Lecture 10: SQL basic 4

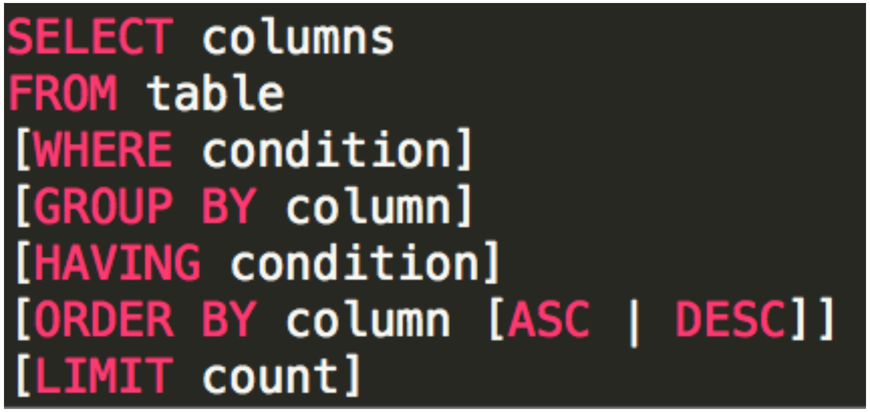
BADM/ACCY 352

Spring 2017

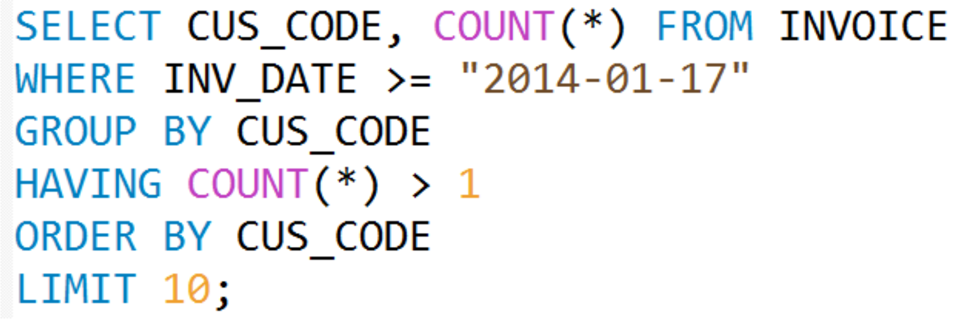
Instructor: Yi Yang, PhD

- Lab 2 due: This Wednesday, 23:59pm
- Test 1 review: This Wednesday, Mar 1st
- Test 1: Next Monday, Mar 6th

Last lecture



```
SELECT columns
FROM table
[WHERE condition]
[GROUP BY column]
[HAVING condition]
[ORDER BY column [ASC | DESC]]
[LIMIT count]
```

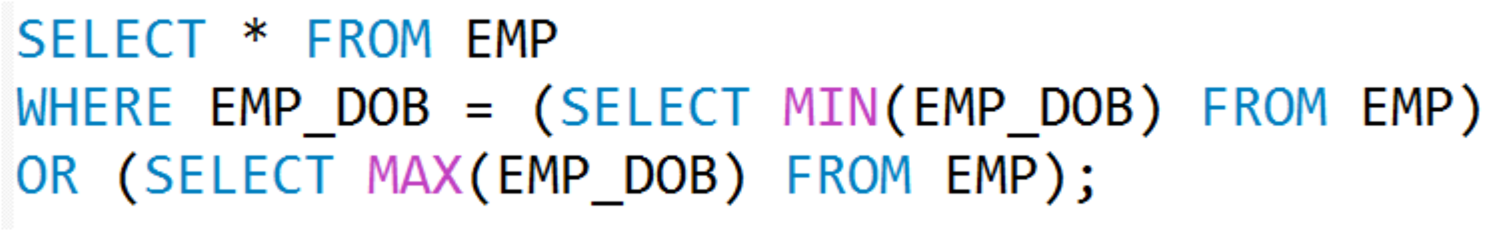


```
SELECT CUS_CODE, COUNT(*) FROM INVOICE
WHERE INV_DATE >= "2014-01-17"
GROUP BY CUS_CODE
HAVING COUNT(*) > 1
ORDER BY CUS_CODE
LIMIT 10;
```

- Know how to use date functions with aggregate functions and Group By clause.

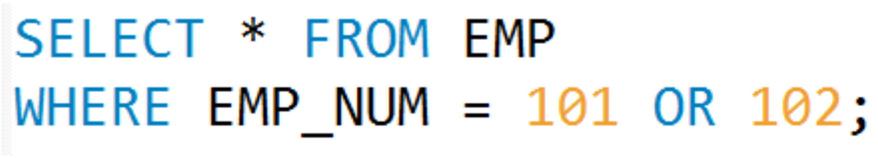
Lab 2

- List the oldest and the youngest employee in one table.



```
SELECT * FROM EMP
WHERE EMP_DOB = (SELECT MIN(EMP_DOB) FROM EMP)
OR (SELECT MAX(EMP_DOB) FROM EMP);
```

❌



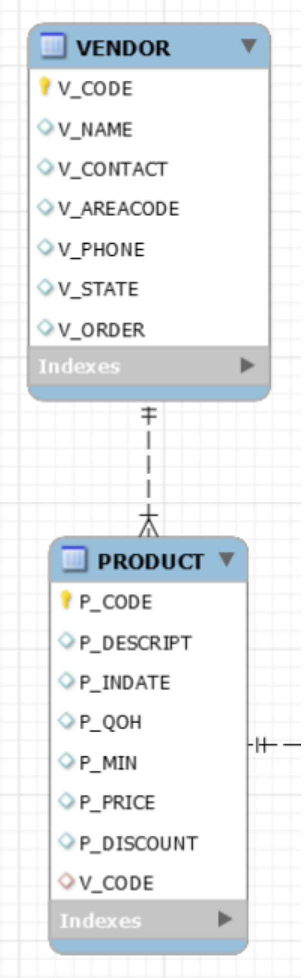
```
SELECT * FROM EMP
WHERE EMP_NUM = 101 OR 102;
```

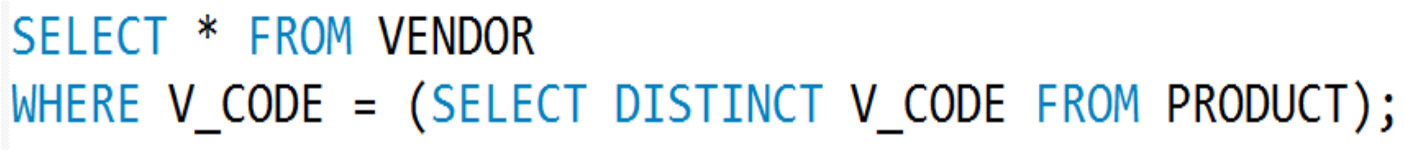
❌

OR needs two comparison statements.

