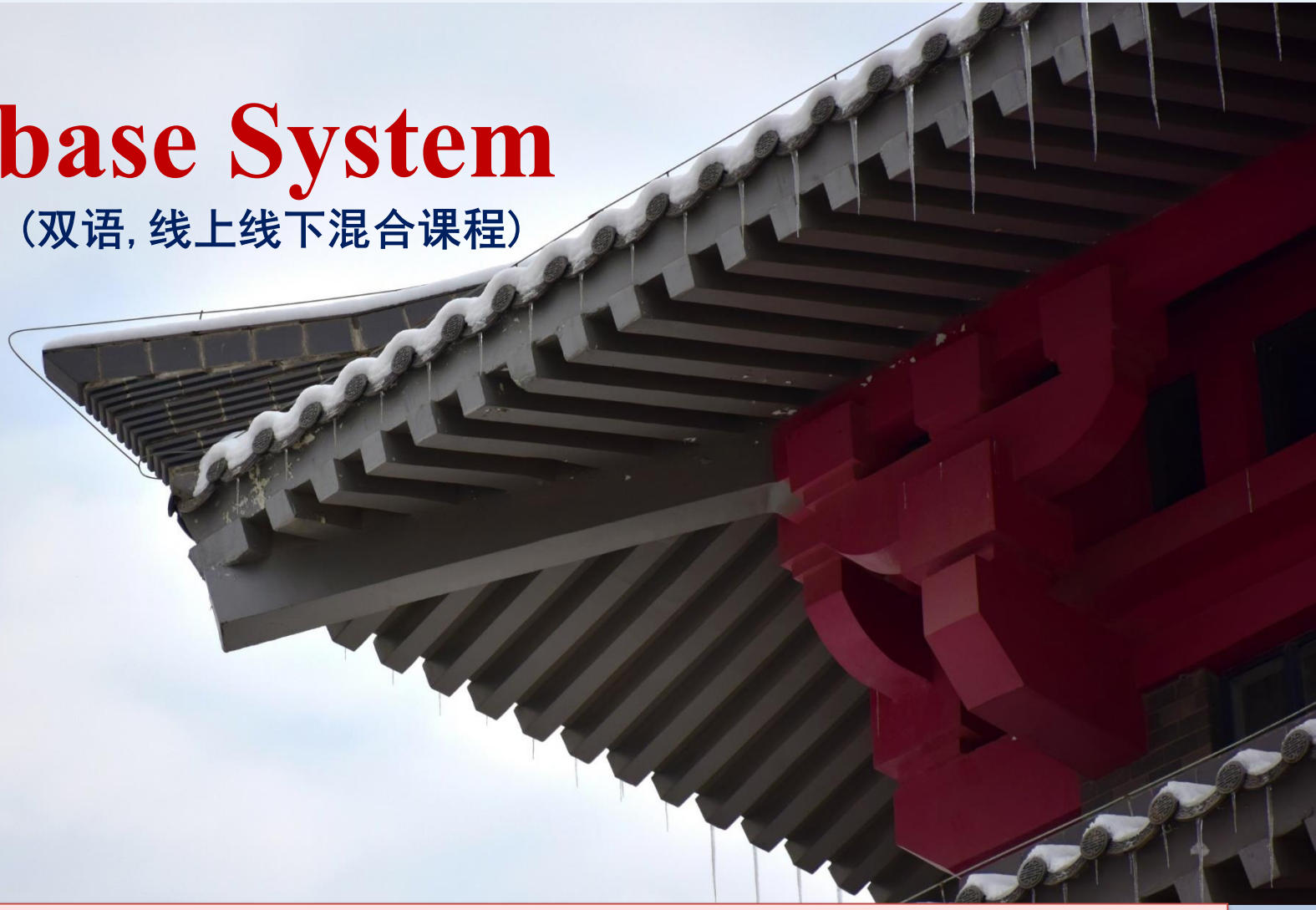
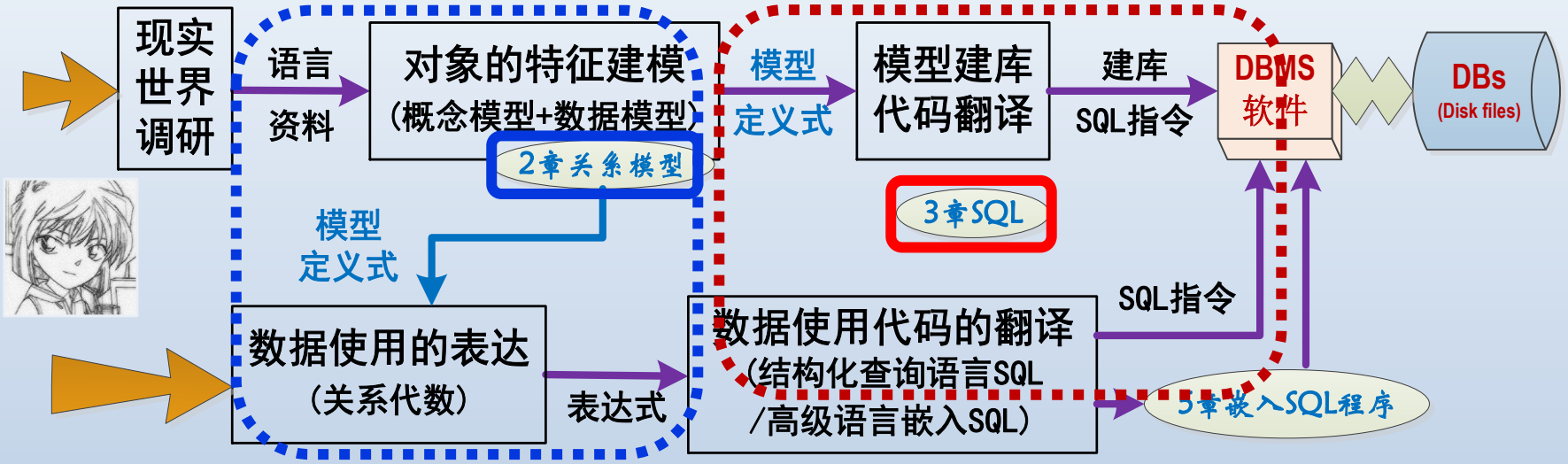
Database System

数据库系统（双语，线上线下混合课程）



Chapter 3: Basic SQL Query Language

Studying Questions



- ***How to build a DB ? How to use a DB ?***

Study Questions

- **How to realize a Relational Database from Relational model?**

  - SQL (Structured Query Language): 结构化查询语言
  - Ex1: Create the CAP database and its tables.

```
CREATE CHEMA CAP;
CREATE TABLE customer (cid char(4), cname varchar(10), discnt int,
                       city char(8), primary key (cid) );
```

- **How to use the Data in a Relational Database?**

  - Make a Relational Algebra　　集合的关系代数运算
  - Its Limitations　　数学逻辑表达，无法驱动计算机

- **How to translate a Relational Algebra into instructions that computers can execute?**

  - Ex2: List customer ID, name and discount of all customers in New York.

```
SELECT cid, cname, discnt FROM customers WHERE city = 'New York';
```

