

Lab01.ipynb

```
1  %% md
2  ### Lab 01 - Assignment - By - Vinay S
3  %% md
4  ### Q1
5  %% md
6  ### Load Required Libraries
7  %%
8  import mysql.connector
9  import pandas as pd
10 from streamlit import columns
11
12 %% md
13 ### Connect to DB using Mysql-connector-python package
14
15 %%
16 # Establish a connection to the database
17 connection = mysql.connector.connect(
18     host='localhost',      # e.g., 'localhost'
19     user='root',          # e.g., 'root'
20     password='',          # e.g., 'password'
21     database='e_commerce' # e.g., 'test_db'
22 )
23
24 %%
25 db_name = 'e_commerce'
26
27 %%
28 #####
29 # Drop the DB
30
31 myquery = f"""DROP DATABASE IF EXISTS {db_name};"""
32 %%
33 cursor = connection.cursor()
34
35 %%
36 cursor.execute(myquery)
37 %%
38 connection.commit()
39 cursor.close()
40 %%
41 connection.close()
42 %%
43 # Establish a connection to the database
44 connection = mysql.connector.connect(
45     host='localhost',      # e.g., 'localhost'
46     user='root',          # e.g., 'root'
47     password='',          # e.g., 'password'
48     database='e_commerce' # e.g., 'test_db'
49 )
50
51 %%
52 db_name = 'e_commerce'
53
54 %%
55 myquery = f"""CREATE DATABASE IF NOT EXISTS {db_name}"""
56 %%
57 cursor = connection.cursor()
58 cursor.execute(myquery)
59
60 %%
61 connection.commit()
62 cursor.close()
63 %%
64 connection.close()
65
66 %%
67 db_name = 'e_commerce'
68
69 # Establish a connection to the database
70 connection = mysql.connector.connect(
71     host='localhost',      # e.g., 'localhost'
72     user='root',          # e.g., 'root'
73     password='',          # e.g., 'password'
74     database=db_name      # e.g., 'test_db'
75 )
76 %%
77 myquery = """
78 CREATE TABLE supplier (
79     SUPP_ID INT PRIMARY KEY,
80     SUPP_NAME VARCHAR(50),
81     SUPP_CITY VARCHAR(50),
82     SUPP_PHONE VARCHAR(10)
83 );
84
85 CREATE TABLE customer (
86     CUS_ID INT NOT NULL,
87     CUS_NAME VARCHAR(20) NULL DEFAULT NULL,
88     CUS_PHONE VARCHAR(10),
89     CUS_CITY VARCHAR(30),
90     CUS_GENDER CHAR,
91     PRIMARY KEY (CUS_ID)
92 );
93
94 CREATE TABLE category (
```

```
95     CAT_ID INT NOT NULL,
96     CAT_NAME VARCHAR(20) NULL DEFAULT NULL,
97     PRIMARY KEY (CAT_ID)
98 );
99
100 CREATE TABLE product (
101     PRO_ID INT NOT NULL,
102     PRO_NAME VARCHAR(20) NULL DEFAULT NULL,
103     PRO_DESC VARCHAR(60) NULL DEFAULT NULL,
104     CAT_ID INT NOT NULL,
105     PRIMARY KEY (PRO_ID),
106     FOREIGN KEY (CAT_ID) REFERENCES category (CAT_ID)
107 );
108
109 CREATE TABLE product_details (
110     PROD_ID INT NOT NULL,
111     PRO_ID INT NOT NULL,
112     SUPP_ID INT NOT NULL,
113     PROD_PRICE INT NOT NULL,
114     PRIMARY KEY (PROD_ID),
115     FOREIGN KEY (PRO_ID) REFERENCES product (PRO_ID),
116     FOREIGN KEY (SUPP_ID) REFERENCES supplier (SUPP_ID)
117 );
118
119 CREATE TABLE orders (
120     ORD_ID INT NOT NULL,
121     ORD_AMOUNT INT NOT NULL,
122     ORD_DATE DATE,
123     CUS_ID INT NOT NULL,
124     PROD_ID INT NOT NULL,
125     PRIMARY KEY (ORD_ID),
126     FOREIGN KEY (CUS_ID) REFERENCES customer (CUS_ID),
127     FOREIGN KEY (PROD_ID) REFERENCES product_details (PROD_ID)
128 );
129
130 CREATE TABLE rating (
131     RAT_ID INT NOT NULL,