Lab 2

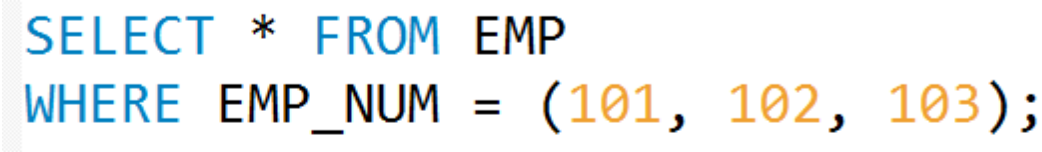
- The VENDOR table contains information about each vendor. List detailed information of vendors that supply products.





```sql
SELECT * FROM VENDOR
WHERE V_CODE = (SELECT DISTINCT V_CODE FROM PRODUCT);
```

✗



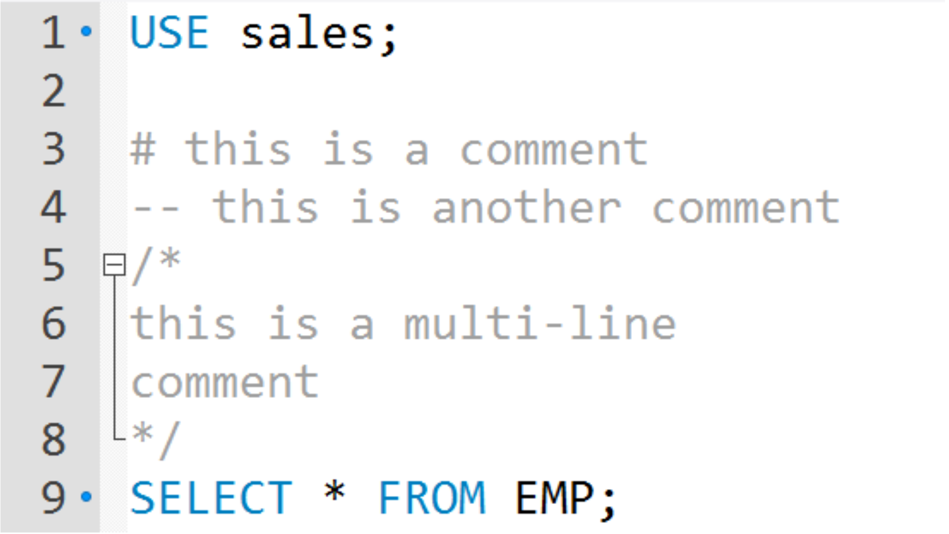
```sql
SELECT * FROM EMP
WHERE EMP_NUM = (101, 102, 103);
```

✗

IN is used to match any value in a value list

SQL comment

- A comment is an annotation to make the code easier for humans to understand, and it will not get executed.
- Three ways to make comment in SQL.
  - From a “#” character to the end of the line.
  - From a “-- ” sequence to the end of the line.
  - From a /* sequence to the following */ sequence

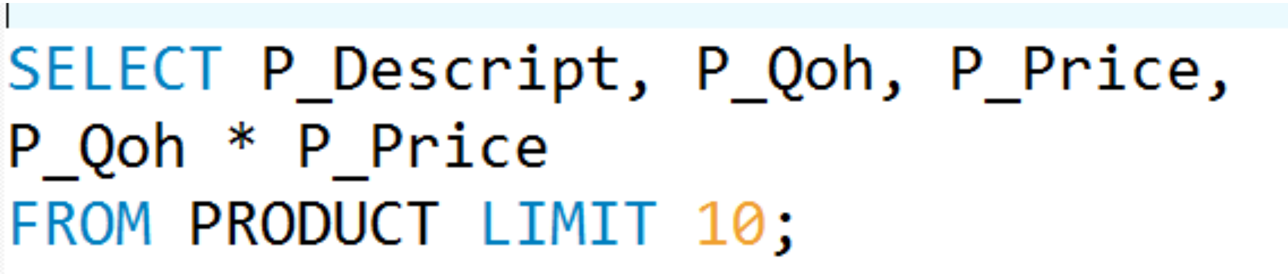


```sql
USE sales;

# this is a comment
-- this is another comment
/*
this is a multi-line
comment
*/
SELECT * FROM EMP;
```

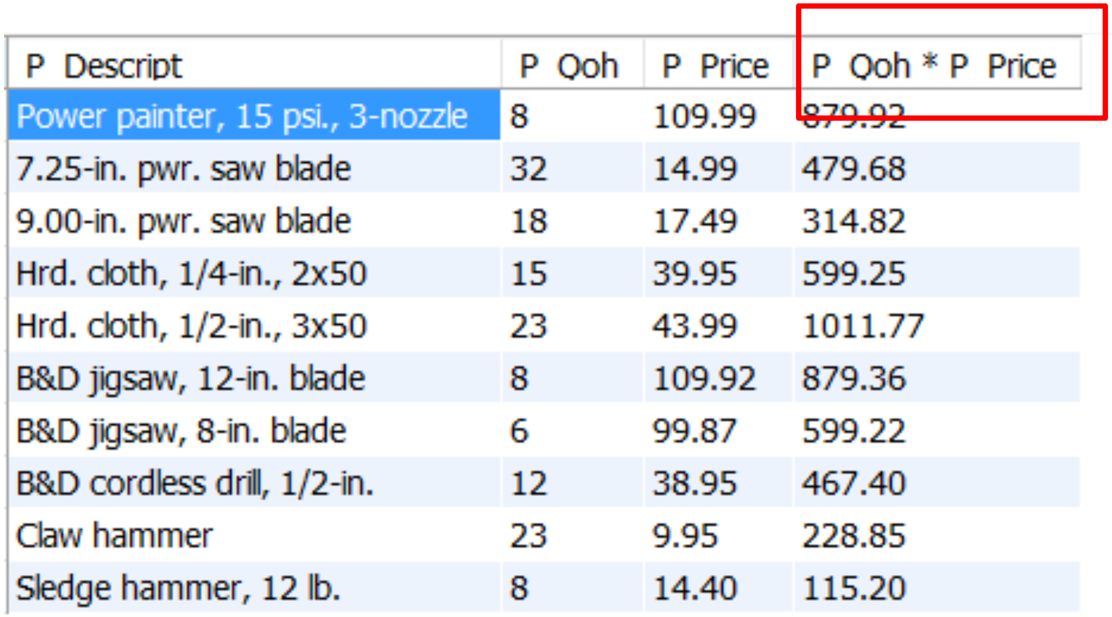
Return a derived column

- You can create a column whose value is computed from other columns.



```
SELECT P_Descript, P_Qoh, P_Price,
P_Qoh * P_Price
FROM PRODUCT LIMIT 10;
```

