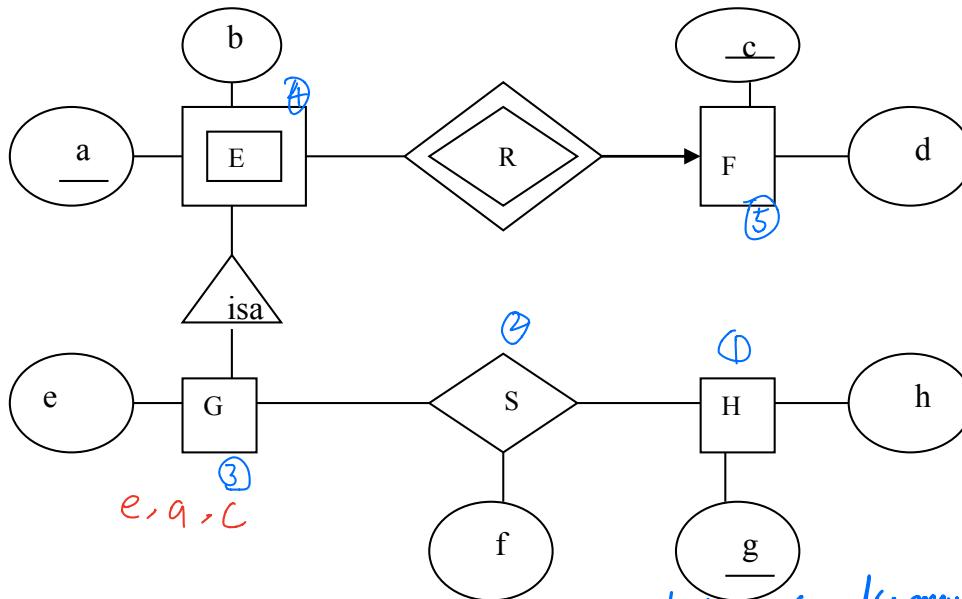


△ 与作业中的内容高度相关

ER/ODL



如何转化 diagram to relations? \rightarrow 如下

If we translate this diagram to relations in the normal way for E/R diagrams described in the textbook, which relation schema would not be in the resulting database schema?

- (A) E(a,c,b) (B) S(a,c,f,g) (C) F(c,d) (D) R(a,c)

弱实体集：自身属性+支持实体集属性

如何用 ODL 法转化？ ODL 与 ER 对比有何不同 \Rightarrow 见下

If we instead used the “object-oriented” approach to translating this E/R diagram to relations, how many relation schema would be different (when compared to the E/R-to-relations translation), either in their schema, or their set of tuples, or both?

- (A) 1 (B) 2 (C) 3 (D) 4

增加了两个子树的 relations. 注意，这

E/R 模式
ODL 转化为 relations
宝值 } P97

Consider the following ODL declarations:

```
class X (key A,B){           class Y (key C) {  
    attribute integer A;       attribute integer C;  
    attribute integer B;       attribute integer D;  
    relationship Y R1;        relationship Set<X> R2;  
    inverse Y::R2;           inverse X::R1;  
};};
```

X 有 1 个
Y 有 1 个

Which of the following relation schemas are produced according to the standard translation in the textbook?

- (A) X(A,B) and Y(A,C,D) \times
 (B) X(A,B,C) and Y(C,D) \checkmark
 (C) X(A,B) and Y(C,D) $\times \Rightarrow$ 无联系
 (D) X(A,B), Y(C,D) and R(A,B,C) $\times \Rightarrow$ 联系转化错

① 类. 联系等转化与 E/R model 相同

② 联系与它们的逆面对后，只创建 1 个 relation.

③ 对复杂的结构属性（字典、数组、包、集合等）转化，先将所有属性

DDL 中联系 $\left\{ \begin{array}{l} \triangle \text{ ODL 中联系需为二元} \\ \text{联系} \end{array} \right.$

注意点

△ 联系类型是多对一时，

可将联系与 联系中“多”的一方的类（此是说为X）建立一个关系，避免违反 BCNF 条件。
这个关系中包含 少的类的 key.

