CS3402 Tutorial 8:

1. Which of the following schedules is (conflict) serializable? For each serializable schedule, determine the equivalent serial schedules.

   (a) $r_1(X)$; $r_3(X)$; $w_1(X)$; $r_2(X)$; $w_3(X)$;
   (b) $r_1(X)$; $r_3(X)$; $w_3(X)$; $w_1(X)$; $r_2(X)$;
   (c) $r_3(X)$; $r_2(X)$; $w_3(X)$; $r_1(X)$; $w_1(X)$;

2. Consider the following concurrent schedule. Draw the serialization graph for the schedule. Is it conflict serializable?

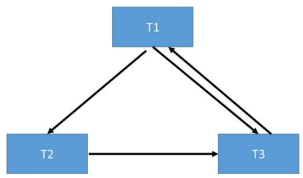
Ta	**Tb**	**Tc**
	Read(x)	
Write(y)		
		Read(y)
	Write(y)	
Write(x)		
	Commit	
		Write(z)
Commit		
		Commit

3. Consider schedules S_1, S_2 and S_3 below. Determine whether each schedule is strict, cascadeless, recoverable, or nonrecoverable. Determine the strictest recoverability condition that each schedule satisfies.
   (a) $r_1(X)$; $w_1(X)$; $r_2(X)$; $r_1(Y)$; $w_2(X)$; c_2; c_1;
   (b) $r_1(X)$; $w_1(X)$; $r_2(X)$; $r_1(Y)$; $w_2(X)$; $w_1(Y)$; c_1; c_2;
   (c) $r_1(X)$; $w_1(X)$; $w_2(X)$; $w_1(Y)$; c_1; $r_2(X)$; c_2;
   Can you change c) into a strict schedule?

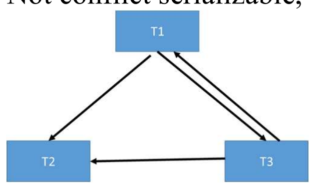
CS3402 Tutorial 8:

1. **Answer**:

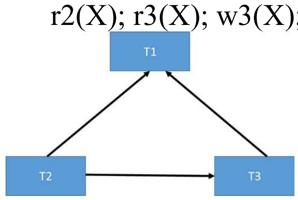
(a) Not conflict serializable, because the serialization graph contains cycle.



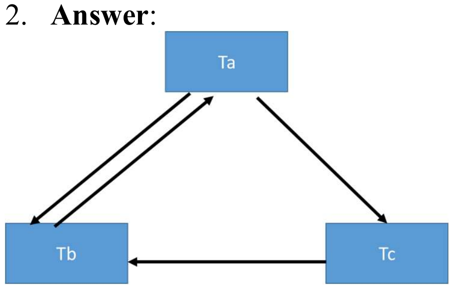
(b) Not conflict serializable, because the serialization graph contains cycle.



(c) Conflict serializable, because the serialization graph contains no cycle.
it is equivalent to this serial schedule:
r2(X); r3(X); w3(X); r1(X); w1(X); (or you can simply write T2, T3, T1)



2. **Answer**:



It is not serializable as the schedule is cyclic.

3. **Answer**:

(a) Non-recoverable, because T2 reads X written by T1， but T1 commits after T2. If T1 abort after T2 commits, the value of X is not recoverable.

(b) Recoverable, because T2 reads X written by T1 and T2 commits after T1, which satisfy the condition of recoverable “ A schedule S is recoverable if no

transaction T in S commits until all transactions T′ that have written some item X that T reads have committed.”
It is not cascadeless， because T2 reads X written by T1 before T1 commits. If T1 fails, T1 has to be rolled back and T2 also need to be rolled back.

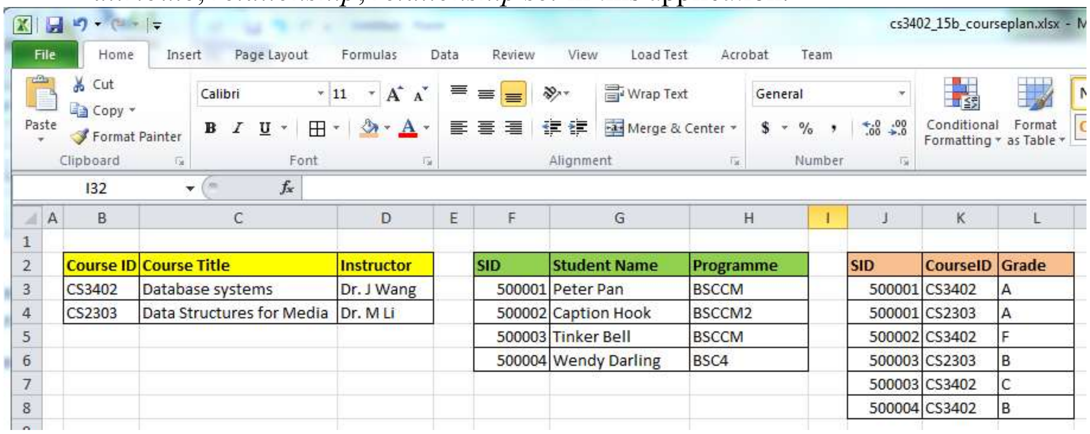
(c) Cascadeless. Because T2 reads X written by T1 after T1 commits. It satisfies the condition of Cascadeless schedule “Every transaction reads only the items that are written by committed transactions.”

It is not the strict schedule because T2 write X after T1 write X but before T1 commits. It does not satisfy the condition of strict schedule： “A schedule in which a transaction can neither read or write an item X until the last transaction that wrote X has committed.”

Schedule (c) can be changed into the strict schedule:
r1(X); w1(X); w1(Y); c1; w2(X); r2(X); c2;

CS3402 Tutorial 1 (Introduction and ER Model):

1. Below are some sample data stored in an Excel file. Identify *entity*, *entity set*, *attribute*, *relationship*, *relationship set* in this application.



Course ID	Course Title	Instructor
CS3402	Database systems	Dr. J Wang
CS2303	Data Structures for Media	Dr. M Li

SID	Student Name	Programme
500001	Peter Pan	BSCCM
500002	Caption Hook	BSCCM2
500003	Tinker Bell	BSCCM
500004	Wendy Darling	BSC4

SID	CourseID	Grade
500001	CS3402	A
500001	CS2303	A
500002	CS3402	F
500003	CS2303	B
500003	CS3402	C
500004	CS3402	B

2. Construct an ER diagram for a car insurance company with a set of customers, each of whom owns a number of cars. Each car has a number of recorded accidents associated with it.

