

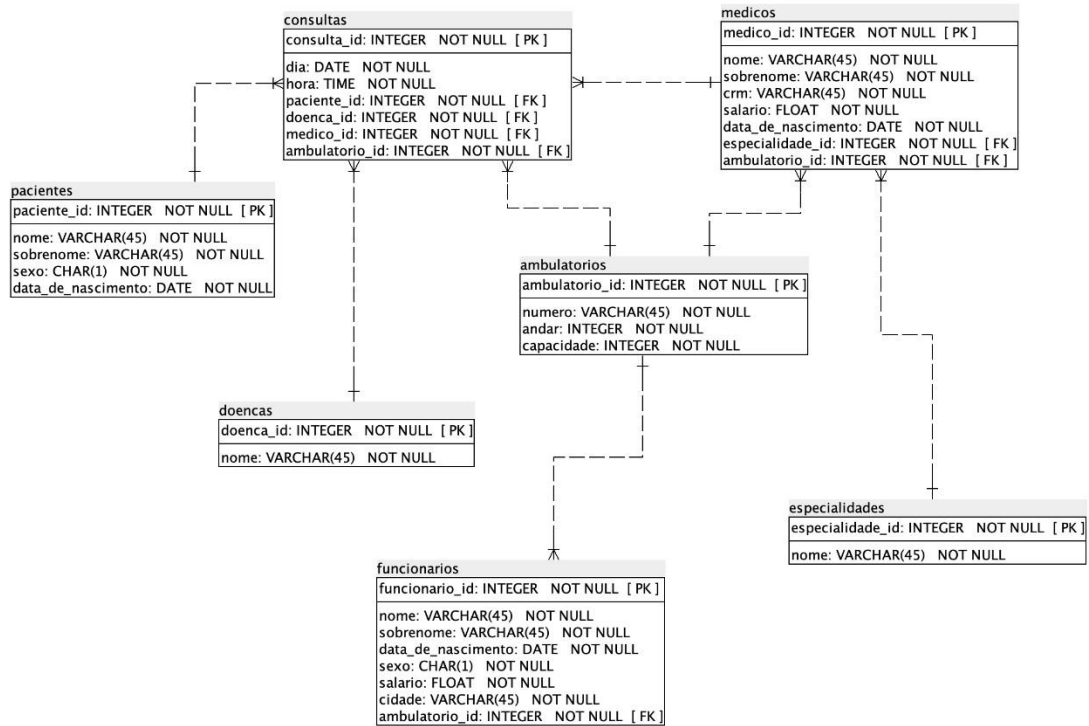
Projeto Segundo Bimestre

Professor: Jean-Rémi Bourguet

Alunos: Vitor Pedruzzi, Renan Manera, Kauan
Modolo, João Pedro Becker, Arthur Rueda

1º Etapa

Vitor Pedruzzi, Renan Manera, Kauan Modolo,
João Pedro Becker, Arthur Rueda



Script SQL:

-- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda -- CREATE

```

TABLE ambulatorios (

    ambulatorio_id INT NOT NULL,

    numero VARCHAR(45) NOT NULL,

    andar INT NOT NULL,        capacidade

    INT NOT NULL,

    PRIMARY KEY (ambulatorio_id)

);
  
```

```

CREATE TABLE funcionarios (
  
```

```
funcionario_id INT NOT NULL,

nome VARCHAR(45) NOT NULL,

sobrenome VARCHAR(45) NOT NULL,

data_de_nascimento DATE NOT NULL,

sexo CHAR(1) NOT NULL,          salario

DOUBLE PRECISION NOT NULL,      cidade

VARCHAR(45) NOT NULL,

ambulatorio_id INT NOT NULL,

PRIMARY KEY (funcionario_id)

);
```

```
CREATE TABLE doencas (

    doenca_id INT NOT NULL,

    nome VARCHAR(45) NOT NULL,

    PRIMARY KEY (doenca_id)

);
```

```
CREATE TABLE especialidades (

    especialidade_id INT NOT NULL,

    nome VARCHAR(45) NOT NULL,

    PRIMARY KEY (especialidade_id)

);
```

```
CREATE TABLE medicos (
```

```
        medico_id INT NOT NULL,

nome VARCHAR(45) NOT NULL,

sobrenome VARCHAR(45) NOT NULL,

crm VARCHAR(45) NOT NULL,        salario

DOUBLE PRECISION NOT NULL,

data_de_nascimento DATE NOT NULL,

especialidade_id INT NOT NULL,

ambulatorio_id INT NOT NULL,

        PRIMARY KEY (medico_id)

);
```

```
CREATE TABLE pacientes (

        paciente_id INT NOT NULL,

nome VARCHAR(45) NOT NULL,

sobrenome VARCHAR(45) NOT NULL,

sexo CHAR(1) NOT NULL,

data_de_nascimento DATE NOT NULL,

        PRIMARY KEY (paciente_id)

);
```

```
CREATE TABLE consultas (

        consulta_id INT NOT NULL,

dia DATE NOT NULL,        hora

TIME NOT NULL,

paciente_id INT NOT NULL,

doenca_id INT NOT NULL,
```

```
medico_id INT NOT NULL,  
  
ambulatorio_id INT NOT NULL,  
  
    PRIMARY KEY (consulta_id)  
  
);
```

```
ALTER TABLE medicos ADD CONSTRAINT ambulatorios_medicos_fk  
  
FOREIGN KEY (ambulatorio_id)  
  
REFERENCES ambulatorios (ambulatorio_id)  
  
ON DELETE NO ACTION  
  
ON UPDATE NO ACTION;
```

```
ALTER TABLE consultas ADD CONSTRAINT ambulatorios_consultas_fk  
  
FOREIGN KEY (ambulatorio_id)  
  
REFERENCES ambulatorios (ambulatorio_id)  
  
ON DELETE NO ACTION  
  
ON UPDATE NO ACTION;
```

```
ALTER TABLE funcionarios ADD CONSTRAINT ambulatorios_funcionarios_fk  
  
FOREIGN KEY (ambulatorio_id)  
  
REFERENCES ambulatorios (ambulatorio_id)  
  
ON DELETE NO ACTION  
  
ON UPDATE NO ACTION;
```

```
ALTER TABLE consultas ADD CONSTRAINT doencas_consultas_fk  
  
FOREIGN KEY (doenca_id)  
  
REFERENCES doencas (doenca_id) ON  
  
DELETE NO ACTION
```

ON UPDATE NO ACTION;

ALTER TABLE medicos ADD CONSTRAINT especialidades_medicos_fk

FOREIGN KEY (especialidade_id)

REFERENCES especialidades (especialidade_id)

ON DELETE NO ACTION

ON UPDATE NO ACTION;

ALTER TABLE consultas ADD CONSTRAINT medicos_consultas_fk

FOREIGN KEY (medico_id)

REFERENCES medicos (medico_id)

ON DELETE NO ACTION

ON UPDATE NO ACTION;

ALTER TABLE consultas ADD CONSTRAINT pacientes_consultas_fk

FOREIGN KEY (paciente_id)

REFERENCES pacientes (paciente_id)


ON DELETE NO ACTION

ON UPDATE NO ACTION;

2º Etapa (Populando + API)

Ref: <https://www.mockaroo.com/>

Print do uso da API:



SCHEMASDATASETSMOCK APISSCENARIOSPROJECTSFUNCTIONS?SIGN INUPGRADE NOW

Looking to generate **fake data** based on your **production data**? Mimic your databases with a trial account from **TONIC**

Need some mock data to test your app? Mockaroo lets you generate up to 1,000 rows of realistic test data in CSV, JSON, SQL, and Excel formats.
Need more data? Plans start at just \$60/year. Mockaroo is also available as a [docker image](#) that you can deploy in your own private cloud.

Field Name	Type	Options
ambulatorio_id	Row Number	blank: 0 % Σ ×
numero	Row Number	blank: 0 % Σ ×
andar	Number	min: 1 max: 4 decimals: 0 blank: 0 % Σ ×
capacidade	Number	min: 10 max: 30 decimals: 0 blank: 0 % Σ ×

+ ADD ANOTHER FIELD

GENERATE FIELDS USING AI...

