October, 2007

Database Systems Engineer Examination (Afternoon, Part 1)

Questions must be answered in accordance with the following:

Question Nos.	**Q1 – Q4**
Question Selection	**Choose 3 questions from the 4 questions**
Examination Time	**12:10 - 13:40 (90 minutes)**

Instructions:

1. **Choose 3 questions** from the 4 questions, and **encircle the question numbers you chose** as seen in the example below. Please note that the answers are not scored if you do not encircle any of the question numbers. When all the 4 questions are encircled, the answers of the first 3 questions will be scored.

Select three
(Q1)
Q2
(Q3)
(Q4)

[An example when Q1, Q3, and Q4 are chosen]

2. Use a pencil to write your answers on the answer sheet. When you need to change an answer, erase your previous answer completely and neatly. Wipe away any eraser debris.
3. Mark your examinee information and test answers in accordance with the instructions below. Your test will not be graded if you do not mark properly. Do not mark or write on the answer sheet outside of the prescribed places.
   (1) **Examinee Number**
   Write your examinee number in the space provided, and mark the appropriate space below each digit.
   (2) **Date of Birth**
   Write your date of birth (in numbers) exactly as it is printed on your examination admission card, and mark the appropriate space below each digit.
   (3) **Answers**
   Write each answer in the space specified for that question.
   Write your answers clearly and neatly. Answers that are difficult to read will receive a lower score.

Do not open the exam booklet until instructed to do so.
Inquiries about the exam questions will not be answered.

Notation Used in the Questions

The notation for conceptual data models, relation schemas, and relational database table structures is given below. This notation applies unless otherwise noted in the text of a question.

1. Notation for Conceptual Data Models

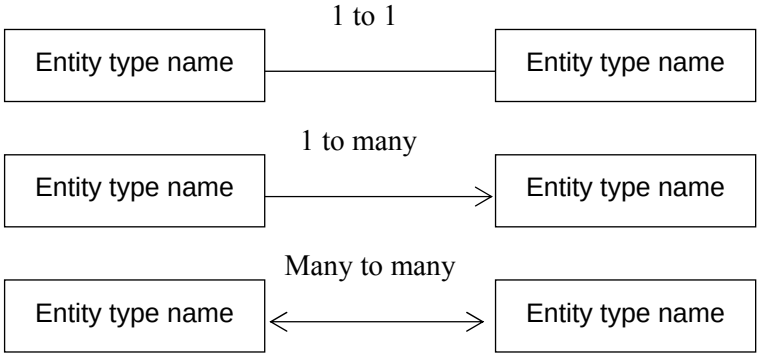


Fig. 1　Notation for Entity Types and Relationships

(1) Entity types are indicated using rectangles.
(2) The entity type name is written inside the rectangle.
(3) The relationship between entity types is indicated using a line.
(4) For a "1-to-1 relationship," neither end of the line is an arrow.
For a "1-to-many relationship," one end of the line is an arrow.
For a "many-to-many relationship," both ends of the line are arrows.

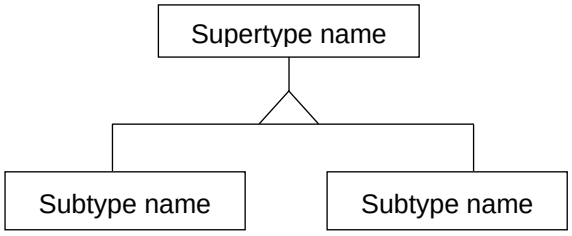


Fig. 2　Notation for Supertypes and Subtypes

(5) When representing supertypes and subtypes, lines are drawn between the supertype and the subtypes, and a "△" is used at the branch point.

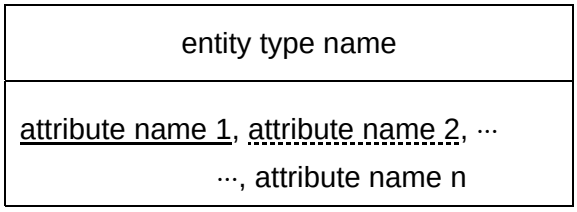


Fig. 3　Notation for the Attributes of Entity Types

(6) When representing the attributes of an entity type, the rectangle is divided into two sections, upper and lower. The entity name is written in the upper section, while the attribute names are listed in the lower section.

(7) When representing a primary key, a solid underline is used for the attribute name or group of attribute names that make up the primary key.

(8) When representing a foreign key, a dotted underline is used for the attribute name or group of attribute names that make up the foreign key. Note, however, that a dotted underline is not used when some of the attributes that make up the primary key are used to make up the foreign key.

2. Notation for Relation Schemas

relation name (attribute name 1, attribute name 2, ···, attribute name n)

Fig. 4　Notation for Relation Schemas

(1) A relation is represented by a relation name and a list of attribute names surrounded by parentheses to the right of the relation name. This is called a relation schema.

(2) When representing a primary key, a solid underline is used for the attribute name or group of attribute names that make up the primary key.

(3) When representing a foreign key, a dotted underline is used for the attribute name or group of attribute names that make up the foreign key. Note, however, that a dotted underline is not used when some of the attributes that make up the primary key are used to make up the foreign key.

3. Notation for Relational Database Table Structures

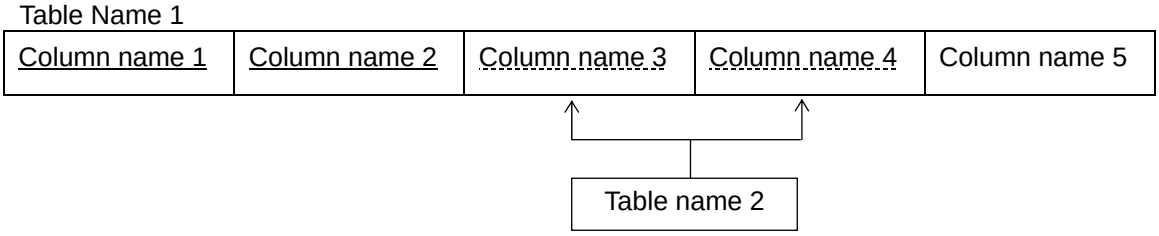


Fig. 5 Notation for Table Structures, Primary Keys, Foreign Keys and Reference Relationships

(1) A table name is entered followed underneath by the column names that make up the table. Each column name is written inside a rectangle.

