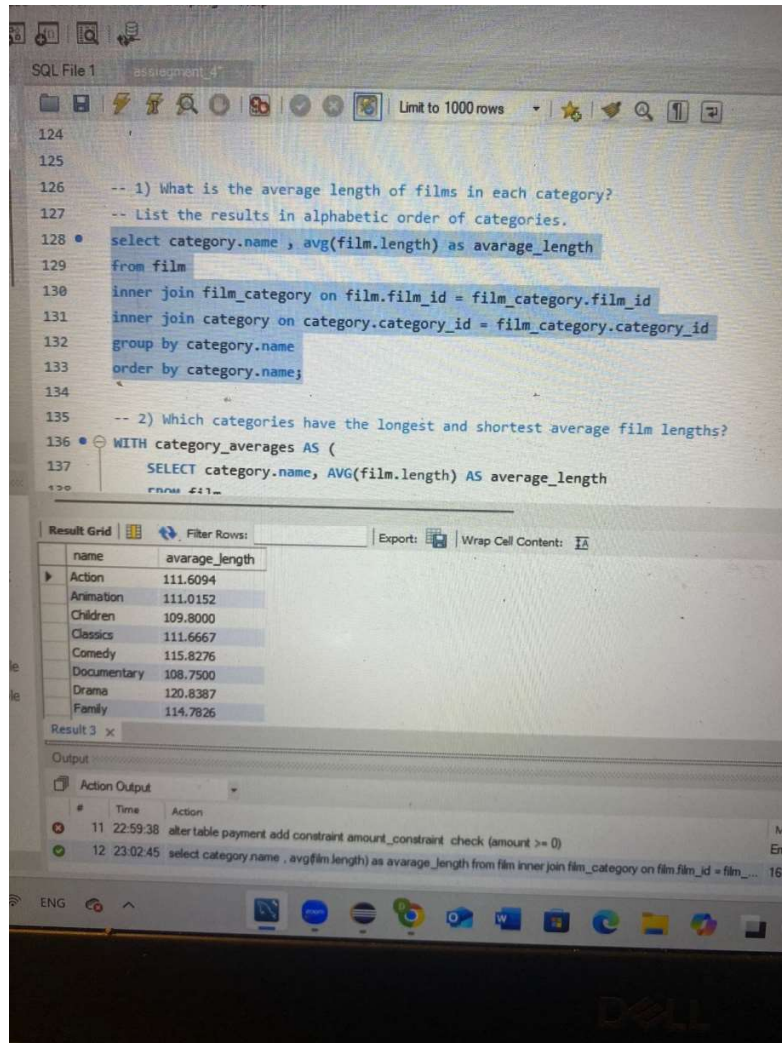


David Schwartzman

10/30/2025

DB Assignment 4

1) What is the average length of films in each category? List the results in alphabetic order of categories.



The screenshot shows a SQL IDE window with a query editor and a results grid. The query editor contains the following SQL code:

```
124
125
126 -- 1) What is the average length of films in each category?
127 -- List the results in alphabetic order of categories.
128 select category.name , avg(film.length) as avarage_length
129 from film
130 inner join film_category on film.film_id = film_category.film_id
131 inner join category on category.category_id = film_category.category_id
132 group by category.name
133 order by category.name;
134
135 -- 2) Which categories have the longest and shortest average film lengths?
136 WITH category_averages AS (
137     SELECT category.name, AVG(film.length) AS average_length
138     from film
139     inner join film_category on film.film_id = film_category.film_id
140     inner join category on category.category_id = film_category.category_id
141     group by category.name
142 )
143 select * from category_averages
144 order by average_length desc;
```

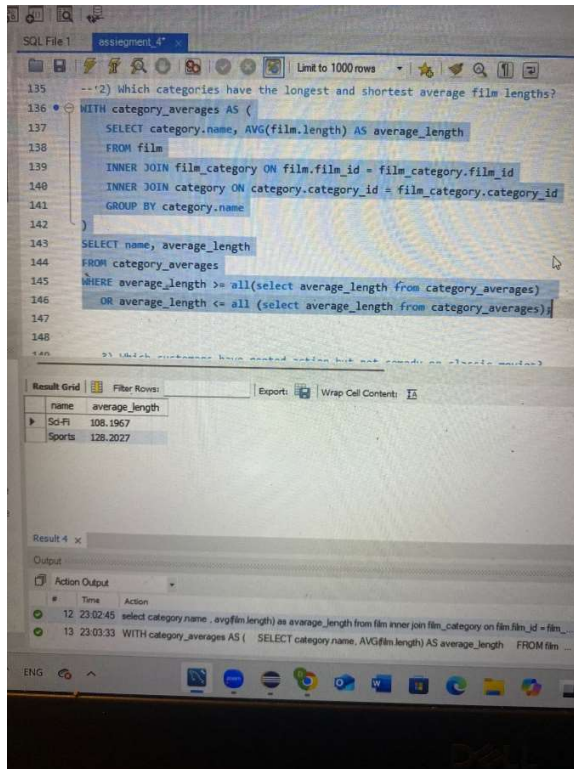
The results grid shows the following data:

name	avarage_length
Action	111.6094
Animation	111.0152
Children	109.8000
Classics	111.6667
Comedy	115.8276
Documentary	108.7500
Drama	120.8387
Family	114.7826

The bottom of the screenshot shows the 'Output' pane with a log of database actions, including a successful query execution at 23:02:45.

