Chapter 5 Questions and Exercises: 5.1, (also define the term Super key, Key, Candidate Key, Primary Key).5.3, 5.4, 5.6, 5.9, 5.11(b, c, e,f, h, i, k), 5.12, 5.14, 5.17. Chapter 7 Exercise: (1) Map the ER diagram in Figure 3.21 on PP 95 into relational schemas. Justify your choice of mapping options. (2) Map the EER diagram in Figure 4.9 on page 120 into relational schemas. Justify your choice of mapping options.

5.1

- Domain – a set of atomic values. Usually the values a data type can take.
- Attribute – the name of a role played by some domain in the relation schema.
- N-tuple – an ordered list of n values where each element belongs to an element of its corresponding attribute domain or is NULL.
- Relation schema – a relation name R and a list of attributes and a set of integrity constraints.
- Relation state – only the valid tuples that represent a particular state of the real world. A set of n-tuples.
- Degree of a relation – aka arity, is the number of attributes in the relation schema.
- Relational database schema – the union of all relation schemas for all relation schemas in the database and a set of integrity constraints.
- Relational database state – the union of all relation states for all relations in the database.
- Super key – specifies a uniqueness constraint that no two distinct tuples in any state r of R can have the same value for SK.
- Candidate key – each of the possible keys. A relation may have more than one key, and each one is known as a candidate key.
- Primary key – the candidate key chosen to uniquely identify tuples in the relation.

5.3

- Duplicate tuples aren't allowed in a relation because a relation is defined as a set of tuples. By definition all elements of a set are distinct, and thus, all tuples in a relation must also be distinct.

5.4

- The difference between key and super key is that a key is a super key in which you cannot remove anymore attributes lest it no longer be a key. A key is a minimal super key.

5.6

- Characteristics of relations that make them different from ordinary tables and files
    - A relation is subject to more constraints than a table.
    - Ordering of tuples in a relation doesn't matter. You can reorder the tuples and the two relations are equivalent. The same isn't true for files.
    - A relation must have unique tuples, whereas a table or file can have duplicate entries.
    - A table is n-dimensional, where a relation is 2-dimensional.
    - Attributes are independent in a table, they aren't in a relation.
    - A table is a physical instance of a relation, but a relation is a logical construct.

5.9

- Foreign key – a set of attributes is a foreign key in R1 that references relation R2 if
  - The attributes in FK have the same domain as the primary key attributes in R2
  - A value of an attribute in FK in R1 exists in the corresponding attribute in R2 or is NULL
  - This is used to reference one relation's key from another relation to enforce referential integrity.

5.11

- B
  - Referential integrity constraint of Dnum. No tuples with DEPARTMENT.Dnumber =2 .
  - Fix by not inserting the tuple and asking the user to correct the error.
- C
  - Uniqueness constraint of Dnumber. Tuple with DEPARTMENT.Dnumber=4 already exists.
  - Referential integrity constraint of Mgr_ssn. No tuples with EMPLOYEE.Ssn='943775543'.
  - Fix by not inserting the tuple and asking the user to correct the error.
- E
  - Violates the domain of attribute DEPENDENT.Relationship. 'spouse' is not equivalent to 'Spouse' (with a capital S).
  - Fix by capitalizing the S.
- F
  - No violations.
- H
  - Referential integrity constraint of Pnumber. WORKS_ON.Pno references this and two tuples will have invalid values for WORKS_ON.Pno.
  - Fix by either rejecting, setting WORKS_ON.Pno=1 to WORKS_ON.Pno=NULL, or by cascading the delete to delete the two tuples in WORKS_ON.Pno=1.
- I
  - No violations.
- K
  - No violations.

5.12

- A – The operations for this are insert operation(s)
  - INSERT <Flight_number, Leg_number, Date, Seat_number, Customer_name, Customer_phone> into SEAT_RESERVATION with the given information. The customer will supply their name and phone number. Flight_number, Leg_number, Date, and Seat_number will all be shown to the user as a list of choices and they will choose a particular one.

