Post-class Exercises: Nested Queries

Schema and data

We will work on the DineSafe database, which is a collection of food safety inspections carried out between 2010 and 2012 in Toronto.

1. (a) Find out the establishment that has received the highest amount of fine for a single infraction. Report the name and the address of the establishment, as well as the amount of the fine. If there are ties, report for each of them.

Solution:

```
SELECT name, addr, fine
FROM establishment, inspection, infraction
WHERE establishment.id = inspection.est_id
AND inspection.id = infraction.insp_id
AND fine = (SELECT MAX(fine)
            FROM infraction);
```

Output:

```
<table>
<thead>
<tr>
<th>name</th>
<th>addr</th>
<th>fine</th>
</tr>
</thead>
<tbody>
<tr>
<td>FARM FRESH SUPERMARKET</td>
<td>4466 SHEPPARD AVE E</td>
<td>31250</td>
</tr>
</tbody>
</table>
```

(b) Find out every establishment that has ever paid any fine because of some infraction, ordered by their names alphabetically.

Solution:

```
SELECT name, addr
FROM establishment
WHERE id IN (SELECT est_id
              FROM inspection, infraction
              WHERE inspection.id = infraction.insp_id
              AND fine > 0)
ORDER BY name;
```

Output:

```
<table>
<thead>
<tr>
<th>name</th>
<th>addr</th>
</tr>
</thead>
<tbody>
<tr>
<td>786 HALAL RESTAURANT</td>
<td>1330 GERRARD ST E</td>
</tr>
<tr>
<td>ABC BAKERY AND COFFEE</td>
<td>3618 VICTORIA PARK AVE</td>
</tr>
<tr>
<td>AJI SAI SUSHI JAPANESE RESTAURANT</td>
<td>813 YONGE ST</td>
</tr>
<tr>
<td>AKIA KING CAFE</td>
<td>387 BROADVIEW AVE</td>
</tr>
<tr>
<td>AL LAGO RISTORANTE</td>
<td>3423 LAKE SHORE BLVD W</td>
</tr>
</tbody>
</table>
```
(c) Find out the establishment which a total amount of fine over $3000 in the history.

Solution:
SELECT name, addr, totalFine
FROM establishment, (  
    SELECT est_id, SUM(fine) as totalFine
    FROM inspection, infraction
    WHERE inspection.id = infraction.insp_id
    AND fine > 0
    GROUP BY est_id
) AS R1
WHERE id = est_id AND totalFine > 3000;

Output:

<table>
<thead>
<tr>
<th>name</th>
<th>addr</th>
<th>totalFine</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHINA CITY SUPERMARKET</td>
<td>247 SPADINA AVE</td>
<td>3125</td>
</tr>
<tr>
<td>EDUARDAS BBQ POULTRY TAKE OUT AND FOOD STUFF</td>
<td>1546 DUPONT ST</td>
<td>3067.5</td>
</tr>
<tr>
<td>HO-LEE CHOW</td>
<td>3466 DUNDAS ST W</td>
<td>6095</td>
</tr>
<tr>
<td>FARM FRESH SUPERMARKET</td>
<td>4466 SHEPPARD AVE E</td>
<td>36250</td>
</tr>
<tr>
<td>MASHION BAKERY</td>
<td>345 SPADINA AVE</td>
<td>4330</td>
</tr>
</tbody>
</table>
(5 rows)

2. Fine the top 5 most popular type of establishment and the number of such establishment on Bloor Street West (Its address contains “BLOOR ST W”).

Solution:
SELECT description, cnt
FROM est_type NATURAL JOIN (  
    SELECT type_id AS id, count(*) as cnt
    FROM establishment
    WHERE addr LIKE '%BLOOR ST W%
    GROUP BY type_id
    ORDER BY cnt DESC LIMIT 1
) AS R;

Output:

<table>
<thead>
<tr>
<th>description</th>
<th>cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restaurant</td>
<td>335</td>
</tr>
<tr>
<td>Food Store (Convenience / Variety)</td>
<td>96</td>
</tr>
<tr>
<td>Food Take Out</td>
<td>70</td>
</tr>
</tbody>
</table>
(5 rows)
3. Find the top 10 establishment that has the most infractions in one single inspection. Report the name and the address of the establishment, as well as the number of infractions in that inspection. If there are ties, report for each of them.

