BLOCK 2: PRACTICAL ON RELATIONAL DATABASE MANAGEMENT SYSTEMS

Structure

1.0 BLOCK INTRODUCTION

In the previous block our emphasis was to present the concepts relating to RDBMS design. This block is an attempt to provide you with some practical exercises. This block is going to different from the other blocks, as it has not been divided into units; however, we have tried to keep separate exercises for separate topics. Please note that an exercise for an advanced topic will involve use of all the application skills acquired by you in the earlier exercises, therefore, you must proceed forward in the block only if you have completed the previous exercises, for your own benefits.

This block covers wide range of solved and unsolved exercises on the basic concepts, normalisation, ER diagram, Database implementation using SQL and a RDBMS package. You must try out these exercises using any RDBMS package. However, for this course because of wide availability the practical may be tried out at least on MS-ACCESS platform: The next block has been kept for the package related activities on MS-ACCESS, so you must use it while implementing these exercises.

We have tried to give you basic competence, however, you are requested to attempt more problems from the reference books:

1.3 C.J. Date, *Database Concepts*, Addison Wesley Publications.

1.4 Korth A. etal, *Database Concepts*, McGraw Hill Publications.

1.1 BLOCK OBJECTIVES

After doing all the exercises and similar type exercises, you should be able to:

- Make ER diagram for a defined problem;
- Define the ormalized table from the ER diagram;

- Write various SQL based queries;
- Implement the table design into an RDBMS package;
- Make simple reports and forms.

1.2 REVIEW QUESTIONS

Please attempt the following questions to begin with:

1) A database system is fully relational if it supports _______________ and _______________.
2) A Relation resembles a ________ a tuple resembles a ____________ and an attribute resembles a ____________.
3) A candidate key, which is not a primary key is known as a ______key.
4) Primitive operations are union, difference, product, selection and projection. The definition of A intersects B will be________.
5) Which one is not a traditional set operator defined on relational algebra?
   (i) Union
   (ii) Intersection
   (iii) Difference
   (iv) Join
6) Write True/False
   a) Any Binary relation is in 3NF.
   b) Any Binary relation is in BCNF.
   c) Any Binary relation is in 4NF.
   d) Relation R (A, B, C) is equal to the join of its projections R1 (A, B) and R2 (A,C) iff the FD **B** → **C** holds in R.
   e) If R.A → R.B and R.B → R.C then R.A → R.C.
   f) If R.A → R.B and R.A → R.C then R.A → R. (B, C).
   g) If R.B → R.A and R.C → R.A then R.(B, C) → R.A.
   h) If R(B, C) → R, A then R.B → R.A and R.C → R.A
   i) Any relation can be non-loss-decomposed into an equivalent collection of 4NF relations.

Model Answers

1) Relational databases and A language as powerful as relational algebra
2) File, Record, Field
3) Alternate
4) A Minus (A minus B)
5) (iv)
6) a) True

b) True
c) True
d) False
e) True
f) True
g) True
h) False
i) True

Exercise 0

The basic objective of this exercise is to make sure that you have gone through the earlier block. Please try to answers questions in section 1 and section 2 yourself using Block 1 of this course.

1.2.1 DBMS and File Oriented Approach

Question 1. What are the five main differences between a file-processing system and a DBMS?

....

....

....

....

Question 2. What are two major disadvantages of a database system?

....

....

....

....

1.2.2 Relational Databases and Integrity Constraints

Question 1. Define a relational, Domain, Attribute, Primary key.

....

....

....

....

Question 2. What are candidate and alternate key?

....

....

Question 3. Describe the two components of a relation in a relational database.

....

....

Question 4. Define a relational database.

....

....

....

Question 5. List two reasons why null values may be introduced into the database.

....

....

Question 6. What is referential integrity?

....

....

1.3 ENTITY – RELATIONSHIP DIAGRAM

Question 1. (i) Construct an E-R diagram for a car-insurance company that has a set of customers, each of whom owns one or more cars. Each car has associated with it zero to any number of recorded accidents.

(ii) Construct appropriate tables.

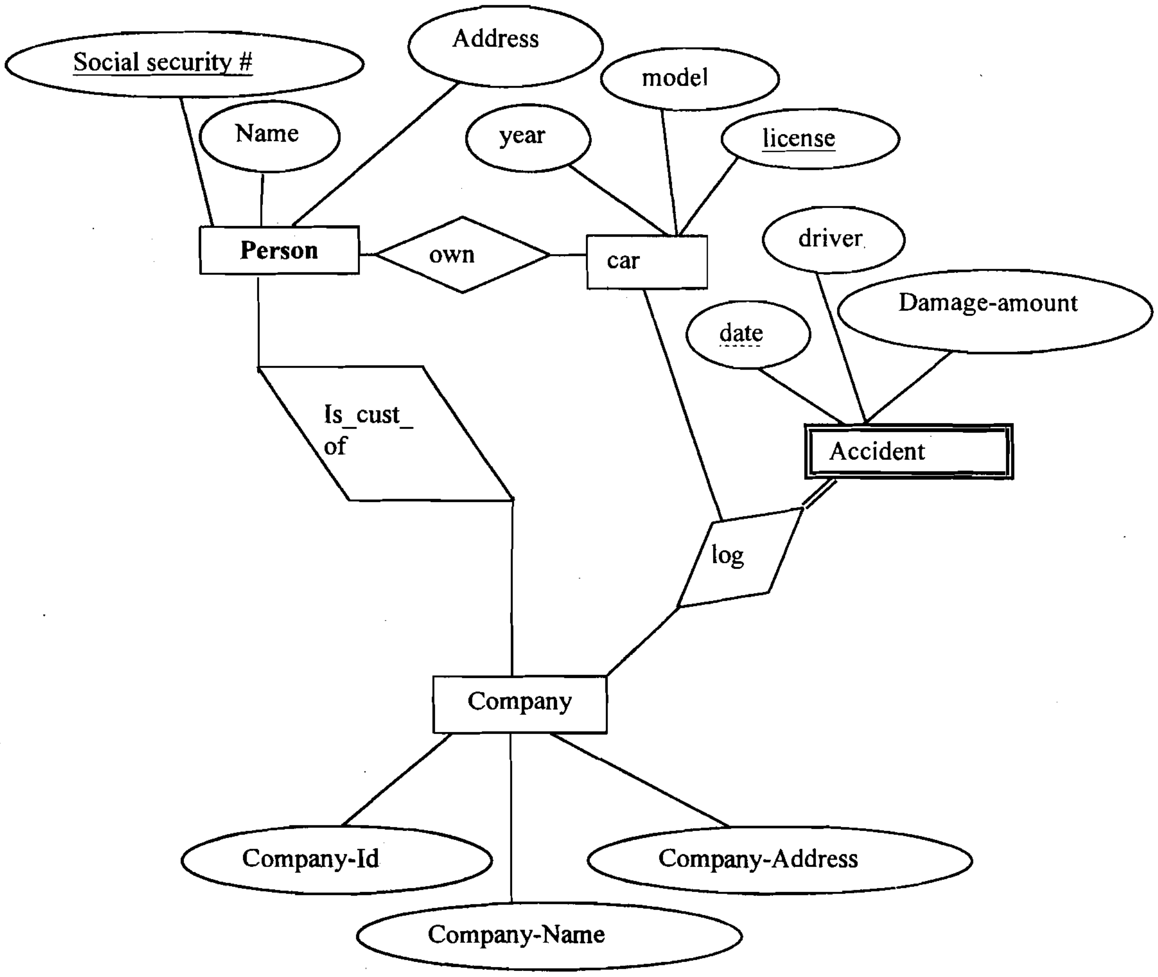
Question 2. Consider an E-R diagram in which the same entity set appears several times. Why is allowing this redundancy a bad practice that one should avoid whenever possible?

Question 3. The room booking side of a small hotel is to be computerized. The hotel has a number of rooms. Each room has a basic (double) price and a supplementary price for extra children. These prices also depend on the time of year- it is more expensive at Christmas, during the summer and around bank holidays, for example. There are 3 seasonal bands. The system must enable the hotel proprietor to answer phone calls from prospective clients (for example, rooms available now and in the future, with costs), make provisional bookings, do mailings of previous clients, prepare clients' bill (ignore extras such as papers, drinks etc). Recognise various entities and relationship among them. Construct an E-R diagram for the above.

Question 4. A student uses a particular computing system to do the computations for a given course using a limited hour account code. Find the appropriate relationship. Draw the E-R diagram.

Model Answers

Answer 1. (i)



(ii) Tables:

Person

Social Security#	Character	10	Primary Key
Name	Character	25	
Address-house-no	Character	20	
Address-street	Character	20	
Address-city	Character	20	
Address-state	Character	20	
Address-pincode	Character	6	

Company

Company-Id	Character	10	Primary Key
Company-name	Character	20	
Company-address	Define as per requirements		

Car

License	Character	10	Primary Key
Model	Character	10	
Year	Numeric		

Own (Assuming many to many relationship)

Social security #	Character	10	Part of Primary Key. Also a Foreign Key referencing Person Table.
License	Character	10	Part of Primary Key. Also a Foreign Key referencing Car Table.

Is-Customer-of

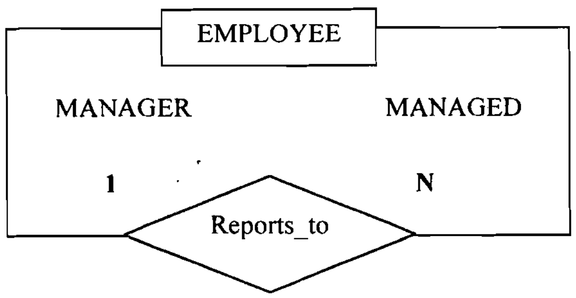
Social security #	Character	10	Part of Primary Key. Also a Foreign Key referencing Person Table.
Company-Id	Character	10	Part of Primary Key. Also a Foreign Key referencing Company Table.

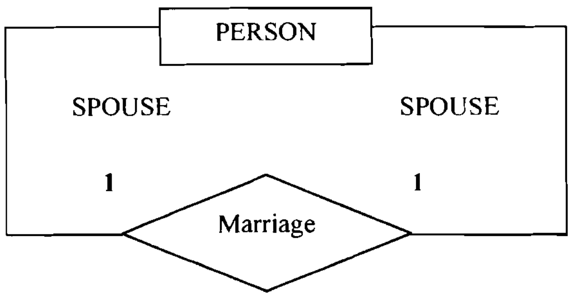
Accident-log

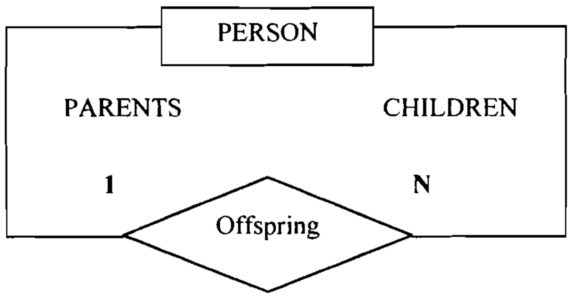
Company-Id	Character	10	Part of Primary Key. Also a Foreign Key referencing Company Table.
License	Character	10	Part of Primary Key. Also a Foreign Key referencing Car Table.

Adate	Date		assumed descriminator in weak entity, part of key.
Driver	Character	25	
Damage-amount	Numeric		

Answer 2. Here are given three examples for an E-R diagram in which the same entity set appears several times







One of the problems with such relationships is that to answer a query you may need to join same tables again and again. However, they may be necessary in some cases.

Answer 3. We are obviously concerned with entities such as CLIENT, ROOM, and BOOKING. A first attempt:

ROOM {<u>RoomNumber</u>, NumberOfDoubles, NumberOfSingles, WashingFacilities, Season, BasicPrice , SupplementaryPrice } are the attributes of ROOM.

Season = [low|middle|high] i.e., The value of season could be either low, middle or high.

CLIENT {<u>ClientName + ClientAddress,</u> ClientTelephoneNumber}

BOOKING {<u>RoomNumber + DateOfArrival</u>, ClientName, ClientAddress, NumberOfGuests, ExpectedStay, Deposit}

The underlined attributes are the primary key of the Entity.

Now there exist attributes in ROOM, which cannot be found *just* from the key. To find the price of a room, you need both to know the RoomNumber and the Season.

We solve this problem by creating a new entity ROOM-PRICE, which allows us to find the prices for any room in any season. These attributes are removed from Room:

ROOM {RoomNumber, NumberOfDoubles, NumberOfSingles, WashingFacilities},

ROOM-PRICE {RoomNumber + Season, BasicPrice, SupplementaryPrice }

Season = ["low"|"middle"|"high"]

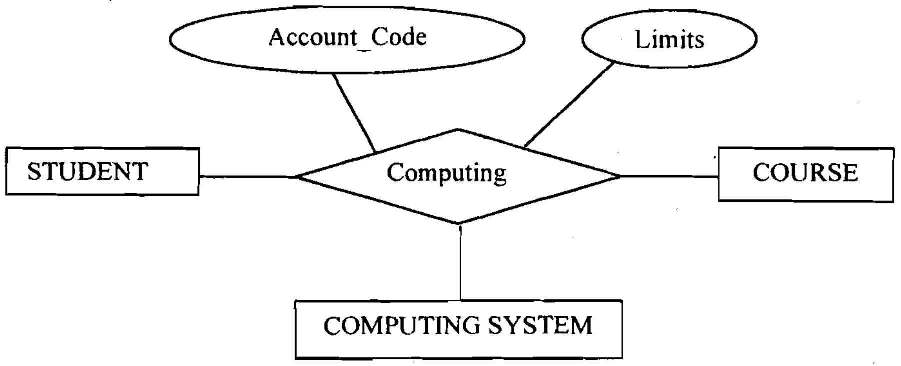
What about current occupancy of the room?