3. Construct an ER diagram for a hospital with a set of patients and a set of medical doctors. A log of the various conducted tests and results is associated with each patient.

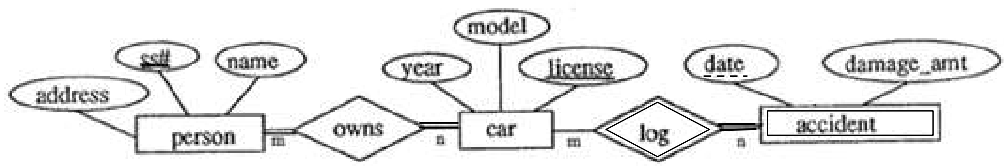
Note the questions (2 & 3) do not contain sufficient information for building the two E/R diagrams. So you can have your own assumptions when drawing your ER diagrams.

CS3402 Tutorial 1:

1. **Answer**:

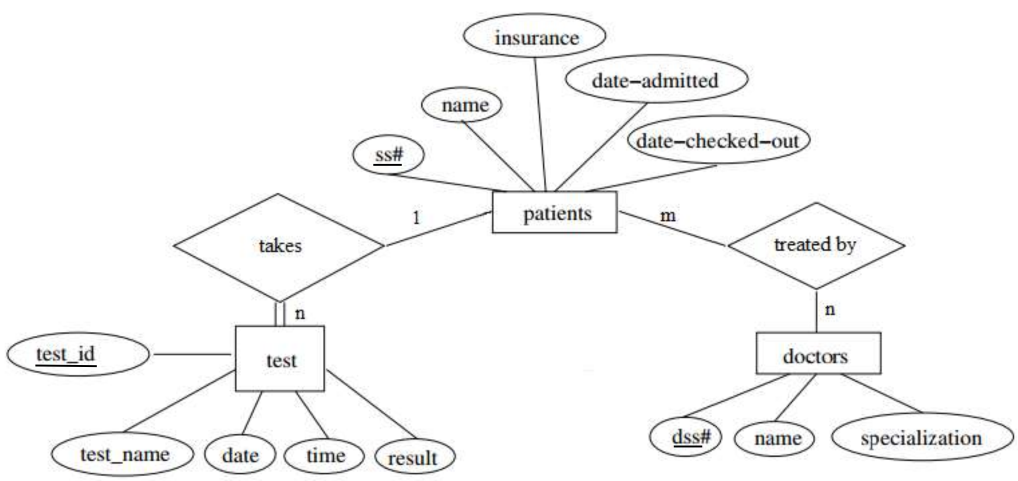
   Entity: every single course, each individual student, each instructor
   Entity set: the set of students, the set of courses, and the set of instructors
   Attributes: CourseID, Course Title, Student ID, Student Name, Student Programme, Student Grade (an attribute of a relationship), Instructor Name
   Relationship: Dr. J Wang teaching CS3402, Dr. M Li teaching CS2303, student Peter Pan taking course CS2303, student Caption Hook taking course CS3402 …
   Relationship set: the set of relationships of which students taking which courses, the set of relationships of which teacher teaching which course

2. **Answer**:



It is assumed that one car may be owned by multiple customers (like family members). An accident may involve more than one car and a car may have several times of accidents or no accident.

3. **Answer**:



It is assumed that a patient may have more than one doctor； a doctor can treat many patients. Some patients may be not treated by any doctor yet; and a new doctor has not treated any patient yet. Each test has a unique test ID.

Note the questions (2 & 3) do not contain sufficient information for building the two E/R diagrams. The answers are only samples.

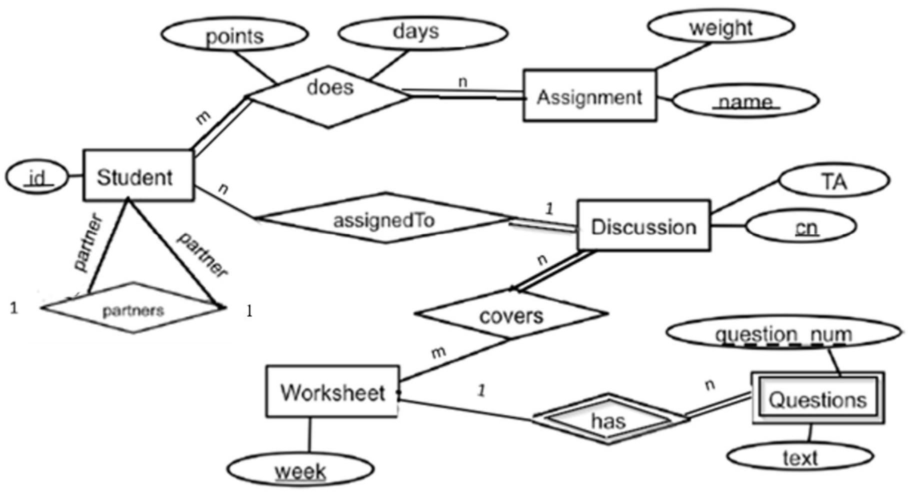
For Q2, it is assumed that one car may be owned by multiple customers (like family members). An accident may involve more than one car. Some car may be not involved in any of accidents.

For Q3, a patient may have more than one doctor； a doctor can treat many patients. Some patients may be not treated by any doctor yet; and a new doctor has not treated any patient yet. Each test has a unique test ID.

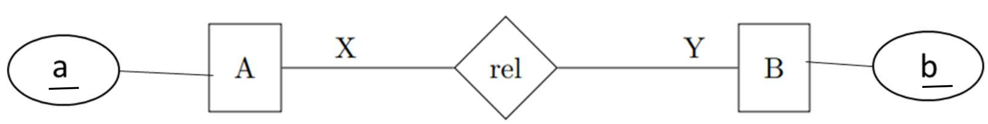
To TA：Please explain some key points in Q2&Q3 answers to our students, including weak entity set, identifying relationship, full participation, cardinality constraint, and key attributes.

