ISYS_Databases_Concepts_Assessment1

Name: Mrwan Alhandi

ID: 3969393

---

databases_projects

links:
1databases_projects
tags:

Overview

The objective of this assignment is to measure your understanding of the basic concepts in the relational database model and using entity-relationship model for database design. The assessment is in two parts, split into four tasks which cover Basic ER Modelling and Basic Relational Modelling.

Elite Construction Case Study

Elite Construction (EC) is an Australian building construction company. The following are the requirements for managing data about staff, clients, construction projects and contractors for EC.

Construction Projects

EC is responsible for managing construction projects throughout Australia. Each construction project has a unique address and the system records the number of rooms and floors the construction has. For each construction project there is a contract which has a start date, end date and a total fee.

Staff

It is important to distinguish between various staff involved in contruction projects. There are project managers that are responsible for overseeing the management of various construction projects – such as organising sub contracts for completing various tasks in the contruction of the project, as well as liasing with the client. They are assigned when the client enters into a contract. In addition there are contractors that are responsible for working on various sub contract tasks associated with the contruction project. Project managers have a distinct employee ID and have a name. Contractors have an Australian Business Number (ABN) and a phone number.

Sub Contracts

A contract consists of several sub contracts which outline the details of the task to be completed (eg Plastering), its fee and the outstanding balance (identifying how much of the fee is still unpaid). One contractor is responsible for the supervision of the sub contract task and must be identified when the sub contract is created. There are many contractors working on the sub contract task itself (the supervisor also works on the sub contract task).

Client

Each client has a unique ID and has a name. The date that clients enter into a contract for a specific project is recorded – note that this may be before the start date of the contract itself. Furthermore, clients nominate a specific bank account which will be used for payment of all contracts that they enter into. The account has a BSB and an account number.

Faster Food Case Study

Faster Food is an Australian food delivery company that allows customers to order food delivery from selected resturants.
You are asked to design a database for managing customer orders. Requirements for the database are as follows:

- Customers order the delivery of various menu items available from resturants. For each customer the database records their credit card number, its expiry date, their name, address and phone number.
- Menu items have a name (eg Vegetarian Thai Green Curry), a description, its cost and preparation time.
- Menu items are prepared by restaurants, each with a unique Australian Business Number (ABN), and also have an address and food style (eg Thai). There could be many resturants that prepare the same menu item (but with potentially different descriptions, costs and preparation times).

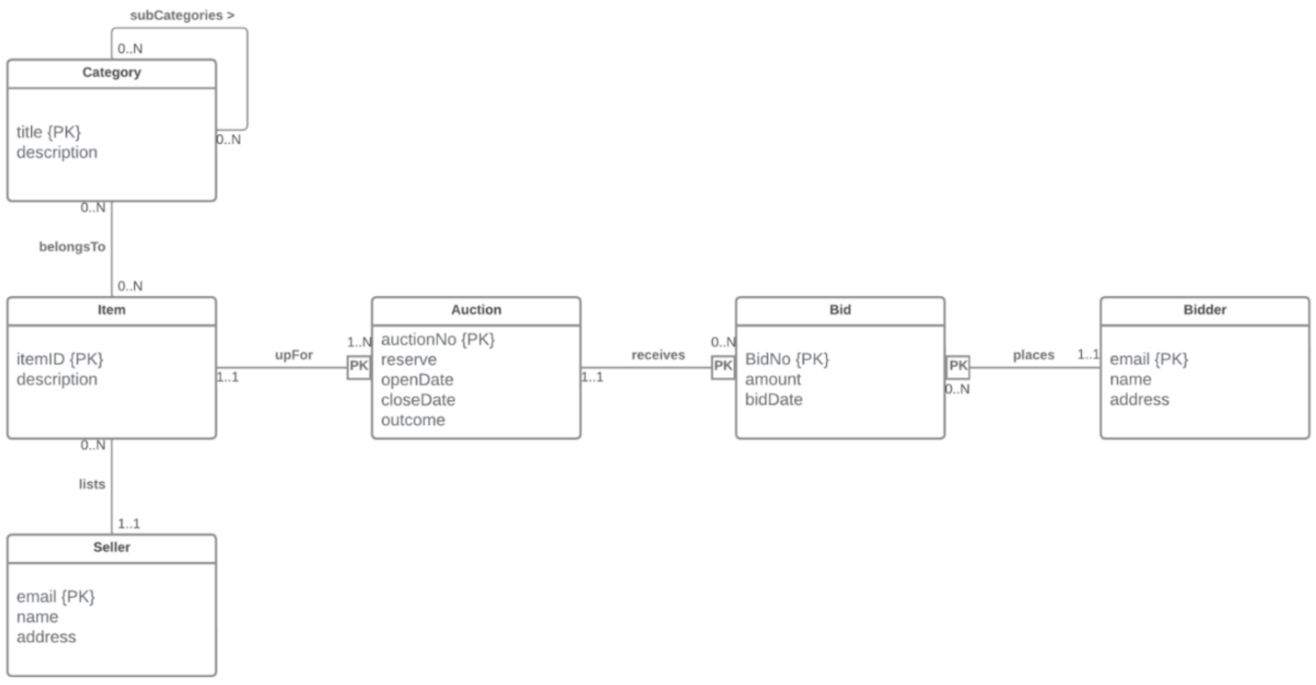
the same menu item (but with potentially different descriptions, costs and preparation times).

- A single menu item can be a part of many orders, with each order, the order date, time, status and quantity (of the menu item) is recorded. Each order is made by a specific customer.

After presenting your ER model to Faster Foods management, you are asked if it can be used to perform the following additional tasks.

- Multiple ordered items can be combined into a single delivery which are are delivered to the customer via a driver. For each delivery, there is a unique ID, a date, time, fee.
- Customers can leave a rating for the driver and the menu item that they received.
- Drivers have a Tax File Number (TFN), a name and a vehicle registration.
- Can your design enforce all ordered items in a delivery to be sourced from only one restaurant?

Consider the following ER diagram, which shows aspects online auction system.



Part A: Entity-Relationship Modelling

Requirements:

- Task 1: design and plan for the implementation of the database system and use UML
- Task 2: ER diagram
- Task 3: Map ER diagram into relational database schema

Task 1: Designing an Entity-Relationship Model

The database development life cycle can consist of eight phases:
1. Planning
2. Requirement Gathering
3. Conceptual Design
4. Logical Design
5. Physical Design
6. Construction
7. Implementation and rollout
8. Ongoing support

This task look into the first three phases which are:
1- Planning
2- Gathering requirements.
3- Conceptual Design

Step one: Planning

This is when an organisation determines whether they actually have a need for a database, what it is goals should be, and the feasibility of successfuly accomplishing those goals. The goal for EC is to build a database that can manage data about staff, clients, construction projects, and contractors.