(2) When representing a primary key, a solid underline is used for the column name or group of column names that make up the primary key.

(3) When representing a foreign key, a dotted underline is used for the column name or group of column names that make up the foreign key. Note, however, that a dotted underline is not used when some of the attributes that make up the primary key are used to make up the foreign key.

(4) When representing a table to be referenced by the foreign key, a line is drawn either above or below the column name or group of column names that make up the foreign key. A rectangle is drawn at the end and the name of the table to be referenced is entered inside. The end of the line on the foreign key side is an arrow.

Q1. Read the following description of the basic theory of databases, and then answer Subquestions 1 and 2.

In order to create a database to manage the book sales information of an online store, we discussed the relational schema of the data model.

[Relational Schema of the Book Sales Information]

Figure 1 shows the relational schema of the book sales information of this online store. The relation "sales record" is a relation concerning the number of books sold per period. Table 1 shows the meanings and constraints of the attributes. Figure 3 shows the main functional dependences between the attributes using the notation shown in Figure 2.

Book (book number, book title, author list, type, binding, publishing editor, publisher, publishing date, edition, price, overview, days required for shipping)
Order (order number, book number, quantity, customer number, shipping address, payment method, order date, shipping date)
Customer (customer number, customer name, address, contact information, email address)
Sales record (period, book number, quantity sold)

Fig. 1 Relational Schema of the book sales information of the online store

Table 1 Meanings and Constraints of the Attributes (abbreviated in part)

Attribute	Meaning and Constraints
Book Number	The number that uniquely identifies the book. Books with identical content could have different publishers and types, so each is given a different book number.
Author List	Character string that handles the list of authors as one value. The order in which they are listed indicates the author ranks, e.g., the first author, second author, etc.
Type	Classes such as paperback and hardcover, size such as A5 and B5.
Binding	Type of binding and the number of pages
Publishing Editor	Person in charge at the publishing company. It's the same person for the same printing, but it could be different for different printing versions.
Publisher	Publishing company
Publishing Date	Publishing date and year, given for each printing
Edition	Edition number and the printing number of the book
Days Required for Shipping	Number of days required until the book can be shipped
Order Number	To each order, at least one order number is assigned. For each order number, at least one copy of the same book can be ordered.
Customer Number	Number that uniquely identifies the registered customer.
Shipping Address	For each order, only one shipping address can be designated.
Payment Method	For each order, the customer chooses either "credit payment" or "payment on arrival."
Period	Tally period for sales record
Quantity Sold	Total quantity sold for each book during a period

