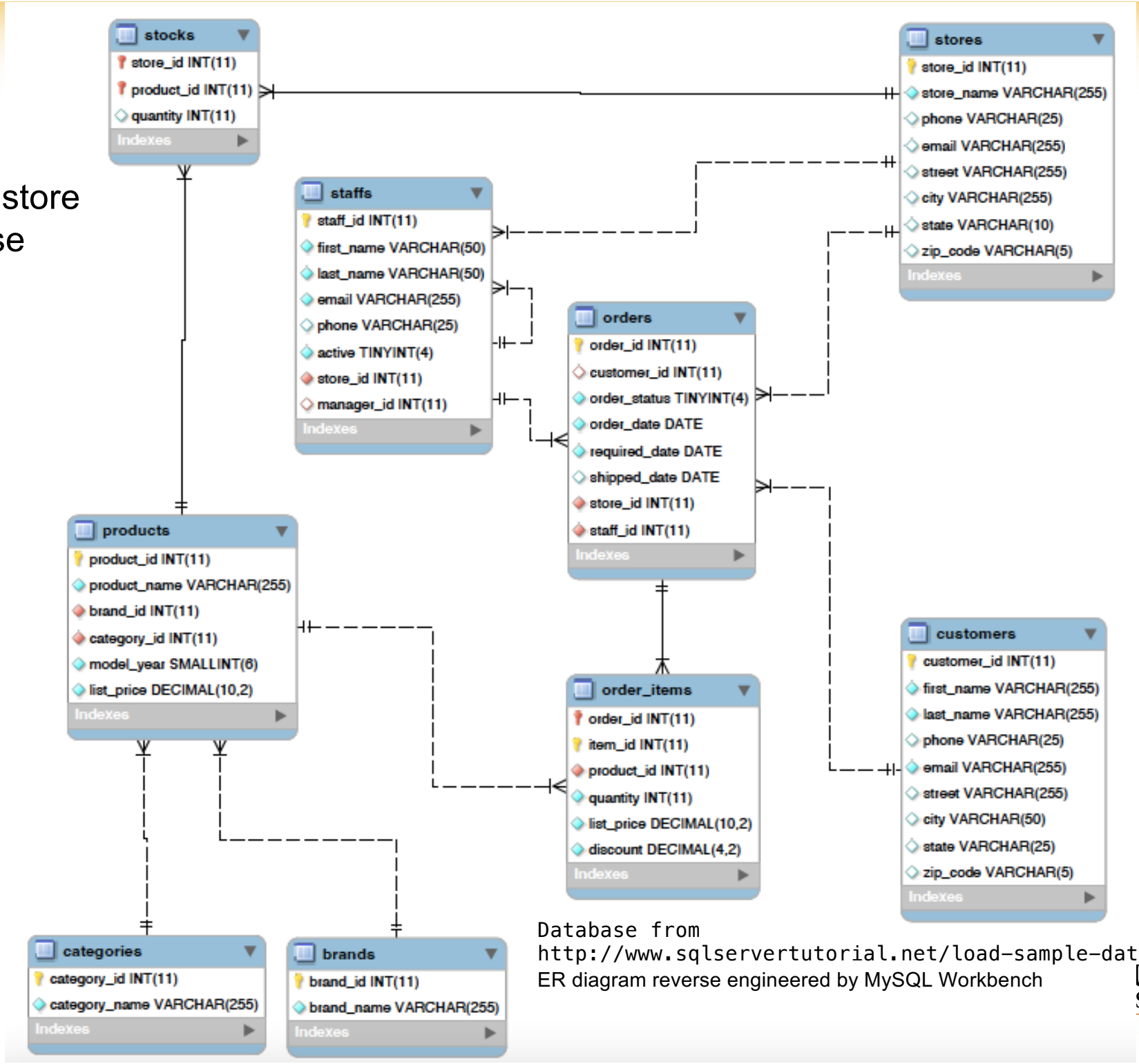
Bicycle store database



Database from http://www.sqlservertutorial.net/load-sample-database/

ER diagram reverse engineered by MySQL Workbench

Questions to try…

- How many of all orders were required on the same day they were ordered?
- Which products were sold in a year that predates their model year?
- How many direct reports does each manager have? Just identify the manager by their staff ID.