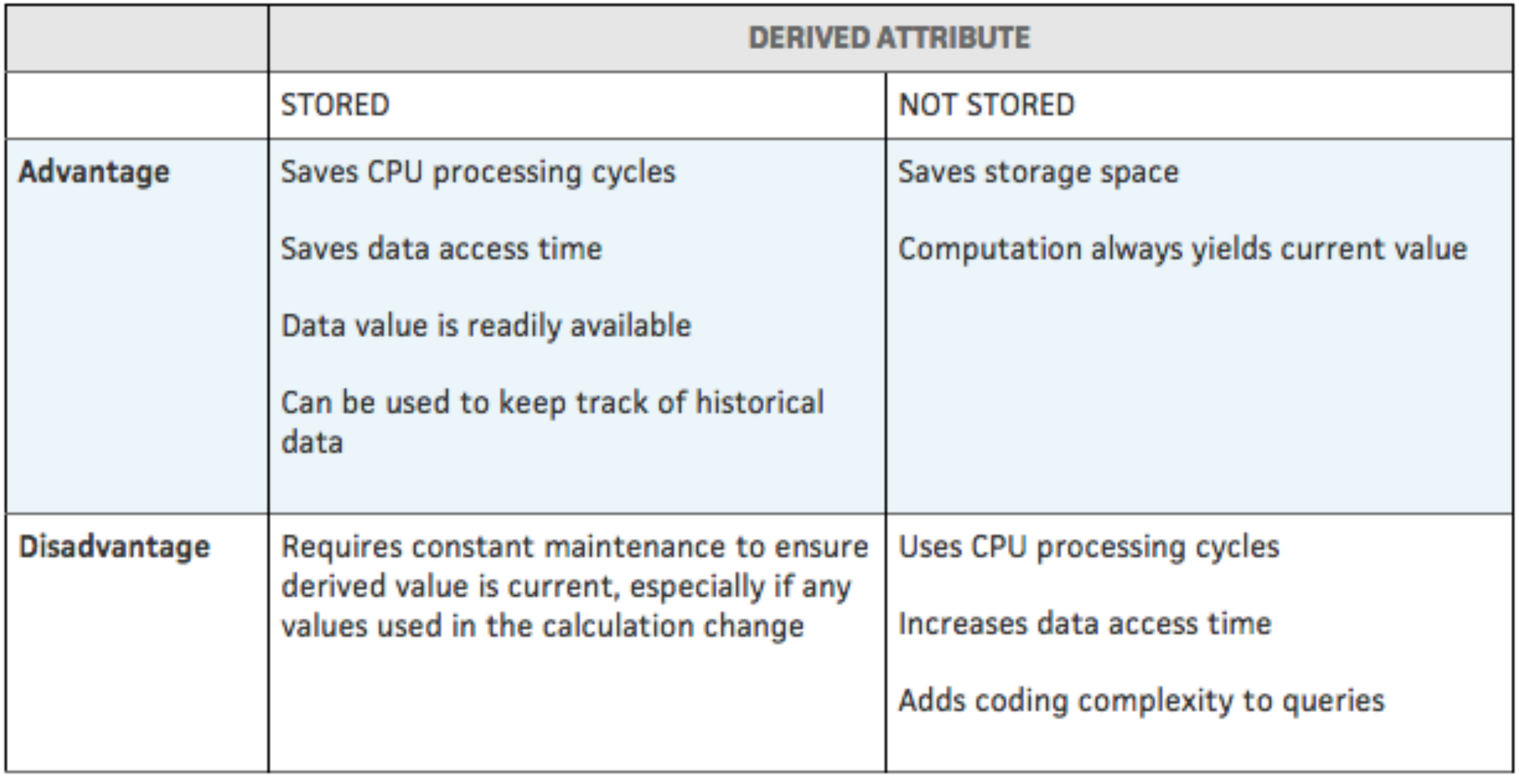
Results:



P Descript	P Qoh	P Price	P Qoh * P Price
Power painter, 15 psi., 3-nozzle	8	109.99	879.92
7.25-in. pwr. saw blade	32	14.99	479.68
9.00-in. pwr. saw blade	18	17.49	314.82
Hrd. cloth, 1/4-in., 2x50	15	39.95	599.25
Hrd. cloth, 1/2-in., 3x50	23	43.99	1011.77
B&D jigsaw, 12-in. blade	8	109.92	879.36
B&D jigsaw, 8-in. blade	6	99.87	599.22
B&D cordless drill, 1/2-in.	12	38.95	467.40
Claw hammer	23	9.95	228.85
Sledge hammer, 12 lb.	8	14.40	115.20

Side note on derived attribute

A derived attribute is an attribute whose value is calculated from other attributes. Example: EMP_AGE.

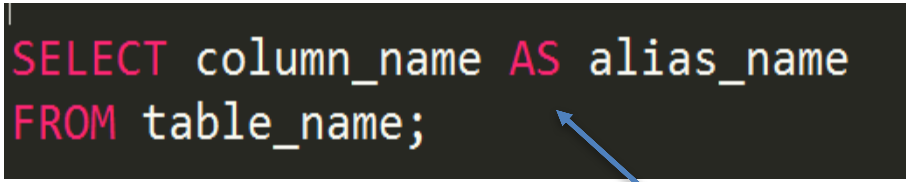


	DERIVED ATTRIBUTE	
	STORED	NOT STORED
Advantage	Saves CPU processing cycles Saves data access time Data value is readily available Can be used to keep track of historical data	Saves storage space Computation always yields current value
Disadvantage	Requires constant maintenance to ensure derived value is current, especially if any values used in the calculation change	Uses CPU processing cycles Increases data access time Adds coding complexity to queries

SQL Alias

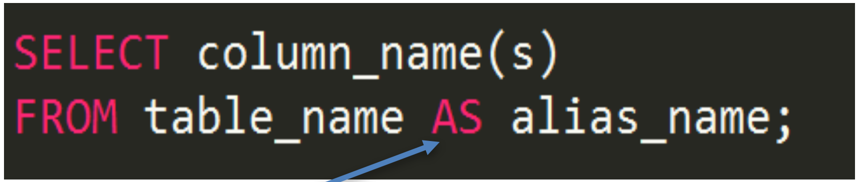
- SQL aliases are used to temporarily rename a table or a column heading.
- Basically aliases are created to make column names more readable.

SQL Alias Syntax for Columns



```
SELECT column_name AS alias_name
FROM table_name;
```

SQL Alias Syntax for Tables

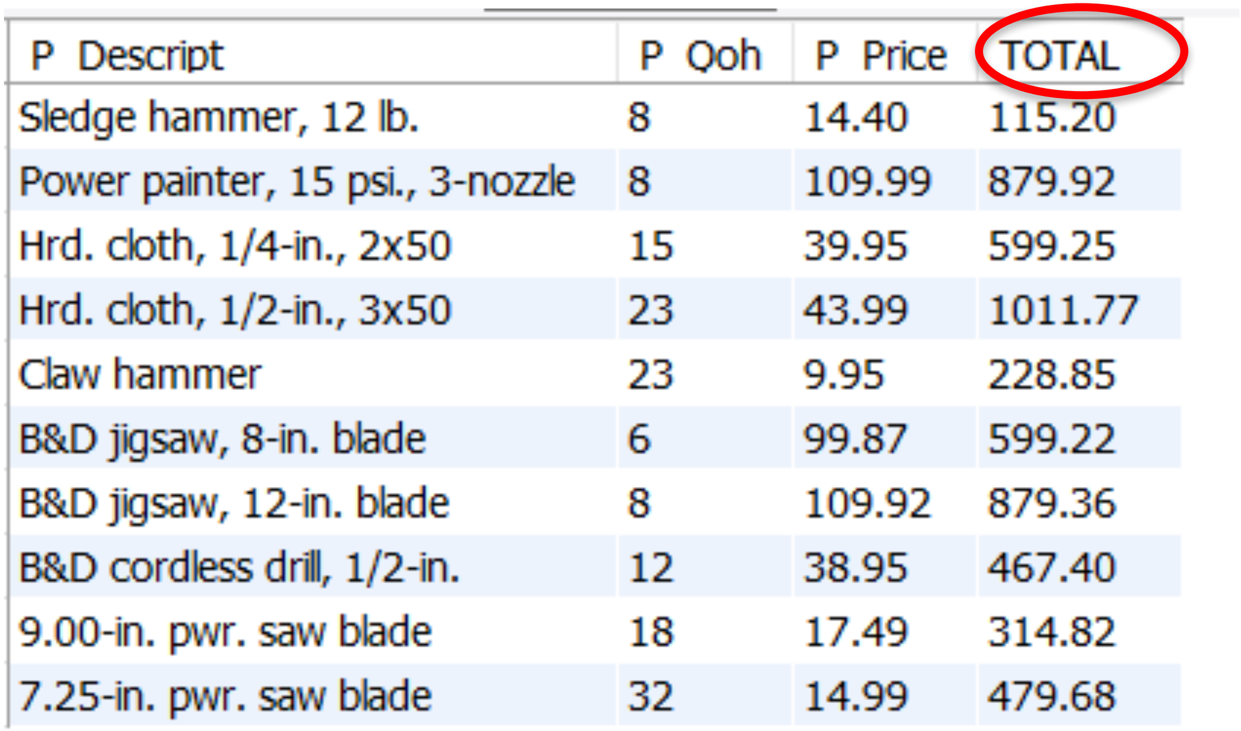


```
SELECT column_name(s)
FROM table_name AS alias_name;
```