级主 relation 中，再做规范化
规范化分解 (BCNF·4NF 分解)

⇒ 使用 `Object` 方法进行子类的转化：

① 对于 isa 一层次 转化为关系，列举所有可能含根的子树，如此题可以是：

注意与根是 ←

否有支持实体无关，

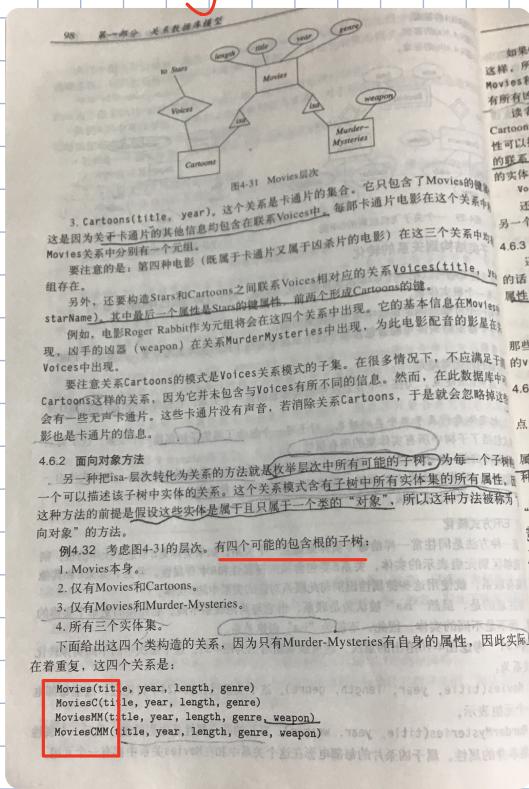
只与根有关。

1. E本身 \Rightarrow 与ER法一致
2. E和G \Rightarrow 与ER法不同
含 b (schema不同)

② 对于联系，支持联系转化为关系：与ER相同

③ 对于实体、弱实体转化为关系：与ER相同

e-9



枚举所有可能包含根的子树

$\{B \rightarrow C\}$

$B \rightarrow D$

$$A^+ = A$$

$$B^t = BDE$$

$$C^+ = C$$

Relational Theory

$D \rightarrow E$

$$\begin{aligned} D^t &= DE \\ AB^t &= ABCDE \\ AC^t &= AC \\ AD^t &= ADE \\ BC^t &= BCDE \\ BD^t &= BDE \\ CD^t &= CDE \end{aligned}$$

Given a relation $R(A,B,C,D,E)$ and FDs $AB \rightarrow CE$, $B \rightarrow D$, and

$D \rightarrow E$, which of the following FD's can *not* be inferred? \Rightarrow FD closure algorithm.

- (A) $AD \rightarrow CE$ (B) $BC \rightarrow D$ (C) $AB \rightarrow A$ (D) $B \rightarrow E$

\times

✓

多值依赖

✓

Suppose that $R(A,B,C,D)$ satisfies MVD: $A \rightarrow\rightarrow B$ and it is known that R has tuples (a_1, b_1, c_1, d_1) , (a_1, b_2, c_2, d_2) , and (a_2, b_1, c_1, d_2) . How many tuples must R have at least?

\Rightarrow 如何根据MVD推导

- (A) 3 (B) 4 (C) 5 (D) 6

RA/Datalog /SQL

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_2	c_2	d_2
a_2	b_1	c_1	d_2
a_1	b_2	c_1	d_1
a_1	b_1	c_2	d_2

For the following relational expressions, the relation schemas are $R(a,b)$ and $S(b,c)$.

Q1: $(R \bowtie S)$ 自然连接

， 几个连接的符号 是什么？

Q2: $\sigma_{R.b=S.b} (R \times S)$

$\pi_L(R)$ } 均有删除
 $R \bowtie S$ } 重复值.

