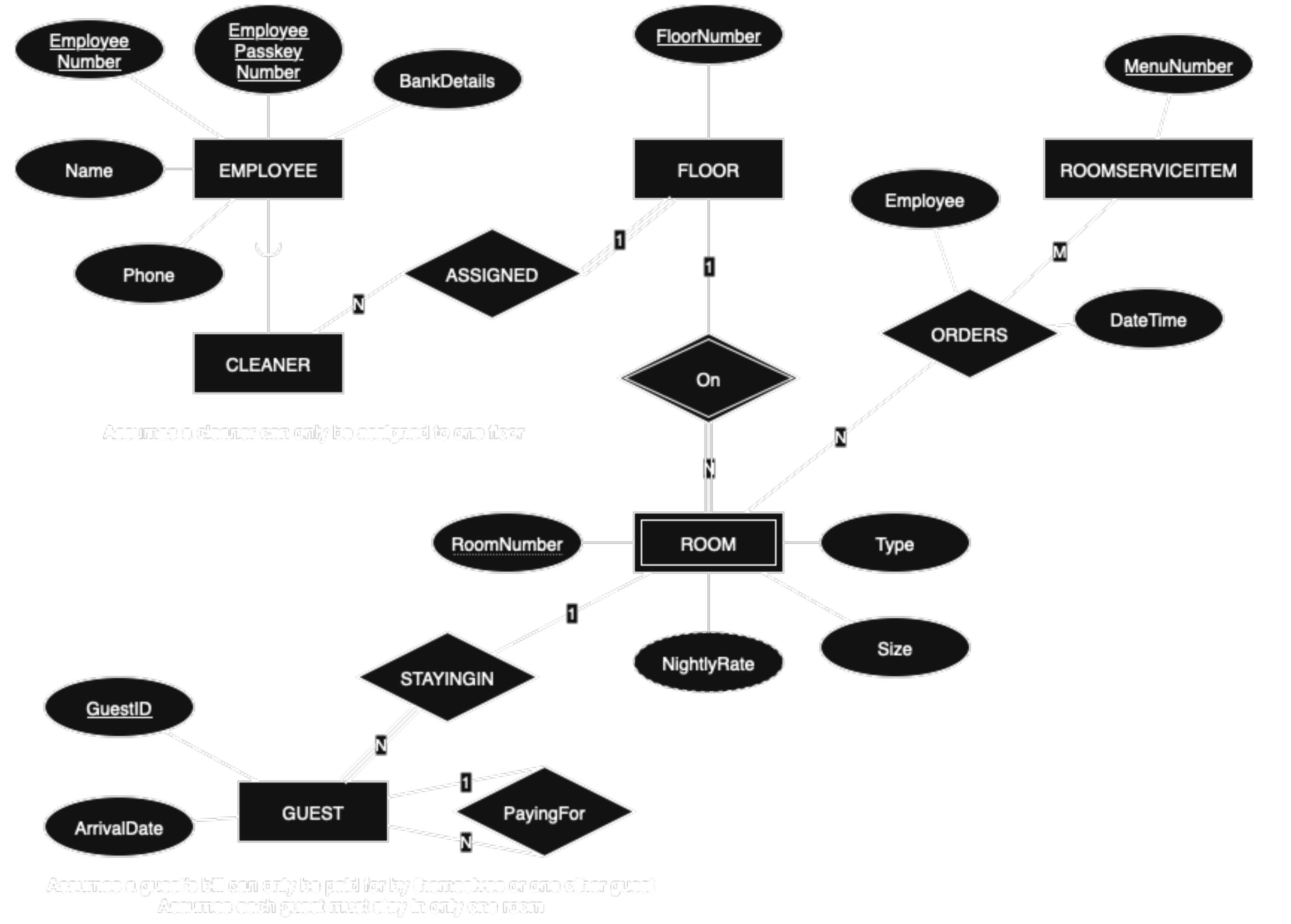
Employees at the hotel each have a unique employee number and a unique passkey number. An employee's name, phone number and bank details are also stored. Some of these employees are cleaners.

Floors at the hotel are identified by their floor number. Each floor must have at least one assigned cleaner.

Each room is located on some floor of the hotel. On any given floor, all room numbers are unique, however rooms on different floors may have the same room number. Each room has its room type and size recorded. Each room's nightly rate (cost of the room per night) can be calculated from the room type and size.