- B – I would expect to check that Flight_number exists in FLIGHT, Leg_number exists in FLIGHT_LEG and LEG_INSTANCE, Date exists in LEG_INSTANCE, Seat_number doesn't exceed Max_seats in AIRPLANE_TYPE, that Number_of_available_seats in LEG_INSTANCE isn't 0, and that the Customer_name and Customer_phone aren't NULL. Make sure that the combination of the 4 attributes for the key type aren't already in SEAT_RESERVATION and that none of these attributes are NULL.
- C
    - Key constraints – Ensuring that {Flight_number, Leg_number, Date, Seat_number} is unique.
    - Entity integrity constraints – Ensuring that none of {Flight_number, Leg_number, Date, Seat_number} are NULL.
    - Referential integrity constraints – Ensuring that Flight_number exists in FLIGHT, Leg_number exists in FLIGHT_LEG and LEG_INSTANCE.
    - Other constraints – Ensuring that Customer_name and Customer_phone are not NULL. Checking to make sure the Number_of_available_seats > 0 and that Seat_number < Max_seats.
- D
    - FLIGHT_LEG.Flight_number references FLIGHT.Flight_number.
    - FLIGHT_LEG.Departure_airport_code references AIRPORT.Airport_code.
    - FLIGHT_LEG.Arrival_airport_code references AIRPORT.Airport_code.
    - LEG_INSTANCE.Flight_number references FLIGHT_LEG.Flight_number.
    - LEG_INSTANCE.Airplane_id references AIRPLANE.Airplane_id.
    - LEG_INSTANCE.Departure_airport_code references FLIGHT_LEG.Airport_code.
    - LEG_INSTANCE.Arrival_airport_code references FLIGHT_LEG.Airport_code.
    - FARE.Flight_number references FLIGHT.Flight_number.
    - CAN_LAND.Airplane_type_name references AIRPLANE_TYPE.Airplane_type_name.
    - CAN_LAND.Airport_code references AIRPORT.Airport_code.
    - AIRPLANE.Airplane_type references AIRPLANE_TYPE.Airplane_type_name.
    - AIRPLANE.Total_number_of_seats references AIRPLANE_TYPE.Max_seats (maybe).
    - SEAT_RESERVATION.Flight_number references FLIGHT.Flight_number.
    - SEAT_RESERVATION.Leg_number references LEG_INSTANCE.Leg_number.
    - SEAT_RESERVATION.Date references FLIGHT.Weekdays and FLIGHT_LEG.Scheduled_departure_time.

5.14

- Referential integrity constraints
    - ORDER.Cust# references CUSTOMER.Cust#.
    - ORDER_ITEM.Order# references ORDER.Order#.
    - ORDER_ITEM.item# references ITEM.Item#.
    - SHIPMENT.Order# references ORDER.Order#.
    - SHIPMENT.Warehouse# references WAREHOUSE.Warehouse#.

- Other constraints
    - In addition to the referential integrity constraints listed, since the foreign keys are used as part of the primary key in some relations, we have some additional NOT-NULL constraints.
    - ORDER_ITEM.Order# may not be NULL.
    - ORDER_ITEM.Item# may not be NULL.
    - SHIPMENT.Order# may not be NULL.
    - SHIPMENT.Warehouse# may not be NULL.

5.17

- Referential integrity constraints
    - OPTION.Serial_no references CAR.Serial_no.
    - SALE.Salesperson_id references SALESPERSON.Salesperson_id.
    - SALE.Serial_no references SALESPERSON.Salesperson_id.
    - Since bartering usually happens, a SALE.Sale_price does not necessarily equal a CAR.Price. Usually the sale price is less than the advertised price.
- Some example populations

CAR

| | | | |
|---|---|---|---|
| 123456789 | Sonata | Hyundai | 12500.00 |
| 987654321 | Cobalt | Chevrolet | 7500.00 |
| 101010101 | Windstar | Ford | 19500.00 |

OPTION

| | | |
|---|---|---|
| 123456789 | power_locks | 250.00 |
| 987654321 | spoiler | 200.00 |

SALE

| | | | |
|---|---|---|---|
| 1 | 123456789 | 08-15-2008 | 11500.00 |
| 1 | 987654321 | 07-25-2010 | 6750.00 |