132     CUS_ID INT NOT NULL,
133     SUPP_ID INT NOT NULL,
134     RAT_RATSTARS INT NOT NULL,
135     PRIMARY KEY (RAT_ID),
136     FOREIGN KEY (SUPP_ID) REFERENCES supplier (SUPP_ID),
137     FOREIGN KEY (CUS_ID) REFERENCES customer (CUS_ID)
138 );
139 """
140
141 ###
142 cursor = connection.cursor()
143
144 ###
145 cursor.execute(myquery)
146
147 ###
148 #connection.commit()
149 cursor.close()
150 connection.close()
151
152 ###
153
154 ### md
155 ### Q2
156 ###
157 ### Inserting Values into Supplier Table
158 ###
159 db_name = 'e_commerce'
160
161 # Establish a connection to the database
162 connection = mysql.connector.connect(
163     host='localhost', # e.g., 'localhost'
164     user='root', # e.g., 'root'
165     password='', # e.g., 'password'
166     database=db_name # e.g., 'test_db'
167 )
168
169 ###
170 myquery = """
171 INSERT INTO supplier (SUPP_ID, SUPP_NAME, SUPP_CITY, SUPP_PHONE) VALUES
172 (1, 'Rajesh Retails', 'Delhi', '1234567890'),
173 (2, 'Appario Ltd.', 'Mumbai', '258963147032'),
174 (3, 'Knome products', 'Bangalore', '9785462315'),
175 (4, 'Bansal Retails', 'Kochi', '8975463285'),
176 (5, 'Mittal Ltd.', 'Lucknow', '7898456532');
177 """
178
179 ###
180 cursor = connection.cursor()
181
182 ###
183 cursor.execute(myquery)
184 connection.commit()
185
186 ###
187 cursor.close()
188 connection.close()
189
190 ###
191 ### Inserting Values into Customer Table
```

```
192 #%%
193 db_name = 'e_commerce'
194
195 # Establish a connection to the database
196 connection = mysql.connector.connect(
197     host='localhost', # e.g., 'localhost'
198     user='root', # e.g., 'root'
199     password='', # e.g., 'password'
200     database=db_name # e.g., 'test_db'
201 )
202
203 #%%
204 myquery = """
205 INSERT INTO customer (CUS_ID, CUS_NAME, CUS_PHONE, CUS_CITY, CUS_GENDER) VALUES
206 (1, 'AAKASH', 9999999999, 'DELHI', 'M'),
207 (2, 'AMAN', 9785463215, 'NOIDA', 'M'),
208 (3, 'NEHA', 9999999998, 'MUMBAI', 'F'),
209 (4, 'MEGHA', 9994562399, 'KOLKATA', 'F'),
210 (5, 'PULKIT', 7895999999, 'LUCKNOW', 'M');
211 """
212 #%%
213 cursor = connection.cursor()
214
215 #%%
216 cursor.execute(myquery)
217 connection.commit()
218
219 #%%
220 cursor.close()
221 connection.close()
222
223 #%%
224 ### Inserting Values into Category Table
225 #%%
226 db_name = 'e_commerce'
227
228 # Establish a connection to the database
229 connection = mysql.connector.connect(
230     host='localhost', # e.g., 'localhost'
231     user='root', # e.g., 'root'
232     password='', # e.g., 'password'
233     database=db_name # e.g., 'test_db'
234 )
235
236 #%%
237 myquery = """
238 INSERT INTO category (CAT_ID, CAT_NAME) VALUES
239 (1, 'BOOKS'),
240 (2, 'GAMES'),
241 (3, 'GROCERIES'),
242 (4, 'ELECTRONICS'),
243 (5, 'CLOTHES');
244 """
245
246 #%%
247 cursor = connection.cursor()
248
249 #%%
250 cursor.execute(myquery)
251 connection.commit()
252
253 #%%
254 cursor.close()
255 connection.close()
256
257 #%%
258 ### Inserting Values into Product Table
259 #%%
260 db_name = 'e_commerce'
261
262 # Establish a connection to the database
263 connection = mysql.connector.connect(
264     host='localhost', # e.g., 'localhost'
265     user='root', # e.g., 'root'
266     password='', # e.g., 'password'
