数据库系统原理

习题汇总

Wengen Li

Email: lwengen@tongji.edu.cn

Department of Computer Science and Technology

Tongji University

Assignment 1: Relation Algebra

- **Exercises**
  - 2.1, 2.7, 2.8, 2.9, 2.12, 2.13
  - 6.11, 6.14

Assignment 1: Relation Algebra

- 2.1 考虑图2-14所示关系数据库。这些关系上适当的主码是什么？

  employee (person-name, street, city)

  works (person-name, company-name, salary)

  company (company-name, city)

- 2.7考虑图2-14所示关系数据库。给出关系代数表达式来表示下列每一个查询：
  - a. 找出居住在“Miami”城市的所有员工姓名；
  - b. 找出工资在100,000美元以上的所有员工姓名；
  - c. 找出居住在“Miami”并且工资在100,000美元以上的所有员工姓名。

  a. $\Pi_{name}\left(\sigma_{city=\text{“Miami”}}(employee)\right)$

  b. $\Pi_{name}\left(\sigma_{salary>100000}(employee)\right)$

  c. $\Pi_{name}\left(\sigma_{city=\text{“Miami”}\wedge salary>100000}(employee)\right)$

Assignment 1: Relation Algebra

- 2.8 考虑图2-15所示银行数据库。对于下列每个查询，给出一个关系代数表达式：
  - a. 找出位于“Chicago”的所有支行名字
  - b. 找出在支行“Downtown”有贷款的所有贷款人姓名

a. $\Pi_{branch\_name}(\sigma_{branch\_city = \text{“Chicago”}}(branch))$

b. $\Pi_{customer\_name}(\sigma_{branch\_name = \text{“Downtown”}}(borrower \bowtie loan))$

branch(branch_name, branch_city, assets)
customer (customer_name, customer_street, customer_city)
loan (loan_number, branch_name, amount)
borrower (customer_name, loan_number)
account (account_number, branch_name, balance)
depositor (customer_name, account_number)

图 2-15　习题 2.8、习题 2.9 和习题 2.13 的银行数据库

Assignment 1: Relation Algebra

- 2.9 考虑图2-15所示银行数据库。
  - a. 适当的主码是什么？
  - b. 给出你选择的主码，确定适当的外码。

branch(branch_name, branch_city, assets)
customer(customer_name, customer_street, customer_city)
load(load_number, branch_name, amount)
borrower(customer_name, load_number)
account(account_number, branch_name, balance)
depositor(customer_name, account_number)

Assignment 1: Relation Algebra

- 2.12 考虑图2-14所示关系数据库。给出关系代数表达式来表示下列每一个查询：
  - a. 找出为“First Bank Corporation”工作的所有员工姓名；
  - b. 找出为“First Bank Corporation”工作的所有员工姓名和居住城市；
  - c. 找出为“First Bank Corporation”工作且挣钱超过10000美元的所有员工姓名、街道地址和居住城市。

a. $\Pi_{person\_name}(\sigma_{company\_name = \text{“First Bank Corporation”}}(works))$

b. $\Pi_{person\_name, city}(employee \bowtie (\sigma_{company\_name = \text{“First Bank Corporation”}}(works)))$

c. $\Pi_{person\_name, street, city}(\sigma_{(company\_name = \text{“First Bank Corporation”} \wedge salary > 10000)}(works \bowtie employee))$

employee (person-name, street, city)

works (person-name, company-name, salary)

company (company-name, city)

Assignment 1: Relation Algebra

- 2.13 考虑图2-15所示银行数据库。对于下列每个查询，给出一个关系代数表达式：
  - a. 找出贷款额度超过10000美元的所有贷款号；
  - b. 找出所有这样的存款人姓名，他拥有一个存款额大于6000美元的账户；
  - c. 找出所有这样的存款人姓名，他在“Uptown”支行拥有一个存款额大于6000美元的账户。

a. $\Pi_{loan\_number}\ (\sigma_{amount > 10000}(loan))$

b. $\Pi_{customer\_name}\ (\sigma_{balance > 6000}\ (depositor \bowtie account))$

c. $\Pi_{customer\_name}\ (\sigma_{balance > 6000 \wedge branch\_name = \text{“Uptown”}}\ (depositor \bowtie account))$

branch(branch_name, branch_city, assets)
customer(customer_name, customer_street, customer_city)
load(load_number, branch_name, amount)
borrower(customer_name, load_number)
account(account_number, branch_name, balance)
depositor(customer_name, account_number)