Using keywords AS

SQL Column Alias

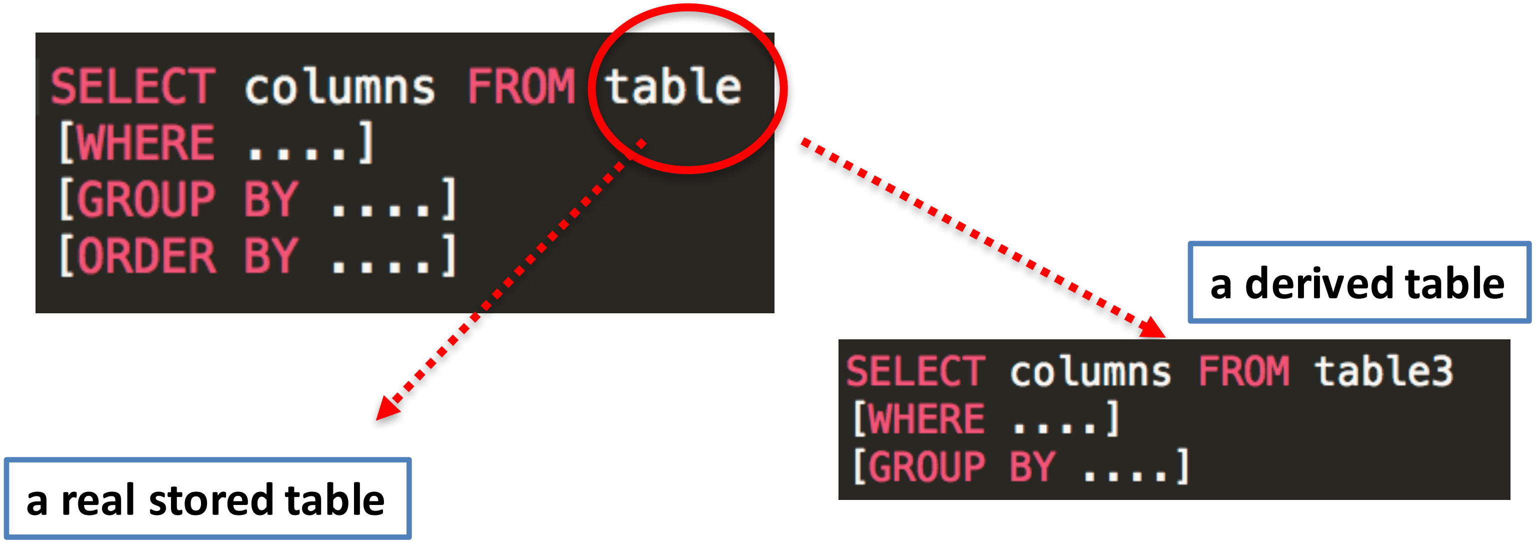
```
SELECT P_Descript, P_Qoh, P_Price,
P_Qoh * P_Price As TOTAL
FROM PRODUCT LIMIT 10;
```



P Descript	P Qoh	P Price	TOTAL
Sledge hammer, 12 lb.	8	14.40	115.20
Power painter, 15 psi., 3-nozzle	8	109.99	879.92
Hrd. cloth, 1/4-in., 2x50	15	39.95	599.25
Hrd. cloth, 1/2-in., 3x50	23	43.99	1011.77
Claw hammer	23	9.95	228.85
B&D jigsaw, 8-in. blade	6	99.87	599.22
B&D jigsaw, 12-in. blade	8	109.92	879.36
B&D cordless drill, 1/2-in.	12	38.95	467.40
9.00-in. pwr. saw blade	18	17.49	314.82
7.25-in. pwr. saw blade	32	14.99	479.68

Derived table

- A **derived table** is essentially a local temporary **table** created by means of a subquery in the FROM clause of a SQL SELECT statement.
- It exists only in memory and behaves like a **table**.
- SQL standards for **Structured** Query Language.



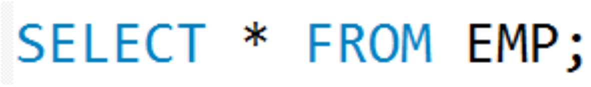
```
SELECT columns FROM table
[WHERE ....]
[GROUP BY ....]
[ORDER BY ....]
```

a real stored table

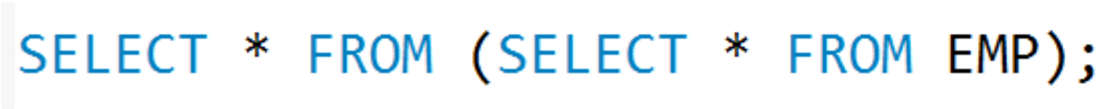
a derived table

```
SELECT columns FROM table3
[WHERE ....]
[GROUP BY ....]
```

Derived table example



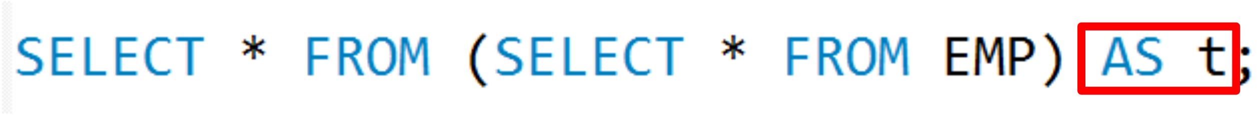
```sql
SELECT * FROM EMP;
```



```sql
SELECT * FROM (SELECT * FROM EMP);
```

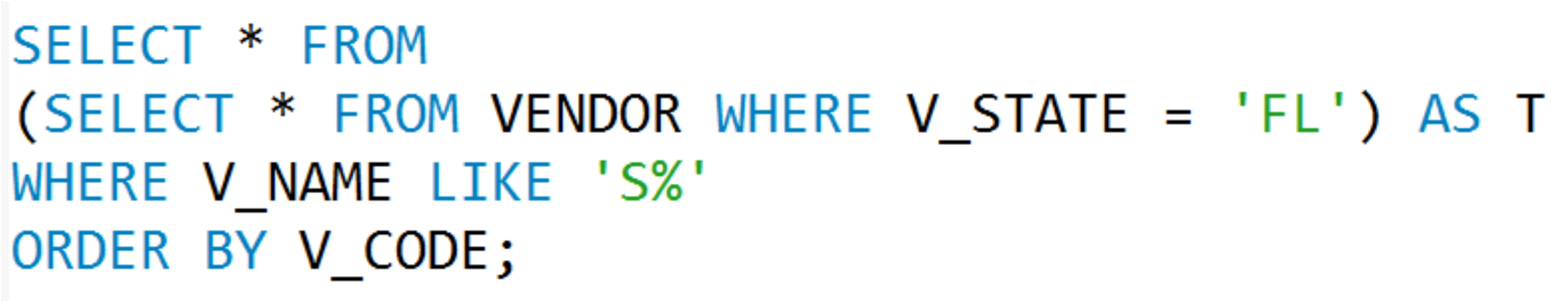
✗

Derived table must have its own alias.

