

Midterm Solution

Problem 1

Answer:

(a) Refer to Figure 1

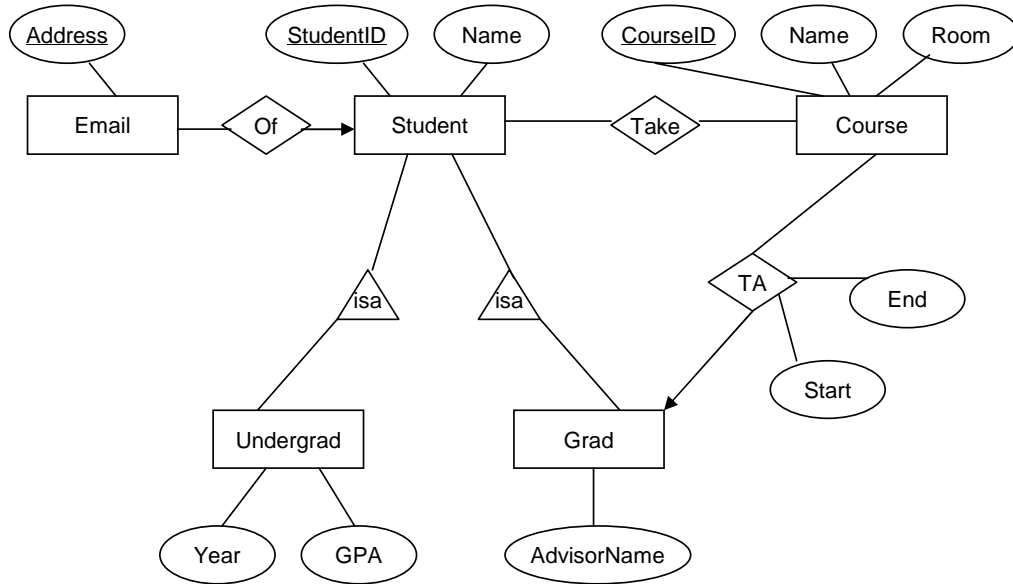


Figure 1: ER Diagram

Note: there is a minor typo in the above figure. The arrow from "TA" to "Grad" must be round-headed, to reflect the referential integrity constraint of "exactly one".

(b)

Undergrad(StudentID, Name, Year, GPA)

Grad(StudentID, Name, AdvisorName)

Email(Address, StudentID)

TakeCourse(StudentID, CourseID)

Course(CourseID, Name, Room, StudentID, Start, End)

(c) Relational model is designed for machines to understand. It only has one concept, the relation. Therefore it has certain inflexibilities to start with. ER diagram, on the other hand, naturally represents domain logic. It offers several complementary concepts (e.g. entity set, relationship). It is convenient for representing an initial, high level database design.

Problem 2

Answer:

(a) Suppose the attributes in X are x_1, x_2, \dots, x_n and the attributes in Y are y_1, y_2, \dots, y_m . Z is a subset of Y , so all the attributes of Z are from the attributes y_1, y_2, \dots, y_m in Y . From the definition of functional dependency $X \rightarrow Y$, we know for any two tuples that agree on attributes x_1, x_2, \dots, x_n , they also agree on attributes y_1, y_2, \dots, y_m . They must agree on all the attributes of Z , because all the attributes of Z are from attributes y_1, y_2, \dots, y_m . Therefore, from the definition of functional dependency, we have $X \rightarrow Z$.

(b) The only non-trivial FDs in a two-attribute relation (name these two attributes A and B) are $A \rightarrow B$ and $B \rightarrow A$. For FD $A \rightarrow B$, we know A decides B and also decides A itself trivially. Therefore, A is key. For FD $B \rightarrow A$, we know B decides A and also decides B itself trivially. Therefore, B is key. We see the left hand side of all the possible non-trivial FDs in a two-attribute relation is key. Therefore, any two-attribute relation is in BCNF.

Problem 3

Answer: R is not in BCNF, all three given FD's are violations. For example, the closure of AB is ABC , which does not cover the entire relation. (Alternatively, you could find the minimal key is AD , and since none of the FD's contain both A and D , they all violate BCNF.)