- (A) Q1 and Q2 produce the same answer. $\Rightarrow \bowtie_B . \bowtie . \times$
 (B) The answer to Q1 is always contained in the answer to Q2.
 (C) The answer to Q2 is always contained in the answer to Q1.

(D) Q1 and Q2 produce different answers. why? 不同之处在哪？ Natural join
has duplicate elimination.

去重与操作

Q1: SELECT DISTINCT a FROM R WHERE $b > 10$;

Q2: SELECT a FROM R WHERE $b > 10$ GROUP BY a; \Rightarrow 分组操作 r_L .

- (A) Q1 and Q2 produce the same answer. \checkmark
 (B) The answer to Q1 is always contained in the answer to Q2.
 (C) The answer to Q2 is always contained in the answer to Q1.
 (D) Q1 and Q2 produce different answers.

三值逻辑是什么？ 0, 1/2, 1, 往小的靠 \Rightarrow 其变

In the 3-valued logic used by SQL, suppose x has the value

NULL and y is TRUE, the result for $(x > 1) \text{ OR } y$ is:

- (A) FALSE (B) UNKNOWN (C) TRUE \checkmark (D) NULL

和各种
组合均弄清
楚.

Transaction \Rightarrow 有什么性质？

RU. RC. RR. S

各有什么特点？

dirty read. repeatable.

phantom.

Assume that we have a relation Employee(ID, salary) where ID is the key, and that Employee initially has two tuples (A,20) and (B,30). Consider the following two concurrent transactions:

T1: BEGIN TRANSACTION;

UPDATE Employee SET salary = 2*salary $20 * 20 = 40 + 10 = 50$
 WHERE ID = 'A';
 UPDATE Employee SET salary = salary+10
 WHERE ID = 'A';
 COMMIT;

T2: BEGIN TRANSACTION;

SELECT AVG(salary) AS sal1 FROM Employee;
 SELECT AVG(salary) AS sal2 FROM Employee;
 COMMIT;

Suppose that T1 and T2 execute at isolation level
 ← SERIALIZABLE and READ COMMITTED respectively,
 transaction which of the following is an impossible pair of values for sal1
 之间? and sal2 returned by T2?

- (A) (35,40) ✗ \Rightarrow 在第二条之后
 (B) (40,40) ✓ \Rightarrow 在 T1 之前读取
 (C) (25,25) ✓ \Rightarrow 在 T1 之前读取
 (D) (25,40) ✓ \Rightarrow 第一个读取在 T1 之前，第二条在 T1 之后
 (4个字母)执行

$$\frac{50+30}{2} = 40$$

Constraint and Trigger

ACID:

Atomicity. Consistency. Isolated. Durability.

Suppose we have the following table declarations:

CREATE TABLE A (w INT PRIMARY KEY);

CREATE TABLE B (x INT PRIMARY KEY)

REFERENCES A(w) ON DELETE SET NULL;

CREATE TABLE C (y INT REFERENCES A(w));

CREATE TABLE D (z1 INT REFERENCES B(x))

ON DELETE SET NULL,

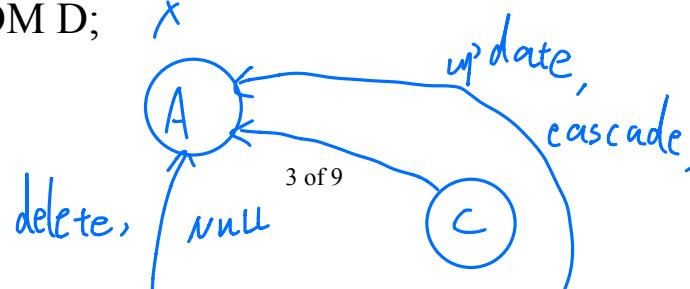
z2 INT REFERENCES A(w)

ON UPDATE CASCADE);

Consider the following scripts:

(1) DELETE FROM C; DELETE FROM B; DELETE FROM A;

DELETE FROM D; ✗



其被系统拒绝的条件

一共有几种保持



启用完整性约束
方式?
SET NULL.CASCADE.