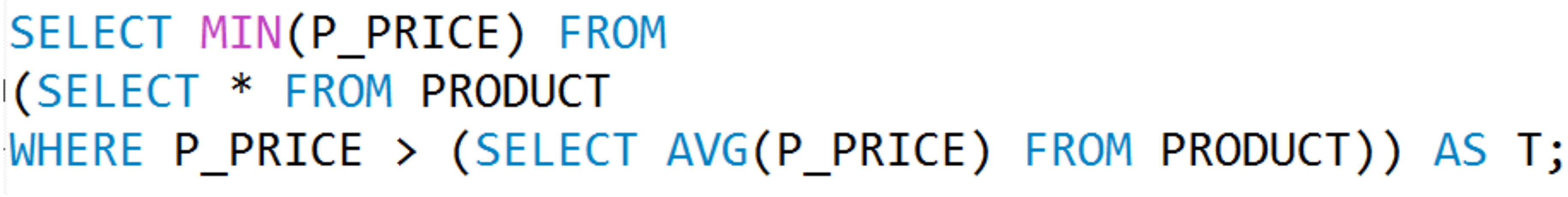


```sql
SELECT * FROM (SELECT * FROM EMP) AS t;
```

Derived table example



```
SELECT * FROM
(SELECT * FROM VENDOR WHERE V_STATE = 'FL') AS T
WHERE V_NAME LIKE 'S%'
ORDER BY V_CODE;
```

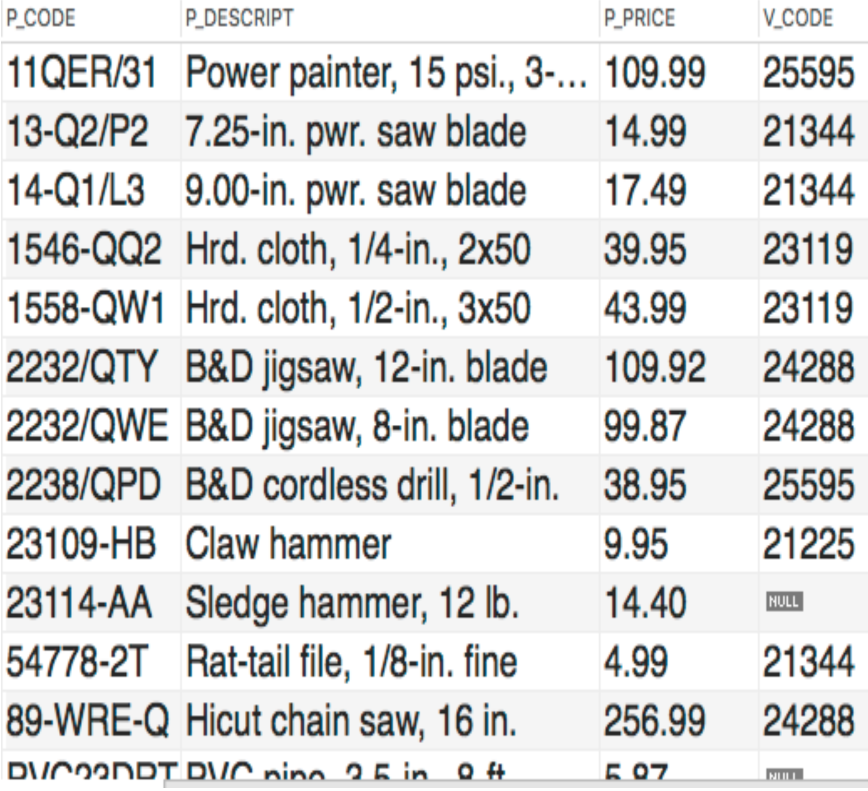


```
SELECT MIN(P_PRICE) FROM
(SELECT * FROM PRODUCT
WHERE P_PRICE > (SELECT AVG(P_PRICE) FROM PRODUCT)) AS T;
```

Derived table is very useful when you want to aggregate results from an aggregate results