Step two: Gathering Requirements

Assuming this is a real world case, the gathering requirements must undergo the following steps:

1- Writing a mission statement.
2- Writing mission objectives.
3- Searching and examining current database/outdated database.
4- Allocate stakeholders (actors).
5- Identify their tasks.
6- Gather details about the buisness by doing interviews with those "subject matter experts".
7- Identify how to deal with historical data.
8- Identify data exceptions and irregularities.

A mission statement is used to guide the database development process and guard from scope creep. It's developed with project initiator and its focused on the goal. From the given information in the assignment document, an appropriate mission statement for the database is: Building a database for the company EC to manage data about staff, clients, constuction projects, and contractors.

A mission obective is a short sentence that supports the mission statement. Usually, it represents the general tasks users can perform against the data in the database. In this case, a mission objective from the information given in the assessment is not defined. An interview with the company experts is required to identify the mission objective.

Step three can be skipped since the company did not state any information about old databases.

The stakeholders also called actors are those who will interact with the system. EC has identified the following actors:

- Project managers: responsible for overseeing the management of various construction projects.
- Contractors: responsible for working on various sub contract tasks associated with the construction project.
- Supervisor: works with contractors on the sub contract task.
  Clients: the clients that sign a contract with EC.

An assumption can be made that the information given in the assessment was collected from an interview.

Step seven can be skipped since there's no current or old databases and the company did not state any requirements about historical data.

The company EC did not state any data exceptions and irregularities. Further interviews can be done identify any if exist.

After the gathering phase is completed and the requirements are clear, the developer can proceed to the next step which is creating an Entity-Relationship Model. To create the model, the developer must identify all the entities, entities attributes, relationships, relationships attributes, keys, cardinality and participation constraints.

Step three: Designing ER Conceptual Model

Assumptions made resulted in the relationships/attributes mentioned below:

1- A contract is signed for only one construction project. Therefore, the relation between ConstructionProject and Contract is one-to-one.
2- A ProjectManager can manage multiple ConstructionProject. Each Construction can only have one ProjectManager. Therefore, the relation is one-to-many.
3- Since Contractor is the one responsible of supervising SubContract, a relation between ProjectManager and SubContracts is not required. Instead, an assumption have been made that ProjectManager supervise Contactor. Since ProjectManager are able to do multiple jobs, an assumption have been made that a ProjectManager not necessary supervise a Contractor and so it is 0..N multiplicity.
4- Since ProjectManager are assigned when the client enters into a contract, a ProjectManager must liase with at least 1 client. Therefore, the relationship of liaise are one-to-many.
5- For relationship number 5 and 6 (supervise and workOn), an assumption have been made that not every Contractor works on a SubContract as some of them just supervise.
6- For relationship number 8, an assumption has been made that a Client can have multiple construction projects but every ConstructionProject have only one Client. If the Client is group of people, then one of them must represent the group.
7- For client, an assumption has been made that the projectDate attribute is formatted in a way that's unique even if there were multiple clients that has signed a contract in the same day. Therefore, projectDate is the primary key of Client entity. This is ideal because now projectDate can be set as a foreign key in ProjectManager entity.

The following are the entities and their attributes given by EC:

1- ContructionProject (Strong entity)
"each construction project has a unique address and the system records the number of rooms and floors the construction has."

- uniqueAddress {PK}
- numberOfRooms
- numberOfFloors

2- Contract (Weak entity dependant on ConstructionProject existance)
"For each construction project there is a contract which has a start date, end date and a total fee."
- totalFee
- startDate
- endDate

3- SubContract (Weak entity dependant on the existance of Contract)
"A contract consists of several sub contracts which outline the details of the task to be completed (eg Plastering), its fee and the outstanding balance (identifying how much of the fee is still unpaid)."
- taskDetails
- taskFee
- unpaidFee

"One contractor is responsible for the supervision of the sub contract task and must be identified when the sub contract is created."
- dateCreated

4- ProjectManager (Weak entity dependant on Client projectDate)
"project managers have a distinct employee ID and have a name."
- staffName
- employeeID

"They are assigned when the client enters into a contract."
- projectDate

5- Contractor (Weak entity dependant on subcontract and subtype of staff)
"one contractor is responsible for the supervision of the sub contract task and must be identified when the sub contract is created."
-staffName
- dateCreated (same as the date of SubContract)

"Contractors have an Australian Business Number (ABN) and a phone number."
- businessNumber
- phoneNumber

6- Client (Strong entity)
"Each client has a unique ID and has a name. The date that clients enter into a contract for a specific project is recorded – note that this may be before the start date of the contract itself. Furthermore, clients nominate a specific bank account which will be used for payment of all contracts that they enter into. The account has a BSB and an account number."
- projectDate {PK}
- uniqueID
- name
- BSB
- accountNumber

The following are the relationships between entities given by EC:

1- ConstructionProject 1..1 have 1..1 Contract
"For each construction project there is a contract which has a start date, end date and a total fee."

2- ProjectManager 1..1 manages 1..N ConstructionProject
"there are project managers that are responsible for overseeing the management of various construction projects"

3- ProjectManager 1..1 supervise 0..N Contractor
"such as organising sub contracts for completing various tasks in the contruction of the project, as well as liasing with the client."

4- ProjectManager 1..1 liaise 1..N Client
"such as organising sub contracts for completing various tasks in the contruction of the project, as well as liasing with the client."

5- Contractor 1..N supervise 1..N SubContract
"One contractor is responsible for the supervision of the sub contract task and must be identified when the sub contract is created."

6- Contractor 1..N workOn 1..N SubContract
"There are many contractors working on the sub contract task itself (the supervisor also works on the sub contract task)."

7- Contract 1..1 outline 1..N SubContract
"A contract consists of several sub contracts which outline the details of the task to be completed (eg Plastering), its fee and the outstanding balance (identifying how much of the fee is still unpaid)."