Database Principles, Programming and Performance

Chapter 3: Basic SQL Query Language

创建DB及对象

3.1 Introduction

译/编制SQL语句

3.2~3.3 Simple Select statements

3.4 Sub-queries

3.5 UNION Operators and for ALL conditions

3.6 Some Advanced SQL syntax

3.7 Set Functions in SQL

3.8 Groups of Rows in SQL

3.9 A complete Description of Select

查询句

3.10 Insert Update and Delete statements

3.1 Introduction

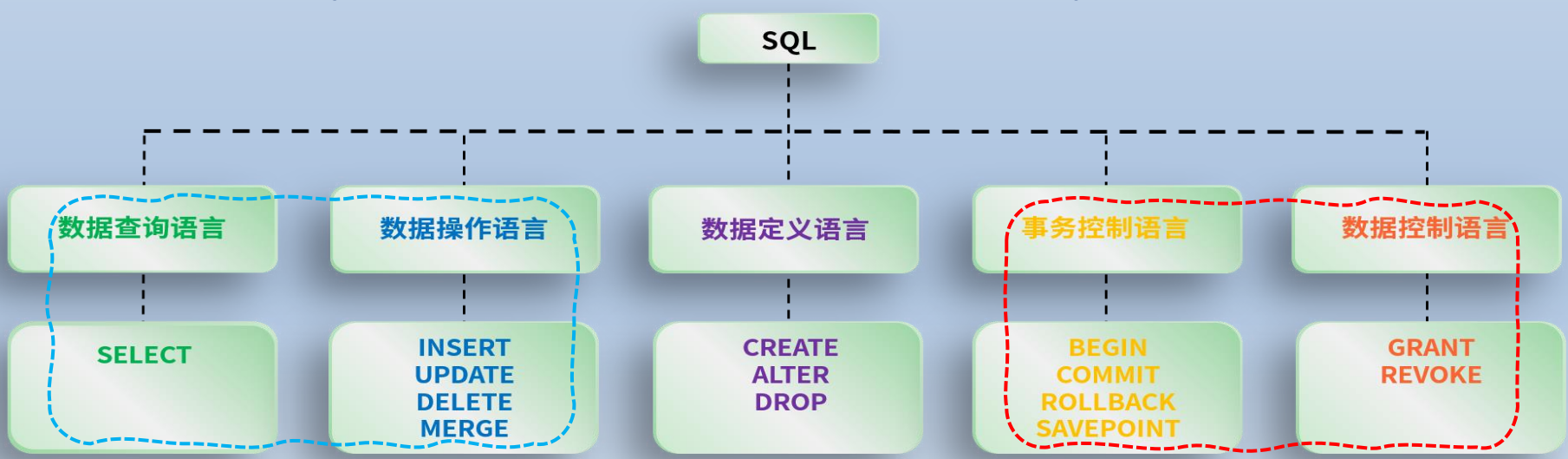
1. History of SQL

- SQL ---- Structured Query Language
- 1986年，ANSI (American National Standard Institute) 的数据库委员会批准SQL作为”关系数据库语言的美国标准”
- 1989年，ANSI 与 ISO (International Standard Origination) 联合完善后，颁布了*SQL_89*国际标准
- 后面公布的*SQL:1999*增加对Object-Relation Model的支持
- 最新的*SQL:2011*支持XML(W3C组织发布的 可扩展标签语言)
- 现多数RDBMS均支持SQL标准
  - ✓ Oracle、DB2 、Informix、Sybase、 SQL Server
  - ✓ 开源数据库：如PostgreSQL、MySQL、NoSQL
  - ✓ 甚至小型产品：如Access

3.1 Introduction

2. Frame of SQL　为便于学习将SQL分为下图几类

- DDL (data definition language)，用于定义数据库的对象(如table, view, index等)，语句包括创建、修改和删除对象 (CREATE, ALTER, DROP) 等
- DML (data manipulation language)，用于表数据的增加、修改、删除、查询等操作 (INSERT, UPDATE, DELETE, SELECT etc.)
- DCL (data control language)，控制数据的访问权限，有授权(GRANT)和撤销(REVOKE)
  - DQL (data query language)，库中数据和信息的查询语句，如SELECT
  - TCL (transaction control language)，管理数据库的事务，包括启动、提交、回退事务等(BEGIN Transaction、COMMIT、ROLLBACK)



3.1 Introduction

3. Capabilities of SQL

[1]综合统一

- 非关系型DB:
  - 分别定义External Schema、Schema、Internal Schema
  - 修改任一模式前，须停止DB运行、转储DB；修改后，重装DB
- 关系型SQL：
  - 集DDL、DML、DCL于一体，用SQL可完成“3类模式定义、数据访问控制、数据操作”等DB生命周期中的全部活动
  - DB运行状态下，可随时修改模式，系统可扩展性良好

[2]高度非过程化

- ✓ SQL操纵数据只需提出“做什么”，无须指明“怎样做”，数据存取路径、操作过程均由DBMS自动完成

3.1 Introduction

[3]面向集合的操作方式

- ✓ 非关系模型下，数据操作为 one time a record
- ✓ 关系模型下，SQL数据操作 one time a set

[4]SQL以同一种语法结构提供两种使用方式

- ✓ 是自含式语言(可联机交互使用)，
  又是嵌入式语言(嵌入到其它高级语言中）
- ✓ 两种使用方式下，SQL的语法结构基本一致

[5]语言简捷，易学易用

- ✓ 数据查询 select
- ✓ 数据操纵 insert, update, delete
- ✓ 数据访问控制 grant, revoke

3.1 Introduction

```
[EXEC SQL] CREATE SCHEMA
    {schemaname
    | AUTHORIZATION authorization_id}
        [{schema_element schema_element ...
```

3. Typographical Conventions (印刷规则)

①SQL 印刷有大、小写之分，但实际使用时不敏感 .

②Meanings of some symbols

- [ ] the optional term / 可选
- | select one term from many / 多选一
- {} the optional and repeat term / 可选且可复选
- ALL the Default term

4. Exercise : read the Manual/Help of the DBMS products

- 创建数据库： CREATE SCHEMA db_xauat ;
- Appendix C: *SQL 语法*
- Appendix B: *Oracle 系统函数*
- Appendix A: (1)A1、A2为Oracle、Informix的DB创建方法
  (2)A3为*常见DBMS的标准数据类型*

3.2 Setting up the Database

1. Prepare Knowledge

[1] 交互式SQL：启动Oracle /SQL Plus　　启动MySQL /Workbench

或第三方数据库管理工具Navicat、SQL developer

登录DBMS；界面下发出SQL指令。

[2] Login with DBMS must have following things

An account/user_id on DBMS + A password

[3] 创建数据库、连接数据库、创建数据表

创建数据库：**CREATE SCHEMA db_xauat ;**

连接数据库：**CONNECT TO db_xauat ;**

创建数据表：**CREATE TABLE usi (ud char(4) not null, pass char(8), primary key (uid));**

[4] SQL语句　(ORACLE/MySQL中的SQL句结束符为“;”)

- DDL:　eg. Create scheme, Create table
- DCL:　eg. Grant, Revoke
- DML:　eg. Select, Insert, Update, Delete

3.2 Setting up the Database

2. CREATE TABLE 语句

CREATE TABLE tablename (colname datatype [not null]

{ , colname datatype [not null] … }

[, PRIMARY KEY (colname { , colname … })])

参考：**create table <表名>(<列名><数据类型>[列级完整性约束]**

[，<列名><数据类型>[列级完整性约束]…

[，<表级完整性约束>]）；

Note:

A) 若完整性约束条件涉及该表的*多列*，则须定义为*表级约束*

B) Datatype中需指明“类型、长度”，DBMS产品支持的Datatype不完全相同

3.2 Setting up the Database

Customers			
cid	cname	city	discnt
c003	mary	Dallas	6

主表

Example 1: Create tables of Customers, Agents, Products and Orders in AP DB.