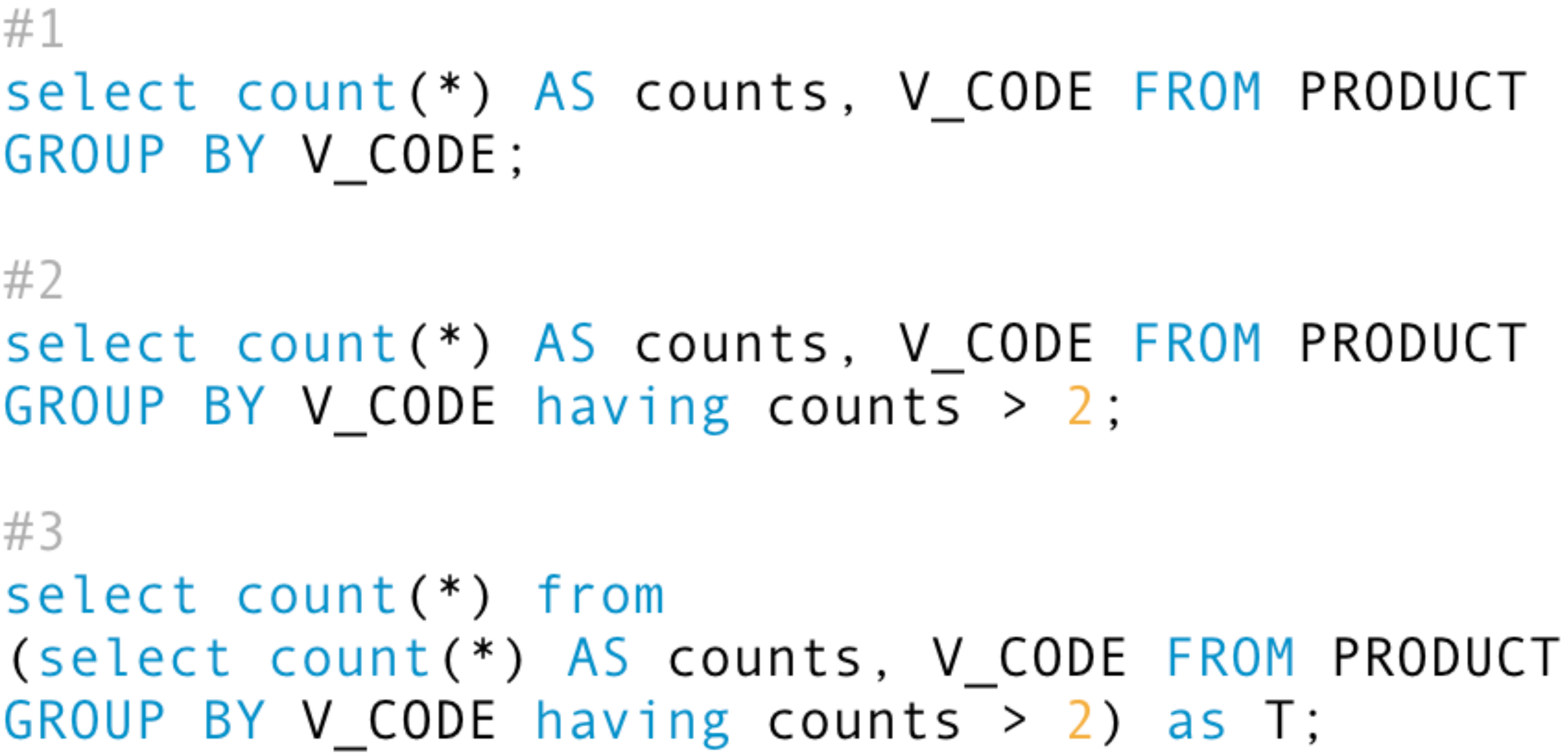
Derived table practice

1. Count the total number of products for each vendor.
2. List the vendor who supplies more than 2 products.
3. Count the number of vendors who supply more than 2 products.



P_CODE	P_DESCRIPT	P_PRICE	V_CODE
11QER/31	Power painter, 15 psi., 3-...	109.99	25595
13-Q2/P2	7.25-in. pwr. saw blade	14.99	21344
14-Q1/L3	9.00-in. pwr. saw blade	17.49	21344
1546-QQ2	Hrd. cloth, 1/4-in., 2x50	39.95	23119
1558-QW1	Hrd. cloth, 1/2-in., 3x50	43.99	23119
2232/QTY	B&D jigsaw, 12-in. blade	109.92	24288
2232/QWE	B&D jigsaw, 8-in. blade	99.87	24288
2238/QPD	B&D cordless drill, 1/2-in.	38.95	25595
23109-HB	Claw hammer	9.95	21225
23114-AA	Sledge hammer, 12 lb.	14.40	NULL
54778-2T	Rat-tail file, 1/8-in. fine	4.99	21344
89-WRE-Q	Hicut chain saw, 16 in.	256.99	24288
PVC23DRT	PVC pipe, 3.5-in., 8-ft	5.97	NULL

Derived table example



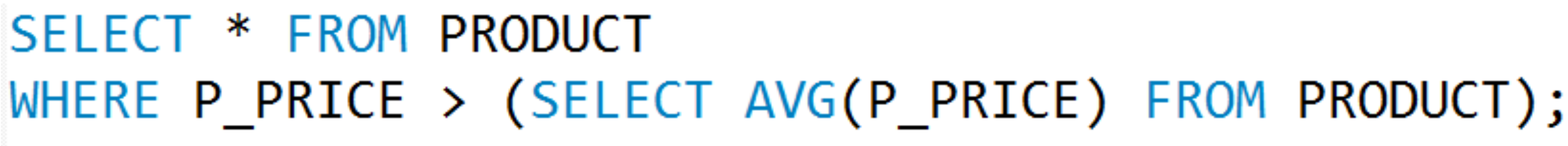
```
#1
select count(*) AS counts, V_CODE FROM PRODUCT
GROUP BY V_CODE;

#2
select count(*) AS counts, V_CODE FROM PRODUCT
GROUP BY V_CODE having counts > 2;

#3
select count(*) from
(select count(*) AS counts, V_CODE FROM PRODUCT
GROUP BY V_CODE having counts > 2) as T;
```

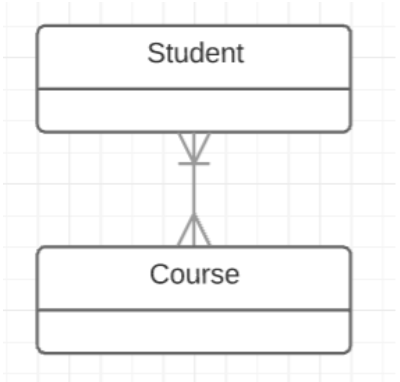
Derived table vs. Subquery

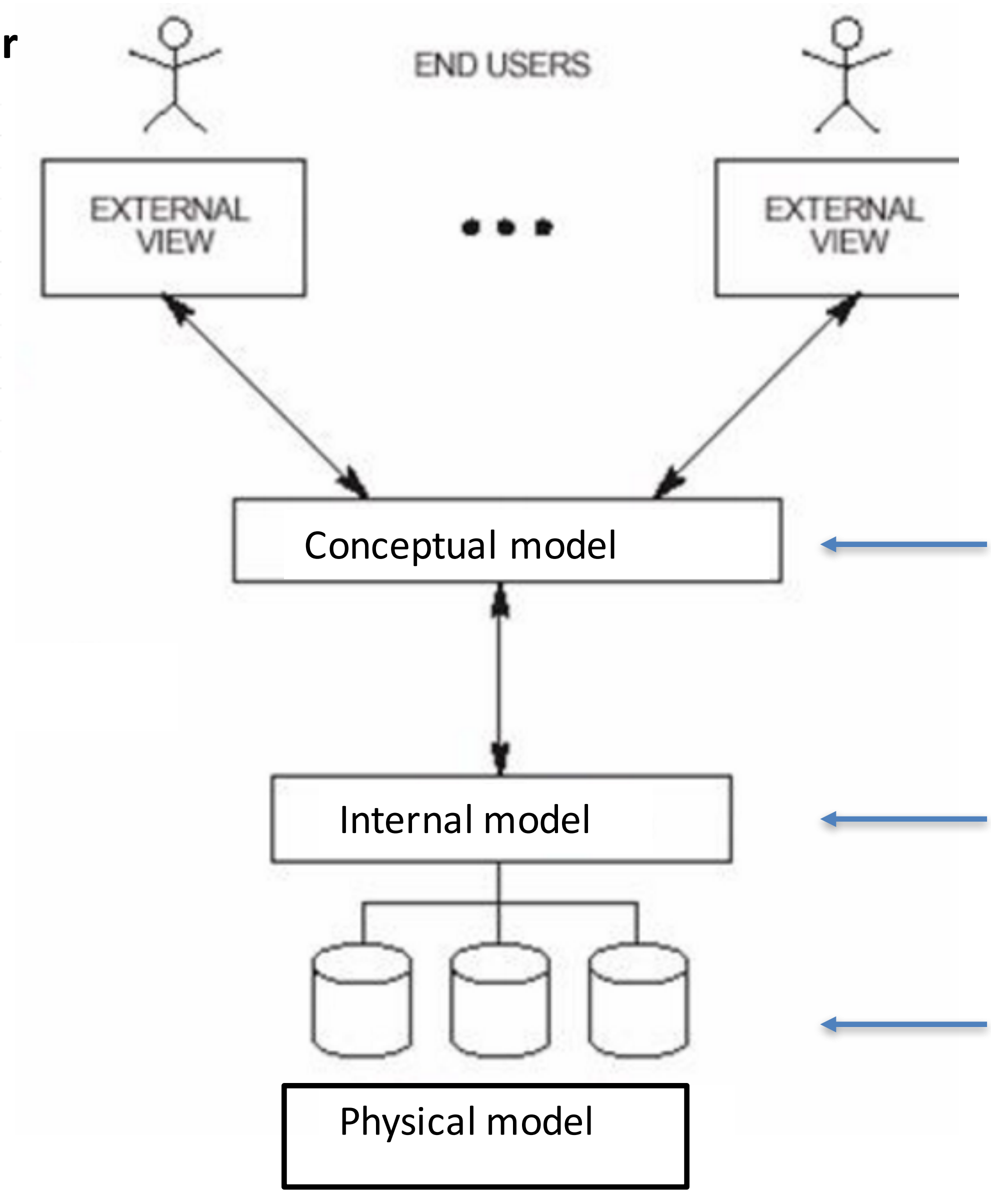
- derived tables are used in the FROM clause
- subqueries are used in the WHERE clause