Rows: 100Format: SQLTable Name: ambulatorios☐ include CREATE TABLE

Field Name	Type	Options
consulta_id	Row Number	blank: 0 % Σ ×
dia	Datetime	05/30/2003 to 05/30/2024 format: yyyy-mm-dd blank: 0 % Σ ×
hora	Time	from 12:00 AM to 11:59 PM format: 24 Hour blank: 0 % Σ ×
paciente_id	Row Number	blank: 0 % Σ ×
doenca_id	Number	min: 1 max: 20 decimals: 0 blank: 0 % Σ ×
medico_id	Number	min: 1 max: 200 decimals: 0 blank: 0 % Σ ×
ambulatorio_id	Number	min: 1 max: 100 decimals: 0 blank: 0 % Σ ×

+ ADD ANOTHER FIELD

GENERATE FIELDS USING AI...

Rows: 1000Format: SQLTable Name: consultas☐ include CREATE TABLE

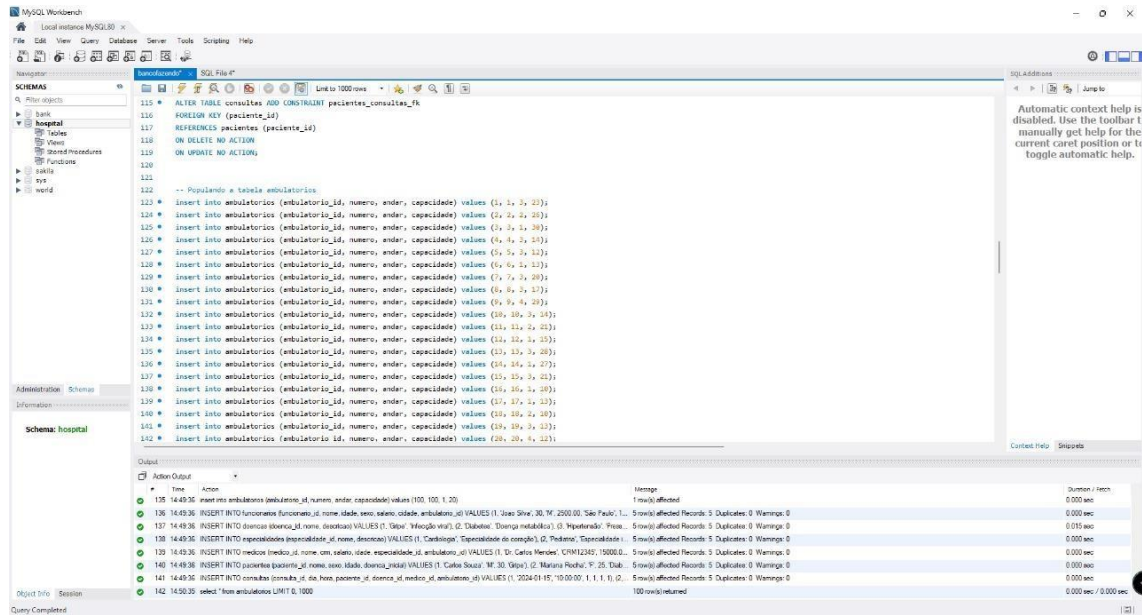
GENERATE DATA

PREVIEW

SAVE AS...

DERIVE FROM EXAMPLE...

MORE



Script dos inserts gerados pela API

Link:

https://drive.google.com/drive/folders/1oJeJZ0UJEI2OF53k1AywfgFaywD8DsEz?usp=drive_link

Procedimento armazenado com laços:

1º Ambulatorios:

```
-- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda --  
/* DELIMITER //  
CREATE PROCEDURE PopulateAmbulatorios(IN num_records INT)  
BEGIN  
    DECLARE i INT DEFAULT 101;  
    DECLARE num INT DEFAULT 101;  
    DECLARE count INT DEFAULT 0;  
  
    WHILE count < num_records DO  
        INSERT INTO ambulatorios (ambulatorio_id, numero, andar, capacidade)  
        VALUES (i, num, FLOOR(RAND() * 4) + 1, FLOOR(RAND() * 31) + 10);  
        SET i = i + 1;  
        SET num = num + 1;  
        SET count = count + 1;  
    END WHILE;  
END //  
DELIMITER ; */  
  
CALL PopulateDoencas(100);
```

SQL File 4*123 - Schema1235 - Schema

Limit to 1000 rows

SQL Additions

```
1 -- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Hanera, Arthur Rueda --
2 /* DELIMITER //
3
4 CREATE PROCEDURE PopulateAmbulatorios(IN num_records INT)
5 BEGIN
6     DECLARE i INT DEFAULT 101;
7     DECLARE num INT DEFAULT 101;
8     DECLARE count INT DEFAULT 0;
9
10    WHILE count < num_records DO
11        INSERT INTO ambulatorios (ambulatorio_id, numero, andar, capacidade)
12            VALUES (i, num, FLOOR(RAND() * 4) + 1, FLOOR(RAND() * 31) + 10);
13        SET i = i + 1;
14        SET num = num + 1;
15        SET count = count + 1;
16    END WHILE;
17 END //
18
19 DELIMITER ; */
20
21 CALL PopulateAmbulatorios(100);
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Output

Context HelpShippets

#	Time	Action	Message	Duration / Fetch
1	18:51:04	CREATE PROCEDURE PopulateAmbulatorios(IN num_records INT) BEGIN DECLARE i INT DEFAULT 101; DECLARE num INT DEFAULT 101; ...	0 row(s) affected	0.000 sec
2	18:52:31	CALL PopulateAmbulatorios(100)	1 row(s) affected	0.063 sec

SQL File 4*123 - Schema1235 - Schema

Limit to 1000 rows

SQL Additions

```
8 DECLARE count INT DEFAULT 0;
9
10 WHILE count < num_records DO
11     INSERT INTO ambulatorios (ambulatorio_id, numero, andar, capacidade)
12         VALUES (i, num, FLOOR(RAND() * 4) + 1, FLOOR(RAND() * 31) + 10);
13     SET i = i + 1;
14     SET num = num + 1;
15     SET count = count + 1;
16 END WHILE;
17 END //
18
19 DELIMITER ; */
20
21 -- CALL PopulateAmbulatorios(100);
22
23 Select * from ambulatorios
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid

ambulatorios 2 x

ambulatorio_id	numero	andar	capacidade
118	118	1	33
119	119	1	14
120	120	3	13
121	121	1	39
122	122	3	17
123	123	2	12
124	124	2	12
125	125	3	40
126	126	4	36
127	127	2	28
128	128	3	34
129	129	4	23
130	130	4	24
131	131	3	15

ambulatorios 2 x

Output

Context HelpShippets

#	Time	Action	Message	Duration / Fetch
1	18:51:04	CREATE PROCEDURE PopulateAmbulatorios(IN num_records INT) BEGIN DECLARE i INT DEFAULT 101; DECLARE num INT DEFAULT 101; ...	0 row(s) affected	0.000 sec
2	18:52:31	CALL PopulateAmbulatorios(100)	1 row(s) affected	0.063 sec
3	18:53:22	Select * from ambulatorios LIMIT 0, 1000	200 row(s) returned	0.000 sec / 0.000 sec