(2) DELETE FROM C; DELETE FROM D; DELETE FROM A;
DELETE FROM B;

(3) DELETE FROM B; DELETE FROM C; DELETE FROM D;
DELETE FROM A; ✓

Which of the above scripts will empty all four tables, without error?

- (A) (3) only
- (B) (1) only
- (C) (2) and (3) only
- (D) (1) and (3) only

A 不能于 D 之前
delete.

顺序之间产生差距.

Suppose we want to enforce the FD $A \rightarrow B$ on R(A,B,C) using triggers. Consider the following operations. For which of them do we need to create a trigger on R?

- (A) delete on R ✗
- (B) update of C on R ✓
- (C) insert on R ✗
- (D) All of the above

C

FD: $A \rightarrow B$

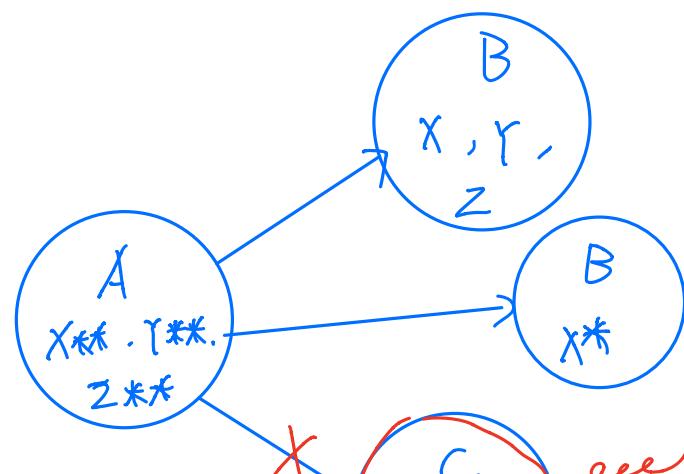
→ A 相同 B 也要相同
→ A 不同 B 相同 没关系

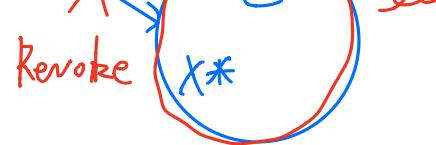
① 使用
trigger 进行
约束

② 使用 trigger
进行 FK 更新

Trigger:

ECA





Privilege = security, 权限

Suppose user A is the owner of relation R(a,b). The following sequence of grants and revocation occurs:

A: GRANT update on R to B

A: GRANT update(a) on R to C WITH GRANT OPTION

C: GRANT update(a) on R to B WITH GRANT OPTION

A: REVOKE update(a) on R FROM C CASCADE

Which of the following statement can user B do?

(I) UPDATE R SET a=a+1; ✓

(II) GRANT update on R to C; X

(III) SELECT * FROM R; X

(A) only (I) ✓

(B) (I)(II)

(C) (I)(II)(III)

(D) None of (I)(II)(III)

privileges: 1. select . update . delete . insert

2. References

3. USAGE

4. TRIGGER

5. EXECUTE 6. Under

⇒ 什么是 recursive Datalog?

The following queries are recursive Datalog, and as for all Datalog queries, the result is a set, not a bag.

Q1: Path(x,y) ← Arc(x,y) 对比 Datalog 不支持包操作

Path(x,y) ← Path(x,a) AND Path(a,b) AND Path(b,y)

Q2: Path(x,y) ← Arc(x,y)

Path(x,y) ← Path(x,z) AND Path(z,y)

(A) Q1 and Q2 produce the same answer.

(B) The answer to Q1 is always contained in the answer to Q2.

(C) The answer to Q2 is always contained in the answer to Q1.

(D) Q1 and Q2 produce different answers.

数据库中不同指有不同 tuples, 为真不同., 包含不能算

OR

Consider the following user-defined type:

→ 有哪些 correct usage?

CREATE TYPE ABC AS (x CHAR(10), y CHAR(20));

Which of the following involves incorrect usage concerning

UDT?

⇒ 什么是 UDT ?