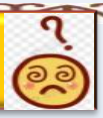
```
create table customers (cid char(4) not null, cname varchar(13), city varchar(20),
        discnt real, primary key (cid));

create table agents (aid char(3) not null, aname varchar(13), city varchar(20),
        percent smallint default 0, primary key (aid));

create table products (pid char(3) not null, pname varchar(13), city varchar(20),
        quantity integer,  price double precision, primary key (pid));

create table orders (ordno integer not null, month char(3), cid char(4),
        aid char(3), pid char(3), qty integer, dollars double precision,
        primary key (ordno),
        foreign key(cid) reference customers(cid),
        foreign key(aid) reference agents(aid),
        foreign key(pid) reference products(pid) );
```

从表

Orders						
ordno	month	cid	aid	pid	qty	dallor

3.3 Simple Select Statements

- **How to Query the Data in DB?**



Example: select cname, city, discnt from customers
where discnt >= 6 order by 3 desc;

1. *SELECT* Statement

[1] 句式：Select [all | distinct] {* | expr [[AS] c_alias] {, expr [[AS] c_alias]…}}
From tableref {, tableref … }
[Where search_condition]
[Group by colname {, colname …}] [Having search_condition]
[Order by colname [ASC | DESC] {, colname [ASC | DESC] …};

[2] Select含义(功能)：

- 依where子句的表达式，从from子句指定的表/视图中，找满足条件的元组；
- 按Select目标列选出元组中的属性，形成查询结果集；
- 结果集按Group指出的列，相同值分为一组；满足Having条件的结果组输出；
- 输出的结果组，按Order指定的列值排序显示；ASC为升序，DESC为降序。

3.3 Simple Select Statements

[3]Example: 生成顾客及其销售商之间的交易额报表

(C×O×A where c.cid=o.cid and o.aid=a.aid) [cid, aid, dollar]

```
select  o.cid, c,cname, o.aid, a.aname sum(o.dollars) as  casales
        from   customers c, orders o, agents a
        where  c.cid = o.cid and o.aid = a.aid
        group by  o.cid, o.aid having sum(o.dollars) >= 900
        order by  5  desc;
```

Alias

Qualifier

Orders

ordno	month	cid	aid	pid	qty	dallor
1011	jan	c001	a01	p01	400	450
1012	jan	c001	a03	p03	50	180
1019	feb	c002	a03	p03	890	54
1017	feb	c001	a01	p05	200	704
1018	mar	c002	a05	p01	100	88

cid	aid	dallor
c001	a01	450
c001	a01	704
c001	a03	180
c002	a04	54
c002	a05	88

cid	aid	dallor
c001	a01	450+704 1154
c001	a03	180
c002	a04	54
c002	a05	88

3.3 Simple Select Statements

[3]Example: 生成顾客及其销售商之间的交易额报表

(C×O×A where c.cid=o.cid and o.aid=a.aid) [cid, aid, dollar]

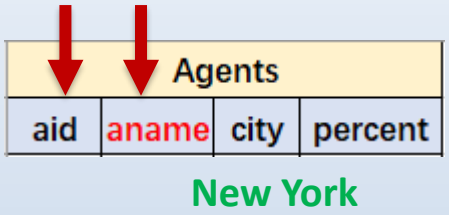
```
select select  o.cid, c,cname, o.aid, a.aname , sum(o.dollars) as  casales
        from   customers c, orders o, agents a
        where  c.cid = o.cid and o.aid = a.aid
        group by  o.cid, o.aid having sum(o.dollars) >= 900
        order by  5  desc;
```

[4] Select与Relational Algebra的关系

- from子句 <===> *Rel. Alg.*中的“Relation”
- where子句 <===> *Rel. Alg.*中的“Selection”运算
- select子句 <===> *Rel. Alg.*中的“Projection”运算
- group子句：查询结果集按列 *i*值分组，值相同的元组排在一组 order[cid,qty]
- group子句带having短语：满足having条件的结果组才输出.
- order by子句，查询结果还应按“列 *j* ”值排序.

3.3 Simple Select Statements

2.Example 3.3.1(P87) We wish to find the aids and names of agents that are based in New York .



[all | distinct]

--*Rel. alg.* (AGENTS where city = 'New York') [aid, aname]

--*SQL* select aid, aname from agents where city = 'New York';

- ✓ duplicate values are possible.

  select all aid, aname from agents where city = 'New York';

- ✓ needless duplicated rows.

  select distinct aid, aname from agents where city='New York';

Eg 1: Wish to find the names of cities that agents are based.

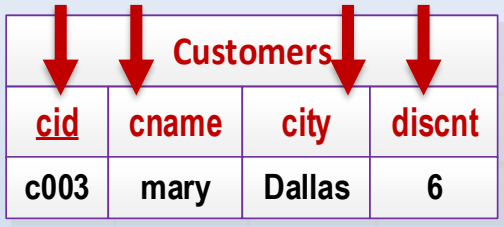
√ select distinct city from agents; select city from agents; ? 【duplicated rows】

Eg 2: Wish to find the names of Agents that are based in New York.

select distinct aname from agents where city='New York'; ?

3.3 Simple Select Statements

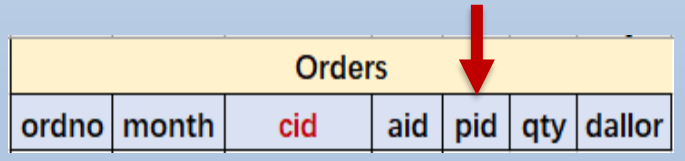
3.Example 3.3.2 Display all values in every row of the customer.



--Rel. alg. Customers [cid, cname, ….]

--SQL **select * from customers;** /* * means all columns */

4.Example 3.3.3 Select all pids for which orders are placed.