8- Client 1..1 have 1..N ConstructionProject

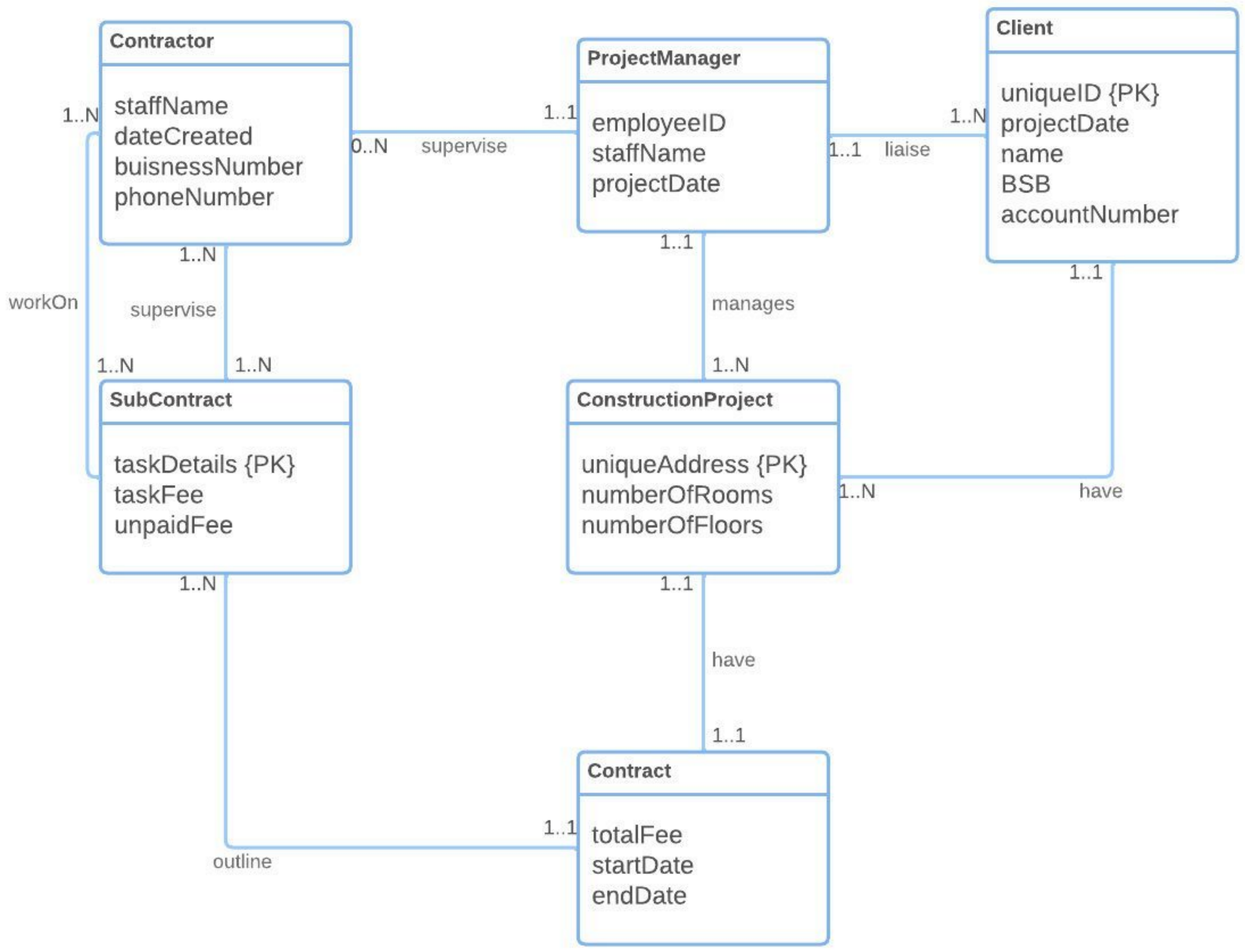


Figure 1: Entity-Relationship Model for EC

Task 2: Designing an Entity-Relationship Model

Part A: Initial Design

Same approach that's similar to task 1 are assumed to be taken where the database designer undergo the steps of the development life cycle of a database. I will jump to step three.

Assumptions made resulted in the relationships/attributes mentioned below:

1- For the relationship "prepare", I have assumed that the restaurant can always add a new item to their menu and that the restaurant must at least have one item in their menu to prepare. Therefore the multiplicity is 1..N on MenuItem side.

The following are the entities and their attributes given by Faster Food delivery company:

1- Customer (Strong entity)
"For each customer the database records their credit card number, its expiry date, their name, address and phone number."
- phone {PK}
- creditCard
- number
- expiryDate
- name
- address

2- Resturants (Strong entity)
"by restaurants, each with a unique Australian Business Number (ABN), and also have an address and food style (eg Thai)."
- ABN {PK}
- address
- foodStyle

3- MenuItem (Strong entity)
"Menu items have a name (eg Vegetarian Thai Green Curry), a description, its cost and preparation time"
- name {PK}

- description
- cost
- prepTime

4- Order (Strong entity)
"with each order, the order date, time, status and quantity (of the menu item) is recorded."
- time {PK}
- date
- status
- quantity

The following are the relationships between entities given by Faster Food delivery company:
1- Customers 1..1 make 1..N order
"Each order is made by a specific customer."
2- Resturant 1..N prepare 1..N MenuItem
"There could be many resturants that prepare the same menu item (but with potentially different descriptions, costs and preparation times)."
3- MenuItem 1..N have 1..N order
"A single menu item can be a part of many orders, with each order, the order date, time, status and quantity (of the menu item) is recorded."