⇒ User Defined Type

(A) CREATE TABLE R OF ABC;

(B) CREATE TABLE R (a REF(ABC));

(C) Both of the above.

(D) None of the above.

什么是 REF ?

⇒ 引用类型

e.g.: A REF(T) SCOPE R

5 of 9

属性A

T 定为 UDT 类型, 例范围为 R
为 R 中的一个类型

Recursion

I. Using Datalog to express logically
(许多涉及递归的概念用Datalog表达比SQL简单的多)

II. SQL sentences:

WITH RECURSIVE R-Name As

<definition of R-Name>

<query involving R-Name>

e.g WITH RECURSIVE Reaches (from,to) As

(select from,to from Flights) UNION

(Select R₁.from R₂.to from

Reaches R₁, Reaches R₂ where

递归，用到自身

R₁.to = R₂.from)

Select to from Reaches where from='k'

选择从 K 城市出发可达终点

• 单调性 (Monotonicity): 作用于 SQL 递归中 否定·聚集

⇒ 在一个关系中插入元组不会引起任何关系中元组的删除

• 线性递归：规则中没有多于一个子目标与 head 相递归

OR(Object Relational)

- OR与 ODL 类似，OR为对象关系模型，ODL为面向对象模型
但二者均为对关系模型的扩展以包含主要面向对象的特征。

① 嵌套关系 (UDT) ② 方法 ③ 引用类型 (该类型变量对一个UDT引用，即指针)

- OR的核心：UDT，OR功能均围绕UDT展开。

• 定义：
• UDT 定义：CREATE TYPE T-name as [<属性>]

• UDT中方法声明：(i) 在 UDT 属性声明后，跟上

METHOD func() RETURNS <TYPE>

典型SQL中方法均无参数

(ii) CREATE METHOD func() RETURNS <TYPE>

FOR <UDT名称>

BEGIN
... (方法定义)

END;

• 引用类型声明：

<属性名> REF (<UDT名>) SCOPE <关系名>

可省略，这样范围
围为全部含 UDT 类型的关系

• 用UDT声明关系：CREATE TABLE <表名> OF <UDT名>

(<元素列表>)

键，外键等，基于元组约束等，
与UDT无关。

• UDT的对象标识 (Object Identity)：UDT类型的元组
有一个DBMS产生的对象标识，为对元组本身的引用。(在ODL
中此标识不可访问，而OR中可通过：

REF IS <name> SYSTEM GENERATED

只赋值给 name 属性。

- VDT 不允量访问

返回其值：生成器 $\Rightarrow x_1 = \text{VDT}.x()$
 $y_1 = \text{VDT}.y()$

修改其值：转换器 $\Rightarrow \text{VDT}.x(x_1)$
 $\text{VDT}.y(y_1)$

- VDT 的排序功能 (少用)

Create ordering for T equals only by <属性>

Create ordering for T ordering full by <属性集>

OLAP

维属性为引用相应
维表的键的外键(FK)

Consider a fact table in an OLAP application: Facts(D1,D2,val) where D1 and D2 are dimension attributes and val is a dependent attribute. Suppose D1 and D2 take on 2 and 3 different values respectively, and all combinations of values are present in Facts. How many new tuples (i.e. not in Facts) are produced in the result of the following query?

SELECT D1, D2, SUM(val)

FROM Facts

GROUP BY D1, D2 WITH CUBE;

(A) 5 (B) 6 (C) 7 (D) 8

$$3+2+1=6$$

查询结果不仅包含 group-by 操作
生成的元组，也包含 CUBE(R) 中的元组

(D₁, NULL, S)

(NULL, D₂, S)

(NULL, NULL, S)

↓

SUM(val)

切片和切块:
group by 得到立
个体按分组属性的块。

where con
From con

中的 condition 使是
立体按条件切片

Consider the following XML DTD:

```
<!DOCTYPE R [
    <!ELEMENT R(A+,B+)>
    <!ELEMENT A(X,Y*)>
    <!ELEMENT X(#PCDATA)>
    <!ELEMENT Y(#PCDATA)>
    <!ELEMENT B(#PCDATA|Z)>
    <!ELEMENT Z(#PCDATA)>
]>
```

How many elements are there in the smallest XML document (i.e., a document containing the fewest possible elements) that conforms to the above DTD? Note that text values are not elements.