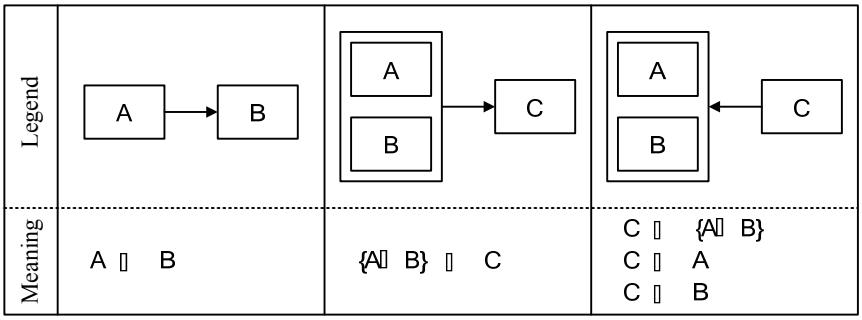


Fig. 2 Notation for Functional Dependences

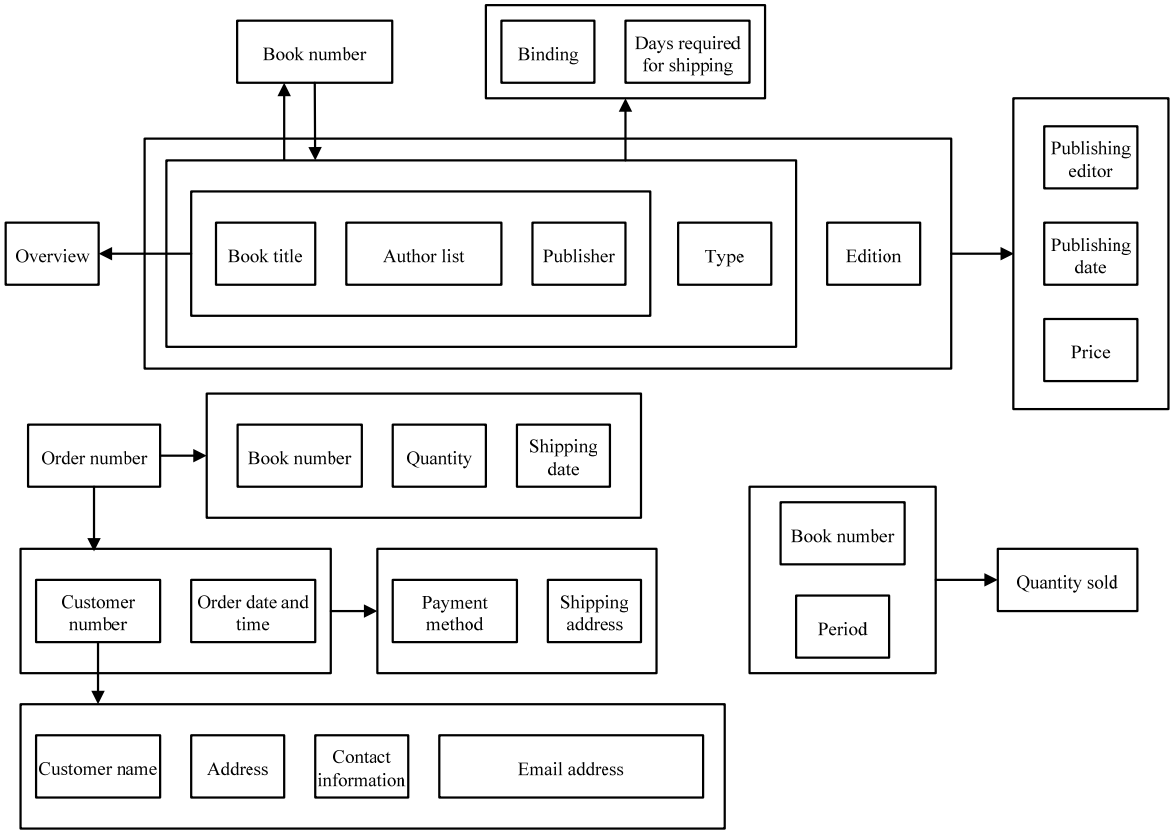


Fig. 3 Main Functional Dependences among Attributes

[Relational model and object-oriented model]

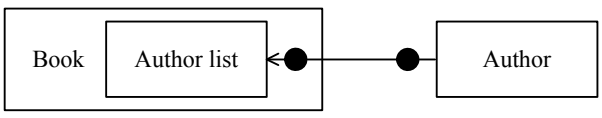
The way in which information concerning the authors in the relation "book" was reviewed, and the following methods (i) through (iii) were considered. Table 2 explains methods (i) through (iii) by giving specific examples using identical values (partially incomplete).

Method (i)

Replace the attribute "author list" in the relation "book" with the relation "author list," and add a new relation "author."

- Book (book number, book title, author list (author order, author number), type, …)
- Author (author number, author name, remarks)

These relations cannot be accurately expressed using the notation rules for this conceptual data model as shown above. For example, expressions like the following diagram would be necessary.

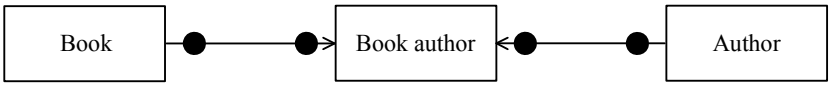


Method (ii)

Delete the attribute "author list" from Book (book number, book title, author list, type, …), and create the following two relations: "book author" and "author."

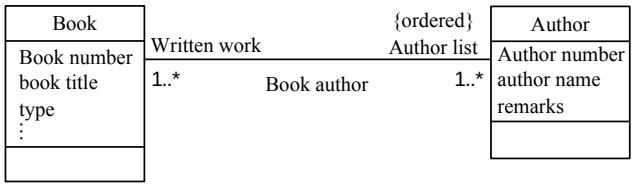
- Book author (book number, author order, author number)
- Author (author number, author name, remarks)

Using the notation rules for this conceptual data model as shown above, these relations can be expressed in the following diagram.



Method (iii)

In the object-oriented model, consider the information concerning authors as the class "book," the class "author," and the relationship between them. Using the notation of UML (Unified Modeling Language), this can be expressed as follows:



Note: "1..*" denotes a multiplicity "at least 1," and "{ordered}" means that the "author list" has the order constraint. The meaning of the UML notation is as follows:

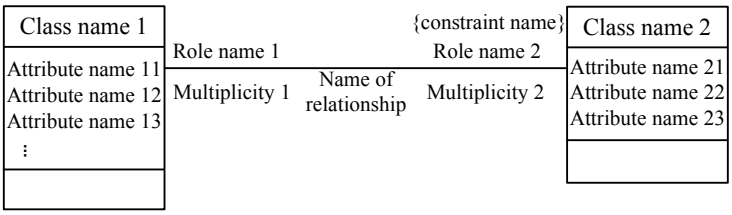


Table 2　Specific Examples of Differences among Methods (i) through (iii)

Method | **Instance**

Method □

Book

Book number	Book title	Author list	Type	...
B001	ABC	Author order / Author number: 1 / N001; 2 / N002	□ □	...
B002	EFG	Author order / Author number: 1 / N001; 2 / N003	□ □	...
...	...	...	...	...

Author

Author number	Author name	Remarks
N001	X	• • • •
N002	Y	□ □ □ □
N003	Z	× × × ×
...	...	...

Method □

Book

Book number	Book title	Type	...
		□ □	...
		□ □	...
...	...	...	...

Author

Author number	Author name	Remarks
N001	X	• • • •
N002	Y	□ □ □ □
N003	Z	× × × ×
...	...	...

book author

Book number	Author order	Author number
...	...	...

Method □

Book object

Identification number	01
Book number	B001
Book title	
Author list	
Type	□ □
...	...

Identification number	02
Book number	B002
Book title	
Author list	□03□ 05□
Type	□ □
...	...

Note: Object identification numbers n1, n2, n3, □c express the link destinations {n1, n2, n3, □c}. If the order here has a meaning, the leftmost item indicates the first object.

Author object

Identification number	03
Author number	N001
Written work	
Author name	
Remarks	• • • •

Identification number	04
Author number	N002
Written work	□01□
Author name	
Remarks	□ □ □ □

Identification number	05
Author number	N003
Written work	
Author name	Z
Remarks	× × × ×

Subquestion 1

Answer (1) through (4) concerning the relational schema of Figure 1.

(1) List all the candidate keys for the relations "book," "order," and "sales record."
(2) For each of the relations "book," "order," and "sales record," if there is any transitional functional dependence from a candidate key, write one example; if not, write "none."
(3) For each of the relations "book," "order," and "sales record," if there is any relation that is partially functionally dependent to a candidate key, write one example of the functional dependence; if not, write "none."
(4) Which normal form (the degree, or order, of normalization) are each of the relations "book," "order," and "sales record"? Choose the most appropriate normal form. Further, state the reason briefly that the relation "sales record" can be considered a normal form.

Subquestion 2

Answer (1) and (2) concerning Methods (i) through (iii).