1

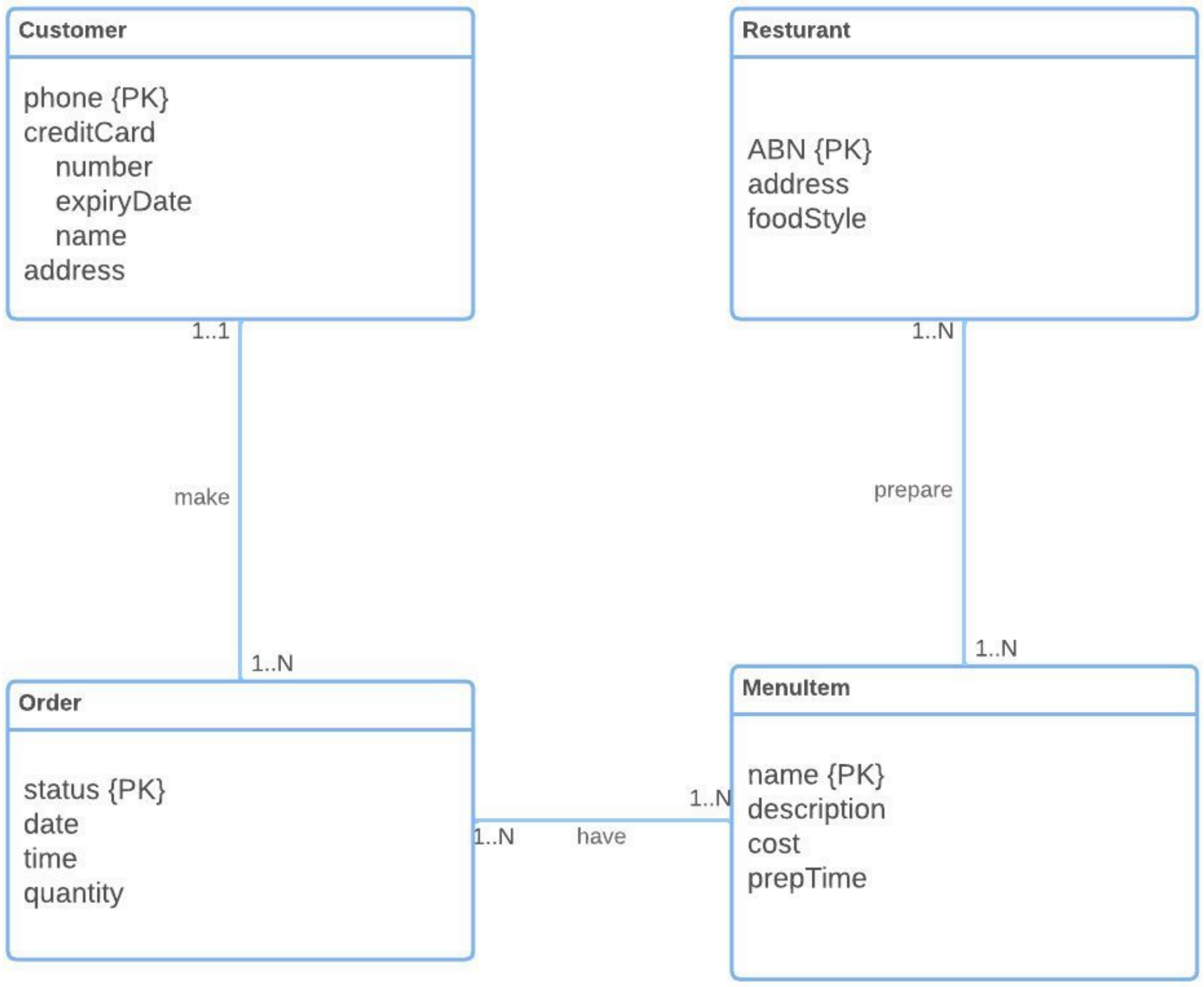


Figure 2: Entity-Relationship Model for Faster Food delivery company

Part B: Client Adjustments

Assumptions made resulted in the relationships/entities mentioned below:
1- I have assumed that the rating consist of a single comment and stars from 1 to 5.
2- To enforce that all MenuItems in an order is delivered from only one resturant, the multiplicity of resturant side of the relationship prepare must be changed from 1..1 to 1..N.

The following are the entities that has been added due to client requirement:
1- Delivery (Weak entity dependant on Order)
- uniqueID

- date
- time
- fee

2- Driver (Weak entity dependant on Delivery)
- TFN
- name
- vehicleRegist

3- Rating (Weak entity dependant on Customer)
- comment
- stars

The following are the relationships that has been added due to client requirement:

1- Customer 1..1 states 1..1 Rating
2- Driver 1..1 deliver 1..N Delivery
3- Delivery 1..1 have 1..N Order

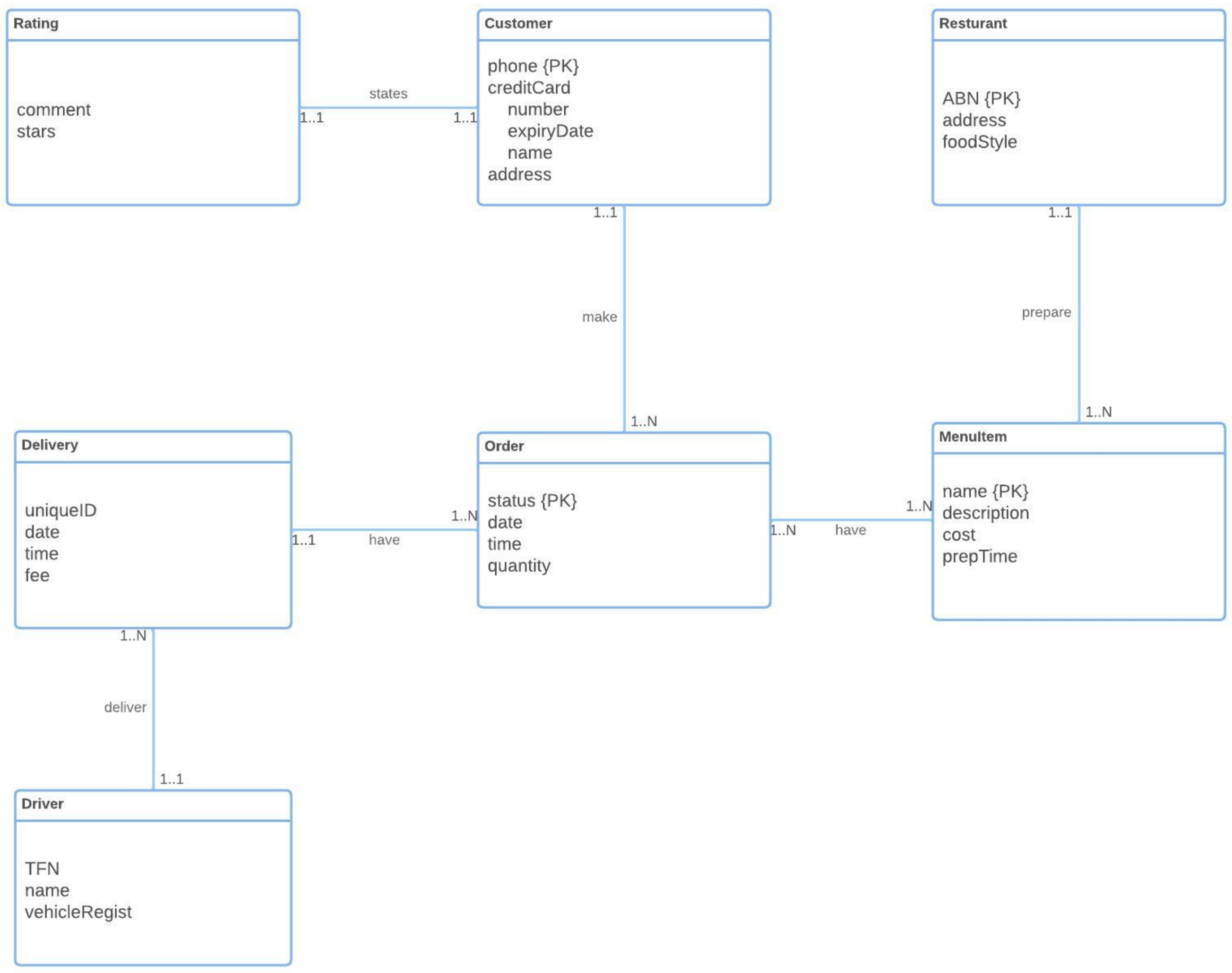


Figure 3: Entity-Relationship Model for Faster Food delivery company

Task 3: Mapping an ER Model to a Relational Database Schema

There are seven steps for the process of converting a conceptual model to a logical model (Relational Database Schema):