Guests are assigned a unique guest ID. Each guest's arrival date is recorded. The database also tracks which room each guest is currently staying in. Guests may choose to pay for the bills of other guests. If so, the database keeps track of who is paying for whom.

Items on the room service menu each have a unique menu number. When a room places an order for an item on the menu, the database records which room the order was for, which employee delivered the order, the item ordered, and the date and time at which the order was made. Rooms have unlimited access to room service.



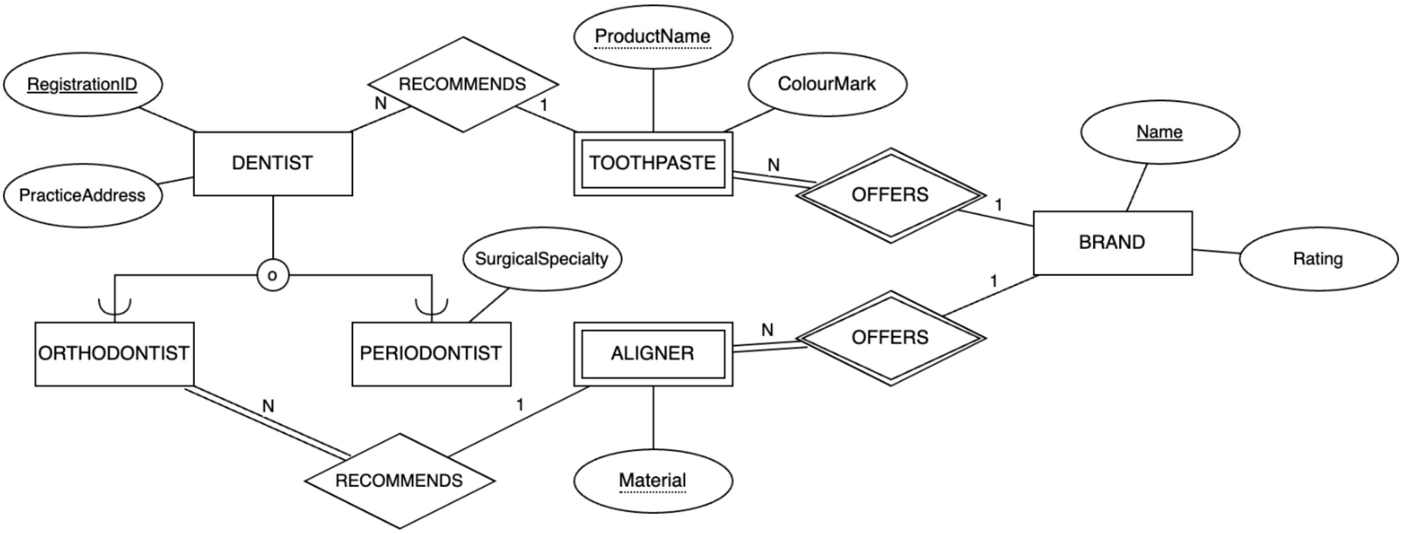
Assume a cleaner can only be assigned to one floor

Assume a guest's bill can only be paid for by themselves or one other guest

Assume each guest must stay in only one room

Question 1.3 [3 marks]

The *Domestic Dental Database* stores information about dentistry and dental products in Australia. An ER diagram for this database is shown below.



Evaluate the following statements and state whether each statement is correct or incorrect. A statement that cannot be inferred from the ER diagram should be considered incorrect. Include a brief justification for each of your responses. A justification is required for full marks. Your justification should include evidence from the ER diagram (where applicable) and shouldn't just restate the question. Each response must be at most 80 words long.

Example: Each brand must have a different name.

Correct. The BRAND entity type has a Name attribute with a solid underline, meaning it is a key so it must be unique.

A) Every dentist must recommend a toothpaste.

Incorrect. The constraints of the relationship RECOMMENDS between the DENTIST entity and the TOOTHPASTE entity is optional, as there is no existential constraint between them (which would have been notated with a double line between DENTIST and RECOMMENDS)

B) Toothpastes offered by different brands may have the same product name

Correct. As TOOTHPASTE is a weak entity, its (candidate) key attribute ProductName must only be unique per BRAND, since its defining relationship is BRAND *OFFERS* TOOTHPASTE. It is a weak entity because its rectangle is outlined twice, and the relationship OFFERS is also outlined twice, making that relationship the defining relationship

C) An orthodontist may also be a periodontist.