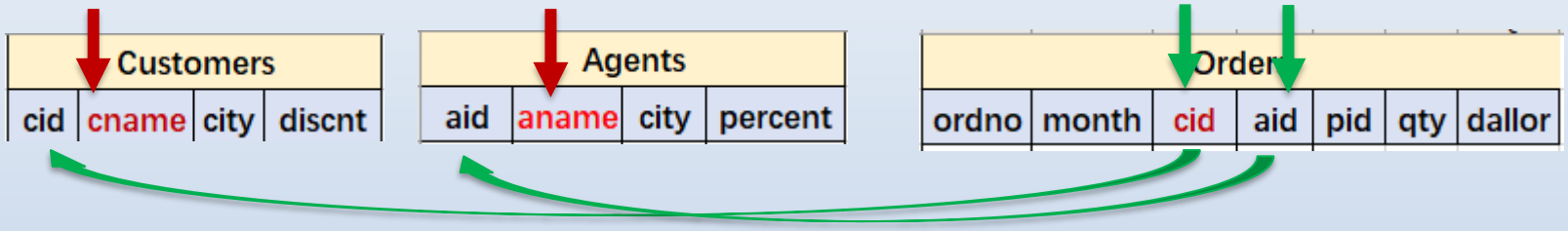


Orders[pid]

select all pid from orders;

3.3 Simple Select Statements

5.Example 3.3.4 Retrieve all (cname, aname) pairs where the customer places an order through the agent.



a) Rel. alg. (C[cid, cname] ∞ O[cid, aid] ∞ A[aid, aname]) [cname, aname]

c) SQL:

```
select C.cname, A.aname
    from customers C, orders O, agents A
    where C.cid = O.cid and O.aid = A.aid ;
```

3.3 Simple S

```
select   o.cid, c,cname, o.aid, a.aname , sum(o.dollars) as  casales
   from   customers c, orders o, agents a
   where  c.cid = o.cid and o.aid = a.aid
   group by  o.cid, o.aid having sum(o.dollars) >=900
   order by 5  desc;
```

Alias

Qualifier

6.The corresponding: *Select Statement <===> Relation Algebra*

- Take *Product* of tables in FROM clause.
- Apply *Selection* in WHERE clause.
- *Projection* on attributes in SELECT-list and report.
- If the column name are duplicated, use qualifier name.

  select C.cid, O.cid …
- If the rows are duplicated, use 'distinct' optional.

  select distinct C.cid, O.cid …
- Using GROUP BY to *re-arranging* rows, and selecting the groups to be shown.

  group by O.cid having O.city<>'New York'
- Using ORDER BY to *re-arranging* the rows to be shown.

  order by O.city desc

3.3 Simple Select Statements

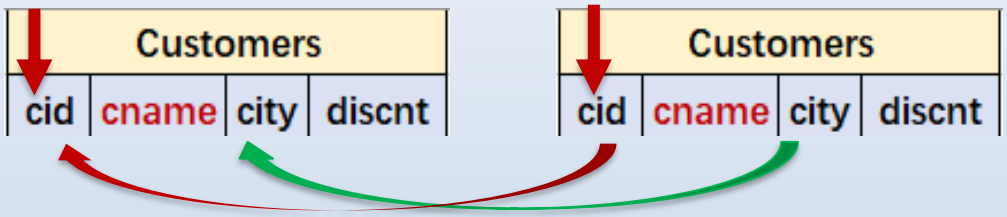
7.Example 3.3.5(p90) Retrieve a table based on orders, with columns ordno, cid, aid, pid, and profit on order. Where the "profit" is calculated from "quantity and price of the product sold(sell) by subtracting(except) 60% for whole sale cost, the discount for the customer and the percent commission for the agent".

Profit = qty*price*((1–0.6) - (discnt*0.01) - (percent*0.01))

```
select  ordno, x.cid, x.aid, x.pid,
        (x.qty*p.price)*(1- 0.6 - 0.01*c.discnt - 0.01*a.percent) as profit
            from  orders x, products p
            where  p.pid = x.pid;
```

3.3 Simple Select Statements

8.Example 3.3.6 (p92) List all pairs of cids based in the same city.



a) Rel. Alg.

c1 := customers

c2 := customers

((c1 ∞ c2) where c1.city = c2.city and c1.cid < c2.cid)[c1.cid, c2.cid]

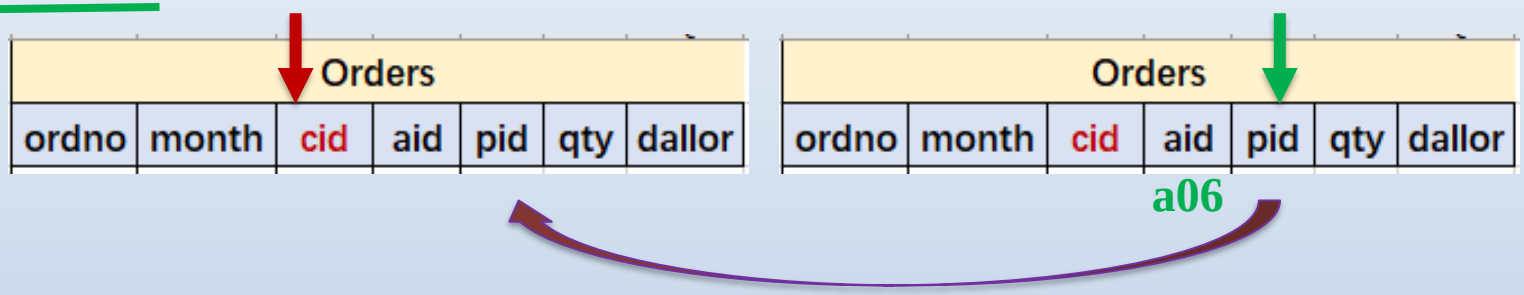
b) SQL: select c1.cid, c2.cid from customers c1, customers c2

where c1.city = c2.city and c1.cid < c2.cid;

3.3 Simple Select Statements

9.Example 3.3.8 (p94) Get cids who order a product for which an order is also placed by agent a06.

(找cids，它们订购的产品，a06也卖给过别人)



Rel. Alg. O1:=orders; PX := (O1 where aid='a06')[pid]

O2:=orders; B := (PX ∞ O2) [cid]

pid

a) Products ordered by a06 :

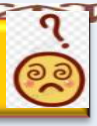
```
select  x.pid  from orders O1  where O1.aid = 'a06' ;
```

b) Customers who order those products:

```
select distinct  O2.cid  from orders O1, orders O2
                where O1.aid = 'a06' and O2.pid = O1.pid;
```

3.4 Sub queries

- *Dose the Query Condition is same as Rel. Alg.?*



1. Predicate

[1]Example 3.4.2 Retrieve information of agents based in Duluth or Dallas.

Alg. (A where city='Duluth' or city='Dallas') [aid, aname,…]

SQL select * from agents where city in ('Duluth' , 'Dallas');

aid∈X

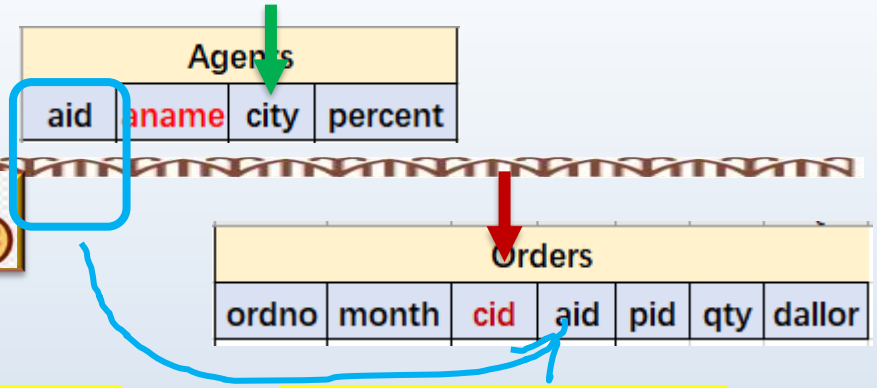
Predicate
In
Between
>=any
<=all 等

[2]Example 1 Find Customers with his discount from 5 to 10.