2º Doencas:

```
-- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda --  
/* DELIMITER //
```

```
CREATE PROCEDURE PopulateNewDoencas(IN num_records INT)  
BEGIN  
    DECLARE i INT;  
    DECLARE count INT DEFAULT 0;  
    DECLARE max_id INT DEFAULT 0;  
  
    -- Lista de nomes específicos de doenças  
    DECLARE disease_name1 VARCHAR(45) DEFAULT 'esclerose';  
    DECLARE disease_name2 VARCHAR(45) DEFAULT 'lupus';  
    DECLARE disease_name3 VARCHAR(45) DEFAULT 'fibromialgia';  
    DECLARE disease_name4 VARCHAR(45) DEFAULT 'psoríase';  
    DECLARE disease_name5 VARCHAR(45) DEFAULT 'esquizofrenia';  
    DECLARE disease_name6 VARCHAR(45) DEFAULT 'depressão';  
    DECLARE disease_name7 VARCHAR(45) DEFAULT 'transtorno de ansiedade';  
    DECLARE disease_name8 VARCHAR(45) DEFAULT 'osteoporose';  
    DECLARE disease_name9 VARCHAR(45) DEFAULT 'câncer';  
    DECLARE disease_name10 VARCHAR(45) DEFAULT 'hipotireoidismo';  
  
    -- Encontrar o máximo doença_id existente  
    SELECT IFNULL(MAX(doença_id), 0) INTO max_id FROM doencas;  
    SET i = max_id + 1;  
  
    WHILE count < num_records DO  
        CASE (count MOD 10)  
            WHEN 0 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name1);  
            WHEN 1 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name2);  
            WHEN 2 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name3);  
            WHEN 3 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name4);  
            WHEN 4 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name5);  
            WHEN 5 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name6);  
            WHEN 6 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name7);  
            WHEN 7 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name8);  
            WHEN 8 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name9);  
            WHEN 9 THEN INSERT INTO doencas (doença_id, nome) VALUES (i, disease_name10);  
        END CASE;  
        SET i = i + 1;  
        SET count = count + 1;  
    END WHILE;  
END //
```

```
DELIMITER ; */  
  
-- CALL PopulateNewDoencas(7)
```

SQL File 4" SQL File 4" 123 - Schema

1 -- Vitor Pedruzzi, Kauan Nodolo, João Pedro Becker, Renan Manera, Arthur Rueda --
2 /* DELIMITER //
3
4 CREATE PROCEDURE PopulateNewDoencas(IN num_records INT)
5 BEGIN
6 DECLARE i INT;
7 DECLARE count INT DEFAULT 0;
8 DECLARE max_id INT DEFAULT 0;
9
10 -- Lista de nomes específicos de doenças
11 DECLARE disease_name1 VARCHAR(45) DEFAULT 'esclerose';
12 DECLARE disease_name2 VARCHAR(45) DEFAULT 'lupus';
13 DECLARE disease_name3 VARCHAR(45) DEFAULT 'fibromialgia';
14 DECLARE disease_name4 VARCHAR(45) DEFAULT 'psoríase';
15 DECLARE disease_name5 VARCHAR(45) DEFAULT 'esquizofrenia';
16 DECLARE disease_name6 VARCHAR(45) DEFAULT 'depressão';
17 DECLARE disease_name7 VARCHAR(45) DEFAULT 'transtorno de ansiedade';
18 DECLARE disease_name8 VARCHAR(45) DEFAULT 'osteoporose';
19 DECLARE disease_name9 VARCHAR(45) DEFAULT 'câncer';
20 DECLARE disease_name10 VARCHAR(45) DEFAULT 'hipotireoidismo';
21
22 -- Encontrar o máximo doença_id existente
23 SELECT IFNULL(MAX(doenca_id), 0) INTO max_id FROM doencas;
24 SET i = max_id + 1;
25
26 WHILE count < num_records DO
27 CASE (count MOD 10)
28 WHEN 0 THEN INSERT INTO doencas (doenca_id, nome) VALUES (i, disease_name1);
29 WHEN 1 THEN INSERT INTO doencas (doenca_id, nome) VALUES (i, disease_name2);
30 WHEN 2 THEN INSERT INTO doencas (doenca_id, nome) VALUES (i, disease_name3);
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
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
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

Output

#	Time	Action	Message	Duration / Fetch
1	20:16:11	CREATE PROCEDURE PopulateNewDoencas(IN num_records INT) BEGIN DECLARE i INT; DECLARE count INT DEFAULT 0; DECLARE max...	0 row(s) affected	0.000 sec
2	20:16:25	CALL PopulateNewDoencas(7) -- Select "from doencas	1 row(s) affected	0.015 sec
3	20:16:30	Select "from doencas LIMIT 0, 1000	27 row(s) returned	0.000 sec / 0.000 sec

SQL File 4" SQL File 4" 123 - Schema

33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
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
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

Output

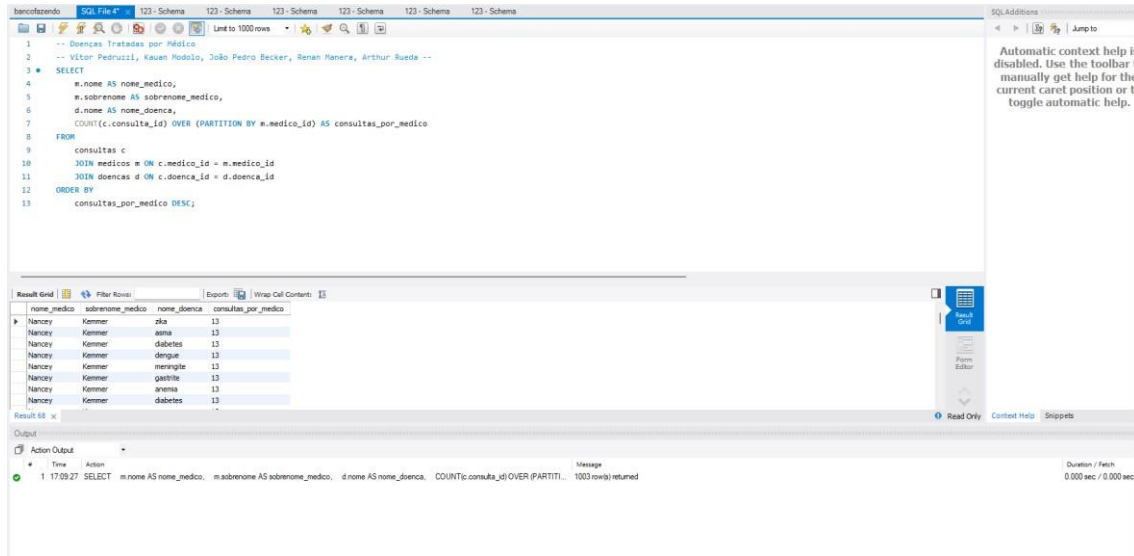
#	Time	Action	Message	Duration / Fetch
1	20:16:11	CREATE PROCEDURE PopulateNewDoencas(IN num_records INT) BEGIN DECLARE i INT; DECLARE count INT DEFAULT 0; DECLARE max...	0 row(s) affected	0.000 sec
2	20:16:25	CALL PopulateNewDoencas(7) -- Select "from doencas	1 row(s) affected	0.015 sec
3	20:16:30	Select "from doencas LIMIT 0, 1000	27 row(s) returned	0.000 sec / 0.000 sec
4	20:20:07	Select "from doencas LIMIT 0, 1000	27 row(s) returned	0.000 sec / 0.000 sec

Result Grid

doenca_id	nome
15	sífilis
16	chikungunya
17	hanseníase
18	anemia
19	malária
20	gripe
21	esclerose
22	lupus
23	fibromialgia
24	psoríase
25	esquizofrenia
26	depressão
27	transtorno de ansiedade

3º Etapa (Consultas)

1º:



The screenshot shows a SQL IDE with a query editor and a results grid. The query is as follows:

```
-- Doenças Tratadas por Médico
-- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda --
1
2
3 SELECT
4     m.nome AS nome_medico,
5     m.sobrenome AS sobrenome_medico,
6     d.nome AS nome_doenca,
7     COUNT(c.consulta_id) OVER (PARTITION BY m.medico_id) AS consultas_por_medico
8 FROM
9     consultas c
10    JOIN medicos m ON c.medico_id = m.medico_id
11    JOIN doencas d ON c.doenca_id = d.doenca_id
12 ORDER BY
13     consultas_por_medico DESC;
```

The results grid shows the following data:

nome_medico	sobrenome_medico	nome_doenca	consultas_por_medico
Nancy	Kemmer	zika	13
Nancy	Kemmer	anemia	13
Nancy	Kemmer	diabetes	13
Nancy	Kemmer	dengue	13
Nancy	Kemmer	meningite	13
Nancy	Kemmer	gastro	13
Nancy	Kemmer	anemia	13
Nancy	Kemmer	diabetes	13