1- Deriving relations for logical data model
2- Validate relations using normalization
3- Validate relations against user transaction
4- Check Integrity Constraints
5- Review logical data model with user
6- Merge logical data models into global model
7- Check for future growth

The only step that is required for this Task is step one.

Step one: Deriving relations for logical data model

For each strong entity, there should be a relation that include all the simple attributes of the entity where the composite attributes must be broken down. All the entities shown in the conceptual model given for the aspects online auction system are strong entities. Therefore, there should be six relations:
1- Category (<u>title</u>, description)
2- Item (<u>itemID</u>, description)
3- Seller (<u>sellerEmail</u>, name, address)
4- Auction (<u>auctionNo</u>, reserve, openDate, closeDate, outcome)
5- Bid (<u>BidNo</u>, amount, bidDate)
6- Bidder (<u>bidderEmail</u>, name, address)

There are no weak entities in the given conceptual model.

The following are one-to-many binary relationship:
1- Seller 1..1 lists 0..N Item
2- Item 1..1 upFor 1..N Auction
3- Auction 1..1 receives 0..N Bid
4- Bidder 1..1 places 0..N Bid

For each one-to-many binary relationship, the entity on the “one side” of the relationship is designated as the parent entity and the entity on the “many side” is designated as the child entity. The primary key of the entity with the "one side"must be moved to the relation that has the "many side". Therefore, adjustions to the previous relations must be made:
1- Bid (<u>BidNo</u>, amount, bidDate, bidderEmail*, auctionNo*)
2- Auction (<u>auctionNo</u>, reserve, openDate, closeDate, outcome, itemID*)
3- Item (<u>itemID</u>, description,sellerEmail*)

There are no one-to-one relationships in the given conceptual model.

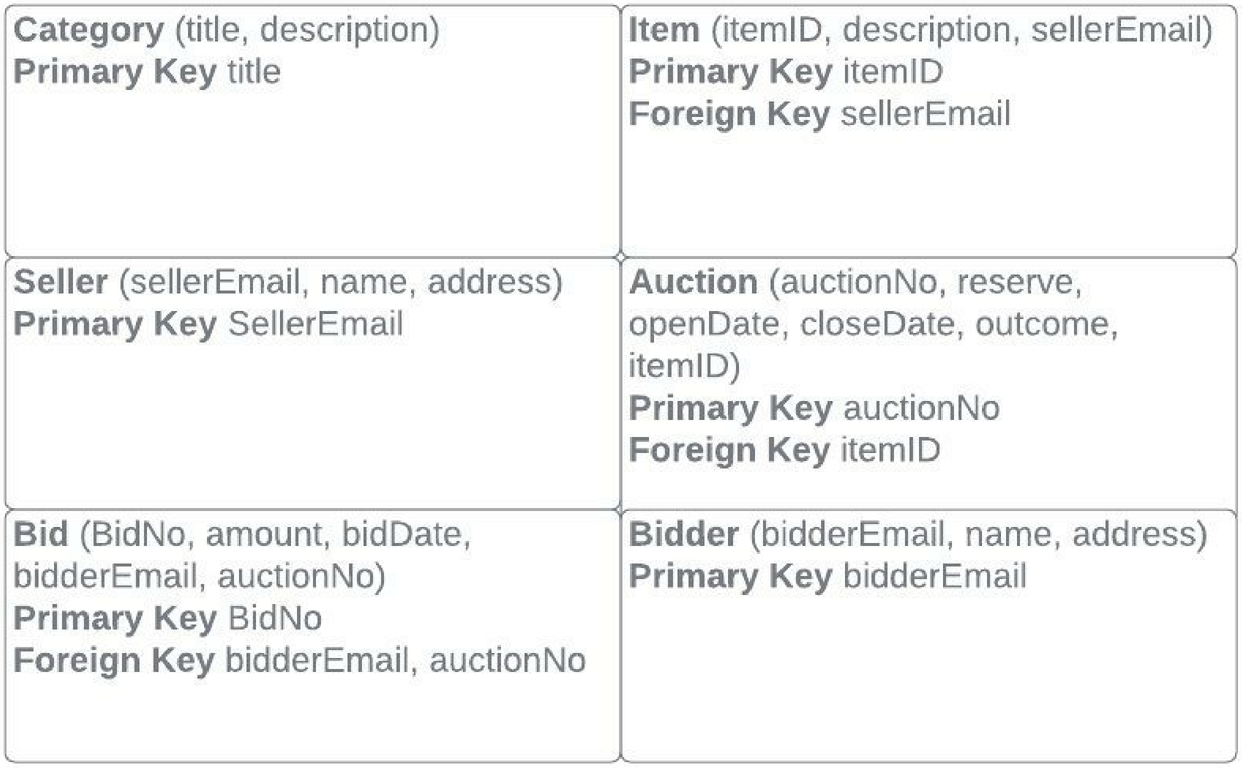
There are no superclass/subclass relationships in the given conceptual model.

The following are many-to-many binary relationship:
1- Category 0..N belongsTo 0..N Item

For each many-to-many binary relationship, there must be a relation that represent the relationship and include any attributes that are part of the relationship. We post the primary keys
1- belongTo (<u>title*</u>, <u>itemID*</u>)

There are no complex relationships in the given conceptual model. All of the relationships are binary.

There are no multi value attribute in the given conceptual model.