Assignment 1: Relation Algebra

- 6.11 考虑图6-22所示关系数据库，主码加了下划线。给出关系代数表达式来表示下列每一个查询：
  - a. 找出First Bank Corporation的所有员工姓名；
  - b. 找出First Bank Corporation所有员工的姓名和居住城市；
  - c. 找出First Bank Corporation所有年收入在10000美元以上的员工姓名和居住的街道、城市；
  - d. 找出所有居住地与工作的公司在同一城市的员工姓名；
  - e. 假设公司可以位于几个城市中。找出满足下面条件的所有公司，它位于Small Bank Corporation所位于的每一个城市

employee (person-name, street, city)
works (person-name, company-name, salary)
company (company-name, city)
Manages(person_name, manager_name)

a. $\Pi_{person\_name}(\sigma_{company\_name = \text{“First Bank Corporation”}}(works))$

b. $\Pi_{person\_name, city}(employee \bowtie (\sigma_{company\_name = \text{“First Bank Corporation”}}(works)))$

c. $\Pi_{person\_name, street, city}(\sigma_{(company\_name = \text{“First Bank Corporation”} \wedge salary > 10000)}(works \bowtie employee))$

d. $\Pi_{person\_name}(employee \bowtie works \bowtie company)$

e. Note: Small Bank Corporation will be included in each answer.
$\Pi_{company\_name}(company \div (\Pi_{city}(\sigma_{company\_name = \text{“Small Bank Corporation”}}(company))))$

Assignment 1: Relation Algebra

6. 14 考虑如下关于图书馆的关系模式：

$$memeber(\underline{memb\_no}, name, dob)$$
$$books(\underline{isbn}, title, authors, publisher)$$
$$borrowed(\underline{memb\_no}, \underline{isbn}, date)$$

用关系代数写出下列查询：

a. 找出借了任何由 McGraw-Hill 出版的书的成员的姓名。

b. 找出借了由 McGraw-Hill 出版的所有的书的成员的姓名。

c. 找出借了由 McGraw-Hill 出版的 5 本以上不同的书的成员的姓名和成员号。

d. 对每个出版商，找出借了该出版商的 5 本以上的书的成员的姓名和成员号。

e. 找出平均每个成员借了多少本书。下面的情况需要考虑在内，如果某个成员没有借任何书，那么他就根本不会出现在关系 *borrowed* 中。

a. $t_1 \leftarrow \Pi_{isbn}(\sigma_{publisher="McGraw-Hill"}(books))$
$\Pi_{name}((member \bowtie borrowed) \bowtie t_1))$

b. $t_1 \leftarrow \Pi_{isbn}(\sigma_{publisher="McGraw-Hill"}(books))$
$\Pi_{name,isbn}(member \bowtie borrowed) \div t_1$

c. $t_1 \leftarrow member \bowtie borrowed \bowtie (\sigma_{publisher="McGraw-Hill"}(books))$
$\Pi_{name}\ (\sigma_{countisbn > 5}\ ((\ memb\_no\ G_{\textbf{count-distinct}(isbn)\ \textbf{as}\ countisbn}(t_1))))$

d. $t_1 \leftarrow member \bowtie borrowed \bowtie books$
$\Pi_{publisher,name}\ (\sigma_{countisbn > 5}\ ((\ publisher,memb\_no\ G_{\textbf{count-distinct}(isbn)\ \textbf{as}\ countisbn}(t_1)))$

Assignment 2: SQL

- **Exercises**
  - 3.8, 3.9, 3.10 (选择其中两个)
  - 3.15, 3.16, 3.17, 3.21 (选择其中两个)

- **注意：**SQL语句可以有不同的形式，后续答案仅供参考！

Assignment 2: SQL

3.8 考虑图 3-19 中的银行数据库，其中加下划线的是主码。为这个关系数据库构造出如下 SQL 查询：

a. 找出银行中所有有账户但无贷款的客户。

b. 找出与“Smith”居住在同一个城市、同一个街道的所有客户的名字。

c. 找出所有支行的名称，在这些支行中都有居住在“Harrison”的客户所开设的账户。

branch (*branch_name*, *branch_city*, *assets*)
customer (*customer_name*, *customer_street*, *customer_city*)
loan (*loan_number*, *branch_name*, *amount*)
borrower (*customer_name*, *loan_number*)
account (*account_number*, *branch_name*, *balance*)
depositor (*customer_name*, *account_number*)

a

```
(select customer_name
from depositor)
except
(select customer_name
from borrower)
```

b

```
select F.customer_name
from customer F join customer S using(customer_street, customer_city)
where S.customer_name = 'Smith'
```

c

```
select distinct branch_name
from account natural join depositor natural join customer
where customer_city = 'Harrison'
```

Assignment 2: SQL

3.9 考虑图 3-20 的雇员数据库，其中加下划线的是主码。为下面每个查询写出 SQL 表达式：

a. 找出所有为“First Bank Corporation”工作的雇员名字及其居住城市。

b. 找出所有为“First Bank Corporation”工作且薪金超过 10 000 美元的雇员名字、居住街道和城市。

c. 找出数据库中所有不为“First Bank Corporation”工作的雇员。

d. 找出数据库中工资高于“Small Bank Corporation”的每个雇员的所有雇员。

e. 假设一个公司可以在好几个城市有分部。找出位于“Small Bank Corporation”所有所在城市的所有公司。

f. 找出雇员最多的公司。

g. 找出平均工资高于“First Bank Corporation”平均工资的那些公司。

employee(*employee_name*, *street*, *city*)
works(*employee_name*, *company_name*, *salary*)
company(*company_name*, *city*)
managers(*employee_name*, *manager_name*)