Alg. (C where discnt>=5 and discnt<=10) [aid, aname,…]

SQL select * from customers C where discnt between 5 and 10;

3.4 Sub queries



- **Dose the Query can be nested?**

2. Subquery

- A **Select** statement appearing **within another** Select **on search-condition**.
- Using **Predicates** and **Logic conditions** with TRUE or FALSE.

[2] Example 3.4.1 (p97) Retrieve cids of customers who place orders with agents in Duluth or Dallas.

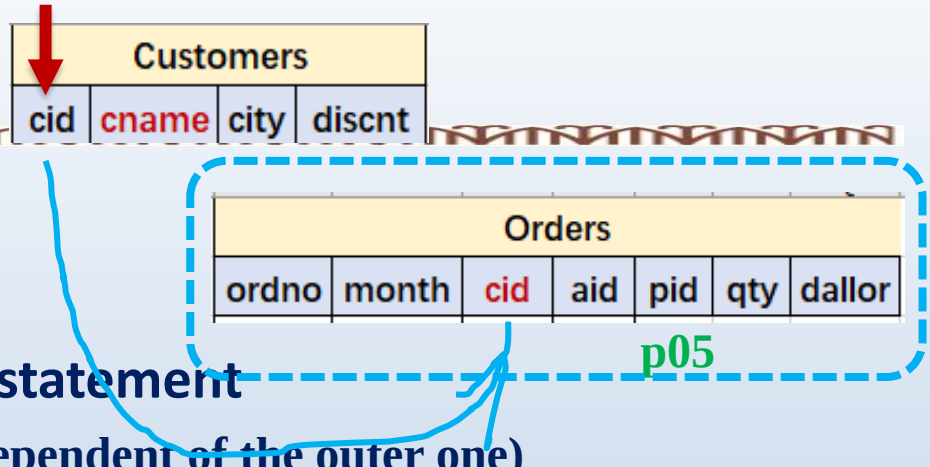
a) select x.cid from orders o, agents a
where o.aid = a.aid and a.city in('Duluth', 'Dallas');

b) Find aids in Duluth or Dallas, make a subquery to select these agents.

```
select distinct cid from orders
    where aid in (select aid from agents
                  where city='Duluth' or city='Dallas' );
```

Subquery

3.4 Sub queries



3. Classification of Subquery

[1] Uncorrelated Subquery

Can pre-calculate the inner SQL statement

(the inner Subquery is completely independent of the outer one)

[2] Correlated Subquery

Can't pre-calculate the inner loop;

(the Subquery using data from an outer Select.

[3] Example 3.4.4 (Correlated Subquery) Find cnames who order product p05.

R.A. T1 := (Orders where pid='p05') [cid] T2 := (T1 ∞ Customers) [cname]

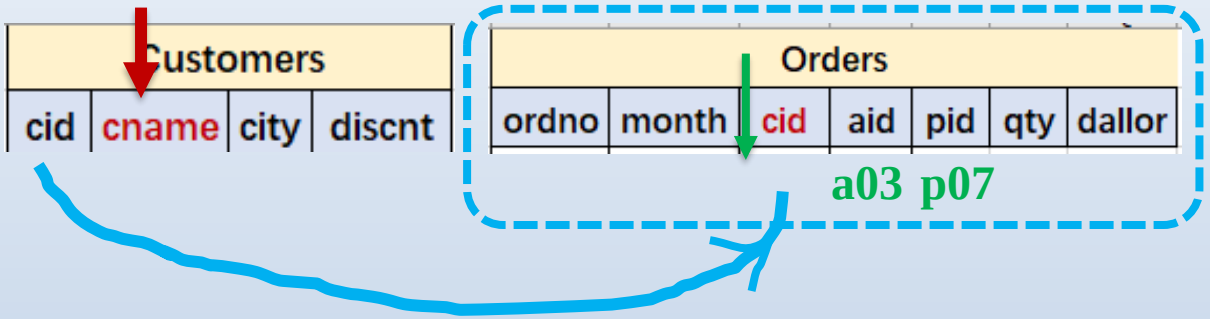
SQL1 select distinct cname from customers c (correlated Subquery)

where 'p05' in (select pid from orders where cid=c.cid);

SQL2 select distinct cname from customers c, orders o where c.cid = o.cid and o.pid ='p05'

3.4 Sub queries

[4] Example 3.4.5 Get cnames who order product p07 from agent a03.



Rel. Alg. T1 := (Orders where pid='p07' and aid='a03') [cid]

T2 := (T1 ∞ Customers) [cname]

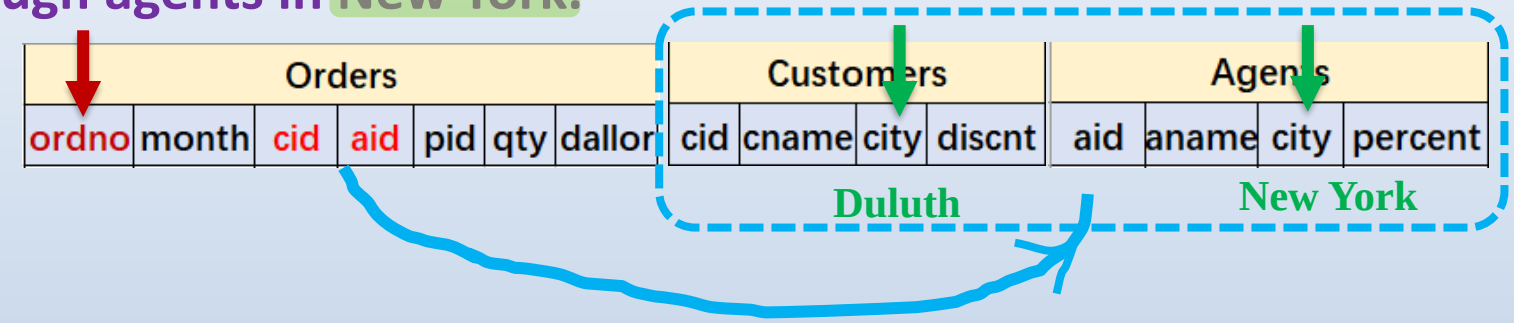
SQL

```
select cname from customers
    where cid in (select cid from orders
        where pid = 'p07' and aid = 'a03');
```

(Uncorrelated Subquery)

3.4 Sub queries

[5]Example 3.4.6 (p100) Retrieve ordno for all orders placed by customers in Duluth through agents in New York.




```
select ordno  from orders o  where (cid, aid) in
    (select cid, aid   from customers c, agents a
          where c.cid=o.cid and a.aid=o.aid and
                c.city='Duluth' and a.city='New York');
                    /* correlated Subquery */
```

3.Summary

[1] *Select* statement is *nonprocedural*

[2] *in* predicate: expr [NOT] in (Subquery) | expr [NOT] in (val1{ , val2…})

3.4 Sub queries

4. Quantified Comparison Predicate

[1] Comparison Predicates:

=some	<>some	<=some	>some	<some	>=some
=any	<>any	<=any	>any	<any	>=any
=all	<>all	<=all	>all	<all	>=all