Category (title, description) **Primary Key** title	**Item** (itemID, description, sellerEmail) **Primary Key** itemID **Foreign Key** sellerEmail
Seller (sellerEmail, name, address) **Primary Key** SellerEmail	**Auction** (auctionNo, reserve, openDate, closeDate, outcome, itemID) **Primary Key** auctionNo **Foreign Key** itemID
Bid (BidNo, amount, bidDate, bidderEmail, auctionNo) **Primary Key** BidNo **Foreign Key** bidderEmail, auctionNo	**Bidder** (bidderEmail, name, address) **Primary Key** bidderEmail

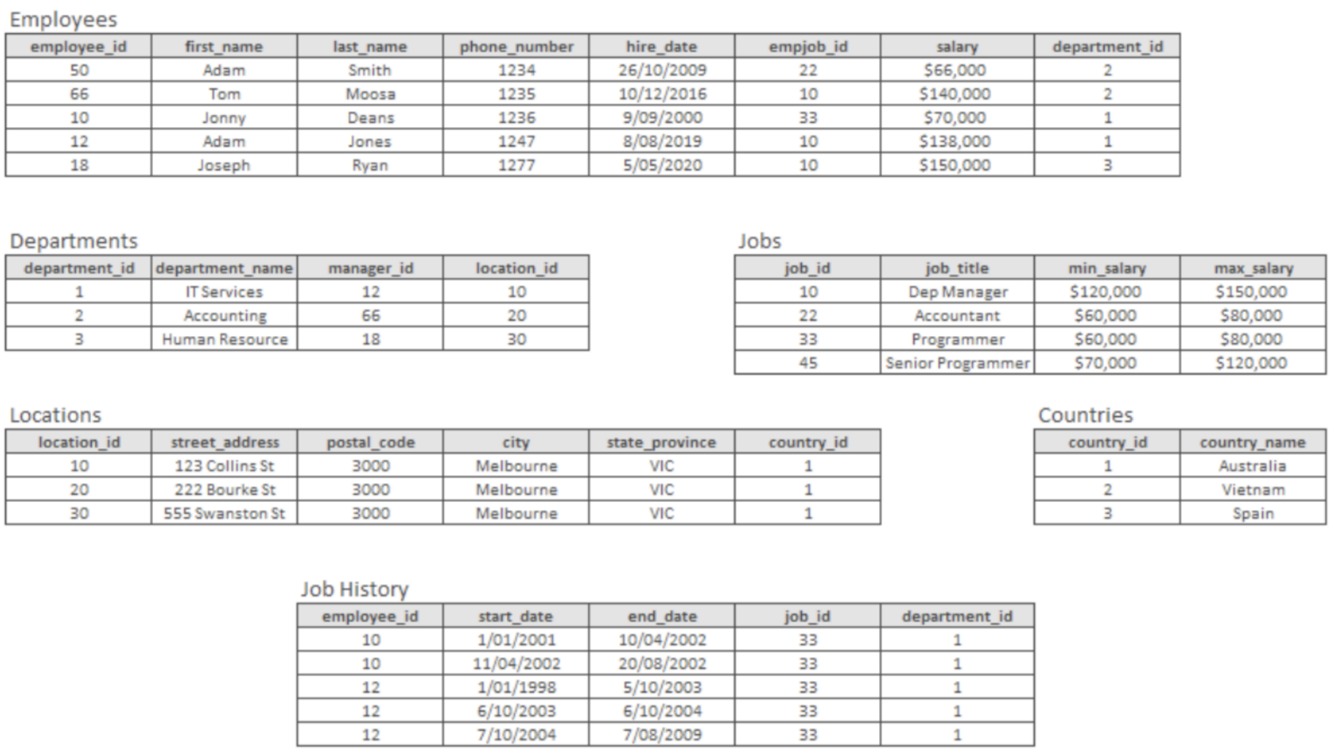
Figure 4: A summary table for all relations.

Part B: Relational Database Model

Requirements:

- Task 4: Answer short questions about relational database model

Task 4: Relational Database Model



Employees

employee_id	first_name	last_name	phone_number	hire_date	empjob_id	salary	department_id
50	Adam	Smith	1234	26/10/2009	22	$66,000	2
66	Tom	Moosa	1235	10/12/2016	10	$140,000	2
10	Jonny	Deans	1236	9/09/2000	33	$70,000	1
12	Adam	Jones	1247	8/08/2019	10	$138,000	1
18	Joseph	Ryan	1277	5/05/2020	10	$150,000	3

Departments

department_id	department_name	manager_id	location_id
1	IT Services	12	10
2	Accounting	66	20
3	Human Resource	18	30

Jobs

job_id	job_title	min_salary	max_salary
10	Dep Manager	$120,000	$150,000
22	Accountant	$60,000	$80,000
33	Programmer	$60,000	$80,000
45	Senior Programmer	$70,000	$120,000

Locations

location_id	street_address	postal_code	city	state_province	country_id
10	123 Collins St	3000	Melbourne	VIC	1
20	222 Bourke St	3000	Melbourne	VIC	1
30	555 Swanston St	3000	Melbourne	VIC	1

Countries

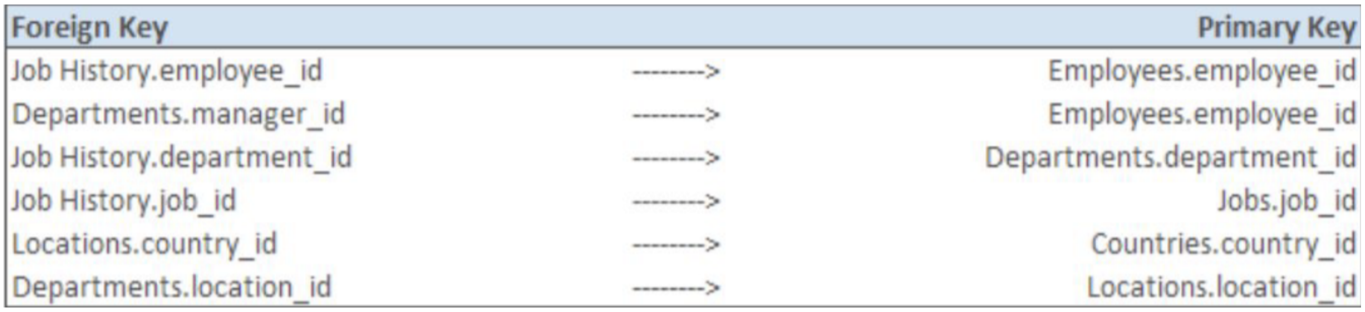
country_id	country_name
1	Australia
2	Vietnam
3	Spain

Job History

employee_id	start_date	end_date	job_id	department_id
10	1/01/2001	10/04/2002	33	1
10	11/04/2002	20/08/2002	33	1
12	1/01/1998	5/10/2003	33	1
12	6/10/2003	6/10/2004	33	1
12	7/10/2004	7/08/2009	33	1

Figure 5: Employee Database Instance.