As some clients will book and then occupy, some will occupy without booking, some will book without occupying......

ROOM-OCCUPANCY { RoomNumber + DateArrived , NameAndAddress, ExpectedLeavingDate }

Can we find the Booking from the Room-Occupancy, and if not what else do we need?

Answer 4.



Exercise 1

Question 1. Construct an E-R diagram for a hospital with a set of patients and a set of medical doctors. Associate with each patient a log of the various tests and examinations conducted.

Question 2. Explain the difference between a weak entity set and a strong entity set.

....

....

Question 3. Consider the enitity EMPLOYEE with following attributes:

- Emp_ID
- Employee_Name
- Address
- Phone
- Dependent_Name
- Relationship_to_Employee
- Skill
- Designation
- Designation_Start_Date
- Salary
- Salary_Start_Date

Using the EMPLOYEE entity, convert each of the one-to-many association into a weak entity and a relationship. Identify the discriminator of each weak entity and the attributes of each relationship.

....

....

....

....

Question 4. Give examples of:

- A many-to-many relationship in which one of the participant is a weak entity.
- A many-to-many relationship in which one of the participant is another relationship.
- An existence dependency.
- A weak entity.
- An entity with composite attributes.
- An entity with multivalued attributes.

....

....

....

....

....

....

Question 5. The people's Bank offers five types of accounts: loan, checking, premium savings, daily interest saving, and money market. It operates a number of branches and a client of the bank can have any number of accounts. Accounts can be joint, i.e., more than one client may be able to operate a given account. Identify the entities of interest and show their attributes. What relationships exist among these entities? Draw the corresponding E-R diagram.

Question 6. A University decides to computerise it's registration system. Identify the possible entities and relationships and Draw the E-R Diagram.

1.4 FUNCTIONAL DEPENDENCY AND NORMALISATION

1.4.1 Functional Dependency

Question 1. Find the FDs in following relations. Identify the problem and give the remedy.

Enrollment

Student-Name	Course	Phone No.	Department	Grade
Jones	353	237-4539	Comp. Science	A
Ng	329	427-7390	Chemistry	A
Jones	328	237-4539	Comp. Science	B
Martin	456	388-5183	Physics	C
Dulles	293	371-6259	Decision Science	B
Duke	491	823-7293	Mathematics	C
Duke	353	823-7293	Mathematics	B

Student-Name	Course	Phone No.	Department	Grade
Jones	491	237-4539	Comp. Science	C
Evan	353	842-1729	Comp. Science	A+
Baxter	379	839-0827	English	B

Question 2. Consider the following relation TIME_TABLE.

Professor	Course	Room	Day	Time
Mrs. Shah	Maths	20	Mon.	9:00 a.m.
Mr. Budhiraja	History	10	Mon	9.00 a.m.
Mrs. Shah	Maths	22	Tue	10:00 a.m.
Mrs. Mehta	Commerce	40	Wed	11:00 a.m.
Mr. Budhiraja	History	11	Fri	12:00 a.m.
Mr. Kumar	Maths	23	Wed	12:00 a.m.

Out of the FDs Course → Prof, and Prof → Course, which one is satisfied and which one not. Give reasons.

Question 3. Consider the relation STUDENT_INFORMATON.

Name	Course	Tel No.	Major	Prof	Grade
Jyoti	History	5579590	Maths	Mrs. Shah	A
Neeta	English	4254589	Hindi	Mrs. Vats	B
Jyoti	Maths	5579590	Maths	Mr. Kumar	B
Mahesh	Hindi	6239387	Physics	Mr. Venkatesh	A
Sumit	Computer Science	2231748	Commerce	Mr. Mishra	C
Vikas	Physical Chemistry	6346666	Chemistry	Mr. Chaubey	B
Vikas	Political Science	6346666	Chemistry	Mr. Pandey	In prog

Find the various FDs in above relation, which are satisfied.

Question 4. Consider the relation STUDENT_ADVISOR (Name, Department, Advisor) with the following functional dependencies:

Name → Department

Name → Advisor

Advisor → Department

Decompose the relation STUDENT_ADVISOR into STUDENT_DEPARTMENT (Name, Department) DEPARTMENT_ADVISOR (Department, Advisor). Tell whether the decomposition is lossy or not.

Question 5. Here are two sets of FDs for a relation R {A, B, C, D, E}. Are they equivalent?

1) A → B   AB → C   D → AC   D → E
2) A → BC   D → AE

Model Answers

Answer 1: The relation has following FD

Student-Name → Phone No.

Student-Name → Department

Student-Name,Course → Grade

The problem with the relation is that unless the student takes atleast one-course, we cannot enter data for the student. And the other problem is that a change in the Phone No. or Department can lead to inconsistencies in the database. To rectify the relation is decomposed as follows:

Student

Student-Name	**Phone No.**	**Department**
Jones	237-4539	Comp. Science
Ng	427-7390	Chemistry
Martin	388-5183	Physics
Dulles	371-6259	Decision Science
Duke	823-7293	Mathematics
Evan	842-1729	Comp. Science
Baxter	839-0827	English

Enroll:

Student-Name	**Course**	**Grade**
Jones	353	A
Ng	329	A
Jones	328	B
Martin	456	C
Dulles	293	B
Duke	491	C
Duke	353	B
Jones	491	C
Evan	353	A+
Baxter	379	B

Answer 2. In order to verify whether a given FD X → Y is satisfied by a relation R on a relational scheme R or not, we find any two tuples with the same X value, the FD X → Y is satisfied in R, if for the same value of X the values of Y in tuples must be same.

Thus, here the FD Prof → Course is satisfied, as for any two same value for Prof, the value of Course will always be same, example, Mr. Budhiraja is teaching history only, and Mrs. Shah is teaching mathematics only.

But the FD Course→ Prof is not satisfied, as Maths could be taught by Mrs. Shah as well as by Mr. Kumar, thus, for the same value of Course the value of Prof is not same.

Answer 3. The following function dependencies are satisfied in the above relation.

Name → Tel No

Name → Major

Name, Course → Grade

Course → Prof (The present instance of table does indicate this FD as well Prof → Course FD, one needs more data to find which of the two actually hold or ask questions as per example 2 above.)

Answer 4. We decompose the STUDENT_ADVISOR into two relations:

STUDENT_DEPARTMENT and DEPARTMENT_ADVISOR. We join the both of the relations STUDENT_DEPARTMENT and DEPARTMENT_ADVISOR into a new relation NEW_ STUDENT_ADVISOR.

We can clearly see that the new relation NEW_ STUDENT_ADVISOR is lossy.

STUDENT_ADVISOR

Name	Department	Advisor
Jyoti	Maths	Mrs. Shah
Neeta	Hindi	Mrs. Vats
Mahesh	Physics	Mr. Venkatesh
Sumit	Commerce	Mr. Mishra
Vikas	Chemistry	Mr. Chaubey

STUDENT_DEPARTMENT

Name	Department
Jyoti	Maths
Neeta	Hindi
Mahesh	Maths
Sumit	Physics
Vikas	Commerce
Mr. Chaubey	Chemistry
Mr. Pandey	Chemistry

DEPARTMENT_ADVISOR

Advisor	Department
Mrs. Shah	Maths
Mrs. Vats	Hindi
Mr. Kumar	Maths
Mr. Venkatesh	Physics
Mr. Mishra	Commerce

NEW_ STUDENT_ADVISOR

Name	Department	Advisor
Jyoti	Maths	Mrs. Shah
Jyoti	Maths	Mr. Kumar
Neeta	Hindi	Mrs. Vats
Mahesh	Physics	Mr. Venkatesh
Sumit	Commerce	Mr. Mishra
Vikas	Chemistry	Mr. Chaubey
Vikas	Chemistry	Mr. Pandey

Please note that Student_Advisor relation suffers from Insertion anomaly. The information that Mr. Kumar is an advisor of Maths department cannot be stored in the table as he is not advisor to any student at present. This information is shown in Department_Advisor. On taking join of Department_Advisor and Student_Department one will get as many tuples per student as the total number of advisors in that department. Mathematically, the joining attribute of the two tables is Department. The decomposition would have been lossless if any of the following had hold: Department → Name OR Department → Advisor. Both of them are not in the FDs, thus decomposition is lossy.

Please note that this decomposition is not preserving the FDs also as the functional dependency Name → Advisor is lost. Please find out which normal form the relation is presently in. If we decompose the relation as Student_Advisor and Advisor_Department, the decomposition will be loss-less. Please note in the said case the dependency Name → Department will be maintained as transitive **dependencies:**

Name → Advisor (in Name_Advisor)
and Advisor → Department (in Advisor_Department)
=> Name → Department

Answer 5. They are equivalent. Let us consider the FDs of the first set as follows:

1) A → B
2) AB → C
3) D → AC
4) D → E

Please note that if A→ B, then for any given value of A, B can be determined, which implies that in FD AB→ C, only by knowing A we can determine B and, therefore, C also.

Thus instead of FD AB → C; A→ C is a better version as A → B exist.

The FD D → AC also means, D → A as well as D→ C, since A → C has been just derived therefore, D → A and A → C implies D → C by transitivity and so can be dropped, leaving the FD.

D → A

The fist set of FDs is thus equivalent to the following irreducible set:

A → B
A → C
D → A
D → E

The second set of FDs

A → BC
D → AE is clearly also equivalent to this irreducible set. Thus, the two given sets are equivalent.

Exercise 2

Question 1: Given a relation R(x, y, z, t) and a set of functional dependence

X → y
x → z
x → t

Is the decomposition of relation in R1 (x, y), R2 (x, z) and R3 (x, t) loss-less? Is it necessary to decompose the table? Suggest situations where you would like to have such decomposition.

....

....

Question 2: Remove the redundant FDs from the following set:

A → BC

AB → CD

B → D

D → E

BC → D

What are the possible candidate keys?

....

....

....

Question 3: In a given instance of a relation, how will you ascertain whether a given FD exists? Explain with examples.

....

....

....

....

Question 4. What is non-key?

....

....

Question 5. Define Second-Normal Form.

....

....

Question 6. Define the Third Normal Form and the Boyce/Codd Normal form with examples.

....

....

....

....

Question 7. What are Multilevel Dependencies?

....

....

....

....

1.4.2 Normalisation

Consider the following relation STUDENT. Normalise the following relation:

Table 4.1

Student-Id	**FirstName**	**LastName**	**Major**
123-45-6789	Jack	Jones	Library and Information Science
222-33-4444	Lynn	Lee	Library and Information Science
987-65-4321	Mary	Ruiz	Pre-Medicine
123-54-3210	Lynn	Smith	Pre-Law
111-33-5555	Jane	Jones	Library and Information Science

You can easily verify for yourself that above relation STUDENT satisfies the definition of 1NF: viz., it has no duplicated rows; each cell is single-valued (i.e., there are no repeating groups or arrays); and all the entries in a given column are of the same kind.

In Table STUDENT, we can see that the key, Student-Id, functionally determines the other attributes; i.e., a given Student-Id implies (determines) a particular value for each of the attributes FirstName, LastName, and Major (assuming, at least for the moment, that a student is allowed to have only one major). In the arrow notation:

Student-Id → FirstName

Student-Id → LastName

Student-Id → Major.

A key attribute will, by the definition of key, uniquely determine the values of the other attributes in a table; i.e., all non-key attributes in a table will be functionally dependent on the key. But there may be non-key attributes in a table that determine other attributes in that table. Consider the following table:

Table 4.2

FirstName	**LastName**	**Major**	**Level**
Jack	Jones	LIS	Graduate
Lynn	Lee	LIS	Graduate
Mary	Ruiz	Pre-Medicine	Undergraduate
Lynn	Smith	Pre-Law	Undergraduate
Jane	Jones	LIS	Graduate

In Table 4.2 the Level attribute can be said to be functionally dependent on the Major attribute. Thus, we have an example of an attribute that is functionally dependent on a non-key attribute. This statement is true in the table *per se*, and that is all that the definition of functional dependence requires; but the statement also reflects the real-world fact that Library and Information Science (LIS) is a major that is open only to graduate students and that Pre-Medicine and Pre-Law are majors that are open only to undergraduate students.

Table 4.2 has another interesting aspect. Its key is a composite key, consisting of the paired attributes, FirstName and LastName. The Level attribute is functionally dependent on this composite key, of course; but, in addition, Level can be seen to be dependent on only the attribute LastName. (This is true because each value of Level is paired with a distinct value of LastName. In contrast, there are two occurrences of the value Lynn for the attribute FirstName, and the two Lynns are paired with different values of Level, so Level is not functionally dependent on FirstName.)

Problems in Table 4.2–

Thus this table fails to qualify as a 2nd Normal Form table, since the definition of 2NF requires that all non-key attributes be dependent on all of the key. (Admittedly, this example of a partial dependency is artificially contrived, but nevertheless it illustrates the problem of partial dependency.)

We can turn Table 4.2 into a table in 2NF in an easy way, by adding a column for the Student-Id, which will then be the natural thing to use as the key.

Table 4.3

Student-Id	FirstName	LastName	Major	Level
123-45-6789	Jack	Jones	LIS	Graduate
222-33-4444	Lynn	Lee	LIS	Graduate
987-65-4321	Mary	Ruiz	Pre-Medicine	Undergraduate
123-54-3210	Lynn	Smith	Pre-Law	Undergraduate
111-33-5555	Jane	Jones	LIS	Graduate

With the Student-Id defined as the key, Table 4.3 is in 2NF, as you can easily verify. This illustrates the fact that any table that is in 1NF and has a single-attribute (i.e., a non-composite) key is automatically also in 2NF.