CS3402 Tutorial 2:

1. Translate the ER diagram below to relational tables in the following steps.
   (a) Map *strong entity* type into relation
   (b) Map *weak entity + identifying relationship* type into relation
   (c) Map binary *1:1 relationship* types into attributes
   (d) Map binary *1:N Relationship* types into attributes
   (e) Map binary *M:N relationship* type into relation
   (f) Map *N-ary* relationship type into relation
   (g) Map *multi-valued* attribute into relation



2 Consider the following ER model with entities A and B (with attributes a and b) connected through a relationship.



2.1 Complete the table below by converting the ER model to relational schema, for all cardinality options. Write down the relations and underline their primary keys.

Hint: Map 1:1 relationship types into attributes; Map 1:N Relationship types into attributes; Map M:N relationship type into relation.

ER Model (X:Y)	Relational Schema
M:N	
1:N	
N:1	
1:1	

2.2 Suppose we want to add elements to the relations. Mark which tuples from below can be inserted into the relational schemas you created for the M:N relationship:
(a1, b1)
(a1, b2)
(a2, b1)
(a2, b2)

2.3 How about the 1:N case?
(a1, b1)
(a1, b2)
(a2, b1)
(a2, b2)

2.4 How about the 1:1 case?
(a1, b1)
(a1, b2)
(a2, b1)
(a2, b2)

CS3402 Tutorial 2:

1. **Answer**:

(a) Map *strong entity* type into relation
- Include simple (or atomic) attributes of the entity
- Include components of composite attributes
- Identify the primary key from the attributes
- Don't include: non-simple component of composite attributes, derived attributes, multi-valued attributes (not yet)

Assignment	**name**	**weights**

Student	**id**

Discussion	**cn**	**TA**

Worksheet	**week**

(b) Map *weak entity* + *identifying relationship* type into relation
- Include simple (or atomic) attributes
- Add the associated strong entity's primary key as attributes (also known as *foreign key* because it refers to another relation's primary key)
- Set the primary key as the combination of the *foreign* key and the partial key of the weak entity

Assignment	**name**	**weights**

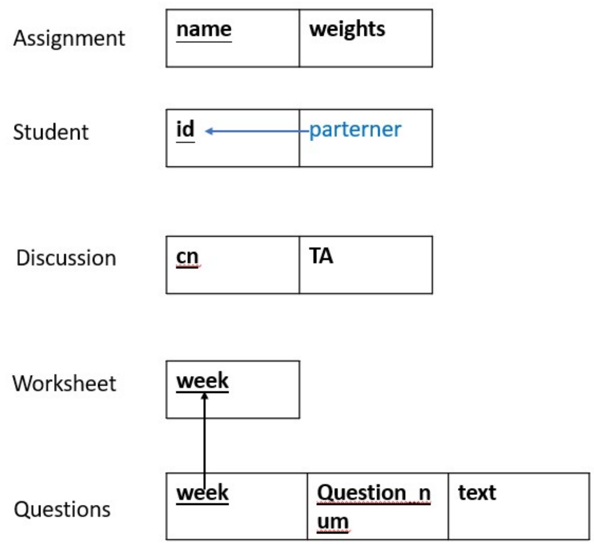
Student	**id**

Discussion	**cn**	**TA**

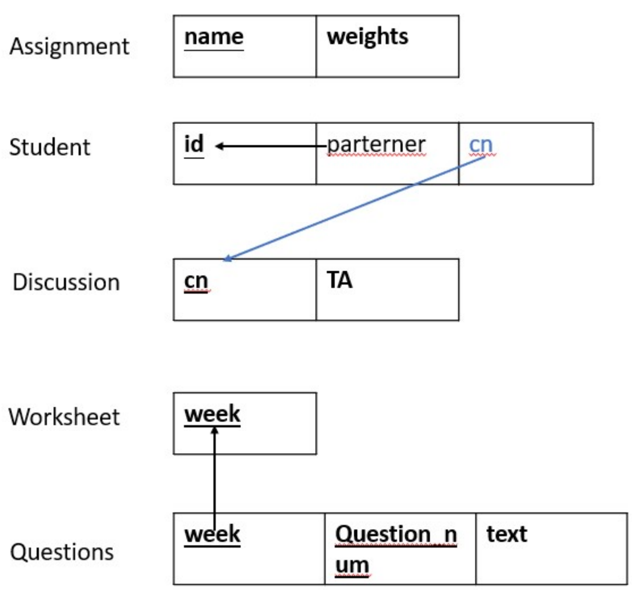
Worksheet	**week**

Questions	**week**	**Question_num**	**text**

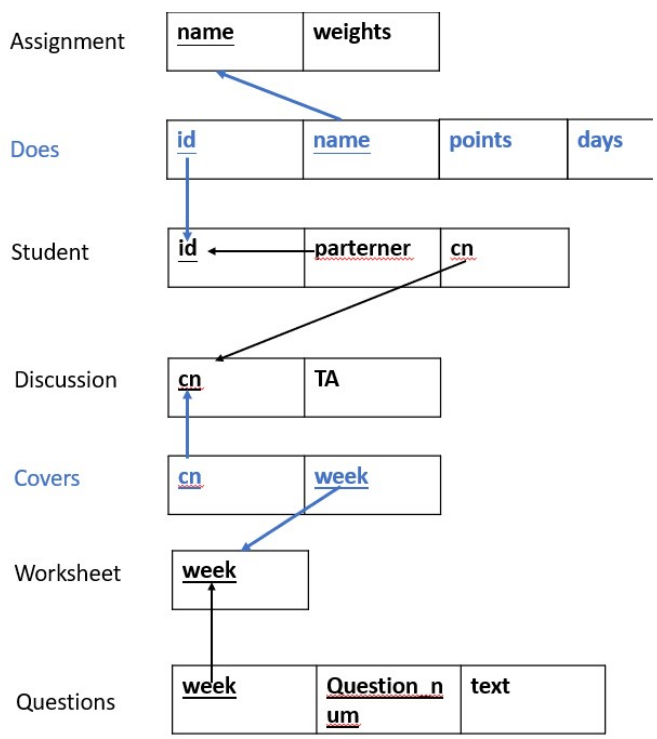
(c) Map binary *1:1 relationship* types into attributes
- Include the primary keys of one entity type as attributes (foreign keys) of the other entity type (*note:* it is better to choose the entity in total participation to include the other entity's key as attribute)
- Include also the simple attributes of the relationship type