The schema for Employee database are given as follows:
1- Employees (employee_id, first_name, last_name, phone_number, hire_date, empjob_id, salary, department_id)
2- Departments (department_id, department_name, manager_id*, location_id*)
3- Jobs (job_id, job_title, min_salary, max_salary)
4- Locations (location_id, street_address, postal_code, city, state_province, country_id*)
5- Countries (country_id, country_name)
6- JobHistory (employee_id*, start_date, end_date, job_id*, department_id*)



Foreign Key		Primary Key
Job History.employee_id	-------->	Employees.employee_id
Departments.manager_id	-------->	Employees.employee_id
Job History.department_id	-------->	Departments.department_id
Job History.job_id	-------->	Jobs.job_id
Locations.country_id	-------->	Countries.country_id
Departments.location_id	-------->	Locations.location_id

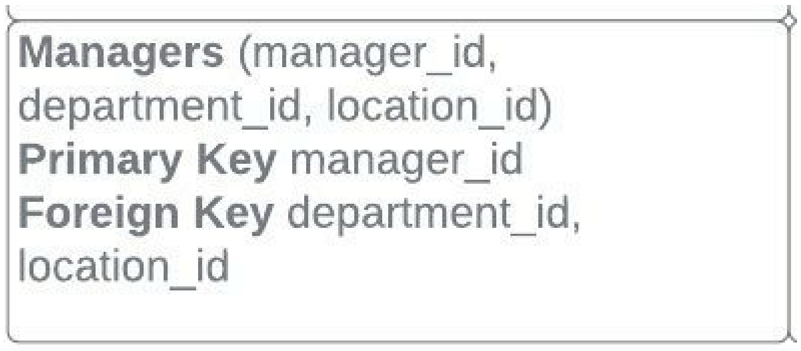
Figure 6: Foreign keys referential.

Question 4.1:

The primary key for the relation Employees is employee_id which means it is possible to uniquely identify a tuple in the relation Employees with only employee_id. It is possible to add another attribute employee_job and make employee_id, employee_job a composite primary key to ensure that each employee is associated with a job. Also, Job History.employee_id* foreign key given in figure 6 refer to Employees.employee_id but not every employee have a job history and that can be seen from the instance in figure 5 E.g. employee_id number 50 is not mentioned as a foreign key in Job History. Therefore, the database schema based on the given instance does not ensure that every employee is associated with a job.

Questions 4.2:

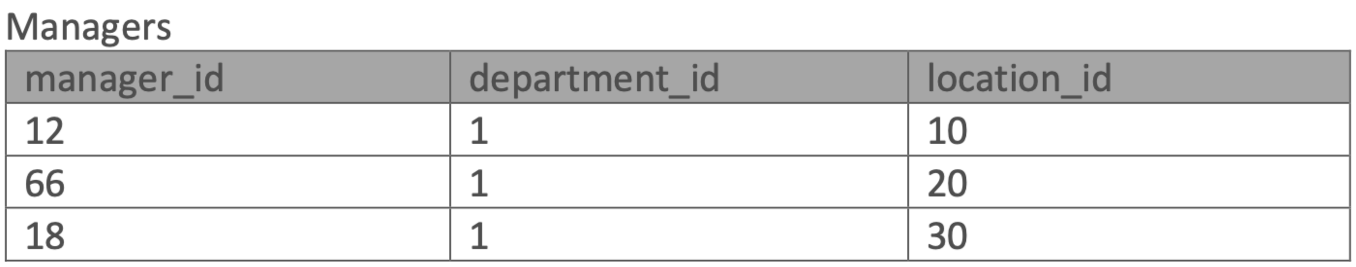
It is possible to create a new relation called Managers as an example shown below:



Managers (manager_id, department_id, location_id)
Primary Key manager_id
Foreign Key department_id, location_id

Figure 7: Managers relation example.

An instance for the relation where the same department have different managers and each manager with a different location is shown below:

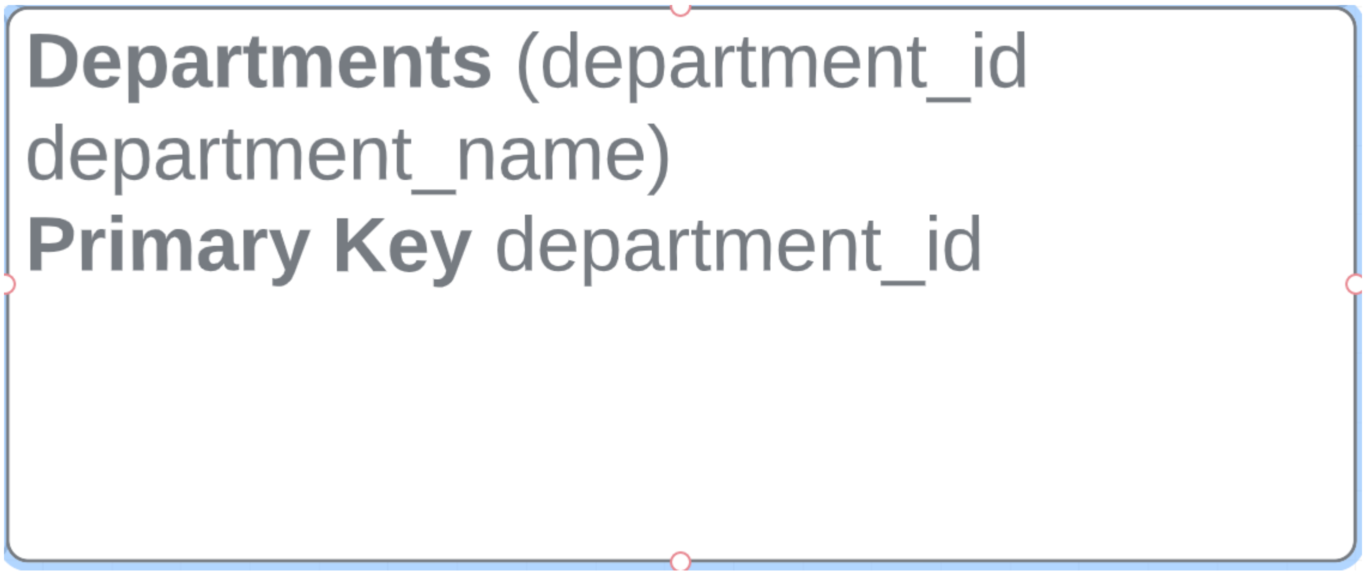


Managers

manager_id	department_id	location_id
12	1	10
66	1	20
18	1	30

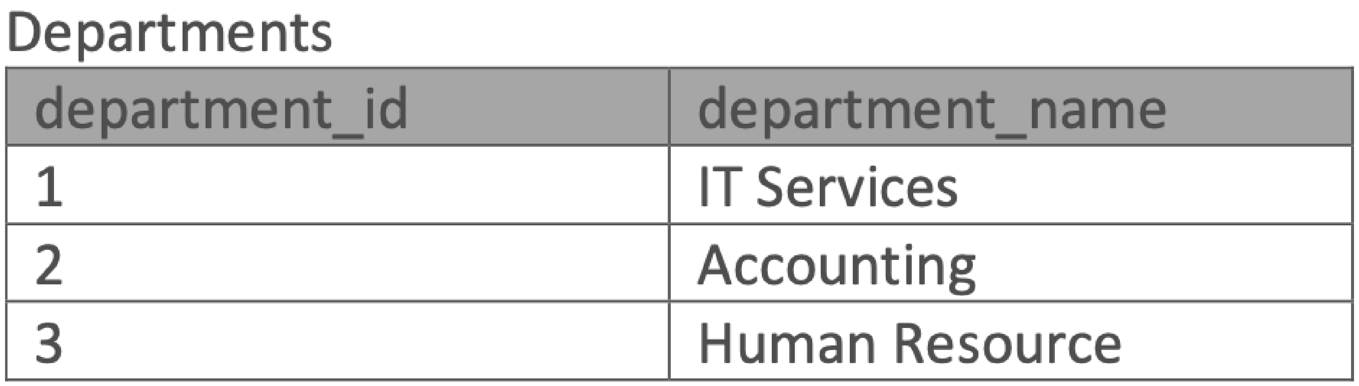
Figure 8: Managers instance.

And we can now remove the two attributes from Departments as shown below:



Departments (department_id department_name)
Primary Key department_id

Figure 9: Departments updated.



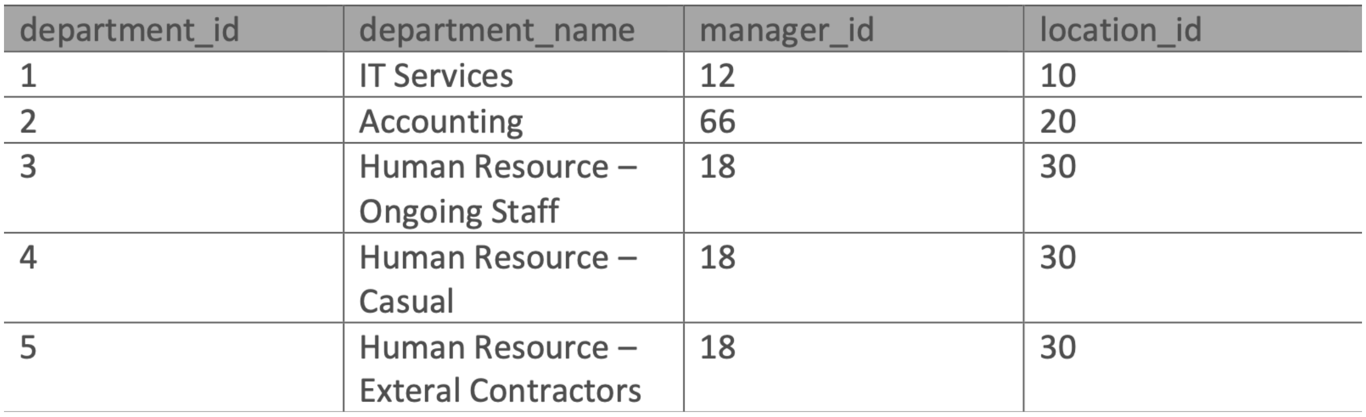
Departments

department_id	department_name
1	IT Services
2	Accounting
3	Human Resource

Figure 10: Instance of updated Departments.

Questions 4.3:

Assuming manager_id number 18 represents Joseph, then this works temporarily. Once new managers are hired for the three sub-departments, then this will not work and we must do a similar approach to Question 4.2 where manager_id is the primary key to ensure that each manager is uniquely identified with an id for each sub-department. The below figure represent what the SQL Query did to Departments relation:



department_id	department_name	manager_id	location_id
1	IT Services	12	10
2	Accounting	66	20
3	Human Resource – Ongoing Staff	18	30
4	Human Resource – Casual	18	30
5	Human Resource – Exteral Contractors	18	30

Figure 11: Department Entity after SQL Query.

Questions 4.4:

The SQL statement will reflect the change of Adam's new position. But there are two Adam?! In order to fix this, we simply need to add an 'AND' statement and specify last name in WHERE condition:

```
UPDATE Employees SET empjob_id=33 WHERE first_name='Adam' AND last_name='Smith';
```

In terms of the past contracts of Adam, the SQL statement/Query is not sufficient. The database schema is able to store past contracts in Job History relation. The following statements will add Adam Smith past contract in Job History:

```sql
INSERT INTO Job History VALUES(50,'26/10/2009',NULL,22,2);
```

I have assumed that end_date attribute in Job History relation can accept NULL.

Questions 4.5:

The statement will deleted the second row of the relation Location where location_id = 20.

Questions 4.6:

```sql
CREATE TABLE Employees
        (employee_id INTEGER NOT NULL,
        first_name VARCHAR(10),
        last_name VARCHAR(10),
        phone_number INTEGER,
        hire_date DATE,
        empjob_id INTEGER,
        salary INTEGER,
        department_id INTEGER,
        PRIMARY KEY(employee_id)
        )
        WITHOUT ROWID;
```

Questions 4.7:

```sql
CREATE TABLE Departments
        (department_id INTEGER NOT NULL,
        department_name VARCHAR(30),
        manager_id INTEGER,
        location_id INTEGER,
        PRIMARY KEY (department_id),
        FOREIGN KEY (manager_id) REFERENCES Employees(employee_id),
        FOREIGN KEY (location_id) REFERENCES Locations(location_id)
        )
        WITHOUT ROWID;
```

Questions 4.8:

```sql
INSERT INTO Departments VALUES(4,'Cyber Security',10,30);
INSERT INTO Job History VALUES(10,'9/09/2000','01/01/2010',33,1);
UPDATE Employees SET empjob_id=45 WHERE first_name='Jonny' AND last_name='Deans';
```

References

Thomas Connolly and Carolyn Begg (2014) _Database systems: A Practical Approach to Design, Implementation, and Management