-- Doenças Tratadas por Médico

-- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda --

SELECT

m.nome AS nome_medico,

m.sobrenome AS sobrenome_medico,

d.nome AS nome_doenca,

COUNT(c.consulta_id) OVER (PARTITION BY m.medico_id) AS consultas_por_medico

FROM

consultas c

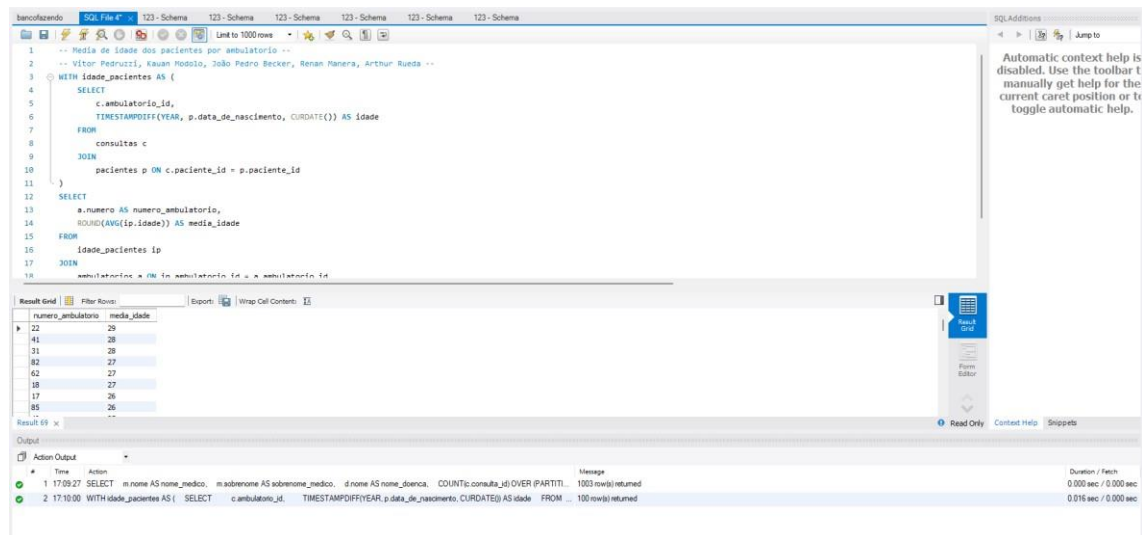
JOIN medicos m ON c.medico_id = m.medico_id

JOIN doencas d ON c.doenca_id = d.doenca_id

ORDER BY

consultas_por_medico DESC;

2º:



-- Media de idade dos pacientes por ambulatorio --

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

```

WITH idade_pacientes AS (
    SELECT
        c.ambulatorio_id,
        TIMESTAMPDIFF(YEAR, p.data_de_nascimento, CURDATE()) AS idade

```

FROM

```

        consultas c    JOIN        pacientes p ON

```

```

c.paciente_id = p.paciente_id

```

)

```

SELECT
    a.numero AS numero_ambulatorio,
    ROUND(AVG(ip.idade)) AS media_idade FROM

```

```

    idade_pacientes ip
JOIN    ambulatorios a ON ip.ambulatorio_id =

```

```

a.ambulatorio_id

```

GROUP BY

```

    ip.ambulatorio_id

```

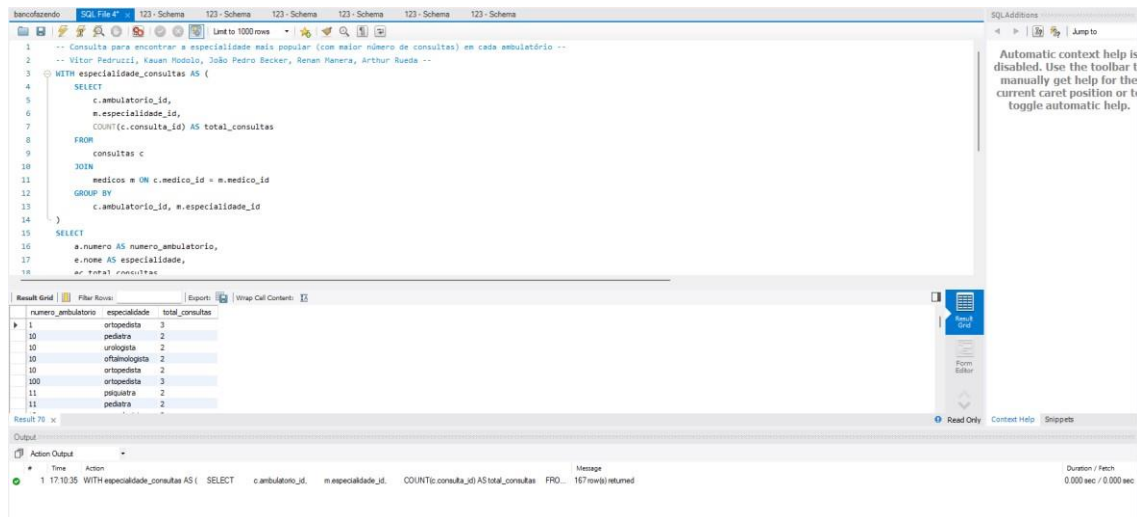
ORDER BY

```

    media_idade DESC;

```

3º:



-- Consulta para encontrar a especialidade mais popular (com maior número de consultas) em cada ambulatório --
 -- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

```

WITH especialidade_consultas AS (

    SELECT

        c.ambulatorio_id,

        m.especialidade_id,

        COUNT(c.consulta_id) AS total_consultas

    FROM

        consultas c

        JOIN      medicos m ON c.medico_id =

m.medico_id

    GROUP BY

        c.ambulatorio_id, m.especialidade_id

)

SELECT

    a.numero AS numero_ambulatorio,

```

```

        e.nome AS especialidade,
ec.total_consultas

FROM

    especialidade_consultas ec JOIN    especialidades e ON
ec.especialidade_id = e.especialidade_id JOIN    ambulatorios
a ON ec.ambulatorio_id = a.ambulatorio_id

WHERE

    (ec.ambulatorio_id, ec.total_consultas) IN (

SELECT        ambulatorio_id,

                MAX(total_consultas)

FROM

    especialidade_consultas

GROUP BY

    ambulatorio_id

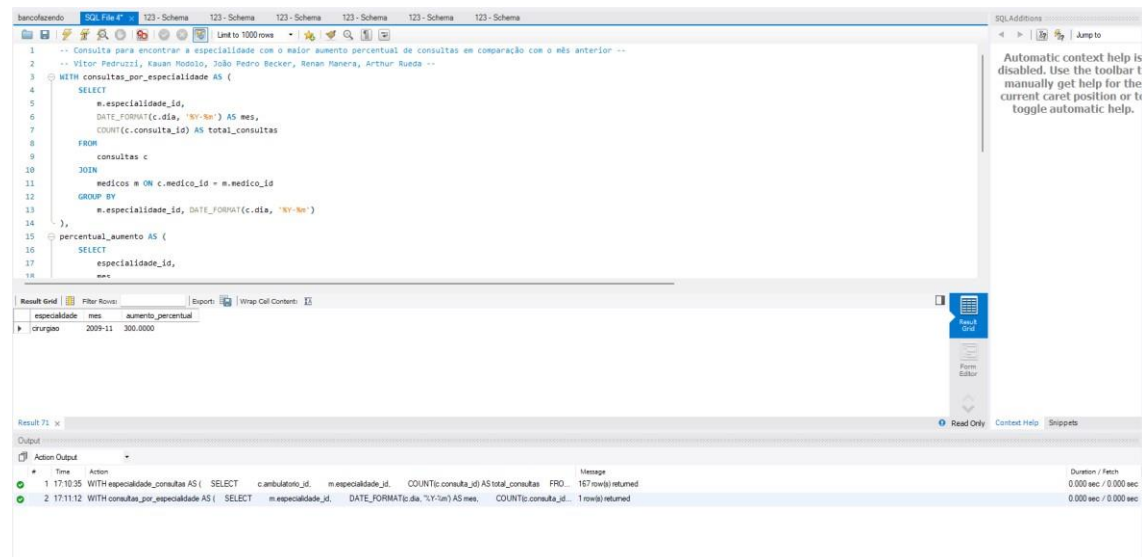
    )

ORDER BY

    a.numero;