267     database=db_name # e.g., 'test_db'
268 )
269
270 #%%
271 myquery = """
272 INSERT INTO product (PRO_ID, PRO_NAME, PRO_DESC, CAT_ID) VALUES
273 (1, 'GTA V', 'DFJDJFDJFDJFJF', 2),
274 (2, 'TSHIRT', 'DFDFJDFJDKFD', 5),
275 (3, 'ROG LAPTOP', 'DFNTTNTNTERND', 4),
276 (4, 'OATS', 'REURENTBTOTH', 3),
277 (5, 'HARRY POTTER', 'NBEMCTHTJTH', 1);
278 """
279
280 #%%
281 cursor = connection.cursor()
282
283 #%%
284 cursor.execute(myquery)
285 connection.commit()
286
287 #%%
288 cursor.close()
```

```
289 connection.close()
290
291 ###
292 ### Inserting Values into Product_Details Table
293 ###
294 db_name = 'e_commerce'
295
296 # Establish a connection to the database
297 connection = mysql.connector.connect(
298     host='localhost',      # e.g., 'localhost'
299     user='root',          # e.g., 'root'
300     password='',          # e.g., 'password'
301     database=db_name      # e.g., 'test_db'
302 )
303
304 ###
305 myquery="""
306 INSERT INTO product_details (PROD_ID, PRO_ID, SUPP_ID, PROD_PRICE) VALUES
307 (1, 1, 2, 1500),
308 (2, 3, 5, 30000),
309 (3, 5, 1, 3000),
310 (4, 2, 3, 2500),
311 (5, 4, 1, 1000);
312 """
313
314 ###
315 cursor = connection.cursor()
316
317 ###
318 cursor.execute(myquery)
319 connection.commit()
320
321 ###
322 cursor.close()
323 connection.close()
324
325 ###
326 ### Inserting Values into Orders Table
327 ###
328 db_name = 'e_commerce'
329
330 # Establish a connection to the database
331 connection = mysql.connector.connect(
332     host='localhost',      # e.g., 'localhost'
333     user='root',          # e.g., 'root'
334     password='',          # e.g., 'password'
335     database=db_name      # e.g., 'test_db'
336 )
337
338 ###
339 myquery = """
340 INSERT INTO orders (ORD_ID, ORD_AMOUNT, ORD_DATE, CUS_ID, PROD_ID) VALUES
341 (20, 1500, '2021-10-12', 3, 5),
342 (25, 30500, '2021-09-16', 5, 2),
343 (26, 2000, '2021-10-05', 1, 1),
344 (30, 3500, '2021-08-16', 4, 3),
345 (50, 2000, '2021-10-06', 2, 1);
346 """
347
348 ###
349 cursor = connection.cursor()
350
351 ###
352 cursor.execute(myquery)
353 connection.commit()
354
355 ###
356 cursor.close()
357 connection.close()
358
359 ###
360 ### Inserting Values into Rating Table
361 ###
362 db_name = 'e_commerce'
363
364 # Establish a connection to the database
365 connection = mysql.connector.connect(
366     host='localhost',      # e.g., 'localhost'
367     user='root',          # e.g., 'root'
368     password='',          # e.g., 'password'
369     database=db_name      # e.g., 'test_db'
370 )
371
372 ###
373 myquery = """
374 INSERT INTO rating (RAT_ID, CUS_ID, SUPP_ID, RAT_RATSTARS) VALUES
375 (1, 2, 2, 4),
376 (2, 3, 4, 3),
377 (3, 5, 1, 5),
378 (4, 1, 3, 2),
379 (5, 4, 5, 4);
380 """
381
382 ###
383 cursor = connection.cursor()
384
385 ###
```

```
386 cursor.execute(myquery)
387 connection.commit()
388
389 ###
390 cursor.close()
391 connection.close()
392
393 ###
394
395 ### md
396 ### Q3
397 ###
398 ### Q3) Display the number of the customer group by their genders who have placed any order of amount greater than or equal to Rs.3000
399 ###
400 db_name = 'e_commerce'
401
402 # Establish a connection to the database
403 connection = mysql.connector.connect(
404     host='localhost',      # e.g., 'localhost'