Problems in Table 4.3–

1. It contains some repeated information about the LIS-Graduate pairing.

2. Consider what would happen to our knowledge (at least, as explicitly contained in a table) of the level of the major, Pre-Medicine, if Mary Ruiz left Enormous State University. With the deletion of the row for Ms. Ruiz, we would lose the information that Pre-Medicine is an Undergraduate major. This is an example of a **deletion anomaly.** We may possess the real-world information that Pre-Medicine is an Undergraduate major, but no such information is explicitly contained in a table in our database.

3. Suppose that a new student wants to enroll in ESU: e.g., suppose Jane Doe wants to major in Public Affairs. From the information in **Table 4.1** we cannot tell whether Public Affairs is an Undergraduate or a Graduate major; in fact, we do not even know whether Public Affairs is an established major at ESU. We do not know whether it is permissible to insert the value, Public Affairs, as a value of the attribute, Major, or what to insert for the attribute, Level, if we were to assume that Public Affairs is a valid value for Major. The point is that while we may possess real-world information about whether Public Affairs is a major at ESU and what its level is, this information is not explicitly contained in any table that we have thus far mentioned as part of our database. This is **insertion anomaly**.

   Although Table 4.3 is in 2NF, it is still open to the problems of insertion and deletion anomalies, as the discussion in the preceding section shows. The reason is that Table 4.3 deals with more than a single theme.

   What can we do to turn it into a set of tables that are, or at least come closer to being, single-theme tables?

   A reasonable way to proceed is to note that Table 4.3 deals with both information about students (their names and STUDENT-IDs) and information about majors and levels. This should strike you as two different themes.

   Presented as follows is one possible set of single-theme tables dealing with the information in Table 4.3. One question which has also been addressed is: Can a student have more than one majors? If the answer is not then we can represent the information in just two relations R1 (Student-Id, FirstName, LastName, major) and R2 (Major, Level). But if the answer to the question is no (please note in that case key in table 4.3 will be (Student-Id + Major) then we need to decompose the relation in following three tables:

Table 4.4

Student-Id	FirstName	LastName
123-45-6789	Jack	Jones
222-33-4444	Lynn	Lee
987-65-4321	Mary	Ruiz
123-45-4321	Lynn	Smith
111-33-5555	Jane	Jones
999-88-7777	Newton	Gingpoor

Table 4.5

Major	Level
LIS	Graduate
Pre-Medicine	Under-graduate
Pre-Law	Under-graduate
Public Affairs	Graduate

Table 4.6

Major	Student-Id
LIS	123-456789
LIS	222-33-4444
Medicine	987-68-4321
Pre-Law	123-54-3210
LIS	111-33-5555

The three preceding tables should strike you as providing a better arrangement of the information in Table 4.3.

For one thing, this arrangement puts the information about the students into a smaller table, Table 4.4, which happily fails to contain redundant information about the LIS-Graduate pairing. For another thing, this arrangement permits us to enter information about students (e.g., Newton Gingpoor) who have not yet identified themselves as pursuing a particular major. For still another thing, it puts the information about the Major-Level pairings into a separate table,

Table 4.5, which can easily be expanded to include information (e.g., that the Public Affairs major is at the Graduate level) about majors for which, at the moment, there may be no students registered.

Table 4.6 provides the needed link between individual students and their majors (note that Newton Gingpoor's STUDENT-ID is not in this Table 4.6, which tells us that he has not yet selected a major).

Tables 4.4 to 4.6 are single-theme tables and are in 3NF, as you can easily verify.

Please note that the example covered here just try to reflect how you may do a design without worrying too much about the definitions of normalisation. However, for complete details on FDs, MVDs and normalisation please refer to block1 unit 3 of this course where overall process is explained through examples.

We are not giving any specific exercises in this section as normalisation is part of overall design process, therefore, you must normalize all the designs, which you will create as per further exercises.

1.5 STRUCTURED QUERY LANGUAGE (SQL)

You have already gone through SQL in CS-06, we present an example here.

Consider the supplier relations.

S

S# (Supplier No.)	SNAME (Supplier Name)	STATUS	CITY
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Adams	30	Athens
S5	Clark	20	London

SP

S#	P# (Part No.)	Quantity
S1	P1	300
S1	P2	200
S2	P1	100
S2	P2	400
S3	P2	200
S4	P2	200

Question 1. Get supplier numbers for suppliers with status > 20 and city is Paris.

....

Question 2. Get Supplier Numbers and status for suppliers in Paris, in descending order of status.

....

Question 3. Get all pairs of supplier numbers such that the two suppliers are located in the same city. (**Hint:** It is retrieval involving join of a table with itself.)

....

Question 4. Get unique supplier names for suppliers who supply part P2.

....

Question 5. Give the same query above by using the operator IN.

....

Question 6. Get part numbers supplied by more than one supplier. (**Hint:** It is retrieval with a sub-query, with interblock reference and same table involved in both blocks).

....

Question 7. Get supplier numbers for suppliers who are located in the same city as supplier S1. (**Hint:** Retrieval with sub query and unqualified comparison operator).

....

Question 8. Get supplier names for suppliers who supply part P1. (**Hint:** Retrieval using EXISTS)

....

Question 9. Get part numbers for parts whose quantity is greater than 200 or are currently supplied by S2. (**Hint:** Use a retrieval using union).

....

Question 10. Suppose for the supplier S5 the value for status is NULL instead of 20. Get supplier numbers for suppliers greater than 25 or is NULL. (**Hint:** Retrieval using NULL).

....

Question 11. Get the number of Suppliers, who are supplying at least one part. (**Hint:** This query is using the built-in function count).

....

Question 12. For each part supplied, get the part no. and the total quantity supplied for that part. (**Hint:** The query using GROUP BY).

....

Question 13. Get part numbers for all parts supplied by more than one supplier. (**Hint:** It is GROUP BY with HAVING).

....

Question 14. For all parts such that the total quantity supplied is greater than 300 (exclude from the total all shipments for which quantity is less than or equal to 200), get the part no. and the maximum quantity of the part supplied, and order the result by descending part no. within those maximum quantity values which are in ascending order.

....

Question 15. Double the status of all suppliers in London. (**Hint:** UPDATE Operation).

....

Question 16. Let us consider the table TEMP has one column, called P#. Enter into TEMP part numbers for all parts supplied by S2.

....

Question 17. Add part P7.

....

Question 18. Delete all the suppliers in London and also the supplies concerned.

....

Question 19. Consider the following Relational database.

employees (eno, ename, address, basic salary)

projects (pno, pname, nos-of-staff-alotted)

workin (pno,eno,pjob)

Two queries regarding the data in the above database have been formulated in SQL. Describe the queries in English sentences.

(i)
```
SELECT ename
FROM employees
WHERE eno IN ( SELECT eno
          FROM workin
          GROUP BY eno
          HAVING COUNT (*) =
         (SELECT COUNT (*) FROM projects));
```

(ii)
```
SELECT pname
FROM projects
WHERE pno IN ( SELECT pno
          FROM projects
          MINUS
          (SELECT DISTINCT pno
            FROM workin
            GROUP BY eno));
```

....

....

....

....

....

....

....

Model Answers

Please note:

In certain Database Management System S# or P# may not be accepted kindly use sno, pno instead.

Answer 1.
```
SELECT S#
FROM S
WHERE CITY = 'PARIS'
AND STATUS > 20
```

Result:

S#
S3

Answer 2.
```
SELECT S#, STATUS
FROM S
WHERE CITY = 'PARIS'
ORDER BY STATUS DESC
```

Result:

S#	STATUS
S3	30
S2	10

Answer 3.

```
SELECT FIRST.S# , SECOND.S#
FROM S FIRST , S SECOND
WHERE FIRST.CITY = SECOND.CITY AND FIRST.S# < SECOND.S#
```

Please note the condition after AND is needed to make sure that you do not get tuples where Supplier number is same; and do not get the tuples which are repeated in reverse order. For example, S1 S5's reverse order is S5 S1. You may try the query by not giving clause after AND also, you will find the difference.

Result:

S#	S#
S1	S5
S2	S3

Answer 4.

```
SELECT  DISTINCT SNAME
FROM S , SP
WHERE S.S# = SP.S#
AND SP.P# = 'P2'
```

Result:

SNAME
Smith
Jones
Blake
Adams

Or,

```
SELECT SNAME
  FROM S
  WHERE S# = ANY ( SELECT S#
                   FROM SP
                   WHERE P# = 'P2' )
```

In this query, we are making a set of supplier numbers using the sub query "who supply a part P2".

Answer 5.

```
SELECT SNAME
FROM S
WHERE S# IN
(SELECT S#
 FROM SP
 WHERE P# = 'P2')
```

Answer 6.

```
SELECT DISTINCT P#
FROM SP  SPX
WHERE P# IN ( SELECT P#
          FROM SP
          WHERE SP.P# = SPX.P#  AND SP.S# < SPX.S#)
```

Result:

P#
P1
P2

Answer 7.

```
SELECT S#
FROM S
WHERE CITY = (SELECT CITY
               FROM S
               WHERE S# = 'S1')
```

Please note that this query will work as S# is the key to relation S and will result in one city for Supplier S1. It may create problem if value of city for S1 is NULL.

Result:

S#
S1
S5

Answer 8.

```
SELECT SNAME
FROM S
WHERE EXISTS (SELECT *
                        FROM SP
                        WHERE SP.S# = S.S#
AND P# = 'P1')
```

Result:

SNAME
SMITH
JONES

Answer 9.

```
SELECT P#
FROM SP
WHERE QUANTITY > 200 UNION SELECT P#
                                          FROM SP
                                          WHERE S# = 'S2'
```

Result:

P#
P1
P2

Answer 10.

```
SELECT S#
FROM S
WHERE STATUS > 25 OR STATUS IS NULL
```

Result:

S#
S3
S4
S5

Answer 11.

```
SELECT COUNT (DISTINCT S#)
FROM SP
```

Result: 4

Answer 12.

```
SELECT P#, SUM(QUANTITY)
FROM SP
GROUP BY P#
```

Result:

P#	
P1	400
P2	1000

Answer 13.

```
SELECT P#
FROM SP
GROUP BY P#
HAVING COUNT (*) > 1
```

Please note it is almost the same query as we had in question 6.

Answer 14.

```
SELECT P#, MAX(QUANTITY)
FROM SP
WHERE QUANTITY > 200
GROUP BY P#
HAVING SUM(QUANTITY) > 300
ORDER BY 2, P# DESC
```

This is a very interesting query, please try various ascending, descending combinations.

Answer 15.

```
UPDATE S
SET STATUS = 2 * STATUS
WHERE CITY = 'LONDON'
```

Answer 16.

```
INSERT INTO TEMP
SELECT P#
FROM SP
WHERE S# = 'S2'
```

Answer 17.

```
INSERT INTO SP( S#,P#,QUANTITY)
VALUES ( 'S5','P7',100)
```

Note: Part P7 can be added only if one Supplier is supplying it; therefore, we assume that Supplier S5 is supplying it.

Answer 18. To make sure that on deletion of few tuples from S will cause Cascading deletion from SP, we need to set up Referential Integrity constraint. If this constraint is set between SP and S, then the deletion of record in S will cause Cascading deletion of records in SP.

Thus, in such cases the following command will be sufficient to perform the required deletion.

```
DELETE FROM S
WHERE S# IN ( SELECT S#
                        FROM S
                        WHERE CITY = 'LONDON')
```

Answer 19. (i) Give names of employees who are working on all projects.

(ii) Give names of the projects, which are currently not being worked upon.

Exercise 3

Question 1. An orchestra database consists of the following relations:

CONDUCTS (Conductor, Composition)

REQUIRES (Composition, Instrument)

PLAYS (Player, Instrument)

LIKES (Player, Composition)

Give the query in SQL,

1) List the players and their instruments who can be part of the orchestra when Latifa Melody conducts.

2) From the list of players on previous page, identify those who would like the composition they are to play.

....

....

....

....

Question 2. Consider the following relational scheme:

COURSES (cno, cname)

STUDENTS (roll no, Sname, age,year)

REGISTERED FOR (cno, roll no)

The underlined attributes indicate the primary key for relations. The "year" attributed for STUDENTS relation indicates the year in which the student is currently studying (First year, second year etc.)

Write a SQL Query to find the age and year of youngest student in each year.

....

....

....

....

Question 3. Consider the Car-insurance database given by Q1 in Section E-R Diagrams. Construct the tables and enter meaningful data and create following SQL Queries for this relational database.

1) Find the total number of people whose cars were involved in accidents in 1998.
2) Find the total number of accidents in which the cars belonging to "Ram Kumar" were involved.
3) Add a new customer to the database.
4) Delete the Mercedes belonging to "Ram Kumar."
5) Add a new accident record for the Toyota belonging to "Rashmi Sinha."

....

....

....

....

....

Question 4. Consider the DIRECTOR table given below:

Director

Dir_id	Dir_lname	Dir_fname	Dir_phone	Dir_address	Dir_city	Dir_state	Dir_zip
D001	Lawrence	Martin	415658-9932	6223, bateman	Berkely	CA	94705
D002	Leker	Larry	415986-7020	309, 63rd floor, St # 441	Oakland	CA	94618
D003	Nichols	Mike	415548-7723	589 Darwin Ln.	Berkely	CA	94705
D004	Schaeffer	Eric	801826-0742	67, Seventh Avenue	Salt lake city	UT	84152
D005	Frears	Stefen	801826-0752	3 balding Place	Salt lake city	UT	84152

Some sample records are given in the table above.

a) Add more records in the table using SQL.

b) Perform following queries:

- Display all the rows in the director table.
- List the name of directors who live in state CA.
- List the id's of all directors. To improve readability change the headings suitably.
- List the name of directors who have id's between 'D003' and 'D005' (both inclusive).
- List the names of all directors whose phone numbers start with 41.

....

....

....

....

