

# Introduction to Data Structure (Data Management) Lecture 6

Felipe P. Vista IV



Chonbuk National University

- 1 -

Global Frontier College

INTRO TO DATA STRUCTURE

# NESTED QUERIES

- More powerful SQL queries
- Subqueries (6.3)

# Subqueries

- Subquery
  - a SQL query nested inside a larger query
  - also called nested queries



# Subqueries

- Subquery
  - a SQL query nested inside a larger query
  - also called nested queries
- Subquery may be used in
  - a **SELECT** clause
  - a **FROM** clause
  - a **WHERE** clause

\* clause – conditions



# Subqueries

- Subquery
  - a SQL query nested inside a larger query
  - also called nested queries
- Subquery may be used in
  - a **SELECT** clause
  - a **FROM** clause
  - a **WHERE** clause
- Rule of thumb:
  - avoid nested queries if possible
  - although sometimes it's impossible to avoid

\* clause – conditions



# Subqueries

- **WHERE** clause
  - can return a single constant & this constant can be compared w/ another value
  - can return relations that can be used in various ways

\* clause – conditions



# Subqueries

- **WHERE** clause
  - can return a single constant & this constant can be compared w/ another value
  - can return relations that can be used in various ways
- **FROM** clause
  - can appear in FROM clause, followed by a tuple variable that represents the tuples in the result of a subquery

\* clause – conditions





# Subqueries

- **WHERE** clause
  - can return a single constant & this constant can be compared w/ another value
  - can return relations that can be used in various ways
- **FROM** clause
  - can appear in FROM clause, followed by a tuple variable that represents the tuples in the result of a subquery
- **SELECT** clause
  - can appear as computed values

\* clause – conditions



# Subqueries in SELECT

→ product (pname, price, cid)

→ company (cid, cname, city)

For each product, return the city where it is manufactured.

```
SELECT X.pname, (SELECT Y.city
                  FROM   Company Y
                  WHERE  Y.cid=X.cid) AS City,
FROM Product X
```

pname	price	cid
bike	119.95	C003
scooter	255.00	c004
genesis	450.99	C001
eBike	210.00	c003

cid	cname	city
C003	Alton	Nabas
c001	Hyundai	Jeonju
C002	BMW	Chennai

What will happen if subquery returns more than one city?

There will be a runtime error!

But, SQLite will simply ignore the extra values

"correlated subquery"

\* Correlated subquery (synchronized subquery) – a subquery that use values from the outer query

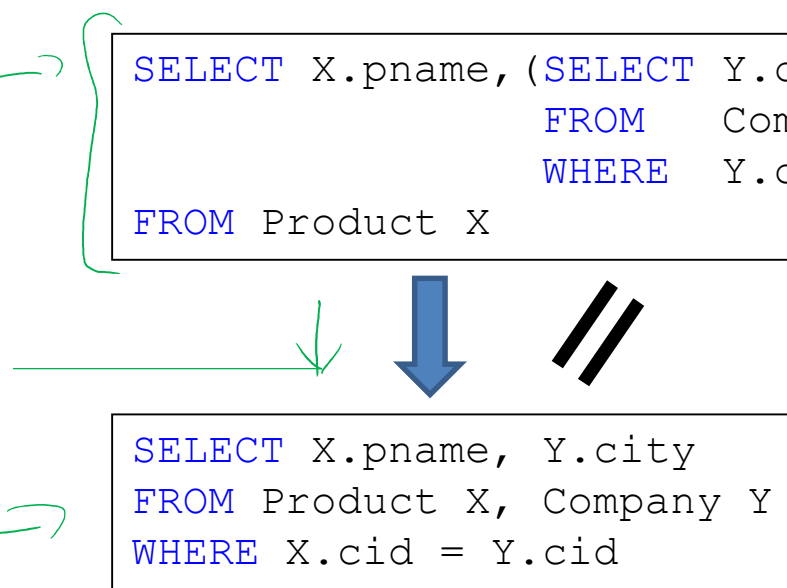


# Subqueries in SELECT

product (pname, price, cid)

company (cid, cname, city)

Whenever possible, do not use nested queries:



```
SELECT X.pname, (SELECT Y.city
                  FROM   Company Y
                  WHERE  Y.cid=X.cid) AS City,
FROM Product X
```

```
SELECT X.pname, Y.city
FROM Product X, Company Y
WHERE X.cid = Y.cid
```

We just  
"unnested"  
the query

DBMS do  
this also

# Subqueries in SELECT

product(pname, price, cid) –  
company(cid, cname, city)

Compute the number of products made by each company

```
SELECT DISTINCT C.cname, (SELECT (*)  
                           FROM Product P  
                           WHERE P.cid=C.cid)  
FROM Product X Company C
```

Suggested:  
Unnest by using **GROUP BY**

```
SELECT C.cname, count(*)  
FROM Company C, Product P  
WHERE C.cid = P.cid  
GROUP BY C.cname
```

# Subqueries in SELECT

product (pname, price, cid)

company (cid, cname, city)

Are they really equivalent?

(1)   
 SELECT DISTINCT C.cname, (SELECT (\*)  
 FROM Product P  
 WHERE P.cid=C.cid),  
 FROM ~~Product X~~ *Company C*

**== ???**

We get different result if a  
company has no product

(2)   
 SELECT C.cname, count(\*)  
 FROM Company C, Product P  
 WHERE C.cid = P.cid  
 GROUP BY C.cname

SELECT C.cname, count(pname)  
 FROM Company C LEFT OUTER JOIN  
 Product P  
 ON C.cid = P.cid  
 GROUP BY C.cname



# Subqueries in FROM

product (pname, price, cid)

company (cid, cname, city)

Find all products whose prices are >200 and < 400

```
→ SELECT X.pname  
→ FROM (SELECT * FROM Product AS Y WHERE price > 200) AS X  
→ WHERE price < 400
```

Unnesting...



```
→ SELECT pname  
FROM Product  
WHERE price > 200 AND price < 400
```