# Aggregate Queries

CS 327E
Feb 19, 2018

## Announcements

- Cloud SQL instance: stop and start to consume fewer billing credits
- Reminder: Midterm in 2 weeks

## 1) Which of the following is an aggregate function?

a) `DISTINCT`
b) `MIN`
c) `REPLACE`
d) All of the above

## 2) How many records are returned by this query?

```
SELECT DISTINCT country
FROM Olympics
WHERE medal IS NOT NULL;
```

Olympics

| id | sport | event | athlete | country | medal |
|---|---|---|---|---|---|
| 33 | Snowboard | Men's Halfpipe | Shaun White | United States | G |
| 36 | Snowboard | Women's Halfpipe | Chloe Kim | United States | G |
| 81 | Alpine Skiing | Men's Downhill | Aksel Lund Svindal | Norway | G |
| 106 | Ski Jumping | Women's Normal Hill Individual | Katharina Althaus | Germany | S |
| 248 | Speedskating | Men's 1,500m | Kim Min Seok | South Korea | B |
| 600 | Ice Hockey | Women's Tournament | | | |

a) 0
b) 1
c) 4
d) 5

## 3) What answer does this query produce?

```
SELECT COUNT(*)
FROM Olympics
WHERE sport IN ('Alpine Skiing', 'Snowboard');
```

Olympics

| id | sport | event | athlete | country | medal |
|---|---|---|---|---|---|
| 33 | Snowboard | Men's Halfpipe | Shaun White | United States | G |
| 36 | Snowboard | Women's Halfpipe | Chloe Kim | United States | G |
| 81 | Alpine Skiing | Men's Downhill | Aksel Lund Svindal | Norway | G |
| 106 | Ski Jumping | Women’s Normal Hill Individual | Katharina Althaus | Germany | S |
| 248 | Speedskating | Men's 1,500m | Kim Min Seok | South Korea | B |
| 600 | Ice Hockey | Women's Tournament | | | |

a) 3
b) 4
c) 5
d) 6

## 4) How many records are returned by this query?

```
SELECT country, COUNT(*)
FROM Olympics
GROUP BY country;
```

Olympics

| id | sport | event | athlete | country | medal |
|---|---|---|---|---|---|
| 33 | Snowboard | Men's Halfpipe | Shaun White | United States | G |
| 36 | Snowboard | Women's Halfpipe | Chloe Kim | United States | G |
| 81 | Alpine Skiing | Men's Downhill | Aksel Lund Svindal | Norway | G |
| 106 | Ski Jumping | Women's Normal Hill Individual | Katharina Althaus | Germany | S |
| 248 | Speedskating | Men's 1,500m | Kim Min Seok | South Korea | B |
| 600 | Ice Hockey | Women's Tournament | | | |

a) 3
b) 4
c) 5
d) 6

## 5) How many records are returned by this query?

```
SELECT country, medal, COUNT(*)
FROM Olympics
GROUP BY country, medal;
```

Olympics

| id | sport | event | athlete | country | medal |
|---|---|---|---|---|---|
| 33 | Snowboard | Men's Halfpipe | Shaun White | United States | G |
| 36 | Snowboard | Women's Halfpipe | Chloe Kim | United States | G |
| 81 | Alpine Skiing | Men's Downhill | Aksel Lund Svindal | Norway | G |
| 106 | Ski Jumping | Women’s Normal Hill Individual | Katharina Althaus | Germany | S |
| 248 | Speedskating | Men's 1,500m | Kim Min Seok | South Korea | B |
| 600 | Ice Hockey | Women's Tournament | | | |

a) 3
b) 4
c) 5
d) 6

# Standard Aggregate Functions

- `SELECT` **`MIN`**`(col)`
- `SELECT` **`MAX`**`(col)`
- `SELECT` **`SUM`**`(col)`
- `SELECT` **`AVG`**`(col)`
- `SELECT` **`COUNT`**`(col)`

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

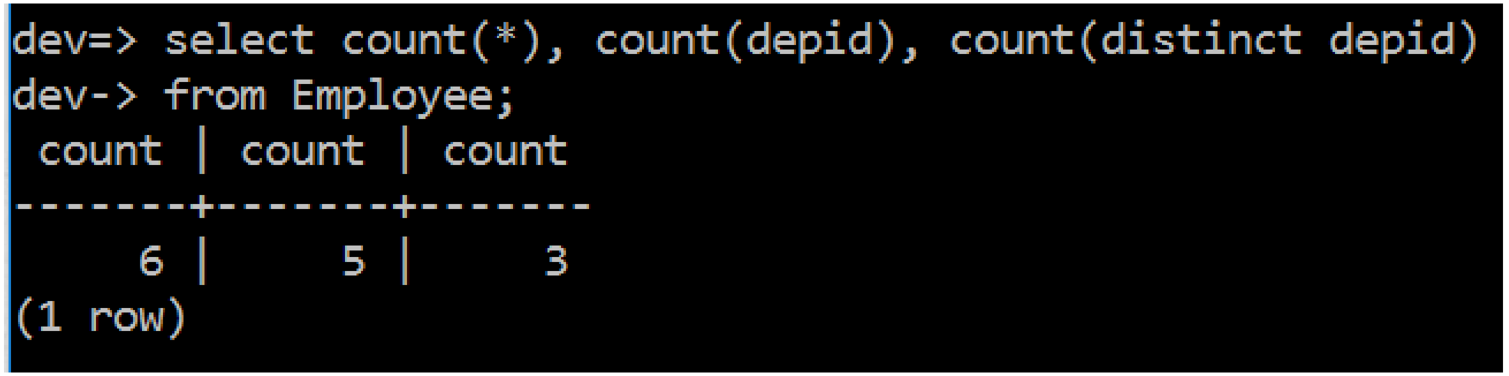
```
SELECT MIN(salary), MAX(salary), SUM(salary), AVG(salary), COUNT(salary)
FROM Employee;
```

## Semantics of COUNT

- `SELECT` **`COUNT`**`(*)`
- `SELECT` **`COUNT`**`(col)`
- `SELECT` **`COUNT`**`(DISTINCT col)`

**Employee**

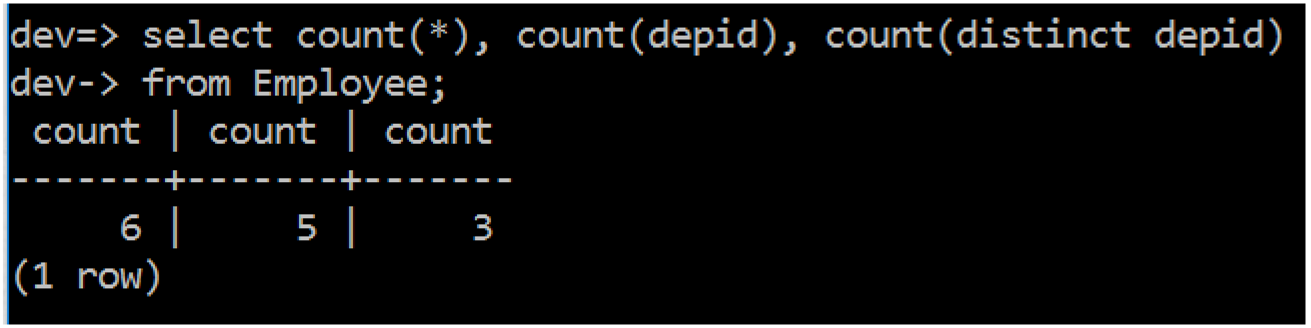
| <u>empid</u> | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 |  |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |



```
dev=> select count(*), count(depid), count(distinct depid)
dev-> from Employee;
 count | count | count
-------+-------+-------
     6 |     5 |     3
(1 row)
```