(d) Map binary *1:N Relationship* types into attributes
- In the relation representing the *N-side* entity type, add the primary keys of the *1-side* entity type as attributes (foreign key)
- Include also the simple attributes of the relationship type



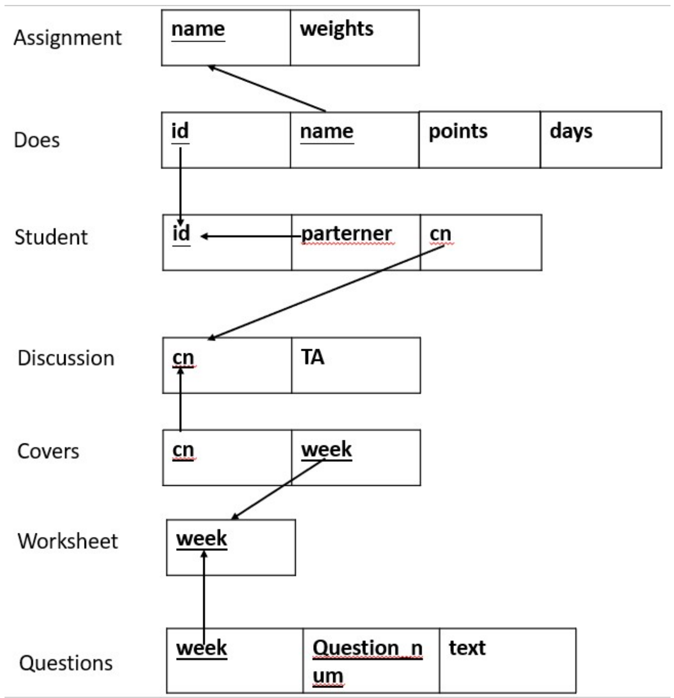
(e) Map binary *M:N relationship* type into relation
- Include the primary keys of the participating entity types as attributes (foreign key)
- Identify the primary key as the combination of the above foreign keys
- Include the simple attributes of the relationship type



(f) Map *N-ary* relationship type into relation
- Similar to binary *M:N* relationship type

(g) Map m*ulti-valued* attribute into relation
- Include the given attribute
- Include the primary attributes of the entity/relationship type owning the multivalued attribute
- Set the primary key to be the combination of foreign key and its original attribute

To summarize, the ER model will be translated into the following relational tables:



2 Answer:

2.1

ER Model (X:Y)	Relational Schema
M:N	A(a) B(b) rel(a,b)
1:N	A(a) B(b,a)
N:1	A(a,b) B(b)
1:1	A(a) B(b,a) or A(a,b) B(b)

2.2
√ (a1, b1)
√ (a1, b2)
√ (a2, b1)
√ (a2, b2)

2.3 How about the 1:N case?
√ (a1, b1)
√ (a1, b2)
(a2, b1)
(a2, b2)

OR
(a1, b1)
(a1, b2)
√ (a2, b1)
√ (a2, b2)

2.4 How about the 1:1 case?
√ (a1, b1)
(a1, b2)
(a2, b1)
√ (a2, b2)

OR

(a1, b1)
√ (a1, b2)
√ (a2, b1)
(a2, b2)

CS3402 Tutorial 3:

1. Suppose each of the following Update operations is applied directly to the database below. Discuss *all* integrity constraints violated by each operation, if any, and the different ways of enforcing these constraints:

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

(a) Insert < 'ProductA', 4, 'Bellaire', 2 > into PROJECT.

(b) Insert < 'Production', 4, '943775543', '01-OCT-88' > into DEPARTMENT.

(c) Insert < '677678989', null, '40.0' > into WORKS_ON.

(d) Delete the EMPLOYEE tuple with SSN= '987654321'.

(e) Delete the PROJECT tuple with PNAME= 'ProductX'.

(f) Modify the SUPERSSN attribute of the EMPLOYEE tuple with SSN= '999887777' to '943775543'.

Answers:

(a) Violates referential integrity because DNUM=2 and there is no tuple in the DEPARTMENT relation with DNUMBER=2. We may enforce the constraint by: (i) rejecting the insertion of the new PROJECT tuple, (ii) changing the value of DNUM in the new PROJECT tuple to an existing DNUMBER value in the DEPARTMENT relation, or (iii) inserting a new DEPARTMENT tuple with DNUMBER=2.

(b) Violates both the key constraint and referential integrity.
Violates the key constraint because there already exists a DEPARTMENT tuple with DNUMBER=4. We may enforce this constraint by: (i) rejecting the insertion, or (ii) changing the value of DNUMBER in the new DEPARTMENT tuple to a value that does not violate the key constraint.
Violates referential integrity because MGRSSN='943775543' and there is no tuple in the EMPLOYEE relation with SSN='943775543'. We may enforce the constraint by: (i) rejecting the insertion, (ii) changing the value of MGRSSN to an existing SSN value in EMPLOYEE, or (iii) inserting a new EMPLOYEE tuple with SSN='943775543'.

(c) Violates both the entity integrity and referential integrity.
Violates entity integrity because PNO, which is part of the primary key of WORKS_ON, is null. We may enforce this constraint by: (i) rejecting the insertion, or (ii) changing the value of PNO in the new WORKS_ON tuple to a value of PNUMBER that exists in the PROJECT relation.
Violates referential integrity because ESSN='677678989' and there is no tuple in the EMPLOYEE relation with SSN='677678989'. We may enforce the constraint by: (i) rejecting the insertion, (ii) changing the value of ESSN to an existing SSN value in EMPLOYEE, or (iii) inserting a new EMPLOYEE tuple with SSN='677678989'.

(d) Violates referential integrity because several tuples exist in the WORKS_ON, DEPENDENT, DEPARTMENT, and EMPLOYEE relations that reference the tuple being deleted from EMPLOYEE. We may enforce the constraint by: (i) rejecting the deletion, or (ii) deleting all tuples in the WORKS_ON, DEPENDENT, DEPARTMENT, and EMPLOYEE relations whose values for ESSN, ESSN, MGRSSN, and SUPERSSN, respectively, is equal to'987654321'.

(e) Violates referential integrity because two tuples exist in the WORKS_ON relations that reference the tuple being deleted from PROJECT. We may enforce the constraint by: (i) rejecting the deletion, or (ii) deleting the tuples in the WORKS_ON relation whose value for PNO=1 (the value for the primary key PNUMBER for the tuple being deleted from PROJECT).