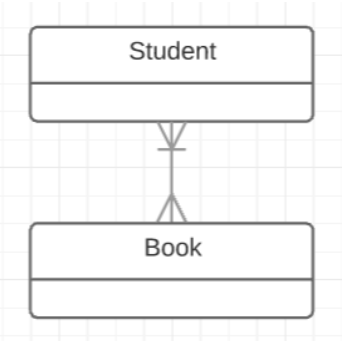
```
SELECT * FROM PRODUCT
WHERE P_PRICE > (SELECT AVG(P_PRICE) FROM PRODUCT);
```

Office of the Registrar





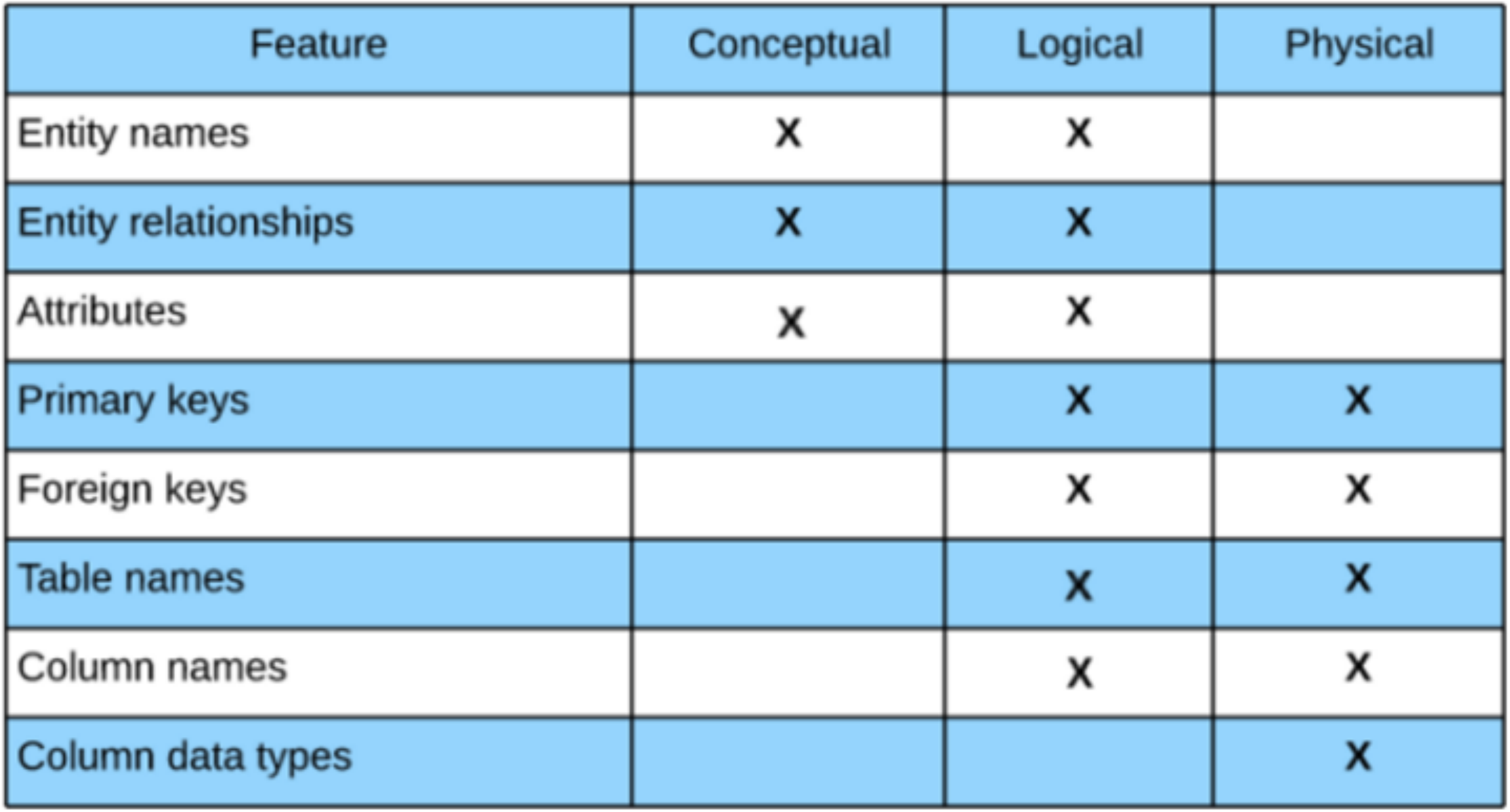
University Library



Entity Relationship Modeling

Relational Modeling

SQL



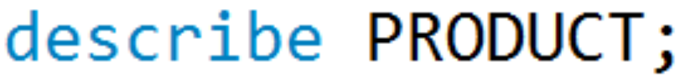
Feature	Conceptual	Logical	Physical
Entity names	X	X	
Entity relationships	X	X	
Attributes	X	X	
Primary keys		X	X
Foreign keys		X	X
Table names		X	X
Column names		X	X
Column data types			X

Data Types

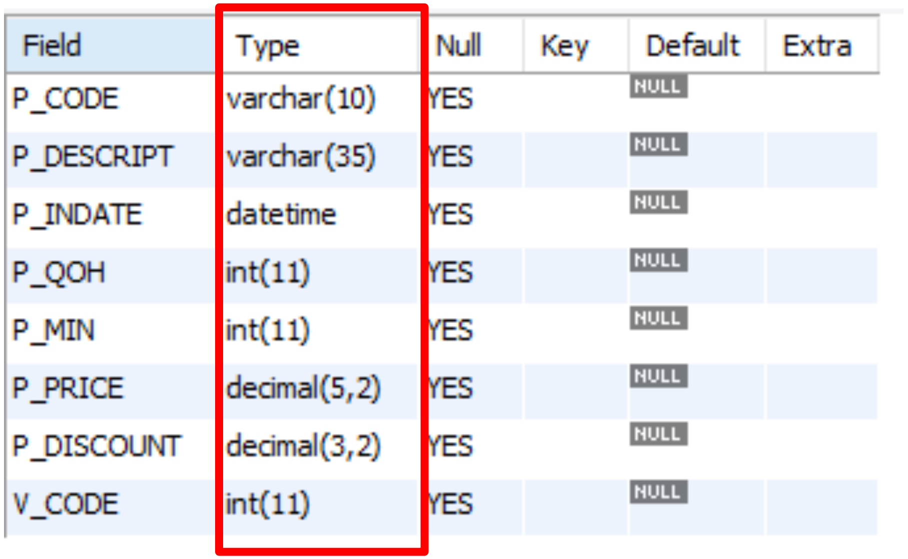
- A data type defines what kind of value a column can contain.
- Each column in a database table is required to have a name and a data type.
- In MySQL there are three main types
  - String/Text
  - Number
  - Date/Time (We have already discussed)
- Data type is usually chosen by nature of data and by intended use