(A) 3 (B) 4 (C) 5 (D) 6

R · A · B · X ·

由根标签开始往下。

什么是DTD? \Rightarrow Document Type Definition,

\Rightarrow 子元素必须出现，且必须少于或等于出现。

什么是XML document?

\Rightarrow XML语句
形成的文件

A+, B+ = 元素必须
至少出现一次。

DTD, XML schema: 为 XML document 提供模式的方法

OLAP

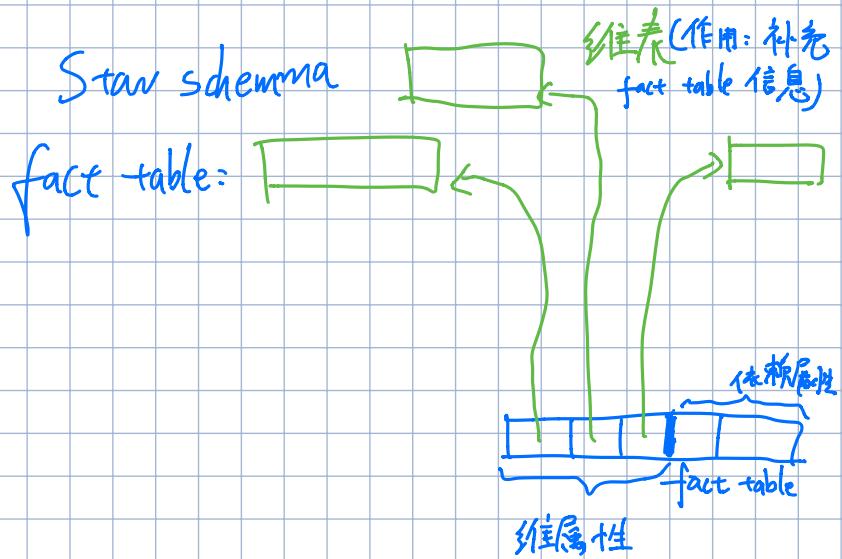
接触复杂大数据，需专门
↑ 数据仓库

- 概念：
OLAP (On-Line Analytic Processing) 联机分析处理
OLTP (On-Line Transaction Processing) 联机事务处理

一般数据库，只接触一小部分数据

- ROLAP (Relational OLAP) \Rightarrow 采用Star schema's relational view
- MOLAP (Multidimensional) \Rightarrow 采用专门数据立方体模型

什么是 fact table? \Rightarrow 事实表，为感兴趣的事件/对象形成中心关系



Drill Down: 划分更精细过程 \Rightarrow 销售车改为销售红色车

Roll-Up: 划分更粗粒的过程 \Rightarrow 月销售改为年销售

- WITH ROLLUP : 只有当元组已聚集尾部分组属性时，才会产生额外的聚集元组 [CUBE (R)]

e.g.
(----, NULL, 100, 200)
(---, NULL, NULL, 200, 600)

而不会出现: (NULL, ---, 100, 200) 这种情况

- WITH CUBE : 不论聚集哪均会产生聚集元组。

XML

- 概念：Extensible Markup Language (可扩展标记语言)