405     user='root',          # e.g., 'root'
406     password='',          # e.g., 'password'
407     database=db_name     # e.g., 'test_db'
408 )
409
410 ###
411 myquery = """
412 SELECT CUS_GENDER, COUNT(*) AS num_customers
413 FROM customer
414 JOIN orders ON customer.CUS_ID = orders.CUS_ID
415 WHERE ORD_AMOUNT >= 3000
416 GROUP BY CUS_GENDER;
417 """
418
419 ###
420 cursor = connection.cursor()
421
422 ###
423 cursor.execute(myquery)
424
425 ###
426 output = cursor.fetchall()
427
428 ###
429 df = pd.DataFrame(output, columns=['CUS_GENDER', 'num_customers'])
430
431 ###
432 df.head()
433
434 ###
435
436 ### md
437 ### Q4
438 ###
439 ### Q4) Display all the order along with product name ordered by a customer having Customer_Id=2;
440
441 ###
442 myquery = """
443 SELECT orders.ORD_ID, orders.ORD_AMOUNT, orders.ORD_DATE, product.PRO_NAME
444 FROM orders
445 JOIN product_details ON orders.PROD_ID = product_details.PROD_ID
446 JOIN product ON product_details.PRO_ID = product.PRO_ID
447 WHERE orders.CUS_ID = 2;
448 """
449
450 ###
451 cursor = connection.cursor()
452 cursor.execute(myquery)
453
454 ###
455 output = cursor.fetchall()
456 print(output)
457
458 ###
459 df = pd.DataFrame(output, columns=['orders.ORD_ID', 'orders.ORD_AMOUNT', 'orders.ORD_DATE', 'product.PRO_NAME'])
460
461 df.head()
462
463 ###
464 ### Alternative Query as provided by Faculty
465 myquery = """
466 select `orders`.*, product.pro_name
467 from `orders`,product_details,product
468 where `orders`.cus_id=2 and `orders`.prod_id=product_details.prod_id and product_details.pro_id=product.pro_id;
469 """
470
471 ###
472 cursor = connection.cursor()
473 cursor.execute(myquery)
474 output = cursor.fetchall()
475
476 ###
477 print(output)
478
479 ###
480 df = pd.DataFrame(output, columns=['orders.ORD_ID', 'orders.ORD_AMOUNT', 'orders.ORD_DATE', "product.cust_id", 'product.prod_id', 'product.pro_name'])
481 df.head()
482
```

```
483 ###
484
485 ### md
486 ### Q5
487
488 ###
489 ### Q5) Display the Supplier details who can supply more than one product.
490
491 ###
492 myquery = """
493 SELECT SUPP_ID, SUPP_NAME
494 FROM supplier
495 WHERE SUPP_ID IN (
496     SELECT SUPP_ID
497     FROM product_details
498     GROUP BY SUPP_ID
499     HAVING COUNT(*) > 1
500 );
501 """
502
503 ###
504 cursor = connection.cursor()
505 cursor.execute(myquery)
506 output = cursor.fetchall()
507
508 ###
509 print(output)
510
511 ###
512 df = pd.DataFrame(output, columns=['SUPP_ID', 'SUPP_NAME'])
513 df.head()
514
515 ###
516 # Alternative Query provided by the Faculty
517 myquery = """
518 select supplier.*
519 from supplier,product_details
520 where supplier.supp_id in
521 (select product_details.supp_id from product_details
522 group by product_details.supp_id having count(product_details.supp_id)>1)
523 group by supplier.supp_id;
524 """
525
526 ###
527 cursor = connection.cursor()
528 cursor.execute(myquery)
529 output = cursor.fetchall()
530
531 ###
532 print(output)
533
534 ###
535 df = pd.DataFrame(output, columns=['SUPP_ID', 'SUPP_NAME', 'SUPP_CITY', 'SUPP_PHONE'])
536 df
537
538 ###
539
540 ### md
541 ### Q6
542
543 ###
544 ### Q6) Find the category of the product whose order amount is minimum.