DALHOUSIE UNIVERSITY
Inspiring Minds

Typical join “shortcut”

- Often join tables on a common set of columns and then ask to only show that common column once.
  - Currently shown using the “from” clause, the “where” clause, and the choice of columns
- Modify the “from” clause to a “from ... join .. using (..)” clause:
  - “join” replacing the comma in the from clause
  - “using” lists the columns to have in common
  - The variant only keeps one copy of all the columns mentioned in the “using” part

DALHOUSIE UNIVERSITY
Inspiring Minds

Natural Join

- When your join is going to use all common columns, you can use the ”natural join” operator
  - Equivalent to join..using (<all common column names>)

DALHOUSIE UNIVERSITY
Inspiring Minds

Example

- Select course.course_id, name, person_id, fees
  from course, registration
  where course.course_id = registration.course_id;

  becomes

  Select *
  from course join registration using (course_id);

  or

  Select * from course natural join registration;

DALHOUSIE UNIVERSITY
Inspiring Minds

Different kinds of join operators

- **Consider 4 join operators:**
  - **Inner join**
    - Returns records that have matching values in both tables
    - **Equi-join – inner join where the join condition is based on equality between values in the common columns**
    - **Natural join – inner join that restricts all same-named columns to match and produces one instance of the common columns**
  - **Left join**
    - Return all records from the left table, and the matched records from the right table, adding NULL values when a match isn't present
  - **Right join**
    - Return all records from the right table, and the matched records from the left table, adding NULL values when a match isn't present
  - **Outer join**
    - Return all records when there is a match in either left or right table

Join examples

sample1

id	value	name
1	100	one
1	200	two
2	300	three

sample2

id	info	value
2	alpha	500
2	beta	300
4	gamma	400

```
mysql> select * from sample1 join sample2 where sample1.id = sample2.id;
+------+-------+-------+------+-------+-------+
| id   | value | name  | id   | info  | value |
+------+-------+-------+------+-------+-------+
|    2 |   300 | three |    2 | alpha |   500 |
|    2 |   300 | three |    2 | beta  |   300 |
+------+-------+-------+------+-------+-------+
```

Equi-join

```
mysql> select * from sample1 natural join sample2;
+------+-------+-------+------+
| id   | value | name  | info |
+------+-------+-------+------+
|    2 |   300 | three | beta |
+------+-------+-------+------+
```

Natural join

DALHOUSIE UNIVERSITY
Inspiring Minds

Join examples

sample1

```
+------+-------+-------+
| id   | value | name  |
+------+-------+-------+
|    1 |   100 | one   |
|    1 |   200 | two   |
|    2 |   300 | three |
+------+-------+-------+
```

sample2

```
+------+-------+-------+
| id   | info  | value |
+------+-------+-------+
|    2 | alpha |   500 |
|    2 | beta  |   300 |
|    4 | gamma |   400 |
+------+-------+-------+
```

```
mysql> select * from sample1 left join sample2 on sample1.id = sample2.id;
+------+-------+-------+------+-------+-------+
| id   | value | name  | id   | info  | value |
+------+-------+-------+------+-------+-------+
|    2 |   300 | three |    2 | alpha |   500 |
|    2 |   300 | three |    2 | beta  |   300 |
|    1 |   100 | one   | NULL | NULL  |  NULL |
|    1 |   200 | two   | NULL | NULL  |  NULL |
+------+-------+-------+------+-------+-------+
```

Left join

```
mysql> select * from sample1 right join sample2 on sample1.id = sample2.id;
+------+-------+-------+------+-------+-------+
| id   | value | name  | id   | info  | value |
+------+-------+-------+------+-------+-------+
|    2 |   300 | three |    2 | alpha |   500 |
|    2 |   300 | three |    2 | beta  |   300 |
| NULL |  NULL | NULL  |    4 | gamma |   400 |
+------+-------+-------+------+-------+-------+
```

