School of Engineering and Computer Science

SWEN 304 **Database System Engineering**

# Assignment 1

The objective of this assignment is to test your understanding of database foundations, basic terms, and the relational data model the entity relational model. It is worth 15% of your final grade. The assignment is marked out of 100.

The assignment is due on **Friday, 26 March, 23:59 pm**. Please submit your assignment in **pdf** via the submission system.

## Question 1 [5 marks]

**a) [2 marks]** Give your own example of a relation schema with at least three attributes to illustrate what a relation schema is.

**Answer:**

Example: Employee (employee_number:Int , name:String, position:String)

Employee is the relation name and is defined by 3 attributes , employee_number,name and position.

**b) [3 marks]** Give a relation (represented as a table) with at least four tuples over your relation schema in part **a)**. Explain how your relation satisfies the properties of schema keys.

**Answer:**

| employee_number | name | position |
|---|---|---|
| 1 | Josh Gardner | General manager |
| 2 | Jessie McNama | Organiser |
| 3 | Patrick Dong | It support |
| 4 | Aimee Oshea | Marketing Officer |

The relation satisfies properties of schema keys and in the given relation, K= {employee_number}. The schema key should be unique,minimal and cannot be null and it identifies the name and position attributes. It is unique because two tuples could not have the

same employee ID, It is minimal because it has the smallest number(only 1) of unique attributes to identify a tuple. Also, it cannot be null as it is used to identify the individual tuples.

## Question 2 [15 marks]

Suppose you are the manager of an IT company in Wellington, and you are using a relational database to manage your business data. The following table shows an instance of the STAFF relation schema that stores basic data on the employees working in your company.

STAFF

| Name | DoB | EmpNo | JobTitle |
|---|---|---|---|
| Andy | 22/01/1988 | 88-11 | Developer |
| Mickey | 11/02/1996 | 96-02 | Project Manager |
| Jerry | 22/02/1989 | 89-08 | Developer |
| Andy | 15/05/1990 | 90-01 | Business Analyst |
| Alice | 22/02/1989 | 89-06 | Developer |
| Mary | 12/07/1990 | 90-04 | Architect |
| Mary | 25/11/1996 | 96-22 | Tester |

**a) [8 marks]** For every set of attributes (that is, for every subset of the set {Name, DoB, EmpNo, JobTitle}) decide whether you can deduce that it is *not* a candidate key, assuming the instance is legal. Justify your answer.

**Answer**:

{Name} and {Dob} and {JobTitle} can not be candidate keys by one attribute because the candidate key must be unique but they all have duplicate values in the database.

But a candidate key may have multiple attributes.

{Name} and {JobTitle} cannot be a candidate key because even combining them together would also avoid duplicates in our current database but it is very likely to have a staff with the same name and job title with a larger dataset.

**b) [4 marks]** For every remaining set of attributes (that is, every set not ruled out as a candidate key in part **a)**), discuss whether you consider it a suitable candidate key? Justify your answer.

**Answer**:

{EmpNo} can be a suitable candidate key because it contains unique values to justify other records stored. It is also a minimum key.

{Name} and {Dob} can be a candidate key , as if we combine them together can avoid duplicates. Staff who have the same name won't have the same DoB so {Name} and {Dob} can identify other records stored in the database. so{Name} and {DoB}can be a candidate key. This principle works the same with {Dob} and {JobTitle} so {DoB} and {JobTitle} can also be a candidate key.

**c) [2 marks]** Which of the candidate keys identified in part **b)** would you choose as the primary key?

**Answer**:

{EmpNo}

**d) [1 mark]** Add a tuple with your own data into the STAFF relation. How would you check that the primary key identified in part **c)** is still valid?

**Answer**:

t = {(Name,Andy),(DoB,25/11/1996),(EmpNo,87-03),(JobTitle,Developer)}

I will check to see if EmpNo is unique and check if it is not null, if these conditions are met then the primary key identified is still valid.