Explanation: This query calculates the average film length for each category. It joins the film, film_category, and category tables to relate films with their respective categories. The AVG function computes the average length, and the results are grouped by category name. Finally, the results are sorted in alphabetical order of categories.

2) Which categories have the longest and shortest average film lengths?



```
135 --2) Which categories have the longest and shortest average film lengths?
136 WITH category_averages AS (
137     SELECT category.name, AVG(film.length) AS average_length
138     FROM film
139     INNER JOIN film_category ON film.film_id = film_category.film_id
140     INNER JOIN category ON category.category_id = film_category.category_id
141     GROUP BY category.name
142 )
143 SELECT name, average_length
144 FROM category_averages
145 WHERE average_length >= all(select average_length from category_averages)
146 OR average_length <= all (select average_length from category_averages)
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
```

name	average_length
Sci-Fi	108.1967
Sports	128.2027

Result 4 x

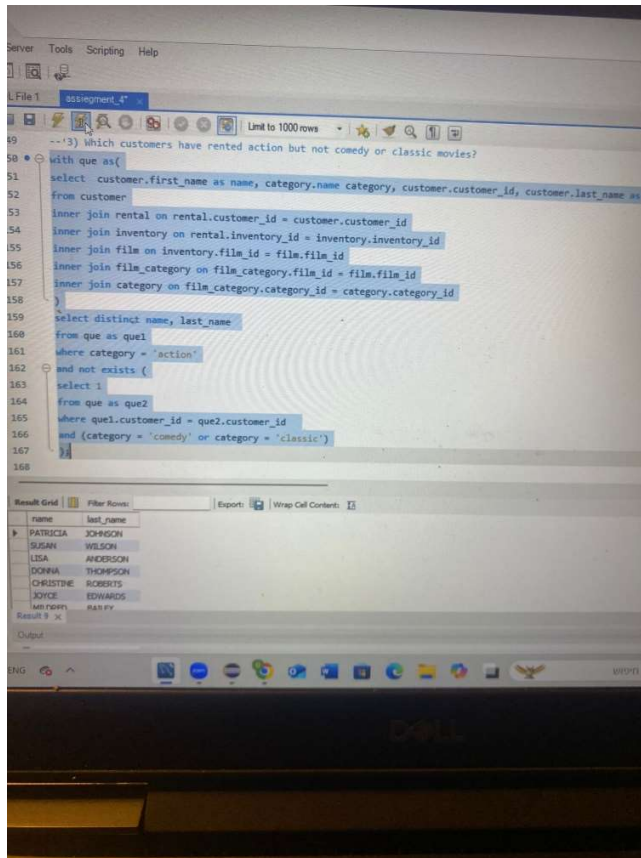
Output

Action Output

#	Time	Action
12	23:02:45	select category name , avg(film.length) as average_length from film inner join film_category on film.film_id = film...
13	23:03:33	WITH category_averages AS (SELECT category name, AVG(film.length) AS average_length FROM film ...

Explanation: This query first creates a common table expression (CTE) named `category_averages` that computes the average length of films for each category. The outer query selects the categories that have the longest and shortest average lengths by comparing each category's average length against all others using the `ALL` keyword.

3) Which customers have rented action but not comedy or classic movies?



Explanation: This query identifies customers who have rented action films but have not rented any comedy or classic films. It uses a CTE (que) to join several tables and gather relevant data. The main query then selects distinct names and last names of customers from que who rented action films and excludes any customer who has rented films in the comedy or classic categories by checking with the NOT EXISTS clause.