与 HTML 对比：同：均是基于标签的，用于标记的语言

异：HTML 标签为文档中信息的表示方式

XML 标签为文档各部分的含义

- XML 元素：标签对 + 其中的内容

或：单一标签 \Rightarrow 只有属性，不可有内容

- XML 模式

Well-formed \Rightarrow 无预定义模式，用户
自由定义标签

Valid \Rightarrow 由 DTD 定义模式

- XML 属性：存在于开始标签中；可将子节点内容直
接作为父节点标签，从而删除子节点

作用：表示连接，标签中值可为其它元素
ID 的引用，从而表示本元素到其它元素的连接

- DTD (Document Type Definition)：与 XML schema
类似，均为规定 XML 文档模式的文件。

格式：`<!DOCTYPE 根标签名 [`

`<!ELEMENT 元素名 (分量)>`

分量

特殊形式：`#PCDATA - EMPTY`

文本值，无嵌套 单配对标签

`]>`

- 控制元素出现次数的操作符： * . + , ?
- 分量中元素若无修饰，则必须出现且必须以
 - () 中顺序出现

- 使用 DTD

文档前包含

开始行引用 \Rightarrow 将 `standalone="no"`

表示使用 DTD，否则置为 "yes"

- 属性列表：

`<!ATTLIST 表名`

`year CDATA #REQUIRED`
`genre (A|B|C|D) #IMPLIED`

→ 必须存在

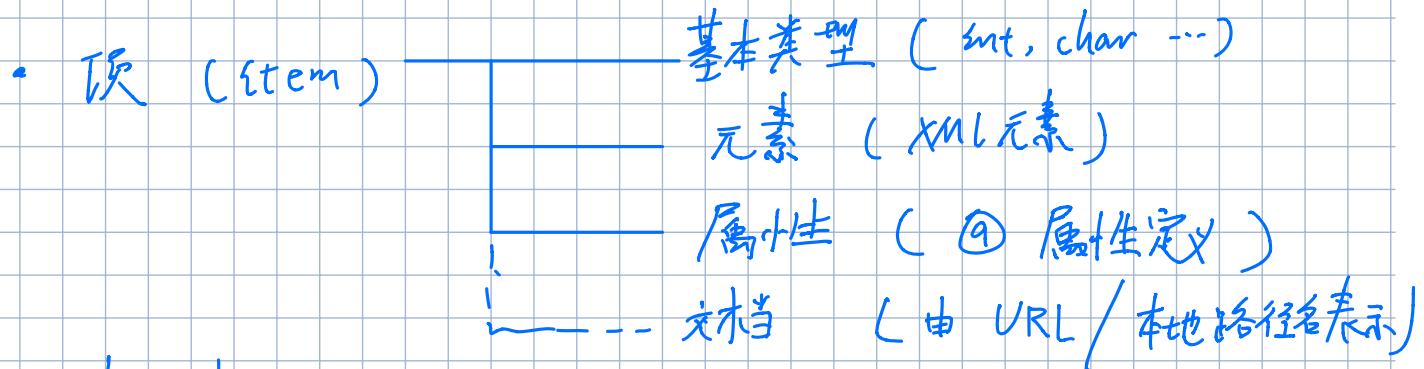
→ 可选

属性类型： CDATA / 枚举

XPath

- 概念：简单XML 程序设计语言

XQuery : XPath 的扩展语言



表达式 = $/T_1/T_2 \dots /T_n$

根节点，在此之前，用一个文档节点表文件，在 T_1 之后，用元素节点表文件。

符号：

.. 父亲

. 自身

① 属性

/ 儿子

* 任何一种标签

② * 任何属性

// 轴 (axes)， //city 表找到任何一个嵌套层次所有 city 元素，并按照它们出现次序返回。若此前已有路径限制，则找路径下所有元素

- 路径表达式中的条件:

1. / home / Start[Address/city = "M"]/Name

[] 中的路径表达式应用在整
个文档中，而非当前路径下。

2. [int] \Rightarrow e.g. [1], 表应用到当前节点
的父节的的第一个子节点元素

3. [标签名] \Rightarrow e.g. /Version[star] 表 version
标签中至少有一个 star 子元素，才将这
样的 Version 标签返回结果。

• XQuery: XPath 的扩展形式，加入 FLWR
表达式。F: For \Rightarrow 创建循环 L: Let \Rightarrow 引入变
量临时定义 (注意:=为赋值，=为比较是否相等)
W: where \Rightarrow 条件 R: Return \Rightarrow 返回查询结果