图 3-20 习题 3.9、习题 3.10、习题 3.16、习题 3.17 和习题 3.20 的雇员数据库

a

```
select e.employee_name, city
from employee e, works w
where w.company_name = 'First Bank Corporation' and
      w.employee_name = e.employee_name
```

b

```
select *
from employee
where employee_name in
    (select employee_name
     from works
     where company_name = 'First Bank Corporation' and salary > 10000)
```

c

The following solution assumes that all people work for exactly one company.

```
select employee_name
from works
where company_name ≠ 'First Bank Corporation'
```

d

The following solution assumes that all people work for at most one company.

```
select employee_name
from works
where salary > all
    (select salary
     from works
     where company_name = 'Small Bank Corporation')
```

e

```
select S.company_name
from company S
where not exists ((select city
                   from company
                   where company_name = 'Small Bank Corporation')
                  except
                    (select city
                     from company T
                     where S.company_name = T.company_name))
```

f

```
select company_name
from works
group by company_name
having count (distinct employee_name) >= all
    (select count (distinct employee_name)
     from works
     group by company_name)
```

Assignment 2: SQL

g

```
select company_name
from works
group by company_name
having avg (salary) > (select avg (salary)
                       from works
                       where company_name = 'First Bank Corporation')
```

Assignment 2: SQL

3.10 考虑图 3-20 的关系数据库，给出下面每个查询的 SQL 表达式：

a. 修改数据库使“Jones”现在居住在“Newtown”市。

b. 为“First Bank Corporation”所有工资不超过 100 000 美元的经理增长 10% 的工资，对工资超过 100 000 美元的只增长 3%。

a

```
update employee
set city = 'Newton'
where person_name = 'Jones'
```

b

```
update works T
set T.salary = T.salary * 1.03
where T.employee_name in (select manager_name
                          from manages)
      and T.salary * 1.1 > 100000
      and T.company_name = 'First Bank Corporation'

update works T
set T.salary = T.salary * 1.1
where T.employee_name in (select manager_name
                          from manages)
      and T.salary * 1.1 <= 100000
      and T.company_name = 'First Bank Corporation'
```

Assignment 2: SQL

3.15 考虑图 3-19 中的银行数据库，其中加下划线的是主码。为这个关系数据库构造出如下 SQL 查询：

a. 找出在“Brooklyn”的所有支行都有账户的所有客户。

b. 找出银行的所有贷款额的总和。

c. 找出总资产至少比位于 Brooklyn 的某一家支行要多的所有支行名字。

a

```
with branchcount as
    (select count(*)
    branch
    where branch_city = 'Brooklyn')
select customer_name
from customer c
where branchcount =
    (select count(distinct branch_name)
    from (customer natural join depositor natural join account
        natural join branch) as d
    where d.customer_name = c.customer_name)
```

b

```
select sum(amount)
from loan
```

c

```
select branch_name
from branch
where assets > some
    (select assets
    from branch
    where branch_city = 'Brooklyn')
```

Assignment 2: SQL

3. 16 考虑图 3-20 中的雇员数据库，其中加下划线的是主码。给出下面每个查询对应的 SQL 表达式：

a. 找出所有为“First Bank Corporation”工作的雇员名字。

b. 找出数据库中所有居住城市和公司所在城市相同的雇员。

c. 找出数据库中所有居住的街道和城市与其经理相同的雇员。

d. 找出工资高于其所在公司雇员平均工资的所有雇员。

e. 找出工资总和最小的公司。

a

```
select employee_name
from works
where company_name = 'First Bank Corporation'
```

b

```
select e.employee_name
from employee e, works w, company c
where e.employee_name = w.employee_name and e.city = c.city and
      w.company_name = c.company _name
```

c

```
select P.employee_name
from employee P, employee R, manages M
where P.employee_name = M.employee_name and
      M.manager_name = R.employee_name and
      P.street = R.street and P.city = R.city
```

d

```
select employee_name
from works T
where salary > (select avg (salary)
                from works S
                where T.company_name = S.company_name)
```

e

```
select company_name
from works
group by company_name
having sum (salary) <= all (select sum (salary)
                            from works
                            group by company_name)
```