Question 5. For the PROJECT, EMPLOYEE, and ASSIGNED_TO relations–

PROJECT (Project#, Project_Name, Chief_Architect)

EMPLOYEE (Emp#, EmpName)

ASSIGNED_TO (Project#, Emp#)

express the following queries in SQL:

1) Get Emp# of employees working on project numbered COMP100.

2) Get details of employees (name and number) working on project COMP100.

3) Get details of employees working on all database projects.

4) Get details of employees working on both COMP100 and COMP101.

5) Get employee numbers of employees who work on at least all those projects that employee105 works on.

6) Get employee numbers of employees who do not work on project COMP200.

7) Get employee numbers of employees who work on all projects.

8) Get employee numbers of employees who work on at least one project that employee 105 works on.

....

....

....

....

....

....

....

....

....

....

Question 6. Consider the employee database given below. Give an expression in SQL for each of the following queries:

EMPLOYEE (Employee-Name, Street, City)

WORKS (Employee-Name, Company-Name, Salary)

COMPANY (Company-Name, City)
MANAGES (Employee-Name, Manager-Name)

1) Find the names of all employees who work for First Bank Corporation.

2) Find the names and cities of residence of all employees who work for the First Bank Corporation.

3) Find the names, street address, and cities of residence of all employees who work for First Bank Corporation and earn more than Rs.10,000.

4) Find the employees in the database who live in the same cities as the companies for which they work.

5) Find all employees in the database who live in the same cities and on the same streets as do their managers.

6) Find all employees in the database who do not work for First Bank Corporation.

7) Find all employees in the database who earn more than every employee of Small Bank Corporation.

8) Assume that the companies may be located in several cities. Find all companies located in every city in which Small Bank Corporation is located.

9) Find all employees who earn more than the average salary of all employees of their company.

10) Find the company that has the most employees.

11) Find the company that has the smallest payroll.

12) Find those companies whose employees earn a higher salary, on average, than the average salary at First Bank Corporation.

....

....

....

....

....

....

....

....

....

....

....

1.6 MICROSOFT-ACCESS

Question 1. Design a package known as "Library Book Collection System" that will make accessing books and author details in a library easier and more efficient. It should make use of simple forms and queries to generate reports on authors and books.

Question 2. Consider a database for EMPLOYEE and their dependents and show the steps of creating tables, giving queries, making forms and reports etc. in MS-ACCESS.

Model Answers

Answer 1. In the beginning we need to know that what are our goals in a "Library Book Collection System", i.e., what should be the functionality in the system.

In broad sense, our "Library Book Collection System" should be able to provide:

- Detailed Report on all authors and books
- List of books on same topics or subjects.
- Displaying all the books written by an author.

Please draw the E-R diagrams of this system yourself assuming at least three entities: Book, Author, and Publisher.

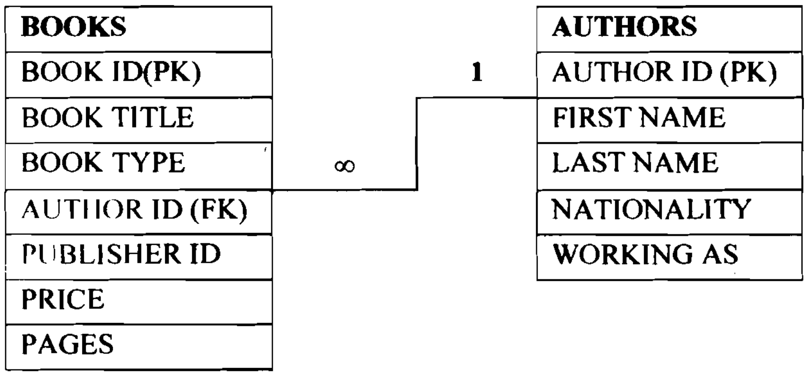
However, the objective of this example is to show how tables and forms may be created, so we are assuming a very elementary design of the database, as follow:

Database Name: Book Collection

Number of tables: 2

Name of tables: 1) Books and, 2) Authors

Now let draw schematic diagram to show table Relationships using primary/foreign key.



Assumption: One book is written by only one author. (You must attempt to remove this assumption and in that case your design will be different. In fact it will consist of three tables.)

PK: PRIMARY KEY

FK: FOREIGN KEY

FORM DESIGN

Form Name	**Form Type**	**Description**	**Table/Query used**
BOOK DETAILS	Columnar	Used to enter data in the BOOKS table	BOOKS & AUTHOR
AUTHOR DETAIL	Columnar	Used to enter data in the AUTHORS table	AUTHORS

The form would like as follows:

AUTHOR DETAILS

AUTHOR ID	
FIRST NAME	
LAST NAME	
NATIONALITY	
WORKING AS	
PHOTOGRAPH	

BOOK DETAILS

BOOK	
BOOK TITLE	
BOOK TYPE	
AUTHOR-ID/NAME	
PUBLISHER-ID	
PRICE	
PAGES	

BOOK ID	**TITLE**	**TITLE TYPE**	**PUB. ID**	**PRICE**	**PAGES**

Validations to be performed:

1) BOOK ID in the BOOKS table to be unique.

2) AUTHOR ID in the AUTHORS table to be unique.

3) BOOK TYPE to be "General", "Technical", "Fiction", "Science" etc.

4) PRICE > 0.

5) TOTAL PAGES > 0

6) None of the fields to contain NULL values.

7) AUTHOR ID in BOOKS table must exist in AUTHORS table (Referential Integrity).

QUERIES USED

Query Name	Query Type	Description	Table used
AUTHORS	Simple	Displays details of all authors	AUTHORS
BOOKS	Simple	Displays details of all books	BOOKS
BOOK AND AUTHORS	Simple	Displays both book details and their corresponding author details	AUTHORS AND DETAILS

REPORT OUTLINE

Report Name	Report Type	Description	Tables/Query used
AUTHORS	AUTO REPORT COLUMNAR	Displays details of all authors	AUTHORS
BOOKS AND AUTHORS	MULTIPLE TYPE REPORT	Displays details of all authors and books	AUTHORS AND BOOKS

REPORTS

AUTHORS USED

AUTHORS

AUTHOR ID	1
FIRST NAME	Hari Shankar
LAST NAME	Parsai
NATIONALITY	Indian
WORKING AS	Artist
PHOTOGRAPH	

AUTHORS AND BOOKS

AUTHOR ID	1
NAME	Hari Shankar Parsai
NATIONALITY	Indian
WORKING AS	Artist

BOOKS BY HIM:

BOOK TYPE Literature (Satire)

BOOK ID	BOOK TITLE	TOTAL PAGES	PRICE
1	Tab ki baat aur thi	100	50/-
2	Kahani Sangrah	500	100/-

The above is just a sample design; you may create your own designs based on the needs. Try implementing above design in MS-Access after going through answer 2 given here.

Answer 2. Let us consider the table Employee and dependents with the description as follows. See the steps for creating the table in the third block of your course i.e.. the block for MS-ACCESS. You must design more complex database after going through that block.

As we know any table in MS-Access has three columns viz., FIELD NAME, DATA TYPE, DESCRIPTION.

Let us consider the case of EMPLOYEE table. The columns of EMPLOYEE table will have following fields:

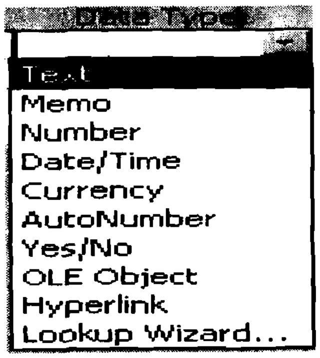
- Employee Name
- Employee Code
- Employee Salary
- Employee Date of joining

Let us consider the case of DEPENDENTS table. The columns of DEPENDENTS table will have following fields:

- Field Name
- Employee Code
- Dependent Name
- Relation with Employee

Datatype

There are different datatypes provided by the MS-Access, as:



For above example, we will choose the following data types for the various attributes.

EMPLOYEE

Field Name	Data Type
Employee Name	text
Employee Code	Text
Employee Salary	number
Employee Date of joining	date/time

DEPENDENTS

Field Name	Data Type
Employee Code	Text
Dependent Name	Text
Relation with Employee	Text

Description Field

It contains the description of the field for example for the Employee name the description will be "this field is for the name of the Employee of the Organisation".

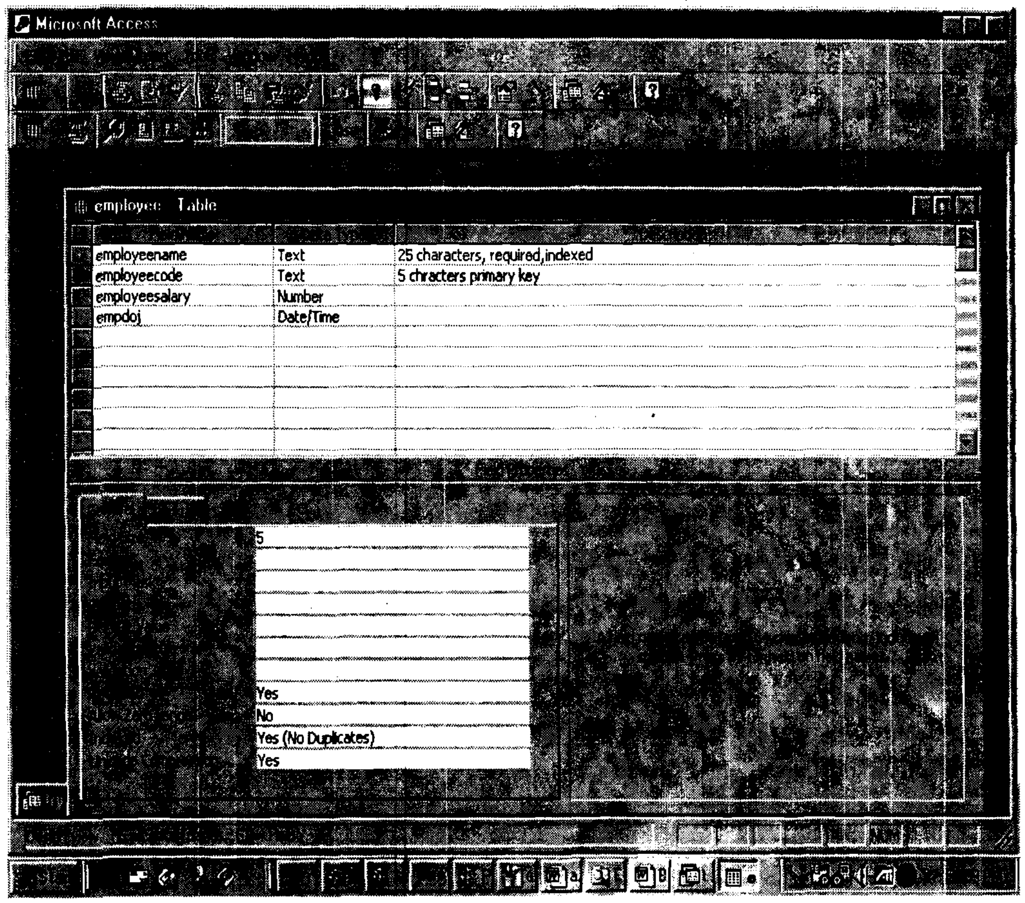
Creating a Table

The easiest way to create most common tables for business or personal use is with the table Wizard however you can also create a table without the Table Wizard. Here we will be considering creating a table without using the wizard. The steps are:

1) In the Database window, click the Table button (or choose Tables from the View menu).

2) Choose the New button. Microsoft Access displays the New Table dialog box.

3) Choose the New Table button. Microsoft Access opens a blank Table window in Design view.



employeename	Text	25 characters, required,indexed
employeecode	Text	5 chracters primary key
employeesalary	Number	
empdoj	Date/Time	

```
5







Yes
No
Yes (No Duplicates)
Yes
```

The above figure shows the creation of the Employee table in the same way you should create the Dependent Table.

When you design or modify a table, you specify the fields that you want it to contain in the upper portion of the Table window. In the lower portion you can enhance your table by setting properties for each field.

Setting a Primary Key

Microsoft Access works most efficiently if you specify a primary key. The primary key of a table consists of one or more fields that uniquely identify each record you store in the table. A primary key is often an ID number or code, since this type of value is always different for each record.

Steps to set the Primary key:

1) Click the field that you want to use for the primary key. To create a multiple-field primary key, hold down the CTRL key and click the field selector to the left of each field that you want to include.

2) From the Edit menu, choose Set Primary Key (or click the set primary key button on the toolbar).

3) Microsoft Access adds a key indicator to the left of the field or fields you specify as the primary key.

In the Employee Table the Primary key is the Employee code and in the Dependent Table the Primary key is a multiple-field primary key, which includes Employee code and Dependent name

Saving the Table

After you add fields to a table, you must save the table design before you can add any records. After you save the Table, its title appears in the list of tables in the Database window.

Steps to save and name a Table:

1) From the File menu, choose Save.