(f) Violates referential integrity because the new value of SUPERSSN='943775543' and there is no tuple in the EMPLOYEE relation with SSN='943775543'. We may enforce the constraint by: (i) rejecting the modification, (ii) changing the value of SUPERSSN to an existing SSN value in EMPLOYEE, or (iii) inserting a new EMPLOYEE tuple with SSN='943775543'.

2 Consider the relation REFRIG(MODEL#, YEAR, PRICE, MANUF_PLANT, COLOR), which is abbreviated as REFRIG(M, Y, P, U, C), and the following set of F of functional dependencies: F={M -> U, {M,Y} -> P, U -> C}. Evaluate each of the following as a candidate key for REFRIG, giving reasons why it can or cannot be a key: {M}, {M,Y}, {M,C}

Answers:
- {M} IS NOT a candidate key since it does not functionally determine attributes Y or P.
{M}+={M,U,C}

- {M, Y} IS a super key since it functionally determines the remaining attributes P, U, and C.
Also {M} and {Y} are not the superkey, so {M,Y} IS a candidate key.
i.e.
We have {M, Y}-> P, and M->U, by augmentation {M, Y}->U
Since U->C, by transitivity M->U, U->C, gives M->C; By augmentation {M, Y} ->C
Thus {M, Y}+={M,Y, P, U,C} and {M, Y} can be a super key.
{M}+={M,U,C}, {M} is not super key.
{Y}+={Y}, {Y} is not super key.
So {M,Y} is the candidate key.

- {M, C} IS NOT a candidate key since it does not functionally determine attributes Y or P.
{M,C}+={M,C,U}.

CS3402 Tutorial 4:

1. Examine the table shown below.

Branch

Branch No	BranchAddress	TelNo
B001	8 Jefferson Way, Portland, OR 97201	503-555-3618, 503-555-2727, 503-555-6534
B002	City Center Plaza, Seattle, WA 98122	206-555-6756, 206-555-8836
B003	14 – 8th Avenue, New York, NY 10012	212-371-3000
B004	16 – 14th Avenue, Seattle, WA 98128	206-555-3131, 206-555-4112

(a) Why this table is not in 1NF?
(b) Describe and illustrate the process of normalizing the data shown in this table to third normal form (3NF).

Answer:
(a) *TelNo* is not an attribute with atomic values, but with multi-values. So, the table is NOT in 1NF.
(b) Create another relation specifically for *TelNo* with *BranchNo* as a foreign key

Branch

BranchNo	BranchAddress
B001	8 Jefferson Way, Portland, OR 97201
B002	City Center Plaza, Seattle, WA 98122
B003	14 – 8th Avenue, New York, NY 10012
B004	16 – 14th Avenue, Seattle, WA 98128

BranchTel

BranchNo	TelNo
B001	503-555-3618
B001	503-555-2727
B001	503-555-6534
B002	206-555-6756
B002	206-555-8836
B003	212-371-3000
B004	206-555-3131
B004	206-555-4112

2. Examine the table shown below.

StaffBranchAllocation

StaffNo	BranchNo	BranchAddress	Name	Position	HoursPer

					Week
S4555	B002	City Center Plaza, Seattle, WA 98122	Ellen Layman	Assistant	16
S4555	B004	16 – 14th Avenue, Seattle, WA 98128	Ellen Layman	Assistant	9
S4612	B002	City Center Plaza, Seattle, WA 98122	Dave Sinclair	Assistant	14
S4612	B004	16 – 14th Avenue, Seattle, WA 98128	Dave Sinclair	Assistant	10

<StaffNo, BranchNo> is the primary key.
<StaffNo> -> <Name, Position>; <BranchNo> -> <BranchAddress>

(a) Why this table is not in 2NF?
(b) Describe and illustrate the process of normalizing the data shown in this table to third normal form (3NF).

Answer:
(a) The primary key of StaffBranchAllocation table is *<StaffNo, BranchNo>*. There exist the partial functional dependencies: *StaffNo → Name, Position* and *BranchNo → BranchAddress*. The non-key attributes are not fully dependent on the key. So, the table is NOT in 2NF.
(b) Remove *BranchAddress*, *Name*, *Position* from StaffBranchAllocation relation to capture the partial functional dependencies separately.

Branch

BranchNo	BranchAddress
B002	City Center Plaza, Seattle, WA 98122
B004	16 – 14th Avenue, Seattle, WA 98128

Staff

StaffNo	Name	Position
S4555	Ellen Layman	Assistant
S4612	Dave Sinclair	Assistant

StaffBranchAllocation

StaffNo	BranchNo	HoursPerWeek
S4555	B002	16
S4555	B004	9
S4612	B002	14
S4612	B004	10

3. Examine the table shown below.

BranchManager

Branch No	BranchAddress	TelNo	MgrStaff No	MgrName
B001	8 Jefferson Way, Portland, OR 97201	503-555-3618	S1500	Tom Daniels
B002	City Center Plaza, Seattle, WA 98122	206-555-6756	S0010	Mary Martinez
B003	14 – 8th Avenue, New York, NY 10012	212-371-3000	S0145	Art Peters
B004	16 – 14th Avenue, Seattle, WA 98128	206-555-3131	S2250	Sally Stern

<BranchNo> is the primary key; <MgrStaffNo> -> <MgrName>

(a) Why this table is not in 3NF?

(b) Describe and illustrate the process of normalizing the data shown in this table to third normal form (3NF).

1. **Answer**:

(a) There exists a non-key attribute transitively dependent on the key, i.e., *MgrName* depends on *MgrStaffNo* and *MgrStaffNo* depends on *BranchNo.*

(b) Create another relation which specifically captures the dependency *MgrStaffNo* → *MgrName*

Branch

BranchNo	BranchAddress	TelNo	MgrStaffNo
B001	8 Jefferson Way, Portland, OR 97201	503-555-3618	S1500
B002	City Center Plaza, Seattle, WA 98122	206-555-6756	S0010
B003	14 – 8th Avenue, New York, NY 10012	212-371-3000	S0145
B004	16 – 14th Avenue, Seattle, WA 98128	206-555-3131	S2250

ManagerStaff

MgrStaffNo	MgrName
S1500	Tom Daniels
S0010	Mary Martinez
S0145	Art Peters
S2250	Sally Stern

4. Examine the table shown below and the set of functional dependency on its attributes：

CourseRmAlloc (CourseId, CourseName, Year, Lecturer, Enrollment, RoomId, RoomCapacity, Day, Time)

FD = {*CourseId -> CourseName,* *CourseName -> CourseId,*

CourseId, Year -> Lecturer, *CourseId, Year -> Enrollment,*

RoomId -> RoomCapacity, *RoomId, Year, Day, Time -> CourseId,*

CourseId, Year, Day, Time -> RoomId }

(a) Find all candidate keys of this table.
(b) Decompose this table into a design into BCNF.