Right join

LHOUSIE IVERSITY
Inspiring Minds

Join conditions

- **Outer joins often invoke “is null” or “is not null” in the where condition to filter the results**

sample1

```
+------+-------+-------+
| id   | value | name  |
+------+-------+-------+
|    1 |   100 | one   |
|    1 |   200 | two   |
|    2 |   300 | three |
+------+-------+-------+
```

sample2

```
+------+-------+-------+
| id   | info  | value |
+------+-------+-------+
|    2 | alpha |   500 |
|    2 | beta  |   300 |
|    4 | gamma |   400 |
+------+-------+-------+
```

- **What does the following produce?**
  Select sample2.id from sample1 right join sample2 on sample1.id = sample2.id where sample1.id is null;
  - **All ids in sample2 not in sample1**

```
mysql> select sample2.id from sample1 right join sample2
    -> on sample1.id = sample2.id where sample1.id is null;
+------+
| id   |
+------+
|    4 |
+------+
```


DALHOUSIE UNIVERSITY
Inspiring Minds

Query Execution

- The DBMS creates an execution plan from your SQL select statement
  - Identifies the order in which to do evaluations
    - From: combine small tables first
    - Where: apply the most restrictive conditions first
  - Query optimizer can re-order elements of the query to increase its performance
    - Estimate the size of the tables to be combined to help manage the total work
- How you specify your query can influence performance.

DALHOUSIE UNIVERSITY
Inspiring Minds

Costs for select statements

- Column selection
  - Identifies what to keep as you process. Low cost, unless you're using transformations
- From clause
  - Generates combinations of records. High cost if you're generating many records that you will just throw away
- Where clause
  - Does winnowing as you process. Can have complex sets of conditions to evaluate. Medium cost
- Group by clause
  - Need the final data to make this work. Throwing away most of the generated records to create summaries.
- Order by clause
  - Need the final data to execute. Cost relative to output size, not generated records size.

Helping performance

- 1. Minimize the size and number of table combinations in the ”from” clause
  - How??? We need a new tool -> subqueries
- 2. include restrictive “where” elements if you can

DALHOUSIE UNIVERSITY
Inspiring Minds

Subqueries

- The output of an SQL statement is a table.
- Use that output table in the place of any other table in a query.
  - Enclose the subquery in parentheses
  - Need to use the “as” keyword to give the output of the subquery a name
- Use a subquery to reduce the size or number of tables to combine in the “from” clause
- A subquery can have its own subquery

DALHOUSIE UNIVERSITY
Inspiring Minds

Subquery example

- Select *
  from person join registration using (person_id)
  where person_id = 3;

versus

```
select *
   from
      (select * from person where person_id = 3) as interested
      join registration using (person_id);
```

Is there a difference?

DALHOUSIE UNIVERSITY
Inspiring Minds

Subquery example

```
select *
   from person as p, course as c, registration as r
   where p.person_id = r.person_id
      and c.course_id = r.course_id
      and p.name like "A%";
```

How many rows does each query create?

versus

```
select * from
   (
      (select * from person where name like "A%") as p
      join registration using (person_id)
   )  join course as c using (course_id);
```

DALHOUSIE UNIVERSITY
Inspiring Minds

Subqueries

- Can also appear in the “where” clause
  - Extract one number for a comparison

```
select name, salary from person
   where salary >= (select avg(salary) from person);
```

  - Extract a set of values for an “in” statement (waiting for mysql to catch up to this functionality)

```
select name, salary from person
   where salary in
      (select distinct salary from person order by salary desc limit
2);
```