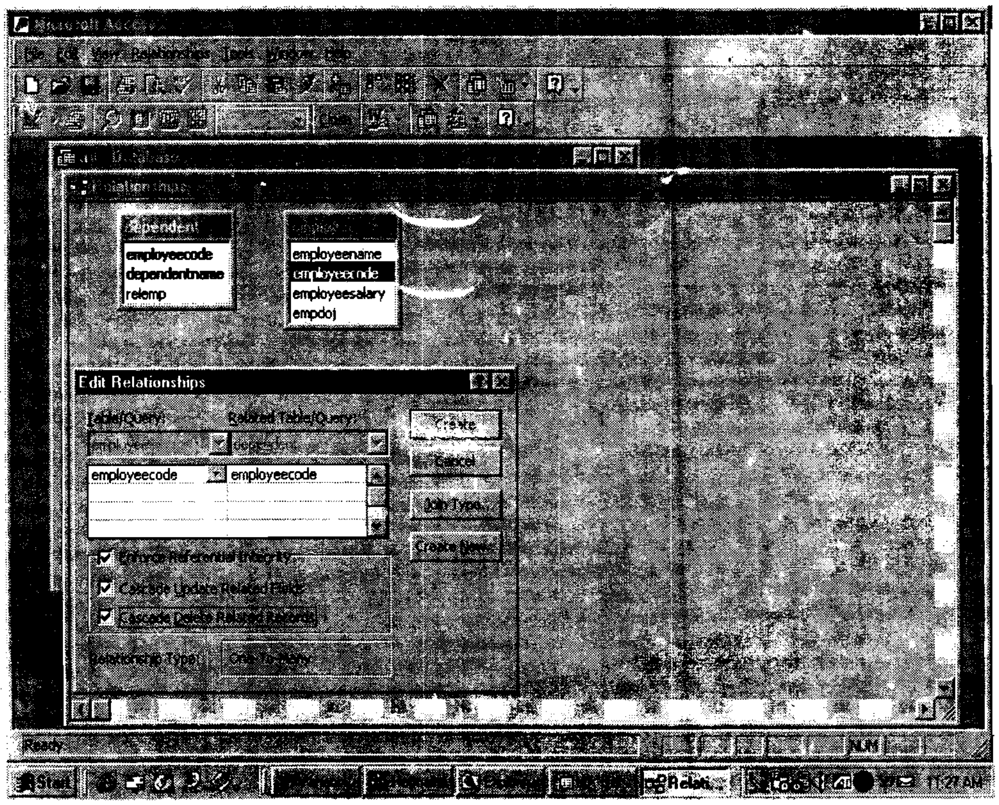
2) If you are saving the table fore the first time, type a name for the table, and then choose OK.

Setting of Relationship

Steps for setting of relationship:

1) Employee code of EMPLOYEE should be linked to employee code of DEPENDENT.

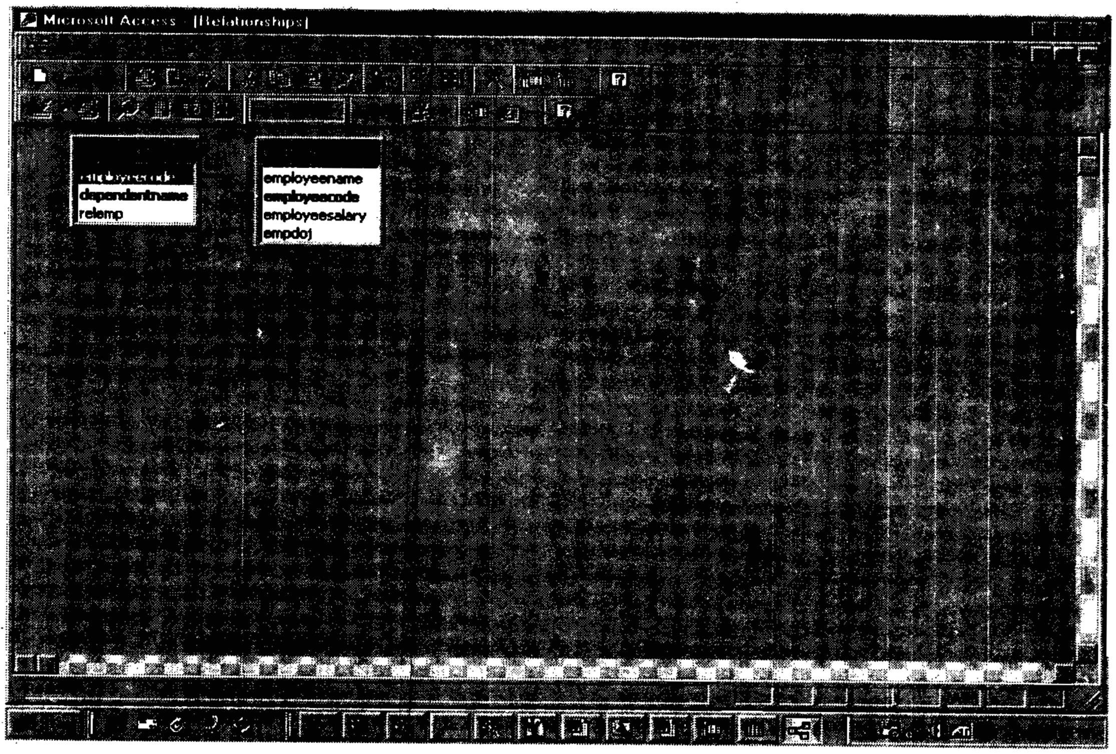
2) Here we would like to enforce referential integrity with Cascade, Delete and Update related records because if some employee leaves the Company then his Dependent records should also be deleted.



Table/Query: employee	Related Table/Query: dependent
employeecode	employeecode

[✓] Enforce Referential Integrity
[✓] Cascade Update Related Fields
[✓] Cascade Delete Related Records

Relationship Type: One-To-Many



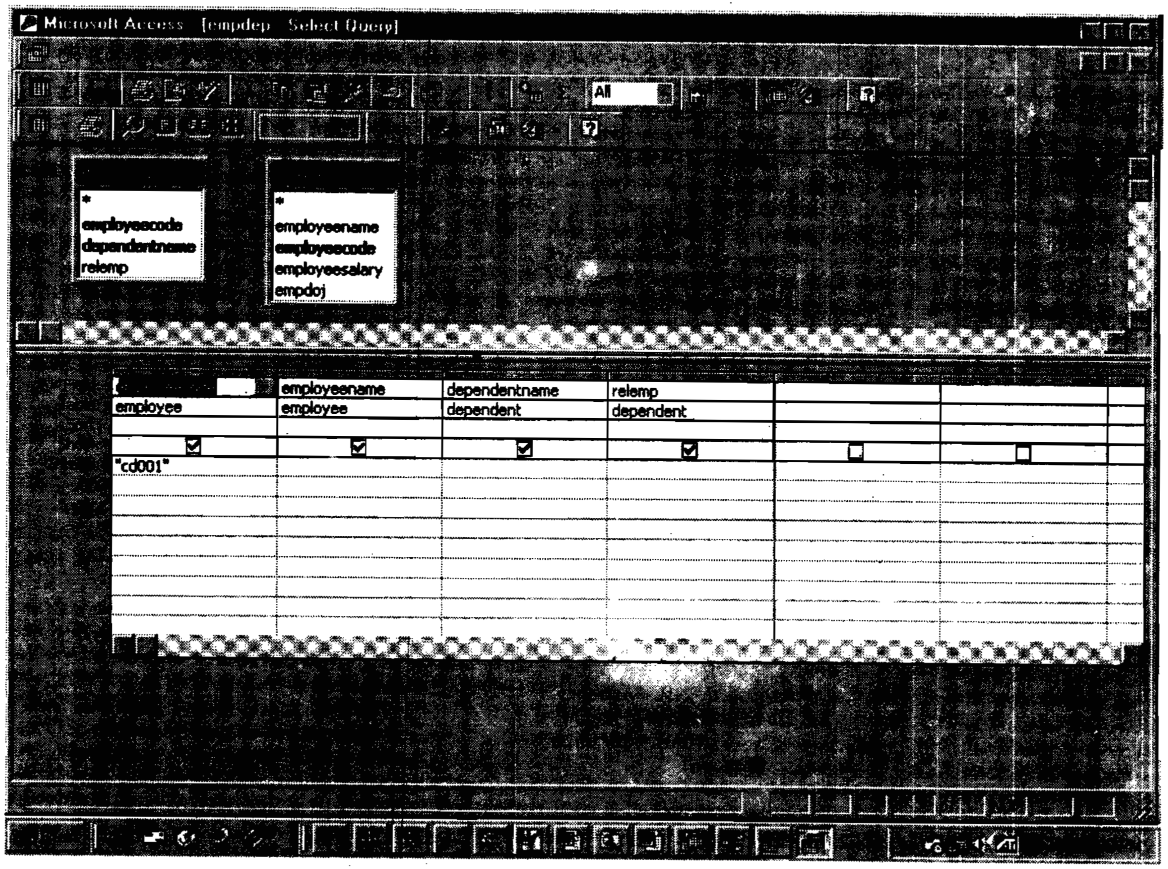
Microsoft Access - [Relationships]

employeecode
dependentname
relemp

employeename
employeecode
employeesalary
empdoj

Creating a Query

Here we will be creating a Query to display the details of an employee including his dependents when the employee code is given.



Microsoft Access - [empdep : Select Query]

*
employeecode
dependentname
relemp

*
employeename
employeecode
employeesalary
empdoj

[illegible]	employeename	dependentname	relemp		
employee	employee	dependent	dependent		
✓	✓	✓	✓	☐	☐
"cd001"					

To Create a Query without a Query Wizard–

1) In the Database window, click the Query button.

2) Choose the New button

3) Microsoft Access displays the New Query dialog box.

4) Choose the New Query button.

5) Microsoft Access opens a Select Query window and displays the Add Table dialog box, which displays the tables and queries in your database.

6) Select the table that contains the data you want to add to your query:

7) In the Query window, Microsoft Access displays a field list for each table you select.

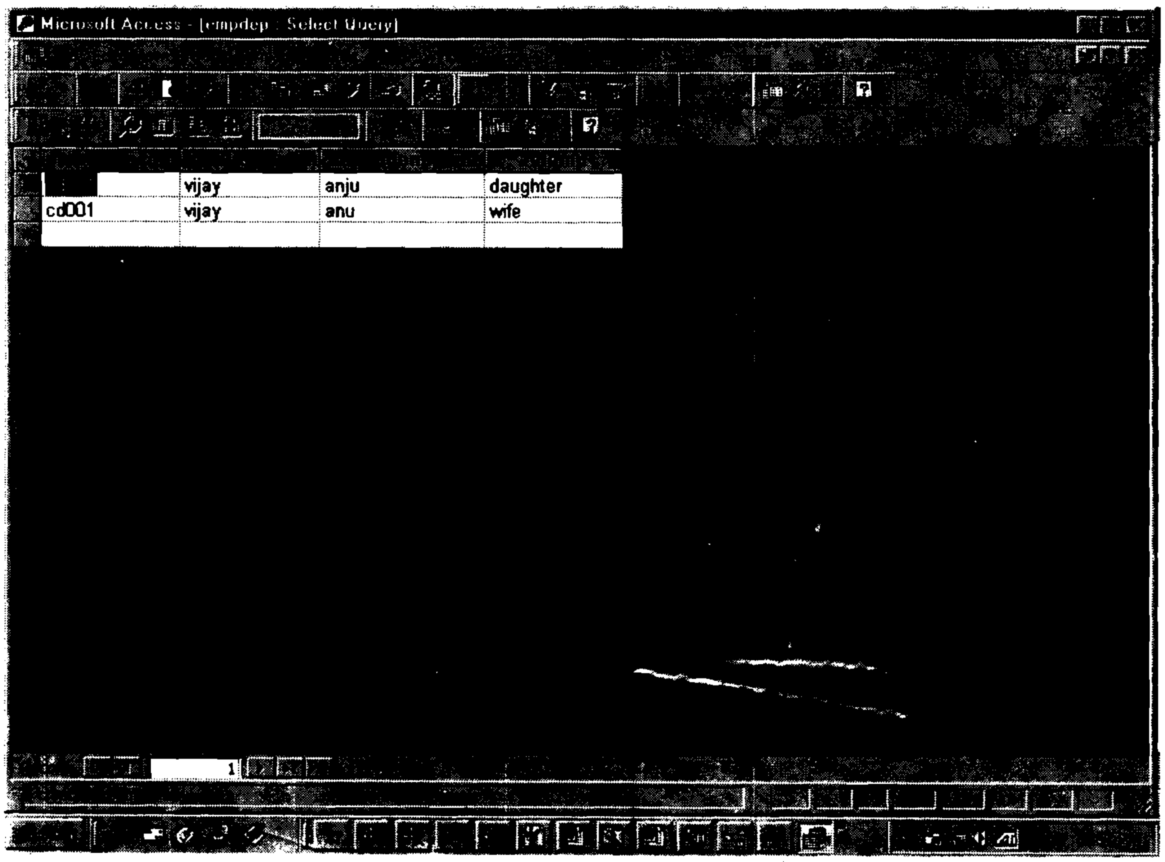
8) Choose the close button.

Saving a Query

1) From the File Menu, choose Save (or click the Save button on the toolbar).