(1) Insert the correct term or phrase in each of the thick blanks in Table 2.
(2) If the information on the relation "author" is expressed by Method (i), the relation "book" will no longer satisfy the conditions of the first normal form. Explain briefly the reason for this.

Q2. Read the following description of the database design for an order entry and management system, then answer Subquestions 1 and 2.

Company E is in the health food mail-order business. The company decided to develop an order entry and management system, and assigned the task of designing the database to Mr. F.

[Required Specifications]

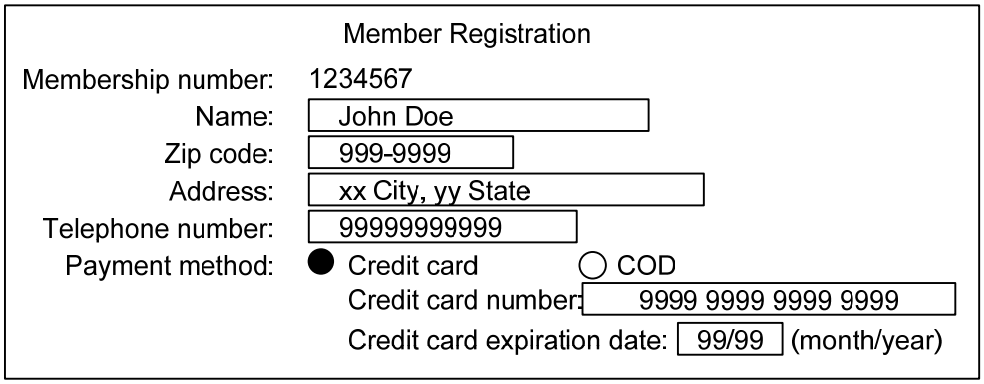
Required specifications of the order receiving and management system are as follows.

1. Member Registration

(1) Customers must register and become a member before ordering products.

(2) Based on the member registration form received from the customer either via postal mail or fax, an employee of Company E inputs the customer's name, zip code, address, telephone number, method of payment (credit card payment or cash on delivery), and, if applicable, the credit card number and credit card expiration date into the member registration screen (Figure 1).

(3) Each member is assigned a unique membership number. The membership numbers of members who have cancelled their membership are not reused.



Note: Fields that can be entered are indicated by the boxes.

Fig. 1 Member Registration Screen

2. Products

(1) Each product is assigned a unique product number. The product numbers of products that are no longer on sale are not reused.

(2) A product can be a single item or a packaged product. The product number of each product is unique for both single-item and packaged products.

(3) A packaged product may contain multiple single-item products of different types, or two or more single-item products of the same type. A single-item product can be included in various types of packaged product. The number of single-item products that forms a packaged product (the constituent quantity) is fixed. The unit price of a packaged product is set to be less than the total of the unit prices of the single-item products that are included in the packaged product.

(4) Each product has a fixed sales period. There are cases where more than one sales period may exist for a product, such as when a product is sold during a certain season for two or more years in a row. In such case, a different product number is assigned for each sales period.

(5) A new product number is assigned when the product's unit price is changed.

3. Orders

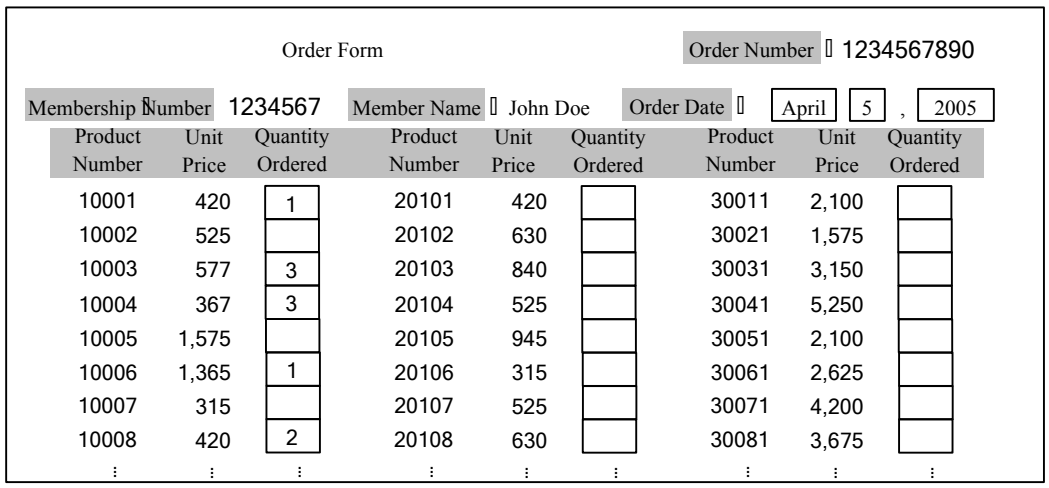
(1) When customers register as members, they are sent a product catalog that contains the product number, product name, product summary, photo, unit price, sales start date and sales end date of each product, and four order forms. New order forms will also be enclosed when ordered items are delivered.

(2) The order form has the order number, membership number, and the member name pre-printed for each member. The form also lists the product numbers, unit prices, and contains blank fields to indicate the order quantity and the order date. The member fills in the order quantity for the product to be ordered, and the order date. (Figure 2). The member then faxes this form to Company E.

(3) The order number is a unique number pre-printed on each order form.

(4) Members can make additional orders by filling new order forms.

(5) Order cancellations are handled individually for each order form. When a member wants to cancel an order, he or she should either write "CANCEL" by hand on the originally-faxed order form and re-send it by fax to Company E, or contact Company E by telephone.