Data Types

- To see the data types of columns in a table. Use the keyword *describe*.



```
describe PRODUCT;
```



Field	Type	Null	Key	Default	Extra
P_CODE	varchar(10)	YES		NULL	
P_DESCRIPT	varchar(35)	YES		NULL	
P_INDATE	datetime	YES		NULL	
P_QOH	int(11)	YES		NULL	
P_MIN	int(11)	YES		NULL	
P_PRICE	decimal(5,2)	YES		NULL	
P_DISCOUNT	decimal(3,2)	YES		NULL	
V_CODE	int(11)	YES		NULL	

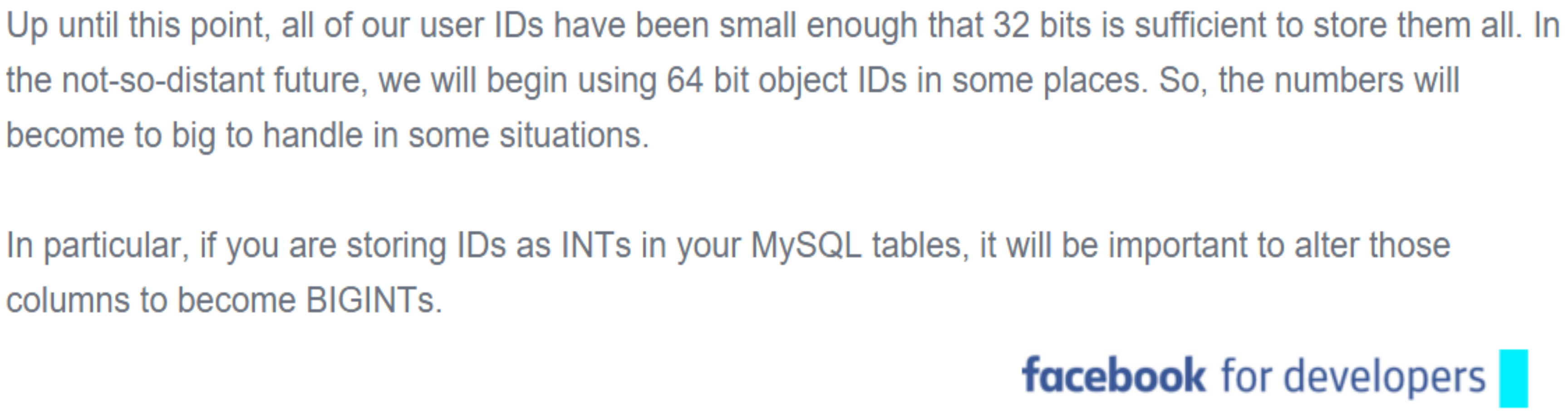
Numeric data type

- **INT** - An integer. The allowable range is from 0 to 4294967295. (2^32, i.e. 32 bits, 4bytes).
- **TINYINT** -- the allowable range is from 0 to 255. (2^8, i.e., 8 bits, 1bytes.)
- **BIGINT --** the allowable range is from 0 to 18446744073709551615. (2^64, i.e., 64bits, 8bytes)

- **DECIMAL(M,D)** - A floating-point number . Defining the display length (M) and the number of decimals (D) is required.
- **DECIMAL(5,2)** be able to store any value with five digits and two decimals, so values that can be from -999.99 to 999.99.

Numeric data type

- Which data type can be used for the following column?
- University student ID
- Facebook User ID



Up until this point, all of our user IDs have been small enough that 32 bits is sufficient to store them all. In the not-so-distant future, we will begin using 64 bit object IDs in some places. So, the numbers will become to big to handle in some situations.

In particular, if you are storing IDs as INTs in your MySQL tables, it will be important to alter those columns to become BIGINTs.

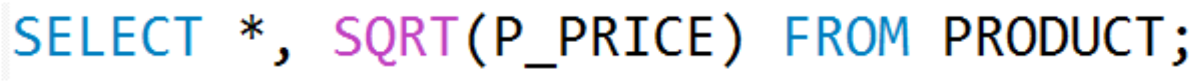
facebook for developers

Random generated number for PK

- Database system automatically assigns a unique random number as PK value when an entry is inserted.
- https://www.facebook.com/illinois.edu/
- https://www.facebook.com/profile.php?id=163536409904

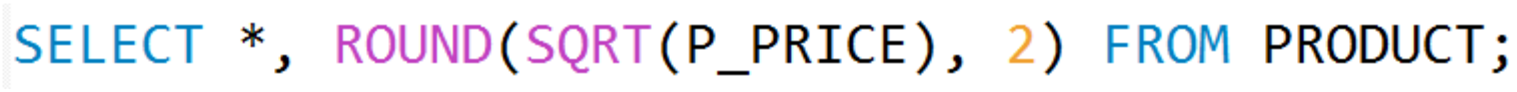
MySQL numeric functions

- MySQL provides several functions that operate on numeric data types for mathematical calculations.
- SQRT(x): Return the square root of the value x.



```
SELECT *, SQRT(P_PRICE) FROM PRODUCT;
```

- ROUND(x, d): Round the value x to d decimals.



```
SELECT *, ROUND(SQRT(P_PRICE), 2) FROM PRODUCT;
```

String data type

- Several common string datatypes in MySQL.
- **CHAR(M)** - A fixed-length string between 1 and 255 characters in length. CHAR(30) can hold up to 30 characters.
- **VARCHAR(M)** - A variable-length string between 1 and 255 characters in length; for example VARCHAR(25).
- **TEXT** - Maximum length of 65535 characters. TEXT is used to hold large amounts of data; TEXT fields. You do not specify a length with TEXT.

String data type

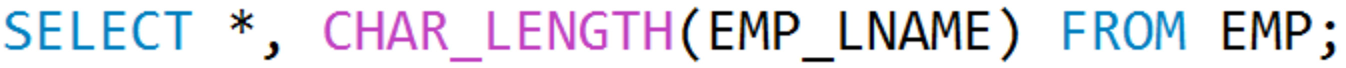
- Which data type can be used for the following columns?
- EMP_LNAME, EMP_FNAME
- EMP_GENDER
- EMP_ADDRESS
- Twitter: a tweet (140 characters limit?)

iClicker question

- What is a reasonable data type choice for the *content* column of POST table in Facebook's user database?
- A. CHAR(255)
- B. VARCHAR(255)
- C. TEXT

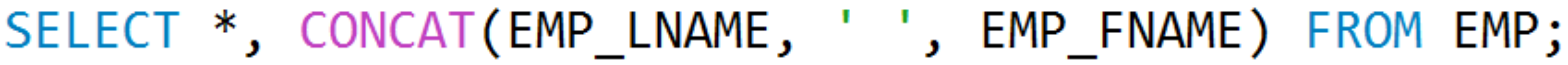
MySQL string functions

- MySQL provides several functions that operate on string data types.
- CHAR_LENGTH(): Return the number of characters in a string.



```
SELECT *, CHAR_LENGTH(EMP_LNAME) FROM EMP;
```

- CONCAT(): Returns concatenated string



```
SELECT *, CONCAT(EMP_LNAME, ' ', EMP_FNAME) FROM EMP;
```

Learning objective

- Understand comments, alias, derived table.
- Understand String and numeric data types.