# Semantics of COUNT

- `COUNT(*)`
- `COUNT(column)`
- `COUNT(DISTINCT column)`



```
dev=> select count(*), count(depid), count(distinct depid)
dev-> from Employee;
 count | count | count
-------+-------+-------
     6 |     5 |     3
(1 row)
```

```
SELECT COUNT(DISTINCT depid), depid
FROM Employee;
```

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

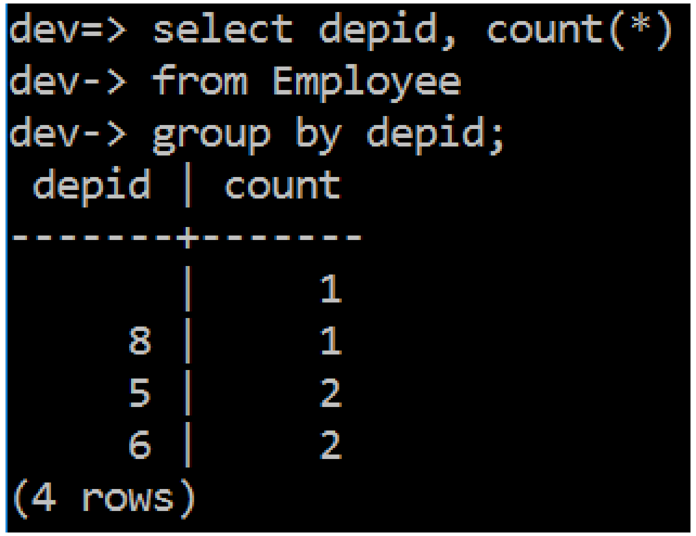
## Aggregates with Groupings

- `SELECT col1, AGGR(col2)` **`GROUP BY`** `col1`

- `SELECT col1, AGGR(col2)` **`GROUP BY`** `col1` `[`**`HAVING AGGR`**`(col2)…]`

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |



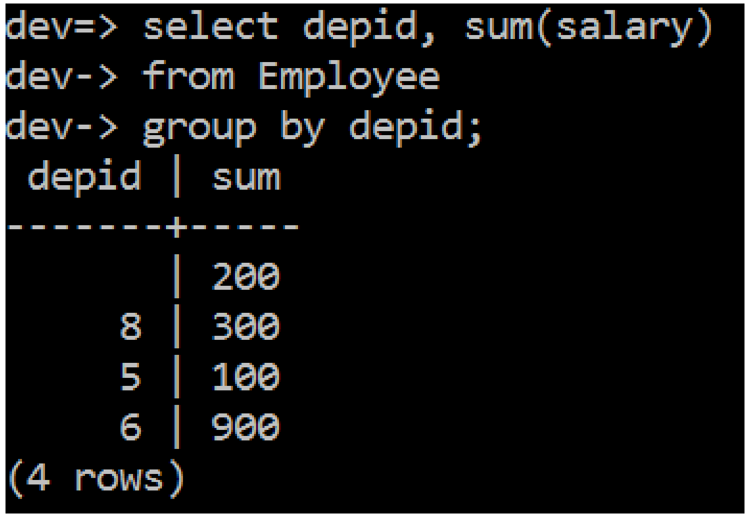
```
dev=> select depid, count(*)
dev-> from Employee
dev-> group by depid;
 depid | count
-------+-------
       |     1
     8 |     1
     5 |     2
     6 |     2
(4 rows)
```

## Aggregates with Groupings

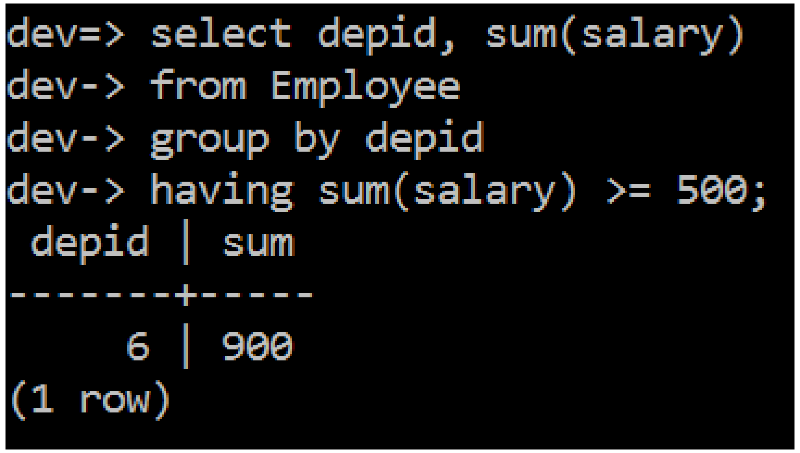
- `SELECT col1, AGGR(col2) GROUP BY col1`
- `SELECT col1, AGGR(col2) GROUP BY col1 [HAVING AGGR(col2)…]`

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |



```
dev=> select depid, sum(salary)
dev-> from Employee
dev-> group by depid;
 depid | sum
-------+-----
       | 200
     8 | 300
     5 | 100
     6 | 900
(4 rows)
```



```
dev=> select depid, sum(salary)
dev-> from Employee
dev-> group by depid
dev-> having sum(salary) >= 500;
 depid | sum
-------+-----
     6 | 900
(1 row)
```

# Aggregation Pitfalls

- SELECT **AGGR**(col)
- HAVING **AGGR**(col)
- ORDER BY **AGGR**(col)

- **Not** FROM **AGGR**(col)
- **Not** JOIN **AGGR**(col)
- **Not** WHERE **AGGR**(col)
- **Not** GROUP BY **AGGR**(col)
- **Not** LIMIT **AGGR**(col)

