1. Explain the following terms and concepts. Use examples if necessary.
2. Atomicity of updates – 원자성 데이터 갱신 데이터베이스의 가장 중요한 점은 데이터들의 변화가 생길 때입니다. 데이터 베이스에 변화가 왔을 때에 변화의 정도가 최소화가 되야 한다는 것입니다. 예를 들어서 은행업무를 들어보겠습니다. 은행업무의 데이터베이스 시스템을 구축하게 되었습니다. 은행 업무 중에서 계좌 이체라고 하는 서비스가 있습니다. 계좌 이체 서비스는 더 의미가 넓습니다. 하지만 같은 은행의 두 개의 계좌로 한정되어서 생각해 보겠습니다. 돈을 보내려는 계좌의 고객이 카드를 넣고, 비밀번호 입력도하고 돈의 금액도 넣고, 상대계좌의 번호까지 입력하고 보내는 도중에 데이터베이스가 다운이 되었다고 생각을 해봅시다. 만약 상대계좌에는 돈이 보내지지 않았습니다. 그리고 보낸 사람의 계좌에는 돈만 빠져나가게 되었습니다. 이러한 상황은 개인의 돈이 사라지는 정도의 수준이 아닙니다. 데이터베이스의 서비스를 제공하고 있는 은행도 큰 타격을 입게 됩니다. 그렇기 때문에 데이터 변경할 때에는 동기화는 될 수가 없지만, 그에 버금가는 데이터 변경시의 주의 사항이 요구됩니다. 그 변경 할 때의 최소한의 변경 그것이 원자성 데이터 데이터 갱신입니다.
3. Physical/logical data independence – 앞의 두 단어인 physical/logical을 살펴 보기 전에 data independence의 뜻에 관해서 알아 볼 필요가 있을 것 같습니다. Data의 저장 형태는 결국 in disk 즉, hard disk에 저장이 됩니다. 물론 속도의 극대화를 위해서 메모리에 상주하는 데이터 베이스도 있습니다. 하지만 여기서는 편의를 위해서 disk에 있다고 가정을 하겠습니다. Hard disk에 저장되어 있는 형태는 물리적인 형태인 Physical level의 저장 형태입니다. 실제로 저장되어 있는 형태라는 것입니다. 그러나 그 저장형태는 사람의 직관적인 판단을 할 수가 없습니다. 데이터간의 저장 형태를 보고 싶어하는 것이 아니기 때문입니다. 그래서 사람이 데이터베이스를 알아보기 편하도록 알아보기 쉬운 형태가 필요한 것입니다. 그것이 conceptual level 논리적(logical)형태의 data가 필요한 것입니다. 그래야지 사람이 어떻게 저장되어있는지 상관 없이 데이터를 볼 수가 있으니까요. 예를 들어서 entity나 relationship 등을 볼 수가 있게 되는 것입니다. 그리고 실제로 데이터베이스가 사용자들에게 실제로 보여지는 단계의 view level이 있습니다. 실제로 application에서 보여지는 database입니다. 모든 데이터베이스를 사용자에게 들어낼 필요는 없습니다. 사용자에게 요청한 데이터베이스만 화면에 보여주게 되면 되는 것입니다. 그렇다면은 logical/physical independence는 무엇을 뜻하는 것일까요? 바로 각 단계는 연관 관계가 있습니다. 하지만 각 관계가 전부를 알 필요는 없습니다. 제일 하위 단계인 physical level은 바로 위의 단계인 logical하고의 관계만 있으면 됩니다. 그리고 그 위에 단계인 logical은 위의 단계인 view level을 알 필요가 없습니다. View level 또한 physical은 알 필요가 없이 logical단계만 알면 됩니다. 이렇듯이 아래의 단계는 위에 단계를 알 필요가 없이 아래 단계 또한 바로 밑 단계만 알면 되는 것이 physical/logical data independence입니다.
4. Schema and instance – schema는 데이터베이스의 논리적인 구조입니다. 데이터베이스의 내용의 빼대라고도 볼 수가 있습니다. Instance는 데이터 베이스에 논리적인 구조에 실제로 들어가는 값입니다. instance에는 찍히는 순간이라는 뜻이 있습니다. 데이터베이스가 찍혔을 때에 들어가 있는 값이라고 생각을 하면은 쉽게 이해할 수 있을 것입니다.
5. Key/super key/ candidate key - key라고 하면은 보통 무엇이 떠오르나요? 자물쇠가 있다고 하면은 그 자물쇠에 맞는 열쇠 구멍이 떠오리 않으시나요? 자물쇠중에서 단하나의 맞는 열쇠 그것이 key입니다. 하지만 데이터베이스에는 약간 그 의미가 다릅니다. 바로 데이터를 여러 개의 information주에서 식별 확인하는 식별자 즉, indentifition의 기능을 하는 것이 key입니다. Super key는 식별할 수 있는 식별자에서 약간 더 의미가 작아지게 됩니다. Entity set 즉, 여러 개의 entity중에서 하나의 entity를 유일하게 지정할 수 있는 attribute또는 attribute의 set을 super key라고 합니다. Candidate key는 앞의 super key보다 더 내용이 적습니다. 그래서 candidate key까지 설명을 하고 밑의 예를 들어서 설명을 하도록 하겠습니다. Candidate key는 super key중에서 가장 minimalize한 가장 작은 key입니다. 물론 하나의 entity를 유일하게 지정할 수 있어야 합니다. 예를 들어서 설명을 해보도록 하겠습니다. 학생이라는 entity가 있습니다. 학생이라는 entity는 이름, 생일, 주소, 학번, 나이, 성별, 학과,주민등록번호 등을 가지고 있습니다. 여기서 super key는 하나의 entity만을 식별하면 됩니다. 그렇기에 여러가지 경우의 super key있을 수가 있습니다. (이름, 생일, 학번)이렇게 composite key도 super key가 될 수가 있습니다. 위의 super key중에서 가장 작은 키가 candidate key가 됩니다. 여기서는 학번과 주민등록번호가 둘 다 candidate key가 될 수가 있습니다. Super key중에서 가장 작은 키이기 때문입니다.