Solution:

```sql
SELECT name, addr, cnt
FROM establishment, inspection, (  
    SELECT insp_id, count(*) AS cnt  
    FROM infraction  
    GROUP BY insp_id  
    ORDER BY cnt DESC LIMIT 10  
) AS R  
WHERE establishment.id = inspection.est_id  
AND inspection.id = R.insp_id;
```

Output:

```
<table>
<thead>
<tr>
<th>name</th>
<th>addr</th>
<th>cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRE MARI BAKERY</td>
<td>1311 ST CLAIR AVE W</td>
<td>24</td>
</tr>
<tr>
<td>B TRUST SUPERMARKET</td>
<td>1105 WILSON AVE</td>
<td>19</td>
</tr>
<tr>
<td>MARIKO JAPANESE RESTAURANT</td>
<td>551 BLOOR ST W</td>
<td>19</td>
</tr>
<tr>
<td>AMAYA BREAD BAR</td>
<td>3305 YONGE ST</td>
<td>18</td>
</tr>
<tr>
<td>PHILTHY MCMASTYS</td>
<td>130 EGLINTON AVE E</td>
<td>18</td>
</tr>
<tr>
<td>FROSHIGHA KABUL HALAL MEAT</td>
<td>1067 DANFORTH AVE</td>
<td>18</td>
</tr>
<tr>
<td>CUISINE OF INDIA CATERING</td>
<td>40 MAGNETIC DR</td>
<td>18</td>
</tr>
<tr>
<td>MI PHO NHA TRANG</td>
<td>1365 WILSON AVE</td>
<td>18</td>
</tr>
<tr>
<td>MANGIA MANGIA MANGIA RESTAURANT</td>
<td>4700 KEELE ST</td>
<td>17</td>
</tr>
<tr>
<td>GOLDEN PIZZA RESTAURANT</td>
<td>1201 BROADVIEW AVE</td>
<td>17</td>
</tr>
</tbody>
</table>
```

4. In this query, we want to study if the minimal annual inspections are properly conducted. We will break this complicated query into some small steps, each of them involves using a nested query itself. You will find it handy to handle these complicated queries using views if possible.

(a) Find out the establishment that was never inspected in year 2010. Report their names and addresses, ordered by their names alphabetically. Hint: for PostgreSQL, use `extract(year from date)` to extract the year from the date; for SQLite, need to use `strftime('%Y', date)`.

Solution:

```sql
SELECT name, addr  
FROM establishment  
WHERE id NOT IN (  
    SELECT est_id  
    FROM inspection  
    WHERE extract(year from date) = 2010  
)  
ORDER BY name;
```
Output:

<table>
<thead>
<tr>
<th>name</th>
<th>addr</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 HOUR CAFETERIA</td>
<td>435 SPADINA AVE</td>
</tr>
<tr>
<td>1 PLUS 3 PIZZA &amp; WINGS</td>
<td>1798 JANE ST</td>
</tr>
<tr>
<td>100 KM FOODS INC.</td>
<td>4478 CHESSWOOD DR</td>
</tr>
<tr>
<td>1000 VARIETY</td>
<td>1000 PAPE AVE</td>
</tr>
<tr>
<td>ZLD</td>
<td>1120 BELLAMY RD N</td>
</tr>
<tr>
<td>ZOBEL</td>
<td>1160 DANFORTH AVE</td>
</tr>
<tr>
<td>ZUCCA TRATTORIA</td>
<td>2150 YONGE ST</td>
</tr>
<tr>
<td>iBENTO</td>
<td>235 COLLEGE ST</td>
</tr>
</tbody>
</table>

For each establishment, find out the actual inspections happened in year 2010. Hint: don’t forget to include those with zero inspection. Verify your result by checking whether the number of returned rows equals to the number of rows in the establishment relation.