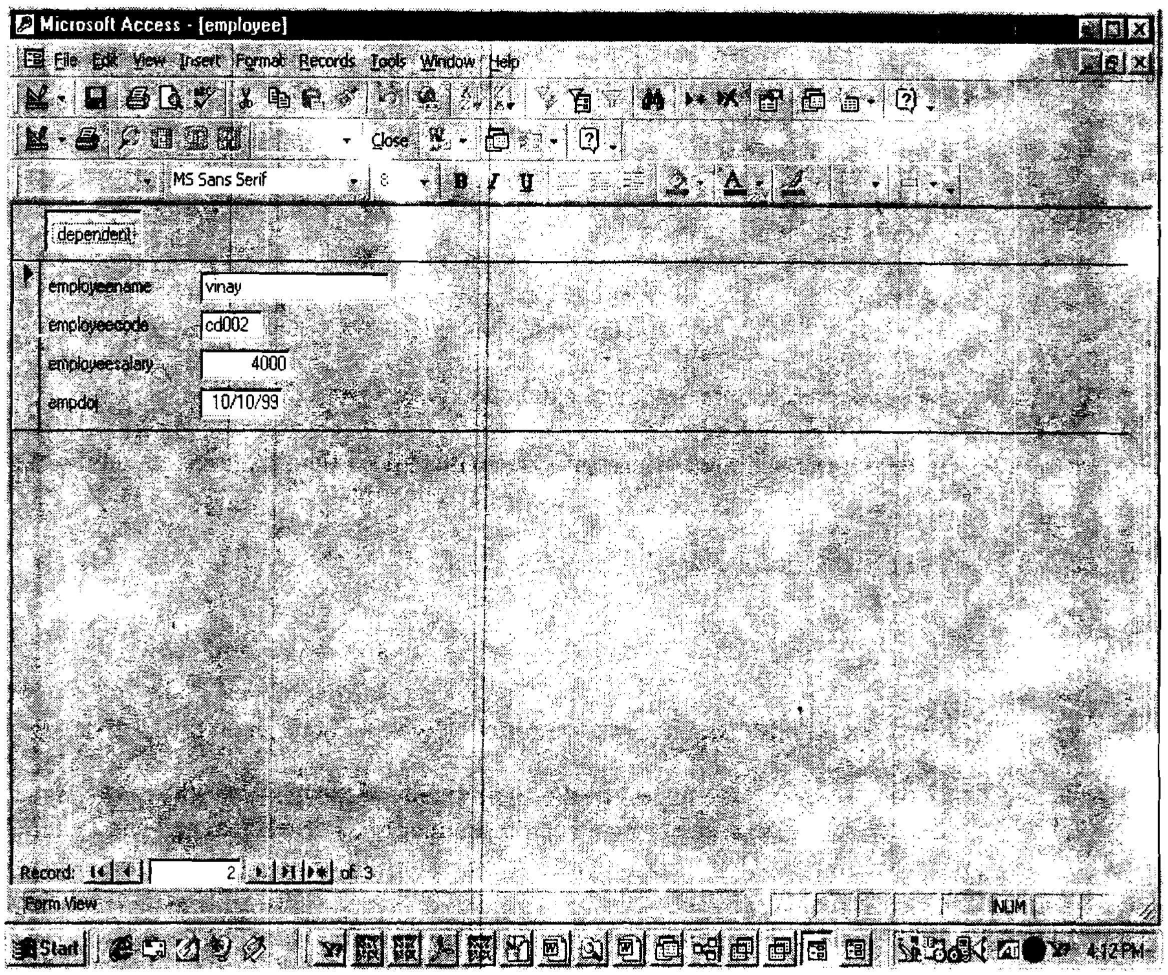
2) If you are saving the query for the first time, type the name for the query, and then choose OK. A query name can contain upto 64 characters and can include spaces.

The above Query should generate the following output:



[illegible]	vijay	anju	daughter
cd001	vijay	anu	wife

Creating a Form


Microsoft Access - [employee]
File Edit View Insert Format Records Tools Window Help
Close
MS Sans Serif
dependent
Record: 2 of 3
Form View
NUM
Start
4:12 PM

employeename	vinay
employeecode	cd002
employeesalary	4000
empdoj	10/10/99

To create a Form without a Form Wizard–

1) In the Database window, click the Form button (or choose Forms from the View menu).

2) Choose the New Button (or choose New from the File menu, and then choose Form).

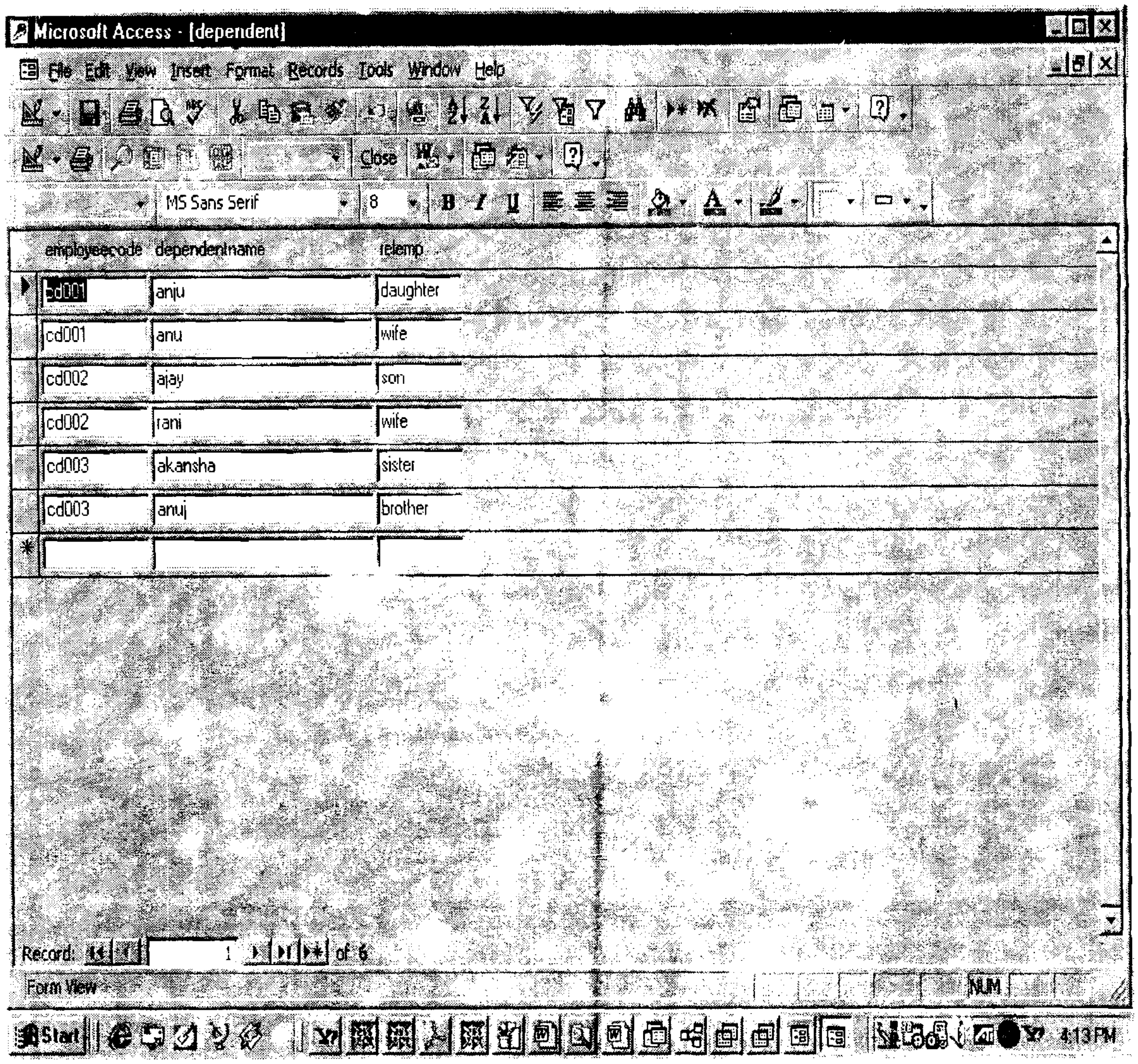
3) Microsoft Access displays the New Form dialog box.

4) From the Select A Table/Query list box, select the table or query that contains the data you want to display on the form. (If the form won't contain data, don't make a selection in this list.)

5) Choose the Blank Form button.

6) Microsoft Access displays the form in Design view.

Here we have created a form employee, please note the button dependent in that form, after entering data of employee, you can enter data of dependents by clicking that button.


Microsoft Access - [dependent]
File Edit View Insert Format Records Tools Window Help
MS Sans Serif
8
Close
Record: 1 of 6
Form View
NUM
Start
4:13 PM

employeecode	dependentname	relemp
cd001	anju	daughter
cd001	anu	wife
cd002	ajay	son
cd002	rani	wife
cd003	akansha	sister
cd003	anuj	brother

Creating a Report

Using a Report Wizard to help you build a report is a fast and foolproof way to create a basic report. You can use the basic report as is or refine it to get exactly the report you want.

To Create a form with the AutoReport Wizard–

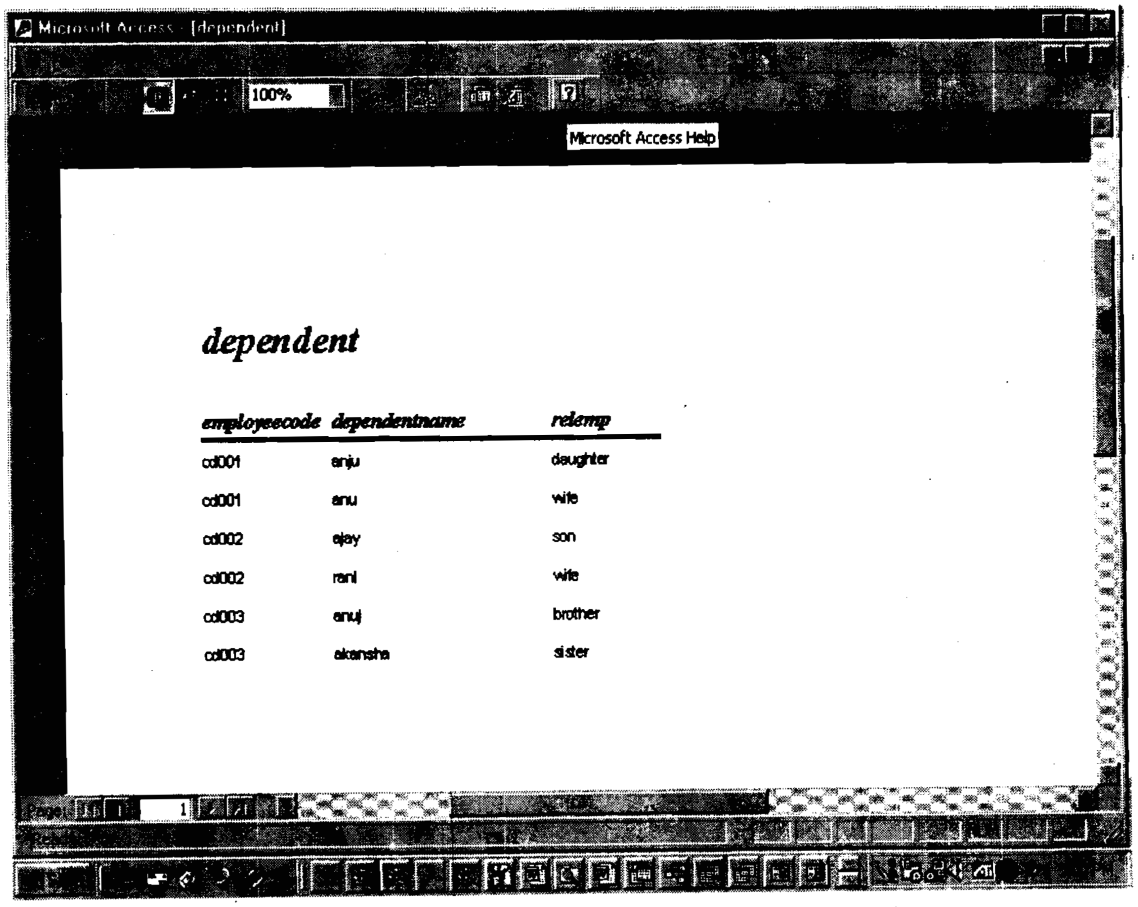
1) In the Database window, click the Table or Query button.

2) Select the Table or Query on which you want to base the

3) Click the Auto Report button on the toolbar.

4) Microsoft Access displays the form.

The Reports generated for the Employee and Dependent tables are:



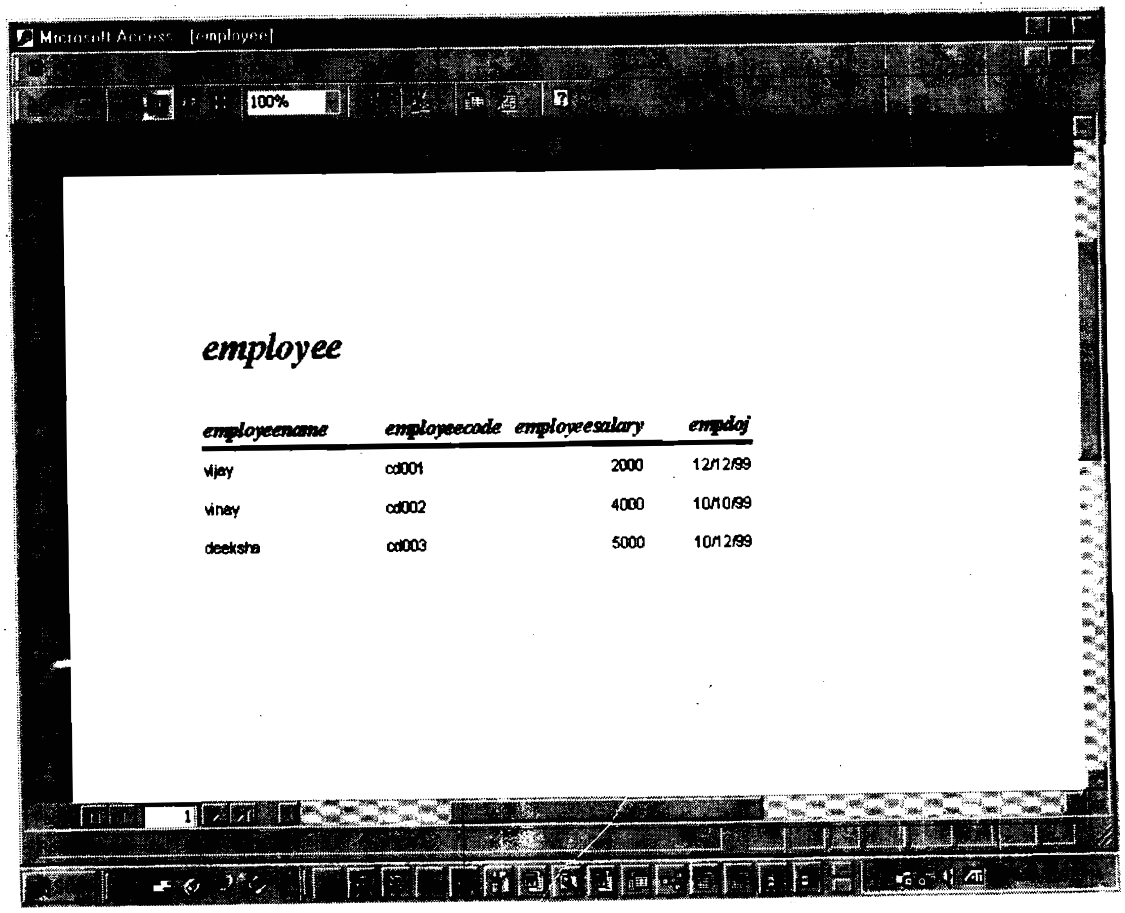
Microsoft Access - [dependent]