```

4°:



-- Consulta para encontrar a especialidade com o maior aumento percentual de consultas em comparação com o mês anterior --

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

WITH consultas_por_especialidade AS (

SELECT

m.especialidade_id,

DATE_FORMAT(c.dia, '%Y-%m') AS mes,

COUNT(c.consulta_id) AS total_consultas

FROM

consultas c

JOIN

medicos m ON c.medico_id = m.medico_id

GROUP BY

m.especialidade_id, DATE_FORMAT(c.dia, '%Y-%m')

),

percentual_aumento AS (

SELECT

especialidade_id,

```

        mes,
total_consultas,
        LAG(total_consultas) OVER (PARTITION BY especialidade_id ORDER BY mes) AS
consultas_mes_anterior,
        (total_consultas - LAG(total_consultas) OVER (PARTITION BY especialidade_id
ORDER BY mes)) / NULLIF(LAG(total_consultas) OVER (PARTITION BY especialidade_id
ORDER BY mes), 0) * 100 AS aumento_percentual FROM
consultas_por_especialidade
)
SELECT
        e.nome AS especialidade,
        p.mes,
        p.aumento_percentual FROM
        percentual_aumento p JOIN especialidades e ON
        p.especialidade_id = e.especialidade_id
WHERE
        p.aumento_percentual IS NOT NULL
ORDER BY
        p.aumento_percentual DESC
LIMIT 1;

```

5°:

The screenshot shows a SQL IDE interface with a query editor at the top and a results grid at the bottom. The query is as follows:

```
1  -- Doenças Mais Comuns nas Consultas
2  -- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda --
3  SELECT
4      d.nome AS nome_doenca,
5      COUNT(c.consulta_id) AS total_consultas
6  FROM
7      consultas c
8  JOIN doencas d ON c.doenca_id = d.doenca_id
9  GROUP BY
10     d.nome
11 ORDER BY
12     total_consultas DESC
13 LIMIT 10;
```

The results grid displays the following data:

nome_doenca	total_consultas
hepatite	65
covid	63
dengue	62
hipertensao	58
pneumonia	58
jaune	57
chikungunya	56
anemia	54

The output pane at the bottom shows the execution message: "1 17:12:22 SELECT d.nome AS nome_doenca, COUNT(c.consulta_id) AS total_consultas FROM consultas c JOIN doencas d ON c.doenca_id = d.doenca_id 10 row(s) returned".

-- Doenças Mais Comuns nas Consultas

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

SELECT

d.nome AS nome_doenca,

COUNT(c.consulta_id) AS total_consultas

FROM

consultas c

JOIN doencas d ON c.doenca_id = d.doenca_id GROUP

BY

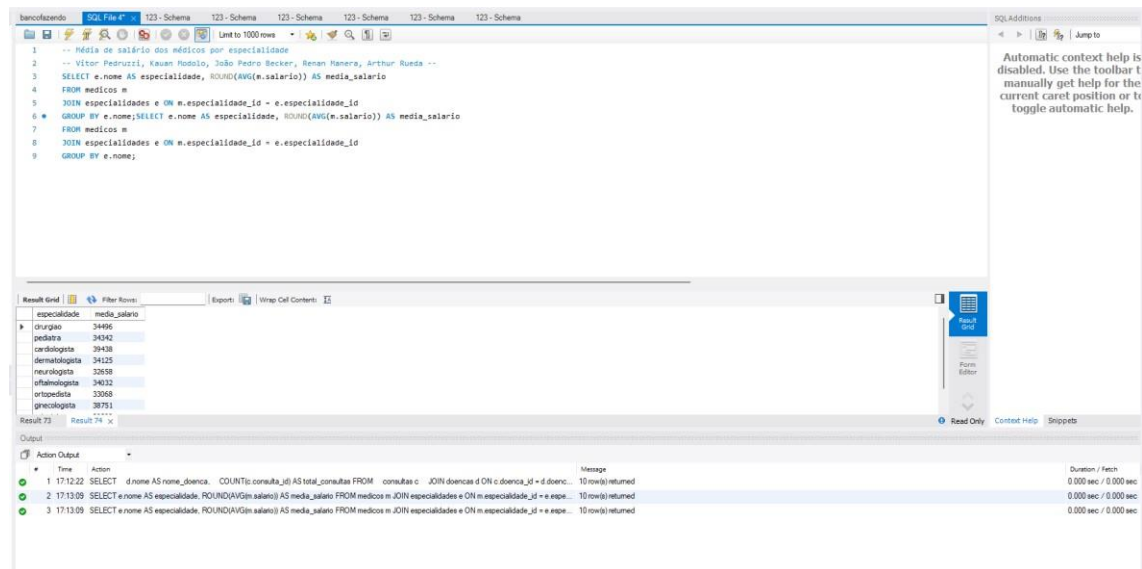
d.nome ORDER

BY

total_consultas DESC

LIMIT 10;

6°:



-- Média de salário dos médicos por especialidade

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

SELECT e.nome AS especialidade, ROUND(AVG(m.salario)) AS media_salario

FROM medicos m

JOIN especialidades e ON m.especialidade_id = e.especialidade_id

GROUP BY e.nome; SELECT e.nome AS especialidade, ROUND(AVG(m.salario)) AS media_salario

FROM medicos m

JOIN especialidades e ON m.especialidade_id = e.especialidade_id

GROUP BY e.nome;

7°:

The screenshot shows a SQL IDE with a query editor at the top and a results grid at the bottom. The query is as follows:

```
-- Total de Consultas por Médico e Andar do Ambulatório --
SELECT
  m.nome AS nome_medico,
  m.sobrenome AS sobrenome_medico,
  a.andar,
  COUNT(c.consulta_id) AS total_consultas
FROM
  consultas c
JOIN medicos m ON c.medico_id = m.medico_id
JOIN ambulatorios a ON m.ambulatorio_id = a.ambulatorio_id
GROUP BY
  m.medico_id, a.andar
ORDER BY
  total_consultas DESC
LIMIT 20;
```

The results grid displays the following data:

nome_medico	sobrenome_medico	andar	total_consultas
Nancy	Kemmer	4	13
Gabriel	Christiane	3	12
Brooks	Hesher	2	12
Dora	Tumassian	2	11
Evelyn	Gault	2	10
Sheldon	Nadde	1	10
Freeland	Burleton	2	10
Nichole	Hennemann	1	10

The output pane at the bottom shows the execution of the query, with the following message:

```
1 17:12:22 SELECT d.nome AS nome_medico, COUNT(c.consulta_id) AS total_consultas FROM consultas c JOIN medicos m ON c.medico_id = m.medico_id JOIN ambulatorios a ON m.ambulatorio_id = a.ambulatorio_id GROUP BY m.medico_id, a.andar ORDER BY total_consultas DESC LIMIT 20; 10 rows returned 0.000 sec / 0.000 sec
```

-- Total de Consultas por Médico e Andar do Ambulatório

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

SELECT

m.nome AS nome_medico,

m.sobrenome AS sobrenome_medico,

a.andar,

COUNT(c.consulta_id) AS total_consultas

FROM

consultas c

JOIN medicos m ON c.medico_id = m.medico_id

JOIN ambulatorios a ON m.ambulatorio_id = a.ambulatorio_id

GROUP BY

m.medico_id, a.andar ORDER

BY

total_consultas DESC

LIMIT 20;

8°:

The screenshot shows the SQL Developer interface. The main window displays a SQL query with comments in Portuguese. The query filters for ambulatories with an average salary greater than 5500 and lists the names of the staff members. The results grid shows the following data:

numero_ambulatorio	media_salario
4	5858
6	9973
7	9838
8	9712
13	6407
20	7728
22	9491
25	5908

The output window shows the execution of the query, indicating that 43 rows were returned.