Assignment 2: SQL

3. 17 考虑图 3-20 中的关系数据库。给出下面每个查询对应的 SQL 表达式：

a. 为“First Bank Corporation”的所有雇员增长 10% 的工资。

b. 为“First Bank Corporation”的所有经理增长 10% 的工资。

c. 删除“Small Bank Corporation”的雇员在 *works* 关系中的所有元组。

a

```
update works
set salary = salary * 1.1
where company_name = 'First Bank Corporation'
```

b

```
update works
set salary = salary * 1.1
where employee_name in (select manager_name
                        from manages)
      and company_name = 'First Bank Corporation'
```

c

```
delete from works
where company_name = 'Small Bank Corporation'
```

Assignment 2: SQL

3.21 考虑图 3-21 中的图书馆数据库。用 SQL 写出如下查询：

a. 打印借阅了任意由“McGraw-Hill”出版的书的会员名字。

b. 打印借阅了所有由“McGraw-Hill”出版的书的会员名字。

c. 对于每个出版商，打印借阅了多于五本由该出版商出版的书的会员名字。

d. 打印每位会员借阅书籍数量的平均值。考虑这样的情况：如果某会员没有借阅任何书籍，那么该会员根本不会出现在 *borrowed* 关系中。

member(*memb_no*, *name*, *age*)
book(*isbn*, *title*, *authors*, *publisher*)
borrowed(*memb_no*, *isbn*, *date*)

图 3-21 习题 3.21 的图书馆数据库

a

```
select name
from member m, book b, borrowed l
where m.memb_no = l.memb_no
      and l.isbn = b.isbn and
          b.publisher = 'McGrawHill'
```

b

```
select distinct m.name
from member m
where not exists
      ((select isbn
      from book
      where publisher = 'McGrawHill')
      except
      (select isbn
      from borrowed l
      where l.memb_no = m.memb_no))
```

c

```
select publisher, name
from (select publisher, name, count (isbn)
      from member m, book b, borrowed l
      where m.memb_no = l.memb_no
      and l.isbn = b.isbn
      group by publisher, name) as
      membpub(publisher, name, count_books)
where count_books > 5
```

d

```
with memcount as
      (select count(*)
      from member)
select count(*)/memcount
from borrowed
```

Assignment 3: SQL

- **Exercises**
  - 4.7, 4.16

Assignment 3: SQL

4.7 考虑图 4-11 所示的关系数据库。给出这个数据库的 SQL DDL 定义。指出应有的参照完整性约束，并把它们包括在 DDL 定义中。

employee(*employee_name*, *street*, *city*)
works(*employee_name*, *company_name*, *salary*)
company(*company_name*, *city*)
manages(*employee_name*, *manager_name*)

图 4-11 习题 4.7 和习题 4.12 的雇员数据库

根据table定义语句书写即可，注意标明主码和外码！

Assignment 3: SQL

4.16 如本章定义的参照完整性约束正好涉及两个关系。考虑包括如图 4-12 所示关系的数据库。假设我们希望要求每个出现在 *address* 中的名字必须出现在 *salaried_worker* 或者 *hourly_worker* 中，但不一定要求在两者中同时出现。

a. 给出表达这种约束的语法。

b. 讨论为了使这种形式的约束生效，系统必须采取什么行动。

salaried_worker (*name*, *office*, *phone*, *salary*)
hourly_worker (*name*, *hourly_wage*)
address (*name*, *street*, *city*)

图 4-12 习题 4.16 的雇员数据库

a. For simplicity, we present a variant of the SQL syntax. As part of the **create table** expression for *address* we include

foreign key (*name*) **references** *salaried_worker* **or** *hourly_worker*

b. To enforce this constraint, whenever a tuple is inserted into the *address* relation, a lookup on the *name* value must be made on the *salaried_worker* relation and (if that lookup failed) on the *hourly_worker* relation (or vice-versa).

Quiz 1: Relational DB Design

- **Given the relational schema R<U, F>, U={A,B,C,D,E }, F={AC→BD, B → C, C → D, B → E}**
  a) Use Armstrong axioms and related rules to prove the functional dependency AC → E
  b) Compute $(A)^+$ and $(AC)^+$
  c) Find a canonical cover F_c of F
  d) Find all candidate keys, and point out R is in which normal form
  e) Decompose R into 3NF such that the decomposition is lossless-join and dependency preserving
  f) *Give related explanation or proof that the above decomposition is lossless-join and dependency preserving
  g) *Decompose the relation into relations in BCNF