Decomposition: (Answers may vary) First, decompose on $AB \rightarrow C$. The closure of AB is ABC , so we have $R_1(A, B, C)$ with key AB and $R_2(A, B, D, E)$ with key AD (which can be determined by computing closures). R_1 is in BCNF since the only non-trivial FD, explicit or derived, that applies is $AB \rightarrow C$, which contains the superkey AB . R_2 is *not* in BCNF, because both $D \rightarrow E$ and $DE \rightarrow B$ still apply, and neither one contains a superkey.

Decompose R_2 on $D \rightarrow E$. The closure of D is DEB , so we have $R_{2,1}(D, E, B)$ with key D and $R_{2,2}(D, A)$ with key AD . $R_{2,1}$ is in BCNF because all FD's now contain a superkey, and $R_{2,2}$ is in BCNF because all two-attribute relations are in BCNF.

Final answer: $R_1(A, B, C)$, $R_{2,1}(D, E, B)$, $R_{2,2}(D, A)$.

Problem 4

There are multiple valid answers to each of the following questions. We list only one for each.

(a) $\pi_{Company.name, stock-price}(\sigma_{city='champaign'}(Person) \bowtie_{ssn=buyer-ssn} Purchase \bowtie_{Purchase.pid=Product.pid} \sigma_{category='TV'}(Product) \bowtie_{Product.cid=Company.cid} Company)$.

(b)

```
SELECT Company.name, stock-price, country
FROM Company, Product, Purchase, Person
WHERE Company.cid=Product.cid AND Product.pid=Purchase.pid AND
buyer-ssn=ssn AND city="Champaign"
EXCEPT
SELECT Company.name, stock-price, country
```

```
FROM Company, Product, Purchase, Person
WHERE Company.cid=Product.cid AND Product.pid=Purchase.pid AND
buyer-ssn=ssn AND city="Urbana";
```

(c)

```
SELECT Company.name
FROM Company, Product, Purchase
WHERE Company.cid=Product.cid AND Product.pid=Purchase.pid
GROUP BY Company.name
HAVING COUNT(DISTINCT Product.pid) >= 2;
```

(d)

```
SELECT name, COUNT(DISTINCT pid) AS total
FROM Person, Purchase
WHERE ssn=buyer-ssn AND city="Champaign"
GROUP BY ssn, name
HAVING total>=5;
```

(e)

```
SELECT name, COUNT(DISTINCT pid) AS total
FROM Person, Purchase
WHERE ssn=buyer-ssn AND city="Champaign"
GROUP BY ssn, name
HAVING total > SELECT AVG(totalurbana) FROM (
SELECT COUNT(DISTINCT pid) AS totalurbana
FROM Person, Purchase
WHERE ssn=buyer-ssn AND city="Urbana"
GROUP BY ssn
);
```

It is fine to remove the DISTINCT in the subquery. For “the average number of products that each person living in Urbana has bought” in the question, you can assume that “different products” is not required.

Problem 5

Answer: (Answers may vary.)

(a) Views are useful in simplifying SQL statements by encapsulating commonly-used calculations. They are also useful in speeding up query answering (see below), defining access-control to hide some of the values from certain users.

(b) Both types of views are derived from other tables in the database, rather than being the original sources of the data. Both can be queried directly as though they were tables. Both can be used to implement the ideas from part (a).

Materialized views are pre-calculated and stored on the database so that we can save com-

puting time; however they require extra work to keep them updated whenever the underlying tables are modified. Virtual views are reconstructed every time they are used so they always reflect the current state of the database.

Problem 6

Answer:

(a) Attribute-based checks are the easiest to implement, since this is simply a constraint on the value of a particular attribute and all we want to do is reject any modification that violates this constraint. (Assertions and triggers could be used, but are overkill for this problem.)

(b) Triggers can be used here, since we not only have to check the given condition, but we also have an action that needs to be performed when the condition fails. (Assertions and checks cannot handle the action.)