Answer:

(a) There are three candidate keys in this table:
(Year, Day, Time, CourseId)
(Year, Day, Time, CourseName)
(Year, Day, Time, RoomId)

(b) This table can be decomposed into the following in BCNF (so also in 3NF):
CourseTeaching (CourseId, Year, Lecturer, Enrollment)
Room (RoomId, RoomCapacity)
CourseRoomAlloc (CourseId, Year, Day, Time, RoomId)
Course (CourseId, CourseName)

CS3402 Tutorial 5:

1. **Answer**:

(a) T1 $\times$ T2

P	Q	R1	A	B	R2
10	a	5	10	b	6
10	a	5	25	c	3
10	a	5	10	b	5
15	b	8	10	b	6
15	b	8	25	c	3
15	b	8	10	b	5
25	a	6	10	b	6
25	a	6	25	c	3
25	a	6	10	b	5

(b) T1 $\bowtie_{T1.P=T2.A}$T2

P	Q	R1	A	B	R2
10	a	5	10	b	6
10	a	5	10	b	5
25	a	6	25	c	3

(c) T1 $\bowtie_{T1.Q=T2.B}$T2

P	Q	R1	A	B	R2
15	b	8	10	b	6
15	b	8	10	b	5

(d) T1 $\bowtie_{T1.R>T2.R}$T2

P	Q	R1	A	B	R2
15	b	8	10	b	6
15	b	8	25	c	3
15	b	8	10	b	5
25	a	6	25	c	3
25	a	6	10	b	5
10	a	5	25	c	3

(e) T1 $*$ T2

P	Q	R	A	B
10	a	5	10	b
25	a	6	10	b

2. **Answer**:

(a) Find the SSn (social security number) of all employees who are not supervisors

π_{ssn} (EMPLOYEE) $- \pi_{Super\_ssn}$ (EMPLOYEE)

(b) Find the SSn of all employees who either work in department 5 or directly supervise an employee who works in department 5

π_{Ssn} ($\sigma_{Dno=5}$ (EMPLOYEE)) $\cup$ π_{Super_ssn} ($\sigma_{Dno=5}$ (EMPLOYEE))

(c) List the names and numbers of all departments locating in 'Houston'

$\pi_{Dname,Dnumber}$ ($\sigma_{Dlocation='Houston'}$ (DEPARTMENT $*$ DEPT_LOCATIONS))

(d) List the first names of all employees who have a dependent with the same first name as themselves

π_{Fname} (EMPLOYEE $\bowtie_{Ssn=Essn \textbf{ AND } Fname=Depende\_name}$ DEPENDENT))

(e) Retrieve the salary of all employees in department 5 who work more than 10 hours per week on the project named 'ProjectX'

WORK_PROJ $\leftarrow$ WORKS_ON $\bowtie_{Pnumber=Pno}$ PROJECT

PROJECTX10 $\leftarrow$ $\sigma_{Pname='ProjectX' \text{ AND Hours}>10}$ (WORK_PROJ)

π_{Salary} (PROJECTX10 $\bowtie_{Essn=Ssn \text{ AND Dno}=5}$EMPLOYEE)

CS3402 Tutorial 6 Solutions:

1.

512/8 = 64 blocks
Sequential search = 64/2 = 32
Binary search = $\log_2 64 = \log_2 2^6 = 6$

2.