## Question 3 [10 marks]

Suppose your software company has developed a relational database for the grocery store "Fruits and more". The underlying database schema contains the following relation schemas:

- COMPANY (Cid: STRING, Name: STRING, Location: STRING) with primary key {Cid}
- FRUITS (Fid: STRING, Name: STRING, Cid: STRING, InStock: INTEGER, Price: INTEGER) with primary key {Fid, Cid} and foreign key Cid ⊆ COMPANY[Cid]

Below you find instances of these two relation schemas:

FRUITS

| Fid | Name | Cid | InStock | Price |
|---|---|---|---|---|
| 557 | Apple | 23XY | 50 | 21 |
| 85520 | Pear | A15F | 0 | 78 |
| 63311 | Pear | FVT35 | 211 | 49 |
| 36773 | Kiwi | 23XY | 50 | 21 |
| 36773 | Kiwi | FVT35 | 29 | 22 |

COMPANY

| Cid | Name | Location |
|---|---|---|
| 23XY | GreatFruits | Wellington |
| FVT35 | Yummy | Wellington |
| F15A | GreatFruits | Levin |
| A15F | BetterFruits | Lower Hutt |
| 5AB32 | NiceFruits | *null* |

Your tasks are as follows.

**a) [5 marks]** Decide which of the following tuples can be added or removed, respectively. *Justify your answers!*

1. Insert tuple ('XYZ4', 'Wellington', 'Yummy') into COMPANY

**Answer**:

Can be added , but the content within the tuple does not make sense. The 'Wellington' should be under Location attribute and the 'Yummy' should be under the 'Name' attribute.

2. Insert tuple (*null*, 'Tasty', 'Wellington') into COMPANY

**Answer**:

Can not be added, because cid is the primary key so it cannot be null value.

3. Insert tuple ('FVT35', 'SweetFruits', 'Porirua') into COMPANY

**Answer**:

Cannot be added , because cid is the primary key so the cid value should be unique and the insert tuple's cid is duplicated.

4. Delete tuple ('A15F', 'BetterFruits', *null*) from COMPANY

**Answer:**

Cannot be deleted directly , because the cid in COMPANY relation is also the forign key in the FRUITS relation.

5. Delete tuple (‘23XY’, ‘GreatFruits’, ‘Wellington’) from COMPANY

**Answer**:

Cannot be deleted directly , because the cid in COMPANY relation is also the forign key in the FRUITS relation.

**b) [5 marks]** Decide which of the following tuples can be added or removed, respectively. *Justify your answers!*

1. Insert tuple (‘55555’, *null*, ‘F15A’, 2, 99) into FRUITS

**Answer**:

Can be added because the primary key unique and forign key exists in the company relation.

2. Insert tuple (‘54556’, ‘Lemon’, ‘FV35’, 20, 43) into FRUITS

**Answer:**

Cannot be added because the forign key doesn’t exist in the company relation,

3. Insert tuple (‘53557’, ‘Apple’, ‘5AB32’, 500, 1) into FRUITS

**Answer**:

Can be added because the primary key unique and forign key exists in the company relation.

4. Delete tuple (‘36773’, ‘Kiwi’, ‘23XY’, 50, 21) from FRUITS

**Answer**:

Can be deleted as the primary key exists and the tuple is valid.

Delete tuple (‘46557’, ‘Apple’, ‘23XY’, 1, 21) from FRUITS

**Answer**:

Can be deleted because the primary key doesn’t exists.

## Question 4 [25 marks]

Suppose your software company is planning to build a relational database for a new event booking system. The following relation schemas are part of the underlying database schema.