545
546 ###
547 myquery = """
548 SELECT category.CAT_NAME
549 FROM category
550 JOIN product ON category.CAT_ID = product.CAT_ID
551 JOIN product_details ON product.PROD_ID = product_details.PROD_ID
552 JOIN orders ON product_details.PROD_ID = orders.PROD_ID
553 WHERE orders.ORD_AMOUNT = (
554     SELECT MIN(ORD_AMOUNT)
555     FROM orders
556 );
557 """
558
559 ###
560 cursor = connection.cursor()
561 cursor.execute(myquery)
562 output = cursor.fetchall()
563
564 ###
565 print(output)
566
567 ###
568 df = pd.DataFrame(output, columns=['category.CAT_NAME'])
569 df.head()
570
571 ###
572 # Alternative Query Provided by the Faculty
573 myquery = """
574 SELECT category.cat_id, category.cat_name, MIN(orders.ord_amount) AS min_order_amount
575 FROM orders
576 INNER JOIN product_details ON orders.prod_id = product_details.prod_id
577 INNER JOIN product ON product.pro_id = product_details.pro_id
578 INNER JOIN category ON category.cat_id = product.cat_id
579 GROUP BY category.cat_id, category.cat_name;
```

```
580 """
581
582 ###
583 cursor = connection.cursor()
584 cursor.execute(myquery)
585 output = cursor.fetchall()
586
587 ###
588 print(output)
589
590 ###
591 # Another Query from Faculty
592 myquery = """
593 select category.*
594 from `orders` inner join product_details on `orders`.prod_id=product_details.prod_id
595 inner join product on product.pro_id=product_details.pro_id
596 inner join category on category.cat_id=product.cat_id having min(`orders`.ord_amount);
597 """
598
599 ###
600 cursor = connection.cursor()
601 cursor.execute(myquery)
602 output = cursor.fetchall()
603
604 ###
605 print(output)
606
607 ###
608 df = pd.DataFrame(output, columns=['category.cat_id','category.cat_name'])
609
610 ###
611 df.head()
612
613 ###
614
615 ### md
616 ### Q7
617
618 ###
619 ### Q7) Display the Id and Name of the Product ordered after "2021-10-05".
620
621 ###
622 myquery = """
623 SELECT product.PRO_ID, product.PRO_NAME
624 FROM product
625 JOIN product_details ON product.PRO_ID = product_details.PRO_ID
626 JOIN orders ON product_details.PROD_ID = orders.PROD_ID
627 WHERE orders.ORD_DATE > '2021-10-05';
628 """
629
630 ###
631 cursor = connection.cursor()
632 cursor.execute(myquery)
633 output = cursor.fetchall()
634
635 ###
636 print(output)
637
638 ###
639 df = pd.DataFrame(output, columns=['product.PRO_ID','product.PRO_NAME'])
640 df.head()
641
642 ###
643
644 ### md
645 ### Q8
646
647 ### Q8) Print the top 3 supplier name and id and rating on the basis of their rating along with the customer name who has given the rating.