(a) Static hashing with 5 buckets, each of which contains at most 3 records

Bucket 0

3760	
7115	
	NULL

Bucket 1

4871	
1821	
	NULL

Bucket 2

4692	
	NULL

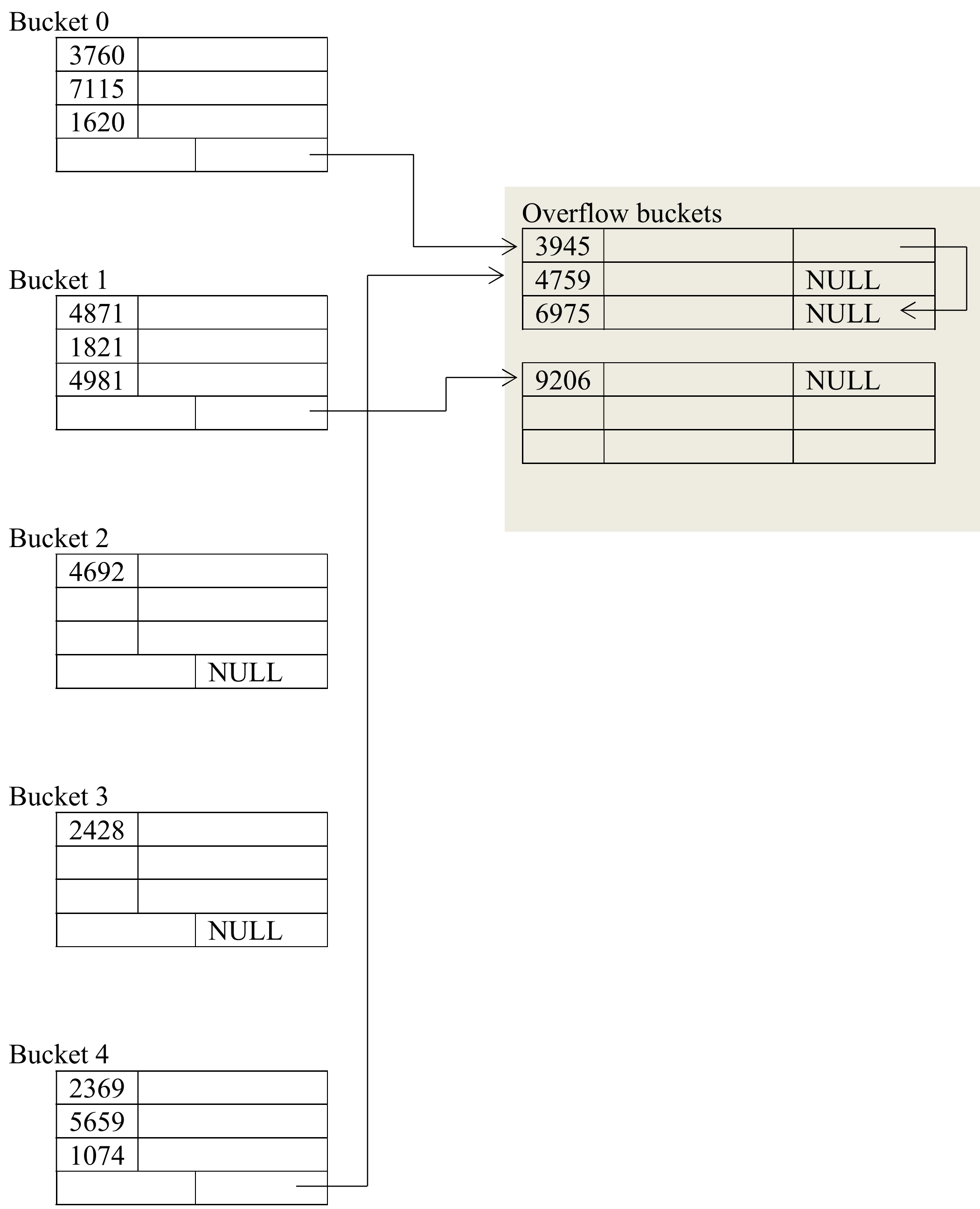
Bucket 3

	NULL

Bucket 4

2369	
5659	
1074	
	NULL

(b) Overflow handling

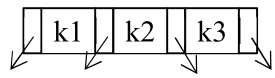


3. 8 hash codes (000, 001, 010, 100, 011, 110, 101, 111)

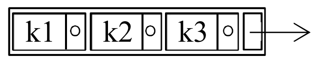
4.
a) R = 40+8+4+65+8+8+1+7+6+2+1 = 150 bytes.
b) The blocking factor: brf=floor(1024/150)=6 records
c) The file will occupy ceil (1000 / brf)= ceil (1000/6)=167 blocks

CS3402 Tutorial 7:

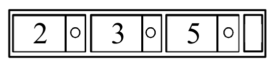
1. **Answer**:
   - When the number of key value in internal nodes is 3, a full internal node of this B+ tree will look like:



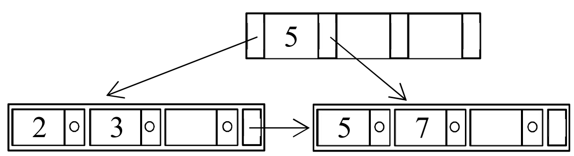
   - When the number of key value in leaf nodes is 3, a full leaf node of this B+ tree will look like:



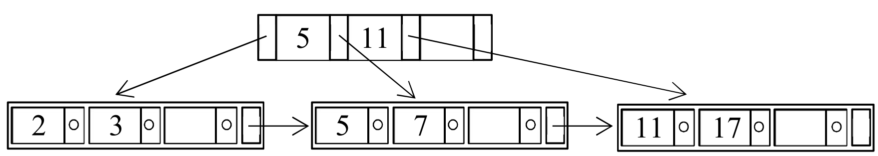
   - After inserting 2, 3, 5, the tree looks like



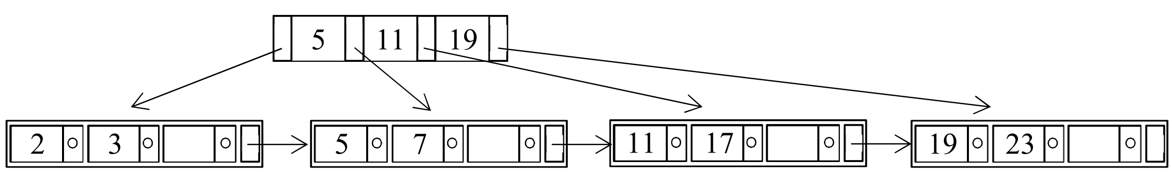
   - After inserting 7, the tree looks like



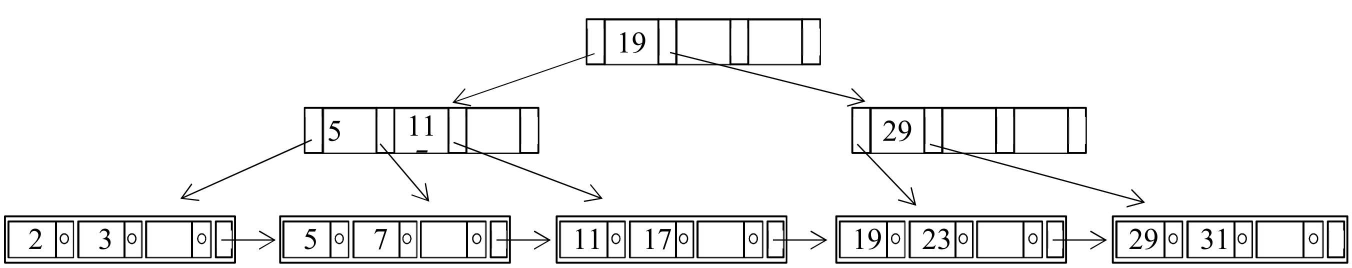
   - After inserting 11, 17, the tree looks like