-- Vamos filtrar apenas os ambulatórios onde a média salarial é superior a 5500:

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

SELECT

a.numero AS numero_ambulatorio,

ROUND(AVG(f.salario)) AS media_salario FROM

funcionarios f JOIN

ambulatorios a ON

f.ambulatorio_id = a.ambulatorio_id

GROUP BY

a.numero

HAVING

AVG(f.salario) > 5500;

9°:

The screenshot shows a SQL IDE interface with a query editor at the top and a results grid at the bottom. The query editor contains the following SQL code:

```
1 -- Listar pacientes com as doenças mais frequentes que foram diagnosticadas
2 -- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda --
3 * SELECT p.nome, p.sobrenome, d.nome AS doenca, COUNT(*) AS frequencia
4 FROM pacientes p
5 JOIN consultas c ON p.paciente_id = c.paciente_id
6 JOIN doencas d ON c.doenca_id = d.doenca_id
7 GROUP BY p.nome, p.sobrenome, d.nome
8 ORDER BY frequencia DESC;
```

The results grid displays the following data:

nome	sobrenome	doenca	frequencia
Patric	Wanda	diabetes	1
Kaube	Tomara	diabetes	1
Deia	Blaydon	diabetes	1
Dorothy	Forge	diabetes	1
Coop	Debrovoldi	diabetes	1
Bryn	Bogays	diabetes	1
Albert	Averay	diabetes	1
Tomken	Keller	diabetes	1

The output pane at the bottom shows the execution message: "1 17:15:06 SELECT p.nome, p.sobrenome, d.nome AS doenca, COUNT(*) AS frequencia FROM pacientes p JOIN consultas c ON p.paciente_id = c.paciente_id JO... 1000 row(s) returned".

-- Listar pacientes com as doenças mais frequentes que foram diagnosticadas

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

SELECT p.nome, p.sobrenome, d.nome AS doenca, COUNT(*) AS frequencia

FROM pacientes p

JOIN consultas c ON p.paciente_id = c.paciente_id

JOIN doencas d ON c.doenca_id = d.doenca_id

GROUP BY p.nome, p.sobrenome, d.nome

ORDER BY frequencia DESC;

4º Otimizar

Script dos códigos para criar os índices:

-- Índice para a tabela consultas

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo,
João Pedro Becker, Arthur Rueda

CREATE INDEX idx_consultas_doenca_id ON
consultas (doenca_id);

Diferença entre rodar o código antes e depois de
criar o index:

The screenshot shows the MySQL Workbench interface. The 'Query Editor' displays a SQL query that selects the disease name and the total number of consultations for each disease, joined with the 'consultas' table. The 'Result Grid' shows the results of the query, with columns 'nome_doenca' and 'total_consultas'. The 'Output' pane shows the execution log, including the time taken for each query and the number of rows affected or returned.

nome_doenca	total_consultas
hepatite	65
covid	63
dengue	62
pneumonia	58
hipertensao	57
asma	56
chikungunya	56
anemia	54
meningite	53

#	Time	Action	Message	Duration / Fetch
2541	15:25:56	insert into consultas (consulta_id, dia, hora, paciente_id, doenca_id, medico_id, ambulatorio_id)	1 row(s) affected	0.015 sec
2542	15:25:56	insert into consultas (consulta_id, dia, hora, paciente_id, doenca_id, medico_id, ambulatorio_id)	1 row(s) affected	0.031 sec
2543	15:25:56	insert into consultas (consulta_id, dia, hora, paciente_id, doenca_id, medico_id, ambulatorio_id)	1 row(s) affected	0.016 sec
2544	15:25:56	insert into consultas (consulta_id, dia, hora, paciente_id, doenca_id, medico_id, ambulatorio_id)	1 row(s) affected	0.016 sec
2545	15:25:42	SELECT * FROM consultas c JOIN medicos m ON c.medico_id = m.medico_id JOIN pacientes p ON c.paciente_id = p.paciente_id	1000 row(s) returned	0.015 sec / 0.032 sec
2546	15:28:04	SELECT d.nome AS nome_doenca, COUNT(c.consulta_id) AS total_consultas FROM consultas c JOIN doencas d ON c.doenca_id = d.doenca_id GROUP BY d.nome	20 row(s) returned	0.062 sec / 0.000 sec

MySQL Workbench

Local instance MySQL80 x MySQL Model x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

banconew

Tables

Views

Stored Procedures

Functions

gerenciamentooncondominios

sakila

sys

world

Administration Schemas

Information

Schema: banconew

Query 1

SQL File 6

Limit to 50000 rows

6 JOIN doencas d ON c.doenca_id = d.doenca_id

7 GROUP BY

8 d.nome

9 ORDER BY

10 total_consultas DESC

11 LIMIT 20;

12 -- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda --

Result Grid

nome_doenca	total_consultas
hepatite	65
covid	63
dengue	62
pneumonia	58
hipertensao	57
asma	56
chikungunya	56
anemia	54
meningite	53

Result: 4 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	16:09:45	show index from pacientes	2 row(s) returned	0.047 sec / 0.000 sec
2	16:10:23	show index from consultas	5 row(s) returned	0.032 sec / 0.000 sec
3	16:11:16	SELECT p.nome, p.sobrenome, d.nome AS doenca, COUNT(*) AS frequencia FROM paciente...	1000 row(s) returned	0.016 sec / 0.000 sec
4	16:15:06	SELECT d.nome AS nome_doenca, COUNT(c.consulta_id) AS total_consultas FROM ...	20 row(s) returned	0.015 sec / 0.000 sec

Object Info Session

Query Completed

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Print mostrando o index:

MySQL Workbench

Local instance MySQL80 x MySQL Model x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

banconew

Tables

Views

Stored Procedures

Functions

gerenciamentooncondominios

sakila

sys

world

Administration Schemas

Information

Schema: banconew

Query 1

SQL File 6

Limit to 50000 rows

1 show index from consultas

2

3

4 -- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda --

Result Grid

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment
consultas	0	PRIMARY	1	consulta_id	A	1000				BTREE	
consultas	1	ambulatorios_consultas_fk	1	ambulatorio_id	A	100				BTREE	
consultas	1	medicos_consultas_fk	1	medico_id	A	200				BTREE	
consultas	1	idx_consultas_paciente_id	1	paciente_id	A	1000				BTREE	
consultas	1	idx_consultas_doenca_id	1	doenca_id	A	20				BTREE	

Result: 6 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	16:09:45	show index from pacientes	2 row(s) returned	0.047 sec / 0.000 sec
2	16:10:23	show index from consultas	5 row(s) returned	0.032 sec / 0.000 sec
3	16:11:16	SELECT p.nome, p.sobrenome, d.nome AS doenca, COUNT(*) AS frequencia FROM paciente...	1000 row(s) returned	0.016 sec / 0.000 sec
4	16:15:06	SELECT d.nome AS nome_doenca, COUNT(c.consulta_id) AS total_consultas FROM ...	20 row(s) returned	0.015 sec / 0.000 sec
5	16:16:45	show index from consultas	5 row(s) returned	0.000 sec / 0.000 sec
6	17:55:18	show index from consultas --Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Mane...	5 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query Completed

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

5º Estruturas

Avançadas:

Função table – valued:

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro
Becker, Arthur Rueda --

DELIMITER //

CREATE PROCEDURE GetDoencasMaisComuns()