648
649 ###
650 myquery = """
651 SELECT supplier.SUPP_ID, supplier.SUPP_NAME, AVG(rating.RAT_RATSTARS) AS avg_rating, customer.CUS_NAME
652 FROM supplier
653 JOIN rating ON supplier.SUPP_ID = rating.SUPP_ID
654 JOIN customer ON rating.CUS_ID = customer.CUS_ID
655 GROUP BY supplier.SUPP_ID, supplier.SUPP_NAME
656 ORDER BY avg_rating DESC
657 LIMIT 3;
658 """
659
660 ###
661 cursor = connection.cursor()
662 cursor.execute(myquery)
663 output = cursor.fetchall()
664
665 ###
666 print(output)
667
668 ###
669 df = pd.DataFrame(output, columns=['SUPPLIER_ID', 'SUPPLIER_NAME', 'AVG_Rating', 'CUSTOMER_NAME'])
670 df
671
672 ###
673 # Alternative Query as provided by Faculty
674 myquery = """
675 select supplier.supp_id,supplier.supp_name,customer.cus_name,rating.rat_ratstars
676 from rating inner join supplier on rating.supp_id=supplier.supp_id
```

```
677 inner join customer on rating.cus_id=customer.cus_id order by rating.rat_ratstars desc limit 3;
678 """
679
680 ###
681 cursor = connection.cursor()
682 cursor.execute(myquery)
683 output = cursor.fetchall()
684
685 ###
686 print(output)
687
688 ###
689 df = pd.DataFrame(output, columns=['Supplier_ID', 'Supplier_Name', 'CUS_NAME', 'Rating'])
690 df.head()
691
692 ###
693
694 ### md
695 ### Q9
696
697 ###
698 myquery = """
699 SELECT CUS_NAME, CUS_GENDER
700 FROM customer
701 WHERE CUS_NAME LIKE 'A%' OR CUS_NAME LIKE '%A';
702 """
703
704 ###
705 cursor = connection.cursor()
706 cursor.execute(myquery)
707 output = cursor.fetchall()
708
709 ###
710 print(output)
711
712 ###
713 df = pd.DataFrame(output, columns=['CUS_NAME', 'CUS_GENDER'])
714 df.head()
715
716 ###
717
718 ### md
719 ### Q10
720
721 ###
722 ### Q10) Display the total order amount of the male customers
723
724 ###
725 myquery = """
726 SELECT SUM(orders.ORD_AMOUNT) AS total_order_amount
727 FROM orders
728 JOIN customer ON orders.CUS_ID = customer.CUS_ID
729 WHERE customer.CUS_GENDER = 'M';
730 """
731
732 ###
733 cursor = connection.cursor()
734 cursor.execute(myquery)
735 output = cursor.fetchall()
736
737 ###
738 print(output)
739
740 ###
741 df = pd.DataFrame(output, columns=['Amount'])
742 df.head()
743
744 ###
745
746 ### md
747 ### Q11
748
749 ###
750 ### Q11) Display all the Customers left outer join with the orders
751
752 ###
753 myquery = """
754 SELECT
755 customer.CUS_ID, customer.CUS_NAME, customer.CUS_PHONE, customer.CUS_CITY, customer.CUS_GENDER, orders.ORD_ID, orders.ORD_AMOUNT, orders.ORD_DATE
756 FROM customer
757 LEFT JOIN orders ON customer.CUS_ID = orders.CUS_ID;
758 """
759
760 ###
761 cursor = connection.cursor()
762 cursor.execute(myquery)
763 output = cursor.fetchall()
764
765 ###
766 print(output)
767
768 ###
769 df = pd.DataFrame(output, columns=['CUST_ID', 'CUST_NAME', 'PHONE', 'CITY', 'GENDER', 'ORD_ID', 'AMOUNT', 'ORD_DATE'])
770 df
771
772 ###
773 # Alternative Query as provided by Faculty
```



```
774
775 #%%
776 myquery = """
777 select * from customer left outer join `orders` on customer.cus_id=`orders`.cus_id;
778 """
779
780 #%%
781 cursor = connection.cursor()
782 cursor.execute(myquery)
783 output = cursor.fetchall()
784
785 #%%
786 print(output)
787
788 #%%
789 df = pd.DataFrame(output, columns=['CUS_ID', 'CUS_NAME', 'CUS_PHONE', 'CUS_CITY', 'CUS_GENDER', 'ORD_ID', 'ORD_AMOUNT', 'ORD_DATE', 'CUS_ID', 'PROD_ID'])
790 df
791
792 #%%
793 cursor.close()
794 connection.close()
795
796 #%%
797
798 #%%
799
800 #%% md
801 ### Lab-01 - Assignment - Concluded
```