Correct. The PERIODONTIST subclass and the ORTHODONTIST subclass are not disjoint, which would have instead been labelled with a ‘d’ instead of an ‘o’ as is the case

D) It is possible for a dentist to be neither an orthodontist nor a periodontist.

Yes, this is possible. “By default” subclasses are optional when single lines are used, if a double line indicating a required subclass was used then this would not be the case

E) Multiple orthodontists may recommend the same aligner.

Correct. The ORTHODONTIST subclass has a cardinality constraint between itself and the ALIGNER weak entity which only requires each ORTHODONTIST to recommend exactly one ALIGNER, but allows for any number of recommendations for a single ALIGNER

F) Every orthodontist has a registration ID.

Correct. From the principle of generalisation, every ORTHODONTIST is a DENTIST, and every DENTIST must have a RegistrationID since RegistrationID is a (candidate) key attribute which is always required

Section 2 – The Relational Data Model (13 Marks)

Question 2.1 [3 marks]

Using the contextual information, relational schema and instance data provided, identify all constraint/s violated by each of the following operations or state that none are violated. For each constraint that is violated, provide reasoning as to how it is violated with reference to the relevant data. Reasoning is not needed for constraints that aren't violated. An example of how to answer the questions is provided below. Consider each operation independently. Do not take into consideration changes from earlier questions. Each response must not exceed **100 words** in length.

You may assume that all attributes in the database have NULL as an acceptable value in their domain *except* in the cases where the contextual information indicates otherwise. You should not assume the existence of any additional constraints which are not stated or implied by the given information.

Contextual Information

The database represents a subset of data related to a recyclable container deposit scheme.

The **User** table serves as a repository for all users participating in the container deposit scheme. It holds each user's unique userID (string of five digits), name (non-empty string), email address (non-empty string), and contact phone number (string of ten digits).

The **Container** relation contains details about the containers that are eligible for deposit within the scheme. Each container is assigned a unique containerID (string of five digits) and has a material (either "Glass", "Aluminium", "Steel", "HDPE" or "LDPE"), capacity (positive integer, represents an amount in millilitres) and refund value (positive number with two decimal places, represents an amount in dollars).

The **Location** relation holds information regarding the various locations where users can deposit containers as part of the scheme. It stores a unique locationID (string of five digits) for each location, along with a location name (non-empty string) and a contact phone number (string of ten digits).

The **Deposit** relation records the deposits made by users that are using the container deposit scheme, where there is one record per container deposited. Each deposit is identified by a userID (of the user making the deposit), the container (via the containerID) deposited and a timestamp (depositDateTime). The date and time of the deposit is recorded in a "YYYY-MM-DD HH:MM:SS" format. The deposit's locationID (not NULL, linking it to the Location relation) is also recorded, referencing the location at which the deposit was made. **Each user can make at most one deposit per day.**

Relational Schema

User [userID, name, email, phoneNumber]

Container [containerID, material, capacity, refundValue]

Location [locationID, locationName, phoneNumber]

Deposit [userID, containerID, depositDateTime, locationID]
Deposit.containerID references Container. containerID
Deposit.locationID references Location.locationID
Deposit.userID references User.userID

Instance Data

User

userID	*name*	*email*	*phoneNumber*
31415	Grace Smith	grace@yahoo.com	0712345678
26589	Jack Smith	brotherofgrace@yahoo.com	0487654321
24444	Levi Ramos	levi.tation@gmail.com	0733653333

Container

containerID	*material*	*capacity*	*refundValue*
27182	Glass	1000	0.20
81828	Glass	200	0.10

Location

locationID	*locationName*	*phoneNumber*
10234	Central Brisbane Container Centre	0733651111
23333	Sunshine Coast Containers	0754601111
25555	Contain Your Excitement	1800080123

Deposit

userID	*containerID*	*depositDateTime*	*locationID*
31415	27182	2022-02-20 20:03:42	23333
31415	81828	2022-02-21 20:04:11	10234
24444	27182	2022-02-22 20:04:11	25555

Example: Insert (31415, “Bob Ross”, “rossinator@bigpond.com”, 0415951951) into *User*.

Constraint/s violated (if any) and reasoning (if applicable):

Key Constraint Violation: The primary key for the User relation is userID but there already exists a tuple with userID = 31415.