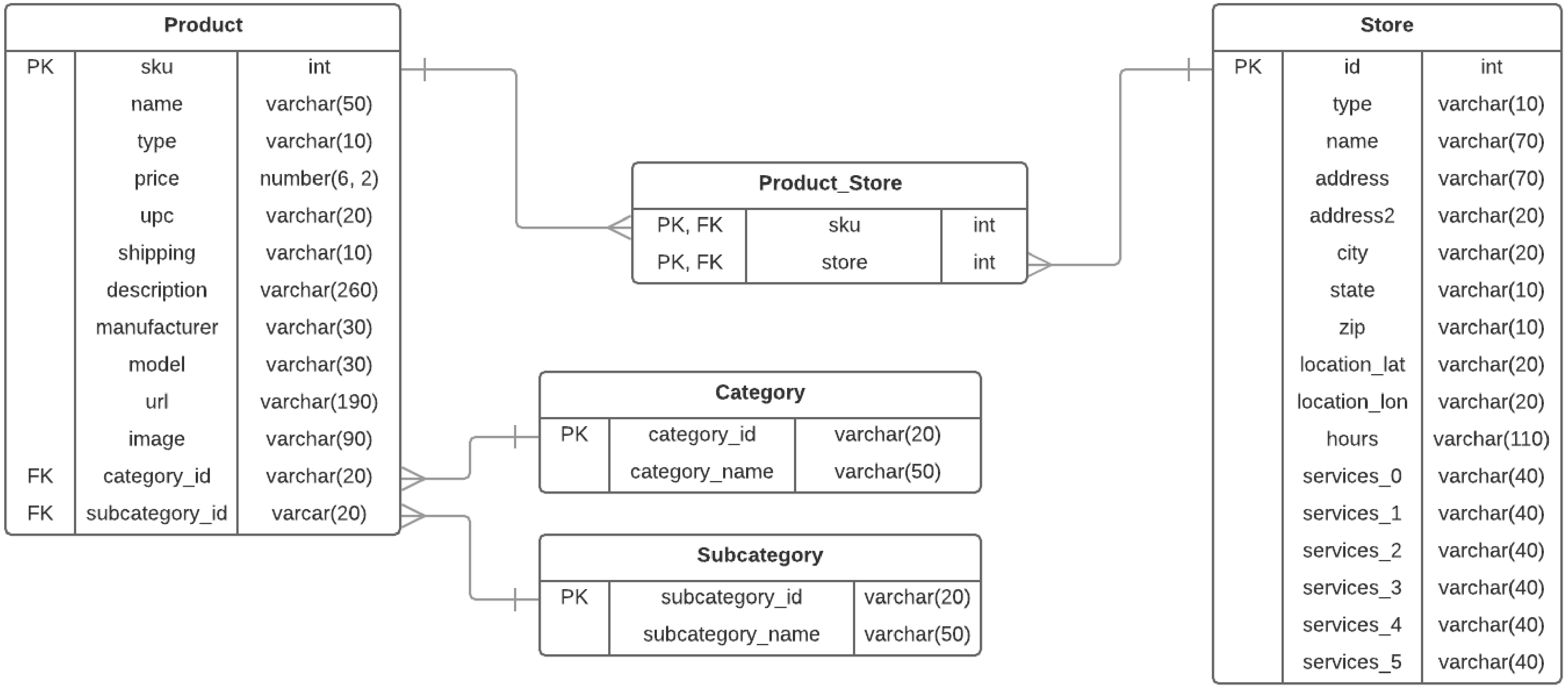
**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

```
SELECT COUNT(DISTINCT depid), depid
FROM Employee;
```

ERROR: column "employee.depid" must appear in the GROUP BY clause or be used in an aggregate function

**Practice Problem 1**: List the manufacturers and the total number of products they have in the database. Only include manufacturers which have at least 5 products. Only include products that cost at least $1.



| Product | | |
|---|---|---|
| PK | sku | int |
| | name | varchar(50) |
| | type | varchar(10) |
| | price | number(6, 2) |
| | upc | varchar(20) |
| | shipping | varchar(10) |
| | description | varchar(260) |
| | manufacturer | varchar(30) |
| | model | varchar(30) |
| | url | varchar(190) |
| | image | varchar(90) |
| FK | category_id | varchar(20) |
| FK | subcategory_id | varcar(20) |

| Product_Store | | |
|---|---|---|
| PK, FK | sku | int |
| PK, FK | store | int |

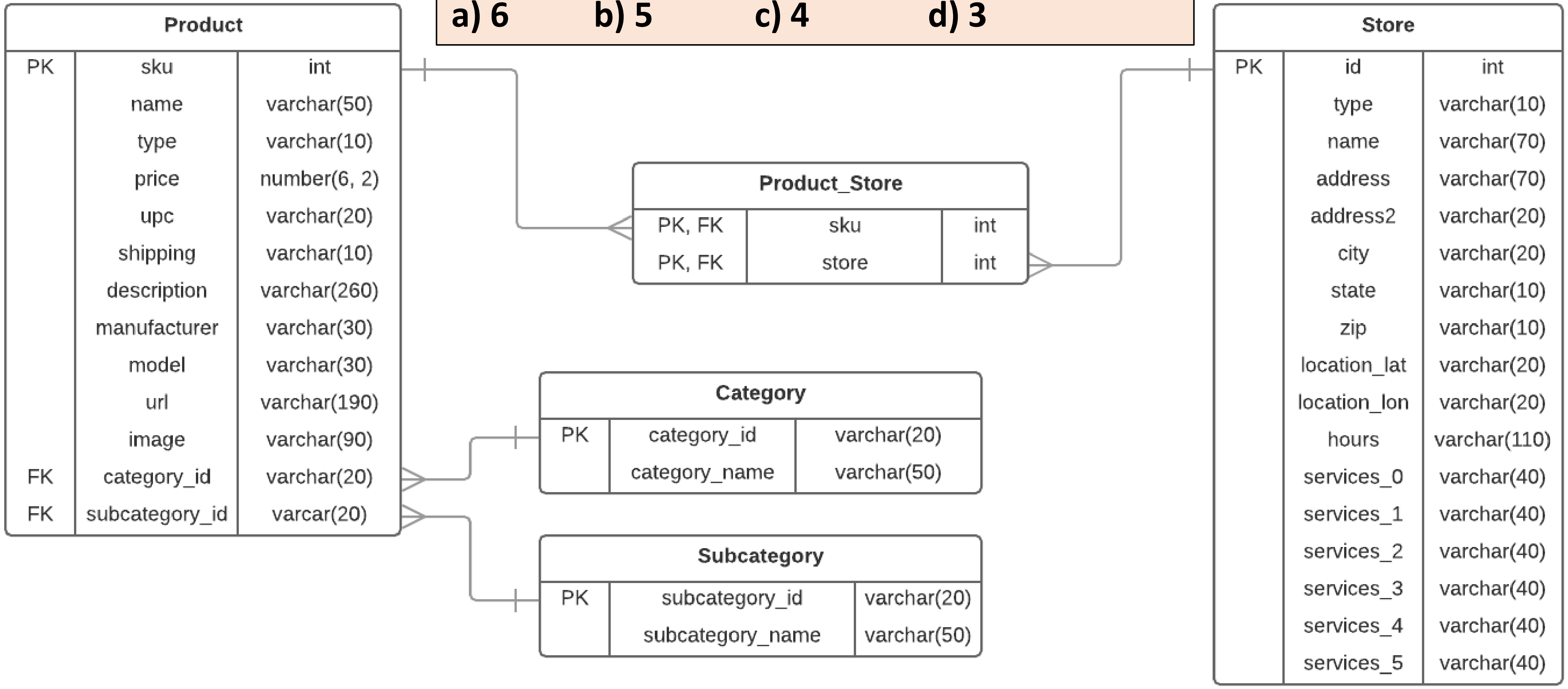
| Category | | |
|---|---|---|
| PK | category_id | varchar(20) |
| | category_name | varchar(50) |