Quiz 1: Relational DB Design

a) 由AC→ BD得AC→ B(分解规则)；再由B→ E，则有AC→E（传递规则）

b) $(A)^+ = A$, $(AC)^+ = ABCDE$

c) 对R<U, F>中的函数依赖集F进行极小化处理，得最小依赖集Fc={AC→B, B→CE, C→D}，仍记为F（可以有多种答案）

d) R的候选码有：AC、AB；主属性为A、B、C。由C→D可见，非主属性D对码AC为部分函数依赖，故R∉2NF, R∈1NF

e) 将关系模式R分解为3NF：全部属性均在F中出现了；不存在X→A∈ F，且XA=U。则对F按相同左部原则分组，有

U1={A,B,C}, F1={AC → B, B → C }

U2={B,C,E}, F2={B → C, B → E }

U3={C,D}, F3={C → D }

分解ρ={R1<U1,F1>, R2<U2,F2>, R3<U3,F3>}为保持函数依赖的分解。

f) 由于码AC、AB都包含在U1中，由检测算法可以找到相应表中的一行可以成为a1, a2, a3, a4, a5（用表格法），分解ρ同时具有无损连接性。

g) U1={A, B, C, D}, U2={B, C, E}, U2={C, D}。

Assignment 4: Relational DB Design

- R<U,F>, U={A,B,C,D,E}, F={A→C, B→C, C→D, DE→C, CE→A}, and a decomposition of R: ρ={R1(A,D), R2(A,B), R3(B,E), R4(C,D,E), R5(A,E)}. ρ is a lossless-join decomposition or a lossy one ?
  - ρ is a lossless-join decomposition

	A	B	C	D	E
R1(A,D)	a1	b12	b13	a4	b15
R2(A,B)	a1	a2	b23	b24	b25
R3(B,E)	b31	a2	b33	b34	a5
R4(C,D,E)	b41	b42	a3	a4	a5
R5(A,E)	a1	b52	b53	b54	a5

	A	B	C	D	E
R1(A,D)	a1	b12	b13→a3	a4	b15
R2(A,B)	a1	a2	b23→a3	b24→a4	b25
R3(B,E)	b31→a1	a2	b33→a3	b34→a4	a5
R4(C,D,E)	b41→a1	b42	a3	a4	a5
R5(A,E)	a1	b52	b53→a3	b54→a4	a5

Assignment 4: Relational DB Design

- R<U,F>, U={A,B,C,D}, F={A→B, A→C, C→D}, ρ={R1(A,B), R2(B,C), R3(C,D)}. ρ is a lossless-join decomposition or a lossy one?
  - ρ is a lossy-join decomposition

	A	B	C	D
R1(A,B)	a1	a2	b13	b14
R2(B,C)	b21	a2	a3	b24
R3(C,D)	b31	b32	a3	a4

	A	B	C	D
R1(A,B)	a1	a2	b13	b14
R2(B,C)	b21	a2	a3	b24→a4
R3(C,D)	b31	b32	a3	a4

Assignment 5: Relational DB Design

- **Exercises (不提交)**
  - 8.1, 8.6, 8.27
  - 8.29 (a, b, c, d)

Assignment 5: Relational DB Design

8.1 假设我们将模式 $r(A, B, C, D, E)$ 分解为

$$r_1(A, B, C)$$
$$r_2(A, D, E)$$

证明该分解是无损分解，如果如下函数依赖集 F 成立：

$$A \rightarrow BC \qquad CD \rightarrow E \qquad B \rightarrow D \qquad E \rightarrow A$$

参考答案：

	A	B	C	D	E
r1	a1	a2	a3	b14	b15
r2	a1	b22	b23	a4	a5

	A	B	C	D	E
r1	a1	a2	a3	b14	b15
r2	a1	~~b22~~ a2	~~b23~~ a3	a4	a5

	A	B	C	D	E
r1	a1	a2	a3	~~b14~~ a4	b15
r2	a1	a2	a3	a4	a5

	A	B	C	D	E
r1	a1	a2	a3	a4	~~b15~~ a5
r2	a1	a2	a3	a4	a5

Assignment 5: Relational DB Design

8.6 计算关于关系模式 $r(A, B, C, D, E)$ 的如下函数依赖集 F 的闭包。

$$A \rightarrow BC$$
$$CD \rightarrow E$$
$$B \rightarrow D$$
$$E \rightarrow A$$

列出 R 的候选码。

参考答案：

F的闭包可以根据Armstrong’s Axioms逐个推导，也可以通过属性的闭包计算。不要求推出所有函数依赖，熟悉推导过程即可。

R的候选码为：A, BC, CD和E

Assignment 5: Relational DB Design

8.27　用实践习题 8.6 中的函数依赖计算 B^+。

参考答案：BD

Assignment 5: Relational DB Design

8.29 考虑如下关系模式 $r(A, B, C, D, E, F)$ 上的函数依赖集 F：

$$A \rightarrow BCD$$
$$BC \rightarrow DE$$
$$B \rightarrow D$$
$$D \rightarrow A$$

a. 计算 B^+。

b. (使用 Armstrong 公理)证明 AF 是超码。

c. 计算上述函数依赖集 F 的正则覆盖；给出你的推导的步骤并解释。

d. 基于正则覆盖，给出 r 的一个 3NF 分解。

e. 利用原始的函数依赖集，给出 r 的一个 BCNF 分解。

f. 你能否利用正则覆盖得到与上面的 r 相同的 BCNF 分解?