6. Disjoint vs Overlapping constraints – disjoint(분기)와 overlapping(중복) 둘 다 하나의 parent entity에서 specialization(일반화)의 방식입니다. Specialization(일반화)같은 경우에는 entity set을 subgroup으로 설계하는 과정입니다. 처음으로 disjoint에 대해서 알아보도록 하겠습니다. Disjoint는 하나의 entity는 하나의 하위 그룹에만 속할 수 있는 것입니다. overlapping같은 경우에는 하나의 entity가 여러가지 subgroup의 속성을 가질 수가 있습니다. 예를 들어서 설명을 해보도록 하겠습니다. People이라고 하는 사람을 나타내는 entity가 있습니다. 그 밑의 하위 subentity로는 customer(고객)이라는 entity와 employee(피고용자)라는 entity가 있습니다. 만약 subgroup의 관계가 overlapping관계가 되게 된다면은 고객이면서 피고용자의 속성까지 가질 수가 있습니다. 하지만 customer의 people과 customer의 관계가 disjoint관계로 되어있다고 가정을 해봅시다. 그렇게 되면은 people에서 분기되어 나온 entity가 employee로 분기 되어 나왔다면은 다른 서브속성을 가질 수가 없습니다.
7. Total vs partical constraints – entity의 참여 관계 중에서 total(전체)적인 참여와 partical(부분)적인 참여로 entity들의 관계가 이루어 질 수 있습니다. Total은 entity관계에서 entity가 빠짐없이 relation관계에 참여를 하는 것입니다. 그리고 partical은 참여를 할 수도 있지만 참여를 안 할 수도 있습니다. 무슨 이야기인가 하면은 예를 들겠습니다. Customer(고객) entity와 loan(대출계좌) entity관계에 대해서 생각을 해 보면 쉽습니다. Customer는 loan을 가질 수도 있지만 가지지 않을 수도 있습니다. 고객이 대출을 하지 않는다. 이렇게 되면은 대출계좌를 가질 필요가 없다는 것입니다. 이 앞에서의 관계는 customer entity는 loan entity와의 relation관계에서 부분적으로 관계를 맺고 있습니다. 반대로 생각을 해 보겠습니다. Loan(대출계좌는) entity는 customer(고객)과의 관계에서 어떤 관계를 맺고 있을까요? 대출계좌를 가지고 있는 고객이 없다면은 대출계좌는 존재를 할 수가 없습니다. 대출계좌가 존재를 한다고 하면은 반듯이 그 대출계좌를 가지고 있습니다. 그렇기 때문에 대출계좌는 고객과의 관계에서 total 적인 관계를 맺고 있습니다. 대출계좌를 가지고 있는 고객이 없다면은 대출계좌는 존재를 할 수가 없기 때문입니다.
8. Relation cardinality – 여러 개의 entity들은 관계를 맺습니다. entity들의 연관 관계는 이어져 있다는 면도 있습니다. 하지만 여러 개의 entity들이 연관을 맺을 수도 있습니다. 여러 개의 entity와 하나의 entity들이 관계를 맺을 수도 있습니다. 관계를 맺는 entity들의 수량은 database를 설계를 할 때의 중요한 설계 부분입니다. 하나와의 하나의 관계가 맺어 질 수가 있습니다. 앞의 정의된 것을 1:1관계라고 합니다. 하나와 여러 개의 관계가 맺어 질 수 있습니다. 앞의 entity들의 연관관계의 수가 많을 때에 m:1 관계라고 합니다. 그리고 뒤의 entity가 연관관계의 수가 많을 수가 있습니다. 그것을 1:n 관계라고 합니다. 그리고 마지막으로 관계들이 여러 개와 여러 개로 연관 될 수가 있습니다. 그러한 관계를 m:n 관계라고 합니다. 예를 들어서 설명을 해 보겠습니다. Customer와 loan entity를 다시 사용해보도록 하겠습니다. 1:1 관계에 customer와 loan이 있게 된다면은 한 명의 고객은 하나의 대출 계좌를 가질 수만 있습니다. 그리고 m:1의 관계가 된다면은 여러 개의 고객은 하나의 대출계좌만을 가질 수가 있습니다. 1:n 관계가 되는 것도 봐볼까요? 그렇게 된다면은 한 명의 고객이 여러 개의 대출 계좌를 가질 수가 있는 것입니다. 그리고 마지막으로 m:n의 관계를 봐볼 까요 그렇게 된다면은 여러 명의 고객이 여러 개의 대출 계좌를 가질 수가 있게 되는 것입니다.
9. Read through the example in 2.8.2 of 4th edition of the text and construct an E-R diagram. You must use the IE(information engineering) notation supported by Erwin.