A. Insert (54321, 23333, “Yesterday”, 87654) into *Deposit*.

Constraint/s violated (if any) and reasoning (if applicable):

Referential integrity violated: the containerID of the inserted tuple 23333 doesn’t exist in the Container table

Referential integrity violated: locationID 87654 doesn’t exist in Location table

Domain constraint violation: “Yesterday” isn’t in the correct date:time format

B. Insert (31415, 27182, “2022-02-20 20:30:15”, 56789) into *Deposit*.

Constraint/s violated (if any) and reasoning (if applicable):

Semantic constraint violation: users should only be able to deposit once per day, and userID=31415 already deposited about 27 minutes before the new tuple

C. Delete (31415, 27182, “2022-02-20 20:03:42”, 23333) from *Deposit*.

Constraint/s violated (if any) and reasoning (if applicable):

Key constraint violation: All three of the key attributes userID, containerID and depositeDateTime are identical, and keys cannot be identical

D. Modify (24444, **27182**, “2022-02-22 20:04:11”, 25555) to (24444, **NULL**, “2022-02-22 20:04:11”, 25555) in *Deposit*.

Constraint/s violated (if any) and reasoning (if applicable):

Domain constraint violation: containerID cannot be NULL

E. Delete (27182, “Glass”, 1000, 0.20) from *Container*.

Constraint/s violated (if any) and reasoning (if applicable):

Assuming the deletion behaviour is “restrict”, this would be a Referential Integrity Violation as there are two deposits referencing the container that is trying to be deleted

If the deletion behaviour is “cascade” this would be allowed, but would delete two deposits which is likely undesirable

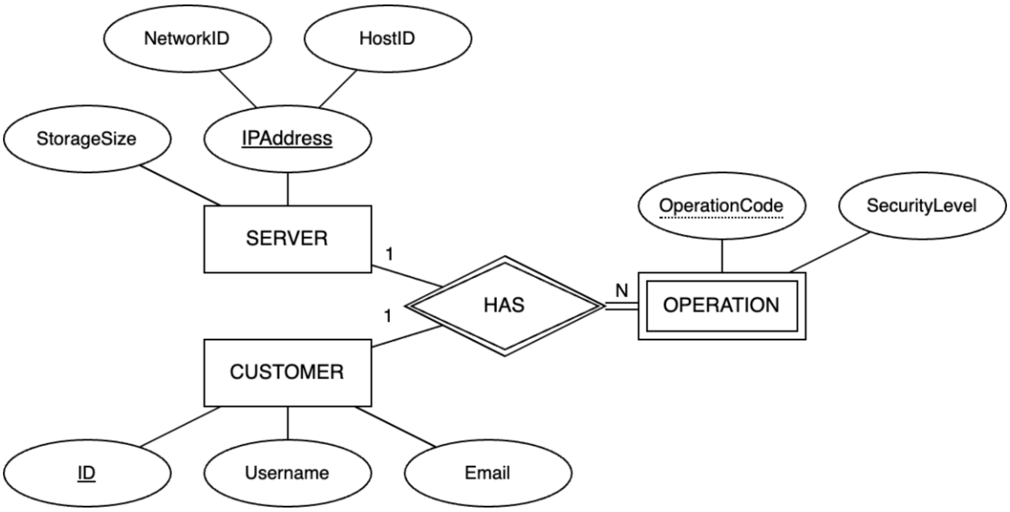
F. Insert (29876, “Wood”, 400, 0.15) into *Container*.

Constraint/s violated (if any) and reasoning (if applicable):

Domain Contraint Violation: The container material must be either “Glass”, “Aluminium”, “Steel”, “HDPE” or “LDPE”, therefore "Wood" is not a valid container material

Question 2.2 [4 marks]

Super Safe Servers provides tech services to IT hobbyists and specialists in Brisbane. They have developed the ER diagram below. Convert this ER diagram to a relational schema, including any foreign keys that are created in the process. Follow the notation guide provided on Blackboard.