Eg: =some (subquery) is TRUE<==> S for some S in the subquery

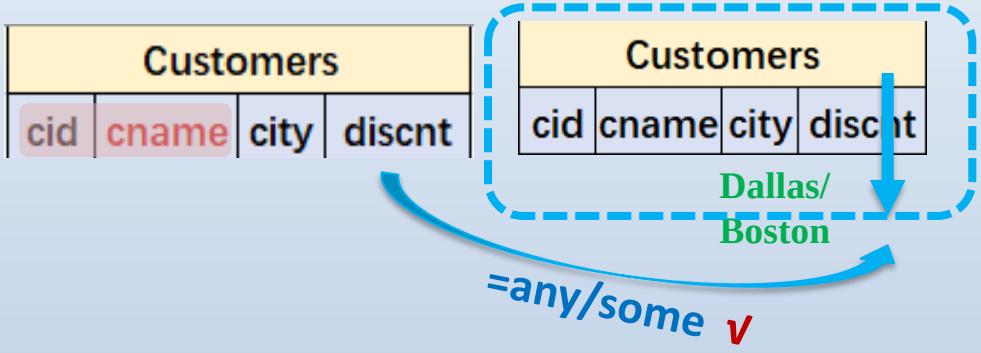
=any (subquery) is TRUE <==> S for some S in the subquery

=all (subquery) is TRUE <==> S is true for all S in the subquery

[Notice] that "some" and "any" mean same thing.

3.4 Sub queries

[2]Example 3.4.8 (p101) Find all customers who have the same discount as that of any of the customers in Dallas or Boston.

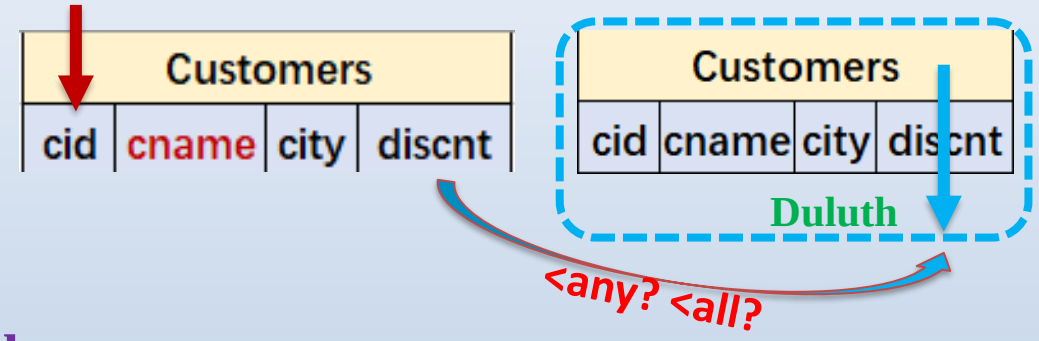


b) SQL:

```
select cid, cname  from customers c
        where discnt=some (select discnt from c
                where city in ('Dallas', 'Boston') );
```

3.4 Sub queries

[3] Example 3.4.9 Get *cid* of customers with *discnt* smaller than *discnt* of any customer who lives in *Duluth*.



a) Wrong:

```
select cid from customers where
        discnt<any(select discnt from customers where city='Duluth');
```

b) Right:

```
select cid from customers where
        discnt<all (select discnt from customers where city='Duluth');
```

[4] Equivalent relation :

expr = some (Subquery) ➔ expr IN (Subquery)

expr NOT IN (Subquery) ➔ expr <> all (Subquery)

expr NOT IN (Subquery) ≠ expr <> some (Subquery)

3.4 Sub queries

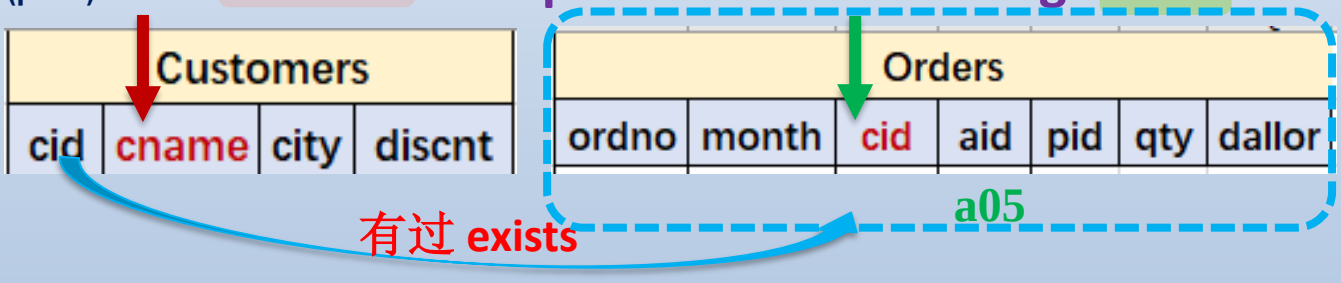
5. EXISTS Predicate

[1] Exists predicate: [NOT] EXIST (Subquery)

EXIST(subquery) is true <==> subquery is a non-empty set

NOT EXIST(subquery) is true <==> subquery is an empty set

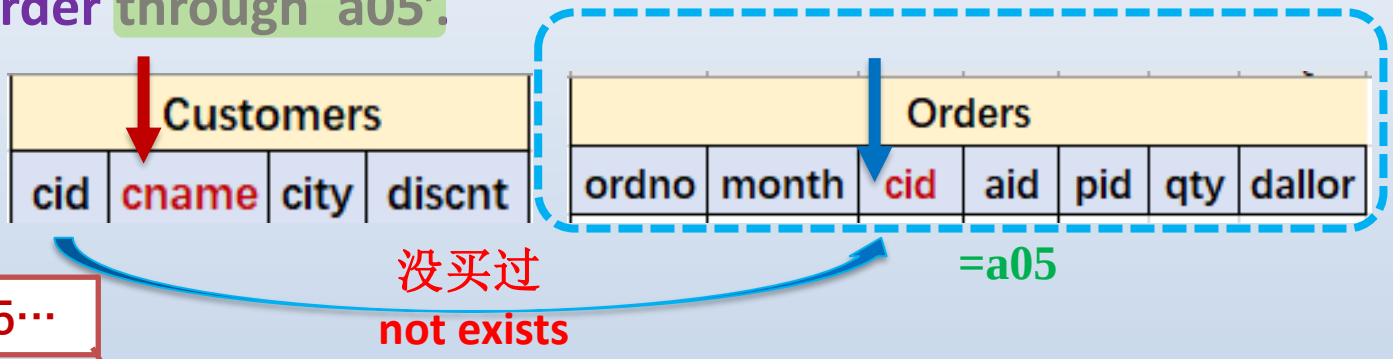
[2] Example 3.4.10 (p103) Find cnames who place an order through a05.




```
SQL1: select distinct c.cname from customers c
        where exists (select * from orders o
                      where o.aid = 'a05' and o.cid = c.cid );

SQL2: select distinct c.cname from customers c, orders o
        where o.aid = 'a05' and o.cid = c.cid;
```

3.4 Sub queries

[3]Example 3.4.12 (p104) Retrieve all customer names where the customer does not place an order through 'a05'.