- EVENT (eventId, venue)
- CLIENT (emailAddress, name, dob, phone) with primary key {emailAddress}
- MANAGER (managerId, name) with primary key {managerId}
- BOOKING (managerId, eventId, emailAddress, date, noOfTickets, promoCode) with primary key {eventId, emailAddress, date}

The following additional constraints are known:

1. Each client may only use a single emailAddress.
2. Managers may also be clients, but may not book an event for themselves.
3. For each event booking, the noOfTickets must be specified, while promoCode may be left blank (if not available).
4. An event may have up to four venues.

Your tasks are as follows:

**a)** **[3 marks]** For the relation schema EVENT, identify all suitable candidate keys. Explain your answer.

**Answer**:

The eventId is the only suitable candidate key as it is unique and cannot be null. venue is possible to have duplicates in the database and an event may have up to four venues so it cannot uniquely identify each attribute within the database.

**b)** **[5 marks]** For each of the relation schemas, identify all suitable foreign keys (if there are any). Explain your answer.

**Answer**:

Event: eventId is the primary key but no forign key.

Client:emailAddress is the primary key but no forign key.

Manager:managerID is the primary key but no forign key.

Booking:eventId is the primary key of the Event relation so eventID is a suitable forign key.

eventId is the primary key of the Event relation so eventID is a suitable forign key.

email Address is the primary key of the Client relation so emailAdress is a suitable forign key.

managertId is the primary key of the Manager relation so managerId is a suitable forign key.

**c)** **[3 marks]** Is it possible to add an event booking to the database with the emailAddress of a client who is not listed in the CLIENT relation? Explain your answer.

**Answer**:
No , it is not possible. Because the email Address in the Booking relation is a forign key that is connected to the Client relation. If we add an emailAddress not in the Client relation into the booking relation, it will cause data inconsistency. So in order to prevent that , it won't be possible to do.

**d)** **[4 marks]** If the attribute eventId would not be part of the primary key of BOOKING, what would be the consequence? Explain your answer.

**Answer**:
The consequence would be that the booking cannot justify which event is in the Booking relation because just using mangerID and email Address would not be enough to justify it. Then the eventID within the Booking relation can be duplicated and can be empty, it will cause a lack of information problem within the relation.

**e)** **[5 marks]** Suppose, a client ('paula@vuw.ac.nz', 'Paula', 22/01/2000, '381-1230') in the CLIENT relation has made several bookings stored in the relation BOOKING. When deleting the record of this client from the CLIENT relation, all her bookings should be deleted, too. How would you ensure this requirement? Explain your answer.

**Answer**:

Because Client contains the forign key to another relation so deleting it directly violates a referential integrity constraint. So we apply cascade which is deleting tuples affected in the Booking relation that refers to Client relation first. Then we delete the Client tuple. Now the client and booking the client made are both deleted , the requirement satisfied.

**f)** **[5 marks]** Suppose, a person (555, 'Mary') in the MANAGER relation is no longer working for the event booking system. When deleting the record of this manager from the MANAGER relation, all the bookings she made for clients should not be lost. How would you ensure this requirement? Explain your answer.