Server [networkID, hostID, storageSize]

Customer [id, username, email]

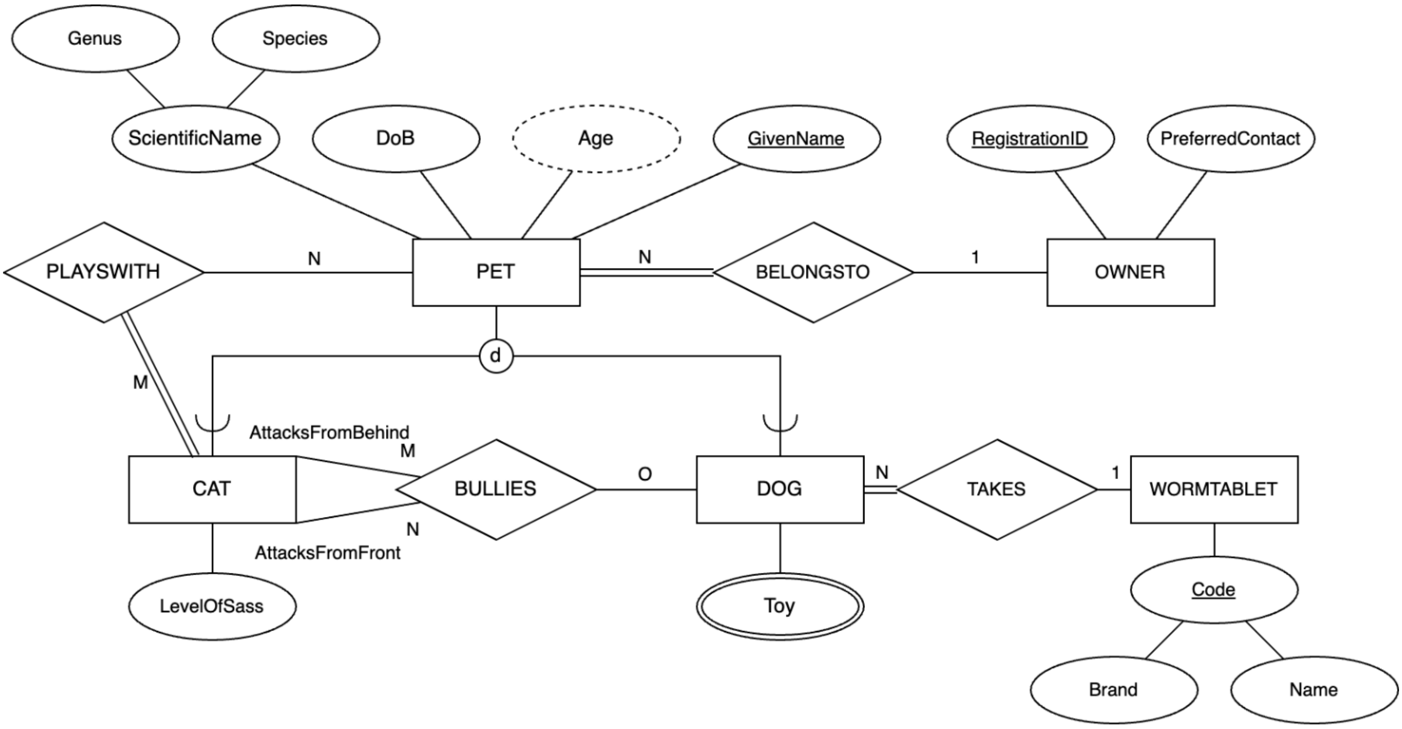
Operation [operationCode, serverNetworkID, serverHostID, customerID, securityLevel]

Operation.{serverNetworkID, serverHostID} references Server.{networkID, hostID}

Operation.customerID references Customer.ID

Question 2.3 [6 marks]

The Pet People have developed the ER diagram below to keep track of their pets. Convert this ER diagram to a relational schema, including any foreign keys that are created in the process. Follow the notation guide provided on Blackboard.



Owner [registrationID, preferredContact]

Pet [givenName, DoB, genus, species, ownerRegistrationID]
Pet.ownerRegistrationID references Owner.registrationID

Cat [givenName, levelOfSass]
Cat.givenName references Pet.givenName

PlaysWith [catGivenName, petGivenName]
PlaysWith.catGivenName references cat.givenName
PlaysWith.petGivenName references Pet.givenName

WormTablet [brand, name]

Dog [givenName, wormTabletBrand, wormTabletName]
Dog.givenName references Pet.givenName
Dog.wormTabletBrand references WormTablet.brand
Dog.workTabletName references WormTablet.name

DogToys [dogGivenName, toy]
DogToys.dogGivenName references Dog.givenName

Bullies [catAttacksFromBehind, catAttacksFromFront, dogGivenName]
Bullies.catAttacksFromBehind references Cat.givenName
Bullies.catAttacksFromFront references Cat.givenName
Bullies.dogGivenName references Dog.givenName