Solution:

SELECT name, addr, cnt
FROM establishment INNER JOIN (  
(SELECT est_id, COUNT(*) AS cnt  
FROM inspection INNER JOIN establishment  
ON inspection.est_id = establishment.id  
WHERE extract(year from date) = 2010  
GROUP BY est_id  
) UNION (  
SELECT id, 0 AS cnt  
FROM establishment  
WHERE id NOT IN (  
SELECT est_id  
FROM inspection  
WHERE extract(year from date) = 2010  
)  
)) AS R  
ON establishment.id = R.est_id
ORDER BY name;

Output:

<table>
<thead>
<tr>
<th>name</th>
<th>addr</th>
<th>cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>(THE) HOLE IN THE WALL IN THE JUNCTION</td>
<td>2867 DUNDAS ST W</td>
<td>1</td>
</tr>
<tr>
<td>1 HOUR CAFETERIA</td>
<td>435 SPADINA AVE</td>
<td>0</td>
</tr>
<tr>
<td>1 PLUS 1 PIZZA &amp; WINGS</td>
<td>361 OAKWOOD AVE</td>
<td>1</td>
</tr>
<tr>
<td>1 PLUS 2 PIZZA &amp; WING</td>
<td>3260 DUNDAS ST W</td>
<td>1</td>
</tr>
<tr>
<td>ZYKA FINE FOODS &amp; BBQ</td>
<td>2535 WARDEN AVE</td>
<td>2</td>
</tr>
<tr>
<td>ZYNG</td>
<td>730 YONGE ST</td>
<td>2</td>
</tr>
<tr>
<td>iBENTO</td>
<td>235 COLLEGE ST</td>
<td>0</td>
</tr>
</tbody>
</table>
(13921 rows)

(c) Find out every establishment that has ever failed to meet the minimal number of required annual inspections in year 2010, ordered by name alphabetically. For each of them, report its name, address, minimal number of required annual inspections, and actual inspections conducted in 2010. Hint: start your work from the results in Q4(b).

Solution:

SELECT name, addr, minspec, cnt
FROM establishment, (
    (SELECT est_id, COUNT(*) AS cnt
     FROM inspection INNER JOIN establishment
     ON inspection.est_id = establishment.id
     WHERE extract(year from date) = 2010
     GROUP BY est_id
    ) UNION (
    SELECT id, 0 AS cnt
    FROM establishment
    WHERE id NOT IN (
      SELECT est_id
      FROM inspection
      WHERE extract(year from date) = 2010
    )
    ) AS R
WHERE minspec > cnt AND id = est_id
ORDER BY name;

Output:

<table>
<thead>
<tr>
<th>name</th>
<th>addr</th>
<th>minspec</th>
<th>cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>THE HOLE IN THE WALL IN THE JUNCTION</td>
<td>2867 DUNDAS ST W</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1 HOUR CAFETERIA</td>
<td>435 SPADINA AVE</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>1 PLUS 1 PIZZA &amp; WINGS</td>
<td>361 OAKWOOD AVE</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1 PLUS 2 PIZZA &amp; WING</td>
<td>3260 DUNDAS ST W</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1 PLUS 3 PIZZA &amp; WINGS</td>
<td>1798 JANE ST</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>ZUPA'S RESTAURANT &amp; DELI</td>
<td>342 ADELAIDE ST W</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>ZYKA FINE FOODS &amp; BBQ</td>
<td>2535 WARDEN AVE</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>ZYNG</td>
<td>730 YONGE ST</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>iBENTO</td>
<td>235 COLLEGE ST</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>
(12112 rows)

5. According to DineSafe Inspection and Disclosure System, a “Conditional Pass” notice will be issued when one or more significant infractions are observed during an inspection, and a second re-inspection will be conducted within 24-48 hours. Find out the number of times that an establishment was issued a “Conditional Pass” notice at first inspection, but no re-inspection was conducted within 24-48 hours (holes in execution!). Hint: for PostgreSQL, use `extract(day from date)` to extract the day from the date; for SQLite, need to use `julianday(R2.date)`.

Solution:
SELECT count(*)
FROM inspection AS R1, est_status
WHERE status_id = est_status.id
  AND description = 'Conditional Pass'
  AND NOT EXISTS (  
    SELECT *
    FROM inspection AS R2
    WHERE R2.est_id = R1.est_id
    AND extract(day from R2.date) - extract(day from R1.date) >= 1
    AND extract(day from R2.date) - extract(day from R1.date) <= 2
  );

Output:

<table>
<thead>
<tr>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1739</td>
</tr>
</tbody>
</table>
(1 row)

6. In the database, you may have found several establishment have the same name but in different locations. These are franchise stores or chain stores, e.g., McDonald’s.