Order Form

Order Number 1234567890

Membership Number 1234567 Member Name John Doe Order Date April 5 , 2005

Product Number	Unit Price	Quantity Ordered	Product Number	Unit Price	Quantity Ordered	Product Number	Unit Price	Quantity Ordered
10001	420	1	20101	420		30011	2,100	
10002	525		20102	630		30021	1,575	
10003	577	3	20103	840		30031	3,150	
10004	367	3	20104	525		30041	5,250	
10005	1,575		20105	945		30051	2,100	
10006	1,365	1	20106	315		30061	2,625	
10007	315		20107	525		30071	4,200	
10008	420	2	20108	630		30081	3,675	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Fig. 2 Order Form

4. Shipping

(1) Because order cancellations often occur on the day of order, or the following day, the total quantity of the products specified on an order form are packed and shipped to the member's address two days after the order receiving date. Shipping is done individually for each order form.

(2) If the quantity ordered is not in stock, it is processed as an out-of-stock order and the out-of-stock products are not shipped. Even if products treated as out-of-stock are later received into stock, these products will not be shipped later.

(3) A shipping slip (Delivery Form and Invoice) similar to that shown in Figure 3 is sent with the product being shipped. The shipping slip not only indicates the shipment date, membership number, member name, shipment number, and amount billed, but will also indicate the order number, product number, product name, unit price, quantity shipped, monetary amount, and remarks as shipping details. For products that are not in stock, the quantity shipped is set to 0, and "out-of-stock (quantity ordered)" is printed in the remarks column.

(4) A unique shipment number is assigned to each shipping slip.

(5) Company E pays for the shipping cost.

Shipping Slip (Delivery Form and Invoice)

Shipping Date: April 7, 2005

Membership Number: 1234567 Member Name: John Doe Shipment Number: 0234567890

Order Number	Product Number	Product Name	Unit Price	Quantity Shipped	Amounts	Remarks
1234567890	10001	[illegible]	420	1	420	
1234567890	10003	[illegible]	577	0	0	Out-of stock (3)
1234567890	10004	[illegible]	367	3	1,101	
1234567890	10006	[illegible]	1,365	1	1,365	
1234567890	10008	[illegible]	420	2	840	
			Amount Billed		3,726	

Fig. 3 Shipping Slip

[Database Design]

Mr. F designed a table structure as shown in Figure 4 based on the required specifications.

Member (membership number, name, zip code, address, telephone number, method of payment, credit card number, credit card expiration date)

Single-item product (product number, product name, product summary, photo, unit price, sales start date, sales end date)

Packaged product (product number, product name, product summary, photo, unit price, sales start date, sales end date)

Order (order number, membership number, order date, order received date, product number, quantity ordered, cancellation flag)

Fig. 4 Table Structure

Mr. G, Mr. F's supervisor, made the following comments, (i) through (v) after studying the table structure in Fig. 4.

(i) The primary and foreign keys are unspecified.

(ii) The "single-item product" table and "packaged product" table should be combined into a "product" table.

(iii) There are no information on what products are included in packaged products.

(iv) The "order" table should be divided into an "order" table and "order details" table.

(v) There is not enough information to create the shipping slips.

When answering the following questions, consult the "Notation Rules for Relational Schemas and Notation Rules for Relational Database Table Structures" for the table structure notation. Also, specify the primary keys and foreign keys.

Subquestion 1

Answer the following questions on comments (i) through (iii) made by Mr. G.

(1) As a response to (i) above, show the primary key and foreign key in each table for "member," "single-item product," and "order" in Fig. 4.

(2) As a response to (ii) above, describe briefly, from a data constraint point-of-view, the advantages of combining the "single-item product" table and "packaged product" table into a "product" table. In addition, when combining these into a "product" table, a new column needs to be added. Describe briefly the reason for adding a new column.

(3) As a response to (iii) above, indicate the structure of the table to be newly added. Note that, when answering this question, the names chosen for the table and columns should describe the meaning of the data to be stored.

Subquestion 2

Answer the following questions assuming that responses to comments (i) through (iii) by Mr. G have been implemented.

(1) As a response to (iv) above, divide the "order" table into an "order" table and "order details" table.
Note that this new table must not be redundant. Furthermore, new columns that were not in the "order" table before the division must not be added.

(2) As a response to (v) above, write the names of the tables to which new columns will be added and the names of those columns in the table below. The answer for (1) above should be taken into account and the column names must be descriptive of the data to be stored and must use names used in the text above.
Note that not all spaces in the table may be filled.

Table Names of Tables to Which Columns are Added and Names of Added Columns

Table Name	Column Name

(3) If additional orders are received from a member before making a shipment to that same member, shipping costs can be reduced by combining those shipments.
In order to make this batching of multiple shipments possible, the structure of one of the existing tables will be modified, and a new table added. Show the structure of the table after it has been modified, as well as the structure of the new table to be added.
Note that the structures of these tables must take into account the answers for (1) and (2) above.

Q3. Read the following description concerning the SQL code for a membership management system, then answer Subquestions 1 and 2.

Company M operates a membership-based sports club. In order to monitor the types of members and the usage of facilities, a membership management system was developed and is now operating.

[Membership Management System Overview]

Membership information is managed in the following way by the membership management system.

(1) A member is assigned a unique membership number. A member is either an individual member or a corporate member (i.e., an individual who belongs to a corporate membership organization).

(2) The time of use (day of week, time of day, etc.) is determined for each usage classification.

(3) The membership type is a classification that determines the monthly membership fee and usage fee (i.e., fee for each time facilities are used) based on the membership and usage classification. The membership type can be modified periodically along with changes in the usage classification.

(4) For corporate members, a registration fee and annual membership fee are required, and the contractual terms determine the individual monthly membership and individual usage fees. There are multiple usage classifications to choose from.

(5) Each time a member uses the facilities, the usage date, membership number, entry time into the facilities, and exit time from the facilities are recorded as part of the usage history.

(6) When members cancel their memberships, the scheduled cancellation date is set as the cancellation date. If membership cancellation is not planned, the cancellation date is set to NULL.

Figure 1 shows the structure of the main tables in the membership management system.

Member (membership number, family name, given name, gender, date of birth, membership classification, usage classification, company ID, address, membership registration date, membership cancellation date)

Company (company ID, company name, address, corporate contract ID, ...)

Corporate contract (corporate contract ID, registration fee, annual membership fee, individual monthly membership fee, individual usage fee, ...)