SALESPERSON

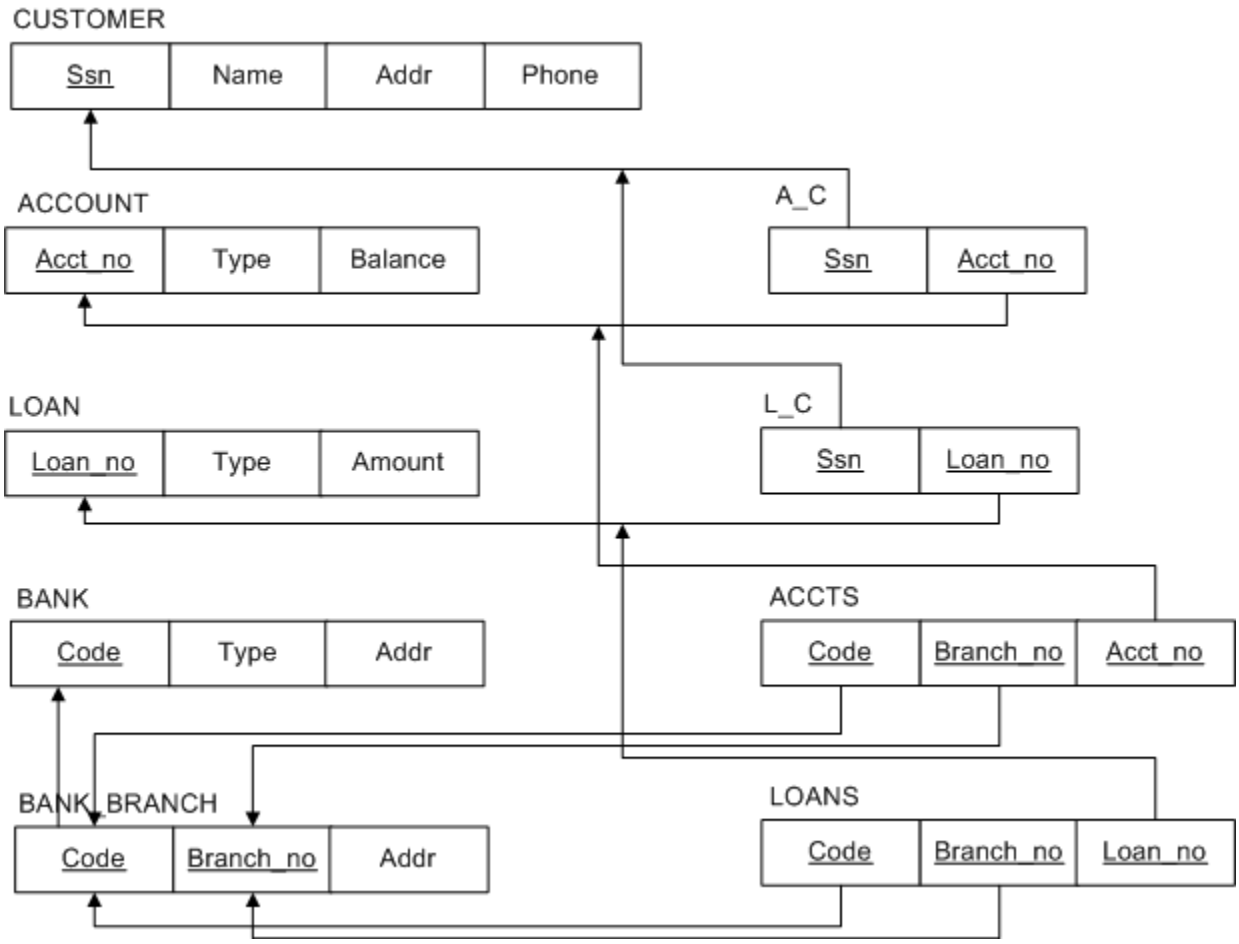
| | | |
|---|---|---|
| 1 | Gary | 888-555-9999 |
| 2 | Rebecca | 888-666-9999 |

- SALE violation: INSERT <3, 111111111, NULL, NULL>.
- SALESPERSON violation: INSERT<1, Jeremy, NULL>.