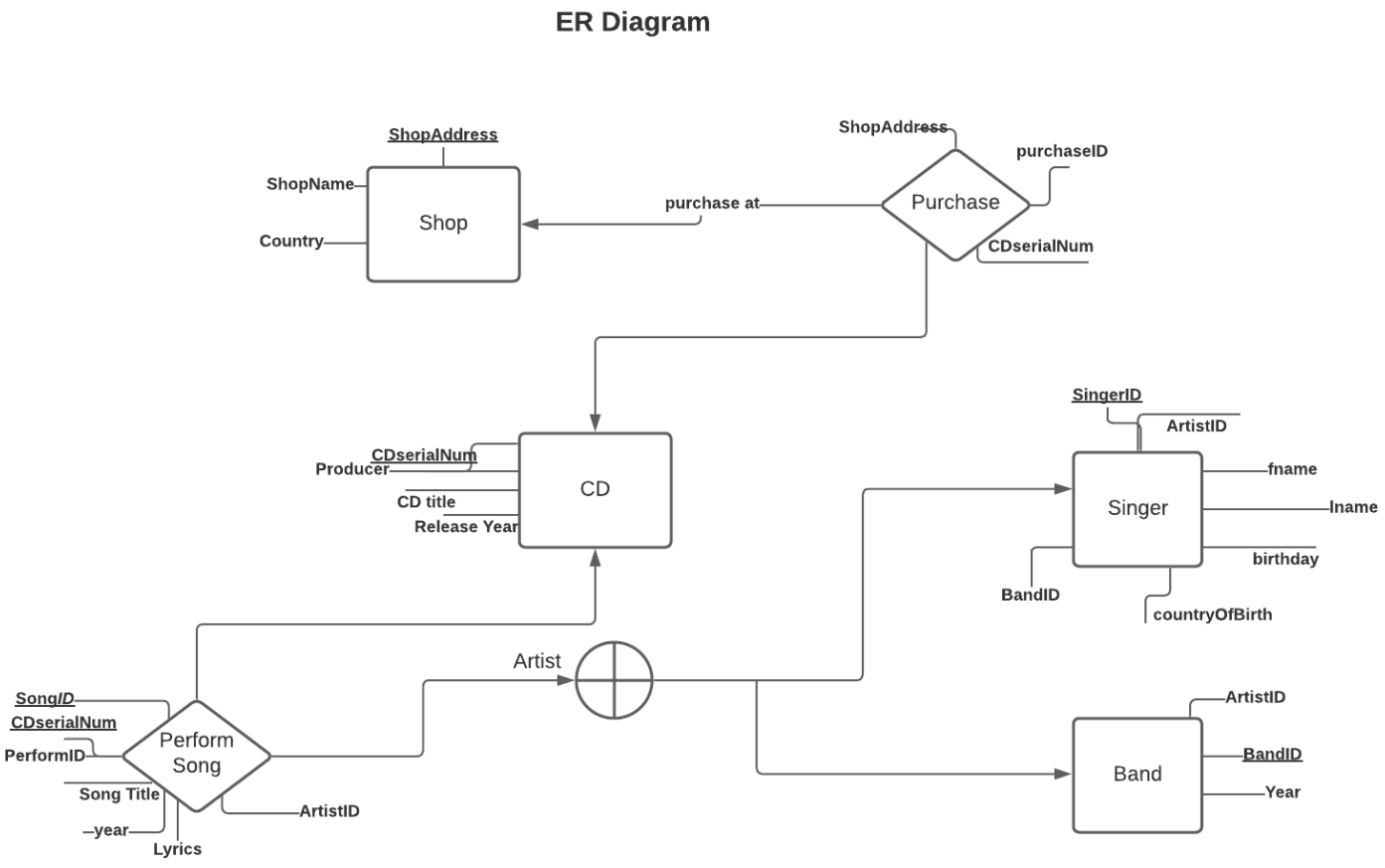
**Answer**:

Manager relation contains the forign key to another relation so deleting it directly violates a referential integrity constraint. So we apply Set null / set default which insert null or a default value in the mangerID attributes of tuples in Booking relation that refer to Manager manager.Then the booking will not be deleted. requirement satisfied.

## Question 5 [30 marks]

You are asked to design a CD_COLLECTION database for your grandma's CD collection. A CD has a title, a release year, a unique serial number and was produced by a certain producer. A CD contains performances of songs by an artist. There are at most 20 songs on a CD. A song has a title, the year in which it was written, and the song lyrics. An artist is a singer or a band. A singer has a name (first name and last name), a birthday and a country of birth. A band has a band name and the year in which it was founded. A singer can be a member of a band. Furthermore, your grandma buys CDs at certain shops, which have a name, an address, and a country.

**a) [24 marks]** Draw an extended ER diagram for the database above. Write down the corresponding extended ER schema, including declarations of all the entity types (showing attributes and keys) and relationship types (showing components, attributes and keys).