100%

Microsoft Access Help

dependent

employeecode	dependentname	relemp
cd001	anju	daughter
cd001	anu	wife
cd002	ajay	son
cd002	rani	wife
cd003	anuj	brother
cd003	akansha	sister



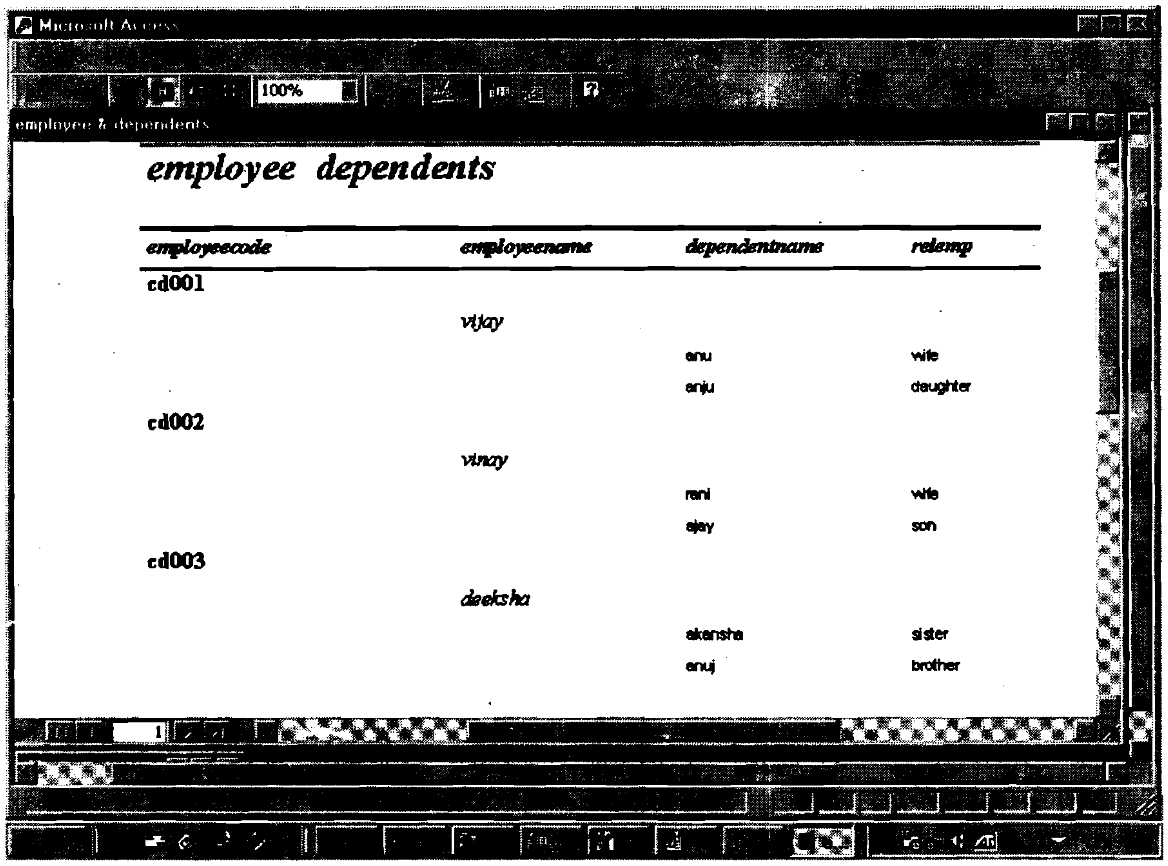
Microsoft Access - [employee]

100%

employee

employeename	employeecode	employeesalary	empdoj
vijay	cd001	2000	12/12/99
vinay	cd002	4000	10/10/99
deeksha	cd003	5000	10/12/99

You can also create a report using more than one table.



employee dependents

```
employeecode      employeename      dependentname     relatnp
ed001
                  vijay
                                    anu               wife
                                    anju              daughter
ed002
                  vinay
                                    rani              wife
                                    ajay              son
ed003
                  deeksha
                                    akansha           sister
                                    anuj              brother
```

Exercise 4

Instructions: You have to perform following exercises in MS-Access.

Question 1. Invoicing is one of the most important activity in Sales management. The selling routine is not considered complete until money due is realised from the customer. To get payment from a customer for goods delivered, the customer will have to be sent an invoice. Once an order has been fulfilled and the goods have been dispatched to the customer, the next step is to prepare and send the invoice. Computerise the whole process.

1) Create the appropriate database and normalised tables. Tables could have following fields:

- Order Number
- Invoice Number
- Invoice Date
- Order Date
- Product Code
- Customer Name
- Customer Address
- Description
- Quantity dispatched

- Price per Unit
- Invoice Amount
- Sales Tax Amount
- Total Amount
- Order Number
- Pending.......................................etc.

2) While creating the structure for the invoice table, the following validations should be taken care of:

   - The invoice number must be unique
   - An invoice can only be generated if the corresponding order number exists
   - The date of the invoice should be current date.

3) Apart from validations, the following tasks should be performed during data entry.

   Data entry from the tables has to be done using Forms, which can be created through the Form wizard.

4) In the form,

   - The invoice amount has to be calculated and displayed.
   - The sales tax at the rate of 8% of the invoice amount needs to be calculated.
   - The total amount due from the customer has to be calculated.

5) Records must be added to the tables after creating a query, linking the various tables (decided by you) on the basis of the order number.

Question 2. 1) Create a new database for any banking Enterprise and name it suitably. The entities you have to consider are following (You can add more entities):

- Employee
- Customer
- Account-Number
- Account Type (Savings, Current etc.)
- Loan
- Payment
- Branch

2) Identify various attributes of the entities above. Identify all the relationships between these entities.

3) First create one single table for the complete banking Enterprise. Identify Key attributes.

4) Normalize the table to various normal forms. In each table identify primary key, foreign keys etc. if possible. Apply both the integrity constraints on the database.

5) Create a suitable form through which we can enter the data for various entities. Associate this form to various tables. More than one form can also be made.

6) At least fill 10 records for each table.

7) Give the following Queries and print the report in a suitable format with suitable Headings for each column:

- Find the names and cities of residence of all employees who work for this Banking Enterprise.
- Find the names, Street Address, and cities of residence of all employees who work for this Banking Enterprise and earn more than Rs. 10,000 per month.
- Find the names of all customers of this banking Enterprise.
- Find the names of the customers who have taken loan from the banking Enterprise and also print the loan amount in the report.
- For the customers who have taken loan, how much amount of loan has been paid back and how much amount is pending.

8) On last day of the month add 2% interest on the minimum amount present in the account after the date 10th of that month. Generate this statistics by using in-built mathematical functions present in MS-Access, e.g.. Count, sum, min etc.... Record this information as one of the field in suitable table.

9) Generate a report in the format of passbook with following fields.

Date	**Mode (Cheque, cash etc.)**	**Interest to date**	**Amount debited**	**Amount credited**	**Balance to date**

10) Taking this database as the backend, create a Graphical User Interface on some other language like Microsoft Visual Basic or PowerBuilder etc., hence exploit ODBC.

11) Add various constraints like,

- A person cannot have more than two accounts in this banking enterprise.
- A person cannot apply for the second loan until he has already paid the first loan completely etc.

Question 3. There is a way in MS-Access through which we can run the SQL. Find it and give all the queries in SQL section in MS-Access and find the results of various queries.

Question 4. Design and implement the database for a University in MS-Access, keeping following in mind.

1) The entities you have to consider are following (You can add more entities):

- Student
- Course
- Registration
- Instructors
- Result
- Scholarship

2) Identify various attributes of the entities above. Identify all the relationships between these entities.

3) First create one single table for the University. Identify Key attributes.

4) Normalize the table to various normal forms. In each table identify primary key, foreign keys etc. if possible. Apply both the integrity constraints on the database.

5) Create a suitable form through which we can enter the data for various entities. Associate this form to various tables. More than one form can also be made.

6) At least fill 20 records for each table.

7) Give the following Queries and print the report in a suitable format with suitable Headings for each column.

- Find the various courses available in University.
- Find the minimum percentage required to enter in the courses like Computer Science, Mathematics, and Physics etc.

- Find the number of students registered in the course Computer course etc.
- Find the toppers of all courses in year 1998.
- Find various Instructors for various Courses.
- Find the Number of students in the third year of course Computer Science in year 1999.
- Find names of students getting Scholarship, Find their marks also, and year of registration, etc.

8) Generate a report in the following format. (You can add more fields)

Name of Student	**Year of Registration**	**Courses taken**	**Instructors**	**Marks in various subject he has taken**	**Position**

9) Taking this database as the backend, create a Graphical User Interface on some other language like Microsoft Visual Basic or PowerBuilder etc., hence exploit ODBC.

10) Add various constraints like,

- A student has to select three courses.
- If students fail in one or more course then he cannot be promoted to next year, thus it is mandatory to clear all the courses before entering next semester or year etc.

Note: For the following questions do proper problem analysis. Draw dataflow diagrams and/or E-R diagrams. Design suitable forms having proper validation checks, reports and queries. You may use menus (if needed). Implement the above in MS-Access. You may also use a front-end (may be Visual Basic) if you need.

Question 5. Design and implement database for a Library in MS-Access.

Question 6. Design and implement database for the hospital in MS-Access.

Question 7. Design and implement database for a Shop (say Electronics Shop) in MS-Access. You can select the any type of shop you want.

1.7 VIEWS AND SECURITY USING SQL

Question 1. Imagine you work for Sandwich shops, and each of the stores in Delhi report to you all their data ... sandwiches sold, of each type, money made that day, supplies used, which supplies are unusually low (each store gets a certain amount of each supply ... Bread, Butter etc. each day, but they may need more the next day if they used a lot of it that day), etc. Each store manager can insert and change data about their store, but shouldn't be able to read data about the other stores. The person in charge of deliveries should be able to look at all the stores to see which supplies need to be delivered. The accountant could look at anything, but is only interested in the money taken in and spent (to buy supplies, for salaries, etc).

1) The Shop just hired you as their data administrator. Do you want to put this in a database system or not? Give reasons. Feel free to make up additional details, which would be likely to occur in a retail store situation like this.

....

....

....

....

2) Let us suppose that you decide to use the database DBMS. How many users views would you create at the external level, and why? (The reason to create a user view is either security or ease of use.)

....

....

....

3) The data dictionary contains at least all the relations and views that are in your database; and which users can access each view. State a few reasons why a dictionary is useful to the database system, and a few reasons why it is useful to you as the data administrator.

....

....

....

....

Question 2. Consider the following relations:

1) **Staff** (sno, fname, name, tel#, position, sex, salary, bno) where Staff numbers (sno) are unique and each staff member has one first name, last name, position, sex, salary, and works for one branch. A staff member may have several telephone numbers that you can reach them at.

2) **Branch** (bno, street, city, tel#, fax#) where Branch numbers (bno) are unique, and each branch is at one place with one main tel# and one fax#.

3) **Property_for_rent** (pno, street, city, type, rooms, rent, sno, bno) where Property numbers (pno) are unique, each property is in one place and is of a particular type (apt, house, duplex, etc), with a certain number of rooms, is listed by one branch and one staff member.

4) **Renter** (rno, fname, lname, street, city, tel#, pref_type, max_rent,bno). What semantics will you use if you say that the primary key of renter is rno, pref_type?

5) **Viewing** (rno, pno, date, comment) where a person looks at a particular property (say room or apartment etc.) on a particular date and might or might not make a comment.

   a) Give the semantics; write down the candidate keys for the above relations. If you think that the given information is not enough then write down any additional assumptions that you need to make. Although the candidate keys are given with the relations, but, write down the real world assumptions to make that a candidate key.

   ....

   ....

   ....

   ....

   b) If there were 1000 different renters, 5000 different properties, 365 dates (lets just consider1 year), and 20 different possible comments (things like "too small", "ugly", "wrong part of town", etc.) what would be the maximum cardinality of viewing? What would be a more realistic maximum estimate, and how would you determine that?

   ....

   ....

   ....

   ....

   c) Write the following queries using SQL for the above relations (You need to attempt this question yourself based on section 6).

      (i) Find those property pno, which are 6 room houses and are in the same city as the branch that listed the property.

      (ii) Find the renters last names who live in Delhi and viewed a property located in Delhi and the staff member that showed the property had a last name of Sharma.

      (iii) Find the property pno that is in Delhi, but no one that viewed the property lives in Delhi.

      (iv) Find those property pno and streets of 6 room houses that are in Delhi and everyone that viewed the property lives in Delhi.

(v) Find those property pno and streets of 6 room houses that are in Delhi and every renter living in Delhi has viewed the property. (What is the difference in the answers between (iv) and (v)? Figure that out first before trying to write the query.) Any logically correct answer for these is fine.

....

....

....

....

....

....

....

....

....

....

Question 3. Write SQL queries for the following English queries. Please change the field names yourself such that they do not conflict with key words if needed.

a) Find those renters who looked at properties in Delhi, where their max_rent is larger than the average apt rent in Delhi. Return their last name, their rno, and the difference between their max_rent and the average apt rent in Delhi.

....

....