- SALE no violation: INSERT<1, 101010101, 10-15-2012, 18750.00>.
- SALESPERSON no violation: INSERT<3, Katy, 888-777-9999>.

7 (1) The following is my derived database schema for the diagram. Some of the arrow heads don't go all the way to the referenced attribute, and instead merge with another arrow that was already pointing to that attribute. The referenced attribute should still be clear.

- Justifications: I followed the procedures set forth in class to make BANK_BRANCH its own entity, whose partial key combined with the identifying entity's key creates a key for it. ACCTS and LOANS are at a specific BANK_BRANCH rather than with a specific BANK.



7(2) The following is my derived database schema for the diagram. I typed it instead of drawing it because it is remarkably cleaner looking since Visio isn't remarkably flexible with its arrowheads.

- I separate PERSON, FACULTY, and STUDENT into three entities for minimal data duplication.
- INSTRUCTOR_RESEARCHER must be its own entity type because it has an M:N relationship.
- SECTION and CURRENT_SECTION are rolled into one because CURRENT_SECTION has only two specific attributes.
- MAJOR and MINOR get rolled into STUDENT because they are 1:N.
- ADVISOR gets rolled into GRAD_STUDENT because it is 1:N.
- TEACH gets rolled into SECTION because it is 1:N.
- CS gets rolled into SECTION because it is 1:N.
- DC gets rolled into COURSE because it is 1:N.
- CD gets rolled into DEPARTMENT because it is 1:N.
- CHAIRS gets rolled into FACULTY because it is 1:1 and I chose that side.
- PI gets rolled into GRANT because it is 1:N.

These are the entities in my schema:

PERSON(Ssn, Fname, Minit, Lname, Bdate, Sex, No, Street, Apt_no, City, State, Zip)
FACULTY(Ssn, Rank, Foffice, Fphone, Salary, Chairs)
STUDENT(Ssn, Class, Minor, Major)
GRAD_STUDENT(Ssn, College, Degree, Year, Advisor)
GRANT(No, Title, Agency, St_date, Faculty)
INSTRUCTOR_RESEARCHER(Ssn)
DEPARTMENT(Dname, Dphone, Office, College)
COLLEGE(Cname, Dean, Coffice)
COURSE(C#, Cname, Cdesc, Department)
SECTION(Sec#, Year, Qtr, Instructor, Course)
REGISTERED(Student, Section)
TRANSCRIPT(Student, Section, Grade)
COMMITTEE(Faculty, Grad_student)
SUPPORT(Instructor, Faculty, Start, End, Time)
BELONGS(Faculty, Department)

These are the referential integrity constraints:

FACULTY.Ssn references PERSON.Ssn
FACULTY.Chairs references DEPARTMENT.Dname
STUDENT.Ssn references PERSON.Ssn
STUDENT.Minor references DEPARTMENT.Dname
STUDENT.Major references DEPARTMENT.Dname
GRAD_STUDENT.Ssn references PERSON.Ssn
GRAD_STUDENT.College references COLLEGE.Cname

GRAD_STUDENT.Advisor references FACULTY.Ssn
GRANT.Faculty references FACULTY.Ssn
INSTRUCTOR_RESEARCHER references FACULTY.Ssn or GRAD_STUDENT.Ssn
DEPARTMENT.College references COLLEGE.Cname
COURSE.Department references DEPARTMENT.Dname
SECTION.Instructor references INSTRUCTOR_RESEARCHER.Ssn
SECTION.Course references COURSE.C#
REGISTERED.Student references STUDENT.Ssn
REGISTERED.Section references SECTION.Sec#
TRANSCRIPT.Student references STUDENT.Ssn
TRANSCRIPT.Section references SECTION.Sec#
COMMITTEE.Faculty references FACULTY.Ssn
COMMITTEE.Grad_student references GRAD_STUDENT.Ssn
SUPPORT.Faculty references FACULTY.Ssn
SUPPORT.Instructor references INSTRUCTOR_RESEARCHER.Ssn
BELONGS.Faculty references FACULTY.Ssn
BELONGS.Department references DEPARTMENT.Dname