差集: 全部cid – 从a05…

```
SQL1:  select distinct cname  from customers c
                where not exists ( select * from orders x
                                   where x.aid = 'a05' and x.cid = c.cid );
```

```
= SQL2 : select distinct cname  from customers c
                where c.cid not in ( select cid  from orders where aid= 'a05' );
```

```
= SQL3 : select distinct c.cname  from customers c
                where c.cid<>all (select cid  from orders where aid='a05');
```

3.5 UNION Operators & FOR ALL Conditions

1.The UNION Operator

[1] Union syntax : Subquery UNION [ALL] Subquery (INTERSECT)

[2]Example 3.5.1 (p108) We wish to list of cities where either a customer or an agent, or both, is based.

[ALL | distinct] [ALL | distinct]

```
SQL1: select city from customers union select city from agents;
                                    /* no duplicate rows */
SQL2: select city from customers
                          union all  select city from agents;
                                    /* has duplicate rows */
```

3.5 UNION Operators & FOR ALL Conditions

- **How to present 全称命题/蕴含关系?**

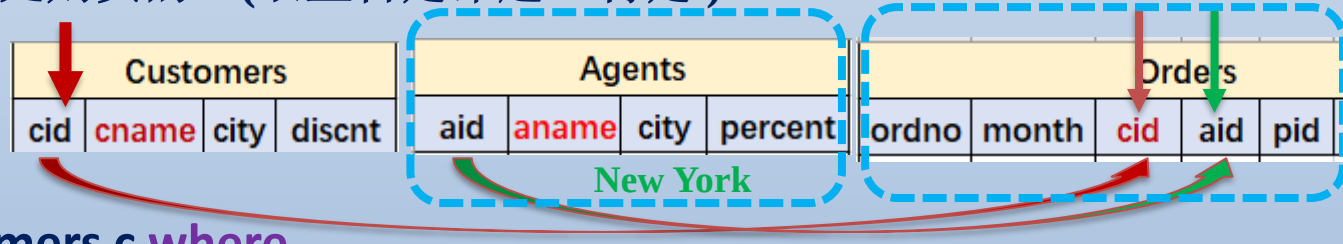
2.Division: SQL "for all…" Conditions(全称量词)

[1] Example 3.5.2 (p109) Find cids of customers who place orders with ALL agents based in New York.

（找cids，他的订单全都是New York的agents销售的）

a) Alg: ORDERS[cid, aid] ÷ (AGENTS where city = 'New York')[aid]

b) SQL: SQL中无全称量词，需做等价转换，题目变为"找cids，他没有一张订单不是从New York的agents处购买的"(双重否定命题➔肯定)



等价变换后SQL:

```
select cid from customers c where
    not exists (select * from agents a where city = 'New York' and
                not exists  (select * from orders x
                             where x.aid=a.aid  and  x.cid=c.cid) );
```

除法

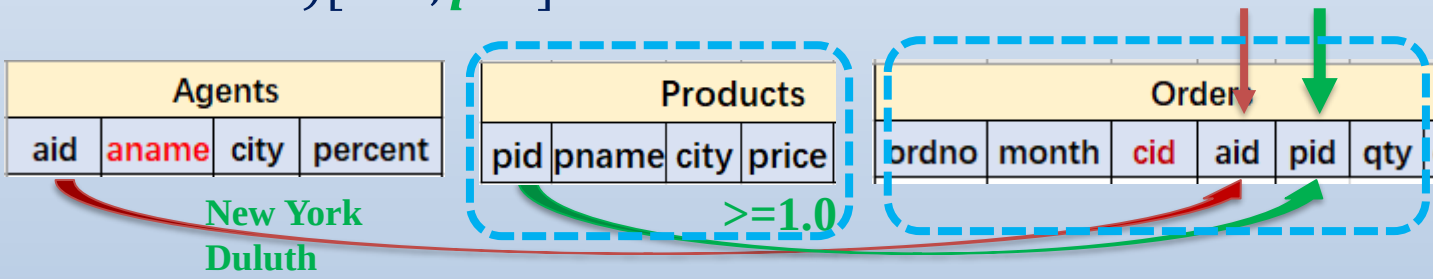
3.5 UNION Operators & FOR ALL Conditions

[2]Example 3.5.3 Get the aids of agents in New York or Duluth who place orders for all products costing more than a dollar. (找New York或Duluth的aids，他们销售所有单价>1美元的产品)

a) Alg $X := AGENTS\ where\ city\ in('NewYork','Duluth')[aid]$

$Y := (PRODUCTS\ where\ price > 1.0)[pid]$

$Z := (X \infty ORDERS \infty Y)[aid, pid] \div Y$



b) 等价：找New York/Duluth的aids，没有一件单价>1美元的产品不是他们销售的

```
select aid from agent a where  (a.city='New York' or a.city='Duluth')
        and not exists (select p.pid from product p where p.price>1.0
                and not exists (select * from orders x
                        where x.pid=p.pid and x.aid=a.aid) );
```

除法

3.5 UNION Operators & FOR ALL Conditions

[3] Summary: "FOR ALL" condition

select . . where not exists(select . . . where not exists(select . . .))

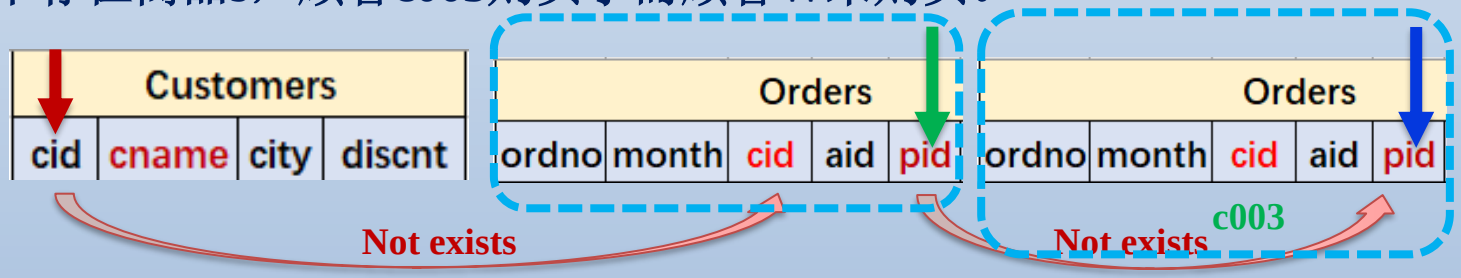
[4] 蕴含(Implication): p→q 在SQL中也不支持，可等价转换为

$$p \rightarrow q = \neg p \vee q$$

[5] Another Eg： 参考书P113~114. 查至少购买了"c003"顾客所购全部商品的顾客。

P：表示"c003"顾客购买了商品s。　Q：表示W顾客购买了商品s

等价表达：不存在商品s，顾客c003购买了而顾客W未购买。




```
select distinct cid from customers c where
    not exists ( select pid/* from orders x where x.cid=c.cid and
        not exists ( select pid/* from orders y
            where y.pid=x.pid and y.cid='c003') );
```

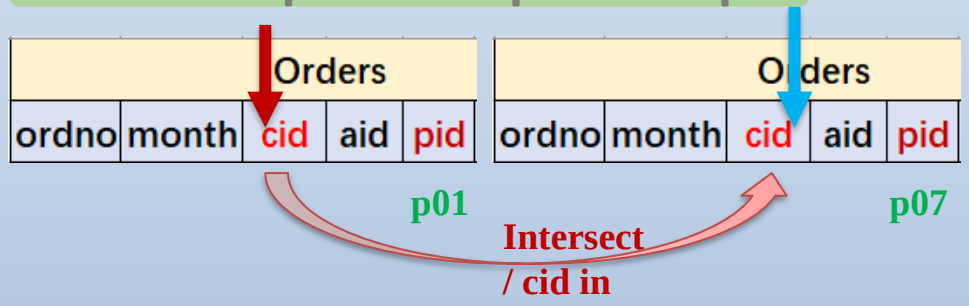
3.6 Some advanced SQL syntax

- **Another SQL Query?**

1. UNION, INTERSECT and EXCEPT Operators

[1] form subquery {UNION [ALL] | INTERSECT [ALL]
| EXCEPT [ALL] subquery}

[2] Example 3.6.1 Find cids who order both products p01 and p07.