**ER Schema:**

**Entity types:**

Shop:

attributes:{ShopName,ShopAddress,Country}

key:{ShopAdress}

CD:

attribute:{CDTitle, ReleaseYear,CDserialNum,Producer}

key:{CDserialNum}

**Cluster:**

Artist = Singer⊕Band

Singer:

attribute:{ArtistID,SingerID,fname,lname,birthday,countryOfBirth,BandID}

key:{SingerID,ArtistID,BandID}

Band:

attribute:{ArtistID,BandID,BandName,Year}

key:{ArtistID,BandID}

**Relationship types:**

Purchase:

components:{Shop,CD}

attribute:{PurchaseID,ShopAddress,CDserialNum}

keys:{PurchaseID,ShopAdress,CDserialNum}

Perform Song:

component:{Artist,Song,}

attribute:{ArtistID,nameSongTitle,Year,lyrics,CDserialNum,SongID}

key:{ArtistID,CDserialNum,SongID}

**b) [6 marks]** There may be information, requirements or integrity constraints that you are not able to represent in your diagram. Give three examples of integrity constraints that have not been represented in your diagram.

*Remark*: Whenever you feel that information is missing in the problem description above, add an assumption and make your assumption explicit. In practice you would consult the domain experts or potential users for clarification.

**Answer**:

1.The first requirement that I couldn't represent is that there should not be more than 20 songs within a CD. As the entity relationship diagram is a data model and it only presents the

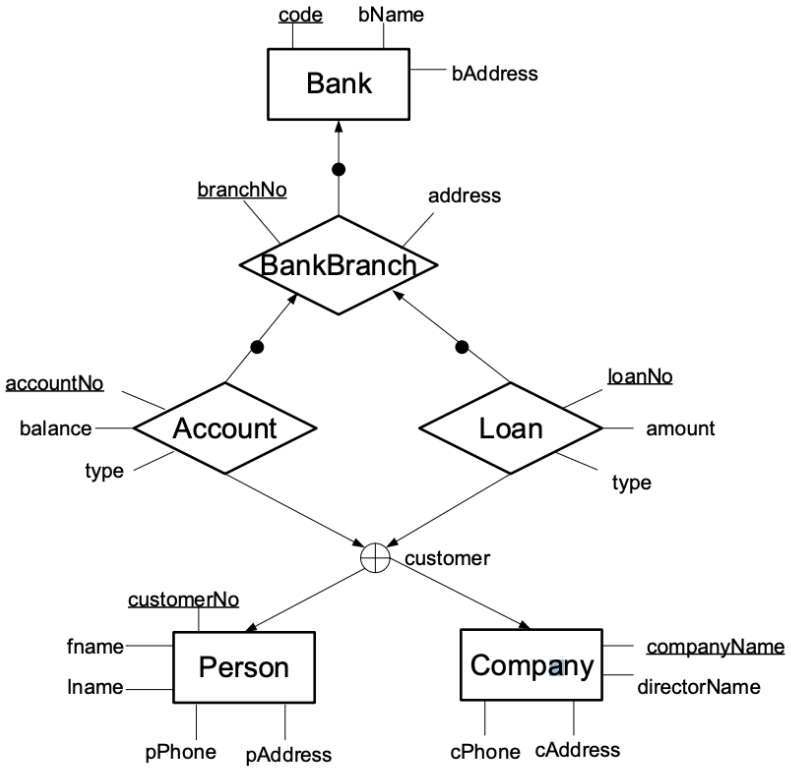
consisting of entities and relationships so I could not show the expectation of data size with ER diagram.

2.My diagram does not present the requirement of should show the band may contain singers. I did add a BandID attribute forign key in the singer entity to show that a singer may belong to a band.I cannot directly draw it on the entity relationship diagram because not all singers belong to a band. So this relationship cannot be presented in the ER diagram.