参考答案：

a. B^+=ABCDE

b. With A→BCD and BC→DE, we have A→BCDE. Further, we have AF→ABCDEF. 因此，AF是超码。

c. A→BC, B→DE, D→A

d. r1(A, B, C), r2(B, D, E), r3(D, A), r4(A, F)

e. r1(A, B, C, D), r2(A, F), r3(A, E)

f. 不能利用正则覆盖得到与上面r相同的分解。不过可以通过正则覆盖导出原始的依赖关系，进而得到相同的分解。

Quiz 2: Indexing & Hashing

- **Q1**: Construct a B^+-tree from an empty tree. Each node can hold four pointers
  - The sequential values to be inserted are: 10, 7, 12, 5, 9, 15, 30, 23, 17, 26
  - Then delete 9, 10, 15, respectively
  - Please give the B^+ trees after each insertion and each deletion
- **Q2**: Compare B^+-tree and B-tree and describe their difference

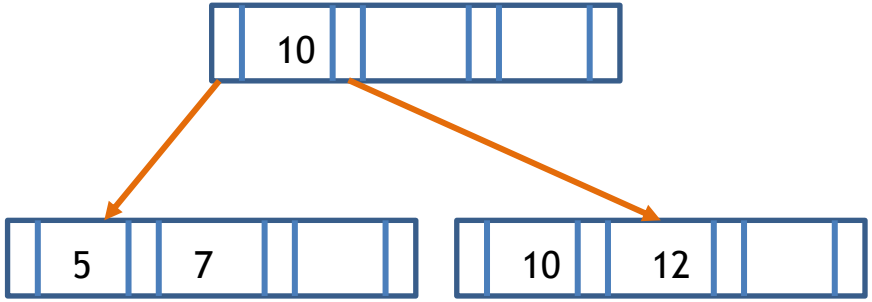
参考答案:

Q1: 见下页，可以有不同形式，符合B^+树的要求即可

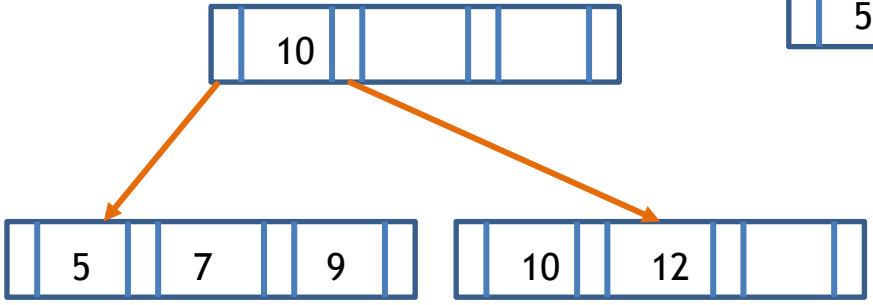
Q2: 主要体现在中间节点是否存储数据指针，具体展开说明。

插入10、7、12

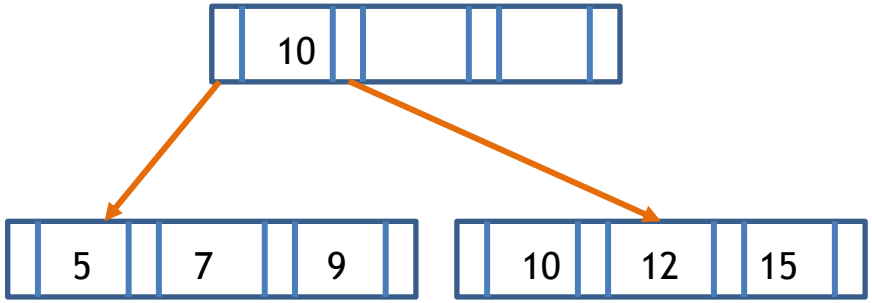
插入5



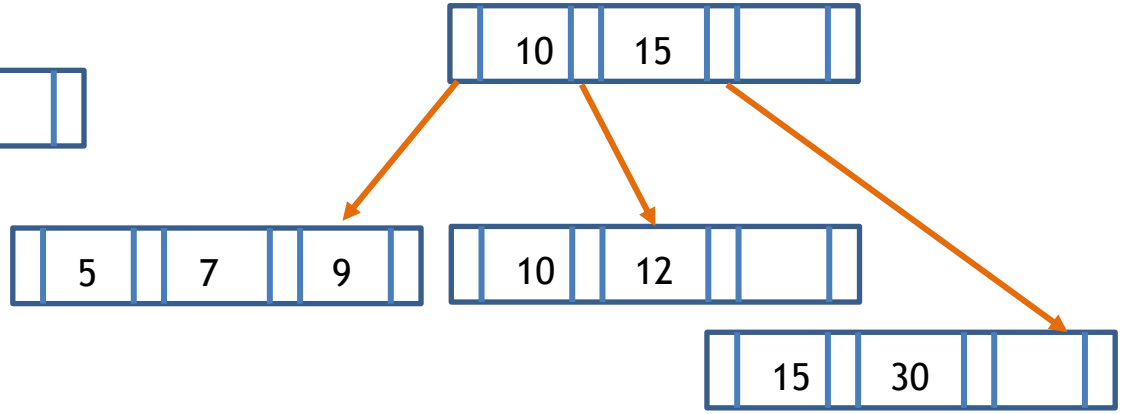
插入9



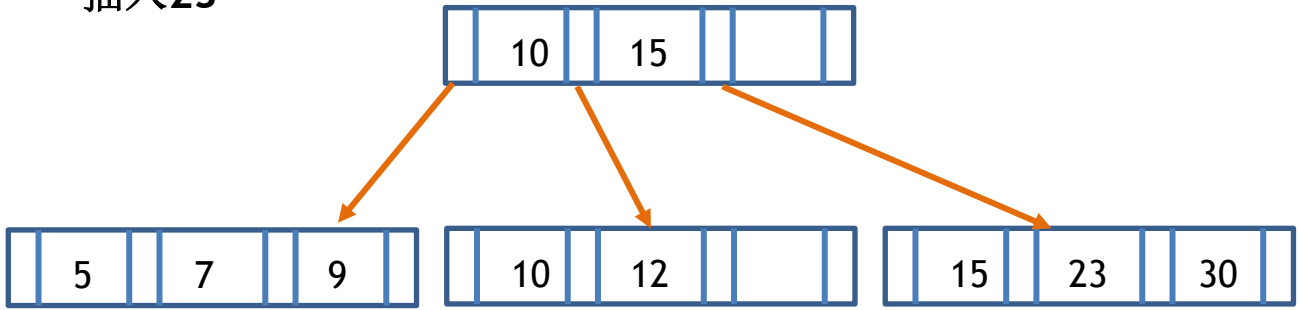
插入15



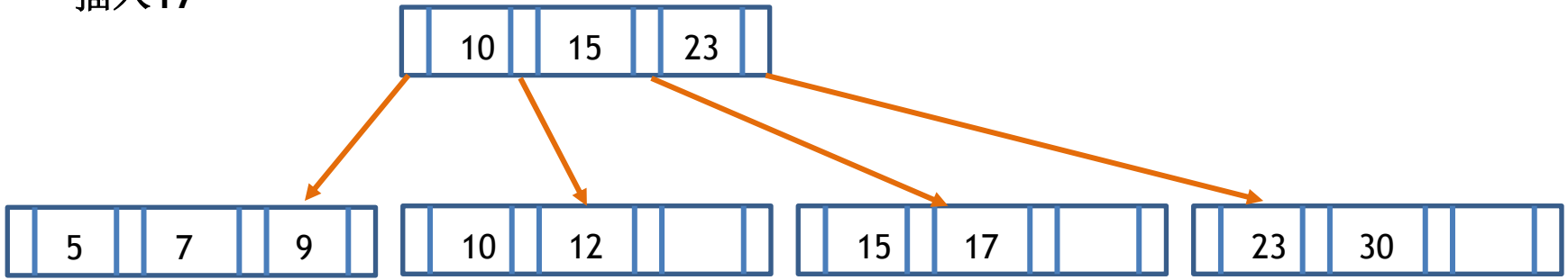
插入30



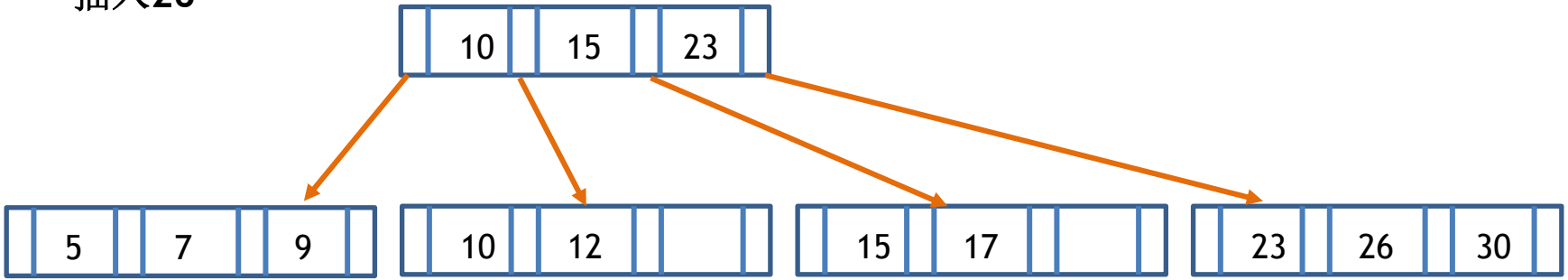
插入23



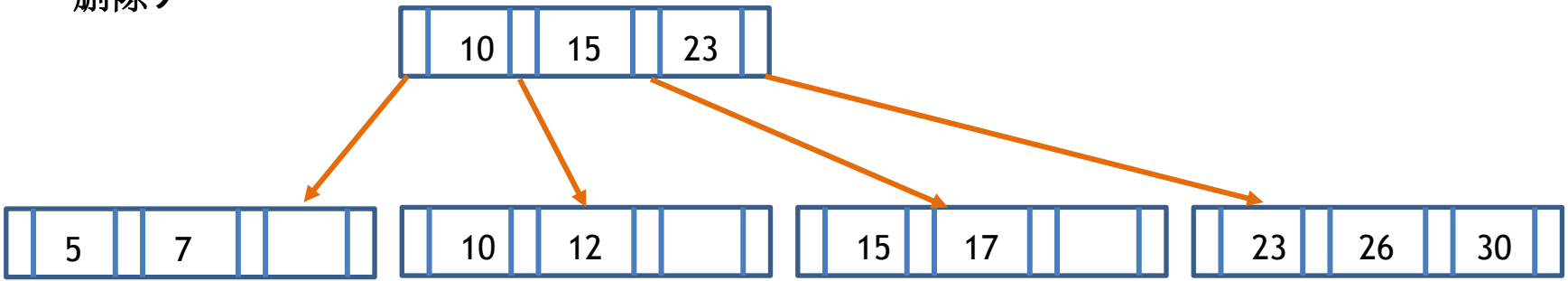
插入17



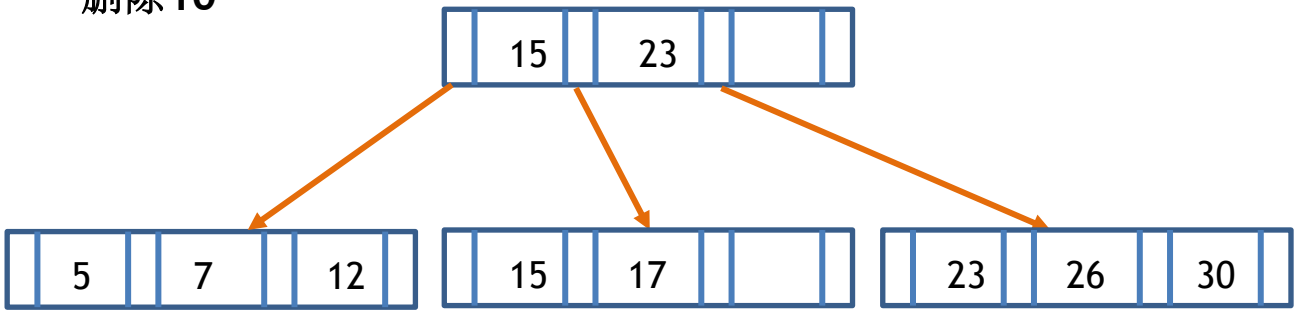
插入26