| Subcategory | | |
|---|---|---|
| PK | subcategory_id | varchar(20) |
| | subcategory_name | varchar(50) |

| Store | | |
|---|---|---|
| PK | id | int |
| | type | varchar(10) |
| | name | varchar(70) |
| | address | varchar(70) |
| | address2 | varchar(20) |
| | city | varchar(20) |
| | state | varchar(10) |
| | zip | varchar(10) |
| | location_lat | varchar(20) |
| | location_lon | varchar(20) |
| | hours | varchar(110) |
| | services_0 | varchar(40) |
| | services_1 | varchar(40) |
| | services_2 | varchar(40) |
| | services_3 | varchar(40) |
| | services_4 | varchar(40) |
| | services_5 | varchar(40) |

**Practice Problem 1**: List the manufacturers and total number of products they have in the database. Only include manufacturers which have at least 5 products. Only include products that cost at least $1.

**How many clauses are required by this query?**
**a) 6 b) 5 c) 4 d) 3**



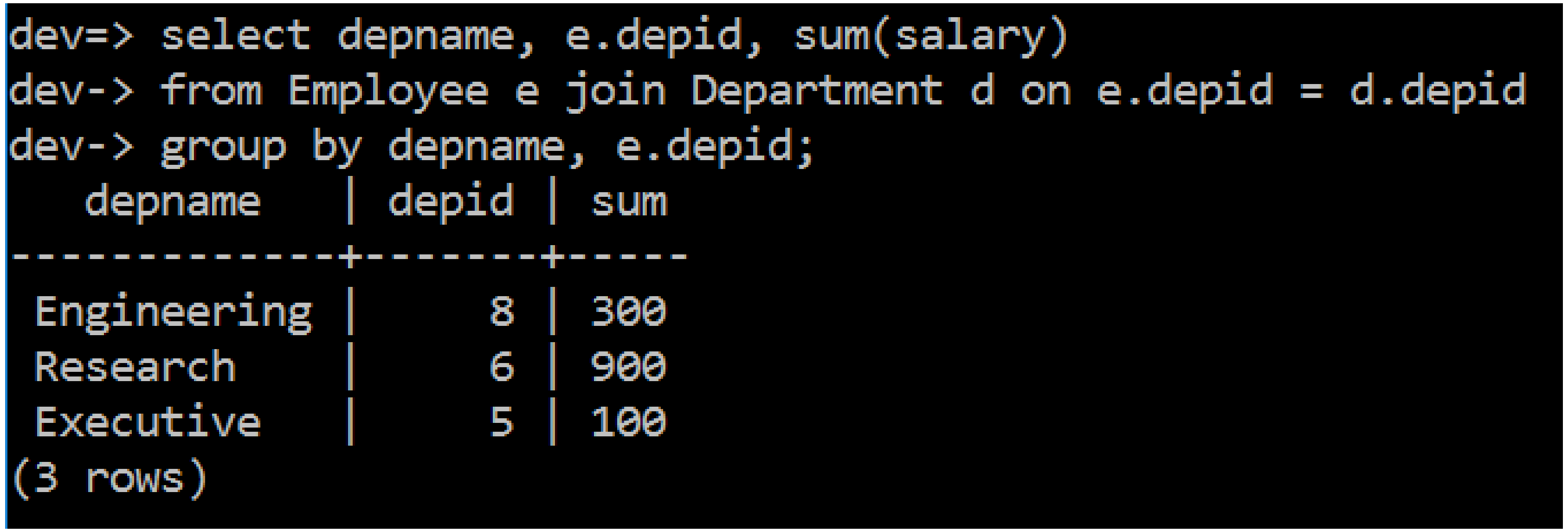
# Aggregates with Groupings

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

**Department**

| depid | depname |
|---|---|
| 5 | Executive |
| 6 | Research |
| 7 | Sales |
| 8 | Engineering |



```
dev=> select depname, e.depid, sum(salary)
dev-> from Employee e join Department d on e.depid = d.depid
dev-> group by depname, e.depid;
   depname   | depid | sum
-------------+-------+-----
 Engineering |     8 | 300
 Research    |     6 | 900
 Executive   |     5 | 100
(3 rows)
```

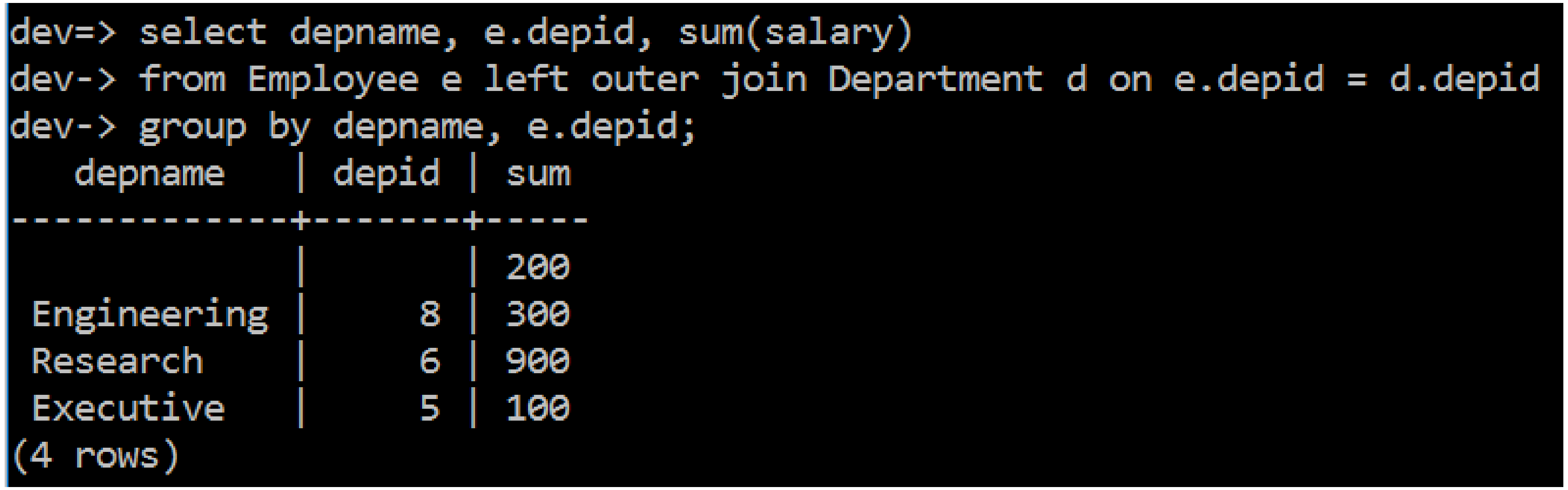
## Aggregates with Groupings

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

**Department**

| depid | depname |
|---|---|
| 5 | Executive |
| 6 | Research |
| 7 | Sales |
| 8 | Engineering |



```
dev=> select depname, e.depid, sum(salary)
dev-> from Employee e left outer join Department d on e.depid = d.depid
dev-> group by depname, e.depid;
   depname    | depid | sum
--------------+-------+-----
              |       | 200
 Engineering  |     8 | 300
 Research     |     6 | 900
 Executive    |     5 | 100
(4 rows)
```