Membership type (membership classification, usage classification, membership type name, monthly membership fee, usage fee)

Usage classification (usage classification, time of use)

Usage history (usage date, membership number, entry time into the facilities, exit time from the facilities)

Fig. 1 Structure of Main Tables of Membership Management System

[Creation of Usage Status List for Each Membership Type]

The amount of usage is checked with the usage status list (Figure 2) for each membership type. Based on this, membership types as well as the monthly membership and usage fees are periodically reviewed.

In the usage status list for each membership type, the number of members at the end of the period, the number of times the facilities were used during this period, the number of people who newly registered as members, and the number who cancelled their memberships during this period are tallied for the specified period, and this data is output in ascending name order of the membership type. Depending on the membership type, it is possible that no members exist in this list.

Usage Status List for Each Membership Type

Period: March 1, 2005 to March 31, 2005

Membership Type Name	Time of Use	Number of Members	Facility Use Count	New Memberships	Membership Cancellations
Individual A	Weekdays, weekends and holidays (opening time to closing time)	50	250	10	5
Individual B	Weekdays (10:00 a.m. to 5:00 p.m.)	40	400	20	2
Individual C	Weekdays (8:00 p.m. to closing time)	40	400	10	3
Individual D	Weekends and holidays (10:00 a.m. to closing time)	30	150	5	1
Corporate A	Weekdays, weekends and holidays (opening time to closing time)	20	200	10	1
...	...	...	...	...	...
Corporate X	Weekdays (7:00 a.m. to 10:00 a.m.)	0			

Fig. 2 Usage Status List for Each Membership Type

[Membership Management System Modifications]

Company M decided to modify the membership management system in the following way.

(1) In order to increase the number of individual members, a family membership classification will be added in which a discount is applied to members of an individual member's family, thereby making it possible to manage the information on family members in the usage status list shown in Fig. 2. Family members will also have individual membership numbers, and will have usage rights equivalent to the usage classification of their parent member (i.e., the person who has registered as an individual member earlier than other family members).

(2) In order to analyze the relationship between the number of users by age bracket and the number of users by time of use, a cross tabulation report will be created. Figure 3 shows SQL code to create a cross tabulation report. If the same member enters the facilities multiple times on the same day, these will be tallied as separate records in the usage history. `GetAge10(`date of birth, calculation date`)` in Fig. 3 is a user-defined function that calculates the age bracket in decades (by rounding the age down to the nearest decade) given the date of birth and the calculation date. Figure 4 shows the data stored in the membership table and usage history table. Another table below shows the correspondence between the date of birth and the age bracket in decades, when the calculation date is March 1, 2005.

```
SELECT age_bracket, gender, COUNT(A.membership_number) AS A1,
    COALESCE(SUM(B1),0) AS A2, COALESCE(SUM(B2),0) AS A3,
COALESCE(SUM(B3),0) AS A4
FROM (SELECT GetAge10(date_of_birth,'2005-03-01') AS age_bracket, gender,
        membership_number FROM member
    WHERE (membership_cancellation_date IS NULL OR
            membership_cancellation_date > '2005-03-31')
    AND membership_registration_date <= '2005-03-31') AS A LEFT OUTER JOIN
    (SELECT membership_number,
        SUM(CASE WHEN facilities_entry_time < '1200' THEN 1 ELSE 0 END) AS B1,
        SUM(CASE WHEN facilities_exit_time BETWEEN '1200' AND '1700' THEN 1
            ELSE 0 END) AS B2,
        SUM(CASE WHEN facilities_exit_time > '1700' THEN 1 ELSE 0 END) AS B3
    FROM usage_history WHERE usage_date BETWEEN '2005-03-01' AND '2005-03-31'
    GROUP BY membership_number) AS B
    ON A.membership_number = B.membership_number
GROUP BY age_bracket, gender
ORDER BY age_bracket, gender
```

Fig. 3 SQL Code to Create Cross Tabulation Report

Member

Membership Number	...	Gender	Date of Birth	...	Membership Registration Date	Membership Cancellation Date
10001	...	Male	1980-01-10	...	2004-09-07	
10002	...	Male	1972-02-15	...	2004-10-06	
10003	...	Female	1972-03-20	...	2004-11-05	
10004	...	Male	1968-04-25	...	2004-12-04	
10005	...	Female	1973-05-01	...	2005-01-03	
10006	...	Male	1969-06-05	...	2005-02-02	
10007	...	Female	1981-07-01	...	2005-03-01	

Usage History

Usage Date	Membership Number	Facilities Entry Time	Facilities Exit Time
2005-03-06	10003	1030	1300
2005-03-07	10005	1200	1430
2005-03-07	10002	2000	2200
2005-03-07	10007	1900	2200
2005-03-08	10005	1200	1430
2005-03-08	10004	1700	1900
2005-03-09	10005	1200	1430
2005-03-09	10007	1900	2200

Fig. 4 Data Stored in Tables

Table Correspondence between Date of Birth and Age Bracket
(Calculation Date: March 1, 2005)

Date of Birth	Age Bracket	Date of Birth	Age Bracket	Date of Birth	Age Bracket	Date of Birth	Age Bracket
1980-01-10	20	1972-03-20	30	1973-05-01	30	1981-07-01	20
1972-02-15	30	1968-04-25	30	1969-06-05	30		

(Date: "YYYY-MM-DD")

Subquestion 1

Answer the following questions on creating a usage status list for each membership type.

(1) Fill in the correct terms in blanks [A] through [G] in the following SQL code to output a usage status list for each membership type from March 1, 2005 to March 31, 2005 (Fig. 2).