첨부파일 2.erwin에 그렸습니다.

1. A university registrar’s office maintains data about the following entities: (a) courses, including number, title, credits, syllabus, and prerequisites; (b) course offerings, including course number, year, semester, section number, instructor(s), timings, and classroom; (c) students, including student-id, name, and program; and (d) instructors, including identification number, name, department, and title. The enrollment of students in courses and grades awarded to students in each course they are enrolled for must be appropriately modeled. Construct an E-R diagram for the registrar’s office. Use the IE(information engineering) notation (using ERwin) to draw your diagram. Document all assumptions that you make about the mapping constraints.

문제에 나와 있는 데로 속성을 넣었습니다. Relation cardinality에 관해서는 다음과 같은 생각을 하였습니다.

1. 하나의 강의는 여러 개의 강의 신청이 있을 수가 있다.
2. 하나의 강의신청은 하나의 강의를 맡게 된다.
3. 하나의 강의는 한 명의 교수가 할 수도 있고 안 할 수도 있다.
4. 하나의 강의 신청은 여러 명의 학생이 신청 할 수 있다.
5. 하나의 학생은 여러 개의 상태를 가질 수도 안 가질 수도 있다.
6. 하나의 강의는 하나의 상태를 가질 수도 안 가질 수도 있다.

|  |  |  |  |
| --- | --- | --- | --- |
| Entity | Primary Key | Foreign Key | Attributes |
| Course | number |  | title, credit, syllabus, prerequisites |
| register |  | number, course\_number |  |
| Course\_offering | course\_number |  | year, semester, section\_number, instructor, timing, timings, classroom |
| Instructor | identification\_number |  | name, department, title |
| reserved |  | student\_id  course\_number |  |
| Student | student\_id |  | name, program |
| stats |  | number  student\_id | grade |

첨부파일 3.erwin에 그렸습니다.

1. Consider a database used to record the scores that students get in different exams of different course offerings.
2. Construct an E-R diagram that models exams as entities, and uses a ternary relationship, for the above database. Use the IE(information engineering) notation (using ERwin) to draw your diagram.

첨부파일 4a).erwin에 그렸습니다.

1. Construct an alternative E-R diagram that uses only a binary relationship between students and course-offerings. Make sure that only one relationship exists between a particular student and course-offering pair, yet you can represent the scores that a student gets in different exams of a course offering. Again use the IE(information engineering) notation (using ERwin) to draw your diagram.

첨부파일 4b).erwin에 그렸습니다.

1. Consider the following E-R diagram, which models an online bookstore.
2. List the entity sets and their primary keys.

|  |  |  |
| --- | --- | --- |
| Entity | Primary key | foreign key |
| Author | name, address |  |
| written\_by |  | name,address,ISBN |
| Publisher | name |  |
| published\_by |  | name,ISBN |
| Customer | email |  |
| basket\_of |  | email,basketID |
| Book | ISBN |  |
| contain |  | ISBN, basketID |
| Shopping\_basket | basketID |  |
| stock |  | ISBN, code |
| Warehouse | code |  |

1. Suppose the bookstore adds music cassettes and compact disks to its collection. The same music item may be present in cassette or compact disk format, with differing prices. Extend the E-R diagram to model this addition, ignoring the effect on shopping baskets. Use the diagramming convention discussed in the class.

첨부파일 5b).erwin에 그렸습니다.

1. Now extend the E-R diagram, using generalization, to model the case where a shopping basket may contain any combination of books, music cassettes, or compact disks. Use the diagramming convention discussed in the class.

첨부파일 5c).erwin에 그렸습니다.

1. Consider the following E-R diagram, which models an airline company.
   1. Extend this E-R model to include catering data, i.e., *meal service.* Use the diagramming convention discussed in the class.

■A meal is associated with a set of dishes.

■A dish has a name, cost, and fat content.

■Each flight is associated with at most one meal.

■If a departure is associated with a flight which is associated with a meal then this departure is also associated with at most one employee who is known to have cooked the meal served on this departure.

첨부파일 6.erwin에 그렸습니다.

* 1. Assign a primary key to each of your entity sets.

첨부파일 6.erwin에 주키(primary key)를 할당 하였습니다.

* 1. Which, if any, of your entity sets should be weak?

Meal, Departure, Aircraft가 weak entity set으로 될 만 합니다.

2.



3.

4a)

4b)

5a)

5b)



5c)



6.