BEGIN

SELECT

d.nome AS nome_doenca,

COUNT(c.consulta_id) AS total_consultas

FROM

consultas c

JOIN doencas d ON c.doenca_id = d.doenca_id

GROUP BY d.nome

ORDER BY

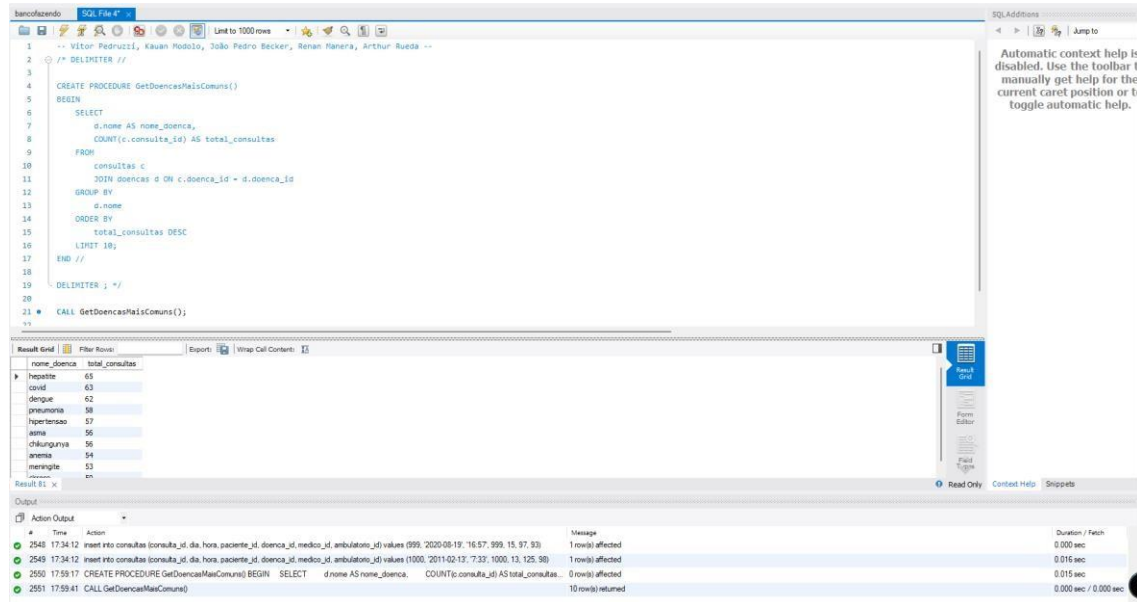
total_consultas DESC

LIMIT 10;

END //

DELIMITER ;

CALL GetDoencasMaisComuns();

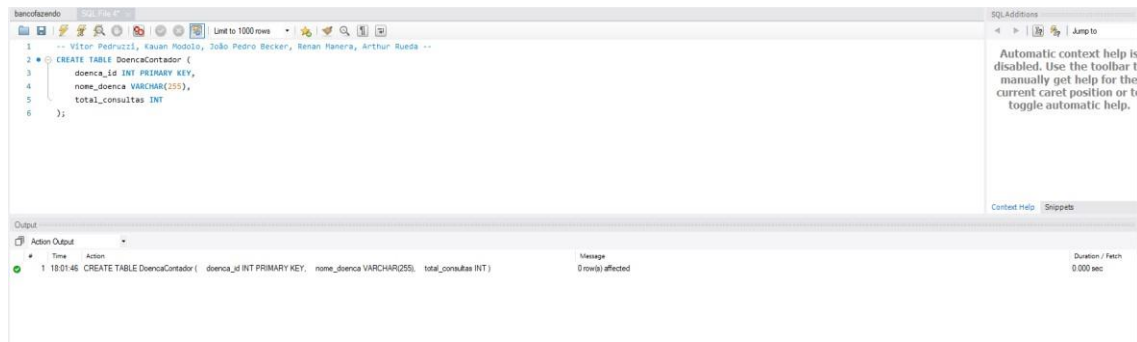


Gatilho:

1º:

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

```
CREATE TABLE DoencaContador (  
doenca_id INT PRIMARY KEY,  
nome_doenca VARCHAR(255),  
total_consultas INT  
);
```



2º:

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

DELIMITER //

CREATE TRIGGER AtualizaDoencaContador

AFTER INSERT ON consultas

FOR EACH ROW

BEGIN

IF EXISTS (SELECT 1 FROM DoencaContador WHERE doenca_id =
NEW.doenca_id) THEN

UPDATE DoencaContador

SET total_consultas = total_consultas + 1

WHERE doenca_id = NEW.doenca_id;

ELSE

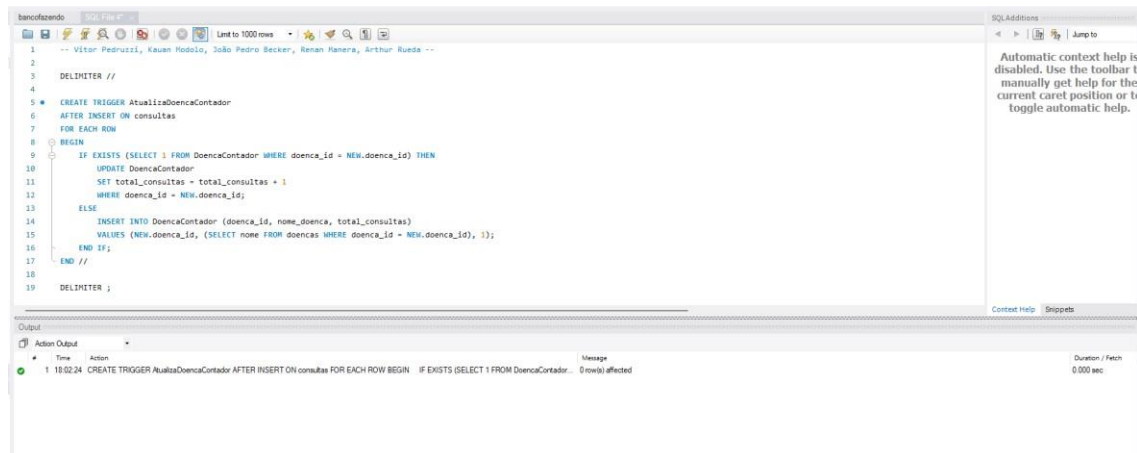
INSERT INTO DoencaContador (doenca_id, nome_doenca,
total_consultas)

VALUES (NEW.doenca_id, (SELECT nome FROM doencas WHERE
doenca_id = NEW.doenca_id), 1);

END IF;

END //

DELIMITER ;



3º:

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

-- Inserir dados na tabela 'doencas'

INSERT INTO doencas (doenca_id, nome) VALUES

(21, 'Asma'),

(22, 'Febre Amarela'),

(23, 'Hipertensao');

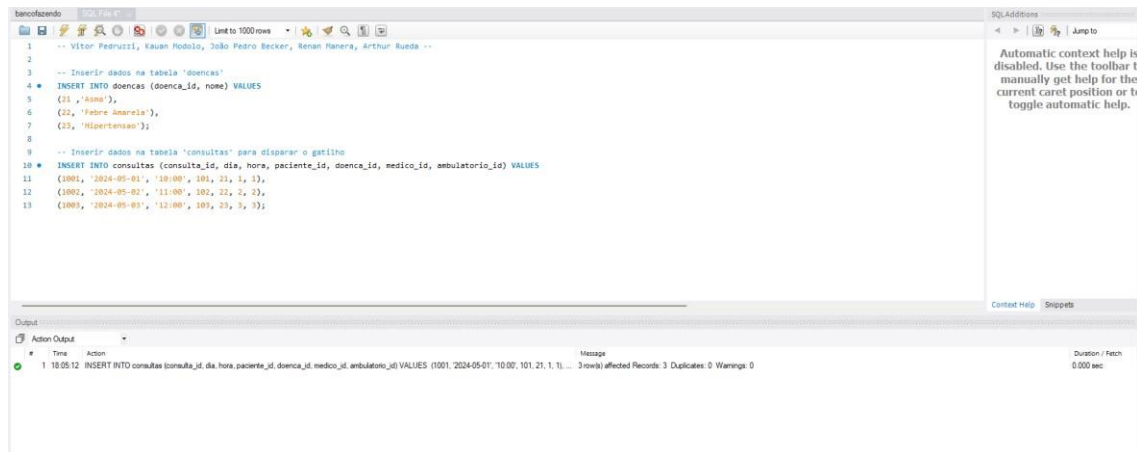
-- Inserir dados na tabela 'consultas' para disparar o gatilho