Note that a gray box [ ] indicates that a piece of the SQL code has been omitted.

```
SELECT membership_type_name, time_of_use, number_of_members, usage_count,
        membership_registration_count, membership_cancellation_count
FROM usage_classification, membership_type LEFT OUTER JOIN
  ( SELECT membership_classification, usage_classification, [A]
          AS number_of_members FROM member
    WHERE (membership_cancellation_date IS NULL OR
          membership_cancellation_date > '2005-03-31')
    AND membership_registration_date <= '2005-03-31'
    [B] ) AS present_member [C]
    LEFT OUTER JOIN
  ( SELECT membership_classification, usage_classification, [A] AS usage_count
    FROM usage_history, member
    WHERE member.membership_number = usage_history.membership_number
    AND [D] '2005-03-01' AND '2005-03-31'
    [B] ) AS usage [ ]
    LEFT OUTER JOIN
  ( SELECT membership_classification, usage_classification, [A]
          AS membership_registration_count FROM member
    WHERE [E] '2005-03-01' AND '2005-03-31'
    [B] ) AS registration [ ]
    LEFT OUTER JOIN
  ( SELECT membership_classification, usage_classification, [A]
          AS membership_cancellation_count FROM member
    WHERE [F] '2005-03-01' AND '2005-03-31'
    [B] ) AS cancellation [ ]
WHERE membership_type.usage_classification = usage_classification.usage_classification
ORDER BY [G]
```

(2) Describe briefly the reasons for using an outer join for the derived table "present member."

(3) In order to manage family member information, a new column needs to be added to an existing table. Make sure the usage status of family members can be checked without any changes to the SQL code in (1) above.
Give the name of the table to which a new column should be added, and the contents of that new column.

Subquestion 2

Answer the following questions on cross tabulation reports.

(1) When data is stored as shown in Fig. 4, the resulting cross tabulation report from applying the SQL code in Fig. 3 is as shown in Figure 5 below. Fill in the numerical values that are output in blanks [H] through [K] in Fig. 5.

Cross Tabulation Report
Period: March 1, 2005 to March 31, 2005

Age Bracket	Gender	A1	A2	A3	A4
20□	Male				
20□	Female				
30□	Male				
30□	Female	H	I	J	K

Note: The data in the gray cells is not shown.

Fig. 5 Cross Tabulation Report

(2) Describe briefly the reasons for using the `COALESCE` function in the SQL code in Fig. 3.
Give the age bracket and gender in Fig. 5 for which the intended condition occurs.

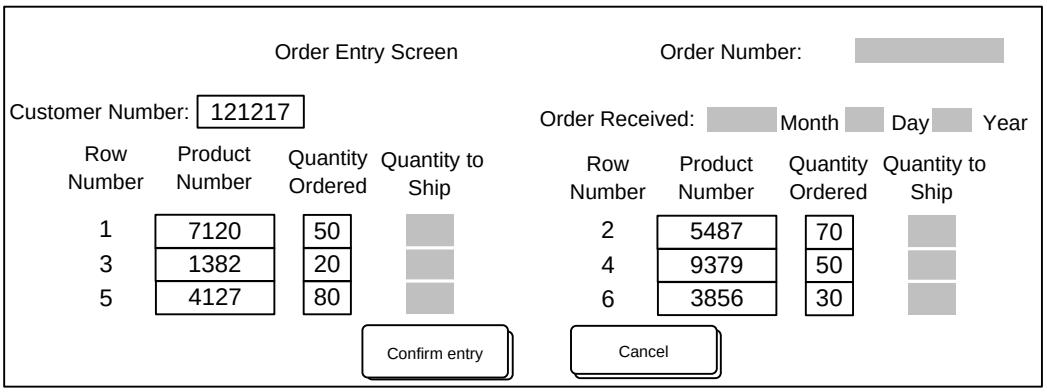
Q4. Read the following description of a program design for updating tables in a relational database, and then answer Subquestions 1 through 3.

Mr. G, who works for Company H, was given the task of designing an order entry processing system to perform order allocations when the details of product orders from customers are entered. Mr. G designed the following order entry screen, order entry processing table structure, and an order entry processing program using embedded SQL.

[Order Entry System Overview]

1. Order Entry Screen and Table Structure for Order Entry Processing

Figure 1 shows the order entry screen, and Figure 2 shows the table structure for order entry processing.



Note: Fields that can be input are indicated by the boxes, and gray locations are displayed after order allocations.

Fig. 1 Order Entry Screen

Order (<u>order number</u>, customer number, order date)
Order details (<u>order number</u>, <u>row number</u>, product number, quantity ordered, quantity to ship)
Inventory (<u>product number</u>, quantity in stock, quantity allocated)

Fig. 2 Table Structure for Order Entry Processing

2. Order Entry Processing Details

(1) The order entry operator inputs the customer number, up to six product numbers for each order, and ordered quantities into the order entry screen. Once the 'confirm entry' button on the order entry screen is pressed, the order entry processing program starts.

(2) The order entry processing program performs the following process. The flow of the program and SQL code are shown in Figure 3.

- The system assigns a unique order number to each order that is input (hereinafter, "number assignment"), and inserts one row into the "order" table (see the step numbers (ii) through (iv) in Fig. 3).
- The system allocates orders by referring to the "inventory" table for each ordered product, and inserts one row into the "order details" table (see the step numbers (v) through (viii) in Fig. 3).
- After order allocations are performed for all ordered products, the order number, order date, product number and quantities to ship are displayed on the order entry screen, and processing is terminated (see the step number (ix) in Fig. 3).

(3) The order entry operator checks the order number and quantities to ship that are displayed on the screen.

(i) Set the ISOLATION LEVEL of the transaction to READ COMMITTED.[1]
(ii) Obtain the customer number, row numbers, product numbers, and the quantities ordered from the order entry screen.
(iii) Add 1 to the current maximum order number in the "order" table, and make this the new order number.
```
SELECT MAX(order_number)+1 INTO :order_number FROM order
```
(iv) Set the order date to the date of program execution, and then insert one row into the "order" table.
```
INSERT INTO order(order_number, customer_number, order_date)
        VALUES(:order_number, :customer_number, :order_date)
```
(v) Repeat the processing in steps (vi) through (viii) within the dotted lines for each product in the order of the row numbers in the order entry screen.

