

```

1  /* *****
2      INTRODUCTION TO STRUCTURED QUERY LANGUAGE FOR DATA ANALYTICS
3          SF25SQL6172025, 2025/11/17 - 2025/12/17
4          https://folvera.commons.gc.cuny.edu/?cat=35
5  *****
6
7  SESSION #1 (2025/11/17): COOKIES & MILK NOW, CODE LATER
8
9  1. For this example with cookies, we first need to create the data. At this
10     point of the course, you should not be concerned about the code, but rather
11     what the language can do. We will have fun writing code starting on day 2.
12
13     1.1. On the right side, there is a quick explanation what each line (part
14           of a single statement) does.
15
16     1.2. Note that a statement starts with an English verb and ends with a
17           semicolon.
18  ***** */
19
20  CREATE DATABASE food_stores;           -- 1. creating database of food
21                                         -- stores (`food_stores`)
22  CREATE SCHEMA nyc;                   -- 2. creating schema for food
23                                         -- stores in NYC (`nyc`)
24  CREATE TABLE nyc.snack_store (      -- 3. creating table for stores
25     record_id INT NOT NULL,           -- where we are shopping
26     store_id INT,                     -- (`snack_store`) in schema
27     snack_id INT,                    -- `nyc`; in other words,
28     food_type VARCHAR(50),           -- another food store in NYC
29     flavor VARCHAR(50),
30     organic VARCHAR(1),
31     dietary_restriction VARCHAR(50),
32     package_size VARCHAR(50),
33     brand_name VARCHAR(50)
34 );
35
36  INSERT INTO nyc.snack_store          -- 4. inserting values in table
37  VALUES (                             -- `nyc.snack_store`
38     1,
39     1,
40     1,
41     'cookie',
42     'chocolate',
43     'y',
44     'gluten free',
45     'small',
46     'Good Cookies'
47 ),
48 (
49     2,
50     1,
51     1,
52     'cookie',
53     'chocolate',
54     'y',
55     '',
56     'small',
57     'Good Cookies'
58 ),
59 (
60     3,
61     1,
62     1,
63     'cookie',
64     'chocolate',
65     'y',
66     'nut free',
67     'small',
68     'Good Cookies'
69 ),

```

```
70  (
71  4,
72  1,
73  2,
74  'donut',
75  'chocolate',
76  'y',
77  'gluten free',
78  'small',
79  'Good Cookies'
80  ),
81  (
82  5,
83  2,
84  1,
85  'cookie',
86  'chocolate',
87  'y',
88  '',
89  'small',
90  'Good Cookies'
91  ),
92  (
93  6,
94  2,
95  1,
96  'cookie',
97  'vanilla',
98  'y',
99  'gluten free',
100 'large',
101 'Cool Snacks'
102 ),
103 (
104 7,
105 2,
106 1,
107 'cookie',
108 'vanilla',
109 'y',
110 'nut free',
111 'single serving',
112 'Cool Snacks'
113 ),
114 (
115 8,
116 1,
117 1,
118 'cookie',
119 'chocolate',
120 'n',
121 'gluten free',
122 'small',
123 'Cookies by Alice'
124 ),
125 (
126 9,
127 3,
128 1,
129 'cookie',
130 'chocolate',
131 'n',
132 'gluten free',
133 'small',
134 'Cookies by Alice'
135 ),
136 (
137 10,
138 3,
```

```
139 1,
140 'cookie',
141 'chocolate',
142 'y',
143 'gluten free',
144 'small',
145 'Good Cookies'
146 ),
147 (
148 11,
149 3,
150 1,
151 'cookie',
152 'vanilla',
153 'y',
154 'gluten free',
155 'small',
156 'Cool Snacks'
157 ),
158 (
159 12,
160 1,
161 2,
162 'donut',
163 'jelly',
164 'y',
165 'gluten free',
166 'small',
167 'Good Cookies'
168 ),
169 (
170 13,
171 1,
172 1,
173 'cookie',
174 'chocolate',
175 'n',
176 'gluten free',
177 'small',
178 'Cookies by Alice'
179 ),
180 (
181 14,
182 3,
183 1,
184 'cookie',
185 'chocolate',
186 'n',
187 'gluten free',
188 'small',
189 'Cookies by Alice'
190 ),
191 (
192 15,
193 3,
194 1,
195 'cookie',
196 'chocolate',
197 'n',
198 'gluten free',
199 'small',
200 'Cookies by Alice'
201 ),
202 (
203 16,
204 3,
205 1,
206 'cookie',
207 'chocolate',
```

```

208     'y',
209     '',
210     'large',
211     'Grand Cookies'
212 ),
213 (
214     17,
215     3,
216     1,
217     'cookie',
218     'vanilla',
219     'y',
220     'vegan',
221     'large',
222     'Good Cookies'
223 ),
224 (
225     18,
226     3,
227     1,
228     'cookie',
229     'vanilla',
230     'y',
231     'vegan',
232     'large',
233     'Good Cookies'
234 ),
235 (
236     19,
237     3,
238     1,
239     'cookie',
240     'chocolate',
241     'y',
242     'gluten free',
243     'small',
244     'Grand Cookies'
245 );
246
247 CREATE TABLE nyc.store_names (
248     store_id INT NOT NULL,
249     store_name VARCHAR(50)
250 );
251
252 INSERT INTO nyc.store_names
253 VALUES (
254     1,
255     'The Nature Shop'
256 ),
257 (
258     2,
259     'Supermarket Cool Banana'
260 ),
261 (
262     3,
263     'The Corner Shop'
264 );
265
266 SELECT *
267 FROM nyc.store_names;
268
269
270 /* *****
271     1.3. In this example, you are an assistant to a boss who wants cookies from
272         `The Corner Shop`.
273
274         We will keep adding line by line to redefine what we need. As you can
275         see, the English request is translated to SQL in an English-like
276         series of expressions.

```

```

277
278     Note that, if you read the statement out loud and it does not sound
279     right as an English imperative sentence (a command, in this case given
280     to a computer), most likely your syntax is wrong.
281     *****/
282
283 SELECT *                               -- 1. getting all information
284 FROM nyc.snack_store                   -- 2. from specific population
285                                         -- `nyc.snack_store`
286 WHERE food_type = 'cookie'             -- 3. where specific
287                                         -- `food_type` is `cookie`
288     AND store_id = 3;                  -- `store_id` is `3`, the
289                                         -- identifier for `The
290                                         -- Corner Shop`
291
292
293 /* *****/
294     1.4. You go to `The Corner Shop` and bring multiple cookies including
295     doubles. Your boss then says that you should not bring doubles.
296     *****/
297
298 SELECT DISTINCT store_id,               -- `DISTINCT` = no doubles,
299     snack_id,                           -- unique cookies, excluding
300     food_type,                           -- the row identifier
301     flavor,
302     organic,
303     dietary_restriction,
304     package_size,
305     brand_name
306 FROM nyc.snack_store
307 WHERE food_type = 'cookie'
308     AND store_id = 3;
309
310
311 /* *****/
312     1.5. You go back to the store and bring back one of each cookie. Then your
313     boss says that he/she wants chocolate cookies.
314     *****/
315
316 SELECT DISTINCT store_id,               -- 1. still looking for all
317     snack_id,                           -- information
318     food_type,
319     flavor,
320     organic,
321     dietary_restriction,
322     package_size,
323     brand_name
324 FROM nyc.snack_store                   -- 2. from specific population
325                                         -- `nyc.snack_store`
326 WHERE food_type = 'cookie'             -- 3. still looking for a
327                                         -- `food_type` of `cookie`
328     AND store_id = 3                    -- and `store_id` 3
329     AND flavor = 'chocolate';          -- 4. new column/value
330                                         -- `flavor` is `chocolate`
331                                         -- 5. hence looking for a
332                                         -- chocolate cookies
333
334
335 /* *****/
336     1.6. You go back to the store and bring back only chocolate cookies. Then
337     your boss says that the cookies must be organic.
338     *****/
339
340 SELECT DISTINCT store_id,
341     snack_id,
342     food_type,
343     organic,
344     dietary_restriction,
345     package_size,

```

```

346     brand_name
347 FROM nyc.snack_store
348 WHERE food_type = 'cookie'
349     AND store_id = 3
350     AND flavor = 'chocolate'
351     AND organic = 'y';
352
353
354 /* *****
355     1.7. You go back to the store and bring back only organic chocolate
356         cookies. Then your boss says that the cookies should also be
357         gluten-free.
358     ***** */
359
360 SELECT DISTINCT store_id,
361     snack_id,
362     food_type,
363     organic,
364     dietary_restriction,
365     package_size,
366     brand_name
367 FROM nyc.snack_store
368 WHERE food_type = 'cookie'
369     AND store_id = 3
370     AND flavor = 'chocolate'
371     AND organic = 'y'
372     AND dietary_restriction = 'gluten free';
373
374
375 /* *****
376     1.8. You go back to the store and bring back only gluten-free and organic
377         chocolate cookies. Then your boss says the package is small.
378     ***** */
379
380 SELECT DISTINCT store_id,
381     snack_id,
382     food_type,
383     organic,
384     dietary_restriction,
385     package_size,
386     brand_name
387 FROM nyc.snack_store
388 WHERE food_type = 'cookie'
389     AND store_id = 3
390     AND flavor = 'chocolate'
391     AND organic = 'y'
392     AND dietary_restriction = 'gluten free'
393     AND package_size = 'small';
394
395
396 /* *****
397     1.9. You go back to the store and bring back only small packages of
398         gluten-free and organic chocolate cookies. Then your boss says that
399         the brand name should be `Good Cookies`.
400     ***** */
401
402 SELECT DISTINCT store_id,
403     snack_id,
404     food_type,
405     organic,
406     dietary_restriction,
407     package_size,
408     brand_name
409 FROM nyc.snack_store
410 WHERE food_type = 'cookie'
411     AND store_id = 3
412     AND flavor = 'chocolate'
413     AND organic = 'y'
414     AND dietary_restriction = 'gluten free'

```

```
415     AND package_size = 'small'
416     AND brand_name = 'Good Cookies';
417
418
419 /* *****
420     1.10. You go back to the store and bring back only small packages of `Good
421         Cookies` gluten-free and organic chocolate cookies. At this point,
422         your boss is finally satisfied.
423
424         Since the values of `cookies` and `donuts` are repeated in several
425         records (rows), these values could be moved to another table and
426         referred to with a unique identifier (a row ID). The values of
427         `chocolate` and `vanilla` can also be moved to another table.
428
429         This is the concept of relational databases where tables have unique
430         values that can be referred by unique identifiers -- for example,
431         `food_type_id` for `cookies` and `donuts` and `flavor_id` for
432         `chocolate` and `vanilla`.
433
434     1.11. Now that you have a general idea of what SQL does, we can start to
435         look at the code.
436
437     *****
438     https://folvera.commons.gc.cuny.edu/?p=1321
439     ***** */
```