3.Also my diagram doesn't show integrity constraints of the singer and band well. Because the Song entity should store the artist information but the artist is actually a cluster.The standard of the singer and the band each have their own primary key which may cause duplication.  How I solved the problem is to create an artist entity to unify the primary key standard but it is still a good solution.

## Question 6 [15 marks]

Consider the extended ER diagram below.




**a) [5 marks]** Present the extended ER schema of the extended ER diagram above.

Level 0 :

Bank: attributes{code,bName,bAddress} key{code}

Person:attributes{customerNo,fname,lname,pPhone,pAddress}key{customerNo}

Company:attributes{companyName,directorName,cPhone,cAddress}key{companyName}

Level 1:

BankBranch: component{Bank}attributes{branchNo,address} key{branchNo}

Level2:

Account:

component{BankBranch,customer},attributes{accountNo,balance,type},key{accountNo}

Loan: component{BankBranch,customer},attributes{loanNo,amount,type}

key{loanNo},customer=Person⊕Company

**b) [10 marks]** Transform your extended ER schema into a relational database schema. In particular, list all the relation schemas in your relational database schema. For each relation schema, list all attributes, the primary key, the NOT NULL constraints, and the foreign keys.

**Answer:**

**Bank**={code **NOT NULL**, bName, bAddress}

primary key :{code}

**Person**={customerNo,fname,lname,pPhone,pAddress}
primary key{customerNo}

**Company**={companyName,directorName,cPhone,cAddress}
primary key{companyName}

**BankBranch**={branchNo **NOT NULL**,BankCode **NOT NULL**,address,accountNo,loanNo}
primary key:{branchNo,BankCode}
forign key:[BankCode]⊆ Bank[code]

**Person-Account** ={accountNo **NOT NULL**,BankBranch-branchNo **NOT NULL**,BankBranch_BankCode **NOT NULL**,Person-customerNo **NOT NULL**,balance,type}
primary key:{accountNo,BankBranch-branchNo,BankBranch_BankCode BankCode,Person-customerNo}
forign key:[BankBranch-branchNo]⊆ BankBranch[branchNo]
[BankBranch_BankCode BankCode]⊆ BankBranch[BankCode]
[Person-customerNo]⊆Person[customerNo]

**Person-Loan** ={loanNo **NOT NULL**,BankBranch-branchNo **NOT NULL**,BankBranch_BankCode BankCode **NOT NULL**,Person-customerNo **NOT NULL**,amount,type}
primary key:{loanNo,BankBranch-branchNo,BankBranch_BankCode BankCode,Person-customerNo}
forign key:[BankBranch-branchNo]⊆ BankBranch[branchNo]
[BankBranch_BankCode BankCode]⊆ BankBranch[BankCode]
[Person-customerNo]⊆Person[customerNo]

**Company-Account** ={accountNo **NOT NULL**,BankBranch-branchNo **NOT NULL**,BankBranch_BankCode **NOT NULL**,Company-companyName **NOT NULL**,balance,type}
primary key:{accountNo,BankBranch-branchNo,BankBranch_BankCode BankCode,Company-companyName}
forign key:[BankBranch-branchNo]⊆ BankBranch[branchNo]
[BankBranch_BankCode BankCode]⊆ BankBranch[BankCode]
[Company-companyName]⊆Company[companyName]

**Person-Loan** ={loanNo **NOT NULL**,BankBranch-branchNo **NOT NULL**,BankBranch_BankCode BankCode **NOT NULL**,Company-companyName **NOT NULL**,amount,type}
primary key:{loanNo,BankBranch-branchNo,BankBranch_BankCode BankCode,Company-companyName}
forign key:[BankBranch-branchNo]⊆ BankBranch[branchNo]
[BankBranch_BankCode BankCode]⊆ BankBranch[BankCode]
[Company-companyName]⊆Company[companyName]

**********