   - After inserting 19, 23, the tree looks like

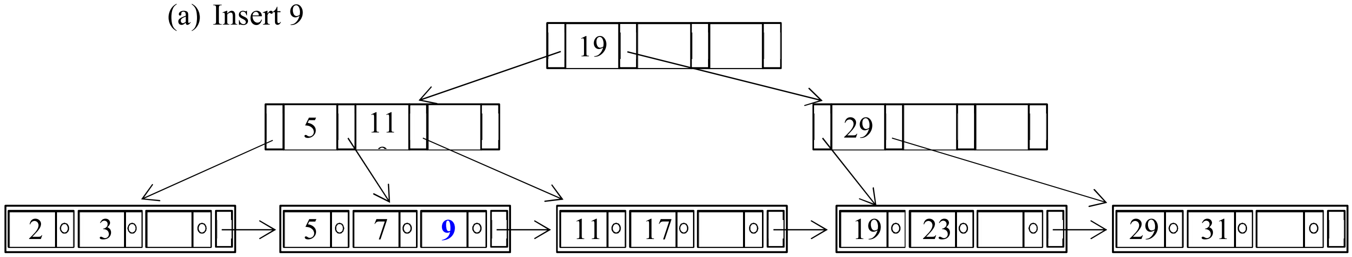


   - After inserting 29, 31, the tree looks like

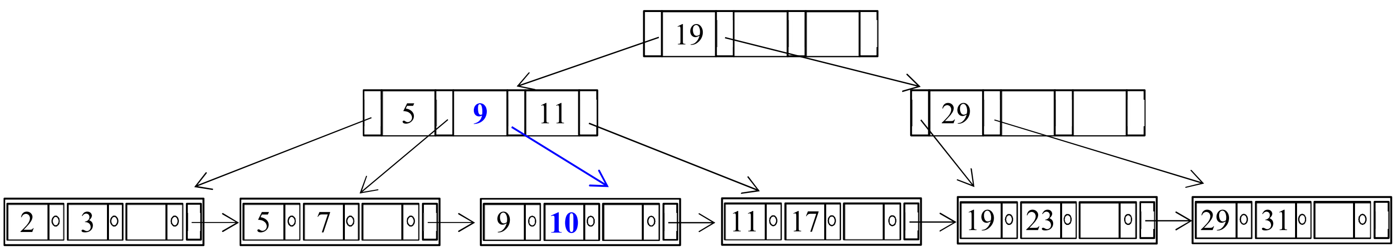


2. **Answer**:

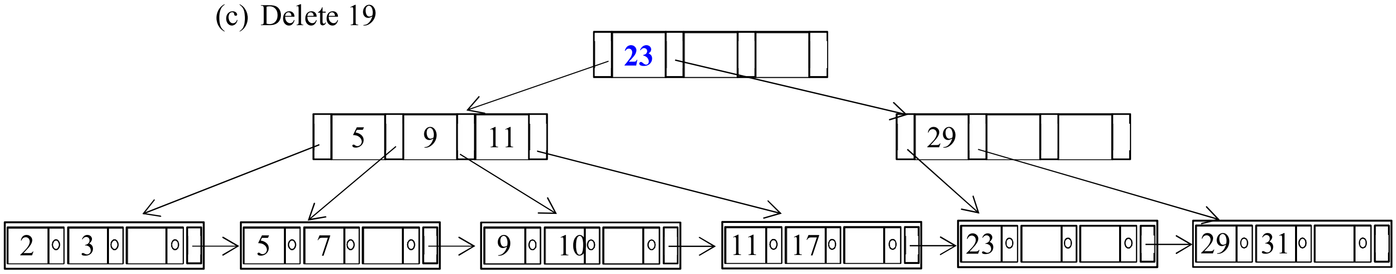
(a) Insert 9



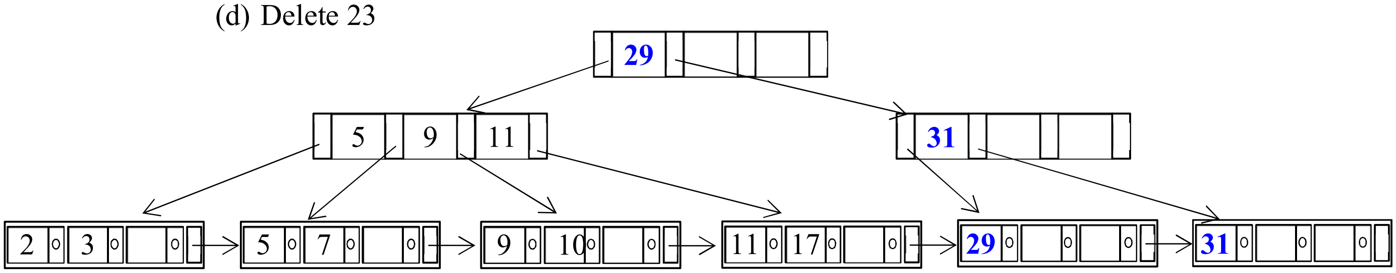
(b) Insert 10



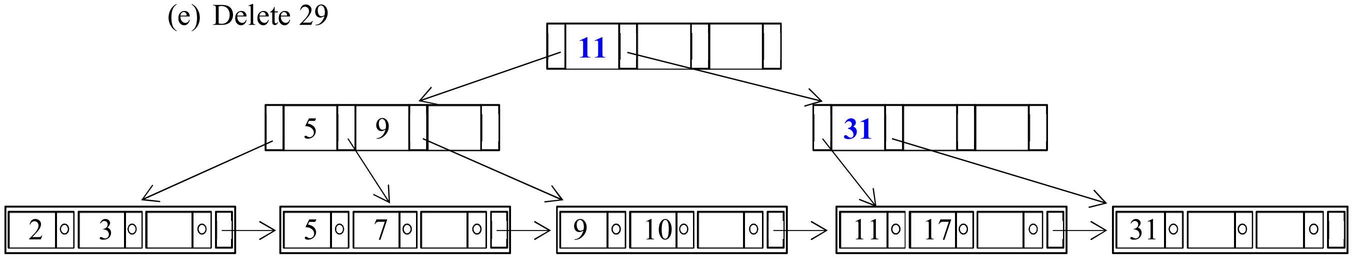
(c) Delete 19



(d) Delete 23



(e) Delete 29



3.

(a) Answer:

Record length R = 32 + 10 + 8 + 40 + 8 + 8 + 1) = 107 bytes

Blocking factor bfr = floor(B/R) = floor(512/107) = 4 records per block

Number of file blocks Nb= ceil(10,000 / 4) =2,500 blocks

(b)

Number of single-level index entries =number of file blocks Nb = 2,500 entries

Index entry size Ri = (V_ID+ P) = (10+ 6) = 16 bytes

Index blocking factor bfr_i = floor(B/R_i) = floor(512/16) = 32 entries per block

Number of index blocks Nb_i=ceil(Nb/bfr_i)= ceil(2,500 / 32) =79 blocks