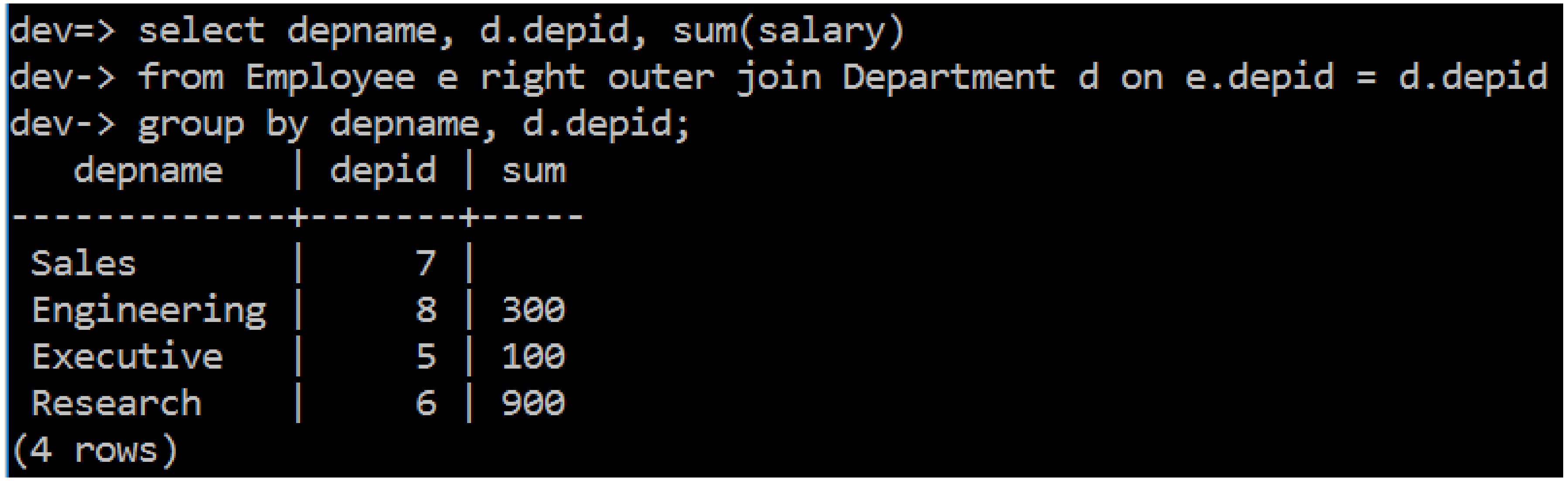
# Aggregates with Groupings

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

**Department**

| depid | depname |
|---|---|
| 5 | Executive |
| 6 | Research |
| 7 | Sales |
| 8 | Engineering |



```
dev=> select depname, d.depid, sum(salary)
dev-> from Employee e right outer join Department d on e.depid = d.depid
dev-> group by depname, d.depid;
   depname    | depid | sum
-------------+-------+-----
 Sales        |     7 |
 Engineering |     8 | 300
 Executive    |     5 | 100
 Research     |     6 | 900
(4 rows)
```

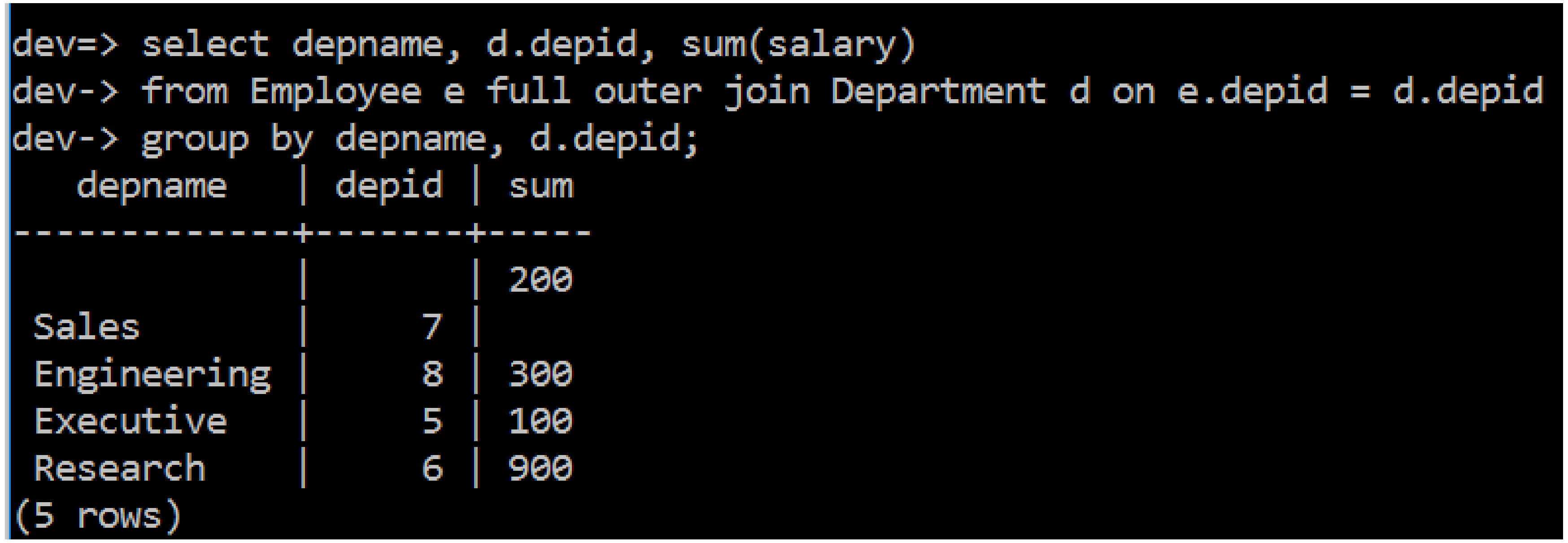
# Aggregates with Groupings

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

**Department**

| depid | depname |
|---|---|
| 5 | Executive |
| 6 | Research |
| 7 | Sales |
| 8 | Engineering |



```
dev=> select depname, d.depid, sum(salary)
dev-> from Employee e full outer join Department d on e.depid = d.depid
dev-> group by depname, d.depid;
   depname    | depid | sum
-------------+-------+-----
             |       | 200
 Sales       |     7 |
 Engineering |     8 | 300
 Executive   |     5 | 100
 Research    |     6 | 900
(5 rows)
```