(a) Find the name of the top 5 largest chains in the database. For each of them, report its name and the number of stores it owns in Toronto, ordered by the number in descending order.

Solution:

SELECT name, count(*) as stores
FROM establishment
GROUP BY name
HAVING count(*) > 1
ORDER BY stores DESC LIMIT 5;

Output:

<table>
<thead>
<tr>
<th>name</th>
<th>stores</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIM HORTONS</td>
<td>203</td>
</tr>
<tr>
<td>SUBWAY</td>
<td>169</td>
</tr>
<tr>
<td>PIZZA PIZZA</td>
<td>107</td>
</tr>
<tr>
<td>SHOPPERS DRUG MART</td>
<td>73</td>
</tr>
<tr>
<td>SECOND CUP</td>
<td>66</td>
</tr>
</tbody>
</table>
(5 rows)

(b) For each of the chain brand in (a), report the average minimal annual inspections per store, in the same order as in (a).

Solution:

SELECT name, SUM(minspec) * 1.0 / stores AS avgMinSpec
FROM establishment NATURAL JOIN(
    SELECT name, count(*) as stores
    FROM establishment
    GROUP BY name
    HAVING count(*) > 1
    ORDER BY stores DESC LIMIT 5;
FROM establishment  
GROUP BY name  
HAVING count(*) > 1  
ORDER BY stores DESC LIMIT 5  
) AS R  
GROUP BY name, stores  
ORDER BY stores;

Output:

<table>
<thead>
<tr>
<th>name</th>
<th>avgminspec</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECOND CUP</td>
<td>1.4696969696969697</td>
</tr>
<tr>
<td>SHOPPERS DRUG MART</td>
<td>1.0000000000000000</td>
</tr>
<tr>
<td>PIZZA PIZZA</td>
<td>2.0093457943925234</td>
</tr>
<tr>
<td>SUBWAY</td>
<td>2.0059171597633136</td>
</tr>
<tr>
<td>TIM HORTONS</td>
<td>1.9901477832512315</td>
</tr>
</tbody>
</table>

(5 rows)

(c) Find out if any of the chain brand in (a), has received any inspections with a “Closed” status in the history. If yes, report the name, the number of “Closed” inspections for each of such chain brand.

Solution:

SELECT name, COUNT(*) AS cnt  
FROM inspection INNER JOIN establishment  
ON inspection.est_id = establishment.id  
NATURAL JOIN (  
SELECT name, count(*) as stores  
FROM establishment  
GROUP BY name  
HAVING count(*) > 1  
ORDER BY stores DESC LIMIT 5  
) AS R  
WHERE status_id = (  
SELECT id  
FROM est_status  
WHERE description = 'Closed'  
)  
GROUP BY name, stores  
ORDER BY stores;

Output:

<table>
<thead>
<tr>
<th>name</th>
<th>cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBWAY</td>
<td>1</td>
</tr>
</tbody>
</table>

(1 row)

7. (a) Find the pass rate of “SUSHI WORLD” on COLLEGE ST. Report the total number of passed inspections, total number of previous inspections, and the pass rate in its history.
Solution:

```
SELECT name, addr, passes, specs, passes * 1.0 / specs AS rate
FROM (  
    SELECT est_id, count(*) AS passes
    FROM inspection, est_status
    WHERE status_id = est_status.id
        AND description = 'Pass'
    GROUP BY est_id
) AS R1 NATURAL JOIN (  
    SELECT est_id, count(*) AS specs
    FROM inspection
    GROUP BY est_id
) AS R2, establishment
WHERE est_id = id
    AND name = 'SUSHI WORLD'
    AND addr LIKE '%COLLEGE ST%';
```

Output:

```
name  | addr             | passes | specs | rate
-------------------------------+-----------------+--------+-------+------------------------
SUSHI WORLD | 281 COLLEGE ST | 5      | 5     | 1.00000000000000000000
(1 row)
```

(b) A very hard one: For each establishment, find its pass rate among all its previous inspections in the database. Report the name, address, total number of passed inspections, total number of previous inspections, and the pass rate. Order your results in an descending order of the pass rate, followed by the total number of inspections (check some of your favorite restaurants!). Verify that the number of rows in your results equals to the number of establishment in the database.

Solution: The follow query will only return those establishment with recorded passes.

```
SELECT name, addr, passes, specs, passes * 1.0 / specs AS rate
FROM (  
    SELECT est_id, count(*) AS passes
    FROM inspection, est_status
    WHERE status_id = est_status.id
        AND description = 'Pass'
    GROUP BY est_id
) AS R1 NATURAL JOIN (  
    SELECT est_id, count(*) AS specs
    FROM inspection
    GROUP BY est_id
) AS R2, establishment
WHERE est_id = id
ORDER BY rate DESC, specs DESC;
```

Need to union those with no recorded passes as well.

```
SELECT name, addr, passes, specs, passes * 1.0 / specs AS rate
FROM (  
    (SELECT est_id, count(*) AS passes
     FROM inspection, est_status
     ```
WHERE status_id = est_status.id
AND description = 'Pass'
GROUP BY est_id
) UNION (
  SELECT est_id, 0 AS passes
FROM inspection
WHERE est_id NOT IN (
  SELECT est_id
  FROM inspection, est_status
  WHERE status_id = est_status.id
  AND description = 'Pass'
)) AS R1 NATURAL JOIN (
  SELECT est_id, count(*) AS specs
  FROM inspection
  GROUP BY est_id
) AS R2, establishment
WHERE est_id = id
ORDER BY rate DESC, specs DESC;

Output:

<table>
<thead>
<tr>
<th>name</th>
<th>addr</th>
<th>passes</th>
<th>specs</th>
<th>rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRESH &amp; WILD</td>
<td>69 SPADINA AVE</td>
<td>10</td>
<td>10</td>
<td>1.000</td>
</tr>
<tr>
<td>LITTLE INDIA RESTAURANT</td>
<td>255 QUEEN ST W</td>
<td>10</td>
<td>10</td>
<td>1.000</td>
</tr>
<tr>
<td>ZELLERS RESTAURANT</td>
<td>500 REXDALE BLVD</td>
<td>9</td>
<td>9</td>
<td>1.000</td>
</tr>
<tr>
<td>MANCHU WOK</td>
<td>1800 SHEPPARD AVE E</td>
<td>9</td>
<td>9</td>
<td>1.000</td>
</tr>
<tr>
<td>BOSTON PIZZA</td>
<td>16 LESLIE ST</td>
<td>8</td>
<td>8</td>
<td>1.000</td>
</tr>
<tr>
<td>HOT DOG CART</td>
<td>80 TORO RD</td>
<td>0</td>
<td>1</td>
<td>0.000</td>
</tr>
<tr>
<td>OAKDALE PARK MIDDLE SCHO...</td>
<td>315 GRANDRAVINE DR</td>
<td>0</td>
<td>1</td>
<td>0.000</td>
</tr>
<tr>
<td>OMM KALSOUM</td>
<td>2175 LAWRENCE AVE E</td>
<td>0</td>
<td>1</td>
<td>0.000</td>
</tr>
<tr>
<td>BENTO</td>
<td>3501 YONGE ST</td>
<td>0</td>
<td>1</td>
<td>0.000</td>
</tr>
<tr>
<td>(13921 rows)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8. Which rule has been violated the most?

Solution:

```
SELECT rule.*, cnt
FROM rule, (
  SELECT rule_id, count(*) AS cnt
  FROM infraction
  GROUP BY rule_id
) AS R1
WHERE id = rule_id AND NOT EXISTS (
  SELECT rule_id, count(*)
  FROM infraction
  GROUP BY rule_id
  HAVING count(*) > cnt
);
```
Output:

<table>
<thead>
<tr>
<th>id</th>
<th>code</th>
<th>sec</th>
<th>subsec</th>
<th>par</th>
<th>subpar</th>
<th>cnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>O. Reg. 562/90</td>
<td>19</td>
<td></td>
<td></td>
<td></td>
<td>3358</td>
</tr>
</tbody>
</table>
(1 row)