INSERT INTO consultas (consulta_id, dia, hora, paciente_id, doenca_id, medico_id, ambulatorio_id) VALUES

(1001, '2024-05-01', '10:00', 101, 21, 201, 1), (1002,

'2024-05-02', '11:00', 102, 22, 202, 2),

(1003, '2024-05-03', '12:00', 103, 23, 203, 3);



4°:

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

SELECT

nome_doenca,

total_consultas

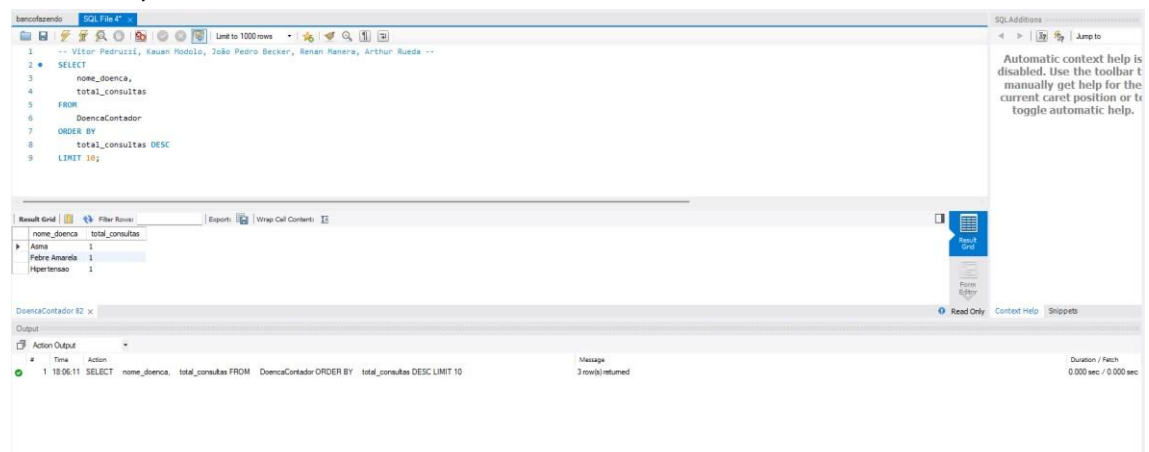
FROM

DoencaContador ORDER

BY

total_consultas DESC

LIMIT 10;



Transação:

1º:

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --

-- Iniciar a transação

START TRANSACTION;

-- Consulta para identificar as doenças mais comuns nas consultas

SELECT

d.nome AS nome_doenca,

COUNT(c.consulta_id) AS total_consultas

FROM

consultas c

JOIN doencas d ON c.doenca_id = d.doenca_id GROUP

BY

d.nome ORDER

BY

total_consultas DESC

LIMIT 10;

-- Finalizar a transação

COMMIT;

The screenshot displays the SQL Developer interface. The top pane shows a SQL script with the following content:

```
-- Vitor Pedruzzi, Kauan Modolo, João Pedro Becker, Renan Manera, Arthur Rueda --
1 -- Iniciar a transação
2 START TRANSACTION;
3
4 -- Consulta para identificar as doenças mais comuns nas consultas
5
6 SELECT
7     d.nome AS nome_doenca,
8     COUNT(c.consulta_id) AS total_consultas
9 FROM
10     consultas c
11 JOIN doencas d ON c.doenca_id = d.doenca_id
12 GROUP BY
13     d.nome
14 ORDER BY
15     total_consultas DESC
16 LIMIT 10;
17
18 -- Finalizar a transação
19 COMMIT;
```

The bottom pane shows the execution results. The top part is a table with the following data:

nome_doenca	total_consultas
hepatite	65
coro	63
dengue	62
hipertensao	58
pneumonia	58
asma	57
chikungunya	56
anemia	54
...	...

The bottom part of the bottom pane shows the execution log with the following entries:

Time	Action	Message	Duration / Fetch
1 10:06:11	SELECT nome_doenca, total_consultas FROM DoencaContador ORDER BY total_consultas DESC LIMIT 10	3 row(s) returned	0.000 sec / 0.000 sec
2 10:08:28	START TRANSACTION	0 row(s) affected	0.000 sec
3 10:08:29	SELECT d.nome AS nome_doenca, COUNT(c.consulta_id) AS total_consultas FROM consultas c JOIN doencas d ON c.doenca_id = d.doenca_id	10 row(s) returned	0.000 sec / 0.000 sec
4 10:08:29	COMMIT	0 row(s) affected	0.000 sec

2°:

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker, Arthur Rueda --
DELIMITER \$\$

CREATE PROCEDURE DoencasMaisComuns()

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION

BEGIN

-- Em caso de erro, desfazer todas as operações

ROLLBACK;

END;

-- Iniciar a transação

START TRANSACTION;

-- Inserir dados na tabela 'doencas'

INSERT INTO doencas (doenca_id, nome) VALUES

(24, 'Diabetes'),

(25, 'Gripe');

-- Inserir dados na tabela 'consultas' para disparar o gatilho

INSERT INTO consultas (consulta_id, dia, hora, paciente_id, doenca_id, medico_id,
ambulatorio_id) VALUES

(1004, '2024-05-04', '09:00:00', 104, 24, 204, 4),

(1005, '2024-05-05', '10:00:00', 105, 25, 205, 5);

-- Consulta para identificar as doenças mais comuns nas consultas

SELECT

d.nome AS nome_doenca,


```

COUNT(c.consulta_id) AS total_consultas

FROM

consultas c

JOIN doencas d ON c.doenca_id = d.doenca_id

GROUP BY

d.nome ORDER BY

total_consultas DESC

LIMIT 10;

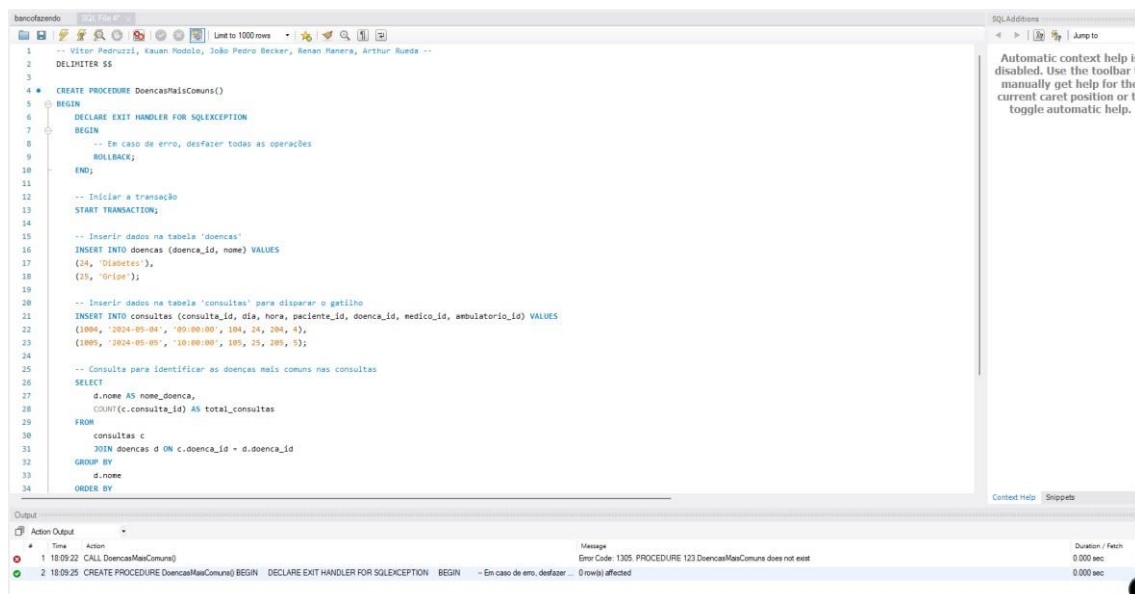
-- Finalizar a transação

COMMIT;

END$$

DELIMITER ;

```



3°:

-- Vitor Pedruzzi, Renan Manera, Kauan Modolo, João Pedro Becker,

Arthur Rueda --
CALL DoencasMaisComuns();