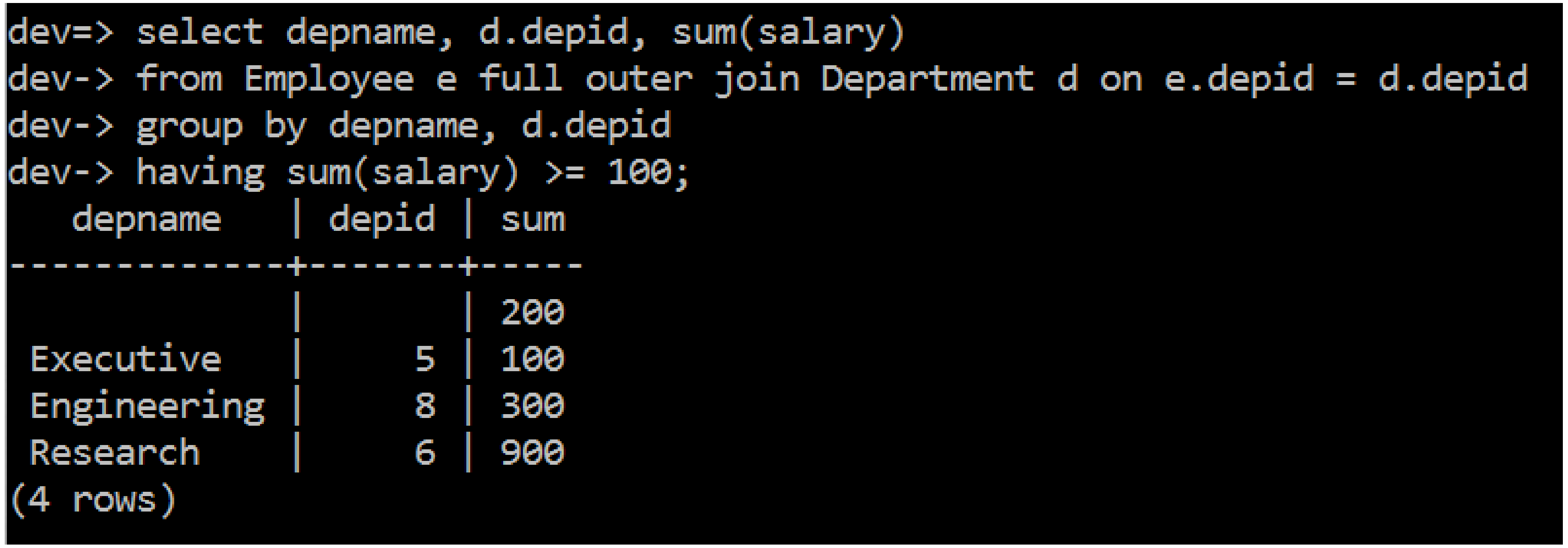
# Aggregates with Groupings

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

**Department**

| depid | depname |
|---|---|
| 5 | Executive |
| 6 | Research |
| 7 | Sales |
| 8 | Engineering |



```
dev=> select depname, d.depid, sum(salary)
dev-> from Employee e full outer join Department d on e.depid = d.depid
dev-> group by depname, d.depid
dev-> having sum(salary) >= 100;
   depname   | depid | sum
-------------+-------+-----
             |       | 200
 Executive   |     5 | 100
 Engineering |     8 | 300
 Research    |     6 | 900
(4 rows)
```

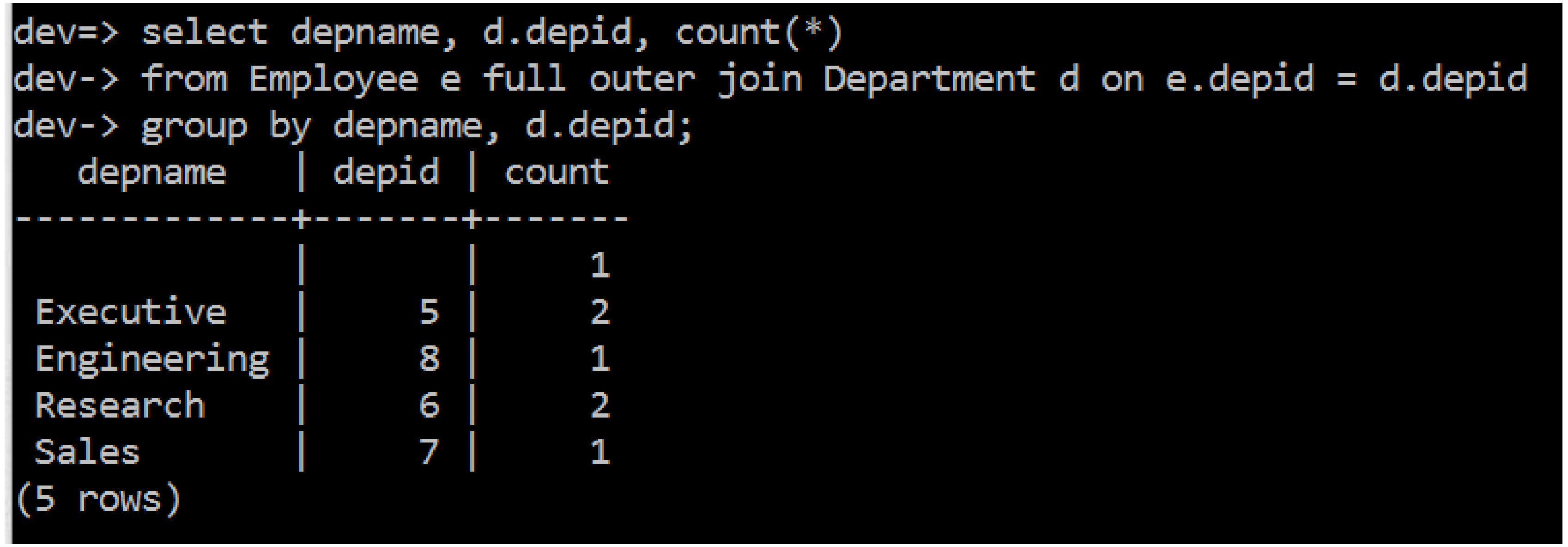
# Aggregates with Groupings

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

**Department**

| depid | depname |
|---|---|
| 5 | Executive |
| 6 | Research |
| 7 | Sales |
| 8 | Engineering |



```
dev=> select depname, d.depid, count(*)
dev-> from Employee e full outer join Department d on e.depid = d.depid
dev-> group by depname, d.depid;
   depname    | depid | count
-------------+-------+-------
             |       |     1
 Executive   |     5 |     2
 Engineering |     8 |     1
 Research    |     6 |     2
 Sales       |     7 |     1
(5 rows)
```