(vi) The allocatable quantity in the product "inventory" table is set to the difference between the quantity in stock and the quantity allocated.
```
SELECT quantity_in_stock, quantity_allocated INTO :quantity_in_stock, :quantity_allocated
        FROM inventory WHERE product_number=:product_number
```
(vii) Compare the quantity ordered with the allocatable quantity, set the quantity to be shipped (a host variable) to the smaller of these two values, and add this quantity to be shipped to the quantity allocated in the "inventory" table.
```
UPDATE inventory SET quantity_allocated=quantity_allocated+:quantity_to_be_shipped
        WHERE product_number=:product_number
```
(viii) Insert one row into the "order details" table.
```
INSERT INTO order_details(order_number, row_number, product_number, quantity_ordered,
                quantity_to_be_shipped)
        VALUES(:order_number, :row_number, :product_number,
                :quantity_ordered, :quantity_to_be_shipped)
```

(ix) If processing results in a normal termination, a COMMIT statement is issued, and the order number, order date and quantities to be shipped for the product numbers are displayed on the order entry screen.
(x) When an exception condition occurs, a ROLLBACK statement is issued.

Note [1]: Exclusive access control by an SQL code is done row by row. When a row that is being updated is accessed by another transaction, that transaction is blocked, and only the rows that were committed can be read in. In addition, it is assumed that the ISOLATION LEVEL of this transaction does not change throughout the operation.

Fig. 3 Flow of Order Entry Processing Program and SQL Code

Assume that the customer numbers that are given to the order entry processing program are correct, and that all product numbers exist in the "inventory" table. When a duplicate key violation is detected for the order number which is the primary key to the "order" table, or when a deadlock occurs during the execution of an SQL code, the transaction is rolled back (see the step number (x) in Fig. 3).

[Problems That Occurred]

When the order entry processing program shown in Fig. 3 processed transactions in parallel, the following problems X through Z occurred.

1. Problem X

When the processing of (iv) in Fig. 3 was performed, a duplicate key exception occurred for the order number, which is the primary key, and the row could not be inserted into the "order" table. In order to solve this problem, a "number assignment" table as shown in Figure 4 was added and the transaction slip type was specified as its primary key, to be used for the number assignment of other processing such as those involving shipping instructions, thereby ensuring that number assignments would be unique. The processing in (iii) in Fig. 3 was also modified as shown in Figure 5. As a result, although duplicate key exceptions for order numbers no longer occurred, there was a large degradation in the transaction throughput.

Number assignment (transaction slip type, current transaction slip number)

Fig. 4 Table Structure

```
(iii)-1  Add 1 to the current transaction slip number in the number assignment table.
  UPDATE number_assignment
        SET cur_transaction_slip_number=cur_transaction_slip_number+1
        WHERE transaction_slip_type='order'

(iii)-2  Obtain the current transaction slip number from the number assignment table.
  SELECT cur_transaction_slip_number INTO :order_number
        FROM number_assignment WHERE transaction_slip_type='order'
```

Fig. 5 Order Entry Processing Program Flow and SQL Code After Modification (partial)

2. Problem Y

A deadlock sometimes occurs when the inventory table update given by (vii) in Fig. 3 is performed. The frequency of deadlocks tends to be especially high between transactions that enter orders for multiple best-selling products.

3. Problem Z

An anomaly occurred in which the allocatable quantity (i.e., the difference between the quantity in stock and the corresponding quantity allocated) in the "inventory" table became negative for a certain product. After modifying the SQL code for (vi) and (vii) in Fig. 3 to that which uses a cursor as shown in Figure 6 below, this anomaly no longer occurred.

(vi)-1 Declare a cursor. Make the cursor name CSR.

```
DECLARE CSR CURSOR FOR SELECT quantity_in_stock, quantity_allocated FROM inventory
  WHERE product_number=:product_number FOR [ A ] OF quantity_allocated
```

(vi)-2 Set the product number to the product number of the host variable.

(vi)-3 Open cursor CSR.

```
OPEN CSR
```

(vi)-4 Use a FETCH statement to read one row from cursor CSR, get the values of the quantity in stock and the quantity allocated, and set the allocatable quantity to be the difference between these two values.

```
FETCH CSR INTO :quantity_in_stock, :quantity_allocated
```

(vii) Compare the quantity ordered with the allocatable quantity, set the quantity to be shipped (a host variable) to the smaller of these two values, and add this quantity to be shipped to the quantity allocated in the inventory table.

```
UPDATE inventory SET quantity_allocated=quantity_allocated+:quantity_to_be_shipped
    WHERE [ B ] OF CSR
```

(vii)-2 Close cursor CSR.

```
CLOSE CSR
```

Fig. 6 Order Entry Processing Program Flow that Uses a Cursor, and SQL Code (partial)

Subquestion 1

Answer the following questions on the solution to Problem X that occurred for order number assignment.

(1) From the point of view of exclusive access control for transactions, give the reasons briefly for the degradation in throughput.

(2) In order to solve the problem of the degradation in throughput, one SQL statement was added to the order entry processing program shown in Fig. 3.
Before which step number should this SQL statement be added in the program shown in Fig. 3?
Give that step number and the SQL statement that should be added.

Subquestion 2

Answer the following questions on Problem Y that occurred during "inventory" table updating.

(1) Give the reason briefly for the deadlock occurring.

(2) In order to prevent deadlocks to the "inventory" table, one step from (ii) through (viii) was modified without any additions, changes, or deletions of SQL codes in the order entry processing program shown in Fig. 3.
Which process in the flow shown in Fig. 3 should be modified?
Give the step number and a brief description of the processing after this modification.
Note that parallel processing of order allocations of multiple and distinct products according to multiple transactions must be possible, as it was prior to this modification.

Subquestion 3

Answer the following questions on Problem Z that occurred during "inventory" table updating.

(1) From the point of view of exclusive access control for transactions, give the reason briefly for the anomaly, under which the allocatable quantity became negative.

(2) Fill in the correct terms in the blanks [ ] in Fig. 6 to complete the SQL code.