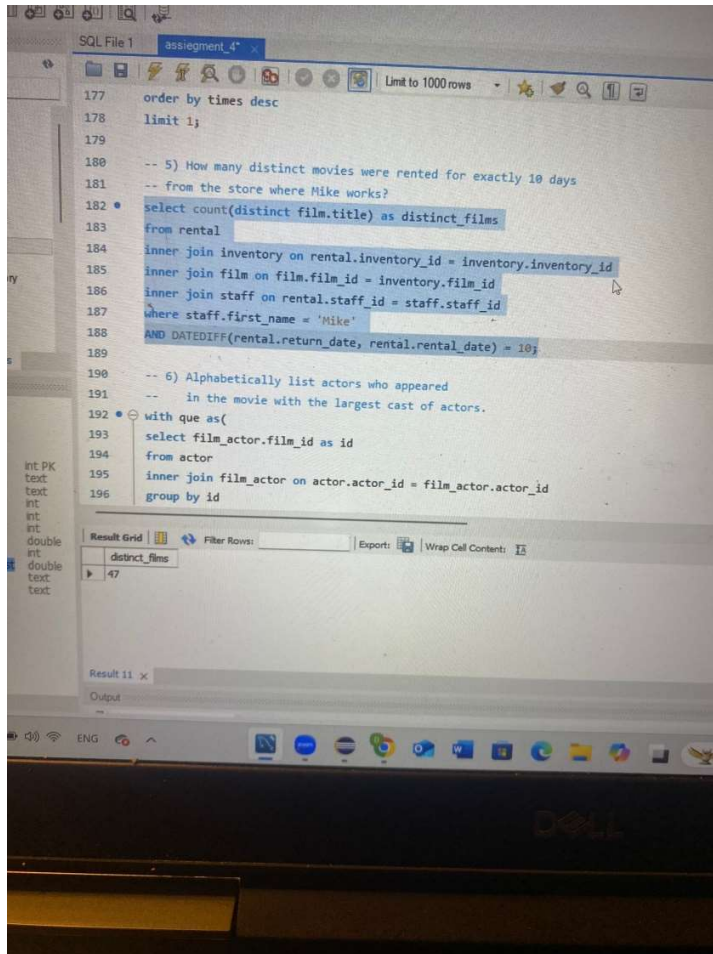
4) Which actor has appeared in the most English-language movies?

```
164 from que as que2
165 where que1.customer_id = que2.customer_id
166 and (category = 'comedy' or category = 'classic')
167 );
168
169 -- 4) Which actor has appeared in the most English-language movies?
170 • select actor.first_name, actor.last_name, count(language.name) as times
171 from actor
172 inner join film_actor on actor.actor_id = film_actor.actor_id
173 inner join film on film_actor.film_id = film.film_id
174 inner join language on film.language_id = language.language_id
175 where language.name = 'English'
176 group by actor.first_name, actor.last_name
177 order by times desc
178 limit 1;
179
180 -- 5) How many distinct movies were rented for exactly 10 days
181 -- from the store where Mike works?
182 • select count(distinct film.title) as distinct_files
183 from rental
```

first_name	last_name	times
SUSAN	DAVIS	54

Explanation: This query finds the actor who has appeared in the most English-language films. It joins the actor, film_actor, film, and language tables. The WHERE clause filters for films with the language 'English', and the COUNT function counts the occurrences of each actor in these films. The results are grouped by actor and ordered by the count in descending order, limiting the output to the top actor.

5) How many distinct movies were rented for exactly 10 days from the store where Mike works?



Explanation: This query counts the distinct titles of movies that were rented for exactly 10 days from the store where Mike works. It joins the rental, inventory, film, and staff tables. The WHERE clause filters for rentals processed by staff named Mike and checks that the difference between the return and rental dates is exactly 10 days. The COUNT(DISTINCT film.title) ensures that only unique movie titles are counted.

6) Alphabetically list actors who appeared in the movie with the largest cast of actors:


```
185 inner join film on film.film_id = inventory.film_id
186 inner join staff on rental.staff_id = staff.staff_id
187 where staff.first_name = 'Mike'
188 AND DATEDIFF(rental.return_date, rental.rental_date) = 10;
189
190 -- 6) Alphabetically list actors who appeared
191 -- in the movie with the largest cast of actors.
192 with que as(
193 select film_actor.film_id as id
194 from actor
195 inner join film_actor on actor.actor_id = film_actor.actor_id
196 group by id
197 order by count(film_actor.actor_id) desc limit 1
198 )
199 select actor.first_name as name, actor.last_name
200 from actor
201 inner join film_actor on film_actor.actor_id = actor.actor_id
202 inner join que on que.id = film_actor.film_id
203 order by name;
204
```

Result Grid

name	last_name
BURT	POSEY
CAMERON	ZELLWEGER
CHRISTIAN	HEESON
FAY	WINSLET
JAYME	NOLTE
JULIA	BARRYMORE
WETA	WILSON

Explanation: This query retrieves actors from the film that has the largest cast. It first creates a CTE (que) that identifies the film ID with the highest number of actors by counting the film_actor entries. The outer query then selects the first and last names of actors who appeared in that film by joining the actor and film_actor tables with the film ID from the CTE. Finally, the results are ordered alphabetically by the actor's first name.

ERD Diagram:

