

```

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 #5 (2025/12/01): MANIPULATING DATA
8
9  1. Using clauses `BETWEEN`, `NOT`, `UNION`, `EXCEPT` and `INTERSECT`
10 2. Understanding function `FORMAT()` for dates and currencies including
11   culture codes
12 *****
13
14 1. LAB #3
15   Write a query
16   1.01. to call all columns and values from `AP1.Vendors` and any related
17         values from `AP1.ContactUpdates` (`LEFT JOIN`),
18   1.02. to put together `FirstName` and `LastName` in one field with alias
19         `ContactName`,
20   1.03. to put together the first letter of `FirstName`, the complete
21         `LastName`, `@`, `VendorName` (removing empty spaces between words and
22         special characters like `&` and `,`) and `.com` as `ContactEmail`
23         presenting the output in lower case,
24   1.04. and to put together `VendorContactFName` and `VendorContactLName`
25         with aliases `VendorContactName` and `VendorContactEmail` (like
26         #1.02).
27 ***** */
28
29 SELECT AP1.Vendors.VendorID,
30         AP1.Vendors.VendorName,
31         AP1.Vendors.VendorAddress1,
32         AP1.Vendors.VendorAddress2,
33         AP1.Vendors.VendorCity,
34         AP1.Vendors.VendorState,
35         AP1.Vendors.VendorZipCode,
36         AP1.Vendors.VendorPhone,
37         AP1.Vendors.VendorContactFName +
38         ' ' +
39         AP1.Vendors.VendorContactLName
40         AS ContactName,
41
42         LEFT (AP1.Vendors.VendorContactFName, 1) +
43         AP1.Vendors.VendorContactLName +
44         '@' +
45
46         REPLACE (
47             REPLACE (
48                 REPLACE (
49                     REPLACE (
50                         REPLACE (
51                             REPLACE (
52                                 REPLACE (
53                                     REPLACE (AP1.Vendors.VendorName, ' ', '')
54                                     -- 2.06. generating value #1
55                                     -- 2.07. generating value #2
56                                     -- 2.08. generating value #3
57                                     -- 2.09. generating value #4
58                                     -- 2.10. generating value #5
59                                     -- 2.11. generating value #6
60                                     -- where `''`
61                                     -- represents a single
62                                     -- quote (`'`) used an
63                                     -- escape character
64                                     -- and `.foo` to complete
65                                     -- email with alias
66                                     -- `VendorContactEmail`
67
68                                     -- 2.01. getting value #5
69                                     -- 2.02. getting value #4
70                                     -- 2.03. getting value #3
71                                     -- 2.04. getting value #2
72                                     -- 2.05. getting value #1
73
74                                     -- 1. concatenation of
75                                     -- `FirstName`, a single
76                                     -- space and `LastName`
77                                     -- using `+` with alias
78                                     -- `ContactName`
79
80                                     -- 2. concatenation of
81                                     -- `VendorContactFName`,
82                                     -- a single space and
83                                     -- `VendorContactLName`,
84                                     -- the `@` sign, followed by
85                                     -- `VendorName` (value #6)
86
87                                     -- 1. concatenation of
88                                     -- `FirstName`, a single
89                                     -- space and `LastName`
90                                     -- using `+` with alias
91                                     -- `ContactName`
92
93                                     -- 2. concatenation of
94                                     -- `VendorContactFName`,
95                                     -- a single space and
96                                     -- `VendorContactLName`,
97                                     -- the `@` sign, followed by
98                                     -- `VendorName` (value #6)
99
100                                    -- 2.01. getting value #5
101                                    -- 2.02. getting value #4
102                                    -- 2.03. getting value #3
103                                    -- 2.04. getting value #2
104                                    -- 2.05. getting value #1
105
106                                    -- 1. concatenation of
107                                    -- `FirstName`, a single
108                                    -- space and `LastName`
109                                    -- using `+` with alias
110                                    -- `ContactName`
111
112                                    -- 2. concatenation of
113                                    -- `VendorContactFName`,
114                                    -- a single space and
115                                    -- `VendorContactLName`,
116                                    -- the `@` sign, followed by
117                                    -- `VendorName` (value #6)
118
119                                    -- 2.01. getting value #5
120                                    -- 2.02. getting value #4
121                                    -- 2.03. getting value #3
122                                    -- 2.04. getting value #2
123                                    -- 2.05. getting value #1
124
125                                    -- 1. concatenation of
126                                    -- `FirstName`, a single
127                                    -- space and `LastName`
128                                    -- using `+` with alias
129                                    -- `ContactName`
130
131                                    -- 2. concatenation of
132                                    -- `VendorContactFName`,
133                                    -- a single space and
134                                    -- `VendorContactLName`,
135                                    -- the `@` sign, followed by
136                                    -- `VendorName` (value #6)
137
138                                    -- 2.01. getting value #5
139                                    -- 2.02. getting value #4
140                                    -- 2.03. getting value #3
141                                    -- 2.04. getting value #2
142                                    -- 2.05. getting value #1
143
144                                    -- 1. concatenation of
145                                    -- `FirstName`, a single
146                                    -- space and `LastName`
147                                    -- using `+` with alias
148                                    -- `ContactName`
149
150                                    -- 2. concatenation of
151                                    -- `VendorContactFName`,
152                                    -- a single space and
153                                    -- `VendorContactLName`,
154                                    -- the `@` sign, followed by
155                                    -- `VendorName` (value #6)
156
157                                    -- 2.01. getting value #5
158                                    -- 2.02. getting value #4
159                                    -- 2.03. getting value #3
160                                    -- 2.04. getting value #2
161                                    -- 2.05. getting value #1
162
163                                    -- 1. concatenation of
164                                    -- `FirstName`, a single
165                                    -- space and `LastName`
166                                    -- using `+` with alias
167                                    -- `ContactName`
168
169                                    -- 2. concatenation of
170                                    -- `VendorContactFName`,
171                                    -- a single space and
172                                    -- `VendorContactLName`,
173                                    -- the `@` sign, followed by
174                                    -- `VendorName` (value #6)
175
176                                    -- 2.01. getting value #5
177                                    -- 2.02. getting value #4
178                                    -- 2.03. getting value #3
179                                    -- 2.04. getting value #2
180                                    -- 2.05. getting value #1
181
182                                    -- 1. concatenation of
183                                    -- `FirstName`, a single
184                                    -- space and `LastName`
185                                    -- using `+` with alias
186                                    -- `ContactName`
187
188                                    -- 2. concatenation of
189                                    -- `VendorContactFName`,
190                                    -- a single space and
191                                    -- `VendorContactLName`,
192                                    -- the `@` sign, followed by
193                                    -- `VendorName` (value #6)
194
195                                    -- 2.01. getting value #5
196                                    -- 2.02. getting value #4
197                                    -- 2.03. getting value #3
198                                    -- 2.04. getting value #2
199                                    -- 2.05. getting value #1
200
201                                    -- 1. concatenation of
202                                    -- `FirstName`, a single
203                                    -- space and `LastName`
204                                    -- using `+` with alias
205                                    -- `ContactName`
206
207                                    -- 2. concatenation of
208                                    -- `VendorContactFName`,
209                                    -- a single space and
210                                    -- `VendorContactLName`,
211                                    -- the `@` sign, followed by
212                                    -- `VendorName` (value #6)
213
214                                    -- 2.01. getting value #5
215                                    -- 2.02. getting value #4
216                                    -- 2.03. getting value #3
217                                    -- 2.04. getting value #2
218                                    -- 2.05. getting value #1
219
220                                    -- 1. concatenation of
221                                    -- `FirstName`, a single
222                                    -- space and `LastName`
223                                    -- using `+` with alias
224                                    -- `ContactName`
225
226                                    -- 2. concatenation of
227                                    -- `VendorContactFName`,
228                                    -- a single space and
229                                    -- `VendorContactLName`,
230                                    -- the `@` sign, followed by
231                                    -- `VendorName` (value #6)
232
233                                    -- 2.01. getting value #5
234                                    -- 2.02. getting value #4
235                                    -- 2.03. getting value #3
236                                    -- 2.04. getting value #2
237                                    -- 2.05. getting value #1
238
239                                    -- 1. concatenation of
240                                    -- `FirstName`, a single
241                                    -- space and `LastName`
242                                    -- using `+` with alias
243                                    -- `ContactName`
244
245                                    -- 2. concatenation of
246                                    -- `VendorContactFName`,
247                                    -- a single space and
248                                    -- `VendorContactLName`,
249                                    -- the `@` sign, followed by
250                                    -- `VendorName` (value #6)
251
252                                    -- 2.01. getting value #5
253                                    -- 2.02. getting value #4
254                                    -- 2.03. getting value #3
255                                    -- 2.04. getting value #2
256                                    -- 2.05. getting value #1
257
258                                    -- 1. concatenation of
259                                    -- `FirstName`, a single
260                                    -- space and `LastName`
261                                    -- using `+` with alias
262                                    -- `ContactName`
263
264                                    -- 2. concatenation of
265                                    -- `VendorContactFName`,
266                                    -- a single space and
267                                    -- `VendorContactLName`,
268                                    -- the `@` sign, followed by
269                                    -- `VendorName` (value #6)
270
271                                    -- 2.01. getting value #5
272                                    -- 2.02. getting value #4
273                                    -- 2.03. getting value #3
274                                    -- 2.04. getting value #2
275                                    -- 2.05. getting value #1
276
277                                    -- 1. concatenation of
278                                    -- `FirstName`, a single
279                                    -- space and `LastName`
280                                    -- using `+` with alias
281                                    -- `ContactName`
282
283                                    -- 2. concatenation of
284                                    -- `VendorContactFName`,
285                                    -- a single space and
286                                    -- `VendorContactLName`,
287                                    -- the `@` sign, followed by
288                                    -- `VendorName` (value #6)
289
290                                    -- 2.01. getting value #5
291                                    -- 2.02. getting value #4
292                                    -- 2.03. getting value #3
293                                    -- 2.04. getting value #2
294                                    -- 2.05. getting value #1
295
296                                    -- 1. concatenation of
297                                    -- `FirstName`, a single
298                                    -- space and `LastName`
299                                    -- using `+` with alias
300                                    -- `ContactName`
301
302                                    -- 2. concatenation of
303                                    -- `VendorContactFName`,
304                                    -- a single space and
305                                    -- `VendorContactLName`,
306                                    -- the `@` sign, followed by
307                                    -- `VendorName` (value #6)
308
309                                    -- 2.01. getting value #5
310                                    -- 2.02. getting value #4
311                                    -- 2.03. getting value #3
312                                    -- 2.04. getting value #2
313                                    -- 2.05. getting value #1
314
315                                    -- 1. concatenation of
316                                    -- `FirstName`, a single
317                                    -- space and `LastName`
318                                    -- using `+` with alias
319                                    -- `ContactName`
320
321                                    -- 2. concatenation of
322                                    -- `VendorContactFName`,
323                                    -- a single space and
324                                    -- `VendorContactLName`,
325                                    -- the `@` sign, followed by
326                                    -- `VendorName` (value #6)
327
328                                    -- 2.01. getting value #5
329                                    -- 2.02. getting value #4
330                                    -- 2.03. getting value #3
331                                    -- 2.04. getting value #2
332                                    -- 2.05. getting value #1
333
334                                    -- 1. concatenation of
335                                    -- `FirstName`, a single
336                                    -- space and `LastName`
337                                    -- using `+` with alias
338                                    -- `ContactName`
339
340                                    -- 2. concatenation of
341                                    -- `VendorContactFName`,
342                                    -- a single space and
343                                    -- `VendorContactLName`,
344                                    -- the `@` sign, followed by
345                                    -- `VendorName` (value #6)
346
347                                    -- 2.01. getting value #5
348                                    -- 2.02. getting value #4
349                                    -- 2.03. getting value #3
350                                    -- 2.04. getting value #2
351                                    -- 2.05. getting value #1
352
353                                    -- 1. concatenation of
354                                    -- `FirstName`, a single
355                                    -- space and `LastName`
356                                    -- using `+` with alias
357                                    -- `ContactName`
358
359                                    -- 2. concatenation of
360                                    -- `VendorContactFName`,
361                                    -- a single space and
362                                    -- `VendorContactLName`,
363                                    -- the `@` sign, followed by
364                                    -- `VendorName` (value #6)
365
366                                    -- 2.01. getting value #5
367                                    -- 2.02. getting value #4
368                                    -- 2.03. getting value #3
369                                    -- 2.04. getting value #2
370                                    -- 2.05. getting value #1
371
372                                    -- 1. concatenation of
373                                    -- `FirstName`, a single
374                                    -- space and `LastName`
375                                    -- using `+` with alias
376                                    -- `ContactName`
377
378                                    -- 2. concatenation of
379                                    -- `VendorContactFName`,
380                                    -- a single space and
381                                    -- `VendorContactLName`,
382                                    -- the `@` sign, followed by
383                                    -- `VendorName` (value #6)
384
385                                    -- 2.01. getting value #5
386                                    -- 2.02. getting value #4
387                                    -- 2.03. getting value #3
388                                    -- 2.04. getting value #2
389                                    -- 2.05. getting value #1
390
391                                    -- 1. concatenation of
392                                    -- `FirstName`, a single
393                                    -- space and `LastName`
394                                    -- using `+` with alias
395                                    -- `ContactName`
396
397                                    -- 2. concatenation of
398                                    -- `VendorContactFName`,
399                                    -- a single space and
400                                    -- `VendorContactLName`,
401                                    -- the `@` sign, followed by
402                                    -- `VendorName` (value #6)
403
404                                    -- 2.01. getting value #5
405                                    -- 2.02. getting value #4
406                                    -- 2.03. getting value #3
407                                    -- 2.04. getting value #2
408                                    -- 2.05. getting value #1
409
410                                    -- 1. concatenation of
411                                    -- `FirstName`, a single
412                                    -- space and `LastName`
413                                    -- using `+` with alias
414                                    -- `ContactName`
415
416                                    -- 2. concatenation of
417                                    -- `VendorContactFName`,
418                                    -- a single space and
419                                    -- `VendorContactLName`,
420                                    -- the `@` sign, followed by
421                                    -- `VendorName` (value #6)
422
423                                    -- 2.01. getting value #5
424                                    -- 2.02. getting value #4
425                                    -- 2.03. getting value #3
426                                    -- 2.04. getting value #2
427                                    -- 2.05. getting value #1
428
429                                    -- 1. concatenation of
430                                    -- `FirstName`, a single
431                                    -- space and `LastName`
432                                    -- using `+` with alias
433                                    -- `ContactName`
434
435                                    -- 2. concatenation of
436                                    -- `VendorContactFName`,
437                                    -- a single space and
438                                    -- `VendorContactLName`,
439                                    -- the `@` sign, followed by
440                                    -- `VendorName` (value #6)
441
442                                    -- 2.01. getting value #5
443                                    -- 2.02. getting value #4
444                                    -- 2.03. getting value #3
445                                    -- 2.04. getting value #2
446                                    -- 2.05. getting value #1
447
448                                    -- 1. concatenation of
449                                    -- `FirstName`, a single
450                                    -- space and `LastName`
451                                    -- using `+` with alias
452                                    -- `ContactName`
453
454                                    -- 2. concatenation of
455                                    -- `VendorContactFName`,
456                                    -- a single space and
457                                    -- `VendorContactLName`,
458                                    -- the `@` sign, followed by
459                                    -- `VendorName` (value #6)
460
461                                    -- 2.01. getting value #5
462                                    -- 2.02. getting value #4
463                                    -- 2.03. getting value #3
464                                    -- 2.04. getting value #2
465                                    -- 2.05. getting value #1
466
467                                    -- 1. concatenation of
468                                    -- `FirstName`, a single
469                                    -- space and `LastName`
470                                    -- using `+` with alias
471                                    -- `ContactName`
472
473                                    -- 2. concatenation of
474                                    -- `VendorContactFName`,
475                                    -- a single space and
476                                    -- `VendorContactLName`,
477                                    -- the `@` sign, followed by
478                                    -- `VendorName` (value #6)
479
480                                    -- 2.01. getting value #5
481                                    -- 2.02. getting value #4
482                                    -- 2.03. getting value #3
483                                    -- 2.04. getting value #2
484                                    -- 2.05. getting value #1
485
486                                    -- 1. concatenation of
487                                    -- `FirstName`, a single
488                                    -- space and `LastName`
489                                    -- using `+` with alias
490                                    -- `ContactName`
491
492                                    -- 2. concatenation of
493                                    -- `VendorContactFName`,
494                                    -- a single space and
495                                    -- `VendorContactLName`,
496                                    -- the `@` sign, followed by
497                                    -- `VendorName` (value #6)
498
499                                    -- 2.01. getting value #5
500                                    -- 2.02. getting value #4
501                                    -- 2.03. getting value #3
502                                    -- 2.04. getting value #2
503                                    -- 2.05. getting value #1
504
505                                    -- 1. concatenation of
506                                    -- `FirstName`, a single
507                                    -- space and `LastName`
508                                    -- using `+` with alias
509                                    -- `ContactName`
510
511                                    -- 2. concatenation of
512                                    -- `VendorContactFName`,
513                                    -- a single space and
514                                    -- `VendorContactLName`,
515                                    -- the `@` sign, followed by
516                                    -- `VendorName` (value #6)
517
518                                    -- 2.01. getting value #5
519                                    -- 2.02. getting value #4
520                                    -- 2.03. getting value #3
521                                    -- 2.04. getting value #2
522                                    -- 2.05. getting value #1
523
524                                    -- 1. concatenation of
525                                    -- `FirstName`, a single
526                                    -- space and `LastName`
527                                    -- using `+` with alias
528                                    -- `ContactName`
529
530                                    -- 2. concatenation of
531                                    -- `VendorContactFName`,
532                                    -- a single space and
533                                    -- `VendorContactLName`,
534                                    -- the `@` sign, followed by
535                                    -- `VendorName` (value #6)
536
537                                    -- 2.01. getting value #5
538                                    -- 2.02. getting value #4
539                                    -- 2.03. getting value #3
540                                    -- 2.04. getting value #2
541                                    -- 2.05. getting value #1
542
543                                    -- 1. concatenation of
544                                    -- `FirstName`, a single
545                                    -- space and `LastName`
546                                    -- using `+` with alias
547                                    -- `ContactName`
548
549                                    -- 2. concatenation of
550                                    -- `VendorContactFName`,
551                                    -- a single space and
552                                    -- `VendorContactLName`,
553                                    -- the `@` sign, followed by
554                                    -- `VendorName` (value #6)
555
556                                    -- 2.01. getting value #5
557                                    -- 2.02. getting value #4
558                                    -- 2.03. getting value #3
559                                    -- 2.04. getting value #2
560                                    -- 2.05. getting value #1
561
562                                    -- 1. concatenation of
563                                    -- `FirstName`, a single
564                                    -- space and `LastName`
565                                    -- using `+` with alias
566                                    -- `ContactName`
567
568                                    -- 2. concatenation of
569                                    -- `VendorContactFName`,
570                                    -- a single space and
571                                    -- `VendorContactLName`,
572                                    -- the `@` sign, followed by
573                                    -- `VendorName` (value #6)
574
575                                    -- 2.01. getting value #5
576                                    -- 2.02. getting value #4
577                                    -- 2.03. getting value #3
578                                    -- 2.04. getting value #2
579                                    -- 2.05. getting value #1
580
581                                    -- 1. concatenation of
582                                    -- `FirstName`, a single
583                                    -- space and `LastName`
584                                    -- using `+` with alias
585                                    -- `ContactName`
586
587                                    -- 2. concatenation of
588                                    -- `VendorContactFName`,
589                                    -- a single space and
590                                    -- `VendorContactLName`,
591                                    -- the `@` sign, followed by
592                                    -- `VendorName` (value #6)
593
594                                    -- 2.01. getting value #5
595                                    -- 2.02. getting value #4
596                                    -- 2.03. getting value #3
597                                    -- 2.04. getting value #2
598                                    -- 2.05. getting value #1
599
600                                    -- 1. concatenation of
601                                    -- `FirstName`, a single
602                                    -- space and `LastName`
603                                    -- using `+` with alias
604                                    -- `ContactName`
605
606                                    -- 2. concatenation of
607                                    -- `VendorContactFName`,
608                                    -- a single space and
609                                    -- `VendorContactLName`,
610                                    -- the `@` sign, followed by
611                                    -- `VendorName` (value #6)
612
613                                    -- 2.01. getting value #5
614                                    -- 2.02. getting value #4
615                                    -- 2.03. getting value #3
616                                    -- 2.04. getting value #2
617                                    -- 2.05. getting value #1
618
619                                    -- 1. concatenation of
620                                    -- `FirstName`, a single
621                                    -- space and `LastName`
622                                    -- using `+` with alias
623                                    -- `ContactName`
624
625                                    -- 2. concatenation of
626                                    -- `VendorContactFName`,
627                                    -- a single space and
628                                    -- `VendorContactLName`,
629                                    -- the `@` sign, followed by
630                                    -- `VendorName` (value #6)
631
632                                    -- 2.01. getting value #5
633                                    -- 2.02. getting value #4
634                                    -- 2.03. getting value #3
635                                    -- 2.04. getting value #2
636                                    -- 2.05. getting value #1
637
638                                    -- 1. concatenation of
639                                    -- `FirstName`, a single
640                                    -- space and `LastName`
641                                    -- using `+` with alias
642                                    -- `ContactName`
643
644                                    -- 2. concatenation of
645                                    -- `VendorContactFName`,
646                                    -- a single space and
647                                    -- `VendorContactLName`,
648                                    -- the `@` sign, followed by
649                                    -- `VendorName` (value #6)
650
651                                    -- 2.01. getting value #5
652                                    -- 2.02. getting value #4
653                                    -- 2.03. getting value #3
654                                    -- 2.04. getting value #2
655                                    -- 2.05. getting value #1
656
657                                    -- 1. concatenation of
658                                    -- `FirstName`, a single
659                                    -- space and `LastName`
660                                    -- using `+` with alias
661                                    -- `ContactName`
662
663                                    -- 2. concatenation of
664                                    -- `VendorContactFName`,
665                                    -- a single space and
666                                    -- `VendorContactLName`,
667                                    -- the `@` sign, followed by
668                                    -- `VendorName` (value #6)
669
670                                    -- 2.01. getting value #5
671                                    -- 2.02. getting value #4
672                                    -- 2.03. getting value #3
673                                    -- 2.04. getting value #2
674                                    -- 2.05. getting value #1
675
676                                    -- 1. concatenation of
677                                    -- `FirstName`, a single
678                                    -- space and `LastName`
679                                    -- using `+` with alias
680                                    -- `ContactName`
681
682                                    -- 2. concatenation of
683                                    -- `VendorContactFName`,
684                                    -- a single space and
685                                    -- `VendorContactLName`,
686                                    -- the `@` sign, followed by
687                                    -- `VendorName` (value #6)
688
689                                    -- 2.01. getting value #5
690                                    -- 2.02. getting value #4
691                                    -- 2.03. getting value #3
692                                    -- 2.04. getting value #2
693                                    -- 2.05. getting value #1
694
695                                    -- 1. concatenation of
696                                    -- `FirstName`, a single
697                                    -- space and `LastName`
698                                    -- using `+` with alias
699                                    -- `ContactName`
700
701                                    -- 2. concatenation of
702                                    -- `VendorContactFName`,
703                                    -- a single space and
704                                    -- `VendorContactLName`,
705                                    -- the `@` sign, followed by
706                                    -- `VendorName` (value #6)
707
708                                    -- 2.01. getting value #5
709                                    -- 2.02. getting value #4
710                                    -- 2.03. getting value #3
711                                    -- 2.04. getting value #2
712                                    -- 2.05. getting value #1
713
714                                    -- 1. concatenation of
715                                    -- `FirstName`, a single
716                                    -- space and `LastName`
717                                    -- using `+` with alias
718                                    -- `ContactName`
719
720                                    -- 2. concatenation of
721                                    -- `VendorContactFName`,
722                                    -- a single space and
723                                    -- `VendorContactLName`,
724                                    -- the `@` sign, followed by
725                                    -- `VendorName` (value #6)
726
727                                    -- 2.01. getting value #5
728                                    -- 2.02. getting value #4
729                                    -- 2.03. getting value #3
730                                    -- 2.04. getting value #2
731                                    -- 2.05. getting value #1
732
733                                    -- 1. concatenation of
734                                    -- `FirstName`, a single
735                                    -- space and `LastName`
736                                    -- using `+` with alias
737                                    -- `ContactName`
738
739                                    -- 2. concatenation of
740                                    -- `VendorContactFName`,
741                                    -- a single space and
742                                    -- `VendorContactLName`,
743                                    -- the `@` sign, followed by
744                                    -- `VendorName` (value #6)
745
746                                    -- 2.01. getting value #5
747                                    -- 2.02. getting value #4
748                                    -- 2.03. getting value #3
749                                    -- 2.04. getting value #2
750                                    -- 2.05. getting value #1
751
752                                    -- 1. concatenation of
753                                    -- `FirstName`, a single
754                                    -- space and `LastName`
755                                    -- using `+` with alias
756                                    -- `ContactName`
757
758                                    -- 2. concatenation of
759                                    -- `VendorContactFName`,
760                                    -- a single space and
761                                    -- `VendorContactLName`,
762                                    -- the `@` sign, followed by
763                                    -- `VendorName` (value #6)
764
765                                    -- 2.01. getting value #5
766                                    -- 2.02. getting value #4
767                                    -- 2.03. getting value #3
768                                    -- 2.04. getting value #2
769                                    -- 2.05. getting value #1
770
771                                    -- 1. concatenation of
772                                    -- `FirstName`, a single
773                                    -- space and `LastName`
774                                    -- using `+` with alias
775                                    -- `ContactName`
776
777                                    -- 2. concatenation of
778                                    -- `VendorContactFName`,
779                                    -- a single space and
780                                    -- `VendorContactLName`,
781                                    -- the `@` sign, followed by
782                                    -- `VendorName` (value #6)
783
784                                    -- 2.01. getting value #5
785                                    -- 2.02. getting value #4
786                                    -- 2.03. getting value #3
787                                    -- 2.04. getting value #2
788                                    -- 2.05. getting value #1
789
790                                    -- 1. concatenation of
791                                    -- `FirstName`, a single
792                                    -- space and `LastName`
793                                    -- using `+` with alias
794                                    -- `ContactName`
795
796                                    -- 2. concatenation of
797                                    -- `VendorContactFName`,
798                                    -- a single space and
799                                    -- `VendorContactLName`,
800                                    -- the `@` sign, followed by
801                                    -- `VendorName` (value #6)
802
803                                    -- 2.01. getting value #5
804                                    -- 2.02. getting value #4
805                                    -- 2.03. getting value #3
806                                    -- 2.04. getting value #2
807                                    -- 2.05. getting value #1
808
809                                    -- 1. concatenation of
810                                    -- `FirstName`, a single
811                                    -- space and `LastName`
812                                    -- using `+` with alias
813                                    -- `ContactName`
814
815                                    -- 2. concatenation of
816                                    -- `VendorContactFName`,
817                                    -- a single space and
818                                    -- `VendorContactLName`,
819                                    -- the `@` sign, followed by
820                                    -- `VendorName` (value #6)
821
822                                    -- 2.01. getting value #5
823                                    -- 2.02. getting value #4
824                                    -- 2.03. getting value #3
825                                    -- 2.04. getting value #2
826                                    -- 2.05. getting value #1
827
828                                    -- 1. concatenation of
829                                    -- `FirstName`, a single
830                                    -- space and `LastName`
831                                    -- using `+` with alias
832                                    -- `ContactName`
833
834                                    -- 2. concatenation of
835                                    -- `VendorContactFName`,
836                                    -- a single space and
837                                    -- `VendorContactLName`,
838                                    -- the `@` sign, followed by
839                                    -- `VendorName` (value #6)
840
841                                    -- 2.01. getting value #5
842                                    -- 2.02. getting value #4
843                                    -- 2.03. getting value #3
844                                    -- 2.04. getting value #2
845                                    -- 2.05. getting value #1
846
847                                    -- 1. concatenation of
848                                    -- `FirstName`, a single
849                                    -- space and `LastName`
850                                    -- using `+` with alias
851                                    -- `ContactName`
852
853                                    -- 2. concatenation of
854                                    -- `VendorContactFName`,
855                                    -- a single space and
856                                    -- `VendorContactLName`,
857                                    -- the `@` sign, followed by
858                                    -- `VendorName` (value #6)
859
860                                    -- 2.01. getting value #5
861                                    -- 2.02. getting value #4
862                                    -- 2.03. getting value #3
863                                    -- 2.04. getting value #2
864                                    -- 2.05. getting value #1
865
866                                    -- 1. concatenation of
867                                    -- `FirstName`, a single
868                                    -- space and `LastName`
869                                    -- using `+` with alias
870                                    -- `ContactName`
871
872                                    -- 2. concatenation of
873                                    -- `VendorContactFName`,
874                                    -- a single space and
875                                    -- `VendorContactLName`,
876                                    -- the `@` sign, followed by
877                                    -- `VendorName` (value #6)
878
879                                    -- 2.01. getting value #5
880                                    -- 2.02. getting value #4
881                                    -- 2.03. getting value #3
882                                    -- 2.04. getting value #2
883                                    -- 2.05. getting value #1
884
885                                    -- 1. concatenation of
886                                    -- `FirstName`, a single
887                                    -- space and `LastName`
888                                    -- using `+` with alias
889                                    -- `ContactName`
890
891                                    -- 2. concatenation of
892                                    -- `VendorContactFName`,
893                                    -- a single space and
894                                    -- `VendorContactLName`,
895                                    -- the `@` sign, followed by
896                                    -- `VendorName` (value #6)
897
898                                    -- 2.01. getting value #5
899                                    -- 2.02. getting value #4
900                                    -- 2.03. getting value #3
901                                    -- 2.04. getting value #2
902                                    -- 2.05. getting value #1
903
904                                    -- 1. concatenation of
905                                    -- `FirstName`, a single
906                                    -- space and `LastName`
907                                    -- using `+` with alias
908                                    -- `ContactName`
909
910                                    -- 2. concatenation of
911                                    -- `VendorContactFName`,
912                                    -- a single space and
913                                    -- `VendorContactLName`,
914                                    -- the `@` sign, followed by
915                                    -- `VendorName` (value #6)
916
917                                    -- 2.01. getting value #5
918                                    -- 2.02. getting value #4
919                                    -- 2.03. getting value #3
920                                    -- 2.04. getting value #2
921                                    -- 2.05. getting value #1
922
923                                    -- 1. concatenation of
924                                    -- `FirstName`, a single
925                                    -- space and `LastName`
926                                    -- using `+` with alias
927                                    -- `ContactName`
928
929                                    -- 2. concatenation of
930                                    -- `VendorContactFName`,
931                                    -- a single space and
932                                    -- `VendorContactLName`,
933                                    -- the `@` sign, followed by
934                                    -- `VendorName` (value #6)
935
936                                    -- 2.01. getting value #5
937                                    -- 2.02. getting value #4
938                                    -- 2.03. getting value #3
939                                    -- 2.04. getting value #2
940                                    -- 2.05. getting value #1
941
942                                    -- 1. concatenation of
943                                    -- `FirstName`, a single
944                                    -- space and `LastName`
945                                    -- using `+` with alias
946                                    -- `ContactName`
947
948                                    -- 2. concatenation of
949                                    -- `VendorContactFName`,
950                                    -- a single space and
951                                    -- `VendorContactLName`,
952                                    -- the `@` sign, followed by
953                                    -- `VendorName` (value #6)
954
955                                    -- 2.01. getting value #5
956                                    -- 2.02. getting value #4
957                                    -- 2.03. getting value #3
958                                    -- 2.04. getting value #2
959                                    -- 2.05. getting value #1
960
961                                    -- 1. concatenation of
962                                    -- `FirstName`, a single
963                                    -- space and `LastName`
964                                    -- using `+` with alias
965                                    -- `ContactName`
966
967                                    -- 2. concatenation of
968                                    -- `VendorContactFName`,
969                                    -- a single space and
970                                    -- `VendorContactLName`,
971                                    -- the `@` sign, followed by
972                                    -- `VendorName` (value #6)
973
974                                    -- 2.01. getting value #5
975                                    -- 2.02. getting value #4
976                                    -- 2.03. getting value #3
977                                    -- 2.04. getting value #2
978                                    -- 2.05. getting value #1
979
980                                    -- 1. concatenation of
981                                    -- `FirstName`, a single
982                                    -- space and `LastName`
983                                    -- using `+` with alias
984                                    -- `ContactName`
985
986                                    -- 2. concatenation of
987                                    -- `VendorContactFName`,
988                                    -- a single space and
989                                    -- `VendorContactLName`,
990                                    -- the `@` sign, followed by
991                                    -- `VendorName` (value #6)
992
993                                    -- 2.01. getting value #5
994                                    -- 2.02. getting value #4
995                                    -- 2.03. getting value #3
996                                    -- 2.04. getting value #2
997                                    -- 2.05. getting value #1
998
999                                    -- 1. concatenation of
1000                                   -- `FirstName`, a single
1001                                   -- space and `LastName`
1002                                   -- using `+` with alias
1003                                   -- `ContactName`
1004
1005                                   -- 2. concatenation of
1006                                   -- `VendorContactFName`,
1007                                   -- a single space and
1008                                   -- `VendorContactLName`,
1009                                   -- the `@` sign, followed by
1010                                   -- `VendorName` (value #6)
1011
1012                                   -- 2.01. getting value #5
1013                                   -- 2.02. getting value #4
1014                                   -- 2.03. getting value #3
1015                                   -- 2.04. getting value #2
1016                                   -- 2.05. getting value #1
1017
1018                                   -- 1. concatenation of
1019                                   -- `FirstName`, a single
1020                                   -- space and `LastName`
1021                                   -- using `+` with alias
1022                                   -- `ContactName`
1023
1024                                   -- 2. concatenation of
1025                                   -- `VendorContactFName`,
1026                                   -- a single space and
1027                                   -- `VendorContactLName`,
1028                                   -- the `@` sign, followed by
1029                                   -- `VendorName` (value #6)
1030
1031                                   -- 2.01. getting value #5
1032                                   -- 2.02. getting value #4
1033                                   -- 2.03. getting value #3
1034                                   -- 2.04. getting value #2
1035                                   -- 2.05. getting value #1
1036
1037                                   -- 1. concatenation of
1038                                   -- `FirstName`, a single
1039                                   -- space and `LastName`
1040                                   -- using `+` with alias
1041                                   -- `ContactName`
1042
1043                                   --
```

```

70     AP1.ContactUpdates.FirstName +
71     ' ' +
72     AP1.ContactUpdates.LastName
73     AS ContactName,
74
75     LEFT (AP1.ContactUpdates.FirstName, 1) +
76     AP1.ContactUpdates.LastName +
77     '@' +
78
79
80
81     REPLACE (
82     REPLACE (
83     REPLACE (
84     REPLACE (
85     REPLACE (
86     REPLACE (AP1.Vendors.VendorName, ' ', '')
87
88     , ' ', '')
89     , ' ', '')
90     , '/' , '')
91     , '&' , '')
92     , ' ', '') +
93
94
95
96
97     '.foo'
98     AS VendorContactEmail
99
100 FROM AP1.Vendors
101 LEFT JOIN AP1.ContactUpdates
102 ON AP1.Vendors.VendorID = AP1.ContactUpdates.VendorID;
103
104
105 /* *****
106     As with previous example, we can use an alias for each table, which
107     in this case, allows us to present neater code.
108
109     `v` for `AP1.Vendors`
110     `c` for `AP1.ContactUpdates`
111 ***** */
112
113 SELECT v.VendorID,
114        v.VendorName,
115        v.VendorAddress1,
116        v.VendorAddress2,
117        v.VendorCity,
118        v.VendorState,
119        v.VendorZipCode,
120        v.VendorPhone,
121        v.VendorContactFName +
122        ' ' +
123        v.VendorContactLName AS ContactName,
124        LEFT (v.VendorContactFName, 1) +
125        v.VendorContactLName + '@' +
126        REPLACE (
127        REPLACE (
128        REPLACE (
129        REPLACE (
130        REPLACE (
131        REPLACE (v.VendorName, ' ', '')
132
133        , ' ', '')
134        , '/' , '')
135        , '&' , '')
136        , ' ', '')
137        + '.foo' AS VendorContactEmail,
138        v.DefaultTermsID,

```

```

139 v.DefaultAccountNo,
140 -- c.VendorID AS Expr1,
141 c.FirstName +
142 ' '+
143 c.LastName AS ContactName,
144 LEFT(c.FirstName, 1) +
145 c.LastName +
146 '@' +
147 REPLACE (
148     REPLACE (
149         REPLACE (
150             REPLACE (
151                 REPLACE (
152                     REPLACE (v.VendorName, ' ', '')
153                     , '' , ''')
154                 , '' , ''')
155             , '/' , ''')
156             , '&' , ''')
157             , '' , ''')
158 + '.foo' AS VendorContactEmail
159 FROM AP1.Vendors AS v
160 LEFT JOIN AP1.ContactUpdates AS c
161 ON v.VendorID = c.VendorID;
162
163
164 /* *****
165     Instead of using a plus sign (`+`), we can use `CONCAT()` since
166     adding a value and NULL returns NULL. In other words, we lose data,
167     which would be logical error (not syntax error).
168     ***** */
169
170 SELECT AP1.Vendors.VendorID,
171        AP1.Vendors.VendorName,
172        AP1.Vendors.VendorAddress1,
173        AP1.Vendors.VendorAddress2,
174        AP1.Vendors.VendorCity,
175        AP1.Vendors.VendorState,
176        AP1.Vendors.VendorZipCode,
177        AP1.Vendors.VendorPhone,
178        CONCAT (
179            AP1.Vendors.VendorContactFName,
180            ' ',
181            AP1.Vendors.VendorContactLName
182        ) AS ContactName,
183        CONCAT (
184            LEFT(AP1.Vendors.VendorContactFName, 1),
185            AP1.Vendors.VendorContactLName,
186            '@',
187
188
189            REPLACE (
190                REPLACE (
191                    REPLACE (
192                        REPLACE (
193                            REPLACE (
194                                REPLACE (AP1.Vendors.VendorName, ' ', '')
195                                , '' , ''')
196                            , '' , ''')
197                        , '/' , ''')
198                        , '&' , ''')
199                        , '' , ''')
200
201            , '.foo'
202        ) AS VendorContactEmail,
203
204
205
206
207

```

```

-- 1. concatenation of
--     `FirstName`, a single
--     space and `LastName` with
--     alias `ContactName`
--
-- 2. concatenation of
--     `VendorContactFName`,
--     a single space and
--     `VendorContactLName`,
--     the `@` sign, followed by
--     `VendorName` (value #6)
--     2.01. getting value #5
--     2.02. getting value #4
--     2.03. getting value #3
--     2.04. getting value #2
--     2.05. getting value #1
--     2.06. generating value #1
--     2.07. generating value #2
--     2.08. generating value #3
--     2.09. generating value #4
--     2.10. generating value #5
--     2.11. generating value #6
--     where `''`
--           represents a single
--           quote (``) used an
--           escape character
--           and `.foo` to complete
--           email with alias
--           `VendorContactEmail`

```

```

208     AP1.Vendors.DefaultTermsID,
209     AP1.Vendors.DefaultAccountNo,
210     -- AP1.ContactUpdates.VendorID AS Expr1,
211     CONCAT (
212         AP1.ContactUpdates.FirstName,
213         ' ',
214         AP1.ContactUpdates.LastName
215     ) AS ContactName,
216     CONCAT (
217         LEFT(AP1.ContactUpdates.FirstName, 1),
218         AP1.ContactUpdates.LastName,
219         '@',
220
221
222     REPLACE (
223         REPLACE (
224             REPLACE (
225                 REPLACE (
226                     REPLACE (AP1.Vendors.VendorName, ' ', '')
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276

```

-- 3. concatenation of
-- `VendorContactFName`, a
-- single space and
-- `LastName` with alias
-- `VendorContactLName`
-- 4. concatenation of
-- `VendorContactFName`,
-- a single space and
-- `VendorContactLName`,
-- the `@` sign, followed by
-- `VendorName` (value #6)
-- 4.01. getting value #5
-- 4.02. getting value #4
-- 4.03. getting value #3
-- 4.04. getting value #2
-- 4.05. getting value #1
-- 4.06. generating value #1
-- 4.07. generating value #2
-- 4.08. generating value #3
-- 4.09. generating value #4
-- 4.10. generating value #5
-- 4.11. generating value #6
-- where `''`
-- represents a single
-- quote (`'`) used an
-- escape character
-- and `.foo` to complete
-- email with alias
-- `VendorContactEmail`

```

FROM AP1.Vendors
LEFT JOIN AP1.ContactUpdates
ON AP1.Vendors.VendorID = AP1.ContactUpdates.VendorID;

/* *****
Using `CONCAT()` also returns a logical error (not syntax error)
since the concatenation to make the second email returns values like
`@USPostalService.foo` since there is no corresponding `FirstName`
and `LastName`. We could use a CASE clause.
***** */

SELECT AP1.Vendors.VendorID,
AP1.Vendors.VendorName,
AP1.Vendors.VendorAddress1,
AP1.Vendors.VendorAddress2,
AP1.Vendors.VendorCity,
AP1.Vendors.VendorState,
AP1.Vendors.VendorZipCode,
AP1.Vendors.VendorPhone,
CONCAT (
    AP1.Vendors.VendorContactFName,
    ' ',
    AP1.Vendors.VendorContactLName
) AS ContactName,
CONCAT (
    LEFT(AP1.Vendors.VendorContactFName, 1),
    AP1.Vendors.VendorContactLName,
    '@',

```

-- 1. concatenation of
-- `FirstName`, a single
-- space and `LastName` with
-- alias `ContactName`
-- 2. concatenation of
-- `VendorContactFName`,
-- a single space and
-- `VendorContactLName`,
-- the `@` sign, followed by
-- `VendorName` (value #6)
-- 2.01. getting value #5
-- 2.02. getting value #4
-- 2.03. getting value #3
-- 2.04. getting value #2
-- 2.05. getting value #1

```

REPLACE (
    REPLACE (
        REPLACE (
            REPLACE (

```

```

277      REPLACE (AP1.Vendors.VendorName, ' ', '')
278      -- 2.06. generating value #1
279      , ' ', ' ')
280      -- 2.07. generating value #2
281      , ' ', ' ')
282      -- 2.08. generating value #3
283      , ' ', ' ')
284      -- 2.09. generating value #4
285      , ' ', ' ')
286      -- 2.10. generating value #5
287      , ' ', ' ')
288      -- 2.11. generating value #6
289      -- where ` `
290      -- represents a single
291      -- quote (` `) used an
292      -- escape character
293      -- and `.foo` to complete
294      -- email with alias
295      -- `VendorContactEmail`
296      ) AS VendorContactEmail,
297      AP1.Vendors.DefaultTermsID,
298      AP1.Vendors.DefaultAccountNo,
299      -- AP1.ContactUpdates.VendorID AS Expr1,
300      CONCAT (
301      AP1.ContactUpdates.FirstName,
302      ' ',
303      AP1.ContactUpdates.LastName
304      ) AS ContactName,
305      CASE
306      WHEN (AP1.ContactUpdates.FirstName <> ''
307      OR AP1.ContactUpdates.FirstName <> ' '
308      OR AP1.ContactUpdates.FirstName IS NOT NULL)
309      -- 3. concatenation of
310      -- `VendorContactFName`, a
311      -- single space and
312      -- `LastName` with alias
313      -- `VendorContactLName`
314      -- 4. checking if
315      -- `ContactUpdates` is not
316      -- an empty string or not
317      -- equal to a space or
318      -- `IS NOT NULL` (in other
319      -- words, not no-value; must
320      -- have a value) using
321      -- parenthesis to make a
322      -- block of three conditions
323      AND (AP1.ContactUpdates.LastName <> ''
324      OR AP1.ContactUpdates.LastName <> ' '
325      OR AP1.ContactUpdates.LastName IS NOT NULL)
326      -- not equal to a space or
327      -- `IS NOT NULL` (in other
328      -- words, not no-value; must
329      -- have a value) using
330      -- parenthesis to make a
331      -- block of three conditions
332      THEN CONCAT (
333      LEFT (AP1.ContactUpdates.FirstName, 1),
334      AP1.ContactUpdates.LastName,
335      '@',
336      REPLACE (
337      REPLACE (
338      REPLACE (
339      REPLACE (
340      REPLACE (
341      REPLACE (AP1.Vendors.VendorName, ' ', ' ')
342      -- 4.01. getting value #5
343      -- 4.02. getting value #4
344      -- 4.03. getting value #3
345      -- 4.04. getting value #2
346      -- 4.05. getting value #1
347      -- 4.06. generating value #1
348      -- 4.07. generating value #2
349      -- 4.08. generating value #3
350      -- 4.09. generating value #4
351      -- 4.10. generating value #5
352      -- 4.11. generating value #6
353      -- where ` `
354      -- represents a single
355      -- quote (` `) used an
356      -- escape character
357      -- and `.foo` to complete
358      )
359      ELSE ''
360      END

```

```

346         AS VendorContactEmail                -- make email with alias
347                                             -- `VendorContactEmail`
348 FROM AP1.Vendors
349 LEFT JOIN AP1.ContactUpdates
350     ON AP1.Vendors.VendorID = AP1.ContactUpdates.VendorID;
351
352
353 /* *****
354     As with previous examples, we can use an alias for each table, which
355     in this case, allows us to present neater code.
356
357         `v` for `AP1.Vendors`
358         `c` for `AP1.ContactUpdates`
359 ***** */
360
361 SELECT v.VendorID,
362        v.VendorName,
363        v.VendorAddress1,
364        v.VendorAddress2,
365        v.VendorCity,
366        v.VendorState,
367        v.VendorZipCode,
368        v.VendorPhone,
369        CONCAT (
370            v.VendorContactFName,
371            ' ',
372            v.VendorContactLName
373        ) AS ContactName,
374        CONCAT (
375            LEFT(v.VendorContactFName, 1),
376            v.VendorContactLName,
377            '@',
378            REPLACE (
379                REPLACE (
380                    REPLACE (
381                        REPLACE (
382                            REPLACE (v.VendorName, ' ', '')
383                            , '.' , '' )
384                            , ',' , '' )
385                            , '/' , '' )
386                            , '&' , '' )
387                            , ' ' , '' ) ,
388            '.foo'
389        ) AS VendorContactEmail,
390        v.DefaultTermsID,
391        v.DefaultAccountNo,
392        -- c.VendorID AS Expr1,
393        CONCAT (
394            c.FirstName,
395            ' ',
396            c.LastName
397        ) AS ContactName,
398        CASE
399            WHEN (
400                c.FirstName <> ''
401                OR c.FirstName <> ' '
402                OR c.FirstName IS NOT NULL
403            )
404            AND (
405                c.LastName <> ''
406                OR c.LastName <> ' '
407                OR c.LastName IS NOT NULL
408            )
409            THEN CONCAT (
410                LEFT(c.FirstName, 1),
411                c.LastName,
412                REPLACE (
413                    REPLACE (

```


553
554 /* *****
555 3. Before you continue learning about SQL
556 (<https://searchsqlserver.techtarget.com/definition/SQL>) syntax
557 (<https://whatis.techtarget.com/definition/syntax>), we should cover some
558 important theory, which you will need whether you need to learn SQL to run
559 queries at work and/or you decide to become a database administrator (DBA).
560
561 3.01. SQL (Structured Query Language) is a standardized programming
562 language used for managing relational databases and performing
563 various operations on the data in them. Initially created in the
564 1970s, SQL is regularly used by database administrators, as well as
565 by developers writing data integration scripts and data analysts
566 looking to set up and run analytical queries.
567 <https://searchsqlserver.techtarget.com/definition/SQL>
568
569 3.02. ISO/IEC 9075-1:2016 [SQL:2016] describes the conceptual framework
570 used in other parts of ISO/IEC 9075 to specify the grammar of SQL and
571 the result of processing statements in that language by an
572 SQL-implementation.
573 ISO/IEC 9075-1:2016 also defines terms and notation used in the other
574 parts of ISO/IEC 9075.
575 <https://www.iso.org/standard/63555.html>
576
577 3.03. T-SQL (Transact-SQL) is a set of programming extensions from Sybase
578 and Microsoft that add several features to the Structured Query
579 Language (SQL), including transaction control, exception and error
580 handling, row processing and declared variables.
581 <https://searchsqlserver.techtarget.com/definition/T-SQL>
582
583 3.04. A relational database is a set of tables containing data fitted into
584 predefined categories. Each table (which is sometimes called a
585 relation) contains one or more data categories in columns. Each row
586 contains a unique instance of data for the categories defined by the
587 columns.
588 <http://searchsqlserver.techtarget.com/definition/relational-database>
589
590 3.05. Microsoft SQL Server is a relational database management system, or
591 RDBMS, that supports a wide variety of transaction processing,
592 business intelligence and analytics applications in corporate IT
593 environments. It's one of the three market-leading database
594 technologies, along with Oracle Database and IBM's DB2.
595 Like other RDBMS software, Microsoft SQL Server is built on top of
596 SQL, a standardized programming language that database administrators
597 (DBAs) and other IT professionals use to manage databases and query
598 the data they contain. SQL Server is tied to Transact-SQL (T-SQL),
599 an implementation of SQL from Microsoft that adds a set of
600 proprietary programming extensions to the standard language. The
601 original SQL Server code was developed in the 1980s by the former
602 Sybase Inc., which is now owned by SAP. Sybase initially built the
603 software to run on Unix systems and minicomputer platforms. It,
604 Microsoft and Ashton-Tate Corp., then the leading vendor of PC
605 databases, teamed up to produce the first version of what became
606 Microsoft SQL Server, designed for the OS/2 operating system and
607 released in 1989.
608 <https://searchsqlserver.techtarget.com/definition/SQL-Server>
609
610 3.06. Another form of flat file is one in which table data is gathered in
611 lines of ASCII text with the value from each table cell separated by
612 a comma and each row represented with a new line. This type of flat
613 file is also known as a comma-separated values file (CSV) file.
614 <http://searchsqlserver.techtarget.com/definition/flat-file>
615
616 3.07. A hierarchical database is a design that uses a one-to-many
617 relationship for data elements. Hierarchical database models use a
618 tree structure that links a number of disparate elements to one
619 `owner,` or `parent,` primary record.
620 <https://www.techopedia.com/definition/19782/hierarchical-database>
621

622 3.08. Data Manipulation Language (DML) is the ``vocabulary used to retrieve
623 and work with data... to add, modify, query, or remove data``
624 (<https://msdn.microsoft.com/en-us/library/ff848766.aspx>).
625

626 SELECT to retrieve records from one or more tables
627 <https://techonthenet.com/sql/select.php>
628
629 INSERT to insert a one or more records into a table
630 <https://techonthenet.com/sql/insert.php>
631
632 UPDATE to update existing records in the tables
633 <https://techonthenet.com/sql/update.php>
634
635 DELETE to delete a one or more records from a table
636 <https://techonthenet.com/sql/delete.php>
637
638 MERGE to insert, update, or delete operations on a target table
639 based on the results of a join with a source table
640 <https://msdn.microsoft.com/en-us/library/bb510625.aspx>
641

642 3.09. Data Definition Language (DDL) is the ``vocabulary used to define
643 data structures... to create, alter, or drop data structures``
644 (<https://msdn.microsoft.com/en-us/library/ff848799.aspx>).
645

646 USE to select any existing database in SQL schema [or output
647 from another query]
648 <http://tutorialspoint.com/sql/sql-select-database.htm>
649
650 CREATE to create and define a table [or other database object]
651 https://techonthenet.com/sql/tables/create_table.php
652
653 ALTER to add a column, modify a column, drop a column, rename a
654 column or rename a table [or other database object]
655 https://techonthenet.com/sql/tables/alter_table.php
656
657 DROP to remove or delete a table [or other database object]
658 https://techonthenet.com/sql/tables/drop_table.php
659
660 TRUNCATE to remove all records from a table
661 <https://techonthenet.com/sql/truncate.php>
662
663 DELETE to delete a one or more records from a table
664 <https://techonthenet.com/sql/delete.php>
665

666 3.10. Note that some of these statements can do more than what is covered
667 in these notes for our first sessions.
668

669 For example, the `CREATE` statement is also used to create other
670 database objects as well as access management, but we will not cover
671 these other statements yet. Refer to
672 <https://msdn.microsoft.com/en-us/library/cc879262.aspx> for more
673 information on the `CREATE` statement.
674

675 On a personal note, when looking for information and/or explanation
676 on how to use Microsoft technologies, in this case SQL Server, go to
677 <https://techonthenet.com/> or <http://tutorialspoint.com/> as
678 <https://msdn.microsoft.com/> and other Microsoft websites often seem
679 to be written for advanced users.
680

681 We will use DML and DDL in detail later in the course.
682

683 4. There are several data types
684 (<https://msdn.microsoft.com/en-us/library/ms187752.aspx>) that you need to
685 know if you are interested in taking the certification exam for Database
686 Fundamentals. In everyday use, these are the most often used data types in
687 T-SQL (<http://searchsqlserver.techtarget.com/definition/T-SQL>) -- the
688 version of SQL (<http://searchsqlserver.techtarget.com/definition/SQL>) used
689 in SQL Server (<http://searchsqlserver.techtarget.com/definition/SQL-Server>)
690 -- are the following.

691
692 INT -2^31 (-2,147,483,648) to 2^31-1 (2,147,483,647)
693 <https://technet.microsoft.com/en-us/library/ms187745.aspx>
694
695 DECIMAL fixed precision and scale numbers, 10^38+1 through 10^38-1
696 <https://msdn.microsoft.com/en-us/library/ms187746.aspx>
697 * instead of DOUBLE or FLOAT, indicating the whole value
698 followed by the number of decimals where pi(1,10) can
699 hold 3.1415926536, but not 3.14159265359 for eleven (11)
700 decimal spaces
701
702 VARCHAR(n) 2^31-1 bytes (2 GB); variable-length, ASCII
703

704 (http://whatis.techtarget.com/definition/ASCII-American-Standard-Code
705 -for-Information-Interchange)
706 string data
707 <https://technet.microsoft.com/en-us/library/ms176089.aspx>
708 not to be confused with NVARCHAR(n) -- variable-length,
709 2^31-1 bytes (2 GB), Unicode
710 (http://whatis.techtarget.com/definition/Unicode) string
711 data, not part of most relational database management
712 systems (RDBMS)
713 <https://technet.microsoft.com/en-us/library/ms186939.aspx>
714
715 DATE date
716 <https://technet.microsoft.com/en-us/library/bb630352.aspx>
717
718 TIME time
719 <https://technet.microsoft.com/en-us/library/bb677243.aspx>
720
721 DATETIME defines a date that is combined with a time of day with
722 fractional seconds that is based on a 24-hour clock
723 <https://technet.microsoft.com/en-us/library/ms187819.aspx>
724
725 MONEY money, not part of most relational database management
726 systems (RDBMS)
727 <https://technet.microsoft.com/en-us/library/ms179882.aspx>

4.01. Conversion may only take place between data similar types.

CONVERSION INPUT	CONVERSION OUTPUT
INT to DECIMAL	no loss; decimal spaces added (.00)
DECIMAL to INT	possible loss of decimal spaces; truncated, value not rounded up or down
DECIMAL to MONEY	truncated and rounded to four decimal spaces for mathematical calculations (.0000 to .9999); two decimal spaces shown for cents (.00 to .99)
DATETIME to DATE	date only; time dropped
DATETIME to TIME	time only; date dropped
DATE to DATETIME	date with default value of '00:00.00.000'
TIME to DATETIME	time with default value of '1900/01/01'
INT DECIMAL DATETIME to VARCHAR	converted to text; no longer numeric data and cannot be used in mathematical calculations

758	DATE	NVARCHAR	
759	TIME		
760	+-----+-----+		
761		INT	straight conversion to proper
762		DECIMAL	data type as long as the string
763	VARCHAR	to DATETIME	field only has numbers and
764	NVARCHAR	DATE	structure is correct (for
765		TIME	example, text with value of
766			`2019/03/11` to DATE); no
767			conversion if the string has
768			letters or special characters
769	+-----+-----+		
770	VARCHAR	to NVARCHAR	straight conversion; no data
771			loss
772	+-----+-----+		
773	NVARCHAR	to VARCHAR	straight conversion if string is
774			encoded as ACIII or UTF-8;
775			possible data loss if string is
776			encoded as Unicode or no
777			conversion at all
778	+-----+-----+		

780 4.02. Refer to <https://technet.microsoft.com/en-us/library/ms187912.aspx>
781 for information on approximate numeric data types -- FLOAT and REAL.
782 If you are considering taking the certification, you should know the
783 concept below and why Microsoft recommends not using approximate
784 numeric data types.

785
786 ``The float and real data types are known as approximate
787 data types. The behavior of float and real follows the
788 IEEE 754 specification on approximate numeric data types.
789 Approximate numeric data types do not store the exact
790 values specified for many numbers; they store an extremely
791 close approximation of the value. For many applications,
792 the tiny difference between the specified value and the
793 stored approximation is not noticeable. At times, though,
794 the difference becomes noticeable. Because of the
795 approximate nature of the float and real data types, do not
796 use these data types when exact numeric behavior is
797 required, such as in financial applications, in operations
798 involving rounding, or in equality checks. Instead, use
799 the integer, decimal, money, or smallmoney data types.
800 Avoid using float or real columns in WHERE clause search
801 conditions, especially the = and <> operators. It is best
802 to limit float and real columns to > or < comparisons. The
803 IEEE 754 specification provides four rounding modes: round
804 to nearest, round up, round down, and round to zero.
805 Microsoft SQL Server uses round up. All are accurate to
806 the guaranteed precision but can result in slightly
807 different floating-point values. Because the binary
808 representation of a floating-point number may use one of
809 many legal rounding schemes, it is impossible to reliably
810 quantify a floating-point value.``

811 <https://technet.microsoft.com/en-us/library/ms187912.aspx>

812
813 Note that FLOAT is commonly used in other relational database
814 management systems (RDBMS) like Oracle (<http://oracle.com/>) and in
815 most programming languages including those distributed by Microsoft.
816

817 5. As we start, we keep in mind that the most basic structure of a `SELECT`
818 statement (<https://techonthenet.com/sql/select.php>) is the following.

```
819
820     SELECT field1, field2...
821     FROM   table1
```

822
823 From the previous structure, you can add clauses in the following order.
824 If you organize the clauses any other order, the query will not work.

```
825
826     SELECT table1.field1,           -- 1. calling columns/fields
```

```

827         table1.field2,          -- (data)
828         ...
829         table2.field1,
830         table2.field2,
831         ...
832         table3.field1,
833         table3.field2,
834         ...
835
836     FROM table1                  -- 2. where to find data
837                                -- (tables/views)
838     INNER|LEFT|RIGHT JOIN table2
839         ON table1.shared_field1 = table2.shared_field1
840         AND table1.shared_field2 = table2.shared_field2
841         ...
842     INNER|LEFT|RIGHT JOIN table3
843         ON table1.shared_field1 = table3.shared_field1
844         AND table1.shared_field2 = table3.shared_field2
845         ...
846
847     WHERE condition1             -- 3. filtering output, what
848         AND|OR condition2        -- rows/records you want to
849         AND|OR condition3        -- retrieve
850         ...
851
852     GROUP BY table1.field1,      -- 4. grouping fields not in an
853            table1.field2,        -- aggregate function
854            ...
855            table2.field1,
856            table2.field2,
857            ...
858            table3.field1,
859            table3.field2,
860            ...
861
862     ORDER BY                     -- 5. organizing rows/records
863            table1.field1 ASC|DESC, -- (output) in ascending
864            table1.field2 ASC|DESC, -- (`ASC`) or descending
865            ...                    -- (`DESC`) order
866            table2.field1 ASC|DESC,
867            table2.field2 ASC|DESC,
868            ...
869            table3.field1 ASC|DESC,
870            table3.field2 ASC|DESC,
871            ...
872

```

5.01. In the example below, we retrieve all (`*`) columns from table `AP1.Vendors`.

```

873     /* ***** */
874     SELECT *
875     FROM AP1.Vendors;          -- retrieves all values from
876                                -- table `AP1.Vendors`
877

```

5.02. The only time you can use `SELECT` without `FROM` is when you want the machine to return a value, similar to `PRINT`.

```

878     /* ***** */
879     SELECT 9 * 8;              -- returns integer 72 (a
880                                -- mathematical equation)
881
882     SELECT 'Hello there';     -- returns string `Hello there`
883                                -- (a simple string)
884

```

5.03. As you can see in the examples above, we are not retrieving data from

```

896         any table. You can get the same results using `PRINT`.
897         ***** */
898
899 PRINT 9 * 8;           -- prints integer 72 (a
900                       -- mathematical equation)
901
902 PRINT 'Hello there';  -- prints string `Hello there`
903                       -- (a simple string)
904
905
906 /* *****
907 6. We have covered built-in functions that affect strings.
908
909     CONCAT()           allows you to concatenate strings together
910                       https://techonthenet.com/sql\_server/functions/concat.php
911
912     + (plus)           allows you to concatenate 2 or more strings together
913                       https://techonthenet.com/sql\_server/functions/concat2.php
914
915     LEFT()             allows you to extract a substring from a string, starting
916                       from the left-most character
917                       https://techonthenet.com/sql\_server/functions/left.php
918
919     LEN()              returns the length of the specified string... does not
920                       include trailing space characters at the end the string
921                       when calculating the length
922                       https://techonthenet.com/sql\_server/functions/len.php
923
924     LTRIM()            removes all space characters from the left-hand side of a
925                       string
926                       https://techonthenet.com/sql\_server/functions/ltrim.php
927
928     LOWER()            converts all letters in the specified string to lowercase
929                       https://techonthenet.com/sql\_server/functions/lower.php
930
931     REPLACE()          replaces a sequence of characters in a string with another
932                       set of characters, not case-sensitive
933                       https://techonthenet.com/sql\_server/functions/replace.php
934
935     RIGHT()            allows you to extract a substring from a string, starting
936                       from the right-most character
937                       https://techonthenet.com/sql\_server/functions/right.php
938
939     RTRIM()            removes all space characters from the right-hand side of a
940                       string
941                       https://techonthenet.com/sql\_server/functions/rtrim.php
942
943     SUBSTRING()        allows you to extract a substring from a string
944                       https://techonthenet.com/sql\_server/functions/substring.php
945
946     UPPER()            converts all letters in the specified string to uppercase
947                       https://techonthenet.com/sql\_server/functions/upper.php
948
949     Now we will see functions used with numeric values.
950
951     AVG()              returns the average value of an expression
952                       https://techonthenet.com/sql\_server/functions/avg.php
953
954     CEILING()          returns the smallest integer value that is greater than or
955                       equal to a number
956                       https://techonthenet.com/sql\_server/functions/ceiling.php
957
958     COUNT()            returns the count of an expression
959                       https://techonthenet.com/sql\_server/functions/count.php
960
961     FLOOR()            returns the largest integer value that is equal to or less
962                       than a number
963                       https://techonthenet.com/sql\_server/functions/floor.php
964

```

965 MAX() returns the maximum value of an expression
 966 https://techonthenet.com/sql_server/functions/max.php
 967
 968 MIN() returns the minimum value of an expression
 969 https://techonthenet.com/sql_server/functions/min.php
 970
 971 PRODUCT() returns the product of an expression
 972
 973 <https://www.sqlservercentral.com/articles/t-sql-in-sql-server-2025-the-operator>
 974 <https://learn.microsoft.com/en-us/sql/t-sql/functions/product-aggregate-transact-sql?view=sql-server-ver17>
 975 * brand new T-SQL 2025
 976 RAND() returns a random number or a random number within a range
 977 https://techonthenet.com/sql_server/functions/rand.php
 978
 979 ROUND() returns a number rounded to a certain number of decimal
 980 places
 981 https://techonthenet.com/sql_server/functions/round.php
 982
 983 SUM() returns the summed value of an expression
 984 https://techonthenet.com/sql_server/functions/sum.php
 985

7. In the examples below, we use each one of the numeric functions with the answer for each on the comment on the right.

***** */

```

989
990 SELECT SUM(InvoiceTotal) AS InvoiceTotalSUM,      -- returns 214290.51
991        AVG(InvoiceTotal) AS InvoiceTotalAVG,      -- returns 1879.7413
992        COUNT(InvoiceTotal) AS InvoiceTotalCOUNT, -- returns 114
993        ROUND(InvoiceTotal, 1) AS InvoiceTotalROUND, -- returns 3813.30
994                                           -- 40.20 ...
995        FLOOR(InvoiceTotal) AS InvoiceTotalFLOOR,  -- returns 3813.00
996                                           -- 40.00 ...
997        CEILING(InvoiceTotal) AS InvoiceTotalCEILING, -- returns 3814.00
998                                           -- 41.00 ...
999        MAX(InvoiceTotal) AS InvoiceTotalMAX,      -- returns 37966.19
1000       MIN(InvoiceTotal) AS InvoiceTotalMIN,      -- returns 6.00
1001       RAND(InvoiceTotal) AS InvoiceTotalRAND,    -- returns 0.713591993212924
1002                                           -- 0.713610626184182...
1003       FORMAT(InvoiceTotal, 'c', 'en-us')         -- `c` for currency with
1004         AS InvoiceTotal,                          -- culture `en-us` (English US)
1005                                           -- returns $3,813.33
1006                                           -- $40.20 ...
1007       FORMAT(InvoiceDueDate, 'd', 'en-us')       -- `d` (lower case) for short
1008         AS InvoiceDueDate,                        -- date returning no leading
1009                                           -- zeros with culture `en-us`
1010                                           -- (English US);
1011                                           -- returns 1/8/2012
1012                                           -- 1/10/2012 ...
1013       FORMAT(InvoiceDueDate, 'D', 'en-us')       -- `D` (upper case) for long
1014         AS InvoiceDueDate,                        -- date returning full day of
1015                                           -- the week, full month, no
1016                                           -- leading zeros with culture
1017                                           -- `en-us` (English US);
1018                                           -- returns
1019                                           -- Sunday, January 8, 2012
1020                                           -- Tuesday, January 10, 2012
1021                                           -- ...
1022       FORMAT(InvoiceDueDate, 'MM/dd/yyyy', 'en-us') -- custom date using format
1023         AS InvoiceDueDate                        -- `MM/dd/yyyy` which overrides
1024                                           -- culture `en-us` (English
1025                                           -- US); returns 01/08/2012
1026                                           -- 01/10/2012 ...
1027 FROM AP1.Invoices
1028 GROUP BY InvoiceTotal,
1029          AP1.Invoices.InvoiceDueDate

```

```

1030
1031
1032 /* *****
1033     7.01. When using an aggregate function, we must use `GROUP BY` and list all
1034         columns not in affected by any aggregate function.
1035
1036         In the example below, we retrieve `VendorState` plus the count of
1037         column `VendorState` for each `VendorState` (`COUNT(VendorState)`).
1038
1039         We can use `DISTINCT` to make sure that duplicate values (rows) are
1040         not included in the output of a query.
1041
1042         We can use `ORDER BY` to organize output by a specific column or list
1043         of columns.
1044
1045         The default option for `ORDER BY` is ascending (`ASC`), which can be
1046         omitted (1, 2, 3... a, b, c...).
1047
1048         The opposite option for `ORDER BY` is descending (`DESC`), which must
1049         be used if needed (...3, 2, 1 ...c, b, a).
1050 ***** */
1051
1052 SELECT DISTINCT           -- 1. to avoid duplicates
1053     VendorState,         -- 2. column not in aggregate
1054                         -- function
1055     COUNT(VendorState)  -- 3. column in aggregate
1056                         -- function (calculation)
1057 FROM AP1.Vendors       -- 4. from table `AP1.Vendors`
1058 GROUP BY VendorState  -- 5. must use `GROUP BY` when
1059                         -- using any aggregate
1060                         -- function, listing all
1061                         -- columns not in the
1062                         -- aggregate function
1063 ORDER BY VendorState ASC; -- 6. organizing results by
1064                         -- column `VendorState` in
1065                         -- ascending order
1066
1067
1068 /* *****
1069     7.02. In the example below, we retrieve `VendorID` plus the sum of column
1070         `PaymentTotal` for each `VendorID` (`SUM(PaymentTotal)`).
1071 ***** */
1072
1073 SELECT DISTINCT           -- 1. to avoid duplicates
1074     VendorID,           -- 2. column not in aggregate
1075                         -- function
1076     SUM(PaymentTotal)  -- 3. column in aggregate
1077                         -- function (calculation)
1078 FROM AP1.Invoices     -- 4. from table `AP1.Invoices`
1079 GROUP BY VendorID    -- 5. must use `GROUP BY` when
1080                         -- using any aggregate
1081                         -- function, listing all
1082                         -- columns not in the
1083                         -- aggregate function
1084 ORDER BY VendorID DESC; -- 6. organizing results by
1085                         -- column `VendorID` in
1086                         -- descending order
1087
1088
1089 /* *****
1090     8. In the example below, the query returns all values from the `AP1.Vendors`
1091         table with all related values from table `AP1.Invoices`,
1092         `AP1.InvoiceLineItems` and `AP1.Terms`.
1093
1094     8.01. The relation between related tables `AP1.Invoices`,
1095         `AP1.InvoiceLineItems` and `AP1.Terms` is `INNER JOIN` since the
1096         value (row ID) of one table is referenced in another.
1097
1098     8.02. Dollar amounts are formatted as `c` (currency) with culture `en-us`

```

1099 (English-United States). Dates are formatted as `MM/dd/yyyy` (two
 1100 digits for month and day, four digits for year) and culture `en-us`
 1101 (English-United States). Refer to
 1102 <https://msdn.microsoft.com/en-us/library/hh213506.aspx> for more
 1103 information. Note that formatting a numeric value changes it to an
 1104 alpha-numeric value -- change in data type.
 1105

1106 8.03. To include the average value of `InvoiceTotal` of all records from
 1107 table `AP1.Invoices`, we use a sub-query (also referred to as nested
 1108 query, <http://tutorialspoint.com/sql/sql-sub-queries.htm>). We use
 1109 alias `AvgInvoiceTotal` to refer to this new column.
 1110

```
1111 (
1112     SELECT FORMAT(AVG(AP1.Invoices.InvoiceTotal),'c','en-us')
1113     FROM AP1.Invoices
1114 )
1115     AS AvgInvoiceTotal
```

1116 There are various values for culture (one per language and country
 1117 combination). The following are just a few, probably the most common
 1118 in American businesses. Refer to
 1119
 1120

<http://sql-server-helper.com/sql-server-2012/format-string-function-culture.aspx>
 x
 for a more detailed list of cultures.

CULTURE	LANGUAGE	COUNTRY	RESULT
en-us	English	USA	dollar
en-gb	English	Great Britain	pound
de-de	German	Germany	euro
zh-cn	Simplified Chinese	China	yuan
jp-jp	Japanese	Japan	yen

1137 Refer to <https://www.iso.org/iso-4217-currency-codes.html> for more
 1138 information on currency codes (ISO 4217).
 1139
 1140

1141 When formatting DATETIME fields, you can use any of the formats below
 1142 and the culture (`en-us`). The default format in data type DATETIME
 1143 is `yyyy-MM-dd hh:mm:ss.nnnnnn`. Refer to
 1144 <https://docs.microsoft.com/en-us/sql/t-sql/functions/datetime-transact-sql>
 1145 for more information about dates.
 1146

OPTION	OUTPUT	FORMAT
c	currency depending on culture (`\$`)	`c`, `en-us`
d	day without leading zero, day without leading zero and complete year (03/25/2024)	`d`, `en-us`
D	whole day of the week, first letter capitalized;	`D`, `en-us`

1166		whole month,	
1167		first letter	
1168		capitalized;	
1169		day without	
1170		leading zero	
1171		and complete	
1172		year (Monday,	
1173		April 17,	
1174		2023)	
1175			
1176			
1177			
1178	DATEPART	OUTPUT	FORMAT
1179			
1180	dw	whole day of	`dw MMMM dd, yyyy`
1181		the week,	`dw MMMM d, yyyy`
1182		first letter	`dw MMMM dd, yy`
1183		capitalized	`dw MMMM d, yy`
1184		(Monday)	
1185			
1186	MMMM	whole month,	`MMMM dd, yyyy`
1187		first letter	`MMMM d, yyyy`
1188		capitalized	`MMMM dd, yy`
1189		(April)	`MMMM d, yy`
1190			
1191	MMM	month in	`MMM dd, yyyy`
1192		abbreviation,	`MMM d, yyyy`
1193		first letter	`MMM dd, yy`
1194		capitalized	`MMM d, yy`
1195		(Apr)	`dd-MMM-yy` (default Oracle)
1196			`d-MMM-yy` (default Oracle)
1197			
1198	MM	month number	`MM/dd/yyyy`
1199		with leading	`MM/d/yyyy`
1200		zero (04)	`MM/dd/yy`
1201			`MM/d/yy`
1202			
1203	M	month number	`M/dd/yyyy`
1204		without	`M/d/yyyy`
1205		leading zero	`M/dd/yy`
1206		(4)	`M/d/yy`
1207			
1208	dddd	day of week	`dddd, MMM d, yyyy`
1209		(Monday)	`dddd, MMMM d, yyyy`
1210			
1211	ddd	day of week	`ddd, MMM d, yyyy`
1212		abbreviation	`ddd, MMMM d, yyyy`
1213		(Mon)	
1214			
1215	dd	day with	`MM/dd/yyyy`
1216		leading zero	`M/dd/yyyy`
1217		(17)	`MM/dd/yy`
1218			`M/dd/yy`
1219			
1220	d	day without	`MM/d/yyyy`
1221		leading zero	`M/d/yyyy`
1222		(17)	`MM/d/yy`
1223			`M/d/yy`
1224			
1225	yy	last two	`M/dd/yy`
1226		digits of year	`M/d/yy`
1227		(23)	`MM/d/yy`
1228			`M/d/yy`
1229			
1230	yyyy	complete year	`M/dd/yyyy`
1231		(2023)	`M/d/yyyy`
1232			`MM/d/yyyy`
1233			`M/d/yyyy`
1234			

1235	HH	24-hour,	`HH:mm:ss`
1236		military time	
1237		with leading	
1238		zero (20)	
1239	+-----+	+-----+	+-----+
1240	H	24-hour,	`H:mm:ss`
1241		military time	
1242		without	
1243		leading zero	
1244		(20)	
1245	+-----+	+-----+	+-----+
1246	hh	12-hour	`hh:mm:ss`
1247		(AM/PM), with	
1248		leading zero	
1249		(08 PM)	
1250	+-----+	+-----+	+-----+
1251	h	12-hour	`h:mm:ss`
1252		(AM/PM),	
1253		without	
1254		leading zero	
1255		(8 PM)	
1256	+-----+	+-----+	+-----+
1257	mm	minutes (13)	`HH:mm:ss`
1258	+-----+	+-----+	+-----+
1259	ss	seconds (58)	`hh:mm:ss`
1260			`h:mm:ss`
1261	+-----+	+-----+	+-----+
1262	nnnnnnn	six decimal	`HH:mm:ss.nnnnnnn`
1263		spaces,	`H:mm:ss.nnnnnnn`
1264		fractions of	`hh:mm:ss.nnnnnnn`
1265		a second	`h:mm:ss.nnnnnnn`
1266	+-----+	+-----+	+-----+

Although we are using aggregate function `AVG()`, we do not need to use `GROUP BY` since the function is inside the sub-query.

Go to <https://docs.microsoft.com/en-us/sql/t-sql/functions/format-transact-sql> for more information on `FORMAT()`.

***** */

```

1275 SELECT DISTINCT AP1.Vendors.VendorID,
1276 AP1.Vendors.VendorName,
1277 CONCAT ( -- 1. concatenating
1278 AP1.Vendors.VendorAddress1, -- `VendorAddress1`, an
1279 ' ', -- empty space and
1280 AP1.Vendors.VendorAddress2 -- `VendorAddress2`
1281 ) AS VendorAddress, -- as `VendorAddress`
1282 AP1.Vendors.VendorCity,
1283 AP1.Vendors.VendorState,
1284 CONCAT ( -- 2. concatenating
1285 AP1.Vendors.VendorZipCode, -- `VendorZipCode` and a
1286 '-0000' -- dummy Plus4 as
1287 ) AS VendorZipCode, -- VendorZipCode
1288
1289 CASE
1290 WHEN AP1.Vendors.VendorPhone <> '' -- 3. checking that
1291 OR AP1.Vendors.VendorPhone <> ' ' -- `VendorPhone` is not an
1292 OR AP1.Vendors.VendorPhone IS NOT NULL -- empty string or not a
1293 -- space or not `IS NOT
1294 -- NULL` (in other
1295 -- words, not no-value; must
1296 -- have a value) using
1297
1298 THEN CONCAT ( -- 4. concatenating an opening
1299 '(', -- parenthesis, the first 3
1300 LEFT(AP1.Vendors.VendorPhone, 3), -- characters of
1301 -- `VendorPhone` (area
1302 -- code), corresponding
1303 ') ', -- closing parenthesis with

```

```

1304         SUBSTRING (AP1.Vendors.VendorPhone, 4,3), -- a space, the substring
1305         -- from `VendorPhone`
1306         -- starting with character 4
1307         -- taking 3 characters
1308         -- (branch exchange), a
1309         '- ', -- hyphen and the 4 four
1310         RIGHT (AP1.Vendors.VendorPhone, 4) -- characters of
1311         -- `VendorPhone`
1312         -- (subscriber number) using
1313         ) -- alias `VendorPhone`
1314     ELSE ''
1315 END
1316     AS VendorPhone,
1317 LTRIM (RTRIM (
1318     -- 5. trimming the output of
1319     -- the concatenation of
1320     CONCAT (AP1.Vendors.VendorContactLName, -- `VendorContactLName`, a
1321     ', ', -- comma with a space and
1322     AP1.Vendors.VendorContactFName)) -- `VendorContactFName`
1323 ) AS VendorContactName, -- using alias
1324 -- `VendorContactName`
1325 AP1.Vendors.DefaultAccountNo,
1326 AP1.Invoices.InvoiceID,
1327 AP1.Invoices.InvoiceNumber,
1328 FORMAT (AP1.Invoices.InvoiceDate, -- 5. formatting column as
1329     'MM/dd/yyyy', 'en-us') -- `MM/dd/yyyy` (date) with
1330 -- culture `en-us` as
1331 AS InvoiceDate, -- `InvoiceDate`
1332 FORMAT (AP1.Invoices.InvoiceTotal, -- 7. formatting column as
1333     'c', 'en-us') -- `c` (currency) with
1334 -- culture `en-us` with
1335 AS InvoiceTotal, -- alias `InvoiceTotal`
1336 (
1337     SELECT -- 8. embedded query calling
1338     FORMAT (AVG (AP1.Invoices.InvoiceTotal), -- `AVG (InvoiceTotal)`
1339     'c', 'en-us') -- formatted as `c`
1340 -- (currency) with culture
1341 -- `en-us`
1342 -- from all values in table
1343 -- `AP1.Invoices` as
1344 -- `AvgInvoiceTotal`
1345 ) AS AvgInvoiceTotal, -- 9. formatting column as `c`
1346 -- (currency) with culture
1347 -- `en-us` as `PaymentTotal`
1348 FORMAT (AP1.Invoices.PaymentTotal, -- 10. formatting column as `c`
1349     'c', 'en-us') -- (currency) with culture
1350 -- `en-us` as `CreditTotal`
1351 AS PaymentTotal, -- 11. formatting column as
1352 -- `MM/dd/yyyy` (date) with
1353 -- culture `en-us` as
1354 -- `InvoiceDueDate`
1355 FORMAT (AP1.Invoices.CreditTotal, -- 12. formatting column as
1356     'c', 'en-us') -- `MM/dd/yyyy` (date) with
1357 -- culture `en-us` as
1358 AS CreditTotal, -- `PaymentDate`
1359 FORMAT (AP1.Invoices.InvoiceDueDate,
1360     'MM/dd/yyyy', 'en-us')
1361 AS InvoiceDueDate,
1362 FORMAT (AP1.Invoices.PaymentDate,
1363     'MM/dd/yyyy', 'en-us')
1364 AS PaymentDate,
1365 AP1.InvoiceLineItems.InvoiceSequence,
1366 AP1.InvoiceLineItems.AccountNo,
1367 FORMAT (AP1.InvoiceLineItems.InvoiceLineItemAmount,
1368     'c', 'en-us') -- 13. formatting column as
1369 -- `c` (currency) with
1370 -- culture `en-us` as
1371 AS InvoiceLineItemAmount, -- `InvoiceLineItemAmount`
1372 AP1.InvoiceLineItems.InvoiceLineItemDescription,
1373 AP1.Terms.TermsDescription,
1374 AP1.Terms.TermsDueDays
1375 FROM AP1.InvoiceLineItems -- 14. from
1376 -- `AP1.InvoiceLineItems`
1377 INNER JOIN AP1.Invoices -- using `INNER JOIN` to
1378 -- to connect to
1379 -- `AP1.Invoices` to get

```

```

1373                                     --      all shared values from
1374     ON AP1.InvoiceLineItems.InvoiceID = AP1.Invoices.InvoiceID
1375                                     --      `AP1.InvoiceLineItems`
1376                                     --      and `AP1.Invoices`
1377 INNER JOIN AP1.Terms                    --      using `INNER JOIN` to
1378                                     --      connect to `AP1.Terms`
1379                                     --      to get all shared values
1380                                     --      from
1381     ON AP1.Invoices.TermsID = AP1.Terms.TermsID --      (`AP1.InvoiceLineItems`
1382                                     --      and `AP1.Invoices`) and
1383                                     --      `AP1.Terms` using
1384 RIGHT JOIN AP1.Vendors                 --      `RIGHT JOIN` to connect
1385                                     --      to `AP1.Vendors` to get
1386                                     --      values from
1387                                     --      `AP1.Vendors` and
1388                                     --      related data from
1389     ON AP1.Invoices.VendorID=AP1.Vendors.VendorID --      (`AP1.InvoiceLineItems`
1390                                     --      and `AP1.Invoices` and
1391                                     --      `AP1.Terms`)
1392 ORDER BY                               -- 15. ordering results by
1393     AP1.Vendors.VendorName,            --      `VendorName` first and
1394     AP1.Invoices.InvoiceID;           --      then by `InvoiceID`
1395
1396
1397 /* *****
1398     As with previous example, we can use an alias for each table, which
1399     in this case, allows us to present neater code.
1400
1401         `il` for `AP1.InvoiceLineItems`
1402         `i`  for `AP1.Invoices`
1403         `t`  for `AP1.Terms`
1404         `v`  for `AP1.Vendors`
1405 ***** */
1406
1407 SELECT DISTINCT v.VendorID,
1408     v.VendorName,
1409     CONCAT (
1410         v.VendorAddress1,
1411         ' ',
1412         v.VendorAddress2
1413     ) AS VendorAddress,
1414     v.VendorCity,
1415     v.VendorState,
1416     CONCAT (
1417         v.VendorZipCode,
1418         '-0000'
1419     ) AS VendorZipCode,
1420     CASE
1421     WHEN v.VendorPhone <> ''
1422     OR v.VendorPhone <> ' '
1423     OR v.VendorPhone IS NOT NULL
1424     THEN CONCAT (
1425         '(',
1426         LEFT(v.VendorPhone, 3),
1427         ') ',
1428         SUBSTRING(v.VendorPhone, 4, 3),
1429         RIGHT(v.VendorPhone, 4)
1430     )
1431     ELSE ''
1432     END AS VendorPhone,
1433     LTRIM(RTRIM(CONCAT (
1434         v.VendorContactLName,
1435         ' ',
1436         v.VendorContactFName
1437     ))) AS VendorContactName,
1438     v.DefaultAccountNo,
1439     i.InvoiceID,
1440     i.InvoiceNumber,
1441     FORMAT(i.InvoiceDate, 'MM/dd/yyyy', 'en-us') AS InvoiceDate,

```

```

1442  FORMAT(i.InvoiceTotal, 'c', 'en-us') AS InvoiceTotal,
1443  (
1444      SELECT FORMAT(AVG(i.InvoiceTotal), 'c', 'en-us')
1445      FROM AP1.Invoices AS i
1446                                     -- 1. alias `i` not needed as
1447                                     --    this is a sub-query, but
1448                                     --    solely included for
1449                                     --    aesthetics in the example
1450                                     -- 2. outer aliases needed
1451      ) AS AvgInvoiceTotal,
1452  FORMAT(i.PaymentTotal, 'c', 'en-us') AS PaymentTotal,
1453  FORMAT(i.CreditTotal, 'c', 'en-us') AS CreditTotal,
1454  FORMAT(i.InvoiceDueDate, 'MM/dd/yyyy', 'en-us') AS InvoiceDueDate,
1455  FORMAT(i.PaymentDate, 'MM/dd/yyyy', 'en-us') AS PaymentDate,
1456  il.InvoiceSequence,
1457  il.AccountNo,
1458  FORMAT(il.InvoiceLineItemAmount, 'c', 'en-us')
1459  AS InvoiceLineItemAmount,
1460  il.InvoiceLineItemDescription,
1461  t.TermsDescription,
1462  t.TermsDueDays
1463 FROM AP1.InvoiceLineItems AS il
1464 INNER JOIN AP1.Invoices AS i
1465   ON il.InvoiceID = i.InvoiceID
1466 INNER JOIN AP1.Terms AS t
1467   ON i.TermsID = t.TermsID
1468 RIGHT JOIN AP1.Vendors AS v
1469   ON i.VendorID = v.VendorID
1470 ORDER BY v.VendorName,
1471         i.InvoiceID;
1472
1473 /* *****
1474 9. To get the difference between two dates, we use `DATEDIFF()`, which
1475   returns the difference between two date values, based on the interval
1476   specified` (https://techonthenet.com/sql_server/functions/datediff.php).
1477
1478   We also call functions `DAY()`,
1479   (https://techonthenet.com/sql_server/functions/day.php), `MONTH()`,
1480   (https://techonthenet.com/sql_server/functions/month.php) and `YEAR()`,
1481   (https://techonthenet.com/sql_server/functions/year.php).
1482
1483   9.01. In the example below, we use `01/01/2017` as the starting date and
1484         `12/01/2025` as the end date.
1485   ***** */
1486
1487 SELECT DATEDIFF(DAY, '01/01/2017', '12/01/2025') AS DatediffDays, -- 3,256 days
1488        DATEDIFF(MONTH, '01/01/2017', '12/01/2025') AS DatediffMonths, -- 107 months
1489        DATEDIFF(YEAR, '01/01/2017', '12/01/2025') AS DatediffYears; -- 8 years
1490
1491
1492 /* *****
1493 9.02. Instead of hard-coding today's date, we can use function `GETDATE()`,
1494   to retrieve the local system datetime -- `2025-12-01 21:52:16.327`.
1495   ***** */
1496
1497 SELECT DATEDIFF(DAY, '01/01/2017', GETDATE()) AS DatediffDays, -- 3,256 days
1498        DATEDIFF(MONTH, '01/01/2017', GETDATE()) AS DatediffMonths, -- 107 months
1499        DATEDIFF(YEAR, '01/01/2017', GETDATE()) AS DatediffYears; -- 8 years
1500
1501
1502 /* *****
1503 https://folvera.commons.gc.cuny.edu/?p=1351
1504 ***** */
1505

```