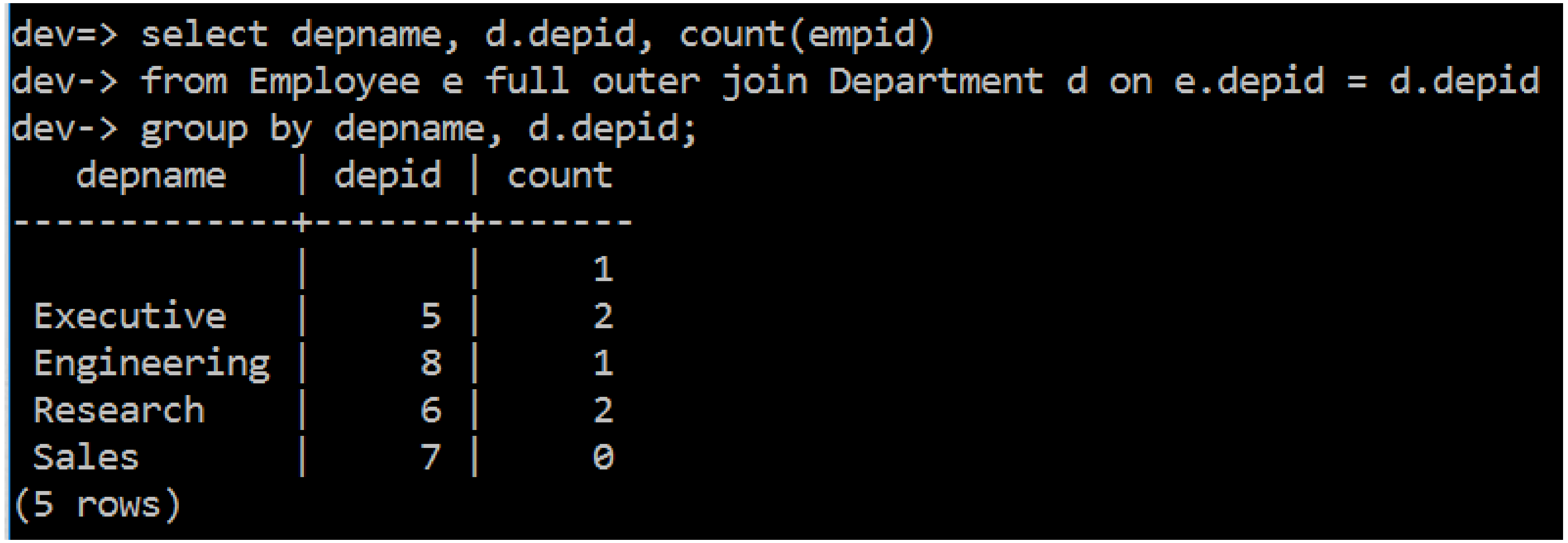
# Aggregates with Groupings

**Employee**

| empid | firstname | lastname | salary | depid |
|---|---|---|---|---|
| 1 | Michael | Dell | 100 | 5 |
| 2 | Betty | Jennings | 200 | |
| 3 | Bill | Gates | 0 | 5 |
| 4 | Kay | McNulty | 300 | 8 |
| 5 | Jim | Gray | 500 | 6 |
| 6 | Gordon | Moore | 400 | 6 |

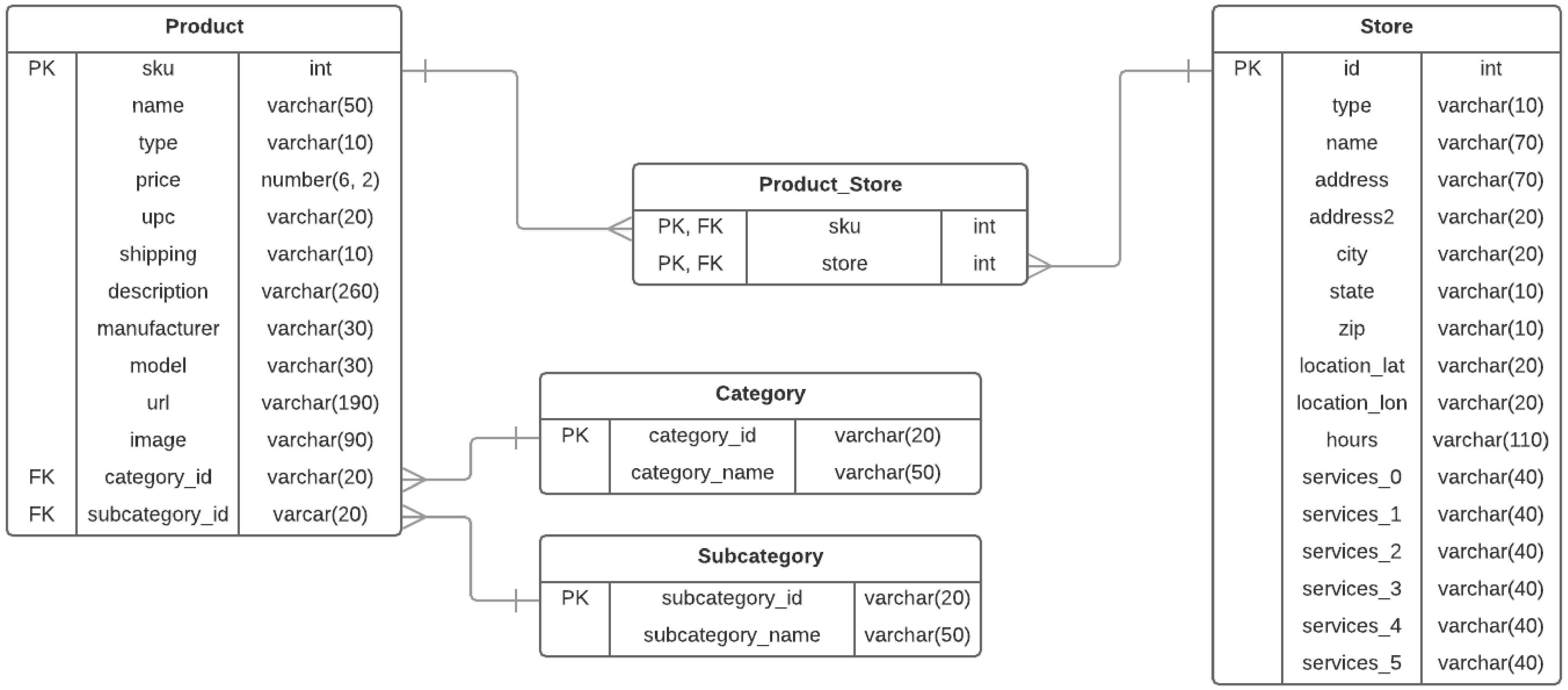
**Department**

| depid | depname |
|---|---|
| 5 | Executive |
| 6 | Research |
| 7 | Sales |
| 8 | Engineering |



```
dev=> select depname, d.depid, count(empid)
dev-> from Employee e full outer join Department d on e.depid = d.depid
dev-> group by depname, d.depid;
   depname    | depid | count
--------------+-------+-------
              |       |     1
 Executive    |     5 |     2
 Engineering  |     8 |     1
 Research     |     6 |     2
 Sales        |     7 |     0
(5 rows)
```