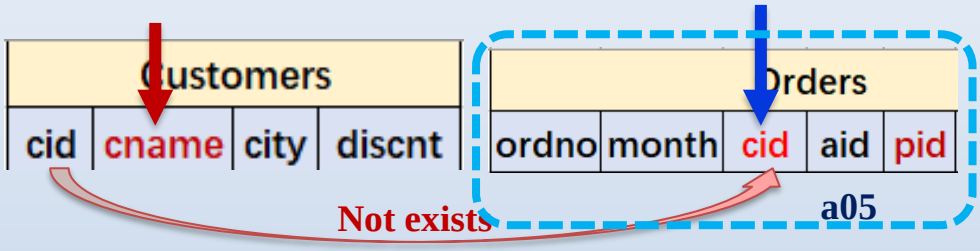


New) (select distinct cid from orders where pid = 'p01')
intersect (select cid from orders where pid = 'p07');

Old) select distinct cid from orders where pid = 'p01' and
cid in (select cid from orders where pid = 'p07');

3.6 Some advanced SQL syntax

[3] Example 3.6.2 (p116) Retrieve all cnames where the customer don't place any order through agent a05.



Old) select c.cname from customers c where
not exists (select * from orders x where x.aid='a05' and x.cid=c.cid);

New) select c.cname from customers c
except (select c.cname from customers c, orders x
where x.aid = 'a05' and x.cid = c.cid);

[4] Operators in DBMS product

- ORACLE 8 : {UNION, UNION ALL, INTERSECT, EXCEPT}
- DB2 Version 5 : {UNION, INTERSECT, EXCEPT} [ALL]
- INFORMIX : UNION [ALL]

3.6 Some advanced SQL syntax

2. Join in advanced SQL

a) Join Predicate

Tableref := tablename [[AS] corr_name[(colname {, colname ...})]]

| (subquery) [AS] corr_name [(colname {, colname ...})]

| tableref1 [INNER | { LEFT | RIGHT | FULL}

[OUTER] JOIN tableref2 ON search_condition |

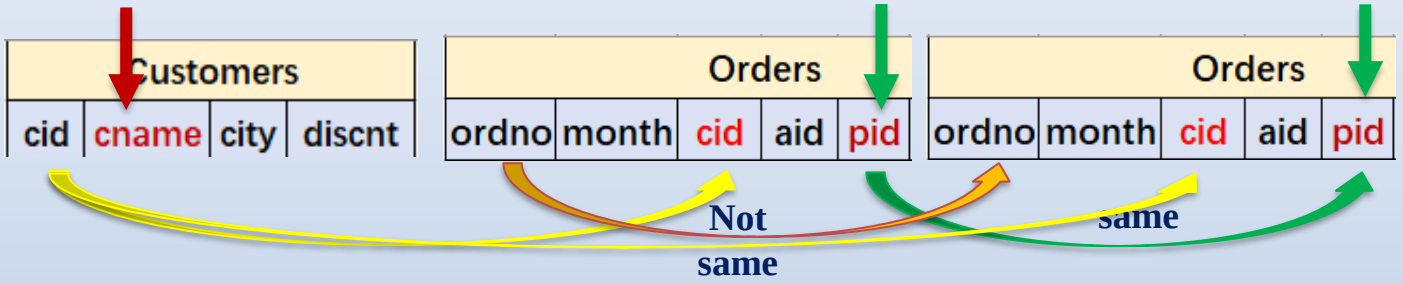
USING (colname {, colname . . .})}

b) FROM 子句中含Subquery

clause := FROM tableref {, tableref . . .}

3.6 Some advanced SQL syntax

[2]Example 3.6.3(p118) Retrieve all cnames where the customer places at least two orders for the same product.



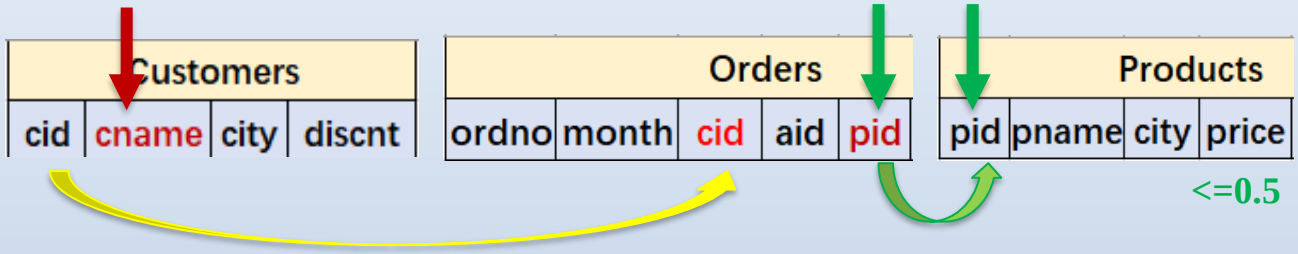

```
S1)  select cname from  (select o.cid as spcid from orders o, orders x
        where o.cid=x.cid and o.pid=x.pid and o.ordno<>x.ordno) as y,
                                        customers c where y.spcid = c.cid;
```

（from子句中出现Subquery：先求View y，再作其它处理）

```
S2)  select cname from customers c, orders o,  orders x
        where c.cid=o.cid and o.pid=x.pid and o.ordno<>x.ordno;
```

3.6 Some advanced SQL syntax

[3] Example 3.6.4 (p119) **Retrieve all customers who has buy at least one product costing less than $0.50.**



a)Alg. (((P where price <0.50)[pid]) ∞ O ∞ C) [cname]

b)New1
```
select distinct cname from (O join P on O.pid = P.pid )
                join C on O.cid =  C.cid where P.price <0.50;
```

New2
```
select distinct cname from  (O join  P using(pid) )
                join C using(cid)  where price<0.50;
```

Old3
```
select cname from Customers C, Orders O, Products P
       where C.cid=O.cid and O.pid=P.pid and P.price<0.50;
```

3.6 Some advanced SQL syntax

3. Outer Join

[1] Form: [{LEFT | RIGHT | FULL} [OUTER]] JOIN

[2] Example: (p120) We have two tables S and T

(a) select * from S full outer join T using(A) ;

(b) select * from S left outer join T using(A)

UNION select * from S right outer join T using(A);

(a)

S.C	S.A	T.A	T.B
c1	a1	a1	b1
c3	a3	a3	b3
c4	a4	a4	b4
		a2	b2

?

(b)

S.C	S.A	T.B
c1	a1	b1
c3	a3	b3
c4	a4	b4
null	a2	b2