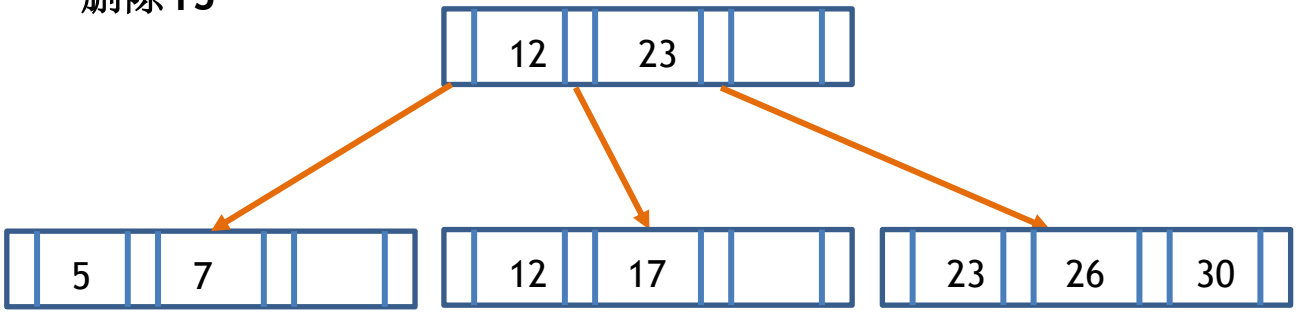
删除9



删除10



删除15



Assignment 6: Query Optimization

13.4 考虑关系 $r_1(A, B, C)$，$r_2(C, D, E)$和 $r_3(E, F)$，它们的主码分别为 A、C、E。假设 r_1 有 1000 个元组，r_2 有 1500 个元组，r_3 有 750 个元组。估计 $r_1 \bowtie r_2 \bowtie r_3$ 的大小，给出一个有效地计算这个连接的策略。

参考答案：

关系r_1、 r_2和r_3自然连接的最后结果与三个关系的连接顺序无关。因此，我们假设按顺序连接，即r_1和r_2先连接，结果同r_3连接。当r_1和r_2连接时，相同属性C是r_2的主码，因此连接结果最多有1000个元组。同理，该结果与r_3连接时相同属性为E，且E为r_3的主码，因此最终结果最多有1000个元组。

针对该连接，我们可以为r_2的属性C和r_3的属性E构建索引，用于快速检索符合连接要求的唯一元组。