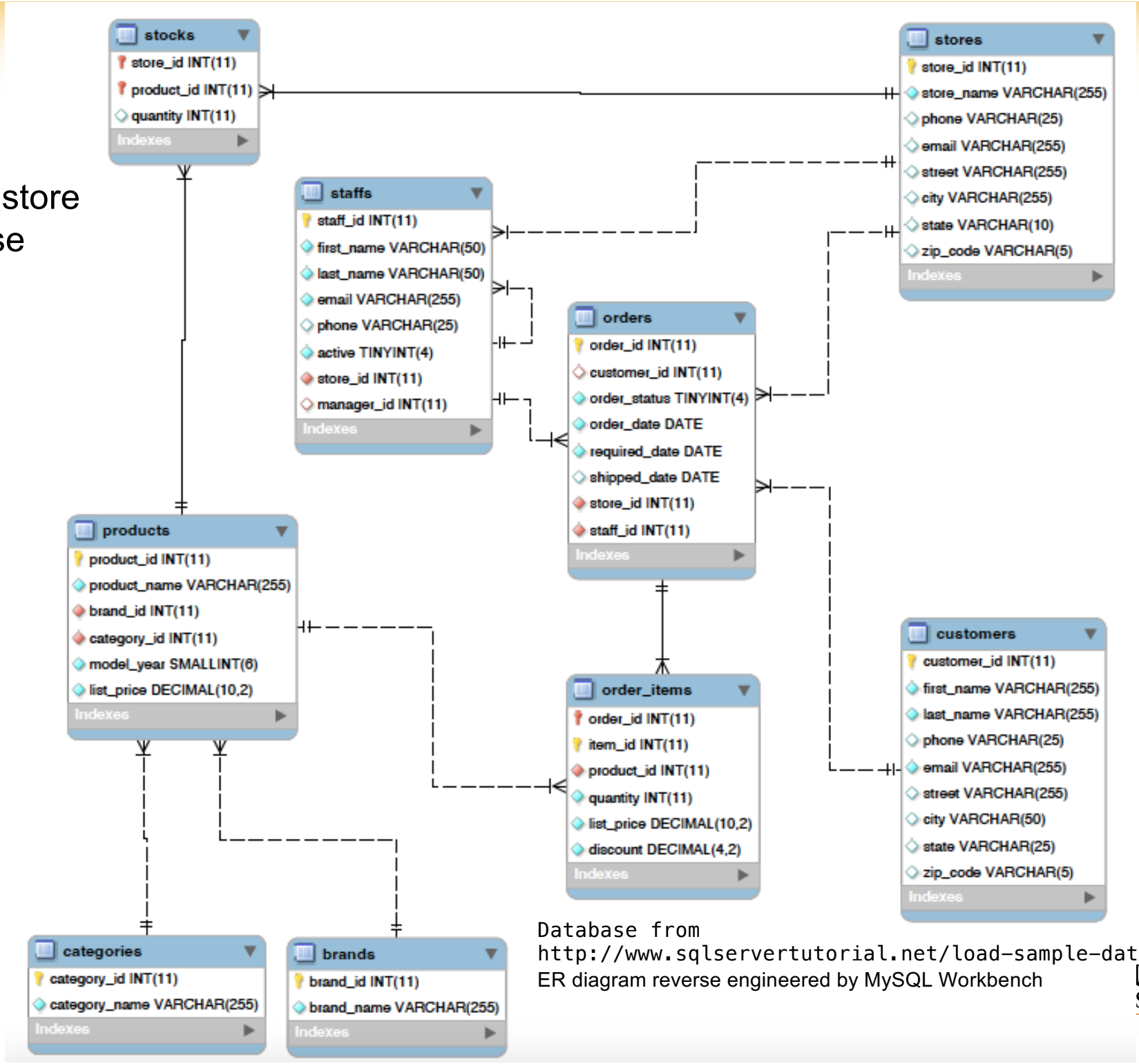

DALHOUSIE UNIVERSITY
Inspiring Minds

Find everyone within 1 standard deviation of the average age – shorter version from class

- Select name, age from person where age between
  (select avg(age) – std(age) from person)
  and
  (select avg(age) + std(age) from person);

DALHOUSIE UNIVERSITY
Inspiring Minds

Bicycle store database



Database from http://www.sqlservertutorial.net/load-sample-database/

ER diagram reverse engineered by MySQL Workbench

Question to try…

- **Which products are out of stock in one or more stores, but company as a whole has some somewhere? Which store is out of stock?**

DALHOUSIE UNIVERSITY
Inspiring Minds