**Practice Problem 2**: List the product subcategory names and average price for each one. Include all products even those that don't have an associated subcategory. Order the results by average price from highest to lowest.



| Product | | |
|---|---|---|
| PK | sku | int |
| | name | varchar(50) |
| | type | varchar(10) |
| | price | number(6, 2) |
| | upc | varchar(20) |
| | shipping | varchar(10) |
| | description | varchar(260) |
| | manufacturer | varchar(30) |
| | model | varchar(30) |
| | url | varchar(190) |
| | image | varchar(90) |
| FK | category_id | varchar(20) |
| FK | subcategory_id | varcar(20) |

| Product_Store | | |
|---|---|---|
| PK, FK | sku | int |
| PK, FK | store | int |

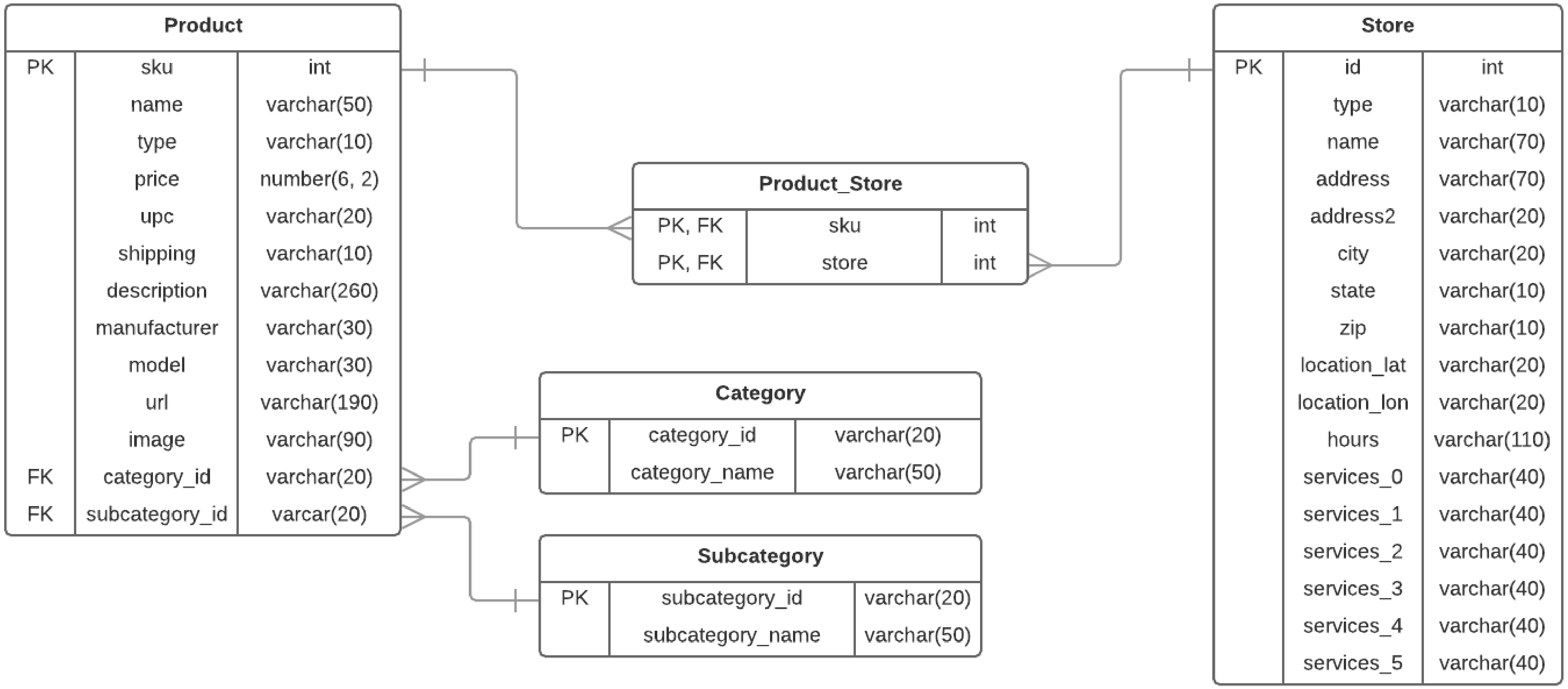
| Category | | |
|---|---|---|
| PK | category_id | varchar(20) |
| | category_name | varchar(50) |

| Subcategory | | |
|---|---|---|
| PK | subcategory_id | varchar(20) |
| | subcategory_name | varchar(50) |

| Store | | |
|---|---|---|
| PK | id | int |
| | type | varchar(10) |
| | name | varchar(70) |
| | address | varchar(70) |
| | address2 | varchar(20) |
| | city | varchar(20) |
| | state | varchar(10) |
| | zip | varchar(10) |
| | location_lat | varchar(20) |
| | location_lon | varchar(20) |
| | hours | varchar(110) |
| | services_0 | varchar(40) |
| | services_1 | varchar(40) |
| | services_2 | varchar(40) |
| | services_3 | varchar(40) |
| | services_4 | varchar(40) |
| | services_5 | varchar(40) |

**Practice Problem 2**: List the product subcategory names and average price for each one. Include all products even those that don't have an associated subcategory. Order the results by average price from highest-to-lowest.

**Which column must be present in both the select and group-by clauses?**
**a) price** **b) subcategory_name** **c) subcategory_id**



Product
PK sku int
name varchar(50)
type varchar(10)
price number(6, 2)
upc varchar(20)
shipping varchar(10)
description varchar(260)
manufacturer varchar(30)
model varchar(30)
url varchar(190)
image varchar(90)
FK category_id varchar(20)
FK subcategory_id varcar(20)
Product_Store
PK, FK sku int
PK, FK store int
Category
PK category_id varchar(20)
category_name varchar(50)
Subcategory
PK subcategory_id varchar(20)
subcategory_name varchar(50)
Store
PK id int
type varchar(10)
name varchar(70)
address varchar(70)
address2 varchar(20)
city varchar(20)
state varchar(10)
zip varchar(10)
location_lat varchar(20)
location_lon varchar(20)
hours varchar(110)
services_0 varchar(40)
services_1 varchar(40)
services_2 varchar(40)
services_3 varchar(40)
services_4 varchar(40)
services_5 varchar(40)