....

b) Find those renters last names that viewed all the 3 room properties for rent in Delhi. (**Hint:** This is division.)

....

....

c) Find those properties pno and addresses that were viewed in January by all the renters living in Delhi whose max_rent was < 600. (**Hint:** This is division.)

....

....

d) Create a view that produces the average apt rent per city. It should return the average rent and the city. Show that it works.

....

....

e) Use the view created in d) that returns the pno of the apartments whose rent is exactly the average apartment rent for the city it is in.

……………………………………………………………………………………

……………………………………………………………………………………

f) You could have directly substituted the code for the view into your query for e), and avoid creating the view. When do you want to actually make it a view?

……………………………………………………………………………………

……………………………………………………………………………………

g) Give one of your friends, permission to insert into and read your "property_for_rent" table. Also see if you can give them permission to insert into and read your view you created in part c). Attempt to read, insert into, and delete from your friend's table, and read and insert into your friend's view. Explain what happens.

……………………………………………………………………………………

……………………………………………………………………………………

h) Create a view with all the attributes of "property_for_rent" except the "rent" field. Try to insert into that view. Try to delete from that view. Explain what happens.

……………………………………………………………………………………

……………………………………………………………………………………

i) Explain how, if you wanted to implement security issues, you could allow certain staff members access to only the properties in Delhi. (It is sufficient to write this in English, you don't have to code it up in SQL).

……………………………………………………………………………………

……………………………………………………………………………………

Model Answers

Answer 1.

1) I would like to put the shop data in a DBMS. In order to arrive at this decision, here are the assumptions I made, the disadvantages of file-based approach I considered and the advantages of DBMS I appreciated.

<u>Assumptions:</u>

(i) Delivery person takes money from the accountant everyday and spends to buy supplies.

(ii) Salaries are managed by store manager.

<u>Data:</u>

(i) Raw supplies received on a day (bread, cheese, sandwich bread etc.)

(ii) Supplies used

(iii) No of sandwiches of each type sold (vegetable sandwich, cheese sandwich etc.)

(iv) Money made on a day

(v) Which supplies are low and how much extra of each item is needed

(vi) Salaries given out that day

(vii) Money taken/spent for buying supply items

(viii) Total money made on a day by selling sandwiches

(ix) Total money given away as salaries.

Let's identify store manager as P1, delivery person as P2 and accountant as P3. Table 1 gives the data (listed above) that each one is concerned about. The numbers 1, 2, 3, 4, 5, 6 etc. denote the data by there respective numbers above e.g.. Raw supplies are denoted by **1**, Supplies used is denoted by **2** etc.

Table 1.

Person	**Concerned Data**
P1	1,2,3,4,5,6
P2	1,2,5,7
P3	1,2,3,4,5,6,7,8,9. Mostly concerned about 4, 6, 7, 8, 9.

So, if we take the file system based approach and have separate files (containing the Concerned Data) for each instance of P1 and each of P2 and P3, we have the following disadvantages:

(i) **Data Duplication:** It's very evident from the above table that there would be a lot of data duplication across the files.

(ii) **Separation of data:** Each store manager has a set of data. The delivery person needs an access to the data of all the stores in order to decide on how much of each item he needs to buy and supply to each store that day. For this he needs access to the individual files. Data is not isolated, since it is all reachable without extensive effort.

(iii) **Data dependence:** Items 1, 3, and 5 are related, as are items 6 and 9. In order for the values to be correct, updates to one field need to check that the other values are still correct.

(iv) **Incompatible file formats:** Depending on who creates the files, incompatible file formats can complicate how to maintain and update these files. If one person created them all, we don't have a problem.

Consider a DBMS based approach and store data in the form shown in Table 2 as follows:

Table 2.

Person	**Concerned Data**
P1	store Id, 1, 2, 3, 4, 5, 6
P3	All store Ids, 7, 8, 9

The DBMS approach shall have the following advantages–

(i) **Controlled redundancy:** Store ids are the only replicated information.

(ii) Since store ids are unlikely to change, we don't have problems in updating. Items 6 and 9 are in separate relations, but the database can keep those correct automatically by the DBA setting it up properly– another advantage of using the database system.

(iii) Could give more information from same amount of data, item 5 might be omitted by setting up an automatic query using information from items 1 and 3.

(iv) **Enables sharing of data:** Each manager is able to look at his individual store, but the accountant can have as much information as desired. Database systems have more flexible sharing abilities.

(v) **Improved data integrity:** For example, a typical integrity constraint can be that the money given out on a day should be strictly less than the money made on that day. Another constraint can be that the money made on a day at a store is consistent with the number of sandwiches sold.

(vi) **Improved security:** The manager of a store will not be allowed to view data about other stores. DBMS approach supports the concept of views to support such security.

(vii) **Backup and/Recovery:** Knowing how much of each supply was already sent out and knowing the salary history is something that we would like to keep even in case of machine failure.

(viii) The other advantages are less obvious, if applicable at all; maybe improved accessibility if the accountant wants to do other kind of monitoring, it might be easier to write queries if the info were in a DBMS, but concurrency and economy of scale aren't really applicable, since there isn't a whole lot of traffic going on anyway, and we don't have huge volumes of data.

Hence a DBMS based approach is suitable for storing the subway data.

2) The situation requires several views:

- One view for <u>store manager</u> - This view is to ensure security. No store manager will have access to information about other stores.

- One view for <u>delivery person</u> - This view will have access to the data 1,2,5 of all stores based on store Id and 7. This view ensures security min that it doesn't allow the delivery person to view data 8,9. The view also ensures ease of use by providing only required data (centrally) to the delivery person.

- One view for accountant - This view gives access to the complete data. But, in order to support ease of use, it will also have an option of filtering out some information and show data 4,6,7,8,9.

3) The data dictionary (also called system catalog) for the SHOP database contains:

- names, types and sizes of data items (The data items include the data that has been listed in part (a))
- Integrity constraints (total money made, money spent etc.)
- Names of authorized users and their views and authentication information (store managers, accountant, delivery person)

The information stored in the system catalog helps the DBMS in the following ways:

(i) Knowing the data types, names of data, and location

(ii) Checking authorization when a user accesses data.

(iii) Data integrity is taken care of internally (makes sure updates keep data consistent).

The information stored in the system catalog helps the Data Administrator in the following ways:

(i) Knowing who has what authorization for which views

(ii) Knowing the structure and organization of the DB when extending or reorganizing it.

(iii) More easily checking errors by referring to this automatic documentation to see what could possibly be set up incorrectly, etc.

Answer 2. The given relations are:

Staff (sno, fname, lname, tel #, position, sex, salary, bno)

branch (bno, street, city, tel #, fax #)

property_for_rent (pno, street, city, type, rooms, rent, sno, bno)

renter (rno, fname, lname, street, city, tel #, pref_type, max_rent, bno)

viewing (rno, pno, date, comment)

Semantics for the renter:

Renter numbers are unique to a renter. Each renter may have any number of preferences. Each renter has only one first name, last name, max_rent, tel#, branch he approached, and address.

a) Candidate keys:

(i) staff

- sno,tel#. If you assumed a tel# goes to only one staff member, then the candidate key would be just tel#

(ii) branch

- bno. there's a unique branch number for each branch.
- tel #. Assuming a phone# goes to only one branch.
- fax #. Same assumptions as on tel#
- street,city. If there's only one branch per address.

(iii)property_for_rent

- pno. There's a unique number for each property.
- street,city. Only one property per address. This may not be true, depending on what is included in the address.

(iv)renter

- rno, pref_type

The assumptions made for making the above attributes a primary key are:

- More than one renter can have same names, addresses, branch numbers, max_rent specifications and/or a combination of these.

- Renter has many preferences on the type of property.

(v) viewing

- rno, pno

One renter can view many properties and one property can be viewed by many renters. Assuming that one renter views one property only once, a combination of rno and pno forms a primary key for viewing.

If a renter can make several different comments about the same property, add it in to the key, but then you have to not allow null in the comment field, so you would have to add a legal value like "none".

b) • In the worst case, if we assume that each renter sees each property each day and makes all possible comments, then the cardinality will be–

no. of renters * no. of properties * no. of dates * no. of comments = 1000*5000*365*20 = 36,500,000,000.

• Using my candidate key above, we get–

no. of renters * no. of properties = 1000 * 5000 = 5,000,000

And even this is big. A person probably won't look at more than 100 properties maximum, so a more realistic estimate would be 100,000. (1000 renters time 100 properties each).

Answer 3.

```
(a) SELECT lname, rno, max_rent – a.avgrent
    FROM (SELECT avg(rent) AS avgrent
              FROM property_for_rent
              WHERE city = 'Delhi' AND type = 'apt' )     a,
              (SELECT lname, r.rno, max_rent
               FROM renter r, viewing v, property_for_rent p
                 WHERE r.rno=v.rno and v.pno=p.pno AND p.city='Delhi'
                 GROUP BY lname, r.rno, max_rent)
    WHERE max_rent > a.avgrent;
```

Alternate to above solution is:

```
SELECT r.lname, r.rno, max_rent-a.avgrent
FROM (SELECT avg(rent) as avgrent
         FROM property_for_rent
         WHERE city='Delhi' AND type = 'apt' )  a, renter r, viewing v,
                                                   property_for_rent p
 WHERE r.rno=v.rno AND v.pno=p.pno AND p.city='Delhi' AND r.max_rent
                                                   > a.avgrent;
```

```
(b) SELECT lname
    FROM renter
    WHERE rno IN (SELECT rno
                  FROM viewing v
                  WHERE pno IN (SELECT v.pno
                                FROM viewing v, property_for_rent p
                                WHERE p.pno = v.pno
                                AND rooms = 3 AND city = 'Delhi')
                  GROUP BY rno
                  HAVING count(distinct pno) = some ( SELECT
                                                 count(distinct v.pno)
                          FROM viewing v, property_for_rent p
                          WHERE p.pno = v.pno AND rooms= 3 AND city
                                                 = 'Delhi'));
```

```
(c) SELECT pno, street
    FROM property_for_rent
    WHERE pno in (SELECT pno
                  FROM viewing v
                  WHERE date = 'Jan' AND rno IN (SELECT r.rno
                                FROM renter r, viewing v
                                WHERE r.max_rent < 600 AND city =
                                                  'Delhi')
                  GROUP BY pno
                  HAVING count(distinct rno) = some (SELECT count(distinct
                                                   r.rno)
                                FROM renter r, viewing v
                                WHERE r.max_rent < 600 AND city =
                                                 'Delhi'));
```

```
(d) CREATE VIEW average_rent_city AS
        (SELECT avg(rent) AS avg_rent, city
          FROM property_for_rent
            WHERE type = 'apt'
              GROUP BY city);
```

(e) SELECT pno
FROM average_rent_city a, property_for_rent p
WHERE p.rent = a.avg_rent AND p.city = a.city;

(f) Views are created only when a certain set of data are to be used often, if we just wanted the relation for that query, we would have just defined it in the from statement itself (as we did in part a) of this question).

(g) GRANT SELECT, INSERT ON property_for_rent TO Ravi (say); This Grant will be succeessful.

GRANT SELECT, INSERT ON average_rent_city TO Ravi; In this case also Grant will be succeessful.

SELECT * FROM Ravi.property_for_rent; Here also respective rows will be selected.

INSERT INTO Ravi.property_for_rent values ('pd2','Khel gaon marg','Delhi','apt', 4,400,'s21','b21'); 1 row will be created.

INSERT INTO Ravi.average_rent_city VALUES (1200,'New York'); Here we will get error as data manipulation operation not legal on this view.

DELETE FROM Ravi.property_for_rent WHERE pno='pd2'; ERROR table or view does not exist as Ravi can insert and read property_for_rent since the permissions were given to Ravi as you, but he cant delete from it because that permission wasn't given.

You can read the view, but you can't insert into it because it makes no sense to do so, given an average rent for Bombay, you cant insert any meaningful tuples in the relation the view is based on.

(h) CREATE VIEW pfrrent AS (SELECT pno, street, city, type, rooms, sno, bno FROM property_for_rent); View gets created.

INSERT INTO pfrrent VALUES ('pf1', NULL, NULL, NULL, 3, NULL, 2 NULL); 1 row is created.

DELETE FROM pfrrent WHERE pno = 'pf1'; 1 row deleted. You can do this since this view is based on only 1 relation, and the missing "rent" field is just filled with null (as are a number of other fields that we actually put NULL in the tuple we inserted).

(i) Please attempt this question by creating a view having a condition.

Exercise 5

Please design views for the problems given in the previous exercise keeping in mind the various users and their expected security levels. Write SQL commands for creation of views and granting and/or revoking of access rights.

1.8 SUMMARY

In this block, we have discussed about various types of problems relating to relational database design and implementation. We have also presented some exercises, which you must try to attempt during your lab hours. Please make sure that before you go to the lab, you are ready with the basic design of your database, queries, forms (having proper validation checks) and reports. The problems presented in the block are just the sample problems, however, you must try attempting additional problems, specially related to the concepts of transactions and web interfaces using a front-end.

The basic topics which are covered in the block include use of ER diagrams, functional dependency and its use for normalisation, a practical example of normalisation, use of SQL for query, you must do more exercises on the topic as SQL is one of the standard platform for all the Commercial Database management systems. One of the prime issues of the block is to implement the designed database into an RDBMS package. Presently you will be using MS-Access for such implementation.

One of the questions you must be asking is why we have selected MS-Access and its version? Well the answer is, it is a package, which helps you as a ladder for higher commercial DBMSs. Please note the issue is not the version or a commercial RDBMS in which you implement the design. The main issue is whether you are adaptable to do so. MS-Access version, which you may be using, may be different, but you must try to understand the design and try to implement using your version.

Finally in the block, we have presented views and security issues using SQL. You must try to find out details of security features of RDBMS package, which you are using.